Data Mining Project

Mohammad Shazil Mahmood, i19-0542
Hannan Ali, i19-2172
Umer Rashid, k19-1257
CS-E

House Price Prediction using Linear Regression

The prices have infinite possible values [continuous values], so it cannot be classified into finite categories, therefore linear regression will be implemented instead of logistic regression.

Note:-

We used 2 programming languages for implementation of this project.

1. **Python:** All the data science related code (Feature Selection, Linear Regression and Final Prediction)
2. **C++:** For data generation, preprocessing and side-tasks such as finding a list of all possible feature set combinations.

Along with this report there are 9 other files (listed below) included in the zip folder.

	File Name	Format	Purpose
1	House Prediction	.py	Main Code Train the model on the Best Features For prediction, take the attribute input from the user and output the predicted price.
2	Feature Selection	.py	Train the model on all possible feature set combinations (510). Calculate the test-error. Finds the best feature set with minimum test-error. Store the info of each feature set and its error in a file (Feature_Error.txt).
3	Feature_Set_Combination	.cpp	List down all possible combinations of feature sets.
4	Training Dataset Generator	.cpp	Generates 10 Million Records (80% Training Data and 20% Test Data) Performs the preprocessing.

5	Feature_Error	.txt	Stores the Test Error of each feature set. Stores the info of the best feature set and its test error. As feature selection had a long execution time (1+ hour), so this stores the results.
6	pre_test	.csv	Test Data before PreProcessing/Cleaning
7	pre_training	.csv	Training Data before PreProcessing/Cleaning
8	final_test	.csv	Test Data after PreProcessing/Cleaning
9	final_training	.csv	Training Data after PreProcessing/Cleaning

Implementation Procedure

❖ Training and Test Data

For house prediction we considered the following Attributes:-

1. Level
   Level represents the Location of The House
   We divided the houses location into 3 segments
   - ➔ Rural Area
   - ➔ Middle Class Area
   - ➔ Upper Class Area

   After Preprocessing each segment is assigned a number.
   Rural Area: 1
   Middle Class Area: 2
   Upper Class Area: 3

2. Area of the house. (Square Feet)
3. No. of Floors (1 or more)
4. No. of Floors Rooms (1 or more)
5. Pool (Yes/No) (1 or 0 after Pre-Processing)
6. No of Garage (1 or more)
7. Garden (Yes/No) (1 or 0 after Pre-Processing)
8. Age of The House (Years)
9. Home Mini-Theater (Yes/No) (1 or 0 after Pre-Processing)
10. Price of the house. (Million-USD)

The data was generated using a C++ Program **(Training Dataset Generator.cpp)**
The **data values (other than price** were generated via **random values** but in a **logical manner** with constraints close to real world data.

For Example:
Minimum Area is 1500 sq-ft
Minimum Rooms are 3.
No. of floors: 1 to 3

If Location is **Middle or Upper Class**, then area can be **upto 26,499 sq-ft.**
Otherwise if **Location in Rural**, then area can be **upto 11,499 sq-ft.**

Houses with Area **less than 5500 sq-ft.**
Don't have a **garden, pool, and theater**.
Have a **maximum** of **3 rooms** per floor and **1 garage**.

House with Area **more than 5500 sq-ft**
Might have a **pool, garden, theater.**
Can have **3 to 6 rooms** per floor, and **1 to 3 Garage.**

House with Area **more than 15000 sq-ft**
It has a **Garden, Pool, Theater** and **3 Garages.**
Can have **3 to 6 rooms** per floor.

Price Calculation:-
Price is calculated in Million USD based on the following formula/method/

1. 28.37 USD per sq-ft. (Checked some websites to see the average land price)
   Price = Area * 28.37
2. Then Price is Multiplied by the Level (Rural-1, Middle Class-2, Upper Class-3)
   Price = Price * Level
3. For every floor, add 40,000 USD.
4. If it has a Pool, then Add 20,000 USD.
5. If it has a Garden, then Add 10,000 USD.
6. If it has a Theater, then Add 15,000 USD.
   Price = Price + (Floors*40,000) + (Pool*20000) + (Theater*15000)
7. Age
   a. If **Age >= 40** then Price **depreciated** to **80%** of original price.
   b. If **40 > Age >= 30** then Price **depreciated** to **90%** of original price.
   c. If **30 > Age >= 20** then Price **remains the same.**
   d. If **20 > Age >= 10** then Price **appreciated** to **110%** of original price.
   e. If **10 > Age >= 5** then Price **appreciated** to **120%** of original price.
   f. If **5** > Age then Price **appreciated** to **130%** of original price.
8. Price is divided by 1,000,000 to convert into Million USD

The program also performs the **preprocessing task.**
Like for location segmentation.

Rural Area:	1
Middle Class Area:	2
Upper Class Area:	3

Pool, Garden, Theater: (Yes/No) Boolean Values ⟶ 0 or 1

A total of 1 Million records are generated.
80% of the data is for training.
20% of the data is for testing.

As a result, the program generates **4 .csv Files** in Total.

1. pre_test (20% Data for Test, Before Preprocessing/Cleaning)
2. pre_training (80% Data for Training, Before Preprocessing/Cleaning)
3. **final_test (20% Data for Test, After Preprocessing/Cleaning)**
4. **final_training (80% Data for Training, After Preprocessing/Cleaning)**

final_test.csv and **final_training.csv** are later used for feature selection, linear regression and prediction

❖ Feature Selection.

First we find the **list of all possible combinations** of the feature set.

Level, Area, Floor, Rooms, Pool, Garage, Garden, Age, Theater

For this we used a simple C++ Program **(Feature_Set_Combination.cpp).**

There were a total of **510 unique possible combinations**.

Output:

['Level'],
['Area'],
[‘Floor'],
['Rooms'],
['Pool'],
['Garage'],
['Garden'],
['Age'],
['Theater'],
['Level','Area'],
['Level','Floor'],
['Level','Rooms'],
['Level','Pool'],
['Level','Garage'],
.
.
.
.
.
['Level','Area','Floor','Pool','Garage','Garden','Age','Theater'],
['Level','Area','Rooms','Pool','Garage','Garden','Age','Theater'],
['Level','Floor','Rooms','Pool','Garage','Garden','Age','Theater'],
['Area','Floor','Rooms','Pool','Garage','Garden','Age','Theater'],
['Level','Area','Floor','Rooms','Pool','Garage','Garden','Age','Theater']

Then to **find the best feature set,** we used a **python** program **(Feature Selection.py)**

It performs linear regression on **final_training.csv (80% training data)** and then checks the test-error using **final_test.csv (20% test data)**.
This is done **iteratively** for **all 510 feature sets,**
The feature set with **minimum error** is **selected** as the **Best Feature Set.**

The test errors of all feature sets are stored in a file **Feature_Error.txt.**
Along with the Best Feature and Minimum error.

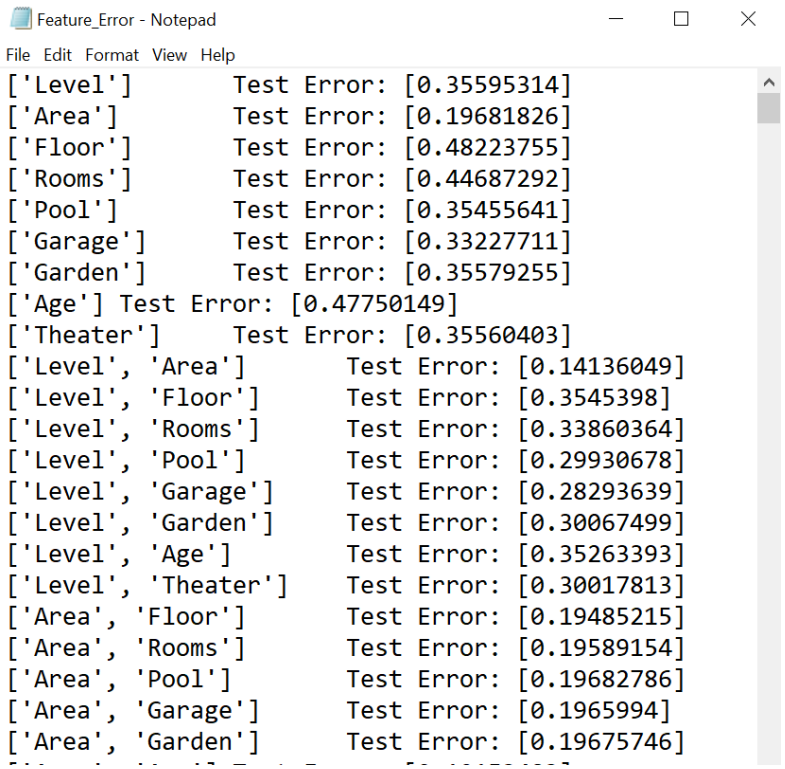
The reason for storing this information was that the feature selection execution took relatively longer time as compared to other operations, as it had to train and test for 510 times.
In our case, it took almost 81 minutes (execution time stored at the end of the same text file).
So instead of running it again and again, we stored all the information, so it can be used later.

Results:
This was the best feature set with minimum test-error.
Best Features: ['Level', 'Area', 'Floor', 'Rooms', 'Pool', 'Garage', 'Garden', 'Age', 'Theater']
Avg Test Error: [0.11171462]

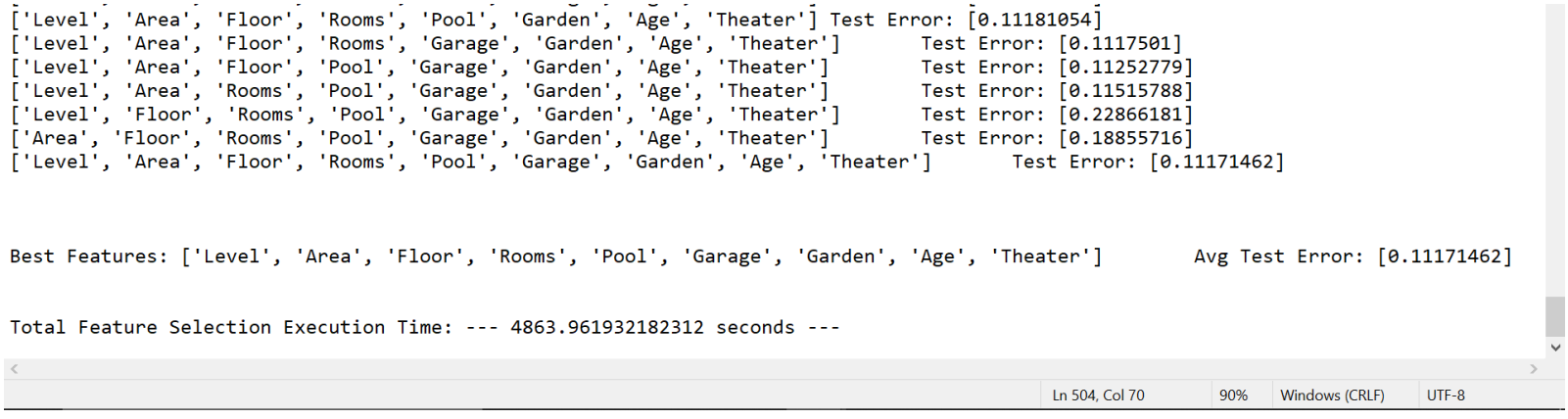
Note: Test Error Value is in Million USD.

0.111 Million USD is almost 20 Million PKR
Which is relatively a low error, considering that there are houses costing upto 460 Million and above.

Feature_Error.txt
Screenshot



-
-
-
-
-
-



❖ Final House Price Prediction

This is our main goal.
Till now all the programs and processes were used to assist us, to perform preprocessing, to generate a list of all feature set and selecting the best feature set.
But **the end user will execute this program only.**

We will use the **python program (House Prediction.py)**

It first performs **linear regression** using **final_training.csv** on the best feature set we found in the previous step of feature selection.
Features: ['Level', 'Area', 'Floor', 'Rooms', 'Pool', 'Garage', 'Garden', 'Age', 'Theater']

After the model is trained.
Now we take attribute inputs from users.

Select Location:	1. Rural Area, 2. Middle Class Area, 3. Upper Class Area,
Enter Area (Sq-ft):	Any +ve Value, > 0
Enter No. Of Floors:	Any +ve Value, > 0
Enter No. Of Rooms:	Any +ve Value, > 0
Pool:	(Yes/No)
Enter No. of Garage:	Any +ve Value, > 0
Garden:	(Yes/No)
Age of House:	Any +ve Value, > 0
Theater:	(Yes/No)

And the system will predict the house price.

Screenshot on Next Page

```
PS C:\Users\LENOVO> & C:/Users/LENOVO/AppData/Loca
Training....

Enter 'Q' to Quit
Enter any other key to Predict House Price
A

Select the location of House
Enter '1' for Rural Area
Enter '2' for Middle Class Area
Enter '3' for Upper Class Area
3

Enter The Area of House (Sq-ft) (+ve value, >0)
12000

Enter No. of Floors (+ve value, >0)
2

Enter No. of Rooms (+ve value, >0)
4

Enter if House have a Pool or Not
Enter '0' for No
Enter '1' for Yes
1

Enter No. of Garages (+ve value, >0)
2
```

```
Enter if House have a Garden or Not
Enter '0' for No
Enter '1' for Yes
1

Enter Age of House (Years) (+ve value, >0)
8

Enter if House have a Theater or Not
Enter '0' for No
Enter '1' for Yes
1

Prediction Summary:-
Predicted Price (Million USD):  [1.23149344]
Predicted Price (Million PKR, @176.38):  [217.21081234]
Location: Upper Class
Area (Sq-Ft):  12000
Floors:  2
Rooms:  4
Garage:  2
Age of House(Years):  8
Pool: Yes
Garden: Yes
Theater: Yes

Enter 'Q' to Quit
Enter any other key to Predict House Price
Q
PS C:\Users\LENOVO>
```

Thank You!