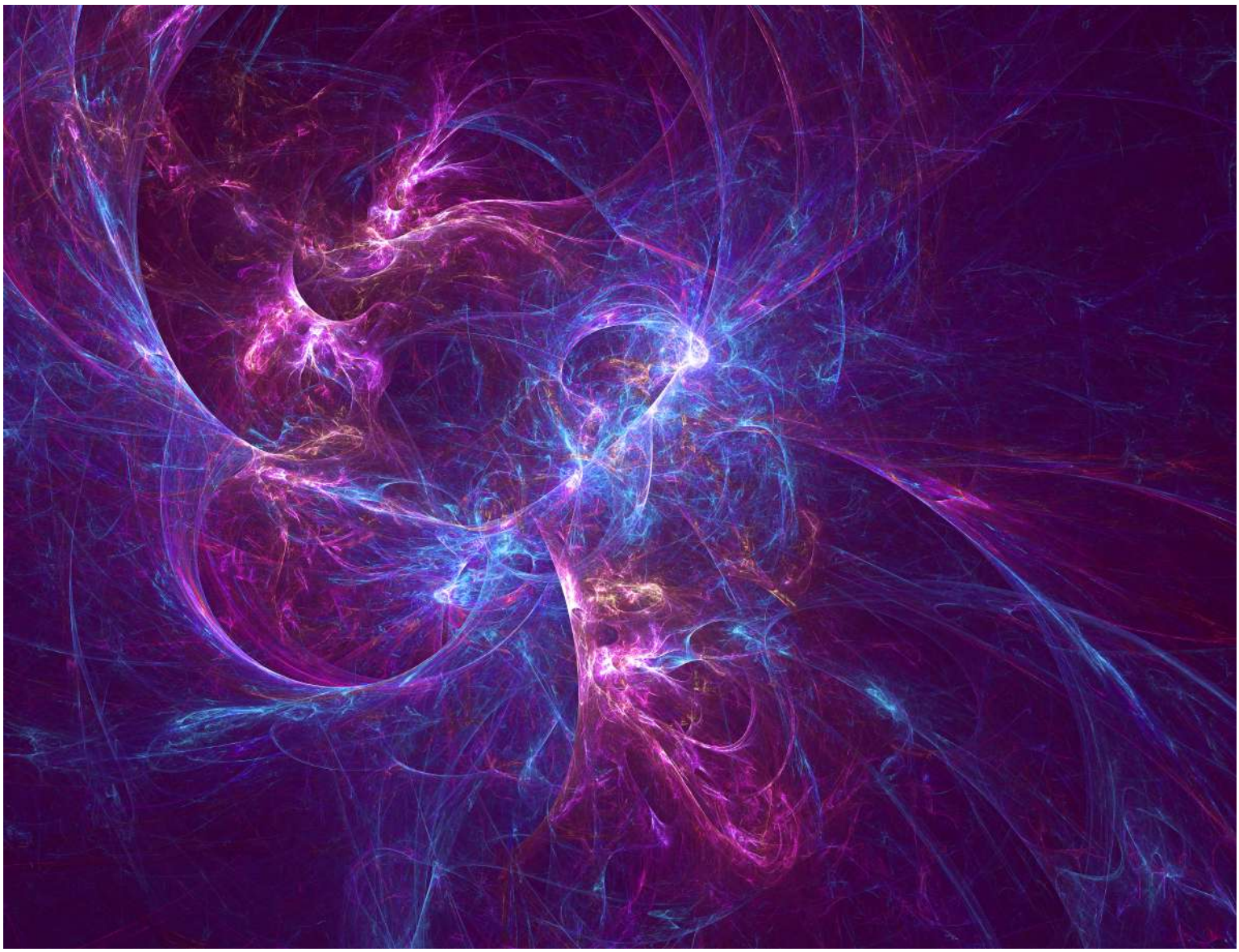
ISSN 2582-8312

Lattice

THE MACHINE LEARNING JOURNAL

VOLUME-2**, ISSUE**-2 (April - June 2021)

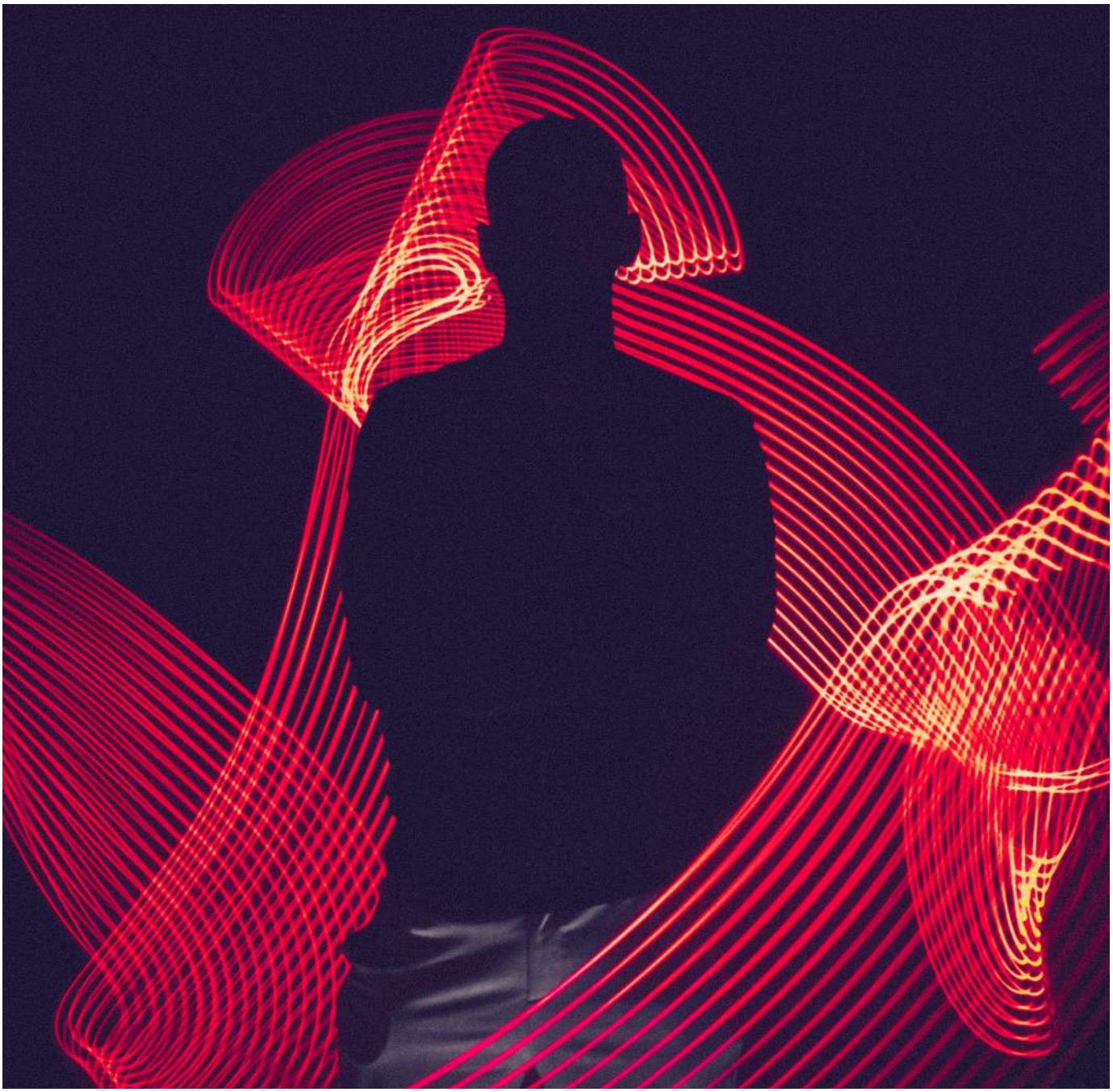


AN INTERNATIONAL PEER REVIEWED OPEN ACCESS JOURNAL
IN DATA SCIENCE & MACHINE LEARNING

ADaSci | THE ASSOCIATION OF DATA SCIENTISTS

www.adasci.org/lattice

Index



PUBLISHED BY:

ADaSci | THE ASSOCIATION OF DATA SCIENTISTS

BANGALORE, INDIA

ONLINE CONTENTS AVAILABLE: EVERY QUARTER ON **www.adasci.org/lattice**

ISSN 2582-8312

Bhasker Gupta
Association of Data Scientists,
#189, 1st Floor, 17th Main Road, Near HSR Club, Sector 3, HSR Layout,
Bengaluru, Karnataka-560102

About THE JOURNAL

Lattice is an international peer-reviewed and refereed journal on machine learning. The journal is hosted and managed by the Association of Data Scientists (ADaSci). Lattice intends to publish high-quality research articles of the researchers and professionals working in the field of data science and machine learning. All the articles published by Lattice have to pass an in-depth doubly blinded review process before publishing. The journal maintains a list of reviewers and editors all belonging to the prestigious institutions/organizations that take part in the functioning of the journal

Scope OF LATTICE

The Association of Data Scientists was formed with the intent to develop, disseminate and implement knowledge, basic and applied research and technologies in analytics, decision-making, and management. Lattice, hosted under the flagship of ADaSci follows the same vision and aims to provide a platform for sharing and exchanging the knowledge and research outcomes in the field of data science and machine learning. Lattice publishes scholarly articles that come under the aim and scope of the journal. The article submitted for consideration must consist of new concepts, theories, methodologies, and applications that are unpublished. It considers articles from the following key areas of data science and machine learning.

Ethics Concerns (PLAGIARISM, MISCONDUCT, ETC.)

The publication of an article in Lattice is considered as a building block in the development of a coherent and respected network of knowledge. The publication is a direct reflection of the quality of the contribution of an author and the organization that supports them. It is, therefore, necessary to adhere to certain standards of expected ethical behaviour. The important points that must be considered before submission are given below.

AUTHORSHIP: Authorship should be limited to those persons only who have made a significant contribution to the conception, design, execution, or interpretation of the reported study. Transparency about the contributions of authors is encouraged.

ORIGINALITY AND PLAGIARISM: The authors should ensure that they have written entirely original works, and if the authors have used the work and/or words of others, that this has been cited or quoted appropriately..

DATA ACCESS AND RETENTION: Authors may be required to provide the raw data in connection with a paper for editorial review, and should be prepared to provide public access to such data.

ACKNOWLEDGEMENT OF SOURCES: Proper acknowledgement of the work of others must always be given. Any funding received for the research must also be acknowledged.

DISCLOSURE AND CONFLICTS OF INTEREST: All submissions must include disclosure of all relationships with any member of the Lattice's editorial team that could be viewed as presenting a potential conflict of interest.

Published by:

ADaSci | THE ASSOCIATION OF DATA SCIENTIST

www.adasci.org

Copyright POLICY

The Lattice requires the transfer of copyrights from the author to the journal. On successful acceptance of every paper to the Lattice, the authors are required to submit a copyright transfer form. The copyright is transferred from the author to the publisher that is meant for the contents in the article. The algorithms and research work is always the intellectual property of the researcher or writer. Consider the case that in future, the author claims that ADaSci has published my work without my knowledge and consent or the author publishes the work with any other publisher and the other publisher claims that ADaSci has published its contents by violating the copyright rules.

The copyright aims to ensure that the researcher has published his work with the publisher to whom the researcher has transferred the copyrights. Now the publisher is the owner of the contents and the publisher of the intellectual property that actually belongs to the researcher. After getting the copyrights transferred from the author, the publisher becomes authorised to publish the contents on his publication mediums such as website, journal, video series etc. The copyright also ensures that the same content is not being published by the author with any other publication without the consent of the current publisher. It also ensures that no other person can publish the contents which are already published where the publisher has the copyrights for the same.

Disclaimer

The Association of Data Scientists (ADaSci) believes that the manuscripts submitted to Lattice by the corresponding authors are their original work as the authors have acknowledged the same while transferring the copyright to the journal. In future, if it is found that the content has been published with any other publication without the knowledge of ADaSci, the Lattice will discontinue the publication of that manuscript from the website.

The Editorial Board

ML based high-cardinality reduction methods to create geo-score to improve auto insurance Tweedie pricing model

Suguna Jayaraj
Senior Manager-Predictive Modelling
Bangalore, India
suguna.srini@gmail.com

Harmandeep Kaur
Senior Predictive Modeller
Bangalore, India
harmankr.56@gmail.com

***Abstract*— A typical automobile insurance rating plan contains a plethora of risk factors, ranging from driver, vehicle, to policy characteristics. Including the geographical risk characteristics into the pricing has been a challenge owing to its high cardinality. The traditional approach groups the postal codes based on the historical loss experience, which suffers from two major drawbacks: a) For geographies with low exposure, the loss cost is almost always zero b) Low confidence as we lose information on the latent variables. In this paper, we demonstrate a case study of Greece automobile insurance product offered by a major US based P&C provider, where a Geo-score was developed at a postal code level to improve risk segmentation in own damage cover pricing.**

The base loss cost(loss/exposure) model was built using Tweedie Compound Poisson regression and geospatial attributes are added in the model without changing the existing rating structure. The external attributes like socio-demographic variables and highway/network data is sourced to create geographical clusters using partitioning around medoids (PAM). Further, various high cardinality feature reduction techniques were used to predict the residual loss cost. This paper illustrates the hybrid approach of the target-based encoding methods and XGBoost to create the geo-score.

Keywords*— high cardinality variables, insurance pricing, GLM models, XGBoost

I. INTRODUCTION

Property-casualty insurance is a complex and dynamic business and the core strategic capability lies in appropriate risk selection/pricing. A pure premium, synonymous with loss-cost, is the amount an insurer would have to charge to cover expected losses. There are several other factors including the commissions, operating costs, target profit which are further added on top of the baseline charges which are out of scope in this discussion. The loss cost model help set the granularity in pricing either by adding new data sources or by using machine learning techniques. In other words, the base price is determined by the risk appetite of the product owner and the slope is decided by the models built [1]. In the actuarial world, commercially available software Emblem is a very popular tool to create these loss cost models and have been in practice for decades. Now, with the availability of machine learning methods to handle multi-collinearity, non-linearities, data sparsity etc., it has opened a new world for the data scientists to integrate them into the pricing models. However, the pricing for personal insurance is heavily regulated in most of the countries by applying cap on the rates, mandating few covers/bundles, fixing tariffs on covers like third party liabilities (TPL)/body injury (BI) to protect the consumer interests. Hence, there is a need to innovate way to work in those boundaries and maintain a profitable portfolio by pricing the risk accurately.

The standard covers sold in Greece by this top insurer based out of US are own damage (OD), Personal damage (PD), BI and Theft [2]. The existing rating plan is based on the risk factors like age, gender, marital status, driving history, vehicle make/model and other policy characteristics. The Greek insurance system operates a bonus-malus system (no-claims bonus), which rewards drivers with no prior accidents or claims as it reduces the premium. Apart from all these attributes, there was a need to distinguish between vehicles from different. PD cover baseline model was not very predictive despite the use of all these attributes, hence there was a need to explore new sources. The geographical level features based on the address given at the time of vehicle registration was something which was unexplored, and the business wanted to tap further into that to build a competitive advantage. Also, in Greece rate filing is not a mandate hence there it was easy to develop machine learning models and create a geo-score.

II. LITERATURE REVIEW

In this section we would like to present some of the studies done by regulators in the major markets on the importance of the geospatial risk factors in pricing personal auto insurance [3]. The NAIC notes that three variables—urban population, miles driven per number of highway miles, and disposable income per capita—are correlated with the state auto insurance premiums. It also notes that high-premium states tend to also be highly urban, with higher wage and price levels, and greater traffic density.

Hence it is evident that the geographical location plays and important role as it captures the latent risk characteristics and the hypothesis is listed as below:

- Pin codes pertaining to denser populations, accident rates and claim frequency is higher.
- In locations with a healthy economy, people are more likely to purchase more expensive cars and the cost of repair can be higher if there is a claim event.
- In areas where the number of unemployed persons is high, there is also a high number of persons that

are driving without insurance, increasing the overall cost burden.
- Road conditions also contribute to wear and tear and hence increasing the claim frequency.
- Location can play a big role in events like staged car accidents, hit and run etc.

Based on the above, the data collection exercise is done to be able to capture most of the factors we think will be helpful in creating a geographic risk score.

The key challenge to capture the geographic risk is that with about ~1000 pin codes available the loss data is very sparse. Most of the commonly available techniques to handle high cardinality variables are either target-based encoding and CATBoost regressor which is specifically optimized to handle categorical variables. However, the literature is limited to fit a Poisson model and hence the strength of the transformation couldn't be tapped here.

III. DATA SOURCING

Internal data is sourced from the policy and claims database for the last 4 years, starting 2016-2019 with the key attributes listed in the table in this section. The address of insured captures the location of the primary insurer and it is assumed that the primary driver is driving within the boundaries of the registered address. The pin codes were available against most of the addresses and wherever it wasn't available a mapping was done based on the area and the closest pin code was assigned to the policy.

Greece is organized into 56 areas, 72 regions, and ~1000 postal codes. The socio-demographic data is available at an area level and had about 50+ features captured though many of them were correlated as expected. The highway/road network data was procured from the public domain for Greece in the form of shape files at a Lat Long level. It classified road type into residential, primary, tertiary, one way, service road etc. The software ArcGIS was used to map these shape files and classify roads at a pin code level.

Table 1. Key variable details

Internal Attributes				**External**
Policy	Policy holder	Vehicle	Claims	Geographical features
Policy Type, Policy Tenure, Bundling Channel	Age Driving ex Address Occupation	Age, Engine size, Body type, Vehicle usage, Make	Amount Type No-claim	*Area level:* Population, Age distribution, Female/Male ratio *Roads:* Type of road (Highway/

IV. GLM OVERVIEW

The state-of-art pricing methods used in the actuarial world is done using exponential distributions which allow the output to be converted to a multiplicative rating table. This allows the flexibility to set a base rate and further increase or decrease premiums based on the risk as in [4].

Generalized Linear Model (GLM) relates the expected value of the target variable ($\mu \equiv E[Y]$) to a linear combination of predictive variables ($\beta \cdot X$) via a "link function" $g(\cdot)$:

$$g(\mu) = \beta 0 + \beta 1 X1 + \beta 2 X2 + \ldots + \beta p X p \equiv \beta \cdot X \quad (1)$$

In a Frequency-Severity approach, the frequency (*number of claims per exposure)* is usually assumed to follow a Poisson distribution, while the severity (*avg. claim amount)* a gamma distribution. Both these models are built separately and then combined. Modeling loss cost (*c/u- claim amount/exposure*) is a little challenging due to their distribution, at policy level it is most often zero and where they do incur a loss, the distribution of losses tends to be highly skewed. As such, the pdf would need to have most of its mass at zero, and the remaining mass skewed to the right which is effectively captured in Tweedie distribution.

One rather interesting characteristic of the Tweedie distribution is that several of the other exponential family distributions are in fact special cases of Tweedie, dependent on the value of p:

- A Tweedie with p = 0 is a normal distribution.
- A Tweedie with p = 1 is a Poisson distribution.
- A Tweedie with p = 2 is a gamma distribution.
- A Tweedie with p = 3 is an inverse Gaussian distribution.

The Tweedie parameter usually lies between $p \in (1,2)$ and was estimated using the as 1.3 and hence Poisson was chosen to fit the loss cost model in this case.

Compound Poisson model that is directly fit on the loss cost is operationally simpler to develop and maintain as it consists in a single scikit-learn estimator instead of a pair of models, each with its own set of hyperparameters. However, at times there is a business requirement to study the effect of the variable on frequency and severity separately and hence F-S models are more common in practice.

V. MODEL DEVELOPMENT

The model development phase is done in two stages, first the socio-demographic variables available at an area level are clustered to identify similar geographic locations. Further, the high cardinal attributes like region, pin code that are transformed using target encoding methods and tested in GLM for identifying the best fit. After the encoding selection, the variables were passed along with the cluster, zone, roadway details into the XGBoost model and tuned to find the best fit.

The following diagram illustrates the flow and the subsequent sections explain them in details:

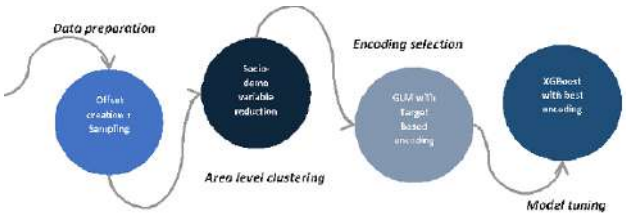


Diagram. 1 Modelling process

A. *Offset creation*

As discussed in the previous sections, the scope of the project is not to update the entire plan and rather, add a geographic risk element into the current pricing scheme. A helpful feature of the GLM framework is the "offset" option which is a model variable with a known or pre-specified coefficient. In this case all the attributes that are existing in the current pricing model can be captured using the offset and the residual can be modelled to create the geo-score. Recall the simple equation of GLM from (1) and now an offset term ξ calculated from the current model β, the equations change as below:

$$\xi = e^{\beta 1}{}_x \; e^{\beta 2}{}_x \; e^{\beta 3}{}_{\ldots .x} \, e^{n} \qquad (2)$$

$$g(\mu) = \beta \cdot X + \log(\xi) \qquad (3)$$

In a completely multiplicative rating plan, the alternative way to incorporate the offset is to create adjusted loss cost by dividing as below:

$$LC_{adj} = c/(u * \xi) \quad (4)$$

Hence, the final parameters for our model are:

- Target variable: LC_{adj}
- Weight: u
- Link function: Log
- Distribution: Poisson

B. *Sampling*

As with any model development exercise, the data is first split randomly into 70:30 split to create a development and validation set. The 2019 data is kept aside to check the out of time validation and stability of the final variables.

TABLE 2. DATASET SUMMARY

	Development	Validation	Out of time
Policy #	667385	286022	270261
Exposure#	389082	166743	164310

C. *Clustering at area level*

There are socio-demographic attributes which are captured at 56 levels are clustered for the following reasons:

- Most of the features were correlated and may not value add when thrown together into the model
- These variables are at an area level and grouping them and creating single score is easy to test

Partitioning around the Medoids (PAM) is one of the most popular hard clustering algorithms used here to divide data into groups (clusters), with *K* number of groups. The k-medoids algorithm requires the user to specify k, the number of clusters to be generated. A useful approach to determine the optimal number of clusters is the average silhouette width method [6]. Silhouette provides a visualization of how well each object lies within its cluster. A silhouette is defined as follows:

$$s=b(i)-a(i)\max\{a(i),b(i)\} \qquad (5)$$

where a(i) is the average distance to all other data points in the cluster and b(i) is the minimum of average distance to other clusters.

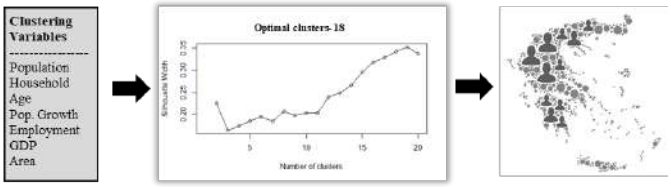


Diagram 2. PAM Clustering

Now, based on the dissimilarity matrix computed based on the Euclidean distance, 18 clusters are created. These clusters are assigned to the policy depending on the area it fell and further flags are created based on the rank order with the LC_{adj} .

D. *Geo variables encoding*

The key focus of this paper is the approach taken to handle the high-cardinality variables. We have explored the target-based encoding techniques where the pin code(or region) is assigned a numeric value based on the function of the target variable chosen. Target (Mean) encoding tends to overfit due to the target leakage. There are various techniques to address this problem. For example, in leave-one-out encoder the current target value is subtracted from the target statistic. This reduces the target leakage. Another technique is to add a Gaussian noise to the encoded value.

TABLE. 3 ENCODING SUMMARY

Variables encoded	Encoding type	Description	Best Transformation
Pin Code Region City	Simple Mean	$\mu = \frac{n\times\bar{x}+m\times w}{n+m}$	Post code WoE Region WoE
	Weight of Evidence	- ln(Sum%/Count%) for that particular category of cardinal variable	
	K-fold	-Split train data into k folds. - Estimate encoding for the sample with the data that left using mean encoding	
	Leave-one-out (LOO)	Similar to K-fold except each observation is left out as a validation sample [7]	

The final encoding function is selected based on the rank order, GINI and the lift obtained from the various iterations.

Overprediction adjustment : The WoE based encoding of the post code is the most predictive, however it is overpredicting for the high-risk policies. This is due to the low exposure in some of the pin codes and hence the credibility criteria in the bucket was not met. There were about 300 postal codes where the no. of policies written were less than 10 and with a very low confidence on the loss cost calculated. For those policies the postal code WoE encoded value was replaced with the WoE encoded value of the region it belonged to.

Postal_code_driver1_WoE=Region_code_WoE(If#policies <10 at postal code level) (6)

This hybrid postal code variable brings the smoothening effect for those small pockets and improved the rank order in the last bucket.

E. XGBoost Model

GLM model with transformed variables was improving the predictability but at the cost of overfitting in last three buckets which was happening on the account of low exposure postal codes with minimum loss history. XGBoost [8] helped overcome the high variance with boosting and cross validated tuning and fixed the overfitting for high loss buckets.

The optimal tuning parameters are those that minimize the cross-validation error, which is obtained by averaging the error on the validation sets.

Table. 4 Tuning parameters (Poisson nloglik)

ETA	Max Depth	Min child weight	Obj.	Colsamp By tree	Sub sample	nrounds
0.3	2	1	Poisson	0.6	0.5	72

The SHAP plot shows the variable importance. As expected, we see that the postal code with WoE is the most significant variable and the flags created from the clusters are also helping in improving the rank order .

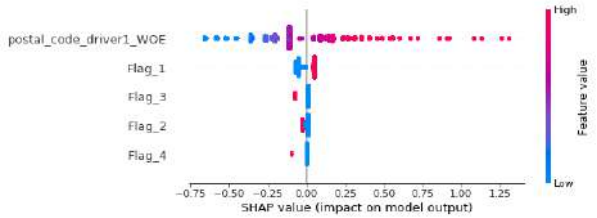


Fig. 1 SHAP Plot with variable importance

VI. DISCUSSION OF RESULTS

In this section we discuss the results obtained from the modelling exercise and how it is integrated to the final rating plan to then study the business impact from the additional of the geo-score.

A. Final Model Selection

First the geo-score was computed by aggregating the policy level scores at a postal code level. The final loss cost is further calculated as below:

Geo-Score (Postal code) = μ (Predicted Adj. LC) (7)

Final Loss cost (Pure Premium) = Geo-Score*Offset (8)

The most important criteria to select the best model for any pricing model are :

- Predictive accuracy : Difference between the actual and the predicted in each decile
- Monotonicity : The actual pure premium should consistently increase across decile
- Lift : Vertical distance between the actual loss cost of the first and the last decile that indicates how well the model distinguishes between the best and the worst risks
- Stability : The performance on the out of time data as well as the importance of the same attributes obtained in the train dataset

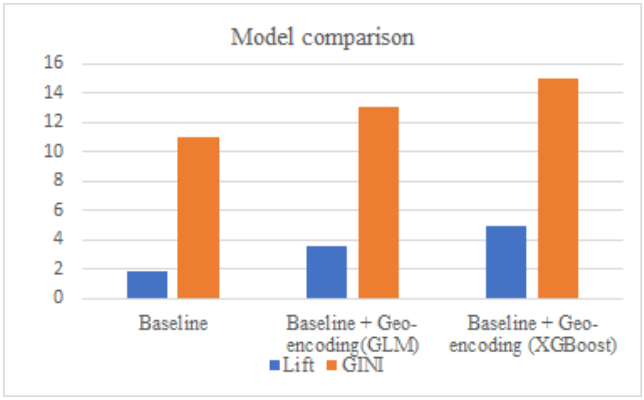


Fig. 2 Model Performance metrics

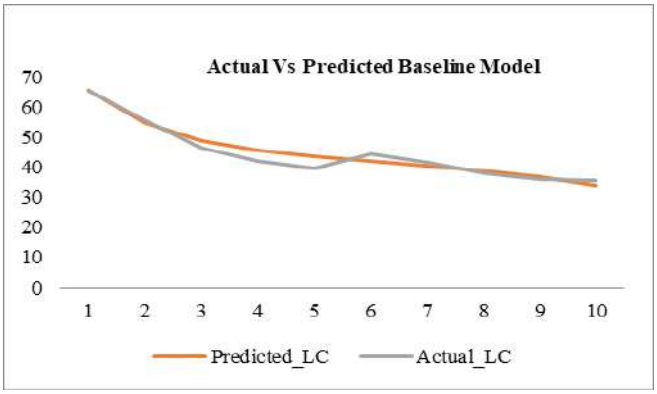


Fig. 3 Baseline model performance

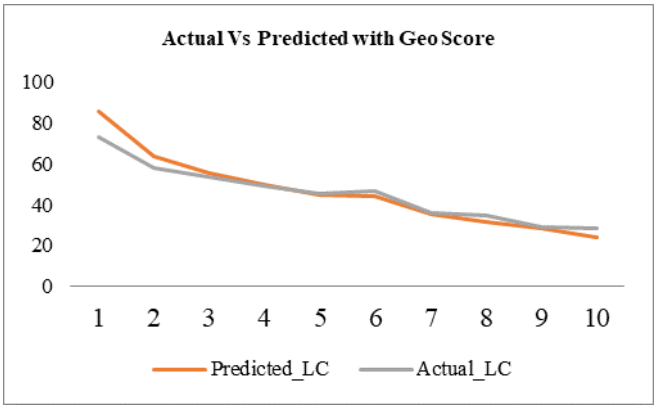


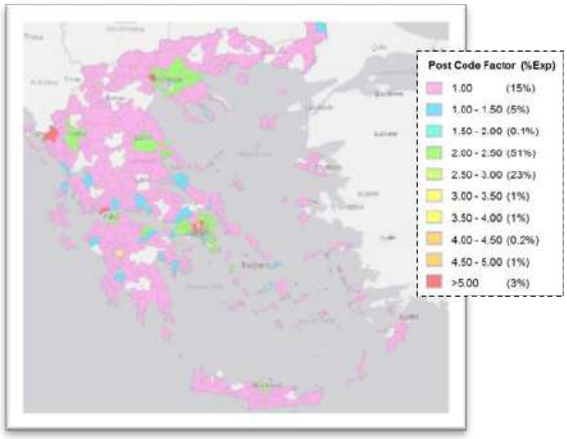
Fig. 3 Model performance with geo score

As observed in the chart above, the lift and the GINI improves due to the addition of the geo attributes. There is slight overprediction in the last decile which can be attributed to heavy vehicles and large losses. They are typically capped and modelled separately as a part of large loss modelling, but not done here as the frequency of such losses is very limited.

VII. BUSINESS IMPACT

A. Implementation

At the end of the modelling exercise, for the consumption by the pricing team, the pure premium is converted to a base price and multiplicative rating factor after adjusting to the latest year claim frequency (also known as trending). The geo-scores at a postal code level translate as below:



Diag. 3 Geo-score for Greece

B. Impact

In order to determine the final business impact, the entire portfolio including the current policies is scored with the new model that includes the geo-score. While single lift chart decides the economic value of a model, a double-lift chart is also a common practice in the pricing to directly compare the results of two models. Double lift charts are like simple decile plots, but rather than sorting based on the predicted loss cost of each model, the sorting is done based on the ratio of the two models predicted loss costs. The x-axis denotes the decile # and the y-axis compares the predicted loss cost from both the models with the actual.

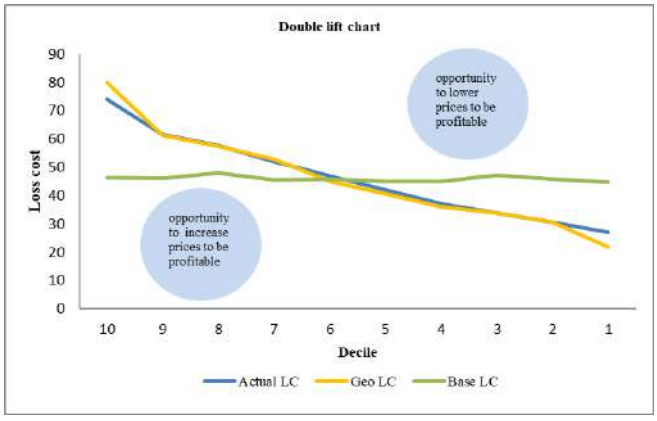


Fig. 4 Business Impact

Above chart is able directionally gives an idea where the premiums need to be adjusted. To find out the exact population that gets impacted in terms of the increase/decrease in the premium to be charged a swap-sets analysis is performed.

VIII. CONCLUSION

From the various approaches taken to handle the high cardinal variables, we see that ML based techniques show a significant improvement. At times we need to follow a hybrid approach by integrating output from the statistical models (clustering, GLM in this case) for a more efficient feature engineering. While XGBoost results are satisfactory, there is still room for improvement and the following can be explored further

- Bayesian approaches to encode the variable
- CATBoost for Poisson

REFERENCES

[1] https://openacttexts.github.io/Loss-Data-Analytics/C-RiskClass.html

[2]https://www.insuranceeurope.eu/sites/default/files/attachments/European%20motor%20insurance%20markets.pdf

[3] https://www.autoinsurance.org/quoting-auto-insurance-rates-by-zip-code

[4] Generalized Linear Models by P. McCullagh, John A. Nelder
http://www.utstat.toronto.edu/~brunner/oldclass/2201s11/readings/glmbook.pdf -1

[5] Casualty Actuarial Society, Basic Ratemaking 5th Edition, May 2016
https://www.casact.org/library/studynotes/Werner_Modlin Ratemaking.pdf

[6] Rousseeuw, P.J.: Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis. Comput. Appl. Math. 20, 53-65

[7] https://link.springer.com/article/10.1186/s40537-020-00305-w

[8] XGBoost: A Deep Dive into Boosting (Introduction Documentation)

Optimizing Cost per Click for Digital Advertising Campaigns

Aditya Jain*, Sahil Khan†
* Data Scientist II R&D, MiQ Digital
† Manager Data Science R&D, MiQ Digital

***Abstract*—Cost per click is a common metric to judge digital advertising campaign performance. In this paper we discuss an approach that generates a feature targeting recommendation to optimise cost per click. We also discuss a technique to assign bid prices to features without compromising on the number of features recommended.**

Our approach utilises impression and click stream data sets corresponding to real time auctions that we have won. The data contains information about device type, website, RTB Exchange ID. We leverage data across all campaigns that we have access to while ensuring that recommendations are sensitive to both individual campaign level features and globally well performing features as well. We model Bid recommendation around the hypothesis that a click is a Bernoulli trial and click stream follows Binomial distribution which is then updated based on live performance ensuring week over week improvement.

This approach has been live tested over 10 weeks across 5 campaigns. We see Cost per click gains of 16-60% and click through rate improvement of 42-137%. At the same time, the campaign delivery was competitive.

I. Introduction

For anyone accessing the internet, Digital marketing is not a new term as you are targeted with advertisements on multiple platforms where businesses try to reach the right audience interested in buying different products and services. Digital marketing is an umbrella where all marketing channels like TV, Desktop and mobile are used to reach out to a particular audience. Digital marketing industry size in 2019 was around $300-310 billions and is expected to grow in 2021 at 15-20%. Digital marketing houses run primarily by agencies do look for proven data science and machine learning methods to strike the right balance between relevancy and growth. Online advertising is driven by the demand side and supply side of the business primarily governed by real time bidding exchange where real time auction takes place. Demand side is dominated by agencies and clients who are looking for an audience interested in their products and service. Publishers and sellers on the other hand dominate the Supply Side. Supply side parties have a placeholder for advertisements to be shown on their websites and mobile apps where traffic of relevant audience arrives and are thus redirected to websites of clients ending up purchasing their products and services.

Advertising on display and videos have been in the industry for long and people do look for ways to innovate it. Although the industry is led by solution owners or programmatic traders with their own assumptive intuition, Data Science practices have opened doors for people to dig deep and dive in further to bring the best value out of any click or conversion happening. With the algorithmic approach, it has gone wide to help the media industry quickly. Targeting the right audience for an advertisement does involve looking at a lot of factors and that can be solved using a model based approach. One of the most common business KPI that marketers look for when they want to call a managed campaign successful is CTR - click through rate. Click Through Rate is a KPI which looks at ratio of users who click the advertisement with respect to people shown an advertisement. In marketing campaigns, you would observe a click through rate of 0.03% while through our methods, we tend to increase the performance of the campaigns by 1.5X to 2X making sure the other media related constraints are satisfied. Technically a good click through rate depends on various factors including the platform. A good CTR for adword's search page could be different from those of Facebook's and would be largely different from media campaigns. It also normally varies from one vertical to another, the place an advertisement is placed, size of the advertisement creative and also the location where an advertisement is targeted.

With this paper, we are looking at data driven techniques to optimise click through rate. While looking at multi-level features selected for better clicks based on initial exploration serves as an initial component, other important aspects that we cover in this paper is the pipeline that allows us to scale our solution to over a Billion rows.

Another form of scale relevant to this paper is scale of impressions. Getting enough scale for any active campaign and at the same time making sure the cost spent on getting those clicks while keeping the cost minimal is the problem we are trying to solve in this paper.

II. Data and Experimental Context

A. Experimental Context

Before we proceed with the discussion of our approach, the experimental context and use-case of the final result needs to be addressed to better explain the practical restrictions governing use of results obtained via the approach outlined in this paper.

Many publications discuss approaches that can be utilised for modelling Click Through Rate. Typical approach is modelling for $P(x = 1)$ using a classification model. Neural Networks, SVMs, Decisions Trees, Random Forests, and various boosted tree approaches have been show to work for this task.

However, our setting and requirement are different form these existing solutions. Typical approaches usually are not very sensitive to the tail end of input features. In practice, we have seen that for our campaigns the highest performing features typically form the tail end of the distribution. It is well known that optimising a model for tail end values is an uphill task. Hence, we need an approach that optimises reasonably well for such features.

Our goal here is to recommend a feature combination along with a reasonable bid value so that the aggregate cost per click is improved. Feature combination is a set of contextual features that define where an impression can be delivered. For example the following table shows a few valid feature combinations:

TABLE I
EXAMPLE FEATURE COMBINATIONS

Site Domain	Device Type	Size	Fold
analyticsindiamag.com	Mobile	300x50	1
yahoo.com	Desktop	300x250	0

Along with such feature combinations (sometimes referred to as context), we need to send bid prices that we are willing to pay for an impression served at each such context. We however have no way of specifying the exact number of impressions that we wish to win for a particular feature combination. That control is not available. Thus bid prices are the only other parameter that we can control. Various methods exist that allow modelling of number of impressions vs a bid price. However, all of them require auction level left censored data that is not available to us [1]. Therefore, the approach discussed in this paper focuses on assigning maximum bid price which is also the price at which the expected CPC is equal to or lower than our target.

Digital advertising campaigns are very dynamic leading to varying week over week performance of same feature sets. Any approach that is chosen for the task should allow for constant feedback and iterative improvement. In case of ill-performing feature set, the approach should be quick in updating its recommendation.

At the same time the approach had to be compliant to GDPR, a European law outlining privacy honouring requirements of data collection and processing. Therefore, our approach does not use user level identifiers and operates at aggregated feature level.

Our chosen approach fulfils all these requirements.

B. Data

We use impression stream and click stream at the organisation level as the input to our process. An impression stream data consists of all impressions that we were able to serve at the account level. Similarly, click stream data consists of every click that happened as a result of an advertisement shown by us.

Along with the information of an impression event or a click event, these streams provide us information about the context of the ad impression. Typical row from this data set contains information about the site domain where the ad impression took place, time stamp of ad impression, device type, geographical information like Zip code, Internet service provider etc. The complete data dictionary contains well over 30 columns of which 7 columns that contain information about price and targeting are of interest to us.

Three types of columns are present in our data set which convey

1) Context of ad slot
2) Cost of ad slot
3) Non Context information

Context information indicates where the ad impression was shown and can be directly used for targeting. This includes the following columns:

- **Timestamp**: Time stamp of click or impression
- **Height**: Height of the image required by the ad slot
- **Width**: Width of the image required by the ad slot
- **Device Type**: Type of device Desktop, Mobile, or Tablet that this ad impression was shown
- **Operating System**: Operating system of device
- **Browser**: Browser type and version where this ad impression was shown
- **Fold Position**: Above fold or below fold. Indicates if the advertisement is visible on page load or after scrolling down the page
- **Geo Country**: Country where this ad impression was shown
- **Geo Region**: DMA [2]
- **Seller Member ID**: Seller via whom the inventory is made available
- **Tag ID**: Unique ID of ad location on a website
- **Publisher ID**: Unique ID of website owner
- **Site Domain**: Mobile Application or Website

Non Context information includes following columns

- **Insertion Order ID**: Advertising campaign identifier
- **Advertiser ID**: ID of Advertiser a particular impression or click belongs to
- **Is Click**: 0 if not click, 1 if click

Cost information contains following columns:

- **Media Cost**: Cost of the impression in USD
- **Data Cost**: Per impression cost of third party data used

Due to the targeting restrictions of our upstream provider, we combine Height and Weight and create Size. Similarly, Geo country and Geo Region are combined to form Geo targeting column. Actual realised cost of an ad impression is $MediaCost + DataCost$. We use the aggregated amount for further analysis and modelling.

From the click and impression stream, we prepare two data sets campaign level, and network level with minor differences. Campaign level data contains all the columns mentioned above that we filter from the larger data set. Network level data however is not processed at campaign level. For this data set, we remove the following columns:

- Insertion Order ID

- Advertiser ID

The final data set contains approximately 1 Billion rows spanning a duration of 7 days. We will discuss more about the usage of these different data sets in section IV.

III. Modelling

An ad-impression can lead to two states that are relevant to this discussion. It can either lead to a click or not lead to a click. We can thus say that a Click is a binary random variable where the value 0 represents a non-click event and 1 represents a click event. We are treating clicks, and impressions as independent events.

This allows us to model a click stream as a Bernoulli Trial [3], [4]

A. Bernoulli Trial

- Let probability of a click be p_c
- Then, Probability of no-click $p_n = 1 - p_c$
- $p_c + p_n = 1$

Since we treat each impression as a Bernoulli Trial, it follows that a series of such trials be modelled as a Binomial experiment where probability of getting n clicks can be expressed as:

$$Pr(X = n) = \binom{i}{n} {p_c}^n (1 - p_c)^{i-n} \tag{1}$$

From data, we can calculate the ratio of clicks vs total impressions. However, consider a coin toss experiment. If we observe coin toss leading to 2 heads and 0 tails in two independent trails, does it follow that the coin only lands on Heads?

This question leads us to the Beta Distribution.

B. Beta Distribution

Beta distribution is the conjugate prior for Binomial and Bernoulli Distributions. Accordingly, we can write

$$f(p_c|n, n_i, i_i, i) \propto {p_c}^{n+n_i-1}(1 - p_c)^{(i-n)+(i_i-n_i)-1} \tag{2}$$

where

- subscript i indicates imaginary trials

The expectation of (2) will give us the expected probability of click P_c from click vs non click data.

For this we leverage Bayesian inference [5] over Beta Distribution as mentioned in equation (3)

$$p(x = 1|Data) = \frac{m + a}{m + a + l + b} \tag{3}$$

where

- $p(x = 1)$ is the probability of Success
- m is prior clicks
- a is real clicks
- l is prior non clicks
- b is real non clicks

This affords us a very simple and explainable approach that we can use to calculate expected click through rate or P_c from the data.

Per the Bayesian approach, we can use the same equation with updated data of a and b to update our belief. This way, we can calculate the posterior probability of clicks by simply adding new data to our data-set without modifying any other part of the system.

C. Final Approach

Our final approach uses equation (3) to calculate expected Cost as well as expected Click through Rate. Along with this we use a heuristic measures to prevent under delivery and high cost.

We utilise data from Network level feed as well as campaign level feed as discussed in section II-B. The reasons for this are twofold:

- Prevent under delivery by using feature combinations with wider reach extracted from network level data
- Bootstrap performance of campaign from known high performing features from network level data.

We first calculate network wide average impressions, and average number of clicks for all feature combinations. This forms the prior part of equation (3). For all feature combinations, we calculate adjusted click through rate by adding the prior to their actual performance.

We repeat this step for cost column to give us a prior cost. Both these steps allow us to handle feature combinations with few data points well.

The same steps are repeated for campaign level data where the prior is again calculated at campaign level. Adjusted Cost and CTR are then calculated for all campaign level features.

We then proceed with bid calculation targeting a specified CPC per the logic below.

$$CTR = \frac{Click}{Impressions} \tag{4}$$

$$CPC = \frac{Cost}{Click} \tag{5}$$

$$adjusted\_ctr = \frac{prior\_click + click}{prior\_imp + imp} \tag{6}$$

$$\begin{aligned} adjusted\_cost = {} & prior\_cpm * prior\_imp \\ & + \\ & feature\_cpm * feature\_imp \end{aligned} \tag{7}$$

$$adjusted\_imp = prior\_imp + feature\_imp \tag{8}$$

$$adjusted\_cpm = \frac{adjusted\_cost}{adjusted\_imp} * 1000 \tag{9}$$

$$CPM = CPC * CTR * 1000 \tag{10}$$

Substituting CPC in equation (10) with a known target, we can calculate the max affordable CPM. By substituting CTR with $adjusted\_ctr$ in equation (10) we can calculate the highest bid price we can recommend given the expected CTR. In this step we introduce a parameter $optimization\_fraction$. Since the goal of this approach is to optimise CPC, we

multiply this variable with the obtained CPM before recommending it to the users. This enables us to always push recommendations that would perform better than the rest of the campaign in terms of CPC. Very aggressive value of $optimization\_fraction$ leads to severe under delivery. Hence, it is advisable to test a few variations or modify this value automatically in a feedback loop.

IV. Implementation Overview

Any task in Digital Advertising industry has to handle at least a few terabytes of data. The approach in this paper is no different and needs to scale to ~24TB of raw input data. PySpark or in our case Databricks is the go to platform.

Figure 1 outlines the overall design of pipeline that we are currently using to generate recommendations. It is divided into 3 parts

- Request creation
- Aggregation and Generation of recommendation
- Activation

Jarvis is an internal tool that takes care of receiving requests for recommendation which is then processed in batch mode once or twice per week.

The bulk of processing happens on Databricks. The first step is to load the raw feeds from S3. For the purpose of these experiments, we loaded 7 days of impression and click stream feeds. Following are the steps that are performed on network level data:

- Filter for relevant Geographical region.
- Group by data with relevant fields
- Remove outliers outside 2 standard deviation
- Calculate average impression & click for use as prior
- Add prior to all feature combinations generated
- Sort by adjusted CTR
- Filter the feed for top 100K impressions with the highest adjusted CTR
- These features are common for all campaigns however bid values are different across each campaign

We perform similar step on campaign data. Prior impressions and clicks are calculated at per campaign level. Another difference is that we do not filter campaign data for top 100K impressions. All impressions and feature combinations are used albeit at lower bids. Once both feeds are individually processed, we proceed with a merger of recommendations from both these sources based on the requested scale of recommendation. Typically, 30% of scale is served from network level features.

Next step is to calculate the bid values using methods discussed in section III. Subsequent steps involve packaging these results into required format and uploading them to our upstream service provider.

We repeat this process twice every week to ensure that bad performing features are kept in check.

A. Effectiveness of Feedback loop

Our hypothesis is that every feature that is not optimally performing will eventually face reduced bids till it starts performing better.

Let's assume a feature combination T that has only delivered 100 impressions so far and has received exactly 1 click. At this point there isn't enough information about T to allow us to make an informed decision. Therefore, we add the prior values calculated from campaign data.

For the sake of argument let's assume that the prior values are at 1 click and 1000 impression. After adding this to the data of T, the effective CTR now becomes

$adj\_ctr = \frac{1+1}{100+1000} = 0.18\%$

This adjusted CTR value is very high and consequently T receives a very high bid value.

In the next iteration of the pipeline, there are 3 possible cases

1) T is performing really well
2) T is delivering a lot of impressions and thus costing us a lot without getting us a lot of clicks
3) T is not able to deliver at all. The delivery is stuck at 100 impressions.

Case 3 is trivial. The algorithm will arrive as bids same as the last time, and we will not see a lot of delivery again in the next week. This is OK as long as other features are delivering sufficient inventory.

Case 1 is also trivial. The algorithm will recalculate the adjusted CTR and increase the bids as applicable.

Case 2 is where we need to ensure that bad performing features stop delivering or deliver at a lower cost per thousand impressions thereby increasing the effective CPC. Let's assume that the total impressions delivered by T in this case is 10,000 without any new clicks.

By calculating the adjusted ctr, we get

$adj\_ctr = \frac{1}{11000} = 0.009\%$

This time around, the algorithm will reduce the bid value allocated to feature combination T as governed by equation (10). Since adjusted CTR is in the numerator of this equation, the effective bids allocated to T will be very low as required by the equally bad performance.

Hence, if a high bid value is assigned to a bad performing feature yet unknown to us, it is benign if it falls under case 3. If it falls under Case 2, we can be sure that the bids will be reduced in response. This dynamic nature of our approach make it responsive to bad performing features and ensures that campaign budget is not wasted.

V. Tests and Results

We tested our approach on 5 live campaigns in the US Region across different verticals and campaign configuration for a duration starting from Late September to Early December. Within each of these campaigns, a new strategy (Line Item) was created and associated with our recommendations. Other strategies that were already delivering on these campaigns included ones optimising for Impressions, CPC, Viewability. No change was made to other line items of these test campaigns.

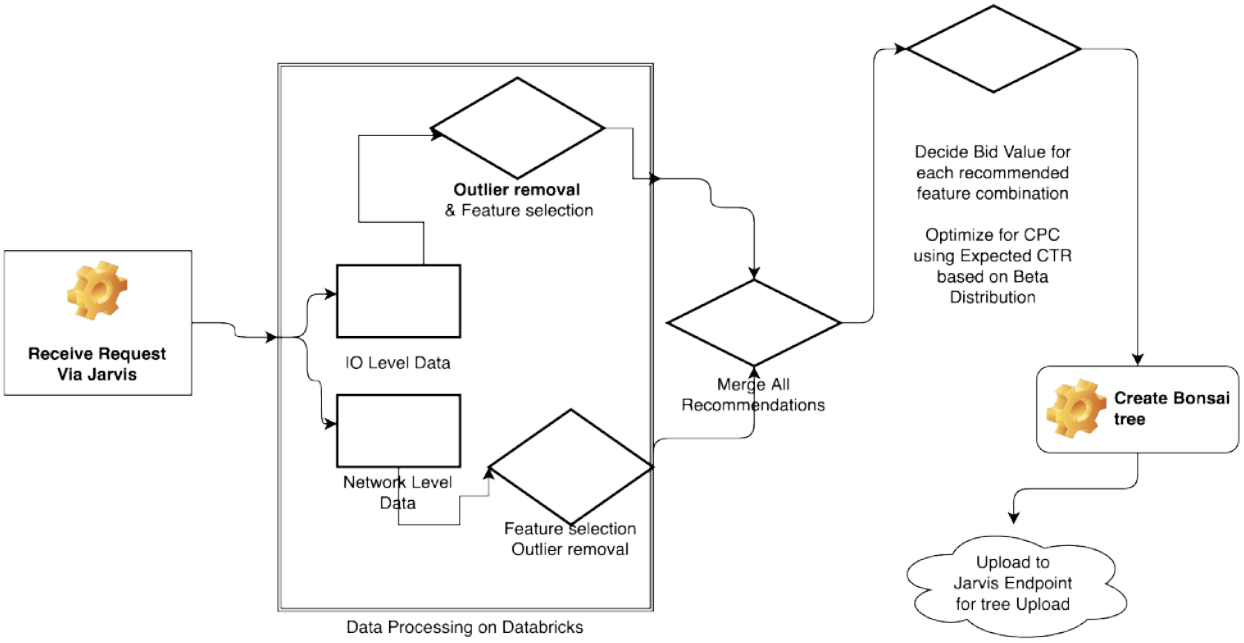


Fig. 1. High level pipeline architecture

TABLE II
DELIVERY COMPARISON OF EXISTING LINES VS NEW RECOMMENDED LINE

Impressions	Daily Budget	Delivery %	**Campaign**	Impressions	LI Budget Daily	Delivery %
C	**C**	**C**	← Type →	**R**	**R**	**R**
4626434	16170000	28.61%	**A**	817478	2560000	31.93%
6537127	13914000	46.98%	**B**	1124843	960000	117.17%
8077947	57446000	14.06%	**C**	1201979	2860000	42.03%
3161197	6910000	45.75%	**D**	170969	9600000	17.81%
2965780	N/A	N/A	**E**	104277	N/A	N/A

TABLE III
KPI COMPARISON OF EXISTING LINES VS NEW RECOMMENDED LINE

Clicks	Media Cost	CPC	CPM	CTR	**Campaign**	Clicks	Media Cost	CPC	CPM	CTR
C	**C**	**C**	**C**	**C**	← Type →	**R**	**R**	**R**	**R**	**R**
3900	8080	2.07	1.75	0.08%	**A**	1538	1693.233	1.1	2.07	0.19%
3616	12486	3.45	1.91	0.06%	**B**	1019	1378.6872	1.35	1.23	0.09%
5823	11472	1.42	1.97	0.07%	**C**	1204	1435.88	1.19	1.19	0.10%
1882	6610	3.51	2.09	0.06%	**D**	336	321	0.96	1.88	0.20%
5594	6626	1.42	1.97	0.07%	**E**	472	258	0.56	2.48	0.45%

Tables II and III compare the result of existing strategies indicated by **C** against type with recommended strategies indicated by **R** against type. Campaign **A** to **C** had a greater geographical coverage. Campaign **D** was configured to deliver on a very restricted geographical area akin to a district. Campaign **E** was a geo-fence campaign using 3rd party data.

In table II, the Impressions column indicates the total number of advertisements show during the test period for a campaign. The daily budget column contains the sum of individual targets of each strategy. Typically, stakeholders over-allocate strategies to ensure campaign delivery. Therefore, we see the Delivery% column containing numbers much below 50%. In practice, the Delivery% of our recommendations should be comparable to corresponding existing strategies.

On the delivery front, we see that Campaigns A to C have a much higher delivery percentage for our recommendations. This percentage when higher indicates that we are able to deliver more than our fair share of the impressions. In campaign D our recommendation as only able to reach 17.81% delivery whereas existing strategies delivered 45.75%. We attribute this to the strict geographic requirement of the campaign.

On the KPI front in table III we see that campaigns that were able to fulfil delivery requirements also have 42.8% to 137.5% better Click through Rate(CTR) and at the same time have 16.19% to 60.86% better Cost Per Click(CPC).

CPM across all well delivering campaigns is lower except for campaign A where it is 18.28% higher than the corresponding strategies. However, this is more than made up by

the much better CTR allowing the line to achieve a lower CPC with respect to control lines.

Campaigns D and E under delivered and the corresponding delivery is much lower than required. However, even in such cases our approach resulted in much lower CPC and much higher CTR. Campaign D realised 233% improvement in CTR corresponding to 72% reduction in CPC. Campaign E realised 641% improvement in CTR corresponding to 60% reduction in CPC.

Thus far, our approach has been able to meet the primary goal of improving Cost per Click for each of the campaigns. After monitoring the week over week performance of these campaigns for the test duration, we can say that the approach is able to react quickly to performance changes, thus satisfying our requirement of responsiveness.

VI. Conclusion and Future Work

In this paper we have discussed the effectiveness of modelling a Click event as a Bernoulli Trial. In digital advertising, many events like Converts, Views, Video completion are suitable candidates for application of this approach. We have seen certain edge cases like restrictive geographical targeting that have resulted in low impression delivery. We would like to explore variations to this approach that will enable us to guarantee delivery for such campaigns. A reinforcement learning approach to modify the parameters of our approach will reduce the manual intervention required in cases of extreme delivery.

References

[1] Y. Cui, R. Zhang, W. Li, and J. Mao, "Bid landscape forecasting in online ad exchange marketplace," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2011, pp. 265–273.

[2] A. C. Nielsen, "US Designated Market Areas," 2000.

[3] A. Papoulis and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes*. Tata McGraw-Hill Education, 2002.

[4] "Bernoulli trial," *Wikipedia*, Oct. 2020.

[5] C. M. Bishop, *Pattern Recognition and Machine Learning*. springer, 2006.

Pneumonia Detection and Classification on Chest Radiographs using Deep Learning

Naveen Raju S G*
AI Engineer
Telerad Tech Pvt. Ltd.
Bengaluru, India
naveenraju100@gmail.com

Kishore Rajendra*
AI Engineer
Telerad Tech Pvt. Ltd.
Bengaluru, India
kishorerajendra000@gmail.com

Tejas Haritsa V K
AI Engineer
Telerad Tech Pvt. Ltd.
Bengaluru, India
tejastejatej@gmail.com

Dr. Arjun Kalyanpur
CEO and Chief Radiologist
Teleradiology Solutions and Image Core Lab
Bengaluru, India
arjun.kalyanpur@telradsol.com

Dr. Pallavi Rao
Senior Scientific Officer and Consultant Radiologist
Image Core Lab
Bengaluru, India
pallavi.rao@imagecorelab.com

Abstract:

Computer Aided Diagnosis (CAD) is progressively becoming a reliable tool in enhancing the productivity and accuracy of a radiologist in detecting abnormalities on chest radiographs. Detection of airspace disease such as pneumonia can be facilitated with the help of image processing and deep learning algorithms. In this study we aim to develop and evaluate the performance of a deep learning model to detect pneumonia on chest radiographs.We have used RetinaNet [1] with Resnet-101 [2] as backbone architecture and trained on chest radiographs with pneumonia findings. The trained model was validated on open source datasets from US National Institutes of Health (NIH) [3], The Society of Thoracic Radiology (STR) [4] and private data repositories [5]. For the purpose of validation, bounding boxes enclosing the ground truth were used as the inference standard. The deep learning model correctly predicted pneumonia on chest radiographs with an accuracy of 96.33%, sensitivity of 97.51%, specificity of 95.55% and Area Under Receiver Operating characteristic Curve (AUROC) of 97%.

Keywords:

Pneumonia, Computer Aided Diagnosis (CAD) systems, Image Processing, Deep Learning Algorithms, Chest Radiographs, CheXNet[11], DenseNet-121, RetinaNet [1], Resnet-101 backbone architecture [2], US National Institutes of Health (NIH), The Society of Thoracic Radiology (STR).

I. INTRODUCTION

Pneumonia is a form of acute respiratory infection that affects the lungs. The lungs are made up of small sacs called alveoli, which fill with air when a healthy person breathes. When an individual has pneumonia, the alveoli are filled with pus and fluid, which limits oxygenation. According to the survey of World Health Organization (WHO) [6], pneumonia is the single largest infectious cause of death in children worldwide. Pneumonia killed 808,694 children under the age of 5 in 2017, accounting for 15% of all deaths of children under five years old. Pneumonia affects children and families everywhere, but is most prevalent in South Asia and sub-Saharan Africa. Detection of pneumonia at an early stage is necessary to prevent deaths and to increase survival rate.

For imaging of the chest various techniques are used, of which Chest Radiographs are preferred in initial evaluation because of their wide availability and low cost. In recent years, increase in computational capacity and advancements in Deep learning technologies in the realm of computer vision have had a direct impact on the medical field. Latest strides in visual information extraction and accessible image processing techniques have improved automated inferences. Data scientists can presently process on larger numbers of radiograph data in shorter periods. This study aims to develop and evaluate the performance of a deep learning model based on RetinaNet[1] to aid radiologists in detection of Pneumonia on chest radiographs. This model was trained on a curated data set consisting of chest radiographs with and without pneumonia from both public and private repositories. The implementation of this algorithm is projected to help in mass screening scenarios by increasing the accuracy of pneumonia detection and reducing the Turn Around Time (TAT) of reporting.

II. DATASET AND METHODS

A. DATASET

The data set was curated from open source repositories such as US National Institutes of Health (NIH) [3], The Society of Thoracic Radiology (STR) [4] and private data repositories totaling 25,798 chest radiographs of varying quality. The data set comprised of 2092 radiographs deemed positive for pneumonia and 300 radiographs deemed negative for pneumonia from NIH [3], 9,555 positive chest radiographs and 8851 negative chest radiographs from STR [4], 2500 positive and negative anonymized chest radiographs gathered by our internal Image Core Lab (ICL). This was approved by our institutional ethics committee (IEC) at ICL. The dataset was used to train the AI system, excluding 3,000 images which were used only for validation during the training phase and 2000 images which were used only for testing. An initial analysis of the data set revealed that the mean age of patients was 58 years ± 28.5 (standard deviation) (52% female).

Since the exact boundaries of pneumonia were hard to establish in most cases, Ground Truth in the form of bounding boxes was the approach followed by our radiologists at ICL. A similar approach was followed by data set annotators of Kaggle [4]. A section of the NIH dataset labeled for pneumonia was utilized in this process. Other abnormalities were being labeled alongside pneumonia

* Equal Contributors

regions on images, so each instance was marked with a bounding box with its coordinates saved in Pascal-VOC format with the abnormality name as its label. Radiologists and technicians at ICL labeled instances of pneumonia using an open source annotating tool.

B. METHODS

Problem statement

The purpose of the algorithm in focus is to detect and localize pneumonia on chest radiographs. On chest radiography pneumonia can have irregular boundaries, ill-defined appearance and sometimes it's features may overlap with those of other benign abnormalities making its detection and localization a matter of contention between observers[6].

1)CheXNet approach

Automating diagnosis from chest radiographs has gained popularity with algorithms for pulmonary tuberculosis classification (Lakhani & Sundaram, 2017)[7] and lung nodule detection (Huang et al.,2017)[8]. Islam et al. (2017)[9] studied the performance of various convolutional architectures on different abnormalities using the publicly available OpenI dataset (Demner-Fushman et al., 2015)[10] and gave multiple conclusions such as the same deep convolutional neural network architecture doesn't perform well across all abnormalities, ensemble models will improve classification significantly compared to a single model when only deep convolutional neural network models are used.

In Rajpurkar et al. [11], a 121-layer Dense Convolutional Network with a modified final layer, christened CheXNet [11], has a chest radiograph image as input data and the probability of pneumonia along with a heatmap localizing the areas of the image most indicative of pneumonia as it's output. The paper claims that this approach achieves an AUROC of 0.7680.

a)CheXNet architecture

CheXNet [11] architecture consists of DenseNet [12] of 121 convolutional neural network. DenseNets are divided into DenseBlocks, where the dimensions of the feature maps remains constant within a block, but the number of filters changes between them. These layers between the blocks are called Transition Layers and implement the downsampling applying a batch normalization, a 1x1 convolution and a 2x2 pooling layers. Since we are concatenating feature maps, this channel dimension is increasing at every layer. In DenseNet [12] architecture each layer receives inputs from all the preceding layers and passes it's own information to all subsequent layers, which means that the final output layer has direct information from every single layer including the very first layer. The DenseNet [12] Architecture can be viewed in Fig.1

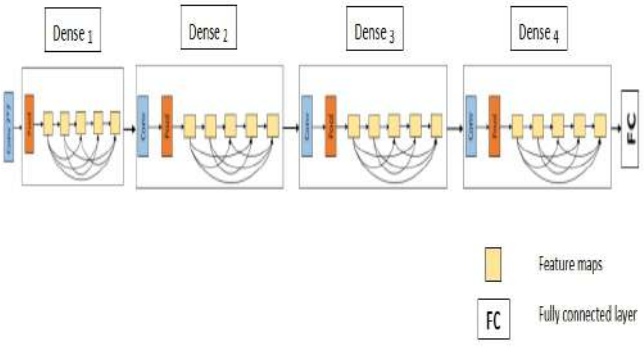


Fig. 1 Dense Net Architecture

b)Training

As suggested by the ChexNet paper [11], we trained the modified Densenet-121 architecture [12] as a binary classifier on the 14-class dataset curated by Wang et al.[3], with pneumonia label being our subject abnormality grouped as positive and the rest thirteen, grouped as negative. The model was trained on the data described in the data section above. The model was trained on NVIDIA 1080 Ti of 11Gb GPU [13] with images of dimension 224x224x1 with batch size of 5. In order to overcome over-fitting and to make the model more generalized to Chest Radiographs of various qualities, several augmentation techniques such as rotation, horizontal flipping, random brightness, random contrast, random Gaussian blur noise were applied on images. Other hyper parameters used were as follows, initial learning rate of 0.001 with reducing learning rate on plateau with patience rate of 2 and reducing factor of 0.1, minimum learning rate of le-8 with monitoring factor as validation loss, Stochastic gradient descent(optimizer), binary cross entropy as loss function. AUROC score was used in callbacks to evaluate and select best weights generated during training hence early stopping was used if AUROC score did not improve for about 10 epochs. Training was done in two stages, in the first stage 0.8:0.2 ratio of pneumonia and normal images were used respectively. Following which the best weight of first stage was selected and used for transfer learning of second stage, where in this stage 0.6:0.4 ratio of pneumonia and normal images were used respectively

2)Retina Net Approach

RetinaNet [1] is one of the best one-stage object detection models that has proven to work well for detecting objects of all scale. RetinaNet [1] has been formed by making two improvements over existing single stage object detection models - Feature Pyramid Networks (FPN) [14] and Focal Loss [1]. For this reason, it has become a popular object detection model to be used in biomedical domain. For example it has been used in - Detection of masses in mammograms using a one-stage object detector based on a deep convolutional neural network[15], DeepRetinaNet for Dynamic Left Ventricle Detection in Multiview Echocardiography Classification [16], Short-term Reproducibility of pulmonary nodule and Mass Detection in chest Radiographs: comparison among Radiologists and Four Different Computer-Aided Detection with convolutional neural net[17].

a)Retina Net Architecture

The architecture chosen was a variant of single shot object detector known as RetinaNet Architecture [1] with Residual Network (ResNet) [2] comprising of 101 layers as its backbone. It also consists of feature pyramid network (FPN) [14] to which inputs are the feature maps from the 4 blocks of ResNet-101 [2] respectively. FPN[14] constructs top down architecture with lateral connections, thus generating high-level semantic feature maps at all scales. These pyramids are scale-invariant which enables the model to detect objects across a large range of scales by scanning the model over both positions and pyramid levels. The main advantage of using each level of an image pyramid is that it produces a multi-scale feature representation in which all levels are semantically strong. Therefore, this helps the model learn less obvious features present in the regions enclosed by the anchors quicker than a regular convolutional network which is fed without localizing annotations. The feature maps obtained from FPN [14] is then sent to a

classification and regression block to generate class probabilities and offset values for the anchor box refinement, used in the steps thereafter. The RetinaNet Architecture [1] can be viewed in Fig.2

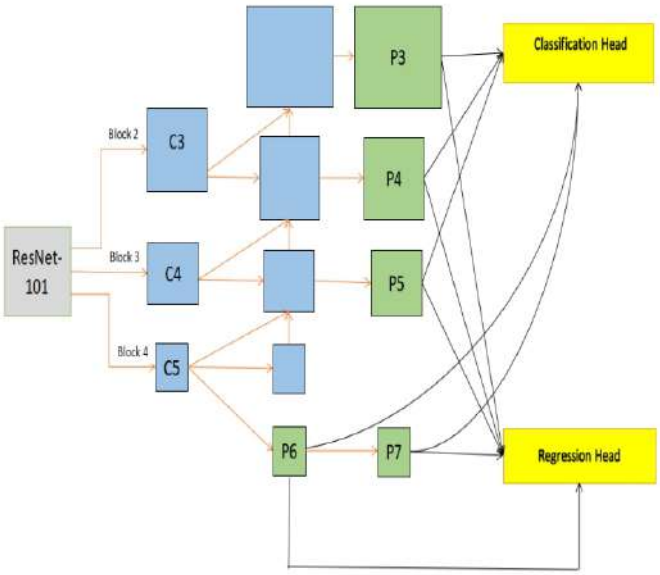


Fig 2 RetinaNet Architecture

b)Training

The model was initialized with pre-trained weights trained on ImageNet datasets. The network was trained with Stochastic Gradient Descent (SGD) as the optimizer with an initial learning rate of 1e-2, Regression loss (smooth L1 loss) with sigma of 3.0, Focal loss[1] with alpha of 0.25, gamma of 2.0, learning rate decay with patience rate of 2, batch size of 5, input image dimensions of 512x512x3 and ground truth annotations of bounding box format along with labels associated with it. Augmentation techniques were used to avoid over fitting of the model and to make it more generalized to various quality of input images as seen on live work list of ER environment. Initially during first stage, the model was trained on a training data set which consists of 0.8:0.2 ratio of pneumonia and normal images respectively. Weights with best regression and classification loss were chosen and used for transfer learning for the 2nd stage of training which was done on training data set comprising of 0.6:0.4 ratio of pneumonia and normal images respectively.

Finally best weights were chosen which had classification loss of 0.35 and regression loss of 1.06 as shown in figures Fig.3 and Fig.4 respectively.

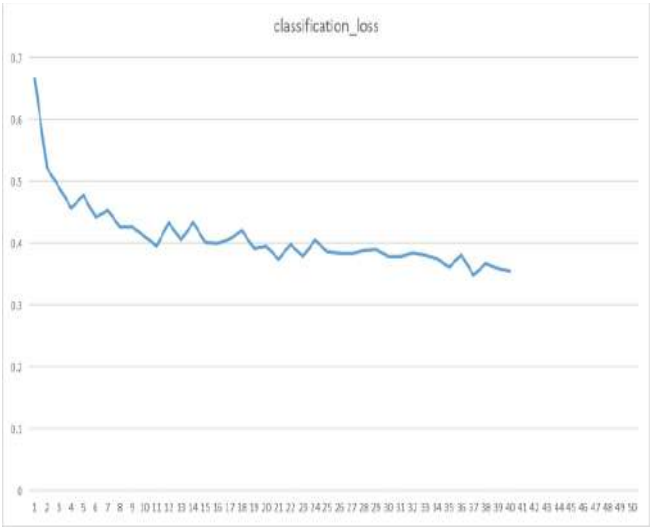


Fig 3 Classification loss

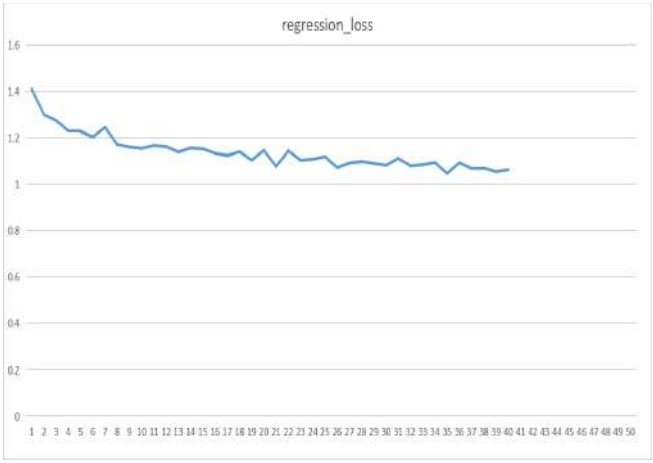


Fig 4 Regression loss

III.RESULTS AND OBSERVATIONS

1.CheXNet APPROACH

This approach yielded a sensitivity of 84.5%, specificity of 80.7%, and an AUROC of 0.84 as shown in Fig.5 However, the architecture fared badly in localization of pneumonia. Irregular boundaries cut across the lung space and stray beyond the body silhouette to even the labels, giving us imprecise markers for location and quantification. As shown in the Fig.6 to Fig.9 , this affects the interpretability of the results greatly. This observation led us to look to an efficient level of labeling- annotating of abnormality using enclosing bounding boxes over a simple yes/no verdict.

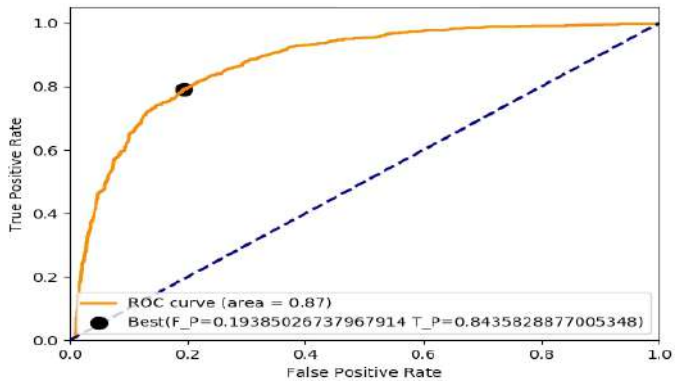


Fig. 5 AUROC of CheXNet

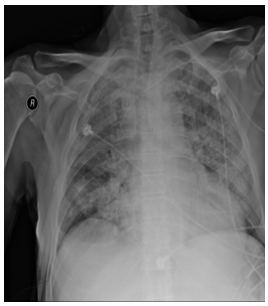


Fig. 6 Input to CheXNet

Fig. 7 CAM output of CheXNet Improper localization

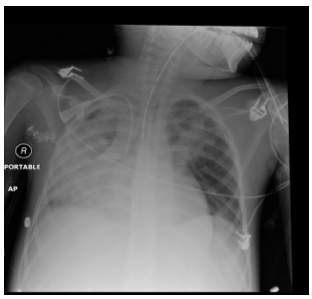


Fig. 8 Input to CheXNet

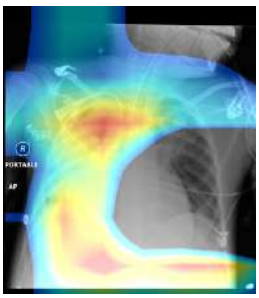


Fig. 9 CAM output of CheXNet Partially correct localization

2.RETINA NET APPROACH

The following metrics were used to evaluate the performance of both the deep learning approaches as seen in Equations (1) to (8).

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + TN + FN + FP)} \quad -- \text{ Equation (1)}$$

$$\text{Focal loss} = [-GT * (\alpha * ((1-P)^{\gamma}) * \log(P))] + [-(1-GT) * ((1-\alpha) * ((P)^{\gamma}) * \log(1-P))] \quad -- \text{ Equation (2)}$$

$$\text{regression\_diff} = \text{absolute}(\text{regression} - \text{regression\_target})$$

$$\text{Regression loss} = \begin{cases} 0.5 * \sigma^2 * \text{regression\_diff}^2, & \text{regression\_diff} < (1/\sigma^2) \\ \text{regression\_diff} - (0.5/\sigma^2), & \text{regression\_diff} > (1/\sigma^2) \end{cases} \quad -- \text{ Equation (3)}$$

$$\text{Recall/Sensitivity} = \frac{TP}{(TP + FN)} \quad -- \text{ Equation (4)}$$

$$\text{Specificity} = \frac{TN}{(TN + FP)} \quad -- \text{ Equation (5)}$$

$$\text{Precision} = \frac{TP}{(TP+FP)} -- \text{ Equation (6)}$$

$$\text{F1 score} = 2 * \frac{(\text{Precision} * \text{recall})}{(\text{Precision} + \text{recall})} -- \text{ Equation (7)}$$

$$\text{F2 score} = 5 * \frac{(\text{Precision} * \text{recall})}{(4 * \text{Precision} + \text{recall})} - - \text{ Equation (8)}$$

Where, TP (True Positive) - Chest Radiographs containing pneumonia were flagged as positive.

FP (False Positive) - Chest Radiographs without pneumonia were flagged as positive.

FN (False Negative) - Chest Radiographs with pneumonia were flagged as negative.

TN (True Negative) - Chest Radiographs without pneumonia were flagged as negative.

The RetinaNet approach yielded an accuracy of 96.33%, sensitivity of 97.51%, specificity of 95.55%, precision of 93.77%, F1 score of 95.6%, F2 score of 96.73% and AUROC of 0.97 as shown in Fig.10, results with bounding boxes and probability is shown in figures Fig.11 and Fig.12

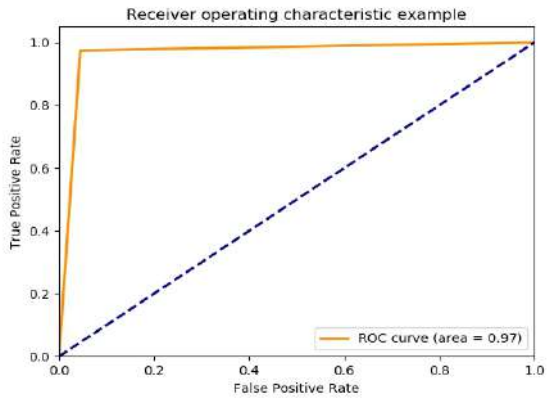


Fig. 10 AUROC of RetinaNet

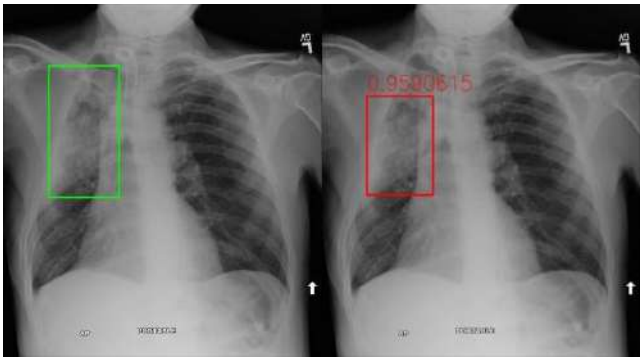


Fig. 11 Ground truth and output of RetinaNet

Fig. 12 Ground truth and output of RetinaNet

IV.CONCLUSION

Though the CheXNet approach yielded reasonable results as a classification task, it was unable to localize regions of pneumonia accurately in its heat maps. This impelled us to look to RetinaNet as a prospective solution to our pneumonia quantification challenge.

This architecture, originally intended to detect and localize objects in real-life situational photographs and images, has been successfully adapted to perform the same functions on radiographs. The observation that a RetinaNet trained model has the capability to pick up ambiguous and ill-defined instances of such abnormalities like pneumonia is a sign of promise for our chosen architecture and training process in the realm of medical imaging. Further research into developing similar models and domain-relevant dataset augmentation techniques is strongly encouraged by the findings of this study.

V.Acknowledgment

We would like to thank Keras [18] and Tensorflow [19] for providing us a reliable framework, which eases the work involved in prototyping and experimentation of deep learning algorithms. We would like to thank Kaggle [4] for hosting the “RSNA Pneumonia Detection Challenge” competition which reduced the work involved in data sourcing, annotation and provided us a platform to learn from many bright minds. We would like to thank Image Core Lab [5] for providing and annotating the dataset. We would like to thank Fizyr for their Keras implementation of RetinaNet [20].

VI.References

[1] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, Piotr Dollár, “Focal Loss for Dense Object Detection”, arXiv:1708.02002v2 [cs.CV]

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, “Deep Residual Learning for Image Recognition”, arXiv:1512.03385v1 [cs.CV]

[3] Dataset, Wang X, Peng Y, Lu L, Lu Z, Bagheri M, Summers RM. ChestX-ray8: Hospital-scale Chest X-ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases. IEEE CVPR 2017, http://openaccess.thecvf.com/content_cvpr_2017/papers/Wang_ChestX-ray8_Hospital-Scale_Chest_CVPR_2017_paper.pdf

[4] Dataset, https://www.kaggle.com/c/rsna-pneumonia-detection-challenge

[5] World Health Organization, https://www.who.int/news-room/fact-sheets/detail/pneumonia

[6] Neuman et al., 2012; Davies et al., 1996; Hopstaken et al., 2004

[7] Lakhani, Paras and Sundaram, Baskaran. Deep learning at chest radiography: Automated classification of pulmonary tuberculosis by using convolutional neural networks.Radiology, pp. 162326, 2017.

[8] Huang, Peng, Park, Seyoun, Yan, Rongkai, Lee,Junghoon, Chu, Linda C, Lin, Cheng T, Hussien,Amira, Rathmell, Joshua, Thomas, Brett, Chen,Chen, et al. Added value of computer-aided CT image features for early lung cancer diagnosis with small pulmonary nodules: A matched case-control study.Radiology, pp. 162725, 2017.

[9] Islam, Mohammad Tariqul, Aowal, Md Abdul, Minhaz, Ahmed Tahseen, and Ashraf, Khalid. Abnormality detection and localization in chest x-raysusing deep convolutional neural networks.arXivpreprint arXiv:1705.09850, 2017

[10] Demner-Fushman, Dina, Kohli, Marc D, Rosenman, Marc B, Shooshan, Sonya E, Rodriguez, Laritza, Antani, Sameer, Thoma, George R, and McDonald,Clement J. Preparing a collection of radiology examinations for distribution and retrieval. Journal of the American Medical Informatics Association, 23(2):304–310, 2015

[11] Pranav Rajpurkar, Jeremy Irvin, Kaylie Zhu, Brandon Yang, Hershel Mehta, Tony Duan, Daisy Ding, Aarti Bagul, Curtis Langlotz, Katie Shpanskaya, Matthew P. Lungren, Andrew Y. Ng, “CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning”, arXiv:1711.05225v3 [cs.CV]

[12] Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger. “Densely Connected Convolutional Networks“, arXiv:1608.06993v5 [cs.CV]

[13] GPU, NVIDIA, https://www.nvidia.com/en-in/

[14] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection”, .InCVPR, 2017

[15] Jung H, Kim B, Lee I, et al. Detection of masses in mammograms using a one-stage object detector based on a deep convolutional neural network. PLoS One. 2018;13(9):e0203355. Published 2018 Sep 18. doi:10.1371/journal.pone.0203355

[16] Meijun Yang, Xiaoyan Xiao, Zhi Liu, Longkun Sun, Wei Guo, Lizhen Cui, Dianmin Sun, Pengfei Zhang, Guang Yang, "Deep RetinaNet for Dynamic Left Ventricle Detection in Multiview Echocardiography Classification", Scientific Programming, vol. 2020, Article ID 7025403

[17] Kim, YG., Cho, Y., Wu, CJ. et al. Short-term Reproducibility of Pulmonary Nodule and Mass Detection in Chest Radiographs: Comparison among Radiologists and Four Different Computer-Aided Detections with Convolutional Neural Net. Sci Rep 9, 18738 (2019). https://doi.org/10.1038/s41598-019-55373-7

[18] Framework, https://keras.io/

[19] Framework, https://www.tensorflow.org/.

[20] Repository, https://github.com/fizyr/keras-retinanet/

Predicting demand offset to react to unforeseen critical events

Priyanka Telang
AI Architect for OMS and Supply Chain Solution,
IBM Sterling Insights & Intelligence Expert Labs,
Bangalore, India
pritelan@in.ibm.com

Mamatha Venkatesh
AI Architect for Supply Chain Control Tower Solutions,
IBM Sterling Insights & Intelligence Expert Labs,
Bangalore, India
mamathakv@in.ibm.com

Naveen Yadav
AI Consultant - Supply Chain insights, IBM Sterling Insights & Intelligence Expert Labs,
Bangalore, India
nayadav4@in.ibm.com

Nithin Mathew
AI Consultant - Supply Chain insights, IBM Sterling Insights & Intelligence Expert Labs
Bangalore, India
nithin10@in.ibm.com

Prarthana M J
AI Consultant - Supply Chain insights, IBM Sterling Insights & Intelligence Expert Labs
Bangalore, India
prarmj10@in.ibm.com

***Abstract*— A lot of industries throughout the supply chain pipeline like manufacturing, retail, and even the services supply chain relies on demand forecasts to anticipate and plan. They use the demand forecasts to plan all activities like production speed, resource allocation, raw materials purchase, and even stock levels. They usually rely on standard demand forecasting models that considers historic demand over a period during different sales cycles to determine the future demand. But in certain unforeseen scenarios be it a catastrophic weather event, or a pandemic like the one we are going through, or even a new trend causing a sales spike, the previously forecasted demand might be inaccurate and causes disruptions in the supply chain due to shortage of supply. The paper discusses a method to continuously monitor the external events in real time, stream such events, cluster them and then arrive at an offset in the demand based on what was seen in the past and the context of occurrence of the events, like the point in the sales cycle when the event is occurring, domain attributes like weather and other sociopolitical climate. This helps to increase the reliability of the demand forecasts and helps supply chain planners react to such unforeseen events effectively and improve the resilience of the supply chain.**

Keywords—supply chain, demand offset, demand forecast.

I. Introduction

With the technological advancement, the supply chain is transformed to insights and data driven. Demand Forecast is one of the crucial insights for strategic planning. For instance, a small change in demand at retail level could cause greater impact in demand to wholesalers, distributors, manufactures etc. This is referred as “bullwhip effect”.

Demand forecast in the process of predicting the sales spike based the previous sales trends, seasonal sales, type of goods etc. The demand forecast is used by supply chain planners for efficient inventory planning. In the year 2001, the failure of the demand planning software for Nike incurred a loss of $100 million on the sales [2]. Therefore, the accuracy of demand forecasted is directly impacts the revenue. One more example is Kimberly-Clark invested on the real time demand trends which accelerated the growth of their annual sales on an average of 5 percent [3]. Hence, the Demand forecast helps in formulating strategic and long-range plans of a business-like budgeting, financial planning, sales and marketing plans, capacity planning, risk assessment and mitigation plans [1].

Demand forecast is part of Supply chain process like Customer Relationship Management (CRM), Order Fulfillment (OF), Manufacturing Flow Management (MFM), Supplier Relationship Management (SRM), Product Development and Commercialization (PDC), and Returns Management (RM). [7]

II. Background and Exisiting Methodologies

A. Demand Forecast

Demand Forecast can be described as initiating the push process for supply chain operations like material planning, purchasing etc which drives the pull process like order management, distribution etc [2]. The methods to predict demand are grouped into the Quantitative assessment and Qualitative assessment. Qualitative Assessment focuses mainly on the expert’s decision and Quantitative assessment on the various Data mining techniques, Machine learning and Deep Learning algorithms etc. Other techniques are on Time series using Moving Averages and Exponential smoothening. Nowadays, Machine learning and Deep learning models are extensively used for demand forecasts.

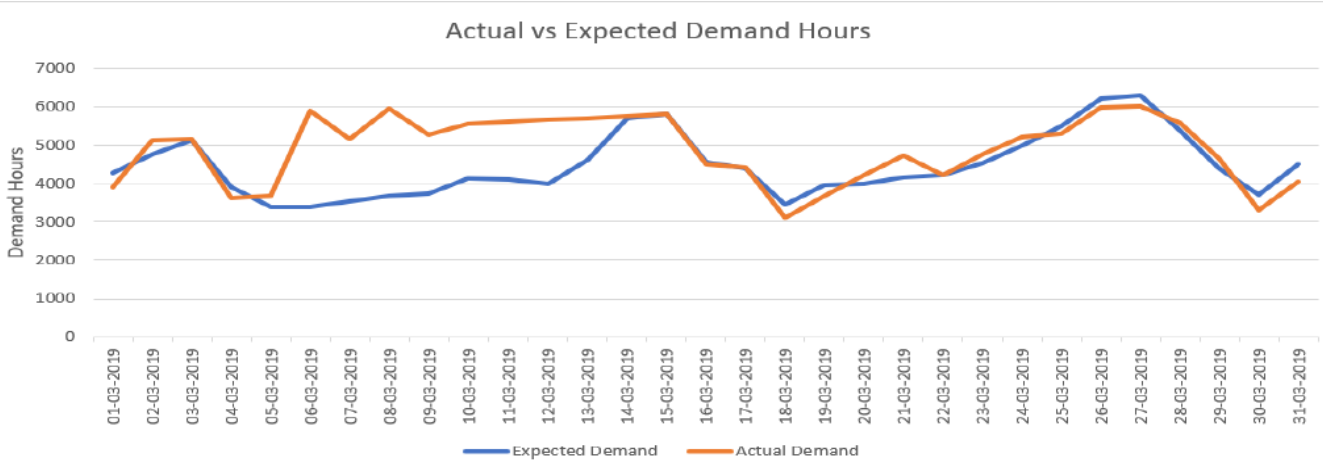


Fig. 1. Graph showing actual demand hours and expected demand hours over time

Sharma, Avinash Kumar, et al. used time series model and regression model and ensembled their results as the outcome. The algorithms used for time series are Weighted moving Average, Simple moving average, Autoregressive integrated moving average (ARIMA) and Holts Winter. For regression Support Vector Regression, Decision Trees Regression, Linear Regression, Ridge Regression and Random Forest Regression were used. The purpose to ensemble 2 models was to nullify the under and over forecast for the better accuracy [4].

J Huber et al. used the demand forecast specifically to focus on special days where the demand pattern varies than the regular days. The problem was formulated as the supervised machine learning type and evaluated it using different methods like artificial neural networks, gradient-boosted decision trees etc. Demand Forecast was outlined as classification problem rather than regression while the results had a greater accuracy for classification rather than regression method [5].

Sanjita et al. tried to reduce the bullwhip affect by applying integrated approach of discrete wavelet transforms and artificial neural network (DWT-ANN model) which would improve the forecasting accuracy. The integrated approach could be applied to both linear and nonlinear data set. In the study, the Mean square error for the DWT-ANN is comparatively less than the traditional time series models like ARIMA. [6]

Javad Feizabadi tried to improve the efficiency of demand forecast by using hybrid method of combining time series data with the leading indicators of machine learning algorithms. A possible factor accounting for inconsistency is inability of pure and non-combined approaches to demand forecasting resulting to lack of ability to handle all sources of uncertainties. This inability of forecasting approaches invoked more research by blending various ML techniques and statistical models to establish the hybrid techniques such as Autoregressive Integrated Moving Average (ARIMA) combined with ANNs [8]

III. Proposal

Current demand forecast systems rely on the seasonality in the demand for a product or service. But this reliance is shattered in case of an extreme event like a pandemic, natural disaster, extreme weather events or even a new unexpected trend. This is because extreme events like these aren't seasonal. For example, most companies manufacturing ventilators where disrupted when due the Covid-19 pandemic there was an unprecedented requirement for ventilators. Current demand planning systems are ill equipped to handle unexpected events such as the pandemic. The paper proposes an approach to correlate the effect of events such as these to demand spikes in the past and predict a context aware offset to the demand when similar events occur in the future.

A. Historic Demand and Event Correlation

The first step of the proposed solution is to correlate historic events to the spike in demand over the previously forecasted demand. Fig. 1, shows the actual versus expected demand hours for field technicians state-wide for a telecom service center for the month of march, 2019. It can be observed here that the expected demand aligns with the actual demand for most of the month (with an RMSE (Root Mean Squared Error): 398.06 hours). But by observing the graph during the portion of time between 6th march to 13th march it can be seen that the difference between the expected demand and actual demand is higher than usual.

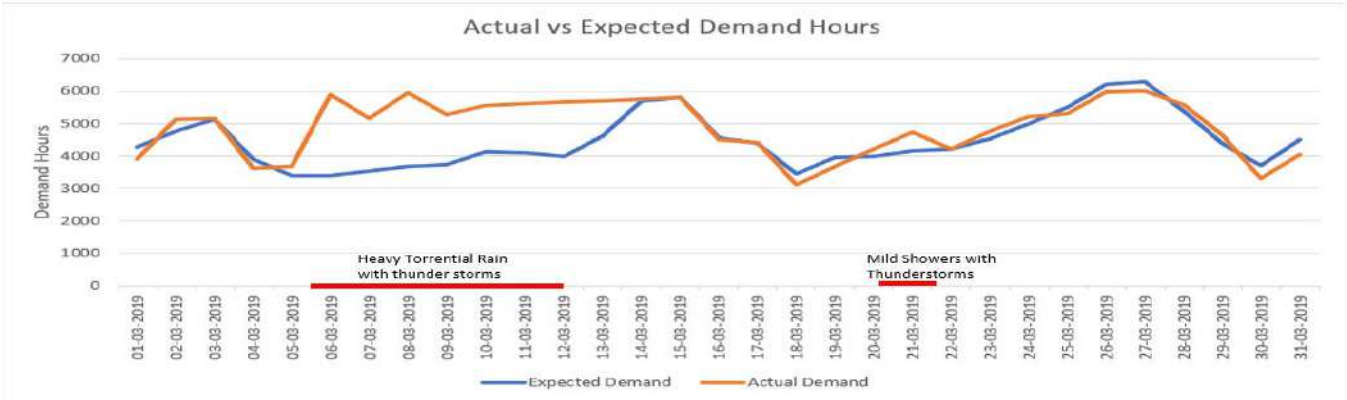


Fig. 2. Graph showing actual demand hours and expected demand hours over time with overlay of events

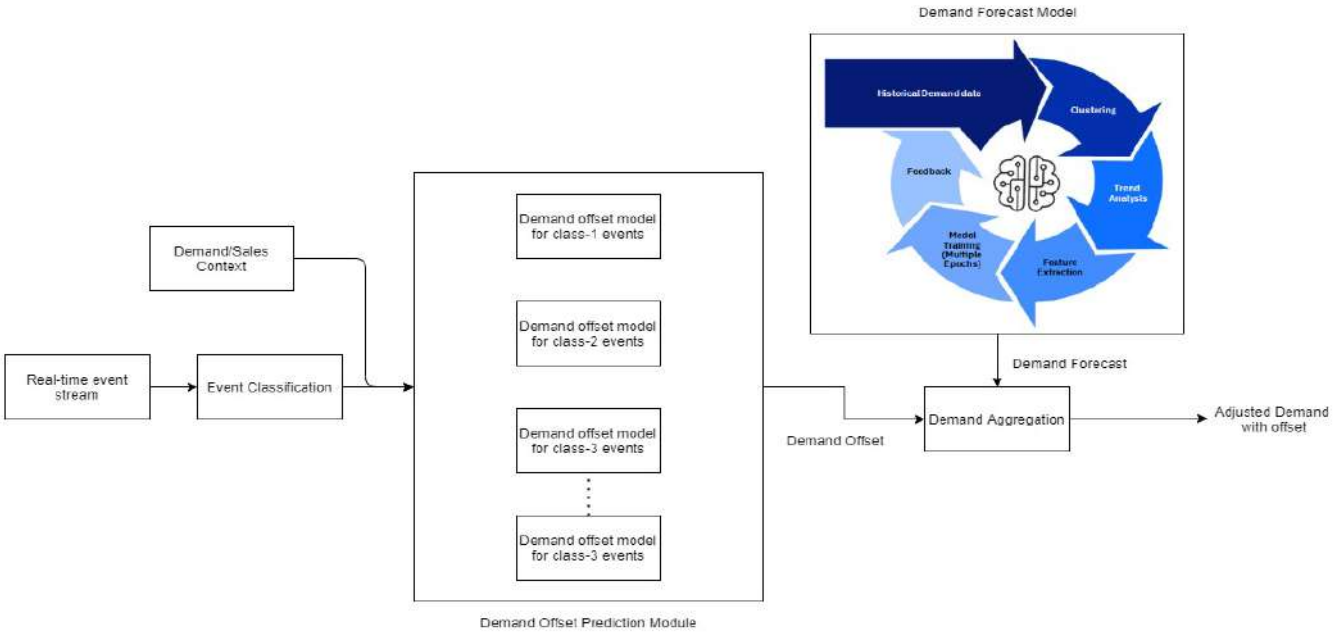


Fig. 3. Proposed process flow for demand offset calculation

Fig. 2, shows the same graph with an overlay of weather events. Here it can be seen that the larger discrepancies in the actual vs expected demand hours usually coincides with the occurrence of a severe event. This was the base intuition for the proposed idea.

The first step in correlating the event to demand offset is quantifying the event. This methodology different for different types of events. For the purpose of demonstration, the paper considers the effect severe weather events. A weather event such as rain can be quantified using features such as intensity, duration, amount, etc. Similarly, heavy winds can be quantified using wind-speed, duration, etc.

Once the features relevant to each individual event type is identified then we can collect the historic events affecting the geographic area and are grouped by event type, their relevant features, the respective difference between the actual demand to the expected demand hours i.e., demand offset during the duration of the event and the context of the event like high, low or medium sales cycle etc. Refer to example Table 1 for sample data.

Table 1. Sample Event Data with respective offset

Event Type	Event Start Timestamp	Event End Timestamp	Demand Context	Intensity	Average Offset
Heavy Torrential Rain with thunder storms	05-03-2020 02:59:00	11-03-2020 18:59:00	High Demand	59 mm/hr	2374
Mild Showers with thunder storms	20-03-2020 08:25:00	21-03-2020 05:35:00	High Demand	24 mm/hr	570
Heavy Torrential Rain with thunder storms	30-07-2020 09:20:00	05-08-2020 10:50:00	Medium Demand	62 mm/hr	1785
...	...	...	...	...	...

Once the data is collected and curated then the data is fed into a regression model considering the event attributes and the event context as the input features and the respective demand offset as the target variable. Such a similar regression model is built for each major event type to predict the offset based on the characteristics of the event.

B. Realtime Event Streaming

To predict the effect of events in real-time first the events must be collected and streamed in real-time. The paper proposes to set up a real-time stream of events either by connecting to an existing real-time event publishing stream or parse RSS feeds of government websites to obtain event alerts. For the purpose of the exploration towards this paper RSS feeds from Australia Bureau of Meteorology were consumed and parsed to obtain the events.

C. Demand Offset Prediction

Once the events are captured in real-time they can be classified to their respective event type either by rule based classification in case of simple event types such as weather in the example considered above or via clustering in case of more complex event types such as trends. The event' characteristics in conjunction with its context such as the point in the sales cycle it has occurred is passed on to the respective

regression model of the event type to predict the current offset in demand with respect to the expected demand. Refer Fig. 3. This offset can then be added to the expected demand to provide a more accurate view of the demand forecast and help planners react to such events.

IV. Results

The proposed solution helps to arrive at more accurate demand forecasts to the demand planners. This helps the planners to react to unexpected events that might affect the supply chain effectively and reduce the ripple effect on their supply chain due to inaccurate demand forecasts. This helps

them to stay on top of such issues and reduce cost to their respective domain arising from disruptions and/or loss of revenue due to unmet demand. In areas like planning of medical equipment such a model can be effectively deployed to react to increased need for medical equipment due to natural disasters or even a pandemic thereby contributing to saving lives.

REFERENCES

[1] "What is Demand Forecasting?" [Online]. Available: https://blog.arkieva.com/demand-forecasting/ . [Accessed: 13-Jan-2021]

[2] "Why is Demand Forecasting important for effective Supply Chain Management?" [Online]. Available: https://blog.arkieva.com/demand-forecasting-for-supply-chain-management/ . [Accessed: 13-Jan-2021]

[3] "Kimberly-Clark Makes Sense of Demand" [Online]. Available: https://consumergoods.com/kimberly-clark-makes-sense-demand . [Accessed: 14-Jan-2021]

[4] Sharma, Avinash Kumar, Neha Goel, Jatin Rajput, and Mohd Bilal. "An Intelligent Model For Predicting the Sales of a Product." In 2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence), pp. 341-345. IEEE, 2020.

[5] Huber, Jakob, and Heiner Stuckenschmidt. "Daily retail demand forecasting using machine learning with emphasis on calendric special days." International Journal of Forecasting (2020).

[6] Jaipuria, Sanjita, and S. S. Mahapatra. "An improved demand forecasting method to reduce bullwhip effect in supply chains."

[7] "How Smart Are Your Supply Chain Predictions?" [Online]. Available: https://towardsdatascience.com/https-medium-com-h-javedani-how-smart-are-your-supply-chain-predictions-daf5a154ac6d . [Accessed: 14-Jan-2021]

[8] Feizabadi, Javad. "Machine learning demand forecasting and supply chain performance." International Journal of Logistics Research and Applications (2020): 1-24.

Predicting missing product taxonomy in retail: An embedded approach using N-gram Mixture Models and Newton's Method

Neeraj Mishra
Science - Media
Dunnhumby
Gurgaon, India
neeraj.mishra@dunnhumby.com

Sanjay Shukla
Science - Media
Dunnhumby
Gurgaon, India
sanjay.shukla@dunnhumby.com

Anthony Kilili
Science - Media
Dunnhumby
USA
anthony.kilili@us.dunnhumby.com

***Abstract*—In retail, taxonomy is a hierarchal and logical arrangement of products such that customers can easily navigate and find what they need in the store or website. Taxonomists, information scientists, and linguistics experts all collaborate to build an effective taxonomy. Clearly this requires a lot of resources in terms of time and effort. It is not always feasible for companies to put these resources for all the products, especially newly launched products. In this research, we have developed a novel machine learning algorithm to predict a product's taxonomy by leveraging N-gram Mixture Model, cross-entropy function, and Newton's optimization method. A modified Naïve Bayes and up to 4-gram models are combined with general heuristics inspired by Jaccard Similarity. A One-vs-all classifier is trained with weights for combining different n-gram models and heuristics scores using cross-entropy loss function and Newton's optimization method. This model is developed and tested on online retail data. The model predicts the correct product taxonomy in 84% of the cases using online retail data.**

Keywords— N-gram model, Jaccard Similarity, Newton's Method, Cross-entropy, Convex Optimization

I. Introduction

Taxonomy of a product helps retailers in grouping related products together. This grouping is essential in deciding how products are displayed and which group of products are shown to which segment of customers. Imagine a customer searching for bread in the 'Bakery' category and coming up empty-handed since that product was incorrectly mapped to the 'Fresh Food' category. With limited shopping time, the customer becomes frustrated and abandons the search. When taxonomy is not properly carried out, the results are reflected in lost sales. Over time, confidence in the brand will also take a hit. On the other hand, when the right product shows up, the chance of it being purchased grows exponentially. Taxonomy organizes everything in the background to make that happen. However, this is a huge task, taxonomists, information scientists, and linguistics experts all collaborate to build an effective taxonomy. Here a big problem is that there could be thousands of categories and each of them could have varying number of products. As there are different brands/suppliers adding hundreds of products regularly, it becomes a mammoth task to group them manually.

In this research, the authors have leveraged the description of the products to build a classifier which automatically detects the product's taxonomy. One of the challenges of this approach is that most product descriptions are brief, with just few words and do not follow rules of grammar. To effectively solve such problem, a novel approach is proposed in this research which uses hybrid n-gram and heuristics (Jaccard Similarity) models. These models are then combined using convex optimization techniques. A cross-entropy loss function is used to train model. Newton method is used to optimize weights, as it has properties of quadratic convergence. Faster convergence method is required as there can be very few data-points for a class. The efficacy of the model developed in this research was validated on online retail data. The model predicts the correct product taxonomy in 84% of the cases using online retail data.

The remainder of the paper is arranged in the following manner. Section 2 discusses previous work done on various building blocks used in this research. Section 3 describes various components of the classifier and how they are put together to solve the problem at hand. Section 4 details the test suit and the experimentation results obtained by the proposed model. Finally, Section 5 concludes the paper.

II. Related Work

In e-commerce, taxonomies are often very heterogeneous, since no standardizations are being used, and hierarchies are often manually created. In the fields of ontology and taxonomy/schema matching, many different algorithms have been proposed to deal with the heterogeneity of information structures [1] [2] [3]. In this research, the problem of taxonomy prediction is treated as a text class prediction and solved with natural language processing algorithms. Text classification in NLP is an old problem and there has been a lot of earlier work done using Naïve Bayes models. In some instances, heuristics were also added to this model and treated like other features coming out of class-conditional word counts. Approaches to create document vectors have also been created using binomial or multinomial Naïve Bayes models or creating document vectors using Tf-Idf [4], which explicitly suppresses effect of stop words, present in many documents. N-grams language models [5] have been used to capture relationships between continuous words. Hybrid models for classification task, using N-grams and naïve bayes have been proposed [6]. A plethora of work has been done to create models which take into consideration the rules of grammar and can easily capture relationships between words in a sentence. Models including Hidden Markov Model [7] and Conditional Random Fields [8] gave good results.

In the past decade, there has been a lot of focus on neural network based research. Various approaches have been

proposed, including word2vec [9] and doc2vec [10], to represent words and documents in a n-dimensional space. Powerful sequence modelling methods, that make use of RNN [11] have come to light. These are further enhanced by GRUs and LSTMs [12] networks, modified to capture the importance of words that are further apart in a sentence through bi-directionality and attention networks. These methods have shown great results in many fields of active research. In addition, these methods do not require much feature engineering effort since multiple non-linear interactions are captured by neural networks. This ability comes with a requirement for enormous data points to train such a model.

The work presented in this paper presents a case with limited data. Although we are dealing with a large number of classes, an individual class can have as few as 3-4 data points. Additionally, product descriptions are not very long and do not follow rules of grammar, hence there is not much advantage in using neural networks approaches.

III. Methodology

This section details various building blocks of the proposed solution.

A. Naïve Bayes

Naïve Bayes [2] is a simple Bayesian method of classification, having an assumption that features are independent of each other given class name.

$$P(W) = \frac{P(C)\prod_{i=1}^{n} P(C)}{P(W)} \quad \dots (1)$$

Although this is a strong assumption, it seems to be fair when product descriptions just have few words on average and does not follow grammar rules. Things which must be kept in mind while creating n-grams is that, there is an explosion on word vocabulary and hence memory space required to store trained model could be huge. Also, when more n-grams are used, there are chances of overfitting. In this paper, number of n-grams to be used is a tunable parameter.

B. N-grams

Instead of using a single n-gram naïve bayes classification model, weighted average of probabilities coming out of these models is used. Normally output of a n-gram model is calculated in a below way

$$C^{*} = argmax\{P_k(W)\} = argmax\{P_k(C)P(C)\} = argmax$$

Where,

$k = size\ of\ n-gram\ used.\ For\ unigram,\ k = 1,\ for\ bi$

$W_{i:i+k} = n-gram\ of\ length\ k,\ starting\ from\ index\ i$

$n = length\ of\ unique\ words\ in\ a\ product\ description$

As output of multiple n-gram (unigram, bigram, …) models are used for weighted average, just finding most probable class from a single n-gram model is not enough. Therefore, class score for each model is calculated.

$$\hat{P}_k(W) \approx P(C)P_k(C) = P(C)\prod_{i=1}^{n-k} P(C) \dots (3)$$

As there are thousands of classes, calculating this score for each of them is a costly operation. Here a simple observation is used. For a given description, score of only those classes are calculated which have non-zero counts in available n-grams.

$$grams = \{w_1 : w_{1+k},\ w_2 : w_{2+k},\ \dots,\ w_{n-k} : w_n\} \dots (4)$$

$$C_e = \{C_j \mid \sum_{i=1}^{n-k} count(C_j,\ grams_i) > 0,\ C_j \in C_{universe}\} \dots (5)$$

$grams = n-grams\ created\ from\ given\ product\ description$

$count(C_j,\ grams_i) = number\ of\ products\ where\ grams_i\ is\ presen$

$C_{universe} = set\ of\ unique\ classes\ present\ in\ data$

Scores calculated using (3) have extremely small values. Normally, this is not much of an issue when only a class name is required having maximum score. However, here these class scores are combined, and a classifier is learnt. Extremely small values can cause floating point errors and hence make it harder for an optimizer to converge. Therefore, all class scores are divided by a normalisation constant (6). For a given model, this brings these scores in a probability space.

$$P_{norm} = \sum_{i=1}^{|C_e|} \hat{P}_k(W) \dots (6)$$

$$P_k(W) = \frac{\hat{P}_k(W)}{P_{norm}} \dots (7)$$

$|C_e| = number\ of\ elements\ in\ set\ of\ eligible\ class$

$P_k(W)$ of (7) is the score of a class, using k-gram model. Multiple such scores are calculated using different number of n-grams.

C. Heuristic

It was observed that n-gram model gives higher weightage to words present in popular classes. For example, let us assume that a product description has three words. There is one word which is present many times in one popular class, while only a couple of times in actual, not so popular class. However, when likelihood using other two classes is calculated, it is observed that actual class is present for one more word, but likelihoods are low. In these scenarios, more weightage should be given to classes, which have non-zero-word count for most of the words. Jaccard index is a simple method of measuring similarity between two sets [13].

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \dots (8)$$

In this text-classification context, similarity between a class and given set of words is observed. When a class has non-zero-word count for all the words in product description, it will have max value, which is one. In this context, this is expressed in (9).

$$J_1(W) = \frac{\sum_{i=1}^{|W|} I(count(C \cap w_i))}{|W|} \dots (9)$$

I is a zero-one indicator function, as given in (10).

$$I(f) = \{1,\ \ if\ f > 0,\ \ 0,\ \ otherwise \quad (10)$$

To give higher weightage to words present in most of the classes, scores from (9), is normalised with log-sum-exp

To give higher weightage to words present in most of the classes, scores from (9), is normalised with log-sum-exp [14] of all eligible class scores. This accentuate difference between high- and low-class scores. Also, this

$$J_1(C_i|W) = \frac{\acute{J}_1(C_i|W)}{log\left(\sum_{j=1}^{|C_e|} exp\left(\acute{J}_1(C_j|W)\right)\right)} \quad \dots(11)$$

function smoothens out scores, giving close to 1 value to a class having higher scores than rest of the class. If lots of class have similar scores, then all will get low scores.

In this section, class scores calculation are presented using just uni-gram. However, this is a tunable parameter and scores can be measured using bi-gram, tri-gram, etc. as well.

D. Loss Function

From above models, a class gets multiple kind of scores. These scores are then combined using a sigmoid function [15][16]. Each class will have their own set of weights. These weights are trained using a cross-entropy loss function [17].

$$Z(X_c) = \sum_{i=1}^{K} {}_c^n X_i P_i(C|W) + \sum_{l=1}^{S} {}_c^j X_l J_l(C|W) \quad \dots(12)$$

$$F(X_c) = \frac{1.0}{1 + exp^{-Z(X_c)}} \quad (13)$$

$$objective: \ \mathcal{O}(X_c) = \min_{X_c}\left(-log\left(F(X_c)\right)\right) \quad \dots(14)$$

X_c
$= weight\ vector\ for\ a\ class\ c, having\ dimension\ of\ K + S$
$K = number\ of\ n-grams\ used\ in\ (7)$
$S = number\ of\ n-grams\ used\ in\ (11)$
${}_c^n X_i = weight\ for\ scores\ generated\ in\ (7)\ using\ i - grams, for\ class\ c$
${}_c^j X_l = weight\ for\ scores\ generated\ in\ (11)\ using\ l - grams, for\ class\ c$

This is a one-vs-all classifier. Loss function is calculated only for a class which is true for a given product description. $\boldsymbol{X_c}$ represents a row (for class c) from weight matrix $\boldsymbol{X}$, which has a dimension of $|C_{universe}| \times (K+S)$. Here, K represents number of different models trained using (7), and S represents number of different scores generated using (11). Both K and S are hyperparameters.

E. Optimization

Starting weights during this optimization is an important aspect. Initially, equal weightage is given to all scores, calculated using (7) and (11). As more evidence is collected for a class, its corresponding weights are updated. Also, it is ensured that convex combination of scores are taken.

$$X_{init} = \mathcal{J}_{|C_{universe}|,(K+S)} \times \frac{1.0}{K+S} \quad \dots(15)$$

$\mathcal{J}_{|C_{universe}|,(K+S)}$
$= a\ mtrix\ of\ all\ ones, having\ |C_{universe}|\ rows\ and\ (K + S)\ columns$

There are thousands of classes, some having less than 10 rows in training set. An optimizer is required which can converge quickly. Newton's method [18] has this property of faster convergence. However, directly updating weights using this method can cause large updates. A normalization of Newton step is done, just before weight updation.

$$\Delta x_{nt} = -\nabla^2 \mathcal{O}(X_c)^{-1} \nabla \mathcal{O}(X_c) \quad \dots(16)$$

$$\Delta x_{step} = \frac{\Delta x_{nt}}{||\Delta x_{nt}||_1} \quad \dots(17)$$

$$\hat{X}_c^{i+1} = X_c^i + \Delta x_{step} \quad \dots(18)$$

$$X_c^{i+1} = \frac{\hat{X}_c^{i+1}}{2.0} \quad \dots(19)$$

$\nabla^2 \mathcal{O}(X_c)^{-1} =$
$inverse\ of\ hessian\ of\ loss\ function, w.r.t.\ weights\ of\ a\ class\ c$
$\nabla \mathcal{O}(X_c)$
$= gradient\ of\ loss\ function\ w.r.t\ weights\ of\ a\ class\ c$
$||\Delta x_{nt}||_1 = \ell_1 - norm\ of\ \Delta x_{nt}$
$X_c^i = weight\ for\ class\ c, at\ i^{th}\ iteration$

After optimizing a loss function (of (14)) using Newton method, weights of features having low values are updated with higher numbers. In context of (12), it would be more desirable to give higher importance to scores of (7) and (11). This implies that for a true class, if $P_2(C|W)$ has value close to one in training sample, corresponding weights should be given higher importance during inference task. To achieve this, scores of (7) and (11) are first subtracted from 1.0001, and then passed to Newton method for updating weights.

F. Prediction

During inference task, $F(X_c)$ of (13) is calculated for all eligible classes, C_e (identified using (5)). Class having max score is output of this approach.

$$C^* = \underset{C \in C_e}{\text{argmax}}(F(X_c)) \quad \dots(20)$$

IV. Results

This algorithm was tested on publicly available Instacart data [19]. This dataset is a relational set of files describing customers' orders over time. It contains a sample of over 3 million grocery orders from more than 200,000 Instacart users. In this research, "aisles.csv" and "products.csv" datasets were merged and treated as a taxonomy data. We had 49,000 distinct products and 134 aisle values in the resultant data set. Table 1 shows a sample of the dataset.

Table 1: Sample Taxonomy data

Product Name	**Aisle**
Chocolate Sandwich Cookies	cookies cakes
Dry Nose Oil	cold flu allergy
Pure Coconut Water With Orange	juice nectars

80% of the dataset was used to fit the model while 20% was used to validate the predicted aisle.
Figure 1 represents distributions of product name's word lengths.

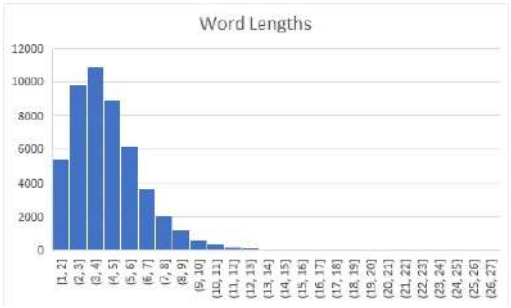


Fig 1: Histogram of word lengths

By looking at Fig 1, it becomes evident that product descriptions have very few words. Also, the class (aisle) counts of this dataset are quite skewed, as shown in figure 2.

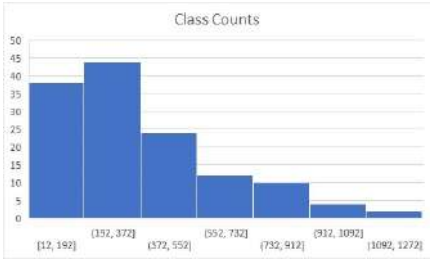


Fig 2: Histogram of aisle lengths

This implies that there are some classes which have sufficient training data points but for many, the model must be built from a small number of training points.

After conducting many iterations and fine tuning the model, parameters *K* and *S* were set to *4* and *2* respectively. The final model was able to predict missing aisles with 84% accuracy. Table 2 captures some examples of actual and predicted aisle examples.

Table 2: Predicted Taxonomy data

Product Name	**Aisle (Actual)**	**Aisle (Predicted)**
Lasting Color Shampoo	hair care	hair care
100% Whole Wheat Pita Bread	tortillas flat bread	bread
Fresh Breath Oral Rinse Mild Mint	oral hygiene	oral hygiene

In this example, model is correctly predicting "hair care" and "oral hygiene" aisle, but for the product "100% Whole Wheat Pita Bread" it is predicting "bread" instead of "tortillas flat bread". Though "bread" and "tortillas flat bread" is not exactly same, the context is not very off. This was the case for many of the 'wrong' predictions.

Therefore, we can conclude that the model proposed in this research can efficiently be used for missing taxonomy prediction. Moreover, it can be trained on extremely small number of training points and can get even more accurate as it gets more data points

V. Conclusion

In this paper, we proposed a ML based classifier for predicting missing taxonomies of products in online retail. To accomplish this, N-gram Mixture Model, cross-entropy function, and Newton's optimization method is used in a unique fashion. The algorithm is developed and tested on Instacart online retail data. It predicts the correct product taxonomy in 84% of the cases. This opens multitude of options to build a model which is explainable, goes well with general intuitions of how things should work and yet creates better decision boundaries. Going forward, relationships between different words in product description can be investigated further. In real world taxonomy dataset, there could be typos and multiple abbreviations used for same word. Also, there could be some data-points which are wrongly labelled. There are proven techniques to solve these problems in silos. But combining all of them to build an explainable AI, which does not take too much of compute resources, is an interesting research topic and needs further investigation.

References

[1] Do, H.-H., Melnik, S., & Rahm, E. (2002). Comparison of schema matching evaluations. In NODe 2002 web and database-related workshops. LNCS (Vol. 2593, pp. 221–237). Springer..

[2] Kalfoglou, Y., & Schorlemmer, M. (2003). Ontology mapping: The state of the art. The Knowledge Engineering Review, 18(1), 1–31

[3] Steven S. Aanen, Damir Vandic, & Flavius Frasincar (2015), Automated product taxonomy mapping in an e-commerce environment, Expert Systems with Applications, 42, 1298–1313

[4] Shi, C., Xu, C., & Yang, X. (2009). Study of TFIDF algorithm. *Journal of Computer Applications*, *29*(6), 167-170.

[5] Katz, S. (1987). Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE transactions on acoustics, speech, and signal processing*, *35*(3), 400-401.

[6] Awwalu, J., Bakar, A. A., & Yaakub, M. R. (2019). Hybrid N-gram model using Naïve Bayes for classification of political sentiments on Twitter. *Neural Computing and Applications*, *31*(12), 9207-9220.

[7] Morwal, S., Jahan, N., & Chopra, D. (2012). Named entity recognition using hidden Markov model (HMM). *International Journal on Natural Language Computing (IJNLC)*, *1*(4), 15-23.

[8] McCallum, A., & Li, W. (2003). Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons.

[9] Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *arXiv preprint arXiv:1310.4546*.

[10] Le, Q., & Mikolov, T. (2014, June). Distributed representations of sentences and documents. In *International conference on machine learning* (pp. 1188-1196). PMLR.

[11] Mikolov, T., Karafiát, M., Burget, L., Černocký, J., & Khudanpur, S. (2010). Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*.

[12] Yin, W., Kann, K., Yu, M., & Schütze, H. (2017). Comparative study of CNN and RNN for natural language processing. *arXiv preprint arXiv:1702.01923*.

[13] Jaccard, P. (1912). The distribution of the flora in the alpine zone. 1. *New phytologist*, *11*(2), 37-50.

[14] Nielsen, F., & Sun, K. (2016). Guaranteed bounds on the Kullback–Leibler divergence of univariate mixtures. *IEEE Signal Processing Letters*, *23*(11), 1543-1546.

[15] Gibbs, M. N., & MacKay, D. J. (2000). Variational Gaussian process classifiers. *IEEE Transactions on Neural Networks*, *11*(6), 1458-1464.

[16] Han, J., & Moraga, C. (1995, June). The influence of the sigmoid function parameters on the speed of backpropagation learning. In *International Workshop on Artificial Neural Networks* (pp. 195-201). Springer, Berlin, Heidelberg.

[17] Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1, No. 2). Cambridge: MIT press.

[18] Boyd, S., Boyd, S. P., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press

[19] Instacart data (2017), https://www.kaggle.com/c/instacart-market-basket-analysis/data

Product Based Store Clustering and Range Recommendation

Seema Mudgil
Dunnhumby
Gurgaon, India
seema.mudgil@dunnhumby.com

***Abstract*- Today's retail market is consumer driven and has become quite competitive. Shopper have wide range of options to choose in terms of products as well as in terms of stores. This has made retailers to re-look at their merchandising strategies more precisely to understand demand at a more granular level. Most retailers group stores into clusters and take strategic decisions on pricing, promotion, assortment & marketing specific to cluster behaviour. This paper provides a clustering solution by comparing two approaches i.e. product-based vs shopping mission based. Our objective to identify group of stores is to provide customer-centric assortment which in turn improves the customer shopping experience and cut down the stock cost by removing non-performing lines which eventually drives retailer growth.**

Keywords— Store Clustering, Shopping mission, k-means++, hierarchical clustering

I. Introduction

Effective store clustering is about letting consumer behavior drive assortment and space decisions, informing promotion and merchandising strategy, increasing sales and winning customer loyalty. Historically, in pursuit of operational efficiency, retailers have tended to group their stores on various aspects such as store size, sales volume, supply chain requirements, or by geography. For example, a retailer might put all large stores (based on store size) in one group, and all those smaller ones in another. There may be operational and supply chain efficiency benefits to such an approach but to maximize sales opportunities, it's imperative to understand customers preferences more deeply and cater to them more effectively while planning assortment, space etc.

An important factor to re-consider in store planning is the type of products being sold. Customers might have different preferences for different categories for e.g. a customer might prefer a premium product in dairy category but when he buys a beer, he might go for a mid-range beer. So, assortment should be driven by the customers purchase behavior at the store.

The study is conducted for one of our retailer clients. The retail organization wanted to make customer-led decision on assortment in stores and identify poor performing product ranges so they can cut down their inventory and reduce stock cost. The current range planning in store is mostly driven by store space capacity and affluence. Due to this gap in customer preferences and product ranges there was increase in stock cost, so it was imperative to re-consider the merchandising strategy in stores based on customer preferences. To achieve this aim, it is important for this retailer to better understand their customers' choices at more granular level and to make it operationally feasible it's needed to group stores based on similar customer purchase behavior. The rest of the paper is organized as follows. In section II, related work on store clustering motivations and methodologies is discussed. In section III, we present the data and methodology used for store clustering. In section IV, we have discussed about the metrics used for profiling clusters created in section III and setting up ranges for the clusters. The conclusion and future works are in section V.

II. Related Work

"Getting close to customer" is the top priority for any business. With online channel in place, it's easy to provide personalization to customers but in offline retail store there is need to place right product in the store. Store segmentation is the way to identify similarities and difference across diverse set of stores. There have been different motivations to find similar behavior stores e.g. Target Marketing [1] where similar stores are identified based on multiple attributes like customer demographics, spatial data, store size, population index etc. There are many other factors like store space capacity, climate or seasonality-based clusters, competition-based clusters that can be considered to group stores [14][15]. Clustering stores based on consumer price elasticity [16] to group stores based on price change behavior of customers and set pricing strategies accordingly. So, the objective of store clustering should drive the approach to clustering.

Different clustering techniques viz. hard clustering and soft clustering have been used by researchers [17]. Hard clustering techniques like K-means & hierarchical clustering allocates each data point to one cluster only whereas Soft Clustering techniques like FCM, Self-Organization Maps, Gaussian Mixture Model where a data point can be allocated to multiple clusters. Looking at the computational time and simplicity of the clustering solution K-means outperforms soft clustering techniques [18].

III. Data and Methodology

The retailer considered in the study operates in different formats targeted to specific customer groups based on affluence and locations. The challenge was to come up with a solution that's interpretable and offers a distinctive group of stores with reasonable difference in product performances because if there is not much difference in terms of range placed in the stores then the whole idea of store groupings is void. So, the research was done in 3 phases

- Store grouping
- Profiling and performance validation
- Range recommendation at cluster level

POS transactional data, store data and product data are leveraged to accomplish all the phases. Two approaches were explored to select final approach for clustering. To reduce noise in the analysis, important categories were selected from the product hierarchy. The selection of categories was based on the sales contribution. Pareto analysis was performed and top 16 categories (20% of total categories) covering 82 subcategories which contributed 98% of sales were selected.

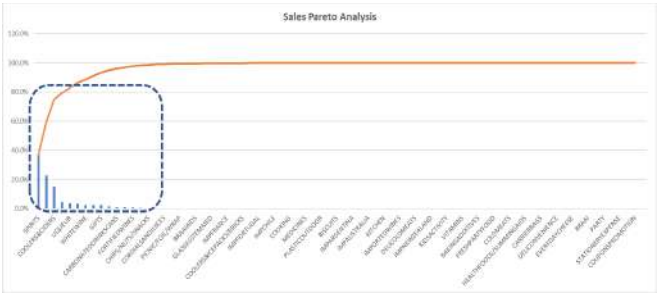


Fig 1: Sales Pareto Analysis

A. Mission based approach

The mission-based based approach requires identification of customers' shopping mission from transactional data. A shopping mission is the purpose for which a customer goes for shopping. So essentially it is a group of products/categories frequently bought by customers in single purchase [8]. For instance, the categories (beer & cigarettes) shown in Fig 2 will have more association than beer and battery.

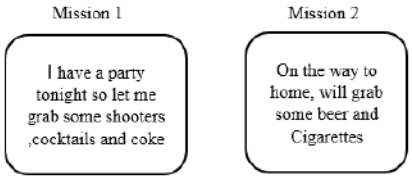


Fig 2: The figure is an example of customer shopping mission

Such association can be identified using frequent item set mining at subcategory level. The objective of frequent Item-set mining is the identification of items that co-occur [3]. Association rule mining [4] is one of the principal problems treated in KDD and can be defined as extracting the interesting correlation and relation among huge amount of transactions. Such interesting correlation between pair of categories can be calculated by looking at the difference between expected and observed frequencies of purchases of the items (categories).

$$assoc\ score(A,B) = observed(A,B)/expected(A,B)$$

$$observed(A,B) = \#\ of\ transactions\ where\ A, B\ are\ bought\ together$$

Expected joint frequency of (A,B) is measured through probability theory [5].

$$expected\ (A,B) = b(B) * b(A) * T$$

$$T = total\ baskets$$

$$b(A) = basket\ penetration\ of\ A$$

$$b(B) = basket\ penetration\ of\ B$$

Association score is measured for every pair of subcategories. Once association score is calculated it's converted into distance matrix to apply hierarchical clustering. Hierarchical clustering is a paradigm of cluster analysis to generate a sequence of nested partitions (clusters) which can be visualized as a tree known as cluster dendrogram. Hierarchical trees can provide a view of data at different levels of abstraction [6]. The dendrogram can be cut at intermediate levels for obtaining clustering results; at one of these intermediate levels meaningful clusters can be found. Hierarchical clustering solutions have been primarily obtained using agglomerative algorithms [5]. Agglomerative clustering strategies function in bottom up manner i.e. in this approach merging of the most similar pair of clusters is achieved after starting with each of the K points in a different cluster. This process is repeated until all data points converge and become members of a same cluster.

As shown in Fig 3, subcategories x, y, z are grouped at lower level so suggest a strong association. Such associations allow to understand relation between product groups and infer customers mission of purchase. Aiding the cluster analysis with business knowledge 29 missions were identified.

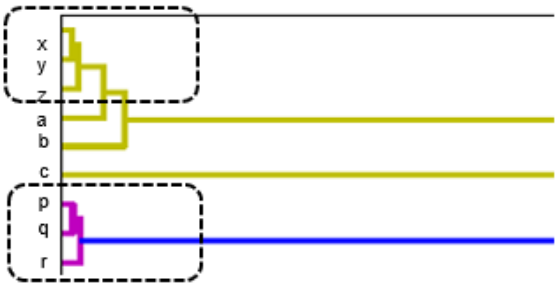


Fig 3: Category Association (mission)

Once customer probable missions are identified, data is prepared at each store on following parameters

Mission Importance	Capture importance of mission in each store using: **Basket** penetration of mission
Store Attributes	**Size** of the Store (Proxied by # of baskets where more #of transaction would mean a bigger size store) Overall **Appeal** of stores (# of customer in different economic value) **Time** of purchase (# of transactions happening in different time of the day

Fig 4: Attributes for store clustering

As shown in Fig 4, data at each store is prepared on mission importance and store attributes that produced a data matrix of 197x38. On this data, K-means++ [10] clustering algorithm was run. The k-means [9] method is a widely used clustering technique that seeks to minimize the average squared distance between points in the same cluster. K-Means++ [11] is a smart centroid initialization technique and the rest of the algorithm is the same as that of K-Means.

The number of clusters were selected based on elbow method [12]. In the elbow method, the variance (within-cluster sum of squared errors) is plotted against the number of clusters. The first few clusters will introduce a lot of variance and information, but at some point, the information gain will become low, thus imparting an angular structure to the graph. The optimal number of clusters is found out from this point; therefore, this is known as the "elbow criterion."

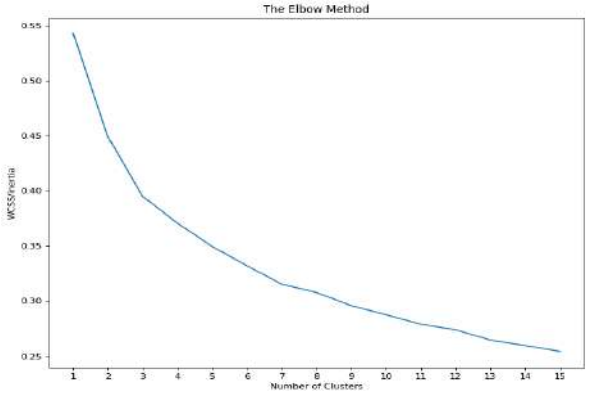


Fig 5: WSS (within cluster sum of squared error) plot

B. *Product Based Approach*

The other approach that was tested was at the most granular level of data i.e. product level. Customer purchase preferences at product level at each store was used to identify store groups with similar purchase behaviour. Some sanity checks on the data was performed as mentioned below to reduce noise in addition to category selection as mentioned in Mission based approach.

1. Store distribution: Products with good store distribution were selected. This made sure that no regional biases introduced in the data.
2. Active Selling Products: Products that are actively selling in recent time are selected so there are no obsolete products in the data that can add noise.

After adding above mentioned filters 1,519 products were selected, giving a matrix of 197x1519. When we work at granular level of data then feature space increases and Clustering algorithm can suffer from curse of dimensionality [19].To reduce data dimensions so that feature space becomes independent of each other and biasness in the clustering is removed, PCA [13] was performed on the data and feature space was reduced to 197. After feature reduction clustering was done on data through K-means++ [10].

The number of clusters were selected based on elbow method. To validate the number if clusters, dendrogram analysis were performed and that also suggested to go with 3 clusters as shown in Fig 6.

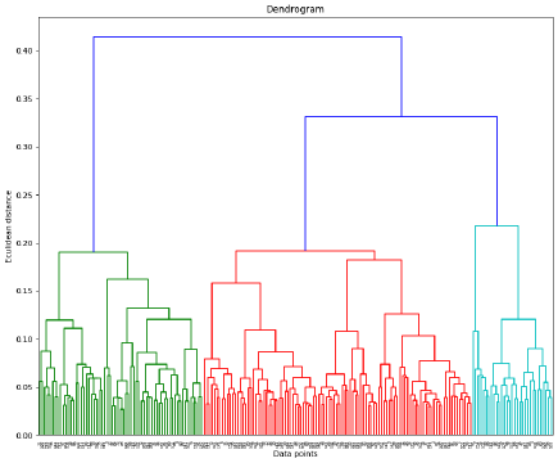


Fig 6: Dendrogram analysis

Fig 7 shows the clusters formed using Product based approach.

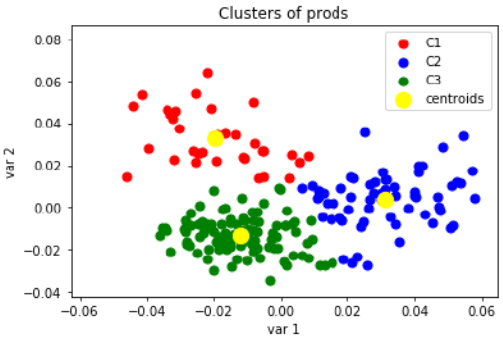


Fig 7: Product-performance based Clusters

IV. Validation , Profiling and Range Selection

A. *Selection of Clustering Method*

As explained in the above section about the two approaches for store clustering, the next step was to finalize clustering methodology that gave better range differentiation across clusters. Since the primary objective was to plan range based on customer preferences, so to finalize the approach we looked at the product performance at cluster level. Table 1 shows the performance metrics used.

Normalized Sales	Estimated weekly sales value if the product was at full distribution (sold in every store)
Normalized Units	Estimated weekly units value if the product was at full distribution (sold in every store)
Penetration	Product penetration in the customers basket when they purchase in category
Sales Contribution	Product sales contribution overall
Units Contribution	Product units contribution overall

Table 1: Product Performance Metrics

As shown in table 1, products are ranked based on their performance score on the composite score calculated on the metrics at each cluster. To identify range difference in both approaches, we looked at the top 50 products in each category in both clustering solution (Mission based & Product based).

It was observed that in categories important to retailer (top categories contributing to ~70% of sales) Product Based approach gave better range difference than Mission based clustering approach. Based on better differentiation in range across clusters formed though Product Based approach, it was selected for further investigation as mentioned in Profiling and Validation Section.

B. Profling and Validation

Post selection of approach to clustering, the next step was to bring clusters to life and show client how clusters differ in various aspects. We created cluster profiles on following measures:

- Promotion data: Products sales on promotion at cluster level to identify if cluster behave differently on promotional sales. This can be used for promotional planning.
- Category Importance: Importance of the category in each cluster to identify customer preferences for specific product ranges. To identify category importance, we looked at category basket share at cluster level and benchmarked it against at total level.

Based on promotion data, there was reasonable difference in terms of % of promotional sales across clusters. Cluster 3 had the least percentage of promotional sales which indicates that it's a Premium cluster.

Different level of preference was also evident for category in each cluster suggesting store grouping is relevant to customer preferences and each cluster should be ranged differently for each category.

C. Range Recommendation

The ranges were recommended by looking at the product performance at each cluster. The idea was to provide range based on cluster behavior for the category and identify poor performing lines at cluster level and suggest retailer to remove them. While recommending range across clusters, there were two important decision made.

- How wide the assortment should be?
- How assortment should vary by cluster

While answering first question, we looked at category importance (# of baskets/store) to identify where there is more need of range. For e.g. if a category X has 2000 baskets/store in cluster 1 and 8000 baskets/store in cluster 2, then cluster 2 needs more range/SKU to be merchandised for category X.

For looking at assortment variation, we looked at customer profiles at cluster level. By customer profile we mean socio-economic class segment So, a more premium cluster will have more premium range in the category as compare to less premium cluster.

Poor performing lines in each cluster was identified based on product performance aiding it with profit-margin data.

With this approach of range recommendation, non-performing lines at cluster level were recommended for delist. This impacted a lower stock to sales ratio, thus saving stock cost to retailer.

V. Conclusion

This paper considered two approaches to store grouping namely: Mission based, and Product performance based. Each technique has different approach of clustering, mission-based approach looks at multiple attributes i.e. mission and store attributes (size, time of purchase) to identify store groups with similar behavior. Product based approach looks at the most granular level of data to identify customer preferences variation among stores. Since the objective of store grouping here was range difference and product-based approach gave better range difference, so this approach was selected. There could be another objective as well for e.g. price perception, store capacity, climatic condition etc. So, what attribute to be used for store grouping should be driven by the objective of the business question. The recommendation that we have provided to retailer are being implemented, so next course of action will be to assess the impact of range changes. We also plan to revisit the clustering to see impact of Covid in customer behavioral changes, if any.

References

[1] Bilgic E., Kantardzic M., Cakir O. (2015) Retail Store Segmentation for Target Marketing. In: Perner P. (eds) Advances in Data Mining: Applications and Theoretical Aspects. ICDM 2015. Lecture Notes in Computer Science, vol 9165. Springer, Cham. https://doi.org/10.1007/978-3-319-20910-4_3

[2] https://www.relexsolutions.com/resources/why-category-management-should-always-start-with-behavioral-clustering/

[3] Alva Erwin, Raj P. Gopalan, N.R. Achuthan, "A BottomUp Projection Based Algorithm for Mining High Utility Itemsets", In Proceedings of the 2nd international workshop on Integrating artificial intelligence and data mining, 2007, Vol. 84: 3-11.

[4] Liu X., Zhai K., & Pedrycz W. An improved association rules mining method. Expert Systems with Applications, 2012 39(1):1362–1374. doi:10.1016/j. eswa.2011.08.018.

[5] C.H.Q Ding, X F he, H Y Zha et al, "A min max cut algorithm for graph partitioning and data clustering", Proc. Of ICDM 2001

[6] Tian Zhang, Raghu Ramakrishnan, MironLivny, BIRCH: an efficient data clustering method for very large databases, Proceedings of the 1996 ACM SIGMOD international conference on Management of data, p.103-114, June 04-06,

[7] Zhang, B., et al., 2012. PCA-subspace method — is it good enough fornetwork-wide anomaly detection. In: Network Operations andManagement Symposium (NOMS), IEEE.

[8] Griva, Anastasia; Bardaki, Cleopatra; Panagiotis, Sarantopoulos; and Papakiriakopoulos, Dimitris, "A DATA MINING-BASED FRAMEWORK TO IDENTIFY SHOPPING MISSIONS" in Mola, L., Carugati, A,. Kokkinaki, A., Pouloudi, N., (eds) (2014) *Proceedings of the 8th Mediterranean Conference on Information Systems*, Verona, Italy, September 03-05. CD-ROM. ISBN: 978-88-6787-273-2.

[9] Sariel Har-Peled and Bardia Sadri. How fast is the k-means method? In SODA '05: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms, pages 877–885, Philadelphia, PA, USA, 2005. Society for Industrial and Applied Mathematics

[10] David Arthur and Sergei Vassilvitskii, k-means++: The Advantages of Careful Seeding, https://theory.stanford.edu/~sergei/papers/kMeansPP-soda.pdf

[11] https://www.geeksforgeeks.org/ml-k-means-algorithm/

[12] https://arxiv.org/ftp/arxiv/papers/1912/1912.00643.pdf

[13] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," Chemometrics and intelligent laboratory systems, vol. 2, no. 1-3, pp. 37–52, 1987.

[14] https://www.dotactiv.com/blog/store-clustering-store-based-vs-category-based-clustering
[15] https://www.dotactiv.com/blog/clustering-methods
[16] Rooij, G., Burg, G.V., & Velden, M.V. (2017). Clustering Stores of Retailers via Consumer Behavior T hesis in Business Analytics and Quantitative Marketing.
[17] Agarwal, Kanika, P. Jain and Mamta A. Rajnayak. "Comparative analysis of Store Clustering Techniques in the Retail industry." *DATA* (2019).
[18] Dibya jyoti Bora, Dr. Anil Kumar Gupta A Comparative study Between Fuzzy Clustering Algorithm and Hard Clustering Algorithm April 2014,International Journal of Emerging Trends & Technology in Computer Science 10(2):108-113
[19]https://developers.google.com/machinelearning/clustering/algorithm/advantages-disadvantages

PyTorch Tabular: A Framework for Deep Learning with Tabular Data

Manu Joseph
Lead Data Scientist
Thoucentric
Bangalore, India
manujoseph@thoucentric.com

***Abstract* — In spite of showing unreasonable effectiveness in modalities like Text and Image, Deep Learning has always lagged Gradient Boosting in tabular data—both in popularity and performance. But recently there have been newer models created specifically for tabular data, which is pushing the performance bar. But popularity is still a challenge because there is no easy, ready-to-use library like Sci-Kit Learn for deep learning. PyTorch Tabular is a new deep learning library which makes working with Deep Learning and tabular data easy and fast. It is a library built on top of PyTorch and PyTorch Lightning and works on pandas dataframes directly. Many SOTA models like NODE and TabNet are already integrated and implemented in the library with a unified API. PyTorch Tabular is designed to be easily extensible for researchers, simple for practitioners, and robust in industrial deployments. The library is available at https://github.com/manujosephv/pytorch_tabular**

Keywords—Tabular, Deep Learning, Machine Learning, PyTorch

I. Introduction

The unreasonable effectiveness of Deep Learning that was displayed in many other modalities—like text[1] and image[2]—have not been thoroughly demonstrated in tabular data. Despite being the most used data type in real-world problems, tabular modality is relatively less explored in Deep Learning literature. The state-of-the-art performance in many problems with tabular data is often achieved by "shallow" models, such as gradient boosted decision trees (GBDT)[3] (XGBoost[4], LightGBM[5], CatBoost[6]). If "*performance*" is one dimension along which GBDTs beat Deep Learning approaches, "*popularity*" is another. If we look at the different machine learning competitions (e.g. Kaggle), we can see the popularity of GBDTs, which are almost always part of the winning solutions.

The past few years, we have seen an increased interest in this modality and many works address this modality to push the state-of-the-art. Deep Forest[7], TabNN[8], TabNet[9], Neural Oblivious Decision Ensembles[10] are just a few architectures proposed specifically for tabular modality. Out of these, NODE and TabNet has shown to beat the GBDT baselines as well.

While research has started to push the "*performance*" bar on tabular data, the "*popularity*" bar is still low. One of the primary reasons behind this is the lack of support on the software side of things. Training Deep Learning models are still and involved process with quite a bit of software engineering required. When this is compared to the ease Scikit-learn[11] and other libraries adopting the Scikit-learn API provides practitioners, we get a clue as to why popularity of Deep Learning for Tabular data is still low.

PyTorch Tabular is a library which aims to make Deep Learning with Tabular data easy and accessible to real-world cases and research alike. The core principles behind the design of the library are:

- Low Resistance Usability
- Easy Customization
- Scalable and Easier to Deploy

PyTorch Tabular attempts to make the "software engineering" part of working with Neural Networks as easy and effortless as possible and let you focus on the model. It also hopes to unify the different developments in the Tabular space into a single framework with a unified API that will work with different state-of-the-art models. It also provides an easily extensible BaseModel to aid Deep Learning researchers create new architectures for tabular data.

PyTorch Tabular is built on the shoulders of giants like PyTorch[12], PyTorch Lightning[13], and Pandas[14]. The library is released under the MIT license and is available on GitHub[1]. Detailed documentation and tutorials are available on documentation page[2].

II. Related Work

The ML community has a strong culture of building open-source tools, which both accelerated research and adoption of new techniques in the industry. Tensorflow[15], PyTorch, and similar frameworks started the journey of abstraction of Deep Learning implementation by providing automatic differentiation, pre-built fundamental blocks of Neural Networks etc. PyTorch Lightning came in and abstracted away the training loop for PyTorch and enabled easy and scalable training. PyTorch Tabular takes that journey of abstraction to the next level by providing domain specific abstraction layer.

The concept of having domain specific abstractions to train neural network models originated from fastai[16] and fastai.tabular provides an easy-to-use API for training deep learning models for tabular data. But where PyTorch Tabular is different is in the fact that it is a strongly

[1] https://github.com/manujosephv/pytorch_tabular
[2] https://pytorch-tabular.readthedocs.io/en/latest/

de-coupled implementation and relies on standard components like Base PyTorch layers, optimizers and loss functions. The training loop is handled by Pytorch Lightning, which is also growing to be a standard in the community. This makes PyTorch Tabular much more extensible for custom use cases.

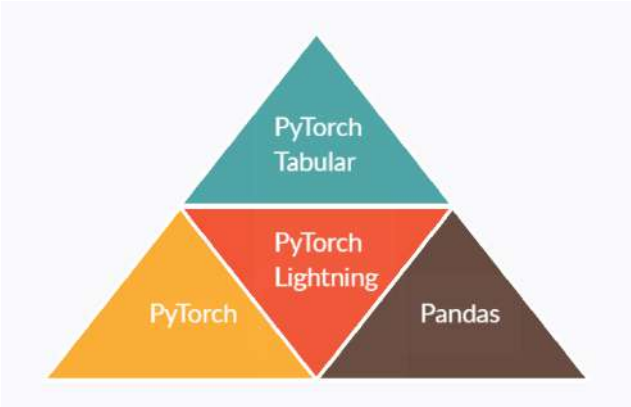


Fig. 1. PyTorch Tabular is built on strong foundations of tried and tested frameworks.

III. Library Design

PyTorch Tabular is designed to make the standard modelling pipeline easy enough for practitioners as well as standard enough for production deployment. In addition to that, it also has a focus on customization to enable wide usage in research.

Pytorch Tabular has adopted a 'config-driven' approach to satisfy these objectives.

A. Config Driven

There are 5 config files which drives the whole process-

1. *DataConfig* – *DataConfig* is where you define the parameters regarding how you manage data within your pipeline. We distinguish between categorical and continuous features, decide the normalization, or feature transformations, etc. in this config.
2. *ModelConfig* – There is a separate *ModelConfig* defined for each model that is implemented in PyTorch Tabular. It inherits from a base ModelConfig which holds common parameters like *task (classification or regression), learning_rate, loss, metrics,* etc. Each model that is implemented inherits these parameters and adds model specific hyperparameters to the config. By choosing the corresponding *ModelConfig*, Pytorch Tabular automatically initializes the right model.
3. *TrainerConfig*—*TrainerConfig* handles all the parameters to control your training process, and most of these parameters are passed down to the PyTorch Lightning layer. You can set parameters like *batch_size, max_epochs, early_stopping*, etc. in here.
4. *OptimizerConfig*—Optimizers and Learning Rate Schedulers are another integral part in training a neural network. These configurations can be done using the OptimizerConfig.
5. *ExperimentConfig* - Experiment Tracking is almost an essential part of machine learning. It is critical in upholding reproducibility. PyTorch Tabular embraces this and supports experiment tracking internally. Currently, PyTorch Tabular supports two experiment Tracking Framework—Tensorboard and Weights & Biases.

   Tensorboard logging is barebones. PyTorch Tabular just logs the losses and metrics to Tensorboard. W&B tracking is much more feature rich - in addition to tracking losses and metrics, it can also track the gradients of the different layers, logits of your model across epochs, etc.

These config files can be set programmatically as well as through YAML files, which makes this easy for both Data Scientists and ML Engineers.

B. BaseModel

PyTorch Tabular uses an abstract class—BaseModel—which implements the standard part of any model definition like loss and metric calculation, etc. This class serves as a template on which any other model is implemented and ensures smooth interoperability between the model and the training engine. Inheriting this class, the only two methods that a new model must implement are the model initialization part and the forward pass. And in case you need to do something non-standard in the loss calculation, all you have to do is overwrite the corresponding methods in your model definition.

C. Data Module

PyTorch Tabular uses Data Module, as defined by Pytorch Lightning, to unify and standardize the data processing. It encompasses the preprocessing, label encoding, categorical encoding, feature transformations, target transformations, etc. and also ensures the same data processing is applied to train and validation splits, as well as new and unseen data. It provides PyTorch dataloaders for training and inference.

D. TabularModel

TabularModel is the core component which brings together the configs, initializes the right model, the data module, and handles the train and prediction functions with methods like '*fit*' and '*predict*'.

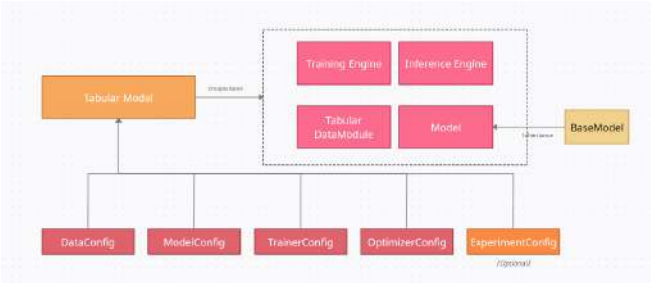


Fig. 2. Overall Structure of PyTorch Tabular

IV. Implemented Models and Unified API

PyTorch Tabular has implemented a few state-of-the-art model architectures and unified them with a single, easy-to-use API which is well suited for rapid iterations. For Deep Learning to gain popularity among practitioners, it is important to be able to provide an easily used API which can compare to the ease of use given by the "Scikit-learn" APIs.

The models which are currently implemented in PyTorch Tabular are:

A. *CategoryEmbeddingModel*

This is a standard feed-forward network with the categorical features passed through a learnable embedding layer. The model architecture is very similar to the Tabular model in fastai with BatchNorm and Dropout Layers in between standard linear layers.

B. *Neural Oblivious Decision Ensembles (NODE)*

NODE[10] is a model architecture presented in ICLR 2020 and shown to beat tuned GBDT models on several datasets. It uses a Neural equivalent of Oblivious Trees (the kind of trees CatBoost[6] uses) as the basic building blocks of the architecture.

There are two variants of this algorithm implemented in PyTorch Tabular—*NODEModel* and *CategoryEmbeddingNODEModel*. The only difference is in the way categorical features are treated. In *NODEModel*, the categorical features are encoded using LeaveOneOutEncoding[17] (as suggested by the authors) and in *CategoryEmbeddingNODEModel* the categorical embeddings are learned from data.

C. *TabNet*

TabNet is a model architecture which deviates from the tree-based hybrid design philosophy and uses Sparse Attention in multiple steps of decision making to model the output. The architecture consists of sequential learnable decision steps which includes feature selection using a learnable mask. The multiple steps create higher representations of the input data which is used for the final task.

D. *Usage*

The basic usage is fairly simple. Below is an example. We define the configs and select the *CategoryEmbeddingModelConfig* as the model config. All the parameters have intelligent defaults so that you can get started as soon as possible.

```
data_config = DataConfig(
    target=['target'],
    continuous_cols=num_col_names,
    categorical_cols=cat_col_names,
)
trainer_config = TrainerConfig(
    gpus=1, #index of the GPU to use. 0, means CPU
)
optimizer_config = OptimizerConfig()

model_config = CategoryEmbeddingModelConfig(
    task="classification"
)
experiment_config = ExperimentConfig(
    project_name="PyTorch Tabular Example"
)
```

Now that the configs are defined, we can put it all together in a *TabularModel*.

```
tabular_model = TabularModel(
    data_config=data_config,
    model_config=model_config,
    optimizer_config=optimizer_config,
    trainer_config=trainer_config,
)
```

The TabularModel takes in the configs and sets up the whole modelling pipeline. Now We just need to call the *fit* method and pass the train and test dataframes. We can also pass in validation dataframe. But if omitted, *TabularModel* will separate 20% (also configurable) at random from the data as validation.

By default, *EarlyStopping* is enabled and is monitoring validation loss with a patience of 3 epochs. The trainer also saves the best model (based on validation loss) and loads that model at the end of training. *TrainerConfig* has the parameters to tweak this default behaviour.

```
tabular_model.fit(train=train, validation=val)
```

After the training, there are three actions that you usually take in a typical modelling pipeline.

1. Evaluate the model on some new data

```
result = tabular_model.evaluate(test)
```

2. Get predictions on new data

```
pred_df = tabular_model.predict(test)
```

3. Save and Load the Model

```
tabular_model.save_model("examples/basic")
loaded_model = TabularModel.load_from_checkpoint(
    "examples/basic"
)
```

Detailed documentation and tutorials for common tasks are present in the documentation.

V. Conclusion

Deep Learning for tabular data is gaining popularity in the research community as well as the industry and in the face of growing popularity, it is essential to have a unified and easy to use API for tabular data, similar to what scikit-learn has done for classical machine learning algorithms. PyTorch Tabular is hoping to fill in that space and reduce the barrier for entry in using new state-of-the-art deep learning model architectures in industry use cases. It

also hopes to reduce the "engineering" work for researchers who are working on new model architectures.

VI. Future Work

PyTorch Tabular is a relatively new library and will continue to grow. We actively invite contributors to help maintain and grow the library. Future roadmap is available on the Github Readme. The items are along the below three paradigms:

1. Adding new models
2. Integrating Hyperparameter Tuning
3. Adding Text and Image modalities for multi-modal problems.
4. Adding new preprocessing techniques

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł Ukasz Kaiser, and Illia Polosukhin, "Attention is all you need" Advances in Neural Information Processing Systems, pp. 5998–6008, 2017.

[2] He, Kaiming, X. Zhang, Shaoqing Ren and Jian Sun. "Deep Residual Learning for Image Recognition." 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 770-778, 2016.

[3] Jerome H Friedman. "Greedy function approximation: a gradient boosting machine." Annals of statistics, pp. 1189–1232, 2001.

[4] Tianqi Chen and Carlos Guestrin. "Xgboost: A scalable tree boosting system." In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794, 2016.

[5] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and TieYan Liu. "Lightgbm: A highly efficient gradient boosting decision tree." In Advances in Neural Information Processing Systems, pp. 3146–3154, 2017.

[6] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. "Catboost: unbiased boosting with categorical features." In Advances in Neural Information Processing Systems, pp. 6638–6648, 2018.

[7] Zhi-Hua Zhou and Ji Feng. "Deep forest: Towards an alternative to deep neural networks." In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI, 2017.

[8] Guolin Ke, Jia Zhang, Zhenhui Xu, Jiang Bian, and Tie-Yan Liu. "Tabnn: A universal neural network solution for tabular data." 2018.

[9] Arik, Sercan Ö. and T. Pfister. "TabNet: Attentive Interpretable Tabular Learning." ArXiv abs/1908.07442. 2019.

[10] Popov, S. et al. "Neural Oblivious Decision Ensembles for Deep Learning on Tabular Data." ArXiv abs/1909.06312. 2020.

[11] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Édouard Duchesnay. "Scikit-learn: Machine Learning in Python" J. Mach. Learn. Res. 12. pp: 2825-2830. 2011.

[12] Paszke, Adam, S. Gross, Francisco Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, Alban Desmaison, Andreas Köpf, E. Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, B. Steiner, Lu Fang, Junjie Bai and Soumith Chintala. "PyTorch: An Imperative Style, High-Performance Deep Learning Library." NeurIPS. 2019.

[13] William Falcon (2019) PyTorch Lightning [Source Code] https://github.com/PyTorchLightning/pytorch-lightning

[14] McKinney, Wes. "Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython." 2017.

[15] Abadi, M., P. Barham, J. Chen, Z. Chen, Andy Davis, J. Dean, M. Devin, Sanjay Ghemawat, Geoffrey Irving, M. Isard, M. Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, D. Murray, B. Steiner, P. Tucker, V. Vasudevan, Pete Warden, Martin Wicke, Y. Yu and Xiaoqiang Zhang. "TensorFlow: A system for large-scale machine learning." OSDI. 2016.

[16] Howard, J. and Sylvain Gugger. "fastai: A Layered API for Deep Learning." ArXiv abs/2002.04688. 2020.

[17] Zhang O. Tips for data science competitions. 2016. https://www.slideshare.net/OwenZhang2/tips-for-data-science-competitions. Accessed 9 Feb 2021.

Real-time social distancing & face mask compliance reporting system for multiple CCTV camera feeds

Archit Kaila
Data Scientist
New Delhi, India

Shrey Gupta
Data Scientist
New Delhi, India

Tanya Kaintura
Data Scientist
New Delhi, India

***Abstract*—Millions of people have been infected by the coronavirus disease of 2019 (COVID-19) and lost their lives to it despite various measures to curb the same. To make the situation worse, a traditional observational method of in-person reporting cannot be used because it poses a risk for the observer to catch the infection. Social Distancing and Face Mask compliance, therefore, remain vital measures to curb the spread of COVID-19. We propose an end-to-end solution that can monitor different social distancing and face mask compliance metrics and be deployed efficiently in Python using open-source libraries. It is scalable and enables the users to implement the solution at a large scale, i.e., cover a broader area using multiple live camera feeds simultaneously. Our solution precisely calculates the distance between two people or objects by mapping the 2-dimensional pixel distances to 3-dimensional actual distances. These attributes make our solution unique, and it can be deployed for usage in various situations and locations such as shopping malls, supermarkets, large workspaces, manufacturing facilities, etc. which can help to dampen the effect of COVID-19 as early as possible**

Keywords— OpenCV, YOLOv3, DeepSORT, Person Detection & Tracking, Multiprocessing, Face Mask Detection, Social distancing, COVID-19

I. Introduction

COVID-19 struck the world in late December 2019, which is believed to be originated in Wuhan, China. As the virus proliferated worldwide, the World Health Organization (WHO) characterised COVID-19 as a pandemic on 11th March 2020. During this pandemic, multiple governmental leaders introduced lockdown and quarantine to suppress the spread of the disease. Around the end of March 2020, 2.6 billion people, one-third of the world's population, were under some type of lockdown. As of 15th January 2021, according to the cases reported to WHO, there are 91,492,398 confirmed cases, including 1,979,507 deaths worldwide.

Various health care organizations, medical experts have put their best efforts to develop a vaccine to fight COVID-19. The WHO issued some basic guidelines that include wearing masks and maintaining social distancing, i.e., maintaining a distance of 6. The main reason for this was that the virus takes up to 7 days to show symptoms. Till then, a person doesn't know if he/she has been affected or not. Even some people are asymptomatic to COVID-19. To stop the spread of this disease, these guidelines were implemented by the government of various countries. Vaccines are out in the market, and the vaccination process has been started, but for every person to get the vaccine throughout the world will require plenty of time. So it all comes down to slowing the rate of spreading the virus; otherwise, the hospitals will be overburdened, and things will deteriorate at a much faster pace. So given the situation, currently, the best solution is to follow the given guidelines by WHO. Research shows that we can flatten the curve if there are controlled measures.

Due to the deteriorating economy, the lockdown was lifted in most geographies of the world. People rushed to malls, parks, and various other public places to help them deal with their mental health, which became a big concern for society during the lockdown. Our proposed solution will help the authorities of shopping malls, factories, restaurants, etc., to monitor if people follow the guidelines given by WHO in real-time and can be deployed to large scale public places efficiently.

A lot of research and development has already been done in this area. Many authors have proposed deep learning-based models to monitor face mask compliance and maintenance of social distancing among the public. Most of these solutions are a proof of concept. They can't be deployed in real life on an extensive scale network of CCTV cameras; hence we came up with an end to end solution with a few novel addons so that the system can be put to use in real-time on an extensive network of CCTV cameras and help curb the spread of COVID-19.

II. Related Work

Most medical and pharmaceutical companies have been researching and trying to develop vaccines to treat COVID-19. Many pharmaceutical companies have recently launched their vaccine in a few countries for emergency use on a priority basis for the elderly and frontline workers. While these vaccines reach out to every individual globally, social distancing remains the most reliable and safest technique to curb the spreading of COVID-19 disease.

In December 2019, the initial cases of COVID-19 emerged in the city of Wuhan, China. As the spread of this infectious disease could not be contained, social distancing opted as the primary measure on 23rd January 2020 [1]. Within a month, the virus outbreak from Wuhan gained a peak in China in the first week of February, with close to 2,000 - 4,000 new cases being confirmed per day. After complying with social distancing norms, wearing masks, and other lockdown measures, there was some relief as no new positive cases were reported for five consecutive – from 18th March 2020 to 23rd March 2020 [2]. Hence, it is evident that the use of face masks, strict and correct social distancing measures imposed in China in the initial phases of the viral spread were beneficial. Therefore, these were later adopted worldwide to contain the community spread of COVID-19.

The conditions of the United States were studied by Adolph et al. [3], indicating the failure of adoption of social distancing measures at an early stage by the government resulted in harming a large number of human lives. The economic activities of the country were also shaken.

Following this study's steps, the relation between the region's financial status and the strictness of social distancing regulations in the country were observed and studied by Kylie et al. [4]. This study stipulated that intermediate levels of activities can avoid a massive outbreak of COVID-19.

The effects of social distancing measures on the widespread COVID-19 pandemic were studied by Prem et al. [5]. The authors used synthetic location-specific contact patterns to mimic this viral outbreak's growth trajectory using susceptible-exposed-infected-removed (SEIR) models. They suggested an earlier secondary peak could be seen in sudden and premature removal of social distancing norms, which could be reduced by gradually relaxing the interventions [5]. Hence, even though social distancing is logically, physically, and economically painful, it remains an essential measure to flatten the COVID-19 spread curve.

After the widespread of this pandemic across the globe, every country has been following and implementing various guidelines and regulations to contain this spread of the virus [6], [7], [8]. Many developing countries such as India made use of Bluetooth and GPS to track and locate the spread and presence of COVID-19 patients in a specified area. The Indian government launched Arogya Setu App, which uses Bluetooth and GPS to help people maintain a safe distance from people already infected by COVID-19 [9]. At the same time, it also helps a person to self-diagnose in case they have COVID-19 symptoms. A few law enforcement departments used drone technology and high-quality video surveillance cameras to detect and avoid vast gatherings of people so that apt and timely regulatory measures can be taken to stop them [10], [11]. This manual intervention by law enforcement helped to flatten the spread-curve in some situations. Still, it simultaneously brought various threats to public health and many health and exposure challenges to the frontline workers.

Human object detection from CCTV cameras is an already established area of work that depends upon various manual methods of identifying unusual activities inside the video footage from surveillance cameras; however, this domain has minimal capabilities [12]. Various recent advancements have led to intelligent systems to detect and capture human objects, their poses, motions, and activities. Even though perfect human object detection is an aspiring objective, because of different constraints such as clothing, lighting, complex backgrounds, low-resolution of the video cameras, varying human poses, and limited machine computer vision capabilities, wherein precedent knowledge on these challenges can improve the object detection performance [13].

Object detection problems have been effectively performed by recent development in this field using advanced techniques. Starting from Convolutional neural networks (CNN), region-based CNN [14] and faster region-based CNN [15] made use of region proposal mechanisms to produce the objectness score anterior to its classification, and later it generates the bounding boxes around the detected object for visualizing them and for various other statistical examinations [16]. All these traditional research works were a part of the last decade. These algorithms require huge amounts of training time; hence even though they are efficient and excellent in terms of accuracy, we look for alternative methods. These CNN based algorithms make use of classification as their backbone. On the other hand, a different approach was followed by YOLO, which makes use of regression techniques to separate the bounding boxes dimensionally and predict the objects' class probabilities [17]. This object detection module manifests powerful generalization capabilities of representing an entire image [18].

Based on the described research results, many similar research findings have been reported in the last few years. One of the promising research areas was People-Crowd counting, and it had various societal, industrial, and market applications. Chen et al. [19] focused on a technique of electronic advertising application that uses the concept of crowd counting. Eshel et al. [20] proposed performing counting of people and crowd detection by suggesting multiple height transformations for head top detection. These authors manage to solve the problems associated with occlusions inside the video feeds. In a similar kind of application, a machine vision-based person counting model was proposed by Chih-Wen et al. [21].

After the rise of neural networks and deep learning techniques, video analytics and video surveillance have become huge research and development areas. A lot of publicly available datasets are being consumed for various machine vision applications. Researchers at the University of Oxford have proposed a dataset named performance evaluation of tracking and surveillance (PETS) [22] for machine vision research comprising a large number of datasets for different kinds of tasks in the field of computer vision, KTH human motion dataset [23] consists of 6 categories of activities. In comparison, another Weizmann human action dataset [24] contains ten types of actions. We have used two datasets in our current work, one for human object detection and the other for correct/in-correct face mask identification. Common Objects In Context (COCO) dataset [25] and VictorLin000's face mask dataset [26] have been used respectively to train models for accurate person detection and correct/in-correct face mask classification.

Computer vision and Deep Learning have managed to turn simple network CCTV cameras into "smart" surveillance cameras that can be used to monitor whether people are following social distancing and mask guidelines correctly or not, along with their traditional use as security cameras. These systems require object detection and tracking algorithms to precisely monitor, track, and check for guidelines' compliance. Although many researchers have already worked in this field recently, the majority of the proposed solutions are limited to small scale applications working on very few CCTV cameras and consuming high CPU and GPU computing power. After understanding and exploring the related works, we realized that object detection applications could be applied to humans to detect them inside a video feed, and an approach of classification can be used to monitor people not wearing masks or wearing masks improperly. We can develop an end-to-end solution to help curb the spread of the COVID-19 virus across the globe.

III. Methodology

We propose an end-to-end solution that is based on: You Only Look Once (YOLO) v3 architecture, Simple Online and Realtime Tracking with a Deep Association Metric (DeepSORT) algorithm, and Open Source Computer Vision Library (OpenCV). It provides us with the following functionalities:

a. ability to monitor different social distancing and face mask compliance metrics

b. flexibility to deploy an efficient and cost-effective system in Python using open-source libraries supported on multiple architectures

c. scalability enabling the users to implement the solution at a large scale, i.e., cover a broader area using various live camera feeds simultaneously through multiprocessing and threading modules

Our solution also solves the inherent problem of precisely calculating the distances among people by mapping the 2-dimensional pixel distances (captured by the camera) to 3-dimensional actual distances based on the concepts of geometry and mathematical approximations.

A. Person Detection

Our objective is to accurately detect humans using existing object detection models, keeping in mind the various types of challenges such as low quality of CCTV footages, variations in clothes, postures present at far and close distances with/without occlusion and under different lighting conditions. To fulfill this objective, we have used the YOLO v3 model trained on the COCO dataset to filter out the "person" class. Joseph Redmon et al. introduced You look only once, also known as YOLO, in 2015 [27]. Later, some improvements were made, and YOLOv2 & YOLOv3 were introduced in 2016 [28] & 2018 [29], respectively. Darknet-53 is the backbone of YOLOv3. It is used as a feature extractor for training a deep neural network. It predicts an object's type and location by looking only once at the image and considers the object detection problem as a regression task instead of classification to assign class probabilities to the anchor boxes. A single convolutional network simultaneously predicts multiple bounding boxes and class probabilities. YOLOv3 uses three scale predictions, i.e., it has three output layers that provide projections based on three different scales.

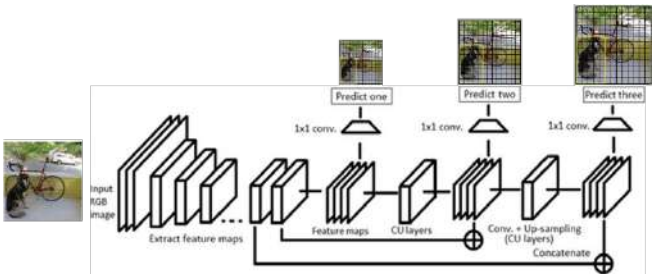


Fig. 1. Schematic representation of YOLOv3 architecture.

B. Person Tracking

As per the Centers for Disease Control and Prevention (CDC) guidelines [30], if two or more people stay within 6 feet, they violate social distancing compliance. Hence, post detecting humans using YOLOv3, the DeepSORT algorithm [31] is used to track all the people appearing in the CCTV footage.

Fig. 2. Output from person detection and person tracking modules.

The DeepSORT algorithm is based on a deep learning approach and is used to track objects in a video. In our solution, DeepSORT is being utilized to track the individuals present in the CCTV footage. It uses patterns learned via detected objects in the images and combines them with the temporal information for predicting associated trajectories of the objects of interest. It keeps track of each item under consideration by mapping unique identifiers for further statistical analysis. DeepSORT also handles the associated challenges with object tracking, such as occlusion, multiple viewpoints, non-stationary cameras, and annotating training data. Kalman filter and the Hungarian algorithm are used in DeepSORT for effectively tracking objects. The Kalman filter, used recursively, improves the association and predicts future positions based on the current position. On the other hand, the Hungarian algorithm is mainly used for association and id attribution. It identifies if an object in the current frame is the same as in the previous frame.

C. Social Distancing

The person detection and tracing results obtained using YOLOv3 and DeepSORT are further processed to check social distancing compliance. The bounding box coordinates of objects detected as "person" class and their respective probability scores are passed as an input to the DeepSORT tracking module, which keeps updating the location of these objects and tracks their movement while they stay inside the region of interest. Time (T) for each unique object detected as "person" class is measured to keep track of the duration they stay in contact with other objects of the "person" class while violating the minimum distance required for social distancing.

To check whether the minimum distance required for social distancing is maintained, the 3-dimensional distance between each pair of persons needs to be calculated while keeping track of their movement. 3-dimensional Euclidean Distance is used to measure the distance between a pair of objects, i.e., the distance between 2 persons in our case. To accurately calculate this 3-dimensional distance, we use the following approach:

a. Using the coordinates of the top-left edge, height, and width of a bounding box (obtained by YOLOv3), we calculate the x and y coordinates of the centroid of each bounding box.

b. As a person walks away from the CCTV camera in a real-world setting, the width of his/her bounding box (created by YOLOv3) decreases. In other words, the distance from the CCTV camera is inversely proportional to the width of the bounding box of the objects. As a result, the inverse of width is a proxy

for the distance of an object from the CCTV camera or the centroid's z coordinate. We can normalize this value range to be in coherence with x and y coordinates using a scale factor.

$$z = (\frac{1}{width}) * scale\ factor$$

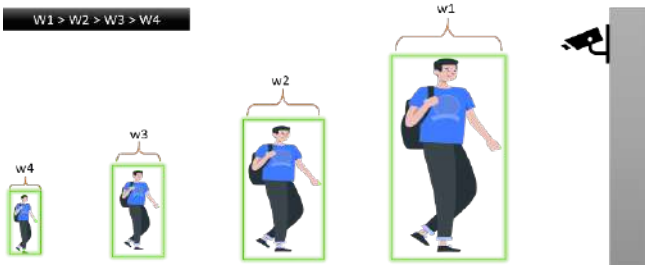


Fig. 3. Relation between width of bounding boxes and distance of objects from camera.

c. Using the coordinates (x, y, z) of the centroids, 3-dimensional Euclidean distance is calculated among all the persons detected in a single frame and stored in a distance matrix.

d. The distance matrix will contain distance values in pixel units, which are converted to real-world distances using the unitary method. For each pair of persons, the height of the bounding box of the person closest to the camera (h) is considered and is assumed to be equivalent to 5.25 feet (average height of human beings). Then the real-world distance between the two persons (D) can be calculated from the pixel distance (d) using:

$$D = (\frac{5.25}{h}) * d$$

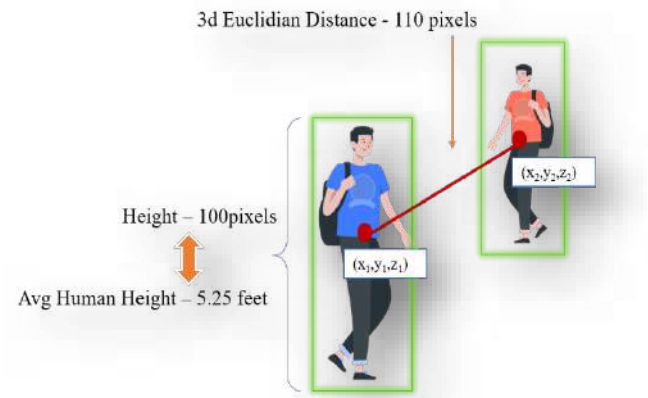


Fig. 4. Conversion of 2-D pixel distances to 3-D real-world distances.

Using the real-world 3-D distances between a pair of persons and the duration they stay in contact, we calculate the social distancing compliance metric for each pair of persons as follows:

Algorithm 1: To compute social distancing metric

```
If distance (D) < 6 feet And time (T) > 15 minutes, then
    social_distancing_compliant = False
Else
    social_distancing_compliant = True
```

D. Face Mask Detection

For face mask detection, we used darknet-53 based YOLOv3 model trained on the COCO dataset as a feature extractor. The head of this deep neural network is re-trained using transfer learning to classify and predict three custom classes:

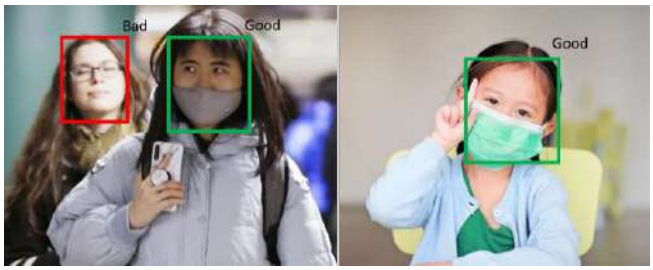


Fig. 5. Sample images from pre-annotated open-source face mask dataset.

a. "Mask" class - refers to a person wearing the mask correctly

b. "No Mask" class - refers to a person not wearing a mask at all

c. "Incorrect Mask" class - refers to a person partially wearing the mask (for example, below the nose)

A pre-annotated open-source dataset [26] containing 678 images of people with and without masks labeled as "good", "bad" and "none" was used to re-train the YOLOv3 model using transfer learning.

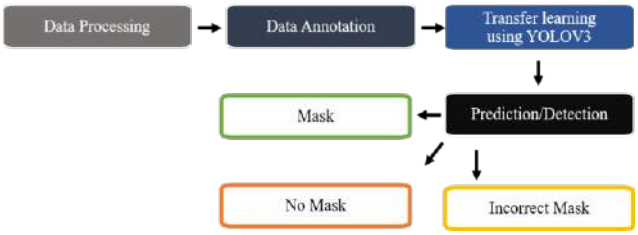


Fig. 6. Workflow for custom YOLOv3 based face mask detection

E. Scaling up our solution

Scalability is defined as the property of a system to be able to handle a growing amount of work. We have made our solution scalable hence it can be implemented industry-wide, i.e., from small retail shops (having 2-3 CCTV cameras) to huge manufacturing facilities (having 50+ CCTV cameras).

The individual social distancing and mask violation detection modules, as explained above, are packaged together using Python and OpenCV [32]. The OpenCV module is used to read, and process live video feeds from multiple CCTV cameras simultaneously with the help of Python's multithreading capabilities. The live CCTV camera feeds are captured using the OpenCV's VideoCapture class, which uses Real Time Streaming Protocol (RTSP) to read and decode the video frames.

In general, the CCTV cameras have a frame rate of 15-30 frames per second (FPS), and as we are working with multiple cameras, they can all have different frame rates. Also, processing each frame of all the cameras would make it practically impossible to run the solution on a personal computer system and require a server-grade system. Therefore, while all frames are continuously read, only a

single frame is processed every 2 seconds for a single camera. To read and process the video feeds from all these cameras simultaneously, multithreading functionality is used. The OpenCV's VideoCapture class has three methods to read and process frames from a camera device or a video file, namely:

a. read() - reads and decodes a frame of a video feed

b. grab() - reads a frame of a video feed

c. retrieve() - decodes a frame of a video feed

Reading a frame is not a compute-intensive task, while decoding a frame is highly compute-intensive. Therefore, the grab() method is executed faster as compared to the read() method, and the frame read using the grab() method can be decoded using the retrieve() method if needed. The frames from all the cameras are read simultaneously using the grab() method running on separate background threads (one thread for each camera).

This way, live video feeds from multiple cameras are read in parallel, which helps us overcome the IO bottleneck. Once all the video feeds start getting captured by the background threads, they are passed on to the social distancing and mask violation detection modules running on the main thread with the help of the 'queue' module. It module uses multiple consumer and producer queue data structures and enables the sharing of data between threads safely and efficiently. It follows the First In First Out (FIFO) rule, i.e., the data item inserted first is processed first. Each background thread uses the put() method to insert a frame in the queue, and it is fetched using the get() method by the main thread for processing and calculating mask and social distancing violations.

Apart from using all the above techniques to efficiently use the system's resources, the GPU version of the OpenCV library is used. Further, OpenCV's DNN module's readFromDarkNet(), blobFromImages(), and setInput() methods are used to prepare a batch of images for inferencing and prediction by social distancing and mask detection modules.

To summarize, we use multithreading and batch processing concepts along with OpenCV's DNN module with GPU support to efficiently process all the live CCTV camera feeds simultaneously close to real-time

IV. Results

The social distancing and mask violation system was tested on sample videos obtained from Twitter [33].

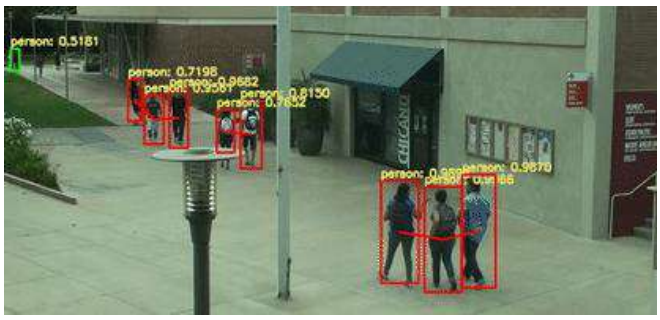
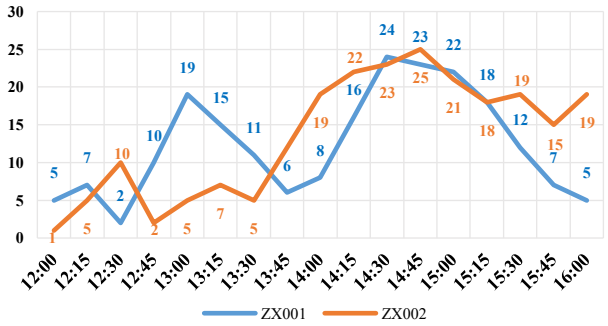


Fig. 7. Processed frame from social distancing module.

It was then implemented in a manufacturing facility having 50 live CCTV cameras. The system's configuration included a 9th Generation Intel Core i7 CPU, Nvidia RTX 2080Ti GPU, and 16GB of RAM. The solution was tested for two weeks for measuring the mask and social distancing compliance across the facility among the workers. 'Dash' module of Python was used to create a dynamic web-based dashboard to visualize the results.

Fig. 8. Processed frames from face mask detection module.



Graph 1. Number of violations within a span of 4 hours for 2 out of 50 CCTV cameras.

V. Conclusion

In our solution, we have developed a fully functional end to end system to monitor social distancing and mask compliance based on multiple CCTV cameras. We have used state of the art YOLOv3 models for accurate person and face mask detections while the DeepSORT algorithm enables tracking persons in live CCTV footage. Concepts of geometry and approximation were used to estimate the 3-D real-world distances among the objects in a frame. Python programming language and it's open-source libraries such as threading, queue, and OpenCV were used to scale up our end-to-end system, making it possible to process live footage from 50 CCTV cameras simultaneously and generate insights in almost real-time successfully.

VI. Future Work

The base version of YOLOv3 trained on the COCO dataset is used for object detection and filtering the humans. YOLOv3 is a compute-intensive architecture and is slower in comparison to Single Shot Detectors (SSD). But it can be re-trained just to detect the 'person' class instead of detecting all the COCO classes, which would make it more efficient and accurate. Quantization of weights in both the YOLOv3 models (person and mask detection models) can be done to further optimize the solution in terms of computing power required and the number of frames to be processed. Different quantization techniques such as weight quantization, integer quantization, full integer quantization, and float-16 quantization can be implemented to reach a sweet spot between efficiency and accuracy.

VII. References

[1] B. News, "China coronavirus: Lockdown measures rise across Hubei province," https://www.bbc.co.uk/news/world-asia-china51217455, 2020, [Online; accessed 23rd January 2020].

[2] N. H. C. of the Peoples Republic of China, "Daily briefing on novel coronavirus cases in China," http://en.nhc.gov.cn/2020-03/20/c 78006.htm, 2020, [Online; accessed 20th March 2020].

[3] K. Prem, Y. Liu, T. W. Russell, A. J. Kucharski, R. M. Eggo, N. Davies, S. Flasche, S. Clifford, C. A. Pearson, J. D. Munday et al., "The effect of control strategies to reduce social mixing on outcomes of the covid- 19 epidemic in wuhan, china: a modelling study," The Lancet Public Health, 2020.

[4] K. E. Ainslie, C. E. Walters, H. Fu, S. Bhatia, H. Wang, X. Xi, M. Baguelin, S. Bhatt, A. Boonyasiri, O. Boyd et al., "Evidence of initial success for china exiting covid-19 social distancing policy after achieving containment," Wellcome Open Research, vol. 5, no. 81, p. 81, 2020.

[5] C. Adolph, K. Amano, B. Bang-Jensen, N. Fullman, and J. Wilkerson, "Pandemic politics: Timing state-level social distancing responses to covid-19," medRxiv, 2020.

[6] S. K. Sonbhadra, S. Agarwal, and P. Nagabhushan, "Target specific mining of covid-19 scholarly articles using one-class approach," 2020.

[7] N. S. Punn and S. Agarwal, "Automated diagnosis of covid-19 with limited posteroanterior chest x-ray images using fine-tuned deep neural networks," 2020.

[8] N. S. Punn, S. K. Sonbhadra, and S. Agarwal, "Covid-19 epidemic analysis using machine learning and deep learning algorithms," medRxiv, 2020. [Online]. Available: https://www.medrxiv.org/content/early/2020/04/11/2020.04.08.20057679

[9] O. website of Indian Government, "Distribution of the novel coronavirus-infected pneumoni Aarogya Setu Mobile App," https://www.mygov.in/aarogya-setu-app/, 2020.

[10] M. Robakowska, A. Tyranska-Fobke, J. Nowak, D. Slezak, P. Zuratynski, P. Robakowski, K. Nadolny, and J. R. Ładny, "The use of drones during mass events," Disaster and Emergency Medicine Journal, vol. 2, no. 3, pp. 129–134, 2017.

[11] J. Harvey, Adam. LaPlace. (2019) Megapixels.cc: Origins, ethics, and privacy implications of publicly available face recognition image datasets. [Online]. Available: https://megapixels.cc/

[12] N. Sulman, T. Sanocki, D. Goldgof, and R. Kasturi, "How effective is human video surveillance performance?" in 2008 19th International Conference on Pattern Recognition. IEEE, 2008, pp. 1–3.

[13] X. Wang, "Intelligent multi-camera video surveillance: A review," Pattern recognition letters, vol. 34, no. 1, pp. 3–19, 2013.

[14] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," IEEE transactions on neural networks and learning systems, vol. 30, no. 11, pp. 3212–3232, 2019.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, 2012, pp. 1097–1105.

[16] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in Advances in neural information processing systems, 2015, pp. 91–99.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788.

[18] M. Putra, Z. Yussof, K. Lim, and S. Salim, "Convolutional neural network for person and car detection using yolo framework," Journal of Telecommunication, Electronic and Computer Engineering (JTEC), vol. 10, no. 1-7, pp. 67–71, 2018.

[19] D.-Y. Chen, C.-W. Su, Y.-C. Zeng, S.-W. Sun, W.-R. Lai, and H.-Y. M. Liao, "An online people counting system for electronic advertising machines," in 2009 IEEE International Conference on Multimedia and Expo. IEEE, 2009, pp. 1262–1265.

[20] R. Eshel and Y. Moses, "Homography based multiple camera detection and tracking of people in a dense crowd," in 2008 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2008, pp. 1–8.

[21] C.-W. Su, H.-Y. M. Liao, and H.-R. Tyan, "A vision-based people counting approach based on the symmetry measure," in 2009 IEEE International Symposium on Circuits and Systems. IEEE, 2009, pp. 2617–2620.

[22] O. Parkhi, A. Vedaldi, A. Zisserman, and C. Jawahar, "The oxford-iiit pet dataset," 2012.

[23] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: a local svm approach," in Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004., vol. 3. IEEE, 2004, pp. 32–36.

[24] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," in Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, vol. 2. IEEE, 2005, pp. 1395–1402.

[25] Common Objects In Context. (2020). COCO. Retrieved from https://cocodataset.org/#home

[26] VictorLin. (2020). Face Mask Dataset. Retrieved from https://github.com/VictorLin000/YOLOv3_mask_detect

[27] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788.

[28] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 7263–7271.

[29] J. R. A. Farhadi and J. Redmon, "Yolov3: An incremental improvement" Retrieved September, vol. 17, p. 2018, 2018.

[30] Centers for Disease Control and Prevention. (2020). CDC. Retrieved from https://www.cdc.gov/coronavirus/2019-ncov/prevent-getting-sick/prevention.html

[31] N. Wojke, A. Bewley, and D. Paulus, "Simple online and real-time tracking with a deep association metric," in 2017 IEEE international conference on image processing (ICIP). IEEE, 2017, pp. 3645–3649.

[32] Open Source Computer Vision Library. (2020). OpenCV 4.3.0 Retrieved from https://docs.opencv.org/4.3.0/

[33] Test Video for Mask Detection. (2020). Retrieved from https://twitter.com/BAG_OFSP_UFSP/status/1321126674692530176

Telecom Churn and Valued Customer Retention

Ishi Khamesra
Hyderabad, India

Srinivasarao Valluru
Hyderabad, India

***Abstract*—Enterprise Telecom services is an immensely competitive market the world over, with tailor-made tariff plans complimented with rebates makes reaching targeted profits to be an astronomical challenge. Consumers are the lifeline that firms have a huge challenge in retaining today. Customer Churn the standard industry buzz word could turn out to be a nightmare for any established service provider, so whilst there is always a renewed focus in bringing new customers that call for intense investments and resources, maintaining existing ones is relatively inexpensive.**

The advancements in predictive modelling are one step in the right direction that alleviates churn for telecom service providers. We propose a progressive analytical approach to predict customer churn one month ahead and the churn pool derived by means of consumer segmentation to identify high valued among potential churn customers, aided by advanced analytics that takes into account the churn severity level, churn priority, etc., to provide a persona-based treatment plan for consumer retention. This will ultimately lead to increased revenue with an ever-growing customer base.

Keywords—Random Forest (RF), Support Vector Machine (SVM), Extreme Gradient Boosting (XGB), Generalized Linear Model (GLM), Logistic Regression (LR), Neural Networks Multi-Layer Perceptron (MLP), Feature Selection, Recursive Feature Elimination (RFE), Decision Tree (DT), Elbow Rule, Density-based spatial clustering of applications with noise (DBSCAN), Mahalanobis distance, Average Silhouette width, K-Prototypes, ROC Curve, AUC, Accuracy, Specificity, Sensitivity, F1-Score, Lift and Gain charts.

I. Introduction

Telecom domain has witnessed tremendous growth in recent years with regard to technology and customer base, coupled with strong competition among the network providers. Consumer retention and acquisition is a central pillar of any successful firm. In order to maintain a robust customer base, network providers are giving steep discounts on tariffs which are leading to poor revenue growth. One of the major concerns for the network providers is customer churn. Prediction of customers who are at high risk of discontinuing the service with the provider is of paramount importance as remedies to acquire new customers are more expensive compared to withhold existing ones. Churn prediction in advance will provide enough time for service providers to take corrective measures to retain the customer. All customers who are at the risk of leaving the service provider may not be prime members. Service providers can save significant revenue by providing additional retention benefits to only high valued customers. Earlier studies[1-4] were focused on predicting customer churn to the best of our knowledge, none attempted to identify high valued customers among potential churn pool for the treatment process. In this study, we provide an analytical solution to predict customers who are likely to churn using machine learning classification techniques and high valued customers among likely churn customer pool using clustering techniques to take corrective measures for retention.

Information related to customer demographics, network usage, billing details, and call details to customer care has been captured. Machine learning classification techniques Support Vector Machines[5] (SVM), Extreme Gradient Boosting[11] (XGB), Generalized Linear Model (GLM) Logistic Regression[5] (LR), Random Forest[7] (RF) and Neural Network Multi-Layer Perception[12] (MLP) has been built to predict customer churn. To form homogenous groups of risky customers who behave similarly, applied clustering technique K-prototypes[10] on historical churn customers. Cluster properties have been studied through rule-based technique Decision Tree[6] (DT) and labelled each cluster. A cluster with prime members was identified and built predictive model SVM[5] to predict the valued customer. Fig.1 represents the overall flow diagram of analytical solution for churn prediction and valued customer identification.

II. Methodology

A. Data

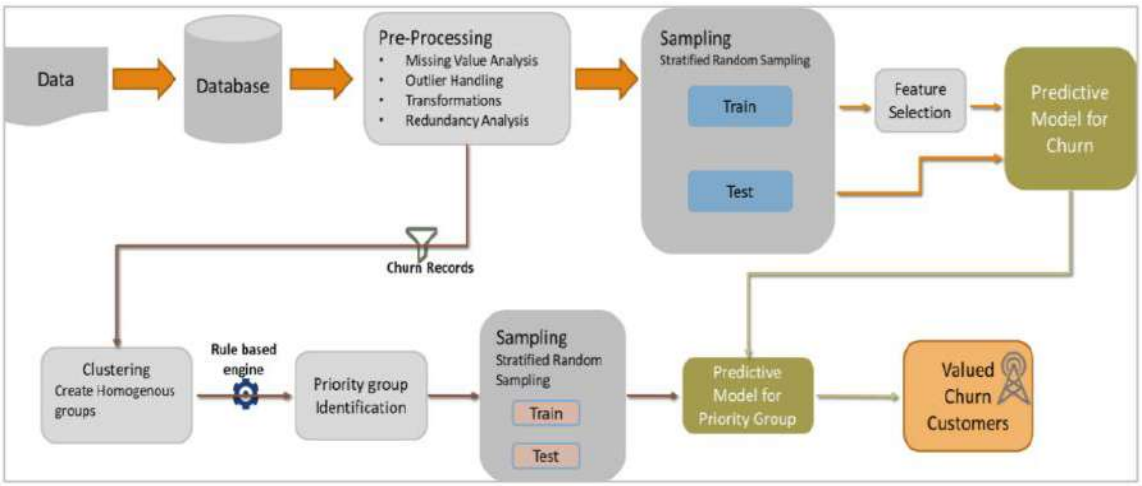


Fig. 1. Churn prediction and valued customer identification

Churn dataset from publicly available source[14] is considered for analysis. A total of 100 features' information related to demography, billing details, usage details ,and calls to customer care has been captured over a period of three

months. Churn is measured in the fifth month to provide one month ahead of predictions for business. A total of 0.1 million customer records, out of which nearly 0.05 million customers were churn. Churn records were oversampled in the actual data to prepare balanced data of churn and non-churn customers for predictive modelling. 'Churn' is the target variable that takes binary values '1' (churn) and '0' (non-churn).

Pre-Processing techniques like Redundancy analysis, Transformations to handle skewness, Binning of categorical levels, Missing value analysis and Data imputations are applied. Identification of outliers through univariate analysis may lead to incorrect results, so applied Density-based spatial clustering of applications with noise (DBSCAN) [6] and Mahalanobis distance[6] to identify outliers. DBSCAN[6] is one of the best algorithms to identify outliers in multivariate scenario. Missing values are imputed through Classification and Regression trees[6] (CART) technique by taking all other variables into consideration.

B. *Churn Prediction*

1) Sampling: Stratified random sampling[8] without replacement is used in the ratio of 70:30 to segregate data into 'Train' and 'Test' respectively. Stratification is done based on variables 'Churn' and 'Area' to provide an equal representation of all categorical levels in model building.

2) Key Feature Identificaiton: All variables might not be important to predict customer churn. Applied feature selection techniques Recursive feature Elimination[9] (RFE) with Random Forest[7] estimator on 'Train' data to identify important variables that are contributing to customer churn. Some of the top 20 key variables identified are Area, Marital status, Equipment days, Revenue, use of off-peak voice calls, total months in service, number of dropped calls, credit class code, and calls to customer care.

3) Analytical Modelling: Key variables identified through feature selection are used to build predictive model to predict customers likely to churn. Using machine learning techniques Support Vector Machines[5] (SVM), Extreme Gradient Boosting[11] (XGB), Generalized Linear Model (GLM), Logistic Regression[5] (LR), Random Forest[7] (RF) and Neural Network Multi-Layer Perceptron[12] (MLP) built predictive models to estimate the probability of customer churn. Compared the performance of models using ROC Curve[13], AUC[13], Accuracy, Specificity, Sensitivity, F1-Score, Lift[13] and Gain charts. For each model respective hyper parameters are tuned to provide optimal performance. Extreme Gradient Boosting[11] (XGB) and Neural Network model Multi-Layer Perceptron[12] (MLP) provided best performance compared to other techniques. Accuracy and AUC[13] values for both XGB[11] and MLP[12] are more than 62% and 0.67 respectively. The accuracy of predictive models is not more than 63% because most of the variables exhibiting same pattern related to churn and non-churn in the data.

4) Analytical Results: Performance of the predictive models has been evaluated on 'Test' data. XGB and MLP provided comparatively better results than other models. Fig.2 shows the performance metrics of customer churn models.

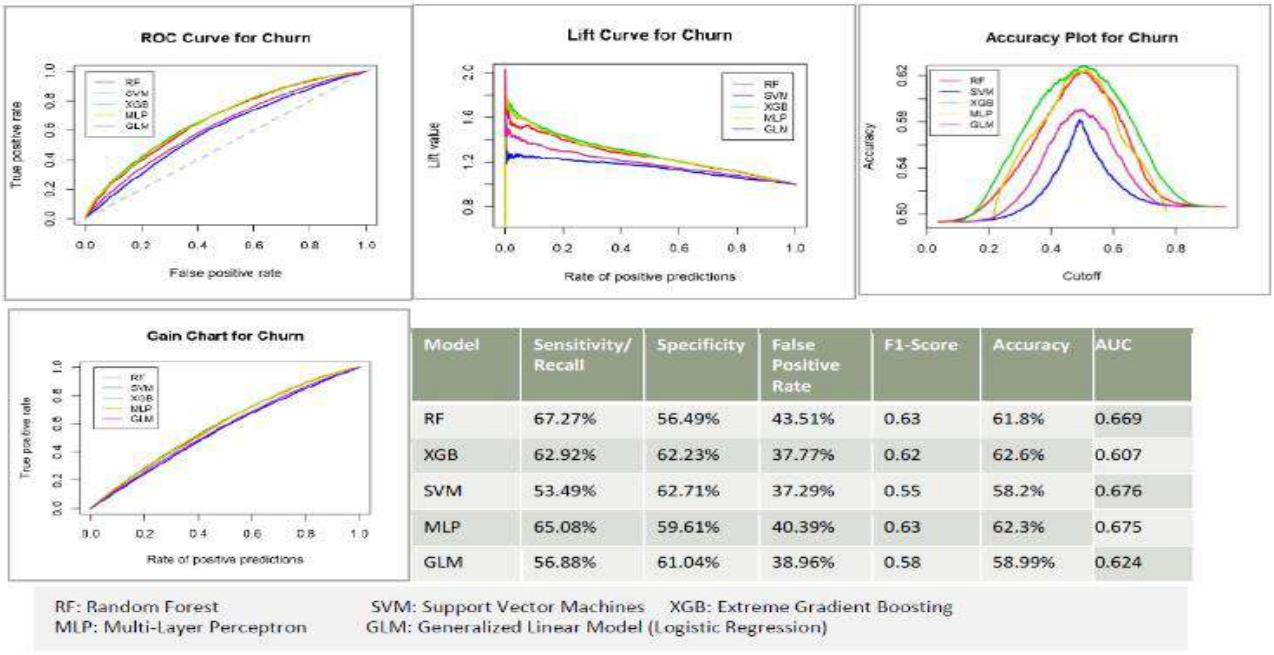
Based on the predicted churn probability, customers are divided into four groups named as 'No risk', 'Low risk', 'Mediumrisk' and 'High risk'.

C. *Valued Customer Identificaiton*

1) Sampling: To study patterns among churn customers, considered data of churn records only in the historical data. Stratified random sampling[8] without replacement is applied on churn records in the ratio of 70:30 to divide data into 'P-Train' and 'P-Test'.

Stratification is done based on the variable 'area' to provide an equal representation of all categorical levels in 'P-Train' data.

2) Clustering: Clustering technique K-Prototypes[10] is applied to 'P-Train' to study patterns among churn customers. K-Prototypes[10] is one of the best algorithms to



Model	Sensitivity/ Recall	Specificity	False Positive Rate	F1-Score	Accuracy	AUC
RF	67.27%	56.49%	43.51%	0.63	61.8%	0.669
XGB	62.92%	62.23%	37.77%	0.62	62.6%	0.607
SVM	53.49%	62.71%	37.29%	0.55	58.2%	0.676
MLP	65.08%	59.61%	40.39%	0.63	62.3%	0.675
GLM	56.88%	61.04%	38.96%	0.58	58.99%	0.624

RF: Random Forest SVM: Support Vector Machines XGB: Extreme Gradient Boosting
MLP: Multi-Layer Perceptron GLM: Generalized Linear Model (Logistic Regression)

Fig. 2. Performance metrics of customer churn prediction models

identify homogeneous groups when both categorical and numerical variables are present. Based on Elbow rule technique and average Silhouette width[6], identified optimal number of clusters (K) as four. To study properties of formed clusters, applied 'Decision Tree[6]', a rule-based technique by considering cluster number as target variable. Study of cluster properties will help in identifying the valued customers' group. Rules provided by 'Decision Tree' are extracted and labelled each cluster as P1, P2, P3 and P4 with the help of domain expert. P1 represents the priority cluster of most valued customers and P4 represents the least priority cluster.

3) Key Feature Identification: All variables might not be important for building classifier to predict the priority (P1, P2, P3 & P4) of each potential churn customer. Applied feature selection technique Recursive Feature Elimination [9] (RFE) and identified top 15 features which are important for classifying customer priority. These features include revenue, network usage, calls to customer care, area, dropped calls, blocked calls, etc.

4) Classifier Building: Built classifier using features selected through feature selection to categorize potential churn customer as P1, P2, P3 and P4. Where P1 represents the high valued customer and P4 represents least valued customer. SVM[5] and RF[7] are used to built classifiers. Model parameters are tuned to provide optimal performance. Performance of the classifiers has been evaluated on 'P-Test' data. SVM based classifier provided comparatively better classification results than RF based classifier.

5) Analytical Results: Model performance has been evaluated using metrics like Sensitivity, Specificity, Accuracy and F1 Score. Support Vector Machine5 (SVM) based classifier provided accuracy of 94.2% compared to Random Forest[7] (RF) 92.8%.

6) Treatment Plan Recommendation: Service providers have to understand and measure customer portfolio of each of the clusters and then provide a profitable customer treatment package for each of the clusters separately. These plans would be based on understanding the reason for discontinuing the service in each of the segments. One of the plan suggested for prime members would be to deliver great customer service and generate loyalty programs based on their history with the firm.

III. Conclusion

In telecom industry the real business value lies in fostering stable relationships with customers and therefore it is necessary to predict the customers who are likely to churn and to identify the priority of each potential churn customer, we presented an analytical solution using advanced machine learning techniques. Identification of risk level (Normal, Low, Medium, and High) and the priority level (P1, P2, P3, and P4) of each potential churn customer will help network providers for persona-based intervention to retain customers. Providing specific benefits to certain groups as a retention measure through persona-based intervention will lead to healthy customer base and increase in revenue. In this study, we also compared the performance of various machine learning models using metrics like AUC[13], Accuracy, F1-Score, Lift[13] and Gain charts. Among all predictive models, XGB[11] and Neural Networks MLP[12] provided the best results to predict customer churn and SVM[5] based classifier performed well for predicting priority of customers.

References

[1] Prashanth, R., Deepak, K., & Meher, A.K. (2017): "High Accuracy Predictive Modelling for Customer Churn Prediction in Telecom Industry", MLDM.

[2] Keramati, A., Jafari-Marandi, R., Aliannejadi, M., Ahmadian, I., Mozaffari, M. and Abbasi, U. (2014): "Improved churn prediction in telecommunication industry using data mining techniques", Applied Soft Computing, Vol.24.

[3] Brandusoiu, I., Toderean, G., Beleiu, H. (2016): "Methods for churn prediction in the pre-paid mobile telecommunications industry", International Conference on Communications (COMM), 97-100.

[4] Castanedo, F., Valverde, G., Zaratiegui, J., Vazquez, A. (2014): "Using Deep Learning to Predict Customer Churn in a Mobile Telecommunication Network".

[5] Hastie, T., Tibshirani, R. and Friedman J (2009): "The Elements of statistical learning: Data Mining, Inference, and Prediction", 2nd Edition, Springer.

[6] Mohammed J Zaki and Meira, W (2014): "Data mining and Analysis: Fundamental Concepts and Algorithms", Cambridge University Press.

[7] Breiman, Leo, (2001): "Random Forests", Machine Learning 45(1):5-32.

[8] Cochran, W (1977): " Sampling Techniques", 3rd Edition, Wiley.

[9] Guyon, Isabelle & Weston, Jason & Barnhill, Stephen & Vapnik, Vladimir. (2002): "Gene Selection for Cancer Classification Using Support Vector Machines. Machine Learning". 46. 389-422. 10.1023/A:1012487302797.

[10] Huang, Z (1997): "Clustering Large Data Sets with Mixed Numeric and Categorical Values".

[11] Chen, Tianqi & Guestrin, Carlos (2016): " XGBoost: A scalable tree boosting system". 785-794. 10.1145/2939672.2939785.

[12] Haykin, Simon. (1998): "Neural Networks: A Comprehensive Foundation", 2nd Edition, Prentice Hall.

[13] Vuk, M., & Curk, T. (2006): "ROC Curve, Lift Chart and Calibration Plot".

[14] Data source: https://www.kaggle.com/abhinav89/telecom-customer

About ADASCI

The Association of Data Scientists is the premier global professional body of data science & machine learning professionals.

ADaSI serves the scientific and professional needs of data science Professionals including educators, scientists, students, managers, analysts, and consultants. It serves as a focal point for data science, permitting them to communicate with each other and reach out their professional goal, as well as the varied clientele of the profession's research and practice.

It provides services such as publishing peer reviewed scholarly journals that describe the latest data science methods and applications, organizing national and international conferences for academics and professional, providing analytics certification and continuing education to assist members and others in furthering their career

PUBLISHED BY:

ADaSci | THE ASSOCIATION OF DATA SCIENTISTS

ONLINE CONTENTS AVAILABLE: EVERY QUARTER ON **www.adasci.org/lattice**

ISSN 2582-8312

Bhasker Gupta
Association of Data Scientists,
#189, 1st Floor, 17th Main Road, Near HSR Club, Sector 3, HSR Layout, Bengaluru, Karnataka-560102