House Sale Price Prediction **With GUI**

Manoj(17001003030)
karan(17001003025)

Mamta kathooria mam
(Mentor)

Agenda

-> Introduction

-> Acknowledgement

-> About Dataset

-> Libraries Used

-> Gradient Boosting

-> Random Forest

-> Decision Tree

-> Algorithm Comparisons

-> GUI With Tkinter

Introduction

- Our goal for this project was to use regression and classification techniques in order to estimate the sale price of a house in King County, Washington given the feature and pricing data for around 21,000 houses sold within one year.

Acknowledgement

We would like to thank our mentor(Mamta Kathooria mam) for her advice and inputs on the project.It could only be possible for us to make this project with the help of our mentor.

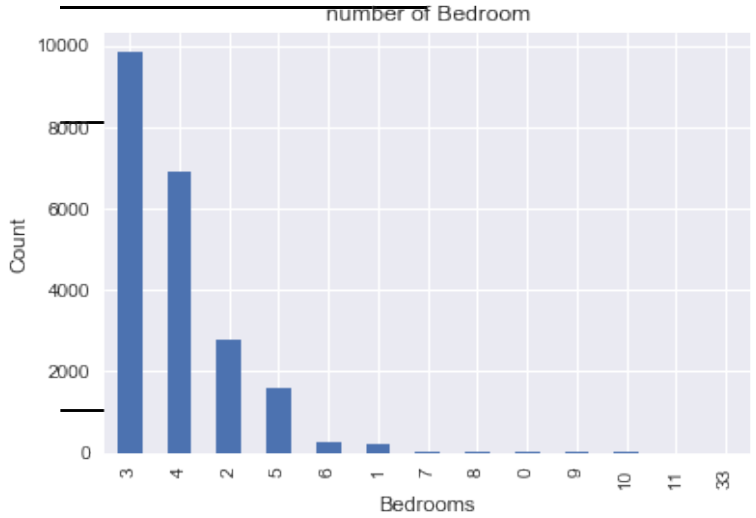
About Dataset

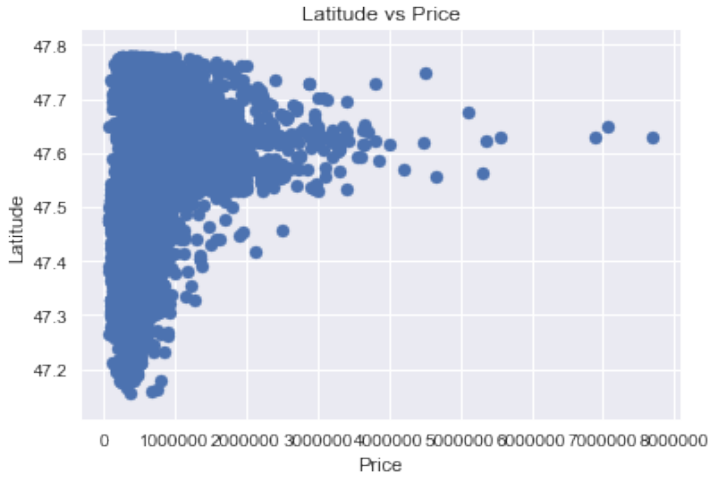
- Our dataset comes from a Kaggle competition.
- Our dataset contains house sale prices and its features for homes sold in King County, Washington between May 2014 and May 2015.
- King County is the most populous county in Washington and is included in the Seattle-Tacoma-Bellevue metropolitan statistical area. The county is considered the 13th most populous county in the United States.
- There are 21,000+ observations in the dataset.
- There are 21 total attributes in the dataset, five of which we derived from current columns. We are planning to use 16 attributes in our models: all attributes except for id,date, yr_renovated,yr_built,zipcode.

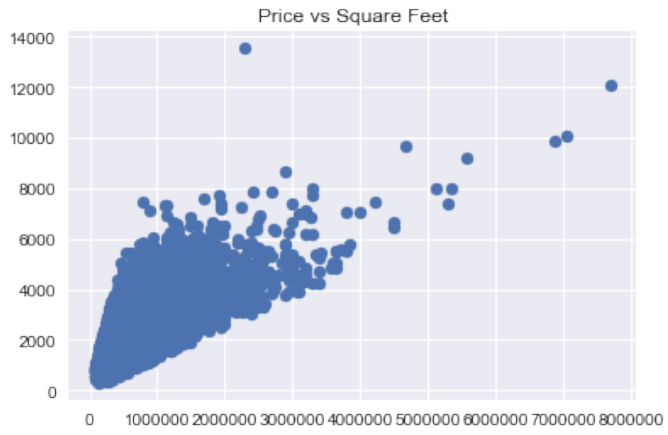
Dataset-Preprocessing

- We chose not to use zipcode because the attributes latitude and longitude, contains the same and more information and easier to work within our models.
- We checked for missing variables and the dataset didn't contain any.
- We performed feature selection by looking at the correlation percentage of each attritube with price.
- Pipeline is used to perform imputer and standardization simultaneously.
- Train and test data are split on the basis of waterfront variable because of uneven number of zeros and ones number of waterfront in dataset.

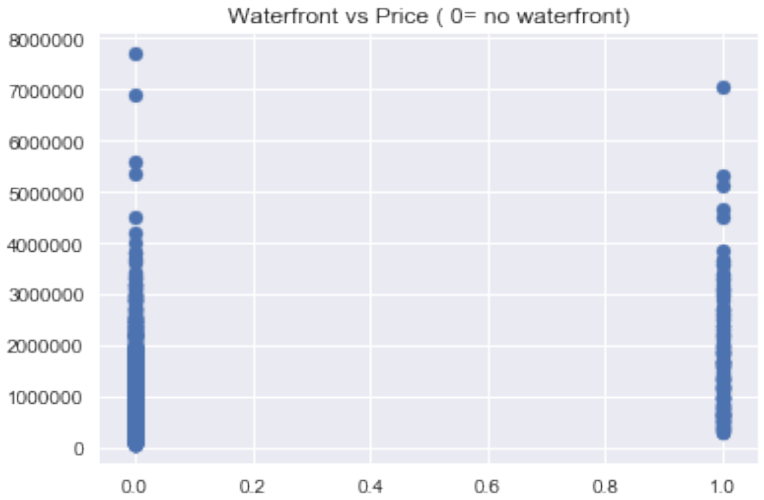
Co-relation b/w attribute and price

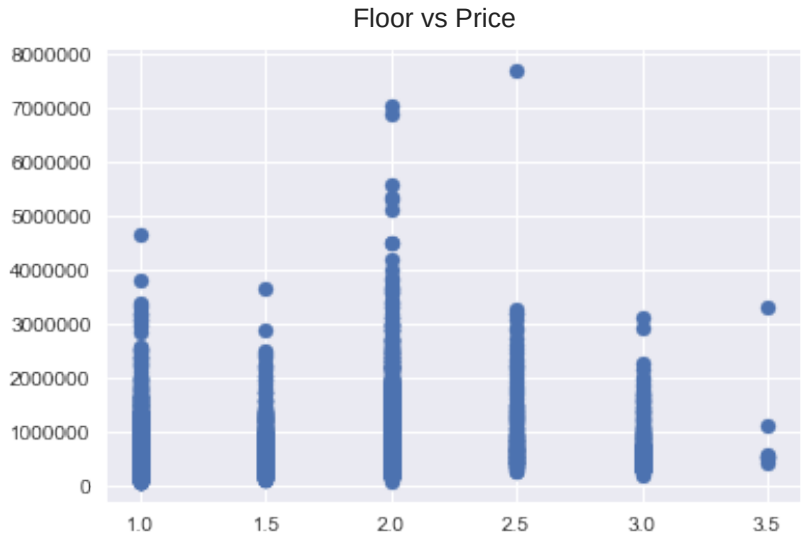


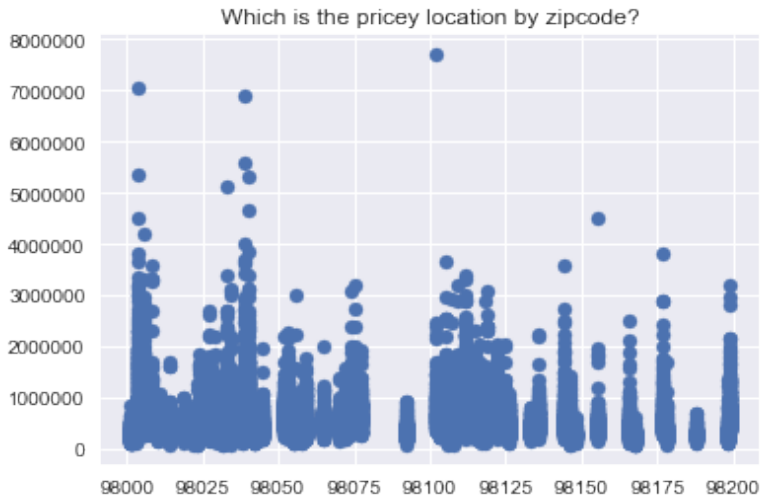




Co-relation b/w attribute and price







Libraries Used

-> Numpy : To perform basic calculation.

->Pandas : To collect dataset and performing changes on dataset.

->Sklearn : It is the most important library.All the major work is done under this library like splitting the dataset,training and prediction.

->Tkinter : It is used to give the model a graphical interface to the user.

Gradient Boosting

Gradient Boosting Regressor:

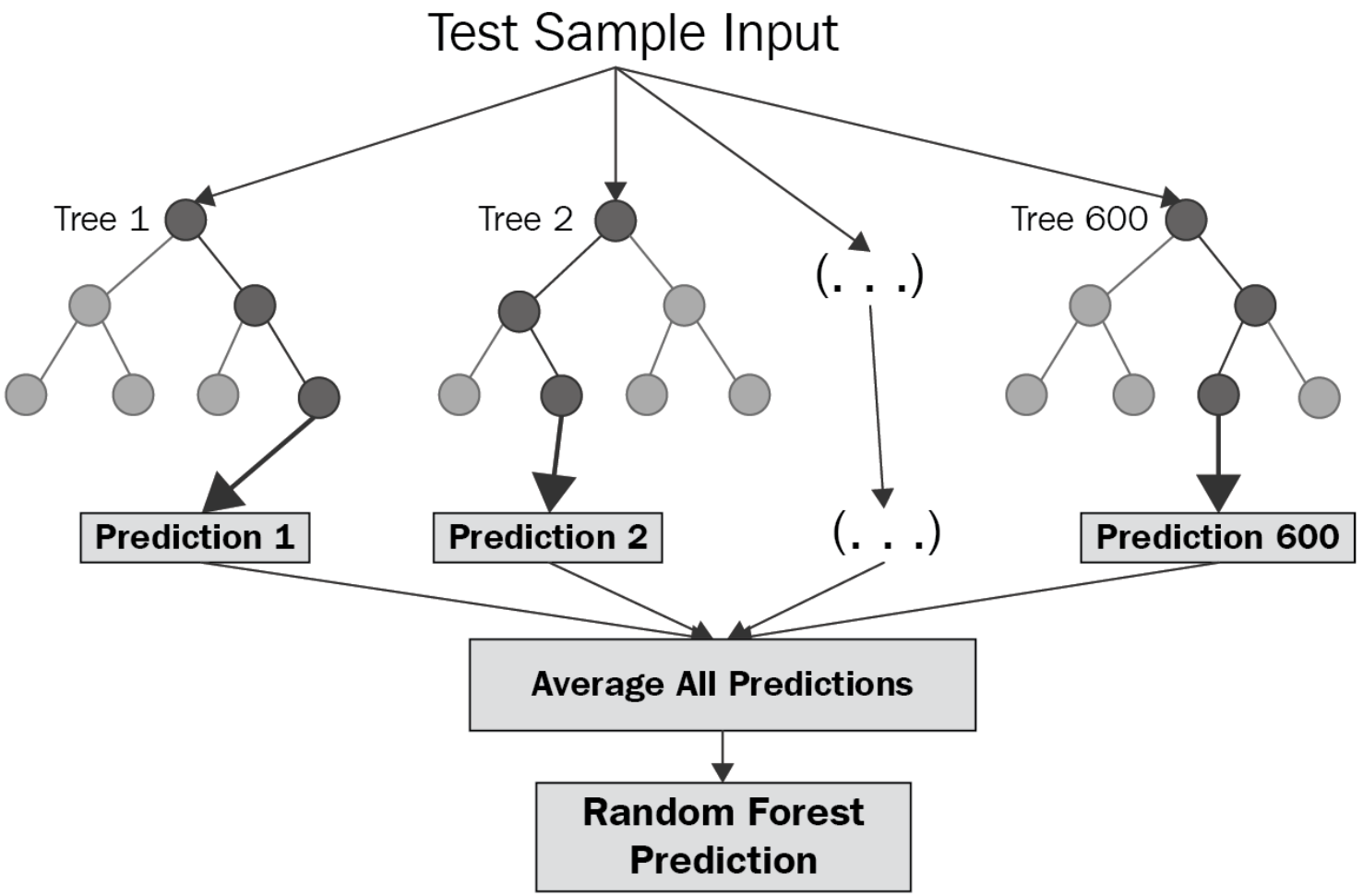
"Boosting" in machine learning is a way of combining multiple simplemodels into a single composite model. This is also why boosting is known as an additive model, since simple models (also known as weak learners) are added one at a time, while keeping existing trees in the model unchanged. As we combine more and more simple models, the complete final model becomes a stronger predictor. The term "gradient" in "gradient boosting" comes from the fact that the algorithm uses gradient descent to minimize the loss.

Random Forest

Random Forest Regressor :

Random forest is a bagging technique and not a boosting technique. The trees in random forests are run in parallel. There is no interaction between these trees while building the trees.It operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.
A random forest is a meta-estimator (i.e. it combines the result of multiple predictions) which aggregates many decision trees.

Random Forest

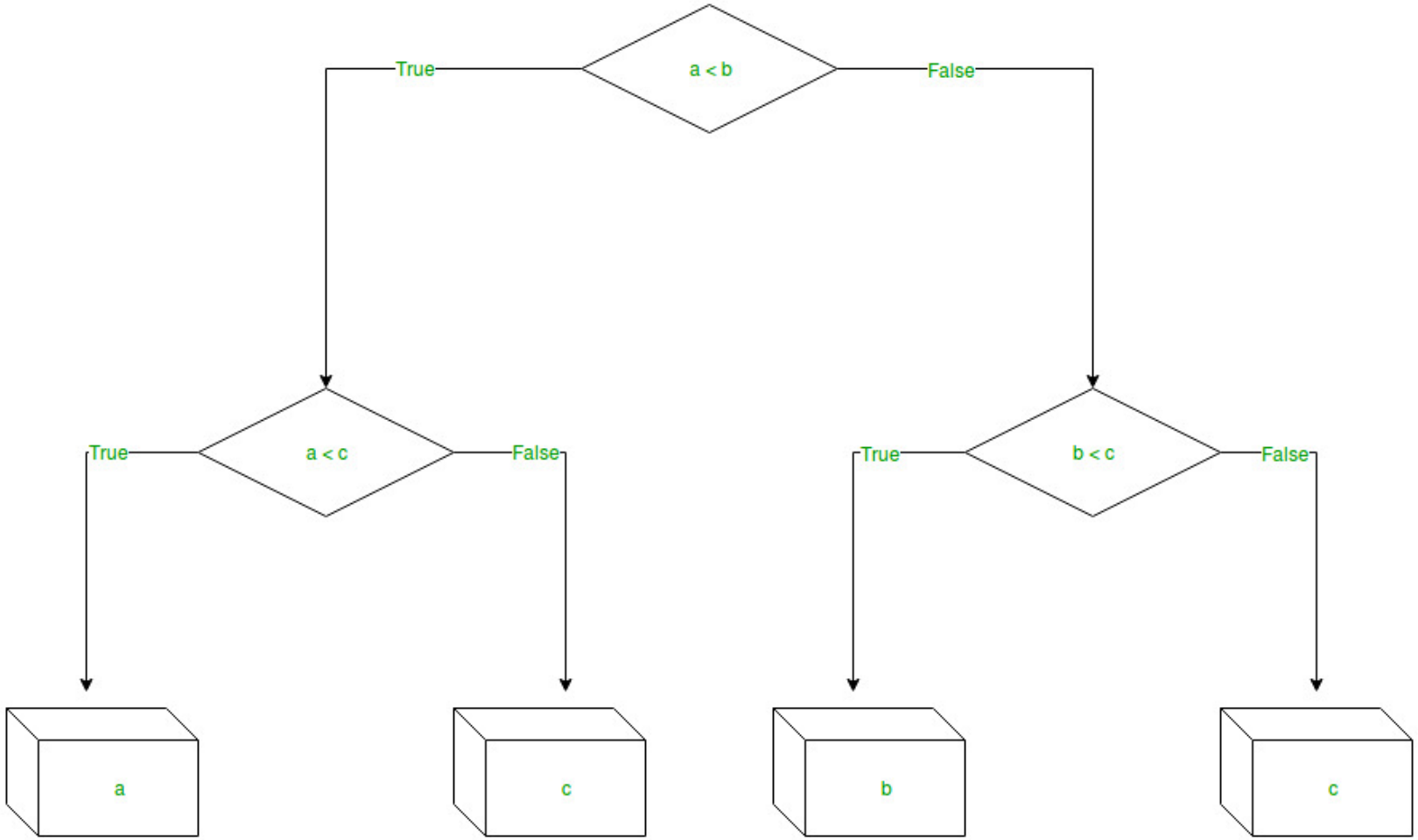


Decision Tree

Decision Tree Regressor :

Decision Tree is a decision-making tool that uses a flowchart-like tree structure or is a model of decisions and all of their possibleresults, including outcomes, input costs and utility.
Decision-tree algorithm falls under the category of supervised learning algorithms.It works for both continuous as well as categorical output variables.
Decision tree regression observes features of an object and trains a model in the structure of a tree to predict data in the future to produce meaningful continuous output.Continuous output means that the output/result is not discrete, i.e., it is not represented just by a discrete,known set of numbers or values.

Decision Tree



Tkinter

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications.Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

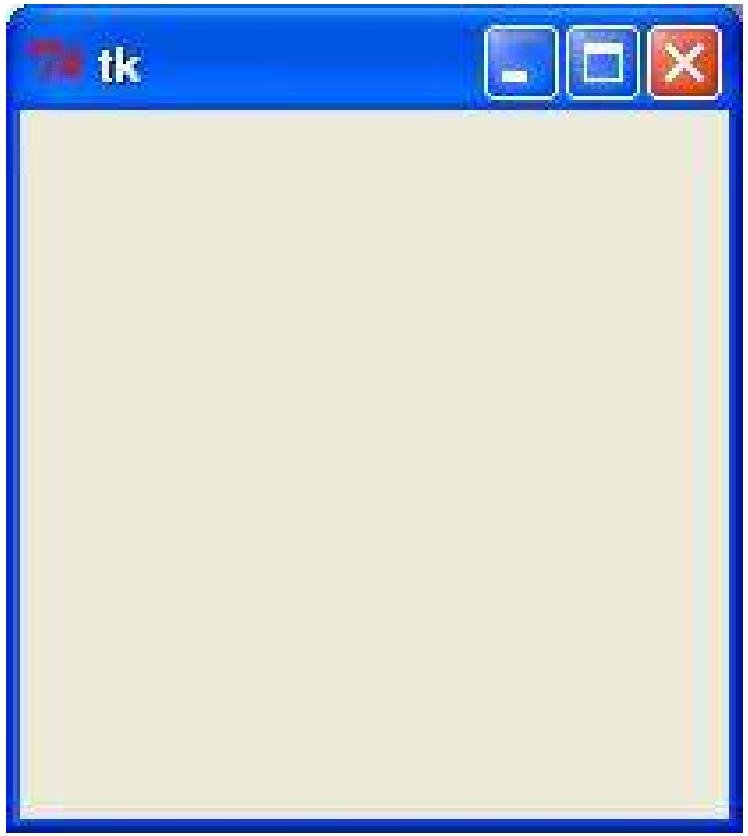
Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps –

->Import the Tkinter module.
->Create the GUI application main window.
->Add one or more of the widgets to the GUI application.
->Enter the main event loop to take action against each event triggered by the user.

Tkinter

Basic Tkinter GUI

```
import Tkinter
top = Tkinter.Tk()
# Code to add widgets will go here...
top.mainloop()
```



Algorithm Comparisons

Regression Algorithm

-> Gradient Boosting:

Accuracy : Approx 89.4%

-> Decision Tree

Accuracy : Approx 76.2%

-> Random forest

Accuracy : Approx 88.4%

Algorithm Comparison

Advantage of Random forest over decision tree

Random Forest aggregates the individual predictions to combine into a final prediction, based on a majority voting on the individual predictions.
It can be shown that an ensemble of independent classifiers, each with an error rate e, when combined significantly reduces the error rate.

Suppose we have 10 independent classifiers, each with error rate of 0.3,$\epsilon=0.3$

In this setting, the error rate of the ensemble can be computed as below (we are taking a majority vote on the predictions)
$\epsilon ensemble=\sum i=10i=6(10i)\epsilon i(1-\epsilon)10-i\approx 0.05$
It can be seen that with the theoretical guarantees stated above an ensemble model performs significantly well over a single decision tree.

Thank you