UNIVERSITY OF PETROLEUM & ENERGY STUDIES
DEHRADUN

Taxi Fare Prediction

Applications of Machine Learning in Industries

Submitted by

Name	SAP ID	Specialization
Simar Katyal	500091861	AIML (Hons.)
Devansh Aggarwal	500092974	AIML (Hons.)
Garvit Shadra	500093671	AIML (Hons.)

Artificial Intelligence Cluster

BACHELORS OF TECHNOLOGY, COMPUTER SCIENCE & ENGINEERING

Submitted To :

Dr. Rohit Srivastava

Course Faculty

Project Title : Taxi Fare Prediction

Project Understanding Report

1.Introduction

1.1 Prologue (Literature Review)

Paper Title	Input Parameters	Methodology	Conclusion	Future Outcomes
Fare Prediction using Supervised Learning[1]	• Historical NYC cab fares dataset • Trip duration • Trip distance • Specific pickup and drop-off locations	• Analysis: XGBoost Random Forest • Evaluation : MSE ,RMSE • Recognizing challenges	Presence of non-random residuals suggests potential for improvement with additional factors . Certain locations disproportionately affect fares, potentially due to demand or congestion.	• Development of user based-tools • Integration of environmental factors
Prediction of Dynamic Price of Ride-on-demand services using Linear Regression[2]	• Pick up and drop location • Distance travelled • Time of day • Weather Condition	• Moving average • Seasonal Naive • STL decomposition • Exponential Smoothing	Can effectively predict taxi fares. Time- series forecasting for taxi demand prediction.	• Rapidly changing traffic conditions • Variability in demand patterns across city areas
Predictive Analysis of Taxi Fare using Machine Learning [3]	• Date & Time • Pickup & Dropoff Locations • Distance • Passenger Count • Base Fare	• Supervised Learning (Regression) • Random Forest Model	Machine Learning predicts taxi fares accurately. It also gives relatable outputs.	• Improved fare estimations for passengers & companies.

Taxi Demand Prediction using ML-RNN [4]	• Historical taxi demand • Pairwise correlation coefficients between different taxi zones • Weather • Temperature	• Pairwise clustering algorithm • Multi-Level RNN • RMSE & MAPE	ML RNN model is superior to other models in terms of Robustness due to the small variance	• Determining the optimal number of clusters • Specific models focusing on heterogeneity of different regions
Taxi services using real-time visualization [5]	• Time • Passenger Count • Travel Distance	• Real-time data stream processing (Apache Flink)	Model predicts traffic & suggests routes in real-time. Focuses on computational efficiency for mobile devices.	• Dynamic route planning based on real-time conditions.
Real-time prediction of taxi demands using random radio networks [6]	• Time of Day • Weather • Taxi Drop-Off Volume	• LSTM Recurrent Neural Network (RNN) with Mixed Density Network (MDN)	LSTM-MDN predicts taxi demand with 83% city-level accuracy.	• Improved prediction with additional data (business locations, taxi configurations).
Hybrid Deep NN combined with Travel Time [7]	• Taxi ID • Time • Latitude • Longitude	• Convolutiona l LSTM combined with travel time • MAE & MSE	Hybrid model outperforms other time series prediction models and deep neural network models	• Optimizing functional layer combinations, such as SeparableCon v2D.
Trip Duration Prediction using MLP & XGBoost [8]	• Real-time data • Pickup and drop-off coordinates • Distance • Start time • Number of passengers • Rate code	• Mini-batch K-means clustering algorithm • Multi-Layer Perceptron • XGBoost • MAE	XGBoost model outperformed the Multi-Layer Perceptron model, with an MAE of 0.25 and an MSE of 0.13	• Dynamic Pricing Models • Scalability • User Experience Enhancement

Fare & Duration Prediction [9]	• Pickup and Dropoff coordinates • Trip distance • Start time • Number of passengers • Rate code	• Linear Regression • Lasso Regression • Random Forest • MAE & MSE	Random Forest Model outperformed Linear Regression & Lasso models	• Exploring parameters for better prediction • Modelling the effect of location & traffic patterns
Realistic prediction of taxi fares using machine learning [10]	• Ride and share • Number of passengers • Date and Time	• Random Forest • Linear Regression	Random Forest outperforms Linear Regression with a lower MAE and higher accuracy Accurate predictions	• Improving prediction accuracy • Development of real-time fare prediction tools for ride-hailing applications.

2. Problem Statement

- Nwadays, flagging down a taxi frequently leads to travelers battling with hazy pricing leading to frustration, disappointment and overpaying. The drivers and the passengers have to face the pricing chaos because the models used do not account for the price modelling, unpredictable demand, and lack of transparency.
- Moreover, the user interface of certain applications is complex to understand and operate specially for the elderly, differently abled & non-tech people. It lacks scalability, and has poor integration leading to different prices displayed to the driver and the passenger.
- This project aims to bridge these gaps by developing a taxi fare prediction model by combining artificial intelligence techniques like RNNs with the machine learning approaches like Linear Regression, Logistic Regression, Random Forest, Support Vector Machine and KNN.
- We propose to utilize python frameworks like Django or Flask for frontend and Streamlit or Tkinter for backend. Thus, the proposed model will provide increased transparency, improved user experience, and more accurate fare estimates, hence beneficial for both the drivers and customers.

3. Objectives

After the conclusive literature review performed, the targets of this project are listed as follows -

1. To predict the taxi fare considering advanced factors like waiting charge, geographical conditions, climatic conditions and traffic conditions.
2. To implement, compare and conclude the best Machine Learning approach like Logistic Regression, Random Forest, Gradient Boosting and Recurrent Neural Network for predicting the accurate taxi fare.
3. To design a simple and user-friendly interface that is convenient to users of all age groups and from all professional backgrounds.

4. Explanation

Here, we have discussed the approach and models that we will be using to build the project 'Taxi Fare Prediction'.

4.1 Data Acquisition:

An attempt to look for various kinds of datasets that include the important factors to be considered for predicting taxi fare have been explored. So far, the best datasets available are NYC Taxi & Limousine Commission (TLC) trip data.

Important Points to Consider while selecting the suitable dataset -

1. **Relevant Data** - The dataset should be relevant to handle the problem at hand. It should contain the essential features which are important factors to forecast the accurate taxi fare.
2. **Data Quality** - The quality of the data affects the reliability & accuracy of the model. It's crucial to ensure the quality of the data by checking for inconsistencies, outliers, metadata and documentation about the dataset. Data profiling tools like DataWrangler may also be used to generate the summary & graphs of the dataset that provide insights into data's quality & characteristics.
3. **Data Accessibility** - The dataset should be easily accessible & readily available. Datasets from reputed sources like government websites, official reports etc. are the best choice.
4. **Data Size** - Large datasets shall be preferred as they lead to better models.

4.2 Data Understanding :

It is crucial to explore the gathered data to understand the characteristics, patterns, and relationships amongst the several features. This is known as Exploratory Data Analysis (EDA). Some of the commonly used data understanding steps include -

1. **Data Summary** - Explore the dataset by computing the summary statistics like mean, median, standard deviation, counts, unique values etc. to understand the central tendency, dispersion and variability of the dataset.

2. **Visualize the Data Distribution** - Visualize the data distributions such as, scatter plots, bar graphs, heatmaps, box plots etc. that help to understand the shape, spread & skewness of the data.
3. **Handling Outliers** - Anomalies and outliers can be detected by visualizing the data through box plot, scatter plot etc. that will help us explore the ways to handle these.

4.3. Data Pre Processing :

It is a crucial step in the Data Analysis Pipeline that involves cleaning, transforming & preparing the data for further analysis or modelling. Key steps in Data Preprocessing are-

1. **Data Cleaning** - Handling the missing values, outliers and duplicates and correcting errors.
2. **Data Transformation** - Data Transformation techniques like normalization, standardization, or logarithmic transformation are incorporated to improve the distribution of variables. This step helps to prevent biases in the training model and improves the performance of the models due to improved data quality and consistency.
3. **Feature Engineering -** This step involves selecting the features from the dataset that are crucial factors in predicting the taxi price. Feature Engineering also involves the creation of new features using the existing features.
4. **Dimensionality Reduction** - Reduce the dimensionality of the dataset by selecting a subset of important features by applying Dimensionality Reduction techniques like Principal Component Analysis (PCA).

4.4. Prediction using various algorithms:

After performing the comprehensive literature review, in this project the following models will be implemented and their accuracy measures will be compared to estimate the best model for Taxi Fare Prediction.

4.4.1 Linear Regression - Linear Regression is a Supervised Machine Learning Algorithm that is trained on the labelled dataset. It is used to compute the relationship between the dependent & the independent variables.

Simple Linear Regression - It is used to predict the independent variable based on one independent variable.

$$\mathbf{y = b_0 + b_1 x} \qquad \mathbf{I}$$

Here, y is the dependent variable, x is the independent variable, b_0 is the intercept & $b_1$is the slope.

Multiple Linear Regression - It is used to predict the dependent variable based on two or more independent variables.

$$\mathbf{y = b_0 + b_1x_1 + b_2x_2 + \ldots\ldots\ldots + b_nx_n} \qquad \mathbf{II}$$

Here, y is the dependent variable, $\{x_1, x_2, x_3, \ldots\ldots\ldots x_n\}$ is the set of independent variables, b_0 is the intercept & $b_1$is the slope.

Algorithm for Linear Regression is as follows -

Step 1 : Import the necessary libraries (numpy, pandas, matplotlib)

Step 2 : Split the dataset into training dataset (80%) & testing dataset (20%)

Step 3 : Initializing the parameters (here, y = fare price & x = pickup location, dropoff location, traffic conditions, weather etc.)

Step 4 : Building the Linear Regression Model

Step 5 : Testing the LR Model on the Testing dataset

Step 6 : Evaluating the LR Model (MAE, MSE, RMSE, R Squared)

Step 7 : Prediction on the new dataset

4.4.2 Ridge & Lasso Regression - Two of the most common Regularization techniques are Lasso (L1) & Ridge (L2) Regression. These techniques help to fix the issue of Overfitting, which arises when the model starts fitting so well on the training data that it starts learning the relationships between the features & target variables. This leads to low generalization ability of the model, i.e, the model does not predict well on the new, unseen data.

Lasso Regression:

Lasso regression adds a penalty term to the Error term, which is the absolute sum of the coefficients.

$$\mathbf{L1 = MSE + \lambda * ||\beta||} \qquad \mathbf{III}$$

Here, MSE is the error term, λ is the hyperparameter that controls the trade-off between reducing the MSE & shrinking the coefficients.

Step 1: Import necessary libraries Lasso Regression from scikit-learn and numpy for mathematical calculations.

Step 2: Define a function to calculate Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared. Use numpy for efficient mathematical operations.

Step 3: Initialize a Lasso Regression model with a specified alpha value (regularization strength).

Step 4: Train the Lasso Regression model. Fit the Lasso Regression model using the training data.

Step 5: Evaluate the training dataset. Calculate the model's score on the training dataset. Make predictions on the training dataset and calculate evaluation metrics (MAE, MSE, R-squared) for the training dataset.

Step 6: Make predictions on the testing dataset. Calculate evaluation metrics (MAE, MSE, R-squared) for the testing dataset.

Ridge Regression:

Ridge regression adds a penalty term to the Error term, which is the squared sum of the coefficients.

$$\mathbf{L2 = MSE + \lambda * ||\beta||^2} \qquad \mathbf{IV}$$

Similarly, here, MSE is the error term, λ is the hyperparameter that controls the trade-off between reducing the MSE & shrinking the coefficients.

Step 1: Import the required libraries: sklearn.linear_model.Ridge

Step 2: Initialize the Ridge Regression model with a specified alpha value (penalty parameter).

Step 3: Train the Ridge Regression model using the training data (x_train, y_train).

Step 4: Evaluate the model performance on the training dataset:

a. Compute the R-squared score of the model on the training data.
b. Make predictions on the training data.
c. Calculate evaluation metrics (e.g., mean squared error, mean absolute error) using the actual and predicted values.

Step 5: Print the evaluation metrics for the training dataset.

Step 6: Make predictions on the testing dataset using the trained model.

Step 7: Evaluate the model performance on the testing dataset: Calculate evaluation metrics (e.g., mean squared error, mean absolute error) using the actual and predicted values.

Step 8: Print the evaluation metrics for the testing dataset.

4.4.3 Support Vector Machine

Support Vector Machines (SVMs) is a Supervised Machine Learning Model commonly used for classification, regression & outlier detection tasks. Objective is to find the best hyperplane in an N-dimensional space that can separate the data points in different classes in the feature space.

$$f(x) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b\right) \quad \text{V}$$

Here, f(x) is the decision function predicting the class label of input x, n is the number of support vectors, α_i are the coefficients obtained during training, y_i are the coefficients obtained during training, K (x_i, x) is the Kernel function which computes similarity between input vectors x_i and x, & b is the bias term.

Step 1 : Import the necessary libraries (numpy, pandas, StandardScaler,)

Step 2 : Split the dataset into training dataset (80%) & testing dataset (20%)

Step 3 : Initializing the parameters (here, y = fare price & x = pickup location, dropoff location, traffic conditions, weather etc.)

Step 4 : Building the Linear Regression Model

Step 5 : Testing the LR Model on the Testing dataset

Step 6 : Evaluating the LR Model (MAE, MSE, RMSE, R Squared)

Step 7 : Prediction on the new dataset

4.4.4 K-Nearest Neighbors - KNN is a Supervised Machine Learning algorithm used for regression and classification tasks. The value of k defines the number of nearest neighbors in the algorithm.

Mathematical Formula for KNN for Classification tasks :

$$\hat{y}_q = \text{argmax}_{yj} \sum_{i=1}^{K} I(y_i = y_j) \quad \text{VI}$$

Mathematical Formula for KNN for Regression tasks :

$$y^{\wedge}_q = 1/K \left(\sum_{i=1}^{K} y_i \right) \quad \text{VII}$$

Here, I is the indicator function that returns the value 1 if true, else 0; y_j iterates over all class labels.

Step 1: Import KNeighborsRegressor for KNN Regression from scikit-learn and StandardScaler for feature scaling.

Step 2: Scale the features of both the training and testing datasets.

Step 3: Initialize a KNeighborsRegressor with a specified number of neighbors. Fit the KNN Regression model using the scaled training data.

Step 4: Calculate the model's score on the scaled training dataset. Make predictions on the scaled training dataset. Calculate evaluation metrics (MAE, MSE, R-squared) for the training dataset.

Step 5: Calculate evaluation metrics (MAE, MSE, R-squared) for the testing dataset.

4.4.5 Decision Trees - Decision Tree is a Supervised Machine Learning algorithm that builds a tree-like structure where internal node denotes a test on an attribute, & each branch represents outcome of the test.

$$f(x) = \text{sign} \left(w^T x + b \right) \quad \text{VIII}$$

Step 1: Import DecisionTreeRegressor for Decision Tree Regression from scikit-learn and GridSearchCV for hyperparameter tuning.

Step 2: Define lists of hyperparameters to search over, such as max_depth, min_samples_split, and min_samples_leaf.

Step 3: Create a parameter grid dictionary containing the hyperparameters to search over.

Step 4: Create a DecisionTreeRegressor object.

Step 5: Initialize GridSearchCV with the DecisionTreeRegressor, parameter grid, cross-validation folds (cv), verbosity level (verbose), and scoring metric (scoring). Fit the GridSearchCV object on the training data to find the best combination of hyperparameters.

Step 6: Get the best estimator and best score from the grid search results.

Step 7: Use the best estimator to make predictions on both the training and testing datasets.

Step 8: Print the evaluation metrics for both the training and testing datasets.

4.4.6 Random Forest : Random Forest Algorithm world by creating multiple number of decision trees by selecting data & features a random subset of the dataset. further, the algorithm aggregates the results of all the decision trees constructed using voting or averaging methods.

Step 1 : Import the necessary libraries (numpy, pandas, matplotlib)

Step 2 : Select the model parameters (number of trees, maximum depth of tree, number of features)

Step 3 : Split the dataset into training dataset (80%) & testing dataset (20%)

Step 4 : Bootstrapping - In this step, copies of the training dataset are created, called bootstraps.

Step 5 : Constructing Decision Trees - For each bootstrap sample, a decision tree is created.

Step 5 : Steps 4 & 5 are repeated to create a large number of trees.

Step 6 : This Random Forest model can now be used for predicting the new data using Majority Vote or Averaging technique.

Step 7 : Evaluating the LR Model (MAE, MSE, RMSE, R Squared)

5. Evaluating Model Performance:

To determine how well each model performs, we employ metrics like:

1. **Mean Absolute Error (MAE) -** MAE calculates the average of the absolute differences between the actual values and the predicted values.

   $$\mathbf{MAE = 1/n\ \Sigma(y_i - \hat{y}_i)}$$

   Here, n - number of data points, y_i - actual value & $\hat{y}_i$ - predicted value.

2. **Mean Squared Error (MSE)** - MSE calculates the average of the squared differences between the actual values and the predicted values.

   $$\mathbf{MSE = 1/n\ \Sigma(y_i - \hat{y}_i)^2}$$

   Here, n - number of data points, y_i - actual value & $\hat{y}_i$ - predicted value.
3. **Root Mean Squared Error (RMSE)** - It calculates the square root of the Mean Squared Error.

   $$\mathbf{RMSE = [\ 1/n\ \Sigma(y_i - \hat{y}_i)^2\ ]^{½}}$$

   Here, n - number of data points, y_i - actual value & $\hat{y}_i$ - predicted value.
4. **R-squared -** R-squared metric, also known as the Coefficient of Determination, represents the proportion of variance explained by the model.

   $$\mathbf{R^2 = 1 - \Sigma(y_i - \hat{y}_i)^2 / \Sigma(y_i - \bar{y})^2}$$

   Here, y_i - actual value, $\hat{y}_i$ - predicted value & $\bar{y}$ - average of the actual y values

6. Key Factors :

The research also identifies the most crucial factors influencing fare prices. This knowledge can be valuable for both riders and taxi drivers. Some potential key factors might include:

- **pickup_datetime -** Analyzing pickup datetime to assess peak hours and potential surge pricing effects on NYC taxi fares
- **dropoff_datetime -** Evaluating dropoff datetime to understand demand patterns and their impact on fare rates.
- **passenger_count -** Considering passenger count to account for fare adjustments based on group size or vehicle type.
- **pickup_longitude -** Utilizing pickup longitude to gauge distance traveled and its influence on fare calculations.
- **pickup_latitude -** Incorporating pickup latitude to understand location-based pricing differentials within NYC.
- **dropoff_longitude -** Leveraging dropoff longitude to estimate travel distance and its effect on final fare amounts.
- **dropoff_latitude -** Incorporating dropoff latitude to assess destination-based fare variations across NYC.
- **trip_duration -** Analyzing trip duration to account for time-based fare adjustments such as waiting times or traffic delays

7. Empowering Users with a Prediction Tool:

The ultimate goal is to leverage these models to create a user-friendly tool. Imagine a mobile app or website that allows users to:

- Input their trip details (pickup and drop-off locations, estimated travel time)
- Utilize the trained models to receive a predicted fare estimate.

This empowers riders to plan their trips more

8. System Requirements

Hardware:

- **Processor**: Intel Core i5 or equivalent (or higher) for efficient language data processing.
- **Memory (RAM)**: Minimum 8 GB RAM is required for handling large datasets and computational tasks.
- **Internet Connectivity**: Required for downloading datasets, libraries, and resources, as well as for collaboration and accessing online documentation.

Software:

- **Programming Language**: Python
- **Development Environment**: Google Collab, or Visual Studio Code for code development and debugging.

Libraries:

- Scikit-learn offers various algorithms for regression.
- NumPy and pandas for data manipulation and analysis.
- Matplotlib or Seaborn for data visualization.

9. System Analysis

9.1 Motivation

9.1.1 Convenience for Passengers:

1. With reliable fare estimates, passengers can plan their journeys more effectively, considering budget constraints.
2. Predictable fares reduce uncertainty for passengers, leading to a smoother travel experience.

9.1.2 Optimized Earnings for Drivers:

1. Accurate fare predictions empower drivers with insights into potential earnings for different routes and times.

2. Drivers can strategically plan their operations to maximize their earnings, contributing to their financial stability.

9.1.3 Informed Decision-Making:

1. Both passengers and drivers can make informed decisions based on predicted fares and optimized routes.
2. Informed decision-making leads to more efficient use of resources and improved overall service quality.

9.1.4 User Experience Improvement:

1. The model enhances the overall user experience by providing clear, upfront pricing information.
2. Improved user experience fosters loyalty and positive word-of-mouth recommendations for taxi services.

9.1.5 Efficiency and Sustainability:

1. Optimized routes and fare predictions contribute to the efficiency of urban transportation systems.
2. By encouraging efficient resource utilization, the model promotes sustainability in urban mobility.

9.2 Proposed System

9.2.1 Data Collection:

1. Gather historical taxi trip data including features such as pickup and drop-off locations, distance travelled, duration of the trip and any additional relevant information.
2. Ensure the dataset is comprehensive, clean, and properly formatted for analysis.

9.2.2 Data Preprocessing:

1. Perform data cleaning to handle missing values, outliers, and inconsistencies.
2. Feature engineering: Extract useful features such as distance between pickup and drop-off points, time-based features (e.g., hour of the day), and categorical encoding (e.g., one-hot encoding for categorical variables like day of the week).
3. Normalize or scale numerical features as necessary to improve model performance.

9.2.3 User Interface:

1. Create a user friendly interface for passengers and drivers to access travel information and get price estimates.
2. Includes features such as interactive maps, competitive pricing and esti mated time of arrival enhance user experience.

9.2.4 Continuous improvement:

1. Write user instructions and monitor performance standards in producti on.
2. Acquire new products and retrain the model regularly to adapt to chang ing the model and improve accuracy over time.

9.2.5 Security and Privacy:

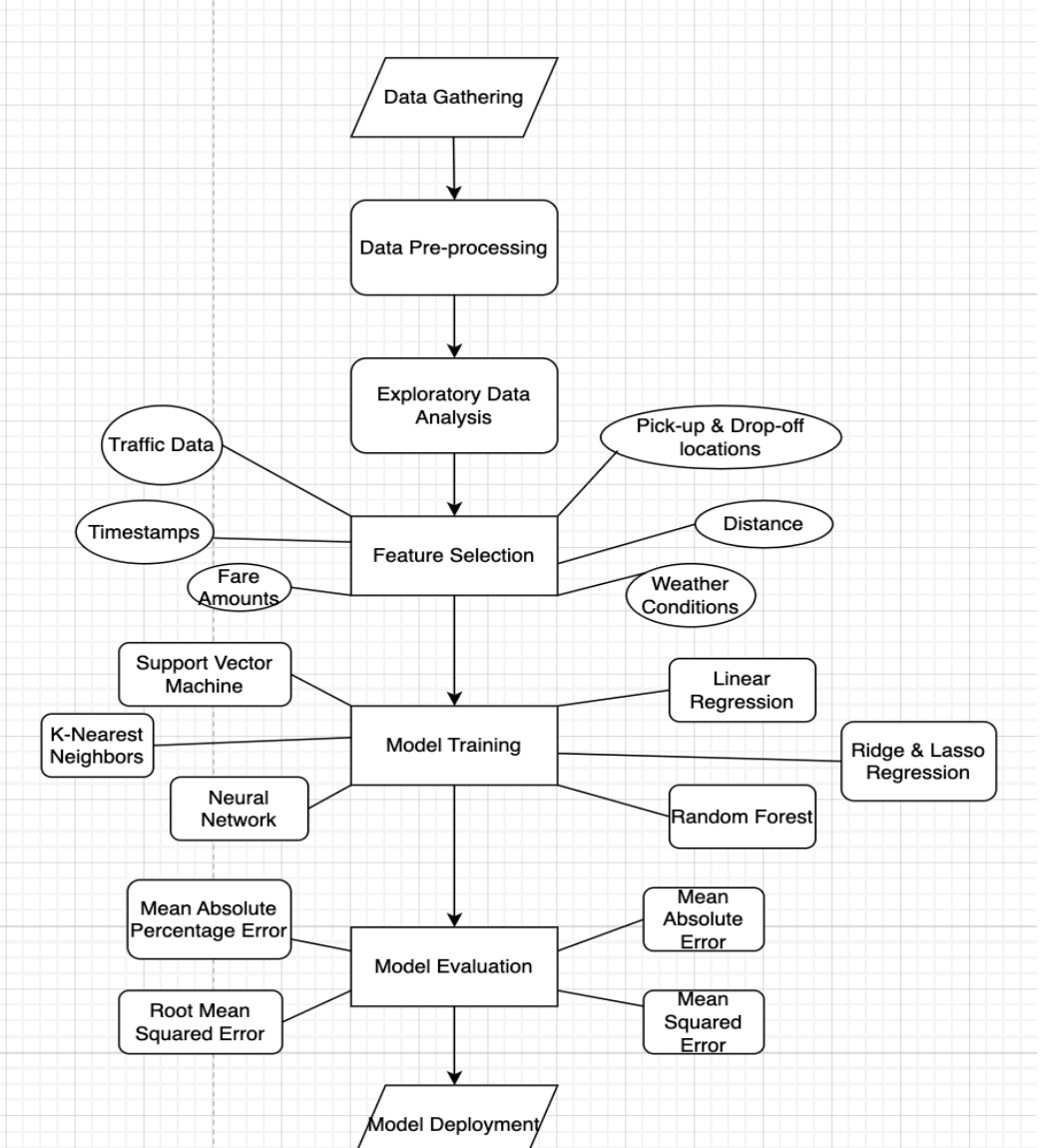
1. Ensure that effective security measures are in place to protect users' se nsitive information and prevent unauthorized access.
2. Ensure compliance with data privacy laws such as GDPR or CCPA to p rotect customer privacy.

9.2.6 Documentation and Maintenance:

1. Maintains information about system architecture, piping information, a nd model specifications.
2. Ensure ongoing maintenance including version control, bug tracking a nd troubleshooting.

10. Design

10.1 Flow Chart Diagram



10.12 Methodology

- **Data Collection:** Collect historical data on New York City taxi cabs and conditions to analyze the relationship between travel type, travel duration, travel time and fares.
- **Linear regression model**: Fit the regression line of the target "value" and all factors and measure the parameter and residual to evaluate the fit.
- **Machine Learning Models:** Four machine learning models have been developed to predict price using a variety of methods, including Multilevel RNN, LSTM , Random Forest, and Linear Regression.
- **Model Performance Evaluation:** The model is evaluated based on various metrics such as R-squared, RMSE, MSE and MAE to determine its accuracy.
- **Analysis of key factors:** The study identified the most important factors in price prediction, including distance, time required, location ID, and whether the travel mode is urban or suburban.
- **How to use a user-based tool:** The model describes the development of a user-based tool that provides users with trip design, essential to helping solve uncertain fares for New York City taxi users. .
- **Model updates and MLOps:** It is recommended that all machine learning architectures that include these models follow MLOps best practices, including model updates through frequent training on updated information, and ensure that performance standards change over time. It's still great as time goes by.

.

11. Limitations

11.1 Data quality and availability:

1. The limited availability of high-quality and comprehensive data on taxi rides may hamper the performance of the model.
2. Inaccurate or incomplete data, such as missing fare or location information, can lead to biased predictions.

11.2 Complexity of fare determinants:

1. Calculating taxi fares involves various factors beyond distance and time, including traffic conditions, tolls, surcharges, and supply and demand dynamics.

2. Accurately capturing all relevant variables in a model can be challenging, leading to potential inaccuracies in fare predictions**.**

11.3 Dynamic and unpredictable factors:

1. External factors such as weather conditions, events, road closures and changes in regulations can unpredictably affect taxi fares.
2. Incorporating real-time data and adapting the model to dynamic conditions is a significant challenge.

11.4 Generalization and adaptation of the model:

1. Models trained on historical data may not generalize well to new regions, time periods, or demographic segments.
2. Adapting the model to different geographical areas, transport infrastructure and user preferences requires constant refinement and adaptation.

12. Future Enhancements

As technology involves and societal needs change, it is important for the project, Taxi Fare Prediction, to stay versatile and forward-thinking. Future enhancements include the potential features that enhance the functionality and usability of the taxi fare prediction system.

After performing the conclusive literature survey and carefully selecting the models to compare and evaluate their performance, there are certain future enhancements that may be adapted in the near future -

1. Integration of real-time sources for weather, fuel pricing, road blockages and traffic updates.
2. Implementation of dynamic pricing algorithms to adjust the fares based on demand and supply dynamics.
3. Integration of multi-modal transportation options for ride-sharing services.
4. Development of mobile applications with location-based services for personalized fare estimates and ride-sharing services.
5. Integration of blockchain technology for transparent fare transactions.

13. Conclusions

This research on taxi fare prediction in New York City offers valuable insights for both riders and policymakers:

- **Empowering Users:** The proposed user-based tool, leveraging trained prediction models, can empower riders to plan their trips more effectively. Estimating fares beforehand allows for informed decisions and budgeting.
- **Reduced Uncertainty:** Riders can feel more confident when hailing a taxi, knowing the approximate fare beforehand. This eliminates the anxiety of unknown costs associated with traditional taxi rides.
- **Improved Efficiency:** For taxi drivers, the ability to predict fares can help optimize routes and potentially reduce wasted time.

- **Data-Driven Decisions:** Policymakers can gain valuable insights into factors influencing taxi fares. This information can be used to optimize taxi services, address traffic congestion issues, and potentially implement dynamic pricing strategies based on demand and location.

14. References

1. AMADXARIF, Z., & OTTER, N. (2023). PREDICTING NEW YORK CITY TAXI FARES WITH SUPERVISED MACHINE LEARNING.
2. Arora, K., Kaur, S., & Sharma, V. (2021). Prediction of Dynamic Price of Ride-On-Demand Services Using Linear Regression. *International Journal of Computer Applications and Information Technology*, *13*(1), 376-389.
3. Banerjee, P., Kumar, B., Singh, A., Ranjan, P., & Soni, K. (2020). Predictive analysis of taxi fare using machine learning. *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol*, 373-378.
4. C. Zhang, F. Zhu, Y. Lv, P. Ye and F. -Y. Wang, "MLRNN: Taxi Demand Prediction Based on Multi-L evel Deep Learning and Regional Heterogeneity Analysis," in IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 7, pp. 8412-8422, July 2022, doi: 10.1109/TITS.2021.3080511
5. Agrawal, S., Sonbhadra, S. K., & Agarwal, S. (2018, September). Favour prediction of Taxi services using real-time visualization. In *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)* (pp. 2276-2282). IEEE.
6. Jacob, T. P., Pravin, A., Prasad, K. M., Judgi, G. T., & Rajakumar, R. (2022, March). Real time prediction of cab fare using machine learning. In *2022 International Conference on Electronics and Renewable Systems (ICEARS)* (pp. 1435-1438). IEEE.
7. Z. Duan et al., "Prediction of City-Scale Dynamic Taxi Origin-Destination Flows Using a Hybrid Deep Neural Network Combined With Travel Time," in IEEE Access, vol. 7, pp. 127816-127832, 2019, doi:1109/ACCESS.2019.2939902
8. Poongodi, M., et al. "New York City taxi trip duration prediction using MLP and XGBoost." *International Journal of System Assurance Engineering and Management* (2022): 1-12
9. Antoniades, Christophoros, Delara Fadavi, and A. F. J. Amon. "Fare and duration prediction: A study of New York city taxi rides." *Unpublished student paper* 104 (2016)
10.Xu, J., Rahmatizadeh, R., Bölöni, L., & Turgut, D. (2017). Real-time prediction of taxi demand using recurrent neural networks. *IEEE Transactions on Intelligent Transportation Systems*, *19*(8), 2572-2581.