TAXI FARE PREDICTION

ML MODELING

Foundation of Machine learning (CS 725)
Final Project

TEAM CONTRIBUTION

Prabhat Patel (24M2005) : Back propogation and Model Training
Anupam (24M2015) : Dataset preparation, Forward propagation
Anuj Yadav(24M2012) : Graph plotting and Weight Initialisation

Project Links

Dataset link : https://drive.google.com/file/d/1pUblDXN2XnR-tnbedxr-19ZGSHKjviCN/view?usp=drive_link

https://www.kaggle.com/datasets/diishasiing/revenue-for-cab-drivers?resource=download

INTRODUCTION

This project aims to prepare a Machine learning Model for predicting taxi fare.

Taxi fare prediction is a machine learning application that aims to estimate the cost of a taxi ride based on various factors like distance, time, and location.

This project involves building predictive models using historical taxi ride data, leveraging machine learning algorithms like Back Propagation, linear regression and random forest .

The goal is to create a system that minimizes prediction errors while being scalable and efficient for real-world deployment.

As a benefit it improves customer experience with upfront pricing and reduces disputes.

Models Used :

ANN model (main model for testing and training)

Linear regression model (secondary model for testing)

Random forest Model(secondary model for testing)

Model Structure

1. ANN model

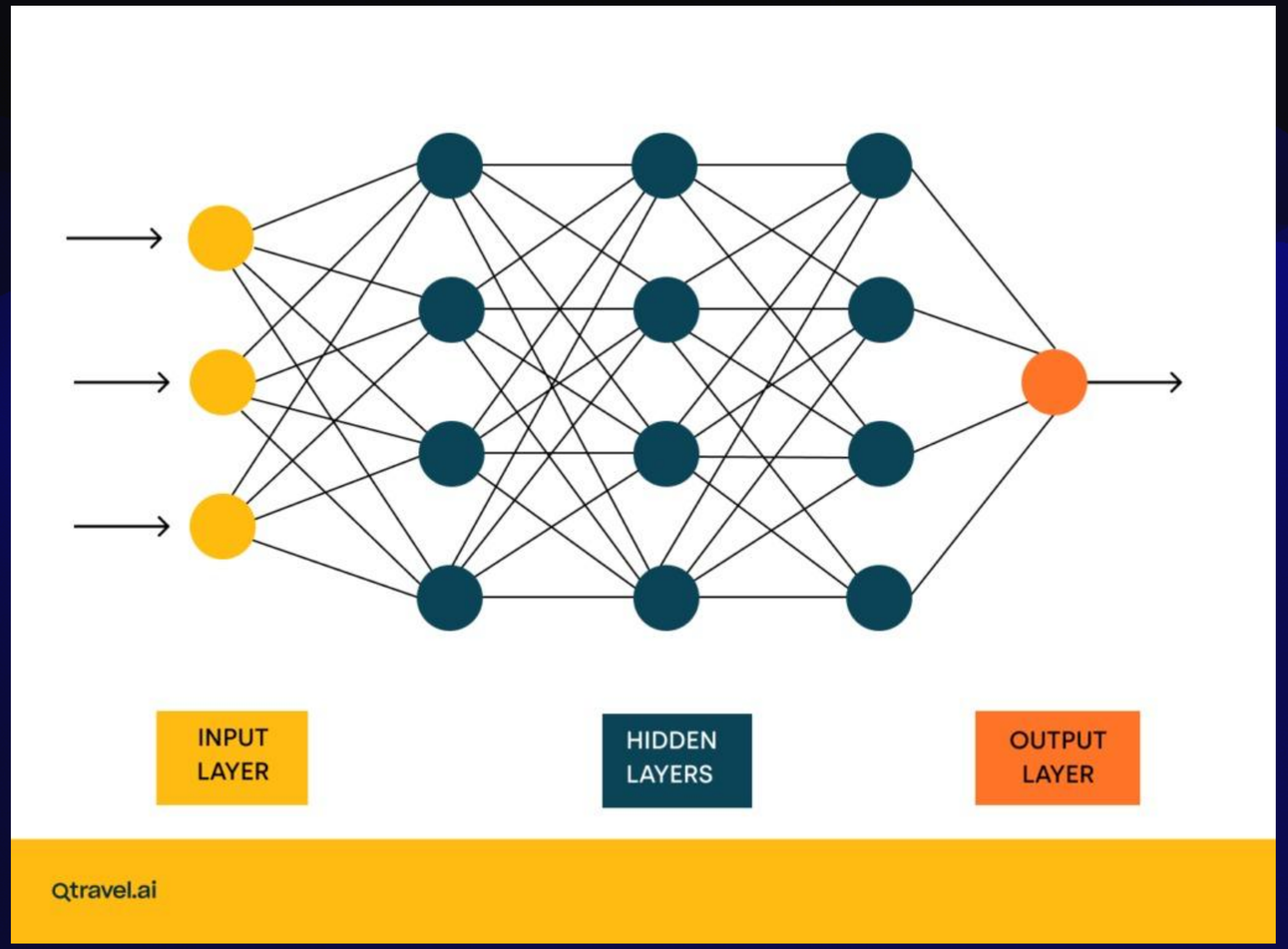
There are 1 input layer, 3 hidden layers and a output layer.

Hidden Layers consisting :

- 64 neurons(H1)
- 32 neurons(H2)
- 21 neurons(h3)

Input layers consists of 8 features such as

Passenger count, week day, distance.



For getting optimised weights parameters we used Adam optimizer.

Parameters :

- Momentum parameter (beta 1 = 0.85)
- adaptive learning rate parameter (beta 2 =0.99)

About output :

It consists single neuron with linear activation function.

Loss function : mean squared loss(MSE), root mean squared loss(RSME)

2. Linear regression

Data collection : Used a dataset for collecting data from Kaggle

Features :

- Date and time
- Pickup/ Drop off location
- passenger count

Error evaluated :

Root mean squared Error

Mean Absolute error

2. Random Forest

Parameters in Random forest :

1. N_estimators: It depicts the number of decision tress in random forest
2. n_jobs: It depicts the number of processor working on the model training.
3. Verbose : It is used for tracing the progress of the model .

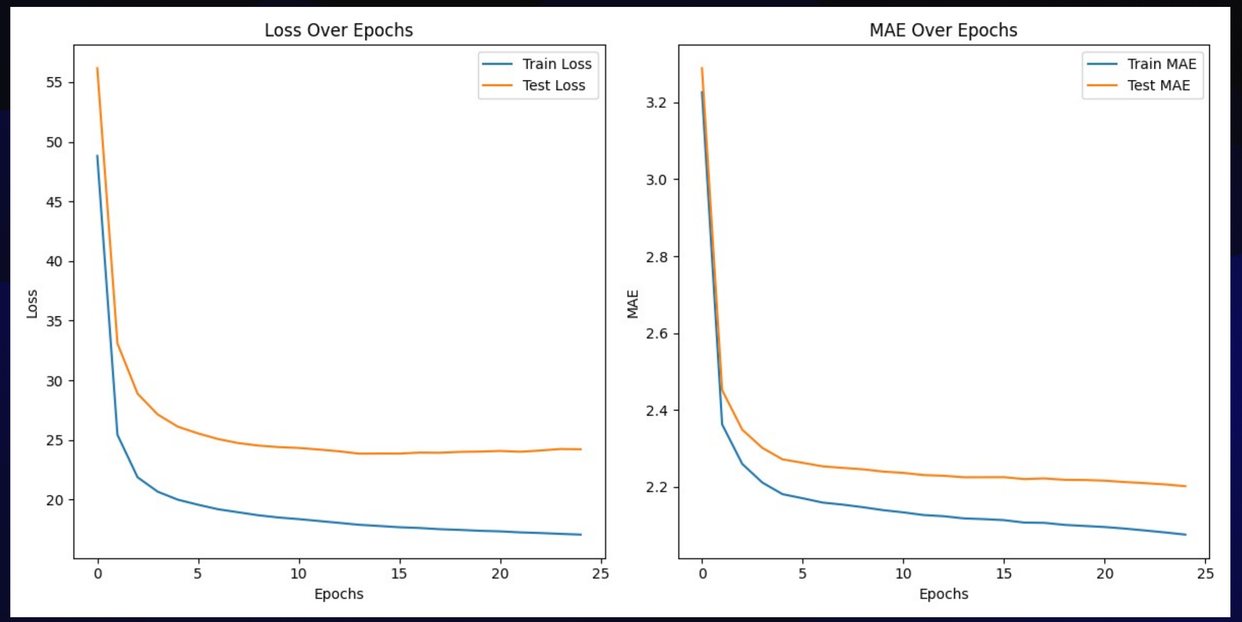
Error calculated :

-Mean Absolute Error (MAE)

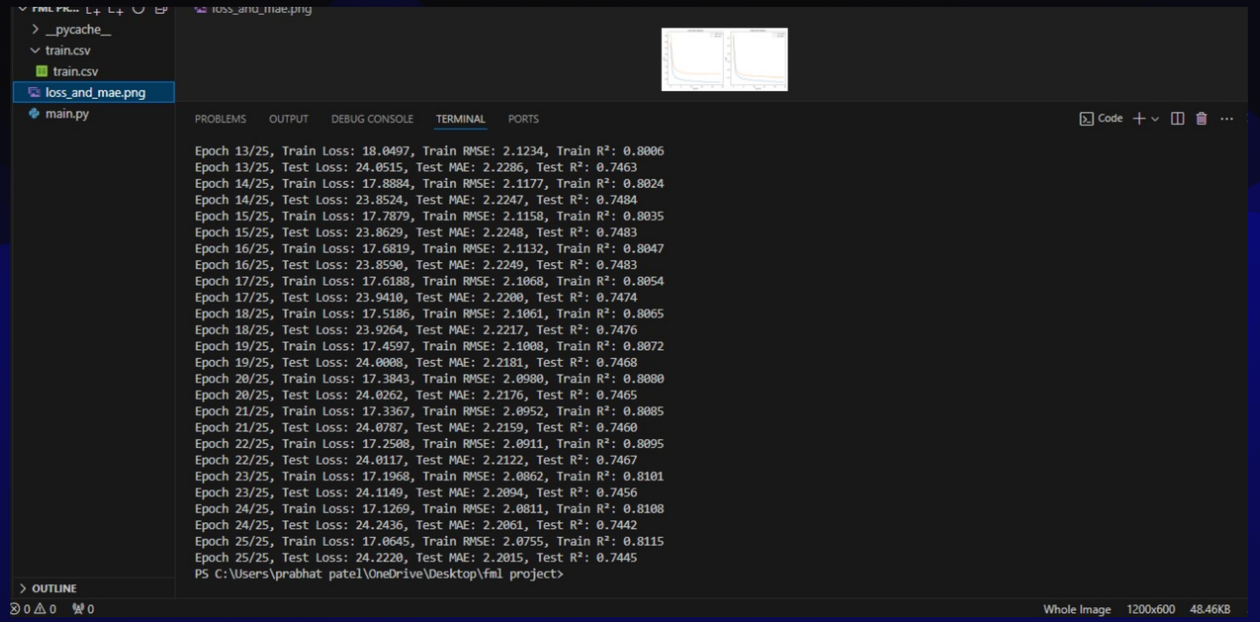
-Root Mean Squared Error(RMAE)

Results

ANN model



Loss and RSME graphs



Error Calculation though Epochs

Linear regression

Mean Absolute Error : 2.066

RMSE: 5.598

```
Mean Absolute Error (MAE): 2.066267466753622
Root Mean Squared Error (RMSE): 5.698491391298586
Average Fare Amount: 12.582708397480648
RMSE as a Percentage of Average Fare: 45.29%
```

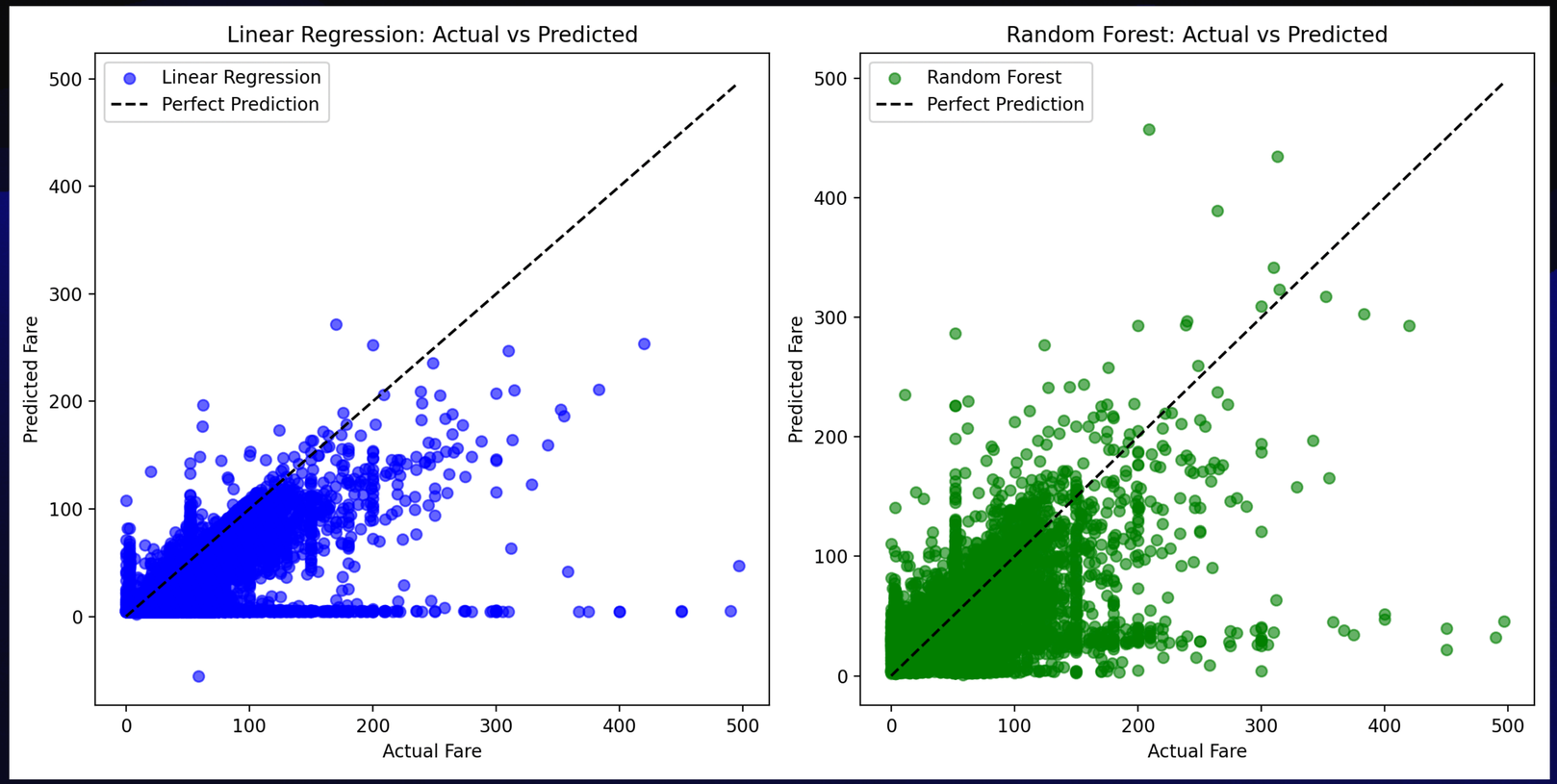
Random forest

Mean Absolute Error : 1.781

RMSE: 5.158

```
[Parallel(n_jobs=-1)]: Using backend ThreadingBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   6 out of  10 | elapsed:   20.9s remaining:   13.9s
[Parallel(n_jobs=-1)]: Done  10 out of  10 | elapsed:   29.4s finished
3
[Parallel(n_jobs=8)]: Using backend ThreadingBackend with 8 concurrent workers.
[Parallel(n_jobs=8)]: Done   6 out of  10 | elapsed:    1.2s remaining:    0.8s
[Parallel(n_jobs=8)]: Done  10 out of  10 | elapsed:    1.7s finished
Random Forest - Mean Absolute Error (MAE): 1.7817573455140276
Random Forest - Root Mean Squared Error (RMSE): 5.158362367358236
```

Random forest V/S linear regression



Conclusion

Model Performance

If the training loss decreases steadily and the gap between training and testing loss is minimal, the model generalizes well.

A high R^2 score close to 1 for both training and testing sets suggests a good fit.

The RMSE provides an interpretable measure of the prediction error in the same units as the target variable.

Next Steps:

Experiment with hyperparameter tuning (e.g., learning rate, number of neurons in hidden layers).

Consider adding dropout layers or L2 regularization to mitigate overfitting.

Use additional feature engineering or external data sources for better predictions.

The final performance metrics and loss plots would give a concrete idea of how the model is performing, which can guide further iterations to improve the results.