INTRODUCTION

The dataset for this project originates from the UCI Machine Learning Repository. The Boston housing data was collected in 1978 and each of the 506 entries represent aggregated data about 14 features for homes from various suburbs in Boston, Massachusetts. The objective of the Boston Housing Study was to extend previous study (examine the effect of air pollution on housing prices, controlling for the effects of other explanatory variables) with **Random Forest** and ***Gradient boosting*** modeling technique.

SUMMARY AND PROBLEM DEFINITION FOR MANAGEMENT

This project as a previous research focuses on ***evaluates the performance and predictive power of a model that has been trained and tested. Random Forest*** and ***Gradient boosting*** among most effective machine learning algorithms that relay on much smaller assumptions set, capable of fitting complex dataset, do not require feature scaling or centering. Both modeling method might produce better prediction results, with less data preparation efforts and less compute consumption.

MEASUREMENT AND STATISTICAL METHODS

The Python's *Pandas* and *Numpy* for data handling, and *Scikit* Learn for machine learning and model evaluation metrics. The housing data was presented to us as a CSV file and loaded into the program using Pandas. For the rendering graphs the graphviz have been download and installed. All regression modeling methods selected from previous research: ***linear regression, stochastic gradient descent, ridge regression, lasso regression, and elastic net***. Also, **Random Forests** and Gradient Boosting were used. For evaluate the scikit-learn a cross-validation and root mean-squared error (RMSE) as an index of prediction error were used.

Decision Tree also was used as alternative to versions of random forests and gradient boosting.

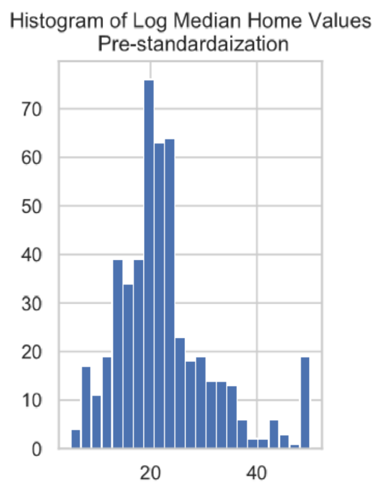
EXPLORATORY DATA ANALYSIS METHODS

506 entries represent aggregated data about 14 features for homes from various suburbs in Boston, Massachusetts. The basic statistics information for data set vectors are presented below.

	crim	zn	indus	chas	nox	rooms	age	dis	rad	tax	ptratio	lstat	mv
count	506	506	506	506	506	506	506	506	506	506	506	506	506
mean	3.613524	11.363636	11.136779	0.06917	0.554695	6.284634	68.574901	3.795043	9.549407	408.237154	18.455534	12.653063	22.528854
std	8.601545	23.322453	6.860353	0.253994	0.115878	0.702617	28.148861	2.10571	8.707259	168.537116	2.164946	7.141062	9.182176
min	0.00632	0	0.46	0	0.385	3.561	2.9	1.1296	1	187	12.6	1.73	5
25%	0.082045	0	5.19	0	0.449	5.8855	45.025	2.100175	4	279	17.4	6.95	17.025
50%	0.25651	0	9.69	0	0.538	6.2085	77.5	3.20745	5	330	19.05	11.36	21.2
75%	3.677082	12.5	18.1	0	0.624	6.6235	94.075	5.188425	24	666	20.2	16.955	25
max	88.9762	100	27.74	1	0.871	8.78	100	12.1265	24	711	22	37.97	50

The response variable was presented in two flavors :

- the median value of homes in thousands of dollars
- the log median value of homes in thousands of dollars



Data Visualization and Correlation

Data Visualization of features, correlation matrix and pair plot could be reviewed in the previous research document.

Overview of Programming Work

In order to compare Random Tree and Gradient bosting results with already used ***Market Linear, Ridge, Lasso, ElasticNet*** new regressors RandomForestRegressor , ExtraTreesRegressor, RradientBoostingRegressor have been added to the regressors array

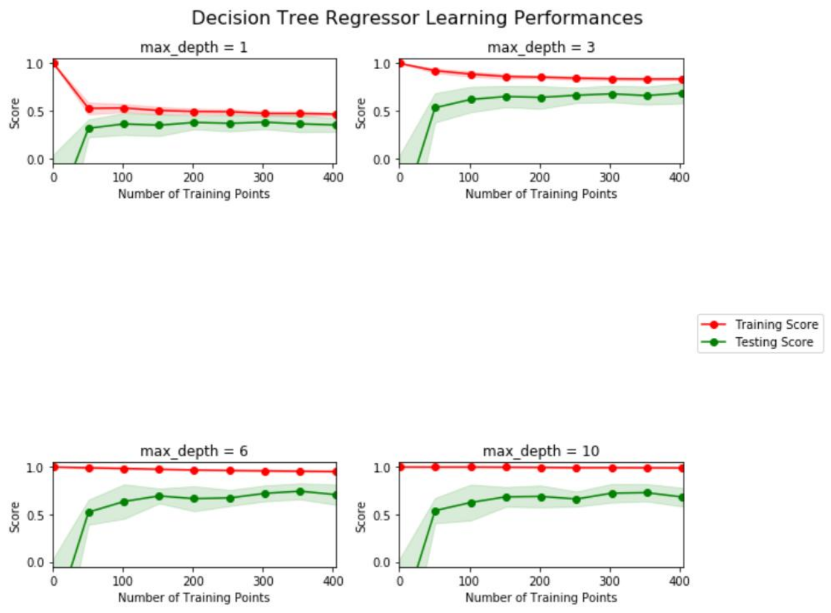
```
names = ['Linear_Regression', 'Ridge_Regression', 'Lasso_Regression',
          'ElasticNet_Regression',
         'RandomForest_Regression',
         'ExtraTrees_Regression',
         'GradientBoosting_Regressor']
```

There are also was an effort to estimate hyperparameter values. The most common way to do this is simply make a bunch of models with different settings, evaluate them all on the same validation set, and see which one does best or use automated methods to do this process in Skicit-learn.

Results and Recommendations

Learning hyperparameters evaluation

The following 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. 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 R2, the coefficient of determination.



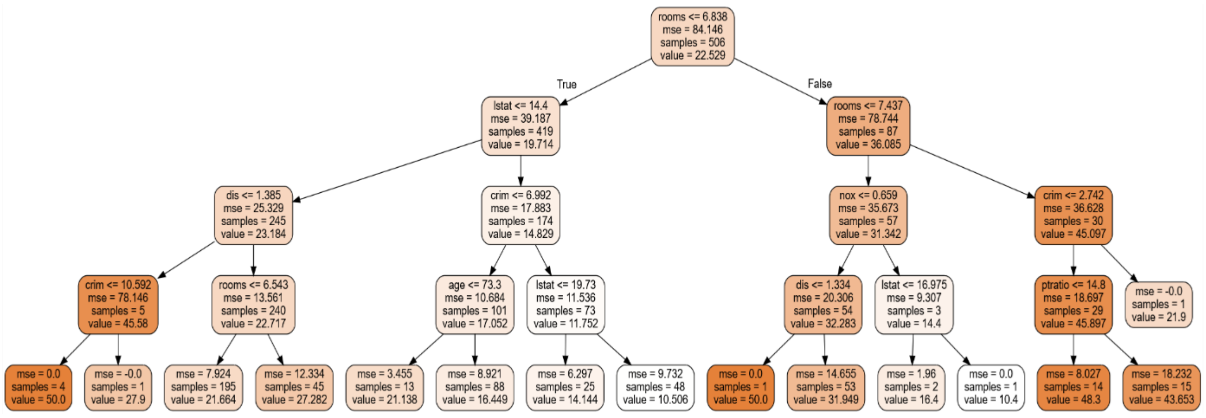
1st and 4th graphs represent biased model and the model suffered from variance respectively. On the first graph with maximum depth 1 training and testing errors converge and are quite high (~0.45). When more training points are added, the scores of the training and testing curves don't improve - no matter how much data we feed it, the model cannot represent the underlying relationship and therefore has systematic high errors.

Results

The results from the 10-fold cross-validation in standardized units for selected models depicted in the following grid

Method	Root mean-squared error
Linear_Regression	0.56194
Ridge_Regression	0.560511
Lasso_Regression	0.587381
ElasticNet_Regression	0.568084
randomForest_Regression	0.440416
ExtraTrees_Regression	0.422097
GradientBoosting_Regressor	0.230492

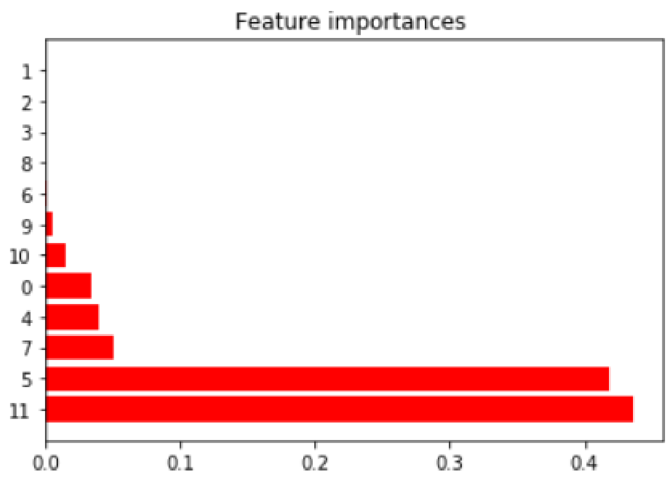
The comparison matrix shows that Gradient boosting with hyperparameter max depth =9 algorithm produces most accurate result. Having said this, this result might be an indication of overfitting. Based on the learning performance and complexity research, the recommendation is to reduce max depth to 3


rooms <= 6.838 mse = 84.146 samples = 506 value = 22.529
True
False
lstat <= 14.4 mse = 39.187 samples = 419 value = 19.714
rooms <= 7.437 mse = 78.744 samples = 87 value = 36.085
dis <= 1.385 mse = 25.329 samples = 245 value = 23.184
crim <= 6.992 mse = 17.883 samples = 174 value = 14.829
nox <= 0.659 mse = 35.673 samples = 57 value = 31.342
crim <= 2.742 mse = 36.628 samples = 30 value = 45.097
crim <= 10.592 mse = 78.146 samples = 5 value = 45.58
rooms <= 6.543 mse = 13.561 samples = 240 value = 22.717
age <= 73.3 mse = 10.684 samples = 101 value = 17.052
lstat <= 19.73 mse = 11.536 samples = 73 value = 11.752
dis <= 1.334 mse = 20.306 samples = 54 value = 32.283
lstat <= 16.975 mse = 9.307 samples = 3 value = 14.4
ptratio <= 14.8 mse = 18.697 samples = 29 value = 45.897
mse = -0.0 samples = 1 value = 21.9
mse = 0.0 samples = 4 value = 50.0
mse = -0.0 samples = 1 value = 27.9
mse = 7.924 samples = 195 value = 21.664
mse = 12.334 samples = 45 value = 27.282
mse = 3.455 samples = 13 value = 21.138
mse = 8.921 samples = 88 value = 16.449
mse = 6.297 samples = 25 value = 14.144
mse = 9.732 samples = 48 value = 10.506
mse = 0.0 samples = 1 value = 50.0
mse = 14.655 samples = 53 value = 31.949
mse = 1.96 samples = 2 value = 16.4
mse = 0.0 samples = 1 value = 10.4
mse = 8.027 samples = 14 value = 48.3
mse = 18.232 samples = 15 value = 43.653

Feature importance for the model

```
For the following columns array(['crim', 'zn', 'indus', 'chas', 'nox', 'rooms', 'age', 'dis', 'rad',
    'tax', 'ptratio', 'lstat', 'mv'], dtype=object)
```

the feature importance feature depicted in the following graph



Recalculated efficiency produces following results.

R^2= 0.78338829102524075

RMSE= 0.3620568426928081