
Group A

Assignment No: 4

Contents for Theory:

1. Linear Regression : Univariate and Multivariate

2. Least Square Method for Linear Regression

3. Measuring Performance of Linear Regression

4. Example of Linear Regression

5. Training data set and Testing data set

---

1. **Linear Regression:** It is a machine learning algorithm based on supervised learning. It targets prediction values on the basis of independent variables.
    - It is preferred to find out the relationship between forecasting and variables.
    - A linear relationship between a dependent variable (X) is continuous; while independent variable(Y) relationship may be continuous or discrete. A linear relationship should be available in between predictor and target variable so known as Linear Regression.
    - Linear regression is popular because the cost function is Mean Squared Error (MSE) which is equal to the average squared difference between an observation's actual and predicted values.
    - It is shown as an equation of line like :

      Y = m*X + b + e

      Where : b is intercepted, m is slope of the line and e is error term.

    This equation can be used to predict the value of target variable Y based on given predictor variable(s) X, as shown in Fig. 1.

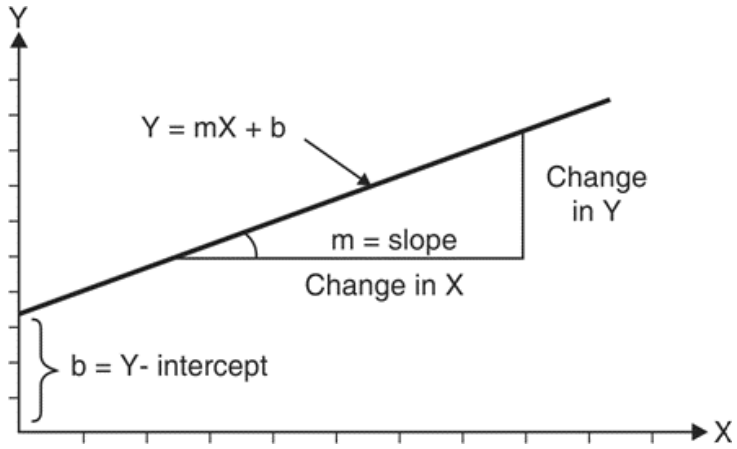
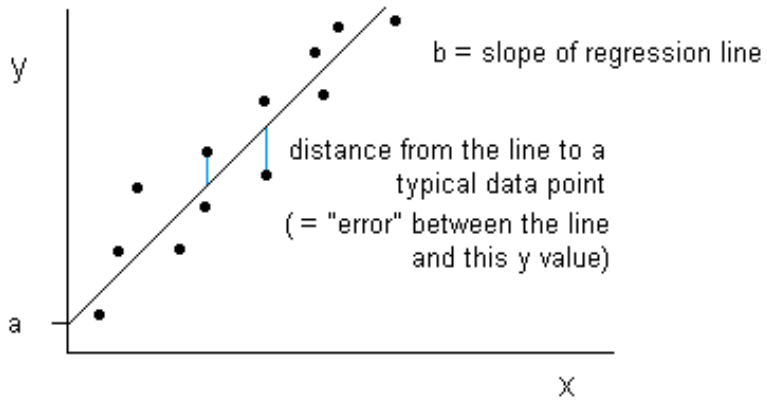


Fig. 1: geometry of linear regression

- Fig. 2 shown below is about the relation between weight (in Kg) and height (in cm), a linear relation. It is an approach of studying in a statistical manner to summarise and learn the relationships among continuous (quantitative) variables.
- Here a variable, denoted by 'x' is considered as the predictor, explanatory, or independent variable.
- Another variable, denoted 'y', is considered as the response, outcome, or dependent variable. While "predictor" and "response" used to refer to these variables.
- Simple linear regression technique concerned with the study of only one predictor variable.

Fig.2 : Relation between weight (in Kg) and height (in cm)



MultiVariate Regression :It concerns the study of two or more predictor variables. Usually a transformation of the original features into polynomial features from a given degree is preferred and further Linear Regression is applied on it.

- A simple linear model Y = a + bX is in original feature will be transformed into polynomial feature is transformed and further a linear regression applied to it and it will be something like

  $$Y=a + bX + cX2$$

- If a high degree value is used in transformation the curve becomes over-fitted as it captures the noise from data as well.

2. Least Square Method for Linear Regression

- Linear Regression involves establishing linear relationships between dependent and independent variables. Such a relationship is portrayed in the form of an equation also known as the linear model.
- A simple linear model is the one which involves only one dependent and one independent variable. Regression Models are usually denoted in Matrix Notations.
- However, for a simple univariate linear model, it can be denoted by the regression equation

$$\hat{y} = \beta_0 + \beta_1 x \qquad (1)$$

where $\hat{y}$ is the dependent or the response variable

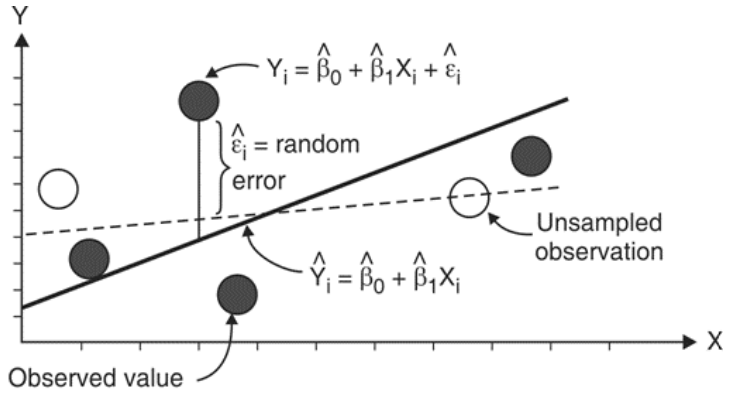
x is the independent or the input variable

β_0 is the value of y when x=0 or the y intercept

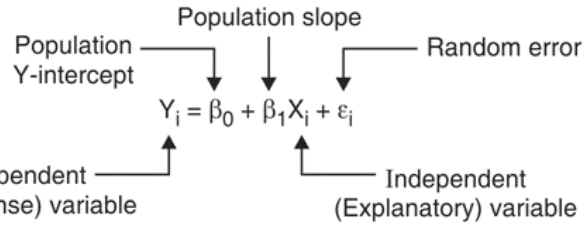
β_1 is the value of slope of the line ε is the error or the noise

- This linear equation represents a line also known as the ‘regression line’. The least square estimation technique is one of the basic techniques used to guess the values of the parameters and based on a sample set.
- This technique estimates parameters β_0 and β_1 and by trying to minimise the square of errors at all the points in the sample set. The error is the deviation of the actual sample
- data point from the regression line. The technique can be represented by the equation.

$$min \sum_{i=0}^{n} (\hat{y} - y)^2 \qquad (2)$$



Where,



Using differential calculus on equation 1 we can find the values of β_0 and β_1 such that the sum of squares (that is equation 2) is minimum.

$$\beta_1 = \sum_{i=1}^{n} (x_i - \overline{x})(y_i - \overline{y}) / \sum_{i=1}^{n} (x_i - \overline{x})^2 \qquad (3)$$

$$\beta_0 = \overline{y} - \beta_1 \overline{x} \qquad (4)$$

Once the Linear Model is estimated using equations (3) and (4), we can estimate the value of the dependent variable in the given range only. Going outside the range is called extrapolation which is inaccurate if simple regression techniques are used.

3. Measuring Performance of Linear Regression

Mean Square Error:

The Mean squared error (MSE) represents the error of the estimator or predictive model created based on the given set of observations in the sample. Two or more regression models created using a given sample data can be compared based on their MSE. The lesser the MSE, the better the regression model is. When the linear regression model is trained using a given set of observations, the model with the least mean sum of squares error (MSE) is selected as the best model. The Python or R packages select the best-fit

model as the model with the lowest MSE or lowest RMSE when training the linear regression models.

Mathematically, the MSE can be calculated as the average sum of the squared difference between the actual value and the predicted or estimated value represented by the regression model (line or plane).

$$MSE = \frac{1}{n}\Sigma\underbrace{\left(y - \hat{y}\right)}^{2}$$

The square of the difference between actual and predicted

An MSE of zero (0) represents the fact that the predictor is a perfect predictor.

RMSE:

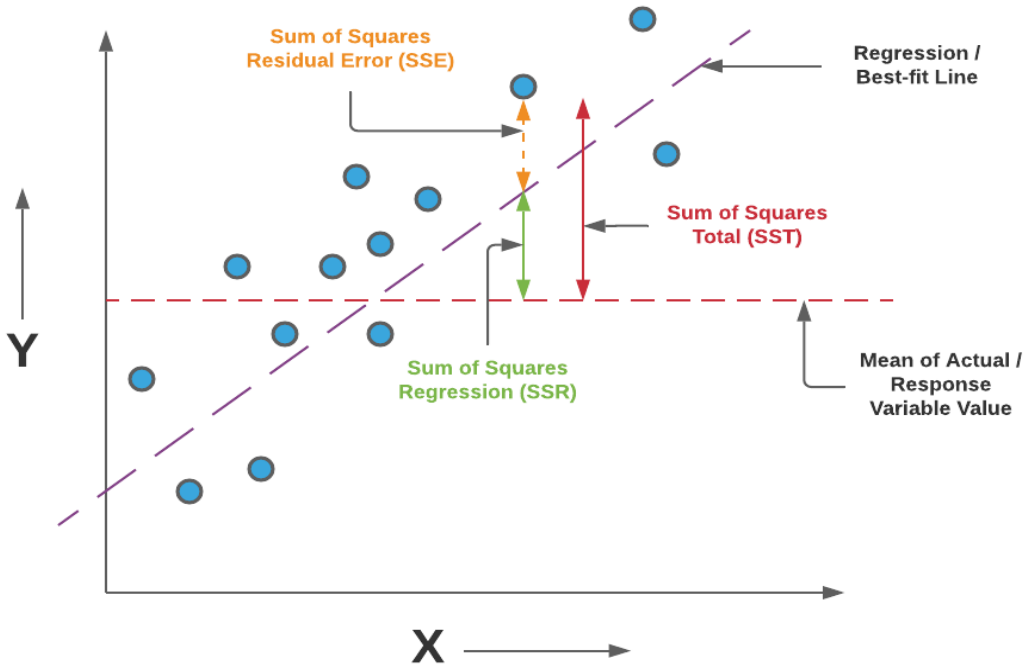
Root Mean Squared Error method that basically calculates the least-squares error and takes a root of the summed values.

Mathematically speaking, Root Mean Squared Error is the square root of the sum of all errors divided by the total number of values. This is the formula to calculate RMSE

$$\text{RMSE} = \sqrt{\sum_{i=1}^{n}\frac{1}{n}(\hat{y}_i - y_i)^2}$$

RMSE - Least Squares Regression Method - Edureka

R-Squared :



R-Squared is the ratio of the sum of squares regression (SSR) and the sum of squares total (SST).

SST : total sum of squares (SST), regression sum of squares (SSR), Sum of square of errors (SSE) are all showing the variation with different measures.

$$SST = \sum_{i=1}^{} (y_i - \bar{y})^2$$

$$SSR = \sum_{i=1}^{n} (\hat{y}_i - \bar{y})^2$$

$$SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

$$R^2 = \frac{SSR}{SST} = \frac{\sum (\hat{y}_i - \bar{y})^2}{\sum (y_i - \bar{y})^2}$$

A value of R-squared closer to 1 would mean that the regression model covers most part of the variance of the values of the response variable and can be termed as a good model.

One can alternatively use MSE or R-Squared based on what is appropriate and the need of the hour. However, the disadvantage of using MSE rather than R-squared is that it will be difficult to gauge the performance of the model using MSE as the value of MSE can vary from 0 to any larger number. However, in the case of R-squared, the value is bounded between 0 and .

4. Example of Linear Regression

Consider following data for 5 students.

Each Xi (i = 1 to 5) represents the score of ith student in standard X and corresponding Yi (i = 1 to 5) represents the score of ith student in standard XII.

(i) Linear regression equation best predicts standard XIIth score

(ii) Interpretation for the equation of Linear Regression

(iii) If a student's score is 80 in std X, then what is his expected score in XII standard?

Student	Score in X standard (Xi)	Score in XII standard (Yi)
1	95	85
2	85	95
3	80	70
4	70	65
5	60	70

x	y	$x - \bar{x}$	$y - \bar{y}$	$(x - \bar{x})^2$	$(x - \bar{x})(y - \bar{y})$
95	85	17	8	289	136
85	95	7	18	49	126
80	70	2	-7	4	-14
70	65	-8	-12	64	96

60	70	-18	-7	324	126
$\bar{x} = 78$	$\bar{y} = 77$			$\mathbf{\Sigma (x - \bar{x})^2 = 730}$	$\mathbf{\Sigma (x - \bar{x})(y - \bar{y}) = 470}$

(i) linear regression equation that best predicts standard XIIth score

$$\hat{y} = \beta_0 + \beta_1 x$$

$$\beta_1 = \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) / \sum_{i=1}^{n} (x_i - \bar{x})^2$$

$$\beta_1 = 470/730 = 0.644$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

$$\beta_0 = 77 - (0.644 * 78) = 26.768$$

$$\hat{y} = 26.76 + 0.644 x$$

(ii) **Interpretation of the regression line.**

Interpretation 1

For an increase in value of x by 0.644 units there is an increase in value of y in one unit.

Interpretation 2

Even if x = 0 value of independent variable, it is expected that value of y is 26.768

Score in XII standard (Yi) is 0.644 units depending on Score in X standard (Xi) but other factors will also contribute to the result of XII standard by 26.768 .

(iii) **If a student's score is 65 in std X, then his expected score in XII standard is 78.288**

For x = 80 the y value will be

$$\hat{y} = 26.76 + 0.644 * 65 = 68.38$$

5. Training data set and Testing data set

- Machine Learning algorithm has two phases
  1. Training and 2. Testing.
- The input of the training phase is training data, which is passed to any machine learning algorithm and machine learning model is generated as output of the training phase.
- The input of the testing phase is test data, which is passed to the machine learning model and prediction is done to observe the correctness of mode.

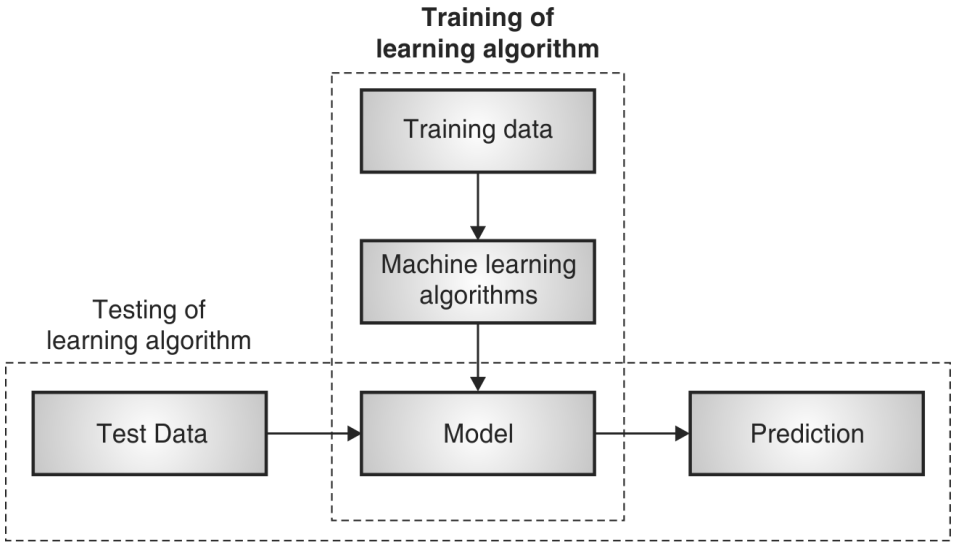


Fig. 1.3.1 : Training and Testing Phase in Machine Learning

(a) Training Phase

- Training dataset is provided as input to this phase.
- Training dataset is a dataset having attributes and class labels and used for training Machine Learning algorithms to prepare models.
- Machines can learn when they observe enough relevant data. Using this one can model algorithms to find relationships, detect patterns, understand complex problems and make decisions**.**
- Training error is the error that occurs by applying the model to the same data from which the model is trained**.**
- In a simple way the actual output of training data and predicted output of the model does not match the training error E_{in} is said to have occurred.
- Training error is much easier to compute.

(b) Testing Phase

- Testing dataset is provided as input to this phase.
- Test dataset is a dataset for which class label is unknown. It is tested using model
- A test dataset used for assessment of the finally chosen model.

- Training and Testing dataset are completely different.
- Testing error is the error that occurs by assessing the model by providing the unknown data to the model.
- In a simple way the actual output of testing data and predicted output of the model does not match the testing error E_{out} is said to have occurred.
- E_{out} is generally observed larger than E_{in}.

(c) Generalization

- Generalization is the prediction of the future based on the past system.
- It needs to generalize beyond the training data to some future data that it might not have seen yet.
- The ultimate aim of the machine learning model is to minimize the generalization error.
- The generalization error is essentially the average error for data the model has never seen.
- In general, the dataset is divided into two partition training and test sets.
- The fit method is called on the training set to build the model.
- This fit method is applied to the model on the test set to estimate the target value and evaluate the model's performance.
- The reason the data is divided into training and test sets is to use the test set to estimate how well the model trained on the training data and how well it would perform on the unseen data.

Algorithm (Synthesis Dataset):

Step 1: Import libraries and create alias for Pandas, Numpy and Matplotlib

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Step 2: Create a Dataframe with Dependent Variable(x) and independent variable y.

```
x=np.array([95,85,80,70,60])
y=np.array([85,95,70,65,70])
```

Step 3 : Create Linear Regression Model using Polyfit Function:

```
model= np.polyfit(x, y, 1)
```

Step 4: Observe the coefficients of the model.

```
model
```

Output:

```
array([ 0.64383562, 26.78082192])
```

Step 5: Predict the Y value for X and observe the output.

```
predict = np.poly1d(model)
predict(65)
```

Output:

```
68.63
```

Step 6: Predict the y_pred for all values of x.

```
y_pred=  predict(x)
y_pred
```

Output:

```
array([81.50684932, 87.94520548, 71.84931507, 68.63013699, 71.84931507])
```

Step 7: Evaluate the performance of Model (R-Suare)

R squared calculation is not implemented in numpy… so that one should be borrowed from sklearn.

```
from sklearn.metrics import r2_score
r2_score(y, y_pred)
```

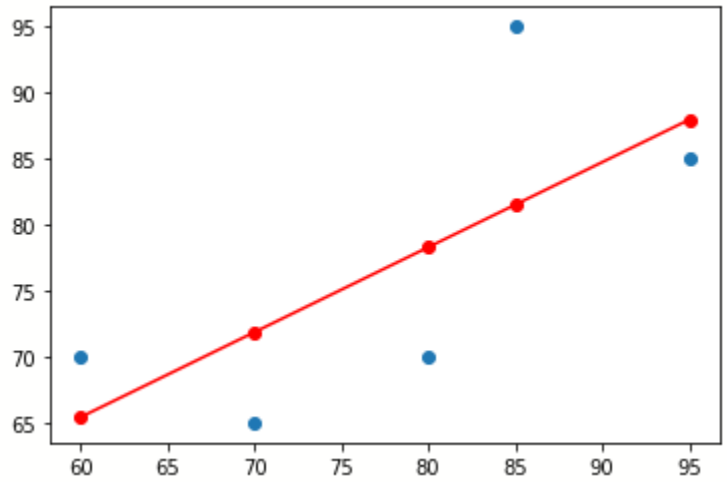
Output:

```
0.4803218090889323
```

Step 8: Plotting the linear regression model

```
y_line = model[1] + model[0]* x
plt.plot(x, y_line, c = 'r')
plt.scatter(x, y_pred)
plt.scatter(x,y,c='r')
```

Output:



Algorithm (Boston Dataset):

Step 1: Import libraries and create alias for Pandas, Numpy and Matplotlib

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Step 2: Import the Boston Housing dataset

```
from sklearn.datasets import load_boston
boston = load_boston()
```

Step 3: Initialize the data frame

```
data = pd.DataFrame(boston.data)
```

Step 4: Add the feature names to the dataframe

```
data.columns = boston.feature_names
data.head()
```

Step 5: Adding target variable to dataframe

```
data['PRICE'] = boston.target
```

Step 6: Perform Data Preprocessing(Check for missing values)

```
data.isnull().sum()
```

Step 7: Split dependent variable and independent variables

```
x = data.drop(['PRICE'], axis = 1)
y = data['PRICE']
```

Step 8: splitting data to training and testing dataset.

```
from sklearn.model_selection import train_test_split
xtrain, xtest, ytrain, ytest =
train_test_split(x, y, test_size =0.2,random_state = 0)
```

Step 9: Use linear regression(Train the Machine) to Create Model

```
import sklearn
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
model=lm.fit(xtrain, ytrain)
```

Step 10: Predict the y_pred for all values of train_x and test_x

```
ytrain_pred = lm.predict(xtrain)
ytest_pred = lm.predict(xtest)
```

Step 11:Evaluate the performance of Model for train_y and test_y

```
df=pd.DataFrame(ytrain_pred,ytrain)
df=pd.DataFrame(ytest_pred,ytest)
```

Step 12: Calculate Mean Square Paper for train_y and test_y

```
from sklearn.metrics import mean_squared_error, r2_score
mse = mean_squared_error(ytest, ytest_pred)
print(mse)
mse = mean_squared_error(ytrain_pred,ytrain)
```

```
print(mse)
```

Output:

```
33.44897999767638
```

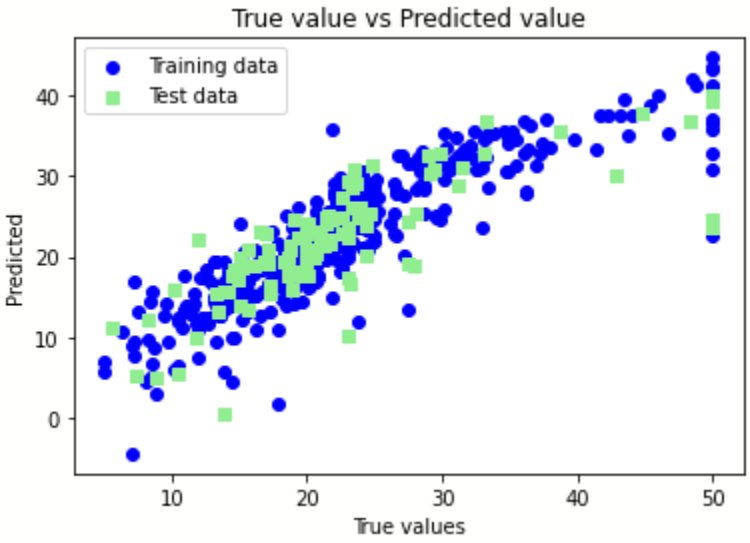
```
mse = mean_squared_error(ytest, ytest_pred)
print(mse)
```

Output:

```
19.32647020358573
```

Step 13: Plotting the linear regression model

```
lt.scatter(ytrain ,ytrain_pred,c='blue',marker='o',label='Training data')
plt.scatter(ytest,ytest_pred ,c='lightgreen',marker='s',label='Test data')
plt.xlabel('True values')
plt.ylabel('Predicted')
plt.title("True value vs Predicted value")
plt.legend(loc= 'upper left')
#plt.hlines(y=0,xmin=0,xmax=50)
plt.plot()
plt.show()
```



Conclusion:

In this way we have done data analysis using linear regression for Boston Dataset and predict the price of houses using the features of the Boston Dataset.

Assignment Question:

1) **Compute SST, SSE, SSR, MSE, RMSE, R Square for the below example .**

Student	Score in X standard (Xi)	Score in XII standard (Yi)

1	95	85
2	85	95
3	80	70
4	70	65
5	60	70

2) **Comment on whether the model is best fit or not based on the calculated values.**
3) **Write python code to calculate the RSquare for Boston Dataset.**
   (Consider the linear regression model created in practical session)

.