FLIGHT DELAYS PREDICTION

Submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY,HYDERABAD

In partial fulfillment of requirements for the award of the degree of

MASTER OF COMPUTER APPLICATION

IN

COMPUTER SCIENCE AND ENGINEERING (MCA)

Submitted by

MUNUJAM ANUSHA(23UK1F0007)

Under the guidance of

MRS.A.Swathi

Assistance Professor

DEPARTMENT OF

COMPUTER SCIENCE AND ENGINEERING(MCA)

VAAGDEVI ENGINEERING COLLEGE(AUTONOMOUS)

Affiliated to JNTUH, HYDERABAD BOLLIKUNTA, WARANGAL (T.S) – 506005

DEPARTMENT OF

COMPUTER SCIENCE AND ENGINEERING(MCA)

VAAGDEVI ENGINEERING COLLEGE(AUTONOMOUS)

CERTIFICATE OF COMPLETION

PROJECT WORK REVIEW-1

This is to certify that the PG Project Phase-1 entitled "**FLIGHT DELAYS PREDICTION**" is being submitted by **MUNJAM ANUSHA(23UK1F0007)** in partial fulfillment of the requirements for the award of the degree of Master of computer application in Computer Science & Engineering to Jawaharlal Nehru Technological University Hyderabad during the academic year 2023-2024.

Project Guide

Mrs.A.Swathi

(Assistant professor)

HOD

Dr.R.Naveen Kumar

(professor)

External

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr.Syed Musthak Ahmed,**Principal, Vaagdevi Engineering College for making us available all the required assistance and for his support and inspiration to carry out this PG Project Phase-1 in the institute.

We extend our heartfelt thanks to **Dr.R.Naveen kumar** , Head of the Department of MCA, Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the PG Project Phase-1.

We express heartfelt thanks to Smart Bridge Educational Services Private Limited, for their constant supervision as well as for providing necessary information regarding the PG Project Phase-1 and for their support in completing the PG P roject Phase-1.

We express heartfelt thanks to the guide, **A.Swathi ,** Assistant professor, Department of CSE for her constant support and giving necessary guidance for completion of this PG Project Phase-1.

Finally, we express my sincere thanks and gratitude to my family members, friends for their encouragement and outpouring their knowledge and experience throughout the thesis.

MUNJAM ANUSHA (23UK1F0007)

ABSTRACT

Flight delays are a constant problem in the airline business, which is made worse by the fact that air travel is more crowded now than it was 20 years ago. There are big financial costs for planes because of these delays, and they also hurt the environment. To deal with this important problem, airlines are constantly looking for different ways to cut down on delays and cancellations.using machine learning model,we can predict flight arrival delays .The input to our algorithm is rows of feature vectors like departure date departure delay,distance between the two airports,scheduled arrival time.We then use decision tree classifier to predict if the flight arrival will be delayed or not

Keywords:flight delay,flight data,prediction,machine learning classifier

TABLE OF CONTENT

1.INTRODUCTION

1.1.OVERVIEW

Flight delays prediction involves forecasting whether a flight will be delayed based on various influencing factors. This field leverages data analysis, statistical methods, and machine learning techniques to provide accurate predictions, aiding airlines, passengers, and airport operators in managing and mitigating delays.

1.2.Purpose: The primary purpose of flight delays prediction is to enhance the efficiency and reliability of air travel by anticipating potential delays and enabling proactive measures. Here are the key objectives:

Operational Efficiency:

Airline Operations: Optimize flight schedules, crew assignments, and maintenance planning to reduce operational disruptions.

Resource Allocation: Better utilization of airport resources such as gates, runways, and ground services.

Passenger Experience:

Information Provision: Offer timely and accurate delay information to passengers, allowing them to make informed travel decisions.

Minimize Disruptions: Reduce passenger inconvenience by anticipating and managing delays effectively.

Cost Reduction:

Fuel Savings: Avoid unnecessary fuel consumption due to delays and holding patterns.

Operational Costs: Decrease costs associated with delays, such as overtime pay, missed connections, and compensation claims.

Safety and Compliance:

Regulatory Adherence: Ensure compliance with aviation regulations and minimize the risk of penalties due to delays.

Safety Management: Identify and mitigate safety risks associated with delays and congested airspace.

Strategic Planning:

Long-term Improvements: Analyze delay patterns to make strategic decisions for infrastructure development and policy-making.

Predictive Maintenance: Schedule maintenance activities more effectively to prevent delays due to technical issues.

1.2 Objectives

- Improve customer satisfaction by providing timely updates and alternatives.
- Enhance airline operational efficiency by optimizing scheduling and resource allocation.
- Assist airports in managing runway and gate usage more effectively.
- Reduce costs associated with delays, such as crew overtime and fuel consumption.
- Aid passengers in making informed travel decisions and managing their itineraries better.

2. LITERATURE SURVEY

2.1 Existing problem

· **Weather Dependencies**: Weather conditions such as thunderstorms, snowstorms, fog, and strong winds can significantly impact flight operations. Predicting the exact timing and severity of these weather events is challenging, making it difficult to accurately forecast their effect on flights.

· **Air Traffic Congestion**: Busy airspace and congested airports can lead to delays as aircraft wait for clearance to take off or land. Predicting the flow of air traffic and potential congestion points is complex, especially during peak travel times or in regions with limited airspace capacity.

· **Airport Operations**: Issues within the airport itself, such as runway closures, ground handling delays, or gate availability problems, can cause delays. Coordinating these operations effectively to minimize disruptions is crucial but often challenging due to the number of stakeholders involved.

· **Aircraft Maintenance**: Unscheduled maintenance or technical issues with aircraft can lead to delays as airlines work to resolve problems before departure. Predicting when these issues might arise and their impact on flight schedules is difficult due to the unpredictability of mechanical failures.

· **Crew Scheduling and Availability**: Flight delays can also be caused by crew scheduling problems, including crew members exceeding their duty time limits or last-minute changes in crew assignments. Balancing crew availability and ensuring compliance with regulatory requirements adds complexity to scheduling and operational planning.

· **Passenger Effects**: Delays not only inconvenience passengers but can also lead to cascading effects on connecting flights, ground transportation, and other travel plans.

Managing passenger expectations and providing timely information during delays is crucial but often challenge

2.2 Proposed Solutions

Enhanced Data Integration and Quality:

- Integrate real-time data from diverse sources such as airlines, airports, weather services, air traffic control, and historical flight data.
- Ensure data quality through validation processes and automated cleaning techniques to minimize errors and inconsistencies.

Advanced Machine Learning Models:

- Develop and deploy sophisticated machine learning models such as ensemble methods, deep learning architectures (e.g., recurrent neural networks), and gradient boosting algorithms.
- Utilize models that can effectively handle temporal data and capture complex interactions between different factors influencing flight delays.

Feature Engineering and Selection:

- Identify and engineer relevant features that contribute to flight delays, such as weather patterns, airport congestion levels, historical flight punctuality, aircraft type, and crew schedules.

- Use feature selection techniques to prioritize the most influential factors for prediction accuracy.

Real-time Predictive Analytics:

- Implement systems for real-time monitoring and prediction of flight delays based on continuously updated data.
- Develop dashboards and visualization tools that provide stakeholders with actionable insights and early warnings about potential delays.

Ensemble and Hybrid Approaches:

- Combine predictions from multiple models or techniques (ensemble methods) to improve robustness and reliability.

Probabilistic Forecasting:

- Move beyond deterministic predictions to provide probabilistic forecasts that quantify uncertainty around delay predictions.
- Incorporate techniques such as Bayesian inference or Monte Carlo simulations to estimate the likelihood of different delay scenarios.

Continuous Model Improvement and Adaptation:

- Implement mechanisms for continuous model training and updating to adapt to evolving patterns and seasonal variations in flight operations.
- Use feedback loops from actual delay outcomes to refine models and improve prediction accuracy over time.

Collaboration and Data Sharing:

- Foster collaboration among airlines, airports, air traffic control agencies, and weather services to share data and insights that can improve prediction capabilities.
- Establish data-sharing agreements and protocols to ensure privacy and security while facilitating joint efforts in delay prediction research.

Operational Decision Support Systems:

- Develop decision support systems that provide actionable recommendations to airlines and airports for proactive management of delays.
- Integrate prediction models with operational planning tools to optimize resource allocation, crew scheduling, and contingency planning in response to predicted delays.

Regulatory and Policy Considerations:

- Advocate for policies that support the adoption of advanced predictive analytics in aviation operations.
- Encourage regulatory bodies to facilitate the sharing of anonymized data and promote innovation in delay prediction technologies.
- Implementing these proposed solutions requires a concerted effort from stakeholders across the aviation industry, supported by investment in technology, infrastructure, and collaborative partnerships. By leveraging advanced analytics and data-driven approaches, the goal is to enhance the accuracy, timeliness, and reliability of flight delay predictions, ultimately improving the efficiency and passenger in air travel

Algorithm:

Random forest:

Random Forest can effectively predict flight delays by analyzing various features such as weather conditions, airline performance, and historical flight data. It builds multiple decision trees on random subsets of data and features, enhancing prediction accuracy and robustness. The ensemble nature of Random Forest reduces overfitting and handles complex, non-linear relationships in the data. By averaging the predictions from all trees, it provides reliable delay predictions. Feature importance scores from the model can also highlight key factors influencing delays.

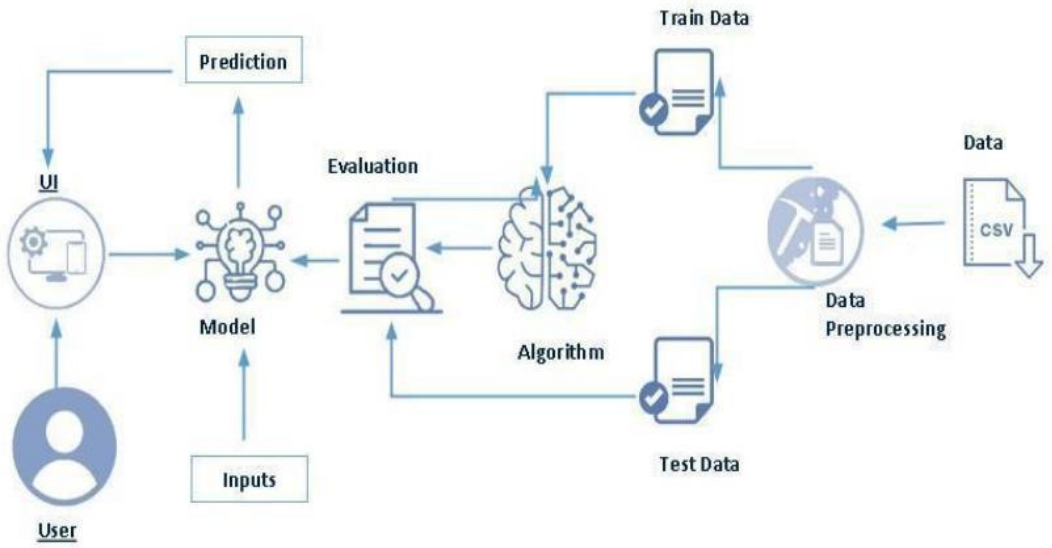
- **Data Preparation**: Collect and preprocess historical flight data, including features such as weather conditions, flight schedules, and airline performance metrics.

- **Model Training**: Train a Random Forest model on this data by building multiple decision trees on random subsets of the data and features, learning patterns associated with flight delays.

- **Prediction and Analysis**: Use the trained model to predict delays for new flights, averaging the predictions from all trees for accuracy, and analyze feature importance to identify key factors affecting delays.

3.THEORITICAL ANALYSIS

3.1 BLOCK DIAGRAM



3.2 SOFTWARE DESIGNING

The following is the Software required to complete this project:

Visual studio code:Visual Studio is a powerful IDE, but when it comes specifically to data reading, preprocessing, and modeling for machine learning projects, the focus typically shifts to using libraries and frameworks within Python, such as Jupyter Notebooks, Anaconda, or specific Python libraries like Pandas, NumPy, and scikit-learn. These tools are extensively used due to their efficiency and ease of use in handling data and building machine learning models. Here’s how Visual Studio can fit into this workflow:

➢ **Dataset (CSV File)**: The dataset in CSV format is essential for training and testing your predictive model. It should include historical air quality data, weather information, pollutant levels, and other relevant features.

➢ **Data Preprocessing Tools**: Python libraries like NumPy, Pandas, and Scikit-learn will be used to preprocess the dataset. This includes handling missing data, feature scaling, and data cleaning.

10➢ **Feature Selection/Drop**: Feature selection or dropping unnecessary features from the dataset can be done using Scikit-learn or custom Python code to enhance the model's efficiency.

➢ **Model Training Tools**: Machine learning libraries such as Scikit-learn, TensorFlow, or PyTorch will be used to develop, train, and fine-tune the predictive model. Regression or classification models can be considered, depending on the nature of the Flight delay prediction task.

➢ **Model Accuracy Evaluation**: After model training, accuracy and performance evaluation tools, such as Scikit-learn metrics or custom validation scripts, will assess the model's predictive capabilities. You'll measure the model's ability to predict Flight Delays categories based on historical data.

➢ **UI Based on Flask Environment**: Flask, a Python web framework, will be used to develop the user interface (UI) for the system. The Flask application will provide a user-friendly platform for users to input Location data or view A Flight Delays predictions, delay information.

In summary, while Visual Studio itself is not typically used for data reading, preprocessing, and modeling directly, it complements these tasks by providing a robust environment for Python development and integration with tools and libraries commonly used in data science and machine learning workflows. For data-intensive projects, developers often switch between Visual Studio (for coding and managing projects) and specialized tools like Jupyter Notebooks (for interactive data exploration and experimentation) based on the specific requirements and preferences of the project.

4.EXPERIMENTAL INVESTIGATION

In this project, we have used Flight delays

Dataset. This dataset is a csv file consisting of labelled data and having the following columns-

YEAR: The calendar year when the flight occurred.

QUARTER: The three-month period (Q1, Q2, Q3, Q4) within the calendar year.

MONTH: The specific month when the flight took place (January to December).

DAY_OF_MONTH: The numerical day of the month (1 to 31) when the flight departed.

DAY_OF_WEEK: The day of the week (Monday to Sunday) when the flight departed.

UNIQUE_CARRIER: The unique code or identifier for the airline carrier operating the flight.

TAIL_NUM: The unique tail number of the aircraft.

FL_NUM: The flight number assigned by the airline for identification.

ORIGIN_AIRPORT_ID: The unique identifier for the origin airport.

ORIGIN: The code or identifier for the origin airport.

DEST_AIRPORT_ID: The unique identifier for the destination airport.

DEST: The code or identifier for the destination airport.

CRS_DEP_TIME: The scheduled departure time of the flight (local time).

DEP_TIME: The actual departure time of the flight (local time).

DEP_DELAY: The difference in minutes between the actual departure time and the scheduled departure time.

DEP_DEL15: Indicates if the flight departure was delayed by 15 minutes or more (0 = No, 1 = Yes).

CRS_ARR_TIME: The scheduled arrival time of the flight (local time).

ARR_TIME: The actual arrival time of the flight (local time).

ARR_DELAY: The difference in minutes between the actual arrival time and the scheduled arrival time.

ARR_DEL15: Indicates if the flight arrival was delayed by 15 minutes or more (0 = No, 1 = Yes).

CANCELLED: Indicates if the flight was cancelled (0 = No, 1 = Yes).

DIVERTED: Indicates if the flight was diverted to an alternate airport (0 = No, 1 = Yes).

CRS_ELAPSED_TIME: The scheduled elapsed time of the flight in minutes.

ACTUAL_ELAPSED_TIME: The actual elapsed time of the flight in minutes.

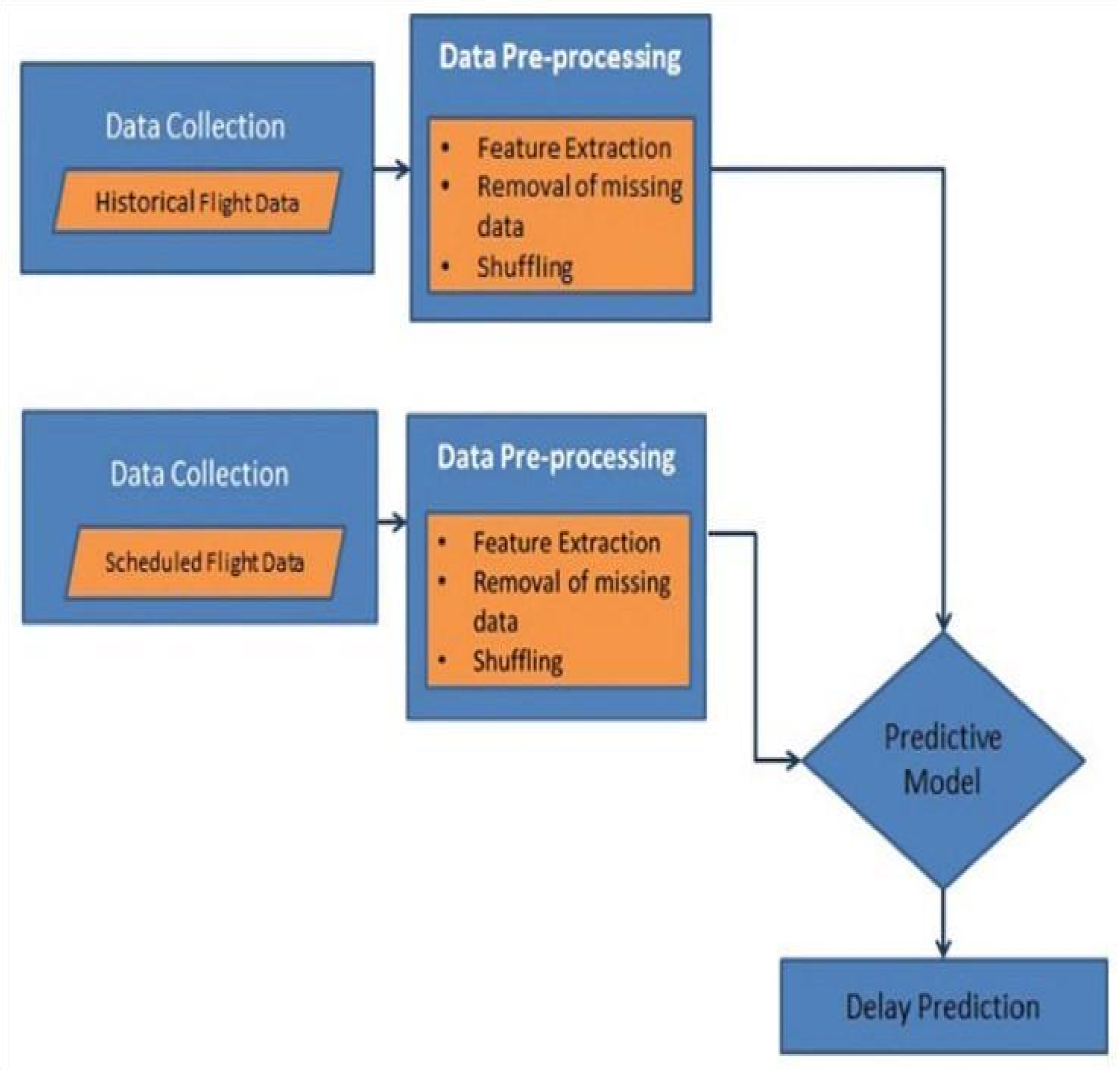
DISTANCE: The distance traveled by the flight in miles.

For the dataset we selected ,it consist of more than the columns we want to predict it So,we have chose the feature drop it contains the columns that I am going to predict the flight delays.

- Feature drop means it drops the columns that we don’t want in our dataset.

- Dataset=dataset.drop(“unnamed:25”,axis=1)

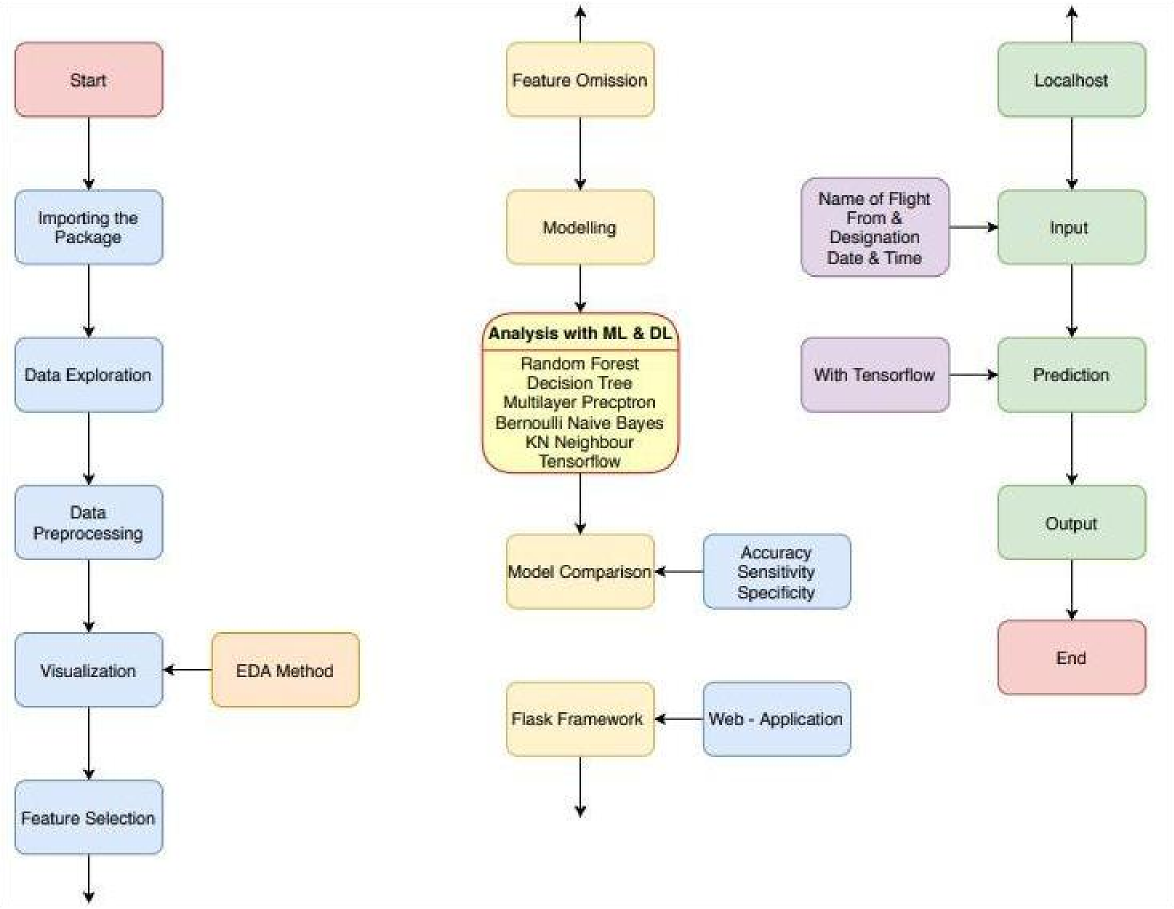
4.BLOCK DIAGRAM OF EXISTING SYSTEMS



4.1 LIMITAIONS OF EXISTING SYSTEM

- Limited to structured data;may struggle with unstructured data.
- Vulnerable to over fitting, especially with complex datasets.
- Time-consuming training process,especially with large datasets.
- Challenges in interpretation due to ensemble complexity

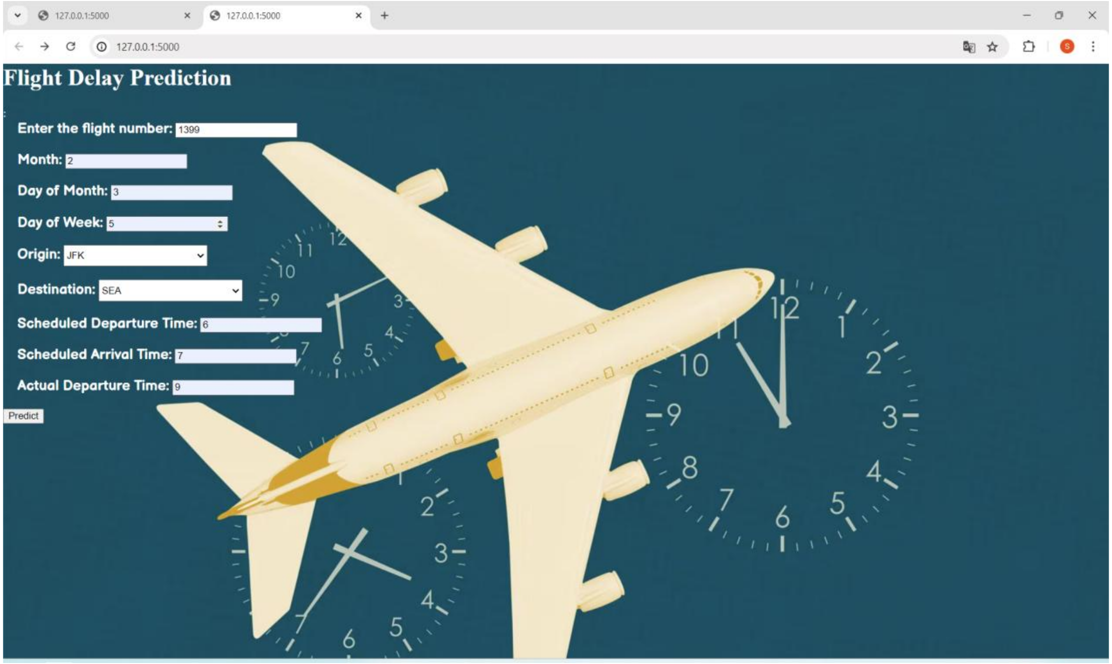
5 . FLOW CHART


Start
Importing the Package
Data Exploration
Data Preprocessing
Visualization
EDA Method
Feature Selection
Feature Omission
Modelling
Analysis with ML & DL
Random Forest
Decision Tree
Multilayer Precptron
Bernoulli Naive Bayes
KN Neighbour
Tensorflow
Model Comparison
Accuracy
Sensitivity
Specificity
Flask Framework
Web - Application
Localhost
Name of Flight
From &
Designation
Date & Time
Input
With Tensorflow
Prediction
Output
End

6.RESULT

The outcomes typically invoves the accuracy and effectiveness of the prediction model used in the project.the result page might include details on how well the model performed in forecasting flight delays.

6.1 PREDICTION PAGE:



Flight Delay Prediction

Enter the flight number: 1399

Month: 2

Day of Month: 3

Day of Week: 5

Origin: JFK

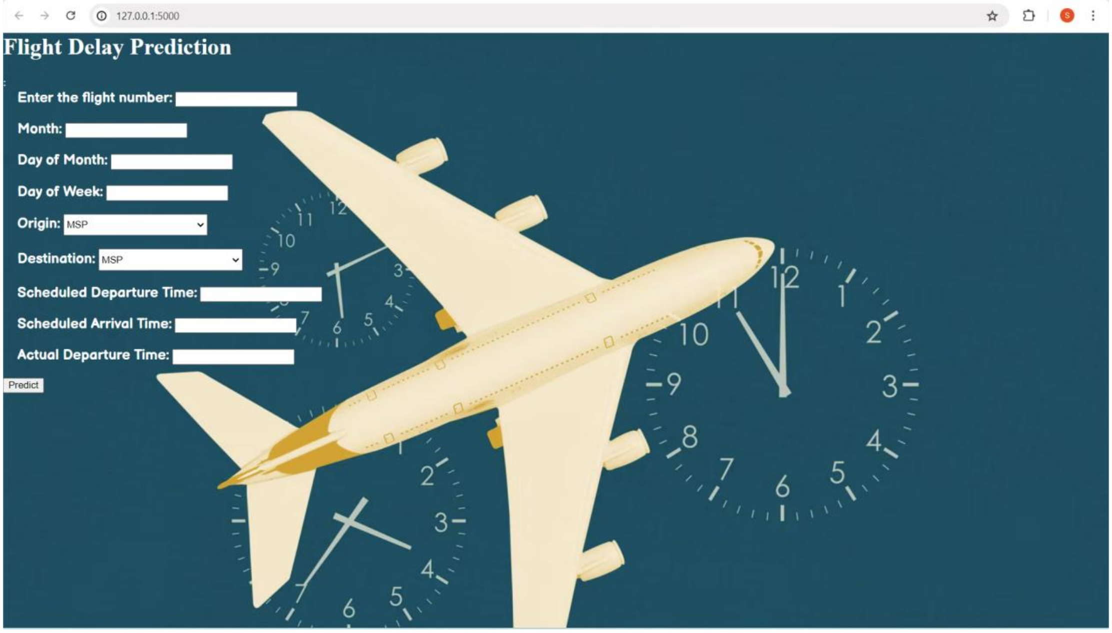
Destination: SEA

Scheduled Departure Time: 6

Scheduled Arrival Time: 7

Actual Departure Time: 9

Enter your flight details to check if your flight is delayed or on time."



Flight Delay Prediction

Enter the flight number:

Month:

Day of Month:

Day of Week:

Origin: MSP

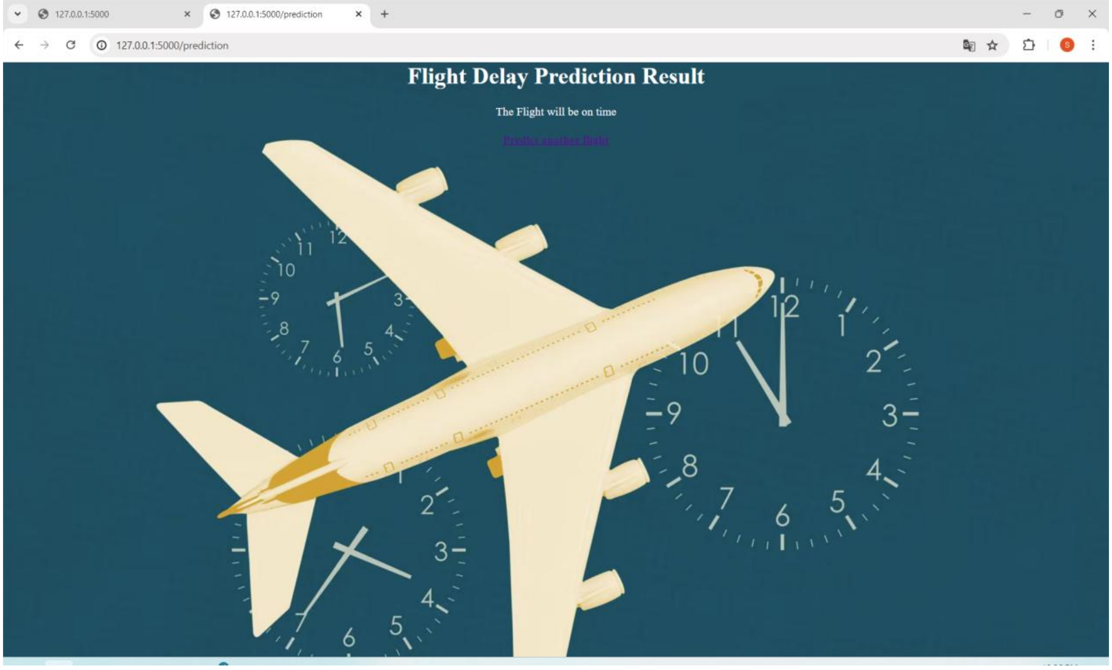
Destination: MSP

Scheduled Departure Time:

Scheduled Arrival Time:

Actual Departure Time:

6.2 PREDICTION RESULTS:

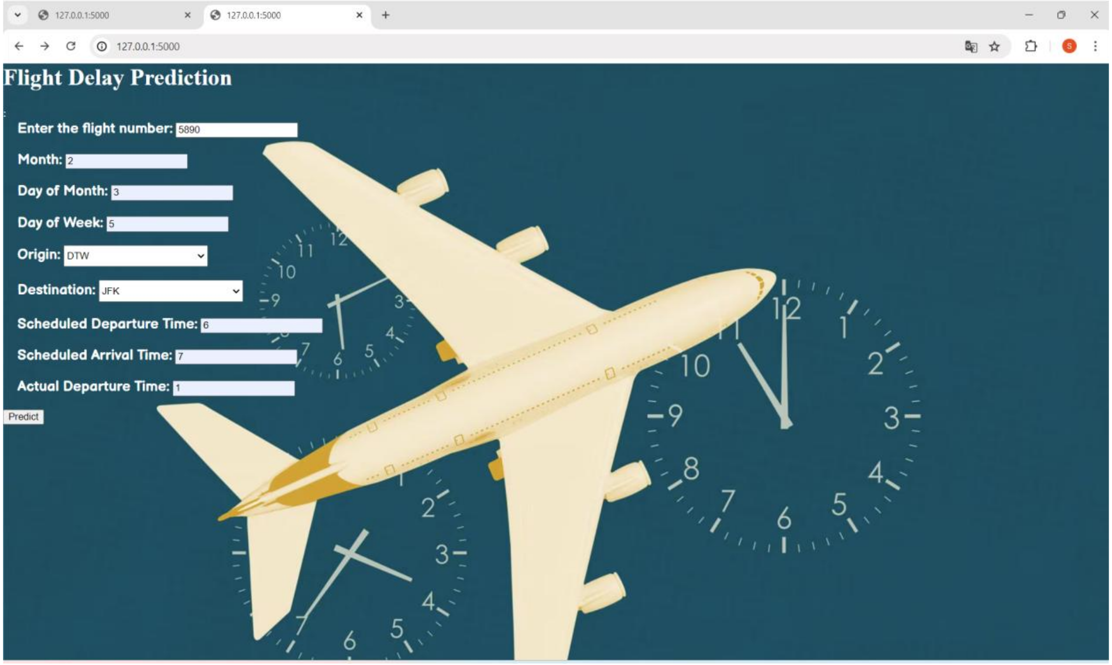


Flight Delay Prediction Result

The Flight will be on time

This image tells ,Your flight is currently scheduled to depart and arrive on time. Please note that flight status can change, so it's always a good idea to check for updates before your Journey.Have a safe and pleasant flight

You want to check the status of another flight? Simply click on "Predict another flight" and enter the new flight details



Flight Delay Prediction

Enter the flight number: 5890

Month: 2

Day of Month: 3

Day of Week: 5

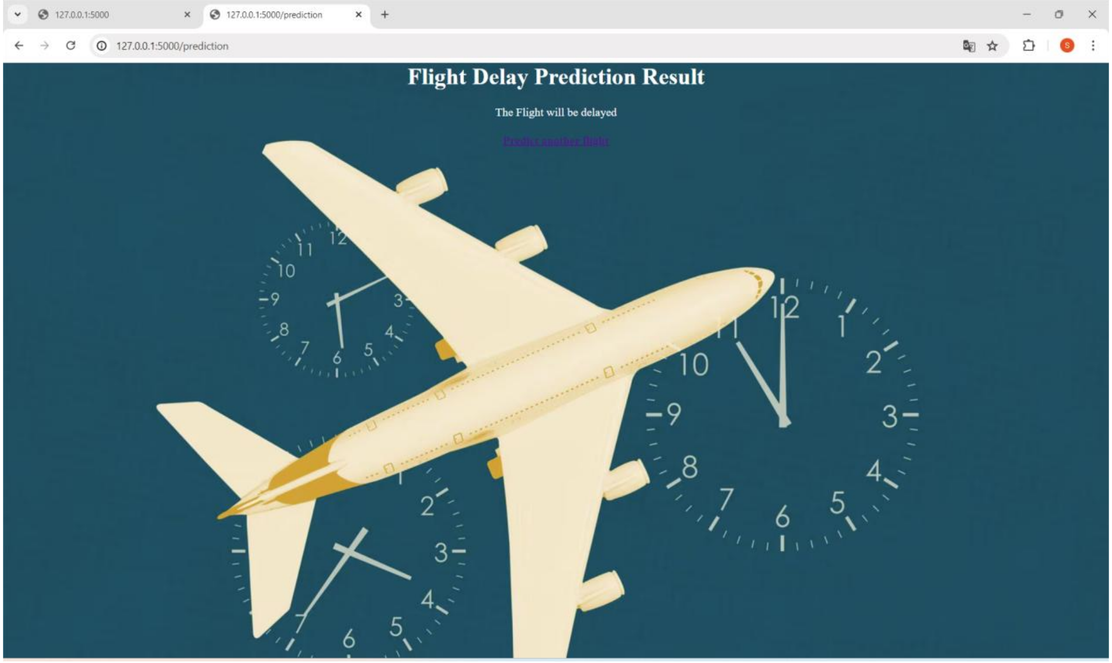
Origin: DTW

Destination: JFK

Scheduled Departure Time: 6

Scheduled Arrival Time: 7

Actual Departure Time: 1



Flight Delay Prediction Result

The Flight will be delayed

Sorry to inform you that your flight is expected to delayed

7. ADVANTAGES AND DISADVANTAGES

ADVATAGES:

While flight delays can be inconvenient and frustrating for travelers, there are a few potential advantages or silver linings that can come from them:

1.Safety Concerns: Sometimes flight delays occur due to safety reasons, such as weather conditions or technical issues. In such cases, the delay ensures that potential risks are addressed before the flight proceeds, prioritizing passenger safety.

2.Opportunity to Rest or Plan: Flight delays can provide passengers with unexpected downtime. This can be a chance to relax, catch up on work, read, or even plan for upcoming activities at the destination.

3.Connecting Flight Adjustment: If you have a connecting flight and the first leg is delayed, airlines often adjust the schedule to accommodate affected passengers. This can prevent missed connections and the subsequent inconvenience of rebooking.

4.Compensation: Depending on the regulations and the circumstances of the delay, passengers may be entitled to compensation or amenities such as meal vouchers, hotel accommodations, or transportation services. This can mitigate some of the inconvenience caused by the delay.

5.Meeting New People: Flight delays can create opportunities for social interaction. Passengers often find themselves in the same situation, leading to conversations and potential new connections or friendships.

6.Appreciation of Timeliness: Experiencing delays occasionally can make travelers more appreciative of smooth and on-time flights in the future. It can also provide a chance to reflect on the complexities of air travel and the efforts involved in keeping flights on schedule.

While flight delays are generally seen as a negative aspect of air travel, these potential advantages can help mitigate some of the frustration and inconvenience they causes

DISADVANTAGES:

Flight delays can certainly be frustrating and come with several disadvantages for passengers:

1.Missed Connections: One of the most significant drawbacks of flight delays is the potential to miss connecting flights. This can disrupt travel plans and lead to further delays in reaching the final destination.

2.Time Wasted: Passengers often spend extended periods of time waiting in airports due to flight delays. This can result in wasted time that could have been used more productively.

3.Financial Loss: Delays can lead to financial losses, especially if passengers miss pre-booked activities, accommodation bookings, or incur additional costs for rebooking flights or changing travel plans.

4.Discomfort and Inconvenience: Being stuck in an airport for an extended period can be uncomfortable and inconvenient, particularly if amenities such as food, seating, or rest areas are insufficient.

5.Increased Stress: Flight delays can cause stress and anxiety, particularly when passengers are unsure of the duration of the delay or whether they will reach their destination on time.

5.**Impact on Plans**: Delays can disrupt carefully planned itineraries, affecting business meetings, family gatherings, or vacation schedules. This can lead to disappointment and frustration among travelers.

6.Customer Service Challenges: Airlines may struggle to manage the expectations and needs of affected passengers during delays, potentially leading to dissatisfaction and strained customer relations.

7.Health Concerns: Prolonged delays can impact passenger well-being, leading to fatigue, discomfort, and even health issues such as stress-related conditions or physical discomfort from prolonged sitting.

Overall, flight delays can have significant negative impacts on passengers' travel experiences,affecting everything from time management

8. APPLICATIONS

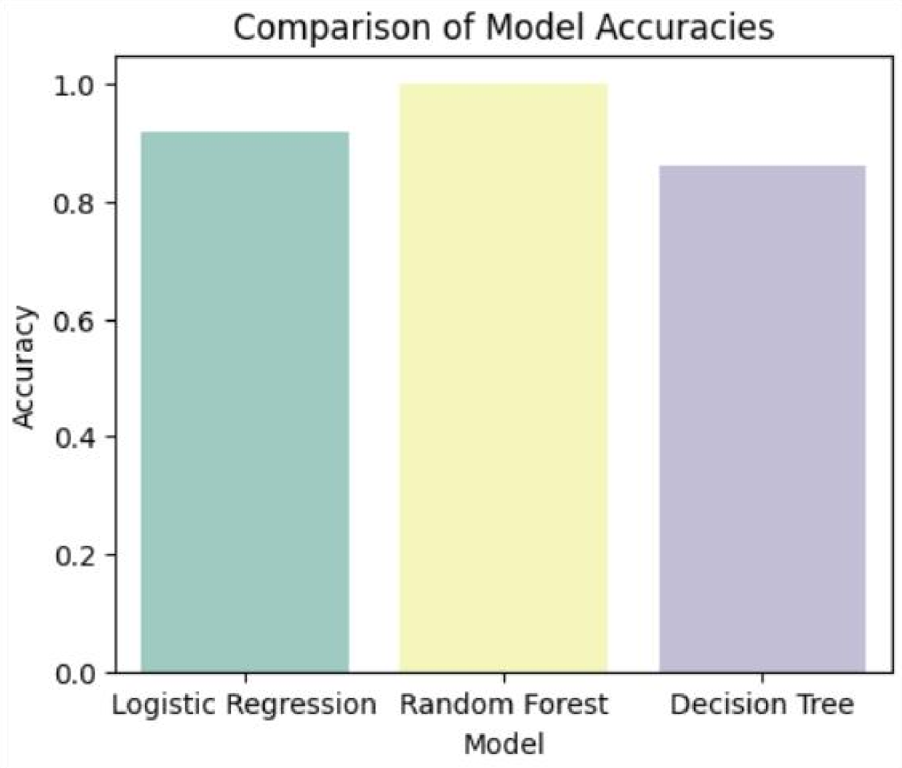
- **Operational Efficiency**: Flight delay data helps airlines optimize schedules and resource allocation.
- **Passenger Communication**: Real-time delay information assists passengers in making informed travel decisions.
- **Regulatory Reporting**: Airlines use delay data to comply with aviation safety and consumer protection regulations.
- **Predictive Analytics**: Analyzing delay patterns enables airlines to predict and mitigate future disruptions.
- **Airport Management**: Airports utilize delay data for efficient terminal and gate operations.

9. CONCLUSION

In conclusion, flight delays pose numerous challenges and inconveniences for travelers, airlines, and airports alike. They can lead to missed connections, wasted time, financial losses, and increased stress. However, advancements in technology and data analytics allow airlines and airports to better manage delays, improve operational efficiency, and enhance passenger communication. Despite these efforts, mitigating delays remains a complex task that requires ongoing collaboration and innovation within the aviation industry.

10.Result Analysis

1.1 Accuracy comparison graphs



Algorithm	**Accuracy**
Logistic Regression	0.92
Random Forest	0.91
Decision Tree	0.86

1.2 Accuracy comparison table

11.FUTURE SCOPE

In the future, addressing flight delays will focus on:

1.Real-Time Data Integration: Utilizing advanced data integration technologies to provide real-time updates and proactive solutions to mitigate delays.

2.Automation and AI: Implementing automation and artificial intelligence to streamline operational processes and predict potential delays more accurately.

3.Infrastructure Modernization: Investing in modern airport infrastructure and air traffic management systems to enhance efficiency and reduce delays.

4.Weather Prediction and Mitigation: Advancing weather forecasting technologies to better predict and mitigate weather-related delays.

5.Collaborative Solutions: Strengthening collaboration among airlines, airports, and regulatory bodies to develop holistic strategies for minimizing delays and improving overall travel experience.

12. APPENDIX

Model building :

1)Dataset

2)VS code Application Building

1. HTML file (Index file, Predict file)

2. Models in pickle format

SOURCE CODE:

INDEX.HTML

```
<html>
<style>
@import
url('https://fonts.googleapis.com/css2?family=Balsamiq+Sans:wght@700&display=swap');
body
{
  width:100%;
  margin:0px;
  color:white;
  background-image:
url("https://www.washingtonpost.com/resizer/XDNLgd1ihU5u3te6FFp4haLzTRg=/1484x0/arc-anglerfish-washpost-prod-washpost.s3.amazonaws.com/public/PBKJ5C6KJJC75BO46RZEWUGL6A.jpg");


}
```

```
.header{

top:0;

width:100%;

height:90px

font-family: 'Balsamiq Sans', cursive;

font-size:25px;

font-weight:800px;

text-align: center;

}

.MAIN  p,label{

 font-size:20px;

 margin-left:20px;

  font-family: 'Balsamiq Sans', cursive;

}

.MAIN  input,select

{

height:30px;

width:200px;

}

.MAIN button

{

height:30px;

width:200px;
margin-left:60px;

background-color:#daa520;

}

.MAIN b{

font-size:20px;

font-weight:800px;

text-align:center;

font-family: 'Balsamiq Sans', cursive;

margin-left:20px;

}



</style>

<body>

   <h1>Flight Delay Prediction</h1>:

   <form action="/prediction" method="post">

       <label for="enter the flight number">Enter the flight number:</label>

       <input type="number" id="enter the flight number" name="enter the flight
number" required><br><br>

       <label for="month">Month:</label>

       <input type="number" id="month" name="month" required><br><br>



       <label for="dayofmonth">Day of Month:</label>

       <input type="number" id="dayofmonth" name="dayofmonth"
required><br><br>
```

```
<label for="dayofweek">Day of Week:</label>
<input type="number" id="dayofweek" name="dayofweek" required><br><br>

<label for="origin">Origin:</label>
<select id="origin" name="origin" required>
  <option value="msp">MSP</option>
  <option value="dtw">DTW</option>
  <option value="jfk">JFK</option>
  <option value="sea">SEA</option>
  <option value="alt">ALT</option>
</select><br><br>

<label for="destination">Destination:</label>
<select id="destination" name="destination" required>
  <option value="msp">MSP</option>
  <option value="dtw">DTW</option>
  <option value="jfk">JFK</option>
  <option value="sea">SEA</option>
  <option value="alt">ALT</option>
</select><br><br>

<label for="dept">Scheduled Departure Time:</label>
```

```
        <input type="number" id="dept" name="dept" required><br><br>


        <label for="arrtime">Scheduled Arrival Time:</label>

        <input type="number" id="arrtime" name="arrtime" required><br><br>


        <label for="actdept">Actual Departure Time:</label>

        <input type="number" id="actdept" name="actdept" required><br><br>


        <button type="submit">Predict</button>

    </form>

</body>

</html>
```

PREDICT.HTML

```
<html>

<style>

@import
url('https://fonts.googleapis.com/css2?family=Balsamiq+Sans:wght@700&display=swap');

body

{

  width:100%;
  margin:0px;
  color:white;
  background-image:
url("https://www.washingtonpost.com/resizer/XDNLgd1ihU5u3te6FFp4haLzTRg=/1
484x0/arc-anglerfish-washpost-prod-
washpost.s3.amazonaws.com/public/PBKJ5C6KJJC75BO46RZEWUGL6A.jpg");


}
.header{
top:0;
width:100%;
height:90px
font-family: 'Balsamiq Sans', cursive;
font-size:25px;
font-weight:800px;
text-align: center;
}
.MAIN  p,label{
 font-size:20px;
 margin-left:20px;
  font-family: 'Balsamiq Sans', cursive;
}
.MAIN  input,select
{
height:30px;
```

```
width:200px;

}

.MAIN button

{

height:30px;

width:200px;

margin-left:60px;

background-color:#daa520;

}

.MAIN b{

font-size:20px;

font-weight:800px;

text-align:center;

font-family: 'Balsamiq Sans', cursive;

margin-left:20px;

}


</style>

<body> <center>

   <h1>Flight Delay Prediction Result</h1>

   <p>{{ showcase }}</p>

   <a href="/"><h4>Predict another flight</h4></a>

</center>
```

```
</body>

</html>
```

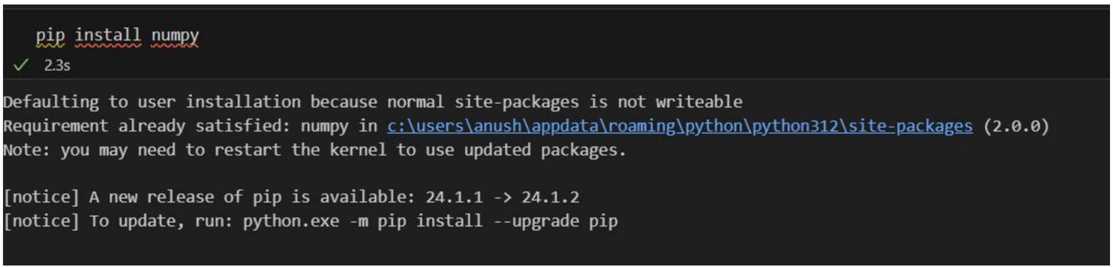
APP.PY

```
from flask import Flask, render_template, request

import pickle

import numpy as np


model = pickle.load(open("E:/FlightDelayPrediction_M/flight.pkl", 'rb'))


app = Flask(__name__)


@app.route('/')

def home():

    return render_template("index.html")


@app.route('/prediction', methods=['POST'])

def prediction():

    #number=int(request.form['enter the flight number'])

    month = int(request.form['month'])

    dayofmonth = int(request.form['dayofmonth'])

    dayofweek = int(request.form['dayofweek'])
```

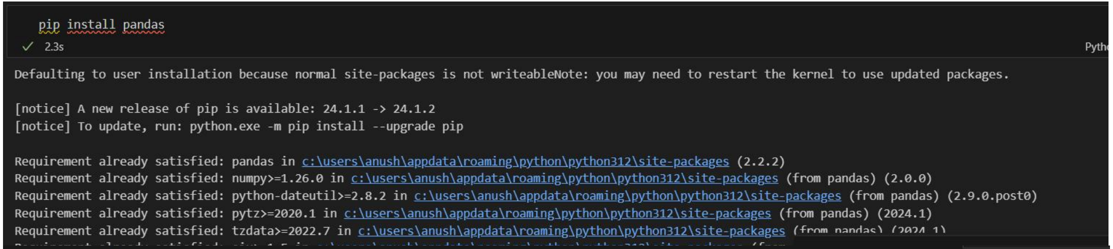
```
origin = request.form['origin']

if origin == "msp":

    origin = 1

elif origin == "dtw":

    origin = 2

elif origin == "jfk":

    origin = 3

elif origin == "sea":

    origin = 4

elif origin == "alt":

    origin = 5


destination = request.form['destination']

if destination == "msp":

    destination = 1

elif destination == "dtw":

     destination = 2

elif destination == "jfk":

    destination = 3

elif destination == "sea":

    destination = 4

elif destination == "alt":
        destination = 5

    dept = int(request.form['dept'])
    arrtime = int(request.form['arrtime'])
    actdept = int(request.form['actdept'])
    dept15 = dept - actdept

    total = np.array([[month, dayofmonth, dayofweek, origin, destination, dept, arrtime, dept15]])

    y_pred = model.predict(total)

    ans = 'The Flight will be on time' if y_pred[0] == 0 else 'The Flight will be delayed'

    return render_template("predict.html", showcase=ans)

if __name__ == '__main__':
    app.run(debug=True)
```

CODE SNIPPETS

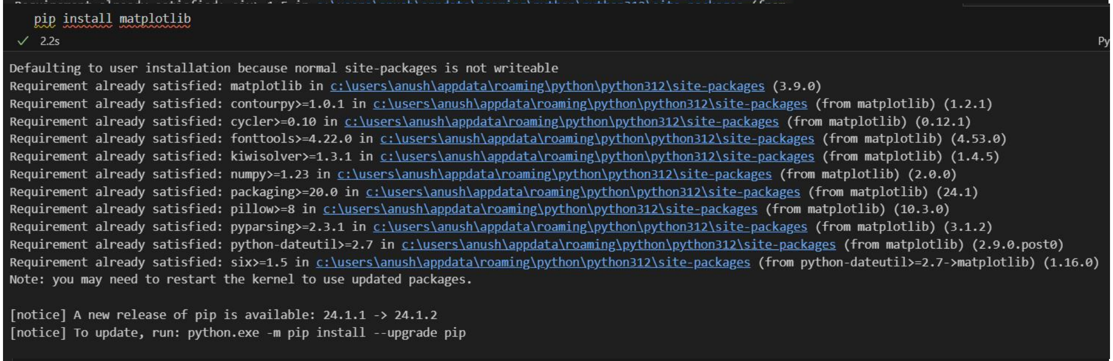
Installation:



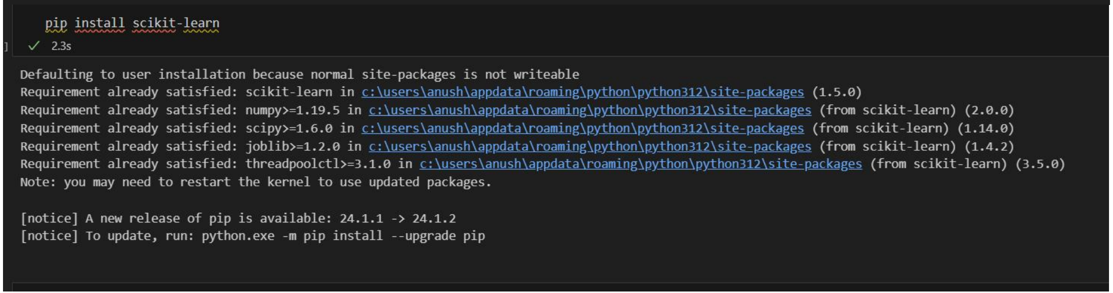
```
pip install numpy
✓ 2.3s

Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: numpy in c:\users\anush\appdata\roaming\python\python312\site-packages (2.0.0)
Note: you may need to restart the kernel to use updated packages.

[notice] A new release of pip is available: 24.1.1 -> 24.1.2
[notice] To update, run: python.exe -m pip install --upgrade pip
```



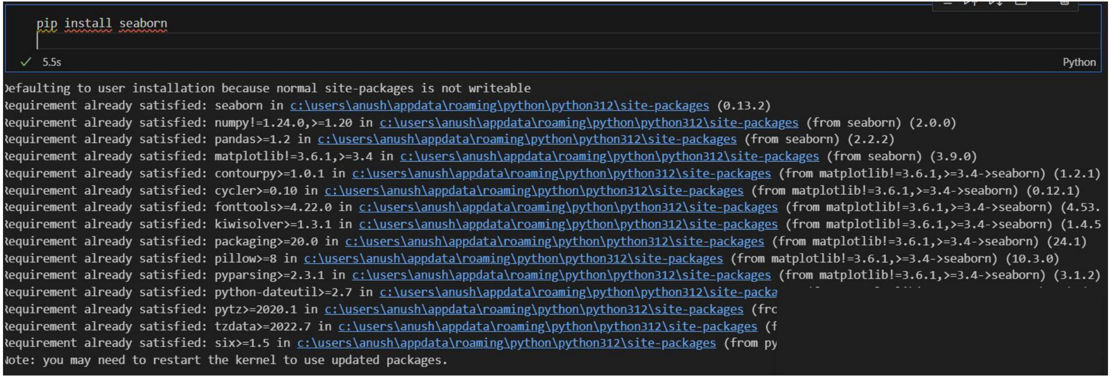
```
pip install pandas
✓ 2.3s                                                                                                              Pyth

Defaulting to user installation because normal site-packages is not writeableNote: you may need to restart the kernel to use updated packages.

[notice] A new release of pip is available: 24.1.1 -> 24.1.2
[notice] To update, run: python.exe -m pip install --upgrade pip

Requirement already satisfied: pandas in c:\users\anush\appdata\roaming\python\python312\site-packages (2.2.2)
Requirement already satisfied: numpy>=1.26.0 in c:\users\anush\appdata\roaming\python\python312\site-packages (from pandas) (2.0.0)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\anush\appdata\roaming\python\python312\site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\anush\appdata\roaming\python\python312\site-packages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\anush\appdata\roaming\python\python312\site-packages (from pandas) (2024.1)
```



```
pip install matplotlib
✓ 2.2s                                                                                                              Py

Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: matplotlib in c:\users\anush\appdata\roaming\python\python312\site-packages (3.9.0)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\anush\appdata\roaming\python\python312\site-packages (from matplotlib) (1.2.1)
Requirement already satisfied: cycler>=0.10 in c:\users\anush\appdata\roaming\python\python312\site-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\anush\appdata\roaming\python\python312\site-packages (from matplotlib) (4.53.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\anush\appdata\roaming\python\python312\site-packages (from matplotlib) (1.4.5)
Requirement already satisfied: numpy>=1.23 in c:\users\anush\appdata\roaming\python\python312\site-packages (from matplotlib) (2.0.0)
Requirement already satisfied: packaging>=20.0 in c:\users\anush\appdata\roaming\python\python312\site-packages (from matplotlib) (24.1)
Requirement already satisfied: pillow>=8 in c:\users\anush\appdata\roaming\python\python312\site-packages (from matplotlib) (10.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\anush\appdata\roaming\python\python312\site-packages (from matplotlib) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\anush\appdata\roaming\python\python312\site-packages (from matplotlib) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in c:\users\anush\appdata\roaming\python\python312\site-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
Note: you may need to restart the kernel to use updated packages.

[notice] A new release of pip is available: 24.1.1 -> 24.1.2
[notice] To update, run: python.exe -m pip install --upgrade pip
```



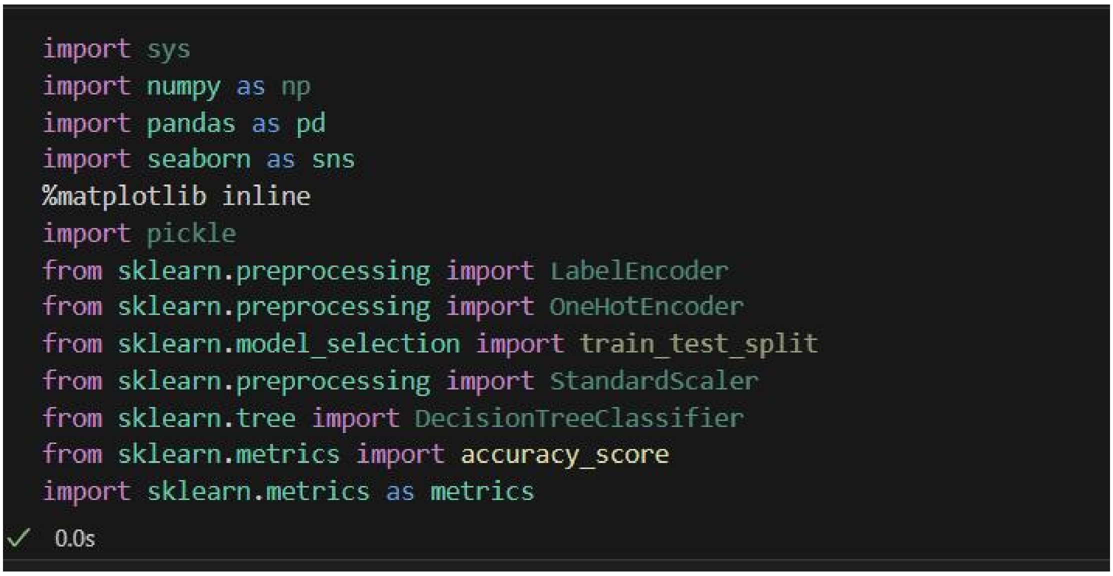
```
pip install scikit-learn
✓ 2.3s

Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: scikit-learn in c:\users\anush\appdata\roaming\python\python312\site-packages (1.5.0)
Requirement already satisfied: numpy>=1.19.5 in c:\users\anush\appdata\roaming\python\python312\site-packages (from scikit-learn) (2.0.0)
Requirement already satisfied: scipy>=1.6.0 in c:\users\anush\appdata\roaming\python\python312\site-packages (from scikit-learn) (1.14.0)
Requirement already satisfied: joblib>=1.2.0 in c:\users\anush\appdata\roaming\python\python312\site-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in c:\users\anush\appdata\roaming\python\python312\site-packages (from scikit-learn) (3.5.0)
Note: you may need to restart the kernel to use updated packages.

[notice] A new release of pip is available: 24.1.1 -> 24.1.2
[notice] To update, run: python.exe -m pip install --upgrade pip
```



```
pip install seaborn

✓ 5.5s                                                                                                              Python

Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: seaborn in c:\users\anush\appdata\roaming\python\python312\site-packages (0.13.2)
Requirement already satisfied: numpy!=1.24.0,>=1.20 in c:\users\anush\appdata\roaming\python\python312\site-packages (from seaborn) (2.0.0)
Requirement already satisfied: pandas>=1.2 in c:\users\anush\appdata\roaming\python\python312\site-packages (from seaborn) (2.2.2)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in c:\users\anush\appdata\roaming\python\python312\site-packages (from seaborn) (3.9.0)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\anush\appdata\roaming\python\python312\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.2.1)
Requirement already satisfied: cycler>=0.10 in c:\users\anush\appdata\roaming\python\python312\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\anush\appdata\roaming\python\python312\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (4.53.
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\anush\appdata\roaming\python\python312\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (1.4.5
Requirement already satisfied: packaging>=20.0 in c:\users\anush\appdata\roaming\python\python312\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (24.1)
Requirement already satisfied: pillow>=8 in c:\users\anush\appdata\roaming\python\python312\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (10.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\anush\appdata\roaming\python\python312\site-packages (from matplotlib!=3.6.1,>=3.4->seaborn) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\anush\appdata\roaming\python\python312\site-packa
Requirement already satisfied: pytz>=2020.1 in c:\users\anush\appdata\roaming\python\python312\site-packages (frc
Requirement already satisfied: tzdata>=2022.7 in c:\users\anush\appdata\roaming\python\python312\site-packages (f
Requirement already satisfied: six>=1.5 in c:\users\anush\appdata\roaming\python\python312\site-packages (from py
Note: you may need to restart the kernel to use updated packages.
```

Data preprocessing:

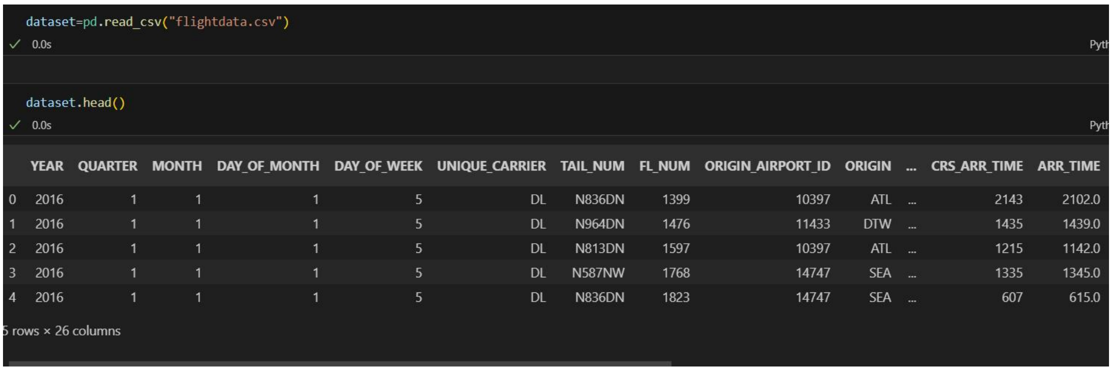
importing the libraries:



```
import sys
import numpy as np
import pandas as pd
import seaborn as sns
%matplotlib inline
import pickle
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
import sklearn.metrics as metrics
```

✓ 0.0s

Read the dataset:



```
dataset=pd.read_csv("flightdata.csv")
```

✓ 0.0s Pyth

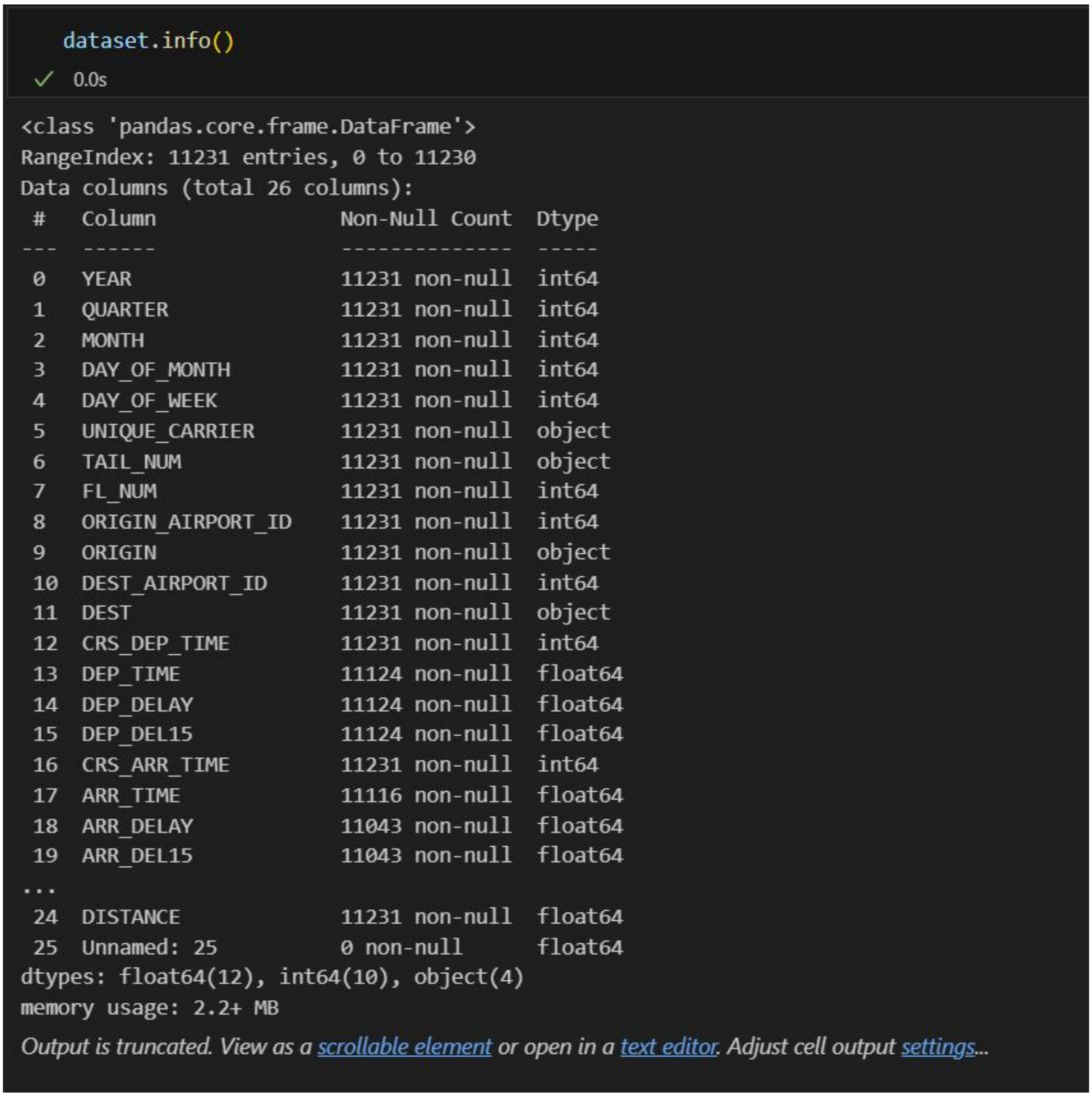
```
dataset.head()
```

✓ 0.0s Pyth

	YEAR	QUARTER	MONTH	DAY_OF_MONTH	DAY_OF_WEEK	UNIQUE_CARRIER	TAIL_NUM	FL_NUM	ORIGIN_AIRPORT_ID	ORIGIN	...	CRS_ARR_TIME	ARR_TIME
0	2016	1	1	1	5	DL	N836DN	1399	10397	ATL	...	2143	2102.0
1	2016	1	1	1	5	DL	N964DN	1476	11433	DTW	...	1435	1439.0
2	2016	1	1	1	5	DL	N813DN	1597	10397	ATL	...	1215	1142.0
3	2016	1	1	1	5	DL	N587NW	1768	14747	SEA	...	1335	1345.0
4	2016	1	1	1	5	DL	N836DN	1823	14747	SEA	...	607	615.0

5 rows × 26 columns

Analyse the data :

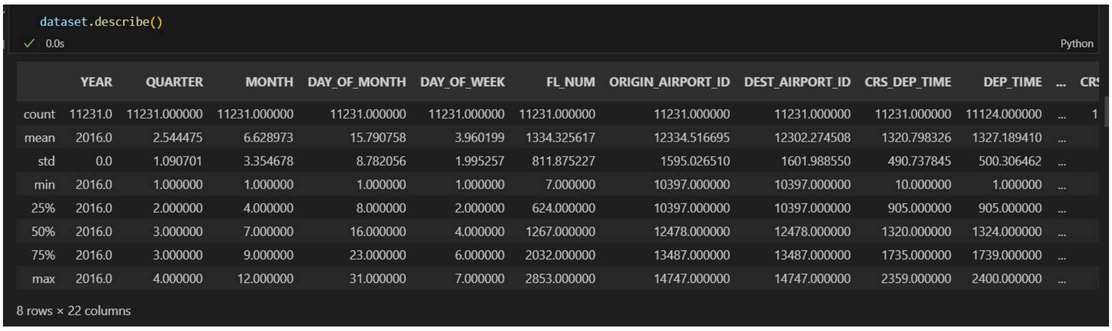


```
dataset.info()
```

✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11231 entries, 0 to 11230
Data columns (total 26 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   YEAR               11231 non-null  int64  
 1   QUARTER            11231 non-null  int64  
 2   MONTH              11231 non-null  int64  
 3   DAY_OF_MONTH       11231 non-null  int64  
 4   DAY_OF_WEEK        11231 non-null  int64  
 5   UNIQUE_CARRIER     11231 non-null  object 
 6   TAIL_NUM           11231 non-null  object 
 7   FL_NUM             11231 non-null  int64  
 8   ORIGIN_AIRPORT_ID  11231 non-null  int64  
 9   ORIGIN             11231 non-null  object 
 10  DEST_AIRPORT_ID    11231 non-null  int64  
 11  DEST               11231 non-null  object 
 12  CRS_DEP_TIME       11231 non-null  int64  
 13  DEP_TIME           11124 non-null  float64
 14  DEP_DELAY          11124 non-null  float64
 15  DEP_DEL15          11124 non-null  float64
 16  CRS_ARR_TIME       11231 non-null  int64  
 17  ARR_TIME           11116 non-null  float64
 18  ARR_DELAY          11043 non-null  float64
 19  ARR_DEL15          11043 non-null  float64
...
 24  DISTANCE           11231 non-null  float64
 25  Unnamed: 25        0 non-null      float64
dtypes: float64(12), int64(10), object(4)
memory usage: 2.2+ MB
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...



```
dataset.describe()
```

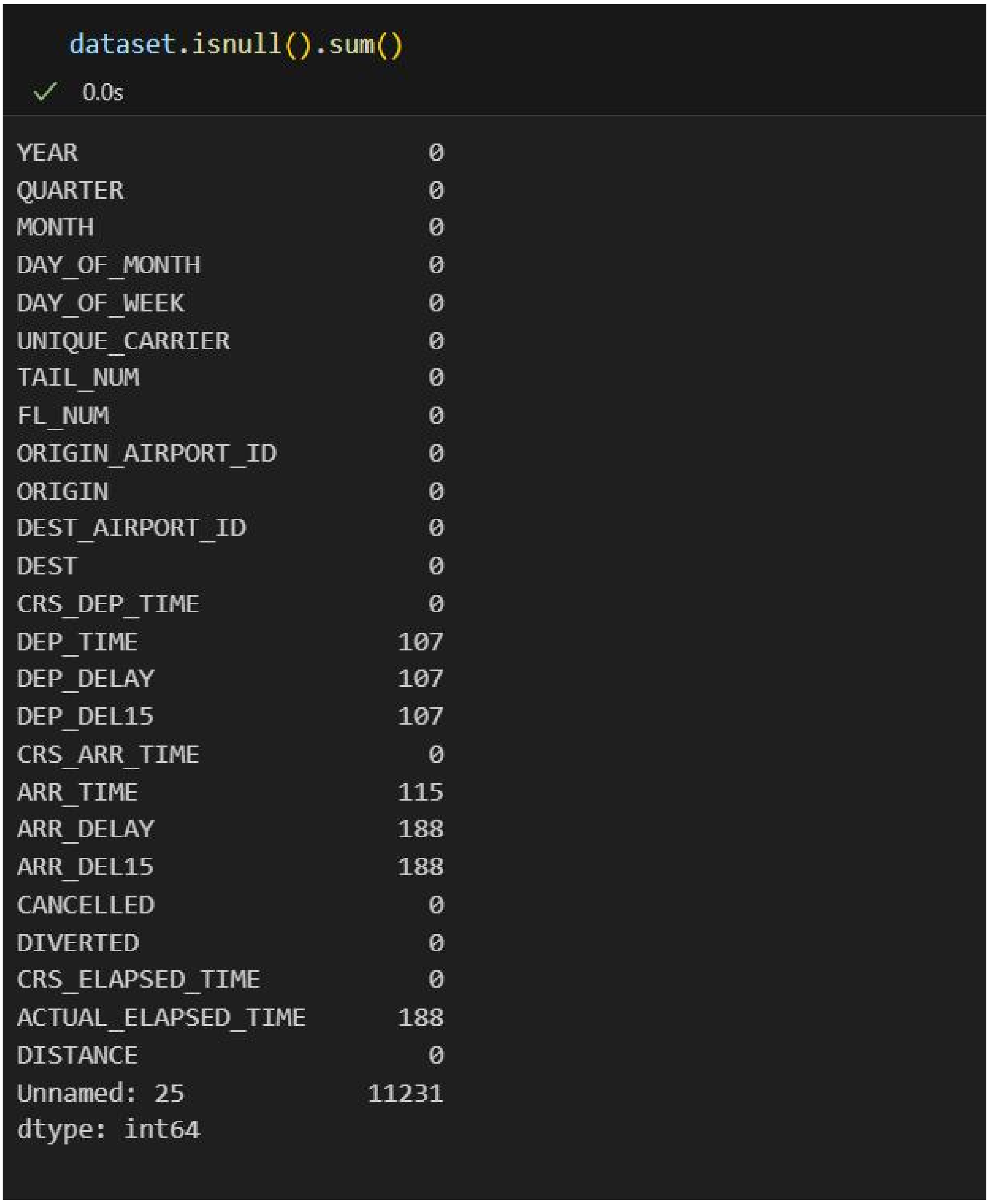
✓ 0.0s

Python

	YEAR	QUARTER	MONTH	DAY_OF_MONTH	DAY_OF_WEEK	FL_NUM	ORIGIN_AIRPORT_ID	DEST_AIRPORT_ID	CRS_DEP_TIME	DEP_TIME	...	CRS
count	11231.0	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11124.000000	...	1
mean	2016.0	2.544475	6.628973	15.790758	3.960199	1334.325617	12334.516695	12302.274508	1320.798326	1327.189410	...	
std	0.0	1.090701	3.354678	8.782056	1.995257	811.875227	1595.026510	1601.988550	490.737845	500.306462	...	
min	2016.0	1.000000	1.000000	1.000000	1.000000	7.000000	10397.000000	10397.000000	10.000000	1.000000	...	
25%	2016.0	2.000000	4.000000	8.000000	2.000000	624.000000	10397.000000	10397.000000	905.000000	905.000000	...	
50%	2016.0	3.000000	7.000000	16.000000	4.000000	1267.000000	12478.000000	12478.000000	1320.000000	1324.000000	...	
75%	2016.0	3.000000	9.000000	23.000000	6.000000	2032.000000	13487.000000	13487.000000	1735.000000	1739.000000	...	
max	2016.0	4.000000	12.000000	31.000000	7.000000	2853.000000	14747.000000	14747.000000	2359.000000	2400.000000	...	

8 rows × 22 columns

Handling missing values:

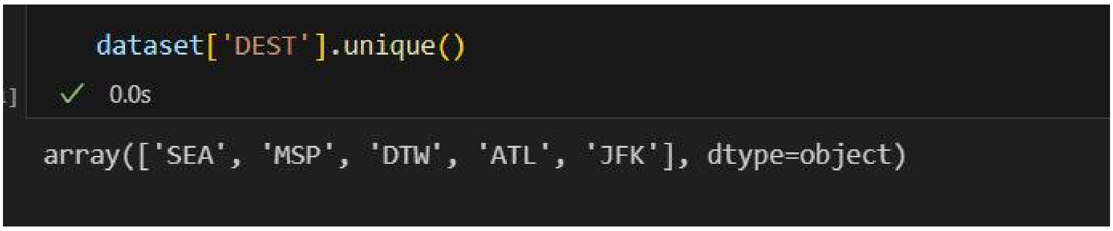


```
dataset.isnull().sum()
```

✓ 0.0s

```
YEAR                       0
QUARTER                    0
MONTH                      0
DAY_OF_MONTH               0
DAY_OF_WEEK                0
UNIQUE_CARRIER             0
TAIL_NUM                   0
FL_NUM                     0
ORIGIN_AIRPORT_ID          0
ORIGIN                     0
DEST_AIRPORT_ID            0
DEST                       0
CRS_DEP_TIME               0
DEP_TIME                 107
DEP_DELAY                107
DEP_DEL15                107
CRS_ARR_TIME               0
ARR_TIME                 115
ARR_DELAY                188
ARR_DEL15                188
CANCELLED                  0
DIVERTED                   0
CRS_ELAPSED_TIME           0
ACTUAL_ELAPSED_TIME      188
DISTANCE                   0
Unnamed: 25            11231
dtype: int64
```

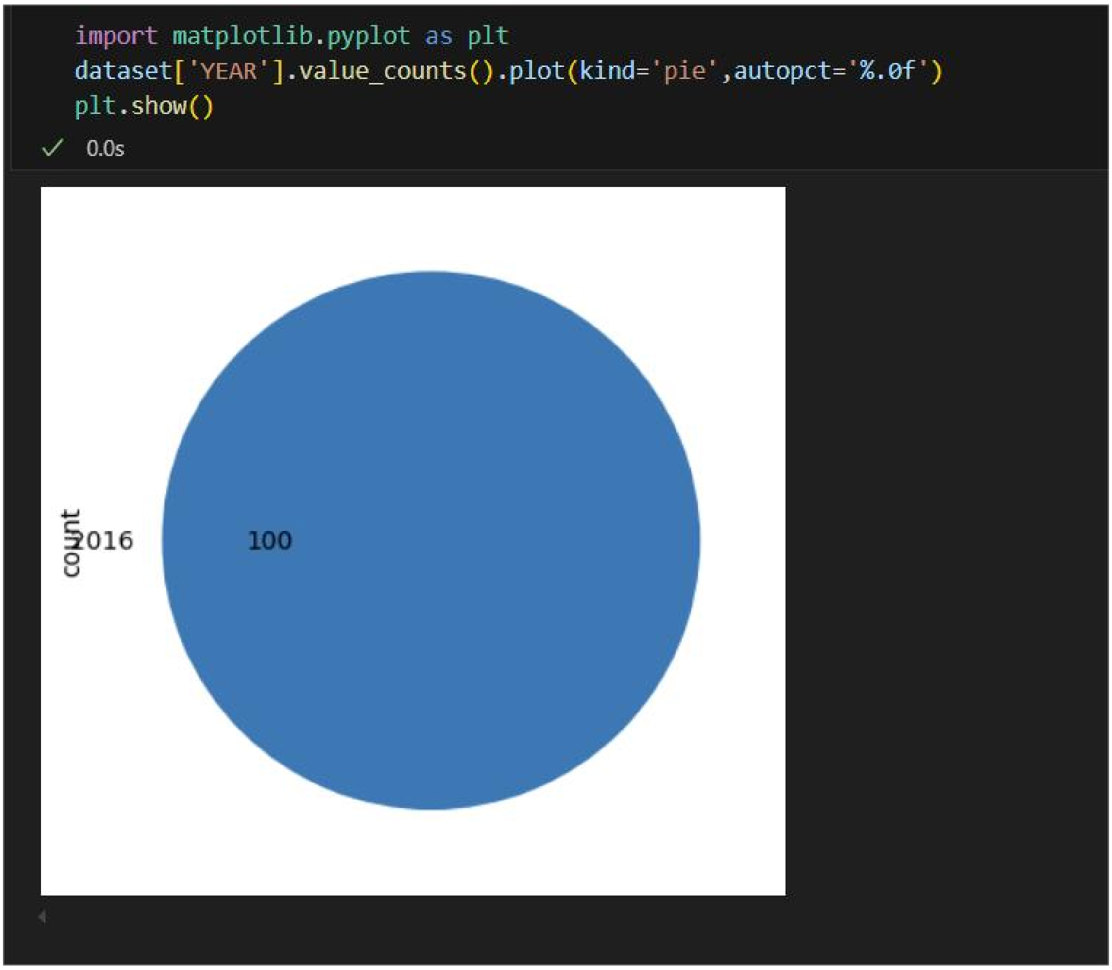
Check unique values in dataset:



```
dataset['DEST'].unique()
✓ 0.0s
array(['SEA', 'MSP', 'DTW', 'ATL', 'JFK'], dtype=object)
```

Data visualization:

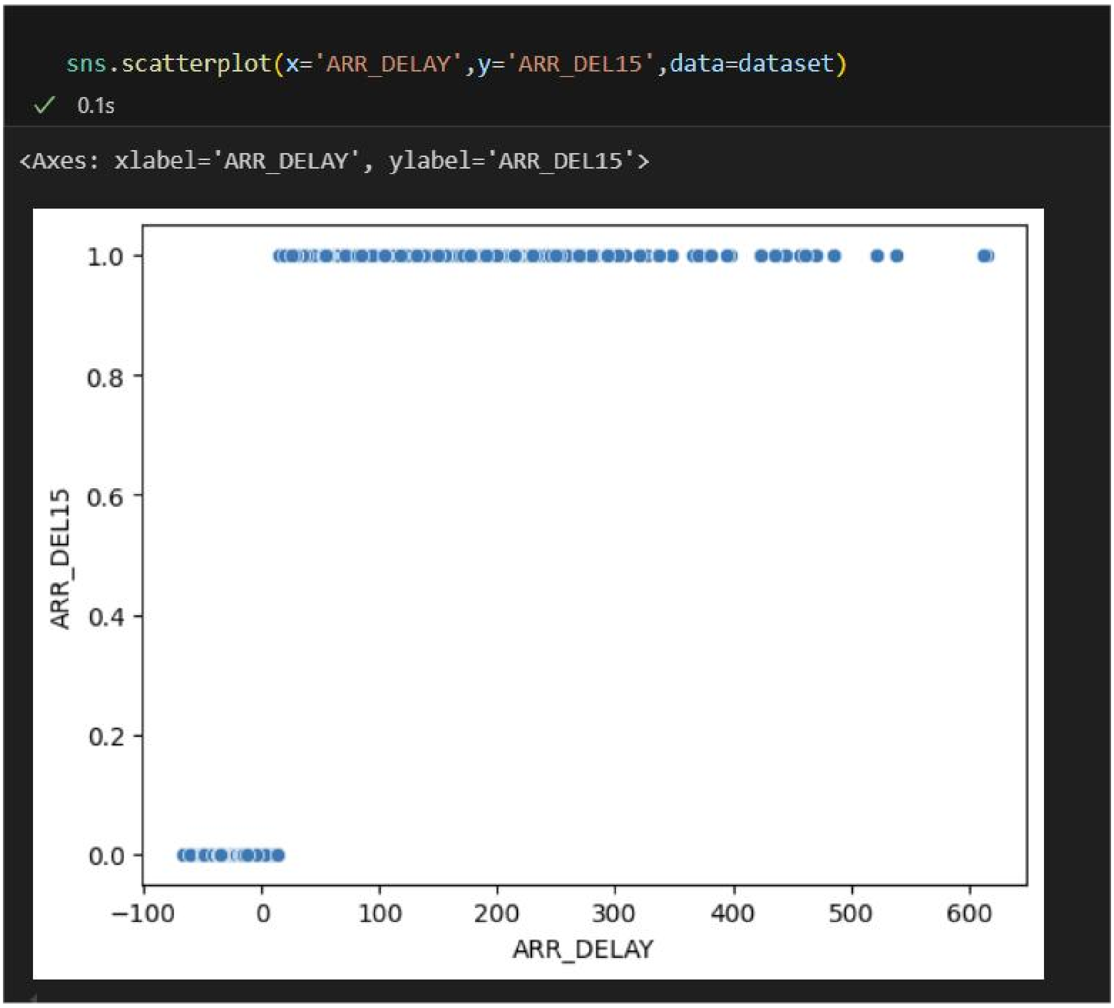
Univariate:



```
import matplotlib.pyplot as plt
dataset['YEAR'].value_counts().plot(kind='pie',autopct='%.0f')
plt.show()
✓ 0.0s
```

Bivariate:

Scatterplot:

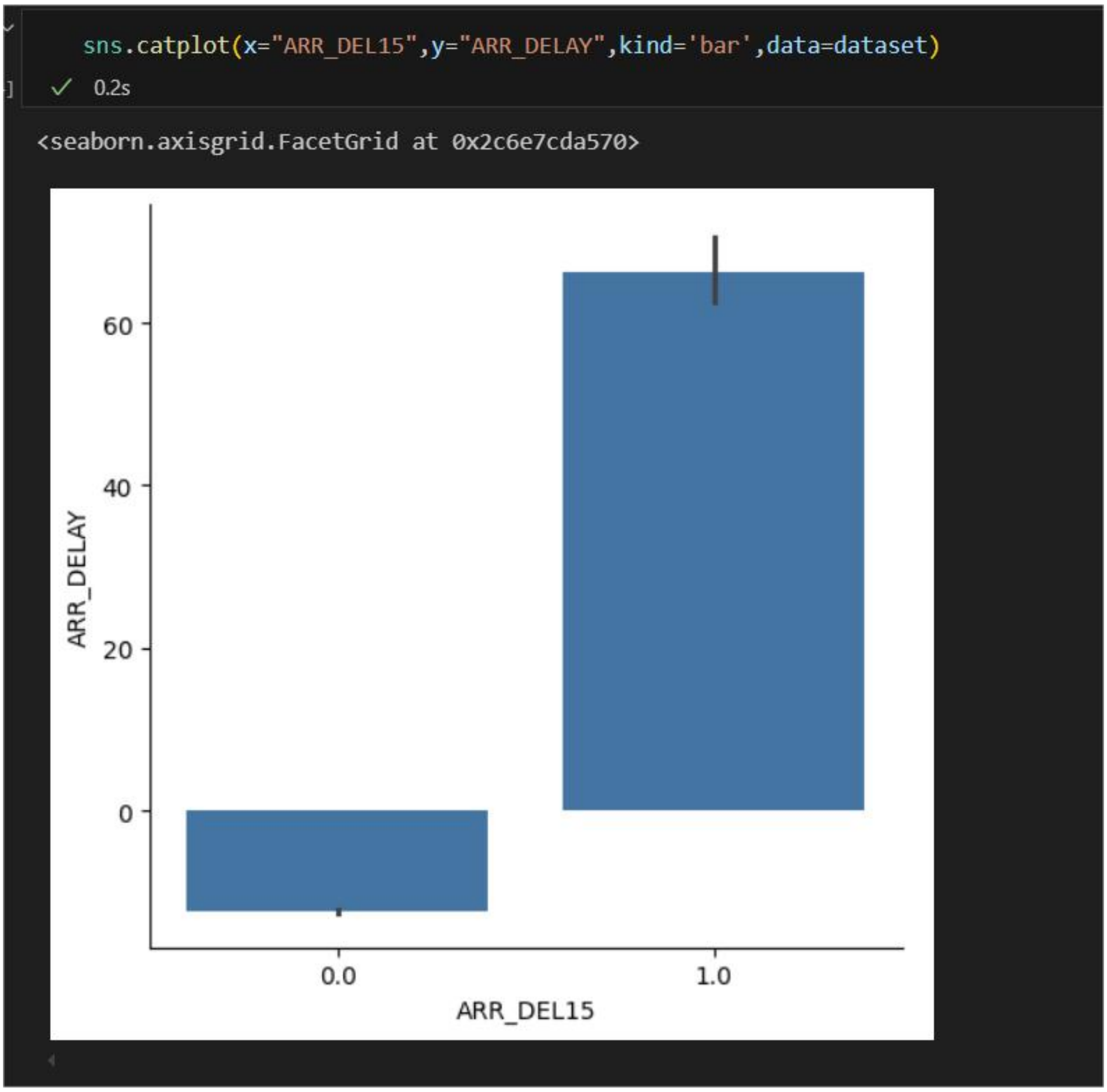


```
sns.scatterplot(x='ARR_DELAY',y='ARR_DEL15',data=dataset)
```

✓ 0.1s

```
<Axes: xlabel='ARR_DELAY', ylabel='ARR_DEL15'>
```

Catplot:

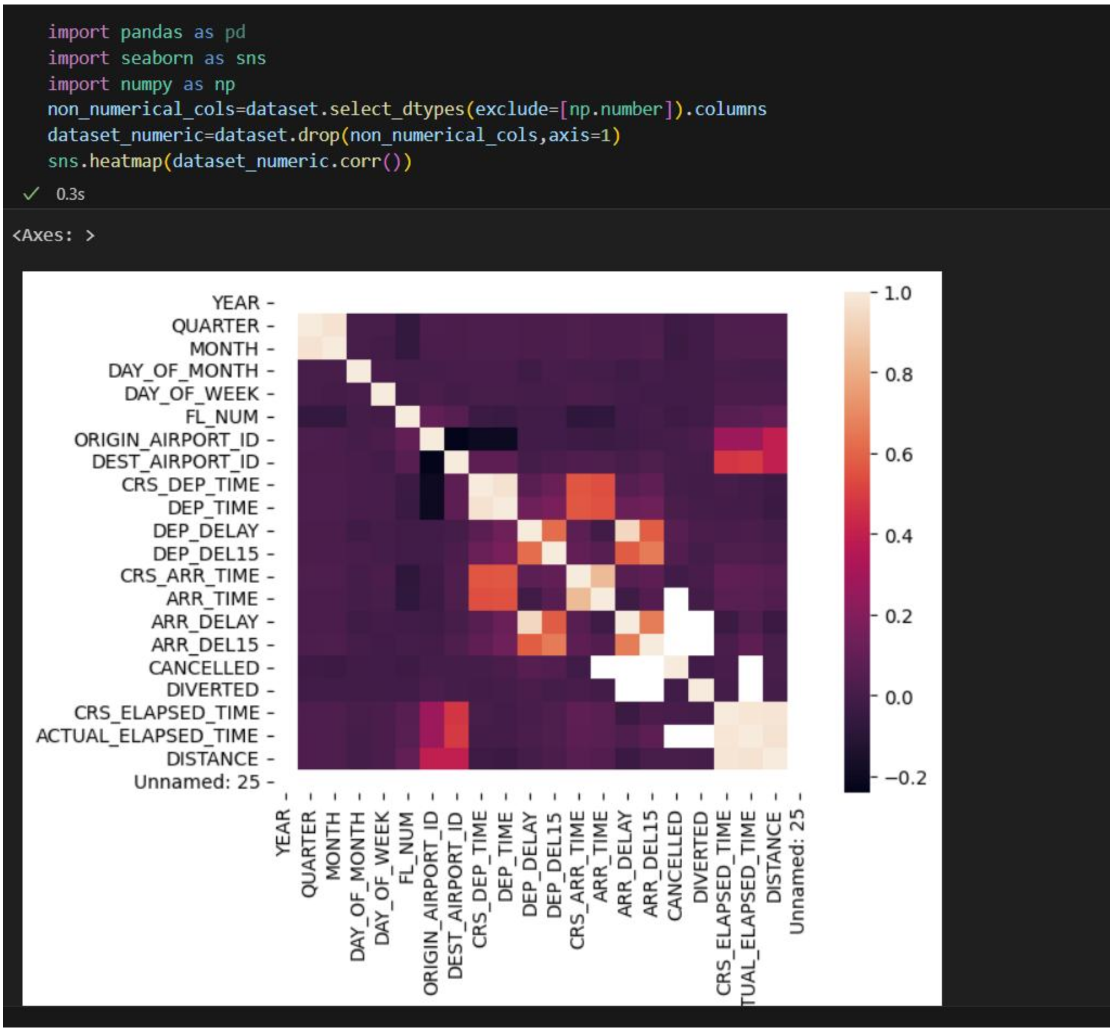


```
sns.catplot(x="ARR_DEL15",y="ARR_DELAY",kind='bar',data=dataset)
```

✓ 0.2s

```
<seaborn.axisgrid.FacetGrid at 0x2c6e7cda570>
```

heatmap:

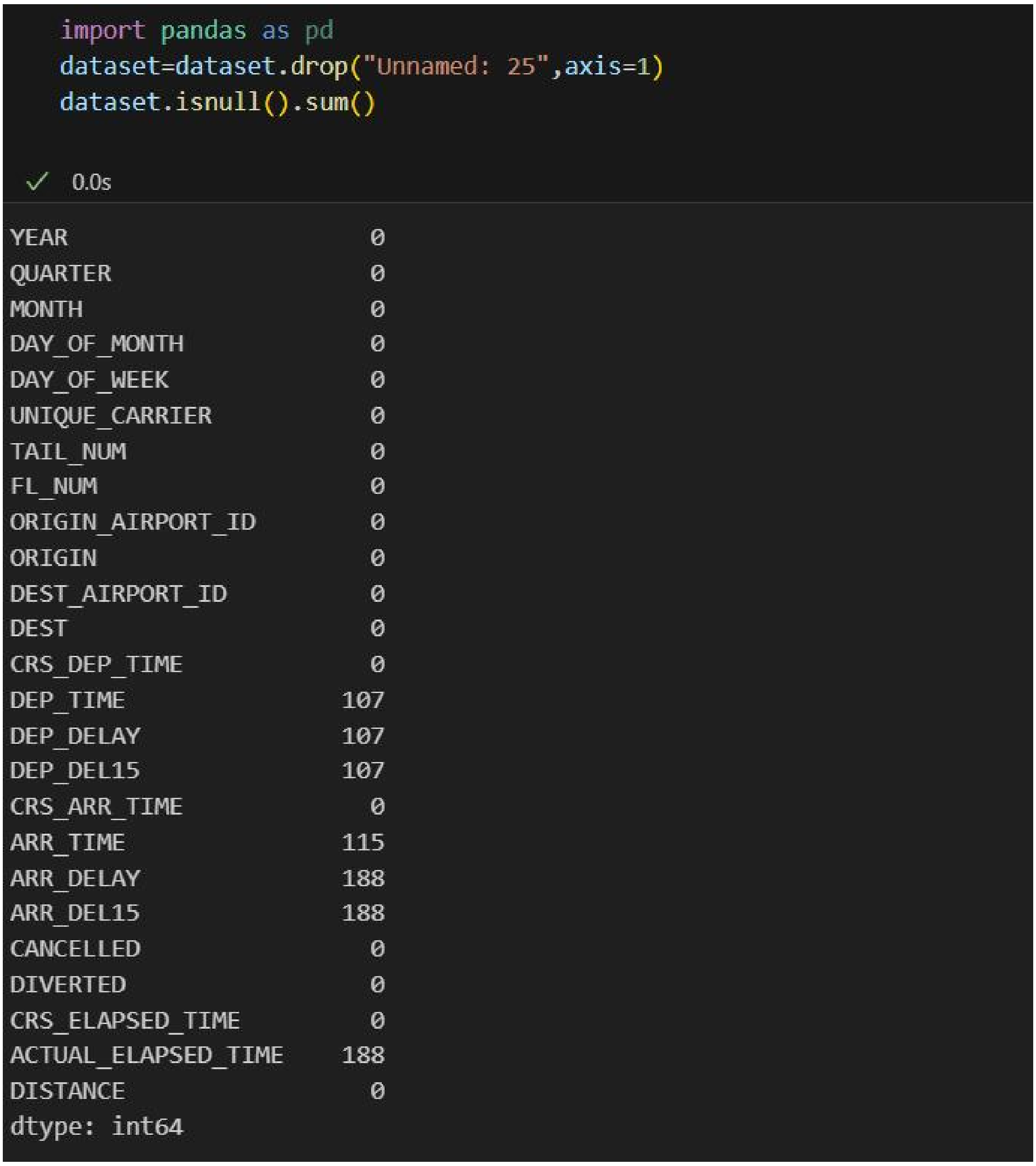


```
import pandas as pd
import seaborn as sns
import numpy as np
non_numerical_cols=dataset.select_dtypes(exclude=[np.number]).columns
dataset_numeric=dataset.drop(non_numerical_cols,axis=1)
sns.heatmap(dataset_numeric.corr())
```

✓ 0.3s

```
<Axes: >
```

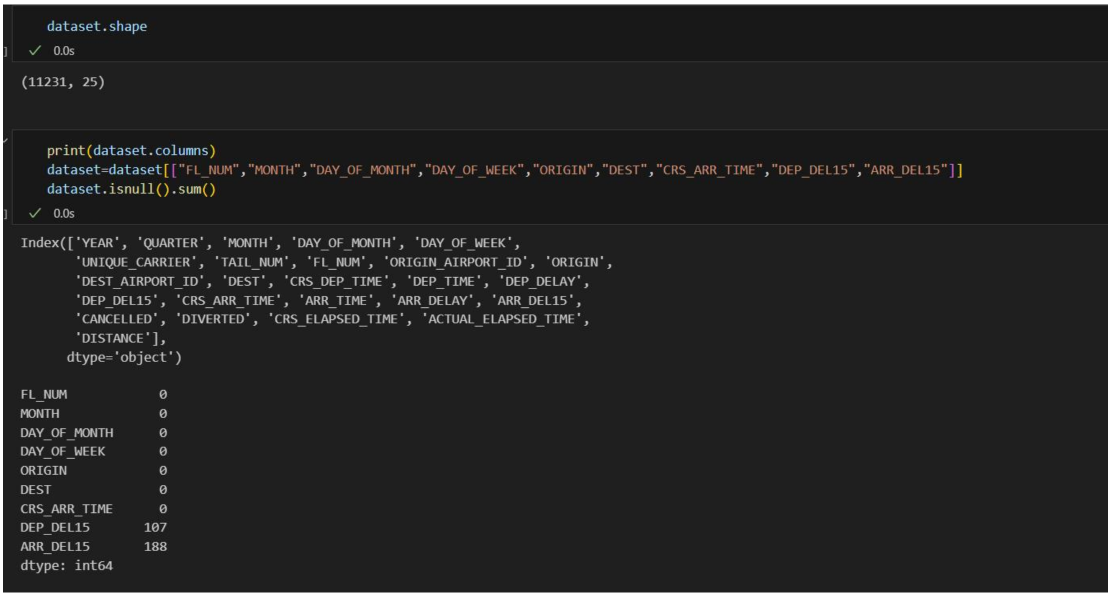
Droping unnecessary columns:



```
import pandas as pd
dataset=dataset.drop("Unnamed: 25",axis=1)
dataset.isnull().sum()
```

✓ 0.0s

```
YEAR                     0
QUARTER                  0
MONTH                    0
DAY_OF_MONTH             0
DAY_OF_WEEK              0
UNIQUE_CARRIER           0
TAIL_NUM                 0
FL_NUM                   0
ORIGIN_AIRPORT_ID        0
ORIGIN                   0
DEST_AIRPORT_ID          0
DEST                     0
CRS_DEP_TIME             0
DEP_TIME               107
DEP_DELAY              107
DEP_DEL15              107
CRS_ARR_TIME             0
ARR_TIME               115
ARR_DELAY              188
ARR_DEL15              188
CANCELLED                0
DIVERTED                 0
CRS_ELAPSED_TIME         0
ACTUAL_ELAPSED_TIME    188
DISTANCE                 0
dtype: int64
```



```
dataset.shape
```

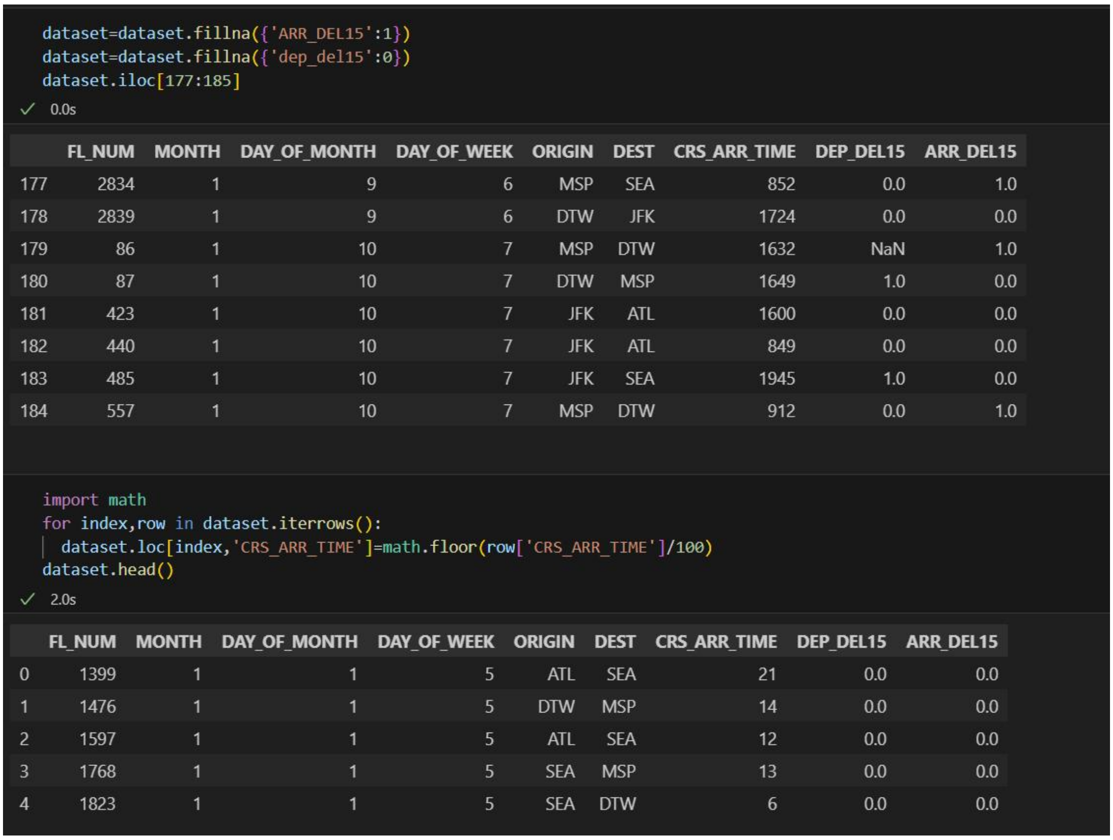
✓ 0.0s

```
(11231, 25)
```

```
print(dataset.columns)
dataset=dataset[["FL_NUM","MONTH","DAY_OF_MONTH","DAY_OF_WEEK","ORIGIN","DEST","CRS_ARR_TIME","DEP_DEL15","ARR_DEL15"]]
dataset.isnull().sum()
```

✓ 0.0s

```
Index(['YEAR', 'QUARTER', 'MONTH', 'DAY_OF_MONTH', 'DAY_OF_WEEK',
       'UNIQUE_CARRIER', 'TAIL_NUM', 'FL_NUM', 'ORIGIN_AIRPORT_ID', 'ORIGIN',
       'DEST_AIRPORT_ID', 'DEST', 'CRS_DEP_TIME', 'DEP_TIME', 'DEP_DELAY',
       'DEP_DEL15', 'CRS_ARR_TIME', 'ARR_TIME', 'ARR_DELAY', 'ARR_DEL15',
       'CANCELLED', 'DIVERTED', 'CRS_ELAPSED_TIME', 'ACTUAL_ELAPSED_TIME',
       'DISTANCE'],
      dtype='object')

FL_NUM            0
MONTH             0
DAY_OF_MONTH      0
DAY_OF_WEEK       0
ORIGIN            0
DEST              0
CRS_ARR_TIME      0
DEP_DEL15       107
ARR_DEL15       188
dtype: int64
```



```
dataset=dataset.fillna({'ARR_DEL15':1})
dataset=dataset.fillna({'dep_del15':0})
dataset.iloc[177:185]
```

✓ 0.0s

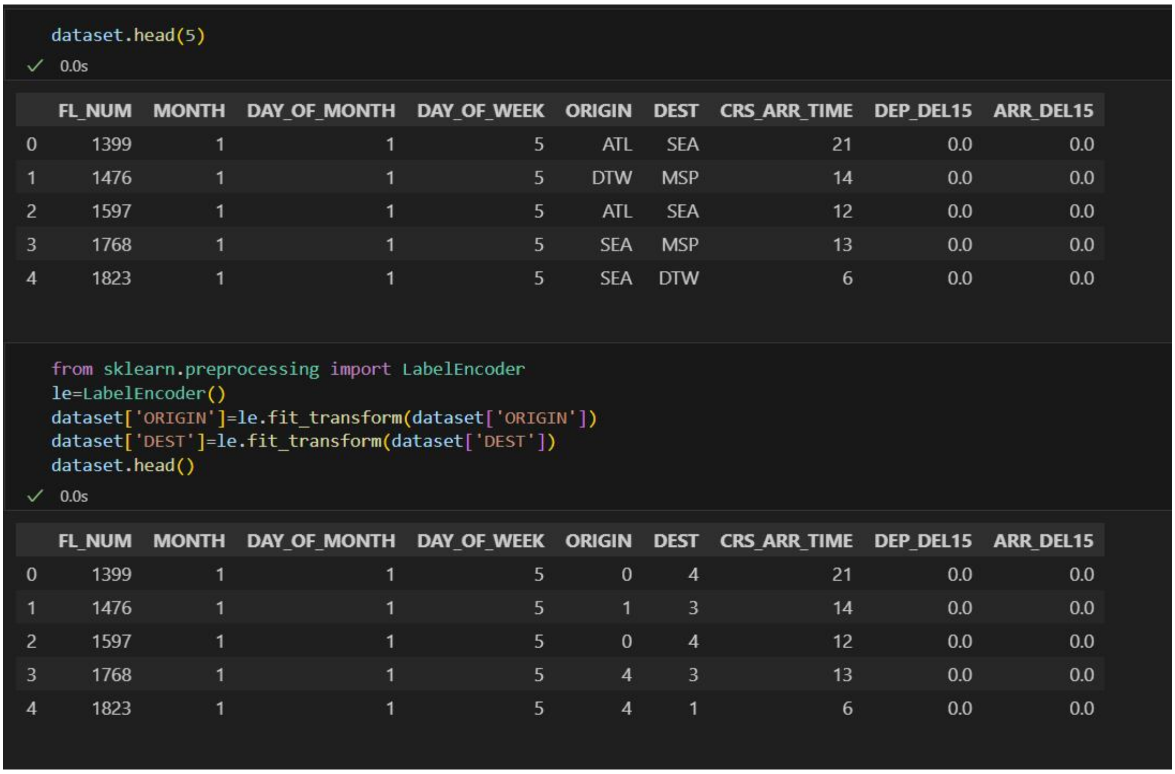
	FL_NUM	MONTH	DAY_OF_MONTH	DAY_OF_WEEK	ORIGIN	DEST	CRS_ARR_TIME	DEP_DEL15	ARR_DEL15
177	2834	1	9	6	MSP	SEA	852	0.0	1.0
178	2839	1	9	6	DTW	JFK	1724	0.0	0.0
179	86	1	10	7	MSP	DTW	1632	NaN	1.0
180	87	1	10	7	DTW	MSP	1649	1.0	0.0
181	423	1	10	7	JFK	ATL	1600	0.0	0.0
182	440	1	10	7	JFK	ATL	849	0.0	0.0
183	485	1	10	7	JFK	SEA	1945	1.0	0.0
184	557	1	10	7	MSP	DTW	912	0.0	1.0

```
import math
for index,row in dataset.iterrows():
    dataset.loc[index,'CRS_ARR_TIME']=math.floor(row['CRS_ARR_TIME']/100)
dataset.head()
```

✓ 2.0s

	FL_NUM	MONTH	DAY_OF_MONTH	DAY_OF_WEEK	ORIGIN	DEST	CRS_ARR_TIME	DEP_DEL15	ARR_DEL15
0	1399	1	1	5	ATL	SEA	21	0.0	0.0
1	1476	1	1	5	DTW	MSP	14	0.0	0.0
2	1597	1	1	5	ATL	SEA	12	0.0	0.0
3	1768	1	1	5	SEA	MSP	13	0.0	0.0
4	1823	1	1	5	SEA	DTW	6	0.0	0.0

Label encoding and one hot encoding:



```
dataset.head(5)
```

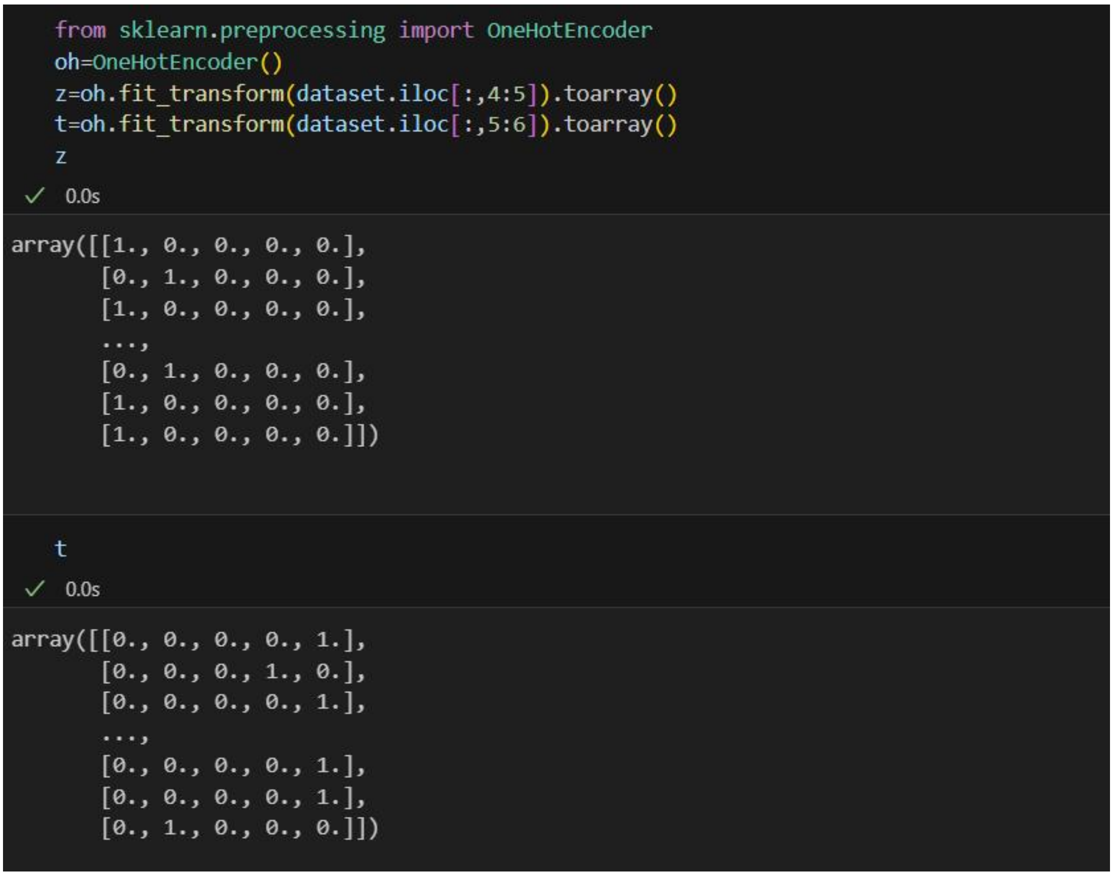
✓ 0.0s

```
   FL_NUM  MONTH  DAY_OF_MONTH  DAY_OF_WEEK  ORIGIN  DEST  CRS_ARR_TIME  DEP_DEL15  ARR_DEL15
0    1399      1             1            5     ATL   SEA            21        0.0        0.0
1    1476      1             1            5     DTW   MSP            14        0.0        0.0
2    1597      1             1            5     ATL   SEA            12        0.0        0.0
3    1768      1             1            5     SEA   MSP            13        0.0        0.0
4    1823      1             1            5     SEA   DTW             6        0.0        0.0
```

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
dataset['ORIGIN']=le.fit_transform(dataset['ORIGIN'])
dataset['DEST']=le.fit_transform(dataset['DEST'])
dataset.head()
```

✓ 0.0s

```
   FL_NUM  MONTH  DAY_OF_MONTH  DAY_OF_WEEK  ORIGIN  DEST  CRS_ARR_TIME  DEP_DEL15  ARR_DEL15
0    1399      1             1            5       0     4            21        0.0        0.0
1    1476      1             1            5       1     3            14        0.0        0.0
2    1597      1             1            5       0     4            12        0.0        0.0
3    1768      1             1            5       4     3            13        0.0        0.0
4    1823      1             1            5       4     1             6        0.0        0.0
```



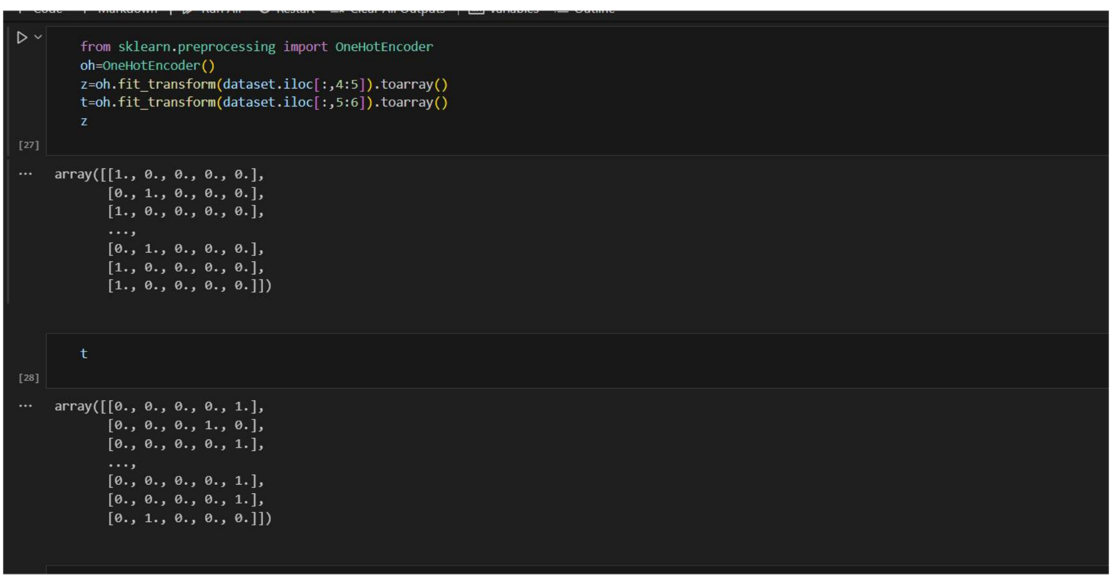
```
from sklearn.preprocessing import OneHotEncoder
oh=OneHotEncoder()
z=oh.fit_transform(dataset.iloc[:,4:5]).toarray()
t=oh.fit_transform(dataset.iloc[:,5:6]).toarray()
z
```

```
array([[1., 0., 0., 0., 0.],
       [0., 1., 0., 0., 0.],
       [1., 0., 0., 0., 0.],
       ...,
       [0., 1., 0., 0., 0.],
       [1., 0., 0., 0., 0.],
       [1., 0., 0., 0., 0.]])
```

```
t
```

```
array([[0., 0., 0., 0., 1.],
       [0., 0., 0., 1., 0.],
       [0., 0., 0., 0., 1.],
       ...,
       [0., 0., 0., 0., 1.],
       [0., 0., 0., 0., 1.],
       [0., 1., 0., 0., 0.]])
```

Creating the independent and dependent variables:

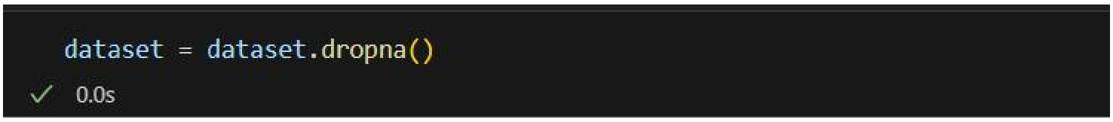


```
from sklearn.preprocessing import OneHotEncoder
oh=OneHotEncoder()
z=oh.fit_transform(dataset.iloc[:,4:5]).toarray()
t=oh.fit_transform(dataset.iloc[:,5:6]).toarray()
z
```

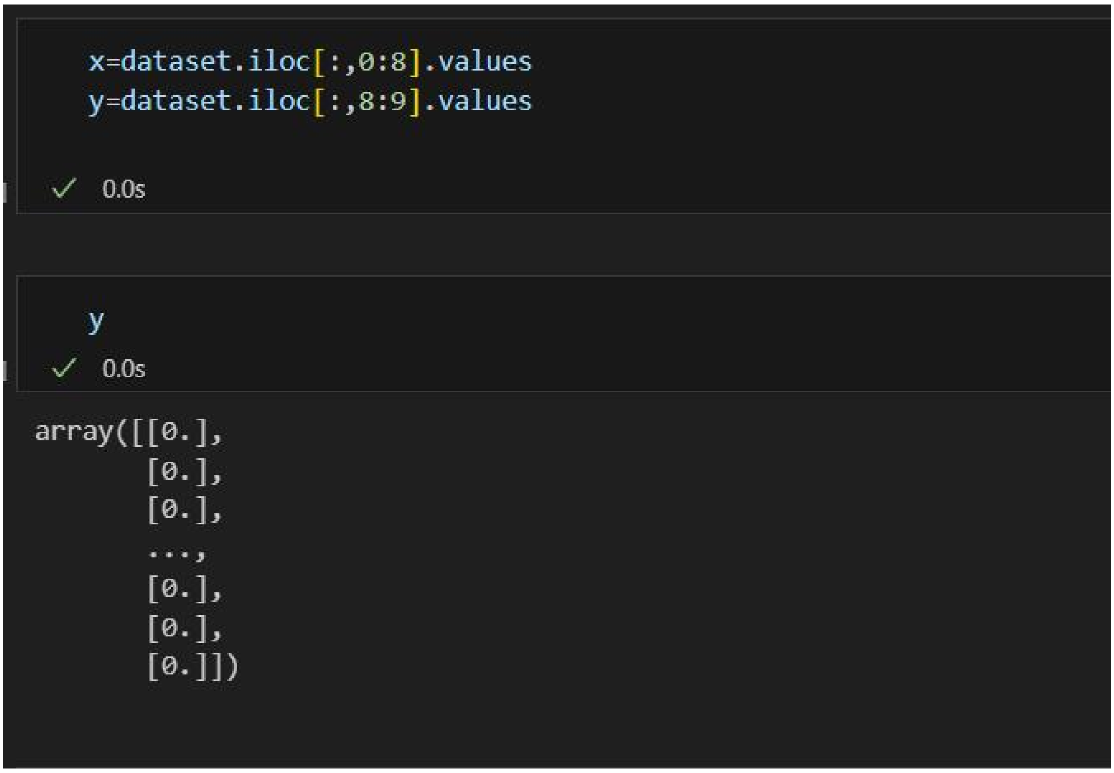
```
array([[1., 0., 0., 0., 0.],
       [0., 1., 0., 0., 0.],
       [1., 0., 0., 0., 0.],
       ...,
       [0., 1., 0., 0., 0.],
       [1., 0., 0., 0., 0.],
       [1., 0., 0., 0., 0.]])
```

```
t
```

```
array([[0., 0., 0., 0., 1.],
       [0., 0., 0., 1., 0.],
       [0., 0., 0., 0., 1.],
       ...,
       [0., 0., 0., 0., 1.],
       [0., 0., 0., 0., 1.],
       [0., 1., 0., 0., 0.]])
```



```
dataset = dataset.dropna()
```



```
x=dataset.iloc[:,0:8].values
y=dataset.iloc[:,8:9].values
```

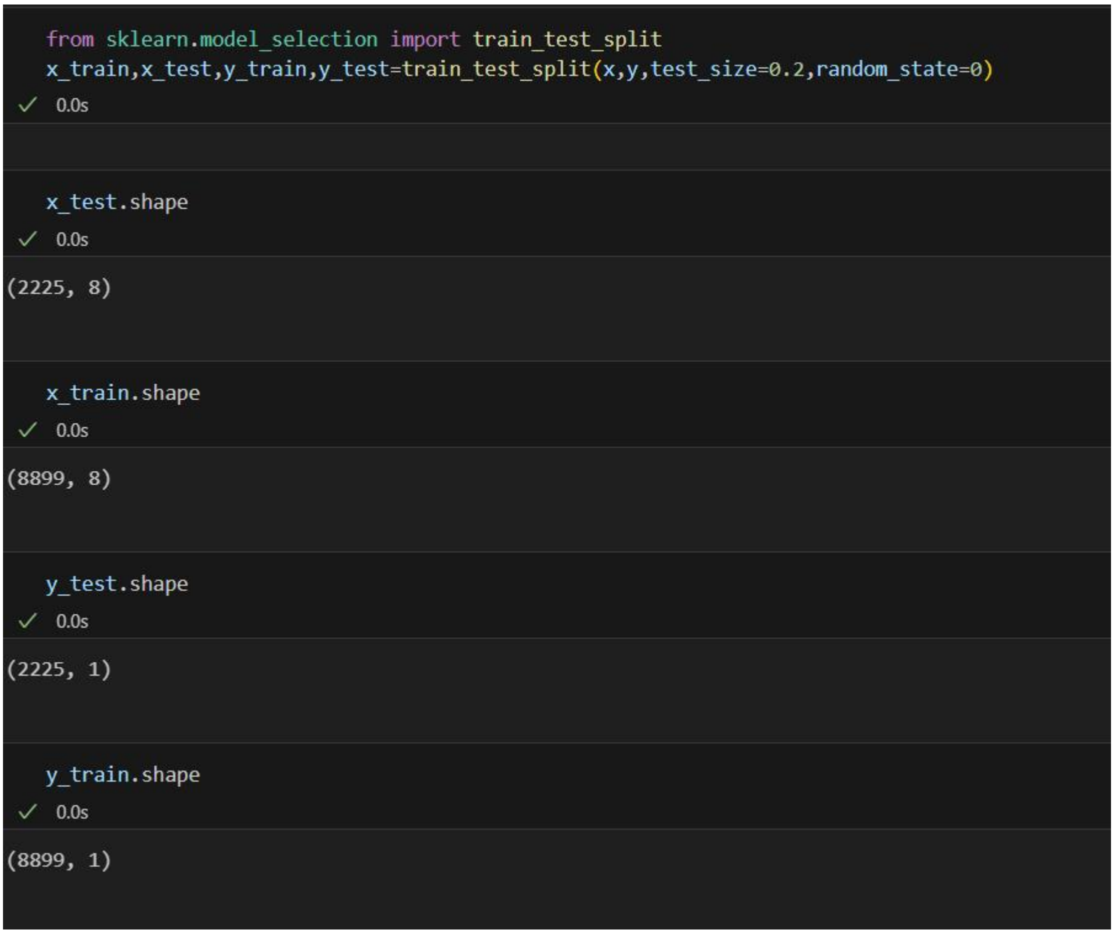
✓ 0.0s

```
y
```

✓ 0.0s

```
array([[0.],
       [0.],
       [0.],
       ...,
       [0.],
       [0.],
       [0.]])
```

Splitting dataset into train and test:



```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

✓ 0.0s

```
x_test.shape
```

✓ 0.0s

```
(2225, 8)
```

```
x_train.shape
```

✓ 0.0s

```
(8899, 8)
```

```
y_test.shape
```

✓ 0.0s

```
(2225, 1)
```

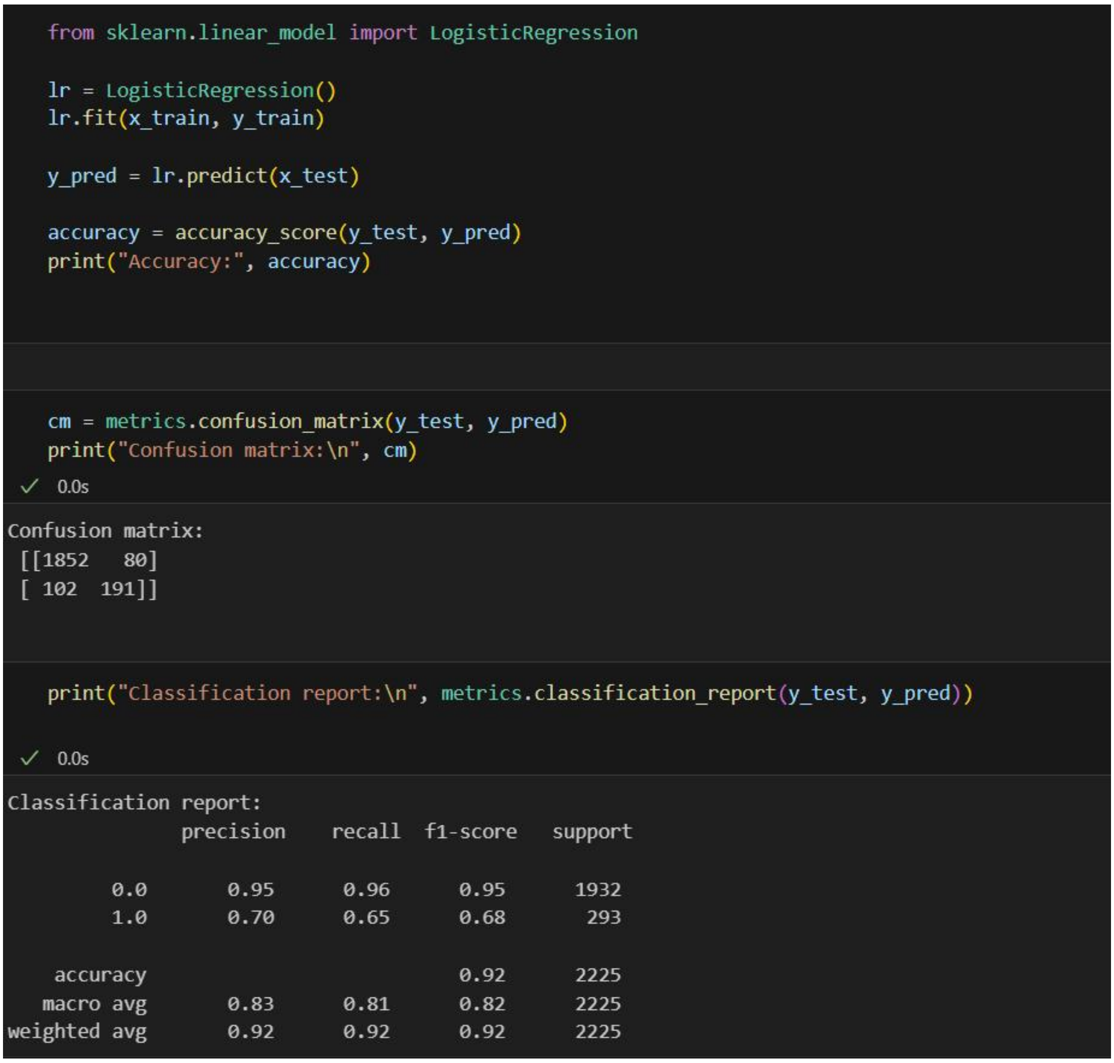
```
y_train.shape
```

✓ 0.0s

```
(8899, 1)
```

Model building:

Logistic regression:



```
from sklearn.linear_model import LogisticRegression

lr = LogisticRegression()
lr.fit(x_train, y_train)

y_pred = lr.predict(x_test)

accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```
cm = metrics.confusion_matrix(y_test, y_pred)
print("Confusion matrix:\n", cm)
```

✓ 0.0s

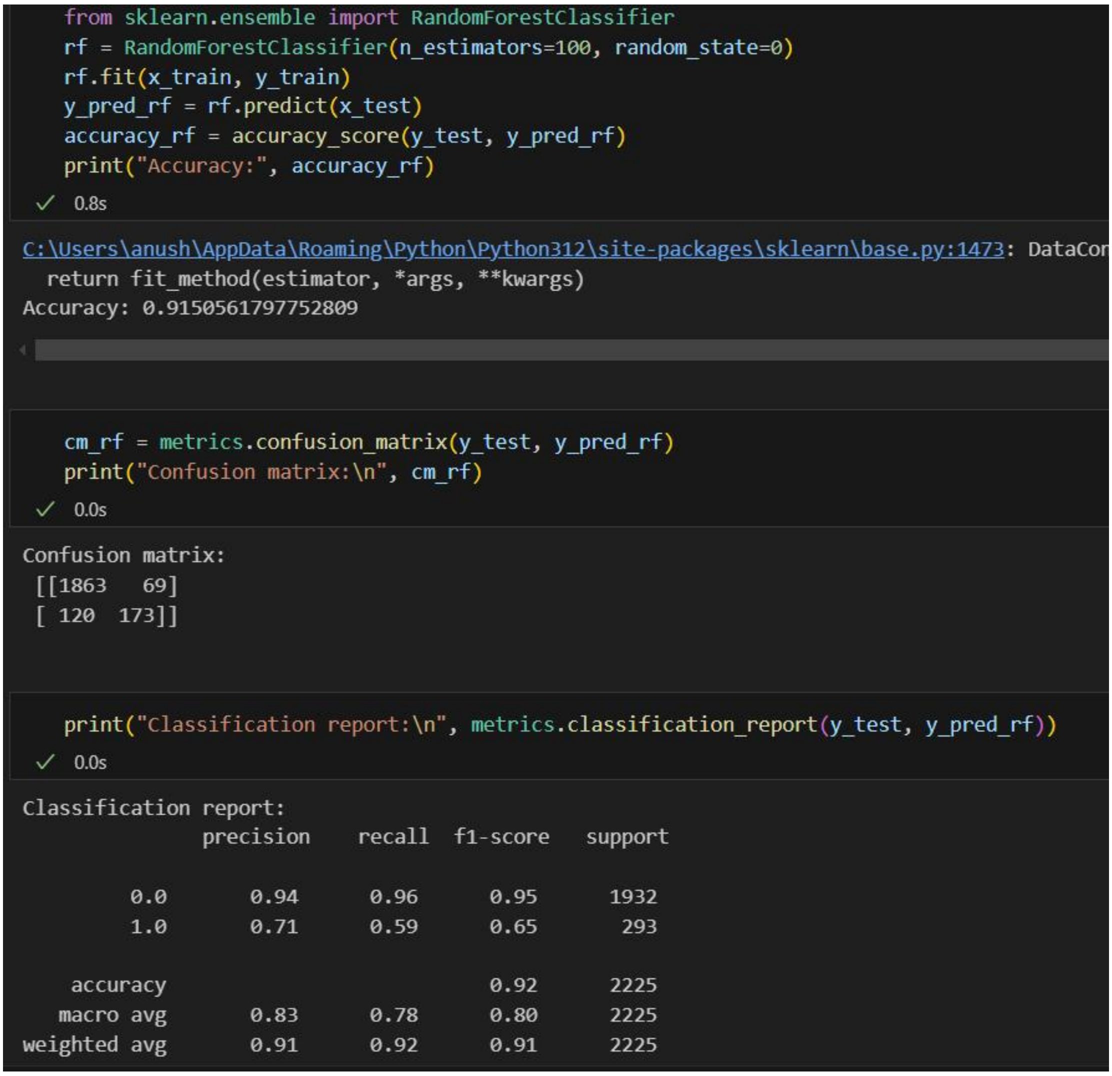
```
Confusion matrix:
 [[1852   80]
 [ 102  191]]
```

```
print("Classification report:\n", metrics.classification_report(y_test, y_pred))
```

✓ 0.0s

```
Classification report:
               precision    recall  f1-score   support

         0.0       0.95      0.96      0.95      1932
         1.0       0.70      0.65      0.68       293

    accuracy                           0.92      2225
   macro avg       0.83      0.81      0.82      2225
weighted avg       0.92      0.92      0.92      2225
```

Randomforest classifier:



```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(n_estimators=100, random_state=0)
rf.fit(x_train, y_train)
y_pred_rf = rf.predict(x_test)
accuracy_rf = accuracy_score(y_test, y_pred_rf)
print("Accuracy:", accuracy_rf)
```

✓ 0.8s

```
C:\Users\anush\AppData\Roaming\Python\Python312\site-packages\sklearn\base.py:1473: DataCon
  return fit_method(estimator, *args, **kwargs)
Accuracy: 0.9150561797752809
```

```
cm_rf = metrics.confusion_matrix(y_test, y_pred_rf)
print("Confusion matrix:\n", cm_rf)
```

✓ 0.0s

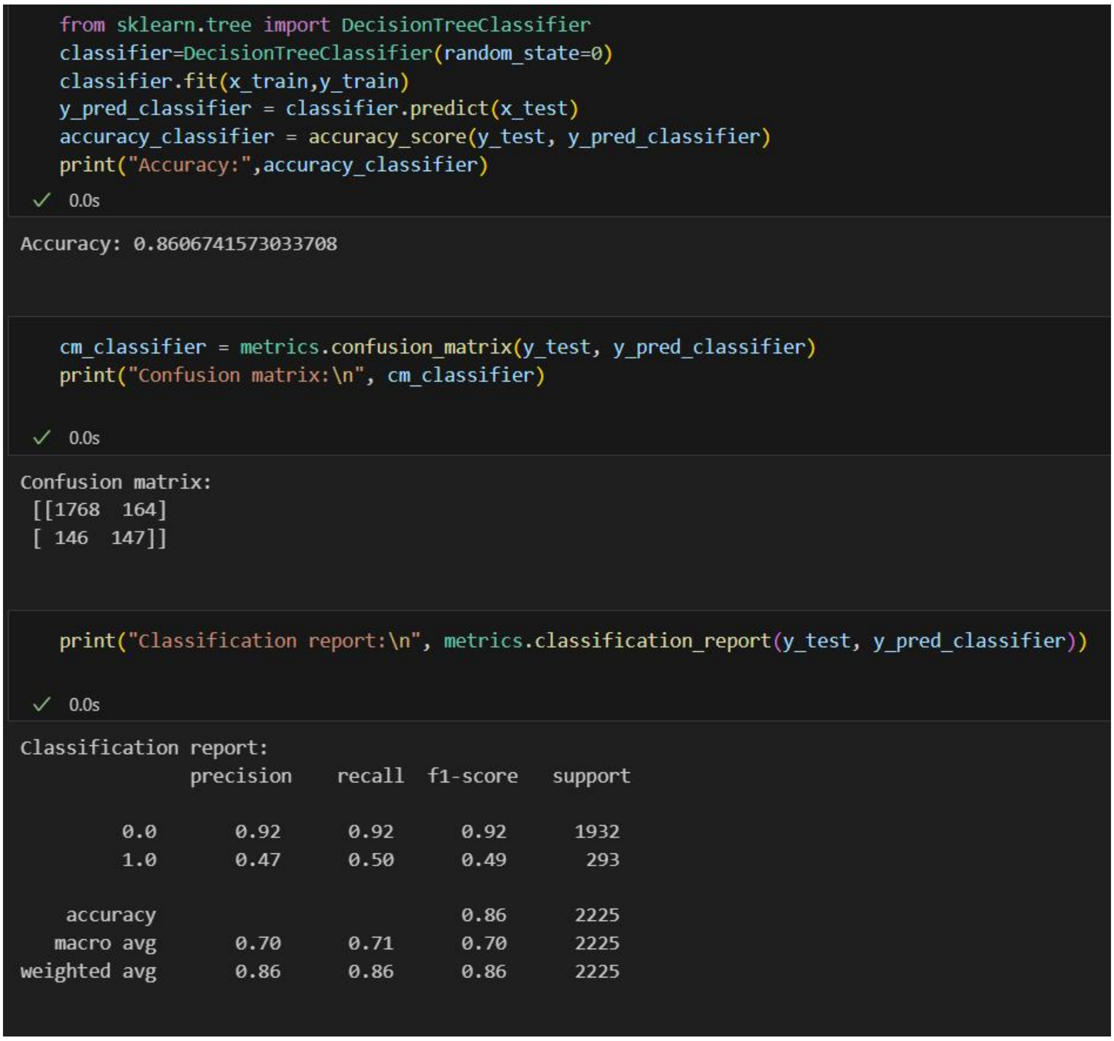
```
Confusion matrix:
 [[1863   69]
 [ 120  173]]
```

```
print("Classification report:\n", metrics.classification_report(y_test, y_pred_rf))
```

✓ 0.0s

```
Classification report:
               precision    recall  f1-score   support

         0.0       0.94      0.96      0.95      1932
         1.0       0.71      0.59      0.65       293

    accuracy                           0.92      2225
   macro avg       0.83      0.78      0.80      2225
weighted avg       0.91      0.92      0.91      2225
```

DecisionTree classifier:



```
from sklearn.tree import DecisionTreeClassifier
classifier=DecisionTreeClassifier(random_state=0)
classifier.fit(x_train,y_train)
y_pred_classifier = classifier.predict(x_test)
accuracy_classifier = accuracy_score(y_test, y_pred_classifier)
print("Accuracy:",accuracy_classifier)
```

```
Accuracy: 0.8606741573033708
```

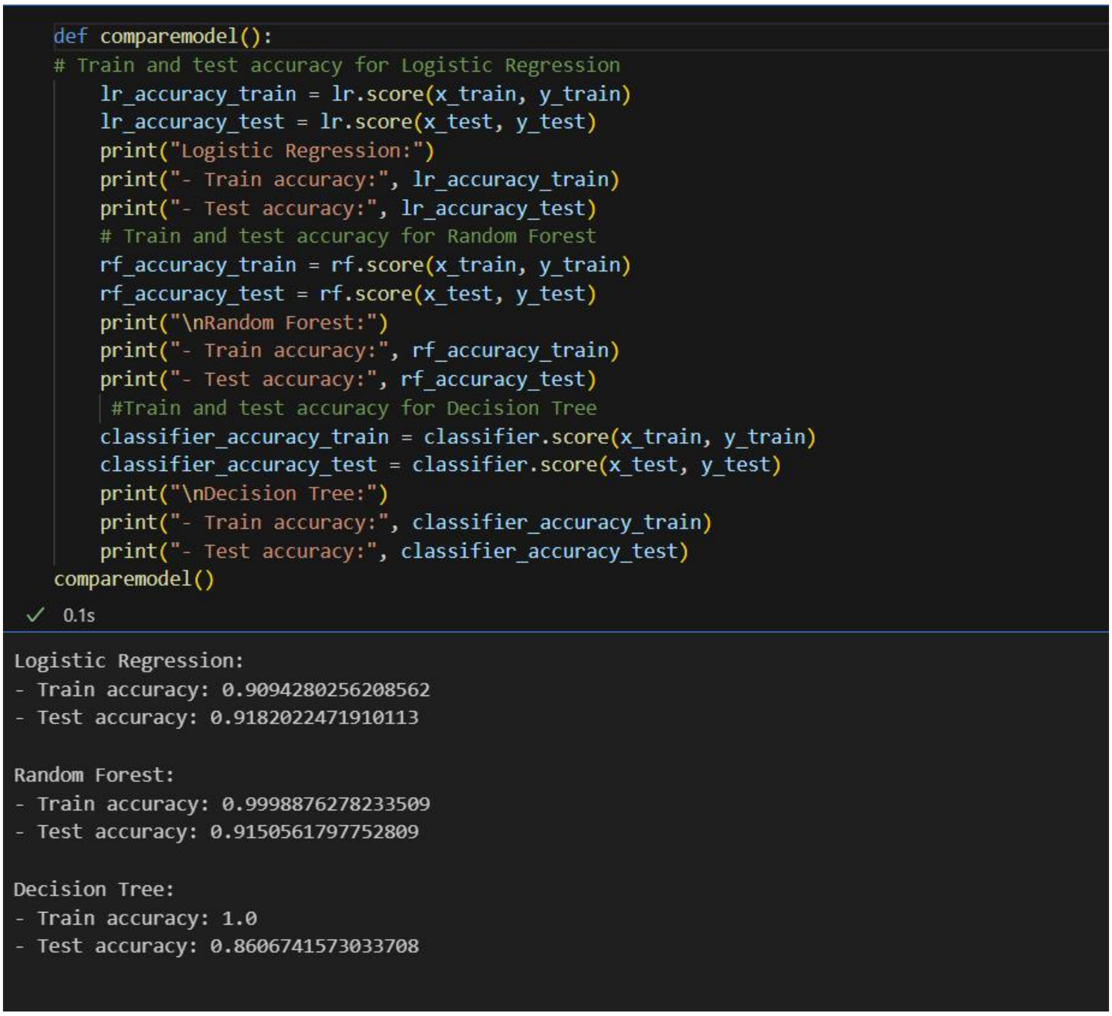
```
cm_classifier = metrics.confusion_matrix(y_test, y_pred_classifier)
print("Confusion matrix:\n", cm_classifier)
```

```
Confusion matrix:
 [[1768  164]
 [ 146  147]]
```

```
print("Classification report:\n", metrics.classification_report(y_test, y_pred_classifier))
```

```
Classification report:
               precision    recall  f1-score   support

         0.0       0.92      0.92      0.92      1932
         1.0       0.47      0.50      0.49       293

    accuracy                           0.86      2225
   macro avg       0.70      0.71      0.70      2225
weighted avg       0.86      0.86      0.86      2225
```

Comparing the models:

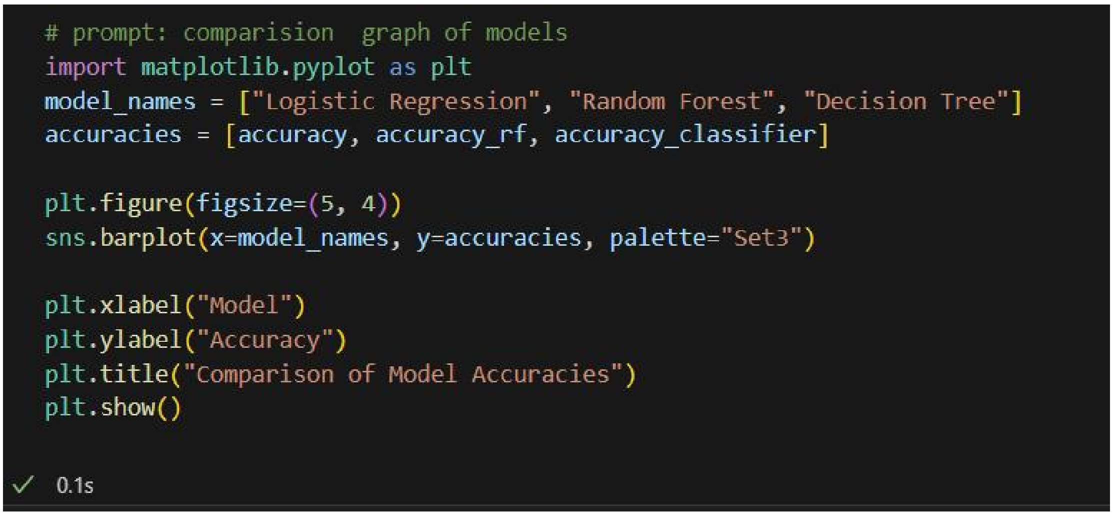


```
def comparemodel():
# Train and test accuracy for Logistic Regression
    lr_accuracy_train = lr.score(x_train, y_train)
    lr_accuracy_test = lr.score(x_test, y_test)
    print("Logistic Regression:")
    print("- Train accuracy:", lr_accuracy_train)
    print("- Test accuracy:", lr_accuracy_test)
    # Train and test accuracy for Random Forest
    rf_accuracy_train = rf.score(x_train, y_train)
    rf_accuracy_test = rf.score(x_test, y_test)
    print("\nRandom Forest:")
    print("- Train accuracy:", rf_accuracy_train)
    print("- Test accuracy:", rf_accuracy_test)
     #Train and test accuracy for Decision Tree
    classifier_accuracy_train = classifier.score(x_train, y_train)
    classifier_accuracy_test = classifier.score(x_test, y_test)
    print("\nDecision Tree:")
    print("- Train accuracy:", classifier_accuracy_train)
    print("- Test accuracy:", classifier_accuracy_test)
comparemodel()
```

✓ 0.1s

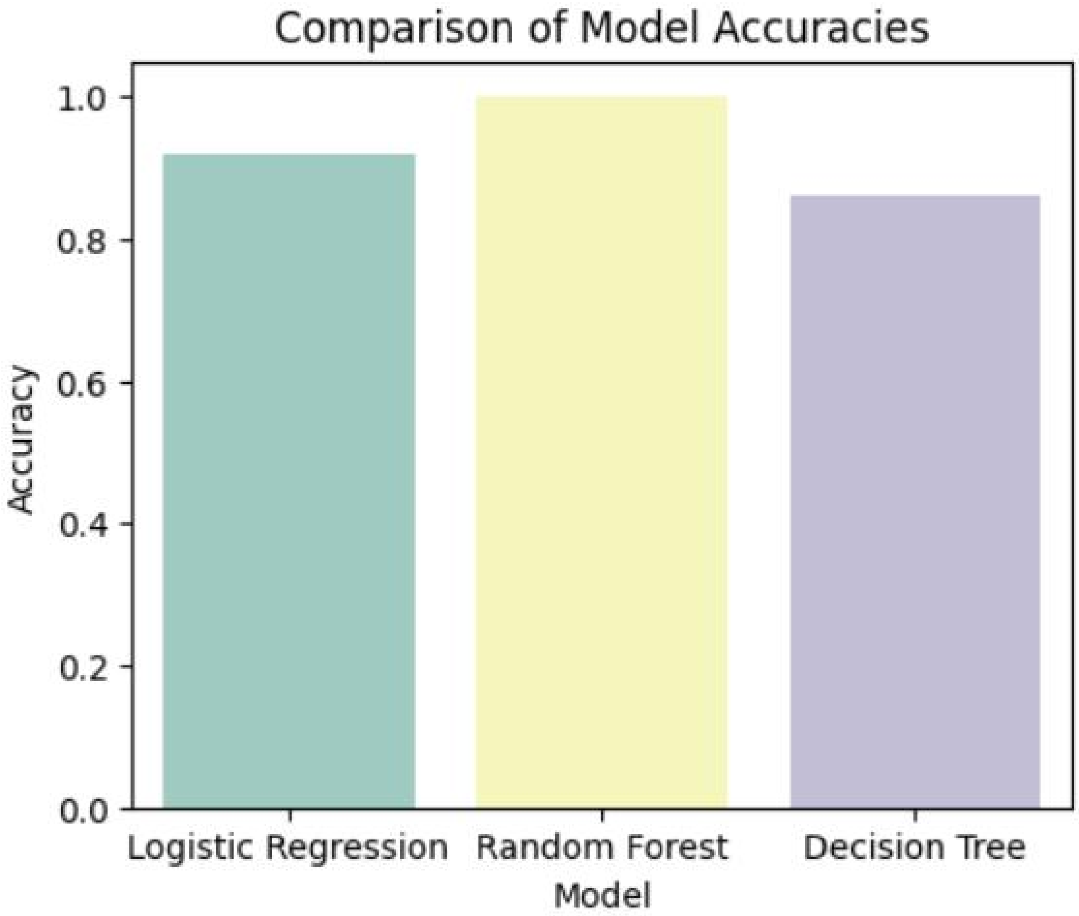
```
Logistic Regression:
- Train accuracy: 0.9094280256208562
- Test accuracy: 0.9182022471910113

Random Forest:
- Train accuracy: 0.9998876278233509
- Test accuracy: 0.9150561797752809

Decision Tree:
- Train accuracy: 1.0
- Test accuracy: 0.8606741573033708
```

Compare models using graph:

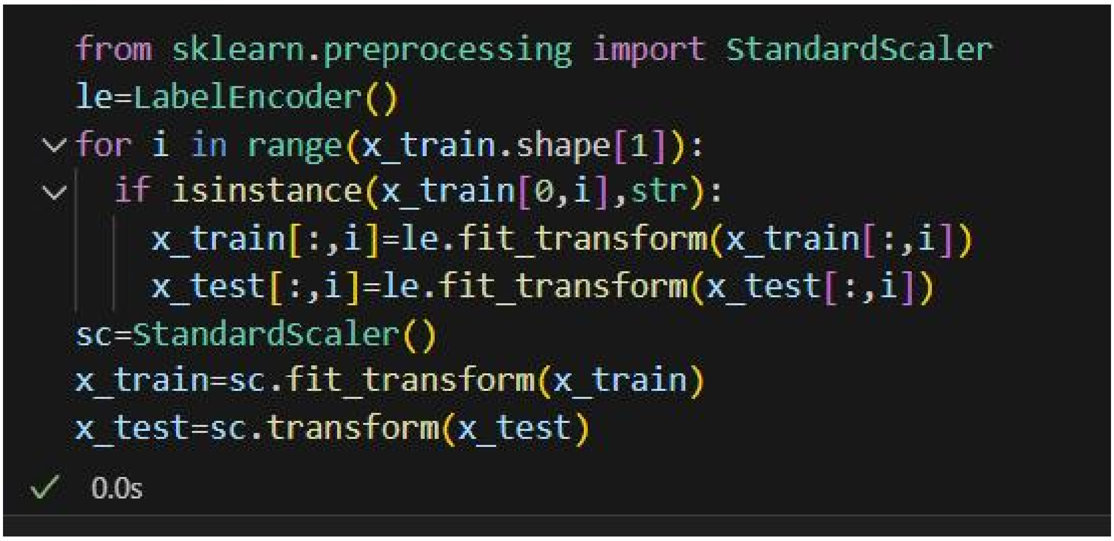


```
# prompt: comparision  graph of models
import matplotlib.pyplot as plt
model_names = ["Logistic Regression", "Random Forest", "Decision Tree"]
accuracies = [accuracy, accuracy_rf, accuracy_classifier]

plt.figure(figsize=(5, 4))
sns.barplot(x=model_names, y=accuracies, palette="Set3")

plt.xlabel("Model")
plt.ylabel("Accuracy")
plt.title("Comparison of Model Accuracies")
plt.show()
```

✓ 0.1s

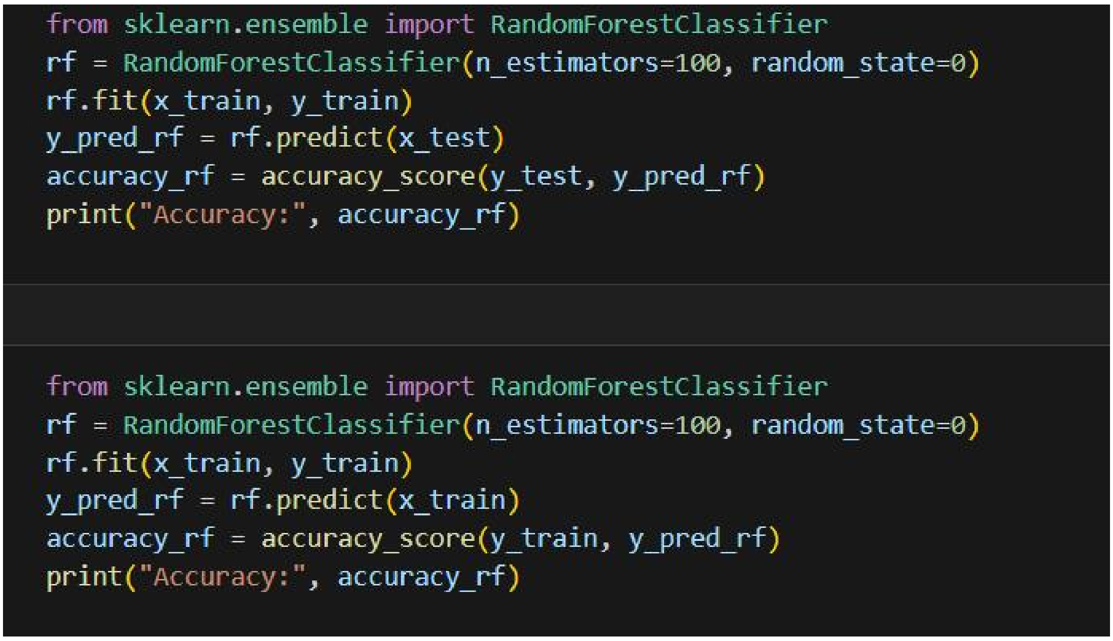


Evaluation of model:



```
from sklearn.preprocessing import StandardScaler
le=LabelEncoder()
for i in range(x_train.shape[1]):
    if isinstance(x_train[0,i],str):
        x_train[:,i]=le.fit_transform(x_train[:,i])
        x_test[:,i]=le.fit_transform(x_test[:,i])
sc=StandardScaler()
x_train=sc.fit_transform(x_train)
x_test=sc.transform(x_test)
```

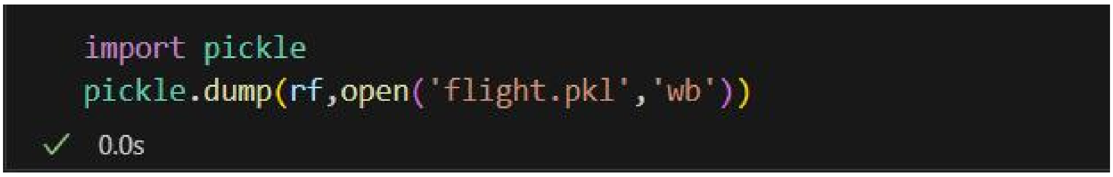
✓ 0.0s



```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(n_estimators=100, random_state=0)
rf.fit(x_train, y_train)
y_pred_rf = rf.predict(x_test)
accuracy_rf = accuracy_score(y_test, y_pred_rf)
print("Accuracy:", accuracy_rf)
```

```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(n_estimators=100, random_state=0)
rf.fit(x_train, y_train)
y_pred_rf = rf.predict(x_train)
accuracy_rf = accuracy_score(y_train, y_pred_rf)
print("Accuracy:", accuracy_rf)
```

Saving the model:



```
import pickle
pickle.dump(rf,open('flight.pkl','wb'))
```

✓ 0.0s