

HOUSE PRICE PREDICTION

By

NAVEEN KUMAR M - 2015103519

UMAPATHI C - 2015103613

A Project report submitted to the

FACULTY OF COMPUTER SCIENCE AND ENGINEERING

For

CREATIVE AND INNOVATIVE PROJECT

In the academic year of 2018 - 2019



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING,
ANNA UNIVERSITY,
CHENNAI-25**

ABSTRACT

The Purpose of this project is to develop an application to predict the Sale price of the house. Since, due to the demand for Houses in recent times, there are a lot of real estate agents arises and tries to cheat the house holder by lowering there house value and try to sell the house to customer for high price and taking the difference amount as a agents commission. So an application which finds the actual price of a house is really very much useful in the field of Real Estate.

Here we building the Decision tree regression model and trained it using a collected data values from the city of boston. By using the high predictive power of Decision Tree model to predict the sale price of house from the user input values.

Finally a Web application is designed to provide interface to the user for getting house attribute values and also to display the output sale price of a house.

TABLE OF CONTENTS

ABSTRACT	ii
LIST OF FIGURES	vii
LIST OF ABBREVIATIONS	vii
1. INTRODUCTION	1
1.1 Problem Domain	1
1.2 Significance	1
1.3 Problem Description	2
1.4 Swot Analysis	3
1.5 Pestel Analysis	4
1.6 Organization of Thesis	5
2. RELATED WORK	6
2.1 Decision Tree	6
2.2 Naive Bayes Classifier	6
2.3 Support Vector Machine	7
2.4 Random Forest	8
2.5 Observation from the Survey	8

3. REQUIREMENT ANALYSIS	9
3.1 Functional Requirements	9
3.2 Non Functional Requirements	9
3.2.1 User interface	9
3.2.2 Hardware Requirements	10
3.2.3 Software Requirements	10
3.3 Constraints and Assumptions	10
3.3.1 Constraints	10
3.3.2 Assumptions	10
 4. SYSTEM DESIGN	 11
4.1 System Architecture	11
4.2 UI Design	12
4.3 Class diagram	13
4.4 Module Design	14
4.4.1 Preprocessing	13
4.4.2 Analyzing the model	16
4.4.3 Model Training	16
4.4.4 Making Prediction	17
4.5 Complexity Analysis	18
4.5.1 Time Complexity	18
4.5.2 Complexity of the Project	18

5. SYSTEM DEVELOPMENT	19
5.1 Prototype across a Modules	19
5.1.1 Preprocessing	19
5.1.2 Analysing the Model	19
5.1.3 Training the Model	19
5.1.4 Make Prediction	20
5.2 Algorithm	20
5.3 Deployment Details	21
6. RESULTS AND DISCUSSION	22
6.1 Dataset for Testing	22
6.2 Output Obtained in Various Stages	23
6.2.1 Input	23
6.2.2 Model Analyzing	24
6.2.3 Model Training	25
6.2.4 Price Prediction	25
6.3 Sample Screenshots of various Input	26
6.3.1 Input	26
6.3.2 Output	26
6.4 Performance Evaluation	27
6.4.1 R-Squared score	27
6.4.1.1 Description	27
6.4.1.2 Algorithm	28
6.4.1.3 Score and obtained discussion	28

7. CONCLUSION

7.1 Summary	30
7.2 Future Work	31
7.3 References	32

LIST OF FIGURES

2.1 Support Vector Machine	7
4.1 System Architecture	11
4.2 UI Architecture	12
4.3 Class Diagram	13
4.4 Data Exploration	14
6.1 Sample Input for UI	22
6.2 Model Analyzing	23
6.3 Model Training	23
6.4 Price Prediction	24
6.5 Sample Input	25
6.6 Sample Output	25

LIST OF ABBREVIATIONS

ML	Machine Learning
GUI	Graphical User Interface
UI	User Interface
CDN	Content Delivery Network
RMSE	Root mean Square Error
MPE	Mean Prediction Error

CHAPTER 1

INTRODUCTION

In this project, we will evaluate the performance and predictive power of a model that has been trained and tested on data collected from homes in suburbs of Boston, Massachusetts. A model trained on this data that is seen as a good fit could then be used to make certain predictions about a home in particular, its monetary value. This model would prove to be invaluable for someone like a real estate agent who could make use of such information on a daily basis.

1.1 PROBLEM DOMAIN

In this project we have deployed Machine Learning to solve the Problem in Predicting the best selling price of the house. Now-a-day Machine learning providing us a way for lot of Complex Problem by Highly efficient Algorithms.

Therefore, we are also chosen Machine learning in solving the House Price Prediction Problem.

1.2 SIGNIFICANCE

Accurate real estate pricing is one of the key issues for countries all over the world. Firstly, at the microeconomic level, real estate, no matter it is a house or an apartment, is the most expensive expenditure that the majority of people will make during their lifetime. Current market housing prices may cause people to be misled by actions of sellers and other bidders, such as price manipulation and shill bidding.

What's more, accurate price models are also desired by real estate agents, whose objective is to sell houses both profitably and quickly. As it is unrealistic to achieve both goals to a high degree, they want to find a balance between profit and speed. A real estate pricing tool is of great help.

1.3 PROBLEM DESCRIPTION

Economists have long recognized that housing markets are geographically localized. Therefore, real estate market price indices are typically assumed to predict for certain geographic area within county area or metropolitan area, which only valid for that particular area. Another crucial issue of, real estate market differing with financial market is assumption of imperfections in real estate.

There are many other methodologies to construct real estate market price indices except for hedonic regressions and repeat-sales analysis (Englund, Quigley and Red feam 1998; Case and Quigley 1991). However, hedonic and repeat-sales models are the fundamental of all other evolution models.

Unfortunately, Both of these two models have their own flaws. Since the hedonic regression model uses data on a vector of key characteristics to control property quality, this method needs large amounts of data across different periods. Since the selected properties must be transacted at least twice, those properties sold only once must be discarded from the sample, which exclude the majority of transactions data.

There fore, it fails to utilize the full information in real estate market, which may affect the results. To the best of my knowledge, regression model, especially regularized regression model has not been performed in real estate data analysis up to now, even though it has been well developed by previous studies and been implemented in other scientific and empirical fields. Regression models, one of the most powerful tools in statistics, focus on learning the relationship among multiple variables statistically by coming up a functional relationship between two or more features such

that the feature that people care mostly can be predicted from the other or the others.

As mentioned above, the scientific goal here is to achieve accurate enough predictions of median property price of tracts basing upon all approachable features, so regression model in statistics science is one of the most reasonable solvents for this problem.

1.4 SWOT ANALYSIS

SWOT Analysis is a planning tool used to understand the Strength, Weakness, Opportunities, and Threats involved in a project. It involves specifying the objective of the project and identifying the internal and external factors that are supportive or unfavourable to achieving that objective.

S: Play to your Strength

W: Address Weaknesses

O: Exploit Opportunities

T: Hedge against Threats

S: Strength

Our project will help to the new house sellers and buyers to easily knowing their house real value. This works as an advantage to the agents and they manipulate the price and increase their own profits.

Our Project will predict a stable reasonable Price of the house. We use feature selection technique so it will give importance to the key features of the house.

W: Weakness

There are various other factors in the market that affect the prices. Parameters like Economy, the inflation rate of an area may result in increase or decrease in the

prices.

The features are not sufficient anymore, in real time we have to take more features like salaries, demand, population into consideration.

O: Opportunities

“House Price Prediction” will have opportunities in the country. There are lot of employment in our India. For eradicate the Unemployment problem only solution is self-employment. Government policies like the Real Estate Regulatory Authority (RERA) have influenced fresh buyer condense into the real estate sector.

T: Threats

People mindset will always changing, so we predict Sometimes, a model is either too complex or too simple to sufficiently generalize to new data. Other times, the data itself could be too noisy or contain too few samples to allow a model to adequately capture the target variable -i.e., the model is underfitted.

1.5 PESTEL ANALYSIS

A **PESTEL** analysis is a framework or tool used by marketers to analyze and monitor the macro-environmental (external marketing environment) factors that have an impact on an organization. The result of which is used to identify threats and weaknesses which is used in a SWOT analysis.

PESTEL stands for:

P: Political

Political interventions will be reduced because of digitalization.

E: Economic

Buying and selling the Real estate properties will increase the tax amount which leads to the increase of Country's Economy.

S: Social

It will reduce the intermediate person (agents). So the buyer need not to give any extra amounts.

T: Technological

It increases the Digitalization of the Society. It paves way for Technological development of the county.

E: Environmental

The application will also analyze the environmental features, so the environmental variables also taken consideration.

L: Legal

It will give a good upto date market of the price for each house. So the correct income tax of the house will be paid by buyer.

1.6 ORGANISATION OF THESIS

Chapter 2 discuss the existing approaches to House Price Prediction in greater detail. Chapter 3 gives the requirement analysis of the system. It explains the functional and nonfunctional requirements, constraints and assumptions made in the implementation of the system.

CHAPTER 2

RELATED WORK

This gives a survey of various machine learning that can be designed for House price prediction. The models include the likes of decision Tree, SVM, Naive Bayes, Random Forest. This helped to us study in details the various methods and choice the best which suits our needs.

2.1 DECISION TREE

Decision Tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, decision tree algorithm can be used for solving regression and classification problems too.

The general motive of using Decision Tree is to create a training model which can use to predict class or value of target variables by learning decision rules inferred from prior data (training data).

The understanding level of Decision Trees algorithm is so easy compared with other classification algorithms. The decision tree algorithm tries to solve the problem, by using tree representation. Each internal node of the tree corresponds to an attribute, and each leaf node corresponds to a class label.

2.2 NAIVE BAYES CLASSIFIER

In machine learning, naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naive) independence assumptions between the features. It remains a popular (baseline) method for text

categorization, the problem of judging documents as belonging to one category or the other (such as spam or legitimate, sports or politics, etc.) with word frequencies as the features. With appropriate pre-processing, it is competitive in this domain with more advanced methods including support vector machines. It also finds application in automatic medical diagnosis.

2.3 SUPPORT VECTOR MACHINE

“Support Vector Machine” (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n -dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well (look at the below snapshot).

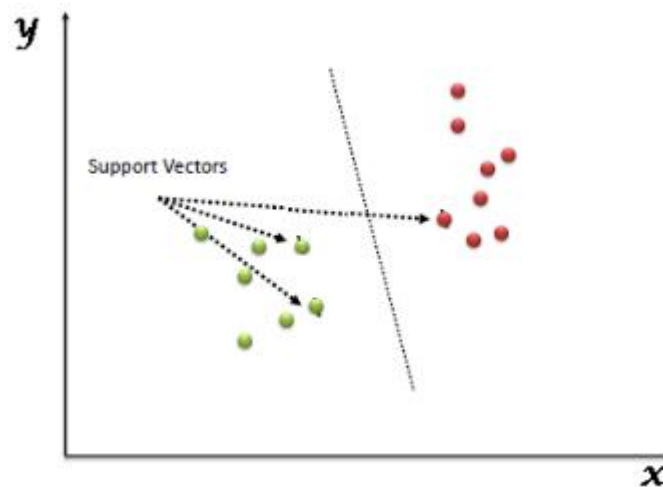


Figure 2.1 Support Vector Graph

Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line).

2.4 RANDOM FOREST

Random forest is like bootstrapping algorithm with Decision tree (CART) model. Say, we have 1000 observation in the complete population with 10 variables. Random forest tries to build multiple CART model with different sample and different initial variables. For instance, it will take a random sample of 100 observation and 5 randomly chosen initial variables to build a CART model. It will repeat the process (say) 10 times and then make a final prediction on each observation. Final prediction is a function of each prediction. This final prediction can simply be the mean of each prediction.

2.5 OBSERVATIONS FROM THE SURVEY

We propose to use Decision Tree Model. Since, One big advantage of the decision tree model is its transparent nature. Unlike other decision-making models, the decision tree makes explicit all possible alternatives and traces each alternative to its conclusion in a single view, allowing for easy comparison among the various alternatives. The use of separate nodes to denote user defined decisions, uncertainties, and end of process lends further clarity and transparency to the decision-making process.

A major decision tree analysis advantages is its ability to assign specific values to problem, decisions, and outcomes of each decision. This reduces ambiguity in decision-making. Every possible scenario from a decision finds representation by a clear fork and node, enabling viewing all possible solutions clearly in a single view. Incorporation of monetary values to decision trees help make explicit the costs and benefits of different alternative courses of action. A decision tree is the best predictive model. It finds use to make quantitative analysis of business problems, and to validate results of statistical tests. It naturally supports classification problems with more than two classes and by modification, handles regression problems.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 FUNCTIONAL REQUIREMENTS

The Developed System outputs a Median value of the sales price of a house. The output prediction should adhere to the following requirements:

- The Predicted value should be reasonable depending on the feature of the house
- The error rate of the prediction shouldn't be larger.
- The Model should be stable when the size of the dataset is modified
- More reliable estimate of out-of-sample performance than train/test split should achieved
- The Model must taken higher consideration for the key features of a house
- The system must be optimized for space complexity.
- The system must be able to make prediction based on the entered input value.
- Server response time must be optimal.

3.2 NON FUNCTIONAL REQUIREMENTS

3.2.1 USER INTERFACE

There must be a simple and easy to use user interface where the user should view his outputs and graphical structures.

3.2.2 HARDWARE REQUIREMENTS

The client side requirement is only a browser with good internet connection. The server side must be able to handle multiple requests & tools to run machine learning python scripts. Server must be able to run javascript runtime environment (NodeJS).

3.2.3 SOFTWARE REQUIREMENTS

- OS-Windows, Linux or Mac.
- Programming Languages -JavaScript, Python.
- Frameworks - JQuery, Bootstrap.
- Libraries - Pandas, Scikit-learn, Numpy.
- Tools-Anaconda.

3.3 CONSTRAINTS AND ASSUMPTIONS

3.3.1 CONSTRAINTS

- Data collected from a rural city may not be applicable as the demographics would change and other features may be better able to fit the dataset instead of a model with features that was learned using urban data.
- The learning algorithm learned from a very old dataset that may not be relevant because demographics have changed a lot since 1978.
- There are only 3 features currently, there are more features that can be included such as crime rates, nearby to city, public transport access and more.

3.3.2 ASSUMPTIONS

- The input dataset is assumed that it contains all the key features of the house.
- The data is assumed that it doesn't undergoes underfitting.
- The input dataset does not have any invalid symbols and characters

CHAPTER 4

SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE

The block diagram of the entire system is shown in figure below. A web application has been developed and its description is explained in section 4.2.

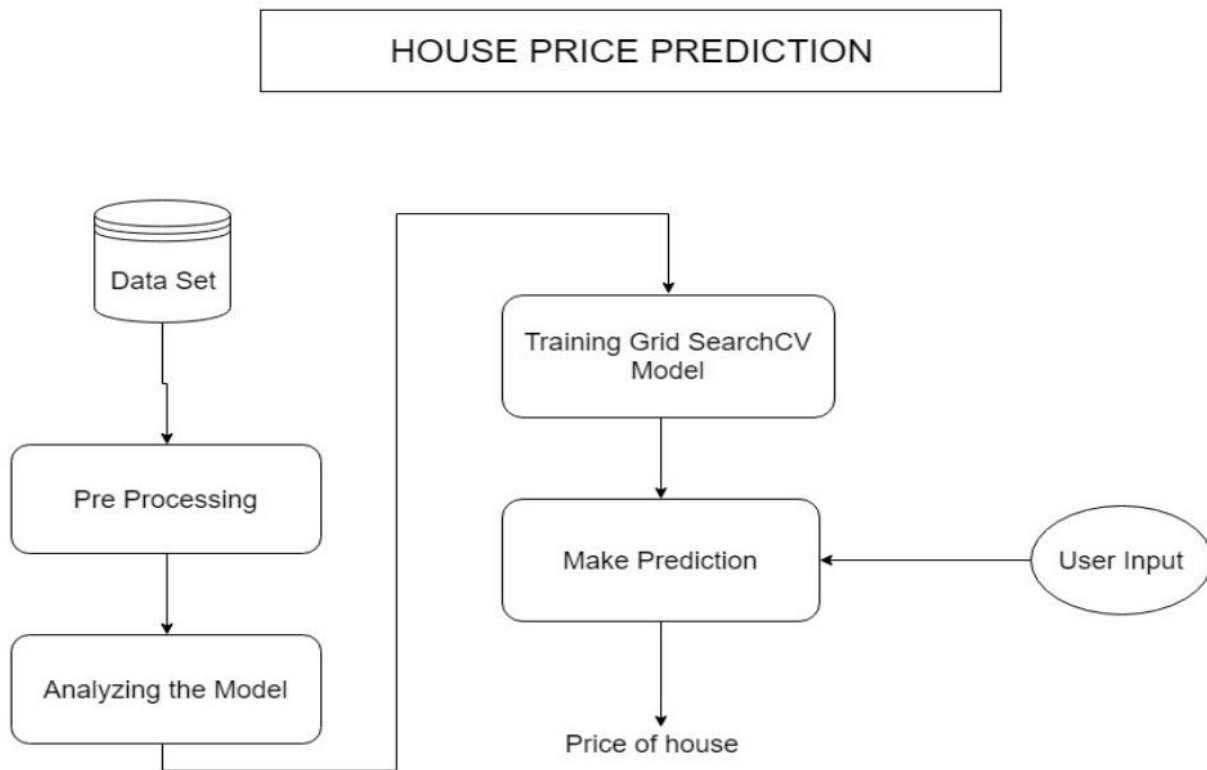
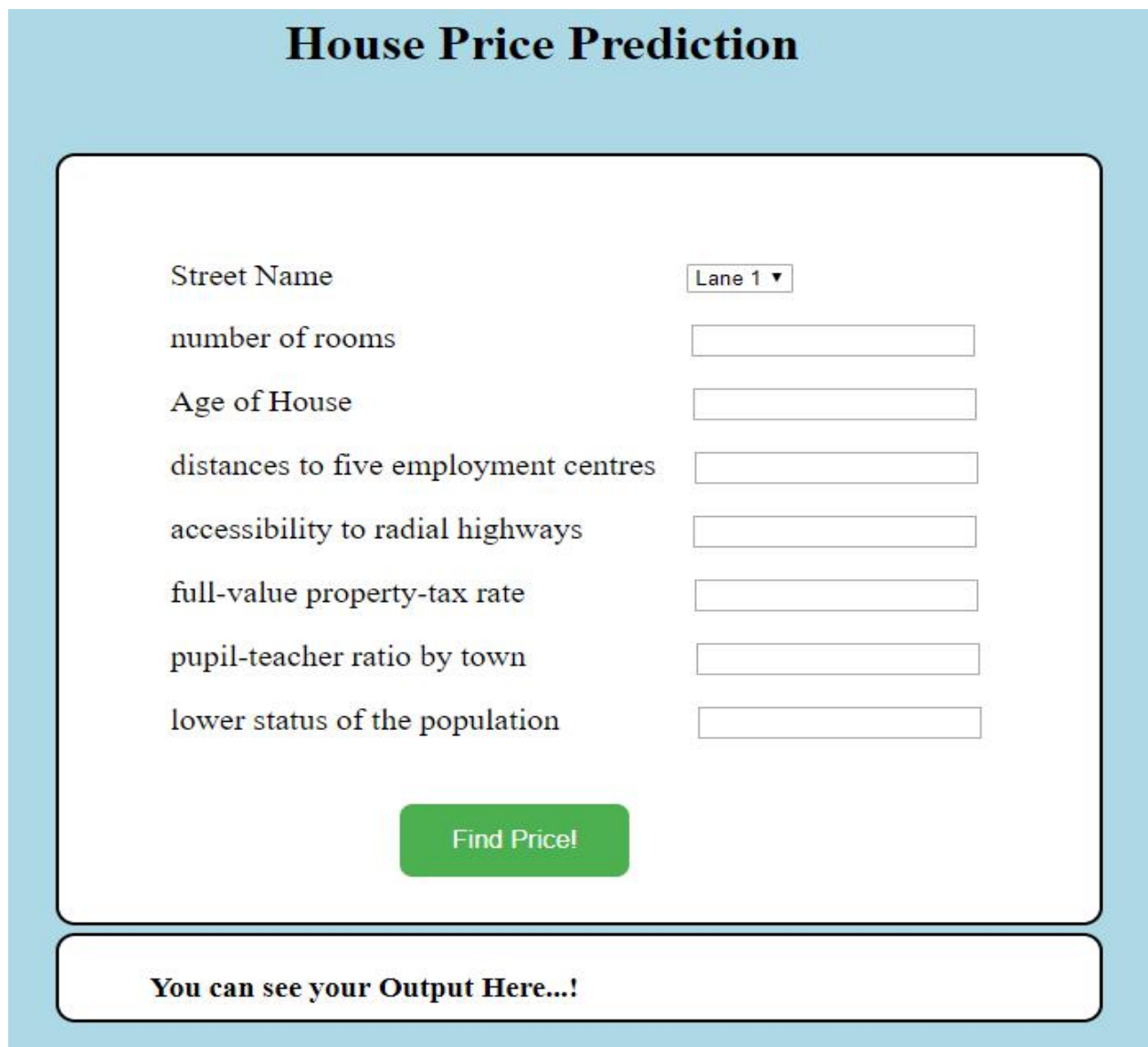


Figure 4.1 System Architecture

4.2 UI DESIGN

HTML and Bootstrap are used to develop the front end. The Graphical user interface, is a type of user interface that allows user to interact with electronic devices through graphical icons and visual indicators such as secondary notation, instead of text-based user interfaces, typed command labels or text navigation. Nearly all digital interfaces are GUIs.



The screenshot displays a web application titled "House Price Prediction" in a bold, black serif font, centered at the top of a light blue header bar. Below the header, a white rectangular form with a thin black border contains the input fields. The form is organized into two columns. The left column lists eight attributes: "Street Name", "number of rooms", "Age of House", "distances to five employment centres", "accessibility to radial highways", "full-value property-tax rate", "pupil-teacher ratio by town", and "lower status of the population". The right column contains a dropdown menu labeled "Lane 1" with a downward arrow, followed by seven empty text input boxes corresponding to the attributes on the left. A prominent green button with the text "Find Price!" is positioned below the input fields. At the bottom of the form, a white box with a black border contains the text "You can see your Output Here...!".

House Price Prediction	
Street Name	Lane 1 ▼
number of rooms	<input type="text"/>
Age of House	<input type="text"/>
distances to five employment centres	<input type="text"/>
accessibility to radial highways	<input type="text"/>
full-value property-tax rate	<input type="text"/>
pupil-teacher ratio by town	<input type="text"/>
lower status of the population	<input type="text"/>
<input type="button" value="Find Price!"/>	
You can see your Output Here...!	

Figure 4.2 UI Screenshot.

4.3 CLASS DIAGRAM

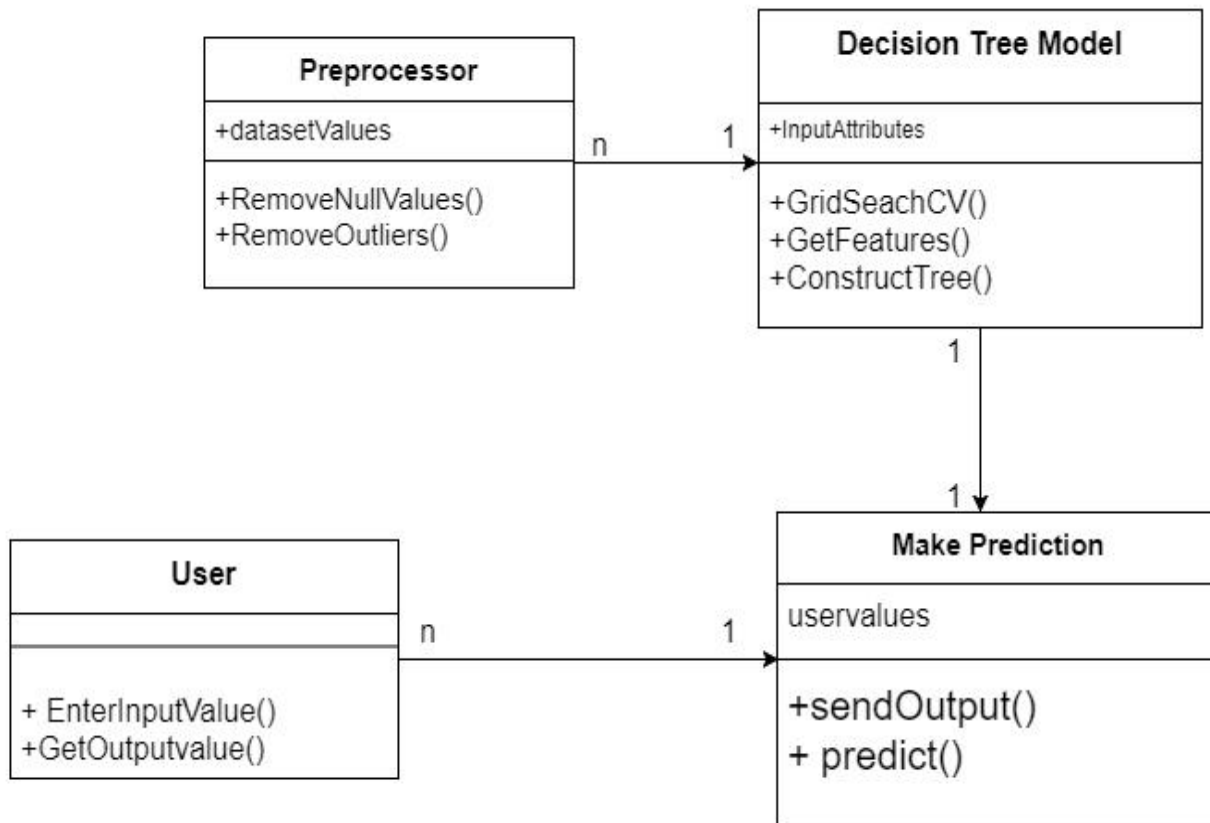


Figure 4.3 Class Diagram

4.4 MODULE DESIGN

4.4.1 PRE PROCESSING

Data preprocessing is a data mining technique that involves transforming raw data into an understandable format. Real world data is often incomplete, inconsistent, and/ or lacking in certain behavior or trends, and is likely to contain many errors. Data preprocessing is a proven method of resolving such issues.

- ❖ **Data cleaning:** fill in missing values, smooth noisy data and resolve inconsistencies.
- ❖ **Data Transformation:** normalization and aggregation.
- ❖ **Data Reduction:** reducing the volume but producing the same or similar analytical results.
- ❖ **Data discretization:** part of data reduction, removing numerical attributes and special characters.

We will make a cursory investigation about the Boston housing data and provide our observations. Familiarizing ourself with the data through an explorative process is a fundamental practice to help you better understand and justify your results.

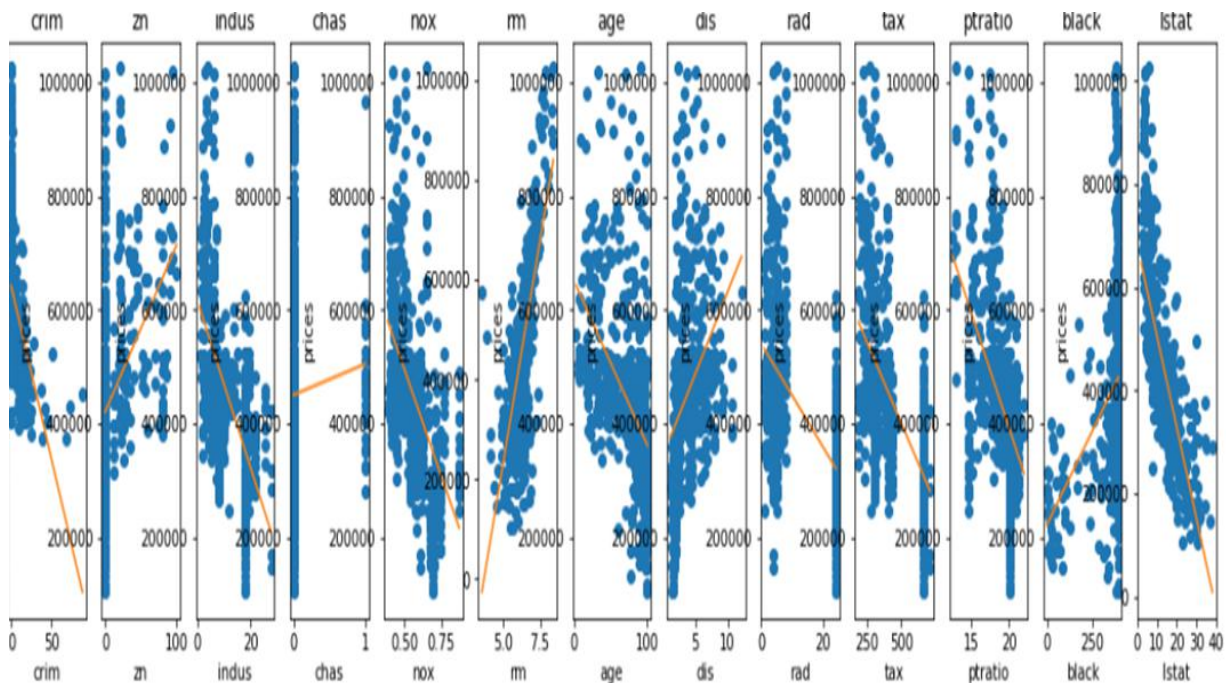


Figure 4.4 Data Exploration

4.4.2 ANALYSING THE MODEL

In this section, we'll take a look at several models' learning and testing performances on various subsets of training data. Additionally, you'll investigate one particular algorithm with an increasing 'max_depth' parameter on the full training set to observe how model complexity affects performance. Graphing your model's performance based on varying criteria can be beneficial in the analysis process, such as visualizing behavior that may not have been apparent from the results alone.

GRID SEARCH CV

In essence, the grid search technique allows one to define a grid of parameters that will be searched using K-fold cross-validation. Importantly, the grid search technique exhaustively tries every combination of the provided hyper-parameter values in order to find the best model. One can then find the highest cross-validation accuracy that matches with the corresponding parameters that optimizes the learning algorithm.

we'll take a look at several models' learning and testing performances on various subsets of training data. Additionally, we'll investigate one particular algorithm with an increasing 'max_depth' parameter on the full training set to observe how model complexity affects performance. Graphing our model's performance based on varying criteria can be beneficial in the analysis process, such as visualizing behavior that may not have been apparent from the results alone.

Learning Curves

The following code cell produces four graphs for a decision tree model with different maximum depths. Each graph visualizes the learning curves of the model for both training and testing as the size of the training set is increased. Note that the shaded region of a learning curve denotes the uncertainty of that curve (measured as the

standard deviation). The model is scored on both the training and testing sets using R^2 , the coefficient of determination.

Complexity Curves

The following code cell produces a graph for a decision tree model that has been trained and validated on the training data using different maximum depths. The graph produces two complexity curves — one for training and one for validation. Similar to the learning curves, the shaded regions of both the complexity curves denote the uncertainty in those curves, and the model is scored on both the training and validation sets using the `performance_metric` function.

4.4.3 MODEL TRAINING

To ensure that you are producing an optimized model, we will train the model using the grid search technique to optimize the 'max_depth' parameter for the decision tree. The 'max_depth' parameter can be thought of as how many questions the decision tree algorithm is allowed to ask about the data before making a prediction. Decision trees are part of a class of algorithms called supervised learning algorithms.

For the `fit_model` function in the code cell below, we will need to implement the following:

Use `DecisionTreeRegressor` from `sklearn.tree` to create a decision tree regressor object. Create a dictionary for 'max_depth' with the values from 1 to 10, and assign this to the 'params' variable. Create a scoring function object. Use `GridSearchCV` from `sklearn.grid_search` to create a grid search object.



4.4.4 MAKING PREDICTION:

Our model has been trained on a given set of data, it can now be used to make predictions on new sets of input data. In the case of a decision tree regressor, the model has learned what the best questions to ask about the input data are, and can respond with a prediction for the target variable. You can use these predictions to gain information about data where the value of the target variable is unknown — such as data the model was not trained on.

Imagine that you were a real estate agent in the Boston area looking to use this model to help price homes owned by your clients that they wish to sell. You have collected the following information from your clients:

1. per capita crime rate by town(CRIM)
2. proportion of residential land zoned for lots over 25,000 sq.ft.(ZN)
3. proportion of non-retail business acres per town.(INDUS)
4. Charles River dummy variable (1 if tract bounds river; 0 otherwise)(CHAS)
5. nitric oxides concentration (parts per 10 million)(NOX)
6. average number of rooms per dwelling(RM)
7. proportion of owner-occupied units built prior to 1940(AGE)
8. weighted distances to five Boston employment centres(DIS)
9. index of accessibility to radial highways(RAD)
10. full-value property-tax rate per \$10,000(TAX)
11. pupil-teacher ratio by town(PTRATIO)
12. % of lower status of the population(LSTAT)

By giving these informations as a input to the trained model. Our Model will predict the Median value of house i.e., Sale price of the house.

4.5 COMPLEXITY ANALYSIS

4.5.1 TIME COMPLEXITY

The time complexity of all preprocessing steps is $O(N)$. In case of Decision Tree the depth of a tree is $O(\log n)$, where n is the number of rows of data and the tree is assumed to be relatively balanced. For each of the splits in the tree, you will need to test every feature for all values to determine the value split that minimizes the loss function.

Assuming number of features is m , the run time is $O(mn \log n)$.

4.5.2 COMPLEXITY OF THE PROJECT

Data collected from a rural city may not be applicable as the demographics would change and other features may be better able to fit the dataset instead of a model with features that was learned using urban data.

The learning algorithm learned from a very old dataset that may not be relevant because demographics have changed a lot since 1978.

There are only 3 features currently, there are more features that can be included such as crime rates, nearby to city, public transport access and more.

CHAPTER 5

SYSTEM DEVELOPMENT

5.1 PROTOTYPE ACROSS THE MODULES

The input and output to each module of the system is described in the section.

5.1.1 PREPROCESSOR

This module takes the house specifications like no of rooms, area, crime rate etc... as input and produces as intermediary output the text without special characters and numbers. This is followed by tokenizing the text and removing stop words. A bag of words approach is used. The countvectorizer produces the bag of words.

5.1.2 ANALYZING THE MODEL

This module takes the preprocessed data as a input and uses the attributes like Features (all features except the target variable) and Prices(target variable). It produces the learning curves for model and Complexity curve for overall model to explore the best parameter. This uses matplotlib for drawing curves.

5.1.3 TRAINING THE MODEL

This module takes the data which we split as a Train and Test. It uses the Train dataset as a input an gives the optimal Trained model as a output. Use DecisionTreeRegressor from sklearn.tree to create a decision tree regressor object.

5.1.4 MAKE PREDICTION

This is our Final module which takes the user given values as a input and predict the best sale price of the house by fitting the input values to our model and shows as the output value as a sale price of a house.

5.2 ALGORITHM

The **GridSearch CV** algorithm used for predict the House Price is shown below

```
def gridsearch(loop):  
    param_opt = []  
    alpha_opt = []  
    l1_ratio_opt = []  
    for x in range(loop):  
        n_splits = 10 #the number of groups  
        k_fold = KFold(n_splits=n_splits, shuffle=True) #make dataset random  
        rmse_10 = []  
        intercept = []  
        alpha = [0.1, 0.5, 1.0, 10]  
        l1_ratio = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]  
        best_score = 100  
        for i in alpha:  
            for j in l1_ratio:  
                clf = linear_model.ElasticNet (alpha = i, l1_ratio = j)  
                for train_index, test_index in k_fold.split(X): #check the overfitting  
                    X_train = X[train_index]  
                    y_train = y[train_index]  
                    X_test = X[test_index]  
                    y_test = y[test_index]  
                    fit = clf.fit(X_train, y_train)  
                    y_pred = clf.predict(X_test)  
                    rmse_10.append(mean_squared_error(y_test, y_pred)**0.5)  
        rmse_mean = np.mean(rmse_10)
```

```

    rmse_10 = []
    if rmse_mean < best_score:
        best_score = rmse_mean
        best_alpha = i
        best_ll_ratio = j
    param_opt.append([best_alpha, best_ll_ratio])
    alpha_opt.append(best_alpha)
    ll_ratio_opt.append(best_ll_ratio)
df_alpha = pd.DataFrame({'alpha' : alpha_opt})
df_ll_ratio = pd.DataFrame({'ll' : ll_ratio_opt})
df_param = pd.concat([df_alpha, df_ll_ratio],axis=1)
df_param =
df_param.groupby(['alpha','ll']).size().sort_values(ascending=False).reset_index(name
='size')
alpha = df_param.iloc[0, 0]
ll_ratio = df_param.iloc[0, 1]
return alpha, ll_ratio, df_param

```

5.3 DEPLOYMENT DETAILS

Python libraries such as Pandas, numpy, sklearn, etc... must be available in the deployment environment with support to deploy NodeJS application. Apart from these front end libraries such as bootstrap and jquery must be made available either locally made in server or through CDN.

CHAPTER 6

RESULTS AND DISCUSSION

6.1 DATASET FOR TESTING

The Housing Dataset consists of price of houses in various places in Boston. Alongside with price, the dataset also provide information such as Crime (CRIM), areas of non-retail business in the town (INDUS), the age of people who own the house (AGE) et... However, because we are going to use scikit-learn, we can import it right away from the scikit-learn itself. We will use several python libraries as required here.

6.1.1 DATASET NAMING

- CRIM - per capita crime rate by town
- ZN - proportion of residential land zoned for lots over 25,000 sq.ft.
- INDUS - proportion of non-retail business acres per town.
- CHAS - Charles River dummy variable (1 if tract bounds river; 0 otherwise)
- NOX - nitric oxides concentration (parts per 10 million)
- RM - average number of rooms per dwelling
- AGE - proportion of owner-occupied units built prior to 1940
- DIS - weighted distances to five Boston employment centres
- RAD - index of accessibility to radial highway.
- 10.TAX - full-value property-tax rate per \$10,000
- PTRATIO - pupil-teacher ratio by town
- B - $1000(B_k - 0.63)^2$ where B_k is the proportion of blacks by town
- LSTAT - % lower status of the population
- MEDV - Median value of owner-occupied homes in \$1000's

6.2 OUTPUT OBTAINED IN VARIOUS STAGES

This section shows the results obtained during module testing.

6.2.1 INPUT

House Price Prediction

Street Name	<input type="text" value="Lane 1"/>
number of rooms	<input type="text"/>
Age of House	<input type="text"/>
distances to five employment centres	<input type="text"/>
accessibility to radial highways	<input type="text"/>
full-value property-tax rate	<input type="text"/>
pupil-teacher ratio by town	<input type="text"/>
lower status of the population	<input type="text"/>

Find Price!

You can see your Output Here...!

Figure 6.1 Sample Input.

6.2.2 MODEL ANALYSING

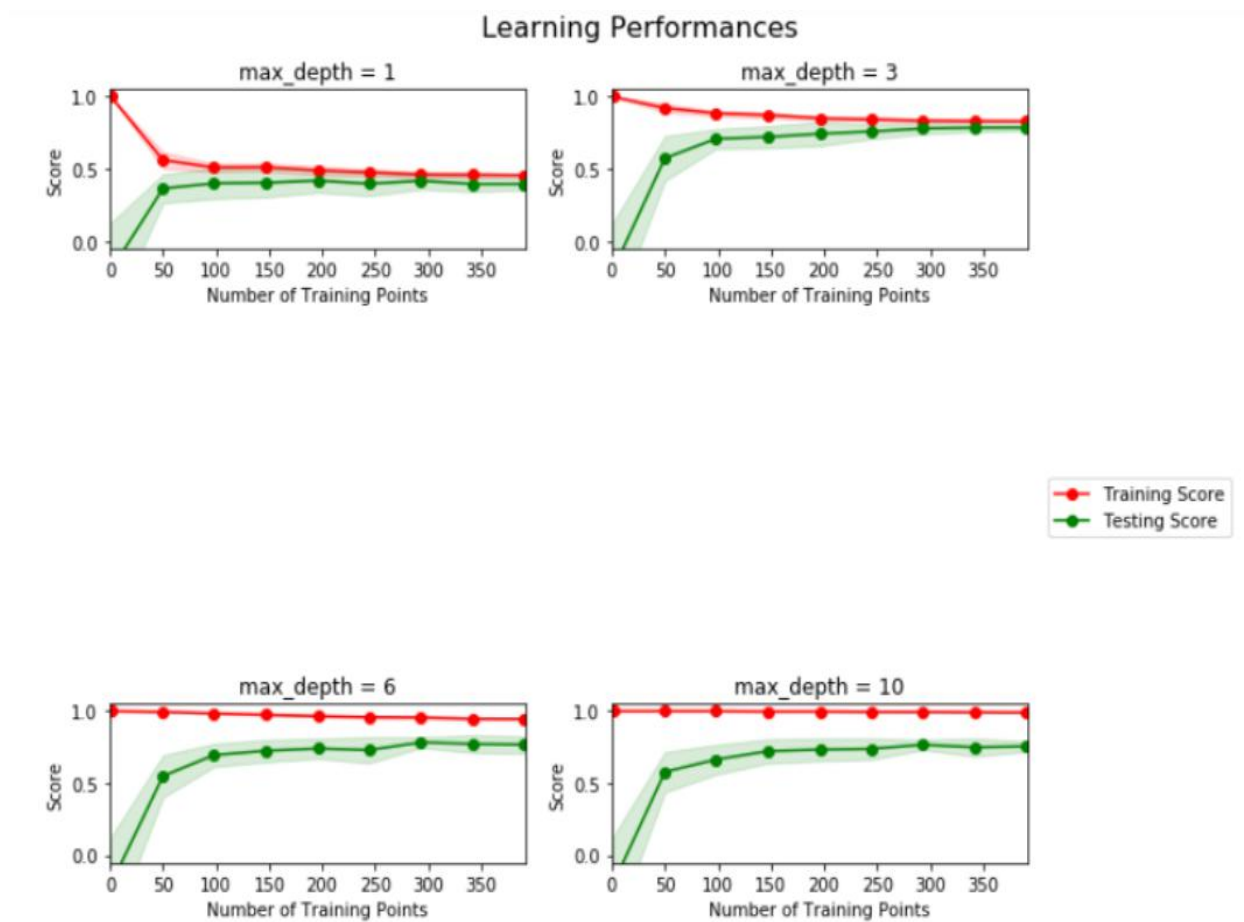


Figure 6.2 Model Analyzing


6.2.3 MODEL TRAINING

Parameter 'max_depth' is 4 for the optimal model.

```
: {'criterion': 'mse',  
   'max_depth': 4,  
   'max_features': None,  
   'max_leaf_nodes': None,  
   'min_impurity_decrease': 0.0,  
   'min_impurity_split': None,  
   'min_samples_leaf': 1,  
   'min_samples_split': 2,  
   'min_weight_fraction_leaf': 0.0,  
   'presort': False,  
   'random_state': 0,  
   'splitter': 'best'}
```

Figure 6.3 Model Training

6.2.4 PRICE PREDICTION

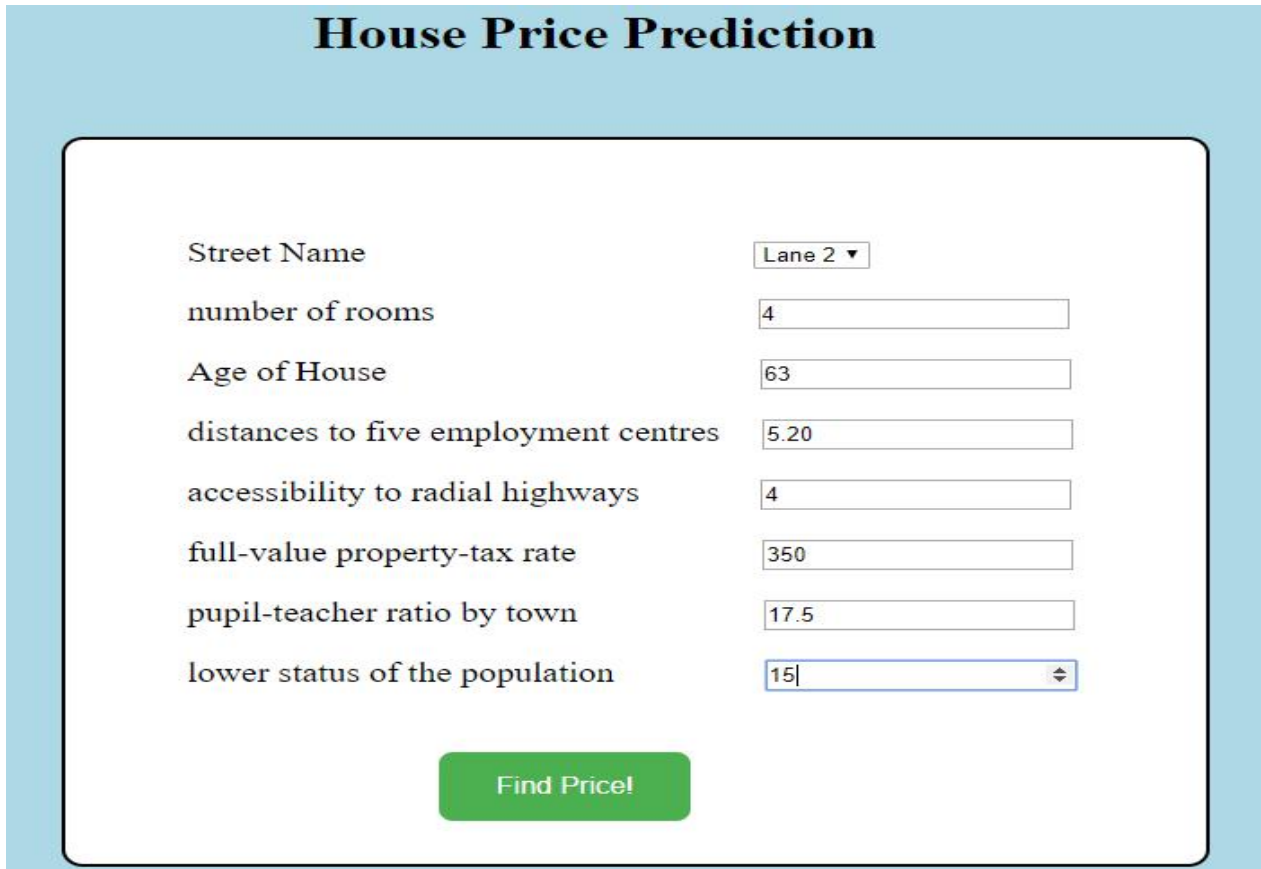


Predicted selling price for Client's home: \$232,200.00

Figure 6.4 Price Prediction.

6.3 SAMPLE SCREENSHOTS DURING TESTING

6.3.1 INPUT




A screenshot of a web application titled "House Price Prediction". The form contains several input fields for user data. The fields are arranged in two columns. The first column contains labels for the inputs, and the second column contains the input values. The inputs are: Street Name (Lane 2), number of rooms (4), Age of House (63), distances to five employment centres (5.20), accessibility to radial highways (4), full-value property-tax rate (350), pupil-teacher ratio by town (17.5), and lower status of the population (15). A green button labeled "Find Price!" is located at the bottom center of the form.

Input Field	Value
Street Name	Lane 2
number of rooms	4
Age of House	63
distances to five employment centres	5.20
accessibility to radial highways	4
full-value property-tax rate	350
pupil-teacher ratio by town	17.5
lower status of the population	15

Find Price!

Figure 6.5 Sample Input

6.3.2 OUTPUT



A screenshot of the output of the "House Price Prediction" application. It shows a single line of text: "Predicted selling price for Client's home: \$232,200.00".

Predicted selling price for Client's home: \$232,200.00

Figure 6.6 Sample Output

6.4 PERFORMANCE EVALUATION

The performance of the entire system is evaluated using the standard parameters described below.

6.4.1 R-SQUARED SCORE

6.4.1.1 Description

R-squared is a statistical measure of how close the data are to the fitted regression line. It is also known as the coefficient of determination, or the coefficient of multiple determination for multiple regression.

R-squared is always between 0 and 100%:

- 0% indicates that the model explains none of the variability of the response data around its mean.
- 100% indicates that the model explains all the variability of the response data around its mean.

The usefulness of R^2 is its ability to find the likelihood of future events falling within the predicted outcomes. The idea is that if more samples are added, the coefficient would show the probability of a new point falling on the line.

Even if there is a strong connection between the two variables, determination does not prove causality. For example, a study on birthdays may show a large number of birthdays happen within a time frame of one or two months. This does not mean that the passage of time or the change of seasons causes pregnancy.

6.4.1.2 Algorithm

To Calculate the R-squared performance score between 'y_true' and 'y_predict'

```
# y_true = np.array(y_true)

# y_predict = np.array(y_predict)

# y_mean = y_true.mean()

# tot = sum((y_true - y_mean)**2)

# res = sum((y_predict - y_true)**2)

# score = 1 - res/tot
```

The definition of R-squared is fairly straight-forward; it is the percentage of the response variable variation that is explained by a linear model. Or:

$$\text{R-squared} = \text{Explained variation} / \text{Total variation}$$

6.4.1.3 Score obtained and discussion

The R-squared Score of our built model is achieved more than 85 %. The accurate value found by value is **0.8567784606316289**.

From the acquired score we can found that our model will predict the price of house at the accuracy of more than 85% . It is a very good accuracy value achieved by applying a limited dataset. This score is further increased by adding feature engineering and by some good data pruning which we are included in the Future work section.

The only drawback is there are only 14 datapoints here. So this might not be statistically significant. Another caveat is that whether the model is successful also

depends largely on the application. So for some projects 0.856 is sufficient, whereas for others it could be a low score. Here there is a very high chance of getting a high variance model which may eventually lead to a 100% accuracy rate with addition of new features, but that's only because it is overfitting the data. It has developed such a complex model that it will have limited or no ability to generalize data and so when we use that model on unknown data, it will give us very very low accuracy. So to avoid that, we can split the data into training and testing sets and train the model on the training data. Then the testing accuracy is a much better estimate than the training accuracy.

But then, the split might create a problem too. If we have a very limited dataset, then even if we take out a small sample of it as testing data, then also, we are losing a portion of the data. So there's an inherent trade off here which might cause underfitting due to limited datasets. This is where we can take advantage of K-fold cross validation where we divide all the datapoints into k number of bins and then run k separate learning experiments. In each of those, we pick one of those k subsets as our testing set and the remaining k-1 bins as our training sets. This is how we can maximize the machine's learning experiment.

CHAPTER 7

CONCLUSIONS

7.1 SUMMARY

Based on the arguments in all previous three chapters, the following conclusions are identical:

Firstly, model fitting result from linear regression method makes sense: after preliminary statistical analysis by numerical and graphical examination, linear regression model was proven to be logical methodology for analysis Boston Housing data; Next, linear regression fitting results were reasonable enough, as interpretation of both the estimated model and influence index are rational in terms of common sense; what's more, different variable selection procedures were compared to ensure that our final model was appropriate; finally, detailed model diagnosis, regarding all five assumptions, which are linearity, normality, independent observations, constant variance, no outliers, was performed to validate our model.

Secondly, effects of adding regularization on the linear regression model varies: Ridge regression and Elastic net both improve the original linear regression mode in terms of average estimation error, i.e., residual standard error, and prediction power, mean prediction error (MPE), while the improvement from Elastic net, the hybrid of Ridge and LASSO regularization, was larger than Ridge regression. These result make sense, as the Boston Housing data is of good statistical property: sample size is 506, large enough for estimating 13 unknown parameters in the final model;

Multicollinearity problem is not severe at all after 'INDUS' and 'AGE' being removed from the linear regression model; over-fitting is not a problem for the Boston Housing data. As a result, regularized regression, which is to fix these issues, will

improve the model by a rather limited level. Even though these improvements are kind of small, the proposed regularized regression model is still meaningful: it did provide better modeling results; it may yield significant improved results for other problem in the field of real estate.

7.2 FUTURE WORK

Some possible future works that are relevant to this project are listed below:

1. Extending linear regression model by other modern regularization term, such as the new hybrid term offered in Owen (2007) [43], could be implemented to see if it provides better model fitting results.
2. The effect of partition into training data and testing data could be studied by repeat the random partition process for a large number of times, for each partition, regression model should be fitted. Then examination about the variation of fitted models should be able to answer this question appropriately.
3. Applying our methodology to other real estate data will be helpful to confirm its effectiveness and generality.
4. The effect of partition into training data and testing data could be studied by repeat the random partition process for a large number of times, for each partition, regression model should be fitted. Then examination about the variation of fitted models should be able to answer this question appropriately.

REFERENCES

- [1] Yu, Jiafu Wu. 2016. Real Estate Price Prediction with Regression and Classification, CS 229 Autumn 2016 Project Final Report, Stanford University.
- [2] Y.-C. Wang, R. Huang, C.-C. Nieh, H.-K. Ou, and M. Chi, 2017. Integration between real estate market and the stock market: Evidence from Taiwan, 2017 International Conference on Applied System Innovation (ICASI).
- [3] L.Li, K.-H. Chu, 2017. Prediction of real estate price variation based on economic parameters, 2017 International Conference on Applied System Innovation (ICASI).
- [4] Nissan Pow, Emil Janulewicz, Liu (Dave) Liu, 2016. Applied Machine Learning Project 4 Prediction of real estate property prices in Montreal: An application of Random Forest, McGill University.
- [5] B. Trawinski, Z. Telec, J. Krasnoborski, M. Piwowarczyk, M. Talaga, T. Lasota, and E. Sawilow, 2017. Comparison of expert algorithms with machine learning models for real estate appraisal, 2017 IEEE International Conference on Innovations in Intelligent Systems and Applications (INISTA).
- [6] E. Antipov and E. Pokryshevskaya, 2010. Mass Appraisal of Residential Apartments: An Application of Random Forest for Valuation and a CART-Based Approach for Model Diagnostics, SSRN Electronic Journal.
- [7] J.-G. Liu, X.-L. Zhang, and W.-P. Wu, 2006. Application of Fuzzy Neural Network for Real Estate Prediction, Advances in Neural Networks - ISNN 2006 Lecture Notes in Computer Science, pp. 1187-1191.

- [8] E. Hromada, 2015. Mapping of Real Estate Prices Using Data Mining Techniques, *Procedia Engineering*, vol. 123, pp. 233-240.
- [9] Byeonghwa Parka Jae, Kwon Bae, 2014. Using machine learning algorithms for housing price prediction: The case of Fairfax County, Virginia housing data: *Expert Syst. Appl.* 42(6), 2928-2934 (2014)
- [10] Da-Ying Li, Wei Xu, Hong Zhao, Rong-Qiu Chen, 2009. A SVR based forecasting approach for real estate price prediction, *International Conference on Machine Learning and Cybernetics*, Baoding.