Team 4

INDEX:

1. DEFINITION

1.1 Project Overview

The technologies used for telecommunications have changed greatly over the last 50 years. Empowered by research into semiconductors and digital electronics in the telecommunications industry, analog representations of voice, images, and video have been supplanted by digital representations. Perhaps the most fundamental change, both in terms of technology and its implications for industry structure, has occurred in the architecture of telecommunications networks. But, is this enough?

Even today, i.e in the 21st century as well one of the major consumer issues is increased network congestion and the consequent Quality of Service (QoS). For sure, network congestion is certainly one of the most important issues that the telecom industry faces.

Cellular service providers, further, have not given due importance to this issue as the basis of competition in this industry in India is pricing and not Quality of Service. But, as now, the Indian Telecom industry is fast transitioning from a growth phase to a maturity phase of the industry life cycle and the competition will get tougher for the players in the years to come. So reducing the network congestion is surely an important factor in the mind of every player in the telecom industry in order to expand their business and get a stronghold on people's mind. It has been shown that congestion, even if for smaller durations, has a negative impact on customer loyalty, especially in price-sensitive markets. To solve this problem effectively, it becomes imperative for firms to be able to predict congestion in advance and take proactive actions thereafter.

In this project, we have used a subset of an original dataset, which also has randomized or masked values. The training dataset that we have used only has sample data for December 2018 transactions.

1.2 Problem Statement

In this project, as mentioned above we aim to determine the type of network congestion occurring in the network in the context of the telecommunication industry. We are required to train machine learning models that use cell tower statistics such as usage, customer count, etc, to predict the type of congestion that might occur in the network. Some fields in the dataset are anonymized to avoid data leakage; while the usage data has also been anonymously scaled/randomized by a single/constant factor. The dataset provided has various features pertaining to both tower level activity and user-level activity.

Data Collection Methodology overview:

Incidents Table	**ESR Records**
1. Tower level activity	1. User level activity

These two types of records are aggregated over 5 mins period and joined internally.

A data dictionary is provided wherein we are enlightened with data for cell towers with all the fields in training dataset, further explanation of which is given in the data table.
Evaluation of the output is based on the Matthews correlation coefficient (MCC).
The data and metrics description is as shown below.

1.3 Data Description

In this project, we have a training dataset (train_upd.csv). This is our main dataset, it consists of 78560 rows and 39 columns. Each cell tower is represented with a unique cell tower id. Then we have "4G_rat" column which has categorical values 0 and 1 (0 – if tower supports 3G indicator and 1 – if supports 4G indicator). Next 4 columns (Par_year, par_month, par_day, par_hour, par_min) define the moment under consideration. The column "par_min" is recorded in buckets of 5 minutes where the values mean the upper limit of the interval. The "ran_vendor" column has to be converted into integers to apply the model on it. So we have performed one hot encoding on it. We also have other features like "beam_direction", this column consists of the angles of the sector which the cell tower covers. Then we have the "tilt" which is the vertical tilt angle i.e the angle the tower makes with the normal.

The following fields have been provided in the training dataset:

1. cell_name: Cell tower number/name – Masked name for cell towers
2. 4G_rat: Tower supports 3G/4G indicator
3. Par_year – Year under consideration (2018)
4. par_month – Month under consideration (December)
5. par_day – Day under consideration
6. par_hour – Hour under consideration
7. par_min – Minute bucket under consideration
   a. Buckets of 5 min interval. Eg: the value of 15 implies statistics are compiled/aggregated over a time period from 10-15 mins
8. subscriber_count: Count of total subscribers for the cell in the specified time period
9. Usage data: Data usage by activity type; includes both upload and download bytes
   Refer Appendix to see these types
10. beam_direction: Tower beam direction
11. cell_range: Cell tower range
12. tilt: Cell tower tilt
13. ran_vendor: Service Vendor
14. Congestion_Type: Type of congestion observed (Target Variable)

The Target variable "Congestion_Type" consists of 4 unique values ('NC', '4G_BACKHAUL_CONGESTION', '4G_RAN_CONGESTION', '3G_BACHAUL_CONGESTION) defining the congestion type occurring in the reported moment for the cell tower.

RAN CONGESTION – Radio Access Network (RAN) resides between devices such as a mobile phone, a computer, or any remotely controlled machine and provides a connection with its network (CN). RAN congestion is the wireless congestion between the cell tower and the network users.

BACKHAUL CONGESTION – The backhaul portion of the network comprises of the intermediate links between the core network, or backbone network, and the small subnetworks at the *edge* of the network. The congestion occurred at backhaul is the wired congestion.

NC - NC means No Congestion, which states that there is no congestion at the moment of recording the data for the specified cell tower.

The target variable is in strings, so we have label encoded our target variable "Congestion_Type" to :

3 – NC (No Congestion)

2 – 4G_RAN_CONGESTION

1 – 4G_BACKHAUL_CONGESTION

0 – 3G_BACKHAUL_CONGESTION

1.4 Evaluation Metrics

In this problem statement, evaluation of outputs is based on the Matthews correlation coefficient, also abbreviated as MCC. In general, the Matthews correlation coefficient is used in machine learning as a measure of the quality of binary (two-class) classifications. It takes into account true and false positives and negatives and is generally regarded as a balanced measure which can be used even if the classes are of very different sizes. The MCC is, in essence, a correlation coefficient between the observed and predicted binary classifications; it returns a value between −1 and +1. A coefficient of +1 represents a perfect prediction, 0 no better than random prediction and −1 indicates total disagreement between prediction and observation. The statistic is also known as the phi coefficient.
The MCC can be calculated directly from the confusion matrix using the formula:

$$\mathrm{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

In this equation, TP is the number of true positives, TN the number of true negatives, FP the number of false positives and FN the number of false negatives. If any of the four sums in the denominator is zero, the denominator can be arbitrarily set to one; this results in a Matthews correlation coefficient of zero, which can be shown to be the correct limiting value.

Example code:

```
>>> from sklearn.metrics import matthews_corrcoef
>>> y_true = [+1, +1, +1, -1]
>>> y_pred = [+1, -1, +1, +1]
>>> matthews_corrcoef(y_true, y_pred)
-0.33…
```

This is equal to the formula given above. As a correlation coefficient, the Matthews correlation coefficient is the geometric mean of the regression coefficients of the problem and it's dual. The component regression coefficients of the Matthews correlation coefficient are Markedness (Δp) and Youden's J statistic (Informedness or Δp'). Markedness and Informedness correspond to different directions of information flow and generalize Youden's J statistic, the delta p statistics and (as their geometric mean) the Matthews Correlation Coefficient to more than two classes.

As in our Problem statement, we are doing multiclass classification, therefore we use the generalized form of MCC score for multiclass classification. The Matthews correlation coefficient has been generalized to the multiclass case. This generalization was called the Rk statistic (for K different classes) by the author and defined in terms of a K X K confusion matrix C.

$$\mathrm{MCC} = \frac{\sum_k \sum_l \sum_m C_{kk} C_{lm} - C_{kl} C_{mk}}{\sqrt{\sum_k (\sum_l C_{kl})(\sum_{k'|k' \neq k} \sum_{l'} C_{k'l'})} \sqrt{\sum_k (\sum_l C_{lk})(\sum_{k'|k' \neq k} \sum_{l'} C_{l'k'})}}$$

When there are more than two labels the MCC will no longer range between -1 and +1. Instead, the minimum value will be between -1 and 0 depending on the true distribution. The maximum value is always +1.

2. ANALYSIS

2.1 Data Explorations

1. Some features like “par_year”, ”par_month” have the same values 2018 and 12 respectively. So their variance is zero and they do not contribute anything to our prediction model. So we removed these two features.

2. The column “cell_name” contains unique cell id of each tower and according to the problem statement, we are not allowed to use this as a predictor variable. So this column was also removed.

3. “subscriber_count” is the feature defining the subscriber’s count at the moment of recording the data. We found that as the count increases, the congestions start to occur. That is we found that for no congestion, the average subscriber count is 197.6368, for 3G backhaul congestion it is 360.6324, for 4G backhaul, it is 658.1238, for 4G ran congestion it is 1078.6633. So as the subscriber count increases, our congestion type changes from ‘no congestion’ to ‘4G ran congestion’.

4. We can find a similar trend in total bytes, as the total bytes consumed through the cell tower increases, the congestion type changes from ‘no congestion’ (average total bytes = 11503.56) to ‘3G backhaul congestion’ (average total bytes = 216242.29) to ‘4G backhaul congestion’ (average total bytes = 40036.81) to ‘4G ran congestion (average total bytes = 67721.85) .

5. Analyzing `Tilt`, `Cell Range` and `Ran Vendor` we get to know that the spread of data in all these variables is almost equal with negligible variation and thus removing any possibility of data bias.

Tilt	Count
2	19559
3	19665
4	19660
5	19676

Cell Range	Count
2	13030
3	13052
4	13387
5	12924
6	13141
7	13026

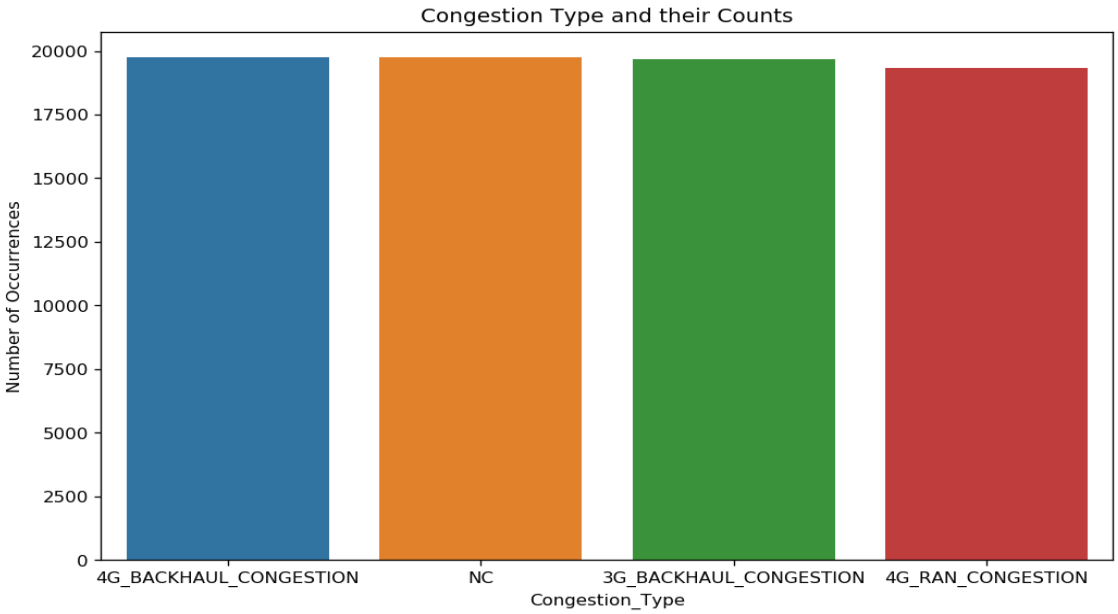
Ran Vendor	Count
Nokia	26533
Ericsson	26216
Huawei	25811

2.2 Feature Engineering

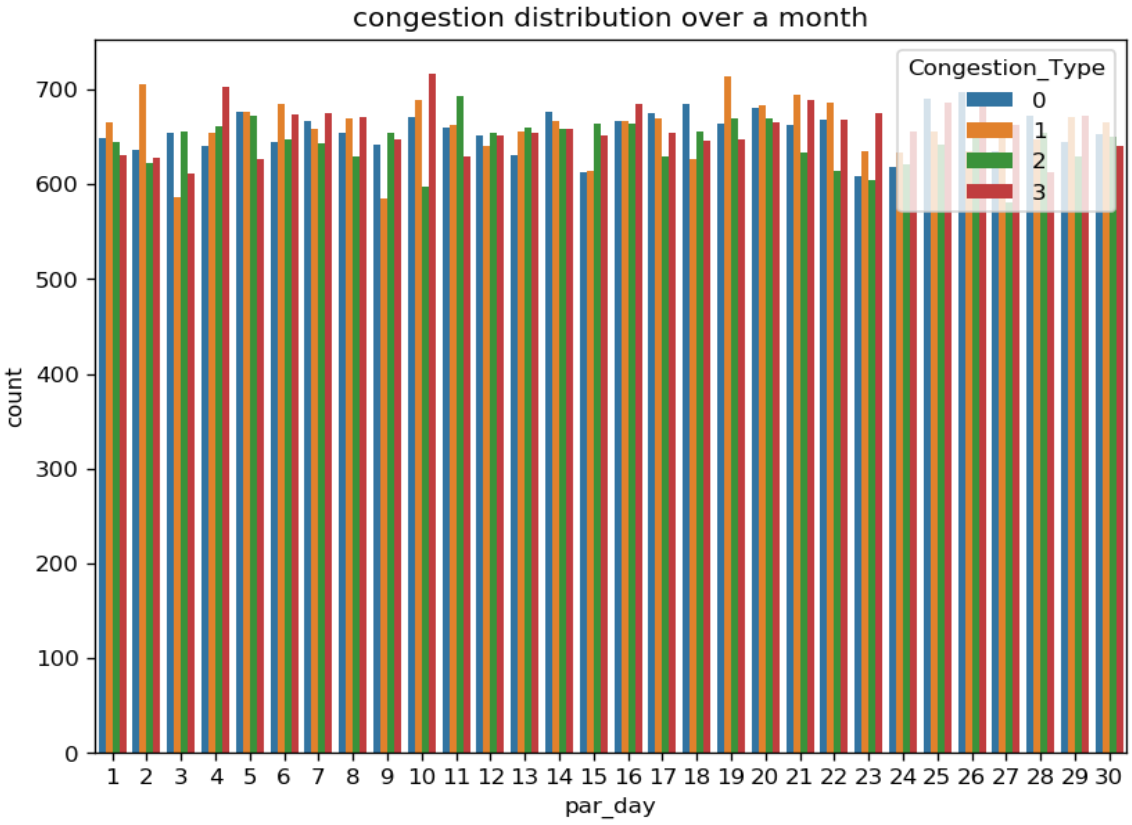
1. Hour buckets:
   We divided the hours into time zones of 5 intervals (1:00 am -7:00 am, 7:00 am - 11:00 am, 11:00 am - 4:00 pm, 4:00 pm - 8:00 pm, 8:00 pm - 1:00 am) and label encoded each time-zone.
2. Total bytes:
   We took the sum of all bytes consumed through the tower and created a new feature for sum_total_bytes consumed.
3. Day of the week feature:
   We divided the 'par_days' feature by 7 and created a new feature of days_of_week, that is if it is Monday then we gave a value 1, 2 for Tuesday,3 for Wednesday and so on. We expected more congestions on Sundays or holidays.

4. Log of each bytes data:
   Due to high variance in data of Bytes consumption we took log of each feature to reduce variance and made its data distribution normal.
5. Square of each feature:
   We created new features by squaring values to capture any hidden pattern.

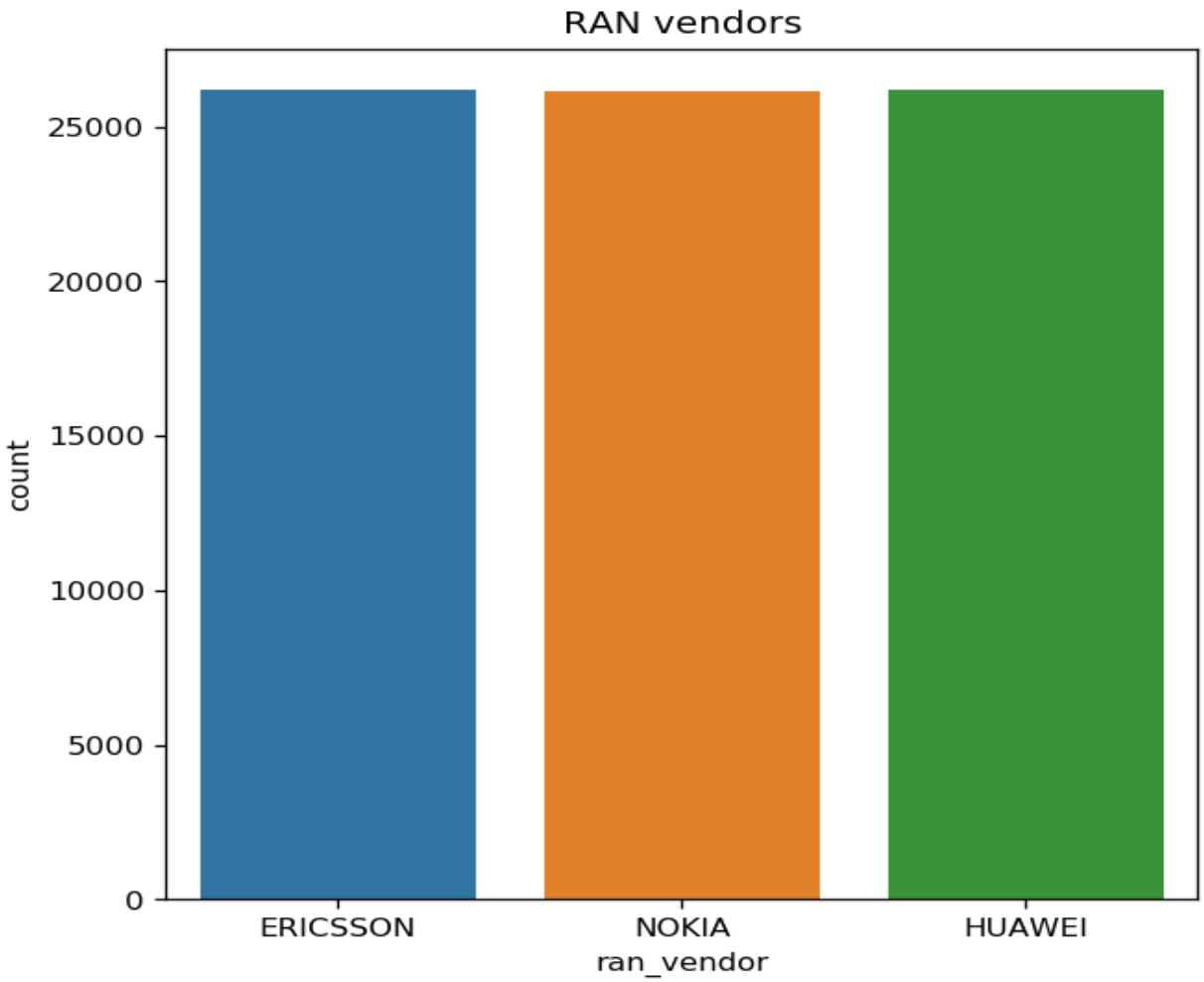
2.3 Exploratory Analysis



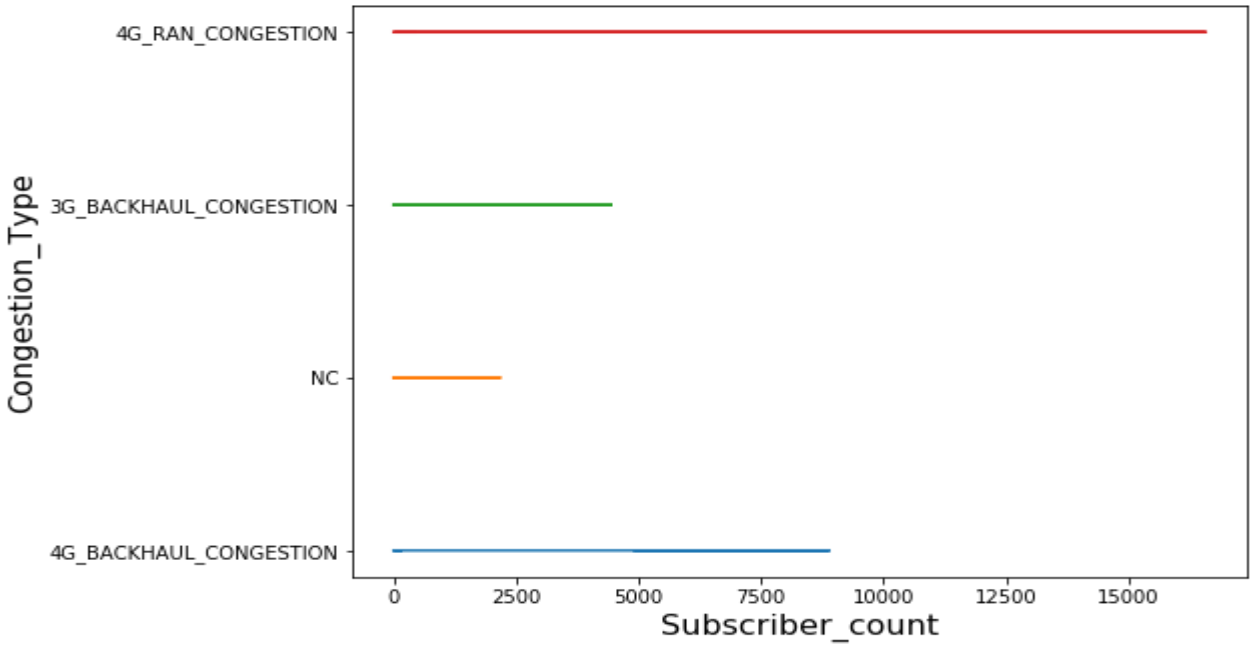
From this basic plot, we can conclude that class distribution between the 4 congestion types is almost similar in the training set.



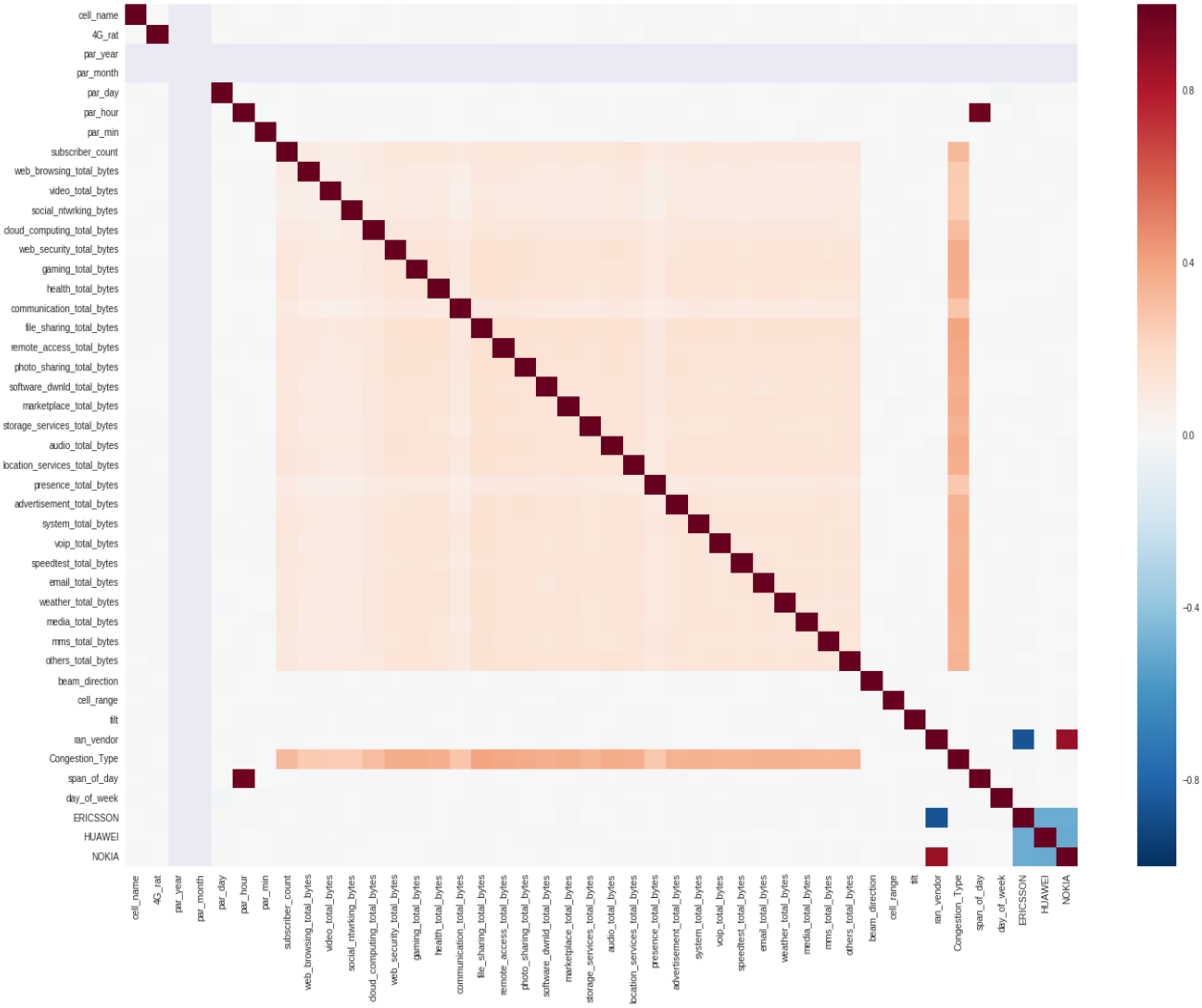
This plot represents the count of each type of congestion during a day. From this, we can conclude that a specific pattern is not generated but the count of each congestion type is randomly distributed throughout the month. Each type being maximum some day and minimum on some other day.



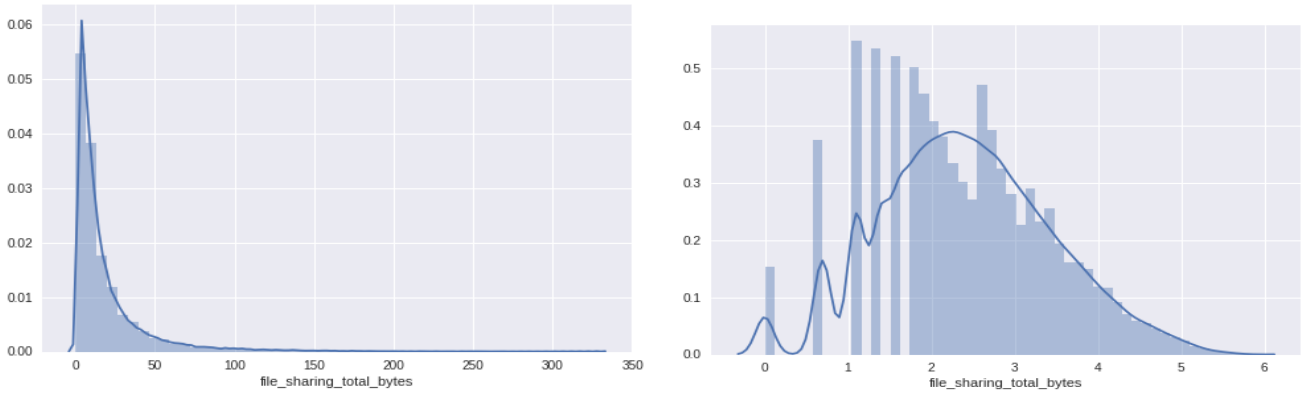
Plot showing the count of each ran_vendor. From this, we can infer that there is no preference given to any specific vendor and this feature may give us detailed hindsight in this data.



This plot represents the type of congestion and the Maximum value of subscriber_count each type has. This clearly indicates that there is always congestion existing beyond 2177 subscribers and after 8890 subscribers, the only type of congestion that exists is 4G_Ran_Congestion.



This is the Correlation matrix of congestion type 4G_BACKHAUL(Y-axis) and 4G_RAN(X-axis) congestion types. This matrix represents the dependency of features on the two classes 4G_Backhaul_ and 4G_RAN_ as these two classes are overlapped to a great extent.



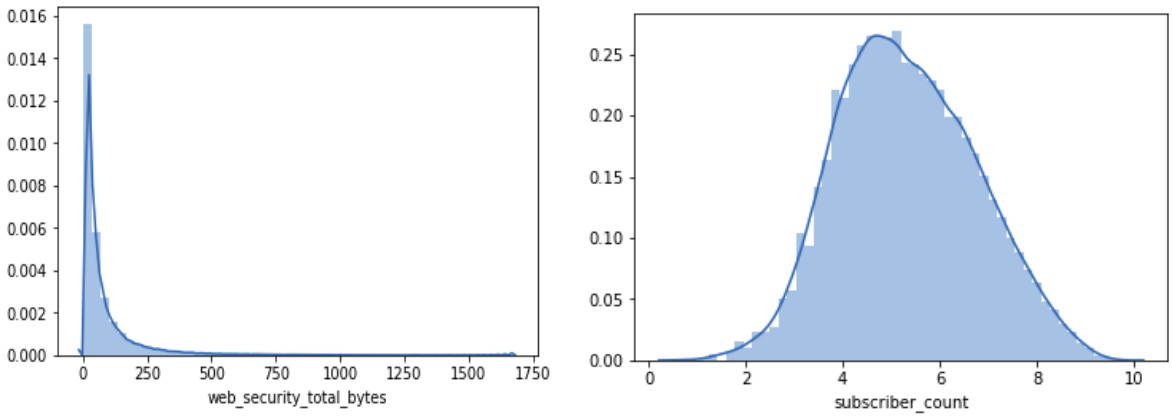
As observed from this plot, the data for 'file_sharing_total_bytes' feature is skewed to a considerable extent. The same is the case with all other types of bytes features.
We hence will apply a log transform to obtain a Gaussian-like distribution as shown below.

3. ALGORITHMS & TECHNIQUES

3.1 Data Pre-processing

After analysing the data, we observed that the 'subscriber count' feature and all the 'bytes transfer' features had skewed data. So, we used logarithmic transformation on the data using the following formula:

X' = log(1+x)



Since skewed data generally reduces the accuracy of the model, after the transformation, our data becomes more meaningful and we are thus able to maximize the efficiency of our model and get the best outputs.

It is seen that despite the above transformation, MCC score on our validation set reduces, hence this is not considered in the final model.

Label Encoding was done on 'ran_vendor' column and 'Congestion_Type' columns as they were of 'Object' data type and needed to be converted into appropriate data type for appropriate use in our model.

Further, normalization of the data was done using two methods, namely Min-Max normalization and Standardisation, with and without using the above-mentioned transformation, respectively. It is observed that standardization yields better accuracy on our validation set.

'par_day' feature was transformed such that each date was replaced by its corresponding day (viz. Sunday, Monday, Tuesday, etc.) according to the calendar.

Some features like 'par_year', 'par_month', 'par_min', were omitted while training the model as these weren't seen to benefit the model particularly month and year as they were the same for all training examples.

3.2 Model Selection

Various different models were applied to the test data in order to get the best output such as Random Forest Classifier, AdaBoost Classifier, Extra Trees Classifier, Logistic Regression, Decision Tree Classifier, MLP Classifier and XGBoost Classifier, Gradient Boosting Classifier, with hyperparameters of a few of them classifiers were changed to get better MCC score as given below along with the custom hyperparameters :-

Model	MCC Score
Random Forest Classifier	0.674
AdaBoost Classifier	0.632
Extra Trees Classifier	0.567
Logistic Regression	0.703
Decision Tree Classifier	0.417
MLP Classifier	0.721
XGBoost Classifier	0.728
GradientBoostingClassifier	0.728
Ensembled Model	**0.732**

We then ensembled the models with our top four models whose MCC score were greater than *0.7*, namely, Logistic Regression, MLP Classifier, XGBoost Classifier, and GradientBoostingClassifier. These models were used for ensembling, so as to get the **best score of 0.732**

3.3 Precautions (Overfitting)

To avoid the negative consequences of overfitting, we split our training data into two parts i.e. the training set (70%) and the validation set (30%) and predictions were made on the latter set. Furthermore, we use K-fold Cross-validation (K=10) to ensure that our model is not biased towards a certain set of data values. On validation dataset, we checked the MCC of training (0.78) and validation set (0.74) and found almost similar score which shows the model is not overfitting.

3.4 Refinement

We found from the exploratory analysis that time series feature of data is not useful because data is given for one particular time for each distinct cell tower. So we removed the “par_min” feature as well. Due to randomization of 3G and 4G, the feature “4G_ran” also became less important. After refining our data for analysis we obtained increased Matthews correlation coefficient values.

3.5 Uncertainty

Since the performance of our model, any model for the matter is not guaranteed to give result one can wholly rely upon. We hence apply the methodology of describing certain **CONFIDENCE INTERVALS** for better determination of the uncertainty in the estimated analysis.
Confidence Interval is a method to restrict the uncertainty of an estimate by the prediction model. The restriction can be applied by bounding the prediction or likelihood of a population parameter like mean.

MCC and other such metrics can be thought of as proportions since they are in [0,1]. The binomial proportion confidence interval can be estimated with (using normal approximation)

$$\hat{p} \pm \hat{\sigma}$$

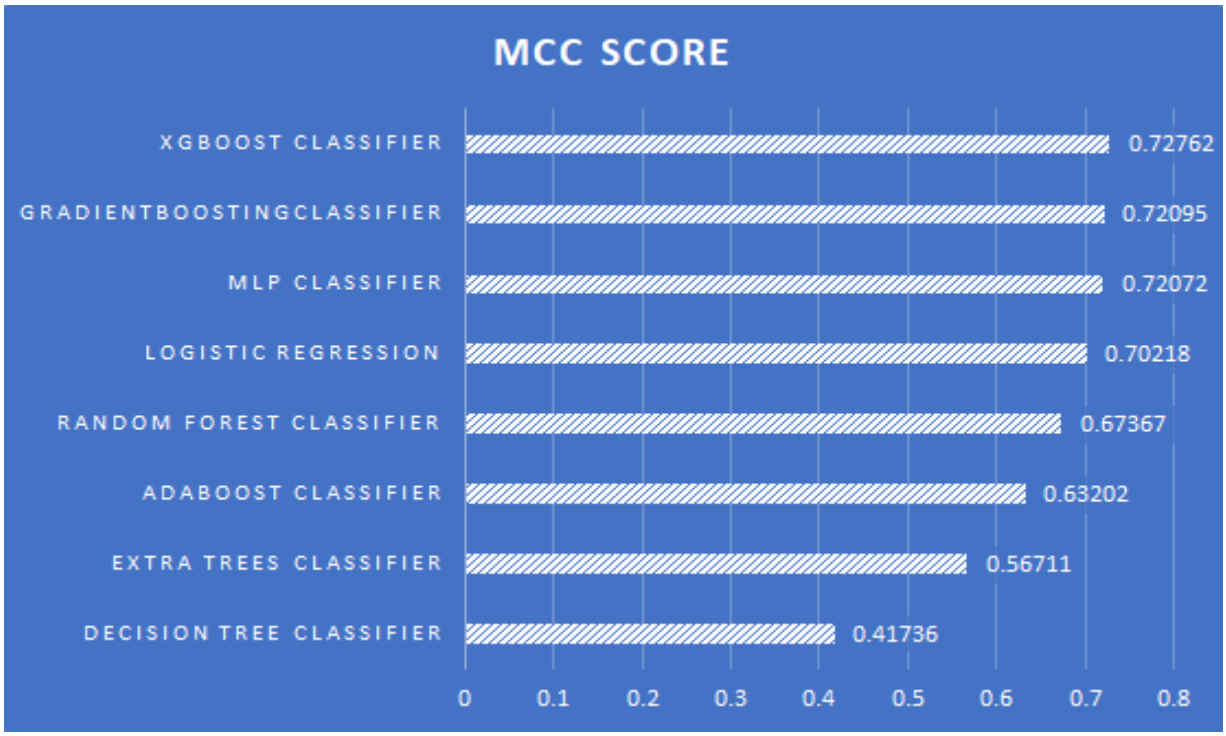
$$\hat{p} \pm z\sqrt{\frac{\hat{p}(1-\hat{p})}{n}}$$

Using the above approximation, we can construct a **90% CI** from our given analysis. With **p_mean = 0.732**, **z = 1.28** and **n = 15** (as we have used K - fold cross-validation with K being 15 in our analysis of the given problem statement, our current model gives the MCC value:

0.732 ± 0.14638

4. RESULT

4.1 Model Performance

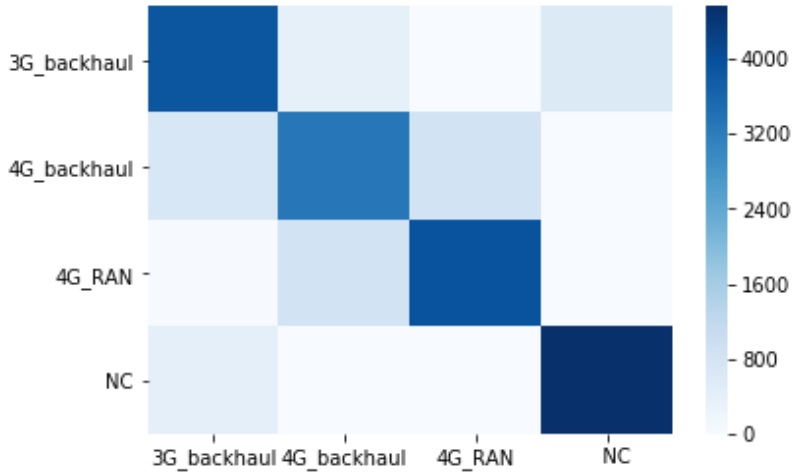


After applying Grid-Search on each classification model for tuning their hyper-parameters we obtained our best results. After ensembling, **MCC score of 0.73169** is obtained.

To predict the final accuracy, Voting Classifier has been used based on four most accurate models have been used i.e., XGBoost Classifier, Gradient Boosting Classifier, MLP Classifier and Logistic Regression which gives a final MCC score of **0.73169**

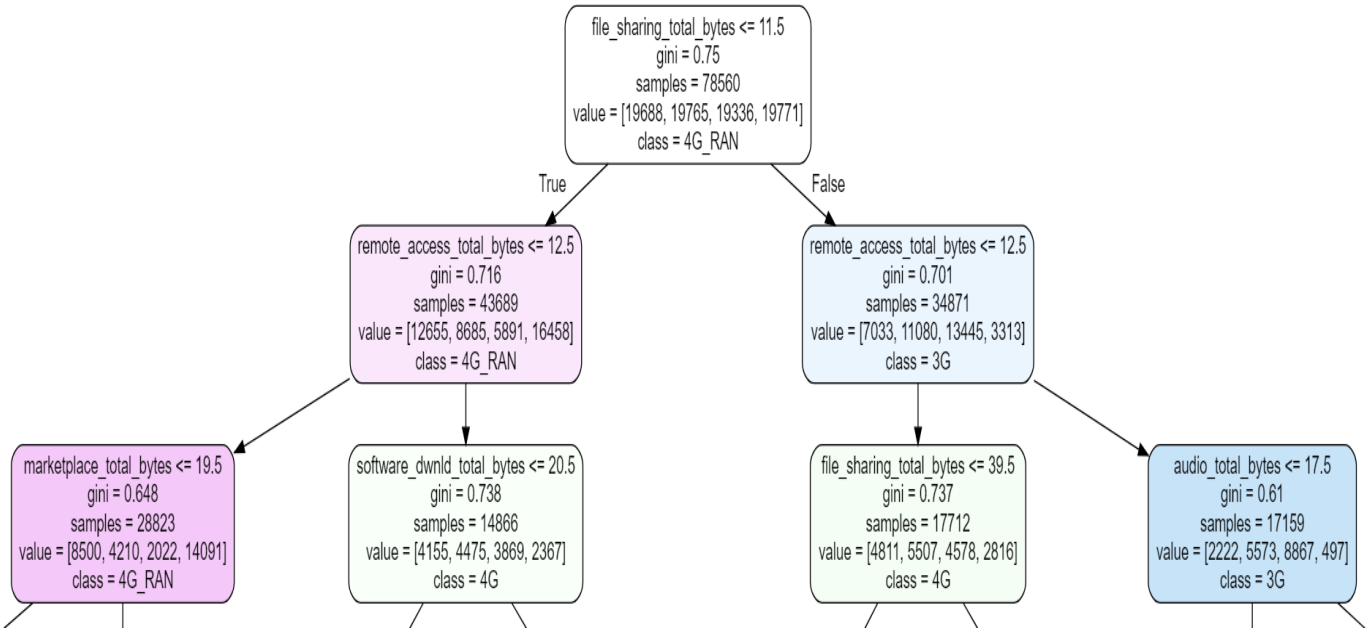
The confusion matrix and corresponding heatmap of the Voting Classifier have been illustrated below for better understanding:

		PREDICTED			
		3G_backhaul	4G_backhaul	4G_RAN	NC
ACTUAL	3G_backhaul	4652	533	0	802
	4G_backhaul	811	4098	937	10
	4G_RAN	28	1138	4684	0
	NC	486	0	0	5389



4.2 Justification

The individual model's performance is highly dependent on features related to Bytes consumption and then slight dependence on other features. This can be seen from the decision tree



The performance obtained from all models didn't change much even after adding extra features or tuning hyper-parameters because the data is highly randomized and each feature has some significant

importance. Also to take into effect the errors caused by different classification models we used ensembling method.

5. CONCLUSION

5.1 Reflection

Throughout the analysis, we came across various hurdles. Some of the prominent ones being to reduce the false positives and true negatives between `4G_BACKHAUL_CONGESTION` and `4G_RAN_CONGESTION`, constructing features that were closely correlated to the congestion classes and finding a relation between the various time patterns. While we were able to address most of these hurdles, we weren't able to use the time features to its full potential primarily because of the reasons that –

1. There were an irregular number of data rows for a particular 5 minute time bin
2. Since cell tower names were unique, we weren't able to make clusters of data values of a particular cell tower and thereafter construction of a Time Series was impractical.

5.2 Improvement

There are several improvements which we could apply to our model but were unable to do so due to various factors such as the requirement of High Processing power and capacity. Some of these improvements are:

1. Using Grid search for tuning hyperparameters of various models such as XGBoost, AdaBoost etc.
2. Using Deep Learning models with efficient number of layers, nodes, and hyper-parameters.

Annexure

Software Stack

1. Python 3.6 - *Language of choice*
2. Pandas - *for Handling files*
3. Numpy - *for complex numerical analysis*
4. Seaborn - *plotting advance visualizations*
5. Matplotlib - *plotting visualizations*
6. Sklearn - *for making machine learning models*
7. XGBoost - *for using boosting models*
8. Datetime - *handling date time data types*

References

1. Tilt in cell tower:
http://www.telecomhall.com/what-is-antenna-electrical-and-mechanical-tilt-and-how-to-use-it.aspx

2. Radiation Pattern of cell tower :
http://neha-wilcom.blogspot.com/2011/05/radiation-pattern-of-cell-tower-antenna.html

3. K-Fold Cross-validation :
https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html

4. Confusion-Matrix:
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html

5. Network Congestion :
https://en.wikipedia.org/wiki/Network_congestion

6. MLP Classifier :
https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html

7. Xgboost Classifier :
https://xgboost.readthedocs.io/en/latest/index.html

8. Gradient Boosting Classifier :
https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html

9. Logistic Regression :
https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

10. Label Encoder :
https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html

11. One Hot Encoder :
https://medium.com/@contactsunny/label-encoder-vs-one-hot-encoder-in-machine-learning-3fc273365621

https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html

12. Heatmap :
https://seaborn.pydata.org/generated/seaborn.heatmap.html

13. Random Forest Classifier :
https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

14. Decision Tree :
https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

15. Grid Search :
https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
https://medium.com/@senapati.dipak97/grid-search-vs-random-search-d34c92946318