(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Time Series Forecasting

A Report for the Evaluation 4 of Project 2

Submitted by

Shantanu Das

(18032030045)

in partial fulfilment for the award of the degree

of

Masters

IN

COMPUTER SCIENCE

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Under the Supervision of

Dr. Arvind Kumar, M.Tech., Ph.D.,

Professor

APRIL / MAY- 2020

SCHOOL OF COMPUTING AND SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report “Time Series Forecasting” is the bonafide work of “Shatanu Das (18032030045)” who carried out the project work under my supervision.

SIGNATURE OF HEAD
Ms.NILANJANA PRADHAN,

Associate Professor
School of Computer Science & Engineering

SIGNATURE OF SUPERVISOR.
Dr. ARVIND KUMAR

M.Tech,Ph.D.
Associate Professor
School of Computer Science & Engineering

TABLE OF CONTENTS

Abstract

Time Series Forecasting finds a lot of applications in many branches of industry or business. It allows to predict product demand (thus optimization production and warehouse storage), Forecast amount of money from sales or predict future values of stock prices .In this article I will try present basic approaches to achieves this goals .We will start with description of most popular models and them move to the models evaluation ,which indicates the best methods for given forecast problem. My research paper based of airline forecasting .In this we have to predict the possibility of route one place to another. Suppose we have to go one city to another city so we have to predict the demand of passenger on that route and chances of business for this we have to calculate the Qualitative Data and Quantitative Data . Using these method we can get the result . In this research paper we are using these method so we get the best future prediction of that business and demand of passenger and demand of aircraft. When demand is getting high then it will grow the GDP and GDP growth tells how many aircraft we need.

Introduction

Whenever data or observation something it record our regular time intervals you looking at time series data. Time series data is looking at over time to forecast the predict what will happen in the next period based on pattern on reoccurring on previous period. History often repeat itself so whatever happens in the past it happens in the future. Most commonly example of time series forecasting is SEASONAL SALES REVENUE is year holiday sales revenue is goes up and during off season sales revenue is goes down. This is not hard to predict we can almost expect what was to get sales in holiday season but we could also get other time based transaction in our data set in consistence upward trend or improvement in your company performance all the time or downward trend when company is consistence falling each year . What makes is different from other types of predicted models is that the prediction is based on the given time looking at the sequence of the observation allt the time. Time series forecast gives us a better estimate of figure that how much this figure/ trend continue.

Example- suppose we have a sensor wires on the road and it will predict how many cars intersect in every twenty minutes and help of this we find the possible ways of traffic so it could help other driver to stop stuck in the traffic.

What happens recently is more useful to guiding us that what will happens next. Than we look at back all of the history than it shows up anything happen.

Forecasting Techniques:

- Simple Moving Average
- Exponential Smoothing
- Autoregressive Integration Moving Average
- Neural Network
- Croston

"[4]Time Series Forecasting is use to predict the future" before Forecasting we have to analyse the pre processed data. Like most forecasting methods didn't handle the missing value. We have to reconstructed the values. Data have to checked it contains the airline pattern since

There are forecast method that cannot handle the airline data. Suppose if data is seasonal than the required method need the frequency of data.

Existing System

In the present System , A customer has need a datasheet of various type of airline or other data which customer can use to find the predict data. Datasheet should be in a excel file or database file . On which two columns must be compulsory for datasheet TIME AND DATE,

Because these column can predict the result .Without these column we can’t predict the result.

- Work should be manually.
- For result we have to Enter the file . This is a manually process.
- Datasheet has time and date column.
- Datasheet should be in excel or database file. It dose not consider any other extension file table.

PROPOSED SYSTEM

The proposed system is a web or desktop application and maintains a centralized repository of all related information. The system need to have python and respected API Example: pandas , matplotlib , . Users can decide what king of datasheet he has to enter in the application.

arima.py - Notepad

File Edit Format View Help

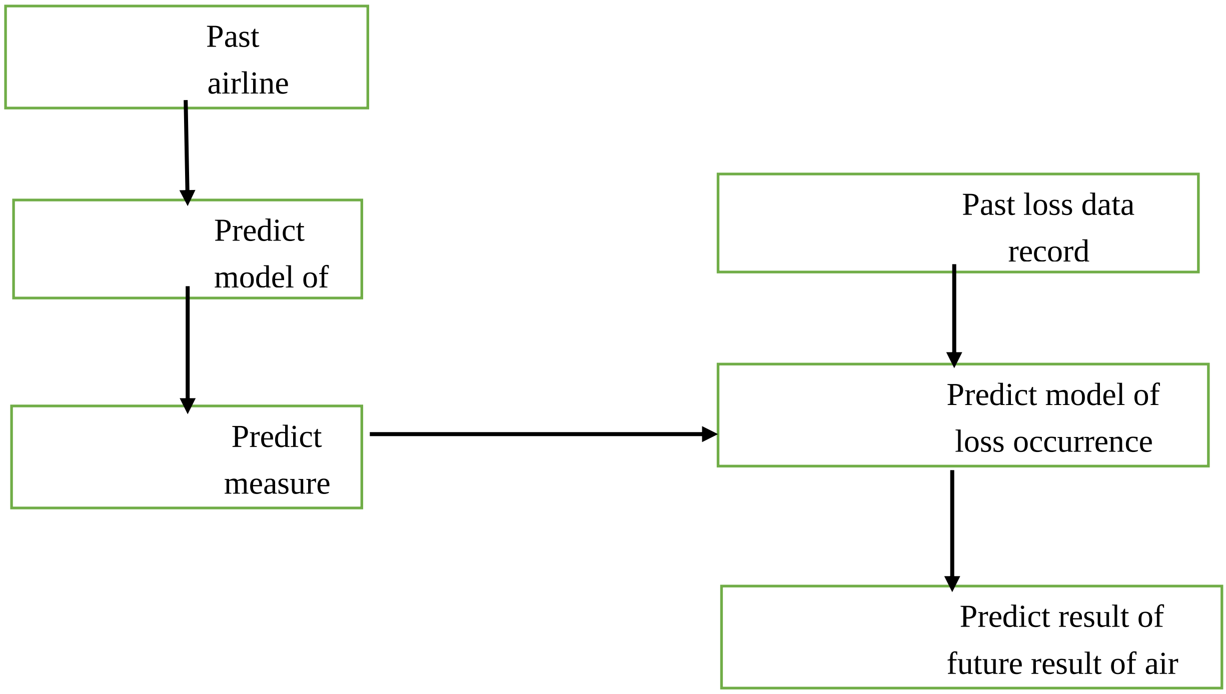
```
from pandas import datetime
from pandas import DataFrame
from statsmodels.tsa.arima_model import ARIMA
from matplotlib import pyplot

def parser(x):
	return datetime.strptime('190'+x, '%Y-%m')

series = read_csv('audomcitypairs-201909.csv', header=0, parse_dates=[0], index_col=0, squeeze=True, date_parser=parser)
# fit model
model = ARIMA(series, order=(5,1,0))
model_fit = model.fit(disp=0)
print(model_fit.summary())
# plot residual errors
residuals = DataFrame(model_fit.resid)
residuals.plot()
pyplot.show()
residuals.plot(kind='kde')
pyplot.show()
print(residuals.describe())
```

The propose system is highly automated and makes the prediction correct at every time. The user can get the very right information at the very right time. Customer has to enter the file and row value which row customer wants the prediction.

Dataflow Diagram



Data Table

Table 1:

	date	births	wday	year	month	day_of_ye	day_of_m	day_of_week
1	1/1/2015	8068	Thu	2015	1	1	1	5
2	1/2/2015	10850	Fri	2015	1	2	2	6
3	1/3/2015	8328	Sat	2015	1	3	3	7
4	1/4/2015	7065	Sun	2015	1	4	4	1
5	1/5/2015	11892	Mon	2015	1	5	5	2
6	1/6/2015	12425	Tue	2015	1	6	6	3
7	1/7/2015	12141	Wed	2015	1	7	7	4
8	1/8/2015	12094	Thu	2015	1	8	8	5
9	1/9/2015	11868	Fri	2015	1	9	9	6
10	1/10/2015	8014	Sat	2015	1	10	10	7
11	1/11/2015	7172	Sun	2015	1	11	11	1
12	1/12/2015	11479	Mon	2015	1	12	12	2
13	1/13/2015	11924	Tue	2015	1	13	13	3
14	1/14/2015	12013	Wed	2015	1	14	14	4
15	1/15/2015	12339	Thu	2015	1	15	15	5
16	1/16/2015	11861	Fri	2015	1	16	16	6
17	1/17/2015	8280	Sat	2015	1	17	17	7
18	1/18/2015	7195	Sun	2015	1	18	18	1
19	1/19/2015	10602	Mon	2015	1	19	19	2
20	1/20/2015	12242	Tue	2015	1	20	20	3
21	1/21/2015	12086	Wed	2015	1	21	21	4
22	1/22/2015	11949	Thu	2015	1	22	22	5
23	1/23/2015	11910	Fri	2015	1	23	23	6

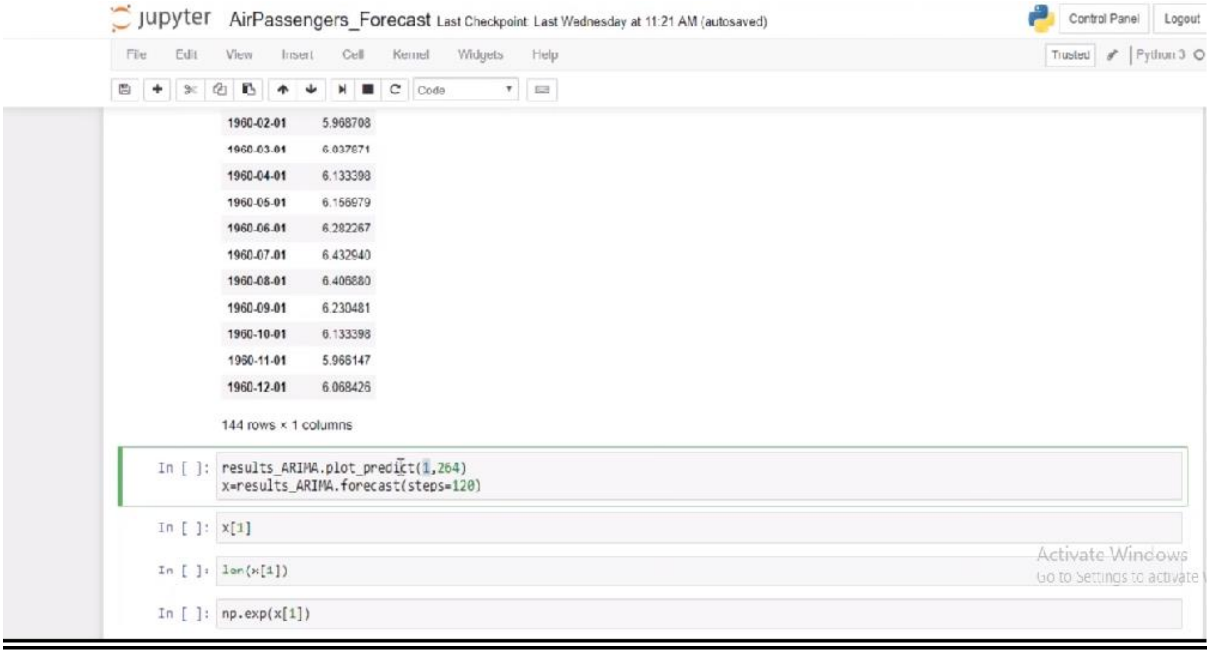
Table 2:

A1 City1

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	City1	City2	Month	Passenger	Aircraft_T	Passenger	Distance_	RPKs	ASKs	Seats	Year	Month_num	
2	ADELAIDE	ALICE SPR	30682	15743	143	81.8	1316	20717788	25327369	19246	1984	1	
3	ADELAIDE	BRISBANE	30682	3781	32	89.8	1622	6132782	6829379	4210	1984	1	
4	ADELAIDE	CANBERRA	30682	1339	12	94.7	972	1301508	1374348	1414	1984	1	
5	ADELAIDE	DARWIN	30682	3050	33	66.8	2619	7987950	11958009	4566	1984	1	
6	ADELAIDE	GOLD COA	30682	1596	16	88.5	1607	2564772	2898047	1803	1984	1	
7	ADELAIDE	MELBOUR	30682	50817	711	66.3	643	32675331	49284059	76647	1984	1	
8	ADELAIDE	PERTH	30682	18670	144	85.1	2120	39580400	46510458	21939	1984	1	
9	ADELAIDE	SYDNEY	30682	37401	358	80.8	1167	43646967	54018524	46288	1984	1	
10	ALBURY	SYDNEY	30682	11478	296	79.8	452	5188056	6501323	14383	1984	1	
11	ALICE SPR	DARWIN	30682	9035	89	71.2	1305	11790675	16559937	12690	1984	1	
12	ALICE SPR	SYDNEY	30682	2896	31	75.8	2022	5855712	7725214	3821	1984	1	
13	BRISBANE	CAIRNS	30682	1397	31	71.9	1391	1943227	2702680	1943	1984	1	
14	BRISBANE	CANBERRA	30682	593	8	72.7	956	566908	779791	816	1984	1	
15	BRISBANE	DARWIN	30682	2627	22	80.7	2852	7492204	9284020	3255	1984	1	
16	BRISBANE	MACKAY	30682	192	15	48.2	797	153024	317477	398	1984	1	
17	BRISBANE	MELBOUR	30682	31791	324	84.4	1381	43903371	52018212	37667	1984	1	
18	BRISBANE	PERTH	30682	566	4	88.2	3615	2046090	2319830	642	1984	1	
19	BRISBANE	PROSERPI	30682	7432	92	76.1	895	6651640	8740657	9766	1984	1	
20	BRISBANE	ROCKHAM	30682	21633	287	77.2	518	11205894	14515407	28022	1984	1	
21	BRISBANE	SYDNEY	30682	95027	990	73.7	753	71555331	97090001	128938	1984	1	

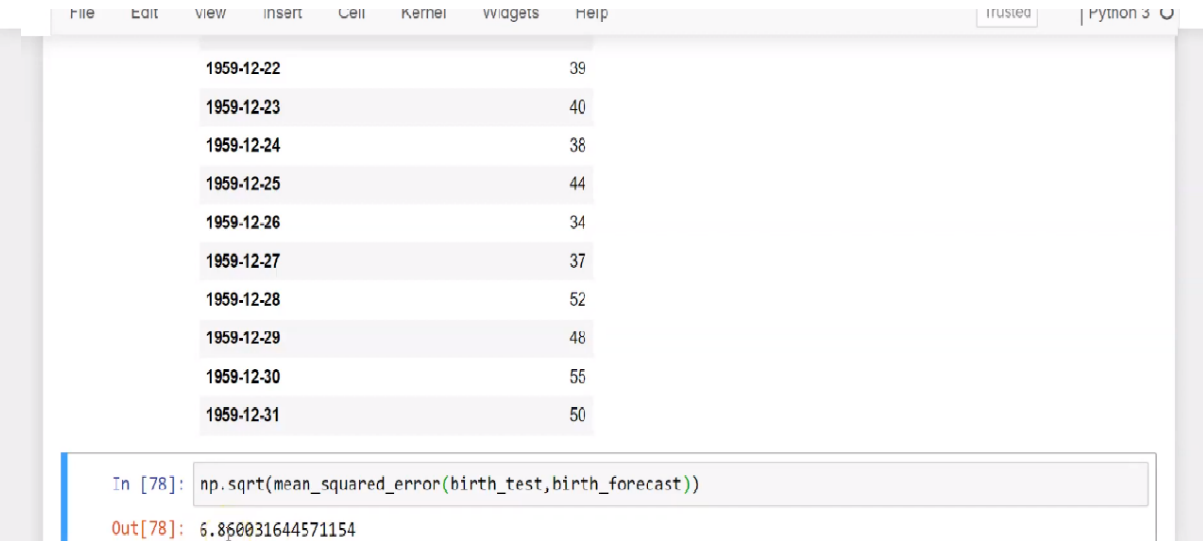
Input

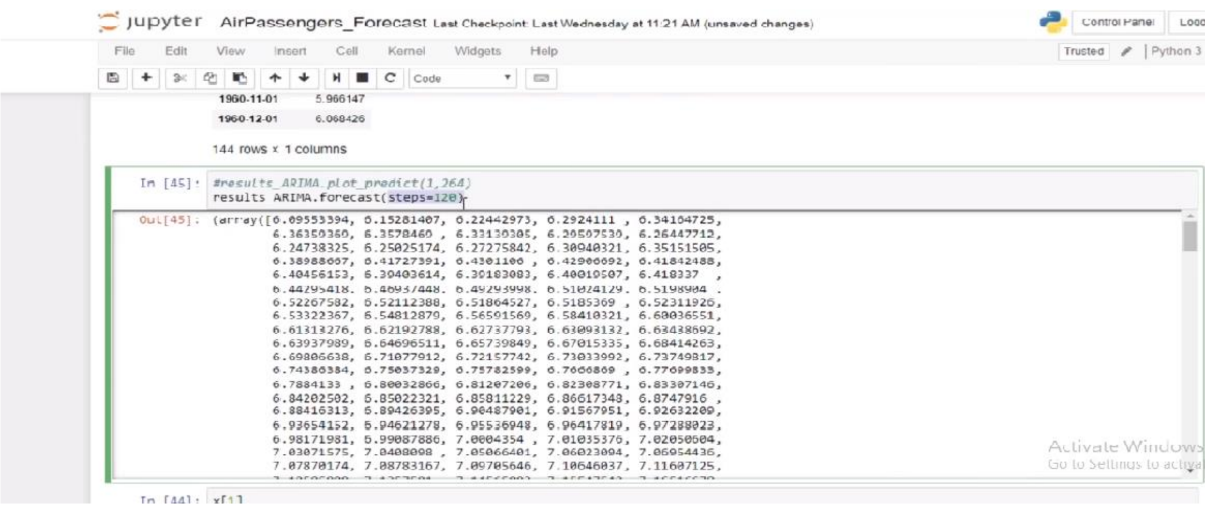
- File name(excel or database).
- Value
- Row   Value



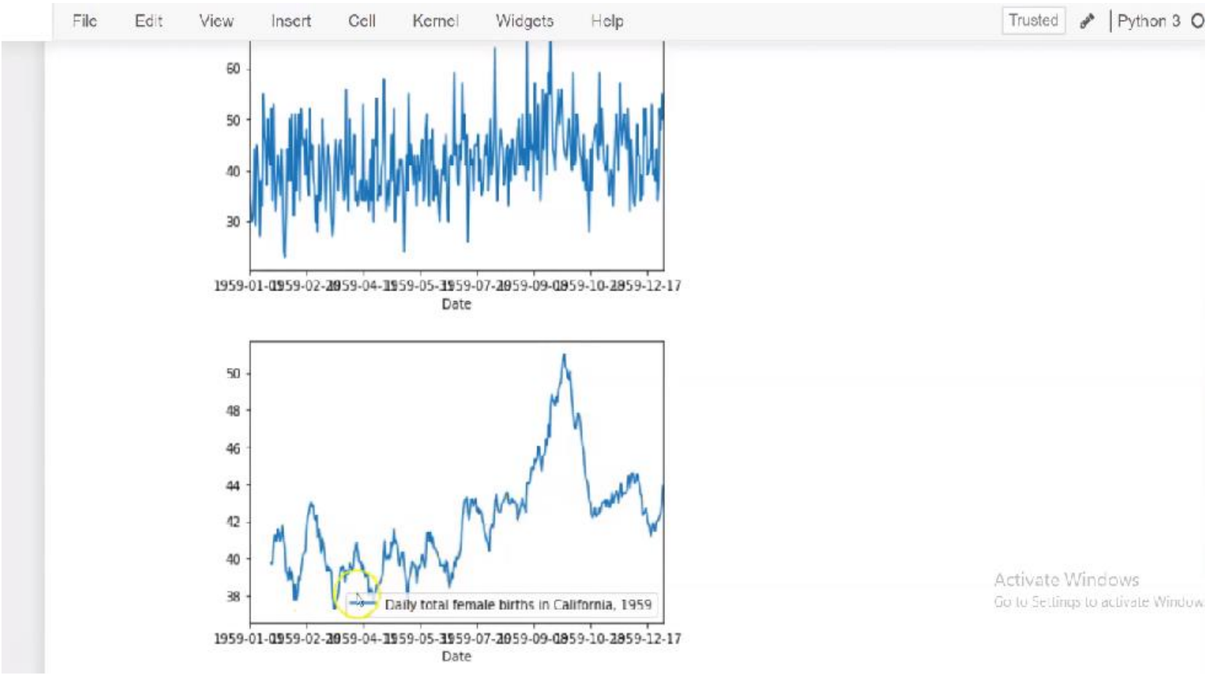
Output

- Output comes in graph format.
- Output comes in array format.
- It show the best predict value or increase and decrease of graph .

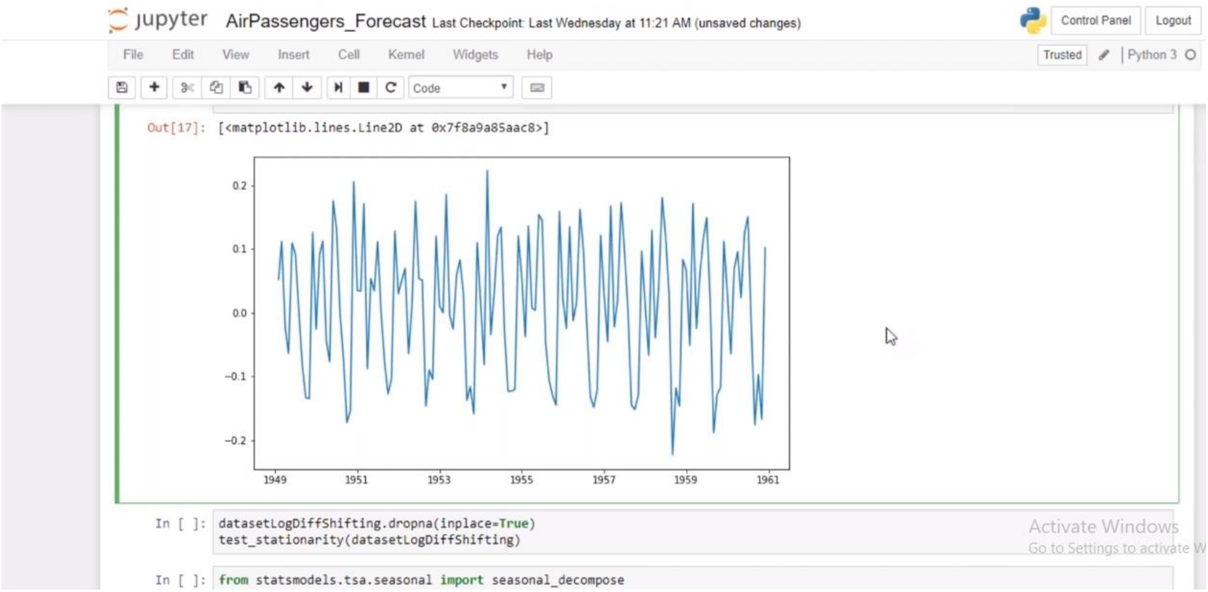




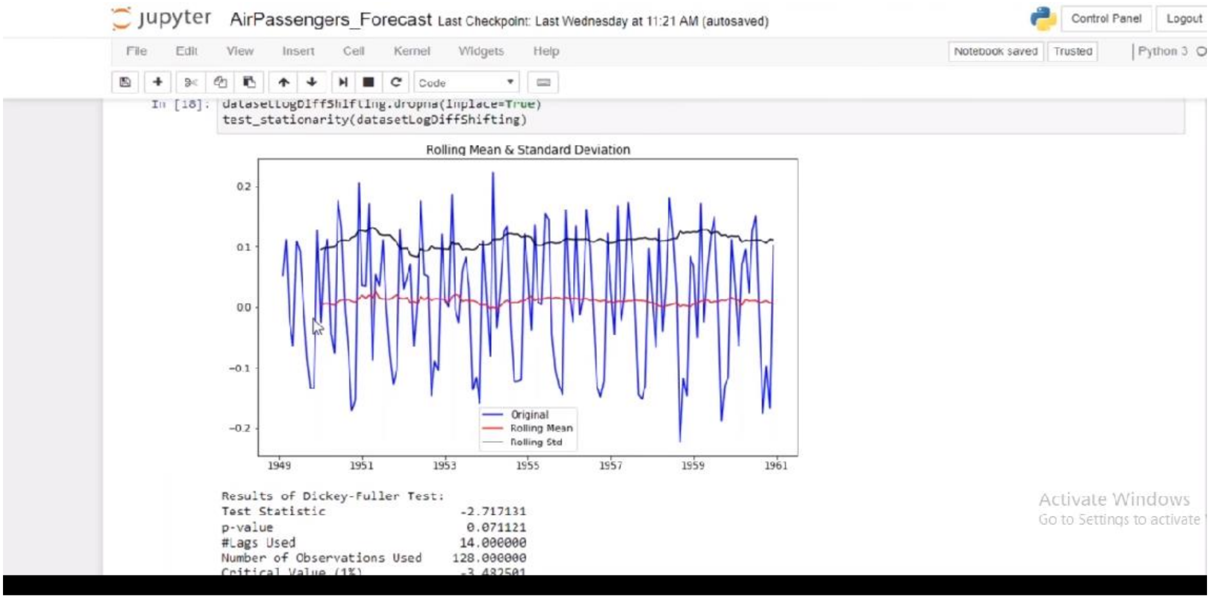
Screenshot



File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Date
Daily total female births in California, 1959
Date
Activate Windows



jupyter AirPassengers_Forecast Last Checkpoint: Last Wednesday at 11:21 AM (unsaved changes)
Control Panel
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Code
Out[17]: [<matplotlib.lines.Line2D at 0x7f8a9a85aac8>]
In [ ]: datasetLogDiffShifting.dropna(inplace=True)
test_stationarity(datasetLogDiffShifting)
In [ ]: from statsmodels.tsa.seasonal import seasonal_decompose
Activate Windows



second resuly.py - Notepad

File Edit Format View Help

```
for param in pdq:
    for param_seasonal in seasonal_pdq:
        try:
            mod = sm.tsa.statespace.SARIMAX(y,
                                            order=param,
                                            seasonal_order=param_seasonal,
                                            enforce_stationarity=False,
                                            enforce_invertibility=False)
results = mod.fit()
print('ARIMA{}x{}12 - AIC:{}'.format(param, param_seasonal, results.aic))
        except:
            continue
```

validation.py - Notepad

File Edit Format View Help

```
pred = results.get_prediction(start=pd.to_datetime('2017-01-01'), dynamic=False)
pred_ci = pred.conf_int()
ax = y['2014':].plot(label='observed')
pred.predicted_mean.plot(ax=ax, label='One-step ahead Forecast', alpha=.7, figsize=(14, 7))
ax.fill_between(pred_ci.index,
                pred_ci.iloc[:, 0],
                pred_ci.iloc[:, 1], color='k', alpha=.2)
ax.set_xlabel('Date')
ax.set_ylabel('Furniture Sales')
plt.legend()
plt.show()
```

FEASIBILITY ANALYSIS

A feasibility study is a preliminary study which investigates the information of prospective users and determines the resources requirements, costs, benefits and feasibility of proposed system. A feasibility study takes into account various constraints within which the system should be implemented and operated. In this stage, the resource needed for the implementation such as computing equipment, manpower and costs are estimated. The estimated are compared with available resources and a cost benefit analysis of the system is made.

- Technical Feasibility
- Operational Feasibility
- Economic Feasibility
- Schedule Feasibility

Technical Feasibility:

Evaluating the technical feasibility is the trickiest part of a feasibility study. This is because, at the point in time there is no any detailed designed of the system, making it difficult to access issues like performance, costs (on account of the kind of technology to be deployed) etc. A number of issues have to be considered while doing a technical analysis; understand the different technologies involved in the proposed system. Before commencing the project, we have to be very clear about what are the technologies that are to be required for the development of the new system.

Operational Feasibility:

Proposed project is beneficial only if it can be turned into information systems that will meet the operating requirements. Simply stated, this test of feasibility asks if the system will work when it is developed and installed. Are there major barriers to Implementation? The proposed was to make a simplified web application. It is simpler to operate and can be used in any webpages. It is free and not costly to operate.

Economic Feasibility:

Economic feasibility attempts to weigh the costs of developing and implementing a new system, against the benefits that would accrue from having the new system in place. This feasibility study gives the top management the economic justification for the new system. A simple economic analysis which gives the actual comparison of costs and benefits are much more meaningful in this case. In addition, this proves to be useful point of reference to compare actual costs as the project progresses

Schedule Feasibility:

A project will fail if it takes too long to be completed before it is useful. Typically, this means estimating how long the system will take to develop, and if it can be completed in a given period of time using some methods like payback period. Schedule feasibility is a measure how reasonable the project timetable is. Given our technical expertise, are the project deadlines reasonable? Some project is initiated 10 with specific deadlines. It is necessary to determine whether the deadlines are mandatory or desirable.

Conclusion

For time series-based forecasting methods were used to construct a passenger traffic forecast .They included two analytical methods: exponential smoothing and ARIMA, and the neural network method and Support Vector Machines (SVMR) approach involving machine learning and artificial intelligence. All methods used to compute a forecast of passenger flows. The forecast retains the characteristics of historical data, i.e. seasonal fluctuations, though its values do not rise substantially. The air services market in the region is stabilising while preserving its seasonal character. The forecasts obtained here are plausible and reliable, but one must note that the passenger traffic is linked to many factors, and the inclusion of the time factor alone is a considerable simplification. The air services market in the region is stabilising while preserving its seasonal character. The main factors determining the demand for transport include: the future amount of the GDP, the population of the country, the volume and value of foreign exchange, consumption levels, household spending structure, the rationalisation of a set of indicators concerning the use of specific means of transport, tendencies to alter travel and transport distances as a result of integration processes and the changes in the geography of manufacturing and settlement in the country. Although the forecasts clearly indicate growth tendencies of the phenomenon, they should be approached with caution. To a certain extent such forecasts enable the right decisions on future activities in the analysed area to be taken.

Refrence

[1] G.E.Box,G.Jenkins,andG.C.Reinsel,“Timeseriesanalysis,prediction and control,” 1970.” Andr´e Bauer, Marwin Z¨ufle, Nikolas Herbst, and Samuel Kounev University of W¨urzburg, Germany”

[2] M. Z¨ufle, A. Bauer, N. Herbst, V. Curtef, and S. Kounev, “Telescope: A Hybrid Forecast Method for Univariate Time Series,” in Proceedings of the International work-conference on Time Series (ITISE 2017), September 2017.” Andr´e Bauer, Marwin Z¨ufle, Nikolas Herbst, and Samuel Kounev University of W¨urzburg, Germany”

[3] J. Hochenbaum, O. S. Vallis, and A. Kejariwal, “Automatic anomaly detection in the cloud via statistical learning,” arXiv preprint arXiv:1704.07706, 2017.” Andr´e Bauer, Marwin Z¨ufle, Nikolas Herbst, and Samuel Kounev University of W¨urzburg, Germany”

[4] G. E. Box and D. R. Cox, “An analysis of transformations,” Journal of the Royal Statistical Society. Series B (Methodological), pp. 211–252, 1964.” Andr´e Bauer, Marwin Z¨ufle, Nikolas Herbst, and Samuel Kounev University of W¨urzburg, Germany”

Coding

1.Data.py

```
import warnings

import itertools

import numpy as np

import matplotlib.pyplot as plt

warnings.filterwarnings("ignore")

plt.style.use('fivethirtyeight')

import pandas as pd

import statsmodels.api as sm

import matplotlib

matplotlib.rcParams['axes.labelsize'] = 14

matplotlib.rcParams['xtick.labelsize'] = 12

matplotlib.rcParams['ytick.labelsize'] = 12

matplotlib.rcParams['text.color'] = 'k'

df = pd.read_excel("Superstore.xls")

furniture = df.loc[df['Category'] == 'Furniture']

furniture['Order Date'].min(), furniture['Order Date'].max()
```

2.dataprocessing.py

```
cols = ['Row ID', 'Order ID', 'Ship Date', 'Ship Mode', 'Customer ID', 'Customer Name', 'Segment', 'Country', 'City', 'State', 'Postal Code', 'Region', 'Product ID', 'Category', 'Sub-Category', 'Product Name', 'Quantity', 'Discount', 'Profit']

furniture.drop(cols, axis=1, inplace=True)

furniture = furniture.sort_values('Order Date')

furniture.isnull().sum()

furniture = furniture.groupby('Order Date')['Sales'].sum().reset_index()

y.plot(figsize=(15, 6))

plt.show()
```

3.Arima.py

```
from pylab import rcParams

rcParams['figure.figsize'] = 18, 8

decomposition = sm.tsa.seasonal_decompose(y, model='additive')

fig = decomposition.plot()

plt.show()

p = d = q = range(0, 2)

pdq = list(itertools.product(p, d, q))

seasonal_pdq = [(x[0], x[1], x[2], 12) for x in list(itertools.product(p, d, q))]

//print('Examples of parameter combinations for Seasonal ARIMA...')

print('SARIMAX: {} x {}'.format(pdq[1], seasonal_pdq[1]))

print('SARIMAX: {} x {}'.format(pdq[1], seasonal_pdq[2]))
```

```
print('SARIMAX: {} x {}'.format(pdq[2], seasonal_pdq[3]))

print('SARIMAX: {} x {}'.format(pdq[2], seasonal_pdq[4]))
```

4.Forecast.py

```
pred_uc = results.get_forecast(steps=100)

pred_ci = pred_uc.conf_int()

ax = y.plot(label='observed', figsize=(14, 7))

pred_uc.predicted_mean.plot(ax=ax, label='Forecast')

ax.fill_between(pred_ci.index,

                pred_ci.iloc[:, 0],

                pred_ci.iloc[:, 1], color='k', alpha=.25)

ax.set_xlabel('Date')

ax.set_ylabel('Furniture Sales')

plt.legend()

plt.show()
```

5.DataExplore.py

```
cols = ['Row ID', 'Order ID', 'Ship Date', 'Ship Mode', 'Customer ID', 'Customer Name', 'Segment', 'Country', 'City', 'State', 'Postal Code', 'Region', 'Product ID', 'Category', 'Sub-Category', 'Product Name', 'Quantity', 'Discount', 'Profit']

furniture.drop(cols, axis=1, inplace=True)
```

```
office.drop(cols, axis=1, inplace=True)

furniture = furniture.sort_values('Order Date')

office = office.sort_values('Order Date')

furniture = furniture.groupby('Order Date')['Sales'].sum().reset_index()

office = office.groupby('Order Date')['Sales'].sum().reset_index()

furniture = furniture.set_index('Order Date')

office = office.set_index('Order Date')

y_furniture = furniture['Sales'].resample('MS').mean()

y_office = office['Sales'].resample('MS').mean()

furniture = pd.DataFrame({'Order Date':y_furniture.index, 'Sales':y_furniture.values})

office = pd.DataFrame({'Order Date': y_office.index, 'Sales': y_office.values})

store = furniture.merge(office, how='inner', on='Order Date')

store.rename(columns={'Sales_x': 'furniture_sales', 'Sales_y': 'office_sales'}, inplace=True)

store.head()
```

6.prophet.py

```
from fbprophet import Prophet

furniture = furniture.rename(columns={'Order Date': 'ds', 'Sales': 'y'})

furniture_model = Prophet(interval_width=0.95)

furniture_model.fit(furniture)

office = office.rename(columns={'Order Date': 'ds', 'Sales': 'y'})
```

```
office_model = Prophet(interval_width=0.95)

office_model.fit(office)

furniture_forecast = furniture_model.make_future_dataframe(periods=36, freq='MS')

furniture_forecast = furniture_model.predict(furniture_forecast)

office_forecast = office_model.make_future_dataframe(periods=36, freq='MS')

office_forecast = office_model.predict(office_forecast)

plt.figure(figsize=(18, 6))

furniture_model.plot(furniture_forecast, xlabel = 'Date', ylabel = 'Sales')

plt.title('Furniture Sales');
```

7.Compare.py

```
furniture_names = ['furniture_%s' % column for column in furniture_forecast.columns]

office_names = ['office_%s' % column for column in office_forecast.columns]

merge_furniture_forecast = furniture_forecast.copy()

merge_office_forecast = office_forecast.copy()

merge_furniture_forecast.columns = furniture_names

merge_office_forecast.columns = office_names

forecast = pd.merge(merge_furniture_forecast, merge_office_forecast, how = 'inner', left_on = 'furniture_ds', right_on = 'office_ds')

forecast = forecast.rename(columns={'furniture_ds': 'Date'}).drop('office_ds', axis=1)

forecast.head()
```

8.Result.py

```
for param in pdq:
    for param_seasonal in seasonal_pdq:
        try:
            mod = sm.tsa.statespace.SARIMAX(y,
                                            order=param,
                                            seasonal_order=param_seasonal,
                                            enforce_stationarity=False,
                                            enforce_invertibility=False)
results = mod.fit()
print('ARIMA{}x{}12 - AIC:{}'.format(param, param_seasonal, results.aic))
        except:
            continue
```