

# Project Report

## **Introduction:**

For both the training and testing dataset, I have taken the housing.csv file from Kaggle.com, which contains data of California Housing Prices for the following feature:

1. housingMedianAge: Median age of a house within a block; a lower number is a newer building
2. totalRooms: Total number of rooms within a block
3. totalBedrooms: Total number of bedrooms within a block
4. population: Total number of people residing within a block
5. households: Total number of families, a group of people living within a home unit, for a block
6. medianIncome: Median income for households within a block of houses (measured in tens of thousands of US Dollars)
7. medianHouseValue: Median house value for homes within a block (measured in US Dollars)

Independent Variable: housingMedianAge, totalRooms, totalBedrooms, population, households, medianIncome

Dependent Variable: medianHouseValue

I have split the first 5,000 rows and saved it as housing\_train.csv to train my algorithm and find out weight and J value for programming purposes. After that, I have split the rows from 10,000 to 12,000 and saved it as housing\_test.csv for testing my weights and deriving the new J value.

The screenshot shows the Spyder Python IDE with a file named 'Project1.py' open. The script implements a linear regression model using pandas and numpy. It reads training and test data from CSV files, preprocesses them by dropping the target variable and selecting features, and then calculates the weights (w) for the model. The console output shows the weights for the training data and the predicted values for the test data.

```
2 """
3 Created on Mon Sep 6 00:57:33 2021
4
5 @author: dines
6 """
7
8 import pandas as pd
9 import numpy as np
10
11 FileName = input("Enter the name of your training file: ")
12 trainD = pd.read_csv(FileName)
13 trainD = trainD.dropna()
14 total_rows = trainD.count
15 trainD = trainD.iloc[:, 2:-1]
16 X_td = trainD.iloc[:, :-1]
17 Y_td = trainD[["median_house_value"]].values
18
19 A = np.linalg.pinv(np.dot(X_td.T, X_td))
20 B = np.dot(X_td.T, Y_td)
21 w = np.dot(A, B)
22
23 temp1 = np.dot(X_td, w) - Y_td
24 Jtrain = (1/len(X_td))*np.dot(temp1.T, temp1)
25
26 np.set_printoptions(formatter={'float_kind': '{:f}'.format})
27
28 print("Weights of the train data: ", w)
29 print("J value of the train data: ", Jtrain)
30
31
32 FileName = input("Enter the name of your test file: ")
33 testD = pd.read_csv(FileName)
34 testD = testD.dropna()
35 testD = testD.iloc[:, 2:-1]
36 X_testD = testD.iloc[:, :-1]
37 Y_testD = testD[["median_house_value"]].values
38
39 temp2 = np.dot(X_testD, w) - Y_testD
40 Jtest = (1/len(X_testD))*np.dot(temp2.T, temp2)
41 print("J value of the test data: ", Jtest)
42
```

Console 3/A

```
weights of the train data: [[900.010070]
 [-21.427765]
 [41.440904]
 [-38.135530]
 [189.714072]
 [41433.718446]]
J value of the train data: [[5158266932.758972]]
Enter the name of your test file: housing_test.csv
J value of the test data: [[5434748199.093115]]
In [2]: |
```

	Predicted output	Ground truth
0	146756.038270	201700
1	133753.477816	162500
2	180471.547838	209600
3	155537.464521	182100
4	188794.783584	206400
...	...	...
1979	138516.420512	153100
1980	155312.793904	75000
1981	167321.532010	203200

J value of the train data: [[5158266932.758972]]

J value of the test data: [[5434748199.093115]]