Boston Housing Prices

The python programming language (+Sklearn) is used to conduct this analysis. The aim of this study is to fit a regression model that best explains the variation in medv (Neural-Network is on-going). The data consist of 506 observations and 14 non constant independent variables.
To see my results and my codes, please go to: https://github.com/DianeDeeDee/Python_Machine_Learning/tree/master/Housing/share

Analyze data

The data is loaded and summarized [data.describe()], a broad overview of the variables is in share/Data_describe.txt file. The data was quite clean and well organized, so I didn't wrote a code to process data (I used my terminal: python commands).
The correlations between the variables are possible sources of multicollinearity, and they affect variation in medv. So let's look at their distribution, and their correlation with medv but also with each other:

- Histograms for the data distributions are in the folder: share/DataDistributions/FeatureHisto
- The distribution of each feature versus house-price: share/FeatureVersusPrice
- Attributes correlation are at: share/FeatureCorrelation

It can be seen that positive and negative correlations are in red (rad,tax) and blue(lstat,medv), respectively.
Charles River area, greater number of rooms and the presence of the black community are factors which increase housing price.
The presence of lower class people, high crime rate, distance to the main employment centers and high teachers/student ratios decrease housing price. The last one is quite surprising...

Evaluate algorithms: CrossValidation.py

I used Cross validation (n_split = 10, scoring='neg_mean_squared_error', all original features) to evaluate estimator performance of 4 models:

Model	Average accuracy from CV =10	Error on average acc (+/-)
Linear Regression	-33.16	5.76
Support Vector Regression	-33.37	5.78
Random Forest	-25.13	5.01
Elastic Net	-33.83	5.82

From the above result, we can see that Random forest and Linear Regression have the least average error (& high average accuracy), indicating that both models have a good performance to evaluate Boston house-price. The numbers are in neg. because of scoring option.
Plots are at: share/ModelComp . From the figures (prediction vs prevision for each model) we can see that some models are more or less sensitive to predict a certain range of house-price. Ex: SVR model doesn't predict very well house-prices greater than ~35, but this model is very sensitive for price predictions in [15 , 35]. This is why, it's better to do the predictions with more than one model.

Linear regression algorithm to fit data: multiple_linear_regression.py

Splitting the dataset into the Training set and Test set: test_size = 0.2. The model is fitted to the training dataset. The predictions are made with 95% CL (p-value < 0.05)
OLS Regression Results are in: share/LR_results.txt We will drop the variables based on their AIC scores. We also check that Adj R^2 and R^2 values are in [0,1].
No further variable can be removed when we start sacrificing accuracy, e.g:
medv ~ 16.71 - 0.06 crim + 2.83 chas + 5.80 rm - 0.46 dis - 0.60 ptratio + 0.01 b - 0.48 lstat
#The first 5 predictions for medv: [31.34 25.51 31.85 30.23 29.52]
Residual: R^2 = 0.957, Adj. R-squared = 0.957

The graph showing fitted values with residuals is: share/True_verus_Predicted_Values.pdf
I should have done it for each feature...

The next step would be to fit data using the other 3 models. It would be when I finish neural network. Finally, to finish this work, I'd write about model Applicability.