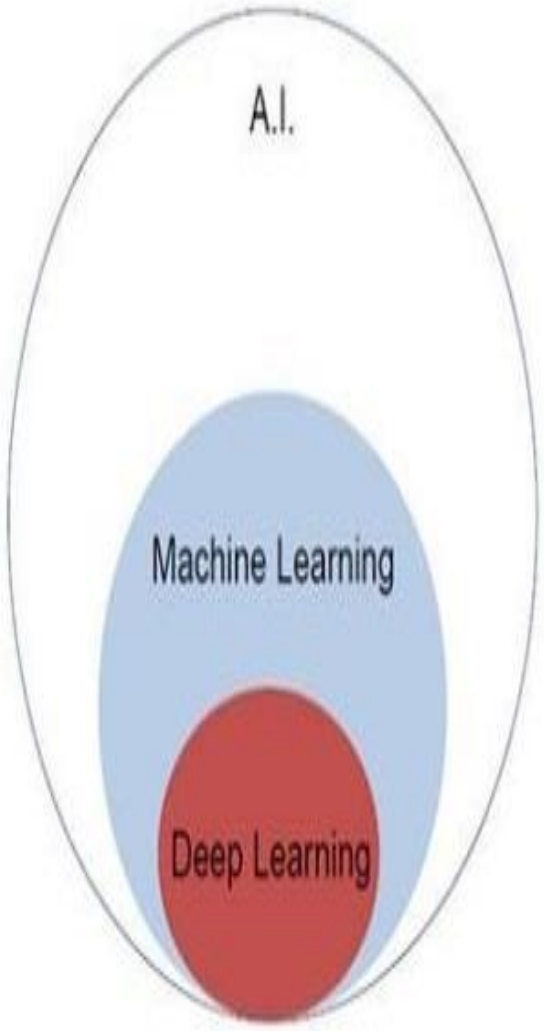
Machine Learning using Python

Introduction

- Artificial Intelligence includes the simulation process of human intelligence by computer systems. Examples of artificial intelligence include learning, reasoning and self-correction. Applications of AI include speech recognition, expert systems, image recognition and computer vision.

- Machine learning is the branch of artificial intelligence, which deals with systems and algorithms that can learn any new data and data patterns. A machine-learning system is trained rather than explicitly programmed.

- Deep learning is a part of machine learning.



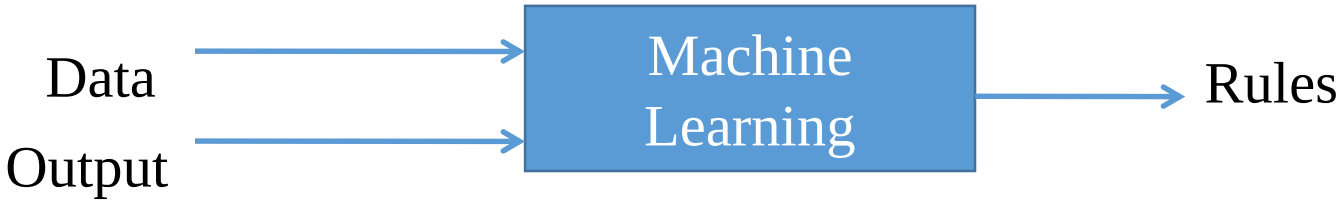
Introduction to ML

- “Machine Learning is the field of study which gives computers the ability to learn without being explicitly programmed.”

  -- Arthur Samuels, 1959

- **Traditional Programming:**

- **Machine Learning:**



Machine Learning Model

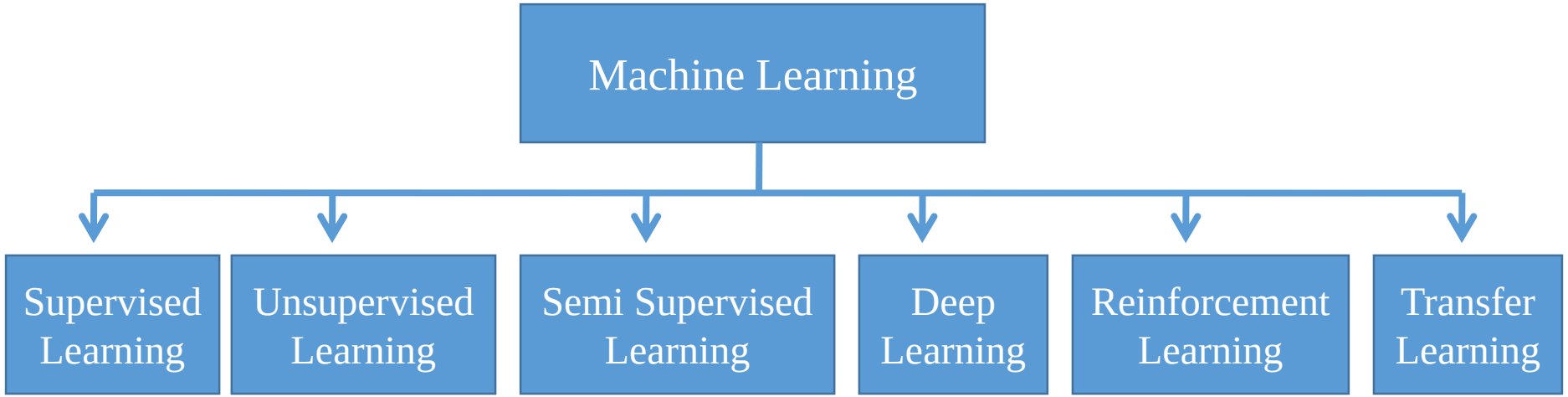
- “A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.”

  -- Tom Michells, 1997

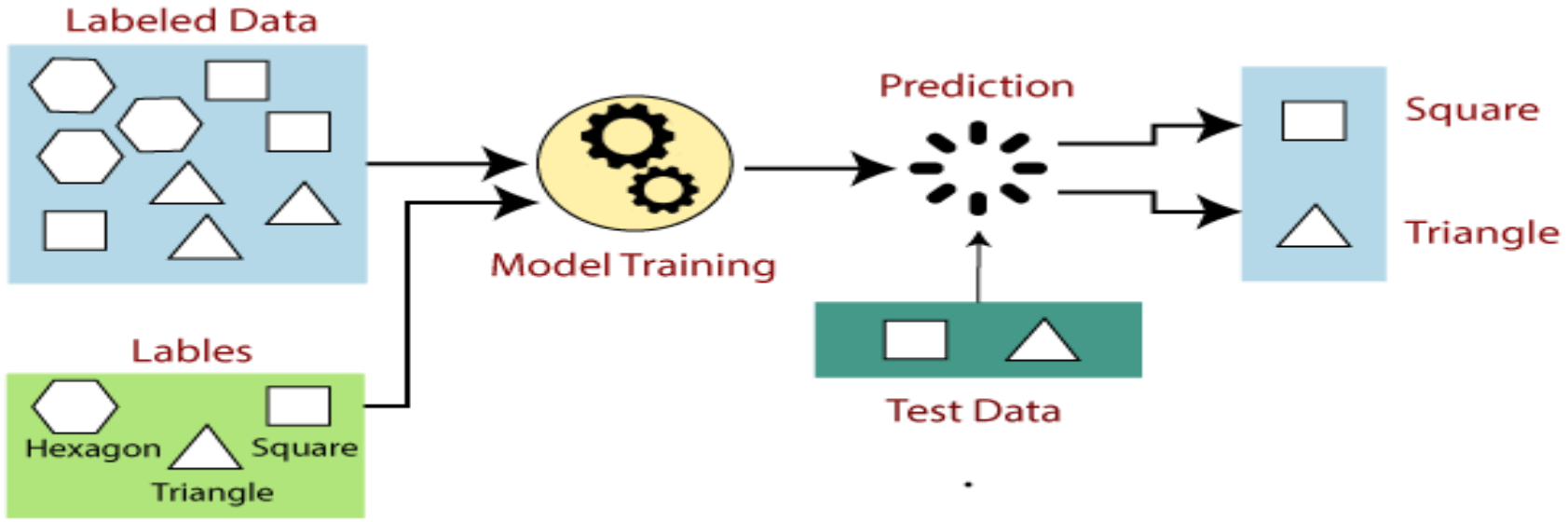
- The main components of any Machine Learning algorithm: **Task(T), Performance(P) and experience (E).**
  - **Task:** We may define the task (T) as any real-world problem that needs to be solved. The examples of ML based tasks are Classification, Regression, Clustering and so on.
  - **Experience:** It is the knowledge gained from inputs provided to the algorithm or model. Once provided with the dataset, the model will run iteratively and try to learn some inherent pattern. The learning thus acquired is called experience(E).
  - **Performance:** It is the measure which tells whether ML algorithm is performing as per expectation or not. (P) is basically a quantitative metric that tells how a model is performing the task, T, using its experience, E. Metrics that help to understand the ML performance are F1 score, confusion matrix, precision, recall and so on.

Types of ML Algorithms

- Machine Learning Algorithms are broadly classified into the following categories:



Supervised Learning



- Supervised learning includes a procedure where the training set and its associated output i.e. labels are given as input to the system.
- After completion of training, the accuracy of each model is measured with respect to the test set or validation set, which is complete disjoint from the training set.

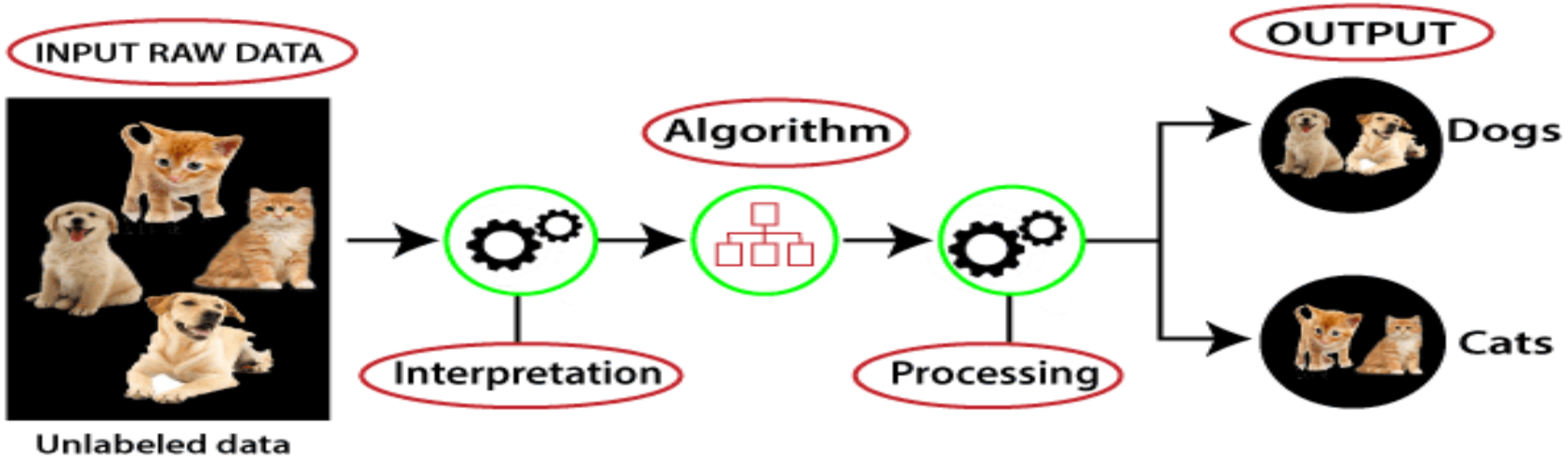
Supervised Learning

- Supervised learning can be grouped further in two categories of algorithms:

1. **Classification:** The key objective of classification problem is to classify objects of similar type. Classification algorithms are used when the output variable is categorical, which means there are two classes such as Yes-No, Male-Female, True-false, etc. Example: Naïve Bayes Classification, Decision Tree, Logistic Regression, Support Vector Machine(SVM) and so on

2. **Regression:** Regression algorithms are used if there is a relationship between the input variable and the output variable. It is used for the prediction of continuous variables, such as Weather forecasting, Market Trends, etc. Example: Linear Regression

Unsupervised Learning



- Unsupervised learning includes a procedure where the training data are not labelled.
- In unsupervised learning, we don't have a predetermined result. The machine tries to find useful insights from the data.

Unsupervised Learning

- Unsupervised learning can be grouped further in two categories of algorithms:

1. **Clustering:** Clustering is a method of grouping the objects into clusters such that objects with most similarities remains into a group and has less or no similarities with the objects of another group. Example: K-Means, K-Nearest Neighbour(KNN), Hierarchical clustering and so on.

2. **Association:** An association rule is an unsupervised learning method which is used for finding the relationships between variables in the large database.

Semi-supervised Learning

- The most basic disadvantage of any Supervised Learning algorithm is that the dataset has to be hand-labelled either by a Machine Learning Engineer or a Data Scientist. This is a very costly process, especially when dealing with large volumes of data. The most basic disadvantage of any Unsupervised Learning is that it’s application spectrum is limited.
- To counter these disadvantages, the concept of Semi-Supervised Learning was introduced. Semi-supervised learning uses a combination of a small amount of labelled data and a large amount of unlabeled data to train models. This approach combines both the concept of Supervised and Unsupervised Learning.
- Example: Text document classifier. This is the type of situation where semi-supervised learning is ideal because it would be nearly impossible to label all the words in a text document.

Loading and Processing of Data

- **Header Files:**

  import pandas as pd

  import numpy as np

  import matplotlib.pyplot as plt

  import seaborn as sns

- **Reading a CSV File in a DataFrame :**

  df=pd.read_csv(“xyz.csv")

  print(df) //print the entire CSV file as a DataFrame

  print(df.shape) //display no of rows and columns present in the CSV file

  print(df.info())// shows different data types of all the columns of the file

  print(df.describe()) // it shows different statistical information of the CSV file such as: mean, standard deviation, minimum, maximum

  print(df.head()) // print first five rows from the CSV file

Pre-Processing of Data

print(df.isnull().sum()) // Check for any missing values in the dataset

Regression Analysis

- Regression analysis is a statistical method to model the relationship between a dependent (target) and one or more independent (predictor) variables.
- More specifically, Regression analysis helps us to understand how the value of the dependent variable is changing corresponding to an independent variable when other independent variables are held fixed.

- **Terminologies Related to the Regression Analysis:**

➢ **Dependent Variable:** The main factor in Regression analysis which we want to predict or understand is called the dependent variable. It is also called **target variable**.

➢ **Independent Variable:** The factors which affect the dependent variables or which are used to predict the values of the dependent variables are called independent variable, also called as a **predictor**.

Regression Analysis

- **Outliers:** Outlier is an observation which contains either very low value or very high value in comparison to other observed values. An outlier may hamper the result, so it should be avoided.

- **Multicollinearity:** If the independent variables are highly correlated with each other than other variables, then such condition is called Multicollinearity. It should not be present in the dataset, because, when independent variables are correlated, it indicates that changes in one variable are associated with shifts in another variable. The stronger the correlation, the more difficult it is to change one variable without changing another.

- **Under-fitting and Over-fitting:** If our algorithm works well with the training dataset but not well with test dataset, then such problem is called **Over-fitting**. And if our algorithm does not perform well even with training dataset, then such problem is called **under-fitting**.

Linear Regression

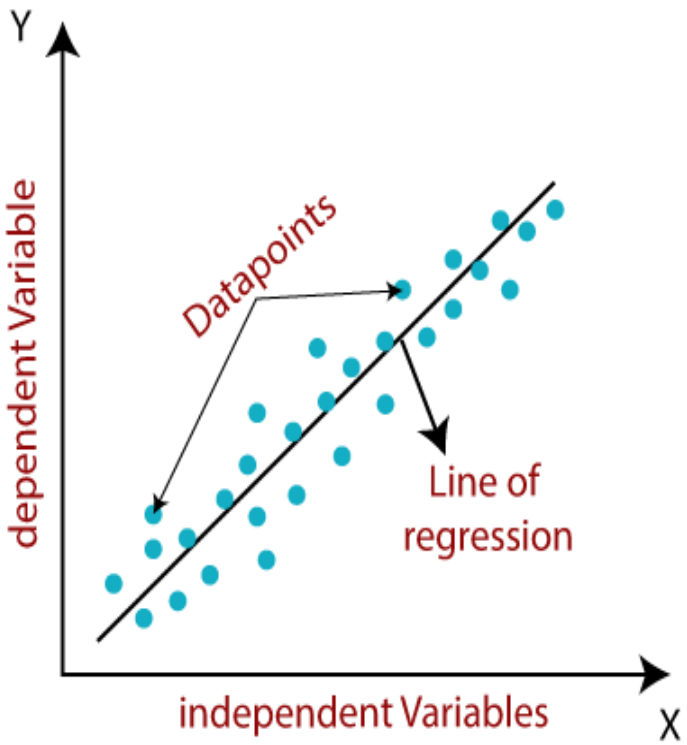
- Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (x) variables. Which means it finds how the value of the dependent variable is changing according to the value of the independent variables.

- Mathematically, we can represent a linear regression as:

$$y= a_0+a_1x_1+a_2x_2+\ldots+a_nx_n+ \varepsilon$$

- Here,

  y = Dependent Variable (Target Variable)
  $x_1,x_2,x_3,\ldots, x_n$ = Independent Variables (Predictor Variables)
  a_0 = intercept of the line (Gives an additional degree of freedom)
  $a_1,a_2,a_3,\ldots,a_n$ = Linear regression coefficients
  ε = random error



Best Fit Line in Linear Regression

- The main objective of Linear Regression model is to find the best fit line that should minimize the error between predicted values and actual values.
- **Cost Function:**

➢ Cost function optimizes the regression coefficients or weights.

➢ The different values for weights or coefficient of lines (a_0, a_1) gives the different line of regression, and the cost function is used to estimate the values of the coefficient for the best fit line.

➢ Here we use the **Mean Squared Error (MSE)** cost function, which is the average of squared error occurred between the predicted values and actual values.

➢ MSE :

$$MSE = \frac{1}{N}\sum_{i=1}^{n}(y_i - (a_1 x_i + a_0))^2$$

Where, N =Total number of observation

yi = Actual value

($a_1 x_i + a_0$)= Predicted value.

Best Fit Line in Linear Regression

- **Residuals:** The distance between the actual value and predicted values is called residual. If the observed points are far from the regression line, then the residual will be high, and so cost function will high. If the scatter points are close to the regression line, then the residual will be small and hence the cost function.

- **Gradient Descent:**

- Gradient descent is used to minimize the MSE by calculating the gradient of the cost function.
- A regression model uses gradient descent to update the coefficients of the line by reducing the cost function.
- It is done by a random selection of values of coefficient and then iteratively update the values to reach the minimum cost function.

Performance of Linear Regression

- The Goodness of fit determines how the line of regression fits the set of observations.
- **R-squared method:**

➢ R-squared is a statistical method that determines the goodness of fit.

➢ The high value of R-square determines the less difference between the predicted values and actual values and hence represents a good model.

➢ It can be calculated from the below formula:

$$R^2 = 1 - \frac{SS_r}{SS_t}$$

SS_t : is the total sum of errors if we take the mean of the observed values as the predicted value.

$$SS_t = \sum_{i=1}^{m} (y^i - \bar{y})^2$$ ss$_t$ = np.sum((y_actual - np.mean(y_actual))**2)

SS_r is the sum of the square of residuals. ss$_r$ = np.sum((y_pred - y_actual)**2)

$$SS_r = \sum_{i=1}^{m} (h(x^i) - y^i)^2$$

Simple Linear Regression

- Simple Linear Regression is a type of Regression that models the relationship between a dependent variable and a single independent variable.
- It can be mathematically represented as:

$$\mathbf{y= a_0+a_1x+ \varepsilon}$$

- Where,

  a0= It is the intercept of the Regression line (can be obtained putting x=0)
  a1= It is the slope of the regression line,

  ε = The error term. (For a good model it will be negligible)

Implementation of Simple Linear Regression using Python

- Step-1: Read Data :

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection
    import train_test_split
from sklearn.linear_model import
    LinearRegression
df= pd.read_csv('salary_data.csv')
print(df)
```

	YearsExperience	Salary
0	1.1	39343
1	1.3	46205
2	1.5	37731
3	2.0	43525
4	2.2	39891
5	2.9	56642
6	3.0	60150
7	3.2	54445
8	3.2	64445
9	3.7	57189
10	3.9	63218
11	4.0	55794
12	4.0	56957
13	4.1	57081
14	4.5	61111
15	4.9	67938
16	5.1	66029

Implementation of Simple Linear Regression using Python

- Step-2: Pre-processing Data :

```
x= data_set.iloc[:,:-1].values  # It selects last column with all rows
y= data_set.iloc[:, 1].values  # It selects column-1 with all rows
```

- Step-3: Splitting the dataset into training and test set:

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 1/3, rando
                                                                    m_state=0)
```

test_size=1/3: Here we will split our dataset (30 observations) into 2 parts (training set, test set) and the ratio of test set compare to dataset is 1/3 (10 observations will be put into the test set).

random_state=0: This is the seed for the random number generator. If we leave it blank or 0, then np.random is used for randomly choosing the elements for training set and test set.

Implementation of Simple Linear Regression using Python

- Step-4: Build the Regression Model:

from sklearn.linear_model import LinearRegression # Here we are importing the LinearRegression class of the linear_model library from the scikit learn.

linreg= LinearRegression()

linreg.fit(x_train, y_train)

- Step-5: Prediction of Test set Result:

test_pred= linreg.predict(x_test)

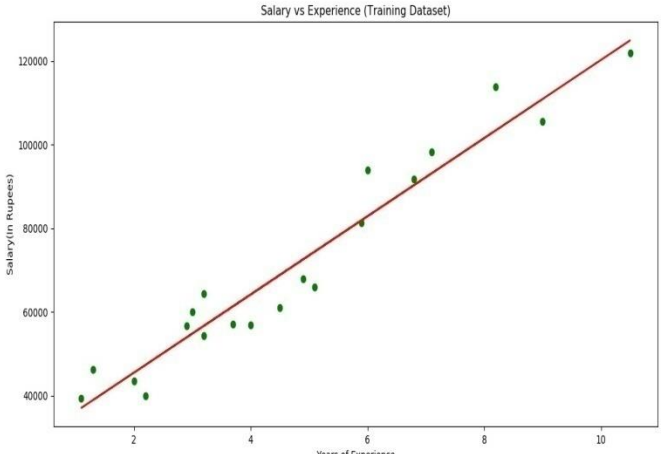
train_pred= linreg.predict(x_train)

Here we create a prediction vector test_pred, and train_pred, which contains predictions of test dataset, and training dataset respectively.

Implementation of Simple Linear Regression using Python

- Step-6: Visualizing the training set results:

```
plt.scatter(x_train, y_train, color="green")
plt.plot(x_train, train_pred, color="red")
plt.title("Salary vs Experience (Training set)")
plt.xlabel("Years of Experience")
plt.ylabel("Salary(In Rupees)")
```



- Step-7:Visualizing the test set results:

```
plt.scatter(x_test, y_test, color="blue")
plt.plot(x_train, train_pred, color="red")
plt.title("Salary vs Experience (Test set)")
plt.xlabel("Years of Experience")
plt.ylabel("Salary(In Rupees)")
```

Implementation of Simple Linear Regression using Python

- Step-8: Predicting the Result:

Method 1:

```
y_pred= linreg.predict(10)
print(y_pred) # 120275.61667525
```

Method 2:

```
print(linreg.coef_) # It gives the value of a1
print(linreg.intercept_) # It gives the value of a0
def cal_salary(x):
    y = ((x*linreg.coef_) + linreg.intercept_)
    return y
print(cal_salary(10)) # 120275.61667525
```

Both of them predicts the salary for 10 years experience