**Machine Learning Project**

**Project Name: House Prices: Advanced Regression Techniques**

**Team members: Shih-Han Wang(sxw173330),Taishan Wang(txw171430), Shakti Suman(sxs177231), Wei Song(wxs170130)**

1. **Introduction and problem description**

In this project, we have to use the raw dataset to predict the house prices, and compare with the expected result we have, so we can know the accuracy of our models. However, the dataset has too many characters and number with vague and sometimes irrelevant meaning, so we need to preprocess the data first, then use it to build our models.

1. **Related work**

* Data preprocessing
* Build multiple models
* Check output accuracy

1. **Dataset description (including features, attributes, etc)**

There are two datasets Train.csv, and Test.csv.

Train.csv: the dataset is for training only.

Test.csv: the dataset is for testing only.

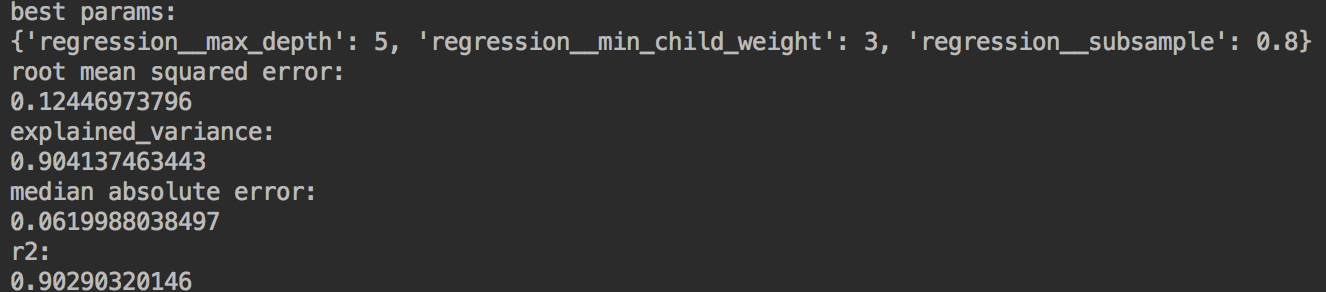
Each dataset has 1460 instances, and 80 features, but only Train dataset has sales price labels.

1. **Preprocessing techniques**
2. Used Log transforms the skewed label
3. Converted the MSSubClass type which is supposed to be category from int to str
4. Used one hot encoder to convert category values to numeric values.
5. Fill the missing value with the mean of corresponding column.
6. Normalize all numeric values expect those converted from one hot encode.
7. **Your proposed solution, and methods [theoretical and practical details]**

We build several regression models like Xgboost, Random Forest, Linear Regression, baggingRegrassor & Neural Network. By using GridSearch, we can tune the hyperparameters of these models on 10-fold cross validation and find the best parameters based on the “neg\_mean\_squared\_error” metric. Meanwhile, we evaluate the best model by using other metrics such as “explained\_variance”, “median\_absolute\_error”, “r2”.

1. **Experimental results and analysis**

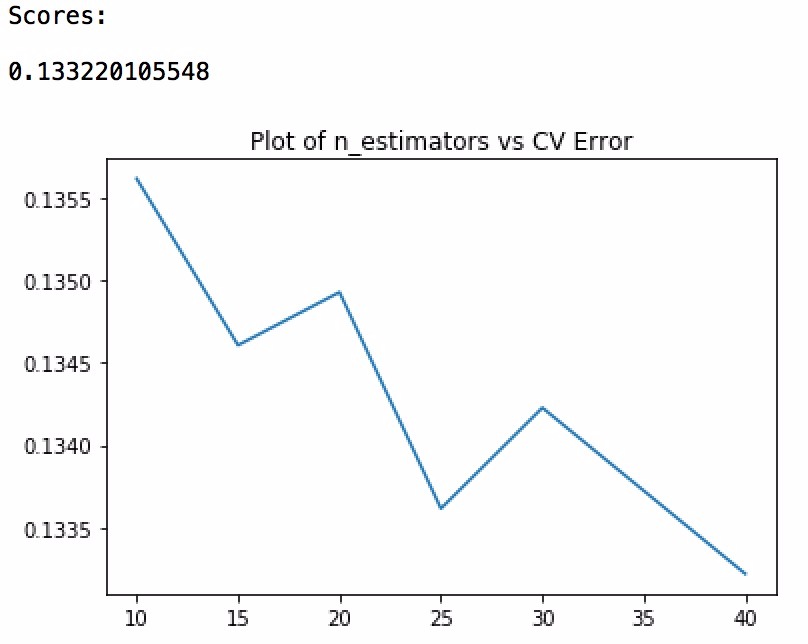
* XGBoost



the xgboost result is the best among these four models. We set max depth to 5, which used to control over-fitting. The higher depth will allow model to learn relations very specific to a particular sample. We set the regression\_subsample to 0.8, which is the fraction of observations to be randomly samples for each tree. Lower values make the algorithm more conservative and prevents overfitting but too small values might lead to under-fitting. We set regression\_min\_child\_weight to 1, which is the minimum sum of weights of all observations required in a child. The higher values prevent a model from learning relations which might be highly specific to the particular sample selected for a tree.

* Ridge

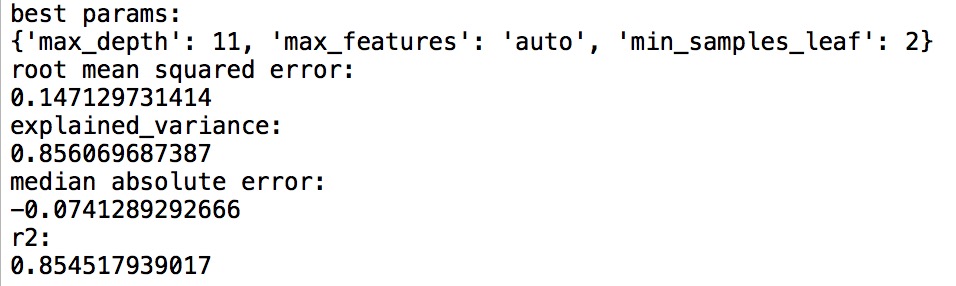
Ridge Regression model of BaggingRegressor in SKLearn gave the next best results with the house prices data.



Score: 0.133220105548.

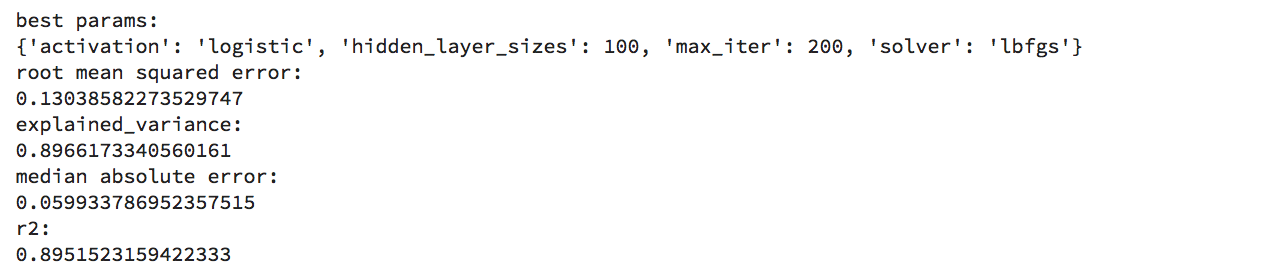
The params finalized were n\_estimators and base\_estimator with Ridge(15) along with Cross validation – 10.

* Random Forest



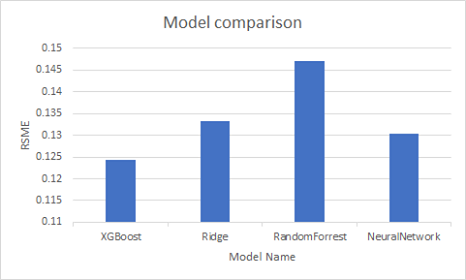
Random forest is not a good model for the house price case. We set up the 'max\_depth' from 10 to 16, however, the higher depth will decrease the accuracy and increase the error. Because if the tree become so complex, it is possibly lead to overfitting, so choosing the right depth is important. For ‘max\_features’, we set up to auto, it can automatically find the best split of the tree. We set the ‘min\_samples\_leaf’ to 2, so we can split the node when we have larger than two samples, the higher ‘min\_samples\_leaf’ made the tree to split much difficult and effect the accuracy to predict the right class.

* Neural Network



Neural Network got the result of squared error with 0.13038. We tuned the parameters by setting ‘hidden layer’ size from 30-100; using ‘activation functions’ identity, logistic, tanh, relu; ‘solver’ we chose lbfgs which is an optimizer in the family of quasi\_Newton methods, ‘max\_iter’ 200 which means to set iterations to 200.

1. **Conclusion**

****

The output result shows that predict error rate is 0.1244 with XGBoost working best.

1. **Contribution of team members**

* **Taishan Wang(txw171430):** Participated in the preprocessing part, normalized all numeric values expect those converted from one hot encode. Built the xgboost model and wrote the gridsearch to tune the parameters and wrote the metrics part to show different evaluation of the model.
* **Shih-Han Wang(sxw173330):** Optimized the preprocessing part, transferred the dataset to numerical only dataset. Built the random forest model and fined tune the parameters by Gridsearch. and evaluate the model with different error square.
* **Shakti Suman(sxs177231):** Wrote the preprocessing code for training data. Worked on optimization for Ridge (BaggingRegressor) model. Worked on final code optimization.
* **Wei Song(wxs170130):** Participated in the preprocessing part. Built the NeuralNetwork model and wrote the gridsearch to tune the parameters and wrote the metrics part to show different evaluation of the model.

1. **Reference**

Sklearn – Preprocessing, models and metrics modules

Matplotlib Graphs

kaggle