Forecast Commuters Inflow For Airline Industry Using Prophet Model

1. INTRODUCTION :

1.1.Overview :

Air passenger traffic forecast is of great importance for airlines and civil aviation authorities. For airlines, accurate forecasts play an increasingly important role in revenue management. It helps to reduce the airlines' risk by objectively evaluating the demand of the air transportation business. For civil aviation authorities, air passenger traffic forecast provides a concrete basis for planning decisions in air transport infrastructure. The main objective of this project is to build a prophet time series model that forecasts the passenger traffic for a given date

1.2.Purpose:

- The project aim is to You'll be able to know the fundamental concepts of time series forecasting.
- Working with Prophet library
- Flask Application Development

2. LITERATURE SURVEY:

a.Existing problem:

The next step after determining what elements should be forecast is to collect and review previous forecasts developed for the airport. The latest FAA Terminal Area Forecast (TAF) for the airport should be obtained. The TAF is updated annually in December and is available at *http://www.apo.data.faa.gov*. If appropriate, regional planning authorities and/or State aviation authorities should be contacted to determine whether they sponsored forecasts of air transportation demand that included the airport. Review of forecasts can provide important information about the previous economic outlook and air transportation demand projections. In addition, the reviews can be used to obtain historic data relevant to the current forecasting effort. Previous projections of aviation activity need to be assessed to determine if they are out of date.

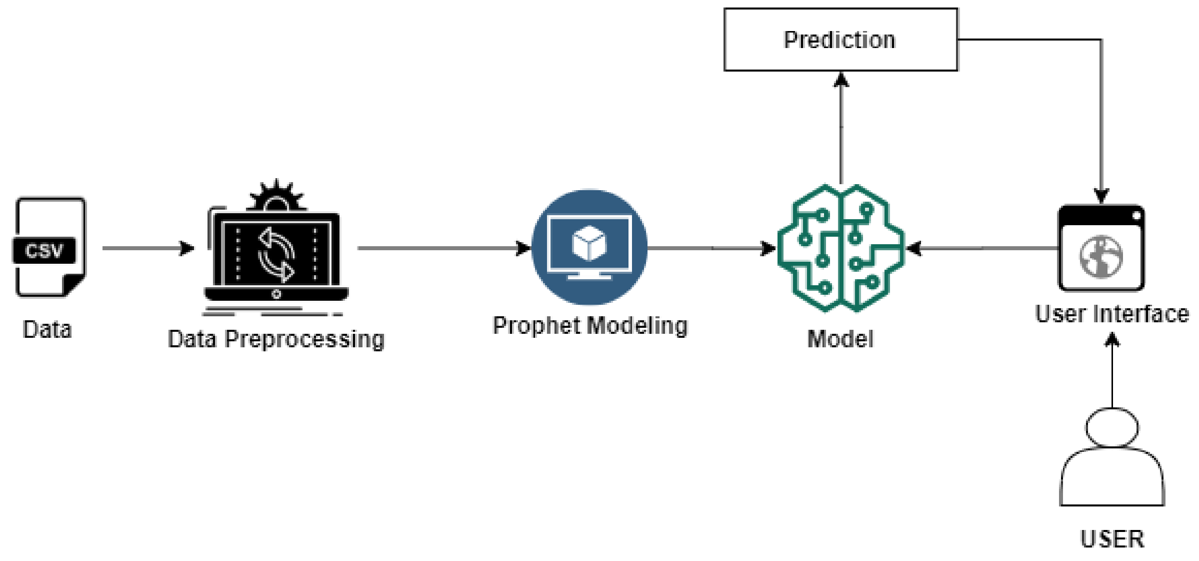
b.Proposed solution:

The next step in preparing an airport forecast is to select appropriate methods to develop the forecast. While there are several acceptable techniques and procedures for forecasting aviation activity at a specific airport, most forecasts utilize basic techniques such as regression or share analysis. The following discussion is an overview of the majority of forecast methods that apply to aviation demand forecasting. The discussion is not all-inclusive, and additional techniques might be acceptable. A forecast effort could involve a number of different techniques.

3. **THEORITICAL ANALYSIS:**

***3.1.Block Diagram*:**

Architecture:



2.Hardware / Software designing :

Hardware Requirements:

Operating System	Windows, Mac, Linux
CPU (for training)	Multi Core Processors (i3 or above/equivalent)
GPU (for training)	NVIDIA AI Capable / Google's TPU

Software Requirements:

Python	v3.10.0 or Above
Python Packages	flask, tensorflow, opencv-python, keras, numpy, pandas, scikit-learn.
Web Browser	Mozilla Firefox, Google Chrome or any modern web browser

2. EXPERIMENTAL INVESTIGATIONS:

The purpose of the present research is estimating the potential traffic for SIA (Sibiu International Airport, SBZ) which is a regional airport in Romania, for the year 2017. For achieving this, the following SIA internal secondary data was used: (i) monthly number of aircraft departures for every route managed by SIA indifferently of the airline company, and (ii) monthly traffic (departures, arrivals, and transit) of SIA. Secondary data analysis was used as research method, and time series analysis and regression analysis as data analysis techniques. Based on the upper literature review of models (parametric and nonparametric) used for predicting passenger traffic, we propose a mixed-model which uses time-series predictions as input for multiple regression analysis.

3. FLOWCHART:

Project Flow

We will be building a Web application where

- The user selects the date from User Interface(UI)
- The passenger traffic for the selected date is analysed by the model
- The count of passengers for the selected date is displayed on UI

To accomplish this, complete all the milestones & activities listed below.

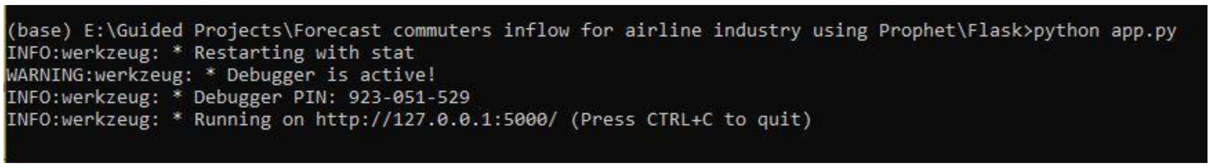
- Installation of Pre-requisites.
  - Installation of Anaconda IDE / Anaconda Navigator.
  - Installation of Python packages.
- Data Collection.
  - Create or Collect the dataset.
- Data Pre-processing.
  - Importing of Libraries.
  - Importing of Dataset & Visualisation.
- Model Building.
  - Fitting the prophet library.
  - Cross validation of the model.
  - Evaluation of the model.

    - Save the model.
- Application Development.

4. **RESULT:**

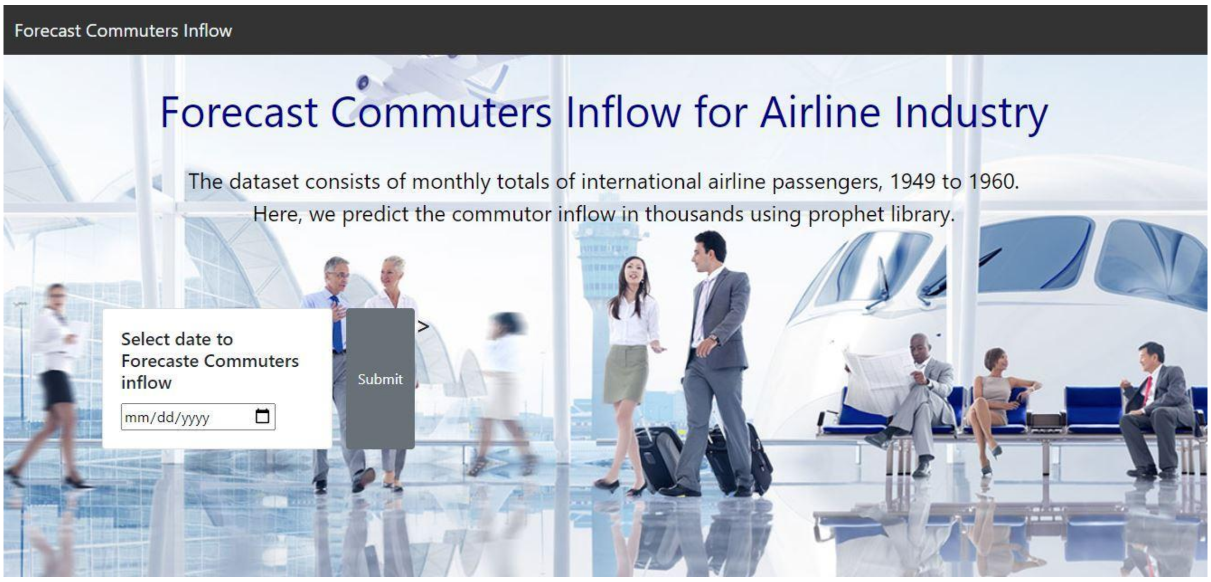
Open anaconda prompt from the start menu.

Navigate to the folder where your app.py resides.
Now type "python app.py" command.
It will show the local host where your app is running on http://127.0.0.1.5000/
Copy that local host URL and open that URL in browser. It does navigate you to the where you can view your web page.



```
(base) E:\Guided Projects\Forecast commuters inflow for airline industry using Prophet\Flask>python app.py
INFO:werkzeug: * Restarting with stat
WARNING:werkzeug: * Debugger is active!
INFO:werkzeug: * Debugger PIN: 923-051-529
INFO:werkzeug: * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Your UI will look like

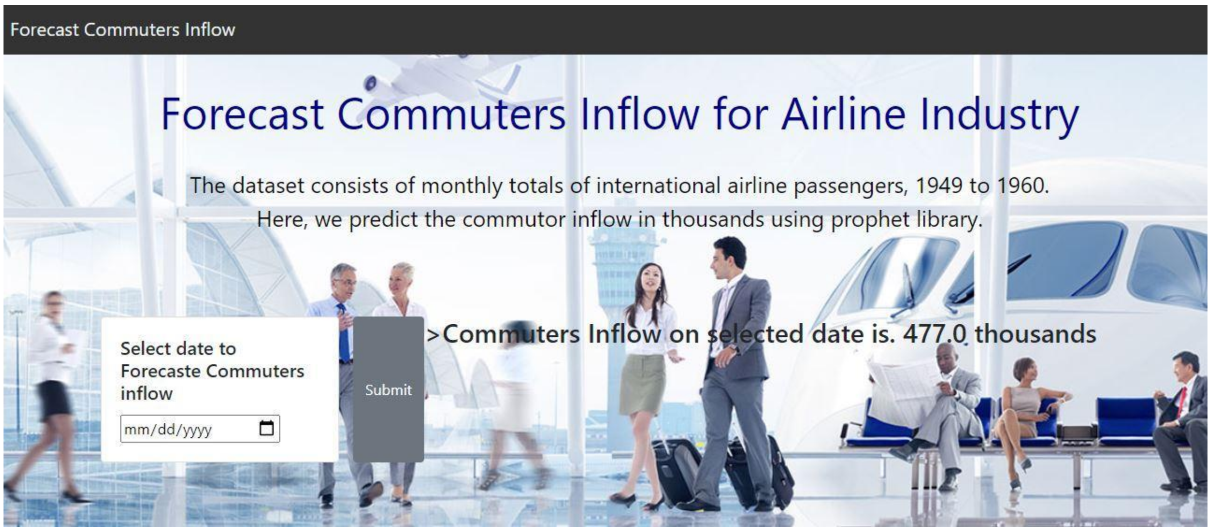


Forecast Commuters Inflow

Forecast Commuters Inflow for Airline Industry

The dataset consists of monthly totals of international airline passengers, 1949 to 1960.
Here, we predict the commutor inflow in thousands using prophet library.

Select the date you would like to predict and click on submit.The output prediction will be like.


Forecast Commuters Inflow

Forecast Commuters Inflow for Airline Industry

The dataset consists of monthly totals of international airline passengers, 1949 to 1960.
Here, we predict the commutor inflow in thousands using prophet library.

Select date to Forecaste Commuters inflow

mm/dd/yyyy

Submit

>Commuters Inflow on selected date is. 477.0 thousands

Check the predictions for the different inputs.With this you have successfully completed the project.

ADVANTAGES AND DISADVANTAGES:

ADVANTAGES:

- It accurately predict the future demand
- it gives the ability to make informed business decisions and develop data-driven strategies.

DISADVANTAGES:

- It is more expensive
- It has capacity limits.
- It is more polluting

APPLICATIONS:

Identifying the proper regression model for traffic estimation based on the number of aircraft departures, and Forecasting the number of aircraft departures for the current routes operated SIA.

CONCLUSION AND FUTURE SCOPE:

The next step in the forecast process is to summarize and document the results. The planning forecast write-up should summarize each forecast element, explain the forecast methods used, highlight significant assumptions, clearly present the forecast results, and provide a brief evaluation of the forecast.

Tables of historical and forecast data should be included for each forecast element, and graphs of key time series and forecasts are suggested. Explanations should be provided if major changes from historic trends are expected in the future. For example, if a new low-fare carrier were forecast at an airport resulting in a high enplanement forecast, special documentation of the carrier's commitment would be necessary.

In order to summarize and document the airport planning forecast, FAA recommends that the preparer of the forecast complete the template provided in Appendix B. This template will help clarify the presentation of the forecast levels for each component, the forecast growth rates, and the operational factors used to derive the forecast levels. Submittal of the completed template to FAA will facilitate the review and approval of proposed forecasts. An Excel worksheet of this template is available at the FAA APO website (*www.apo.data.faa.gov*).

BIBILOGRAPHY:

Ashford, Norman and Messaoud Benchemam. "Passengers' Choice of Airport: An Application of the Multinomial Logit Model," in Air Transportation Issues. Transportation Research Record 1147. Washington, DC: Transportation Research Board, 1987, p. 1-5.

Bowles, Robert and Gene Mercer. Deregulation and its Effect on FAA Methods for Forecasting Commercial Air Carrier Demand. Washington, DC: Federal Aviation Administration, September 1989.

Caves, Robert E. "Forecasting Traffic at Smaller Airports in a Free Market Environment," in Airport Planning, Operation and Management. Transportation Research Record 1423. Washington, DC: Transportation Research Board, 1993, p. 1-7.

Corsi, Thomas, Martin Dresner and Robert Windle. "Air Passenger Forecasts: Principles and Practices," Journal of the Transportation Research Forum, 36(2): 42-62, 1997.

Diebold, Frances X. Elements of Forecasting. South-Western College Publishing, Cincinnati, Ohio, 1998.

Federal Aviation Administration. Airport Master Plans. Advisory Circular 150/5070-6A. Washington, DC, 1985.

APPENDIX:

Import Packages And Load Data

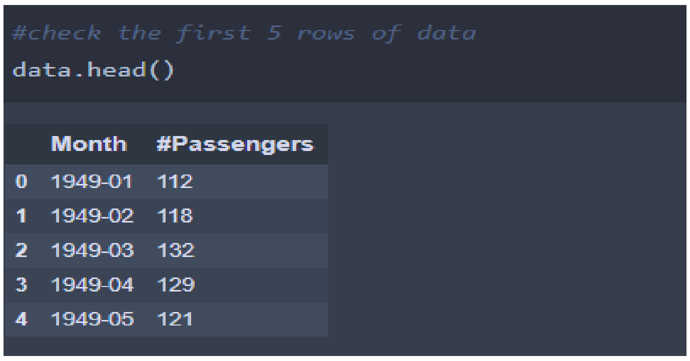
Start by importing the libraries you will need at the top of your notebook .

```
# Importing of Libraries
#import numpy
import numpy as np
#import pandas
import pandas as pd
#import visualization library
import matplotlib.pyplot as plt
```

Import the dataset, using the below command

```
# Importing & reading of dataset
data=pd.read_csv('air_passenger.csv')
```

- List the first five row of the dataset using head function.



```
#check the first 5 rows of data
data.head()
```

	Month	#Passengers
0	1949-01	112
1	1949-02	118
2	1949-03	132
3	1949-04	129
4	1949-05	121

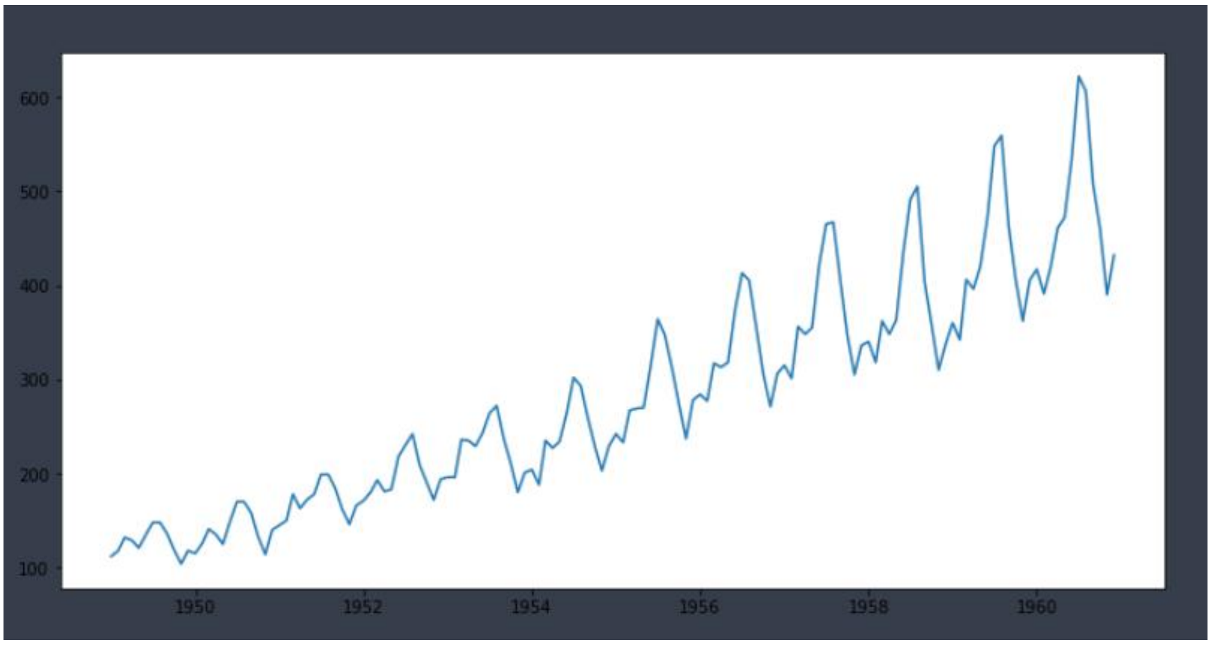
```
df.dtypes
```

```
Output
Month              object
AirPassengers       int64
dtype: object
```

```
# Date Coversion
data['Month']=pd.to_datetime(data['Month'], format='%Y-%m')
```

```
#import datetime conversion
from pandas import to_datetime
#prepare expected column names
data.columns = ['ds', 'y']
data['ds']= to_datetime(data['ds'])
```

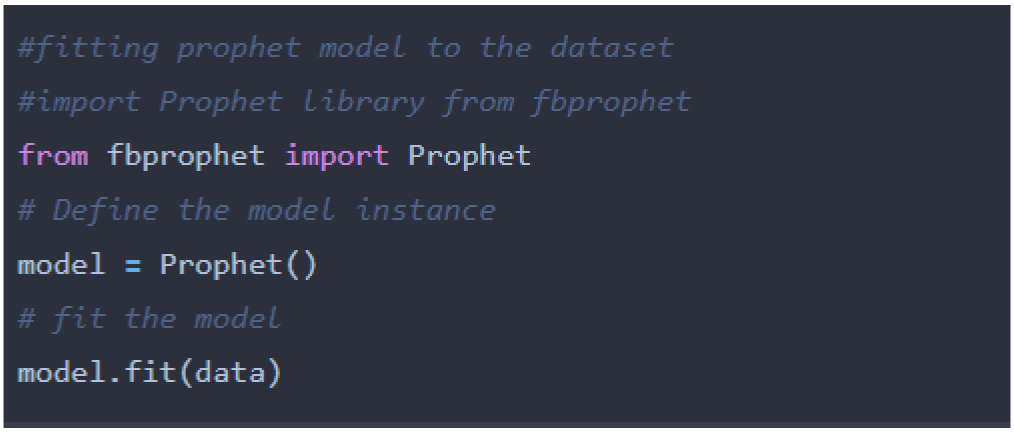
```
#visualizing the ds column
#configure the figure size
plt.figure(figsize=(12,6))
plt.plot(data.set_index(['ds']))
```



With our data now prepared, we are ready to use the Prophet library to produce forecasts of our time series.

Model Building

Fit The Model

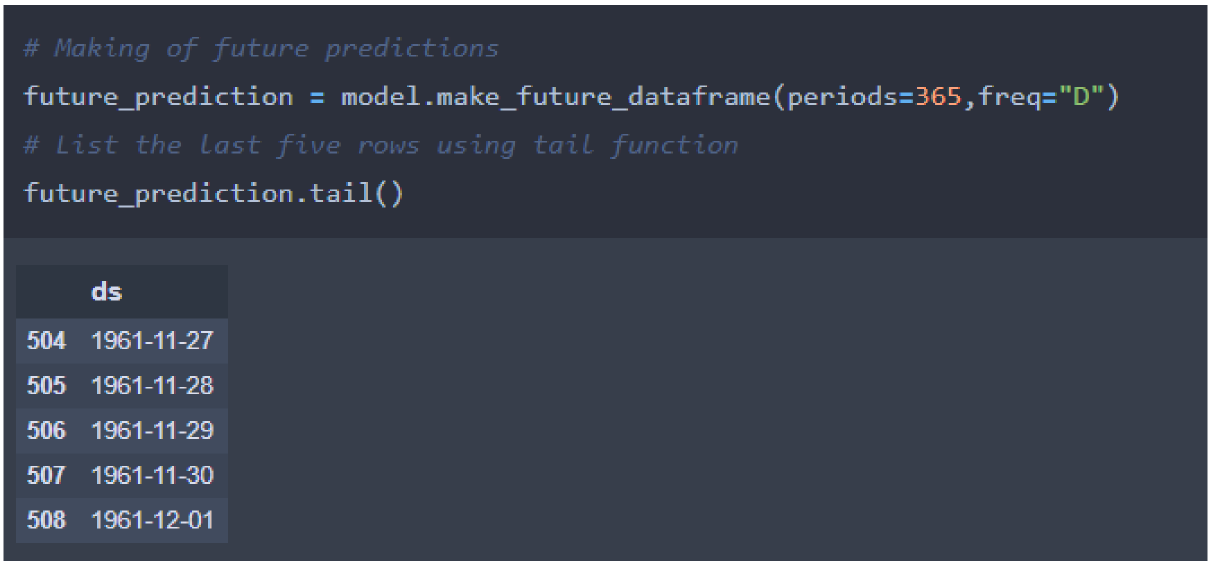


```
#fitting prophet model to the dataset
#import Prophet library from fbprophet
from fbprophet import Prophet
# Define the model instance
model = Prophet()
# fit the model
model.fit(data)
```

Note: It will take few minutes to fit the model.

Making Future Predictions

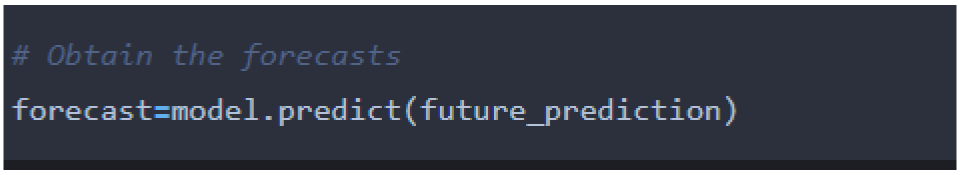
This is achieved using the Prophet.make_future_dataframe method and passing the number of days we'd like to predict in the future.



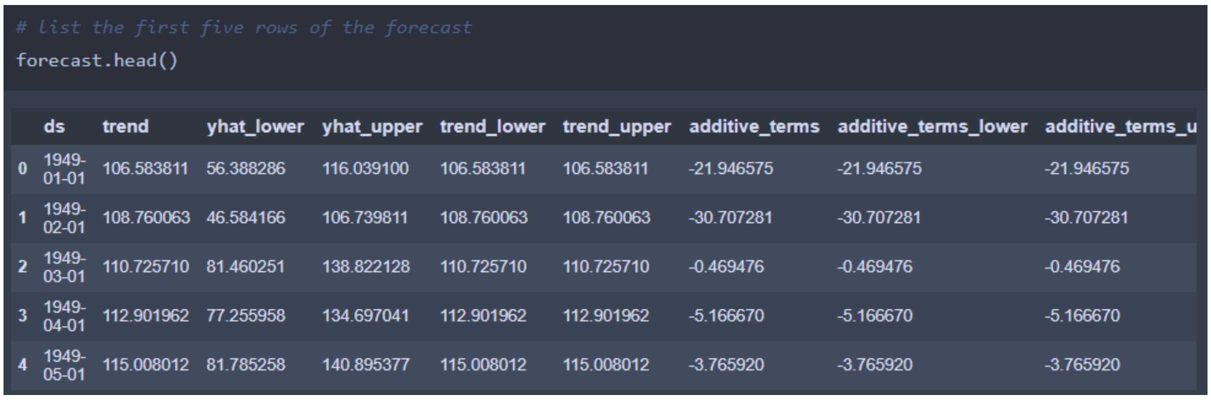
```
# Making of future predictions
future_prediction = model.make_future_dataframe(periods=365,freq="D")
# List the last five rows using tail function
future_prediction.tail()
```

	ds
504	1961-11-27
505	1961-11-28
506	1961-11-29
507	1961-11-30
508	1961-12-01

Obtaining the Forecasts



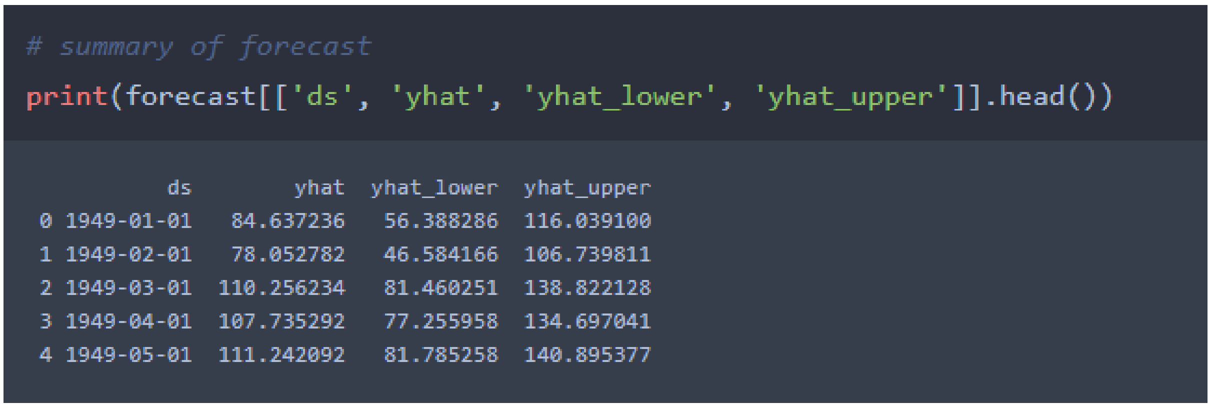
```
# Obtain the forecasts
forecast=model.predict(future_prediction)
```



```
# list the first five rows of the forecast
forecast.head()
```

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_terms_lower	additive_terms_u
0	1949-01-01	106.583811	56.388286	116.039100	106.583811	106.583811	-21.946575	-21.946575	-21.946575
1	1949-02-01	108.760063	46.584166	106.739811	108.760063	108.760063	-30.707281	-30.707281	-30.707281
2	1949-03-01	110.725710	81.460251	138.822128	110.725710	110.725710	-0.469476	-0.469476	-0.469476
3	1949-04-01	112.901962	77.255958	134.697041	112.901962	112.901962	-5.166670	-5.166670	-5.166670
4	1949-05-01	115.008012	81.785258	140.895377	115.008012	115.008012	-3.765920	-3.765920	-3.765920

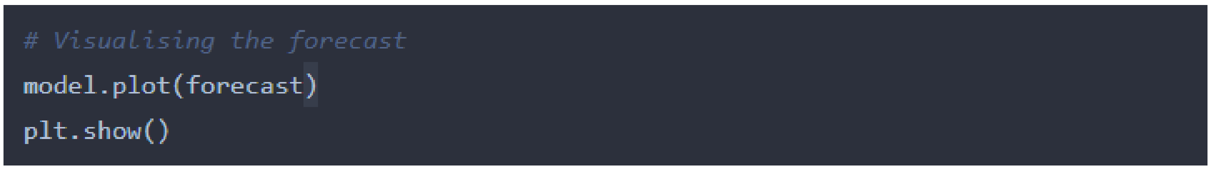
- Get the summary of the forecast using the below command.



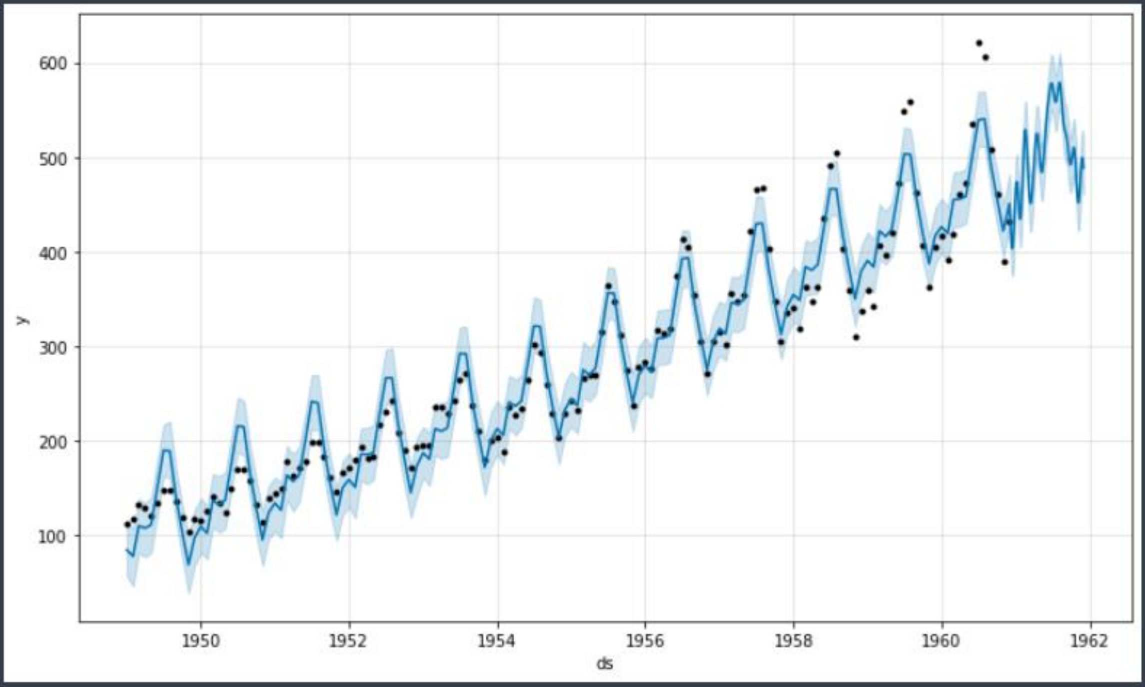
```
# summary of forecast
print(forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].head())
```

```
          ds        yhat  yhat_lower  yhat_upper
0 1949-01-01   84.637236   56.388286  116.039100
1 1949-02-01   78.052782   46.584166  106.739811
2 1949-03-01  110.256234   81.460251  138.822128
3 1949-04-01  107.735292   77.255958  134.697041
4 1949-05-01  111.242092   81.785258  140.895377
```

- Now, let’s visualize the forecast using the below commands.



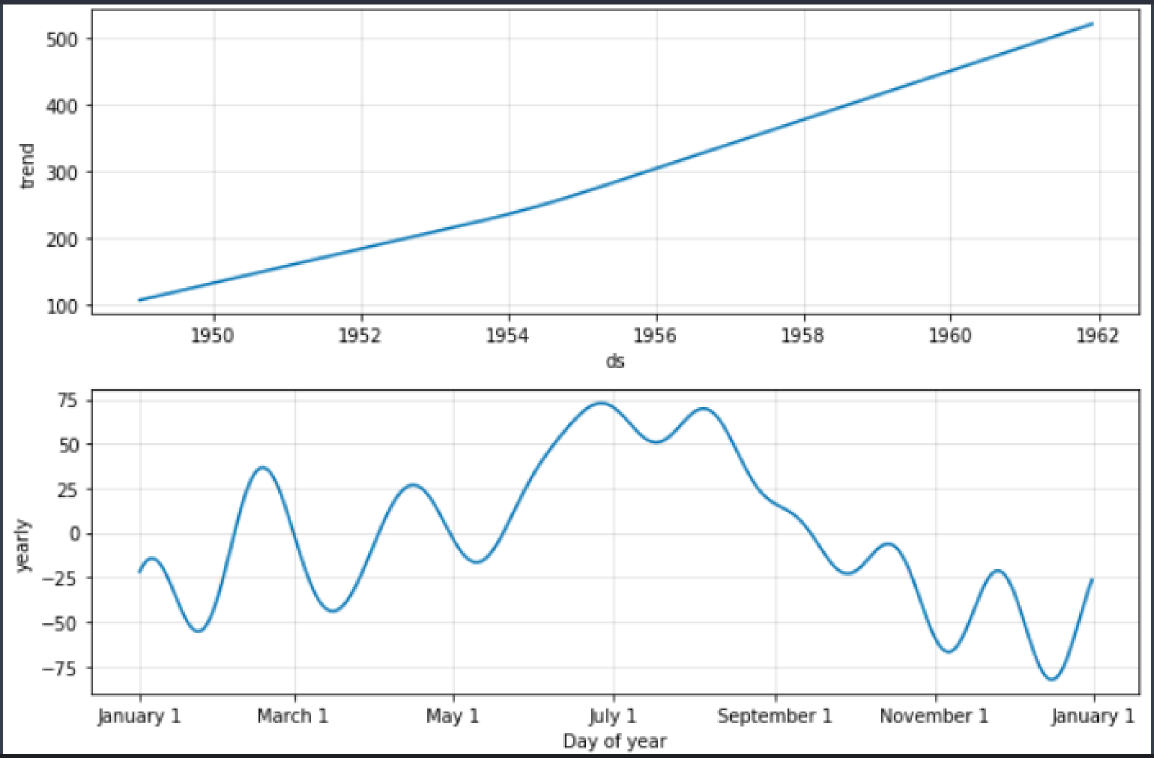
```
# Visualising the forecast
model.plot(forecast)
plt.show()
```



- Visualise the components using plot components

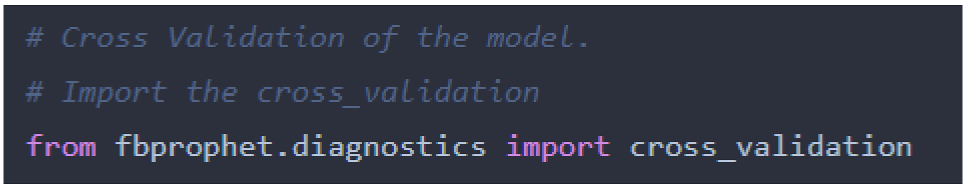
```
# Visualise the plot components
model.plot_components(forecast)
```

- The output visualisation will be like



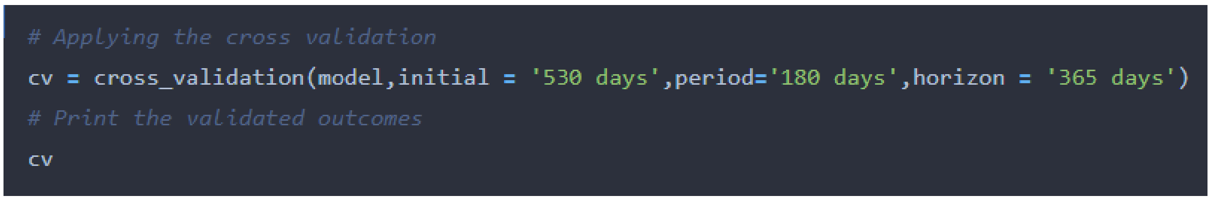
Cross Validation of the model

.



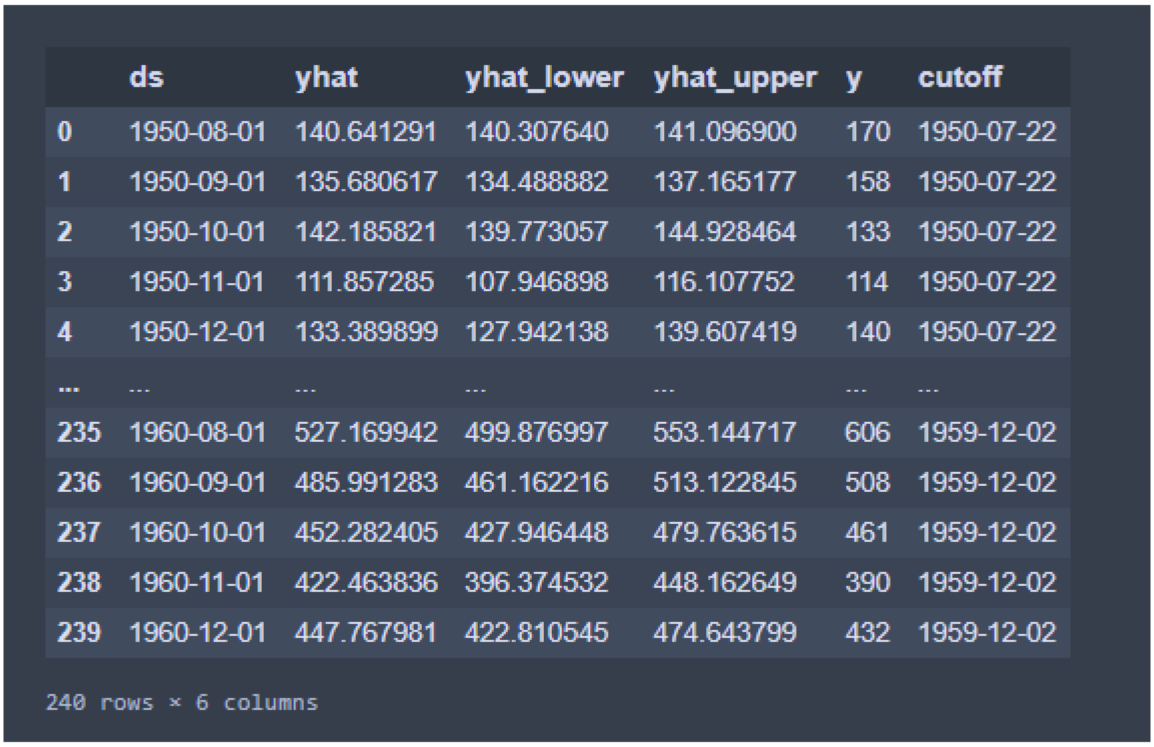
```python
# Cross Validation of the model.
# Import the cross_validation
from fbprophet.diagnostics import cross_validation
```

Fitting the cross validation.



```python
# Applying the cross validation
cv = cross_validation(model,initial = '530 days',period='180 days',horizon = '365 days')
# Print the validated outcomes
cv
```

Outcomes of the cross validation.

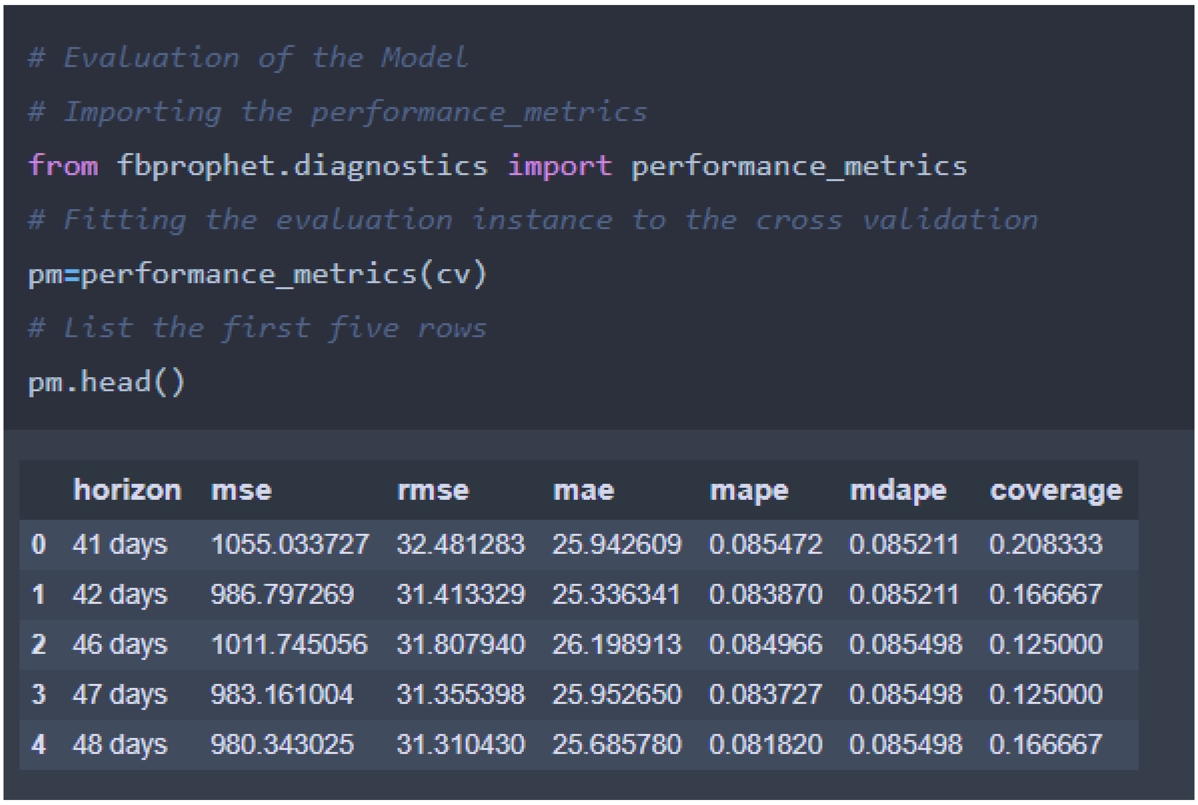


	ds	yhat	yhat_lower	yhat_upper	y	cutoff
0	1950-08-01	140.641291	140.307640	141.096900	170	1950-07-22
1	1950-09-01	135.680617	134.488882	137.165177	158	1950-07-22
2	1950-10-01	142.185821	139.773057	144.928464	133	1950-07-22
3	1950-11-01	111.857285	107.946898	116.107752	114	1950-07-22
4	1950-12-01	133.389899	127.942138	139.607419	140	1950-07-22
...	...	...	...	...	...	...
235	1960-08-01	527.169942	499.876997	553.144717	606	1959-12-02
236	1960-09-01	485.991283	461.162216	513.122845	508	1959-12-02
237	1960-10-01	452.282405	427.946448	479.763615	461	1959-12-02
238	1960-11-01	422.463836	396.374532	448.162649	390	1959-12-02
239	1960-12-01	447.767981	422.810545	474.643799	432	1959-12-02

```
240 rows × 6 columns
```

Evaluation Of The Model

- Import the evaluation model libraries and fit it to the cross validated outcomes.
- The Outcomes of the evaluation will be like.

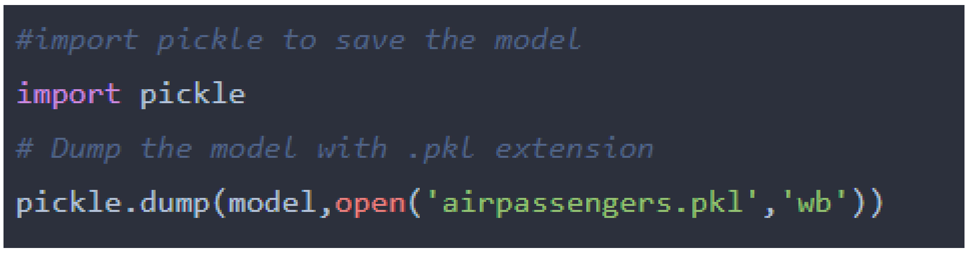


```python
# Evaluation of the Model
# Importing the performance_metrics
from fbprophet.diagnostics import performance_metrics
# Fitting the evaluation instance to the cross validation
pm=performance_metrics(cv)
# List the first five rows
pm.head()
```

```
   horizon  mse          rmse       mae        mape      mdape     coverage
0  41 days  1055.033727  32.481283  25.942609  0.085472  0.085211  0.208333
1  42 days  986.797269   31.413329  25.336341  0.083870  0.085211  0.166667
2  46 days  1011.745056  31.807940  26.198913  0.084966  0.085498  0.125000
3  47 days  983.161004   31.355398  25.952650  0.083727  0.085498  0.125000
4  48 days  980.343025   31.310430  25.685780  0.081820  0.085498  0.166667
```

Save The Model.

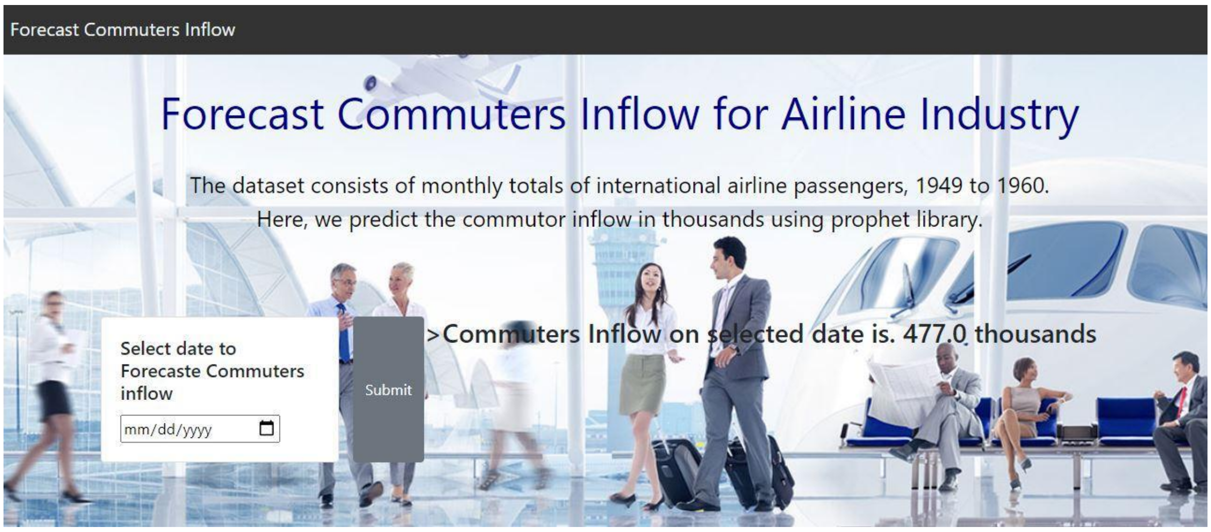
- Follow the commands to save your model.



```python
#import pickle to save the model
import pickle
# Dump the model with .pkl extension
pickle.dump(model,open('airpassengers.pkl','wb'))
```

Application Building

Build HTML Code


Forecast Commuters Inflow

Forecast Commuters Inflow for Airline Industry

The dataset consists of monthly totals of international airline passengers, 1949 to 1960.
Here, we predict the commutor inflow in thousands using prophet library.

Select date to Forecaste Commuters inflow

mm/dd/yyyy

Submit

>Commuters Inflow on selected date is. 477.0 thousands

Build Python Code

Import the following libraries

```
# Importing of libraries
import numpy as np
import pandas as pd
from flask import Flask, request, jsonify, render_template
import pickle
```

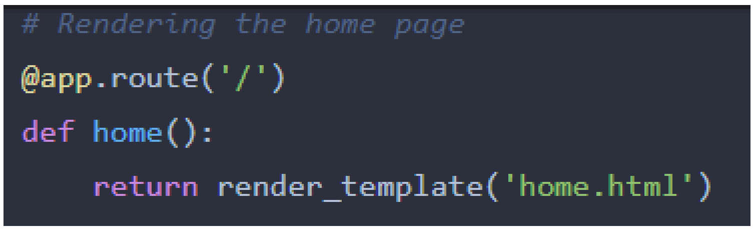
Rendering to html page

```
# Defining the app
app = Flask(__name__)
```

Loading the saved model using pickle library.

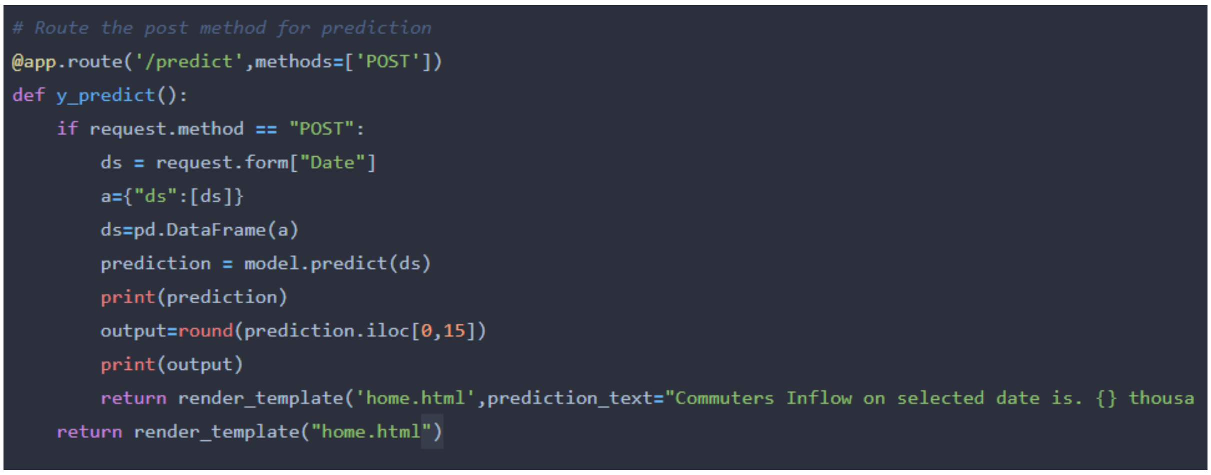
```
# Loading the saved model
model = pickle.load(open('airpassengers.pkl', 'rb'))
```

Rendering of home page html.



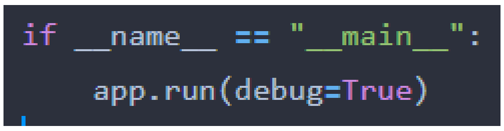
```python
# Rendering the home page
@app.route('/')
def home():
    return render_template('home.html')
```

Routing the prediction to the home page.



```python
# Route the post method for prediction
@app.route('/predict',methods=['POST'])
def y_predict():
    if request.method == "POST":
        ds = request.form["Date"]
        a={"ds":[ds]}
        ds=pd.DataFrame(a)
        prediction = model.predict(ds)
        print(prediction)
        output=round(prediction.iloc[0,15])
        print(output)
        return render_template('home.html',prediction_text="Commuters Inflow on selected date is. {} thousa
    return render_template("home.html")
```

Calling of Main Function



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