Title

Airline Price Prediction: A Data-Driven Approach for Predicting Airline Prices

Data Detectives – Cumulative Progress Report 2

Team Members	**Email**
Swetha Tanikonda	swethatanikonda@loyalistcollege.com
Mohamed Maaz Rehan	mohamedmaazrehan@loyalistcollege.com
Devendra Singh Shekhawat	devendrasinghshek@loyalistcollege.com
Fatemi Sadikbhai Lokhandwala	fatemisadikbhailo@loyalistcollege.com
Gaurav Singh Rawat	gauravsinghrawat@loyalistcollege.com
Gaurav Singh	gauravsingh3@loyalistcollege.com
Jankiba Viralsinh Zala	jankibaviralsinhz@loyalistcollege.com
Comfort Iroha Onuoha	comfortirohaonuoh@loyalistcollege.com
Isha Savaliya	ishasavaliya@loyalistcollege.com
Tirth Patel	tirthpatel@loyalistcollege.com
Urjeet Parmar	urjeetparmar@loyalistcollege.com

Presentation link:
In-class_presentation2

Contents

1.Introduction

Project Overview

This report details the development of airline price prediction project, a data-driven solution to help travelers find the best flight prices. Our project aimed to leverage machine learning techniques and data visualization to predict airfare based on various factors. The primary objective was to build a user-friendly and reliable system capable of providing accurate flight price estimations.

Project Objectives

The key objectives of the project are:

- To gather flight price data from various sources such as Skyscanner, Kayak.
- To clean and preprocess the data for accurate and effective analysis.
- To engineer relevant features that will enhance the performance of predictive models.
- To build and evaluate predictive models and deploy them for user accessibility.
- To create an interactive dashboard for effective data visualization and user interaction.

Scope of the Project

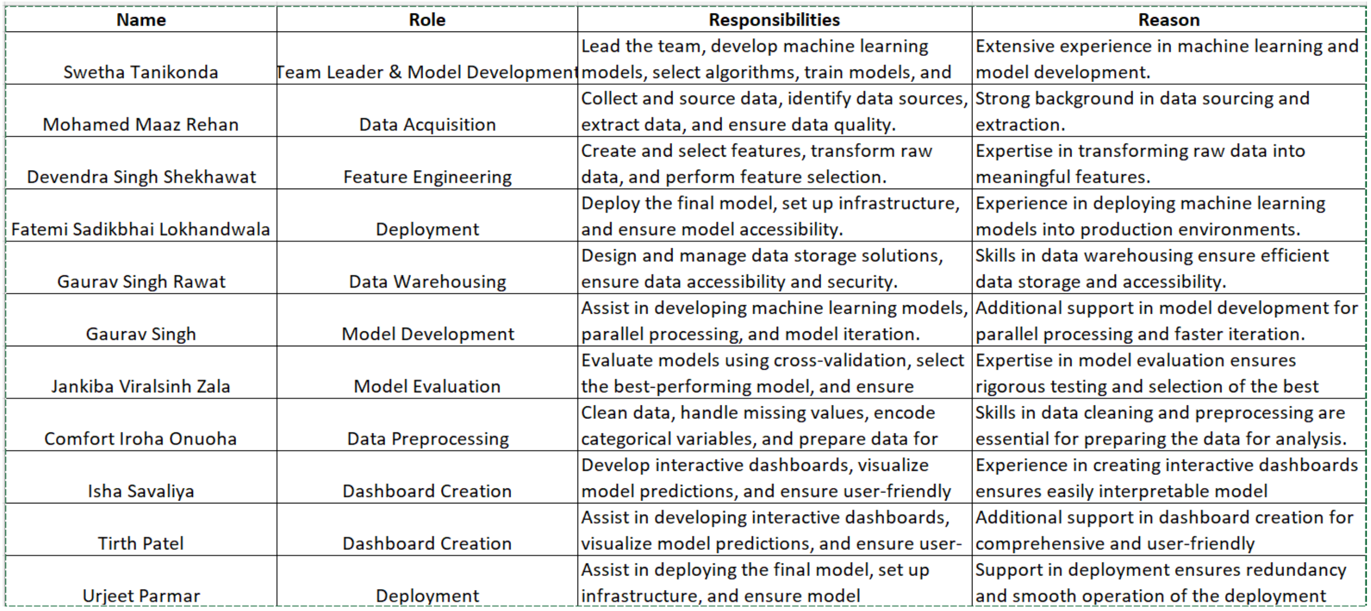
The project will analyze data such as past ticket prices, flight dates, number of stops, and other relevant factors to train the predictive models. The data will include features such as:

- Date of Journey: The date the flight is scheduled.
- Booking date: The date the flight is booked.
- Airline: The Airline operating the flight.
- Flight code: Unique code for each flight.
- Class: Travel class (e.g. economy class, first class, premium class).
- Source and destination: The departure and arrival cities.
- Total Stops: The number of stops during the journey.
- Fare: The price of the ticket (Target variable).

2.Project Management

Roles and Responsibilities

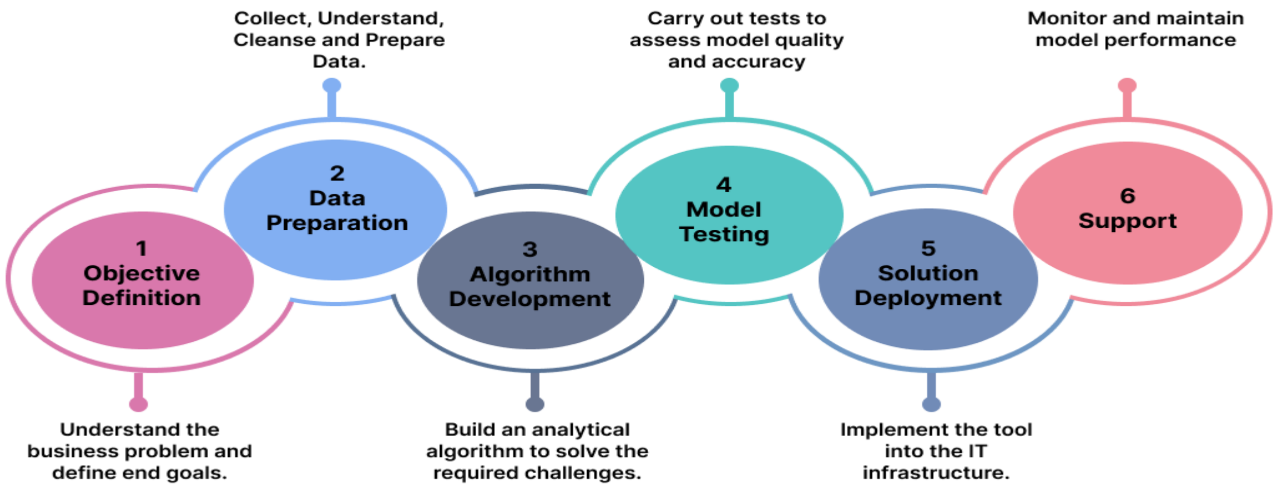
This table shows what roles and responsibilities we have chosen and the reason behind the roles chosen.



Name	Role	Responsibilities	Reason
Swetha Tanikonda	Team Leader & Model Development	Lead the team, develop machine learning models, select algorithms, train models, and	Extensive experience in machine learning and model development.
Mohamed Maaz Rehan	Data Acquisition	Collect and source data, identify data sources, extract data, and ensure data quality.	Strong background in data sourcing and extraction.
Devendra Singh Shekhawat	Feature Engineering	Create and select features, transform raw data, and perform feature selection.	Expertise in transforming raw data into meaningful features.
Fatemi Sadikbhai Lokhandwala	Deployment	Deploy the final model, set up infrastructure, and ensure model accessibility.	Experience in deploying machine learning models into production environments.
Gaurav Singh Rawat	Data Warehousing	Design and manage data storage solutions, ensure data accessibility and security.	Skills in data warehousing ensure efficient data storage and accessibility.
Gaurav Singh	Model Development	Assist in developing machine learning models, parallel processing, and model iteration.	Additional support in model development for parallel processing and faster iteration.
Jankiba Viralsinh Zala	Model Evaluation	Evaluate models using cross-validation, select the best-performing model, and ensure	Expertise in model evaluation ensures rigorous testing and selection of the best
Comfort Iroha Onuoha	Data Preprocessing	Clean data, handle missing values, encode categorical variables, and prepare data for	Skills in data cleaning and preprocessing are essential for preparing the data for analysis.
Isha Savaliya	Dashboard Creation	Develop interactive dashboards, visualize model predictions, and ensure user-friendly	Experience in creating interactive dashboards ensures easily interpretable model
Tirth Patel	Dashboard Creation	Assist in developing interactive dashboards, visualize model predictions, and ensure user-	Additional support in dashboard creation for comprehensive and user-friendly
Urjeet Parmar	Deployment	Assist in deploying the final model, set up infrastructure, and ensure model	Support in deployment ensures redundancy and smooth operation of the deployment

Project Lifecycle

Below image shows us the project lifecycle we are following for this project



Collect, Understand, Cleanse and Prepare Data.
Carry out tests to assess model quality and accuracy
Monitor and maintain model performance
1 Objective Definition
2 Data Preparation
3 Algorithm Development
4 Model Testing
5 Solution Deployment
6 Support
Understand the business problem and define end goals.
Build an analytical algorithm to solve the required challenges.
Implement the tool into the IT infrastructure.

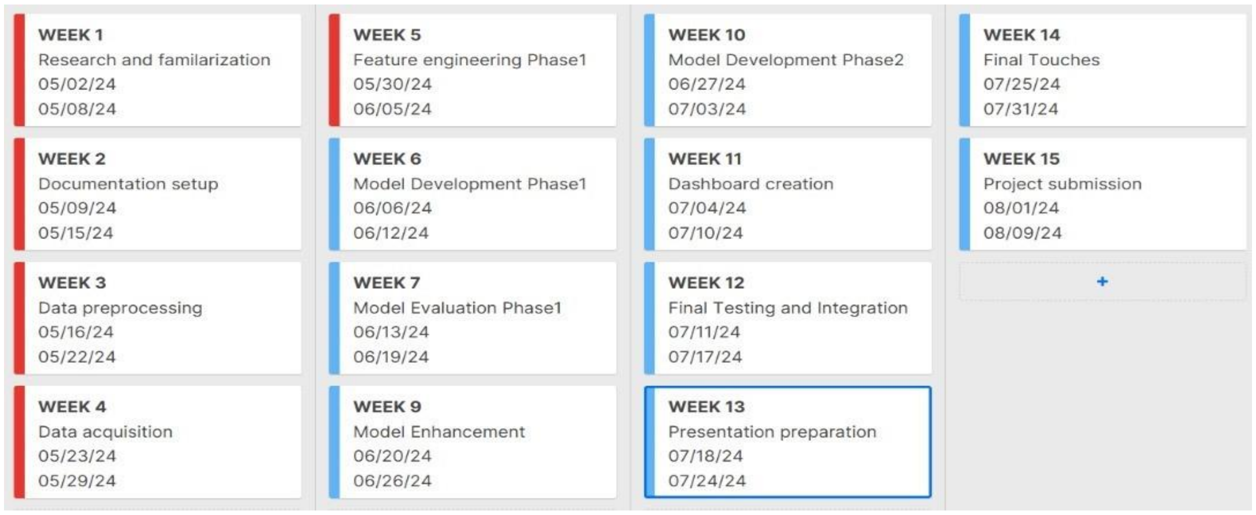
Cost of work and number of hours worked

The table shows us the time invested till now for the project and how much it cost from our pocket for the project.

S.No	Student Name	Role	Total Hours Worked	Cost
1	Swetha Tanikonda	Team Leader & Model Development	46hr	$0
2	Mohamed Maaz Rehan	Data Acquisition	48hr	$0
3	Devendra Singh Shekhawat	Feature Engineering	40hr	$0
4	Fatemi Sadikbhai Lokhandwala	Deployment	42hr	$0
5	Gaurav Singh Rawat	Data Warehousing	44hr	$0
6	Gaurav Singh	Model Development	44hr	$0
7	Jankiba Viralsinh Zala	Model Evaluation	36hr	$0
8	Comfort Iroha Onuoha	Data Preprocessing	44hr	$0
9	Isha Savaliya	Dashboard Creation	36hr	$0
10	Tirth Patel	Dashboard Creation	36hr	$0
11	Urjeet Parmar	Deployment	46hr	$0

Project Timeline

We created our new timeline using Smartsheet which is less cluttered and have a better representation of what we have done till now and what we are going to do in the next weeks.



WEEK 1 Research and familarization 05/02/24 05/08/24	WEEK 5 Feature engineering Phase1 05/30/24 06/05/24	WEEK 10 Model Development Phase2 06/27/24 07/03/24	WEEK 14 Final Touches 07/25/24 07/31/24
WEEK 2 Documentation setup 05/09/24 05/15/24	WEEK 6 Model Development Phase1 06/06/24 06/12/24	WEEK 11 Dashboard creation 07/04/24 07/10/24	WEEK 15 Project submission 08/01/24 08/09/24
WEEK 3 Data preprocessing 05/16/24 05/22/24	WEEK 7 Model Evaluation Phase1 06/13/24 06/19/24	WEEK 12 Final Testing and Integration 07/11/24 07/17/24	+
WEEK 4 Data acquisition 05/23/24 05/29/24	WEEK 9 Model Enhancement 06/20/24 06/26/24	WEEK 13 Presentation preparation 07/18/24 07/24/24	

Accountability and Conflict Resolution:

Methods to Ensure Accountability:

- **Regular Meetings:** We have held weekly meetings to track progress and address any issues.
- **Task Management Tools:** Used tools like Microsoft teams and excel to assign tasks and monitor their completion.
- **Progress Reports:** Required team members to submit progress reports at the end of each week.
- Provided a clear definition of roles and responsibilities to avoid overlap and ensure each member is aware of their tasks.

Conflict Resolution Strategies:

- **Open Communication:** Frequent team meetings and discussions encouraged open communication to address issues promptly.
- **Mediation:** A designated team member facilitated discussions to find mutually acceptable solutions.

Adaptations and Changes:

- **Adjustment of Data Sources:** Initial data source is asking whether we are human or robot while fetching the data; so, we switched from using a Skyscanner to Kayak.
- **Algorithm Switch:** Based on preliminary results, switched from a basic regression model to a more complex ensemble method to improve predictions.
- **Timeline Adjustments:** Extended model development phase to allow for thorough testing and optimization.

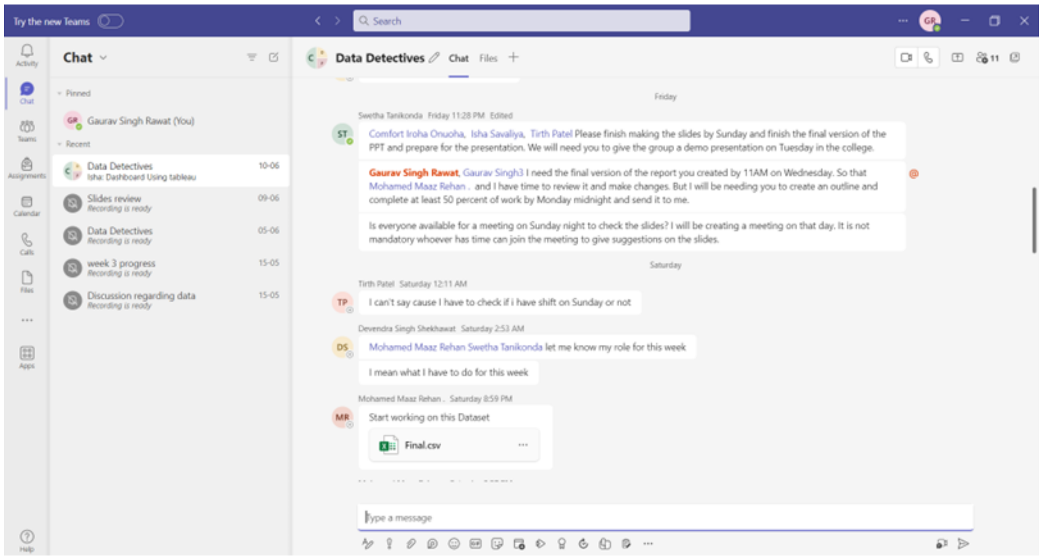
Steps Taken to Execute the Project Plan:

1. **Initial Planning:** Defined project scope, objectives, and team roles.
1. **Data Acquisition:** Mohamed sourced data from multiple APIs and databases.
2. **Data Preprocessing:** Comfort cleaned the data, handled missing values, and encoded categorical variables.
3. **Feature Engineering:** Devendra created new features based on domain knowledge and selected the most relevant ones.
4. **Data Warehousing:** Gaurav designed and managed data storage solutions, ensuring data accessibility and security.
5. **Model Development:** Swetha and Gaurav developed and trained multiple machine learning models.
6. **Model Evaluation:** Jankiba evaluated models using different metrics as necessary.
7. **Deployment:** Fatemi and Urjeet deployed the model using cloud infrastructure and set up APIs.
8. **Dashboard:** Isha and Tirth developed interactive dashboards to visualize the model's predictions using Tableau and PowerBI.

Meeting Documentation/Minutes & Proof of Discussion

For more reference: Link

Effective communication and collaboration are pivotal to the success of any project. Throughout the Flight Price Prediction project, our team maintained active discussions on Microsoft Teams. These discussions included project planning, task assignments, and progress updates. Key decisions and action items were frequently discussed and documented, ensuring that all team members were aligned with the project's goals and objectives.


Chat

Pinned
Gaurav Singh Rawat (You)

Recent
Data Detectives — Isha: Dashboard Using tableau — 10-06
Slides review — Recording is ready — 09-06
Data Detectives — Recording is ready — 05-06
week 3 progress — Recording is ready — 15-05
Discussion regarding data — Recording is ready — 15-05

Data Detectives

Friday

Swetha Tanikonda Friday 11:28 PM Edited

Comfort Iroha Onuoha, Isha Savaliya, Tirth Patel Please finish making the slides by Sunday and finish the final version of the PPT and prepare for the presentation. We will need you to give the group a demo presentation on Tuesday in the college.

Gaurav Singh Rawat, Gaurav Singh3 I need the final version of the report you created by 11AM on Wednesday. So that Mohamed Maaz Rehan . and I have time to review it and make changes. But I will be needing you to create an outline and complete at least 50 percent of work by Monday midnight and send it to me.

Is everyone available for a meeting on Sunday night to check the slides? I will be creating a meeting on that day. It is not mandatory whoever has time can join the meeting to give suggestions on the slides.

Saturday

Tirth Patel Saturday 12:11 AM

I can't say cause I have to check if i have shift on Sunday or not

Devendra Singh Shekhawat Saturday 2:53 AM

Mohamed Maaz Rehan Swetha Tanikonda let me know my role for this week

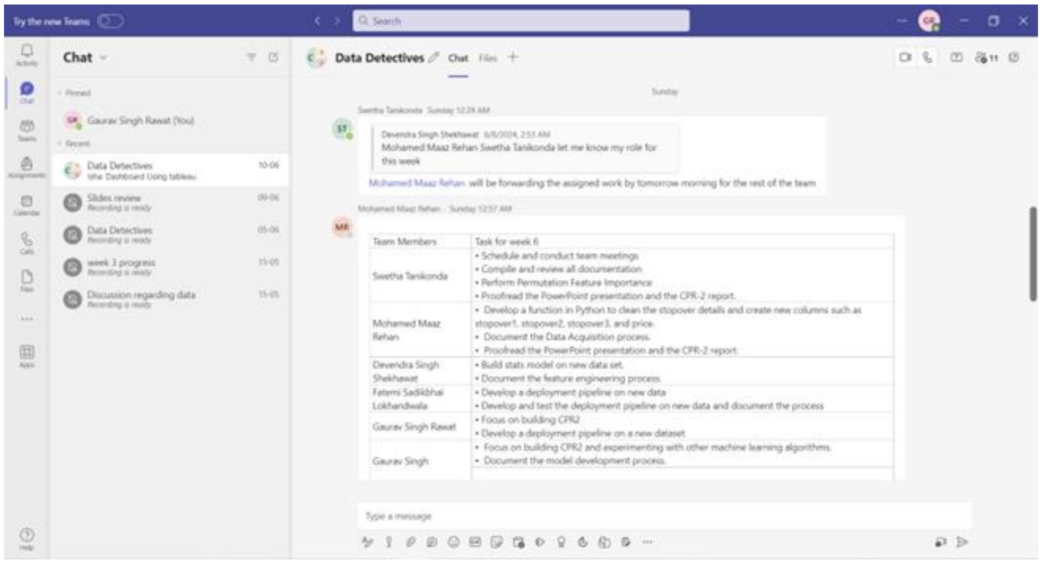
I mean what I have to do for this week

Mohamed Maaz Rehan . Saturday 8:59 PM

Start working on this Dataset

Final.csv

Type a message


Data Detectives

Sunday

Swetha Tanikonda Sunday 12:29 AM

Devendra Singh Shekhawat 6/8/2024, 2:53 AM
Mohamed Maaz Rehan Swetha Tanikonda let me know my role for this week

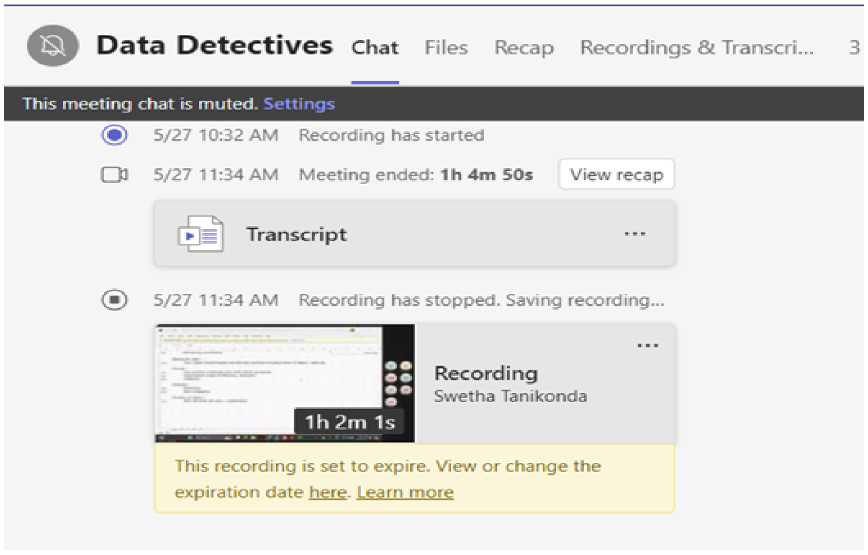
Mohamed Maaz Rehan will be forwarding the assigned work by tomorrow morning for the rest of the team

Mohamed Maaz Rehan . Sunday 12:57 AM

Team Members	Task for week 6
Swetha Tanikonda	• Schedule and conduct team meetings • Compile and review all documentation • Perform Permutation Feature Importance • Proofread the PowerPoint presentation and the CPR-2 report.
Mohamed Maaz Rehan	• Develop a function in Python to clean the stopover details and create new columns such as stopover1, stopover2, stopover3, and price. • Document the Data Acquisition process. • Proofread the PowerPoint presentation and the CPR-2 report.
Devendra Singh Shekhawat	• Build stats model on new data set. • Document the feature engineering process.
Fatemi Sadikbhai Lokhandwala	• Develop a deployment pipeline on new data • Develop and test the deployment pipeline on new data and document the process
Gaurav Singh Rawat	• Focus on building CPR2 • Develop a deployment pipeline on a new dataset
Gaurav Singh	• Focus on building CPR2 and experimenting with other machine learning algorithms. • Document the model development process.

Type a message

In addition to chat discussions, our team conducted regular meetings to review progress and make strategic decisions. We recorded these meetings and transcribed the discussions to ensure thorough documentation. This practice not only provided a reference for future decision-making but also ensured transparency and accountability. The meeting minutes captured key outcomes and milestones, contributing to the project's structured and systematic approach.



This meeting chat is muted. Settings

5/27 10:32 AM Recording has started

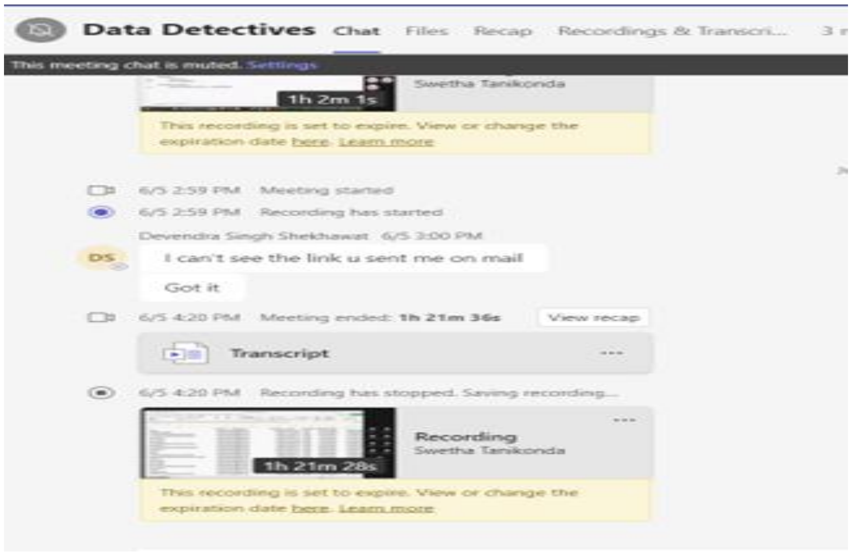
5/27 11:34 AM Meeting ended: **1h 4m 50s** View recap

Transcript ...

5/27 11:34 AM Recording has stopped. Saving recording...

1h 2m 1s
Recording
Swetha Tanikonda

This recording is set to expire. View or change the expiration date here. Learn more



This meeting chat is muted. Settings

1h 2m 1s
Swetha Tanikonda

This recording is set to expire. View or change the expiration date here. Learn more

6/5 2:59 PM Meeting started

6/5 2:59 PM Recording has started

Devendra Singh Shekhawat 6/5 3:00 PM

DS
I can't see the link u sent me on mail

Got it

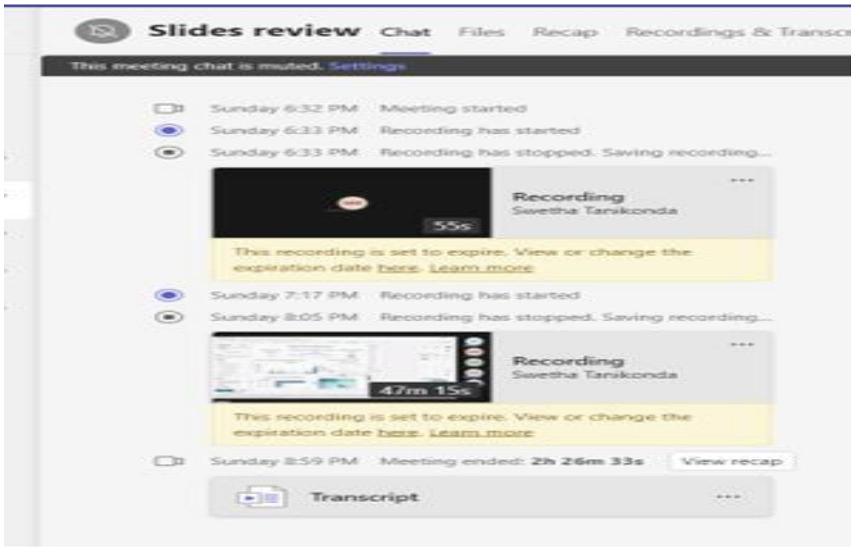
6/5 4:20 PM Meeting ended: **1h 21m 36s** View recap

Transcript ...

6/5 4:20 PM Recording has stopped. Saving recording...

1h 21m 28s
Recording
Swetha Tanikonda

This recording is set to expire. View or change the expiration date here. Learn more



This meeting chat is muted. Settings

Sunday 6:32 PM Meeting started

Sunday 6:33 PM Recording has started

Sunday 6:33 PM Recording has stopped. Saving recording...

55s
Recording
Swetha Tanikonda

This recording is set to expire. View or change the expiration date here. Learn more

Sunday 7:17 PM Recording has started

Sunday 8:05 PM Recording has stopped. Saving recording...

47m 15s
Recording
Swetha Tanikonda

This recording is set to expire. View or change the expiration date here. Learn more

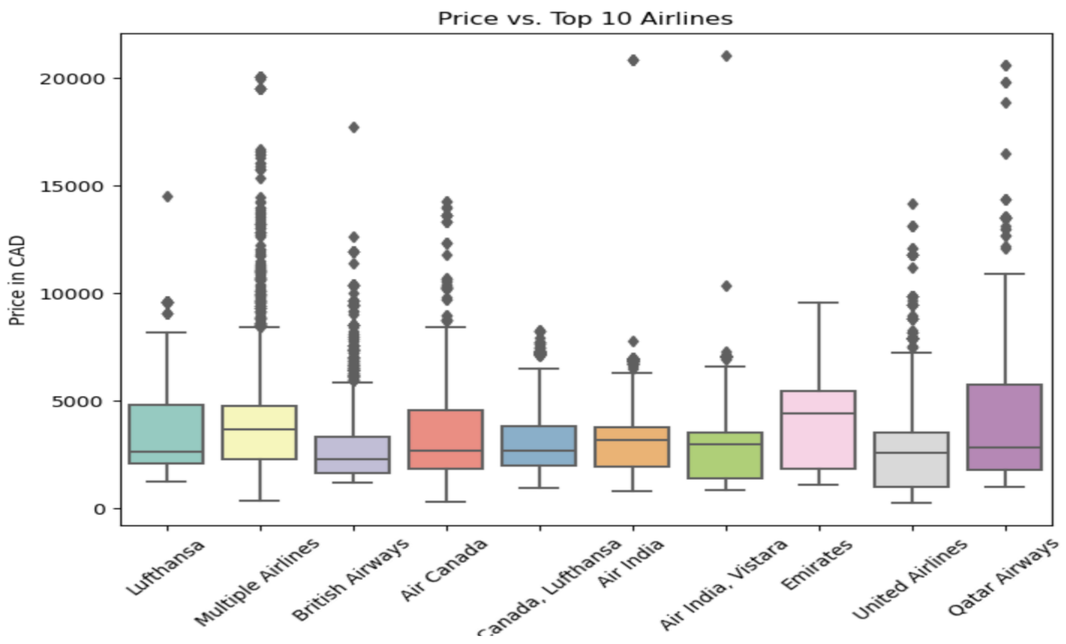
Sunday 8:59 PM Meeting ended: **2h 26m 33s** View recap

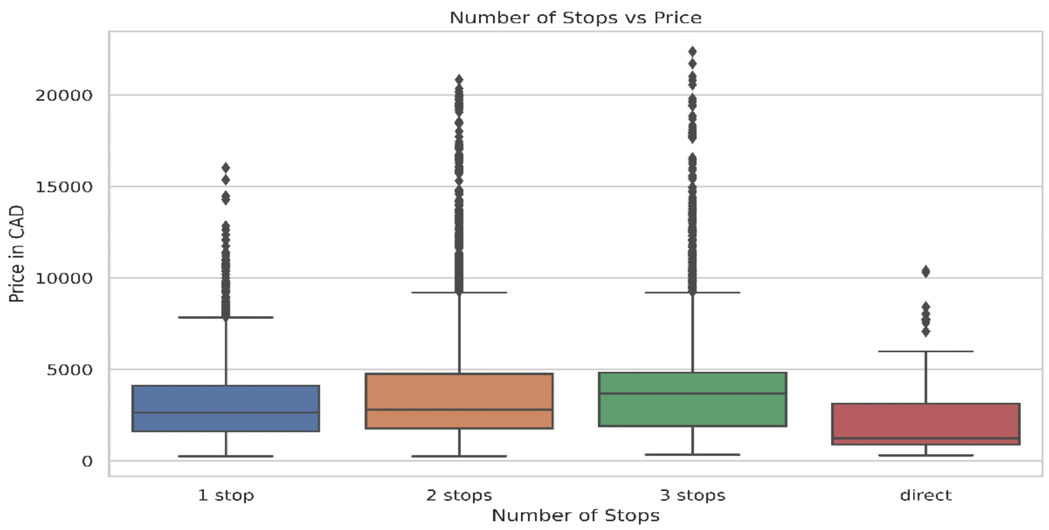
Transcript ...

3.Visualization

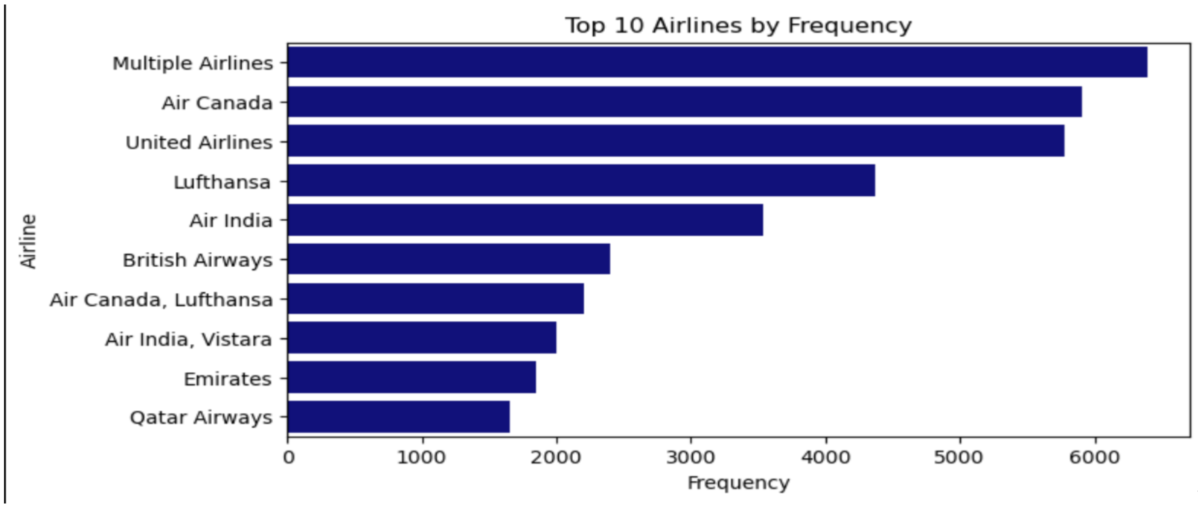
Visualization Techniques

For the EDA, Comfort used various graphs some of them are here present as the data set is new, we have much more outliers than before we used Boxplot Visualization this was used to display the distribution of flight prices, highlighting the median, quartiles, and outliers. Bar Chart were employed to compare the frequency or count of flights across different categories such as airlines. This is effective for categorical data comparison.





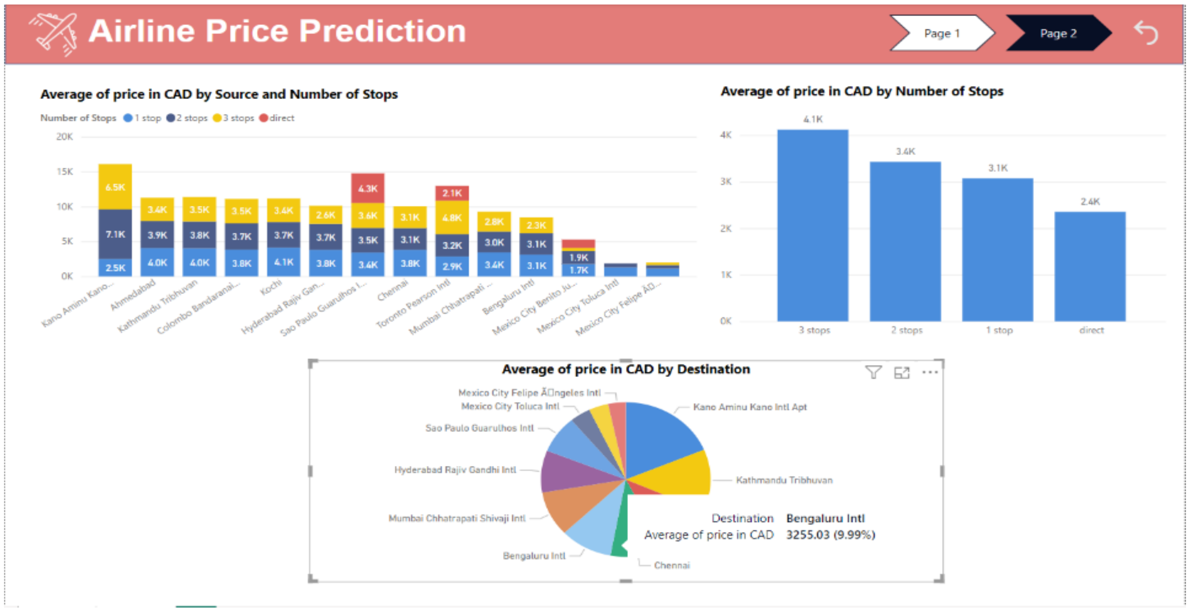
we chose boxplot as part of EDA because it's a straightforward, robust, and efficient way to visualize the distribution, detect outliers, and compare multiple datasets. Its ability to provide a clear summary of large datasets makes it a fundamental tool in exploratory data analysis. We used Python's matplotlib, seaborn and pandas to achieve this.

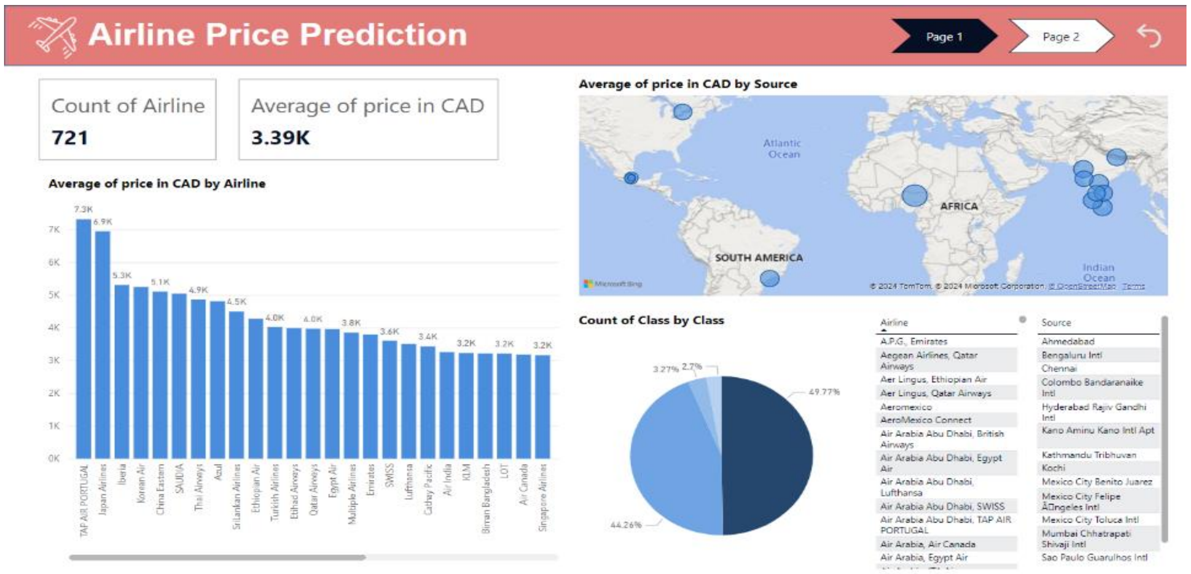


we choose the bar chart because of the need to compare categorical feature (top airlines) The bar chart emphasizes differences, or present aggregated information in a clear, intuitive manner. Its versatility and simplicity make it a fundamental tool for data visualization.

Matching Analysis

Dashboard (Tirth and Isha): The visualizations done in EDA part are also available in dashboard in simpler and interactive way. The visualizations are designed with a general audience in mind, including travelers and data analysts. The techniques used are standard in data visualization, ensuring that they are accessible to both novices and those more experienced with data interpretation. The choice of clear, straightforward graphs ensures that the information is digestible without requiring advanced statistical knowledge. The tableau dashboard is clutter, so we have decided to go with our PowerBI dashboard



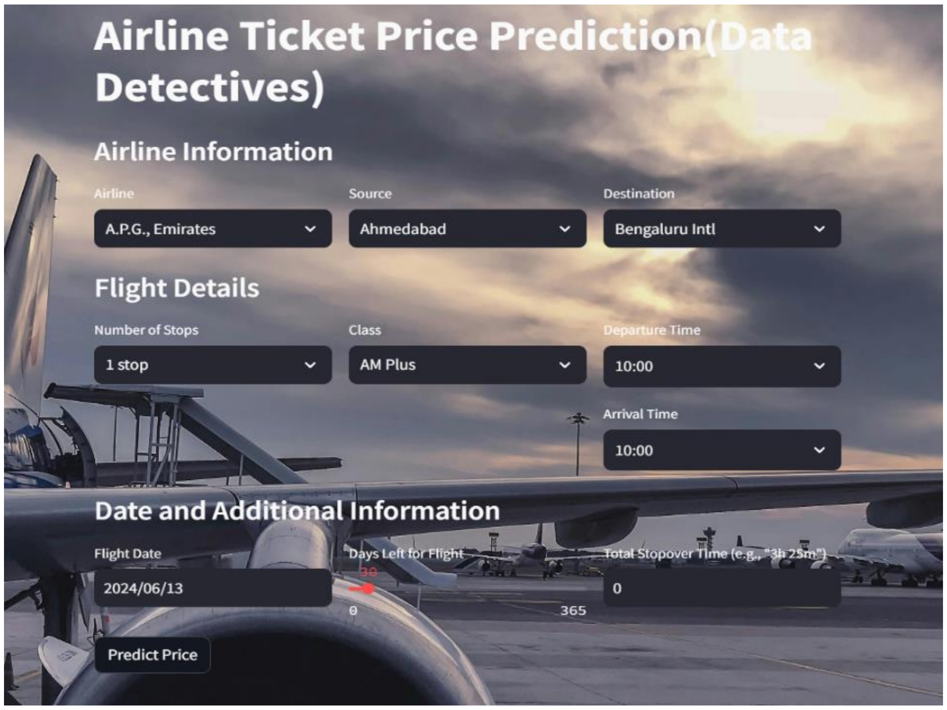


Addressing Multiple Audiences Interactivity

Model Deployment Link Click here: Website_Link

Deployment (Fatemi and Urjeet): we have deployed our website with basic features to predict the price based on training with old dataset. This helps us to connect with the audience who is not interested in the stats or how the things work just and just want to see the result

This is our current deployed website



Airline Ticket Price Prediction(Data Detectives)

Airline Information

Airline: A.P.G., Emirates
Source: Ahmedabad
Destination: Bengaluru Intl

Flight Details

Number of Stops: 1 stop
Class: AM Plus
Departure Time: 10:00
Arrival Time: 10:00

Date and Additional Information

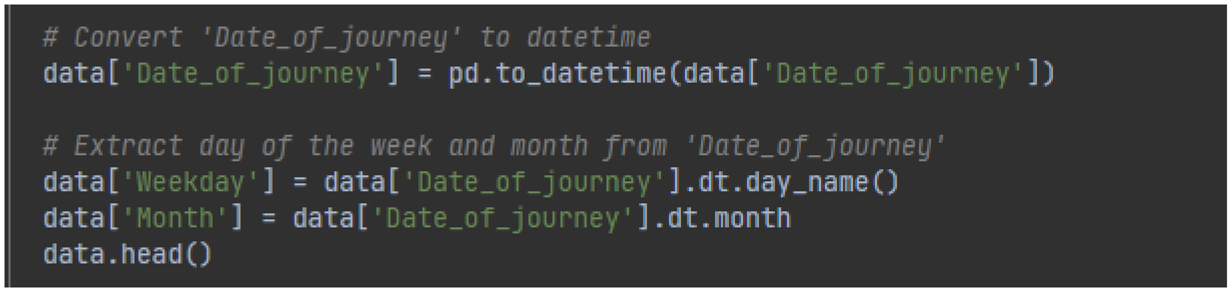
Flight Date: 2024/06/13
Days Left for Flight: 30 (0 – 365)
Total Stopover Time (e.g., "3h 25m"): 0

4.Coding a Solution

Implementation and Version Control

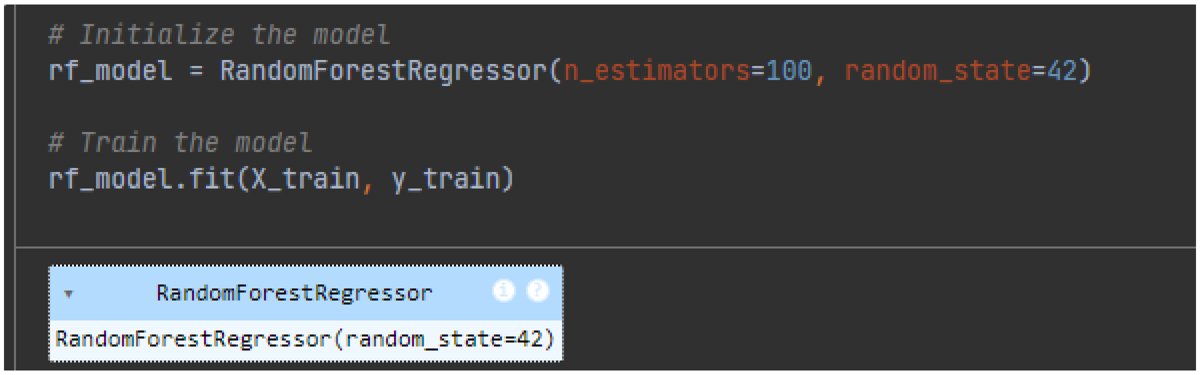
Code Structure-The coding solution for the flight price predictor is designed to accurately predict flight prices based on historical data. The solution involves data preprocessing, feature engineering, model training, and evaluation:

- **Data Preprocessing:** Cleaning and preparing the dataset for analysis.
- **Feature Engineering:** Creating new features to improve model performance (e.g., travel duration, day of the week).



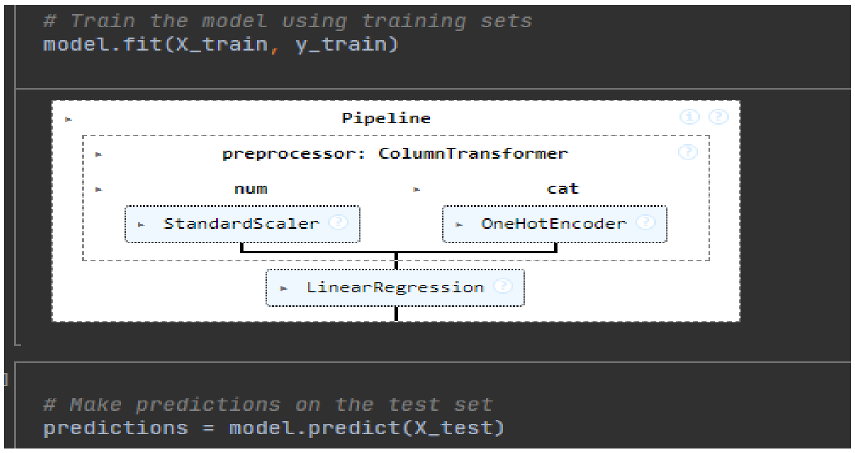
```python
# Convert 'Date_of_journey' to datetime
data['Date_of_journey'] = pd.to_datetime(data['Date_of_journey'])

# Extract day of the week and month from 'Date_of_journey'
data['Weekday'] = data['Date_of_journey'].dt.day_name()
data['Month'] = data['Date_of_journey'].dt.month
data.head()
```

- **Model Training:** Using algorithms like linear regression, Lasso, Ridge, and Elastic Net, XGboost, random forest regressor, decision tree



```python
# Initialize the model
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)

# Train the model
rf_model.fit(X_train, y_train)
```

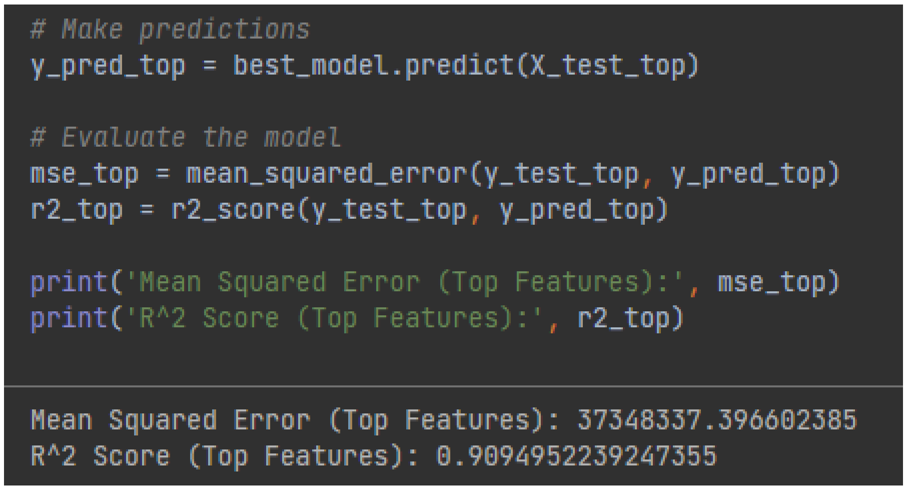
```
RandomForestRegressor
RandomForestRegressor(random_state=42)
```



```python
# Train the model using training sets
model.fit(X_train, y_train)
```

```python
# Make predictions on the test set
predictions = model.predict(X_test)
```

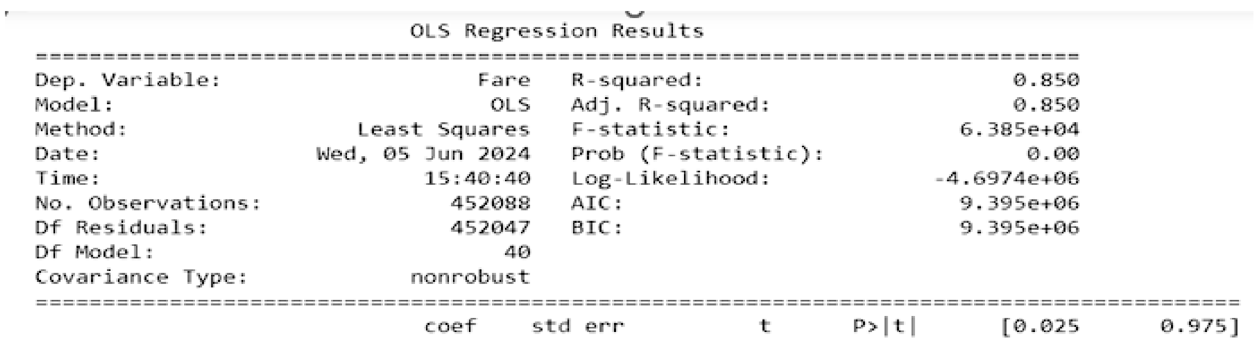
- **Model Evaluation:** Assessing model performance using metrics like accuracy, precision, and recall.



```
# Make predictions
y_pred_top = best_model.predict(X_test_top)

# Evaluate the model
mse_top = mean_squared_error(y_test_top, y_pred_top)
r2_top = r2_score(y_test_top, y_pred_top)

print('Mean Squared Error (Top Features):', mse_top)
print('R^2 Score (Top Features):', r2_top)
```

```
Mean Squared Error (Top Features): 37348337.396602385
R^2 Score (Top Features): 0.9094952239247355
```

Model summary from Stats Model. (Devendra)

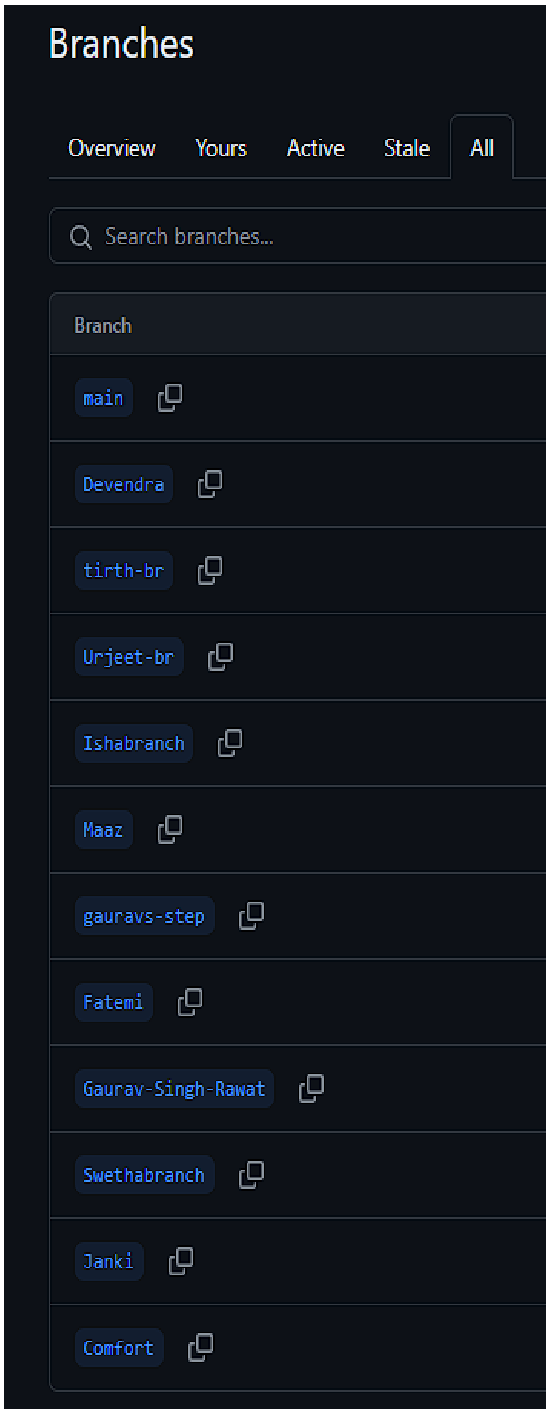


```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   Fare   R-squared:                       0.850
Model:                            OLS   Adj. R-squared:                  0.850
Method:                 Least Squares   F-statistic:                 6.385e+04
Date:                Wed, 05 Jun 2024   Prob (F-statistic):               0.00
Time:                        15:40:40   Log-Likelihood:            -4.6974e+06
No. Observations:              452088   AIC:                         9.395e+06
Df Residuals:                  452047   BIC:                         9.395e+06
Df Model:                          40
Covariance Type:            nonrobust
==========================================================================================
                             coef    std err          t      P>|t|      [0.025      0.975]
```

- **Deployment:** Making the model accessible through a user-friendly interface or API.

Code Commit/Versioning

We used Git for version control to manage code commits and track changes, enabling team members to work collaboratively on branches and merge changes into the main branch. Each team member created branches for their tasks and merged them into the main branch after review. This facilitated tracking code revisions and ensuring consistency.

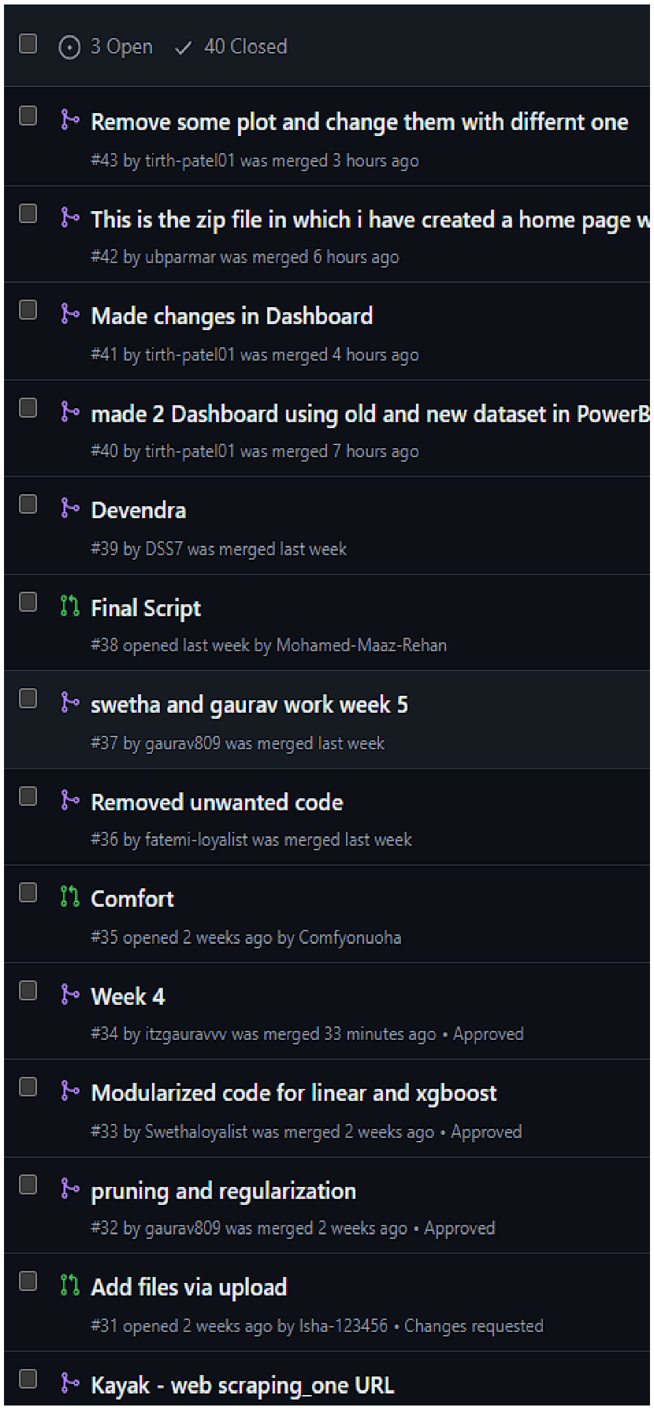


Branches

Overview Yours Active Stale All

Search branches...

Branch
main
Devendra
tirth-br
Urjeet-br
Ishabranch
Maaz
gauravs-step
Fatemi
Gaurav-Singh-Rawat
Swethabranch
Janki
Comfort



3 Open ✓ 40 Closed

Remove some plot and change them with differnt one
#43 by tirth-patel01 was merged 3 hours ago

This is the zip file in which i have created a home page w
#42 by ubparmar was merged 6 hours ago

Made changes in Dashboard
#41 by tirth-patel01 was merged 4 hours ago

made 2 Dashboard using old and new dataset in PowerB
#40 by tirth-patel01 was merged 7 hours ago

Devendra
#39 by DSS7 was merged last week

Final Script
#38 opened last week by Mohamed-Maaz-Rehan

swetha and gaurav work week 5
#37 by gaurav809 was merged last week

Removed unwanted code
#36 by fatemi-loyalist was merged last week

Comfort
#35 opened 2 weeks ago by Comfyonuoha

Week 4
#34 by itzgauravvv was merged 33 minutes ago • Approved

Modularized code for linear and xgboost
#33 by Swethaloyalist was merged 2 weeks ago • Approved

pruning and regularization
#32 by gaurav809 was merged 2 weeks ago • Approved

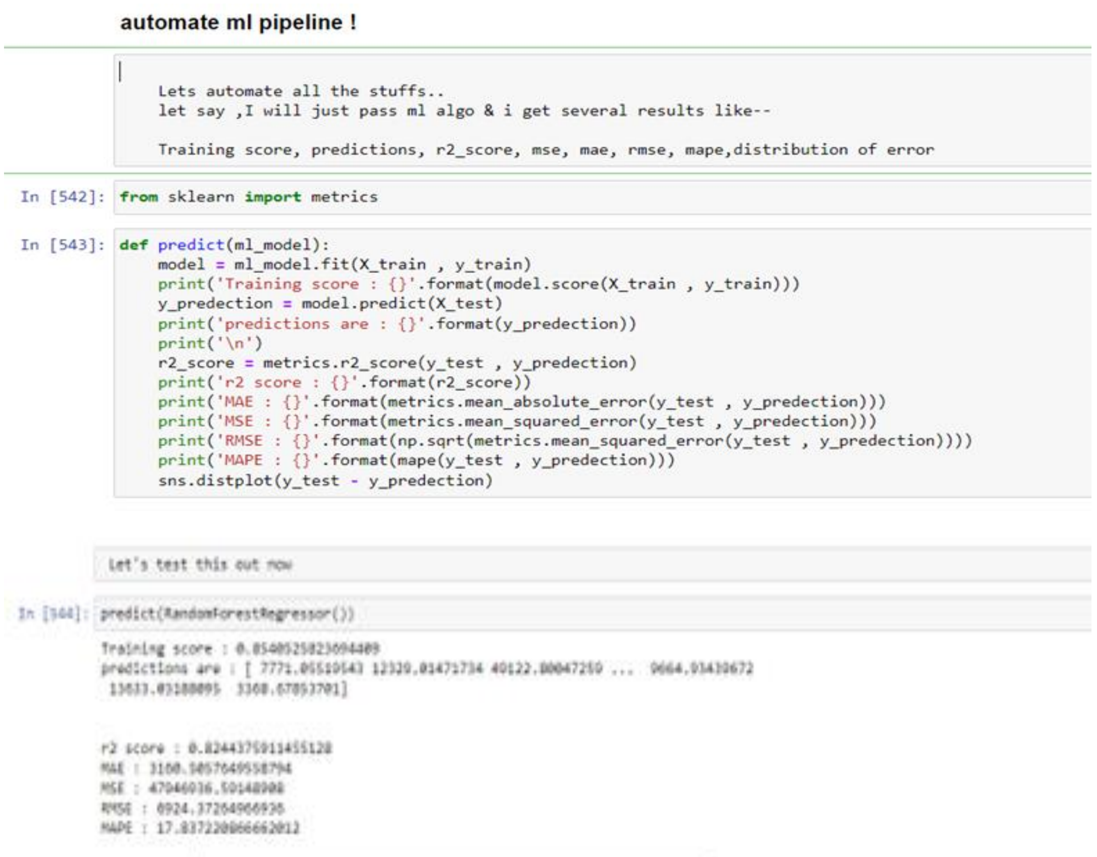
Add files via upload
#31 opened 2 weeks ago by Isha-123456 • Changes requested

Kayak - web scraping_one URL

Testing and Quality Assurance

Unit Testing

We automated a pipeline and created an interesting function that can let say ,I will just pass ml algorithm & I will get several results like - Training score, predictions, R2_score, MSE, MAE, RMSE, MAPE, distribution of error which gave pretty good information with a nice and clean distribution and scores.



automate ml pipeline !

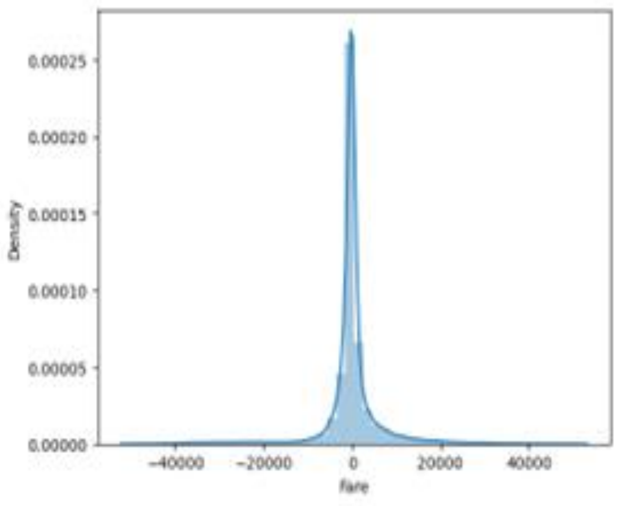
```
Lets automate all the stuffs..
let say ,I will just pass ml algo & i get several results like--

Training score, predictions, r2_score, mse, mae, rmse, mape,distribution of error
```

```
In [542]: from sklearn import metrics
```

```
In [543]: def predict(ml_model):
              model = ml_model.fit(X_train , y_train)
              print('Training score : {}'.format(model.score(X_train , y_train)))
              y_predection = model.predict(X_test)
              print('predictions are : {}'.format(y_predection))
              print('\n')
              r2_score = metrics.r2_score(y_test , y_predection)
              print('r2 score : {}'.format(r2_score))
              print('MAE : {}'.format(metrics.mean_absolute_error(y_test , y_predection)))
              print('MSE : {}'.format(metrics.mean_squared_error(y_test , y_predection)))
              print('RMSE : {}'.format(np.sqrt(metrics.mean_squared_error(y_test , y_predection))))
              print('MAPE : {}'.format(mape(y_test , y_predection)))
              sns.distplot(y_test - y_predection)
```

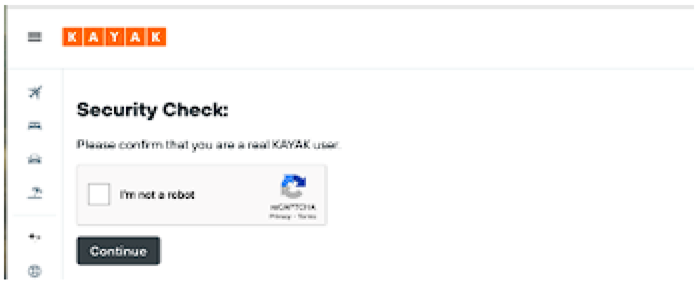
```
let's test this out now
```

```
In [544]: predict(RandomForestRegressor())

Training score : 0.8540525823094409
predictions are : [ 7771.05510543 12320.01471734 40122.80047250 ...  9664.93430672
 13633.03188095  3368.67853701]


r2 score : 0.8244375911455128
MAE : 3160.5057649558794
MSE : 47946036.50148908
RMSE : 6924.37264966936
MAPE : 17.837220866662012
```



Reliability and Edge Case Handling

As the data acquires from Skyscanner is not reliable like we were not able to get our desired dataset and get hit with anti-bot security checks, So we explored alternative solutions like using the Skyscanner API and switching to Kayak for data scraping. We still get security checks, but they were not that frequent

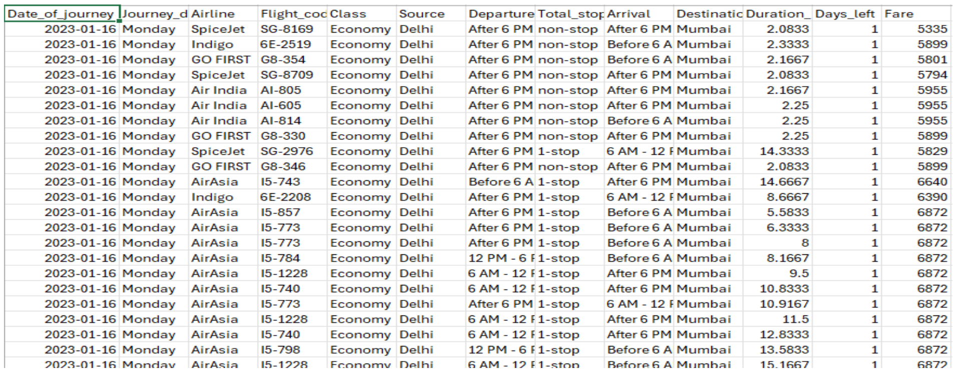


Security Check:

Please confirm that you are a real KAYAK user.

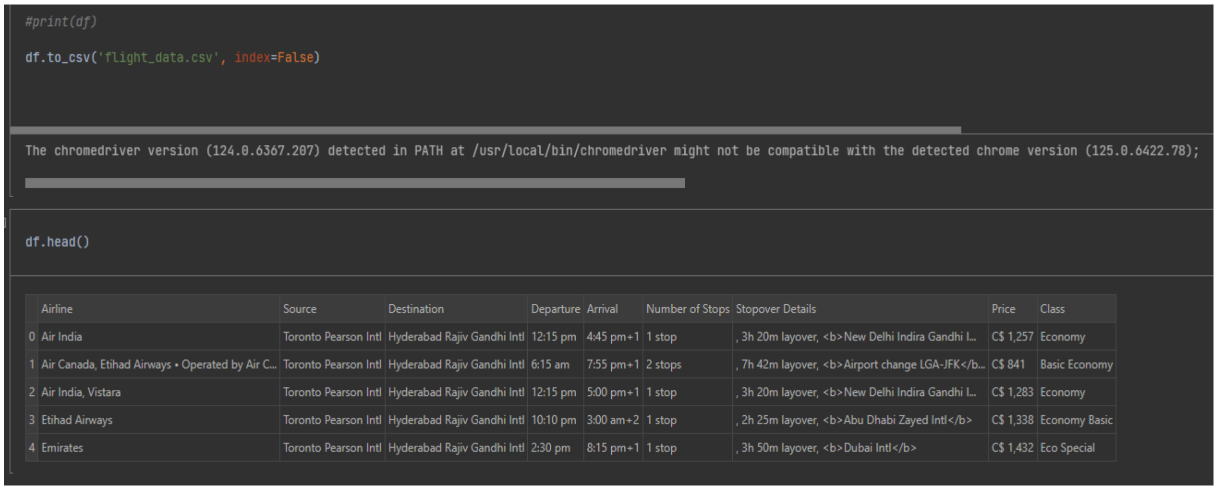
Use of Dummy Data

Dummy data was used during the initial stages of development to test the code's functionality before using the actual dataset. The dataset is found on Kaggle and had the necessary features that we are going to have while scarping the data. For majority of our project, we used flightprice.csv data and now we are shifting to our live dataset



Date_of_journey	Journey_d	Airline	Flight_coc	Class	Source	Departure	Total_stop	Arrival	Destinatic	Duration_	Days_left	Fare
2023-01-16	Monday	SpiceJet	SG-8169	Economy	Delhi	After 6 PM	non-stop	After 6 PM	Mumbai	2.0833	1	5335
2023-01-16	Monday	Indigo	6E-2519	Economy	Delhi	After 6 PM	non-stop	Before 6 A	Mumbai	2.3333	1	5899
2023-01-16	Monday	GO FIRST	G8-354	Economy	Delhi	After 6 PM	non-stop	Before 6 A	Mumbai	2.1667	1	5801
2023-01-16	Monday	SpiceJet	SG-8709	Economy	Delhi	After 6 PM	non-stop	After 6 PM	Mumbai	2.0833	1	5794
2023-01-16	Monday	Air India	AI-805	Economy	Delhi	After 6 PM	non-stop	After 6 PM	Mumbai	2.1667	1	5955
2023-01-16	Monday	Air India	AI-605	Economy	Delhi	After 6 PM	non-stop	After 6 PM	Mumbai	2.25	1	5955
2023-01-16	Monday	Air India	AI-814	Economy	Delhi	After 6 PM	non-stop	Before 6 A	Mumbai	2.25	1	5955
2023-01-16	Monday	GO FIRST	G8-330	Economy	Delhi	After 6 PM	non-stop	After 6 PM	Mumbai	2.25	1	5899
2023-01-16	Monday	SpiceJet	SG-2976	Economy	Delhi	After 6 PM	1-stop	6 AM - 12 F	Mumbai	14.3333	1	5829
2023-01-16	Monday	GO FIRST	G8-346	Economy	Delhi	After 6 PM	non-stop	After 6 PM	Mumbai	2.0833	1	5899
2023-01-16	Monday	AirAsia	I5-743	Economy	Delhi	Before 6 A	1-stop	After 6 PM	Mumbai	14.6667	1	6640
2023-01-16	Monday	Indigo	6E-2208	Economy	Delhi	After 6 PM	1-stop	6 AM - 12 F	Mumbai	8.6667	1	6390
2023-01-16	Monday	AirAsia	I5-857	Economy	Delhi	After 6 PM	1-stop	Before 6 A	Mumbai	5.5833	1	6872
2023-01-16	Monday	AirAsia	I5-773	Economy	Delhi	After 6 PM	1-stop	Before 6 A	Mumbai	6.3333	1	6872
2023-01-16	Monday	AirAsia	I5-773	Economy	Delhi	After 6 PM	1-stop	Before 6 A	Mumbai	8	1	6872
2023-01-16	Monday	AirAsia	I5-784	Economy	Delhi	12 PM - 6 F	1-stop	Before 6 A	Mumbai	8.1667	1	6872
2023-01-16	Monday	AirAsia	I5-1228	Economy	Delhi	6 AM - 12 F	1-stop	After 6 PM	Mumbai	9.5	1	6872
2023-01-16	Monday	AirAsia	I5-740	Economy	Delhi	6 AM - 12 F	1-stop	After 6 PM	Mumbai	10.8333	1	6872
2023-01-16	Monday	AirAsia	I5-773	Economy	Delhi	After 6 PM	1-stop	6 AM - 12 F	Mumbai	10.9167	1	6872
2023-01-16	Monday	AirAsia	I5-1228	Economy	Delhi	6 AM - 12 F	1-stop	After 6 PM	Mumbai	11.5	1	6872
2023-01-16	Monday	AirAsia	I5-740	Economy	Delhi	6 AM - 12 F	1-stop	After 6 PM	Mumbai	12.8333	1	6872
2023-01-16	Monday	AirAsia	I5-798	Economy	Delhi	12 PM - 6 F	1-stop	Before 6 A	Mumbai	13.5833	1	6872
2023-01-16	Monday	AirAsia	I5-1228	Economy	Delhi	6 AM - 12 F	1-stop	Before 6 A	Mumbai	15.1667	1	6872

We have the live dataset (unclean): we are cleaning the dataset and including it in our pipeline



```
#print(df)

df.to_csv('flight_data.csv', index=False)
```

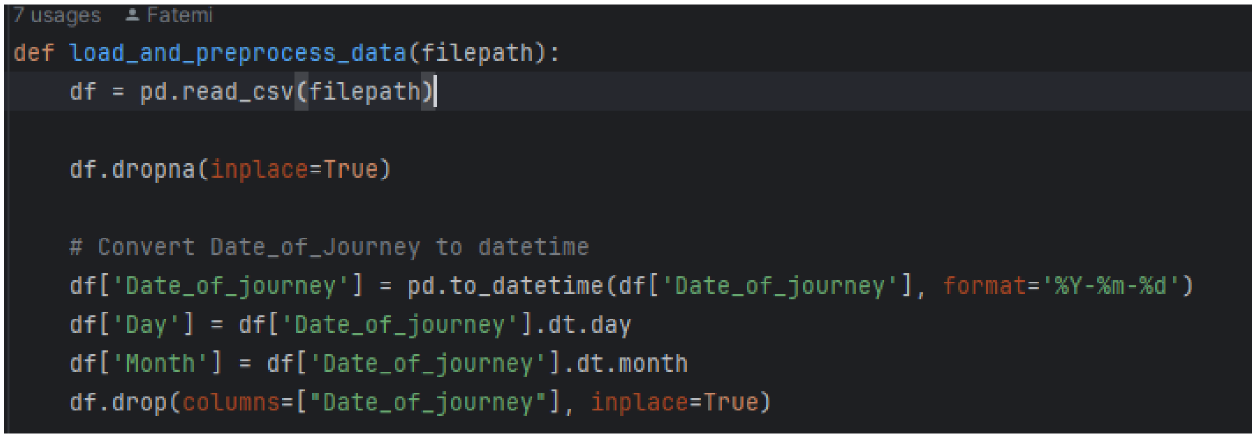
```
The chromedriver version (124.0.6367.207) detected in PATH at /usr/local/bin/chromedriver might not be compatible with the detected chrome version (125.0.6422.78);
```

```
df.head()
```

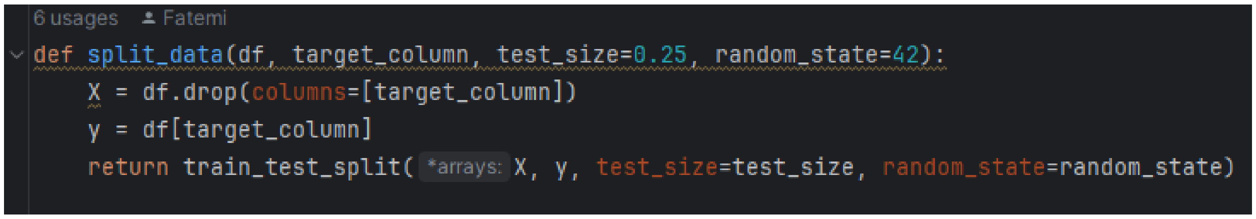
	Airline	Source	Destination	Departure	Arrival	Number of Stops	Stopover Details	Price	Class
0	Air India	Toronto Pearson Intl	Hyderabad Rajiv Gandhi Intl	12:15 pm	4:45 pm+1	1 stop	, 3h 20m layover, <b>New Delhi Indira Gandhi I...	C$ 1,257	Economy
1	Air Canada, Etihad Airways • Operated by Air C...	Toronto Pearson Intl	Hyderabad Rajiv Gandhi Intl	6:15 am	7:55 pm+1	2 stops	, 7h 42m layover, <b>Airport change LGA-JFK</b...	C$ 841	Basic Economy
2	Air India, Vistara	Toronto Pearson Intl	Hyderabad Rajiv Gandhi Intl	12:15 pm	5:00 pm+1	1 stop	, 3h 20m layover, <b>New Delhi Indira Gandhi I...	C$ 1,283	Economy
3	Etihad Airways	Toronto Pearson Intl	Hyderabad Rajiv Gandhi Intl	10:10 pm	3:00 am+2	1 stop	, 2h 25m layover, <b>Abu Dhabi Zayed Intl</b>	C$ 1,338	Economy Basic
4	Emirates	Toronto Pearson Intl	Hyderabad Rajiv Gandhi Intl	2:30 pm	8:15 pm+1	1 stop	, 3h 50m layover, <b>Dubai Intl</b>	C$ 1,432	Eco Special

Modular Structure and Efficiency

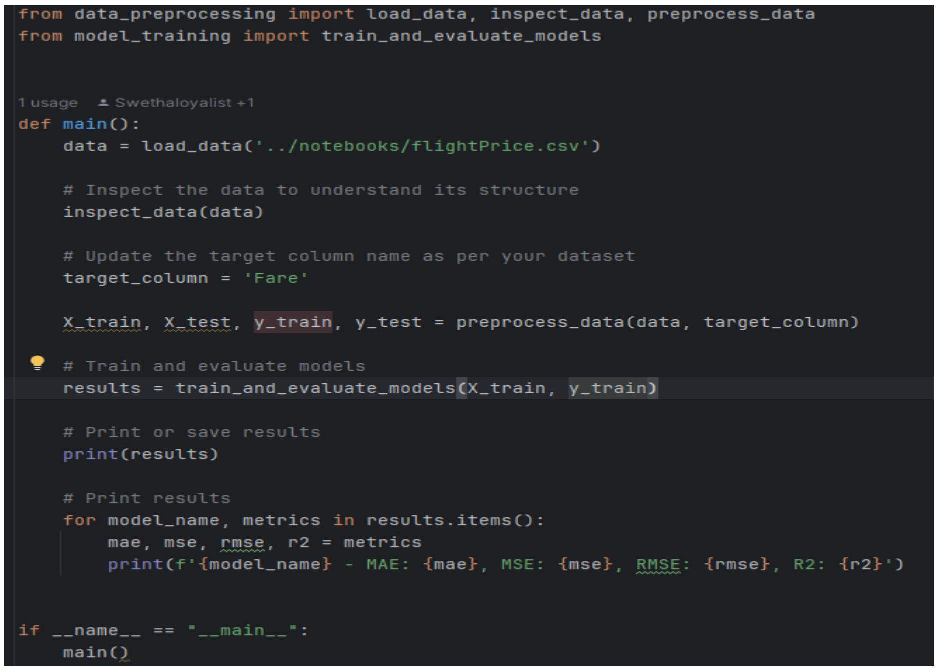
The code is organized into modules, each responsible for a specific task, such as data preprocessing, feature engineering, model training, and evaluation. This modular structure makes the code easy to understand, maintain, and extend. The code is optimized for efficiency by using vectorized operations with libraries like NumPy and pandas. This reduces the computational overhead and speeds up the processing time.



```python
def load_and_preprocess_data(filepath):
    df = pd.read_csv(filepath)

    df.dropna(inplace=True)

    # Convert Date_of_Journey to datetime
    df['Date_of_journey'] = pd.to_datetime(df['Date_of_journey'], format='%Y-%m-%d')
    df['Day'] = df['Date_of_journey'].dt.day
    df['Month'] = df['Date_of_journey'].dt.month
    df.drop(columns=["Date_of_journey"], inplace=True)
```



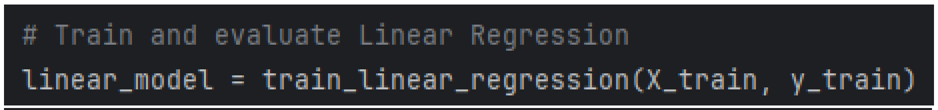
```python
def split_data(df, target_column, test_size=0.25, random_state=42):
    X = df.drop(columns=[target_column])
    y = df[target_column]
    return train_test_split(X, y, test_size=test_size, random_state=random_state)
```



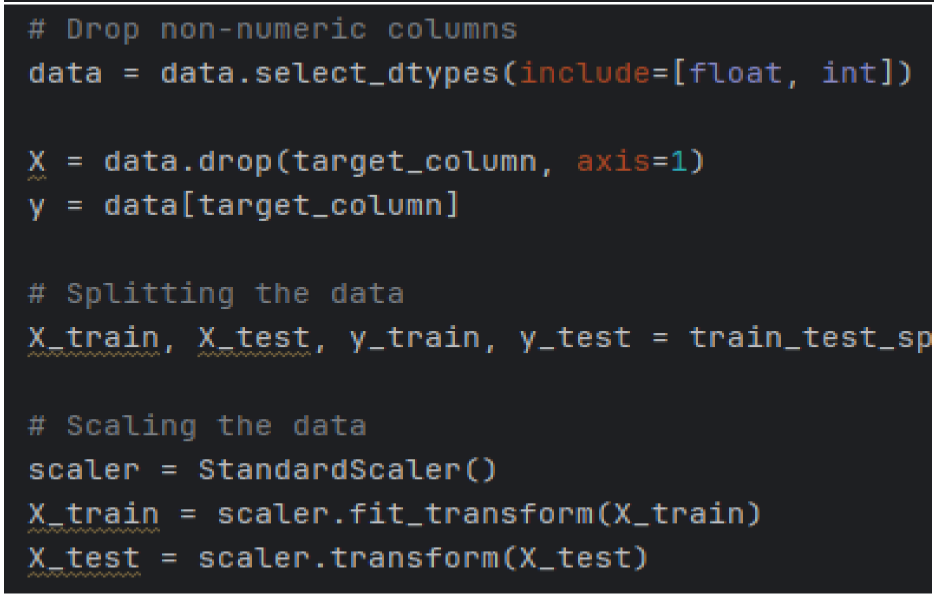
```python
from data_preprocessing import load_data, inspect_data, preprocess_data
from model_training import train_and_evaluate_models


def main():
    data = load_data('../notebooks/flightPrice.csv')

    # Inspect the data to understand its structure
    inspect_data(data)

    # Update the target column name as per your dataset
    target_column = 'Fare'

    X_train, X_test, y_train, y_test = preprocess_data(data, target_column)

    # Train and evaluate models
    results = train_and_evaluate_models(X_train, y_train)

    # Print or save results
    print(results)

    # Print results
    for model_name, metrics in results.items():
        mae, mse, rmse, r2 = metrics
        print(f'{model_name} - MAE: {mae}, MSE: {mse}, RMSE: {rmse}, R2: {r2}')


if __name__ == "__main__":
    main()
```

Code Quality and Best Practices

Commenting -The code snippet shows comments explaining the purpose of each block of code, we try to make it a regular practice to add comment for each block of code. Such as in below example Clear and concise comments were added throughout the code to explain its purpose and functionality.

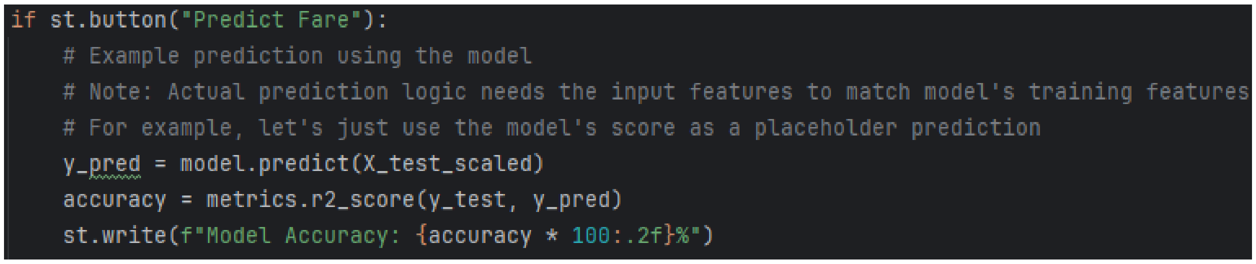


```
# Train and evaluate Linear Regression
linear_model = train_linear_regression(X_train, y_train)
```



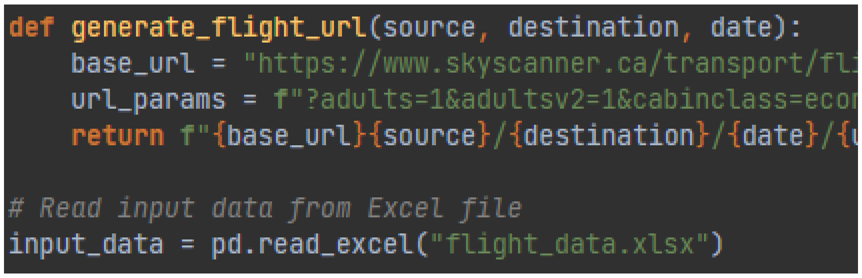
```
# Drop non-numeric columns
data = data.select_dtypes(include=[float, int])

X = data.drop(target_column, axis=1)
y = data[target_column]

# Splitting the data
X_train, X_test, y_train, y_test = train_test_sp

# Scaling the data
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Readability and Self-Explanatory Code- The code includes a button to predict fares and displays model accuracy using Streamlit. Naming conventions, code indentation, and logical organization were implemented to enhance readability and maintainability.



```
if st.button("Predict Fare"):
    # Example prediction using the model
    # Note: Actual prediction logic needs the input features to match model's training features
    # For example, let's just use the model's score as a placeholder prediction
    y_pred = model.predict(X_test_scaled)
    accuracy = metrics.r2_score(y_test, y_pred)
    st.write(f"Model Accuracy: {accuracy * 100:.2f}%")
```

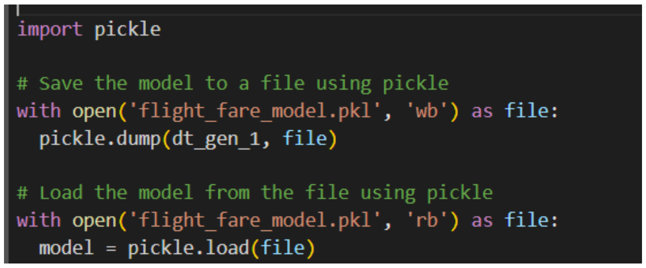
The function generate_flight_url clearly indicates its purpose through its name, which is to generate a URL for flight searches based on given parameters**.**



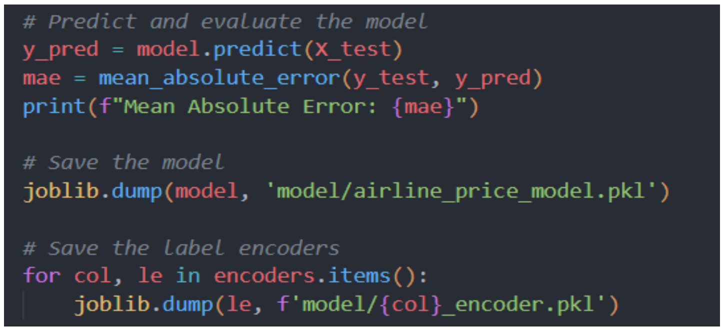
```
def generate_flight_url(source, destination, date):
    base_url = "https://www.skyscanner.ca/transport/fl
    url_params = f"?adults=1&adultsv2=1&cabinclass=eco
    return f"{base_url}{source}/{destination}/{date}/{

# Read input data from Excel file
input_data = pd.read_excel("flight_data.xlsx")
```

Performance

Speed/Latency: for the deployment team **(Fatemi and Urjeet) instead** of using function we used dumped model and used joblib and pickle tested which is fast and reliable. Which reduces the time need to predict the price on the website.



```
import pickle

# Save the model to a file using pickle
with open('flight_fare_model.pkl', 'wb') as file:
  pickle.dump(dt_gen_1, file)

# Load the model from the file using pickle
with open('flight_fare_model.pkl', 'rb') as file:
  model = pickle.load(file)
```

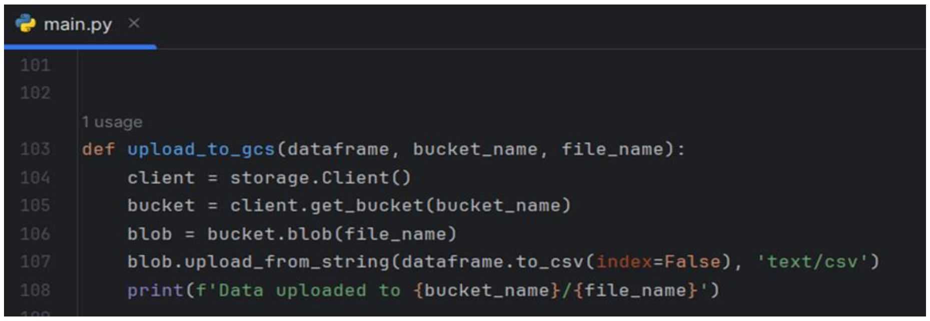


```
# Predict and evaluate the model
y_pred = model.predict(X_test)
mae = mean_absolute_error(y_test, y_pred)
print(f"Mean Absolute Error: {mae}")

# Save the model
joblib.dump(model, 'model/airline_price_model.pkl')

# Save the label encoders
for col, le in encoders.items():
    joblib.dump(le, f'model/{col}_encoder.pkl')
```

Adaptability and Scalability

For scaling and adaptability, we use Google Cloud Storage (GCS) which will help us to make the pipeline more reliable and make it more autonomous, below are the things we have done:

Cloud Storage Integration: By uploading data to GCS, you leverage the scalable infrastructure of Google Cloud. GCS is designed to handle large amounts of data efficiently, ensuring that your data storage can grow as your dataset increases.
Centralized Data Repository: Storing data in GCS provides a centralized repository that can be accessed by various data processing and machine learning services within Google Cloud. This centralization is essential for scaling operations across different services.
Cost Efficiency: Google Cloud Storage offers various storage classes (e.g., Standard, Nearline, Coldline) that can be used to optimize cost depending on your data access patterns. This flexibility helps manage storage costs effectively as data scales.
Easy Data Management: The use of the `upload_to_gcs` function allows for programmatically uploading data, which is crucial for automated data pipelines. Automated pipelines are key to scaling as they reduce the need for manual intervention and ensure consistent data handling.
High Availability and Durability: GCS provides high availability and durability for stored data, ensuring that your data is safe and accessible even as the volume increases.
Integration with Other GCP Services: Data in GCS can be easily integrated with other GCP services like BigQuery for analytics, Dataflow for data processing, and AI Platform for machine learning. This integration is essential for building scalable data processing and analysis pipelines.


main.py ×

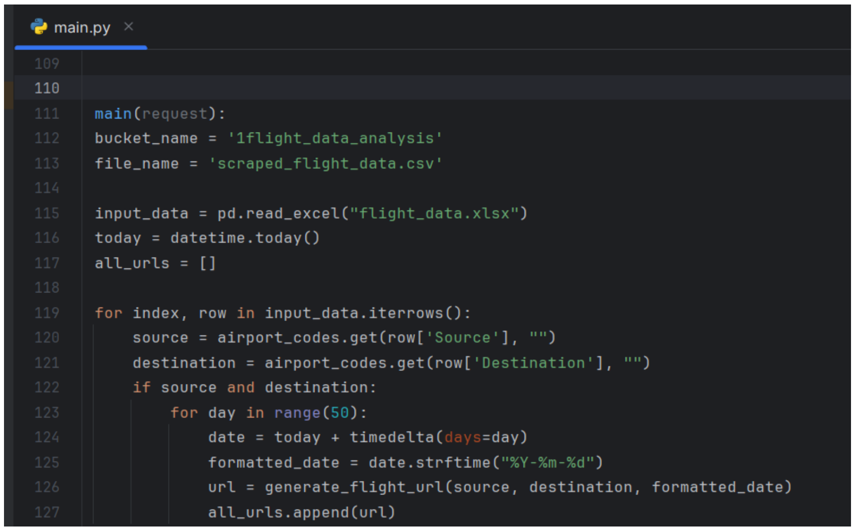
```python


1 usage
def upload_to_gcs(dataframe, bucket_name, file_name):
    client = storage.Client()
    bucket = client.get_bucket(bucket_name)
    blob = bucket.blob(file_name)
    blob.upload_from_string(dataframe.to_csv(index=False), 'text/csv')
    print(f'Data uploaded to {bucket_name}/{file_name}')
```

The upload_to_gcs function uploads the extracted data to GCS as a CSV file.

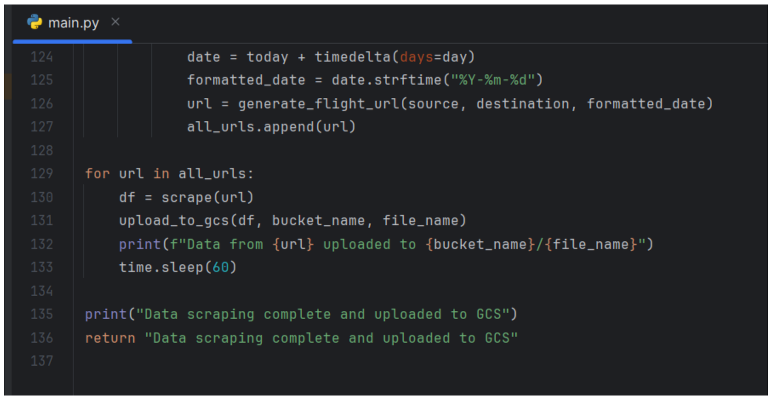
The function upload_to_gcs has the following parameters:

- **dataframe**: The pandas Data Frame that you want to upload.
- **bucket_name**: The name of the GCS bucket where the file will be stored.
- **file_name**: The name of the CSV file to be created in GCS.

By using this code, you ensure that your data is stored in a scalable, reliable, and easily accessible manner, which is critical for handling growing amounts of data and increasing the number of users or operations that rely on this data.


main.py ×

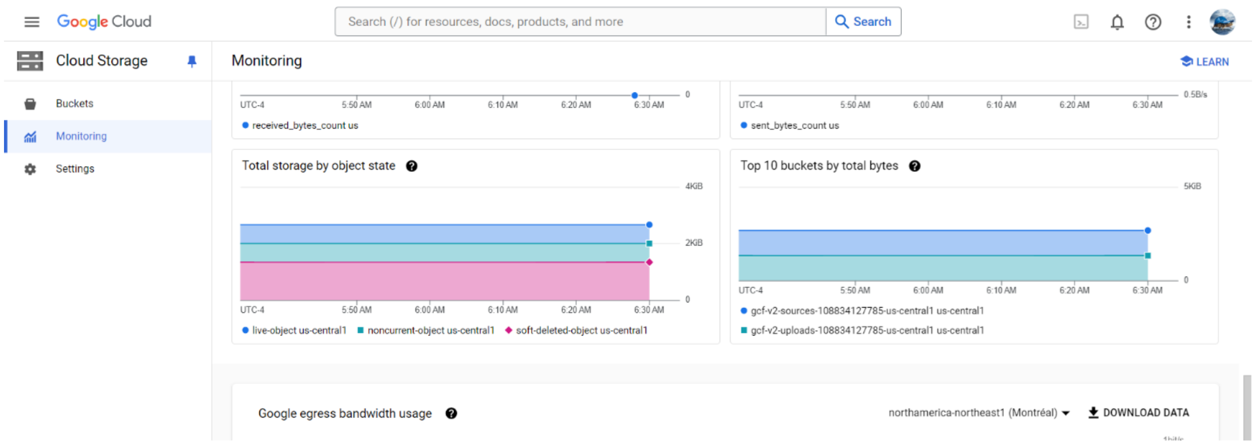
```python


main(request):
bucket_name = '1flight_data_analysis'
file_name = 'scraped_flight_data.csv'

input_data = pd.read_excel("flight_data.xlsx")
today = datetime.today()
all_urls = []

for index, row in input_data.iterrows():
    source = airport_codes.get(row['Source'], "")
    destination = airport_codes.get(row['Destination'], "")
    if source and destination:
        for day in range(50):
            date = today + timedelta(days=day)
            formatted_date = date.strftime("%Y-%m-%d")
            url = generate_flight_url(source, destination, formatted_date)
            all_urls.append(url)
```


main.py ×

```python
            date = today + timedelta(days=day)
            formatted_date = date.strftime("%Y-%m-%d")
            url = generate_flight_url(source, destination, formatted_date)
            all_urls.append(url)

for url in all_urls:
    df = scrape(url)
    upload_to_gcs(df, bucket_name, file_name)
    print(f"Data from {url} uploaded to {bucket_name}/{file_name}")
    time.sleep(60)

print("Data scraping complete and uploaded to GCS")
return "Data scraping complete and uploaded to GCS"

```

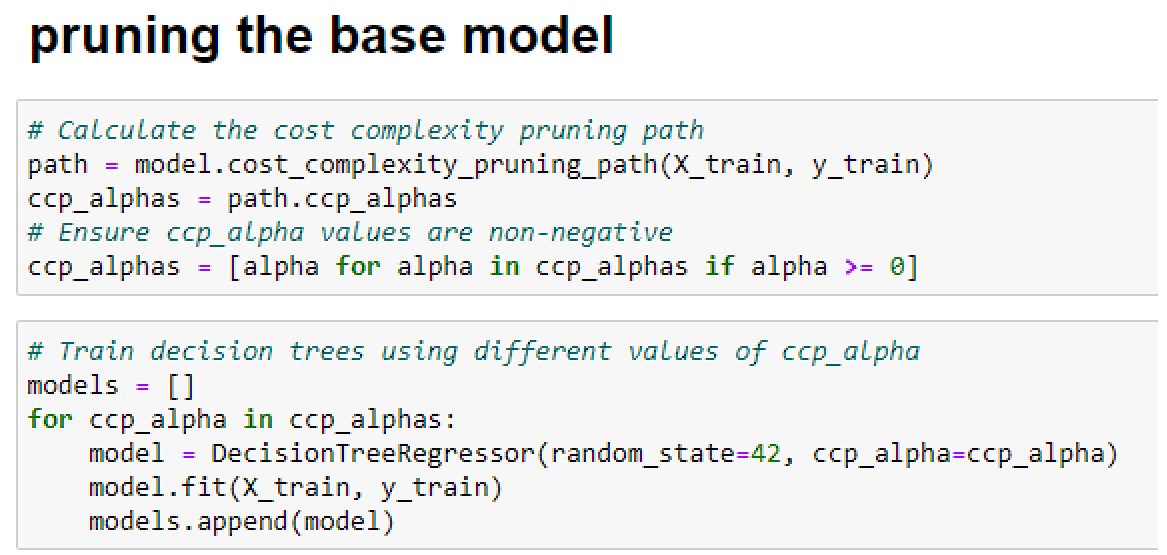
This function serves as the entry point for the script. It:

- Reads the input data from an Excel file.
- Generates flight search URLs for a range of dates.
- Scrapes flight data from each URL.
- Uploads the scraped data to Google Cloud Storage.
- Includes a delay of 60 seconds between scraping each URL to avoid overloading the server.


Google Cloud
Search (/) for resources, docs, products, and more
Search
Cloud Storage
Monitoring
LEARN
Buckets
Monitoring
Settings
received_bytes_count us
sent_bytes_count us
Total storage by object state
4KB
2KB
0
UTC-4
5:50 AM
6:00 AM
6:10 AM
6:20 AM
6:30 AM
live-object us-central1
noncurrent-object us-central1
soft-deleted-object us-central1
Top 10 buckets by total bytes
5KB
0.5B/s
gcf-v2-sources-108834127785-us-central1 us-central1
gcf-v2-uploads-108834127785-us-central1 us-central1
Google egress bandwidth usage
northamerica-northeast1 (Montréal)
DOWNLOAD DATA

Robustness and Stress Testing

Stress Testing (CCP) during calculating cost complexity pruning it was to much for the system to handle and ends in crashing or giving error as not enough memory to allocate.

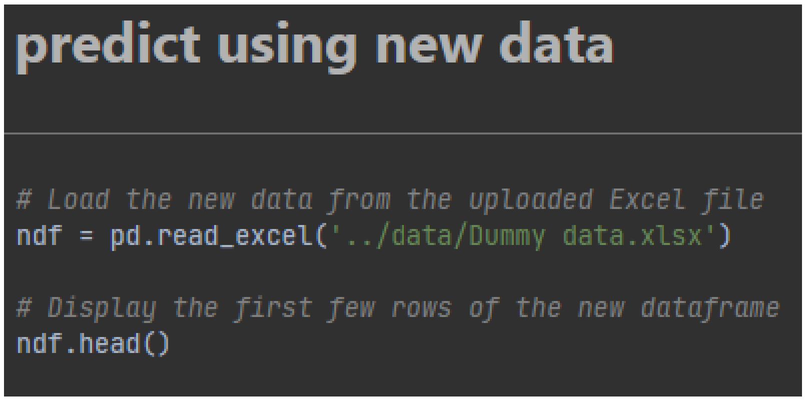


pruning the base model

```
# Calculate the cost complexity pruning path
path = model.cost_complexity_pruning_path(X_train, y_train)
ccp_alphas = path.ccp_alphas
# Ensure ccp_alpha values are non-negative
ccp_alphas = [alpha for alpha in ccp_alphas if alpha >= 0]
```

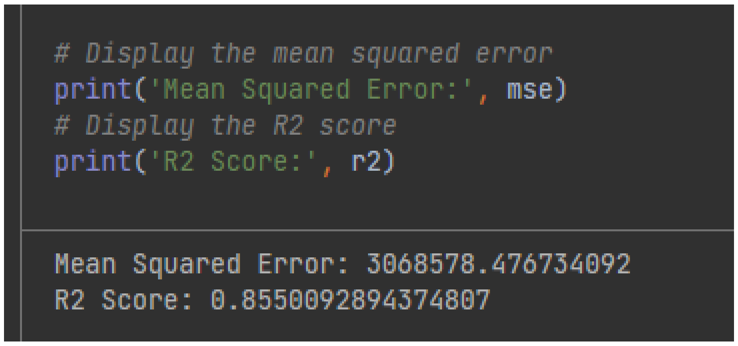
```
# Train decision trees using different values of ccp_alpha
models = []
for ccp_alpha in ccp_alphas:
    model = DecisionTreeRegressor(random_state=42, ccp_alpha=ccp_alpha)
    model.fit(X_train, y_train)
    models.append(model)
```

The code was designed for robustness and maintainability, incorporating exception handling and error checks to minimize potential issues. As we also try checking if the model can predict correctly with new data set (still dummy) we were able to do that after some preprocessing of the data, though r2 value fall.



predict using new data

```python
# Load the new data from the uploaded Excel file
ndf = pd.read_excel('../data/Dummy data.xlsx')

# Display the first few rows of the new dataframe
ndf.head()
```



```python
# Display the mean squared error
print('Mean Squared Error:', mse)
# Display the R2 score
print('R2 Score:', r2)
```

```
Mean Squared Error: 3068578.476734092
R2 Score: 0.8550092894374807
```

Collaboration and Consistency

Collaboration:

- Regular Meetings: Team members held regular meetings to track progress, discuss challenges, and plan the next steps. This ensured that everyone was on the same page and any issues were promptly addressed.
- Task Management Tools: Tools like Microsoft Teams and Excel were used to assign tasks, monitor their completion, and facilitate communication among team members.
- Weekly Progress Reports: Each team member was required to submit weekly progress reports, which helped in tracking individual contributions and overall project progress.
- Shared Documentation: The team maintained shared documentation for the project, including meeting minutes, project updates, and technical details. This ensured everyone was informed about project decisions and progress.
- Knowledge Sharing: Team members actively shared their expertise and insights during discussions and code reviews. This fostered a collaborative learning environment and ensured a comprehensive understanding of the project.
- Defined Roles and Responsibilities: Clear roles and responsibilities were assigned to each team member to avoid overlap and ensure accountability.

Consistency:

- Code Style and Structure: The team adhered to a consistent coding style and structure, making the codebase easier to read, understand, and maintain. This consistency was crucial for collaborative development and debugging.
- Naming Conventions: Consistent naming conventions were followed for variables, functions, and classes, improving code clarity and reducing ambiguity.
- Commenting and Documentation: The team committed to writing clear and concise comments throughout the code and maintaining comprehensive documentation for the project. This facilitated understanding and future maintenance of the project.

5.Conclusion

The project successfully developed a flight price predictor using machine learning techniques and data visualization. The model demonstrated promising performance in predicting airfare based on historical data, offering travelers a valuable tool for informed flight booking decisions.

Challenges Faced

Name	Failure/setback/problems	Solution
Fatemi Sadikbhai Lokhandwala	I am encountering an issue where the dataset is not being retrieved from the specified GitHub path. Despite ensuring that the path is correct, the system fails to access the data.	I have resolve the issue with the path to ensure the dynamic retrieval of the dataset from the GitHub URL. Verify the accuracy of the URL, check for any permissions or access restrictions, and ensure there are no connectivity problems. By addressing these factors, we can ensure that the dataset is successfully fetched from the GitHub repository.
Urjeet Parmar	The issue I was facing was scaling the data during prediction using interactive dashboard.	I pickled the scaler and pickled encoded columns that fixed the dashboard errors showing during development phase in flask and streamlit.
Swetha Tanikonda	The issue I was facing was with the PyCharm. I was not able to work on Python code in my PyCharm connected to the GitHub due to libraries not finding correct path.	It was solved by making a virtual environment
Gaurav Singh	Ram allocation for cost complexity pruning	It is on hold as we have shifted on live dataset and if it needs upgrade we have to see.
Devendra Singh Shekhawat	I am experiencing an issue with the accuracy of my model. When I use Recursive Feature Elimination (RFE) alone, I achieve an accuracy of 85%. However, when I combine RFE with Stats Models, the accuracy drops to around 82%. My goal is to improve the accuracy while using both methods together.	To improve accuracy, I will ensure that RFE's selected features align with Stats Models and fine-tune the hyperparameters. I will also use cross-validation and apply regularization to prevent overfitting. Additionally, I will experiment with different models within Stats Models to find the best results.

Mohamed Maaz Rehan	While scraping data, I encountered challenges with security checks designed to confirm that I'm not a robot. These checks disrupted the automated task, causing interruptions and preventing access to certain URLs.	To resolve this, I avoided URLs with security check questions and implemented a delay time between each URL request. This approach successfully mitigated the issue, allowing the scraping process to continue smoothly without triggering the security checks.

Future Work:

- **Model Enhancement:** Further refine the model by exploring more advanced algorithms and feature engineering techniques.
- **Real-Time Predictions:** Integrate real-time data feeds from flight booking platforms to provide up-to-date price predictions.
- **User Interface Development:** Develop a user-friendly web interface for easier access to flight price predictions.

6.References

(n.d.). Retrieved from https://www.webscraper.io/tutorials
(n.d.). Retrieved from https://rapidapi.com/ntd119/api/sky-scanner3
(n.d.). Retrieved from https://www.skyscanner.ca/
(n.d.). Retrieved from https://www.ca.kayak.com/
(n.d.). Retrieved from https://scikit-learn.org/stable/auto_examples/tree/plot_cost_complexity_pruning.html
(n.d.). Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.ElasticNet.html
(n.d.). Retrieved from https://streamlit.io/
(n.d.). Retrieved from https://docs.python.org/3/library/pickle.html
(n.d.). Retrieved from https://joblib.readthedocs.io/en/stable/
(n.d.). Retrieved from https://machinelearningmastery.com/rfe-feature-selection-in-python/