**ML EXPERIMENT 4**

# Import libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model\_selection import train\_test\_split

from pandas.core.common import random\_state

from sklearn.linear\_model import LinearRegression

# Get dataset

df\_sal = pd.read\_csv('/content/Salary\_Data.csv')

df\_sal.head()



# Describe data

df\_sal.describe()

# Data distribution

plt.title('Salary Distribution Plot')

sns.distplot(df\_sal['Salary'])

plt.show()

# Relationship between Salary and Experience

plt.scatter(df\_sal['YearsExperience'], df\_sal['Salary'], color = 'lightcoral')

plt.title('Salary vs Experience')

plt.xlabel('Years of Experience')

plt.ylabel('Salary')

plt.box(False)

plt.show()

# Splitting variables

X = df\_sal.iloc[:, :1]  # independent

y = df\_sal.iloc[:, 1:]  # dependent

# Splitting dataset into test/train

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.2, random\_state = 0)

# Regressor model

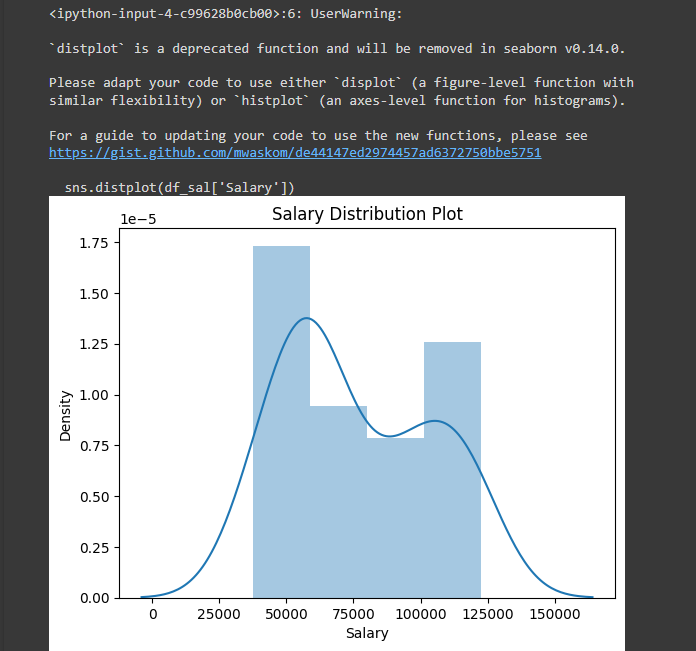
regressor = LinearRegression()

regressor.fit(X\_train, y\_train)

# Prediction result

y\_pred\_test = regressor.predict(X\_test)     # predicted value of y\_test

y\_pred\_train = regressor.predict(X\_train)   # predicted value of y\_train

# Prediction on training set

plt.scatter(X\_train, y\_train, color = 'lightcoral')

plt.plot(X\_train, y\_pred\_train, color = 'firebrick')

plt.title('Salary vs Experience (Training Set)')

plt.xlabel('Years of Experience')

plt.ylabel('Salary')

plt.legend(['X\_train/Pred(y\_test)', 'X\_train/y\_train'], title = 'Sal/Exp', loc='best', facecolor='white')

plt.box(False)

plt.show()



# Prediction on test set

plt.scatter(X\_test, y\_test, color = 'lightcoral')

plt.plot(X\_train, y\_pred\_train, color = 'firebrick')

plt.title('Salary vs Experience (Test Set)')

plt.xlabel('Years of Experience')

plt.ylabel('Salary')

plt.legend(['X\_train/Pred(y\_test)', 'X\_train/y\_train'], title = 'Sal/Exp', loc='best', facecolor='white')

plt.box(False)

plt.show()



# Regressor coefficients and intercept

print(f'Coefficient: {regressor.coef\_}')

print(f'Intercept: {regressor.intercept\_}')

