

**CSE303: Statistics for Data Science**

**[Spring 2022]**

**Project Report**

**Group No. – A**

**SEC - 2**

**Submitted by:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Student ID** | **Student Name** | **Contribution Percentage** | **Signature** |
| **2019-1-60-063** | **SazzadHossen** | **20** |  |
| **2019-1-60-006** | **Amit Roy** | **17.5** |  |
| **2018-3-60-079** | **Md Mahmud Hassan** | **15** |  |
| **2019-2-60-051** | **OrpaNuman** | **17.5** |  |
| **2019-2-60-054** | **Abdullah Ahnaf** | **15** |  |
| **2019-1-60-174** | **FarjanaYeasminLimu** | **15** |  |

**Submitted to :**

**Md Al-Imran**

# Introduction

In this project we tried to implement the inverse matrix without using any library function.To find the inverse of a 3x3 matrix, first calculate the determinant of the matrix. If the determinant is 0, the matrix has no inverse. Next, transpose the matrix by rewriting the first row as the first column, the middle row as the middle column, and the third row as the third column. But this inverse matrix is going to be correct only if the identity matrix is correct.

We implemented multiple linear equation by using normal algebraic method.Multivariate Multiple Linear Regression is used when there is one or more predictor variables with multiple values for each unit of observation.We tried to use least possible library function.It is the basic and commonly used type for predictive analysis. It is a statistical approach to modelling the relationship between a dependent variable and a given set of independent variables. These are of two types: Simple linear Regression and Multiple Linear Regression.

Let’s Discuss Multiple Linear Regression using Python. It attempts to model the relationship between two or more features and a response by fitting a linear equation to observed data. The steps to perform multiple linear Regression are almost similar to that of simple linear Regression. The Difference Lies in the evaluation. We can use it to find out which factor has the highest impact on the predicted output and now different variables relate to each other.

# Data Pre-processing

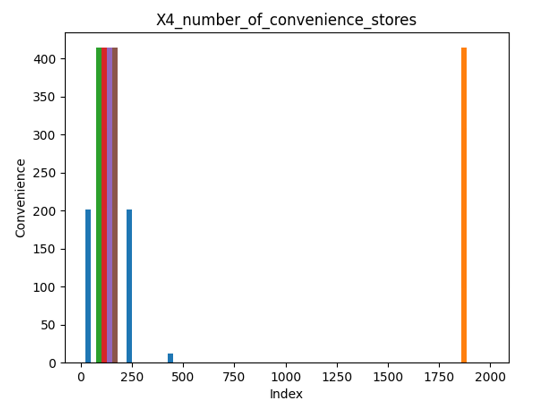
There was a given csv file which we had to pre-process for doing further work. By comparing a row with another row then simply writing command data drop, we have dropped the duplicate values. Using lock function we tried to upper right the train accuracy and loss accuracy. X train and Y train was fitted by using sklearn linear regression. There is no use of dimensionality reduction we directly used the entire values. We used Linear regression for modelling relationships between a dependent variable with a given set of independent variables. We also showed How to improve the accuracy of a Regression Model Handling Null/Missing Values. Data Visualization. Linear regression is one of the most commonly used predictive modelling techniques.It is represented by an equation 𝑌 = b0+b1\*x1+b2\*x2, where b0 is the intercept, b1 is the slope of the line and b2 is the error term. This equation can be used to predict the value of a target variable based on given predictor variable(s).

# Dataset Characteristics and Exploratory Data Analysis (EDA)

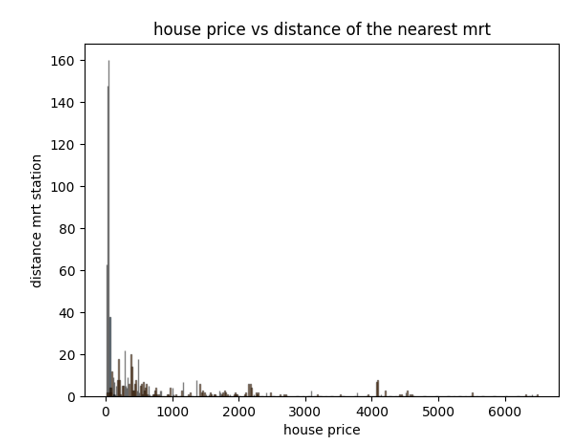
This is a dataset on Real estate. There are 415 rows and 8 columns. The columns are:No,X1 transaction data,X2 house age,X3 distance to the nearest MRT station, X4 number of convenience stores,X5 latitude, X6 longitude, Y house price of unit area.

At this EDA phase, one of the algorithms we often use is Linear Regression. Linear Regression is an algorithm to draw an optimized straight line between two or more variables. Being able to draw such a straight line helps us not only predict the unknown but also understand the relationship between the variables better.

**HISTOGRAM:**



**Figure: X4 No of covariance stores**



**Figure: House price VS Distance of the nearest MRT**

# Data Science Perspective Model

Linear regression is a linear model, e.g. a model that assumes a linear relationship between the input variables (x) and the single output variable. More specifically, that y can be calculate from a linear combination of the input variables (x). Regression analysis is a reliable method of identifying which variables have impact on a topic of interest. The process of performing a regression allows you to confidently determine which factors matter most, which factors can be ignored, and how these factors influence each other.

**Library function:**

MAE is 5.58

MSE is 47.98

R2 score is 0.68

**Implemented Function:**

b0 : 2195.079723896365

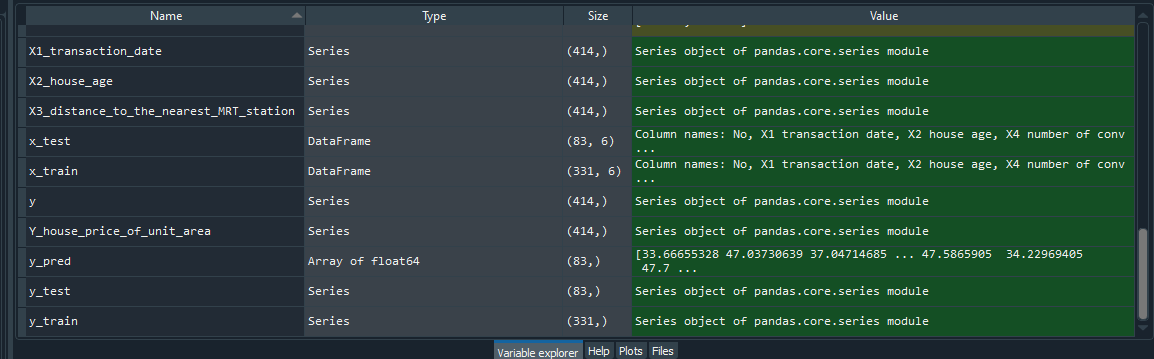
b1 : 3.317327477818516

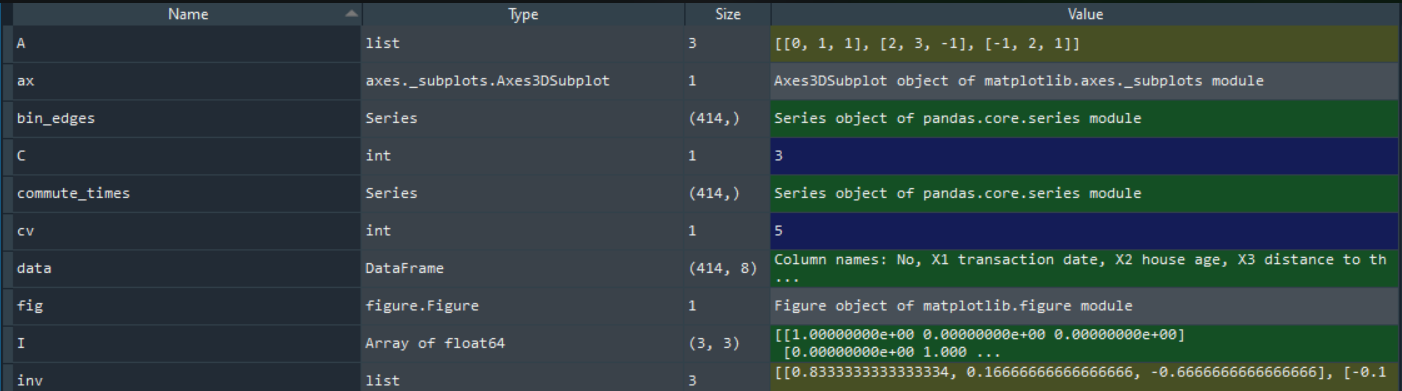
b2 : -450.40201572150306

slope: 45.26388888888876

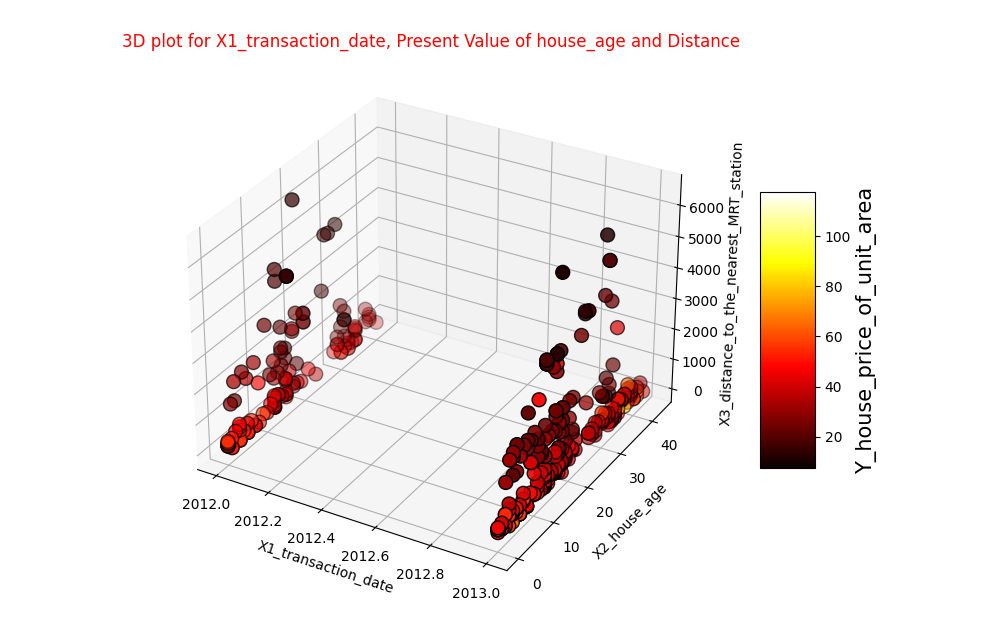
intercept: -90019.05555555529

regression: 0.0002729530353069721

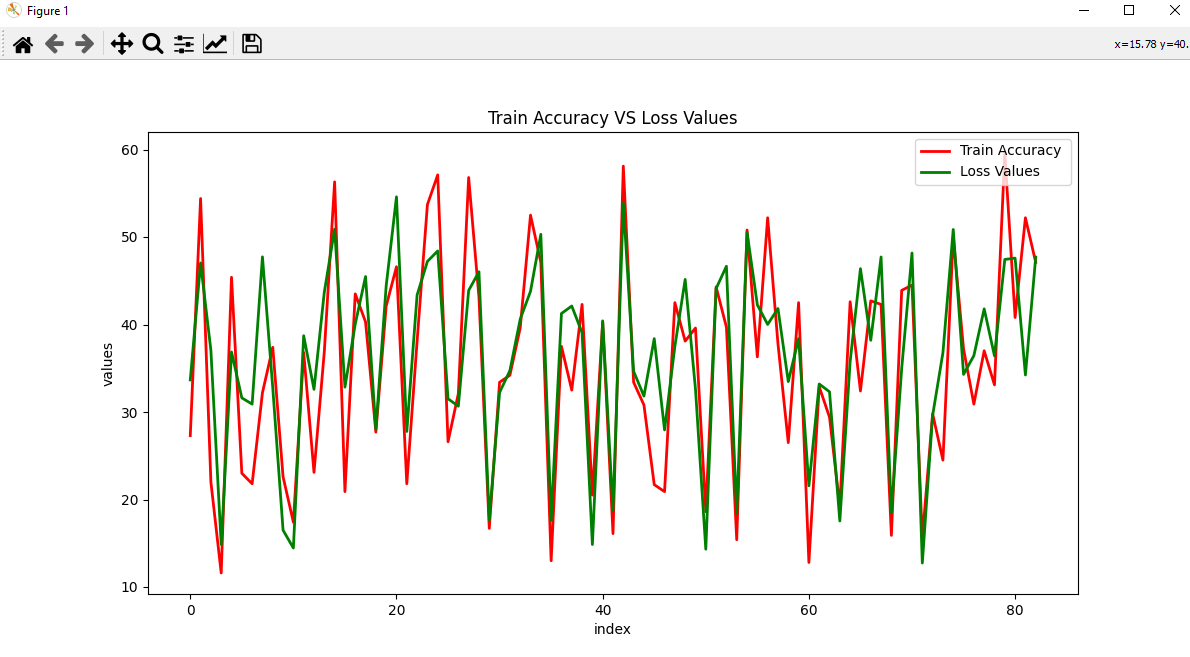




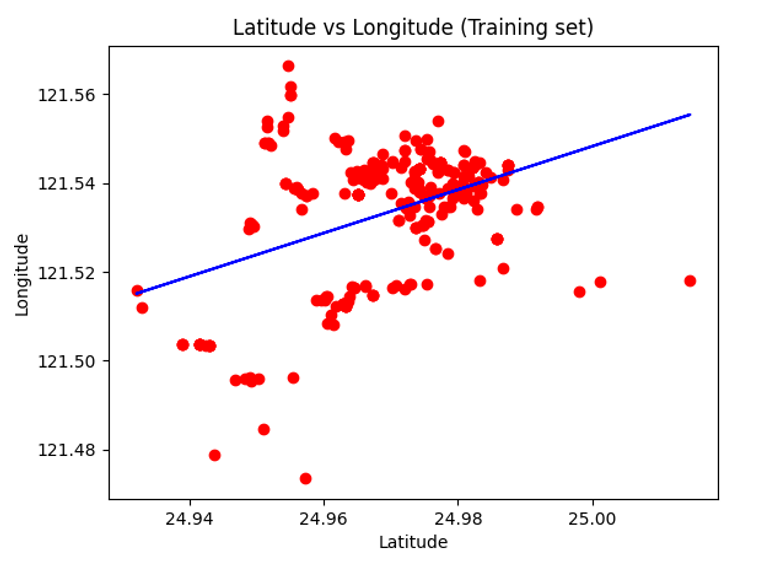
# Loss and accuracy



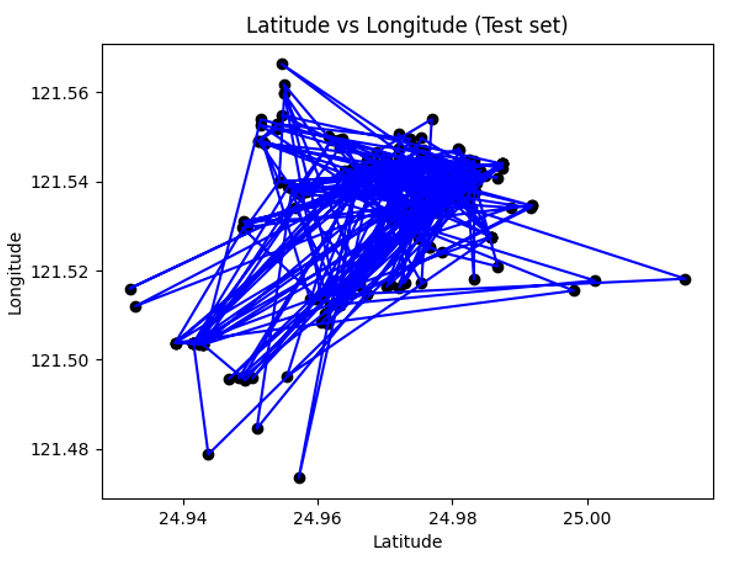
This is a 3D plot which takes 3 type of values (X2,X1,X3).



This is a plot which represents Loss and Accuracy of the values.



This scatter plot is a Latitude VS Longitude training set.



**Figure:Latitude VS Longitude test set**

This is a plot which measures the distance of each point from each other.

**Model:**

defmodel(algorithm,x\_train\_,y\_train\_,x\_test\_,y\_test\_):

    algorithm.fit(x\_train\_,y\_train\_)

    predicts=algorithm.predict(x\_test\_)

    prediction=pd.DataFrame(predicts)

    R\_2=r2\_score(y\_test\_,prediction)

    cross\_val=cross\_val\_score(algorithm,x\_train\_,y\_train\_,cv=cv)

    r\_2.append(R\_2)

    CV.append(cross\_val.mean())

    print("algorithm","\n")

    print("r\_2 score :",R\_2,"\n")

    print("CV scores:",cross\_val,"\n")

    print("CV scores mean:",cross\_val.mean())

    test\_index=y\_test\_.reset\_index()["Y\_house\_price\_of\_unit\_area"]

    ax=test\_index.plot(label="Train Accuracy ",figsize=(12,6),linewidth=2,color="r")

    ax=prediction[0].plot(label = "Loss Values",figsize=(12,6),linewidth=2,color="g")

    plt.legend(loc='upper right')

    plt.title("Train Accuracy VS Loss Values")

    plt.xlabel("index")

    plt.ylabel("values")

    plt.show()

# Discussion

In our first question to do inverse of a matrix, for the output we used an extra module. In this module the more effective part was the time complexity. It was time complexity was more faster in case of giving output. We also used an identity matrix for checking the correction of the inverse matrix.

In the linear regression part, we tried 3D plotting.This is a 3D plot taking 3 typ of values (X1, X2, X3). Using linear regression, we fit the values of Y house price of unit area which represents Loss and Accuracy value plotting. Even though there was a lot of library functions but we tried to implement the code using our own module.

# Conclusion

To implement the project, we could have used library functions to easily implement it. But as we cannot use the library function, we had to use extra module then use it in the code it was a bit challenging. Because of time shortness we have done the project more efficiently. We visualized the test data.

Lickings: [10]we printed the overall dataset which was not required for the project but we did it to look it more realistic. But it increased the execution time of the program.

[16-19] we count the value which was not required but we did it for easier calculation which increased the execution time.

# References

* [*https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiv7PCl0q\_3AhVMSGwGHbKTBxoQFnoECBMQAQ&url=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FLinear\_regression&usg=AOvVaw3EP2QPkOuzWQYU6o8yIH4r*](https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiv7PCl0q_3AhVMSGwGHbKTBxoQFnoECBMQAQ&url=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FLinear_regression&usg=AOvVaw3EP2QPkOuzWQYU6o8yIH4r)
* [*https://www.youtube.com/*](https://www.youtube.com/)
* [*https://en.wikipedia.org/wiki/Linear\_regression*](https://en.wikipedia.org/wiki/Linear_regression)
* [https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiv7PCl0q\_3AhVMSGwGHbKTBxoQFnoECBMQAQ&url=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FLinear\_regression&usg=AOvVaw3EP2QPkOuzWQYU6o8yIH4r](https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiv7PCl0q_3AhVMSGwGHbKTBxoQFnoECBMQAQ&url=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FLinear_regression&usg=AOvVaw3EP2QPkOuzWQYU6o8yIH4r&fbclid=IwAR2rQ45fOks6sVPWEVGCL7ieLZkPFztP_7B0u05AsSrF0tA5IBtPeGxRm8s)
* <https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwjXzILC0q_3AhVkT2wGHS6iCBQQFnoECAcQAw&url=https%3A%2F%2Fwww.wikihow.com%2FFind-the-Inverse-of-a-3x3-Matrix%23%3A~%3Atext%3DTo%2520find%2520the%2520inverse%2520of%2520a%25203x3%2520matrix%252C%2520first%2520calculate%2Crow%2520as%2520the%2520third%2520column.&usg=AOvVaw13e6NLiPJ3HmXfz58hprLn>
* <https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwirrMXN0q_3AhWiSWwGHXJHDtgQFnoECCUQAw&url=https%3A%2F%2Fwww.investopedia.com%2Fterms%2Fh%2Fhistogram.asp%23%3A~%3Atext%3DA%2520histogram%2520is%2520a%2520graphical%2Cinto%2520logical%2520ranges%2520or%2520bins.&usg=AOvVaw3S5rzvo1yhzSbFuYVFwWbd>
* <https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwjRuYbY0q_3AhXtSWwGHYEbC9UQFnoECAsQAQ&url=https%3A%2F%2Fseaborn.pydata.org%2Fgenerated%2Fseaborn.histplot.html&usg=AOvVaw3k9N7S9DfyQrieAVGcbfN7>

# Appendix:

**Inverse matrix main function:**

importinv\_moduleas iv

importnumpyas np

importpprint

R=3

C=3

#A = [[int(input()) for x in range (C)] for y in range(R)]

A = [[0, 1, 1],

    [2, 3, -1],

    [-1, 2, 1]]

inv=iv.inverse\_3X3\_matrix(A)

print("Inverse Matrix : ")

pprint.pprint(inv)

# JUST For TESTING THAT THIS IS WRITE OR WRONG BY USING IDENTITY MATRIX

I = np.matmul(A,inv)

print("IDENTITY MATRIX : \n",I)

**Inverse matrix module inv\_module:**

definverse\_3X3\_matrix(A):

       n = 3

       inv = [[0.0] \* 3] \* 3

       n=3

       det=0.0

       foriinrange(0, n):

              det = det + ( A[0][i] \* (A[1][(i + 1) % 3] \* A[2][(i + 2) % 3] - A[1][(i + 2) % 3] \* A[2][(i + 1) % 3]) )

       print("Det : ",det)

       if det == 0.0:

              print("can't possible")

       else:

              foriinrange(n):

                     a = []

                     for j inrange(n):

                           a.append( (A[(j + 1) % n][(i + 1) % n] \* A[(j + 2) % n][(i + 2) % n] -A[(j + 1) % 3][(i + 2) % 3] \* A[(j + 2) % 3][(i + 1) % 3]) / det )

                     inv[i] = a

       return inv

**Linear regression main function :**

importnumpyas np

import pandas as pd

import seaborn assns

importmatplotlib.pyplotasplt

importos

import sys

data=pd.read\_csv("RealEstate.csv")

print("DATASET : \n",data)

X1\_transaction\_date = data['X1\_transaction\_date']

X2\_house\_age = data['X2\_house\_age']

X3\_distance\_to\_the\_nearest\_MRT\_station = data['X3\_distance\_to\_the\_nearest\_MRT\_station']

print(X1\_transaction\_date.value\_counts(),"\n")

print(X2\_house\_age.value\_counts(),"\n")

print(X3\_distance\_to\_the\_nearest\_MRT\_station.value\_counts(), "\n")

X1\_transaction\_date.replace(regex={"a":"0","b":"1","c":"2"},inplace=True)

X2\_house\_age.replace(regex={"Dealer":"0","Individual":"1"},inplace=True)

X3\_distance\_to\_the\_nearest\_MRT\_station.replace(regex={"Manual":"0","Automatic":"1"},inplace=True)

data[["X1\_transaction\_date","X2\_house\_age","X3\_distance\_to\_the\_nearest\_MRT\_station"]]=data[["X1\_transaction\_date","X2\_house\_age","X3\_distance\_to\_the\_nearest\_MRT\_station"]].astype(int)

from mpl\_toolkits.mplot3d import Axes3D

fig = plt.figure(figsize=(16,9))

ax = fig.gca(projection = "3d")

plot = ax.scatter(data["X1\_transaction\_date"],

                    data["X2\_house\_age"],

                    data["X3\_distance\_to\_the\_nearest\_MRT\_station"],

                    linewidth=1,edgecolor ="k",

                    c=data["Y\_house\_price\_of\_unit\_area"],s=100,cmap="hot")

ax.set\_xlabel("X1\_transaction\_date")

ax.set\_ylabel("X2\_house\_age")

ax.set\_zlabel("X3\_distance\_to\_the\_nearest\_MRT\_station")

lab = fig.colorbar(plot,shrink=.5,aspect=5)

lab.set\_label("Y\_house\_price\_of\_unit\_area",fontsize = 15,)

plt.title("3D plot for X1\_transaction\_date, Present Value of house\_age and Distance",color="red")

plt.show()

Y\_house\_price\_of\_unit\_area = data['Y\_house\_price\_of\_unit\_area']

y=Y\_house\_price\_of\_unit\_area

x=data.drop(["Y\_house\_price\_of\_unit\_area","X3\_distance\_to\_the\_nearest\_MRT\_station"],axis=1)

fromsklearn.model\_selectionimporttrain\_test\_split

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2,random\_state=1)

print("x train: ",x\_train.shape)

print("x test: ",x\_test.shape)

print("y train: ",y\_train.shape)

print("y test: ",y\_test.shape)

fromsklearn.metricsimport r2\_score

fromsklearn.model\_selectionimportcross\_val\_score

cv=5

r\_2 = []

CV = []

# Main function for models

defmodel(algorithm,x\_train\_,y\_train\_,x\_test\_,y\_test\_):

    algorithm.fit(x\_train\_,y\_train\_)

    predicts=algorithm.predict(x\_test\_)

    prediction=pd.DataFrame(predicts)

    R\_2=r2\_score(y\_test\_,prediction)

    cross\_val=cross\_val\_score(algorithm,x\_train\_,y\_train\_,cv=cv)

    r\_2.append(R\_2)

    CV.append(cross\_val.mean())

    print("algorithm","\n")

    print("r\_2 score :",R\_2,"\n")

    print("CV scores:",cross\_val,"\n")

    print("CV scores mean:",cross\_val.mean())

    test\_index=y\_test\_.reset\_index()["Y\_house\_price\_of\_unit\_area"]

    ax=test\_index.plot(label="Train Accuracy ",figsize=(12,6),linewidth=2,color="r")

    ax=prediction[0].plot(label = "Loss Values",figsize=(12,6),linewidth=2,color="g")

    plt.legend(loc='upper right')

    plt.title("Train Accuracy VS Loss Values")

    plt.xlabel("index")

    plt.ylabel("values")

    plt.show()

defLinearRegression1():

    x1 = data['X1\_transaction\_date']

    x2 = data['X2\_house\_age']

    y = data['X3\_distance\_to\_the\_nearest\_MRT\_station']

    x1\_square = 0

    x2\_square = 0

    x1y = 0

    x2y = 0

    x1x2 = 0

    x1\_sum = 0

    x2\_sum = 0

    y\_sum = 0

    n = len(x1)

    foriinrange(n):

        x1\_square = x1\_square + x1[i] \* x1[i]

        x2\_square = x2\_square + x2[i] \* x2[i]

        x1y = x1y + x1[i] \* y[i]

        x2y = x2y + x2[i] \* y[i]

        x1x2 = x1x2 + x1[i] \* x2[i]

        x1\_sum = x1\_sum + x1[i]

        x2\_sum = x2\_sum + x2[i]

        y\_sum = y\_sum + y[i]

    x1\_square = x1\_square - ((x1\_sum \* x1\_sum) / n)

    x2\_square = x2\_square - ((x2\_sum \* x2\_sum) / n)

    x1y = x1y - ((x1\_sum \* y\_sum) / n)

    x2y = x2y - ((x2\_sum \* y\_sum) / n)

    x1x2 = x1x2 - ((x1\_sum \* x2\_sum) / n)

    b1 = ( (x2\_square \* x1y) - (x1x2 \* x2y) ) / (x1\_square \* x2\_square - x1x2 \* x1x2)

    b2 = ( (x1\_square \* x2y) - (x1x2 \* x1y) ) / (x1\_square \* x2\_square - x1x2 \* x1x2)

    b0 = (y\_sum / n) - (b1 \* (x1\_sum / n)) - (b2 \* (x2\_sum / n))

    print("b0 :  ", b0 , "b1 : " ,b1, " b2 : ", b2)

defmean(a):

    n=len(a)

    sum=0.0

    foriin a:

        sum+=i

    return (sum/n)

defsolve():

    a13=data['X1\_transaction\_date']

    a14=data['X3\_distance\_to\_the\_nearest\_MRT\_station']

    xbar=mean(a13)

    ybar=mean(a14)

    x=0

    y=0

    x2=0

    y2=0

    xxbar=0

    yybar=0

    xybar=0

    n=len(a13)

    foriinrange(0,n,1):

      x+=a13[i]

      y+=a14[i]

      x2+=a13[i]\*a13[i]

      y2+=a14[i]\*a14[i]

    foriinrange(0,n,1):

      xxbar += (a13[i] - xbar) \* (a13[i] - xbar)

      yybar += (a14[i] - ybar) \* (a14[i] - ybar)

      xybar += (a13[i] - xbar) \* (a14[i] - ybar)

    slope  =xybar / xxbar

    intercept = ybar - slope \* xbar;

    print("slope: ",slope)

    print("intercept: ",intercept)

    rss=0

    ssr=0

    foriinrange(0,n,1):

     fit = slope\*a13[i] + intercept

     rss += (fit - a14[i]) \* (fit - a14[i])

     ssr += (fit - ybar) \* (fit - ybar)

    degreesOfFreedom = n-2

    r2    = ssr / yybar

    svar  =rss / degreesOfFreedom

    svar1 = svar / xxbar

    svar0 = svar/n + xbar\*xbar\*svar1

    print("regression: ",r2)

#applying linear regression

fromsklearn.linear\_modelimportLinearRegression

lr = LinearRegression()

model(lr,x\_train,y\_train,x\_test,y\_test)

LinearRegression1()

solve()

importlat\_longasll

print(ll)

## ERROR VALUE

y\_pred=lr.predict(x\_test)

fromsklearnimport metrics

mae=metrics.mean\_absolute\_error(y\_test,y\_pred)

msc=metrics.mean\_squared\_error(y\_test,y\_pred)

r2=metrics.r2\_score(y\_test,y\_pred)

print("The model Performance for testing set")

print("---------------------------------------------------")

print('MAE is %.2f'% mae)

print('MSE is %.2f'% msc)

print('R2 score is %.2f'% r2)

##hist

xx=data[["X4\_number\_of\_convenience\_stores"]]

xx = np.random.normal(170, 10, 250)

plt.hist(x)

plt.title("X4\_number\_of\_convenience\_stores")

plt.xlabel("Index")

plt.ylabel("Convenience")

plt.show()

#Two  lines to make our compiler able to draw:

## hist 2

value1 = data['Y\_house\_price\_of\_unit\_area']

value2 = data['X3\_distance\_to\_the\_nearest\_MRT\_station']

n = len(value1)

x = [value1, value2]

plt.hist(x,

         bins = n,

         density=False,

         histtype='bar',

         edgecolor='k',

         alpha=0.5,

         width = 10)

plt.title("house price vs distance of the nearest mrt")

plt.xlabel("house price")

plt.ylabel("distance mrt station")

plt.show()

**Linear regression Module lat\_long:**

from turtle importcolor

importnumpyas np

importmatplotlib.pyplotasplt

import pandas as pd

data=pd.read\_csv("RealEstate.csv")

print("DATASET : \n",data)

x=data[['X5\_latitude']]

y=data[['X6\_longitude']]

fromsklearn.model\_selectionimporttrain\_test\_split

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=1/3,random\_state=0)

##Fitting the DS

fromsklearn.linear\_modelimportLinearRegression

regression= LinearRegression()

regression.fit(x\_train,y\_train)

##PREDICTION the value

y\_pred=regression.predict(x\_test)

## visualization of lat long relation

plt.scatter(x\_train,y\_train,color='red')

plt.plot(x\_train,regression.predict(x\_train),color='blue')

plt.title('Latitude vs Longitude (Training set)')

plt.xlabel ('Latitude')

plt.ylabel('Longitude')

plt.show()

### visualizing test set result of lat long

plt.scatter(x\_train,y\_train,color='black')

plt.plot(x\_train,y\_train,color='blue')

plt.title('Latitude vs Longitude (Test set)')

plt.xlabel ('Latitude')

plt.ylabel('Longitude')

plt.show()

**Output of this project:**

