

```

import pandas as pd # data
import numpy as np # array

import matplotlib.pyplot as plt

# read data set
dataset = pd.read_csv(r"C:\Users\ammyg\Desktop\DATA SCIENCE\classroom material\statistic_workshop_ni

#create dependent and independent variable

x = dataset.iloc[:, :-1]
y = dataset.iloc[:, [-1]]

# we have already cleaned data so need for imputer
#decorator is what?
#decorater that lets us edit code and add additional future

from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.2,random_state = 0)

from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(x_train,y_train)

y_pred = regressor.predict(x_test)

plt.scatter(x_test,y_test,color = 'red') # real salary data
plt.plot(x_train,regressor.predict(x_train),color = 'blue')

plt.xlabel('years of exp')
plt.ylabel('salary')
plt.show()

#stats integration to ML
dataset.mean()
dataset['Salary'].mean()
dataset['Salary'].median()
dataset['Salary'].mode()
dataset.var
dataset['YearsExperience'].var()
dataset.std()

dataset['Salary'].std()

# corfficient of variance(cv)
from scipy.stats import variation
variation(dataset.values)

variation(dataset['Salary'])
dataset.corr()
#skewness
dataset.skew() # this will give skewness of entire column
dataset['Salary'].skew()

#standard error
dataset.sem()

```

```

# z score
import scipy.stats as stats
dataset.apply(stats.zscore)
stats.zscore(dataset['Salary'])

#degree of freedom
y_mean = np.mean(y)
ssr = np.sum((y_pred-y_mean)**2)
print(ssr)

#sse
y =y[0:6]
sse = np.sum((y-y_pred)**2)
print(sse)

#sst
mean_total =np.mean(dataset.values)
sst = np.sum((dataset.values-mean_total)**2)
print(sst)
#r2
r_square = 1 - ssr/sst
print('r2:',r_square)

bias = regressor.score(x_train,y_train)
print('bias:',bias)

variance = regressor.score(x_test,y_test)
print('variance',variance)

#deploy to website
import pickle
import os

# Save the trained model
filename = 'linear_regression_model1.pkl' # Fixed typo
with open(filename, 'wb') as file: # Fixed 'bw' to 'wb'
    pickle.dump(regressor, file)

print(f'Model saved as {filename}')
print(f'Location: {os.getcwd()}')

```

