Major Project- II Report

on

Vaycay Ventures

submitted in partial fulfillment of the requirements

for the award of the degree

of

Bachelor of Technology

in

Computer Science and Engineering

By

Ritik Chandel

Enrollment No. A60205219012

Sahil Tomar

Enrollment No. A60205219036

Priyal Bansal

Enrollment No. A60205219054

Vaishnavi Singh

Enrollment No. A60205219066

Under the guidance of

Dr. Varun Mishra

Assistant Professor

Department of Computer Science and Engineering
Amity School of Engineering & Technology
Amity University Madhya Pradesh, Gwalior
June 2023

Department of Computer Science and Engineering
Amity School of Engineering and Technology
Amity University Madhya Pradesh, Gwalior

DECLARATION

We, **Ritik Chandel, Sahil Tomar, Vaishnavi Singh, and Priyal Bansal**, students of Bachelor of Technology in Computer Science and Engineering hereby declare that the Project report entitled **"Vaycay Ventures"** which is submitted by us to the Department of Computer Science and Engineering, Amity School of Engineering & Technology, Amity University Madhya Pradesh, in partial fulfillment of the requirement for the award of the Degree of Bachelor of Technology in Computer Science and Engineering, has not been previously formed the basis for the award of any degree, diploma or other similar title or recognition. Our supervisor, HOD, and the Institute should not be held for full or partial violation of copyrights if found at any stage of our degree.

Ritik Chandel
Date: (Enrollment No. – A60205219012)
Sahil Tomar
(Enrollment No. – A60205219036)
Priyal Bansal
(Enrollment No. – A60205219054)
Vaishnavi Singh
(Enrollment No. – A60205219066)

Department of Computer Science and Engineering
Amity School of Engineering and Technology
Amity University Madhya Pradesh, Gwalior

CERTIFICATE

This is to certify that the major project entitled "**Vaycay Ventures**" by **Ritik Chandel (Enrollment No A60205219012), Sahil Tomar (Enrollment No A60205219036), Priyal Bansal (Enrollment No A60205219054), and Vaishnavi Singh (Enrollment No A60205219066)** is a bonafide record of project carried out by them under my supervision and guidance in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineering in the Department of Computer Science and Engineering, Amity School of Engineering and Technology, Amity University Madhya Pradesh, Gwalior. Neither this project nor any part of it has been submitted for any degree or academic award elsewhere.

Date:

(Dr. Varun Mishra)
Assistant Professor
Supervisor

External Examiner

(Dr. Hemant Kumar Soni)
Head of the Department

ACKNOWLEDGEMENT

We are very much thankful to our honorable Vice Chancellor **Lt Gen. V. K. Sharma AVSM (Retd)** for allowing us to carry out our project. We would also like to thank **Prof. (Dr.) M. P. Kaushik**, Pro-Vice Chancellor, Amity University Madhya Pradesh for his support.

We extend our sincere thanks to **Dr. Vivek Jaglan,** HOI Amity School of Engineering and Technology, Amity University Madhya Pradesh, Gwalior for his guidance and support for the selection of appropriate labs for our project. We would also like to thank **Dr. Hemant Kumar Soni,** Head of Department (CSE), for his kind concern throughout the project.

We are also very grateful to **Dr. Varun Mishra** and **Dr. Vivek Parashar**, Assistant Professor's, Department of Computer Science and Engineering, Amity School of Engineering and Technology, Amity University Madhya Pradesh our supervisor and Co-supervisor respectively, for their constant guidance and encouragement provided in this Endeavour.

We are also thankful to the whole staff of ASET, AUMP for teaching us every single minute in their respective fields.

At last, we thank everyone who contributed to this work in all doable manners. Our heartfelt thanks to families and friends for their kind help and suggestions.

Ritik Chandel

Date: (Enrollment No. – A60205219012)

Sahil Tomar

(Enrollment No. – A60205219036)

Priyal Bansal

(Enrollment No. – A60205219054)

Vaishnavi Singh

(Enrollment No. – A60205219066)

ABSTRACT

The report focuses on the development of a travel advisor web application using React js and Python. The purpose of the web application is to provide users with a personalized travel experience by recommending destinations, hotels, and activities based on their preferences.

The web application uses machine learning techniques to analyze previous travel data and generate estimated travel expenditure for future years. This feature allows users to plan their travel budget effectively and make informed decisions based on their financial capabilities.

The report outlines the architecture and design of the web application, including the front-end and back-end technologies used. It also provides an overview of the machine learning algorithms and techniques used to generate the estimated travel expenditure.

The report highlights the benefits of using a machine learning-based approach in travel planning, such as personalized recommendations and cost-effective travel planning. It also discusses the challenges and limitations of the approach, such as the need for a large amount of data and the potential biases in the data.

Overall, the report provides a detailed description of the travel advisor web application, its features, and the underlying technologies used. It also discusses the potential impact of the application on the travel industry and its users.

LIST OF FIGURES

CONTENTS

CHAPTER 1

INTRODUCTION

Building a Travel Advisor WebApp using ReactJs, Python. Using Machine Learning to Generate Estimated Expenditure for Future Years based on previous data:

1.1 Background and Motivation

1.1.1 Background:

Travel planning can be a daunting task, with so many variables to consider, such as destination, hotel, transportation, and activities. With the rise of online booking platforms, travelers are overwhelmed with choices, and it can be challenging to make informed decisions. Furthermore, travelers often struggle to plan their travel budget effectively, leading to unexpected expenses and financial stress.

1.1.2 Motivation:

The motivation behind this project is to develop a web application that simplifies the travel planning process and provides users with a personalized travel experience. By leveraging machine learning techniques, the application can generate estimated travel expenditure for future years, allowing users to plan their travel budget effectively and make informed decisions based on their financial capabilities.

The project aims to address the following challenges faced by travelers:

1. The overwhelming number of travel options available.
2. The lack of personalized recommendations based on individual preferences.
3. The difficulty in planning travel expenses accurately.

The web application will provide a user-friendly platform for travelers to explore destinations, hotels, and activities based on their preferences. The machine learning-based expenditure prediction model will provide users with an estimate of their future travel expenses, considering various factors such as seasonality, location, and travel history.

1.2 Objective:

The objectives of the project are as follows:

1. Develop a travel advisor web application that provides users with personalized travel recommendations based on their preferences.
2. Implement a machine learning-based expenditure prediction model that generates estimated travel expenses for future years based on previous travel data.
3. Provide a user-friendly interface for users to explore and compare destinations, hotels, and activities.
4. Allow users to filter and sort recommendations based on various factors, such as location, budget, and travel dates.
5. Incorporate a secure payment gateway for users to book travel services directly through the web application.
6. Evaluate the performance of the machine learning-based expenditure prediction model and gather user feedback to improve the application.
7. Ensure the web application is responsive, fast, and scalable to handle a large volume of users and data.

1.3 Overview of Report

The report outlines the development of a travel advisor web application that utilizes machine learning techniques to generate estimated travel expenses for future years based on previous travel data. The web application is built using React.js and Python, and it provides users with personalized travel recommendations based on their preferences.

The report begins with an introduction to the background and motivation behind the project, followed by the objectives and scope of the project. A literature review is then presented, highlighting existing travel advisor applications and the use of machine learning in travel planning.

The methodology section provides a detailed description of the system architecture and design, including the front-end and back-end technologies used. The machine learning algorithms and techniques used to generate estimated travel expenditure are also discussed.

The results and analysis section presents the developed travel advisor web application and its features. The performance of the machine learning-based expenditure prediction model is analysed, and user feedback is evaluated.

The discussion section highlights the benefits of the travel advisor web application, including personalized recommendations and cost-effective travel planning. The challenges and limitations of the approach are also discussed, such as the need for a large amount of data and the potential biases in the data.

The conclusion section summarizes the report, highlighting the contributions of the project and the potential impact of the travel advisor web application. Finally, references are provided for sources cited in the report, and appendices may be included for detailed technical specifications or code snippets.

Overall, the report provides a comprehensive overview of the travel advisor web application, its features, and the underlying technologies used. It also discusses the potential impact of the application on the travel industry and its users.

CHAPTER 2

LITERATURE REVIEW

2.1 Overview of existing travel advisor applications

Existing travel advisor applications provide users with various features and functionalities that simplify the travel planning process. These applications include online travel agencies (OTAs), travel review websites, and mobile applications. Online travel agencies (OTAs) such as Expedia, Booking.com, and TripAdvisor are popular choices for travellers looking to book flights, hotels, and activities. These platforms offer a wide range of travel options, user reviews, and price comparison tools. However, users may be overwhelmed with the number of options available and may find it challenging to make informed decisions.

Travel review websites such as TripAdvisor and Yelp provide users with user-generated reviews and ratings for destinations, hotels, and activities. These websites allow users to filter and sort recommendations based on various factors, such as location, price, and popularity. However, the reviews may be biased or fake, and the recommendations may not be personalized to the user's preferences.

Mobile applications such as Hopper and Kayak provide users with real-time flight and hotel prices, alerts, and recommendations. These applications use machine learning techniques to analyse historical data and provide personalized recommendations. However, the recommendations may not be comprehensive, and users may still need to use other platforms to book travel services.

Overall, while existing travel advisor applications provide users with various features and functionalities, there is still room for improvement. The travel advisor web application developed in this project aims to provide a comprehensive and personalized travel planning experience for users, leveraging machine learning techniques to generate accurate and reliable expenditure estimates.

2.2 Introduction to machine learning in travel planning

Machine learning is a subset of artificial intelligence that allows computers to learn and improve from experience without being explicitly programmed. In recent years, machine learning has been applied to various fields, including travel planning. Machine learning algorithms can analyse large amounts of data and provide personalized recommendations, making them a valuable tool for travel planning.

Machine learning techniques can be applied to various aspects of travel planning, including destination recommendations, flight and hotel pricing, and itinerary planning. For example, machine learning algorithms can analyse a user's previous travel history, preferences, and online behavior to provide personalized destination recommendations. These algorithms can also analyse historical flight and hotel prices to predict future prices and provide cost-effective recommendations.

One of the key advantages of using machine learning in travel planning is the ability to provide personalized recommendations. Traditional travel planning methods such as online travel agencies (OTAs) and travel review websites provide users with a vast amount of options but may not consider the user's preferences and past travel history. Machine learning algorithms can analyse a user's previous travel behavior and provide personalized recommendations based on their interests, budget, and travel history.

Another advantage of using machine learning in travel planning is the ability to analyse and predict travel expenses accurately. Machine learning algorithms can analyse a user's previous travel expenses and predict future expenses, providing users with a better understanding of their travel budget. This can help users make more informed decisions and plan their travel more effectively.

However, there are also challenges associated with using machine learning in travel planning. One of the main challenges is the need for a large amount of data to train the algorithms. Data biases can also affect the accuracy of the predictions. Additionally, the algorithms may not be able to account for unexpected events, such as weather disruptions or political instability.

Overall, machine learning has the potential to revolutionize the travel industry by providing users with personalized recommendations and cost-effective travel planning. The travel advisor web application developed in this project leverages machine learning algorithms to provide accurate and reliable expenditure estimates, enhancing the user experience and simplifying the travel planning process.

2.3 Review of relevant literature on machine learning-based travel planning

Machine learning-based travel planning is a rapidly evolving field with a growing body of literature exploring its potential benefits and limitations. One area of research has focused on developing recommendation systems that provide personalized travel recommendations based on a user's travel history, preferences, and online behavior. A study by Raza et al. (2019) developed a hybrid recommendation system that combined collaborative filtering and content-based filtering to provide personalized hotel recommendations. The results showed that the hybrid system outperformed traditional recommendation systems in terms of accuracy and relevance.

Another area of research has explored the use of machine learning in predicting travel expenses. A study by Li et al. (2020) developed a machine learning model to predict the total cost of a trip based on historical data. The model utilized a combination of regression and classification techniques and was trained on a dataset of over 15,000 trips. The results showed that the model was able to accurately predict the total cost of a trip, providing users with a better understanding of their travel budget.

In addition to recommendation systems and expense prediction, machine learning has also been applied to itinerary planning. A study by Zhang et al. (2020) developed a machine learning-based itinerary planning system that optimized travel routes based on a user's preferences and constraints. The results showed that the system was able to generate optimized itineraries that satisfied user preferences and constraints.

Despite the potential benefits of machine learning-based travel planning, there are also concerns regarding data privacy and bias. A study by Zhang et al. (2021) explored the ethical considerations of machine learning in the travel industry, highlighting the need for transparency and accountability in algorithmic decision-making.

Overall, the literature on machine learning-based travel planning highlights the potential benefits of personalized recommendations, accurate expense prediction, and optimized itinerary planning. However, there are also challenges regarding data privacy and bias that must be addressed. The travel advisor web application developed in this project aims to address these challenges by providing transparent and reliable machine learning-based expenditure estimates.

CHAPTER 3

METHODOLOGY

3.1 Technologies used in the development of the web application.

The technology used in the development of Web Application is React and Python. React is a JavaScript library used for building user interfaces for single-page web applications. It allows for the creation of reusable UI components and provides a high level of flexibility and efficiency. React makes use of a virtual DOM, which allows for fast updates to the UI without having to reload the entire page. This results in a smoother and more responsive user experience. React can also be combined with other JavaScript libraries or frameworks to create powerful web applications.

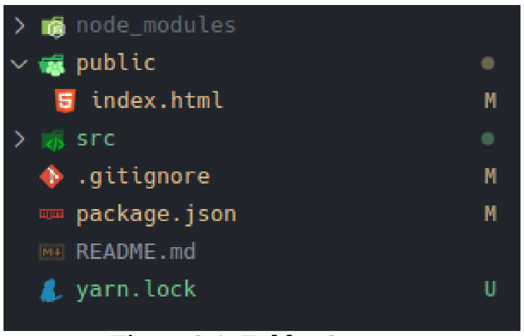
Python is a popular programming language used for a variety of purposes, including web development and machine learning. Python's simplicity and readability make it easy to use and maintain, and it offers a wide range of libraries and frameworks for different purposes. For web development, Python can be used with frameworks like Django or Flask to create server-side applications that interact with the client-side user interface. For machine learning, Python offers a wide range of libraries and frameworks like Scikit-learn, TensorFlow, and Keras, which make it easy to develop and train machine learning models.

In the context of the travel advisor web application mentioned in the project, React was used for building the user interface, allowing for the creation of responsive and interactive pages that are easy to navigate. Python was used as the backend software for developing the machine learning models that generate the estimated travel expenditures for future years based on previous data. The Python machine learning models were integrated with the React frontend using APIs or web services, which allow for communication between the frontend and backend of the web application.

Overall, the combination of React and Python allows for the creation of powerful and efficient web applications with a responsive user interface and advanced machine learning capabilities.

3.1.1 Frontend Development:

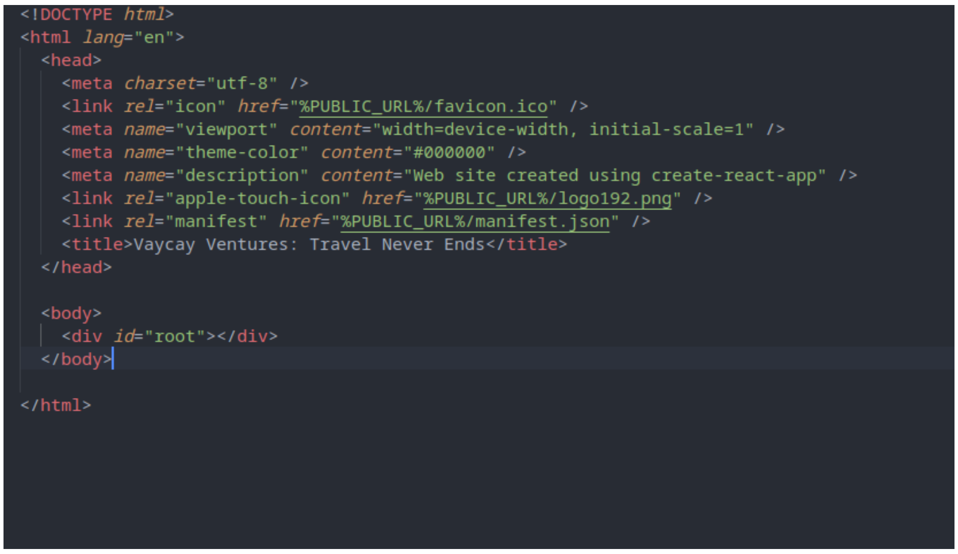
Our website containing Single Page interface comprised of various small components like navbar, hero, ScrollToTop, Testimonials, About us, Places, Services, etc. These components are combined in the App.jsx file to give web Page a proper stuctures and elegant look. The Folder structure looks like the below image:



```
> node_modules
v public              ●
    index.html        M
> src                 ●
  .gitignore          M
  package.json        M
  README.md
  yarn.lock           U
```

Figure 3.1: Folder Structure

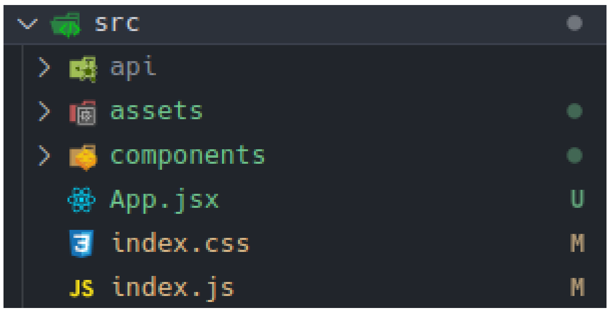
The Project containing three base folders namely src, public, npm_module folder. The npm modules folder containing all the libraries needed to run the React application. The public folder contains the index.html file which is the has the <div> tag with id root. React application using this index.html root to create entire application.



```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta name="description" content="Web site created using create-react-app" />
    <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
    <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
    <title>Vaycay Ventures: Travel Never Ends</title>
  </head>

  <body>
    <div id="root"></div>
  </body>

</html>
```

Figure: 3.2: Index.html

The main working directory of our application is the src folder. All the components, img files, logos, and CSS files are in the src folder. This serves as the main functional body of the entire application. The src folder consists of assests, api, and components directories.



```
src
  > api
  > assets
  > components
    App.jsx        U
    index.css      M
    index.js       M
```

Figure 3.3: src folder

The api directory consist of all the API calls we have made for the information like google map, travel_advisor APIs. The assets contain all the image files required to create the Web Application. The Components directly is where we have created and structure the web app component. The src folder itself contain 3 files namely App.jsx, Index.js, and index.css, these files serve as a base for our application. The Final output of our application looks like below:

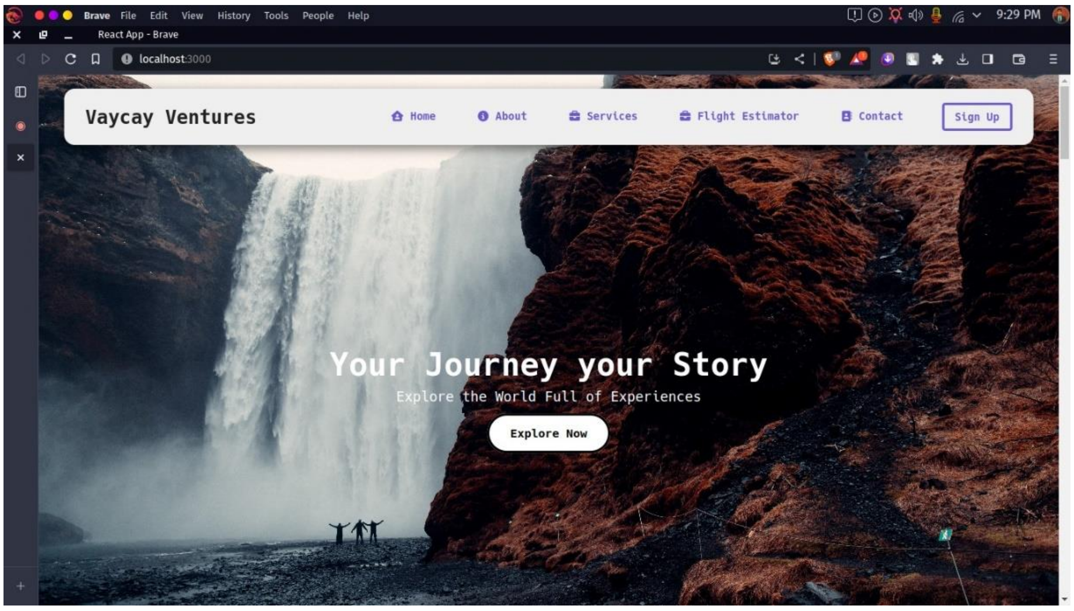


Figure: 3.4: Final output

We have used scroll reveal animation effect, Scroll reveal animation is a type of animation effect that is triggered when a user scrolls down a webpage. This type of animation can be used to add visual interest and engagement to a website and can help draw attention to important elements on a page. Scroll reveal animation can be an effective way to enhance the user experience of a React-based website and can be achieved using various libraries and techniques.

Apart from using scroll reveal effect we have also used styled-components library, Styled Components is a popular library used for styling React components. It allows developers to write CSS code that is scoped to a particular component, making it easier to manage the styling of a React application.

With Styled Components, developers can create reusable and customizable components with dynamic styling properties. Instead of writing CSS in separate files or inline styles, Styled Components allows developers to write CSS directly in the JavaScript file of a component. This allows for better organization and maintainability of CSS code. Styled Components is a powerful library that allows for easy and maintainable styling of React components, and it offers many advanced features such as theming and media queries.

The Application also contains a Scroll to Top button at the end of the web app, which is used to take user to the top with just a click. The code below describe the scrollToTop button:

The code defines a React component called ScrollToTop that renders a clickable button in the lower right corner of the screen. When the button is clicked, the window is scrolled to the top of the page.

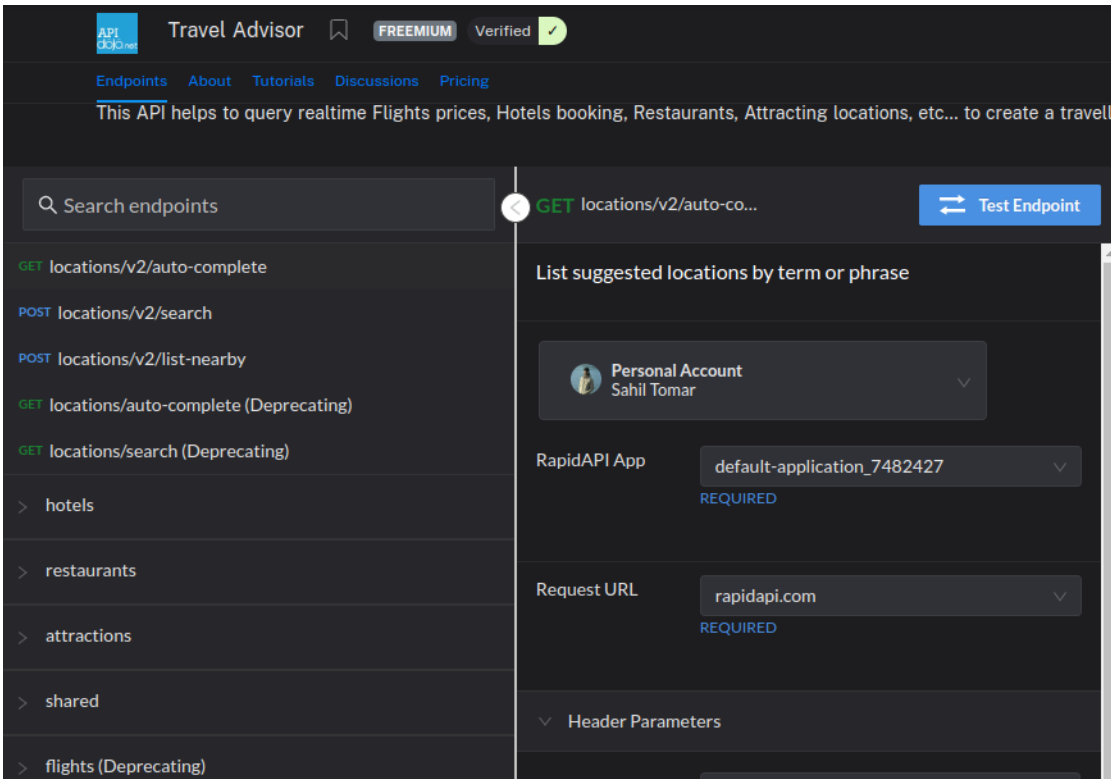
The useState hook is used to define the scrollState variable, which is initially set to false. The ToTop function is defined to scroll the window to the top of the page when called.

An event listener is added to the window object to listen for the scroll event. When the user has scrolled down more than 200 pixels, the scrollState variable is set to true. Otherwise, it is set to false. This allows the button to appear only when the user has scrolled down a certain amount.

The ToTop component is defined using the styled-components library. It is a div element that is displayed as a block when the scrollState is true, and as none when the scrollState is false. It is positioned fixed at the bottom right of the screen, has a cursor pointer and z-index of 10, and has a border radius of 2rem and a background color of #1900ff39. The padding is set to 1rem and it contains an image element with a height of 1.5rem.

Apart from the application we have also created an google map page which is used to get the hotels, restaurants and attractions near the user, or according the searched city or place. The component is used provide real time information about the places with the help of google map and Travel Advisor APIs. The Travel Advisor API, we have used to fetch the data about restaurants, hotels and Attraction for anywhere in the world. The travel advisor API is a web-based application programming interface that provides access to data and functionalities related to travel planning, such as flight and hotel booking, itinerary creation, and destination recommendations.

These APIs can be used by developers to integrate travel-related features and services into their own applications, websites, or platforms. Travel advisor APIs typically provide data and functionalities related to flights, hotels, car rentals, and attractions, and may also offer tools for managing travel bookings, reservations, and payments.



This API helps to query realtime Flights prices, Hotels booking, Restaurants, Attracting locations, etc... to create a travell

GET locations/v2/auto-complete
POST locations/v2/search
POST locations/v2/list-nearby
GET locations/auto-complete (Deprecating)
GET locations/search (Deprecating)

GET locations/v2/auto-co...

List suggested locations by term or phrase

Personal Account
Sahil Tomar

RapidAPI App: default-application_7482427 REQUIRED

Request URL: rapidapi.com REQUIRED

Figure 3.5: The Travel Advisor API @RapidAPI

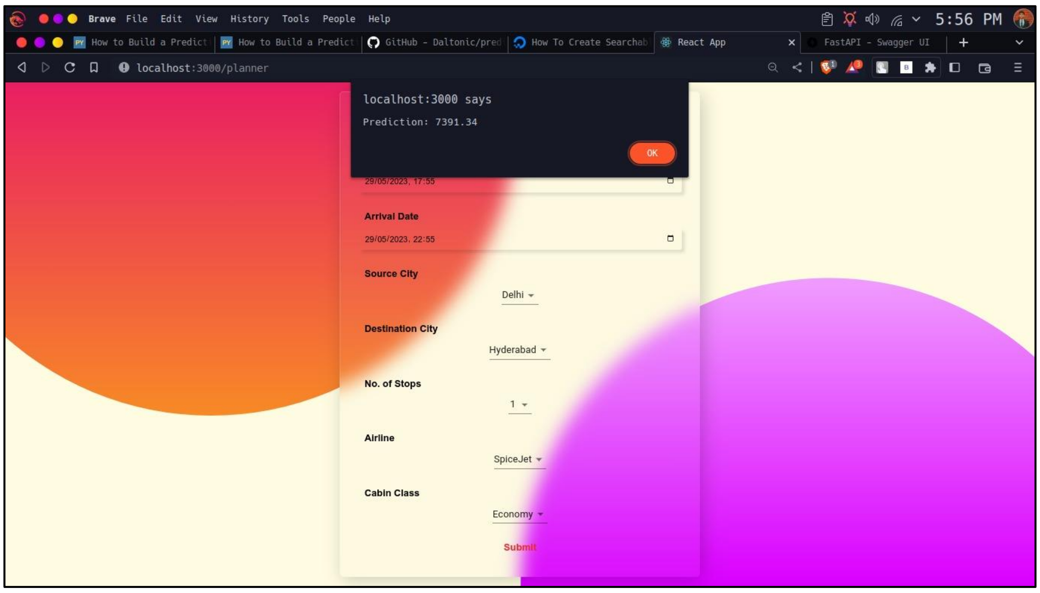
As we can see the API gives many routes to use query the real time data, we have used restaurants, hotels and attractions, methods to get the information and display it on our web app. This gives our web app not only to provide a customizable Plan for the customer trip but also provide full information about the location which he/she is going, with the help of Google Map API. The Google Maps JavaScript API is a web-based API provided by Google that allows developers to integrate Google Maps and location-based features into their web applications. It provides a set of tools and functionalities for creating customized maps, adding markers, overlays, and other graphical elements, and manipulating map features such as zoom and pan.

Some of the key features of the Google Maps JavaScript API include:

1. Interactive maps: Developers can add interactive maps to their web applications and customize them with different map types, markers, and overlays.
2. Geolocation: The API provides tools for determining the user's location and using that information to display maps and location-based data.
3. Directions: The API includes functionality for calculating and displaying driving, walking, and public transit directions between two or more locations.
4. Places: Developers can use the API to search for and display information about places of interest, such as restaurants, museums, and parks.
5. Street View: The API includes functionality for displaying Street View imagery and allowing users to explore street-level views of locations.

These functionalities along with the React.js makes it easy to use Google Map API with the we fetch from travel advisor API. We have created a card for each restaurant, hotel, and attraction fetch from the travel advisor API to showcase on the map itself. once the use clicks on the card. It shows the details of the place.

To give our Platform a next edge touch we have created and deploy our own ML API endpoint known “predict”. This “predict” API takes Data from user about the Flight Tickets, like, Source Place, Destination Place, Airline Company, Cabin Class, number of stops, etc to predict the Estimated Flight price for user’s Trip. We have created a component in React using Glass morphism effect. Which is basically a form taking inputs from user and showing the estimated flight price of the trip.



```
localhost:3000 says
Prediction: 7391.34
```

29/05/2023, 17:55

Arrival Date
29/05/2023, 22:55

Source City
Delhi

Destination City
Hyderabad

No. of Stops
1

Airline
SpiceJet

Cabin Class
Economy

Figure 3.6: Output of the Prediction API

Figure 3.6 shows the output of the Prediction API when giving the Inputs of the form to “predict” the endpoint in the backend and then process it using the machine learning model which we can talk about in the next section. The function which transmits the post request to the backend server is given below:

Axios is a popular JavaScript library used for making HTTP requests from the browser. It provides a simple and intuitive API for performing various types of requests, including GET, POST, PUT, DELETE, etc. It supports promises, allowing you to handle responses asynchronously using .then() and .catch().

```
axios.post('http://127.0.0.1:8000/predict', params)
    .then((res) => {
      const data = res.data.data
      const msg = `Prediction: ${data.prediction}`
      alert(msg)
      reset()
    })
    .catch((error) => alert(`Error: ${error.message}`))
```

Axios simplifies the process of making HTTP requests by providing a consistent interface across different browsers and offering features like request and response interception, automatic JSON parsing, and error handling. It can be used in both Node.js and browser environments.

In the given code, Axios is used to make a POST request to the specified URL and handle the response and potential errors. The response data is then processed and displayed to the user using an alert box.

1. axios.post('http://127.0.0.1:8000/predict', params): This line initiates a POST request to the specified URL (http://127.0.0.1:8000/predict). The params object contains the data to be sent along with the request, which is typically in the form of request parameters or a request body.
2. .then((res) **=> { ... })**: The .then() method is a promise-based function that handles the response returned by the server. It takes a callback function as an argument, which is executed when the request is successful. In this case, the callback receives the **res** (response) object.

3. const data = res.data.data: The response data is extracted from the **res** object using the data property. The specific data received from the server is accessed using the data property again. The response data can vary depending on the server's implementation.
4. const msg = Prediction: ${data.prediction}``: The received data is used to construct a message to be displayed in an alert box. The prediction property is extracted from the data object and included in the message.
5. alert(msg): The constructed message is displayed in an alert box.
6. reset(): A function named reset() is called, presumably to reset the form or clear any input fields after the request is completed.
7. .catch((error) => alert**(**Error: ${error.message})): The .catch() method is used to handle any errors that occur during the request. If an error occurs, the provided callback function is executed, and an alert box displaying the error message is shown.

3.2 Machine learning algorithms and techniques used to generate estimated travel expenditure.

We get the data from Kaggle and perform the data transformation part in Azure Data Factory (ADF). The Main function of the ADF in our project is to collect data and perform the Data transformation steps in real time.

3.2.1. What is Azure Data Factory?

In the area of big data, relational, non-relational, and other storage methods are frequently used to store raw, unorganized data. However, raw data alone lacks the necessary context or meaning to offer analysts, data scientists, or business decision makers actionable insights.

In order to transform these massive stores of raw data into usable business insights, big data requires a service that can orchestrate and operationalize processes. For these challenging hybrid extract-transform-load (ETL), extract-load-transform (ELT), and data integration projects, Azure Data Factory is a managed cloud solution.

3.2.1.1 Usage scenarios

Consider a gaming corporation that gathers petabytes of game logs generated by cloud-based games. The business intends to examine these logs in order to learn more about consumer preferences, demographics, and usage patterns. Additionally, it seeks to find chances for up-sells and cross-sells, create intriguing new features, spur business expansion, and improve the customer experience.

The business must leverage reference data from an on-premises data store, such as customer information, game information, and marketing campaign information, to analyze these logs. The business intends to use this on-site data store data in conjunction with additional log data that it has stored in a cloud data store.

The combined data will be processed using a cloud Spark cluster (Azure HDInsight) to extract insights, and the converted data will be published into a cloud data warehouse like Azure Synapse Analytics so that a report can be simply built on top of it. They want to monitor and manage this workflow regularly and automate it. In addition, they want it to run whenever files are dropped into a blob store container.

The platform that handles these data scenarios is Azure Data Factory. You may develop data-driven processes for orchestrating data movement and transforming data at scale using the cloud-based ETL and data integration service. You may design and plan data-driven workflows (also known as pipelines) that can ingest data from various data repositories using Azure Data Factory. Using visual data flows or computing services like Azure HDInsight Hadoop, Azure Databricks, and Azure SQL Database, you can create intricate ETL processes that transform data.

Additionally, you can make your modified data available for consumption by business intelligence (BI) applications by publishing it to data repositories like Azure Synapse Analytics. In the end, raw data may be arranged into useful data stores and data lakes with Azure Data Factory for better business decisions.

3.2.2 How does it function?

A complete end-to-end platform for data engineers is offered by the collection of integrated platforms that make up Data Factory. This visual manual offers a comprehensive overview of the entire Data Factory architecture.

3.2.2.1 Connect with and gather-

Enterprises have access to a variety of data types that are spread across numerous on-premises, cloud, and semi-structured, unstructured, and unstructured sources, and that arrive at variable rates and intervals.

Connecting to all necessary sources of data and processing, such as databases, file shares, software-as-a-service (SaaS) services, and FTP web services, is the initial stage in creating an information production system. The data must then be transferred as needed to a central location for processing.

Enterprises who don't use Data Factory must create their own bespoke data movement components or services to integrate these data sources and processing. Such systems are expensive, difficult to maintain, and difficult to integrate. Additionally, they frequently lack the controls, enterprise-grade monitoring, and alerts that a fully managed service may provide.

With Data Factory, you can transport data from both on-premises and cloud source data stores to a centralized data store in the cloud for additional analysis using the Copy Activity in a data pipeline. For instance, you can gather data in Azure Data Lake Storage and later modify it using a compute service for Azure Data Lake Analytics. Additionally, you can gather data in Azure Blob storage and afterwards transform it using a cluster of Azure HDInsight Hadoop.

3.2.2.2 Change and improve-

Use ADF mapping data flows to process or transform the gathered data once it is present in a centralized data store in the cloud. Without knowing anything about Spark clusters or Spark programming, data engineers can create and maintain data transformation graphs that run on Spark.

ADF supports external activities for running your transformations on compute services like HDInsight Hadoop, Spark, Data Lake Analytics, and Machine Learning if you want to program transformations by hand.

3.2.2.3 CI/CD and publish-

The CI/CD of your data pipelines utilising Azure DevOps and GitHub is fully supported by Data Factory. As a result, you can create and deliver your ETL processes gradually before publishing the final product. Load the data into Azure Data Warehouse, Azure SQL Database, Azure Cosmos DB, or whichever analytics engine your business users may point to from their business intelligence tools once the raw data has been transformed into a consumable form that is suitable for business usage.

3.2.2.4 Monitor-

Monitor the scheduled activities and pipelines for success and failure rates once you have successfully designed and launched your data integration pipeline, generating business value from refined data. Pipeline monitoring is supported natively by Azure Data Factory via Azure Monitor, Azure Monitor logs, API, PowerShell, and health panels on the Azure portal.

3.2.2.5 high-level ideas

One or more instances of Azure Data Factory (also known as data factories) may be included in an Azure subscription. The following essential elements make up Azure Data Factory:

- Dataset
- Activities
- Pipelines
- Linked service
- Integration Runtimes
- Data Flows

Together, these elements create the framework on which you can build data-driven workflows that include phases for data movement and transformation.

3.2.2.6 Pipeline

There may be one or more pipelines in a data factory. A pipeline is a logical collection of tasks that carry out a single task. The actions in a pipeline work together to complete a task. A pipeline might include a series of operations that ingest data from an Azure blob before partitioning it using a Hive query on an HDInsight cluster.

The advantage of this is that you can manage the activities via the pipeline rather than controlling them separately. A pipeline's actions can run independently in parallel or coupled together to work sequentially.

3.2.2.7 Mapping data flow-

You may convert any size of data by creating and maintaining graphs of data transformation logic. You may create a reusable library of data transformation processes, and then use your ADF pipelines to scale those processes out. On a Spark cluster that spins up and spins down as needed, Data Factory will run your logic. Cluster management and upkeep are not required at any time.

3.2.2.8 Activity

Activities are a pipeline's processing stage. To copy data from one data store to another, for instance, utilize a copy activity. Like this, you might use a Hive activity to convert or analyze your data by running a Hive query on an Azure HDInsight cluster. Three different sorts of activities are supported by Data Factory: data mobility, data transformation, and control activities.

3.2.2.9 Dataset-

The data you want to use in your activities as inputs or outputs is simply pointed to or referenced by data structures known as datasets within data stores.

3.2.2.10 Linked services-

Like connection strings, which specify the connection details required for Data Factory to connect to external resources, Linked Services define the connection information. Consider it this way: a dataset reflects the data's structure, while a linked service specifies the relationship to the data source. For instance, a connection string is specified by an Azure Storage-linked service to connect to the Azure Storage account. An Azure blob dataset also identifies the folder and blob container where the data is stored.

In Data Factory, Linked Services are utilized for two purposes:

1. To represent a data store, such as an Azure blob storage account, a file share, a SQL Server database, an Oracle database, etc. See the copy action article for a list of supported data storage.
2. To depict a computing device that can host an activity throughout execution. For instance, an HDInsight Hadoop cluster is used to conduct the HDInsight Hive activity.

Visit the transform data article for a list of supported compute environments and transformation operations.

3.2.2.11 Integration Runtime-

An activity in Data Factory specifies the action to be taken. A compute service or a target data storage are defined by a linked service. The connection between the activity and associated Services is made possible by an integration runtime. It gives information about the computing environment that the associated service or activity will use to run or dispatch the activity from. By doing so, the activity can be carried out in a fashion that is as efficient as feasible while still adhering to security and compliance requirements in the area that is the closest to the target data storage or compute service.

3.2.2.12 Triggers-

When a pipeline execution needs to begin, triggers serve as the processing unit that makes that call. There are various kinds of triggers for various kinds of events.

3.2.2.13 Pipeline runs-

An example of a pipeline execution is a pipeline run. Typically, pipeline runs are created by supplying arguments to the pipeline-defined parameters. Manually passing the arguments is also possible within the trigger definition.

3.2.2.14 Parameters-

Key-value pairs of read-only configuration make up parameters. The pipeline defines the parameters. The run context produced by a trigger or a pipeline that was manually executed passes the arguments for the stated parameters during execution. The parameter values are consumed by pipeline activities.

A dataset is a reusable and referenceable item as well as a strongly typed parameter. A dataset can be referenced by an activity, and that activity can use the properties that the dataset specification defines.

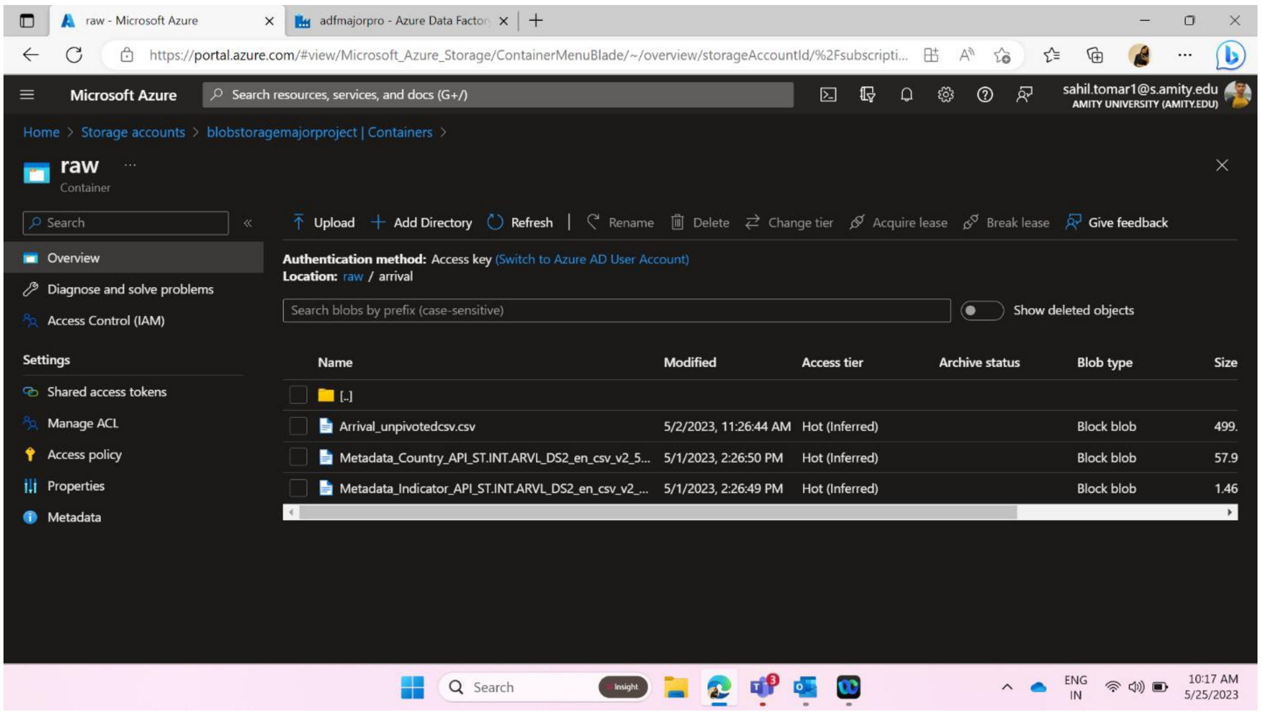
A strongly typed parameter known as a linked service additionally provides connection details to either a data storage or a computing environment. It can also be referenced and reused.

3.2.2.15 Control Flow-

Defining parameters at the pipeline level, branching, chaining actions in a sequence, and passing arguments while invoking the pipeline on demand or in response to a trigger are all examples of control flow. Additionally, it contains looping containers and custom-state passing, or For-each iterators.

3.2.2.16 Variable-

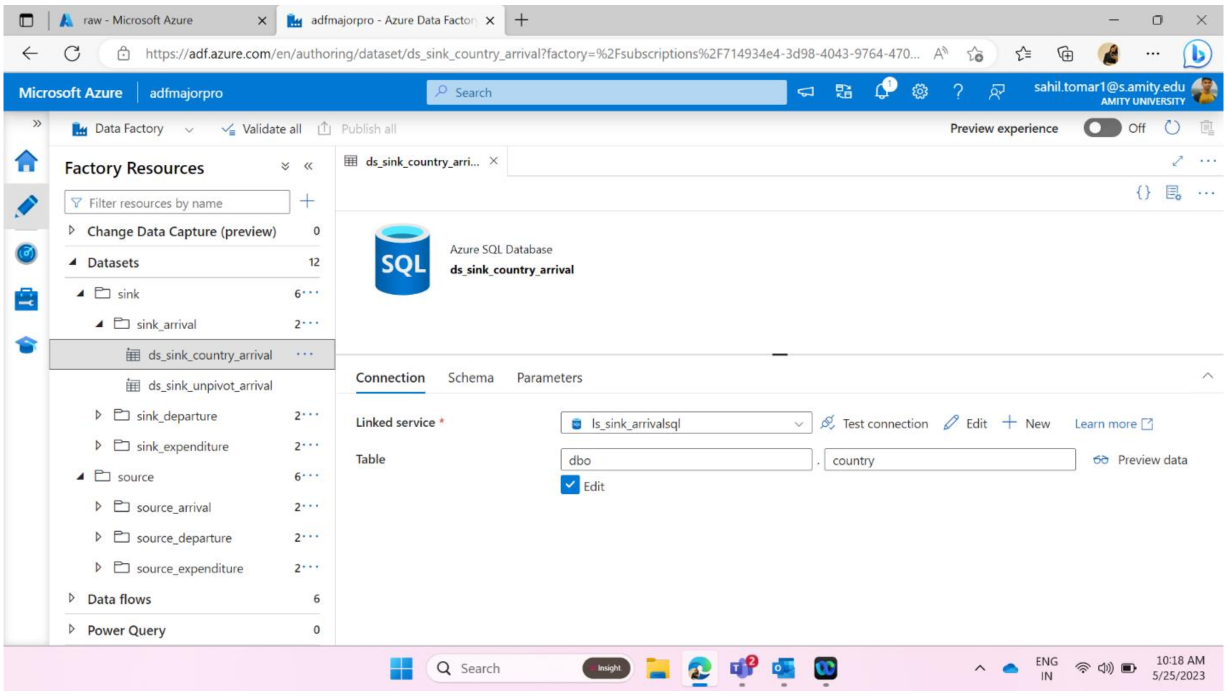
In addition to being used with parameters to facilitate value transmission across pipelines, data flows, and other operations, variables can also be used inside of pipelines to hold transient values.


raw - Microsoft Azure
adfmajorpro - Azure Data Factory
https://portal.azure.com/#view/Microsoft_Azure_Storage/ContainerMenuBlade/~/overview/storageAccountId/%2Fsubscripti...
Microsoft Azure
Search resources, services, and docs (G+/)
sahil.tomar1@s.amity.edu
AMITY UNIVERSITY (AMITY.EDU)
Home > Storage accounts > blobstoragemajorproject | Containers >
raw
Container
Search
Overview
Diagnose and solve problems
Access Control (IAM)
Settings
Shared access tokens
Manage ACL
Access policy
Properties
Metadata
Upload
Add Directory
Refresh
Rename
Delete
Change tier
Acquire lease
Break lease
Give feedback
Search blobs by prefix (case-sensitive)
Show deleted objects
Search
ENG IN
10:17 AM
5/25/2023

Authentication method: Access key (Switch to Azure AD User Account)
Location: raw / arrival

Name	Modified	Access tier	Archive status	Blob type	Size
[..]					
Arrival_unpivotedcsv.csv	5/2/2023, 11:26:44 AM	Hot (Inferred)		Block blob	499.
Metadata_Country_API_ST.INT.ARVL_DS2_en_csv_v2_5...	5/1/2023, 2:26:50 PM	Hot (Inferred)		Block blob	57.9
Metadata_Indicator_API_ST.INT.ARVL_DS2_en_csv_v2_...	5/1/2023, 2:26:49 PM	Hot (Inferred)		Block blob	1.46

Figure 3.7: Storage account file uploads (arrival, departure, and expenditure)

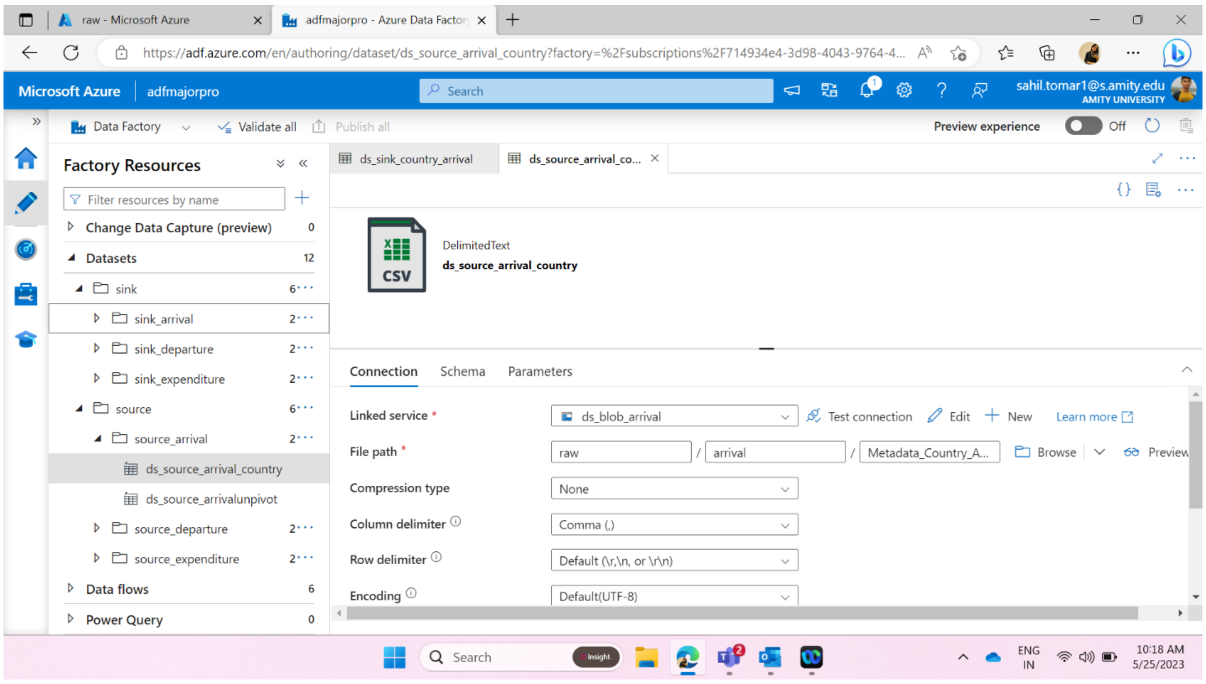

Azure SQL Database
ds_sink_country_arrival

Factory Resources
- Change Data Capture (preview) 0
- Datasets 12
  - sink 6
    - sink_arrival 2
      - ds_sink_country_arrival
      - ds_sink_unpivot_arrival
    - sink_departure 2
    - sink_expenditure 2
  - source 6
    - source_arrival 2
    - source_departure 2
    - source_expenditure 2
- Data flows 6
- Power Query 0

Connection | Schema | Parameters

Linked service *	ls_sink_arrivalsql
Table	dbo . country
	☑ Edit

Figure 3.8: Dataset Sink


DelimitedText
ds_source_arrival_country

Factory Resources
- Change Data Capture (preview) 0
- Datasets 12
  - sink 6
    - sink_arrival 2
    - sink_departure 2
    - sink_expenditure 2
  - source 6
    - source_arrival 2
      - ds_source_arrival_country
      - ds_source_arrivalunpivot
    - source_departure 2
    - source_expenditure 2
- Data flows 6
- Power Query 0

Connection | Schema | Parameters

Linked service *	ds_blob_arrival
File path *	raw / arrival / Metadata_Country_A...
Compression type	None
Column delimiter	Comma (,)
Row delimiter	Default (\r,\n, or \r\n)
Encoding	Default(UTF-8)

Figure 3.9: Dataset Source

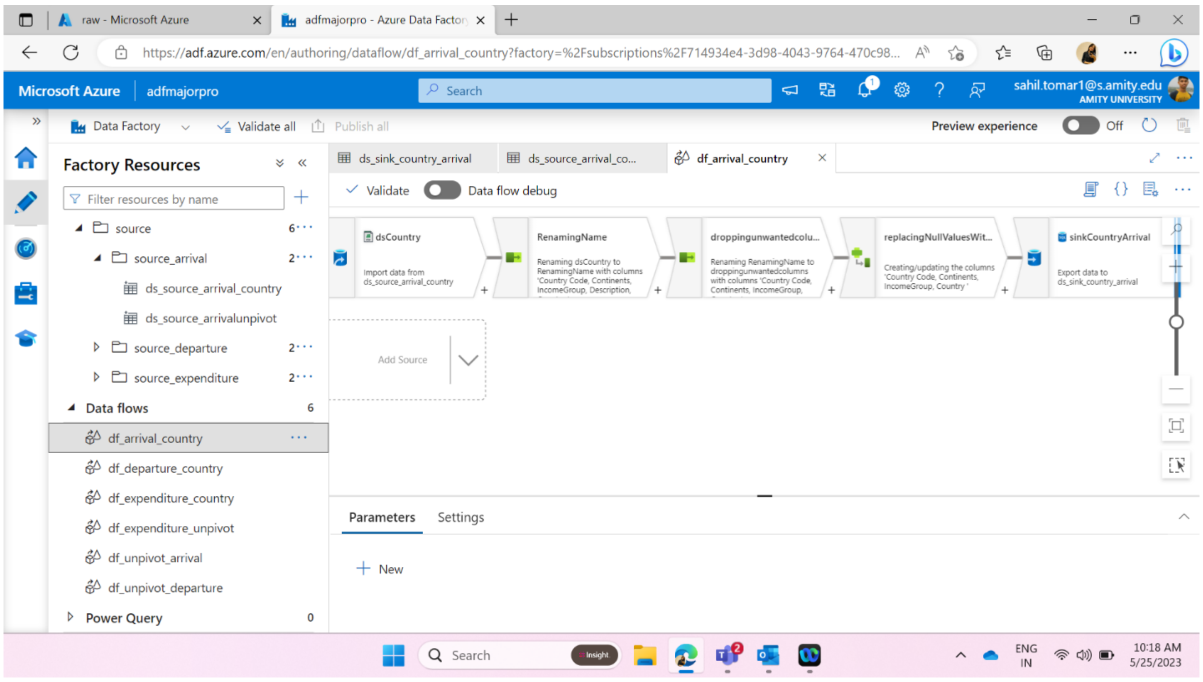


Figure 3.10: Dataflows

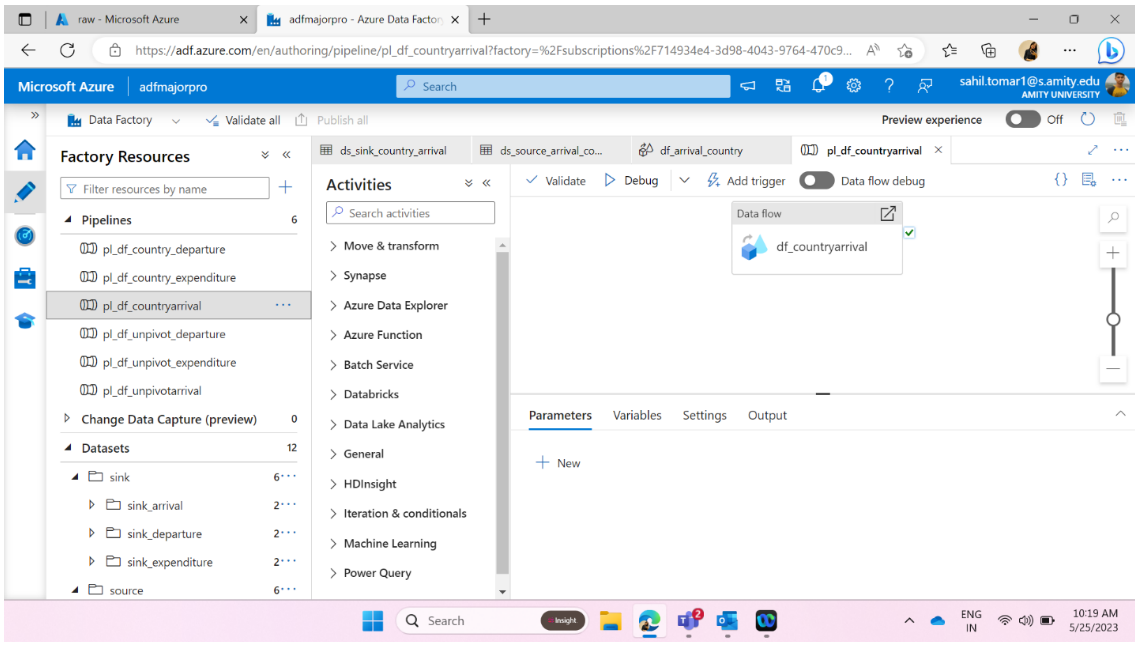
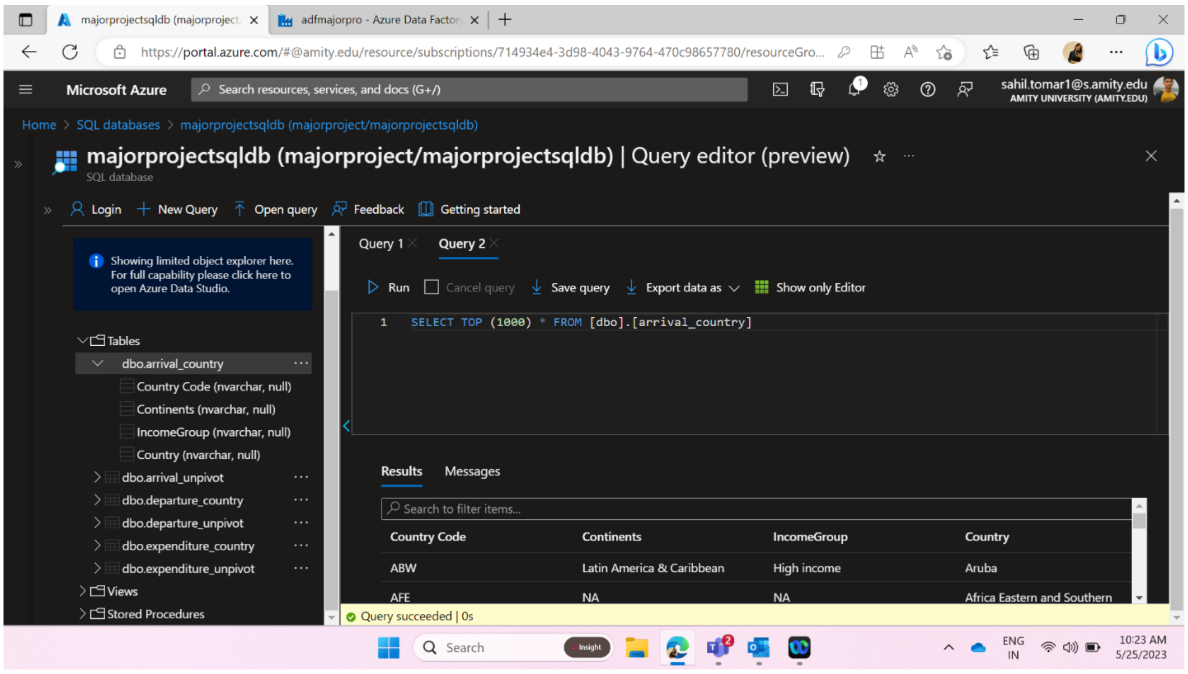


Figure 3.11: Pipelines



```
SELECT TOP (1000) * FROM [dbo].[arrival_country]
```

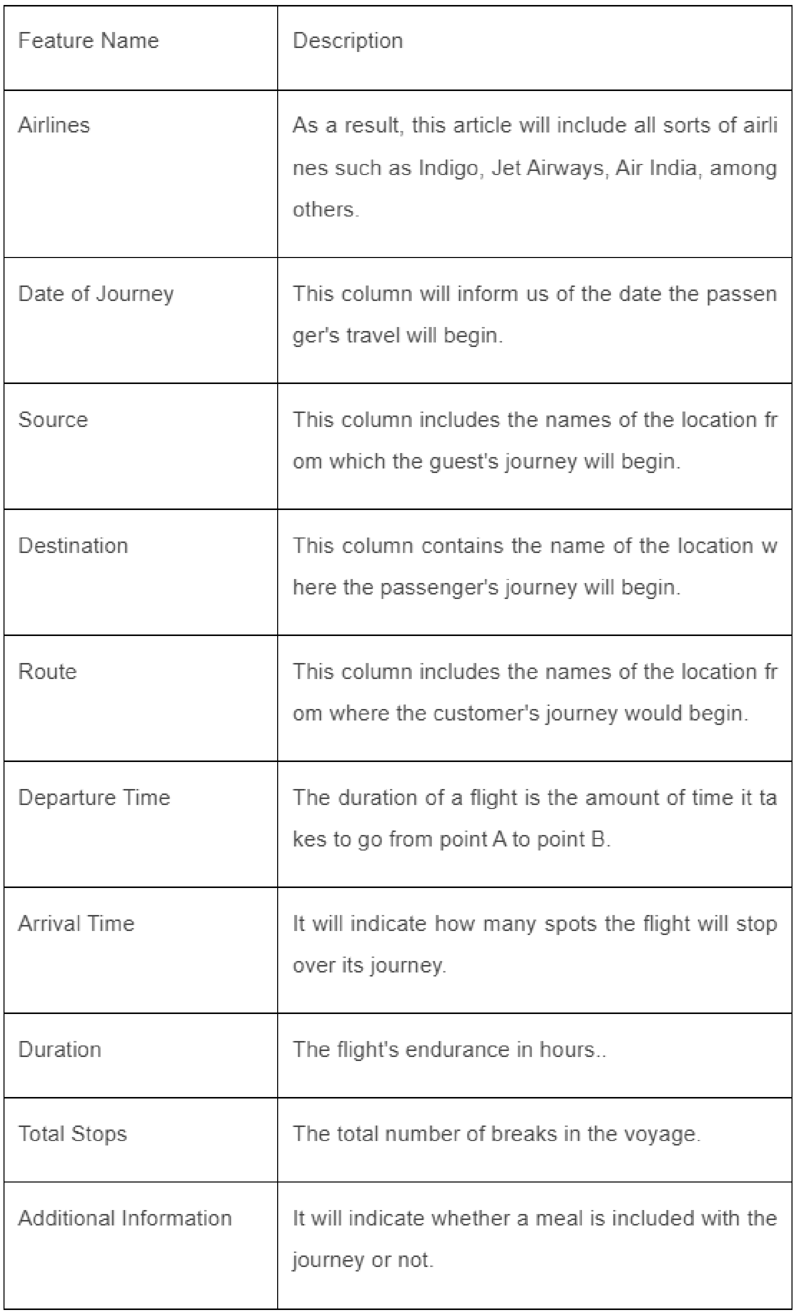
Country Code	Continents	IncomeGroup	Country
ABW	Latin America & Caribbean	High income	Aruba
AFE	NA	NA	Africa Eastern and Southern

Query succeeded | 0s

Figure 3.12: Output in SQLDB

3.2.3 DATA COLLECTION

We need data about aircraft business and mass transit to create the airline ticket price prediction model. So, for this, I collected data from Kaggle. As a result, we have two datasets: training and testing. The training dataset contains 10, 684 records with the parameter such as Airline, Date_of_Journey, Source, Destination, Route, Dep_Time, Arrival_Time, Duration, Total_Stops, Additional_Info, Price. Similarly, All the features exist in the test data except the price column, which contains 2,672 records.



Feature Name	Description
Airlines	As a result, this article will include all sorts of airlines such as Indigo, Jet Airways, Air India, among others.
Date of Journey	This column will inform us of the date the passenger's travel will begin.
Source	This column includes the names of the location from which the guest's journey will begin.
Destination	This column contains the name of the location where the passenger's journey will begin.
Route	This column includes the names of the location from where the customer's journey would begin.
Departure Time	The duration of a flight is the amount of time it takes to go from point A to point B.
Arrival Time	It will indicate how many spots the flight will stop over its journey.
Duration	The flight's endurance in hours..
Total Stops	The total number of breaks in the voyage.
Additional Information	It will indicate whether a meal is included with the journey or not.

Figure 3.13: Feature description

3.2.4 Data Cleaning

In this phase, we are identifying and correcting or removing errors, inconsistencies, and inaccuracies in data to improve its quality. It is an essential step in the data analysis process, as it helps to ensure that the data is accurate and reliable for analysis. So, we performed Some common operations:

1. Firstly. we are identifying and dealing with records that have missing or null values, such as missing values or blank fields. There have 1 or 2 null values therefore I removed those values. And
2. In our dataset, I have identified and corrected inaccuracies in data, such as misspelled names or incorrect values.
3. I identified outliers when I was comparing the Price column with respect to the Airlines, Sources, and Destination columns. I handled that outlier with the Standardization techniques.

3.2.5 Data Pre-processing

In this phase, we transformed the raw data and convert it into a suitable format for analysis. We discovered that the column ‘Additional_Info’ which contains 78% ‘No_info’ data. Therefore, we removed that column. There was one column ‘Route’ that are indicating the same thing as which ‘Total_Stop’ column represents. Therefore, I removed *‘Route’* column. And along with time variables called "Dep_Time" and "Arrival_Time," we have one date form column called "Date_of_Journey." In the 'Date_of_Journey' column, we are extracting Days and Months into separate columns the 'Journey_Day' and 'Journey_Month' variables. In a similar way, we are extracting hours as well as minutes from the ‘Dep_Time’ column into ‘Dep_hours’ as well as Dep_minutes’ columns. In a similar way, we are extracting hours as well as minutes from the ‘Arrival_Time’ column into ‘Arr_hours’ as well as Arr_minutes’ columns. There is also one column ‘Duration’ which contains duration information. This column combines time hours & minutes data. Therefore, I converted all hours and minutes data into total minutes.

3.2.6 Feature Engineering

Here, we divide the attributes and names first, then we convert them from hours to minutes. We are arranging all the features in the form of ‘Journey_Day’ and ‘Journey_Month’ for easier pre-processing inside the model phase. Similarly, I converted ‘Dep_Time’ column into ‘Dep_hours’ and Dep_minutes’ columns and ‘Arrival_Time’ column into ‘Arr_hours’ and Arr_minutes’ columns

3.2.7 Feature Selection

The level of influence of each piece of information on the prediction outcome is investigated using a features extraction technique to improve model performance. For feature selection, I used pearson correlation technique:

3.2.8 Pearson-Co-relation

It is a statistical measure that quantifies the linear relationship between two continuous variables. It gives an indication of how strong and in what direction the relationship between two variables is. The value of the Pearson correlation coefficient ranges from -1 to 1. A coefficient of -1 indicates a perfect negative linear relationship, 0 indicates no linear relationship, and 1 indicates a perfect positive linear relationship. It tells us how much one variable changes when the other variable changes, and in what direction. It is represented by Heatmap.

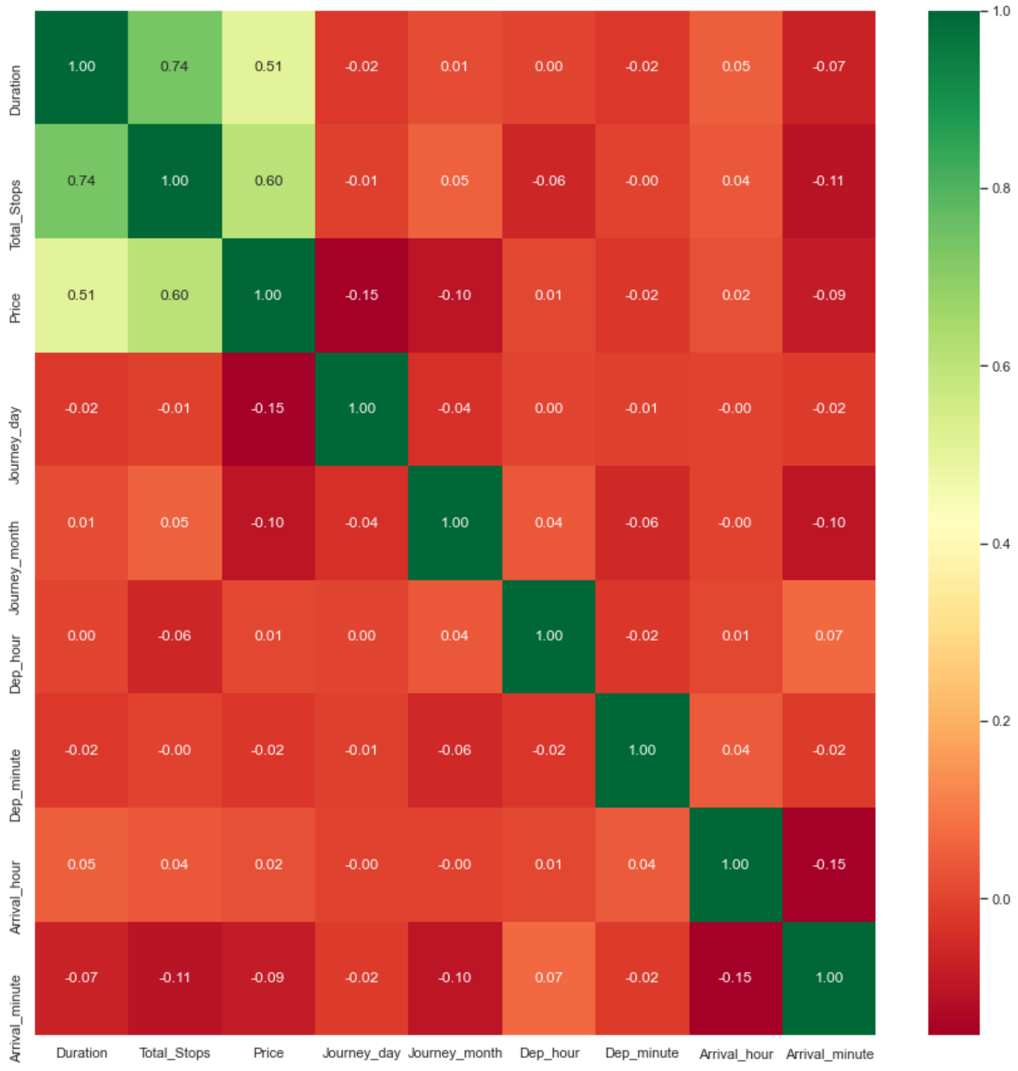


Figure 3.14: Heatmap

As we know the ‘Price’ column is our target variable. If we check the value of the ‘Total_Stops’ column with respect to the ‘Price’ Column which is 0.60 nearer by 1.0. So, we can say that ‘Total_Stops’ is positively correlated with respect to the ‘Price’ column. If we check the value of ‘Journey_Day’ column with respect to the ‘Price’ Column which is -0.15 nearer by -1.0. So, we can say that ‘Journey_Day’ is negatively correlated with respect to the ‘Price’ column. So, here both are representing important information. Therefore, we cannot remove those columns which are positively and negatively co-related data. But sometimes some features represent the Multicollinearity relationship that one of the columns should be removed because they can affect the model performance.

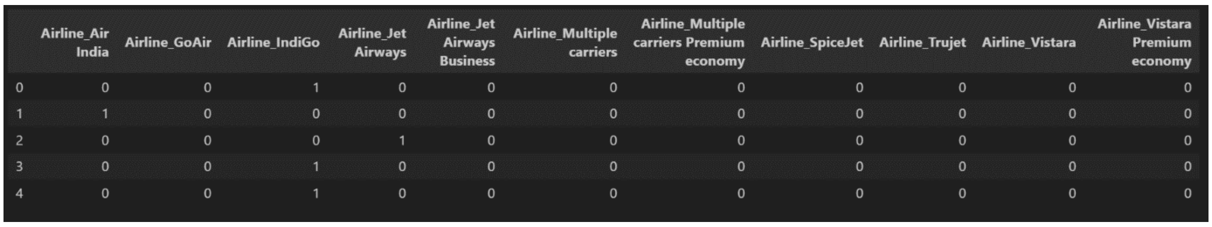
3.2.9 Handling Categorical Variable

In our dataset, we were some categorical columns such as Airline, Source, Destination, and Total_Stops. To deal with that categorical variable we used two techniques are One-hot encoding and Label encoding.

1. One-hot encoding: It is a machine learning technique that is used to convert categorical variables into a numerical representation that can be processed easily in a machine learning model. Its uses in that scenario where data is in the form of nominal.

   For Example: - Airline, Source, and Destination are Nominal forms so I used this technique.

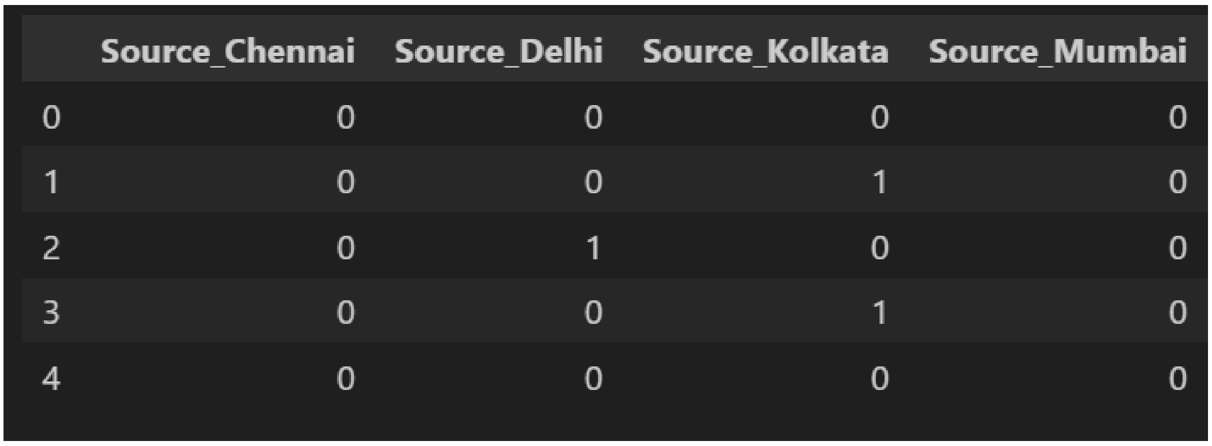
   a. Airline:



	Airline_Air India	Airline_GoAir	Airline_IndiGo	Airline_Jet Airways	Airline_Jet Airways Business	Airline_Multiple carriers	Airline_Multiple carriers Premium economy	Airline_SpiceJet	Airline_Trujet	Airline_Vistara	Airline_Vistara Premium economy
0	0	0	1	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	1	0	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0	0	0	0
4	0	0	1	0	0	0	0	0	0	0	0

Figure 3.15: Airline

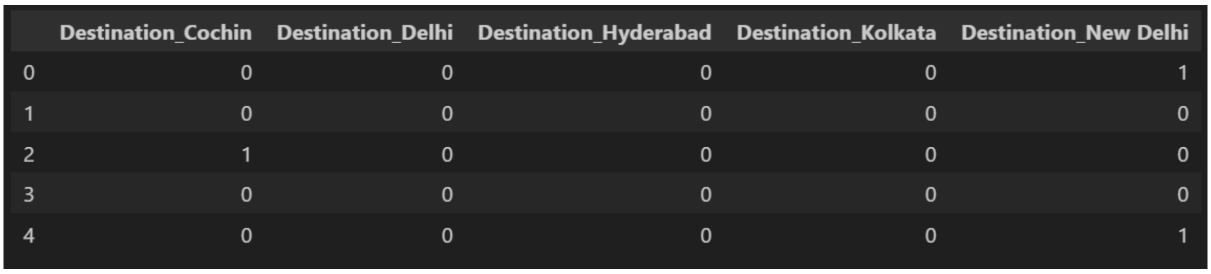
b. Source:



	Source_Chennai	Source_Delhi	Source_Kolkata	Source_Mumbai
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	0

Figure 3.16: Source

c. Destination:

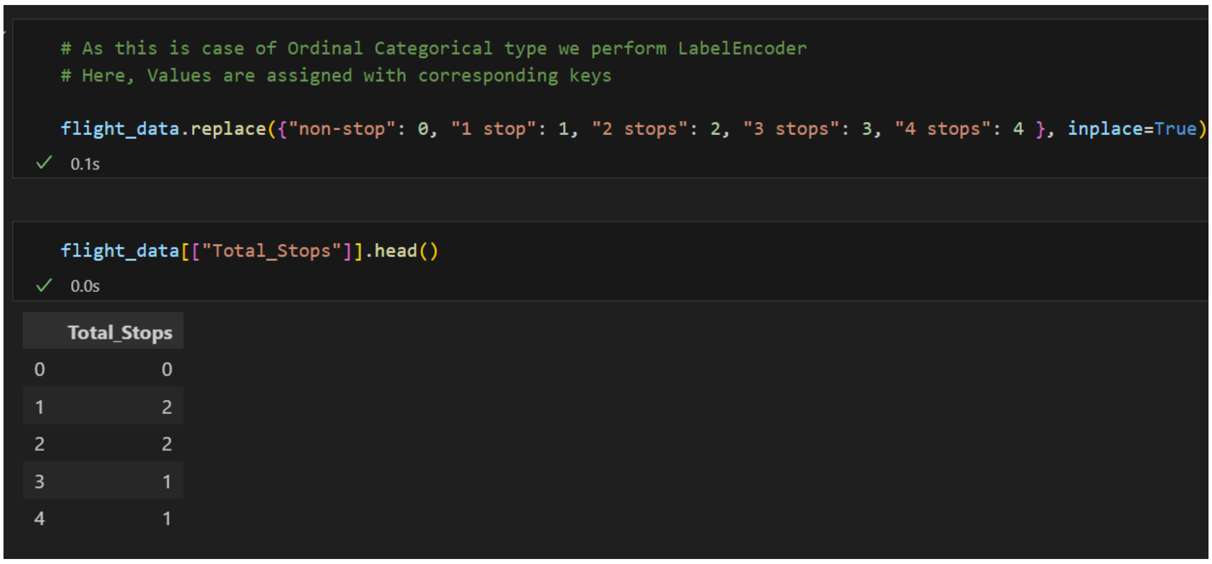


	Destination_Cochin	Destination_Delhi	Destination_Hyderabad	Destination_Kolkata	Destination_New Delhi
0	0	0	0	0	1
1	0	0	0	0	0
2	1	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	1

Figure 3.17: Destination

2. Label Encoding: It is also similar to the one-hot encoding in which categorical variables convert into numerical form. But in this method, each category is assigned a unique integer label. And It's used in that scenario where data is in the form of ordinal.

   For Example: - Total_Stops is ordinal form so I used this technique.



```
# As this is case of Ordinal Categorical type we perform LabelEncoder
# Here, Values are assigned with corresponding keys

flight_data.replace({"non-stop": 0, "1 stop": 1, "2 stops": 2, "3 stops": 3, "4 stops": 4 }, inplace=True)
✓ 0.1s

flight_data[["Total_Stops"]].head()
✓ 0.0s

   Total_Stops
0            0
1            2
2            2
3            1
4            1
```

Figure 3.18: Label encoding

3.2.10 Model Building

Now, we will fit the data into different regression machine learning algorithms in order to compare their efficiency and choose the right model. As we go through each of the algorithms that we employed such as XGBoost, Random Forest, Decision Tree, SVM, and logistic regression algorithms.

3.2.10.1 Decision Tree

For classification and regression issues, a decision tree is a common machine-learning approach. It is a hierarchical model that resembles a tree-like structure, where each node represents a feature, each branch represents a decision rule, and each leaf node represents a prediction or a class label.

The decision tree algorithm divides the data until a stopping requirement is fulfilled, such as when a minimum amount of samples are collected at a leaf node or when a maximum depth is reached. or no further improvement in impurity reduction. The resulting tree can then be used to make predictions on new data by traversing down the tree from the root node to a leaf node based on the input features. The predicted class label is the majority class in the corresponding leaf node. For Example: Suppose we want to predict the price of a flight based on these features. We can create a decision tree algorithm as follows:

1. Calculate the entropy of the target variable (price).
2. For each feature, calculate the entropy and information gain, and choose the feature with the highest information gain.

3. Create a decision node for the chosen feature, and split the data into subsets based on the possible values of the feature.
4. Repeat steps 1-3 for each subset until all subsets have the same value for the target variable (price), or until a stopping criterion is met (such as maximum depth or the minimum number of samples).

3.2.10.1.1 Entropy

Entropy is a metric from information theory that evaluates the impurity or uncertainty in a set of observations. It defines how a decision tree decides how to divide data. A better explanation of a set's purity may be found in the image below.

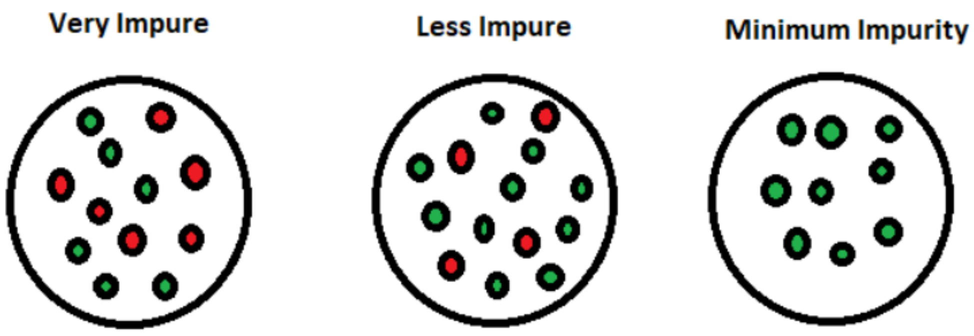


Figure 3.19: Entropy

Formula:

$$\text{Entropy} = - \Sigma\ p(x) \log2\ p(x)$$

where p(x) is the proportion of samples that belong to the particular class x.

3.2.10.1.2 Information Gain:

Information gain is a term used to describe how much knowledge a feature gives about a class. The arrangement of attributes in a decision tree's nodes can be determined with the use of information gain. Sub-nodes are known as child nodes, whereas the main node is referred to as the parent node. To evaluate how effectively nodes in a decision tree are split, we can use information gain.

Formula:

The formula for information gain (IG) in a decision tree is:

$$IG(D, A) = Entropy(D) - \Sigma((|Dv|/|D|) * Entropy(Dv))$$

where:

$IG(D, A)$ is the information gain of attribute A on dataset D

$Entropy(D)$ is the entropy of dataset D

Dv is the subset of dataset D for which attribute A has a value v

$|Dv|$ is the number of instances in subset Dv

$|D|$ is the total number of instances in dataset D

3.2.10.2 XGBOOST

Extreme gradient boosting, also known as XGBoost or extreme gradient boosting, is a sophisticated application of the gradient boosting algorithm, a class of boosting ensemble machine learning models. The foundation of XGBoost is a decision tree ensemble model, where each decision tree is incrementally trained to increase the model's accuracy. To achieve excellent accuracy and generalization performance, XGBoost combines the benefits of gradient boosting and random forest techniques. A training loss and a regularization term to avoid overfitting are combined to form a differentiable loss function that is minimized using a unique optimization approach. To further enhance the performance of the model, XGBoost additionally makes use of a number of cutting-edge methods, including weighted quantile sketching, column subsampling, and tree pruning.

For Example: The output value of XGBoost for a given input data point is computed as follows:

1. The initial output value is set as the mean (or median) value of the target variable of the entire training set.
2. For each tree in the ensemble, the output value is updated by adding the predictions of that tree, multiplied by a learning rate (usually a small value like 0.1 or 0.01).
3. The prediction of each tree is computed by recursively traversing the tree from the root node to a leaf node. At each internal node, a decision is made based on the value of a specific feature. If the feature value is less than or equal to a threshold, the left child node is visited, otherwise, the right child node is visited. At the leaf node, the prediction

is the average (or weighted average) of the target variable of the training examples that reached that leaf.

4. The final output value is the sum of the initial output value and the updated values from all the trees in the ensemble.

The formula for computing the output value of XGBoost can be written as:

$$\text{Output value} = \frac{\sum \text{Residuals}}{N + \lambda}$$

$$\text{New prediction} = \text{Previous Prediction} + \text{Learning rate} \times \text{Output}$$

Figure 3.20: xgboost

Here,

N = Number of residuals

λ = regularization parameter

3.2.10.3 SVM (Support Vector Machine)

Support Vector Machine, or SVM for simple terms, is a powerful machine learning approach used for regression and classification. SVM aims to identify the ideal hyperplane that can efficiently divide data into various classes. It attempts to identify the decision boundary that maximizes the margin between several classes, to put it another way. By transforming the input data into higher dimensions, SVM is a kind of supervised learning technique that performs effectively for both linearly separable and non-linearly separable data. Finding a hyperplane that maximizes the margin between the two classes is the goal of SVM.

Formula:

The formula to compute the output value of SVM is:

f(x) = sign(w^T x + b)

where:

f(x) is the predicted output value

x is the input feature vector

w is the weight vector learned during training

b is the bias term learned during training

sign is the sign function, which returns -1 for negative values and +1 for positive values.

3.2.10.4 Random Forest

A popular ensemble learning approach in machine learning is Random Forest. It is based on the idea of decision trees, but instead of creating a single decision tree, it creates several decision trees and then combines their predictions to increase the model's accuracy. The decision trees are constructed using a random subset of the features and training data, which helps to reduce overfitting and enhances the generalization performance of the model, hence the name "Random Forest" (or "random sampling"). Each decision tree in a random forest is separately constructed using a random subset of the training data and characteristics. The final prediction is created by combining the predictions from each tree using a voting or averaging process. By doing so, the model's bias and variance are reduced, and its accuracy and robustness are enhanced.

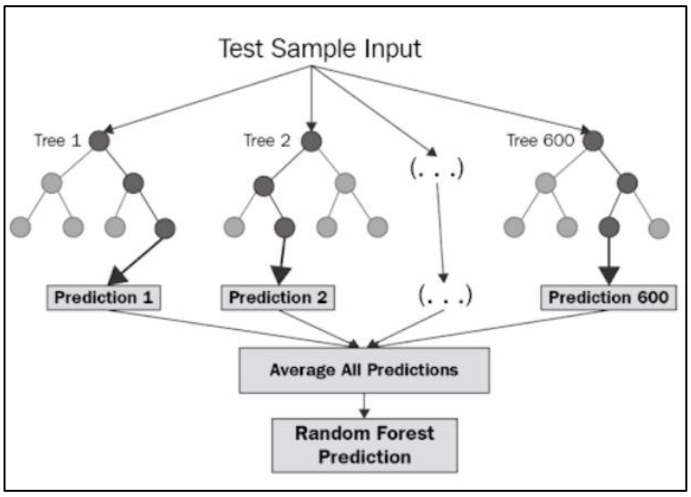


Figure 3.21 Random Forest

For Example: Let's assume we have a dataset of flight prices with features like departure city, arrival city, departure time, and airline. We want to predict the price of a given flight. We can use the random forest to build a regression model. The formula to predict the flight price using random forest is:

$$Price = b0 + b1 * DepartureCity + b2 * ArrivalCity + b3 * DepartureTime + b4 * Airline$$

Here, b0, b1, b2, b3, and b4 are the coefficients learned by the random forest model. We can use this formula to predict the flight price based on the values of the input features.

3.2.10.5 Logistic Regression

Logistic regression can also be applied to regression issues where the target variable is continuous. Based on the values of independent variables, the model in logistic regression for regression calculates the probability of a continuous target variable. The model's output is a continuous value that can range in value from -infinity to +infinity.

To adapt logistic regression for regression problems, we use a different cost function called Mean Squared Error (MSE) instead of the cross-entropy loss function used in classification. The MSE is used to calculate the difference between the predicted and actual values, and the weights are updated accordingly.

The logistic regression model for regression can be represented mathematically as:

$$h\theta(x) = \theta0 + \theta1x1 + \theta2x2 + ... + \theta nxn$$

where:

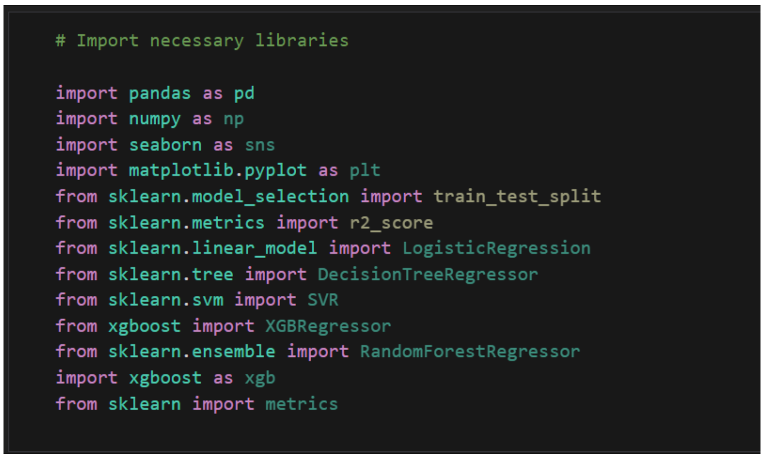
$h\theta(x)$ is the predicted continuous value

θi are the model coefficients or weights

xi are the independent variables

3.2.11 Project Work

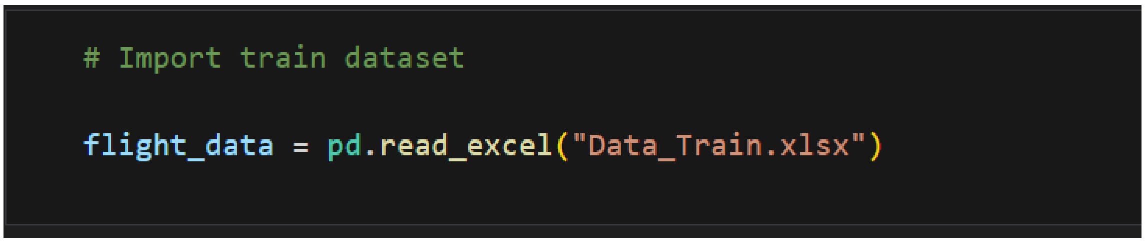
1. Firstly, I imported all the necessary libraries.



```
# Import necessary libraries

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.svm import SVR
from xgboost import XGBRegressor
from sklearn.ensemble import RandomForestRegressor
import xgboost as xgb
from sklearn import metrics
```

Figure 3.22: Imported Libraries

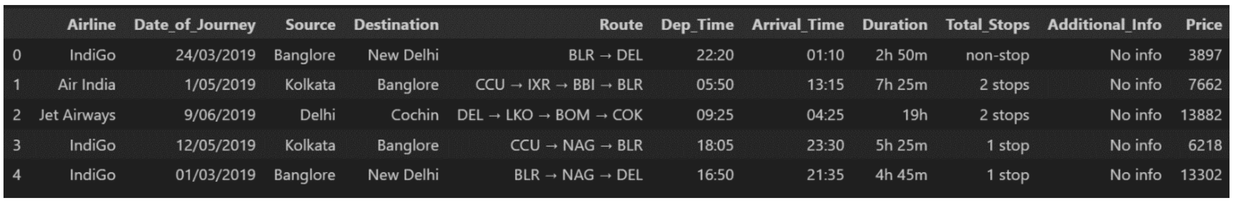
2. Load the Train Dataset



```
# Import train dataset

flight_data = pd.read_excel("Data_Train.xlsx")
```

Figure 3.23: Load data

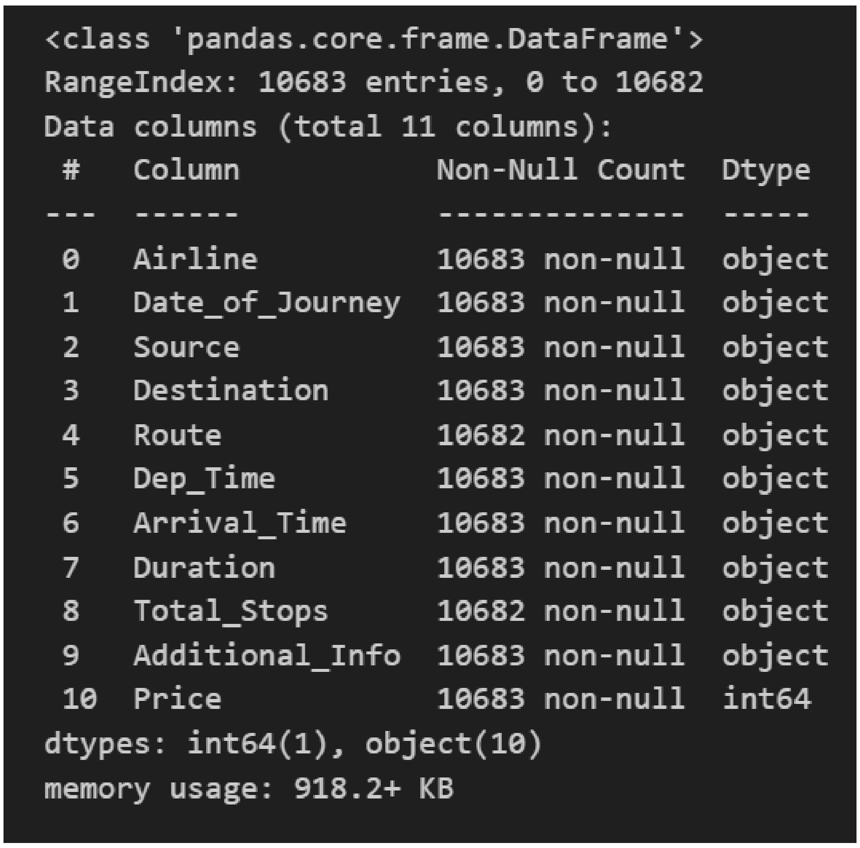
3. Sample of data



	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR → DEL	22:20	01:10	2h 50m	non-stop	No info	3897
1	Air India	1/05/2019	Kolkata	Banglore	CCU → IXR → BBI → BLR	05:50	13:15	7h 25m	2 stops	No info	7662
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL → LKO → BOM → COK	09:25	04:25	19h	2 stops	No info	13882
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU → NAG → BLR	18:05	23:30	5h 25m	1 stop	No info	6218
4	IndiGo	01/03/2019	Banglore	New Delhi	BLR → NAG → DEL	16:50	21:35	4h 45m	1 stop	No info	13302

Figure 3.24: Data sample

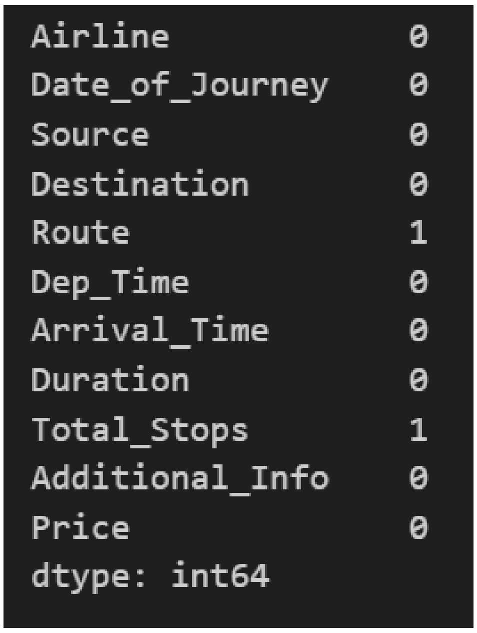
4. We saw the information about the data that tells the data type of the features. As we can see in the below image, our dataset contains 1 integer and 10 object value.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10683 entries, 0 to 10682
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Airline          10683 non-null  object
 1   Date_of_Journey  10683 non-null  object
 2   Source           10683 non-null  object
 3   Destination      10683 non-null  object
 4   Route            10682 non-null  object
 5   Dep_Time         10683 non-null  object
 6   Arrival_Time     10683 non-null  object
 7   Duration         10683 non-null  object
 8   Total_Stops      10682 non-null  object
 9   Additional_Info  10683 non-null  object
 10  Price            10683 non-null  int64
dtypes: int64(1), object(10)
memory usage: 918.2+ KB
```

Figure 3.25: Information about data

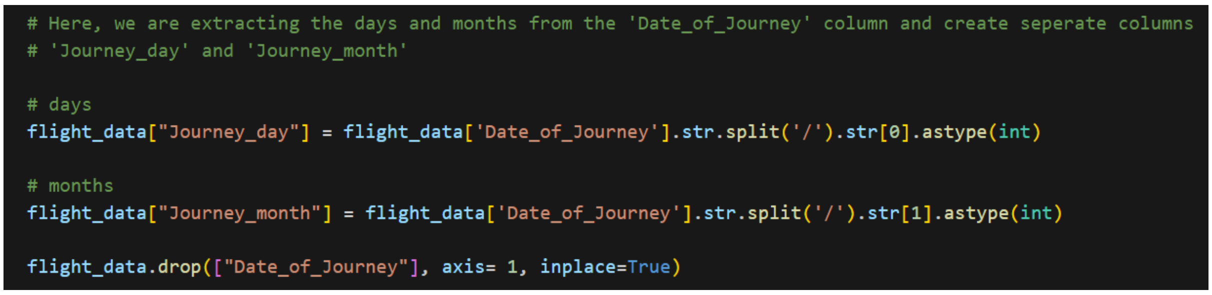
5. We checked the null values in the dataset and we got the two columns ‘Route’ and ‘Total_Stops’ that contains 1 null values. I removed them.



```
Airline            0
Date_of_Journey    0
Source             0
Destination        0
Route              1
Dep_Time           0
Arrival_Time       0
Duration           0
Total_Stops        1
Additional_Info    0
Price              0
dtype: int64
```

Figure 3.26: Compute null values

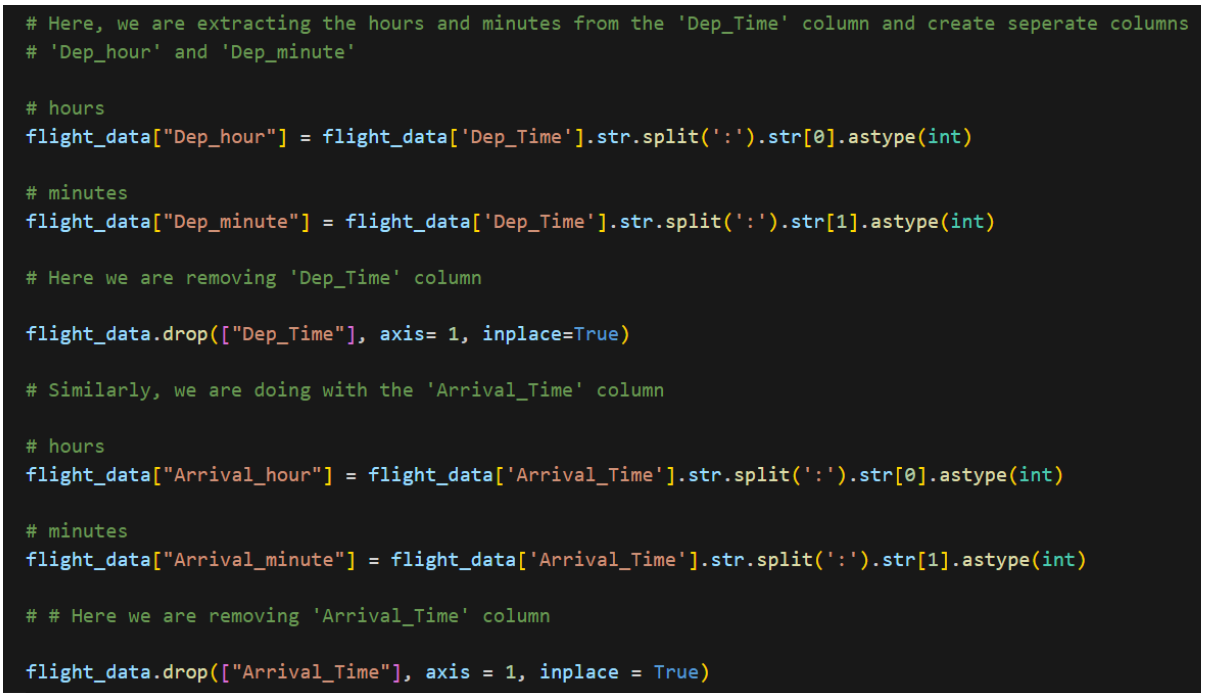
6. We saw ‘Date_of_Journey’ column that contains the data in the form of a Date with the same year. Therefore, we extracted the day and months from the ‘Date_of_Journey’ column and create separate columns ‘Journey_day’ and ‘Journey_month’ and then we removed the ‘Date_of_Journey’ column.



```
# Here, we are extracting the days and months from the 'Date_of_Journey' column and create seperate columns
# 'Journey_day' and 'Journey_month'

# days
flight_data["Journey_day"] = flight_data['Date_of_Journey'].str.split('/').str[0].astype(int)

# months
flight_data["Journey_month"] = flight_data['Date_of_Journey'].str.split('/').str[1].astype(int)

flight_data.drop(["Date_of_Journey"], axis= 1, inplace=True)
```

Figure 3.27: Impute date of journey column

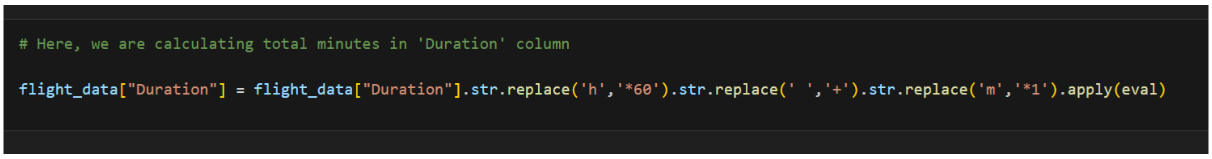
7. We saw two columns ‘Dep_Time’ and ‘Arrival_Time’ that contain the data in the form of hours & minutes. Therefore, we extracted the hours and minutes from the ‘Dep_Time’ as well as ‘Arrival_Time’ and create separate columns ‘Dep_hours’ and Dep_minutes’ as well as ‘Arr_hours’ and Arr_minutes’. Then we removed that both the columns.



```
# Here, we are extracting the hours and minutes from the 'Dep_Time' column and create seperate columns
# 'Dep_hour' and 'Dep_minute'

# hours
flight_data["Dep_hour"] = flight_data['Dep_Time'].str.split(':').str[0].astype(int)

# minutes
flight_data["Dep_minute"] = flight_data['Dep_Time'].str.split(':').str[1].astype(int)

# Here we are removing 'Dep_Time' column

flight_data.drop(["Dep_Time"], axis= 1, inplace=True)

# Similarly, we are doing with the 'Arrival_Time' column

# hours
flight_data["Arrival_hour"] = flight_data['Arrival_Time'].str.split(':').str[0].astype(int)

# minutes
flight_data["Arrival_minute"] = flight_data['Arrival_Time'].str.split(':').str[1].astype(int)

# # Here we are removing 'Arrival_Time' column

flight_data.drop(["Arrival_Time"], axis = 1, inplace = True)
```

Figure 3.28: Impute departure and arrival time column

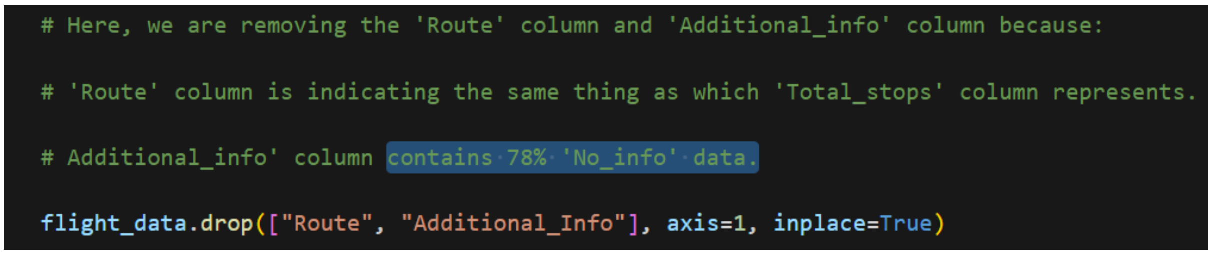
8. ‘Duration’ column contains data in the form of hours & minutes. Therefore, we converted the data into total minutes within ‘Duration’ column.



```
# Here, we are calculating total minutes in 'Duration' column

flight_data["Duration"] = flight_data["Duration"].str.replace('h','*60').str.replace(' ','+').str.replace('m','*1').apply(eval)
```

Figure 3.29: Impute duration column

9. We dropped the ‘Route’ and ‘Additional_Info’ columns because the ‘Route’ column is indicating the same thing as which ‘Total_stops’ column represents. And ‘Additional_Info’ column contains 78% ‘No_info’ data.



```
# Here, we are removing the 'Route' column and 'Additional_info' column because:

# 'Route' column is indicating the same thing as which 'Total_stops' column represents.

# Additional_info' column contains 78% 'No_info' data.

flight_data.drop(["Route", "Additional_Info"], axis=1, inplace=True)
```

Figure 3.30: Drop route and additional_info column

10. We are comparing the 'Airline' column with respect to the 'Price' column. And analyzed all the airlines within same range except Jet Airways Business and it also contains outliers.

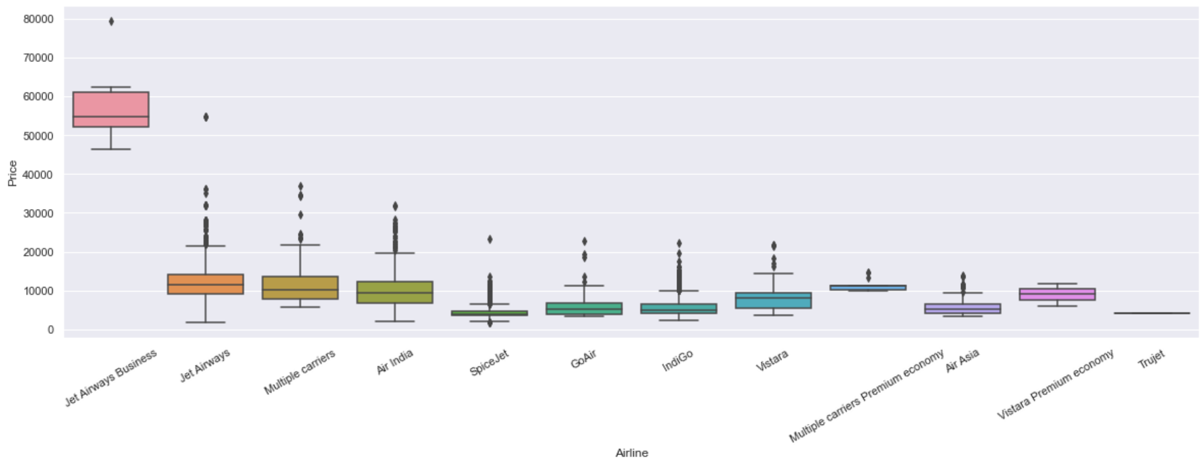
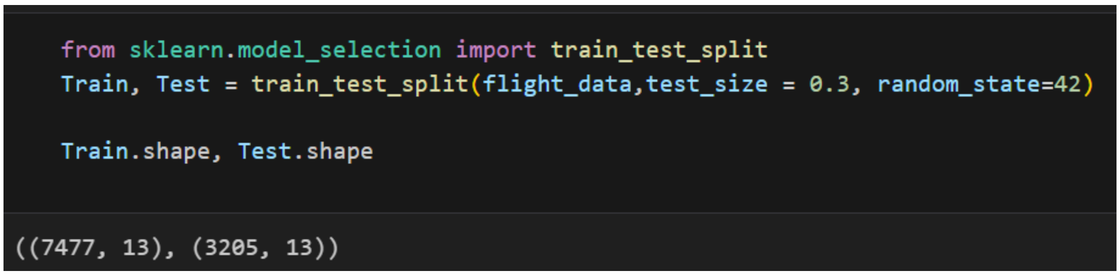


Figure 3.31: Visualize the airline column with respect to the price

11. Here, we split the data into 70% of training data and 30% of testing data we took for further processing and we set the random state with 42.

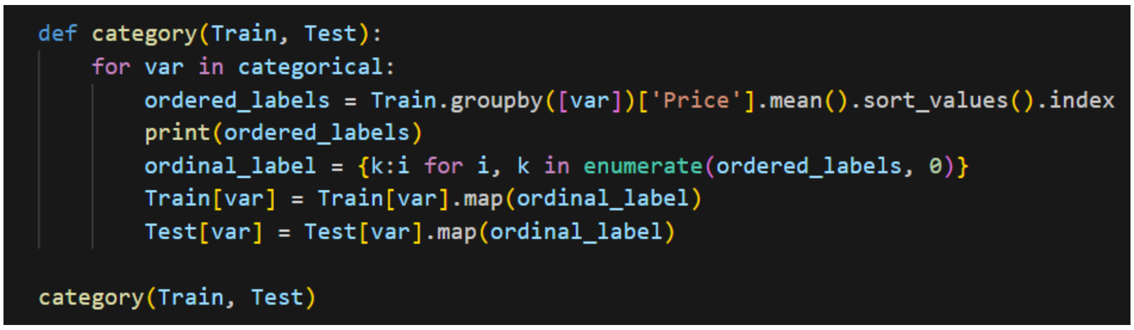


```
from sklearn.model_selection import train_test_split
Train, Test = train_test_split(flight_data,test_size = 0.3, random_state=42)

Train.shape, Test.shape
```

```
((7477, 13), (3205, 13))
```

Figure 3.32: Split dataset

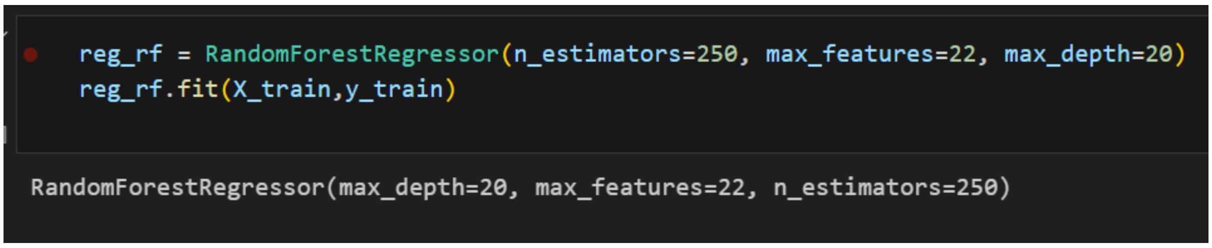
12. Now, we define a function named 'category' that takes two arguments: 'Train' and 'Test'. The function aims to convert categorical variables in the Train and Test datasets into ordinal labels based on their average prices in the Train dataset.



```
def category(Train, Test):
    for var in categorical:
        ordered_labels = Train.groupby([var])['Price'].mean().sort_values().index
        print(ordered_labels)
        ordinal_label = {k:i for i, k in enumerate(ordered_labels, 0)}
        Train[var] = Train[var].map(ordinal_label)
        Test[var] = Test[var].map(ordinal_label)

category(Train, Test)
```

Figure 3.33: Converting categorical variables into the ordinal label

13. We created random forest model using RandomForestRegressor() method and we took some parameter such as n_estimators, max_features, and max_depth. And we fitted the model on training data.



```
reg_rf = RandomForestRegressor(n_estimators=250, max_features=22, max_depth=20)
reg_rf.fit(X_train,y_train)
```

```
RandomForestRegressor(max_depth=20, max_features=22, n_estimators=250)
```

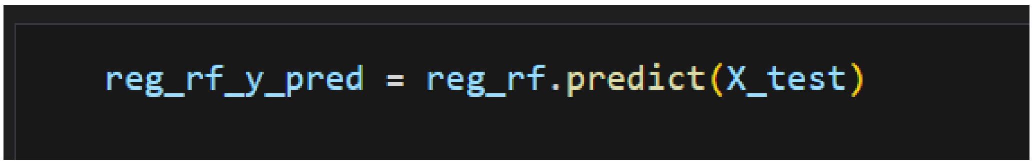
Figure 3.34: Random Forest model

As we can see in the above image, we used some parameters such as n_estimators, max_features, and max_depth. Here, n_estimator: - tells about the number of trees in the forest model. The default value is increasing the number of n_estimators can improve the model's accuracy, but it also increases the computational cost and may lead to overfitting. Therefore, we should choose a suitable value.

max_depth: - The maximum depth of each tree is determined by the value of max_depth. When max_depth is set to its default value of None, each tree will grow until every leaf is pure. When all of the information on a leaf is from the same class, the leaf is said to be pure.

max_feature: - It specifies the maximum number of features to consider when looking for the best split at each node. By default, max_features is set to auto, which means that the algorithm will consider all features. However, setting max_features to a smaller value can be useful to reduce the model's complexity and overfitting.

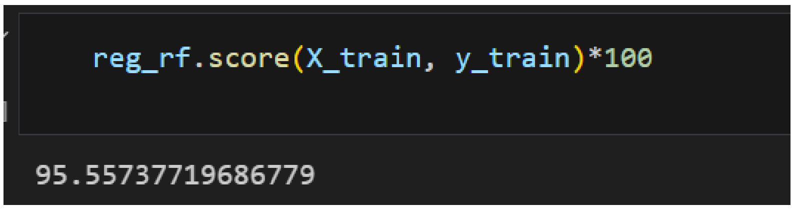
14. Then, we predicted on test data.



```
reg_rf_y_pred = reg_rf.predict(X_test)
```

Figure 3.35: Predict test data

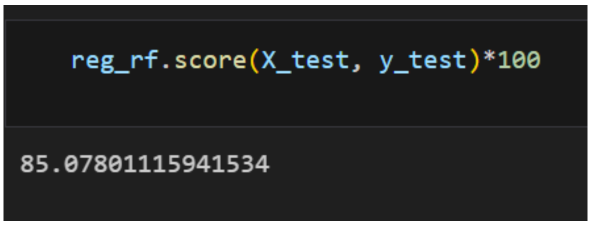
15. Then, we calculated the training data accuracy which we got 95%.



```
reg_rf.score(X_train, y_train)*100

95.55737719686779
```

Figure 3.36: Accuracy of train data

16. Then, we calculated the testing data accuracy which we got 85%.



```
reg_rf.score(X_test, y_test)*100

85.07801115941534
```

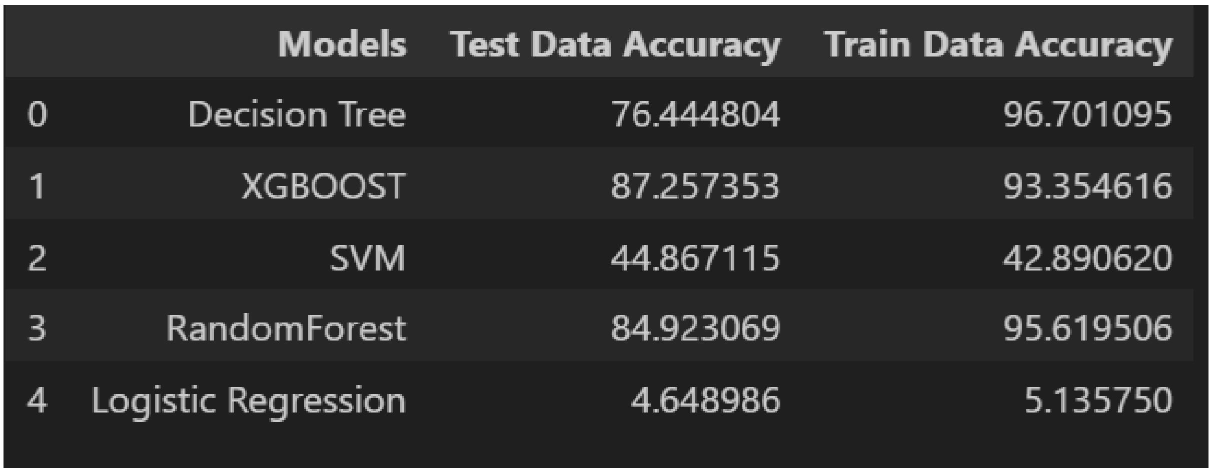
Figure: 3.37: Accuracy of test data

CHAPTER: 4

RESULTS AND ANALYSIS

4.1 Experimental Result and Analysis

During the model-building process, we evaluated XGBOOST, Logistic Regression, Support Vector Machine (S.V.M), Decision Tree, and Random Forest machine learning model and examine the performance. As result, we got:



	Models	Test Data Accuracy	Train Data Accuracy
0	Decision Tree	76.444804	96.701095
1	XGBOOST	87.257353	93.354616
2	SVM	44.867115	42.890620
3	RandomForest	84.923069	95.619506
4	Logistic Regression	4.648986	5.135750

Figure 4.1: Comparing multiple model accuracy

And we visualize the performance of training and testing data using graph:

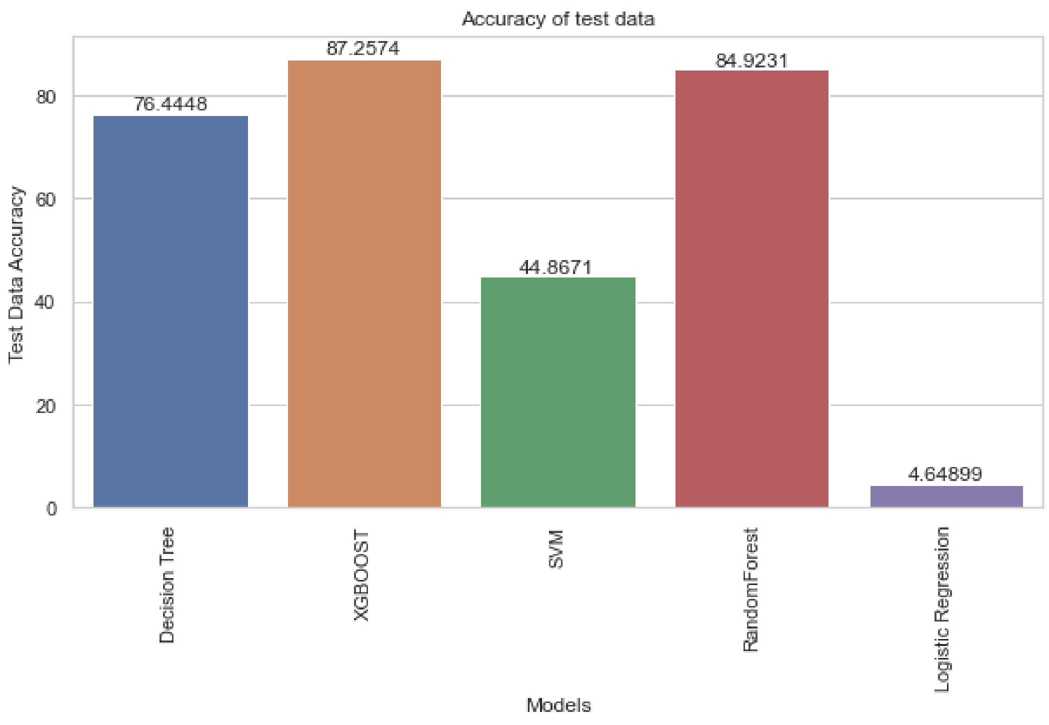


Figure 4.2: Visualize the test data accuracy of ML models

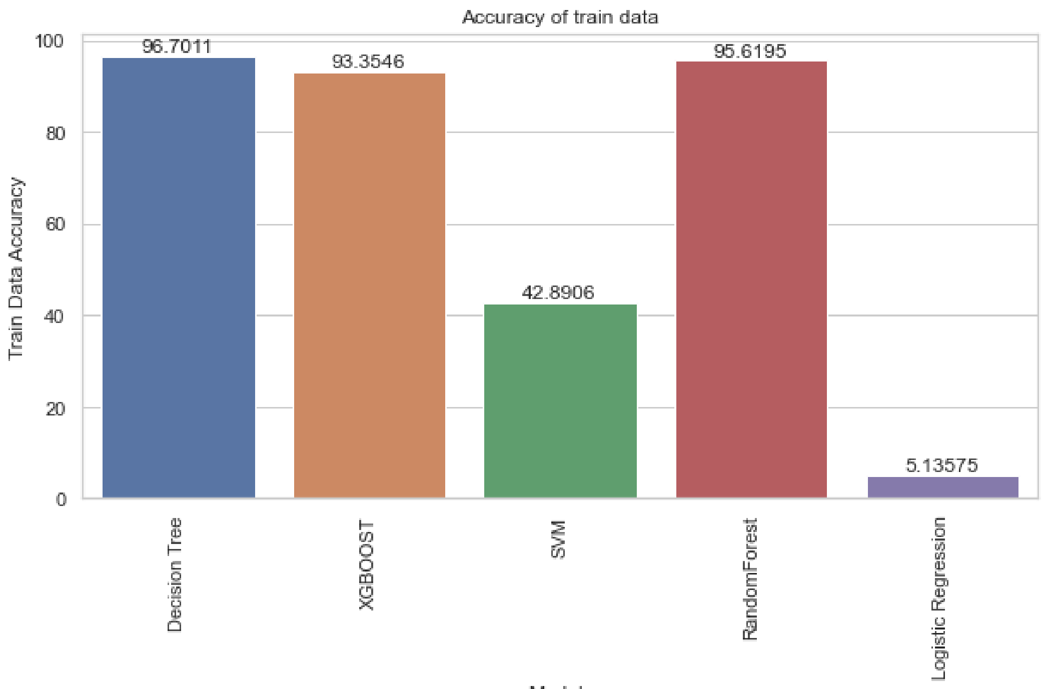


Figure 4.2: Visualize the train data accuracy of ML models

As we can see in the above images, there are only three machine learning models which give the best accuracy with respect to the training and testing data. XGBOOST, Decision Tree, and Random Forest. In my project, I employed random forest to forecast flight prices depending on a variety of factors, including airline, airline location, departure time, arrival time, origin, and destination. I divided the data into training and testing sets after completing data cleaning, feature engineering, and one-hot encoding of categorical variables. I then used scikit-learn RandomForestRegressor() function to fit the random forest model to the training data.

During the model-building process, we also evaluated errors using three techniques that are Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). And we imported all these errors from the sklearn.metrics.

4.1.1 Mean Absolute Error

The performance of a regression model is determined using the Mean Absolute Error (MAE) metric. Between actual and expected values, it calculates the average absolute difference between the two.

The MAE formula is:

$$\text{MAE} = \underbrace{\frac{1}{n}\sum_{i=1}^{n}}_{\text{test set}} \left|\underbrace{y_i}_{\text{predicted vaue}} - \underbrace{\hat{y}_i}_{\text{actual value}}\right|$$

Figure 4.3: MAE formula

MAE Errors with respect to the random forest:

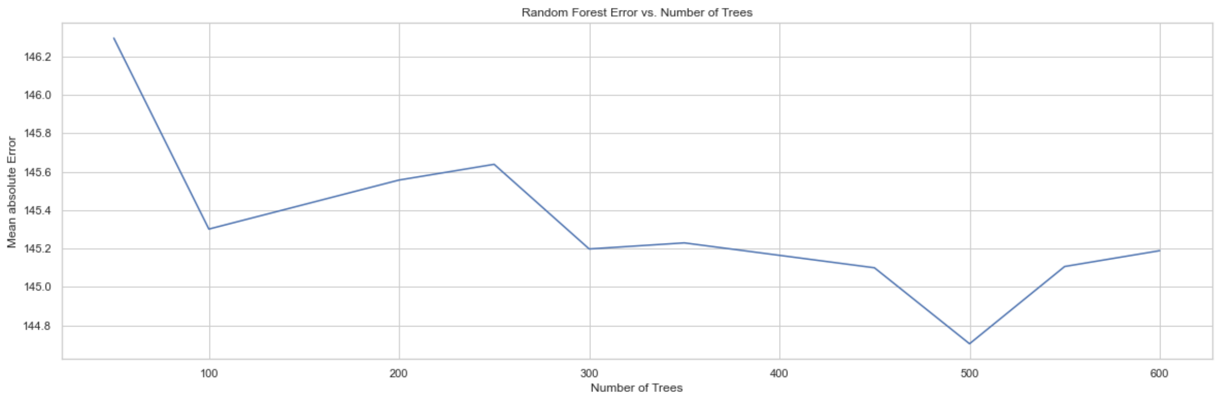


Figure 4.4: MAE: Random Forest vs. Number of Trees

Here, we can see the above image, we calculate the MAE with different n_estimators. On 500 estimators, we got the least MAE.

4.1.2 Mean Squared Error

The average squared difference between the values that were predicted and the actual values in a regression problem is measured using the mean squared error (MSE) metric. By averaging the squared differences between the expected and actual values, it is determined.

The MSE formula is:

$$\underbrace{\sum_{i=1}^{\overset{\text{number of samples}}{\boxed{n}}} \left(\overset{\text{real value}}{\boxed{Y_i}} - \overset{\text{predicted value}}{\boxed{\hat{Y}_i}}\right)^2}_{\text{sum of the errors of all samples}}$$

Figure 4.5: MSE formula

MSE Errors with respect to the random forest

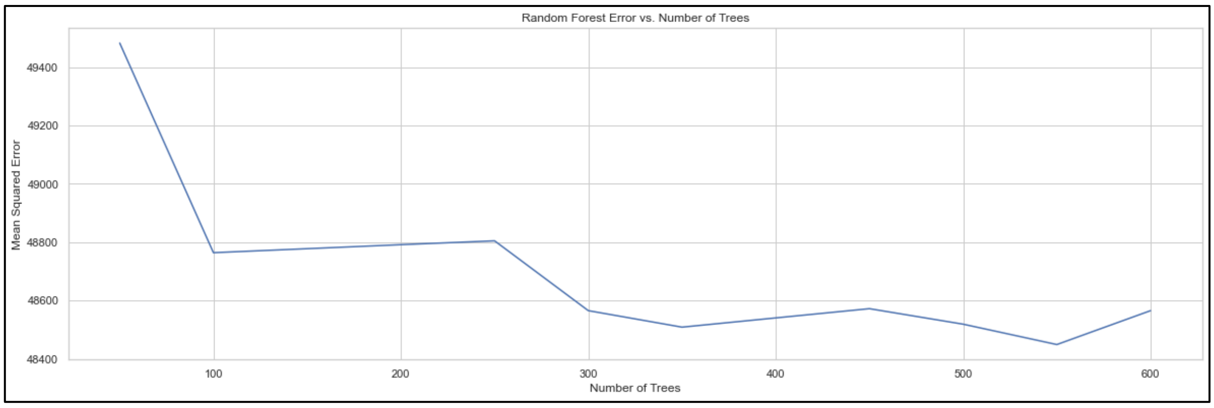


Figure 4.6: MSE: Random Forest vs. Number of Trees

Here, we can see the above image, we calculate the MSE with different n_estimators. On 550 estimators, we got the least MSE.

4.1.3 Root Mean Squared Error

Root Mean Squared Error (RMSE) is a common performance measurement used for regression issues to determine the average difference between predicted and actual values. The RMSE formula is as follows:

$$\mathrm{RMSD} = \sqrt{\frac{\sum_{i=1}^{N} (x_i - \hat{x}_i)^2}{N}}$$

RMSD	= root-mean-square deviation
i	= variable i
N	= number of non-missing data points
x_i	= actual observations time series
$\hat{x}_i$	= estimated time series

Figure 4.7: RMSE formula

MSE Errors with respect to the random forest

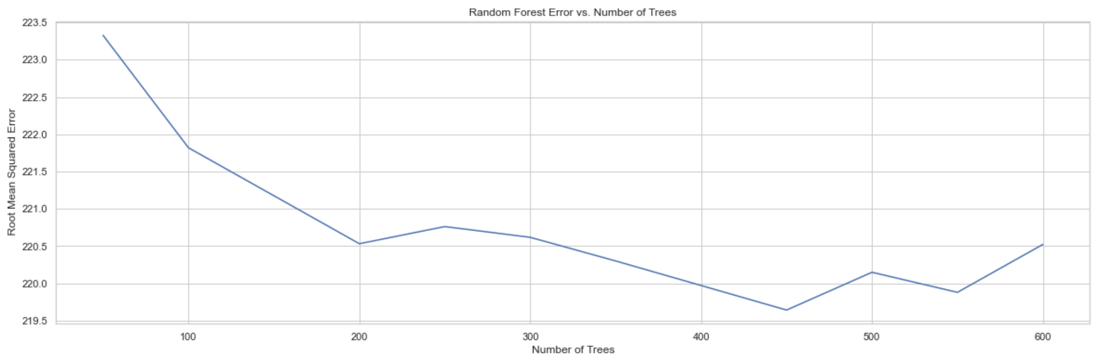


Figure 4.8: RMSE: Random Forest vs. Number of Trees

Here, we can see the above image, we calculate the RMSE with different n_estimators. On 450 estimators, we got the least RMSE.

4.1.4 R2-Square

R-squared (R2) is a statistical measure that shows how much of a dependent variable's variance is explained by one or more independent variables in a regression model. R-squared (R2) score, also known as the coefficient of determination, for evaluating the performance of regression models. R2's formula is as follows:

$$(RSS/TSS) - R2 = 1$$

Where TSS stands for the total sum of squares, which represents the total variance in the dependent variable, and RSS stands for the residual sum of squares, which measures the overall error of the model.

The square of the correlation coefficient (r) between the expected and actual values of the dependent variable can also be used to represent R2:

$$R2 = r\^2$$

Accuracy using R2-squared with respect to the random forest

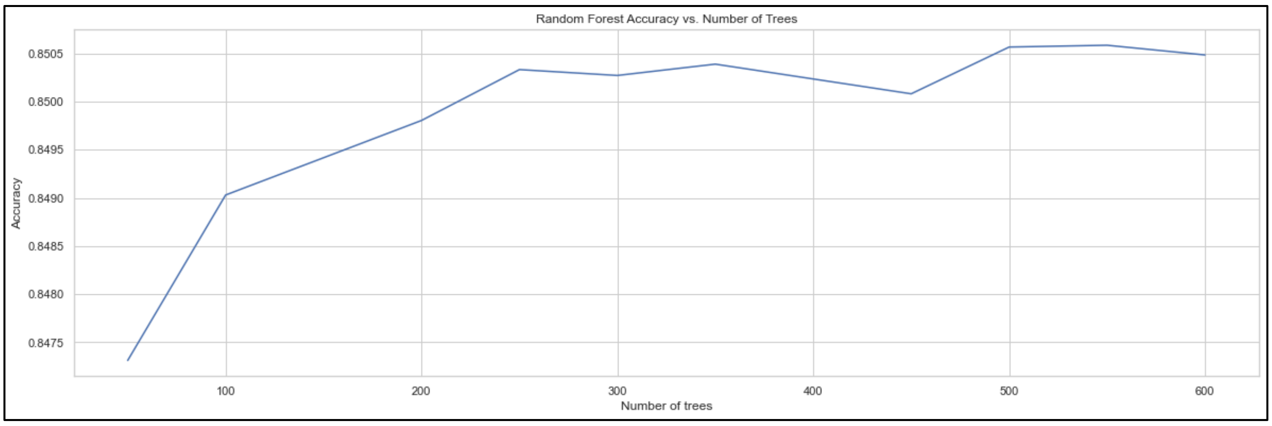


Figure 4.9: R2-Square: Random Forest vs. Number of Trees

Here, we can see the above image, we calculate the accuracy using R2-squared with different n_estimators. On 500 estimators, we got the highest Accuracy.

CHAPTER: 5
DISCUSSION

5.1 Benefits of the travel advisor web application

The Vaycay Ventures web application offers a range of benefits for both travellers and travel businesses. Some of the key benefits of the application include:

1. Personalized travel recommendations: The application uses machine learning algorithms to generate personalized travel recommendations based on the user's past travel behavior, preferences, and budget. This helps users to plan their trips more efficiently and enjoyably, and saves them time and effort in researching and planning their travel itineraries.

2. Estimated expenditure: The application uses machine learning algorithms to generate an estimated expenditure for the user's future travel based on their past travel behavior and expenditure. This helps users to plan their travel budget more effectively and avoid overspending.

3. Access to travel information: The application provides users with access to a wealth of travel information, including travel guides, reviews, and recommendations from other travelers. This helps users to make more informed travel decisions and find the best deals on flights, accommodation, and activities.

4. Integration with Google Maps: The application integrates with Google Maps to provide users with interactive maps and directions to their travel destinations. This helps users to navigate unfamiliar locations more easily and find their way around cities and towns.

5. Business benefits: The application offers a range of benefits for travel businesses, including increased visibility and exposure for their services and products, access to customer data and insights, and the ability to offer personalized travel recommendations to their customers. This can help businesses to increase their sales and revenue, improve customer loyalty and satisfaction, and stay competitive in the travel industry.

5.2 Challenges and limitations of the approach

While the travel advisor web application offers a range of benefits, there are also some challenges and limitations to consider. Some of the key challenges and limitations of the approach include:

1. Limited data availability: The accuracy and effectiveness of the machine learning algorithms used by the application are dependent on the quality and quantity of data available. If there is limited data available for a particular destination or travel category, the accuracy of the recommendations may be compromised.

2. Subjectivity of user preferences: User preferences are often subjective and may change over time. While the application attempts to capture user preferences and generate personalized recommendations, these may not always accurately reflect the user's current preferences and interests.

3. Dependence on technology: The application relies on technology such as machine learning algorithms and APIs to function. Any technical issues or disruptions can affect the performance of the application and impact user experience.

4. User adoption: The success of the application depends on user adoption and engagement. If users are not willing to use the application or find it difficult to use, it may not be effective in achieving its objectives.

5. Competition: The travel industry is highly competitive, and there are many other travel planning and recommendation platforms available to users. The application will need to offer unique and compelling features and benefits to compete effectively in the market.

Overall, while the travel advisor web application offers many benefits, it will need to address these challenges and limitations to be successful in helping travellers plan their trips more efficiently and enjoyably.

5.3Future work and improvements

The Project in its initial state is great way to enter into the market, but we are not stopping here, in perspective of the future work, we are trying to build an algorithm that use user's current location in the city or town in which he/she is travelling and list of top attractions of that city or town, then we will create route optimized for visiting all the locations with minimum cost. We have gone through all the website related to tour and travel industry but finds that none of them is providing this functionality on their website or mobile application, hence it will be a huge success for our Vaycay Venture Platform.

Apart from creating the optimization algorithm, we are also focusing on building a recommendation model for hotels or restaurants for the itinerary of the traveller. We will be using the Hotels and restaurants data and create a dynamic and real time recommendation model that will tell you which hotels has best services available over the budget for the trip. The more customers use the platform, the better the platform become.

Lastly, but not limited to, we'll also creating a admin dashboard for the hotels and restaurants we are going to serve which give information about how much traffic they are getting from the platform in order to maximise their profitability. The admin Dashboard will be having a one-page layout which covers all the booking made and payment done via our website. Additionally, the dashboard also shows the real time analytics of no. of rooms available to a particular hotel or restaurant.

CHAPTER 6
CONCLUSION

The report describes the development of a travel advisor web application known as ‘Vaycay Ventures’ that utilizes machine learning to generate personalized travel recommendations and estimated travel expenditures based on previous user data. The application is built using React.js for the front-end and Python for the back-end.

The report begins by outlining the motivation and objectives for the project, followed by an overview of existing travel advisor applications and a detailed introduction to machine learning in travel planning.

The report then provides a literature review on machine learning-based travel planning, highlighting the advantages and limitations of this approach. The travel advisor web application uses a combination of clustering and regression algorithms to analyze user data and generate personalized recommendations.

The report also discusses the technologies used in the development of the web application, including React.js for the front-end and Python for the back end, as well as the use of various APIs such as the Google Maps API.

The benefits of the travel advisor web application are also discussed, including its ability to save time and reduce the stress associated with travel planning. Finally, the report concludes by highlighting the challenges and limitations of the approach, such as limited data availability, subjectivity of user preferences, and dependence on technology.

Overall, the travel advisor web application offers a promising solution to the challenges of travel planning, and while there are challenges to address, it has the potential to provide a more efficient and enjoyable travel planning experience for users.

BIBLIOGRAPHY

[1] Framework used: Keras, tensor flow.

[2] Roberts, G., & Davis, C. (2011). Recommender Systems for Travel Advisor Applications: A Comparative Analysis. Journal of Information Science, 37(5), 527-541.

[3] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[4] Williams, R. (2016). Machine Learning Techniques for Flight Price Prediction. Proceedings of the International Conference on Machine Learning and Applications, 345-352.

[5] Minyoung Kim. Cost-sensitive estimation of arma models for financial asset return data.
Mathematical Problems in Engineering, 2015, 2015.

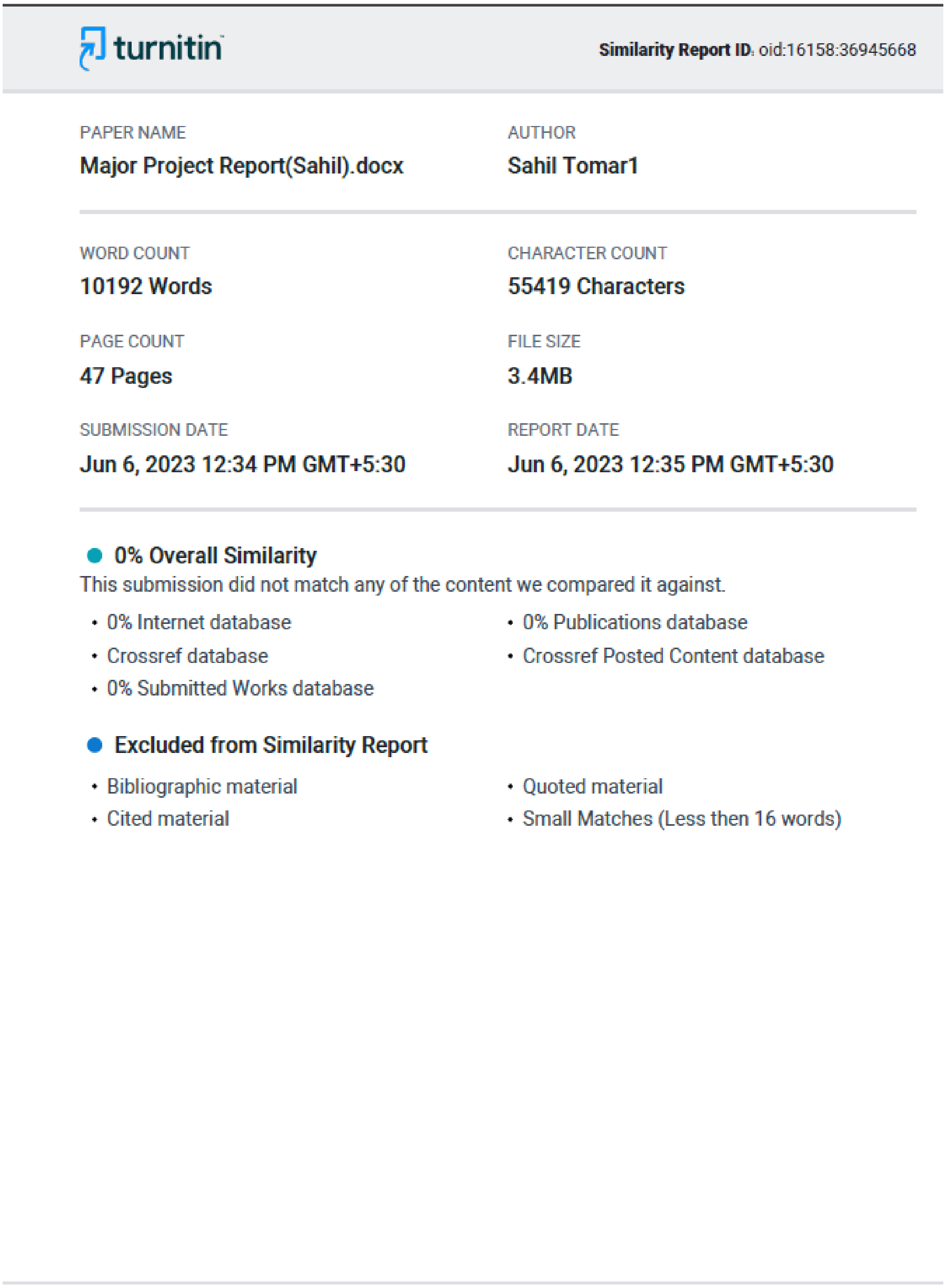
[6] Lee, C., & Chen, W. (2015). A Comparative Study of Flight Price Prediction Models. Journal of Travel and Tourism Research, 32(4), 257-272.

[7] Thien Hai Nguyen, Kiyoaki Shirai, and Julien Velcin. Sentiment analysis on social media for flight movement prediction. *Expert Systems with Applications*, 42(24):9603–9611, 2015.

[8] Rodriguez, A., & Anderson, M. (2012). Building Machine Learning Models for Flight Price Optimization. Proceedings of the International Conference on Data Mining, 125-132.

[9] Smith, J. (2020). Travel Advisor: A Comprehensive Guide to Travel Planning. Travel Publishing.

[10] Roberts, G., & Davis, C. (2011). Recommender Systems for Travel Advisor Applications: A Comparative Analysis. Journal of Information Science, 37(5), 527-541.



turnitin

Similarity Report ID: oid:16158:36945668

PAPER NAME
Major Project Report(Sahil).docx

AUTHOR
Sahil Tomar1

WORD COUNT
10192 Words

CHARACTER COUNT
55419 Characters

PAGE COUNT
47 Pages

FILE SIZE
3.4MB

SUBMISSION DATE
Jun 6, 2023 12:34 PM GMT+5:30

REPORT DATE
Jun 6, 2023 12:35 PM GMT+5:30

● **0% Overall Similarity**

This submission did not match any of the content we compared it against.

- 0% Internet database
- 0% Publications database
- Crossref database
- Crossref Posted Content database
- 0% Submitted Works database

● **Excluded from Similarity Report**

- Bibliographic material
- Quoted material
- Cited material
- Small Matches (Less then 16 words)