

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belagavi-590018, Karnataka, INDIA



PROJECT PHASE-2 REPORT

on

“Housing Price Prediction”

Submitted in partial fulfillment of the requirements for the VIII Semester

Bachelor of Engineering IN COMPUTER SCIENCE AND ENGINEERING

For the Academic year

2018-2019

BY

Mohammed Yaseen

Shubharthi Dey

Saleha Afsari

1PE15CS086

1PE15CS157

1PE16CS421

Under the Guidance of

Prof. Lakshmi Nagaprasanna R

Assistant Professor

Dept. of CSE

PESIT-BSC, Bengaluru-560100



Department of Computer Science and Engineering

PESIT BANGALORE SOUTH CAMPUS

Hosur Road, Bengaluru -560100

PESIT BANGALORE SOUTH CAMPUS

Hosur Road, Bangalore -560100

Department of Computer Science and Engineering



CERTIFICATE

*Certified that the project work entitled “**Housing Price Prediction**” is a bonafide work carried out by **Mohammed Yaseen, Shubharthi Dey and Saleha Afsari** bearing USN **1PE15CS086, 1PE15CS157 and 1PE16CS421** respectively, students of **PESIT Bangalore South Campus** in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the **Visvesvaraya Technological University, Belagavi** during the year 2018-2019. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated and the project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.*

Signatures:

Project Guide
Prof. Lakshmi Nagaprasanna R
Assistant Professor,
Dept. of CSE
PESIT-BSC, Bengaluru

Head Dept of CSE
Dr. Sandesh B J
Prof. and HOD,
Dept. of CSE,
PESIT-BSC, Bengaluru

Director/Principal
Dr. J. Suryaprasad
PESIT-BSC, Bengaluru

External Viva

Name of the Examiners

Signature with date

1. _____

2. _____

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned my effort with success.

We are indebted to our Guide, **Prof. Lakshmi Nagaprasanna R**, Assistant Professor, Department of Computer Science and Engineering, PESIT - Bangalore South Campus, who has not only coordinated our work but also given suggestions from time to time.

We are also extremely grateful to our Project Co-ordinators, **Prof. Keerti Torvi**, Assistant Professor, **Prof. Sangeetha R**, Assistant Professor, **Prof. Jyoti Desai**, Assistant Professors, Department of Computer Science and Engineering, PESIT Bangalore South Campus, for their constant support and advice throughout the course of preparation of this document.

We are greatly thankful to **Dr. Sandesh B J**, Professor and HOD, Department of Computer Science and Engineering, PESIT Bangalore South Campus, for his able guidance, regular source of encouragement and assistance throughout this project.

We would like to express our immense gratitude to **Dr. J. Suryaprasad**, Director and Principal, PESIT Bangalore South Campus, for providing us with excellent infrastructure to complete our project work.

We gratefully acknowledge the help lent out to us by all faculty members of the Department of Computer Science and Engineering, PESIT Bangalore South Campus, at all difficult times. We would also take this opportunity to thank our college management for the facilities provided during the course of the project. Furthermore, we acknowledge the support and feedback of my parents and friends.

Mohammed Yaseen
Shubharthi Dey
Saleha Afsari

ABSTRACT

The real estate market is one of the most competitive in terms of pricing and the same tends to vary significantly based on a lot of factors, hence it becomes one of the prime fields to apply the concepts of machine learning to optimize and predict the prices with high accuracy.

Therefore in this project, we present various important features to use while predicting housing prices with good accuracy. While using features in a regression model some feature engineering is required for better prediction. Often a set of features (multiple regressions) or polynomial regression (applying a various set of powers in the features) is used for making better model fit. While in Neural Networks, solely the difference between the predicted and existing values determines the change in weights, hence determining the best fit.

This project thus directs to the best application of regression models in comparison to the neural networks model towards the same goal and examines the efficiency of each with respect to each other.

The project will be implemented as a web application using python machine learning libraries.

CONTENTS

Acknowledgement	i
Abstract	ii
Contents	iii
List of figures	vi

Chapter 1

1.	Introduction	1
1.1	Purpose of the project	1
1.2	Scope	2
1.3	Abbreviations	2
1.4	Literature survey	2
1.5	Existing System	4
1.6	Proposed system	4
1.7	Statement of the Problem	4
1.8	Summary	4

Chapter 2

2.	Software Requirements Specifications	5
2.1	Operating Environment	5
2.1.1	Hardware Requirements	5
2.1.2	Software Requirements	5
2.2	Functional Requirements	6
2.3	Non-functional requirements	6
2.4	User characteristics	7
2.5	Applications of System	7
2.6	Summary	7

Chapter 3

3.	High Level Design	8
3.1	Design Approach	8
3.2	System Architecture	8
3.3	Sequence Diagram	9

3.4	Data Flow Diagram	10
3.4	Summary	11
Chapter 4		
4.	Detailed design	12
4.1	Purpose	12
4.2	Training Dataset	12
4.3	User Interface Design	12
4.5	Module 1: Data Processing	13
4.6	Module 2: Prediction Modelling and Evaluation	14
4.7	Summary	17
Chapter 5		
5.	Implementation	18
5.1	Programming Language Selection	18
5.1.1	Overview of Python	18
5.1.2	Overview of Flask framework	18
5.1.3	Overview of Jupyter notebook using Anaconda	19
5.2	Platform Selection	19
5.3	Steps in Implementation	19
5.4	Summary	20
Chapter 6		
6.	Testing	21
6.1	Software Testing	21
6.2	Testing Process	21
6.3	Levels of Testing	23
6.3.1	Functional Testing	23
6.3.2	Integration Testing	23
6.3.3	System Testing	24
6.3.4	Acceptance Testing	24
6.3.5	Validation	24

6.3.6	Compilation Test	24
6.3.7	Output Test	25
6.4	Screen Shots	26
6.5	Summary	29
Chapter 7		
7.	Conclusion	30
7.1	Conclusion	30
7.2	Limitations	30
7.3	Summary	30
Bibliography		31

LIST OF FIGURES

Fig No.	Description	Page no
3.1	System Architecture(SNN)	9
3.2	System Architecture (RNN)	9
3.3	Sequence Diagram	10
3.4	Data Flow Diagram	11
4.1	Training Dataset	12
4.2	Home(UI)	13
4.3	Graph-1(Left Skewed)	13
4.4	Graph-2(Centre Skewed)	14
4.5	Structured Neural Network (Graph)	15
4.6	XGBoost (Graph)	16
4.7	Recurrent Neural Network (Graph)	17
6.1	Black-Box Testing	22
6.2	White- Box Testing	22
6.3	Gray-Box Testing	22
6.4	Input of XGBoost	26
6.5	Output of XGBoost	26
6.6	Input of RNN (LSTM)	27
6.7	Output of RNN (LSTM)	27
6.8	Input of SNN	28
6.9	Output of SNN	28

Chapter 1

Introduction

The real estate market is a standout amongst the most focused regarding pricing and the equivalent will in general shift fundamentally dependent on a ton of components, consequently it winds up one of the prime fields to apply the ideas of machine learning to enhance and anticipate the prices with high accuracy.

Therefore in the project, we present different significant highlights to utilize while predicting lodging prices with great accuracy. While utilizing highlights in a regression model some element designing is required for better forecast. Regularly a lot of highlights (different regressions) or polynomial regression (applying a different arrangement of forces in the highlights) is utilized for improving model fit. While in Neural Networks, exclusively the contrast between the anticipated and existing qualities decides the adjustment in loads, consequently deciding the best fit.

Even among neural networks, we have different approaches we can follow in order to reduce the computation required by the system with no loss of accuracy. For example, structured neural networks get rid of all the unrequired connections. Simply speaking, they cut off the links between the result and the values not essential to it. This reduces computation time and at the same time, increases the accuracy. Then we have our recurrent neural networks which are just neural networks where the output of the network is passed again as the input to the same network recurrently. They are very useful in areas where the current output depends upon the input processed so far. Typically used for language processing, but used here for price evaluation.

1.1 Purpose

Given a set of values representing the various attributes of the property, our objective is to predict the selling price of the property by using structured neural networks and then using recurrent neural networks.

We will then compare the accuracy of the respective predictions by comparing it with the real selling price of the property.

With this, we will be able to establish the difference in the accuracies of both methods and will be able to tell with certainty which is closer to the real deal and by the same line of reasoning, which is a better model.

1.2 Scope

The scope of our project is somewhat specific. It is intended to be used by architects, investors and government policy makers. Estimating the net worth of the property is a modern day problem with no properly established solution. In our case, we are using Machine Learning to decide the worth of the property. And this helps them decide whether a project is worth investing in and if it is then what's a good investment to it ?

1.3 Abbreviations

- et al. = et alia (and others)

1.4 Literature Survey

- Comparison between fuzzy regression and ANN

Reza Ghodsi et. al. compared Fuzzy Regression and Artificial Neural Networks even though Fuzzy Regression performs superior to anything conventional linear regression, by reason of the way that the previously mentioned useful connections could likewise be resolved not with standing when there are no fresh qualities for both of the reliant or autonomous factors (or them two) or not with standing when they are as intervals or fuzzy numbers. The fact still remains that the MAPE for both the models were compared and the ANN model performed relatively well with respect to Fuzzy Regression.

- Classical rule based classifier RIPPER vs Naive Bayes, AdaBoost.

Byeonghwa Park, Jae Kwon Bae compared a classical rule-based classifier RIPPER to various other algorithms like Naive Bayes, AdaBoost. The use of RIPPER is justified by the reasoning that RIPPER is able to handle imbalanced class distribution and more robust to noisy data. The study concluded accuracy of RIPPER was better than Naive Bayes by approximately 16% and better than AdaBoost roughly by 7%. But again, it was implemented in a certain country hence it failed to cover the overall pricing across the globe. Performance evaluation was done only on the basis of certain chosen classifiers which might not prove the basis of the research paper. With this, we thought it might be a good idea to move towards the neural networks side of the problem, gain a new perspective.

- ANN vs ARMA

Wan Teng Lim et al. compared ANN vs. ARMA. Neural networks are proven to show better results compared to Linear or Multivariate Regression. Moreover, not all the data is linear. subsequently, regression may come up short. Then again, one of the potential impediments is that neural systems expect an adequately vast data set, which may not be accessible in all cases. Henceforth, the exhibition of the ANN isn't upgraded because of the absence of training data set and inadequate confirmation yield data. In addition, a portion of the data for the autonomous factors, for example, the populace and yearly expansion rate are changed over from yearly into quarterly premise; the precision of the anticipated yields will be influenced.

- Application of neural network in predicting housing prices

Tom Kauko et. al discusses about application of Neural Network in predicting housing prices. As expected, he concludes that the neural networks provide an accuracy much better than conventional models like regression and tree based algorithms. And the approach is fairly robust to noise. It comes handy when dealing with non-linear data. However, because of the black box nature of the strategy it will stay vague which practical relations represent the variety in results. The understanding of results depends vigorously on the expert knowledge of the researcher. Obviously the neural network approach isn't well-suited in testing predefined theories and that not with standing connecting results to existing hypothesis is tricky.

- Regression Model for Appraisal of Real Estate Using Recurrent Neural Network and Boosting Tree(IEEE)

Junchi Bin et. al. compared Recurrent NN to some well known algorithms like SVR, LASSO. Recurrent Neural Networks (RNNs) are a family of neural networks for processing sequential data Essentially ANNs in which output is given back into input in the next iteration/layer along with new input. Hence, input can be variable size. This method is usually used for analysis of speech, text etc. Some possible shortcomings include - The amount of training required which is comparatively much more than other kinds of networks. And secondly, it cannot be stacked into very deep models. This is mostly because of the saturated activation function used in RNN models, making the gradient decay over layers. Which is why we use a model called long short term memory model.

- Smart Real Estate Assessments using structured DNNs

Haiping Xu et. al. compared Structured DNN to other algorithms like Regression, fully connected deep neural networks etc. Neurons in a structured DNN are fundamentally associated, which makes the network existence proficient. a structured DNN with a diminished number of connections and neurons requires considerably less computational assets, yet would in any case lead to satisfactory or far and away superior outcomes. The structured DNN model has been intended to gain from the

most as of late caught information points; Therefore, it enables the model to adjust to the most recent market patterns. In spite of the fact that a DNN works more proficient than a shallow one, an average DNN for taking care of a down to earth issue may even now require a substantial number of shrouded neurons. Generally, a DNN is viewed as a black-box approach, where the semantics of the concealed neurons are obscure, and there is no reasonable method to decide an appropriate size of the system.

1.5 Existing System

Existing systems have established the fact that both of the suggested approaches i.e Structured neural Networks and Recurrent Neural Networks are better than any other Machine Learning techniques available to use to predict housing prices, including fully connected ANNs

1.6 Proposed System

In our system, we hope to find out the more superior of the two, and combine both the approaches to form a accurate, more efficient model.

1.7 Statement of the problem

This project directs to the best application of recurrent neural networks models in comparison to the structured neural networks model towards the common goal of predicting the real estate prices most accurately and compare how close each of them are to the correct output.

The project will be implemented as a web application using python machine learning libraries.

1.8 Summary

This chapter deals with the initial problem statement and provides a brief introduction to the topic. We define a few key terms and specify the literature survey carried out before beginning work on the project. We also show the systems presently available in the market and how ours differs from those systems.

Chapter 2

Software Requirements Specifications

The software requirements specification (SRS) spreads out functional and non-functional requirements, and may likewise incorporate a lot of utilization cases that portray client associations that the software must give. Software requirements specification permits an intensive examination of essentials before setup can begin and decreases later update. Software requirements specification develops the explanation behind a comprehension among customers and authoritative laborers or suppliers (in market-driven endeavor, these employments might be played by the exhibiting and improvement divisions) on what the item thing is to do too as what it isn't depended upon to do. It should in like manner give a sensible reason to surveying thing costs, threats, and timetables. Right when used fittingly, Software requirements specification can help balance programming adventure dissatisfaction.

2.1 Operating Environment

The operating environment gives a brief overview about the hardware and the software requirements of the system.

2.1.1 Hardware Requirements

- **Processor:** 2GHz or faster processor
- **RAM:** 2 GB(64bit)
- **Storage:** 5GB of available hard disk space
- Other general hardwares such as a mouse and keyboard for inputs and a monitor for display.

2.1.2 Software Requirements

- **Operating system:** Linux or Windows
- **Programming languages:** Python/R

2.1.3 Functional Requirements

Functional requirements are a formal way of expressing the expected services of a project. We have identified the functional requirements for our project as follows:

- The framework ought to have the option to gather data from data sets.
- The system should be able to check for correctness of data extracted from data sets.
- The framework ought to have the option to process the data before showing the accuracy and output.
- The system should be able to predict price correctly.

2.1.4 Non-Functional Requirements

In structures building and necessities planning, a non-functional requirement is an essential that demonstrates criteria that can be used to condemn the action of a system, rather than unequivocal practices. This should be showed up contrastingly in connection to pragmatic necessities that describe express lead or limits. Non-Functional Requirements are every now and again called as qualities of a system.

- Performance: The system should have good performance and ensure that all the operations provide good accuracy and reliable results.
- Usability: This ensures that the system is easily understood and can be used by any novice. The system has a user friendly interface and can be used by any person regardless of prior knowledge or experience.
- Correctness: The correctness of the results are ensured through rigorous testing and using the best and more robust algorithms.
- Maintainability: The straight forwardness with which the framework can be altered to address deformities or meet prerequisites, makes maintenance simpler.
- Scalability: The system should be scalable to include more functionalities.
- Availability: System availability is the amount of time for which the system can be running. Our system is run on demand and has no constraints on the usage time.

2.1.5 User Characteristics

The following type of users are considered in the making of this project-

- Architect : Architects can actually plan their construction projects according to the ongoing demand, the selling price, and the market trend.
- Investor : This model will give an idea to the investors about how profitable a given investment in the real estate would be, based on it's net worth to the user and the current market.
- Public policy makers : If this model can accurately predict the cost of the real estate, it can help in making policies for the general public. For instance, the tax to be paid for such kind of property.

2.1.6 Applications

There could be various applications of this system in the different aspects. One of the first things that comes to mind is what cost to expect for the features required by you when looking for a house. Also, given your price range, what features can u avail.

Second thing is as a landlord who wants to rent his property, you are entitled to know the best price possible.

2.1.7 Summary

In this chapter, the specific hardware and software requirements that must be adhered to during the development and usage of the product are elucidated. The functional and non-functional requirements were described along with the advantages and applications of this project.

Chapter 3

High Level Design

This section mainly covers the design technique of the entire system which involves the implementation of 2 modules which are as follows:

- **Structured neural network** : This is a variant of artificial neural network where the unnecessary links between a result and its contributing attributes are cut off. Simply speaking, a result will only consist of attributes that contribute significantly to its calculation. It helps in reducing non-essential computations.
- **Recurrent neural network**: Output of the network is passed as the input to the same network in order to be evaluated again, keeping track of its gradient.

3.1 Design Approach

Here are two methodologies for software designing:

- **Top-down Design**: It takes the entire programming framework as one entity and after that disintegrates it to accomplish in excess of one subsystem or some components based on few attributes.
- **Bottom-up Design**: The model begins with most particular and essential components. It accedes with making more elevated amount out of subsystems by utilizing essential or lower level components.

As mentioned above the project requires two main modules to be implemented. Each module has its own components to be developed. We use bottom-up design strategy in this product design phase as we start designing the basic components in each module and finally we interlink both the modules to get the final product.

3.2 System Architecture

System architecture is the determined model that portray the structure, lead and viewpoints on a system. It is a formal depiction and portrayal of a system, the basic structure of proposed system is depicted in the underneath figure 3.1 and figure 3.2

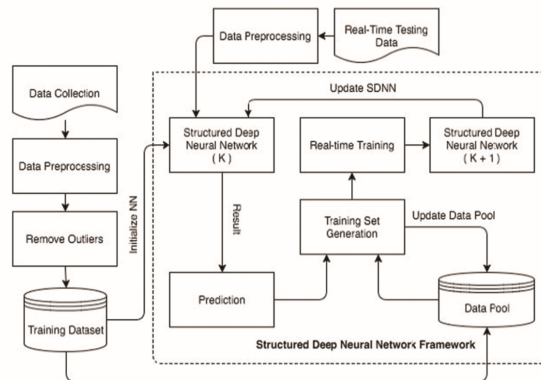


Figure 3.1: System Architecture (Structured Neural Network)

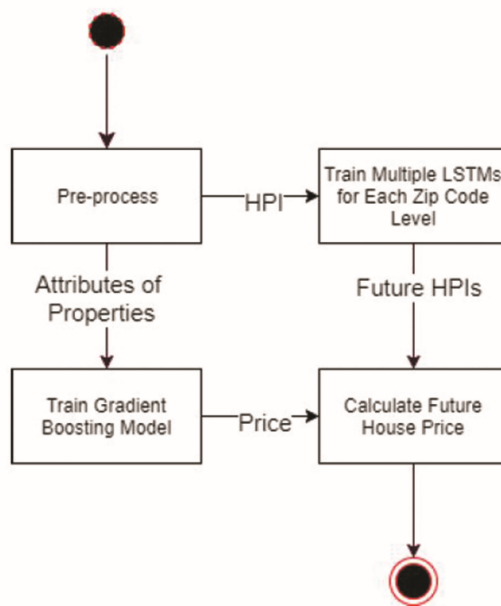


Figure 3.2: System Architecture (Recurrent Neural Network)

3.3 Sequence Diagram

A sequence diagram in UML is a sort of cooperation figure that indicates how forms work with each other. Figure 3.2 illustrates the same.

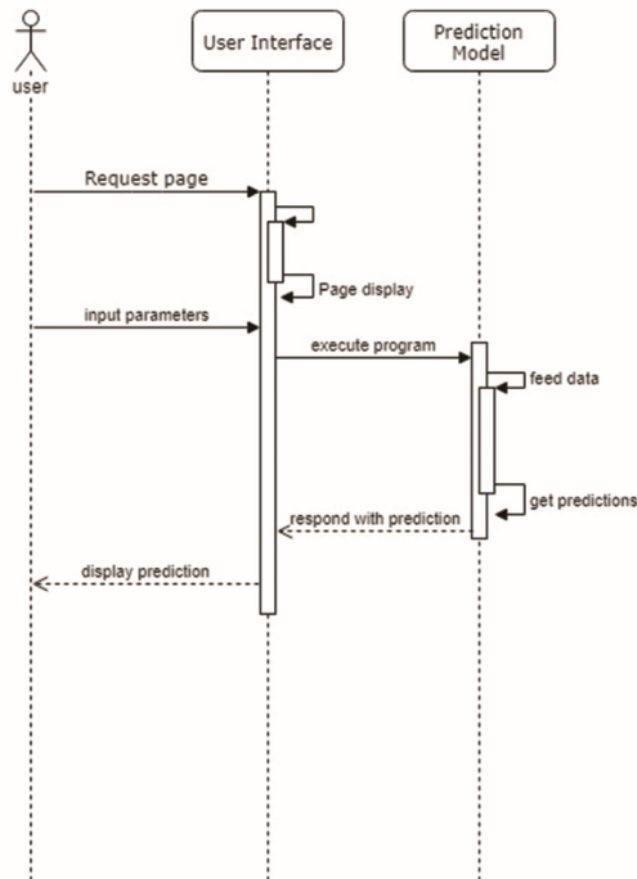


Figure 3.3: Sequence Diagram

3.4 Data Flow Diagram

Data Flow Diagram(DFD) is the beginning stage of the design phase. A DFD comprises of a progression of air pockets joined by lines. The air pockets speak to data change and the lines speak to data flows in the framework. A DFD portrays what data flow as opposed to how they are prepared, so it does exclude equipment, programming and data structure.

A Data Flow Diagram is a graphical depiction of the progression of information through an data structure. A data flow configuration is a critical demonstrating system for examining and developing data forms. DFD actually implies an illustration that clarifies the course or development of data in a procedure. DFD illustrates this flow of data in a procedure dependent on inputs and outputs.

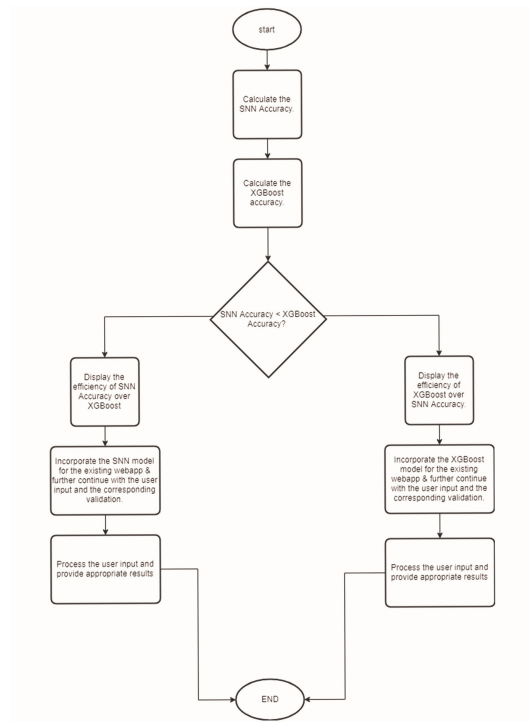


Figure 3.4: Data Flow Diagram

3.5 Summary

In this chapter we give a survey for the plan of the structure and how it is dealt with and the progression of information through the system. By scrutinizing this document the client ought to have a general perception of the issue and its answer.

Chapter 4

Detailed Design

4.1 Purpose

In this chapter, a high-level design for the interior rationale of each of the modules is chosen. It presents further details and algorithmic plan of each of the modules is indicated. Each low-level parts and sub-parts are additionally depicted. Each sub area of this segment contains an itemized depiction of a product framework part. Here, we talk about the control stream, with substantially more details about modules, by clearing up their motivation, usefulness, input and output.

4.2 Training Dataset

LotArea	LotShape	LotConfig	Neighborhood	BldgType	HouseStyle	OverallQual	OverallCond
8450	Reg	Inside	CollgCr	1Fam	2Story	7	5
9600	Reg	FR2	Veenker	1Fam	1Story	6	8
11250	IR1	Inside	CollgCr	1Fam	2Story	7	5
9550	IR1	Corner	Crawfor	1Fam	2Story	7	5
14260	IR1	FR2	NoRidge	1Fam	2Story	8	5

Figure 4.1: Train Dataset

We use the training dataset taken from Kaggle.com which contains 1460 rows and 67 columns. The source for the dataset is as follows:

<https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data>

4.3 User Interface Design

A graphical UI or GUI empowers the customers to coordinate with the structure through graphical images and visual pointers, rather than substance based interfaces, created request stamps or substance course. The customer cooperates with the system with a front end organized using a Python scaled down scale web structure called Flask.

Python Flask relies upon Werkzeug tool stash and Jinja2 format motor. It is known as a miniaturized scale system since it doesn't require explicit mechanical assemblies or libraries. It has no database thought layer, structure endorsement or whatever different fragments where past outcast libraries give standard limits.

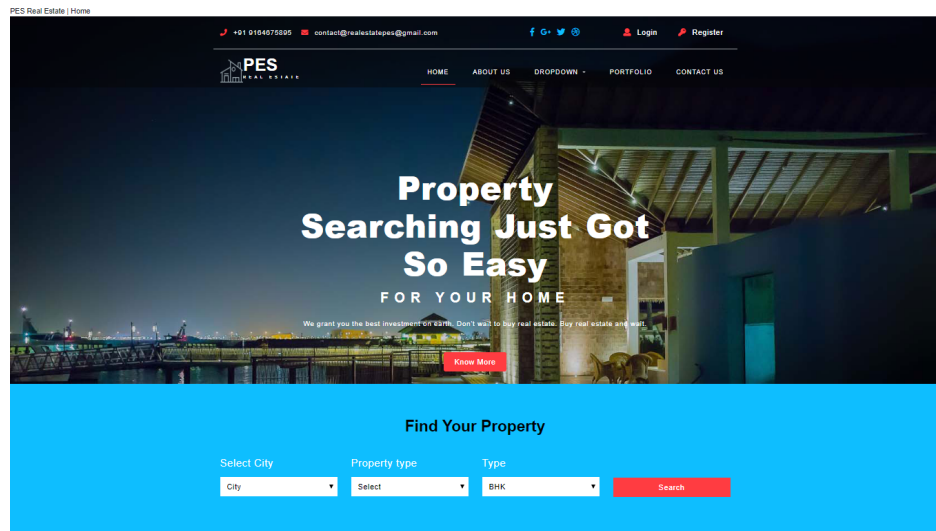


Figure 4.2: UI (Home)

4.4 Module 1: Data Preprocessing

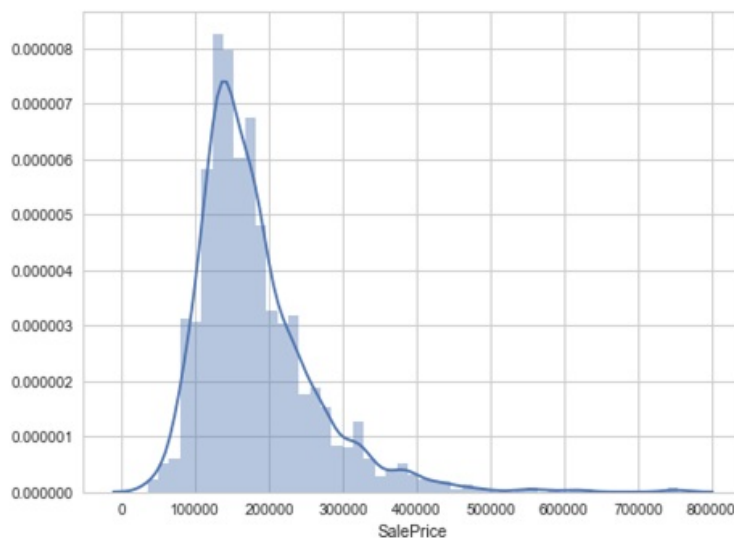


Figure 4.3: Graph-1 (Left Skewed)

The above figure shows a skewness in the nature of the data distribution. The figure suggests a right-skewed distribution and hence it is asymmetrical about the mean. Hence we propose a log-transformation technique to rectify this asymmetry. After applying the log-transform we get the data distribution "equally stretched" about the mean as shown below.

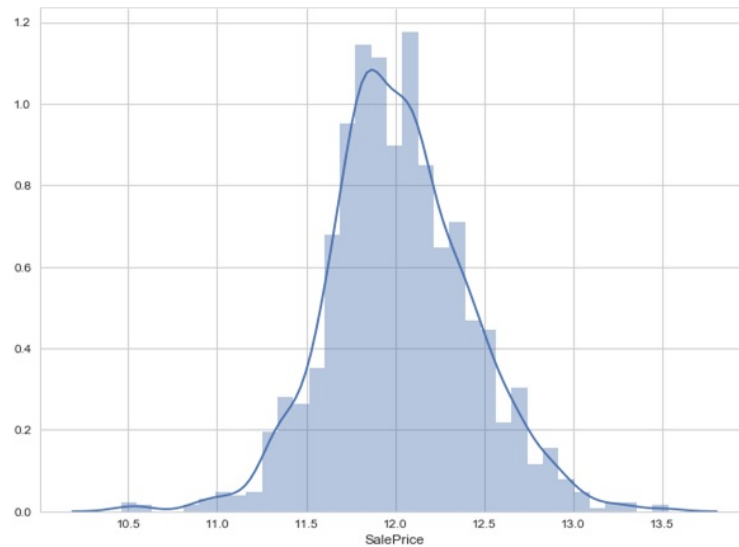


Figure 4.4: Graph-2 (Center Skewed)

4.5 Module 2: Prediction Modelling and Evaluation

- Structured Neural Networks :

The structured neural network will be implemented by 2 steps - Constructing a structured knowledge graph leading from input to output keeping only important connections. Then, constructing a neural network corresponding to the finished knowledge graph.

The knowledge graph will be constructed by the following steps:

- Initialize a completely associated DNN X with k layers and n_i neurons in each layer
- Train X utilizing the training dataset
- Test X utilizing the test dataset and record its precision
- Repeat the accompanying except if there's a critical reduction in accuracy.
 - * Identify the weak links in X , and remove them
 - * Identify the weak neurons in X , and remove them
 - * Identify the weak layers in X , and remove them
 - * Train the decreased X utilizing the training dataset

- * Test the decreased X utilizing the test dataset and record it's precision
- Restore X from it's previous iteration
- Create information chart Y from X by disregarding all the weights.
- return layered information chart Y

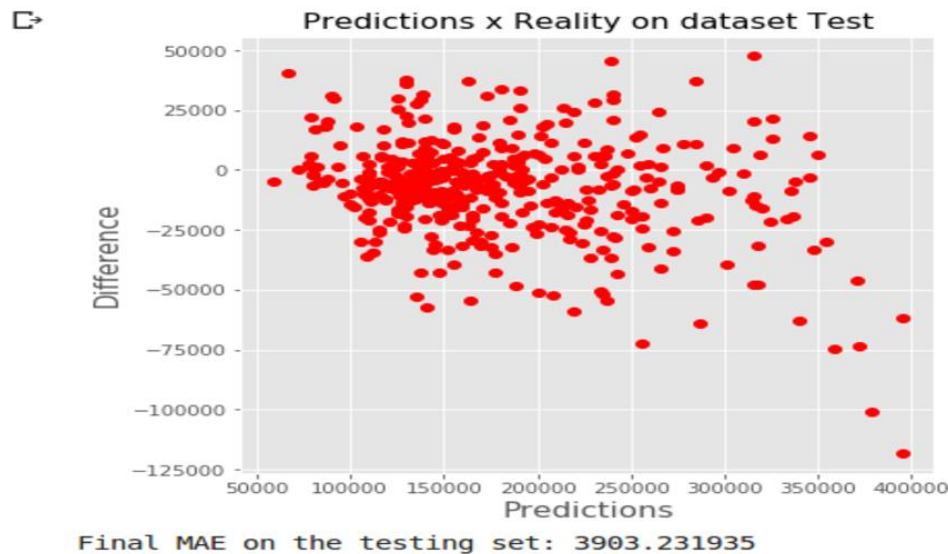
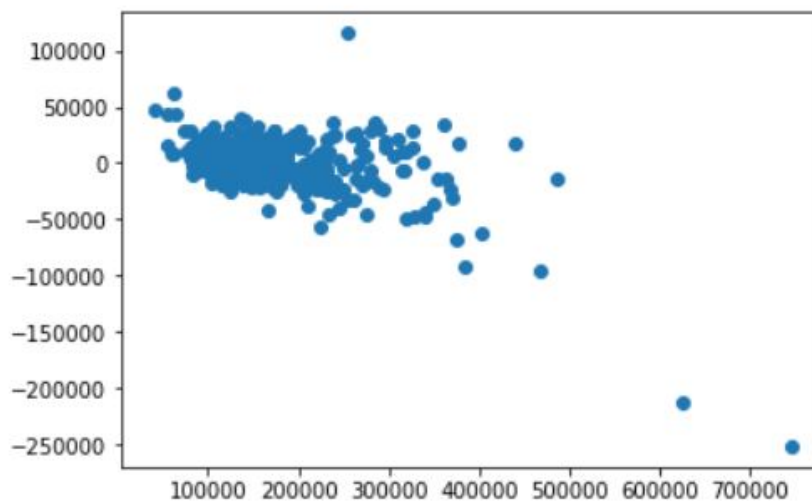


Figure 4.5: Structured Neural Network

- XGBoost :

XGBoost is another method which comes under non-metric classifier family which is based on the concept of Decision Tree, but there are significant differences between the two. XGBoost is an ensemble of decision trees wherein weighted combinations of predictors is taken. XGBoost works on the same lines of Random Forest, but there is a difference in working procedures. The similarities are that the features extracted in both the cases is completely random in nature.



Validation MAE for XGBoost Model : 15350.46401969178

Validation MAE for multi-pass XGBoost Model : 14161.433465325337

[123983.45 152999.73 178252.39 ... 168134.14 122767.21 230023.5]

Figure 4.6: XGBoost

- Recurrent Neural Networks(LSTM) :

Recurrent Neural Network has become one of the most preferred algorithm used extensively for sequential data. The main reason of its popularity is attributed to its ability to remember the input given to it. It consists of internal memory which makes it most suitable for any type of Machine Learning problem consisting of sequential data. It has been one of the very few algorithms which had been the stepping stone of success behind the enormous success of the concept of Deep Learning.

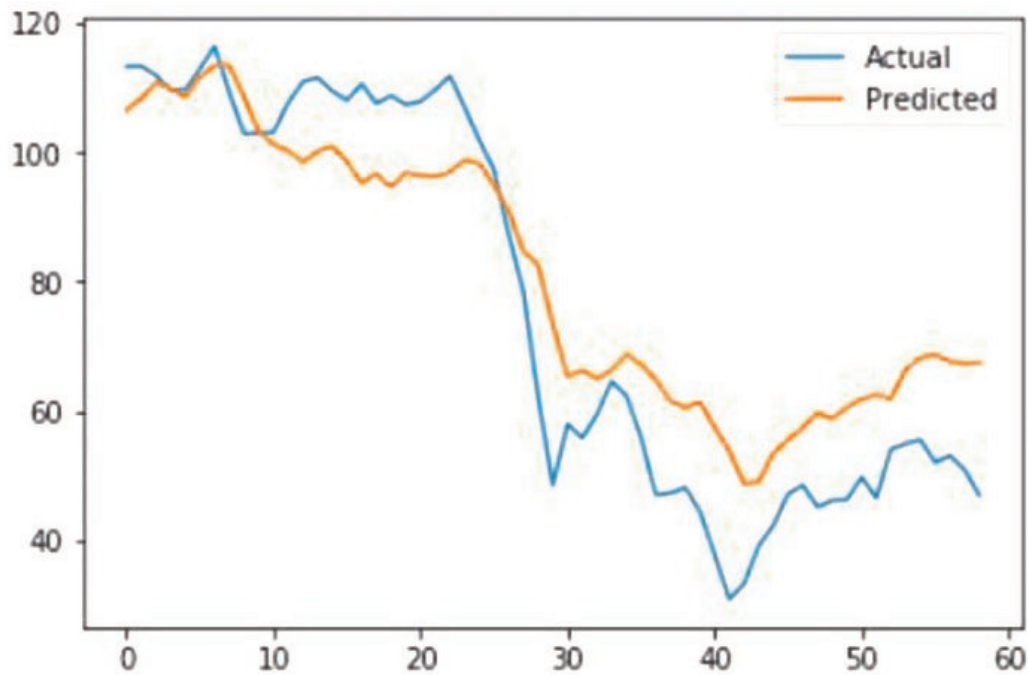


Figure 4.7: Recurrent Neural Network

4.6 Summary

The modules of the system are the basic units of functionality of a system. They interact with the user to complete the working of the system. The various modules that are implemented along with the expected inputs and outputs are mentioned in this chapter. Defining these parameters will help the user as well as the developer get a clear understanding of the project in addition to easing out the testing process.

Chapter 5

Implementation

the implementation chapter, the framework is manufactured utilizing different apparatuses and bundles either starting with no outside help or by creation. Given the engineering report from the structure stage and the necessity record from the examination stage, the framework ought to be worked by the determinations settled upon between the client and the customer.

Implementation of any product is constantly gone before by significant choices in regards to determination of the stage, the language utilized, and so forth. These choices are affected by a few factors, for example, the genuine condition where the framework works, the speed and precision required and other implementation explicit details.

5.1 Programming language selection

5.1.1 Overview of Python:

Python is an extensively used strange state, all around helpful, interpreted, dynamic programming language. Its structure hypothesis emphasizes code understandability, and its punctuation empowers programming architects to express thoughts in less lines of code than would be possible in languages, for instance, C++ or Java. This language gives assembles proposed to engage clear projects on both a little and significant scale. Python supports various programming perfect models, tallying object-arranged, fundamental and down to earth programming or procedural styles. It features a dynamic sort system and customized memory the officials and has a huge and extensive standard library. It is similarly a not too bad prototyping programming which is the motivation behind why we have picked this language for execution.

5.1.2 Overview of Flask framework:

A web templating framework joins a layout with a specific information source to render dynamic pages. Flask is a web application structure written in Python. It is utilized in associating the entire of the backend to the individual frontend in our task. Flask is frequently alluded to as a miniaturized scale system. It expects to keep the center of an application basic yet extensible.

5.1.3 Overview of Jupyter notebook using Anaconda:

We unequivocally endorse presenting Python and Jupyter using the Anaconda Distribution, which fuses Python, the Jupyter Notebook, and other normally used packs for coherent figuring and data science. The Jupyter Notebook application empowers us to make and change reports that demonstrate the data and yield of our Python substance.

5.2 Platform Selection

For our project, we have picked Windows platform for the execution. In straight forward terms, it is a working framework. It is the product on a PC that engages applications and the PC overseer to get to the gadgets on the PC to perform needed limits. Microsoft Windows has made much movement and changes which made it easy to use the working framework. Notwithstanding the way that it isn't the least requesting, it is less difficult than linux. Since there are progressively number of Microsoft customers there are more programming projects, redirections and utilities for windows. Almost all games are perfect to windows, some CPU genuine and practical heightened games are moreover maintained. While using a standard site design program having windows encouraging makes it bundle logically less complex. If you mean to make windows based applications then windows platform is most suggested as linux does not support windows applications.

5.3 Steps in Implementation

The project involves the comparison of two major neural network techniques namely the Recurrent Neural Network and Structured Neural Network.

Recurrent Neural Network

- The main base of this algorithm is upon using the XgBoost model to understand the pattern.
- Using a specified number of weak learners (here decision trees) and using them in sequential order, the model "boosts" the performance by gradually decreasing the error.
- The decision trees work sequentially wherein the error committed by the first one is gets rectified by the second tree. Consequently, the error committed by the second one is rectified by the one which follows it.

- The RNN tries to predict the Housing Price Index to figure out the trend in the price in the near future. The RNN works well with time series data having a sequential nature.

Structured Neural Network

- The model is initially "unstructured" with some n nodes, k layers and associated random weights.
- The model is made to propagate in the forward direction with the random weights which have been assigned.
- The final output node commits error in prediction of the price, which needs to be rectified using backpropagation algorithm.
- However, while moving backwards the weights which are below a specified threshold are removed as they contribute significantly less towards the actual output.
- On a similar understanding, the value of the nodes which have a value lesser than a specified threshold are eliminated, reason being the same as mentioned in the previous step.
- Even the hidden layer with lesser contribution is removed.
- This entire process of removing the least significant nodes, weights, layers is continued for a certain number of iterations or epochs until a suitable accuracy is achieved in the prediction.

This makes the network more structured and helps to reduce the computation required in the entire process by a significant amount and in fact proves to be a better model as compared to the Recurrent Neural Network(See the MAE value for the models).

5.4 Summary

The implementation details of the entire system were discussed in this chapter. Also, the details of the programming languages, platform and their architectures are discussed. The reasons for choosing the above mentioned programming languages and respective details are also discussed.

Chapter 6

Testing

6.1 Software Testing

Software testing is the way toward practicing or assessing a framework or framework part by manual or automated intends to confirm that is fulfills determined necessities or to distinguish contrasts/among expected and actual results.

Software testing should be possible in two different ways-

- **Manual Testing:** Manual Testing is a procedure completed to discover the defects in software. In this procedure analyzer assumes a significant job as end client and confirm all highlights of the application to guarantee that the conduct of the application is functioning according to client necessities. It isn't important to know about any testing device for manual software testing. As the software testing crucial dependably says that 100 percent automation is no possible so the manual testing is very important.
- **Automation Testing:** Automation Testing refers to the way toward composing the test content (or) test code by the test engineer in the test tool(selenium, QPT and so forth). where the test tool will execute the test content on the software under test and produce the results.

There are eight essential standards of Testing:

- Define the normal yield or result.
- Don't test your very own projects.
- Inspect the results of each test totally.
- Include experiments for invalid or surprising conditions.
- Test the program to check whether it does what it should do well as what it should do.
- Avoid dispensable experiments except if the program itself expendable.
- Do not design tests accepting that no blunders will be found.

6.2 Testing Process

The three noteworthy strategies for testing are:

- **Black-Box Testing:** If one performs testing just on the practical piece of an application without having any auxiliary (code) Knowledge then that strategy for testing is known as Black-Box testing, as a rule the test engineers play out this testing.

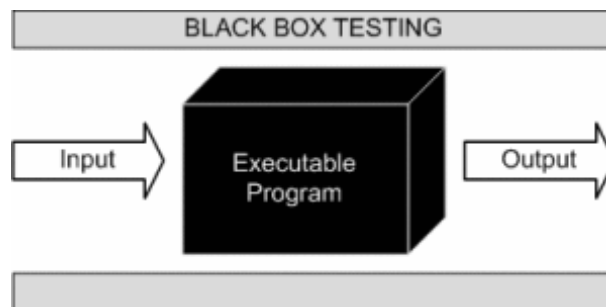


Figure 6.1: Black-Box Testing

- **White-Box Testing:** If one performs testing on the auxiliary (code) some portion of an application then that strategy for testing is known as White-Box testing, more often than not the developers or white box testers performs it

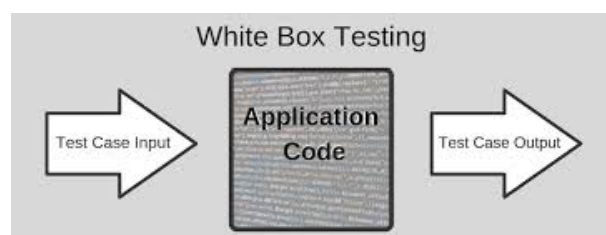


Figure 6.2: White-Box Testing

- **Gray-Box Testing:** If one performs testing on both the utilitarian part just as the auxiliary (code) some portion of an application then that technique for testing is known as gray box testing, typically the test engineers with code learning play out this testing. At the end of the day it is a mix of both black-box and white-box testing.



Figure 6.3: Gray-Box Testing

6.3 Levels of Testing

The testing process in this project involves the testing of all the modules. It covers the below mentioned levels of testing:

- Functional Testing
- Integration Testing
- System Testing
- Acceptance Testing
- Validation
- Compiling Test
- Output Test

6.3.1 Functional Testing:

It is additionally called as segment testing. Testing every single segment thoroughly against necessity particulars is known as functional testing. Functional testing will be tried with substantial information first. In the event that it works for substantial information, at that point test for invalid information. Here we do both positive and negative testing and we don't need to perform under testing and over testing. Functional testing falls into two classes:

- Positive Functional Testing: This testing convey practicing the application's capacities with legitimate info and furthermore confirming that the yields are right.
- Negative Functional Testing: This testing includes practicing application functionality utilizing a blend of invalid information sources, some surprising working conditions and by some startling working conditions and by some other "too far out" situations.

6.3.2 Integration Testing:

It is a dimension of testing where the designers will build up certain interfaces to coordinate the modules and test whether the interfaces are working fine or not. It is a white box testing generally engineers

or white box analyzers perform. Here we test dataflow between 2 modules.

6.3.3 System Testing:

System testing of programming or gear is testing coordinated on an aggregate, incorporated system to survey the system's consistence with its foreordained requirements. System testing falls inside the degree of disclosure testing, and everything considered, should require no data of the inward structure of the code or method of reasoning. If all else fails, system testing takes, as its info, most of the "co-ordinated" programming parts that have passed combination testing and furthermore the item system itself incorporated with any appropriate hardware system(s). System testing is start to finish testing where testing condition is like generation condition.

6.3.4 Acceptance Testing

Acceptance testing is the dimension of testing wherein one will play out a similar system testing within the sight of the client so as to influence him to acknowledge the application. It is a pre-conveyance testing in which entire structure is attempted at client's site on certifiable data to find mistakes.

6.3.5 Validation

The system has been tried and executed adequately and thusly ensured that all of the essentials as recorded in the item necessities detail are completely fulfilled. On the off chance that there ought to emerge an event of wrong data relating error messages are appeared.

6.3.6 Compilation Test

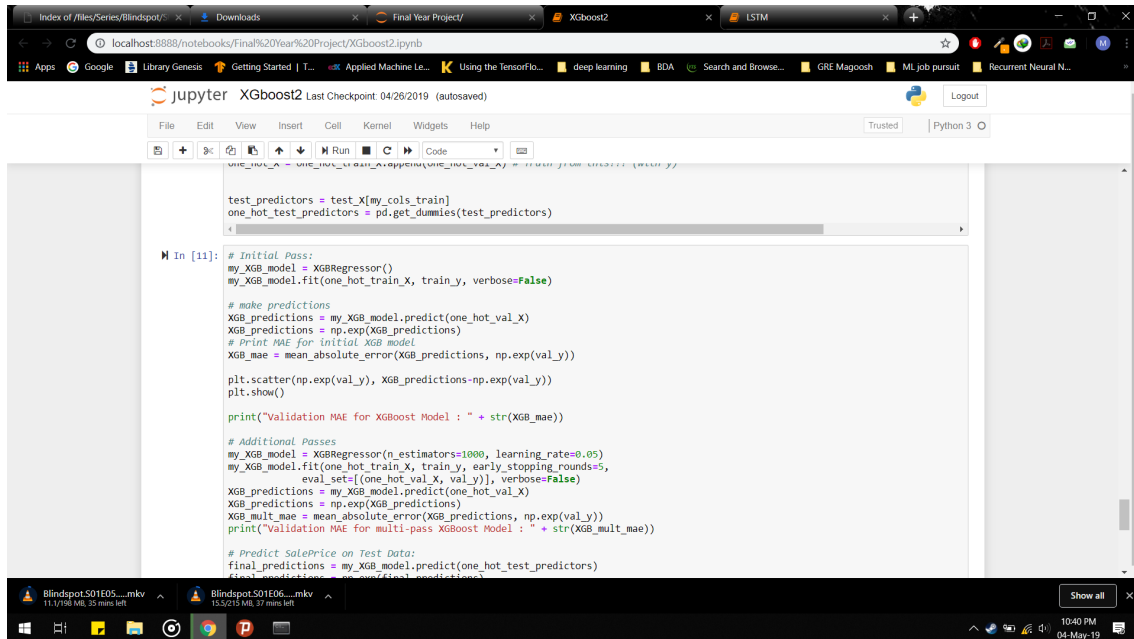
It was a brilliant idea to do our Stress testing right off the bat, since it allowed us a chance to fix some of startling halts and security issues that perhaps happened when parts were exhibited to especially high trade volumes.

This program was viably stacked and executed. By virtue of good programming there were no execution mistakes.

6.3.7 Output Test

The fruitful yield of calculations are put in the screen shots segment.

6.4 Screen Shots



```

test_predictors = test_X[my_cols_train]
one_hot_test_predictors = pd.get_dummies(test_predictors)

# Initial Pass:
my_XGB_model = XGBRegressor()
my_XGB_model.fit(one_hot_train_X, train_y, verbose=False)

# make predictions
XGB_predictions = my_XGB_model.predict(one_hot_val_X)
XGB_predictions = np.exp(XGB_predictions)
# Print MAE for initial XGB model
XGB_mae = mean_absolute_error(XGB_predictions, np.exp(val_y))

plt.scatter(np.exp(val_y), XGB_predictions - np.exp(val_y))
plt.show()

print("Validation MAE for XGBoost Model : " + str(XGB_mae))

# Additional Passes
my_XGB_model = XGBRegressor(n_estimators=1000, learning_rate=0.05)
my_XGB_model.fit(one_hot_train_X, train_y, early_stopping_rounds=5,
                  eval_set=[(one_hot_val_X, val_y)], verbose=False)
XGB_predictions = my_XGB_model.predict(one_hot_val_X)
XGB_predictions = np.exp(XGB_predictions)
XGB_mult_mae = mean_absolute_error(XGB_predictions, np.exp(val_y))
print("Validation MAE for multi-pass XGBoost Model : " + str(XGB_mult_mae))

# Predict SalePrice on Test Data:
final_predictions = my_XGB_model.predict(one_hot_test_predictors)

```

Figure 6.4: Input of XGBoost

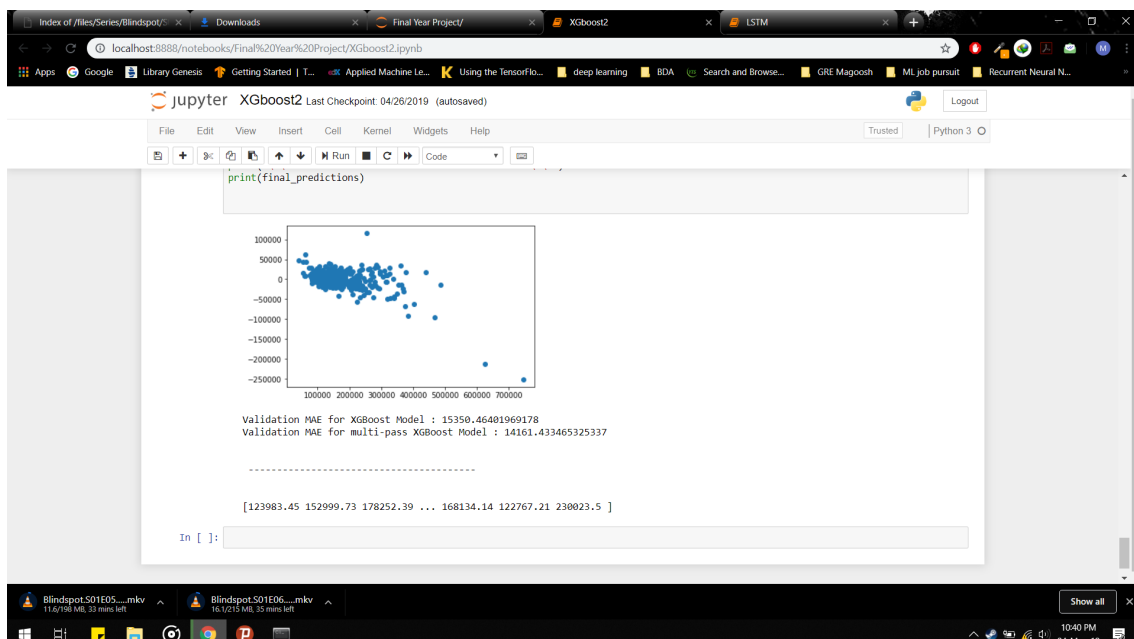


Figure 6.5: Output of XGBoost

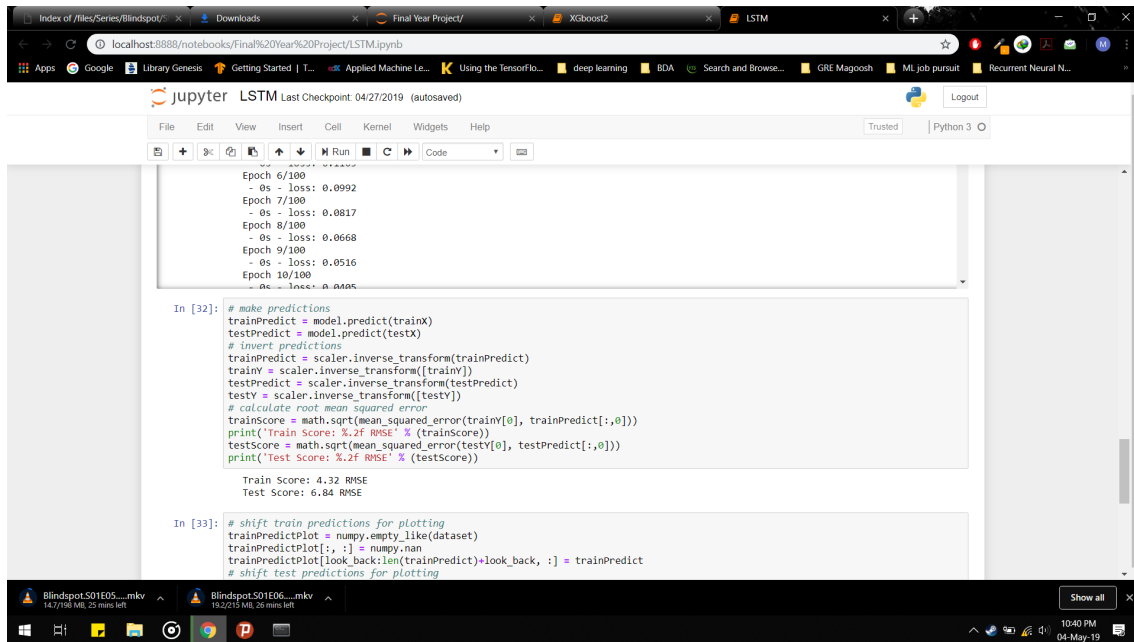


Figure 6.6: Input of RNN (LSTM)

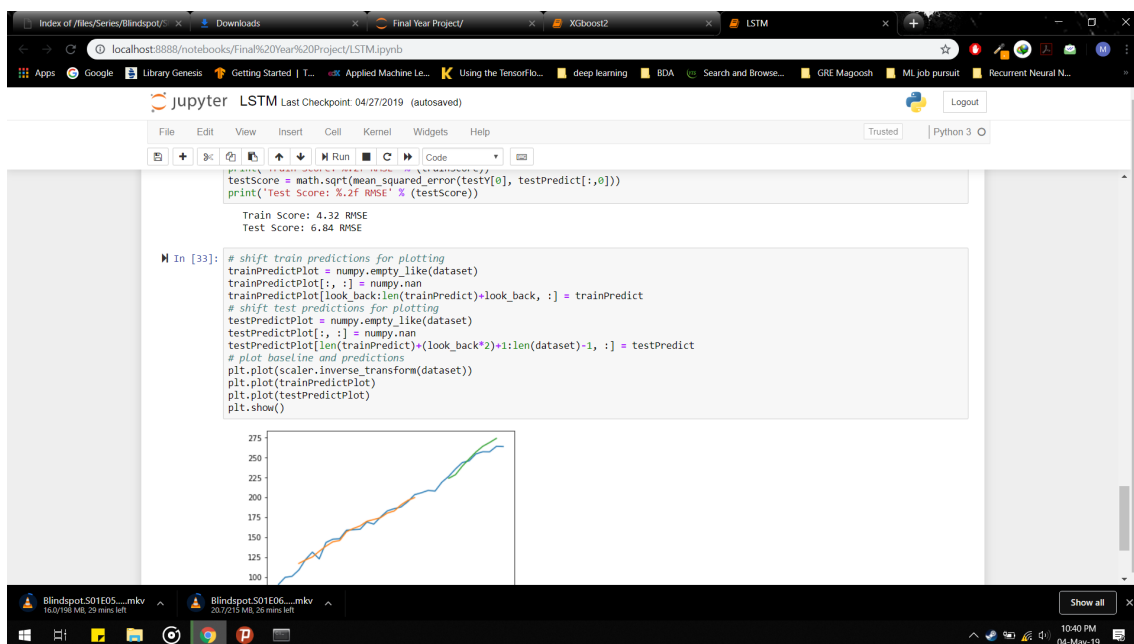


Figure 6.7: Output of RNN (LSTM)

```

[ ] feature_cols = [k: tf.constant(data_set[k].values) for k in FEATURES]
    labels = tf.constant(data_set[LABEL].values)
    return feature_cols, labels

if pred == True:
    feature_cols = [k: tf.constant(data_set[k].values) for k in FEATURES]
    return feature_cols

[ ] regressor.fit(input_fn=lambda: input_fn(training_set), steps=2000)

DNNRegressor(params={'head': <tensorflow.contrib.learn.python.learn.estimators.head._RegressionHead object at 0x7f03669ca1d0>, 'hidden_units': [200, 100, 50, 25, 12], 'feature_column

[ ] ev = regressor.evaluate(input_fn=lambda: input_fn(testing_set), steps=1)

[ ] loss_score1 = ev["loss"]
    print("Final loss on the testing set: {0:f}".format(loss_score1))
    loss=loss_score1*545676
    print(loss)

Final loss on the testing set: 0.002225
3438.8216777816415

[ ] y = regressor.predict(input_fn=lambda: input_fn(testing_set))
    predictions = list(itertools.islice(y, testing_set.shape[0]))

[ ] predictions = pd.DataFrame(prepro_y.inverse_transform(np.array(predictions).reshape(434,1)), columns = ['Prediction'])
    print(predictions)

```

Figure 6.8: Input of SNN



Figure 6.9: Output of SNN

6.5 Summary

In this chapter, we characterize software testing and give a concise depiction of black-box, white-box and gray-box testing. We specify the 3 levels of testing (Functional, integration system testing) and we also specify the validation, compilation test and output test of our project with the screen shots .

Chapter 7

Conclusion

7.1 Conclusion

After the project, we hereby prove and conclude that the accuracy of the structured neural network is better than the recurrent neural network and the it is also more efficient.

7.2 Limitations

- The size of dataset, may not be sufficiently large. The clarity of the results depends a lot upon the dataset, and only with sufficient data can we have good results
- The data needs to be available exactly in the required form. For that, we need a good amount of data cleaning. Both the models are very far from being versatile.

7.3 Summary

In this report we see the design and implementation of the house price prediction system which has 2 Modules i.e Data Preprocessing, Prediction modelling and evaluation. Finally, we learnt that both models have their flaws, but comparatively, the structured neural network is more accurate.

Bibliography

1. Haiping Xu and Amol Gade- Smart Real Estate Assessments
2. Reza Ghodsi, Abtin Boostani, Farshid Faghihi- Fuzzy Regression and Artificial Neural Network. 2010 Fourth Asia International Conference on Mathematics
3. Wan Teng Lim, Lipo Wang, Yaoli Wang, and Qing Chang- Housing Price Prediction Using Neural Networks. 2016 12th International Conference on Natural Computation
4. Junchi Bin, Shiyuan Tang, Yihao Liu, Gang Wang, Bryan Gardiner, Zheng Liu- Regression Model for Appraisal of Real Estate Using Recurrent Neural Network and Boosting Tree. 2017 2nd IEEE International Conference on Computational Intelligence and Applications
5. TOM KAUKO, PIETER HOOIMEIJER, JACCO HAKFOORT- Capturing Housing Market Segmentation: An Alternative Approach based on Neural Network Modelling. TOM KAUKO, PIETER HOOIMEIJER, JACCO HAKFOORT (2002)
6. Elsevier- Using machine learning algorithms for housing price prediction: The case of Fairfax County, Virginia housing data.

TEAM INFORMATION

<p>1. NAME: Mohammed Yaseen</p> <p>USN: 1PE15CS086</p> <p>CONTACT NUMBER:+91 8792306643</p> <p>EMAIL ID: 47.yaseen@gmail.com</p> <p>ADDRESS: PESIT BSC Boys Hostel, Konappana Agrahara, Hosur road, Bengaluru-560100</p>	<p>2. NAME: Shubharthi Dey</p> <p>USN:1PE15CS157</p> <p>CONTACT NUMBER: +91 7022768067</p> <p>EMAIL ID: deysubharthi15@gmail.com</p> <p>ADDRESS: PESIT BSC Boys Hostel, Konappana Agrahara, Hosur road, Bengaluru-560100</p>
<p>3. NAME: Saleha Afsari</p> <p>USN: 1PE16CS421</p> <p>CONTACT NUMBER: +91 8884956437</p> <p>EMAIL ID: saleha.afsari@gmail.com</p> <p>ADDRESS:#17/23 11th Main, 3rd Cross, Near Oxford college of Engineering, Bommanahalli, Bengaluru -560068</p>	

