

Practical No. 5

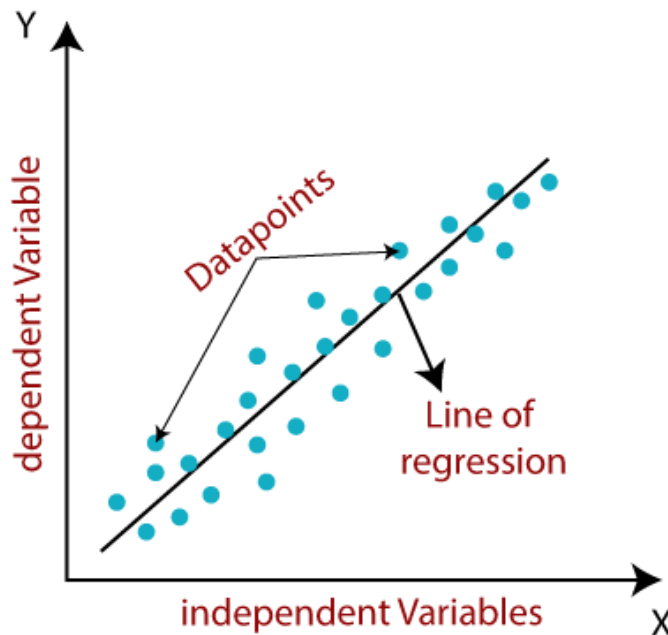
Title: Implementation Supervised Learning algorithms : Linear Regression, Logistic regression, Support Vector Machine or SVM

Aim: Understanding basics of Linear Regression, Logistic regression and Support Vector Machine or SVM

Introduction:

Linear Regression

Linear Regression is a machine learning algorithm based on **supervised learning**. It performs a **regression task**. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between dependent and independent variables they are considering, and the number of independent variables getting used.



Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (x) variables, hence called linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

The linear regression model provides a sloped straight line representing the relationship between the variables.

Mathematically, we can represent a linear regression as:

$$y = mx + b$$

Here,

Y= Dependent Variable (Target Variable)

X= Independent Variable (predictor Variable)

b= intercept of the line (Gives an additional degree of freedom)

m= Linear regression coefficient (scale factor to each input value).

The values for x and y variables are training datasets for Linear Regression model representation.

Finding the best fit line:

When working with linear regression, our main goal is to find the best fit line that means the error between predicted values and actual values should be minimized. The best fit line will have the least error.

The different values for weights or the coefficient of lines (a_0 , a_1) gives a different line of regression, so we need to calculate the best values for a_0 and a_1 to find the best fit line, so to calculate this we use cost function.

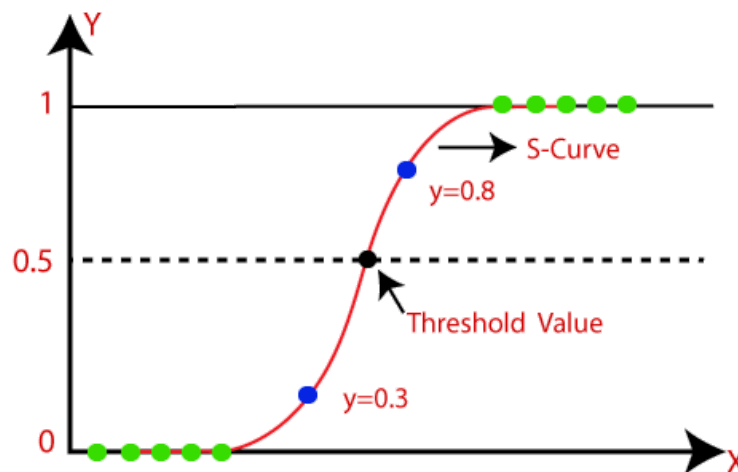
Logistic Regression in Machine Learning

- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
- Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or

No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1.**

- Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas **Logistic regression is used for solving the classification problems.**
- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.
- Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification.

The below image is showing the logistic function:

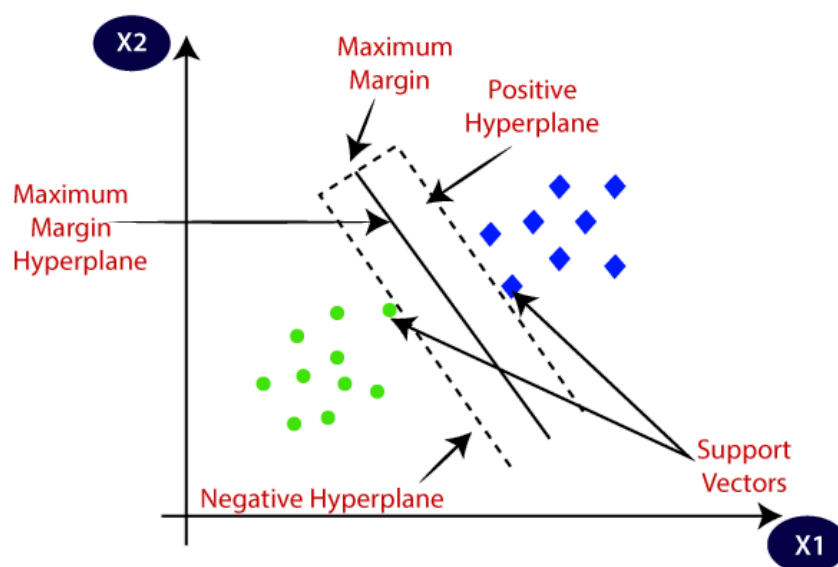


Support Vector Machine or SVM

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

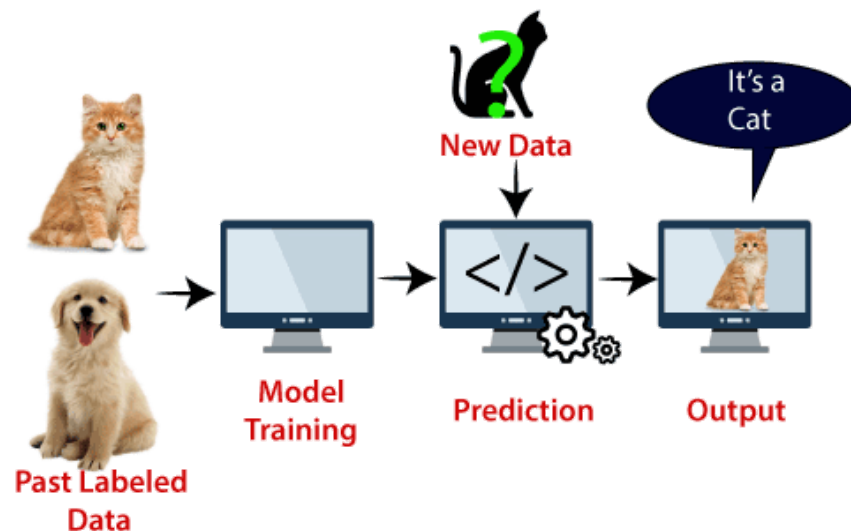
The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called support vectors, and hence the algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:



Example: SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange

creature. So as the support vector creates a decision boundary between these two data (cat and dog) and chooses extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify it as a cat. Consider the below diagram:



SVM algorithm can be used for Face detection, image classification, text categorization, etc.

Types of SVM

SVM can be of two types:

- **Linear SVM:** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
- **Non-linear SVM:** Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

Hyperplane and Support Vectors in the SVM algorithm:

Hyperplane: There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.

The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.

We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

Support Vectors:

The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

Exercise -

1. Implementation of Linear regression
2. Implementation of Logistic regression
3. Support Vector Machine Tutorial Using Python Sklearn

Implementation:

Program:

```
#Linear Regression : Supervised Machine Learning
```

```
import numpy as np
```

```
import pandas as pd
```

```
from sklearn import linear_model
```

```
from matplotlib import pyplot as plt
```

```
url="https://raw.githubusercontent.com/codebasics/py/master/ML/1_linear_reg/homeprices.csv"
```

```
hp=pd.read_csv(url)
```

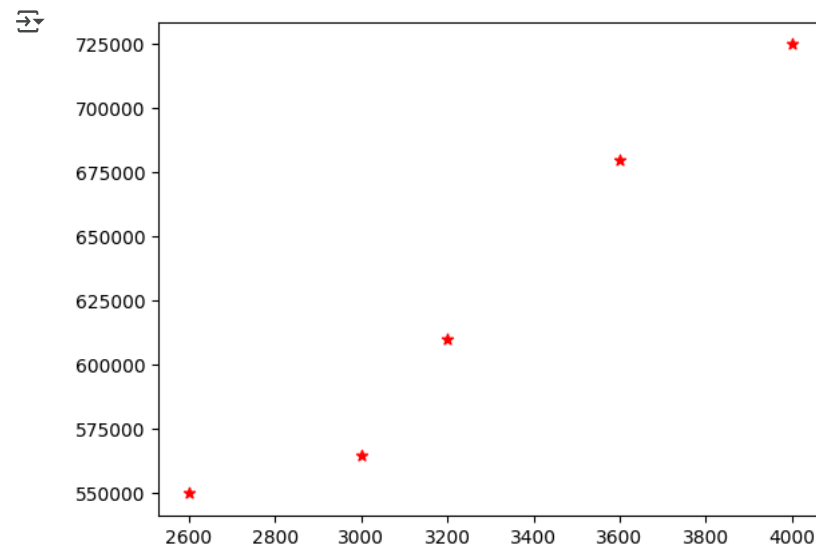
```
hp
```

```
↕
```

	area	price
0	2600	550000
1	3000	565000
2	3200	610000
3	3600	680000
4	4000	725000

```
plt.scatter(hp.area, hp.price, color="r", marker="*")
```

```
plt.show()
```



```
rg=linear_model.LinearRegression()
```

```
rg.fit(hp[['area']], hp.price)
```

```
↕
```

LinearRegression ⓘ ?

LinearRegression()

```
rg.predict([[3300]])
```

```
↕ /usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but Lin
warnings.warn(
array([628715.75342466])
```

```
rg.predict([[3900]])
```

```
↕ /usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but Lin
warnings.warn(
array([710188.35616438])
```

```
rg.coef_
```

```
↕ array([135.78767123])
```

```
rg.intercept_
```

```
np.float64(180616.43835616432)
```

```
135.78767123*3300+180616.43835616432
```

```
628715.7534151643
```

```
#Linear Regression : Supervised Machine Learning
```

```
url="https://raw.githubusercontent.com/codebasics/py/master/ML/1_linear_reg/areas.csv"
```

```
ar1=pd.read_csv(url)
```

```
ar1
```

```
↕
```

	area
0	1000
1	1500
2	2300
3	3540
4	4120
5	4560
6	5490
7	3460
8	4750
9	2300
10	9000
11	8600
12	7100

```
p=rg.predict(ar1)
```

```
ar1['prices']=p
```

```
ar1
```

```
↕
```

	area	prices
0	1000	3.164041e+05
1	1500	3.842979e+05
2	2300	4.929281e+05
3	3540	6.613048e+05
4	4120	7.400616e+05
5	4560	7.998082e+05
6	5490	9.260908e+05
7	3460	6.504418e+05
8	4750	8.256079e+05
9	2300	4.929281e+05
10	9000	1.402705e+06
11	8600	1.348390e+06
12	7100	1.144709e+06

```
ar1.to_csv('newareas.csv')
```


#Linear Regression : Supervised Machine Learning Multivariable

```
import numpy as np
import pandas as pd
from sklearn import linear_model
from matplotlib import pyplot as plt
url="https://raw.githubusercontent.com/codebasics/py/master/ML/2_linear_reg_multivariate/homepr
homep=pd.read_csv(url)
homep
```




	area	bedrooms	age	price
0	2600	3.0	20	550000
1	3000	4.0	15	565000
2	3200	NaN	18	610000
3	3600	3.0	30	595000
4	4000	5.0	8	760000
5	4100	6.0	8	810000

```
homep.bedrooms=homep.bedrooms.fillna(homep.bedrooms.median())
homep
```




	area	bedrooms	age	price
0	2600	3.0	20	550000
1	3000	4.0	15	565000
2	3200	4.0	18	610000
3	3600	3.0	30	595000
4	4000	5.0	8	760000
5	4100	6.0	8	810000

```
rg=linear_model.LinearRegression()
rg.fit(homep[['area','bedrooms','age']],homep.price)
```




LinearRegression ⓘ ?
LinearRegression()

```
rg.predict([[3000,3,40]])
```




```
/usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but Lin
warnings.warn(
array([498408.25158031])
```

```
rg.coef_
```




```
array([ 112.06244194, 23388.88007794, -3231.71790863])
```

```
rg.intercept_
```



```
np.float64(221323.00186540396)
```

```
112.06244194*3000+23388.88007794*3+-3231.71790863*40+221323.00186540396
```



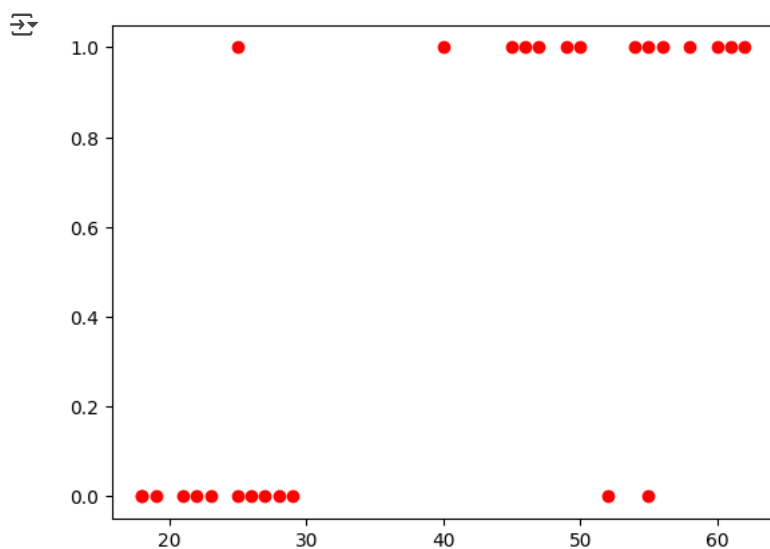
```
498408.251574024
```

#Logistic Regression : Supervised Machine Learning

```
import numpy as np
import pandas as pd
from sklearn import linear_model
from matplotlib import pyplot as plt
url="https://raw.githubusercontent.com/WamanParulekar/AI ML/main/lic.csv
```

	age	lic_member
0	22	0
1	25	0
2	47	1
3	52	0
4	28	0
5	27	0
6	29	0
7	49	1
8	55	1
9	25	1
10	58	1
11	19	0
12	46	1
13	56	1
14	55	0
15	60	1
16	62	1
17	61	1
18	18	0
19	18	0
20	21	0
21	26	0
22	40	1
23	45	1
24	50	1
25	54	1
26	23	0

```
plt.scatter(lic.age,lic.lic_member,color="r")
plt.show()
```



```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(lic[['age']],lic.lic_member,train_size=0.9)
```

```
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()
lr.fit(X_train,y_train)
```

LogisticRegression

```
lr.predict([[54]])
```

/usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but Log warnings.warn(array([1])

X_test

	age
17	61
25	54
22	40

```
lr.score(X_test,y_test)
```

0.6666666666666666

#Support Vector Machine : Supervised Machine Learning

```
import numpy as np
import pandas as pd
from sklearn import linear_model
from matplotlib import pyplot as plt
url="https://gist.githubusercontent.com/curran/a08a1080b88344b0c8a7/raw/0e7a9b0a5d22642a06d3d5t
iris=pd.read_csv(url)
iris
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa
...
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica


150 rows × 5 columns

```
iris1=iris.drop(['species'],axis='columns')
iris1.head()
```

	sepal_length	sepal_width	petal_length	petal_width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2


```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(iris1,iris[['species']],train_size=0.9)
```

```
X_test.head()
```



	sepal_length	sepal_width	petal_length	petal_width
52	6.9	3.1	4.9	1.5
145	6.7	3.0	5.2	2.3
85	6.0	3.4	4.5	1.6
149	5.9	3.0	5.1	1.8
45	4.8	3.0	1.4	0.3

```
from sklearn.svm import SVC
model=SVC()
model.fit(X_train,y_train)
```




```
/usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:1408: DataConv
y = column_or_1d(y, warn=True)
```

▼ SVC ⓘ ?

SVC()

```
model.predict([[5.4,3.7,1.5,0.2]])
```



```
/usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but SVC
warnings.warn(
array(['setosa'], dtype=object)
```