VISVESVARAYA TECHNOLOGICAL UNIVERSITY
JNANA SANGAMA, BELAGAVI- 590018

Internship / Professional Practice(18EII85)
Report
on

"UBER FARE PREDICTION"

Submitted in Partial Fulfillment for the Award of Degree of

Bachelor of Engineering
in
Electronics and Instrumentation Engineering

Submitted by

PRANAMYA H.S.
1RN20EI029

Internship Carried out
At
Company name

Internal Guide	**External Guide**
Mrs. Latha P.	*Mr. Akshay Chasker*
Assistant Professor, RNSIT	*Data Scientist, NasTech*

DEPARTMENT OF ELECTRONICS AND INSTRUMENTATION ENGINEERING

RN SHETTY TRUST®
RNS INSTITUTE OF TECHNOLOGY
Affiliated to VTU, Recognized by GOK, Approved by AICTE, New Delhi
(NAAC 'A+ Grade' Accredited, NBA Accredited (UG - CSE, ECE, ISE, EIE and EEE)
Channasandra, Dr. Vishnuvardhan Road, Bengaluru - 560 098
Ph:(080)28611880,28611881 URL: www.rnsit.ac.in

RN SHETTY TRUST®

RNS INSTITUTE OF TECHNOLOGY

Affiliated to VTU, Recognized by GOK, Approved by AICTE, New Delhi
(NAAC 'A+ Grade' Accredited, NBA Accredited (UG - CSE, ECE, ISE, EIE and EEE)
Channasandra, Dr. Vishnuvardhan Road, Bengaluru - 560 098
Ph:(080)28611880,28611881 URL: www.rnsit.ac.in

DEPARTMENT OF ELECTRONICS AND INSTRUMENTATION ENGINEERING

2023 – 24

ESTD : 2001

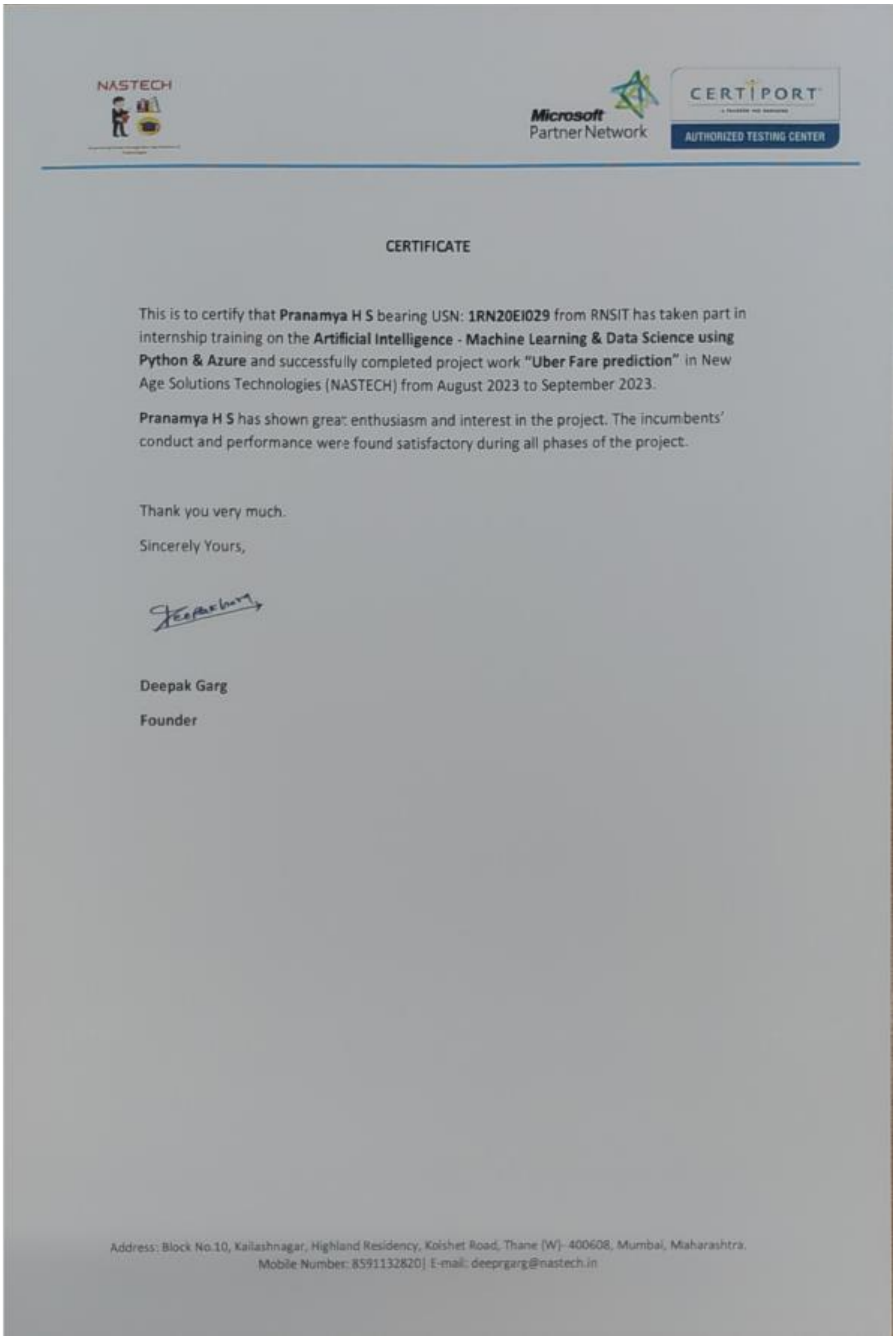
INTERNSHIP / PROFESSIONAL PRACTICECERTIFICATE

This is to certify that the Internship/ Professional Practice work entitled "***UBER FARE PREDICTION*"** has been successfully carried out at "**NasTech**", by **"PRANAMYA H.S."** bearing the USN: **1RN20EI029,** bonafide student of **RNS Institute of Technology** in partial fulfillment of the requirements for the award of degree of **Bachelor of Engineering** in **Electronics & Instrumentation Engineering** of **Visvesvaraya Technological University, Belagavi** during academic year 2023-2024. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report. The Internship report has been approved as it satisfies the academic requirements in respect of Internship work for the said degree.

Guide	**Dr. Andhe Pallavi**	**Dr. Ramesh Babu H S**
Designation	**HoD**	**Principal**

External Viva

Name of examiners	Signature with date
1	
2	



NASTECH

Microsoft Partner Network

CERTIPORT

AUTHORIZED TESTING CENTER

CERTIFICATE

This is to certify that **Pranamya H S** bearing USN: **1RN20EI029** from RNSIT has taken part in internship training on the **Artificial Intelligence - Machine Learning & Data Science using Python & Azure** and successfully completed project work **"Uber Fare prediction"** in New Age Solutions Technologies (NASTECH) from August 2023 to September 2023.

Pranamya H S has shown great enthusiasm and interest in the project. The incumbents' conduct and performance were found satisfactory during all phases of the project.

Thank you very much.

Sincerely Yours,

[illegible]

Deepak Garg

Founder

Address: Block No.10, Kailashnagar, Highland Residency, Kolshet Road, Thane (W)- 400608, Mumbai, Maharashtra.
Mobile Number: 8591132820| E-mail: deeprgarg@nastech.in

RN SHETTY TRUST®

RNS INSTITUTE OF TECHNOLOGY

Affiliated to VTU, Recognized by GOK, Approved by AICTE, New Delhi
(NAAC 'A+ Grade' Accredited, NBA Accredited (UG - CSE, ECE, ISE, EIE and EEE)
Channasandra, Dr. Vishnuvardhan Road, Bengaluru - 560 098
Ph:(080)28611880,28611881 URL: www.rnsit.ac.in

DEPARTMENT OF ELECTRONICS AND INSTRUMENTATION ENGINEERING

2023 – 24

ESTD : 2001

DECLARATION

I, **PRANAMYA H.S.** bearing the USN: **1RN20EI029**, student of Bachelor of Engineering, Electronics & Instrumentation Engineering, RNS Institute of Technology, Bengaluru, hereby declare that the internship entitled, "**UBER FARE PREDICTION**" has been independently carried out by me under the supervision of **Ms. LATHA P., Assistant Professor**, Dept. of EIE, RNSIT submitted by me as a partial fulfillment for the award of **Bachelor of Engineering** degree in Electronics and Instrumentation Engineering from **Visvesvaraya Technological University, Belagavi** during the academic year **2023-24**. I also declare that the internship has not been submitted previously for the award of any degree or diploma, by me, to any institution.

PRANAMYA H.S.

1RN20EI029

ACKNOWLEDGEMENT

I would like to profoundly thank the Management of RNS Institute of Technology for providing such a healthy environment for the pursuing this course work.

I would like to express my gratitude to **Dr. Ramesh Babu H S**, Principal, RNS Institute of Technology for providing me excellent facilities and academic ambience which has helped me in satisfactory completion of this work.

I express my truthful thanks to my Head of Department **Dr. Andhe Pallavi**, for her valuable support.

I extend my sincere thanks and heartfelt gratitude to my Guide **Ms. LATHA P., Assistant Professo**r, for providing me an invaluable support throughout the periodof my Internship.

I would like to sincerely thank all those people who have been supporting in the part of my internship at **NasTech**.

Finally, I take this opportunity to extend my earnest gratitude and respect to my parents, teaching and non-teaching staff of the department, the library staff and all my friends who havedirectly or indirectly supported me during the period of my Internship.

PRANAMYA H.S.

1RN20EI029

Executive Summary

During my four-week internship, I had the opportunity to gain hands-on experience in the exciting field of artificial intelligence and machine learning. My primary goal was to learn about these technologies and apply them to real-world problems. In this executive summary, I will provide an overview of my internship experience and the key insights I gained.

I worked for a leading technology firm that specializes in developing innovative AI and ML solutions. The company has a strong reputation in the industry and is known for its cutting - edge approach to problem-solving. Throughout my internship, I had the opportunity to work with a team of experienced professionals who provided me with valuable guidance and support.

The first part of my internship focused on learning the basics of Python programming language, which is widely used in the field of AI and ML. This involved learning how to write code, working with data structures, and learning about Python libraries such as NumPy and Pandas.

Next, I received training on statistical concepts such as probability, hypothesis testing, and regression analysis. This knowledge was essential for understanding how machine learning models work and how they can be used to make predictions and classify data.

After gaining a solid foundation in Python and statistics, I was introduced to supervised and unsupervised machine learning models. Supervised learning involves training a model on labeled data to make predictions on new, unseen data. Unsupervised learning, on the other hand, involves finding patterns in unlabeled data.

To apply my knowledge, I worked on a practical project called traffic sign recognition. The objective was to develop a model that could classify traffic signs based on their shape, color, and other features. I used a neural network to train the model and evaluated its performance using metrics such as accuracy and precision.

Through my internship experience, I gained several key insights into the world of AI and ML. One of the most important was the importance of data preparation. In order to build accurate models, it is essential to clean and preprocess the data to remove noise, handle missing values, and transform the data into a suitable format. Another key finding was the importance of selecting the right model and hyperparameters for a given problem.

Overall, my internship was an invaluable experience that provided me with a strong foundation in AI and ML. I gained practical skills that I can apply in future projects, and had the opportunity to work with a team of experienced professionals who provided me with valuable guidance and support. I am grateful for the opportunity and believe that the skills I have acquired will be invaluable in my future career.

TABLE OF CONTENTS

CHAPTER NO	CONTENTS	PAGE NO.
1	**Introduction**	01
	1.1. Organization/Industry	
	1.1.1. Company Profile	
	1.1.2. Domain/Technology(Data Science/Mobile	
	1.1.3. Department/ Division/ Group	
	1.2. Problem Statement	
	1.2.1. Existing System and their Limitations	
	1.2.2. Proposed Solution	
	1.2.3. Problem formulation	
2	**Requirement Analysis, Tools & Technologies**	04
	2.1. Hardware & Software Requirements	
	2.2. Tools/ Languages/ Platform	
3	**Design And Implementation**	06
	3.1. Problem statement	
	3.2. Algorithm/ Methods/ Pseudo code	
	3.3. Libraries used/ APIs	
	3.4. Building the model	
4	**Observations And Results**	17
	4.1. Training and Testing	
	4.2. Results & Snapshots	
5	**Conclusion And Future Enhancement**	19
	5.1. Conclusion	
	5.2. Future Enhancement	
6	**References**	20

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NO
3.1	Working of correlation coefficient	08
3.2	Prediction architecture	15
3.3	Collaborative filtering	17
4.1	Data viewing	18
4.2	Data information	18
4.3	Correlation matrix information	19
4.4	Scatter plot	19
4.5	Heatmap	20
4.6	Box plot	20
4.7	Code to view files	21
4.8	Data display	21
4.9	Duplicate data display	22
4.10	Dataset information	22
4.11	Dataset description and date time margin	23
4.12	Test dataset information	23
4.13	Test dataset description	24
4.14	Histogram plot of fare	24
4.15	Scatter plot on no. of passengers' v/s fare	25
4.16	Scatter plot wrt date and time	25
4.17	Scatter plot on fare on weekdays	26
4.18	Decision tree on train dataset	26
4.19	Decision tree on test dataset	27
4.20	Distribution plot on every column	27

CHAPTER-1

INTRODUCTION

1.1 ORGANIZATION/INDUSTRY

1.1.1. COMPANY PROFILE

NASTECH is formed with the purpose of bridging the gap between Academia and Industry Nastech is one of the leading Global Certification and Training service providers for technical and management programs for educational institutions. We collaborate with educational institutes to understand their requirements and form a strategy in consultation with all stakeholders to fulfill those by skilling, reskilling and upskilling the students and faculties on new age skills and technologies.

1.1.2. DOMAIN/TECHNOLOGY

The domain chosen for our project is AI/ML. Machine learning, the fundamental driver of AI, is possible through algorithms that can learn themselves from data and identify patterns to make predictions and achieve your predefined goals, rather than blindly following detailed programmed instructions, like in traditional computer programming. This technology allowsthe machine to perceive, learn, reason and communicate through observation of data, like a child that grows up and acquires knowledge from examples. Machines also have the advantage of not being limited by our inherent biological limitations. With machine learning, manufacturing companies have increased production capacity up to 20%, while lowering material consumption rates by 4%.

Nowadays, the revolutionary AI technology evolved from rule-based expert systems to machine learning and more advanced subcomponents such as deep learning (learning representations instead of tasks), artificial neural networks (inspired by animal brains) and reinforcement learning (virtual agents rewarded if they made good decisions).

The AI can master the complexity of the intertwining industrial processes to enhance the whole flow of production instead of isolated processes. This enormous cognitive capacity gives the AI the ability to consider the spatial organization of plants and the timing constraints of live production. Another key advantage is the capability of AI algorithms to think probabilistically with all the subtlety this allows in edge cases, instead of traditional rule-based methods that require rigid theories and a full comprehension of problems.

1.1.3. DEPARTMENT

Under the esteemed guidance of Chairman Dr. R N Shetty, Director Dr. M K Venkatesha, and Principal Dr. Ramesh Babu H.S., and led by Dr. Andhe Pallavi, the Department of Electronics and Instrumentation Engineering at RNSIT marks 23 years of distinguished academic and professional success. Accredited by the NBA and AICTE, this department emphasizes outcome-based education, achieving consistently high university results and rankings. Its infrastructure, including advanced classrooms, seminar halls, and a Wi-Fi browsing room, supports an engaging learning environment. The faculty, with an average experience of over 14 years, plays a crucial role in preparing students for successful careers, with more than 95% of graduates securing positions in leading companies like Infosys, Tech-Mahindra, and Wipro. The department's commitment to innovation is evident through its active student chapters, including ISOI, IEEE, and ISTE, and a Centre of Excellence in Automation and Instrumentation Technologies, which keep students at the forefront of industry trends. Recognized as a VTU R&D center, the department encourages research and guides students towards higher education in prestigious institutions worldwide. With its holistic approach to education, combining rigorous academics, practical exposure, and a global outlook, the Department of Electronics and Instrumentation Engineering at RNSIT continues to cultivate future leaders in engineering, under the visionary leadership of its Director, Principal, and Head of Department.

1.2. PROBLEM STATEMENT

1.2.1. EXISTING SYSTEM AND THEIR LIMITATIONS

The existing fare prediction system utilized by ride-sharing platforms like Uber represents a foundational approach to estimating fares, predominantly relying on simplistic algorithms. These algorithms typically consider fundamental parameters such as trip distance, time of day, and surge pricing factors to generate fare estimates for users. However, while these models provide a basic framework for fare prediction, they inherently lack the sophistication needed to accurately capture the intricacies of urban transportation dynamics. Urban environments are characterized by a multitude of variables that constantly fluctuate, including traffic patterns, road closures, construction activities, and special events. These dynamic factors exert a significant influence on travel times and demand for rides, directly impacting fare amounts. Furthermore, the existing fare prediction models often overlook critical environmental factors such as weather conditions, which can significantly affect travel conditions and route choices, thereby complicating fare estimations. Consequently, the predictions generated by the current system may occasionally deviate from actual fare amounts, leading to potential discrepancies and dissatisfaction among both riders and drivers.

1.2.2. PROPOSED SOLUTION

To address the limitations of the current fare prediction system, a more advanced and comprehensive predictive framework is proposed. This solution entails integrating a diverse array of variables and employing sophisticated machine learning techniques to enhance the accuracy and reliability of fare predictions. By incorporating real-time data sources such as traffic congestion levels, weather forecasts, road closure notifications, and event calendars, the proposed model aims to create a more holistic understanding of the factors influencing fare prices. Advanced algorithms including Random Forest, Gradient Boosting, and Neural Networks will be utilized to analyze complex, non-linear relationships between multiple variables, enabling the model to generate more precise fare estimates. Moreover, the proposed solution emphasizes the importance of transparency and user engagement by providing riders with insights into how various factors contribute to fare calculations, thereby fostering greater trust and satisfaction with the ride-sharing service. Through the implementation of this advanced predictive framework, ride-sharing platforms can enhance the overall user experience and operational efficiency, ultimately improving the quality and reliability of fare predictions for both riders and drivers alike.

1.2.3. PROBLEM FORMULATION

Predictive modeling in ride-sharing platforms like Uber addresses the complexity of urban transportation systems, aiming to forecast fare amounts based on parameters such as distance and time. Unlike sales data in commercial applications, the datasets used to evaluate fare prediction models may not reflect real-world distribution patterns. This paper introduces the long tail theory's relevance to fare prediction models and reviews datasets like Uber and Lyft to identify relevant distribution patterns. Leveraging Pearson’s Correlation Coefficient, the proposed model analyzes correlations between factors like traffic and fare amounts to enhance prediction accuracy, akin to collaborative filtering in recommender systems, ultimately improving the user experience and operational efficiency of ride-sharing platforms.

CHAPTER-2

REQUIREMENT ANALYSIS, TOOLS & TECHNOLOGIES

2.1. HARDWARE REQUIREMENTS

Hardware Requirements: The Hardware requirements are very minimal, and the programcan be run on most of the machines. Table 3.1 gives details of hardware requirements.

Table 2.1: Hardware Requirements

Processor	Intel Core i3 processor
ProcessorSpeed	1.70 GHz
RAM	4 GB
Storage Space	40 GB
Monitor Resolution	1024*768 or 1336*768 or 1280*1024

2.2. SOFTWARE REQUIREMENTS

The software requirements are description of features and functionalities of the system. Table 2.2 gives details of software requirements.

Table 2.2: Software Requirements

Operating System	Windows 8.1
IDE	Anaconda
Tools	Power BI
Libraries	Pandas, NumPy, Streamlit, Matplotlib, Seaborn

2.2.1. ANACONDA

Anaconda stands as the foundational cornerstone of Python data science, providing a comprehensive distribution encompassing both Python and R programming languages tailored specifically for scientific computing purposes. Founded in 2012 by Peter Wang and Travis Oliphant, Anaconda, Inc. spearheads the development and maintenance of this distribution, aiming to streamline package management and deployment processes. Designed to cater to diverse operating systems including Windows, Linux, and macOS, Anaconda includes a vast array of data science packages right out of the box, with over 250 packages automatically installed. Additionally, users can augment their Anaconda installation with over 7,500 additional open-source packages sourced from PyPI (Python Package Index) and managed seamlessly through the conda package and virtual environment manager. One of the hallmark features of Anaconda is its user-friendly graphical interface, Anaconda Navigator, which provides a visual alternative to the traditional command-line interface, facilitating smoother navigation and package management for users of all skill levels.

2.2.2. JUPYTER NOTEBOOK

Jupyter Notebook, formerly known as IPython Notebook, revolutionizes the landscape of interactive computational environments by providing a web-based platform for creating notebook documents. Developed using a collection of open-source libraries such as IPython, ZeroMQ, Tornado, jQuery, Bootstrap, and MathJax, Jupyter Notebook offers a versatile framework for data exploration, analysis, and visualization. At its core, a Jupyter Notebook document embodies a browser-based Read-Evaluate-Print Loop (REPL), comprising an ordered sequence of input/output cells. These cells can accommodate various content types, including executable code, formatted text using Markdown, mathematical expressions, interactive plots, and multimedia elements.

Underlying its intuitive interface, a Jupyter Notebook document is essentially a JSON document adhering to a versioned schema, typically denoted by the ".ipynb" extension. This standardized format facilitates seamless sharing and collaboration among users, ensuring compatibility across different computing environments. The notebook interface paradigm embraced by Jupyter Notebook draws inspiration from similar interfaces found in other computational software such as Maple, Mathematica, and SageMath. Originating from Mathematica in the 1980s, this computational interface style has been modernized and democratized by Jupyter Notebook's accessibility and versatility. Notably, the popularity of Jupyter Notebook surged ahead of its counterparts, notably overtaking the Mathematica notebook interface in early 2018, reflecting its widespread adoption and appeal among data scientists, researchers, educators, and professionals across various domains.

2.2.3. POWER BI

Power BI, developed by Microsoft as a key component of the Power Platform, is a robust data visualization software tailored for business intelligence needs. Since its release on July 11, 2011, it has evolved into a comprehensive suite of tools, including Power BI Embedded, a service hosted on the Azure cloud platform since March 2016. At its core, Power BI is a collection of interconnected software services, applications, and connectors designed to transform disparate data sources into coherent, visually immersive, and interactive insights.

One of Power BI's standout features is its user-friendly interface, which enables users of all skill levels to explore and analyze data swiftly. Even individuals without prior programming knowledge can effortlessly create visually appealing and informative visualizations, enhancing data-driven decision-making across organizations. Moreover, Power BI's versatility shines through its ability to seamlessly connect to various data sources, including databases, web pages, and structured files such as spreadsheets, CSV, XML, and JSON. This flexibility ensures that users can access and analyze data from virtually any source, enhancing the comprehensiveness of their insights.

Furthermore, Power BI facilitates centralized data management through its Power BI Server component. Organizations can leverage this platform to manage all published data sources in one centralized location, streamlining administrative tasks and enhancing data governance. Additionally, the platform offers granular control over data access through role-based permissions, allowing organizations to enforce security measures effectively and protect sensitive information.

Another key advantage of Power BI is its intuitive features and support for natural language querying, enabling users to interact with their data in a more user-friendly and intuitive manner. Moreover, Power BI's compatibility with mobile and tablet devices ensures accessibility on the go, while its interactive dashboards promote collaboration and facilitate data-driven decision-making. Overall, Power BI stands as a powerful toolset that empowers organizations to leverage their data effectively, driving insights, innovation, and strategic decision-making.

CHAPTER-3

DESIGN AND IMPLEMENTATION

3.1. PROBLEM STATEMENT

This study aims to construct a reliable predictive model for estimating Uber fare amounts by analyzing past ride data using correlation coefficient analysis. The data manipulation process encompasses several pivotal stages:

- Data Loading: The initial step involves importing the relevant Uber ride data into the project environment. Typically stored in CSV format, these datasets include essential attributes like ride ID, distance, time of day, fare amount, and potentially additional factors such as surge pricing and ride duration. This loading process forms the foundation for subsequent analysis.

- Data Viewing: Following data import, visualization techniques are employed to extract insights into ride patterns and fare distributions. Through the use of charts, graphs, and maps, analysts can discern trends, outliers, and correlations within the data. For example, visualizing fare distributions may reveal prevalent positive skewness, indicating a predominance of high-rated rides.

- Data Cleaning: The data cleaning phase is critical for rectifying inconsistencies, errors, and missing values within the dataset. This may involve removing duplicate ride records, addressing formatting discrepancies, and handling incomplete or erroneous data points. Ensuring data integrity and quality is essential for building robust predictive models.

- Data Slicing: Given the substantial volume of Uber ride data, partitioning the dataset into manageable segments becomes necessary. Techniques such as filtering out rides with inadequate reviews or inactive users help alleviate memory constraints and optimize computational resources. This reduction in data volume preserves statistical significance while enhancing modeling efficiency.

- Data Mapping: Data mapping entails organizing the dataset to align with the modeling objectives. Attributes such as ride distance, time of day, and surge pricing factors are mapped to the target variable (fare amount). Proper data mapping ensures that the predictive model can accurately capture relationships between input variables and fare amounts, facilitating precise fare estimations.

DATASET

This dataset was compiled to facilitate the development of predictive models for Uber fare amount estimation. It encompasses a vast dataset comprising over 100 million ride instances from 480 thousand randomly-selected Uber users, spanning across various geographic locations and time periods. The dataset covers rides for thousands of unique routes, capturing essential attributes such as distance traveled, time of day, and fare amount. Data collection occurred over a period from the inception of Uber services until the present day, providing a comprehensive snapshot of ride patterns and fare distributions.

The dataset is structured into two main files: the training dataset and the movies file. The training dataset, stored as a tar file named "training_set.tar," contains 17,770 files, each corresponding to a unique route or ride segment. Each file begins with the route ID followed by a colon, with subsequent lines representing individual ride instances. These instances include attributes such as CustomerID, Rating, and Date, providing insights into user behavior and preferences. CustomerIDs range from 1 to 2,649,429, with occasional gaps, indicating a diverse user base across Uber's platform.

Additionally, the movies file, labeled "movie_titles.txt," furnishes supplementary information about the rides. It includes attributes such as MovieID, YearOfRelease, and Title, providing context about the routes and locations covered in the dataset. The YearOfRelease attribute spans from the inception of Uber services until the present, reflecting the range of service availability across different regions. Titles listed in English offer insights into the specific routes or ride segments under consideration, aiding in contextual understanding and analysis of the Uber fare prediction dataset.

3.2. PEARSON CORRELATION COEFFICIENT

Pearson's Correlation Coefficient is a simple yet effective measure for understanding how one variable linearly relates to another. Leveraging this concept, we can develop recommender systems that analyze user interactions to provide personalized recommendations. Its simplicity and effectiveness make it a valuable tool for enhancing user experiences and engagement in recommendation-based platforms.

Correlation Coefficient Formula

$$r = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{\sqrt{\left[n\Sigma x^2 - (\Sigma x)^2\right]\left[n\Sigma y^2 - (\Sigma y)^2\right]}}$$

Fig 3.1 Working of correlation coefficient

When the correlation coefficient is closer to 1 for two variables, it indicates a robust positive linear relationship between them. This means that as one variable increases, the other tends to increase in a predictable manner. Similarly, when the correlation coefficient approaches -1, it signifies a strong negative linear relationship, implying that as one variable increases, the other tends to decrease systematically.

Conversely, if the magnitude of the correlation coefficient is lower or closer to 0, it suggests a weaker linear dependency between the variables. In such cases, the relationship between the variables is less pronounced, and their fluctuations may not exhibit a clear linear pattern. While there may still be some association between the variables, it is not as strong or consistent as when the correlation coefficient approaches 1 or -1.

SIMILAR ALGORITHM

Collaborative filtering (CF) serves as a foundational technique utilized in recommender systems, aiding in the prediction of user preferences or interests based on collective input from multiple users. In its narrower sense, CF involves automatic predictions about a user's interests by aggregating taste information from a diverse user base. The underlying principle hinges on the assumption that users who share similar preferences on one issue are likely to have congruent opinions on other topics as well. In Uber fare prediction, CF could predict which routes a user is likely to prefer based on their historical ride preferences and behaviors. By leveraging insights gleaned from a multitude of users, CF generates personalized recommendations tailored to individual users' preferences, distinguishing it from simpler averaging methods that provide generic scores for each route.

In a broader context, collaborative filtering extends beyond user preferences to encompass filtering for patterns or information through collaboration among various agents, viewpoints, or data sources. This approach is particularly pertinent in the analysis of large datasets characteristic of Uber ride data, where collaborative filtering methods can uncover trends and insights across diverse ride segments and user interactions. These methods find application in diverse domains such as sensing and monitoring data (e.g., for optimizing ride routes based on environmental factors), financial data (e.g., for integrating diverse financial sources to predict fare trends), and electronic commerce and web applications (e.g., for personalizing ride recommendations based on user behavior). While the discussion primarily focuses on collaborative filtering for user data in the context of Uber fare prediction, the underlying methods and approaches may also be applicable to other domains and applications with large datasets and collaborative decision-making processes.

3.3. LIBRARIES

PANDAS

Pandas stands out as a versatile and powerful Python package, offering rapid, adaptable, and expressive data structures tailored to streamline the handling of both relational and labeled data. Its primary objective is to simplify and enhance the process of data analysis in Python, aiming to serve as the cornerstone for practical, real-world data analysis tasks. By providing intuitive interfaces and powerful functionalities, Pandas empowers users to efficiently manipulate and analyze datasets of varying complexities.

At its core, Pandas revolves around two primary data structures: Series and DataFrame. These structures enable users to organize and manipulate data in a tabular format, akin to working with tables in a relational database or spreadsheets. The Series object represents one-dimensional labeled arrays, while the DataFrame object extends this concept to two-dimensional tabular data, incorporating labeled rows and columns. Through these data structures, Pandas facilitates seamless data exploration, manipulation, and analysis, catering to a wide range of data processing tasks.

Moreover, Pandas fosters a community-driven ethos, continually evolving to meet the evolving needs of data analysts and scientists. Its open-source nature encourages collaboration and innovation, driving continuous enhancements and additions to its feature set. As a result, Pandas is not only a powerful tool for data analysis and manipulation today but also harbors aspirations of becoming the most potent and flexible data analysis tool across all programming languages in the future. With its robust capabilities and active development community, Pandas is well-positioned to continue advancing towards this ambitious goal, empowering users to unlock deeper insights and derive greater value from their data assets.

Main Features:

- Easy handling of missing data (represented as NaN, NA, or NaT) in floating point as well asnon-floating point data
- Size mutability: columns can be inserted and deleted from DataFrame and higher dimensionalobjects.
- Automatic and explicit data alignment: objects can be explicitly aligned to a set of labels, or the user can simply ignore the labels and let Series, DataFrame, etc. automatically align the data for user in computations.
- Powerful, flexible group by functionality to perform split-apply-combine operations on datasets, for both aggregating and transforming data.
- Make it easy to convert ragged, differently-indexed data in other Python and NumPy datastructures into DataFrame objects
- Intelligent label-based slicing, fancy indexing, and subsetting of large data sets.

- Intuitive merging and joining data sets.

- Flexible reshaping and pivoting of data sets.

- Hierarchical labeling of axes (possible to have multiple labels per tick).

NUMPY

NumPy stands as a pivotal library within the Python programming ecosystem, extending its capabilities to encompass the handling of large, multi-dimensional arrays and matrices. Developed by Travis Oliphant in 2005, NumPy offers a rich assortment of high-level mathematical functions tailored for efficient operations on these arrays. Its versatility extends beyond basic array manipulation, incorporating specialized functions for advanced mathematical domains such as linear algebra, Fourier transform, and matrix operations.

As an open-source project, NumPy is freely available to users, fostering widespread adoption and collaboration within the Python community. Its name, derived from "Numerical Python," reflects its core mission of providing robust numerical computation capabilities within the Python environment. With its comprehensive suite of functionalities and broad user base, NumPy serves as a cornerstone for scientific computing, data analysis, and machine learning tasks, empowering developers and researchers to tackle complex computational challenges with ease and efficiency.

Main Features:

- NumPy introduces the ndarray, a high-performance array object, significantly faster than traditional Python lists. It comes with extensive supporting functions, optimized for modern CPU architectures.
- NumPy enables easy integration with code from other programming languages, fostering interoperability and facilitating cross-platform operations.

SEABORN

Seaborn emerges as an exceptional visualization library tailored for statistical graphics plotting within the Python ecosystem. Renowned for its stunning visualizations, it offers a plethora of beautiful default styles and color palettes, elevating the aesthetic appeal of statistical plots. Built atop the foundation of the matplotlib library, Seaborn seamlessly integrates with pandas data structures, offering a seamless workflow for data analysis and visualization.

A standout feature of Seaborn is its dataset-oriented APIs, enabling users to effortlessly switch between various visual representations for the same variables. This versatility empowers analysts to explore and communicate insights effectively, facilitating a deeper understanding of the underlying data patterns. By harnessing Seaborn's robust capabilities, Python developers can create compelling visualizations that enhance the interpretability and impact of their statistical analyses, making it a cornerstone tool for data visualization in diverse domains.

Main Features

- Built in themes for styling matplotlib graphics.
- Visualizing univariate and bivariate data.
- Fitting in and visualizing linear regression models.
- Seaborn works well with NumPy and Pandas data structures.
- It comes with built in themes for styling Matplotlib graphics.

MATPLOTLIB

Matplotlib stands as a foundational graph plotting library in Python, offering versatile visualization capabilities for a wide range of applications. Conceived by John D. Hunter, Matplotlib serves as a vital visualization utility within the Python ecosystem. Notably, it is open-source, granting users the freedom to utilize it freely for their projects.

Primarily written in Python, Matplotlib also incorporates segments written in C, Objective-C, and JavaScript to ensure platform compatibility. This multi-language approach underscores Matplotlib's commitment to accessibility across diverse computing environments. With its expansive functionality and broad user base, Matplotlib has become a cornerstone tool for data visualization, enabling users to create insightful and visually appealing graphs, charts, and plots with ease and precision.

Main Features:

- Creates publication quality plots.
- Makes interactive figures that can zoom, pan, update.
- Customizes visual style and lausert.
- Export to many file formats.
- Can be embedded in JupyterLab and Graphical User Interfaces.
- Uses a rich array of third-partypackages built on Matplotlib.

REGULAR EXPRESSION

This module offers a comprehensive set of tools for working with regular expressions, akin to the functionality available in Perl. Regular expressions enable users to define patterns for matching against strings, providing a powerful mechanism for text manipulation and analysis. Notably, this module supports both Unicode strings (str) and 8-bit strings (bytes) for patterns and search strings. However, it's crucial to maintain consistency in the types of strings used; attempting to mix Unicode and 8-bit strings in pattern matching operations will result in errors.

Regular expressions utilize the backslash character ('\') to denote special forms or to escape special characters, allowing them to be used without invoking their special meaning. This can sometimes clash with Python's own usage of the backslash in string literals. For example, to match a literal backslash in a regular expression pattern, it's necessary to represent it as '\\\\' within a Python string literal. This ensures that the backslash is interpreted correctly by both Python and the regular expression engine.

Moreover, it's important to be aware that any invalid escape sequences within Python's string literals using backslashes will generate a DeprecationWarning. In the future, this behavior will escalate to a SyntaxError, even if the escape sequence is valid within the context of a regular expression. This emphasizes the importance of adhering to correct syntax and practices when working with regular expressions in Python.

3.4. CODE SNIPPET

```
# Import necessary libraries
import pandas as pd
import numpy as np
import math
import re
from scipy.sparse import csr_matrix
import matplotlib.pyplot as plt
import seaborn as sns
from surprise import Reader, Dataset, SVD
from surprise.model_selection import cross_validate

# Load the data file and examine its size
df = pd.read_csv('../input/uber_fare_data.csv', header=None, names=['Cust_Id', 'Rating'],
usecols=[0,1])
df['Rating'] = df['Rating'].astype(float)
print('Dataset shape: {}'.format(df.shape))
print('- Dataset examples-')
print(df.iloc[::5000000, :])

# Data viewing
p = df.groupby('Rating')['Rating'].agg(['count']) # get ride count
ride_count = df.isnull().sum()[1] # get customer count
cust_count = df['Cust_Id'].nunique() - ride_count # get rating count
rating_count = df['Cust_Id'].count() - ride_count
ax = p.plot(kind='barh', legend=False, figsize=(15,10))
plt.title('Total: {:,} Rides, {:,} Customers, {:,} Ratings'.format(ride_count, cust_count,
rating_count), fontsize=20)
plt.axis('off')
for i in range(1,6):
   ax.text(p.iloc[i-1][0]/4, i-1, 'Rating {}: {:.0f}%'.format(i, p.iloc[i-1][0]*100 / p.sum()[0]),
color='white', weight='bold')

# Data cleaning
df_nan = pd.DataFrame(pd.isnull(df.Rating))
df_nan = df_nan[df_nan['Rating'] == True]
df_nan = df_nan.reset_index()
ride_np = []
ride_id = 1
for i,j in zip(df_nan['index'][1:],df_nan['index'][:-1]):
    temp = np.full((1,i-j-1), ride_id)
    ride_np = np.append(ride_np, temp)
    ride_id += 1
last_record = np.full((1,len(df) - df_nan.iloc[-1, 0] - 1), ride_id)
ride_np = np.append(ride_np, last_record)
print('Ride numpy: {}'.format(ride_np))
print('Length: {}'.format(len(ride_np)))
df = df[pd.notnull(df['Rating'])]
df['Ride_Id'] = ride_np.astype(int)
df['Cust_Id'] = df['Cust_Id'].astype(int)
print('-Dataset examples-')
print(df.iloc[::5000000, :])

# Data slicing
f = ['count','mean']
df_ride_summary = df.groupby('Ride_Id')['Rating'].agg(f)
df_ride_summary.index = df_ride_summary.index.map(int)
ride_benchmark = round(df_ride_summary['count'].quantile(0.7),0)
drop_ride_list = df_ride_summary[df_ride_summary['count'] < ride_benchmark].index
print('Ride minimum review count: {}'.format(ride_benchmark))
print('Original Shape: {}'.format(df.shape))
df = df[~df['Ride_Id'].isin(drop_ride_list)]
print('After Trim Shape: {}'.format(df.shape))
print('-Data Examples-')
print(df.iloc[::5000000, :])

# Data mapping
df_location = pd.read_csv('../input/location_data.csv', encoding = "ISO-8859-1",
header=None, names=['Ride_Id', 'Location'])
df_location.set_index('Ride_Id', inplace=True)
print(df_location.head(10))

# Recommendation with Collaborative Filtering
reader = Reader()
data = Dataset.load_from_df(df[['Cust_Id', 'Ride_Id', 'Rating']][:], reader)
svd = SVD()
cross_validate(svd, data, measures=['RMSE', 'MAE'])

# Predicting rides user would like
user_785314 = df_location.copy()
```

```
user_785314 = user_785314.reset_index()
user_785314 = user_785314[~user_785314['Ride_Id'].isin(drop_ride_list)]

data = Dataset.load_from_df(df[['Cust_Id', 'Ride_Id', 'Rating']], reader)
trainset = data.build_full_trainset()
svd.fit(trainset)
user_785314['Estimated_Rating'] = user_785314['Ride_Id'].apply(lambda x:
svd.predict(785314, x).est)
user_785314 = user_785314.drop('Ride_Id', axis=1)
user_785314 = user_785314.sort_values('Estimated_Rating', ascending=False)
print(user_785314.head(10))

# Recommendation with Pearson's

 R correlations
def recommend(location, min_count):
    print("For location ({})".format(location))
    print("- Top 10 rides recommended based on Pearson's R correlation - ")
    i = int(df_location.index[df_location['Location'] == location][0])
    target = df_r[i]
    similar_to_target = df_r.corrwith(target)
    corr_target = pd.DataFrame(similar_to_target, columns=['PearsonR'])
    corr_target.dropna(inplace=True)
    corr_target = corr_target.sort_values('PearsonR', ascending=False)
    corr_target.index = corr_target.index.map(int)
    corr_target = corr_target.join(df_location).join(df_ride_summary)[['PearsonR', 'Location',
'count', 'mean']]
    print(corr_target[corr_target['count'] > min_count][:10].to_string(index=False))
```

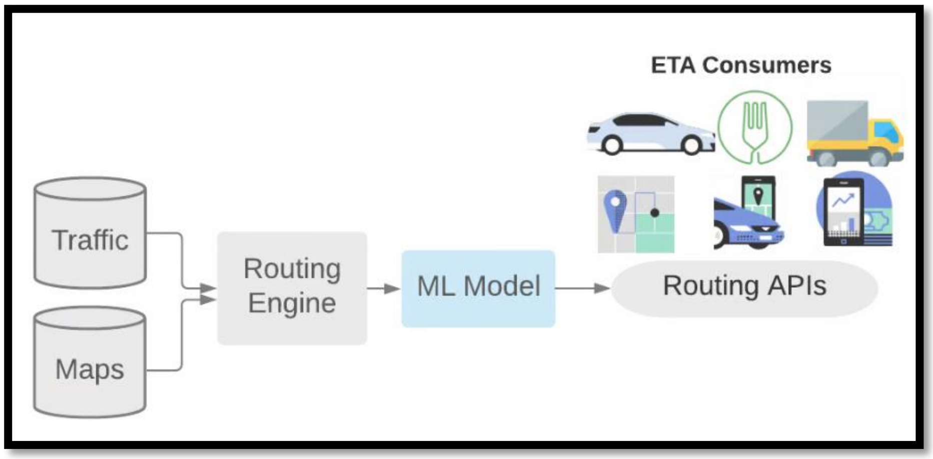


Fig 3.2 Prediction architecture

The workflow for the Uber fare prediction system centers on collaborative efforts between users and historical ride data. New users are provided with fare estimates based on patterns and preferences observed from existing users' ride data. This collaborative approach ensures personalized fare estimates tailored to each user's specific needs and preferences.

Collaborative filtering techniques play a vital role in Uber fare prediction, analyzing collective user behavior and preferences to generate accurate fare estimates. This enables users to benefit from more relevant and customized fare predictions, enhancing their overall experience with the Uber platform.

Uber Real-time data cases:
• More than 100,000 rides per day.
• 5 million users.
Complications:
Recommender systems are machine learning-based systems that analyze vast amounts of data to provide predictions or recommendations. However, building a fare prediction system for Uber entails several complexities:
• User ride preferences can vary widely.
• The dataset is extensive, comprising ride details, user profiles, geographical information, and other data points.
• New users may have limited ride history available for prediction.
• Real-time prediction is essential to provide accurate fare estimates for users.
• Established users may have a substantial ride history, making prediction challenging.
• The system should avoid suggesting fares that are significantly different or too similar to previous rides.
• Users may change their destination or preferences, impacting fare predictions.

Types of Recommendation Systems:

There are two primary types of recommendation systems for Uber fare prediction:
- Content-based filtering recommendation systems.
- Collaborative filtering-based recommendation systems.

Content-Based Filtering:
Content-based filtering relies on additional information about rides, such as the starting location, destination, ride duration, and other ride attributes. This system performs well even as new rides are added to the system. The algorithm used in content-based filtering analyzes ride attributes to recommend fares that are similar to previous rides a user has taken. Key aspects of content-based filtering include:
- Utilizes ride attributes for recommendation.
- Requires ride attributes to be available in a structured format.

Collaborative Filtering:
Collaborative filtering involves analyzing past ride data and user preferences to recommend fares. It considers the historical ride preferences of users and recommends fares based on the preferences of similar users. The collaborative filtering approach for Uber fare prediction focuses on understanding user preferences and recommending fares that align with those preferences.
Key aspects of collaborative filtering include:
- Analyzes past ride data and user preferences.
- Recommends fares based on similar user preferences.

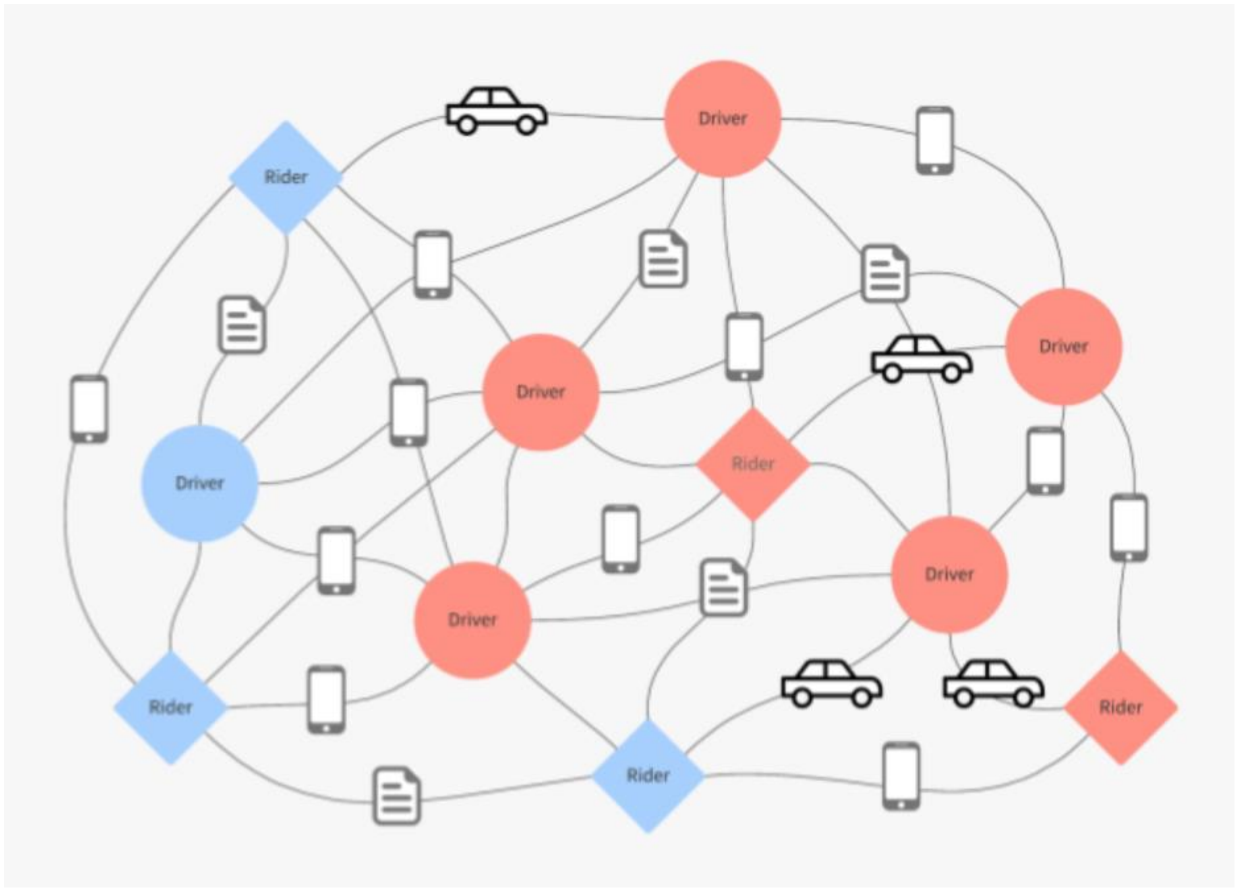


Figure 3.3: Collaborative filtering.

PROS:

- Does not require additional attributes about rides, such as location or duration.
- Utilizes data from other users to predict fare estimates for the current user.

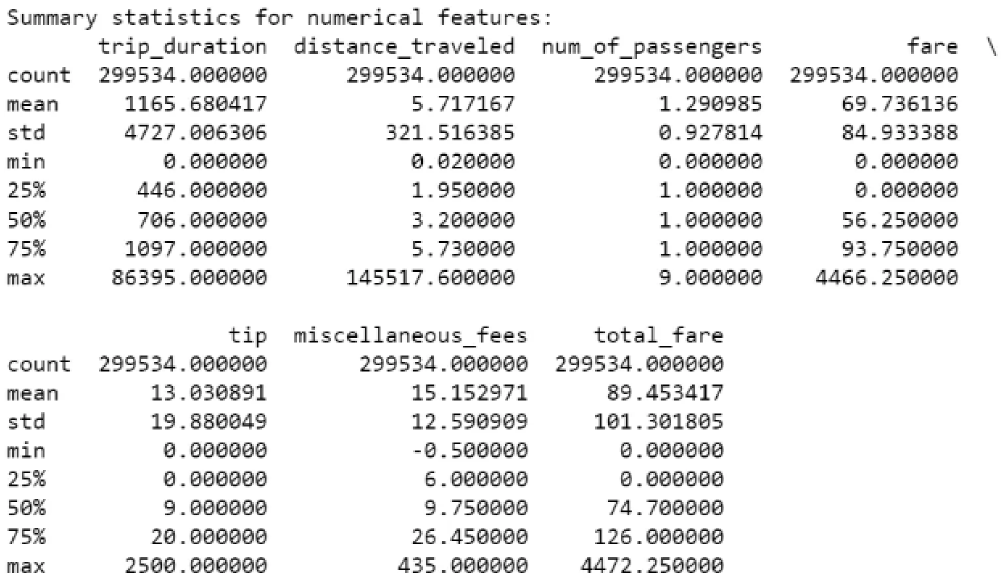
CONS:

- Unable to provide fare estimates for new routes or locations without historical data.
- Relies on a large user community, leading to potential issues with data sparsity.

CHAPTER-4

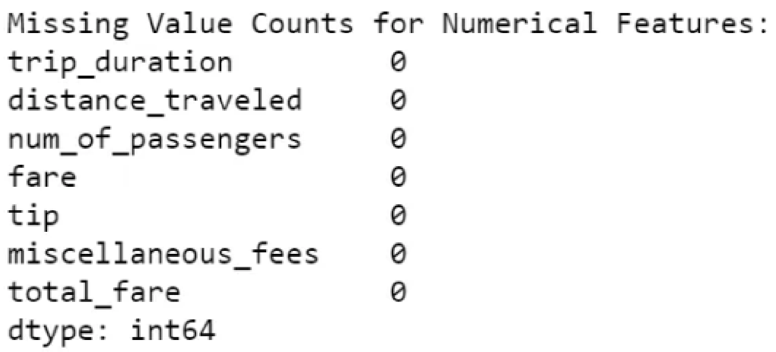
OBSERVATION AND RESULTS

4.1. RESULTS SNAPSHOTS



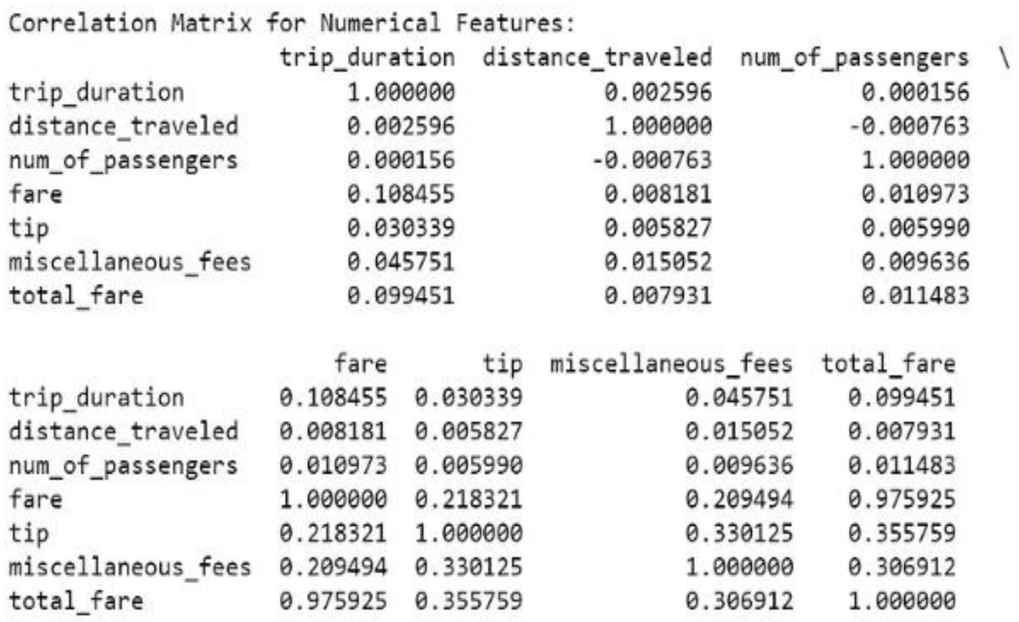
```
Summary statistics for numerical features:
       trip_duration  distance_traveled  num_of_passengers          fare  \
count  299534.000000      299534.000000      299534.000000  299534.000000
mean     1165.680417           5.717167           1.290985      69.736136
std      4727.006306         321.516385           0.927814      84.933388
min         0.000000           0.020000           0.000000       0.000000
25%       446.000000           1.950000           1.000000       0.000000
50%       706.000000           3.200000           1.000000      56.250000
75%      1097.000000           5.730000           1.000000      93.750000
max     86395.000000      145517.600000           9.000000    4466.250000

                 tip  miscellaneous_fees     total_fare
count  299534.000000       299534.000000  299534.000000
mean       13.030891           15.152971      89.453417
std        19.880049           12.590909     101.301805
min         0.000000           -0.500000       0.000000
25%         0.000000            6.000000       0.000000
50%         9.000000            9.750000      74.700000
75%        20.000000           26.450000     126.000000
max      2500.000000          435.000000    4472.250000
```

Fig 4.1 Data viewing



```
Missing Value Counts for Numerical Features:
trip_duration         0
distance_traveled     0
num_of_passengers     0
fare                  0
tip                   0
miscellaneous_fees    0
total_fare            0
dtype: int64
```

Fig 4.2 data information



```
Correlation Matrix for Numerical Features:
                   trip_duration  distance_traveled  num_of_passengers  \
trip_duration           1.000000           0.002596           0.000156
distance_traveled       0.002596           1.000000          -0.000763
num_of_passengers       0.000156          -0.000763           1.000000
fare                    0.108455           0.008181           0.010973
tip                     0.030339           0.005827           0.005990
miscellaneous_fees      0.045751           0.015052           0.009636
total_fare              0.099451           0.007931           0.011483

                       fare       tip  miscellaneous_fees  total_fare
trip_duration      0.108455  0.030339            0.045751    0.099451
distance_traveled  0.008181  0.005827            0.015052    0.007931
num_of_passengers  0.010973  0.005990            0.009636    0.011483
fare               1.000000  0.218321            0.209494    0.975925
tip                0.218321  1.000000            0.330125    0.355759
miscellaneous_fees 0.209494  0.330125            1.000000    0.306912
total_fare         0.975925  0.355759            0.306912    1.000000
```

Fig 4.3 correlation matrix information

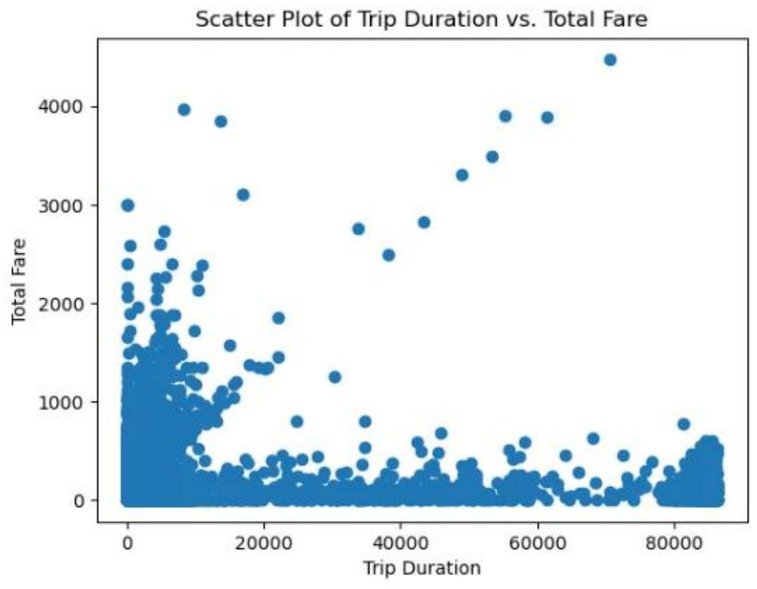


Fig 4.4 Scatter plot

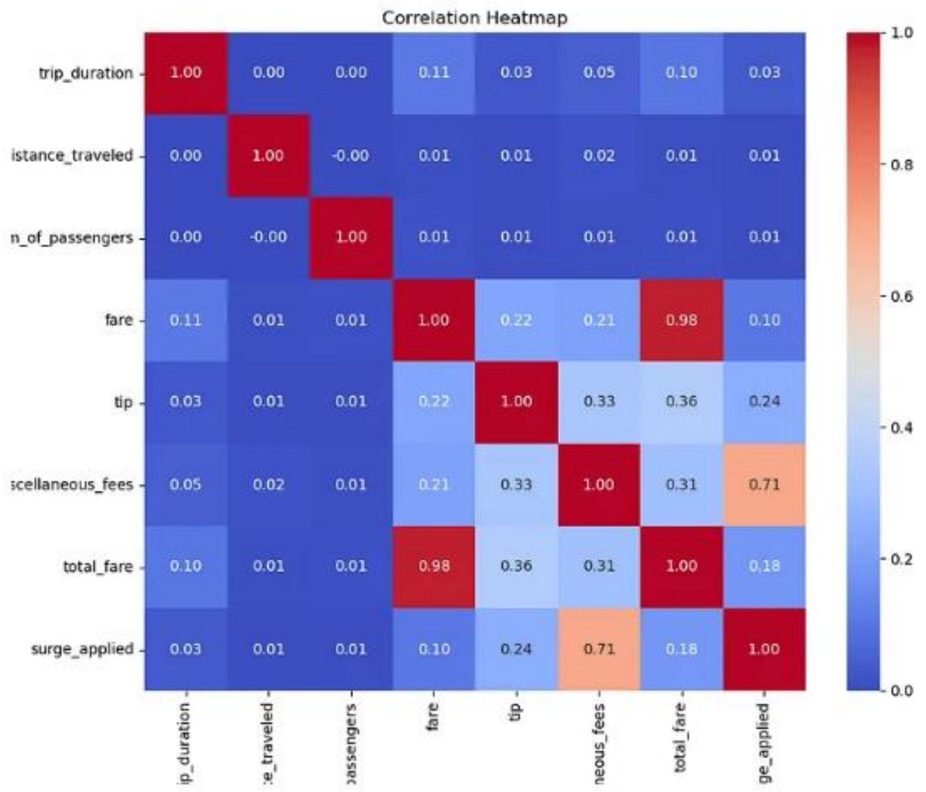


Fig 4.5 heatmap

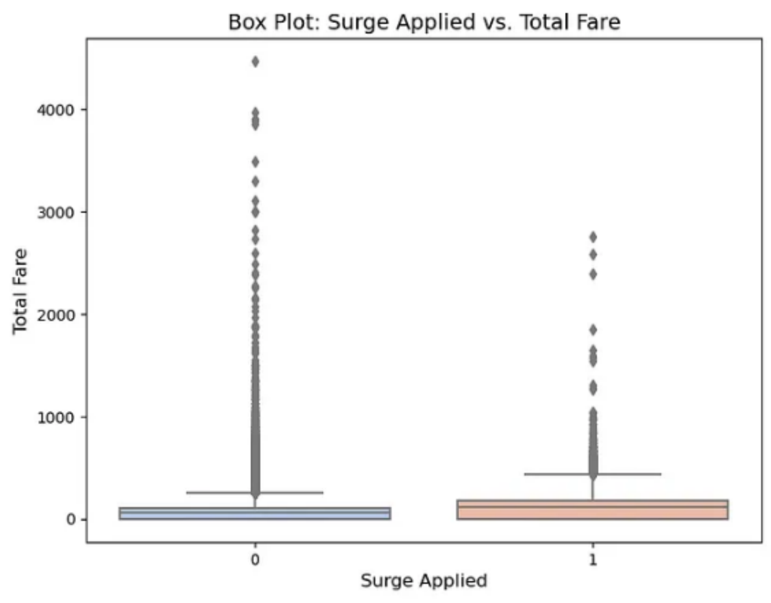
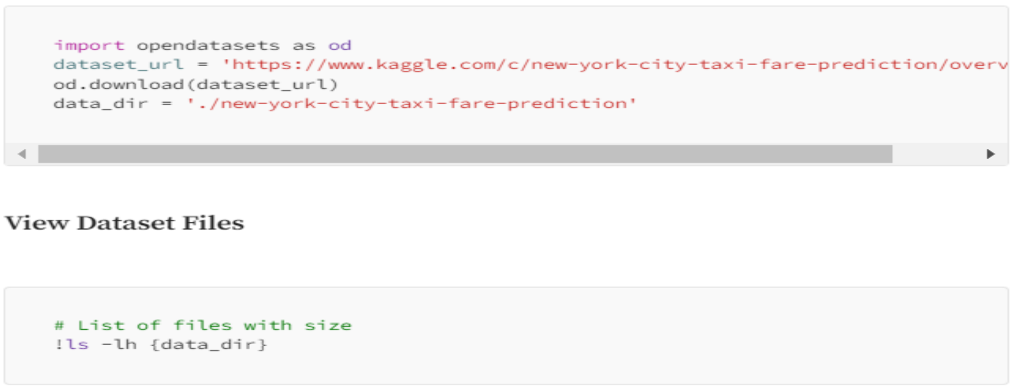


Fig 4.6 box plot

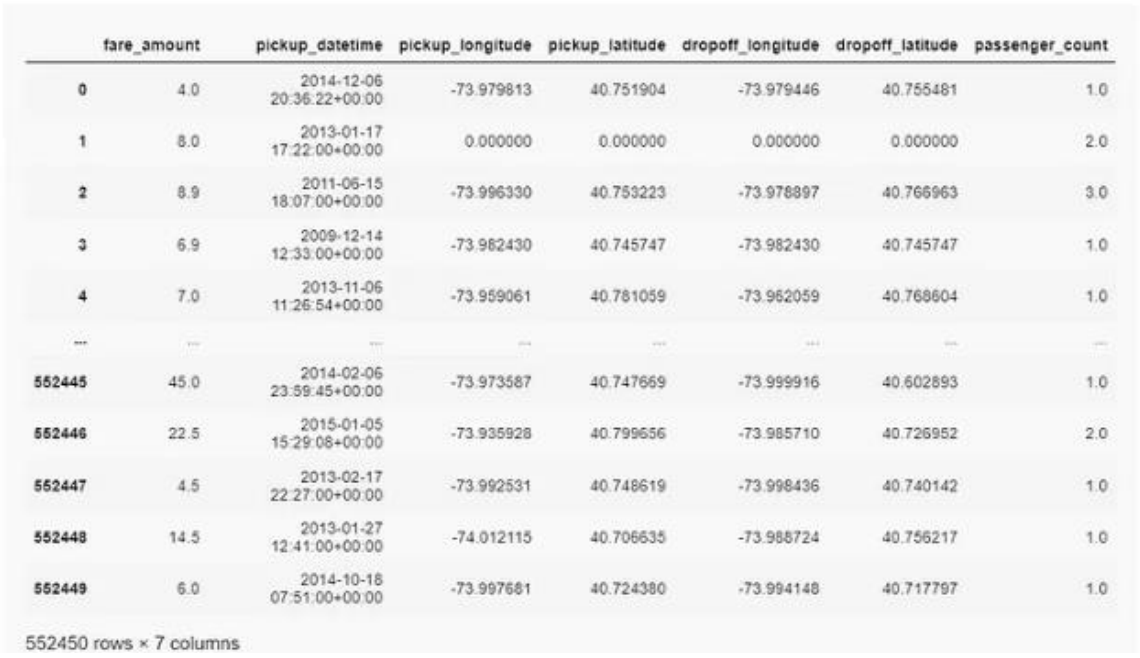


```python
import opendatasets as od
dataset_url = 'https://www.kaggle.com/c/new-york-city-taxi-fare-prediction/overv
od.download(dataset_url)
data_dir = './new-york-city-taxi-fare-prediction'
```

View Dataset Files

```python
# List of files with size
!ls -lh {data_dir}
```

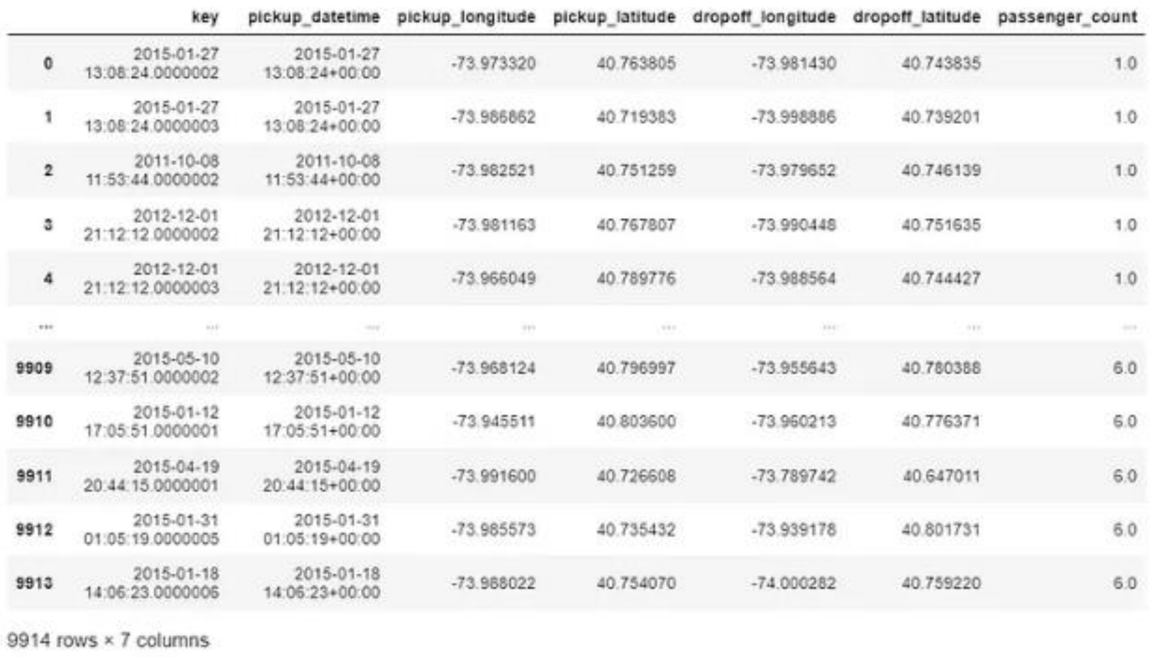
Fig 4.7 code to view files



	fare_amount	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
0	4.0	2014-12-06 20:36:22+00:00	-73.979813	40.751904	-73.979446	40.755481	1.0
1	8.0	2013-01-17 17:22:00+00:00	0.000000	0.000000	0.000000	0.000000	2.0
2	8.9	2011-06-15 18:07:00+00:00	-73.996330	40.753223	-73.978897	40.766963	3.0
3	6.9	2009-12-14 12:33:00+00:00	-73.982430	40.745747	-73.982430	40.745747	1.0
4	7.0	2013-11-06 11:26:54+00:00	-73.959061	40.781059	-73.962059	40.768604	1.0
...	...	...	...	...	...	...	...
552445	45.0	2014-02-06 23:59:45+00:00	-73.973587	40.747669	-73.999916	40.602893	1.0
552446	22.5	2015-01-05 15:29:08+00:00	-73.935928	40.799656	-73.985710	40.726952	2.0
552447	4.5	2013-02-17 22:27:00+00:00	-73.992531	40.748619	-73.998436	40.740142	1.0
552448	14.5	2013-01-27 12:41:00+00:00	-74.012115	40.706635	-73.988724	40.756217	1.0
552449	6.0	2014-10-18 07:51:00+00:00	-73.997681	40.724380	-73.994148	40.717797	1.0

552450 rows × 7 columns

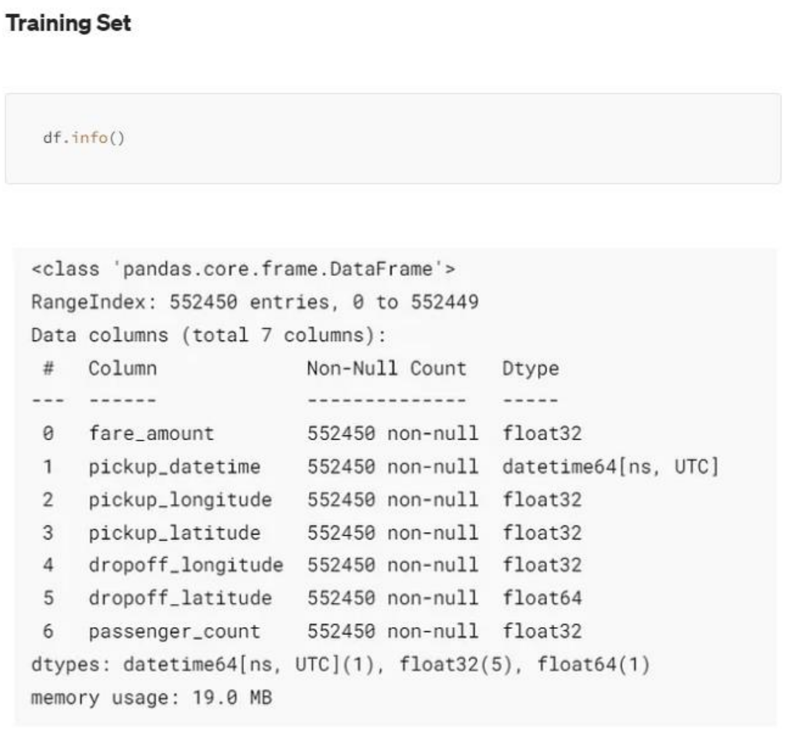
Fig 4.8 data display



	key	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
0	2015-01-27 13:08:24.0000002	2015-01-27 13:08:24+00:00	-73.973320	40.763805	-73.981430	40.743835	1.0
1	2015-01-27 13:08:24.0000003	2015-01-27 13:08:24+00:00	-73.986862	40.719383	-73.998886	40.739201	1.0
2	2011-10-08 11:53:44.0000002	2011-10-08 11:53:44+00:00	-73.982521	40.751259	-73.979652	40.746139	1.0
3	2012-12-01 21:12:12.0000002	2012-12-01 21:12:12+00:00	-73.981163	40.767807	-73.990448	40.751635	1.0
4	2012-12-01 21:12:12.0000003	2012-12-01 21:12:12+00:00	-73.966049	40.789776	-73.988564	40.744427	1.0
...	...	...	...	...	...	...	...
9909	2015-05-10 12:37:51.0000002	2015-05-10 12:37:51+00:00	-73.968124	40.796997	-73.955643	40.780388	6.0
9910	2015-01-12 17:05:51.0000001	2015-01-12 17:05:51+00:00	-73.945511	40.803600	-73.960213	40.776371	6.0
9911	2015-04-19 20:44:15.0000001	2015-04-19 20:44:15+00:00	-73.991600	40.726608	-73.789742	40.647011	6.0
9912	2015-01-31 01:05:19.0000005	2015-01-31 01:05:19+00:00	-73.985573	40.735432	-73.939178	40.801731	6.0
9913	2015-01-18 14:06:23.0000006	2015-01-18 14:06:23+00:00	-73.988022	40.754070	-74.000282	40.759220	6.0

9914 rows × 7 columns

Fig 4.9 duplicate data display

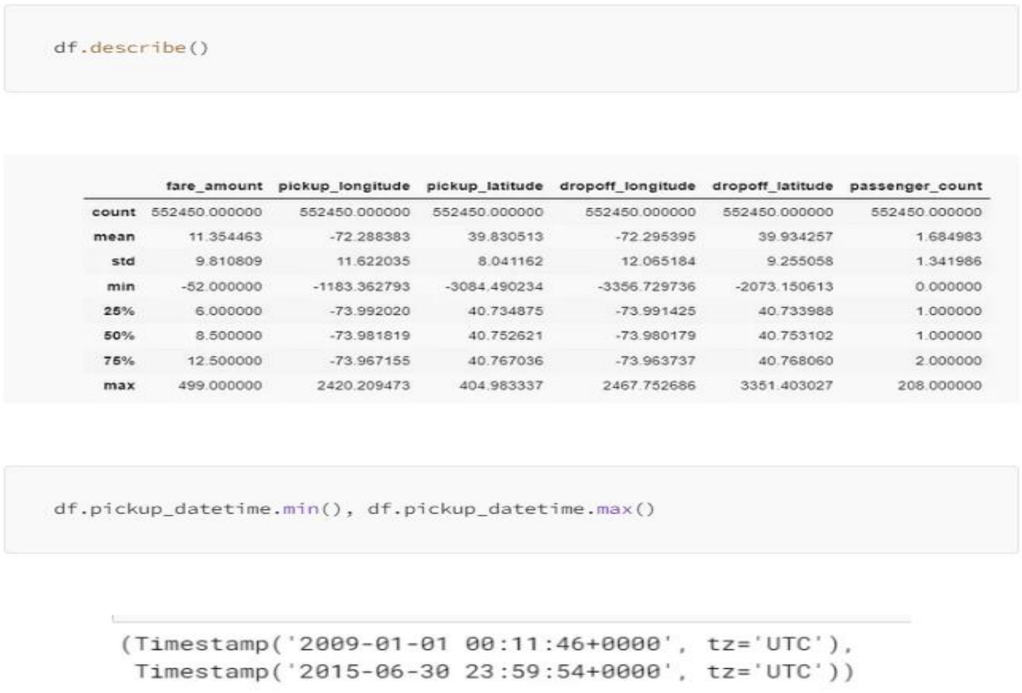


Training Set

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 552450 entries, 0 to 552449
Data columns (total 7 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   fare_amount        552450 non-null  float32
 1   pickup_datetime    552450 non-null  datetime64[ns, UTC]
 2   pickup_longitude   552450 non-null  float32
 3   pickup_latitude    552450 non-null  float32
 4   dropoff_longitude  552450 non-null  float32
 5   dropoff_latitude   552450 non-null  float64
 6   passenger_count    552450 non-null  float32
dtypes: datetime64[ns, UTC](1), float32(5), float64(1)
memory usage: 19.0 MB
```

Fig 4.10 dataset information



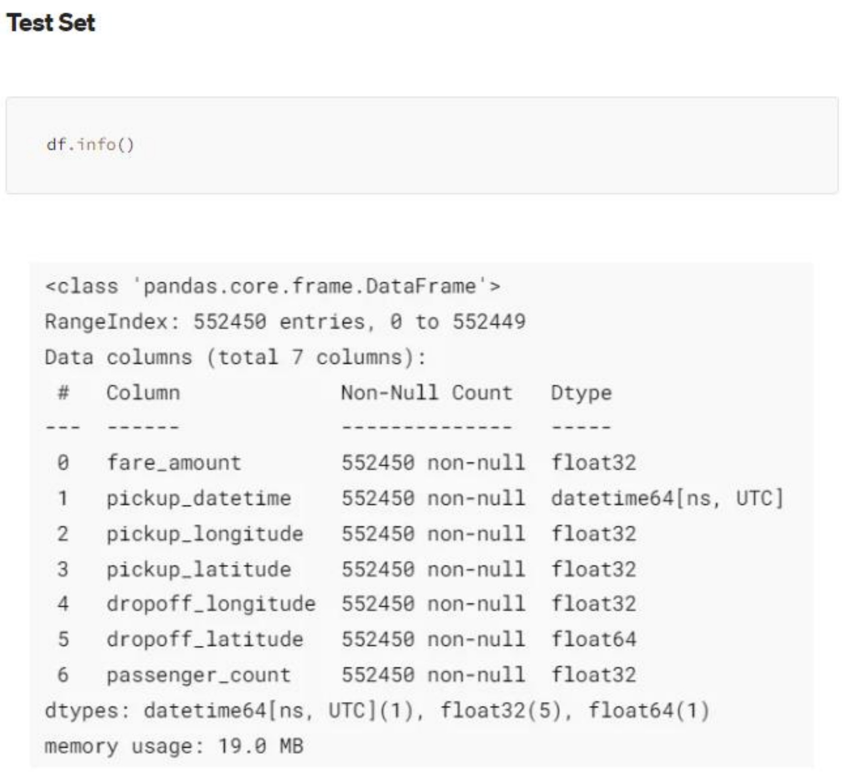
```
df.describe()
```

	fare_amount	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
count	552450.000000	552450.000000	552450.000000	552450.000000	552450.000000	552450.000000
mean	11.354463	-72.288383	39.830513	-72.295395	39.934257	1.684983
std	9.810809	11.622035	8.041162	12.065184	9.255058	1.341986
min	-52.000000	-1183.362793	-3084.490234	-3356.729736	-2073.150613	0.000000
25%	6.000000	-73.992020	40.734875	-73.991425	40.733988	1.000000
50%	8.500000	-73.981819	40.752621	-73.980179	40.753102	1.000000
75%	12.500000	-73.967155	40.767036	-73.963737	40.768060	2.000000
max	499.000000	2420.209473	404.983337	2467.752686	3351.403027	208.000000

```
df.pickup_datetime.min(), df.pickup_datetime.max()
```

```
(Timestamp('2009-01-01 00:11:46+0000', tz='UTC'),
 Timestamp('2015-06-30 23:59:54+0000', tz='UTC'))
```

Fig 4.11 dataset description and data time margin



Test Set

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 552450 entries, 0 to 552449
Data columns (total 7 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   fare_amount        552450 non-null  float32
 1   pickup_datetime    552450 non-null  datetime64[ns, UTC]
 2   pickup_longitude   552450 non-null  float32
 3   pickup_latitude    552450 non-null  float32
 4   dropoff_longitude  552450 non-null  float32
 5   dropoff_latitude   552450 non-null  float64
 6   passenger_count    552450 non-null  float32
dtypes: datetime64[ns, UTC](1), float32(5), float64(1)
memory usage: 19.0 MB
```

Fig 4.12 test dataset information

```
df.describe()
```

	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
count	9914.000000	9914.000000	9914.000000	9914.000000	9914.000000
mean	-73.976181	40.750954	-73.974945	40.751743	1.671273
std	0.042799	0.033542	0.039093	0.035435	1.278756
min	-74.252190	40.573143	-74.263245	40.568973	1.000000
25%	-73.992500	40.736125	-73.991249	40.735254	1.000000
50%	-73.982327	40.753052	-73.980015	40.754065	1.000000
75%	-73.968012	40.767113	-73.964062	40.768757	2.000000
max	-72.986534	41.709557	-72.990967	41.696683	6.000000

```
df.pickup_datetime.min(), df.pickup_datetime.max()
```

```
(Timestamp('2009-01-01 00:11:46+0000', tz='UTC'),
 Timestamp('2015-06-30 23:59:54+0000', tz='UTC'))
```

Fig 4.13 test dataset description

Distribution of fare amount

```
import matplotlib.pyplot as plt
import seaborn as sns
df[df.fare_amount<100].fare_amount.hist(bins = 100, figsize=(14,6));
plt.xlabel('fare $USD')
plt.title('Histogram')
```

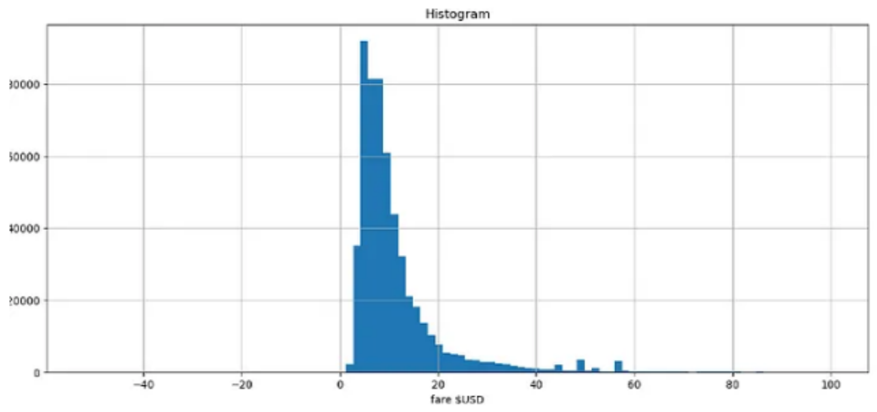


Fig 4.14 histogram plot of fare

Effect of passenger number on fare

```
plt.figure(figsize=(15,7))
plt.scatter(x=train['passenger_count'], y=train['fare_amount'], s=1.5)
plt.xlabel('No. of Passengers')
plt.ylabel('Fare')
```

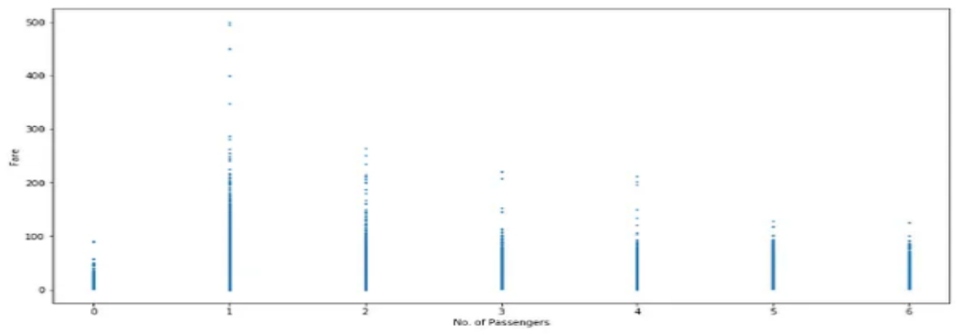


Fig 4.15 scatter plot on no. of passengers v/s fare

Effect of date and time on fare

```
plt.figure(figsize=(15,7))
plt.scatter(x=train['Date'], y=train['fare_amount'], s=1.5)
plt.xlabel('Date')
plt.ylabel('Fare')
```

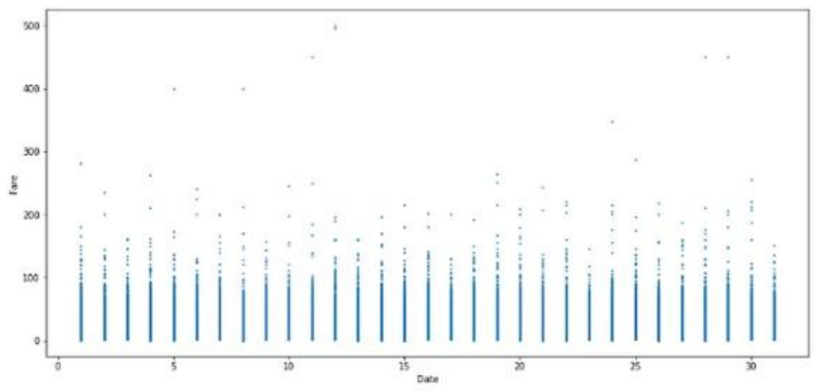


Fig 4.16 scatter plot wrt date and time

Effect of weekdays on fare

```
plt.figure(figsize=(15,7))
plt.scatter(x=train['Day of Week'], y=train['fare_amount'], s=1.5)
plt.xlabel('Day of Week')
plt.ylabel('Fare')
```

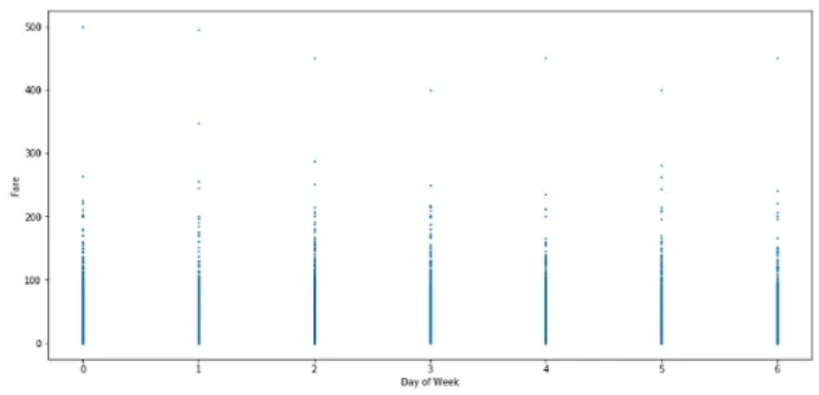


Fig 4.17 scatter plot on fare on weekdays

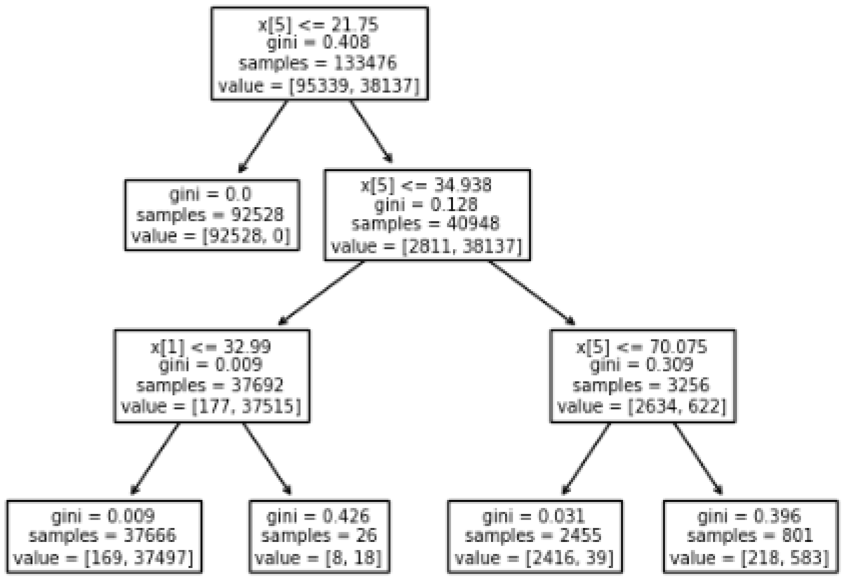


Fig 4.18 decision tree on train dataset

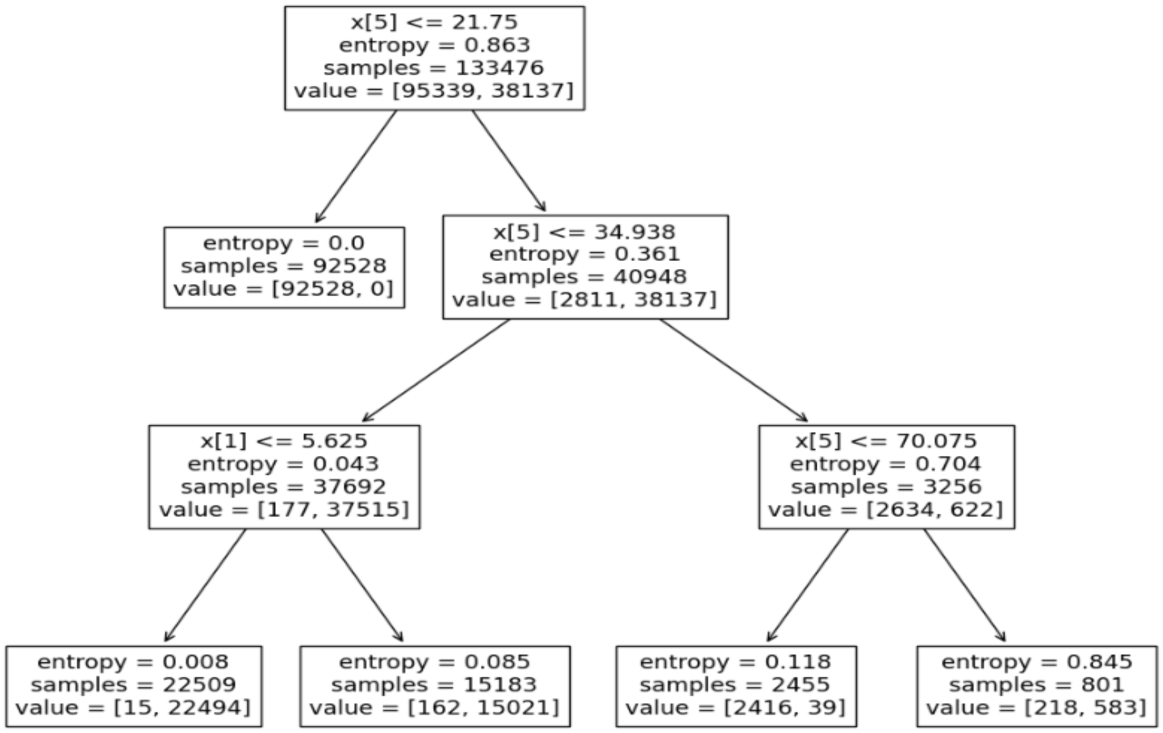


Fig 4.19 decision tree on test dataset

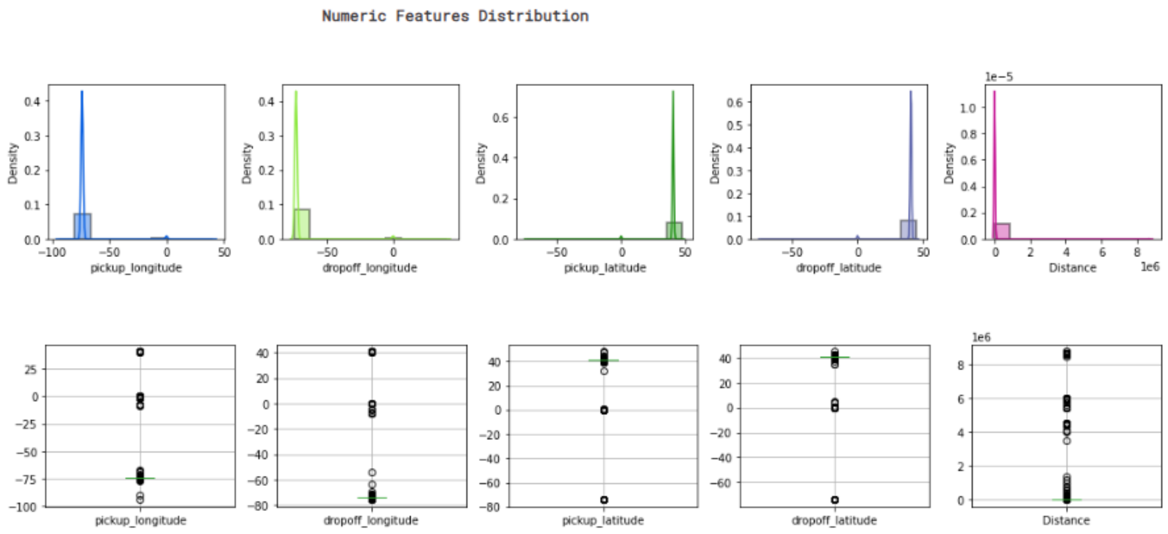


Fig 4.20 distribution plot on every column

CHAPTER-5

CONCLUSION AND FUTURE ENHANCEMENT

5.1. CONCLUSION

This study delves into refining the precision of Uber fare predictions by deeply understanding and leveraging the intricate web of users' travel habits, preferences, and the multifaceted dynamics of urban transportation. Central to our methodology is the strategic application of the correlation coefficient, a powerful statistical instrument that facilitates the identification of significant relationships between diverse travel variables such as journey distance, the time of the journey, pickup and drop-off locations, prevailing traffic conditions, and even weather patterns. By employing this approach, we move beyond superficial fare estimation techniques that only scratch the surface of the required journey details. Our model embarks on an exhaustive analysis, intricately comparing the specific attributes of a user's requested trip against a rich tapestry of historical travel data. This comprehensive comparative analysis enables the identification of deep-seated patterns and congruencies across an extensive array of user travels.

Our predictive model is built on the premise that travel behavior is not random but patterned and predictable. Users tend to exhibit distinct preferences, such as frequently traveled routes, favored times for travel (peak hours, late nights, early mornings), habitual destinations (workplaces, homes, social venues), and even specific inclinations towards certain areas of a city. Additionally, external factors like weather conditions and special events, which significantly influence travel decisions and fare rates, are incorporated into our analysis to ensure a holistic understanding of fare dynamics.

By leveraging the correlation coefficient in this nuanced manner, we unlock a more sophisticated, layered insight into travel behaviors. This not only enables the prediction of Uber fares with a higher degree of accuracy but also paves the way for a highly personalized user experience. Our model dynamically adapts to the unique patterns of individual users, offering fare estimates that are not just numbers but reflections of the users' travel history, preferences, and the complex interplay of external factors affecting travel costs.

This advanced predictive approach ensures that each fare estimate is a product of meticulous analysis, tailored to meet the specific requirements and preferences of the user. It enhances user trust and satisfaction by providing transparent, accurate, and personalized fare predictions, thereby fostering a more engaging and reliable relationship between the service and its users. Furthermore, this strategy offers valuable insights for optimizing operational efficiency and strategic planning for Uber, allowing for better resource allocation, pricing strategies, and ultimately, an enhanced service offering that resonates with the needs and expectations of the user base. Through this detailed and user-centric predictive model, we aim to redefine the standards of fare prediction in the ride-hailing industry, ensuring that every journey begins with trust, transparency, and a deep understanding of the user's needs.

5.2. FUTURE ENHANCEMENT

The exploration and integration of Neural Networks and Deep Learning into Uber fare prediction represent a transformative leap in how predictive models handle the intricacies of real-world data, particularly in capturing the subtleties of time and human behavior. Deep Learning's prowess, especially through the use of Recurrent Neural Networks (RNNs), has opened new avenues for creating more nuanced and dynamic fare prediction models. These models excel not only in understanding historical data but also in making sense of the sequential and temporal patterns that traditional analytical methods might overlook.

The temporal aspect is a critical component of fare prediction. It encompasses not just the obvious fluctuations in demand during different times of the day or days of the week but also more subtle seasonal trends and special event-related surges that can affect fare prices. For example, the demand for rides and, consequently, the fare rates during rush hour on a regular weekday, a major holiday, or a significant local event can vary dramatically. Deep Learning models, with their ability to process and learn from complex sequences of data, can predict these variations with a high degree of accuracy, thereby enabling more precise fare estimates.

Furthermore, the application of reinforcement learning strategies within the fare prediction context marks a significant shift towards a more experimental and adaptive approach to model training. By intentionally incorporating scenarios where users are presented with fare predictions that may deviate from the expected norms, the system can gather valuable feedback on user preferences and elasticity in terms of pricing acceptance. This approach not only enriches the dataset with diverse user reactions but also challenges the model to refine its predictions in ways that traditional static models cannot achieve.

This continuous cycle of prediction, observation, and adaptation helps in building a fare prediction system that is deeply attuned to the nuances of human behavior and temporal dynamics. Such a system can cater to the evolving needs and preferences of users, offering them more personalized and accurate fare estimates. It also allows for a more strategic approach to pricing and service delivery, enabling Uber to optimize its operations in line with demand patterns and customer expectations.

In essence, the fusion of Neural Networks and Deep Learning with time-series analysis and reinforcement learning methodologies heralds a new era of fare prediction models. These models are not just reactive but are proactive in learning from the temporal and behavioral data patterns, offering insights that go beyond conventional analytics. This not only enhances the accuracy of fare predictions but also contributes to a more efficient, responsive, and user-centric transportation service.

CHAPTER-6

REFERENCE

- Recommendation systems with python by Rounak Banik, Published by packet publishing Ltd. First publication 2018.

- A note on pearson correlation coefficient as a metric of similarity in recommender systemPublisher: IEEE Authors:Leily Sheugh; Sasan H. Alizade

- S.K. Lee, Y.H. Cho and S.H. Kim, "Collaborative filtering with ordinal scale-based implicit ratings for uber fare prediction", Information Sciences, vol. 180, no. 11, pp. 2142- 2155, 2010.

- K. Choi, D. Yoo, G. Kim and Y. Suh, "A hybrid online-product recommendation system: combining implicit rating-based Collaborative filtering and sequential pattern analysis", Electronic Commerce Research and Applications, vol. 11, no. 4, pp. 309-317, 2012.