**SAFE TRIP ANALYSIS - UK ACCIDENT DATABASE**

**Ch.1 Introduction**

**1.1 Motivation**

The number of road accidents contributes to a high number of casualties all over the world. We wanted to study the factors affecting these accidents and classify their severity and predict future accidents after studying different attributes.

As said on the problem statement, we have two classification problems: Given a route and certain parameters, we want to

1) Predict the number of future accidents

2) If it is risk prone, what kind of an accident severity can occur.

**1.2 Objectives**

Based on the data, wefurther divide our problem into three parts as follows:

1. Predict the accident severity based on Casualties dataset over 3 years
2. Predict the accident severity based on Vehicles dataset over 3 years
3. Predict the number of future accidents using Time Series forecasting

**Ch.2 System Design & Implementation details**

**2.1 Technologies & Tools used (and why)**

1. Pandas: For reading csv files. We also performed different data manipulation operations such as reshaping, merging, data cleaning, and data analysis.
2. Numpy: For supporting capacities that make working with ndarray exceptionally simple and fast. We used it while encoding the features acceptable to our model.
3. matplotlib.pyplot and seaborn: For visualizations while exploring the different data files - Accident, Casualties, Vehicles. Various plotting options provided by matplotlib and seaborn helped us understand the relationship between features.
4. Scikit-learn: Several classification algorithms and the evaluation functions.
5. Other classification models: XGBoost, AdaBoostClassifier and RandomForestClassifier.

**2.2 Algorithms considered in our project:**

We have defined three major parts in our project.

1. Data exploration and classification algorithms - Casualties dataset
2. Data exploration and classification algorithms - Vehicles dataset
3. Time Series Forecasting

**Classification algorithms**

We are using two datasets for classification:

1. Casualties database for three years (2017, 2018 and 2019)
2. Vehicles database for three years (2017, 2018 and 2019)
3. **Data exploration and classification algorithms - Casualties dataset**

The target column is: Casualty\_Serverity (classes = 1, 2, 3). This is a multi class problem. We are using various classification models, known for their good performance. We chose these algorithms because these are well known models for classification problems.

We are using 11 classification models for this dataset.

1. Logistic Regression
2. Logistic Regression with 10-fold CV
3. Random Forests with 10-Fold
4. Polynomial Kernel SVM
5. LinearSVC
6. Decision Tree Classifier
7. SGD Classifier
8. OneVsRestClassifier SVC model
9. KNeighborsClassifier & GridSearchCV
10. XGBoost
11. AdaBoost

Analysis of Casualties data:

For analyzing the casualties data, we are considering the data for the years 2017, 2018, and 2019. We performed the following steps for this process.

1. Gathering and loading data
2. Data Exploration
3. Data Preprocessing & Cleaning (Imputing missing values)
4. Machine Learning models (Classification)
5. Training and testing the model with train and test datasets
6. Evaluation of each model (Accuracy)
7. Comparison of these models for best accuracy.
8. **Data exploration and classification algorithms - Vehicles dataset**

The classification problem is to analyze the vehicle attributes which might lead to accidents. The vehicle dataset is linked to the casualty dataset by the column vehicle reference. Following are the classification algorithms used for this dataset:

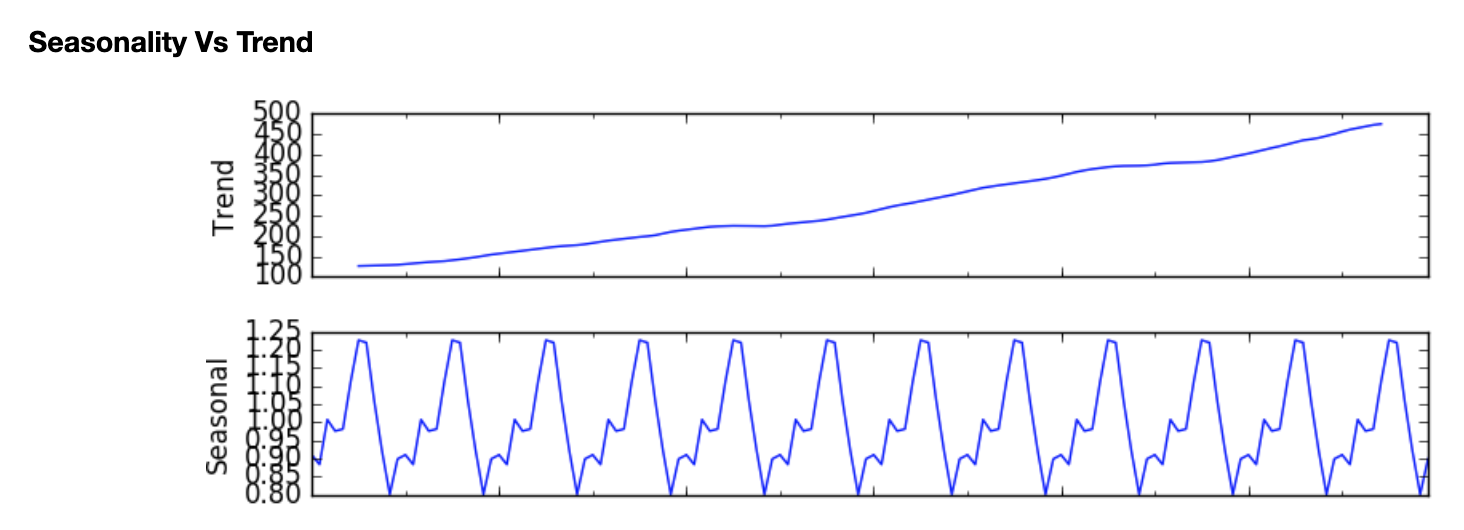
1. Logistic Regression
2. AdaBoost Default Classifier
3. AdaBoost custom setting Classifier
4. RandomForestClassifier
5. Bagging Classifier
6. Decision Tree
7. SVM

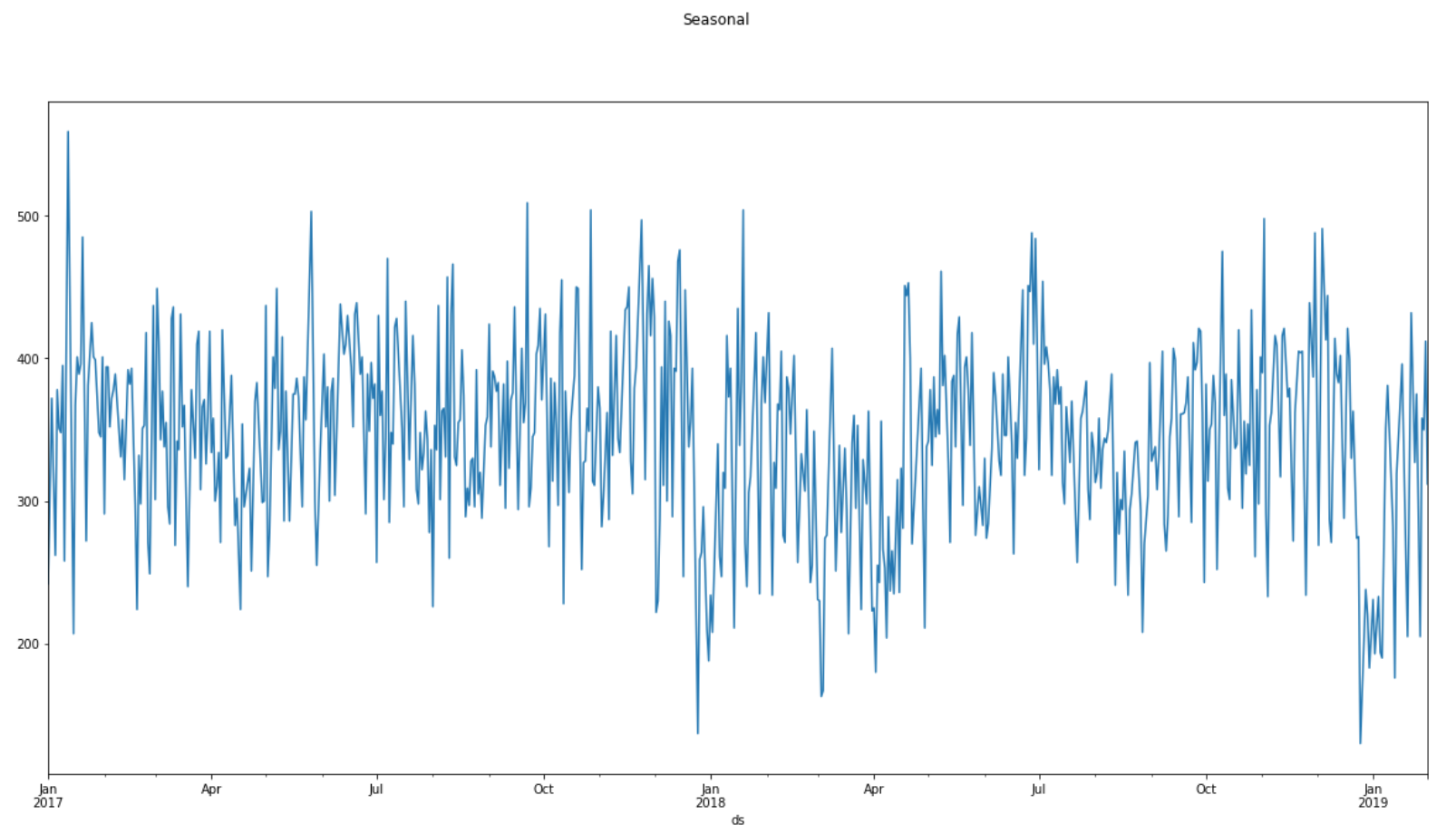
Analysis of Vehicles data:

We combined the data of the years 2017, 2018 and 2019 to cover larger data for better prediction accuracy. Given below are the steps used in the classification of the vehicles dataset.

1. Gathering data and loading data
2. Merging the data of the above mentioned three years
3. Data exploration
4. Data cleaning and feature manipulation
5. Classification models
6. Training and testing the merged dataset
7. Evaluation of models
8. Model accuracy comparison to get better results
9. **Time Series Forecasting:**

|  |  |
| --- | --- |
| **Algorithm** | **Reason** |
| **SARIMAX** | Seasonality and exogenous influences (seasonality and normal ARIMA do not blend well) are the contrast between ARIMA and SARIMAX. We tested the model with both simple ARIMAX and SARIMAX, SARIMAX performed better for our data. The reason for this is that our dataset possesses Seasonal characteristics. |
| **Prophet**  ***(Selected)*** | Facebook Prophet is not searching for any causal links between the past and the future. Instead, it simply attempts to use a linear or logistic curve and Fourier coefficients for the seasonal components to find the best curve to match the results. There is also an aspect of regression, but it is for external regressors, not the time series itself. The regressors help in adding additional features while model building. In our application, we needed a model to understand the effect of features such as ‘Road Type’ was important. |

****

****

**Plot for our dataset**

1. statsmodels.api

Statsmodels is a Python module which provides classes and functions for many different statistical models to be calculated, statistical tests to be carried out, and statistical data exploration to be carried out. Statsmodels.tsa(part of statsmodels.api) includes groups and functions of the algorithm that are useful for study of time series. Univariate autoregressive (AR), vector autoregressive (VAR) and univariate autoregressive moving average (ARMA) models are simple models.

1. fbprophet

Prophet is a method focused on an additive model for forecasting time series data where non-linear patterns match with annual , monthly, and regular seasonality, plus holiday results. It fits well for time series that have clear seasonal influences and historical evidence from many seasons. The Prophet is immune to lost data and pattern fluctuations, and usually manages outliers well.

Prophet is open source software published by the Core Data Science team of Facebook. Facebook recommends using Prophet for forecasting with data that has the following characteristics:

* hourly, daily, or weekly observations with at least a few months (preferably a year) of history
* strong multiple “human-scale” seasonalities: day of week and time of year
* important holidays that occur at irregular intervals that are known in advance (e.g. the Super Bowl)
* a reasonable number of missing observations or large outliers” [3]

**Architecture Diagrams**

1. **Classification algorithms (for both casualties and vehicles datasets)**

Below is the flow chart of the classification model training in our project:

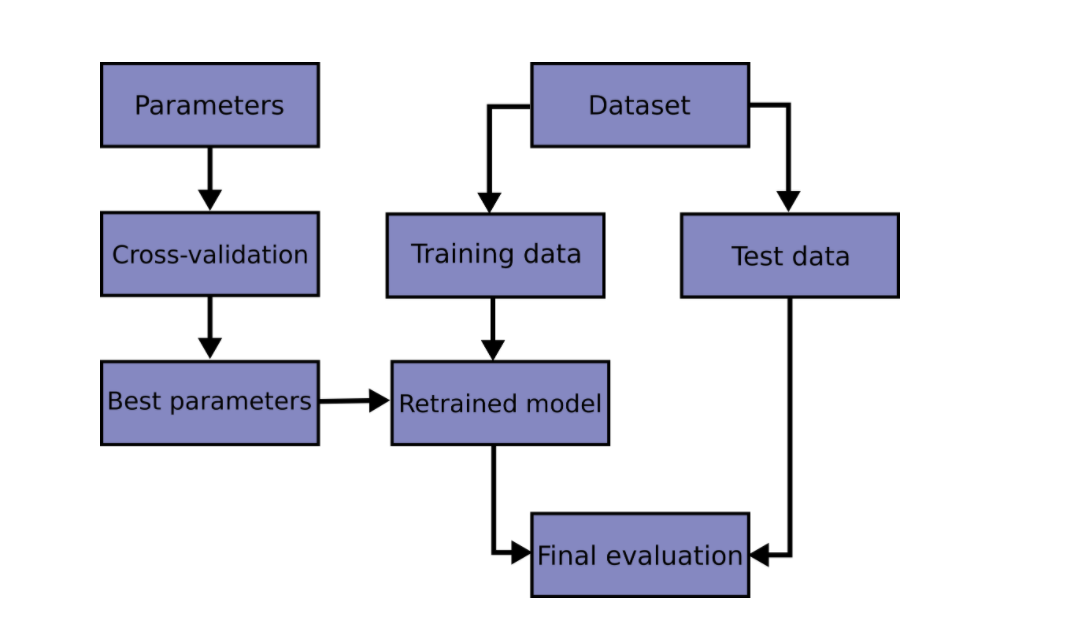
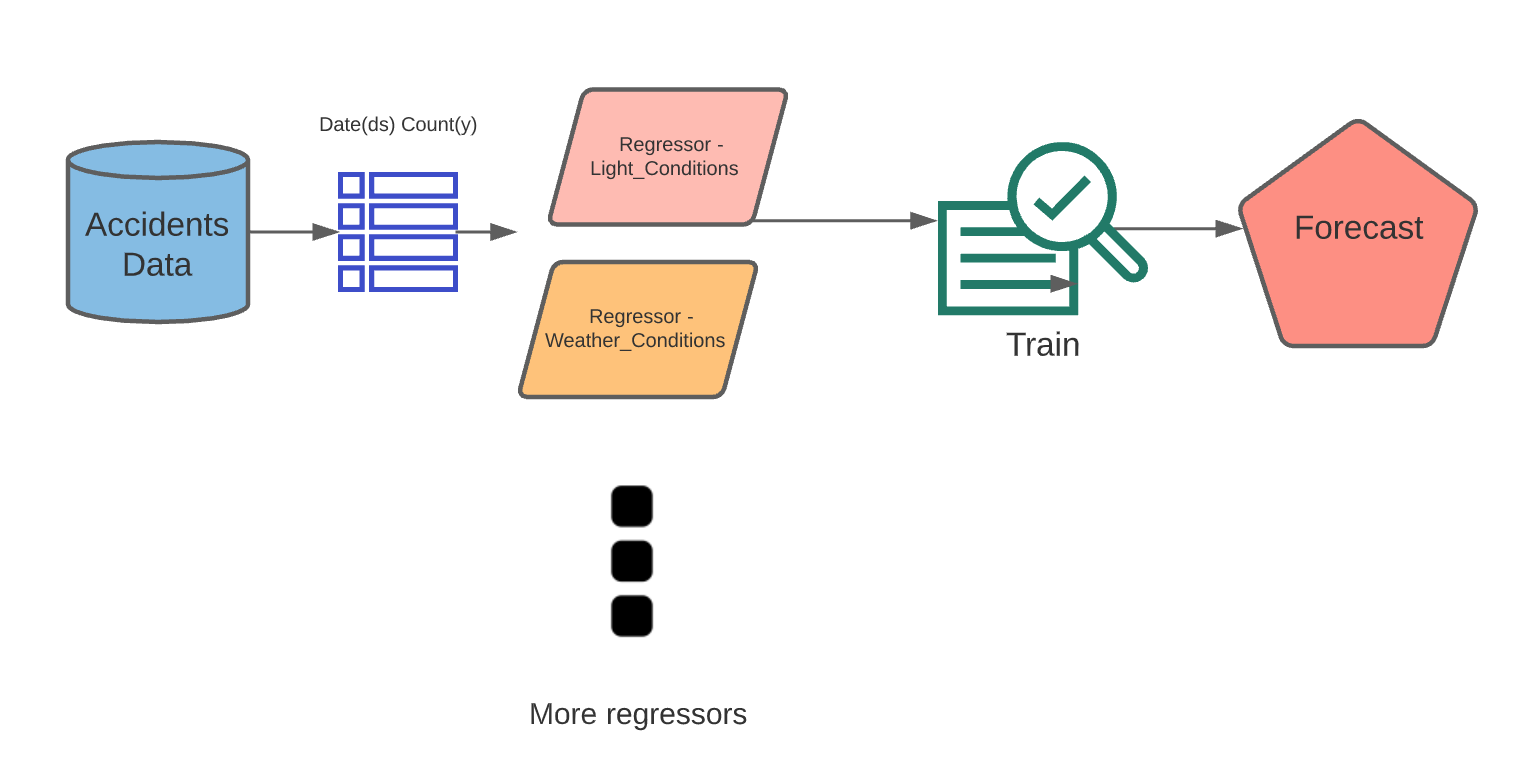


Fig 2.2.1: Workflow representation of classification (general) model training

Image source: <https://scikit-learn.org/stable/modules/cross_validation.html>

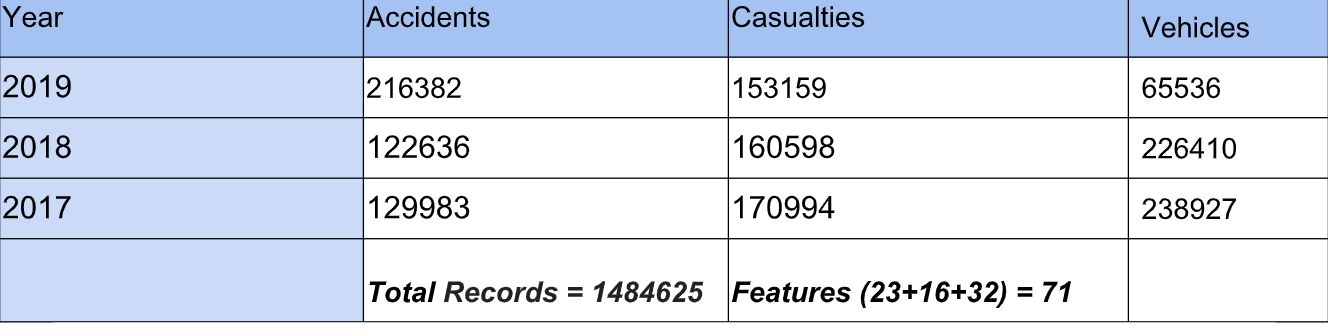
1. **Time Series Forecasting:**



**Ch.3 Experiments / Proof of concept evaluation**

**3.1 Total data of the project**

The features in these three files have been referenced using a common “Accident\_Index” column. The dataset also contains a variable look-up excel sheet which provides information about various features in the three CSVs. The dataset is available on the UK's government [website](https://data.gov.uk/dataset/cb7ae6f0-4be6-4935-9277-47e5ce24a11f/road-safety-data).



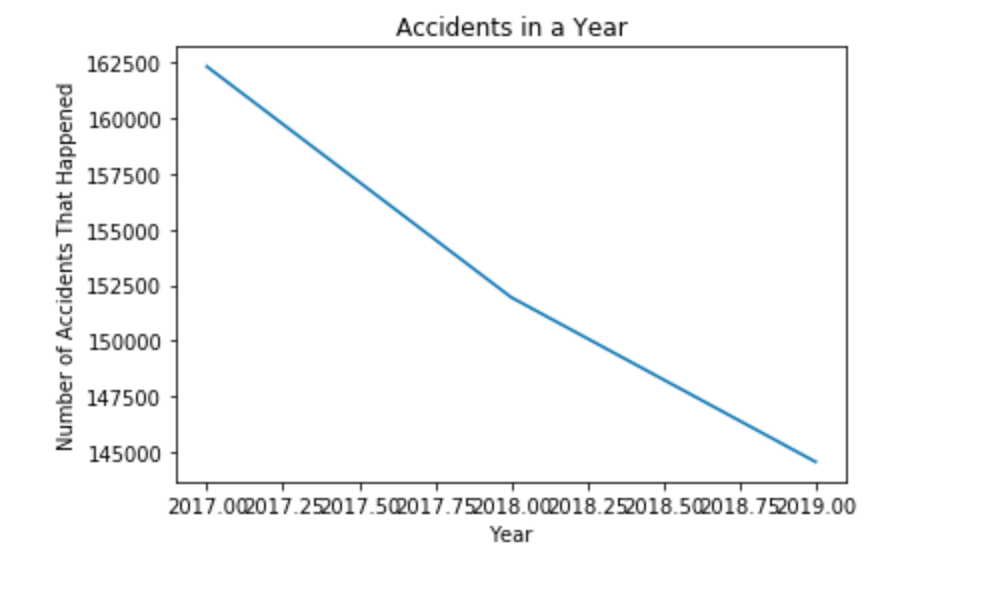
**3.2 Data preprocessing:**

1. Casualties dataset:

The main important steps of preprocessing done in this dataset are

1. Gathering and loading data
2. Data Exploration
3. Data Preprocessing & Cleaning (Imputing missing values)
4. Gathering and loading data of Casualties

Analysis on casualties by year wise comparison.



1. Data Exploration of Casualties

Our main focus in this step is to analyze and observe as much meaning/sense as possible from the available data. In this data, we have 16 columns.

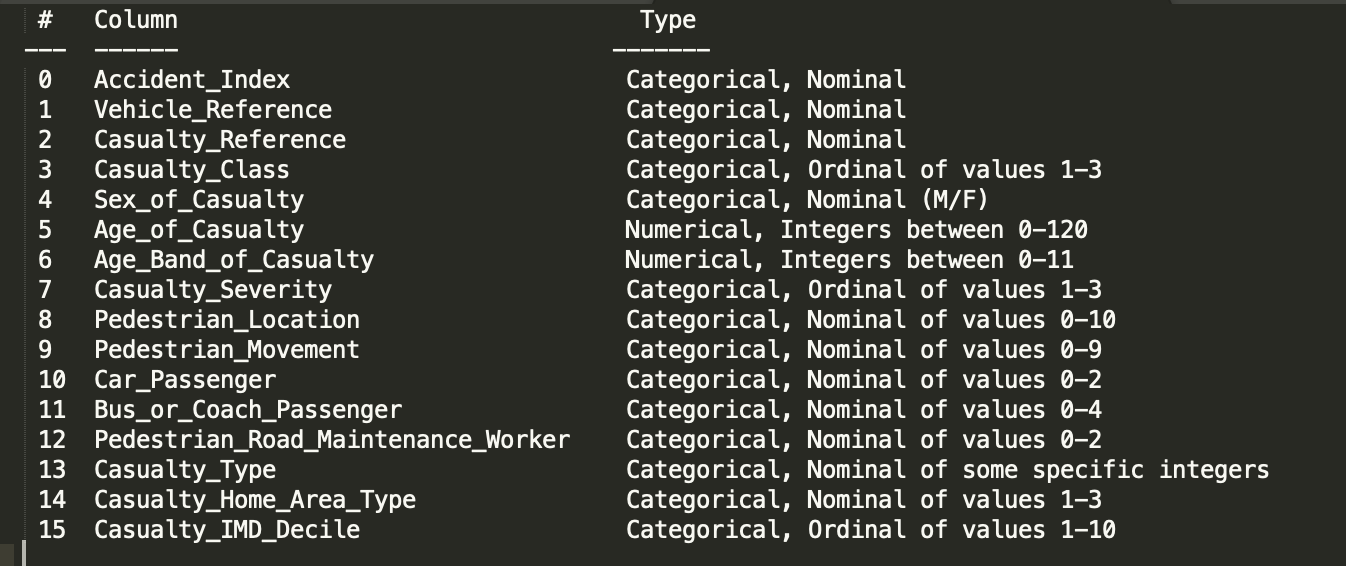


Figure: Casualties data types

**Feature Selections and Data Visualizations of Casualties:**

We used the below visualizations in the ipython notebook.

1. Correlation Matrix shown in the notebook
2. Scatter plot shown in the notebook
3. Histogram of all the columns.

While we have elaborated all the observations in detail in the Data\_Exploration.ipynb notebook. We are writing here only the most important findings considering the conciseness of the report.

1. More risk prone people are those who travel in minibuses.
2. Most of the people who underwent injuries were males.
3. Most of the accidents were slight injuries. The second most was serious injuries. Fatal cases were very rare (relatively).

**Data Preprocessing of Casualties -**

In this step, we are removing the dirty/missing/unknown data based on the data lookup (added in data/variable lookup.xls).

As explained in the Data\_Exploration.ipynb notebook, we encountered many missing data.

We explored the three options of

1) Dropping the rows of the missing values

2) Dropping the column itself

3) Impuring the missing values with a suitable value.

1. In the data preprocessing process, we dropped the rows of missing values w.r.t the following columns, since it was not even 1% of the total data.

1) Sex of causality

2) Age of causality

3) Pedestrian\_Location

4) Car\_Passenger

5) Bus\_or\_Coach\_Passenger

6) Pedestrian\_Road\_Maintenance\_Worker

7) Casualty\_Type

1. Valid range of Casualty\_Home\_Area\_Type is [1-3]. -1 is bad data. Bad data is 64465/484748 = 13.2%. This is a lot of data. We do not want to end up risking the data.

Solution: Imputed this data by mode.

1. Casualty\_IMD\_Decile has 458848 missing values. However, as seen in the correlation matrix and histogram, this does not look like an important feature in this data. So, we could safely drop the column.

Feature selection for casualties:

All features except Casualty\_IMD\_Decile are selected in the dataset.

**2. Feature Selection and Data processing in vehicles dataset:**

The Vehicles dataset as mentioned above is merged data of three years.

1. Data gathering and loading:

The vehicles data of all the years is loaded from the website <https://data.gov.uk/dataset/cb7ae6f0-4be6-4935-9277-47e5ce24a11f/road-safety-data>.

Merged the dataset of mentioned years.

Total no. of rows = 681716

No. of columns are 23 as the datasets are merged w.r.t the columns.

1. Data cleaning and preprocessing:
2. Getting the data type of each column to know which type of processing can be done on a particular column.
3. The data collected has a significant count of null values. Given below is the count of original column names and their corresponding null values.
4. Most of the columns of the dataset have values which are codes that represent different labels.
5. As the first step of processing we dropped the rows which have null values for all the columns.
6. We have plotted bar graphs of all the attributes to get the idea of the existing values. Box plots are plotted to detect outliers.
7. Based on the data available and the study of the data, missing values of the columns are imputed using different techniques.
8. Outlier detection with lower and upper threshold at 0.01% and 99.99% respectively. These values are replaced in the dataset.

After preprocessing and imputing values, we decided to drop 5 columns from the vehicles dataset.

The columns dropped were the following:

|  |  |
| --- | --- |
| Column Name | Reason for dropping |
| Journey\_Purpose\_of\_Driver | More than 70% of the data was missing. Imputing values will not help our model. |
| Age\_of\_Driver | Another column Age\_Band\_of\_Driver represents the similar data in a concise form which is much cleaner. |
| Driver\_IMD\_Decile | No information about what the column is in reference to. This column does not give any meaningful information that helps the classification. |
| Driver\_Home\_Area\_Type | No information about what the column is in reference to. This column does not give any meaningful information that helps the classification. |
| Vehicle\_IMD\_Decile | No information about what the column is in reference to. This column does not give any meaningful information that helps the classification. |

Post all the pre-processing the dataset has 0 null, 0 outliers 681716 and 18 columns.

**3. Feature Selection for TimeSeries forecasting**

**Time Series Forecasting:** After the initial data exploration, we found out the important contributors to frequency of accidents. Furthermore, finding correlations between different features helped us narrow down the input features for the model.

Following are the selected features for Time series forecasting model :

*Day\_of\_Week*

*Light\_Conditions*

*Weather\_Conditions*

*Speed\_limit*

*Pedestrian\_Crossing-Physical\_Facilities*

*Special\_Conditions\_at\_Site*

*Number\_of\_Vehicles*

*Police\_Force*

*2nd\_Road\_Class*

**Data preprocessing for Time Series Forecasting**

We had to convert the columns to forms accepted by Prophet (ds:Date and y: Target)

And ARIMA needs in the format (Date as index column and second column is the Target column)

Algorithms

1. **Classification Models for Casualties**

Test and Train dataset split:

In this problem, we used an 80:20 test and train split.

Machine Learning Models for Classification on Casualties:

We are comparing 11 classification algorithms to analyze the Casualties dataset over 3 years (2017, 2018, 2019). We experimented with the below classification models, which are widely known for their good performance.

Evaluation metrics:

This is a classification problem and we defined the Classification accuracy as the evaluation metric. Accuracy% = Number of correct predictions / (Total number of predictions) \* 100

In the confusion matrix, this is equal to Accuracy% = TP+TN/(TP+TN+FP+FN) \* 100

Aim of the Classification:

We aim for the model which gives the highest accuracy.

N-fold cross-validation:

In our project, we used n-fold cross-validation for the Logistic Regression and Random Forests algorithms. We used 10-Fold in this step i.e., n=10.

1. **Classification models on Casualties:**
2. ***Logistic Regression***: This model is commonly used to estimate the probability of a prediction of a particular class. We experimented with C = 0.01, 0.1 and 1. All three experiments resulted in an accuracy of ***82.82%.***
3. ***Logistic Regression with 10-Fold Cross-Validation:*** achieved an accuracy of ***82.94%.***
4. ***Random Forests with 10-Fold Cross-Validation:*** An ensemble learning method, achieved an accuracy of ***82.028%***.
5. ***Polynomial Kernel SVM:*** A 3 polynomial SVM gave an accuracy of ***82.82%.***
6. ***LinearSVC:*** The algorithm aims to get the ‘best fit’ hyperplane that classifies the data, achieved an accuracy of ***82.82%.***
7. ***Decision Tree Classifier:*** Builds builds the model in the form of a tree structure. Achieved an accuracy of ***81.337%.***
8. ***SGD Classifier:*** A popular algorithm for its ease of use and efficiency. In this step, we achieved an accuracy of ***81.33%***
9. ***OneVsRestClassifier SVC model:*** Also known as one-vs-all, this strategy consists of fitting one classifier per class. For each classifier, the class is fitted against all the other classes. In addition to its computational efficiency (only n\_classes classifiers are needed), one advantage of this approach is its interpretability [5]. In this step, we achieved an accuracy of ***82.61%.***
10. ***KNeighborsClassifier & GridSearchCV:*** KNeighborsClassifier is implemented based on a k-nearest neighbor vote. In this step, we achieved an accuracy of ***80.38%.***
11. ***XGBoost:*** XGBoost provides parallel tree boosting. We used XGBoost for this problem and achieved an accuracy of ***82.81%.***
12. ***AdaBoost***: AdaBoost iteratively learns from the mistakes of weak classifiers, and tries to turn them into strong ones. In this step, we achieved an accuracy of ***82.83%***
13. **Classification Models for Vehicles:**

For classification we merged the Casualty\_Severity column from the Casualty dataset. The pivot point for the merge is Vehicle\_Reference.

The train and test split for the data is *70%* and *30%* respectively.

Following are the classifiers used in the vehicles dataset:

1. ***Logistic Regression:***It is a linear classifier. It is best suited for binary classification. It is used here for the multi class classification as a starter classification model. It gave an accuracy of ***83.06%***.

Scaling of dataset: Scaling required due to inability of processing the dataset.Accuracy dropped to ***1.14%*** with the scaled dataset.

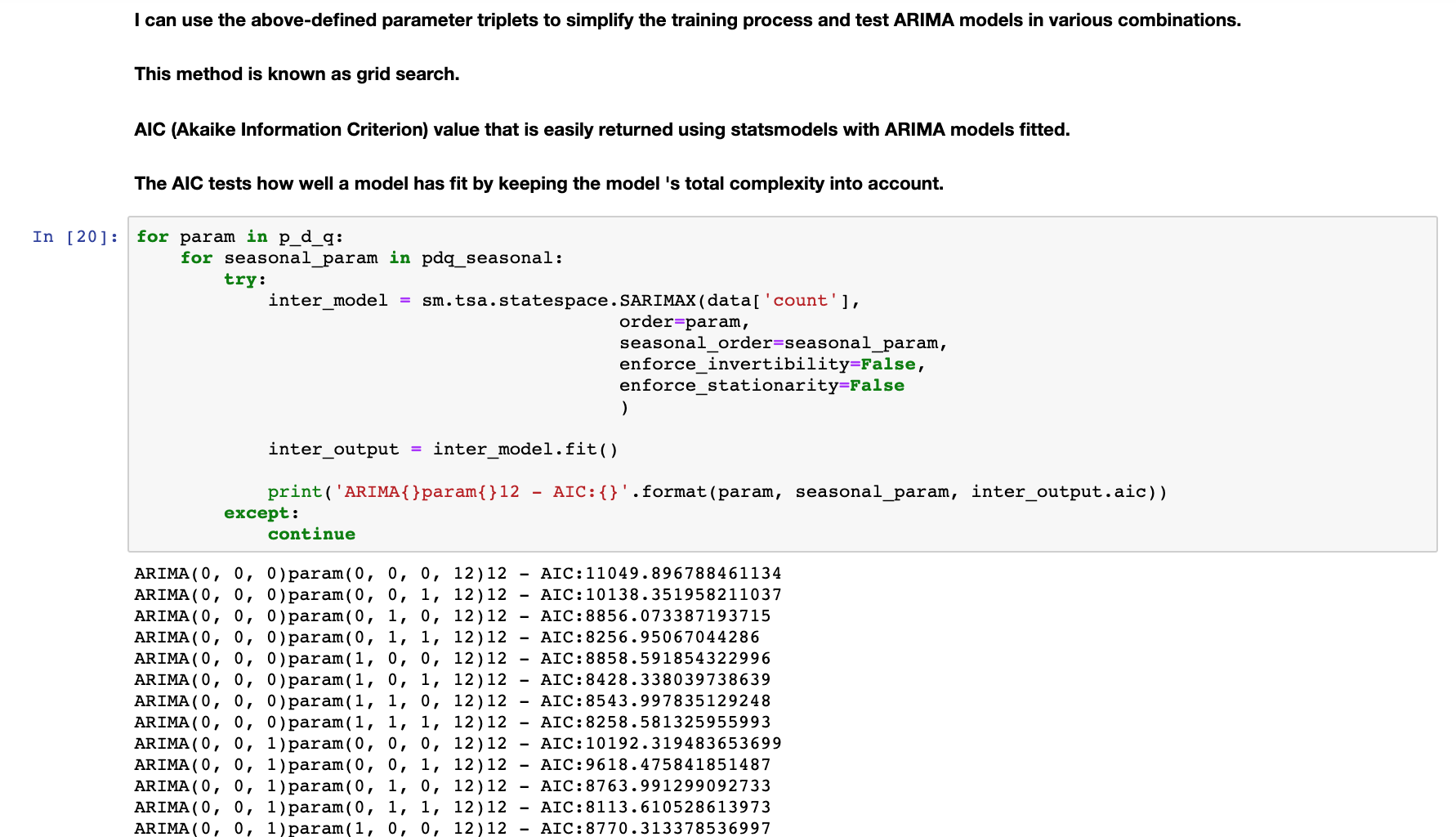
1. ***AdaBoost Default Classifier***:It is an ensemble classifier. It takes multiple average classifiers and creates a strong one. Gives better results when data is clean.Its gave an accuracy of ***83.16%***.
2. ***AdaBoost Custom setting Classifier***:It is basically default AdaBoost but with custom settings. It gave an accuracy of ***83.17%***
3. ***RandomForest Classifier:*** It is an ensemble machine learning algorithm. It is useful since we have a multiclass classification problem. It gave an accuracy of ***81.12%***.
4. ***Bagging Classifier:*** A Bagging classifier is an ensemble meta-estimator that fits base classifiers each on random subsets of the original dataset and then aggregate their individual predictions. It gave an accuracy of ***78.69%.***
5. ***Decision Tree:*** A Decision Tree is a Supervised Machine Learning classifier where the data is continuously split according to the parameters. It gave an accuracy of ***73.74%***.
6. ***SVM:*** SVM uses a hyperplane to separate the input variable space into different classes. SVM works better for binary classification, but can also work good for multiclass. It gave an accuracy of ***83.06%***

***For evaluating the models we have used confusion matrix in the vehicle dataset. The details and implementation can be found in the jupyter notebook.***

**2. Time Series Forecasting:**

For the Prophet, the important contributors were the external regressors(features). We manually added and removed them to understand when the model performed best.

For ARIMA, we performed grid search for different combinations of hyperparameters. The combination having the least AIC is the best option.



**Evaluation methodology for Time Series Forecasting**

We split the data by date since time series need the date in appropriate order.

Training Set - 01-01-2017 to 31-01-2019

Test Set - 01-02-2019 to 31-12-2019

**Analysis and Results:**

1. **Classification models on Casualties data:**

Here is the summary of all the models:

1. Logistic Regression - Accuracy 82.8%

2. Logistic Regression with 10-fold CV - Accuracy 82.94%

3. Random Forests with 10-Fold - Accuracy 82.028%

4. Polynomial Kernel SVM - Accuracy 82.82%

5. LinearSVC - Accuracy 82.82%

6. Decision Tree Classifier - Accuracy 81.337%

7. SGD Classifier - Accuracy 81.33%

8. OneVsRestClassifier SVC model - Accuracy 82.61%

9. KNeighborsClassifier & GridSearchCV - Accuracy 80.38%

10.XGBoost - Accuracy 82.81%

11.AdaBoost - Accuracy 82.83%

So far, we have run and evaluated 11 popular classification models.

Best Classification Models for Casualties:

Logistic Regression with 10-fold CV (82.94% accuracy) and AdaBoost (82.83% accuracy) are the best performing models amongst all.

Another observation is: In the SVM model, a high degree of polynomial degree creates a huge number of features. In this problem, both linear and polynomial degrees gave the same accuracy. Also, the logistic regression model gave the same accuracy for all 3 different values of C.

1. **Classification models on Vehicles data:**

We used 7 classification models for the vehicle dataset. For comparing all the classifiers we started by using undersampling and oversampling.

***Undersampling*** reduced the accuracy of the logistic regression model to ***57.30%***.

***Oversampling*** reduced the accuracy to ***79.94%*** for the logistic regression model.

Observing this huge drop in accuracy from undersampling and oversampling, we did not pursue this approach for the other classification models.

The best accuracy is achieved using the ***AdaBoost Custom setting Classifier*** which is ***83.17%***

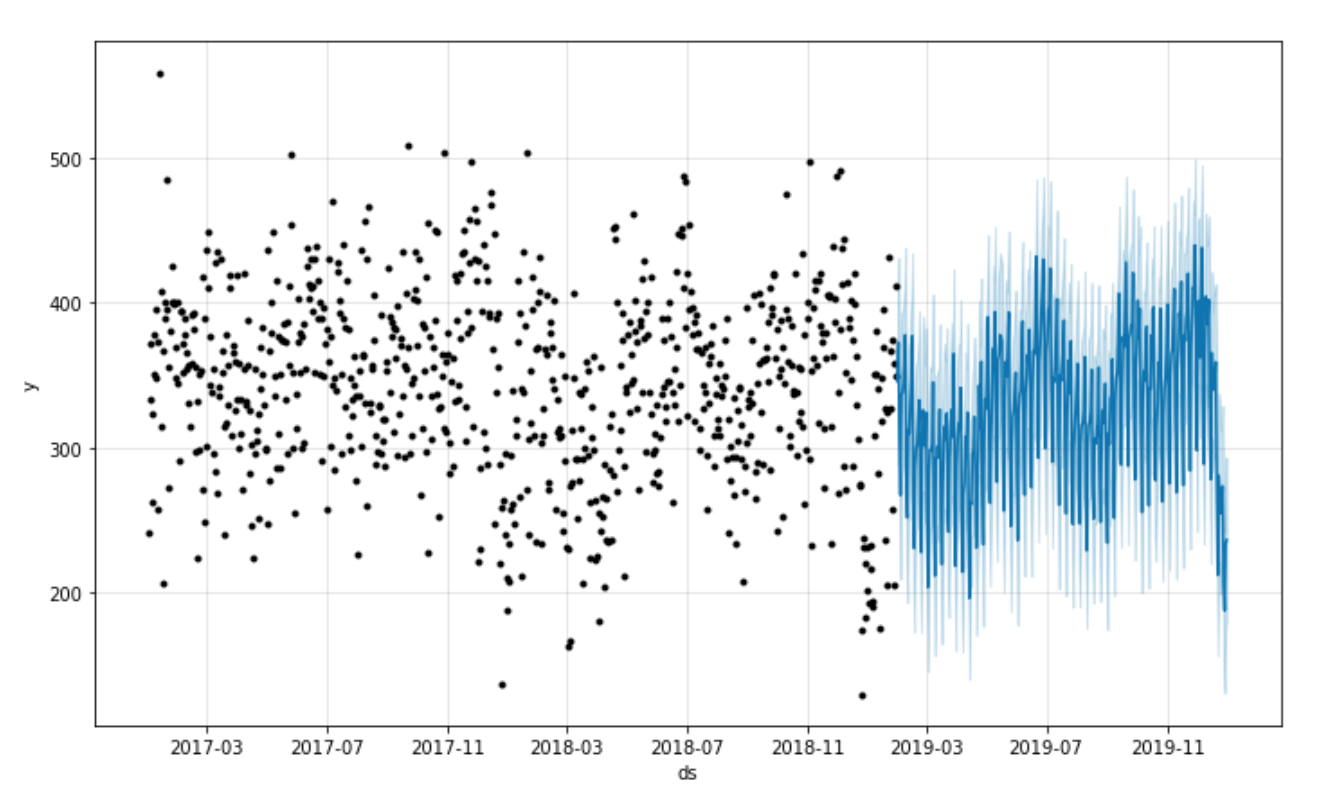
1. **Time Series Forecasting:**

Following is a screenshot of the forecast generated by Prophet.

Dark blue is yhat. At the top, the light blue is yhat-upper and at the bottom, the light blue is yhat-lower.

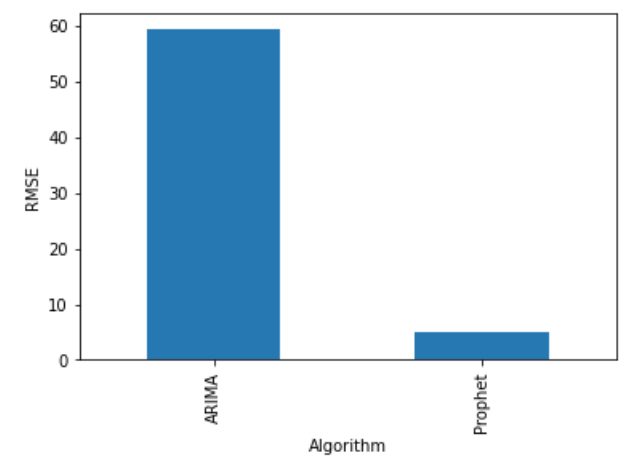
*yhat* is the predicted value. *yhat\_lower, yhat\_upper* are the uncertainty interval (predicted range can be between these two values)

If outliers (dots) impact the prediction, they can be excluded.



Historical data \*-------------------------------------------\* Forecasted data

The final score was evaluated using RMSE(Root Mean Square Error) and Prophet was a clear winner for our dataset

****

**Ch.4 Discussion & Conclusions**

* Decisions made/Things that worked

1. Division of tasks was decided early and that helped members to focus on respective responsibilities.
2. Data preprocessing helped in the classification models. Otherwise, it could have been error-prone.
3. Regular meetings and peer evaluations gave us the best team work experience.
4. Timely planned tasks done.

* Difficulties faced/Things that didn’t work well

1. Exploring time series forecasting algorithms was time consuming.
2. Since we explored both ARIMA and Prophet, it was difficult to understand subtlety initially.
3. Some of the classification models like SVM with polynomial kernel, XGBoost, RandomForest and any model with Cross Validation and Grid Search were very slow and time consuming. We have initially implemented ensemble learning algorithms but we had to remove them later because they were taking many hours to run and sometimes, the kernel crashed too.

* Conclusions

1. We were able to explore the data and decide important features from a large set of features.
2. We were able to understand how time series algorithms work.
3. We were able to predict whether the given route or not based on given circumstances. We hope this will help people and the concerned authorities to be aware of the accident risk and take necessary preventive steps.

Ch.5 Project Plan / Task Distribution

|  |  |
| --- | --- |
| Tasks | Group Member |
| Dataset Selection | All the team members |
| Data Exploration - Accidents | Harshada |
| Data Exploration - Casualties | Manasa |
| Data Exploration - Vehicles | Pragati |
| Feature Selection for Time Series | Harshada |
| Feature Selection for Classification - Casualties | Manasa |
| Feature Selection for Classification - Vehicles | Pragati |
| Preprocessing - Time Series - ARIMA | Harshada |
| Preprocessing - Time Series - Prophet | Harshada |
| Preprocessing - Classification - Casualties | Manasa |
| Preprocessing - Classification - Vehicles | Pragati |
| Model Building - Time Series - ARIMA | Harshada |
| Model Building - Time Series - Prophet | Harshada |
| ModelBuilding - Casualties - 11 models | Manasa |
| Model Building - Classification - Vehicles | Pragati |
| Evaluation - Time Series | Harshada |
| Evaluation - Classification - Casualties | Manasa |
| Evaluation - Classification - Vehicles | Pragati |
| Report | All the team members |
| Presentation | All the team members |

**References:**

[1]<https://stats.stackexchange.com/questions/472266/inference-in-time-series-prophet-vs-arima>

[2]<https://facebook.github.io/prophet/docs/quick_start.html>

[3]<https://mode.com/blog/how-facebook-made-business-forecasting-scalable-for-the-masses-with-prophet/>

[4] <https://towardsdatascience.com/cross-validation-in-machine-learning-72924a69872f>

[5]<https://scikit-learn.org/stable/modules/generated/sklearn.multiclass.OneVsRestClassifierhtml>

[6]<https://towardsdatascience.com/data-cleaning-in-python-the-ultimate-guide-2020-c63b88bf0a0d>

[7]<https://machinelearningmastery.com/machine-learning-algorithms-mini-course/>

[8]<https://towardsdatascience.com/machine-learning-part-17-boosting-algorithms-adaboost-in-python-d00faac6c464>

[9]<https://github.com/codebasics/py/blob/master/DeepLearningML/14_imbalanced/handling_imbalanced_data.ipynb>

[10]<https://github.com/GenTaylor/Traffic-Accident-Analysis>