Business Problem

*Forecast the airlines data set. Prepare a document for each model explaining how many dummy variables you have created and RMSE value for each model. Finally which model you will use for Forecasting

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

In [2]:
```
d = pd.read_excel("Airlines+Data.xlsx")
d
```

Out[2]:

	Month	Passengers
0	1995-01-01	112
1	1995-02-01	118
2	1995-03-01	132
3	1995-04-01	129
4	1995-05-01	121
...	...	...
91	2002-08-01	405
92	2002-09-01	355
93	2002-10-01	306
94	2002-11-01	271
95	2002-12-01	306

96 rows × 2 columns

In [3]:
```
d.isnull().sum()
```

Out[3]:
```
Month         0
Passengers    0
dtype: int64
```

In [4]:
```
d.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 96 entries, 0 to 95
Data columns (total 2 columns):
 #   Column      Non-Null Count  Dtype         
---  ------      --------------  -----         
 0   Month       96 non-null     datetime64[ns]
 1   Passengers  96 non-null     int64         
dtypes: datetime64[ns](1), int64(1)
memory usage: 1.6 KB
```

In [5]:
```
d.describe()
```

Out[5]:

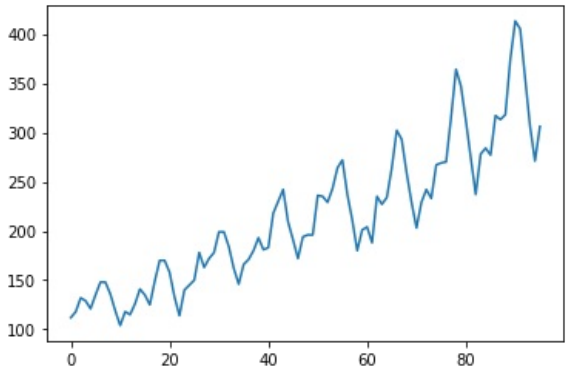
	Passengers
count	96.000000
mean	213.708333
std	71.918216
min	104.000000
25%	156.000000
50%	200.000000
75%	264.750000
max	413.000000

In [6]:
```
d.shape
```

Out[6]:
```
(96, 2)
```

In [7]:
```
d['Passengers'].plot()
#plt.show()
```

Out[7]:
```
<AxesSubplot:>
```

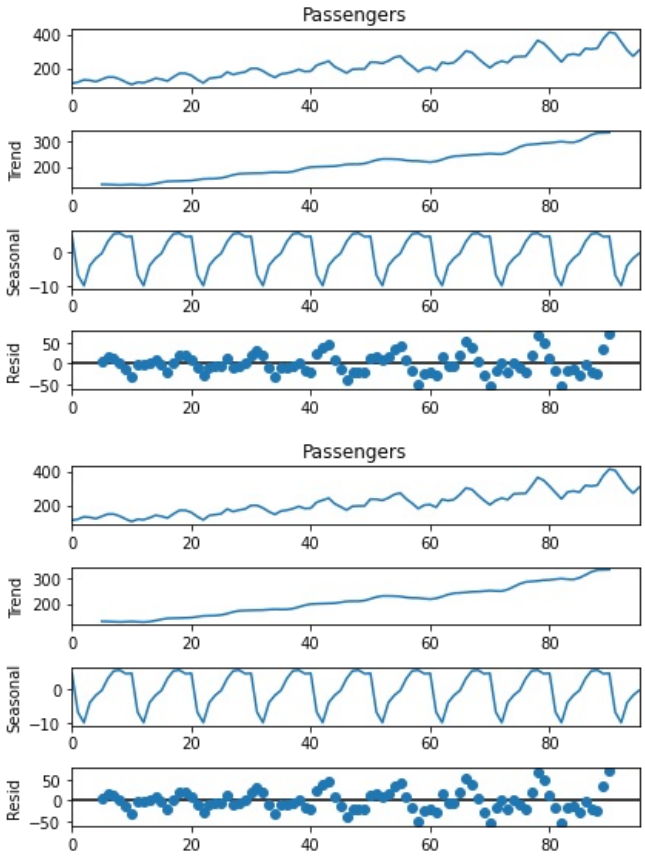


In [8]:
```
import statsmodels.api as smf
from statsmodels.tsa.seasonal import seasonal_decompose
```

In [9]:
```
seasonal_ts_add=smf.tsa.seasonal_decompose(d["Passengers"],freq=10)
seasonal_ts_add.plot()
```

```
C:\Users\rajesh\AppData\Local\Temp/ipykernel_16224/1459645772.py:1: FutureWarning: the 'freq'' keyword is deprecated, use 'period' instead
  seasonal_ts_add=smf.tsa.seasonal_decompose(d["Passengers"],freq=10)
```

Out[9]:



In [10]:
```
d['Month'] = pd.to_datetime(d['Month'])
```

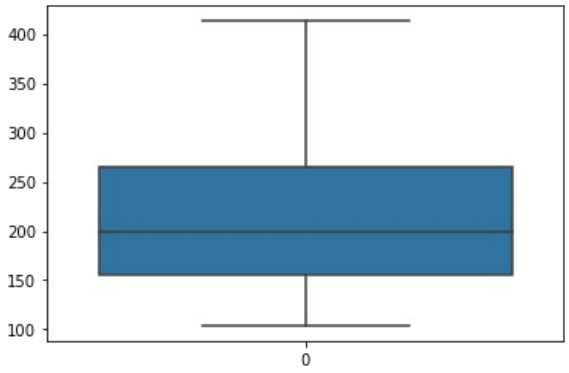
```
d['Months'] = d['Month'].dt.strftime('%b')
d.head()
```

Out[10]:

	Month	Passengers	Months
0	1995-01-01	112	Jan
1	1995-02-01	118	Feb
2	1995-03-01	132	Mar
3	1995-04-01	129	Apr
4	1995-05-01	121	May

In [11]:
```
sns.boxplot(data=d['Passengers'])
```

Out[11]:
```
<AxesSubplot:>
```



In [12]:
```
month_dummies = pd.DataFrame(pd.get_dummies(d['Months']))
```

In [13]:
```
df1 = pd.concat([d,month_dummies],axis = 1)
df1.head()
```

Out[13]:

	Month	Passengers	Months	Apr	Aug	Dec	Feb	Jan	Jul	Jun	Mar	May	Nov	Oct	Sep
0	1995-01-01	112	Jan	0	0	0	0	1	0	0	0	0	0	0	0
1	1995-02-01	118	Feb	0	0	0	1	0	0	0	0	0	0	0	0
2	1995-03-01	132	Mar	0	0	0	0	0	0	0	1	0	0	0	0
3	1995-04-01	129	Apr	1	0	0	0	0	0	0	0	0	0	0	0
4	1995-05-01	121	May	0	0	0	0	0	0	0	0	1	0	0	0

In [14]:
```
df1["t"] = np.arange(1,97)
df1["t_squared"] = df1["t"]*df1["t"]
df1["log_Passengers"] = np.log(df1["Passengers"])
df1.columns
df1.head()
```

Out[14]:

	Month	Passengers	Months	Apr	Aug	Dec	Feb	Jan	Jul	Jun	Mar	May	Nov	Oct	Sep	t	t_squared	log_Passengers
0	1995-01-01	112	Jan	0	0	0	0	1	0	0	0	0	0	0	0	1	1	4.718499
1	1995-02-01	118	Feb	0	0	0	1	0	0	0	0	0	0	0	0	2	4	4.770685
2	1995-03-01	132	Mar	0	0	0	0	0	0	0	1	0	0	0	0	3	9	4.882802
3	1995-04-01	129	Apr	1	0	0	0	0	0	0	0	0	0	0	0	4	16	4.859812
4	1995-05-01	121	May	0	0	0	0	0	0	0	0	1	0	0	0	5	25	4.795791

In [15]:
```
Train = df1.head(75)
Test = df1.tail(25)
```

In [16]:
```
# L I N E A R
import statsmodels.formula.api as smf
```

```
linear_model = smf.ols('Passengers~t',data=Train).fit()
pred_linear =  pd.Series(linear_model.predict(pd.DataFrame(Test['t'])))
rmse_linear = np.sqrt(np.mean((np.array(Test['Passengers'])-np.array(pred_linear))**2))
print("RMSE Linear: ",rmse_linear)
```

```
RMSE Linear:  51.83809749584507
```

In [17]:

```
# Exponential

Exp = smf.ols('log_Passengers~t',data=Train).fit()
pred_Exp = pd.Series(Exp.predict(pd.DataFrame(Test['t'])))
rmse_Exp = np.sqrt(np.mean((np.array(Test['Passengers'])-np.array(np.exp(pred_Exp)))**2))
print("RMSE Exponential: ",rmse_Exp)
```

```
RMSE Exponential:  42.775259750198
```

In [18]:

```
# Quadratic

Quad = smf.ols('Passengers~t+t_squared',data=Train).fit()
pred_Quad = pd.Series(Quad.predict(Test[["t","t_squared"]]))
rmse_Quad = np.sqrt(np.mean((np.array(Test['Passengers'])-np.array(pred_Quad))**2))
print("RMSE Quadratic: ",rmse_Quad)
```

```
RMSE Quadratic:  54.03140645625428
```

In [19]:

```
# Additive seasonality

add_sea = smf.ols('Passengers~Jan+Feb+Mar+Apr+May+Jun+Jul+Aug+Sep+Oct+Nov+Dec',data=Train).fit()
pred_add_sea = pd.Series(add_sea.predict(Test[['Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov',
rmse_add_sea = np.sqrt(np.mean((np.array(Test['Passengers'])-np.array(pred_add_sea))**2))
print("RMSE Additive seasonality: ",rmse_add_sea)
```

```
RMSE Additive seasonality:  123.0276378808424
```

In [20]:

```
#Additive Seasonality Quadratic

add_sea_Quad = smf.ols('Passengers~t+t_squared+Jan+Feb+Mar+Apr+May+Jun+Jul+Aug+Sep+Oct+Nov+Dec',data=Train).fit()
pred_add_sea_quad = pd.Series(add_sea_Quad.predict(Test[['Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','O
rmse_add_sea_quad = np.sqrt(np.mean((np.array(Test['Passengers'])-np.array(pred_add_sea_quad))**2))
print("RMSE Additive Seasonality Quadratic:",rmse_add_sea_quad )
```

```
RMSE Additive Seasonality Quadratic: 36.536274445464215
```

In [21]:

```
# Multiplicative Seasonality

Mul_sea = smf.ols('log_Passengers~Jan+Feb+Mar+Apr+May+Jun+Jul+Aug+Sep+Oct+Nov+Dec',data = Train).fit()
pred_Mult_sea = pd.Series(Mul_sea.predict(Test))
rmse_Mult_sea = np.sqrt(np.mean((np.array(Test['Passengers'])-np.array(np.exp(pred_Mult_sea)))**2))
print("RMSE Multiplicative Seasonality:",rmse_Mult_sea)
```

```
RMSE Multiplicative Seasonality: 128.16622817596138
```

In [22]:

```
# Multiplicative Additive Seasonality

Mul_Add_sea = smf.ols('log_Passengers~t+Jan+Feb+Mar+Apr+May+Jun+Jul+Aug+Sep+Oct+Nov+Dec',data = Train).fit()
pred_Mult_add_sea = pd.Series(Mul_Add_sea.predict(Test))
rmse_Mult_add_sea = np.sqrt(np.mean((np.array(Test['Passengers'])-np.array(np.exp(pred_Mult_add_sea)))**2))
print("RMSE Multiplicative Additive Seasonality:",rmse_Mult_add_sea )
```

```
RMSE Multiplicative Additive Seasonality: 11.565825437685964
```

In [23]:

```
# Testing

data1 = {"MODEL":pd.Series(["rmse_linear","rmse_Exp","rmse_Quad","rmse_add_sea","rmse_add_sea_quad","rmse_Mult_se
        "RMSE_Values":pd.Series([rmse_linear,rmse_Exp,rmse_Quad,rmse_add_sea,rmse_add_sea_quad,rmse_Mult_sea,rmse
```

```
table_rmse=pd.DataFrame(data1)
table_rmse
```

Out[23]:

	MODEL	RMSE_Values
0	rmse_linear	51.838097
1	rmse_Exp	42.775260
2	rmse_Quad	54.031406
3	rmse_add_sea	123.027638
4	rmse_add_sea_quad	36.536274
5	rmse_Mult_sea	128.166228
6	rmse_Mult_add_sea	11.565825

In [24]:

```
data = [['2003-01-01', 'Jan'], ['2003-02-01', 'Feb'], ['2003-03-01', 'Mar'],['2003-04-01', 'Apr'],['2003-5-01', '
        ['2003-07-01', 'Jul'],['2003-08-01', 'Aug'],['2003-09-01', 'Sep'],['2003-10-01', 'Oct'],['2003-11-01', 'N
forecast = pd.DataFrame(data, columns = ['Date', 'Months'])
forecast
```

Out[24]:

	Date	Months
0	2003-01-01	Jan
1	2003-02-01	Feb
2	2003-03-01	Mar
3	2003-04-01	Apr
4	2003-5-01	May
5	2003-06-01	Jun
6	2003-07-01	Jul
7	2003-08-01	Aug
8	2003-09-01	Sep
9	2003-10-01	Oct
10	2003-11-01	Nov
11	2003-12-01	Dec

In [25]:

```
# Create dummies and T and T-Squared columns

dummies = pd.DataFrame(pd.get_dummies(forecast['Months']))
forecast1 = pd.concat([forecast,dummies],axis = 1)

forecast1["t"] = np.arange(1,13)
forecast1["t_squared"] = forecast1["t"]*forecast1["t"]
print("\nAfter Dummy, T and T-Square\n\n",forecast1.head())
```

```
After Dummy, T and T-Square

          Date Months  Apr  Aug  Dec  Feb  Jan  Jul  Jun  Mar  May  Nov  Oct  \
0  2003-01-01    Jan    0    0    0    0    1    0    0    0    0    0    0
1  2003-02-01    Feb    0    0    0    1    0    0    0    0    0    0    0
2  2003-03-01    Mar    0    0    0    0    0    0    0    1    0    0    0
3  2003-04-01    Apr    1    0    0    0    0    0    0    0    0    0    0
4   2003-5-01    May    0    0    0    0    0    0    0    0    1    0    0

   Sep  t  t_squared
0    0  1          1
1    0  2          4
2    0  3          9
3    0  4         16
4    0  5         25
```

In [26]:

```
# Forecasting using Multiplicative Additive Seasonality Model

model_full = smf.ols('log_Passengers~t+Jan+Feb+Mar+Apr+May+Jun+Jul+Aug+Sep+Oct+Nov+Dec',data =df1).fit()
pred_new  = pd.Series(model_full.predict(forecast1))

forecast1["Forecasted_log"] = pd.Series(pred_new)
forecast1['Forecasted_Passengers'] = np.exp(forecast1['Forecasted_log'])
```

In [27]:

```
# Final Prediction for next 12 months
```

```
Final_predict = forecast1.loc[:, ['Date', 'Forecasted_Passengers']]
Final_predict
```

Out[27]:

	Date	Forecasted_Passengers
0	2003-01-01	109.176148
1	2003-02-01	110.331245
2	2003-03-01	127.315234
3	2003-04-01	123.200587
4	2003-5-01	122.399578
5	2003-06-01	138.536397
6	2003-07-01	154.066959
7	2003-08-01	153.741209
8	2003-09-01	137.693733
9	2003-10-01	120.894736
10	2003-11-01	106.109309
11	2003-12-01	121.633998

In [ ]: