



Name of the Contest: House Prices: Advanced Regression Techniques

Team Name: Asrafiya

Contestants Name: Asrafiy Malek Choya

ID : 1640CSE00499

Kaggle Account: <https://www.kaggle.com/asrafia>

Git Account : <https://github.com/Asrafia>

Introduction

Project Goal& problem statement: The Goal is to predict the sale price for houses based on various features like building characteristics, size and location.

Data Preprocessin:

Handling outlier, missing and redundant samples:

The following data preprocessing process will consist of the following steps:

Data Exploration

Variable Identification

- DataFrame size and shape

- Label

- Features (Numeric, Categorical)

Univariate Analysis

- Label

- Numeric features

- Categorical features

Bivariate Analysis

- Numeric features

- Categorical features

- Correlation Matrix

Data Manipulation

- Label Manipulation

- Cutting Outliers

- Impute missing values

- Create new features

- Turn some numeric features into categorical features

- Correct Feature Skewness

- Create Dummy Variables

Data Exploration

In the first part of the analysis we will look at the data. We will investigate the size and shape of the test dataset and identify the variables. Then we will examine the distributions of the label and of each single feature. Afterwards we will explore the relationship between the features and the label and look at their correlation.

We import all necessary modules.

```
In [1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from scipy.stats import norm, skew
import warnings
warnings.filterwarnings('ignore')
```

We load the training data und test data

```
df_train = pd.read_csv("../input/train.csv")  
df_test = pd.read_csv("../input/test.csv")
```

Variable Identification

First, we investigate the training dataset. We look at the shape of the dataset, the name of the columns, the type of the columns and have a first look at the amount of missing data.

In [3]:

```
df_train.shape
```

Out[3]:

```
(1460, 81)
```

In [4]:

```
df_train.columns
```

Out[4]:

```
Index(['Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street',  
      'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig',  
      'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType',  
      'HouseStyle', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',  
      'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',  
      'MasVnrArea', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',  
      ],  
      dtype='object',  
      name='columns')
```



```
'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType',  
'HouseStyle', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',  
'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',  
'MasVnrArea', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',  
'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1',  
'BsmtFinType2', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',  
'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF',  
'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath',  
'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual',  
'TotRmsAbvGrd', 'Functional', 'Fireplaces', 'FireplaceQu', 'GarageType',  
'GarageYrBlt', 'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual',  
'GarageCond', 'PavedDrive', 'WoodDeckSF', 'OpenPorchSF',  
'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'PoolQC',  
'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleType',
```

```
C',  
    'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleTyp  
e',  
    'SaleCondition', 'SalePrice'],  
dtype='object')
```


Feature Engineering(Feature subset selection, creation, dimensionality reduction):

After investigating the SalePrice, we will examine the other features. Our features have two different data types: numeric and string features. We will split them into numeric and categorical features for further analysis. We will look at histograms for the numerical features and countplots for the categorical features.

In [10]:

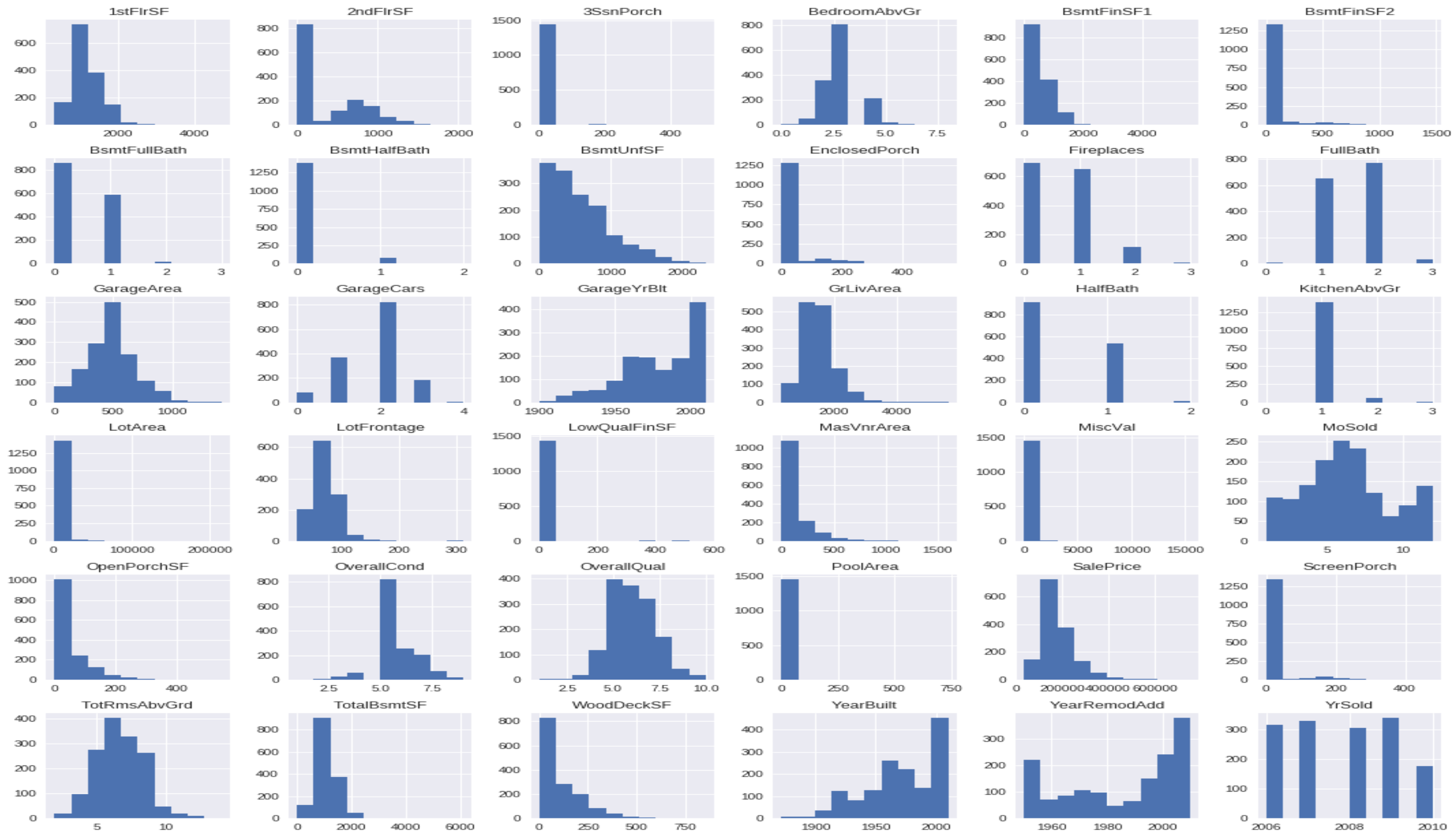
```
numeric_features = df_train.dtypes[df_train.dtypes != "object"].index  
numeric_features = numeric_features.drop("SalePrice")
```

In [11]:

```
categorical_features = df_train.dtypes[df_train.dtypes == "object"].index
```

In [12]:

```
f = pd.melt(df_train, value_vars=numeric_features)  
g = sns.FacetGrid(f, col="variable", col_wrap=3, sharex=False, sharey=False, size = 5)  
g = g.map(sns.distplot, "value")  
plt.show()
```



Data Science is the process of making some assumptions and hypothesis on the data, and testing them by performing some tasks. Initially we could make the following intuitive assumptions for each feature:

Houses with more rooms (higher 'RM' value) will worth more. Usually houses with more rooms are bigger and can fit more people, so it is reasonable that they cost more money. They are directly proportional variables.

Neighborhoods with more lower class workers (higher 'LSTAT' value) will worth less. If the percentage of lower working class people is higher, it is likely that they have low purchasing power and therefore, they houses will cost less. They are inversely proportional variables.

Neighborhoods with more students to teachers ratio (higher 'PTRATIO' value) will be worth less. If the percentage of students to teachers ratio people is higher, it is likely that in the neighborhood there are less schools, this could be because there is less tax income which could be because in that neighborhood people earn less money. If people earn less money it is likely that their houses are worth less. They are inversely proportional variables.

Scatter Plot Matrix

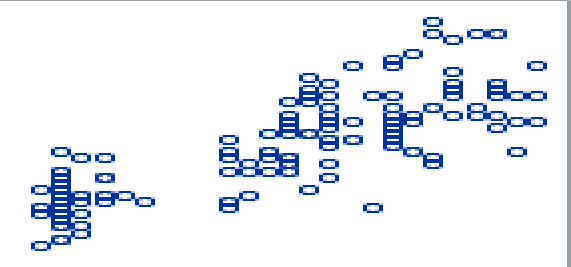
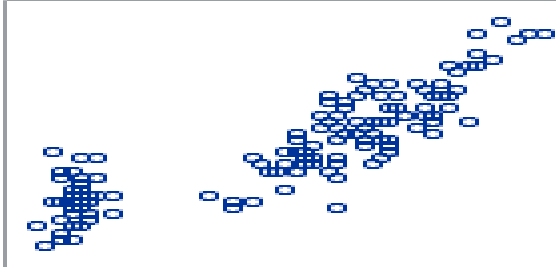
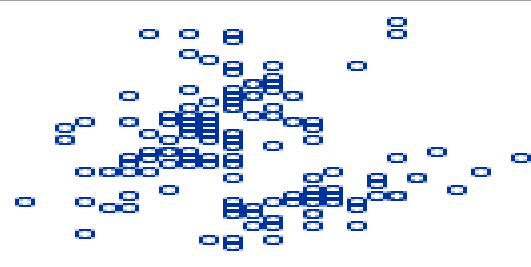
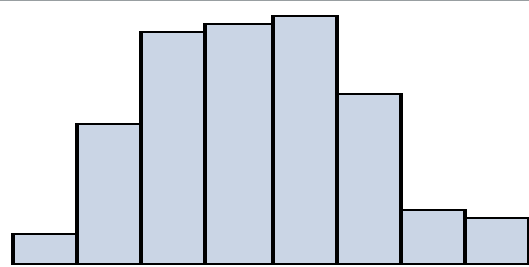
SepalLength

SepalWidth

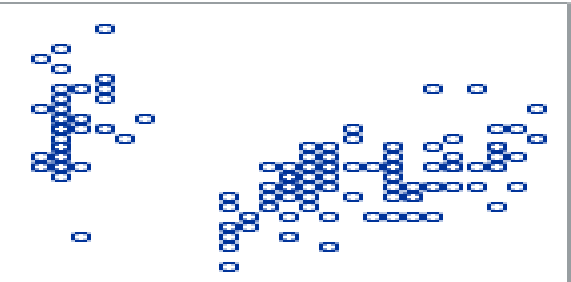
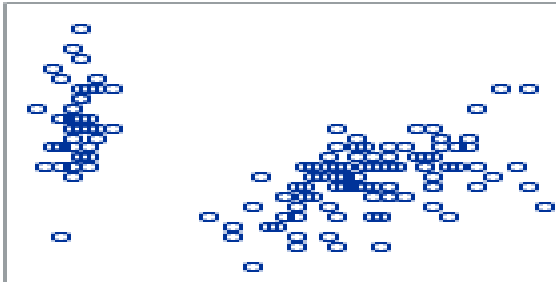
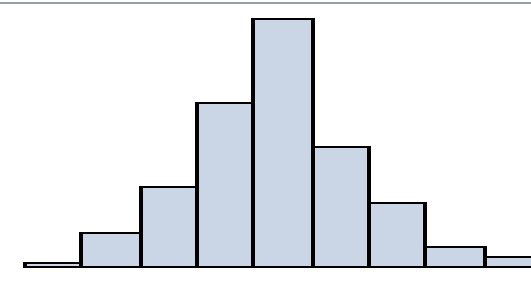
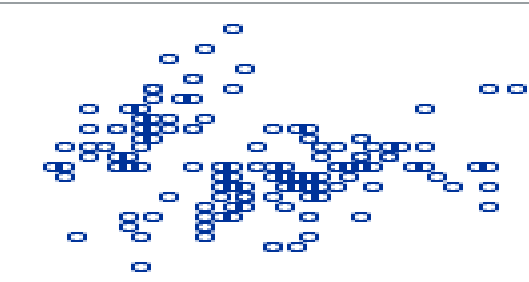
PetalLength

PetalWidth

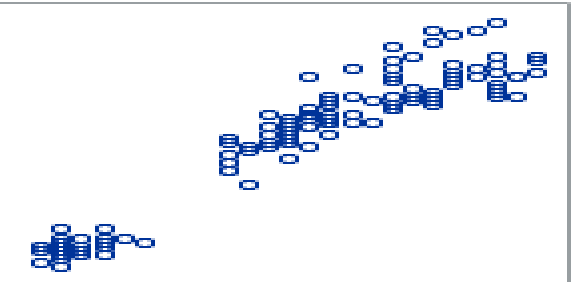
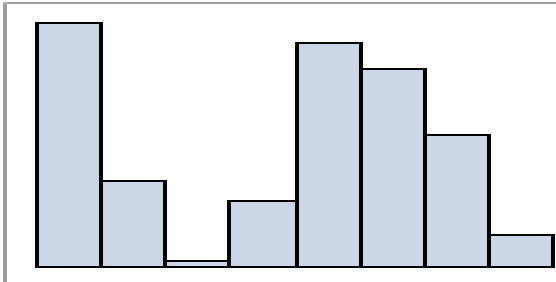
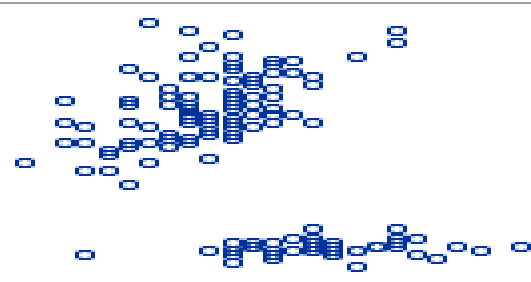
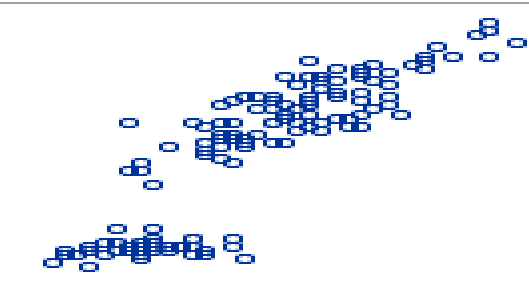
SepalLength



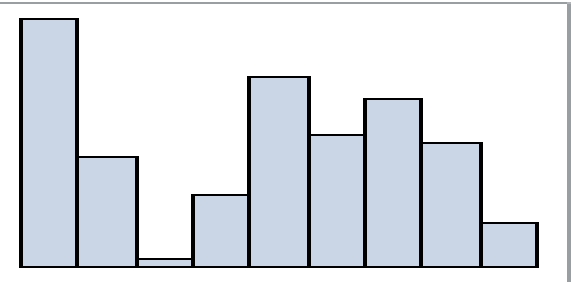
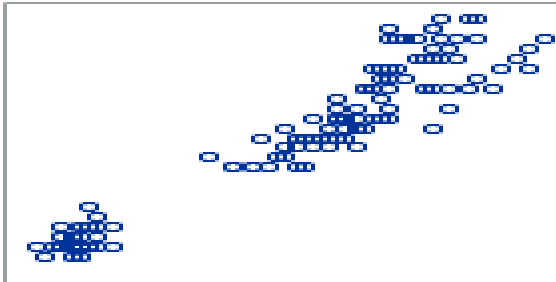
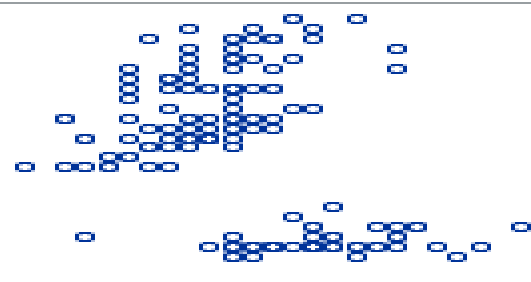
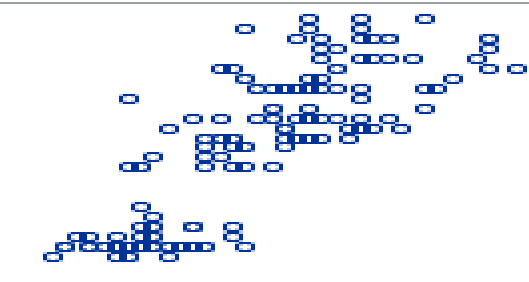
SepalWidth



PetalLength



PetalWidth



Modeling Methods (cross-validation, ensemble) :

Time to build some models! We began by creating some benchmarks using a Linear Regression model on both the scaled and non-scaled data. We then prepared a series of fits using three regularized linear regression models. The models we fit were:

- A naive Ridge Regression against the raw data

- A naive Lasso Regression against the raw data

- A naive ElasticNet Regression against the raw data

- A naive Ridge Regression against the scaled data

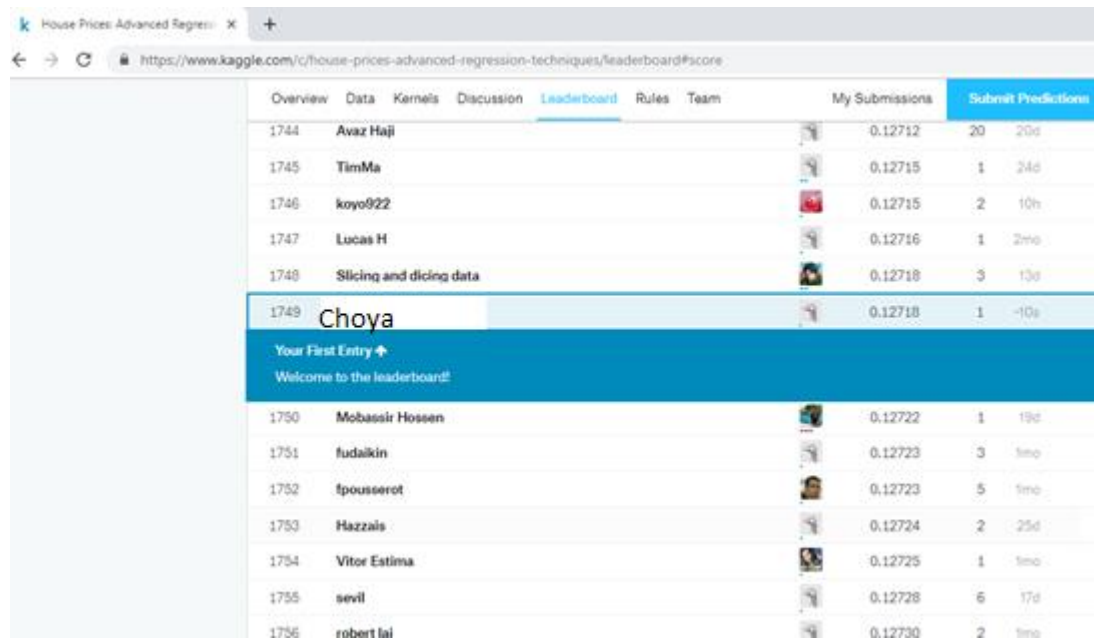
- A naive Lasso Regression against the scaled data

- A naive ElasticNet Regression against the scaled data

We then imported the Cross-Validation Models for each of the Regularized Linear Models.

Results & Discussion

Our performing models were



	Overview	Data	Kernels	Discussion	Leaderboard	Rules	Team	My Submissions	Submit Predictions
1744	Avaz Haji							0.12712	20 20d
1745	TimMa							0.12715	1 24d
1746	koyo922							0.12715	2 10h
1747	Lucas H							0.12716	1 2mo
1748	Slicing and dicing data							0.12718	3 13d
1749	Choya							0.12718	1 ~10s
Your First Entry Welcome to the leaderboard!									
1750	Mobassir Hossen							0.12722	1 19d
1751	fudaikin							0.12723	3 5mo
1752	fpousseerot							0.12723	5 5mo
1753	Hazzais							0.12724	2 25d
1754	Vitor Estima							0.12725	1 5mo
1755	sevil							0.12728	6 17d
1756	robert lai							0.12730	2 5mo

Conclusion:

During this analysis we investigated and cleaned the test and train datasets of the Kaggle House Price Competition. The analysis consisted of two parts: the data exploration and the data manipulation. In the first part, we identified the label and the features, looked at their distributions and relationships. In the second part we cutted outliers, imputed missing values, created new features, corrected the skewness of the label and the features. The train and test datasets are cleaned and ready for modeling.