ABHINAV RANJAN
RA1911003010003
CSE A1 SECTION
SRMIST , KTR

# AI LAB EXP 8

# MACHINE LEARNING ALGORITHMS TO SOLVE REAL WORLD PROBLEMS

**PROBLEM STATEMENT :**

Develop a program to implement supervised and unsupervised ML algorithms on a given dataset

**TOOLS USED :** python3 , excel sheet (dataset) , google colab

**ALGORITHM :**

## LINEAR REGRESSION

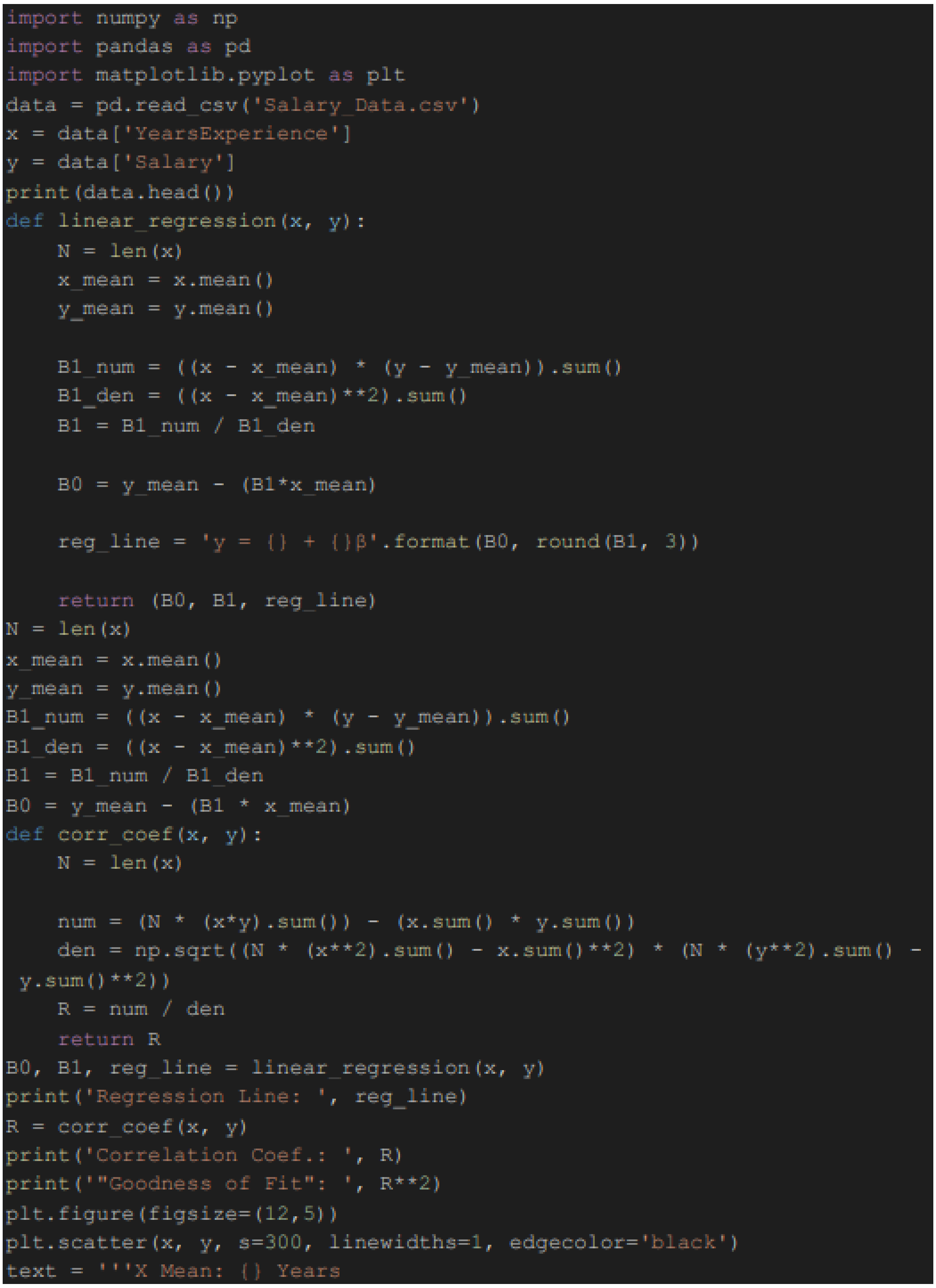
1. Import the packages and classes you need.
2. Provide data to work with and eventually do appropriate transformations ( convert categorical variable into dummy variable)
3. Create a regression model and fit it with existing data.
4. Check the results of model fitting to know whether the model is satisfactory.
5. Apply the model for predictions.

## CLUSTERING

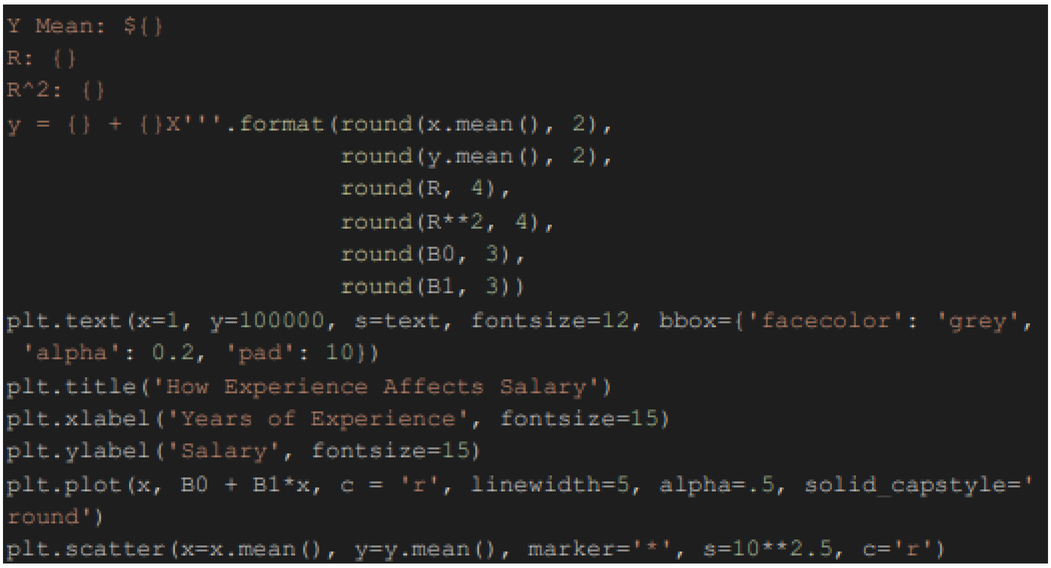
```
1) Setup arrays to store train and test accuracies
neighbors = np.arange(1, 9)
train_accuracy = np.empty(len(neighbors))
test_accuracy = np.empty(len(neighbors))
2) Loop over different values of k
for i, k in enumerate(neighbors):
  3) Setup a k-NN Classifier with k neighbors
  knn = KNeighborsClassifier(n_neighbors=k)
  4) Fit the classifier to the training data
  knn.fit(X_train, y_train)
  5) Compute accuracy on the training set
  train_accuracy[i] = knn.score(X_train, y_train)
  6) Compute accuracy on the testing set
  test_accuracy[i] = knn.score(X_test, y_test)
7) Generate plot
```

## SUPERVISED LEARNING :-

## CODE :-

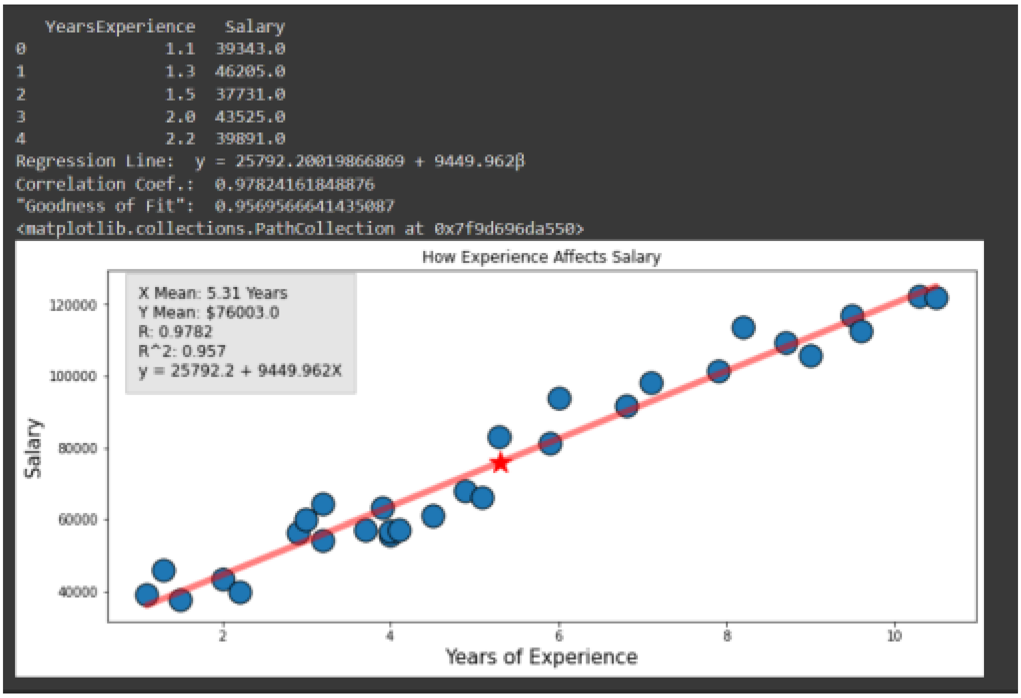


```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
data = pd.read_csv('Salary_Data.csv')
x = data['YearsExperience']
y = data['Salary']
print(data.head())
def linear_regression(x, y):
    N = len(x)
    x_mean = x.mean()
    y_mean = y.mean()

    B1_num = ((x - x_mean) * (y - y_mean)).sum()
    B1_den = ((x - x_mean)**2).sum()
    B1 = B1_num / B1_den

    B0 = y_mean - (B1*x_mean)

    reg_line = 'y = {} + {}β'.format(B0, round(B1, 3))

    return (B0, B1, reg_line)
N = len(x)
x_mean = x.mean()
y_mean = y.mean()
B1_num = ((x - x_mean) * (y - y_mean)).sum()
B1_den = ((x - x_mean)**2).sum()
B1 = B1_num / B1_den
B0 = y_mean - (B1 * x_mean)
def corr_coef(x, y):
    N = len(x)

    num = (N * (x*y).sum()) - (x.sum() * y.sum())
    den = np.sqrt((N * (x**2).sum() - x.sum()**2) * (N * (y**2).sum() -
 y.sum()**2))
    R = num / den
    return R
B0, B1, reg_line = linear_regression(x, y)
print('Regression Line: ', reg_line)
R = corr_coef(x, y)
print('Correlation Coef.: ', R)
print('"Goodness of Fit": ', R**2)
plt.figure(figsize=(12,5))
plt.scatter(x, y, s=300, linewidths=1, edgecolor='black')
text = '''X Mean: {} Years
```



```
Y Mean: ${}
R: {}
R^2: {}
y = {} + {}X'''.format(round(x.mean(), 2),
                       round(y.mean(), 2),
                       round(R, 4),
                       round(R**2, 4),
                       round(B0, 3),
                       round(B1, 3))
plt.text(x=1, y=100000, s=text, fontsize=12, bbox={'facecolor': 'grey',
 'alpha': 0.2, 'pad': 10})
plt.title('How Experience Affects Salary')
plt.xlabel('Years of Experience', fontsize=15)
plt.ylabel('Salary', fontsize=15)
plt.plot(x, B0 + B1*x, c = 'r', linewidth=5, alpha=.5, solid_capstyle='
round')
plt.scatter(x=x.mean(), y=y.mean(), marker='*', s=10**2.5, c='r')
```

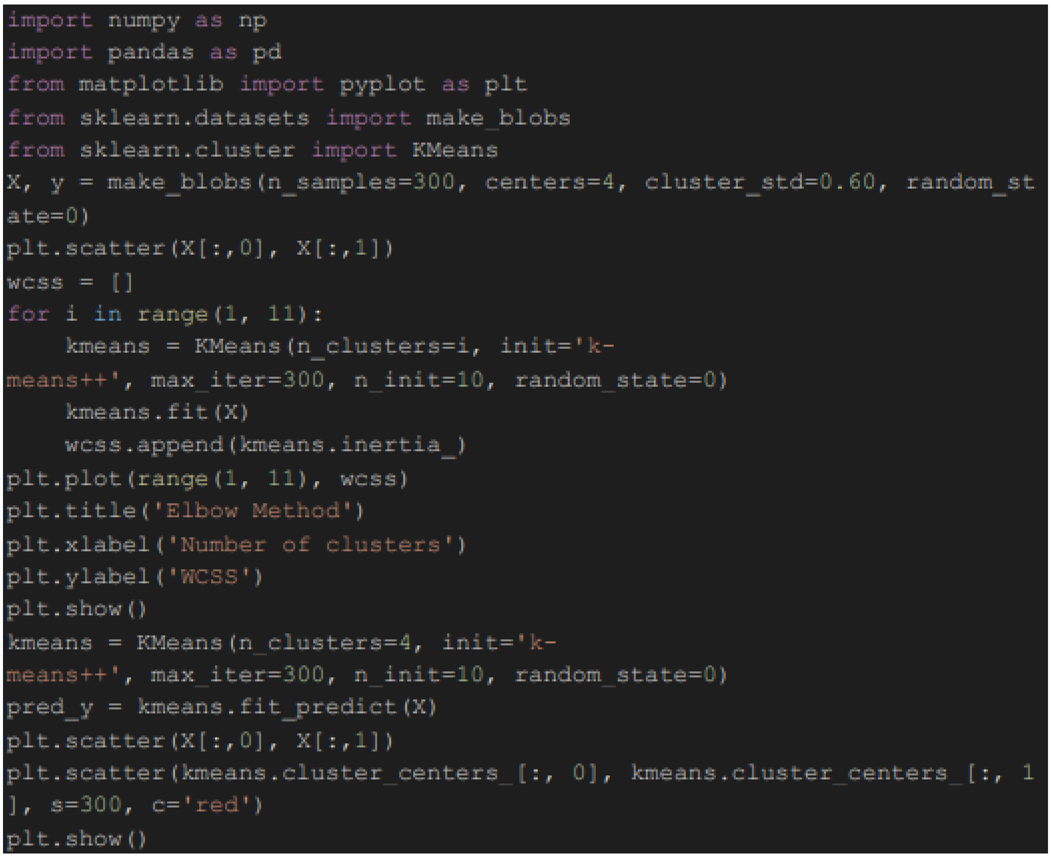
## OUTPUT:-



```
   YearsExperience   Salary
0              1.1  39343.0
1              1.3  46205.0
2              1.5  37731.0
3              2.0  43525.0
4              2.2  39891.0
Regression Line:  y = 25792.20019866869 + 9449.962β
Correlation Coef.:  0.97824161848876
"Goodness of Fit":  0.9569566641435087
<matplotlib.collections.PathCollection at 0x7f9d696da550>
```

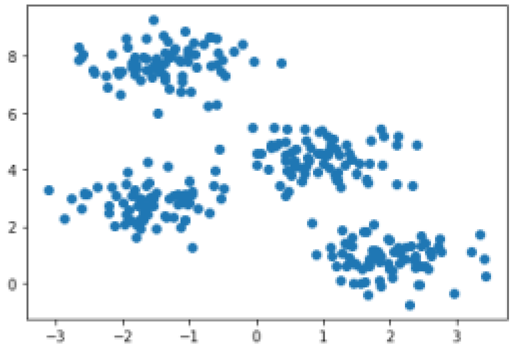
## UNSUPERVISED LEARNING:-

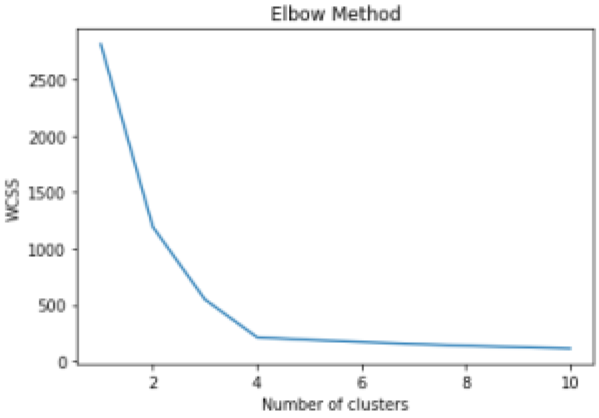
## CODE :-

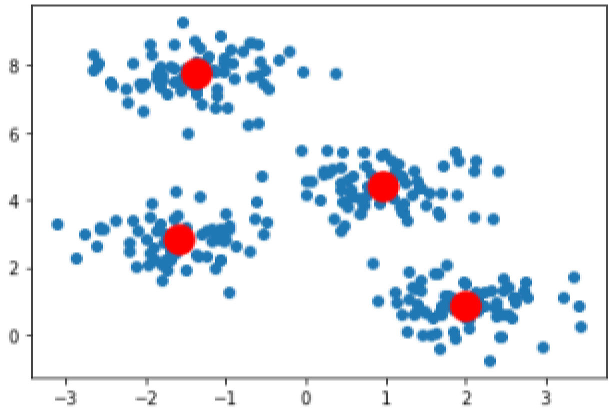


```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
from sklearn.datasets import make_blobs
from sklearn.cluster import KMeans
X, y = make_blobs(n_samples=300, centers=4, cluster_std=0.60, random_st
ate=0)
plt.scatter(X[:,0], X[:,1])
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-
means++', max_iter=300, n_init=10, random_state=0)
    kmeans.fit(X)
    wcss.append(kmeans.inertia_)
plt.plot(range(1, 11), wcss)
plt.title('Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
kmeans = KMeans(n_clusters=4, init='k-
means++', max_iter=300, n_init=10, random_state=0)
pred_y = kmeans.fit_predict(X)
plt.scatter(X[:,0], X[:,1])
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1
], s=300, c='red')
plt.show()
```

# OUTPUT:







## RESULT :

Thus we have successfully implemented supervised and unsupervised ML algorithms and obtained the desired results