PRODUCT DEMAND PREDICTION WITH MACHINE LEARNING

REG NO:912421104050

NAME: K.TAMILOLI

PHASE 3SUBMISSION DOCUMENT

INTRODUCTION:

A product company plans to offer discounts on its product during the upcoming holiday season. The company wants to find the price at which its product can be a better deal compared to its competitors. For this task,the company provided a dataset of past changes in sales based on price changes. You need to train a model that can predict the demand for the product in the market with different price segments.

The **dataset**that we have for this task contains data about:

1.the product id;

2.store id;

3.total price at which product was sold;

4.base price at which product was sold;

5.Units sold (quantity demanded);

Product Demand Prediction using Python:

Example:

```
import pandas aspd
import numpy asnp
import plotly.express aspx
import seaborn as sns
import matplotlib.pyplot asplt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor

data =
pd.read_csv("https://raw.githubusercontent.com/amankharwal/Website-data/master/demand.csv")
data.head()
```

Output:

	ID	Store ID	Total Price	Base Price	Units Sold
0	**1**	**8091**	**99.0375**	**111.8625**	**20**
1	**2**	**8091**	**99.0375**	**99.0375**	**28**
2	**3**	**8091**	**133.9500**	**133.9500**	**19**
3	**4**	**8091**	**133.9500**	**133.9500**	**44**
4	**5**	**8091**	**141.0750**	**141.0750**	**52**

Content:

Consider incorporating time series forecasting techniques like ARIMA or Prophet to capture temporal patterns in demand data.

Data source:

To predict the future, statistics utilize data from the past. That's why statistical forecasting is often called *historical*. The common recommendation is collecting data on sales for at least two years.

Dataset Link:

https://www.kaggle.com/datasets/chakradharmattapalli/product-demand-prediction-with-machine-learning

ID	Store ID	Total Price	Base Price	Units Sold
1	8091	99.0375	111.8625	20
2	8091	99.0375	99.0375	28
3	8091	133.95	133.95	19
4	8091	133.95	133.95	44
5	8091	141.075	141.075	52
9	8091	227.2875	227.2875	18
10	8091	327.0375	327.0375	47

1 3	8091	210.9 50	210.9	
1 4	8091	190.2375 82	234.4125	
1 7	8095	99.0375 99	99.0375	
1 8	8095	97.6125	97.6125	120
1 9	8095	98.325 40	98.325	
2 2	8095	133.2375 68	133.2375	
2 3	8095	133.95 87	133.95	
2 4	8095	139.65	139.65	186
2 7	8095	236.55 54	280.0125	
2 8	8095	214.4625 74	214.4625	

29	8095	266.475	296.4	102
30	8095	173.85	192.375	214
31	8095	205.912528	205.9125	
32	8095	205.9125	205.9125	7
33	8095	248.662548	248.6625	
34	8095	200.92578	200.925	
35	8095	190.237557	240.825	
37	8095	427.550	448.1625	
38	8095	429.637562	458.1375	
39	8095	177.412522	177.4125	
42	8094	87.6375	87.6375	109

43	8094	88.35	88.35	133
44	8094	85.5	85.5	11
45	8094	128.25	180.975	9
47	8094	127.5375	127.5375	19
48	8094	123.975	123.975	33
49	8094	139.65	164.5875	49
50	8094	235.8375	235.8375	32
51	8094	234.4125	234.4125	47
52	8094	235.125	235.125	27
53	8094	227.2875	227.2875	69

54	8094	312.7875	312.7875	49

Data collection and preprocessing:

- Data Collection: Data collection is the process of gathering and measuring information on variables of interest, in an established systematic fashion that enables one to answer stated research questions, test hypotheses, and evaluate outcomes.

- Data preprocessing: A component of data preparation, describes any type of processing performed on raw data to prepare it for another data processing procedure. It has traditionally been an important preliminary step for the data mining process.

MODEL SELECTION:

There are some regression algorithms are present

- Linear regression
- Random forest

- XG boost

FEATURE ENGINEERING:

Create new features or transform existing ones to capture valuable information.

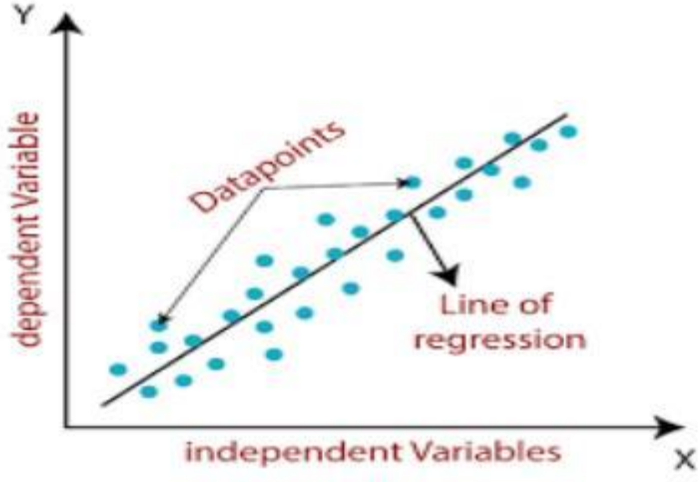
- **Enhanced Model Performance**
- **Improved Interpretability**
- **Handling Non-linearity**

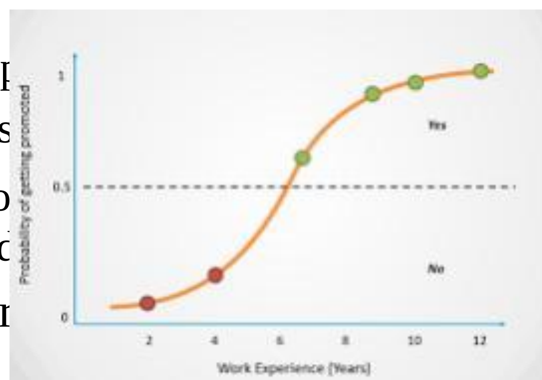
Time-based features: These features capture trends and patterns overtime. Examples include day of the week, month, year, and holidays.

Store-based features: These features capture pharmacy-specific characteristics. Examples include the location of the pharmacy, the size of the pharmacy, and the customer demographics.

ADVANCED REGRESSION TECHNIQUES:

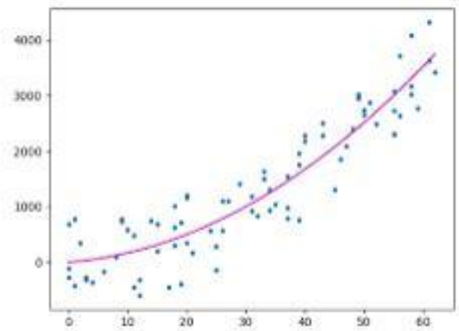
- Linear : Linear regression analysis is used to predict the value of a variable based on the value of another variable.



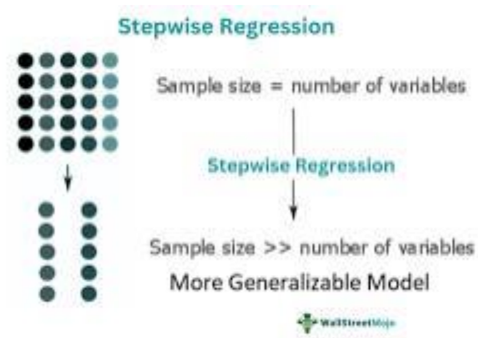
- Logistic : This typ  wn as logit model) is often us ive analytics.

  Logistic regressio event occurring, such as voted or d aset of independent var

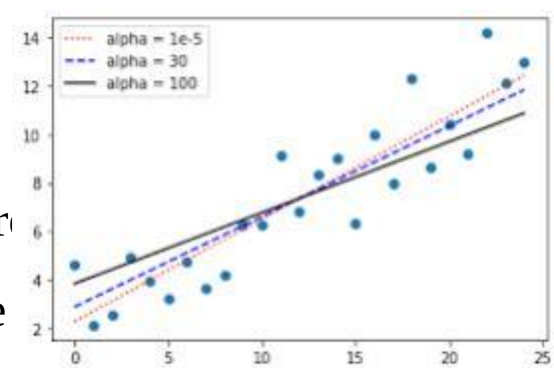
- Polynomial : Polynomial regression is a form of Linear regression where only due to the Non-linear relationship between dependent and independent variables.



- Stepwise : Stepwise regression is the step-by-step iterative construction of a regression model that involves the selection of independent variables to be used in a final model.

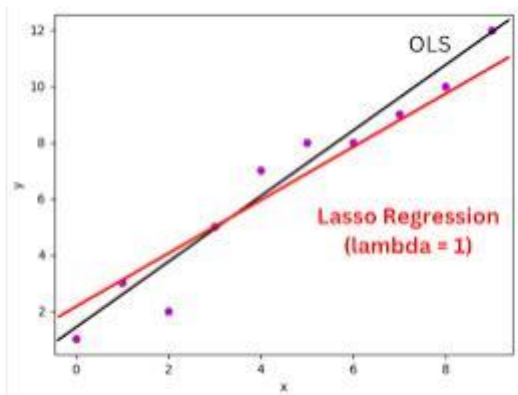


•

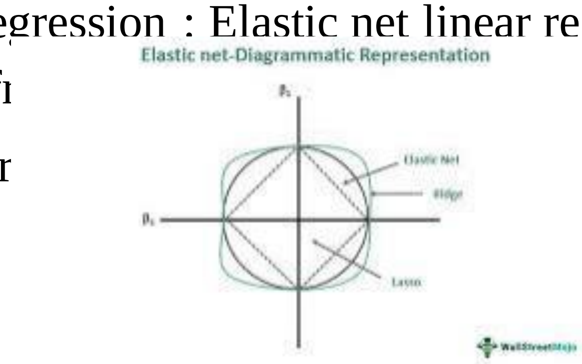


- Ridge : Ridge r nethod that is used to analyse ulticollinearity.

- Lasso : LASSO regression, also known as L1 regularization, is a popular technique used in statistical modeling and machine learning to estimate the relationships between variables and make predictions.



- Elastic Net Regression : Elastic net linear regression uses the penalties fr chniques to regularize regr



.

ARIMA technique:

ARIMA is a method for forecasting or predicting future outcomes based on a historical time series. It is based on the statistical concept of serial correlation, where past data points influence future data points.

Why is ARIMA good for forecasting?

Advantages of ARIMA models:

ARIMA models can account for various patterns, such as linear or nonlinear trends, constant or varying volatility, and seasonal or non-seasonal fluctuations.

ARIMA Time-series Forecasting Methods:

Autoregressive integrated moving average (ARIMA) forecasting methods were popularized by G. E. P. Box and G. M. Jenkins in the 1970s. These techniques, often called the Box-Jenkins forecasting methodology, have the following steps:

1.Model identification and selection

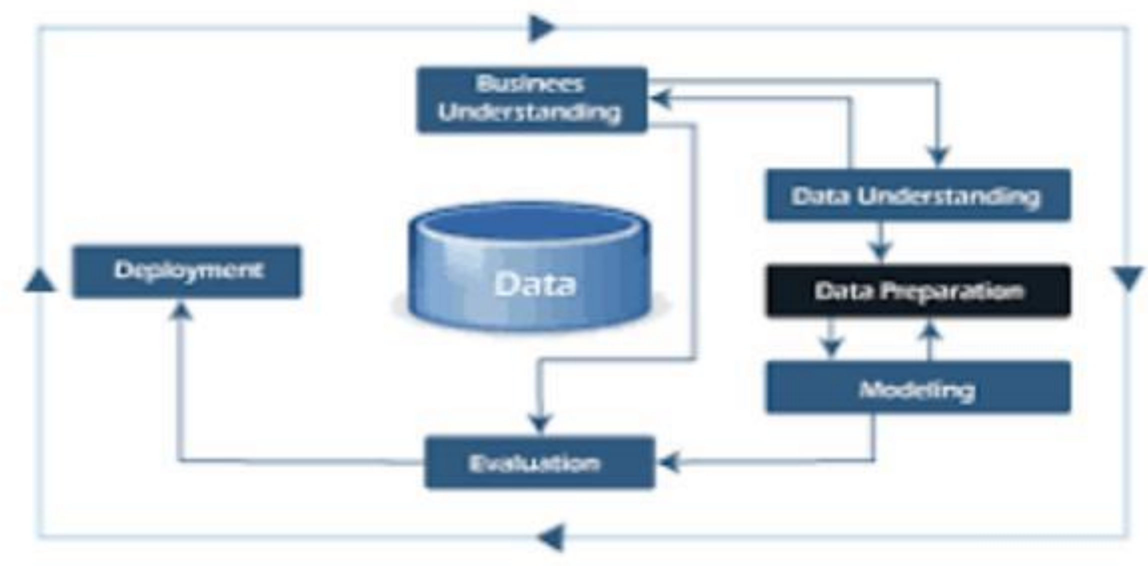
2.Estimation of autoregressive (AR), integration or differencing
3.(I), and moving average (MA) parameters
Model checking

ARIMA is a univariate process. Current values of a data series are correlated with past values in the same series to produce the AR component, also known as *p*. Current values of a random error term are correlated with past values to produce the MA component, *q*. Mean and variance values of current and past data are assumed to be stationary, unchanged overtime. If necessary, an I component (symbolized by *d*) is added to correct for a lack of stationarity through differencing.

In a nonseasonal ARIMA(p,d,q) model, p indicates the number or order of AR terms, d indicates the number or order of differences, and q indicates the number or order of MA terms. The p, d, and q parameters are integers equal to or greater than 0.

Data wrangling technique:

- Merging several data sources into one data-set for analysis.
- Identifying gaps or empty cells in data and either filling or removing them.
- Deleting irrelevant or unnecessary data.
- Identifying severe outliers in data and either explaining the inconsistencies or deleting them to facilitate analysis.



 SARIMA (Seasonal ARIMA):

Extends ARIMA to handle seasonal patterns in data.

 Exponential Smoothing Methods:

These include Holt-Winters for capturing trends and seasonality.

 Prophet:

Developed by Facebook, Prophet is useful for data with daily observations, holidays, and seasonality.

- Deep Learning Models (e.g., LSTM and GRU):

Suitable for capturing complex temporal patterns, but they may require more data and computational resources.

Model Training:

Train the selected time series forecasting model using historical demand data. This involves estimating model parameters and seasonal components, if applicable.

Supervised learning:

Supervised learning, also known as supervised machine learning, is a subcategory of machine learning and artificial intelligence. It is defined by its use of labeled datasets to train algorithms that to classify data or predict outcomes accurately.

CLASSIFICATION:

Classification is a supervised machine learning method where the model tries to predict the correct label of a given input data. In classification, the model is fully trained using the training data, and then it is evaluated on test data before being used to perform prediction on new unseen data.

Hyper parameter optimization:

Hyperparameter optimization or tuning is the problem of choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a parameter whose value is used to control the learning process. By contrast, the values of other parameters (typically node weights) are learned.

Data:

1. ID Store ID Total Price Base Price Units Sold
2. 1 8091 99.0375 111.8625 20
3. 2 8091 99.0375 99.0375 28
4. 3 8091 133.95 133.95 19
5. 4 8091 133.95 133.95 44

6. 5	8091	141.075	141.075	52

7.	9	8091	227.2875	227.2875	18
8.	10	8091	327.0375	327.0375	47
9.	13	8091	210.9	210.9	50
10.	14	8091	190.2375	234.4125	82
11.	17	8095	99.0375	99.0375	99
12.	18	8095	97.6125	97.6125	120
13.	19	8095	98.325	98.325	40
14.	22	8095	133.2375	133.2375	68
15.	23	8095	133.95	133.95	87

16.	24	8095	139.65	139.65	186
17.	27	8095	236.55	280.0125	54

18.	28	8095	214.462574	214.4625	
19.	29	8095	266.475	296.4	102
20.	30	8095	173.85	192.375	214
21.	31	8095	205.912528	205.9125	
22.	32	8095	205.9125	205.9125	7
23.	33	8095	248.662548	248.6625	
24.	34	8095	200.92578	200.925	
25.	35	8095	190.237557	240.825	
26.	37	8095	427.550	448.1625	
27.	38	8095	429.637562	458.1375	

28.	3 9	8095	177.4125	177.4125 22
29.	4 2	8094	87.6375	87.6375 109
30.	4 3	8094	88.35	88.35 133
31.	4 4	8094	85.5 85.5	11
32.	4 5	8094	128.25	180.975 9
33.	4 7	8094	127.5375	127.5375 19
34.	4 8	8094	123.975	123.975 33
35.	4 9	8094	139.65	164.5875 49
36.	5 0	8094	235.8375	235.8375 32
37.	5 1	8094	234.4125	234.4125 47
38.	5 2	8094	235.125	235.125 27

39.	53	8094	227.2875	227.2875	69
40.	54	8094	312.7875	312.7875	49

<u>PROGRAM:</u>

Product Demand Prediction:

Import pandas aspd

Import numpy asnp

Import plotly.express aspx

Import seaborn as sns

Import matplotlib.pyplot asplt

From sklearn.model_selection import train_test_split

From sklearn.tree import DecisionTreeRegressor

Data=pd.read_csv("C:\Users\mabir\AppData\Local\Microsoft\Windows\INetCache\IE\AHLGJQP8\archive[1].zip ")

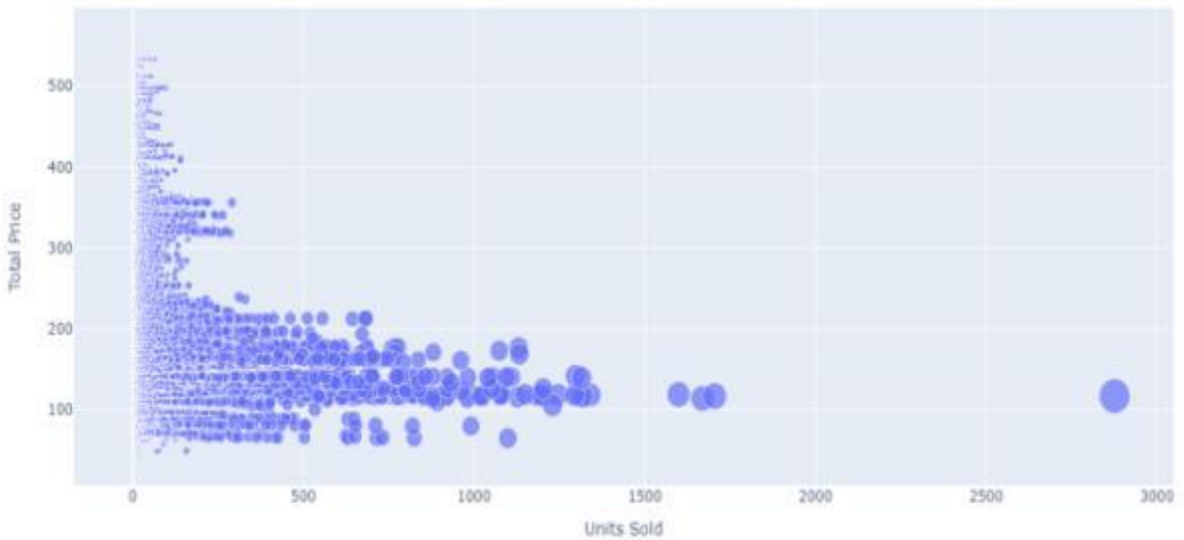
Data.head()

<u>Relationship between price and demand for the product:</u>

```
Fig = px.scatter(data, x=”Units Sold”, y=”Total Price”,
            Size=’Units Sold’)
```

Fig.show()

Output:



Correlation between the features of the dataset:

Print(data.corr())

Output:

ID Store ID Total Price Base Price Units Sold

ID 1.000000 0.007464 0.008473 0.018932 -
0.010616

Store ID 0.007464 1.000000 -0.038315 -0.038848 -
0.004372

Total Price 0.008473 -0.038315 1.000000 0.958885 -
0.235625

Base Price 0.018932 -0.038848 0.958885 1.000000 -
0.140032

Units Sold -0.010616 -0.004372 -0.235625 -0.140032
1.000000

1

Correlations = data.corr(method='pearson')

2

Plt.figure(figsize=(15, 12))

3

Sns.heatmap(correlations, cmap=”coolwarm”, annot=True)

4

Plt.show()

<u>Output:</u>

ID
Store ID
Total Price
Base Price
1
0.0075
0.0085
0.019
-0.011
0.0075
1
-0.038
-0.039
-0.0044
0.0085
-0.038
1
0.96
-0.24
0.019
-0.039
0.96
1
-0.14
1.0
0.8
0.6
0.4
0.2

```
# fit an ARIMA model and plot residual errors
From pandas import datetime
From pandas import read_csv
From pandas import DataFrame
From statsmodels.tsa.arima.model import ARIMA
From matplotlib import pyplot
# load dataset
Def parser(x):
Return datetime.strptime('190'+x, '%Y-%m')
Series = read_csv('shampoo-sales.csv', header=0, index_col=0,
parse_dates=True, squeeze=True, date_parser=parser)
Series.index = series.index.to_period('M')
# fit model
Model = ARIMA(series, order=(5,1,0))
Model_fit = model.fit()
# summary offit model
Print(model_fit.summary())
# line plot of residuals
```

```
Residuals = DataFrame(model_fit.resid)

Residuals.plot()
Pyplot.show()
# density plot of residuals

Residuals.plot(kind='kde')
Pyplot.show()
# summary stats of residuals

Print(residuals.describe())
```

Output:

SARIMAX Results

Dep. Variable: Sales No. Observations:

36

408.969

Model: ARIMA(5, 1, 0) Log Likelihood -
198.485

Date: Thu, 10 Dec 2020 AIC

408.969

Time: 09:15:01 BIC 418.301

Sample: 01-31-1901 HQIC
412.191

- 12-31-1903

Covariance Type: opg

	Coef	std err	z	P>|z|	[0.025	0.975]
Ar.L1	-0.9014	0.247	-3.647	0.000	-1.386	-
0.417						
Ar.L2	-0.2284	0.268	-0.851	0.395	-0.754	
0.298						

Ar.L3	0.0747	0.291	0.256	0.798	-0.497

0.646

Ar.L4	0.2519	0.340	0.742	0.458	-0.414	0.918
Ar.L5	0.3344	0.210	1.593	0.111	-0.077	0.746
Sigma2	4728.9608	1316.021	3.593	0.000	2149.607	7308.314

Ljung-Box (L1) (Q):	0.61	Jarque-Bera (JB):	0.96
Prob(Q):	0.44	Prob(JB):	0.62
Heteroskedasticity (H):	1.07	Skew:	0.28

Prob(H) (two-sided): 0.90 Kurtosis:
2.41

Prophet:

```
# make an in-sample forecast
from pandas import read_csv
from pandas import to_datetime
from pandas import DataFrame
from fbprophet import Prophet
from matplotlib import pyplot
# load data
path =
'https://raw.githubusercontent.com/jbrownlee/Datasets/master/monthly-car-sales.csv'
df = read_csv(path, header=0)
# prepare expected column names
df.columns = ['ds', 'y']
df['ds']= to_datetime(df['ds'])
# define the model
model = Prophet()
# fit the model
model.fit(df)
# define the period for which we want a prediction
future = list()
for i in range(1, 13):
date = '1968-%02d' % i
```

```
future.append([date])
future = DataFrame(future)

future.columns = ['ds']
future['ds']= to_datetime(future['ds'])

# use the model to make a forecast
forecast = model.predict(future)

# summarize the forecast
print(forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].head())
# plot forecast
model.plot(forecast)

pyplot.show()
```

Running the example forecasts the last 12 months of the dataset.

The first five months of the prediction are reported and we can see that values are not too different from the actual sales values in the dataset.

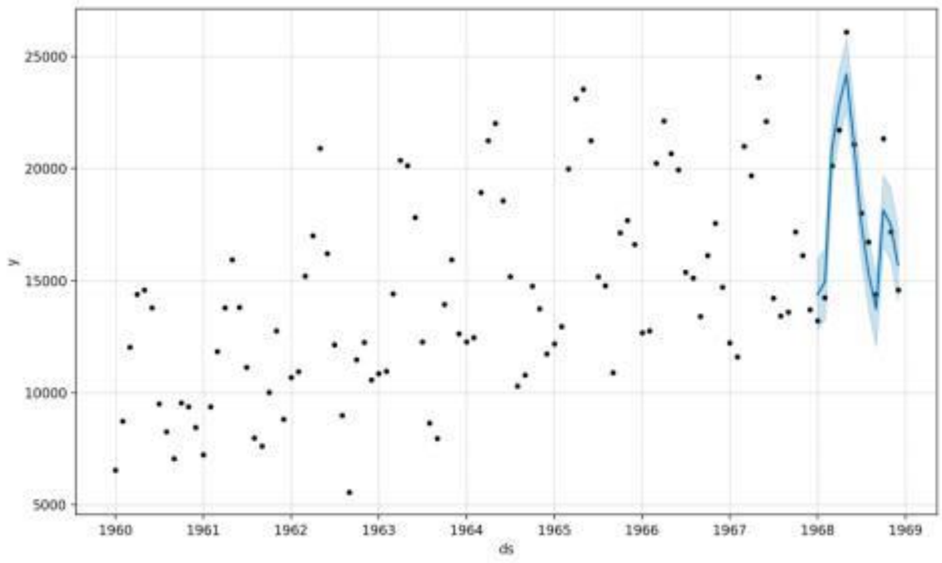
OUTPUT:

	ds	yhat	yhat_lower	yhat_upper
0	1968-01-01	14364.866157	12816.266184	15956.555409
1	1968-02-01	14940.687225	13299.473640	16463.811658

2	1968-03-01	20858.282598	19439.403787	22345.747821

3 1968-04-01 22893.610396 21417.399440 24454.642588

4 1968-05-01 24212.079727 22667.146433 25816.191457



EVALUATION:

- Mean Absolute Error(MAE) is the mean size of the mistakes in collected predictions. We know that an error basically is the absolute difference between the actual or true values and the values that are predicted. The absolute difference means that if the result has a negative sign, it is ignored.

What is Mean Squared Error or MSE

- The Mean Absolute Error is the squared mean of the difference between the actual values and predictable values.