Mini Project Report

on

Railway tracking and arrival time prediction

(CSE V Semester)

2021-2022

Submitted to:

Mr. C.D. Bhatt

(CC-CSE-B-V-Sem)

Guided by:

Mrs. Rishika Yadav

(External Guide)

Submitted by:

Mr. Priyanshu Gupta

Roll. No.: 1918567

CSE-B-V-Sem

Session: 2021-2022

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GRAPHIC ERA HILL UNVERSITY, DEHRADUN

TABLE OF CONTENTS **Page No.**

1. ABSTRACT

The objective of this project is to predict the delay in arrival of train(s) given certain features like destination, day of week etc. These features are discussed in detail in the latter part of the report.

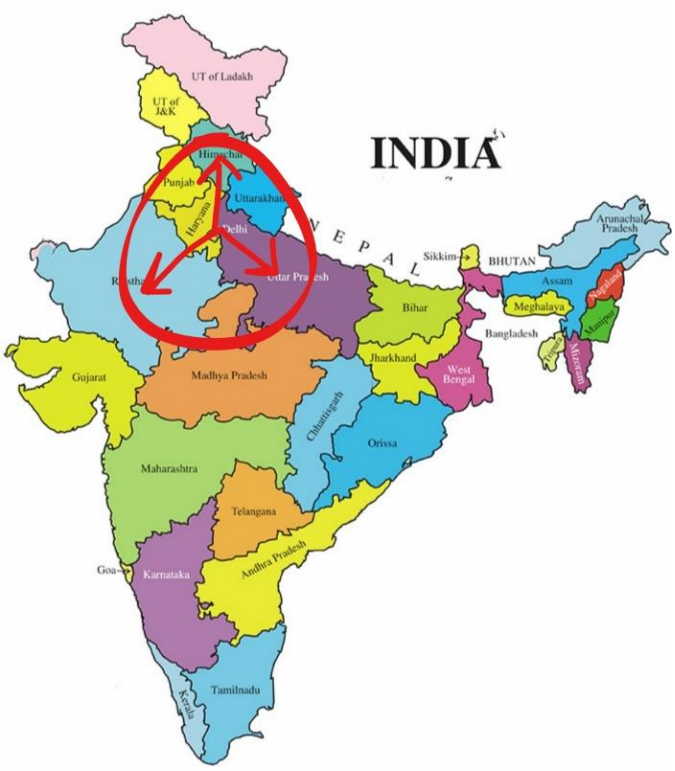
Rail transportation is a convenient and safe in many countries. However, Rail transportation in some countries has significant long delays. Arrival time prediction and rescheduling the time table are partial solutions to tackle the delay problem. In this paper, the relationship between measurable properties and the delay time are studied in order to develop an arrival time prediction. The result of this experiment has three parts. The relationship between properties and arrival late are then visualized and discussed.

2. PROJECT INTRODUCTION AND MOTIVATION

In order to make the analysis simpler we set out for identifying some of major station in IRN in terms of degree of station (connecting stations) and weight of station (traffic). Three key points were observed:

- Major stations are located in close vicinity to the metropolitan cities in India (e.g. Hazrat Nizamuddin near Delhi, Kalyan near Mumbai etc.).
- Major stations that are located in the central parts of the country or at the meeting points of railway lines connecting different zones.
- Most of traffic resided on tracks connecting these major stations.

Thus we decided to use New Delhi and NCR regions to be the origin of station we analyse. We broadly divided the destinations to the four zones:



i) UP/Bihar Region
ii) Chandigarh/Jammu Region
iii) Rajasthan/South UP Region

All the destinations were under radius of 1500 km from Delhi. This provided for a feasible yet reliable area of analysis. If required the model can be extended to other regions of India in a similar fashion.

OUR ZONES OF INTEREST

3. FEATURES

The next step was to come up with features to be used for delay prediction. Feature extraction is the most important part of any machine learning task and so is the case with us. To build effective train delay predictor, our aim is to try to model attributes like quality of train, the duration of its journey, area of its journey etc.

At present, our model is using the following set of features:

a) Type of Train: It has often been noticed that certain train are given preference over other when it comes to passing over congested network. Thus, we have divided the train into two categories. Type-0 train that include the best category trains of IRN like Rajdhani, Shatabdi etc. Type-1 includes the normal trains like express, superfast etc.

b) Travelling Distance: The trains travelling over long distance are often more vulnerable to delay. Thus we have also taken into account the distance that a train travels. It has been classified into two categories - Category-0: short-distance, less than 500km, and Category-1: long-distance, more than 500km.

c) Region of Journey: Certain regions experience more railway traffic than others. For instance in the UP/Bihar region we have extensive cases of redundant chain-pulling, frequent track disorder, high concentration of passenger trains and the heavy volume of traffic. The regions are same as discussed above, and each of them have been assigned a label from 1 to 3.

d) Day of Week: We are expecting certain days of week to be more congested than others. For example, the weekends are expected to experience more rush than other days of week. The days are assigned label starting from Mon:1, Tue:2, …., Sun:7.

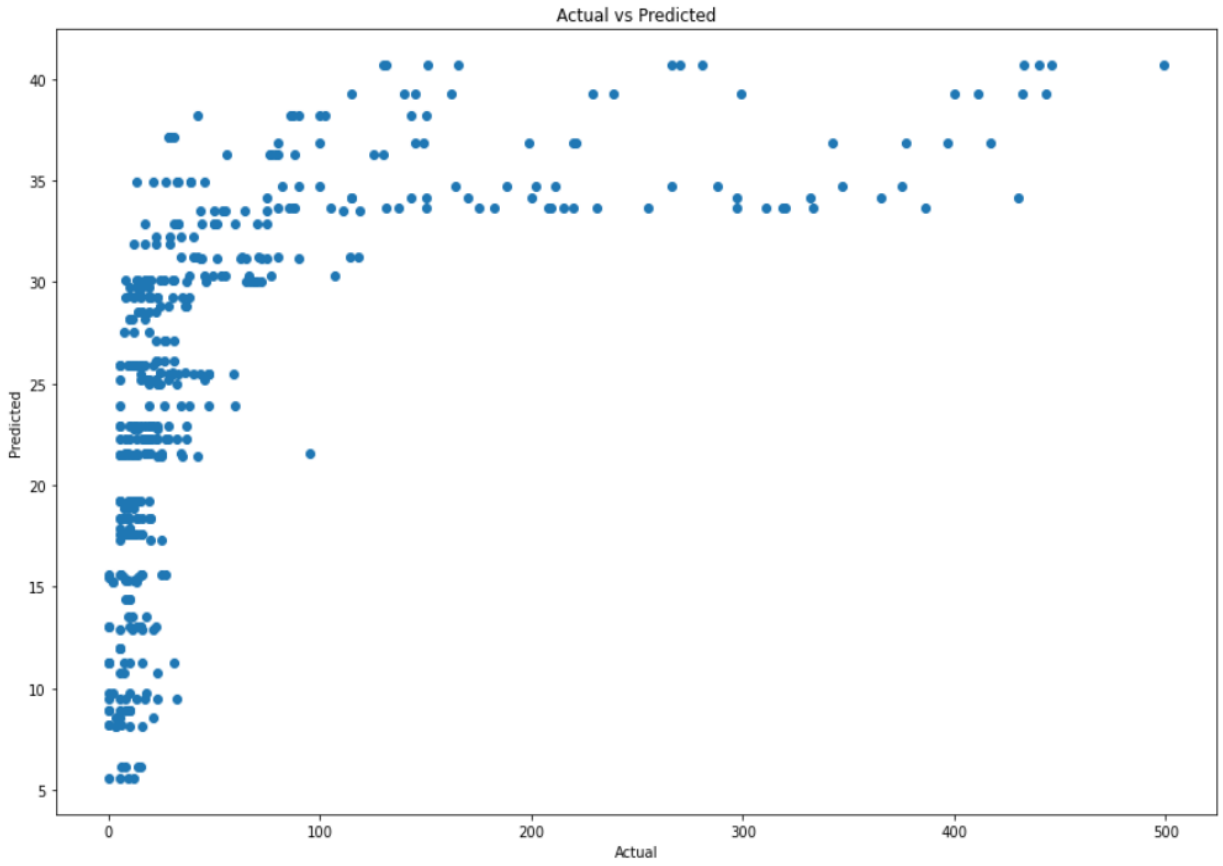
4. MODELS WE LOOKED UPON :

i) SUPPORT VECTOR REGRESSION

We performed Support Vector Regression over the data. We trained our model and visualised the predicted values to actual values.

We analysed for the problems which are causing this. One of the reasons we thought could be causing this is the variety and variance of parameters, and our inability to capture them in this limited time. Even if we reduce the feature space to make it more optimal we could not get even near to the desired results as the number of data

points we had could not be enough to train it completely (it would take about tens of millions of data point for just one route), and it would make the calculations very slow.
Reducing the number of feature space also had this disadvantage of being non-consistent with our original theme of giving at least some knowledge about train late pattern. Narrowing down the problem to, say, one station- one train- one day etc. will not result in generalized view of the problem.



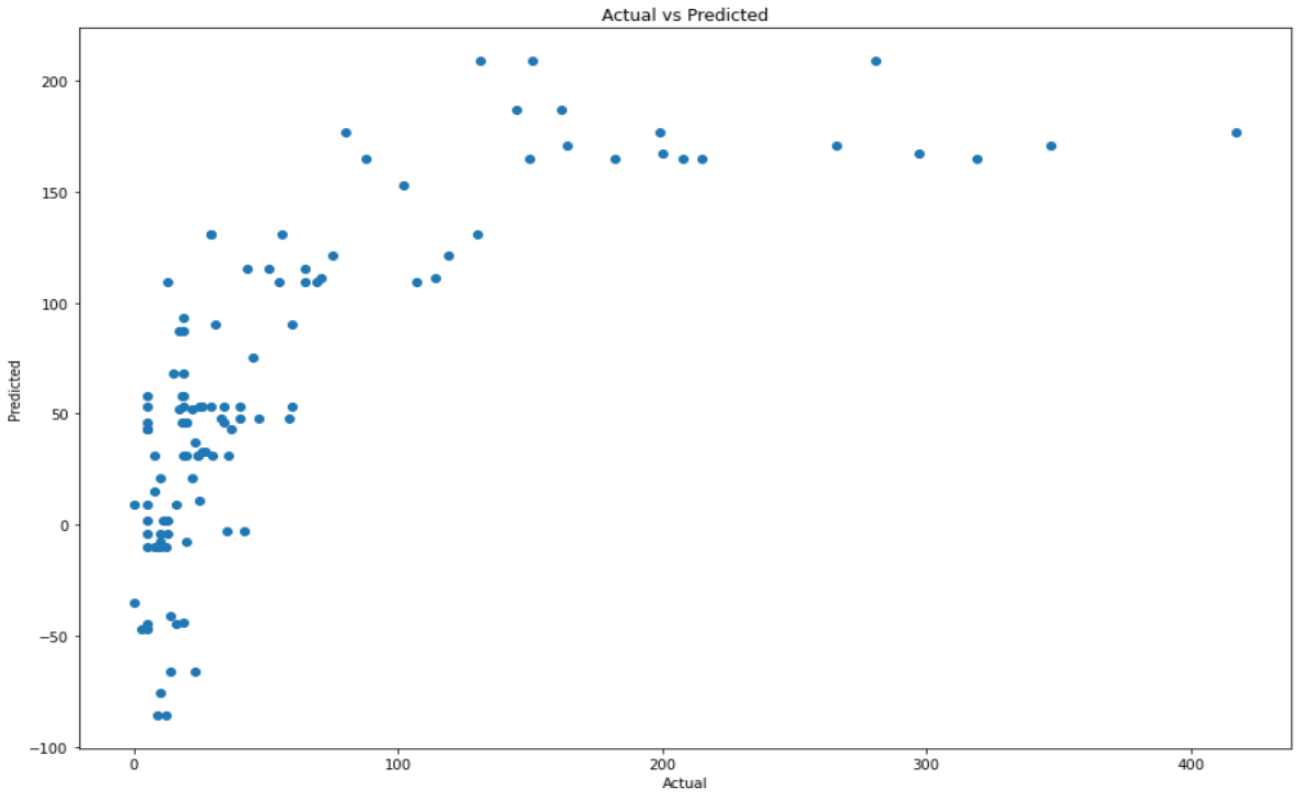

```
In [163]: print("SVR R^2 Score: %.2f"
          % r2_score(y, lst))
```

```
SVR R^2 Score: -0.04
```

The Support Vector Regression (SVR) uses the same ideas as the SVM for classification, with a few small differences. For starters, because output is a real number, it becomes incredibly difficult to forecast the information at hand, which has an infinite number of possibilities. **This model is not considered due to negative r2 Score.**

ii) MULTIPLE LINEAR REGRESSION:

We also used Multiple Linear Regression for predicting results. We used Pandas, Numpy, Matplotlib, Pickle and Sklearn. We thought the result would be good for Multiple Regression, but it underperformed due to low accuracy.



We analysed for the problems which are causing this. One of the reasons we thought could be causing this is the variety and variance of parameters, and our inability to capture them in this limited time. Even if we reduce the feature space to make it more optimal we could not get even near to the desired results as the number of data points we had could not be enough to train it completely .

```
In [182]: print('Train Score: ', regressor.score(X_train, y_train))
          print('Test Score: ', regressor.score(X_test, y_test))

          Train Score:  0.5156826967583082
          Test Score:  0.5188893575153475
```

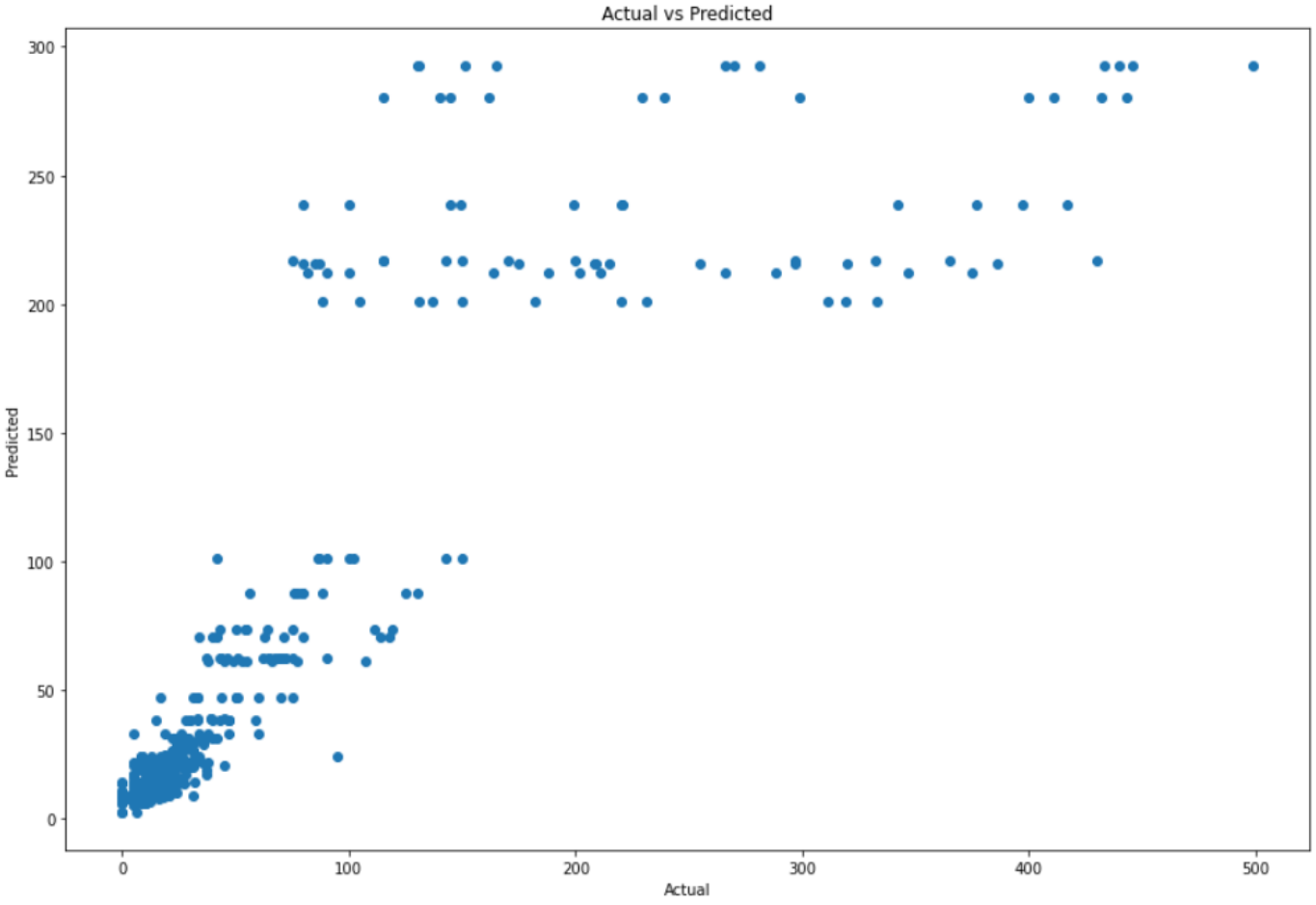
The above score tells that our model is 51% accurate with the training dataset and 53% accurate with the test dataset.

Reducing the number of feature space also had this disadvantage of being non-consistent with our original theme of giving at least some knowledge about train late pattern. Narrowing down the problem to, say, one station- one train- one day etc. will not result in generalized view of the problem.

iii) RANDOM FOREST

A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as **bagging**. The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees. Random Forest has multiple decision trees as base learning models. We randomly perform row sampling and feature sampling from the dataset forming sample datasets for every model. This part is called Bootstrap.

Prediction based on the trees is more accurate because it takes into account many predictions. This is because of the average value used. These algorithms are more stable because any changes in dataset can impact one tree but not the forest of trees.



In our model too , it gave best results compared to rest of the model we trained our data from. It uses the ensemble technique to predict values for the given data. **This model has the highest accuracy of about 76%.**

5. RESULT ANALYSIS

We used 5-fold cross validation for selecting the final model. SVM regression did not turn out well as it did not converge for squared error on training and the test data set. The following table summarise the accuracy of different model

MODEL	ACCURACY
SUPPORT VECTOR	Worst fit dala model with negative r2 Score.
MULTIPLE LINEAR	Gave accuracy of about 53% on the test set.
RANDOM FOREST	Gave the accuracy of 76%.

As can be seen from the above table, SVM overfits the data and performs poorly on test set. Multiple Linear Regression also didn't performed as expected. Random forest performs well in terms of accuracy compared to other models.

6. FEW SCREENSHOTS:

In [149]:
```
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import pickle
```

In [150]:
```
dataset = pd.read_csv('train_data.csv')
dataset.head()
```

Out[150]:

	TrainNo	TrainName	Type	Distance	Region	Day	DelayInMin
0	12004	Lucknow Swarn Shatabdi	0	0	1	1	22
1	12004	Lucknow Swarn Shatabdi	0	0	1	2	36
2	12004	Lucknow Swarn Shatabdi	0	0	1	3	32
3	12004	Lucknow Swarn Shatabdi	0	0	1	4	26
4	12004	Lucknow Swarn Shatabdi	0	0	1	5	37

Dataset

In [151]:
```
X = dataset.iloc[:, 2:6].values
y = dataset.iloc[:, -1].values
```

1. Support Vector Regressor Prediction :

In [162]:
```
y_pred_df = pd.DataFrame({'Actual Value':y,'Predicted Values': lst, 'Difference':y-lst})
y_pred_df[10:20]
```

Out[162]:

	Actual Value	Predicted Values	Difference
10	23	26.100035	-3.100035
11	28	28.826716	-0.826716
12	34	32.196626	1.803374
13	39	34.920971	4.079029
14	31	27.099826	3.900174
15	24	25.509848	-1.509848
16	19	24.956220	-5.956220
17	22	26.100035	-4.100035
18	24	28.826716	-4.826716

2. Multiple Linear Regressor Prediction :

In [179]:
```
y_pred_df = pd.DataFrame({'Actual Value':y_test,'Predicted Values': y_pred, 'Difference':y_test-y_pred})
y_pred_df[10:20]
```

Out[179]:

	Actual Value	Predicted Values	Difference
10	19	31.0	-12.0
11	164	171.0	-7.0
12	19	68.0	-49.0
13	40	48.0	-8.0
14	266	171.0	95.0
15	27	33.0	-6.0
16	29	131.0	-102.0
17	36	31.0	5.0
18	23	37.0	-14.0

3. Random Forest Regressor Prediction :

In [144]:
```
y_pred_df = pd.DataFrame({'Actual Value':y,'Predicted Values': lst, 'Difference':y-lst})
y_pred_df[10:20]
```

Out[144]:

	Actual Value	Predicted Values	Difference
10	23	25.572440	-2.572440
11	28	31.176393	-3.176393
12	34	30.815197	3.184803
13	39	38.803089	0.196911
14	31	26.495587	4.504413
15	24	28.760119	-4.760119
16	19	24.923921	-5.923921
17	22	25.572440	-3.572440
18	24	31.176393	-7.176393

7. REFERENCES

1. https://www.researchgate.net/publication/318160378_Arrival_Time_Prediction_and_Train_Tracking_Analysis

2. https://www.udemy.com/course/machinelearning/learn/lecture/19041246?start=0#overview
3. https://www.youtube.com/watch?v=mrExsjcvF4o&list=PLZoTAELRMXVOAvUbePX1lTdxQR8EY35Z1&index=2

4. www.runningstatus.in