House Prices: Advanced Regression Techniques

Pawel Sobieralski

August 6, 2020

Abstract

A variant of Kaggle House Price competition where house prices are predicted as either high or low. The implementation consists of the below steps.

· Automatic Data Profiling with Scikit-Learn

· Exploratory Data Analysis

· Preprocessing and Imputations with Pipelines

· Features Selection with Recursive Feature Elimination and Cross Validation

· Modeling and Parameters Grid Search with CV

· Models Ensemble

· Visualisations

The solution evaluation is assessed by cross validations calculations over stratified K-Folds. The area under the curve AUC is 0.98 ± 0.01 shows has high predictive power and separability in both classes with balanced accuracy 0.91.

1 Statistical Profiling

Scikit-learn enables automatic profiling by calculating quantile and descriptive statistics and presening them in a visual manner by interactive widget or export to html format

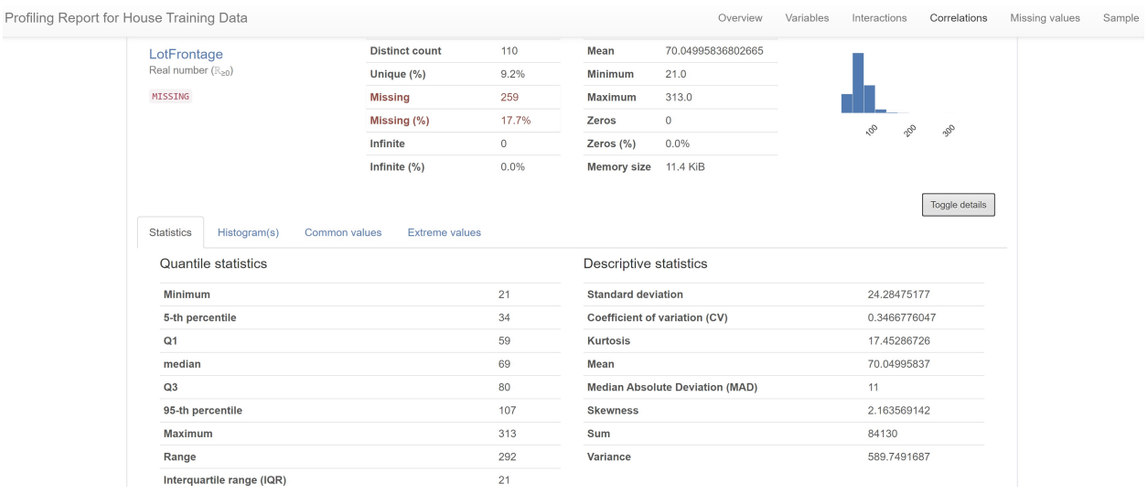


Figure 1: Profiling

2 Exploratory Data Analysis

2.1 Target Feature

2.1.1 Summary

The target feature is an indicator of house price above or below 200, 000 threshold:

- 1 - price above threshold
- 0 - price below threshold

This below is a visual summary of the target feature that reveals its asymmetric nature, particularly in the high value class, greater dispersion and outliers. Clear cut off at the threshold level.

- High value sales greater dispersion and outliers
- Low value sales cluster around their median

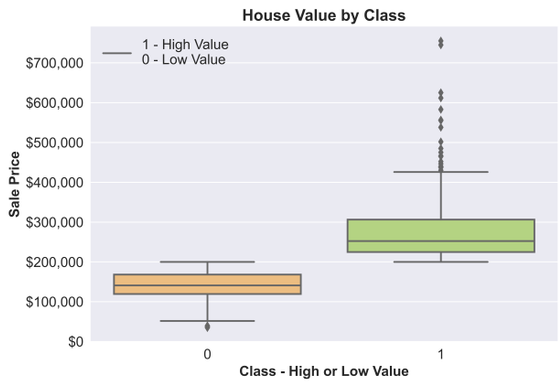
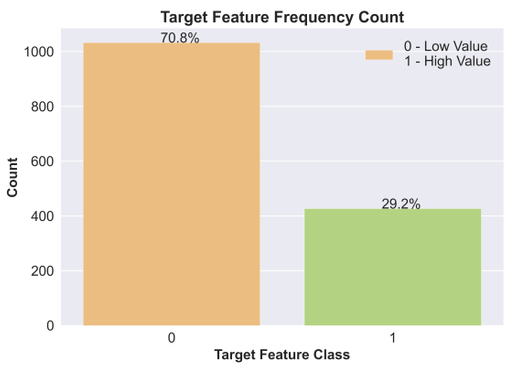


Figure 2: Target Feature Classes

2.1.2 Class Imbalance

The target feature has a Gaussian-like distribution with a long tail and some skewness.

- 70% of sales are classifed as low value
- 30% are high value sales



(a) Target Feature Classification

(b) Target Feature Regression

I recognize this is a moderate imbalance and I address it using stratification. If the Target had bigger imbalance it could benefit from over-sampling or one of the box-cox transforms respectively for classification and regression.

2.2 Dependent Features

2.2.1 Missing Values Ratio

I calculate missing value ratio for each feature and below is its graphical representation. Features with highest missing ration are on the left.

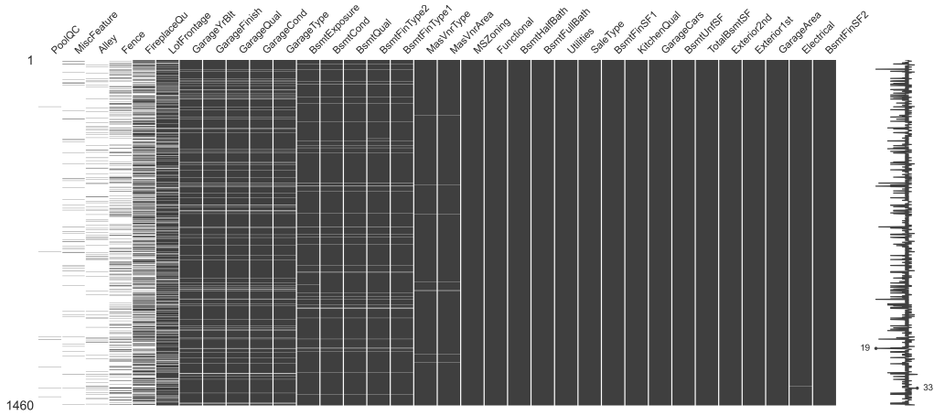


Figure 4: Missing Value Ratio

The following transformation are applied on the missgin data:

- None - denoting not really missing values but rather level absence
- Mode - categorical text type features
- Median - numeric type features with discrete values
- Mean - decimal values features
- Custom Estimator - regression for domain specific features

2.2.2 Features Highly Correlated

- Target feature has strong linear relationship with with two independent features Living Area and Overall Quality. These are potentially our two strongest predictors driving high value sales.
- Some independent features are also strongly correlated between themselves and we will seek to reduce this by considering their presence in the training set.

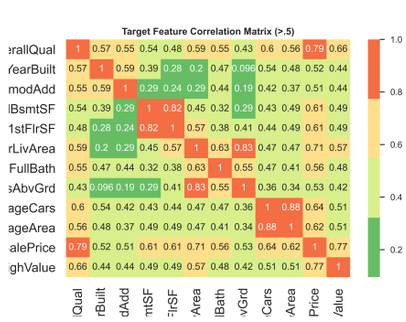


Figure 5: Target Feature Correlation

	Feature 1	Feature 2	Correlation
0	GarageArea	GarageCars	0.889700
2	YearBuilt	GarageYrBlt	0.834812
4	GrLivArea	TotRmsAbvGrd	0.808354
6	TotalBsmtSF	1stFlrSF	0.801670
8	TotRmsAbvGrd	BedroomAbvGr	0.669737
10	GrLivArea	2ndFlrSF	0.655085
12	GarageYrBlt	YearRemodAdd	0.652365

Figure 6: Independent Features Correlation

3 Processing

The processing components are arrange in custom class Pipelines. Pipelines are supposed to minimize knowledge leaking. They also help arrange blocks of processing in more production application. For the Knowledge leaking consideration Scalers are placed down the processing line and enclosed together with model.

- Custom Estimator Pipeline for Domain Specific Imputations
- Mean, Mode, Median Imputation and Label Encoding
- Systematic Feature Search and Selection with Custom RFE CV class
- Model Enclosed Together with Scaler

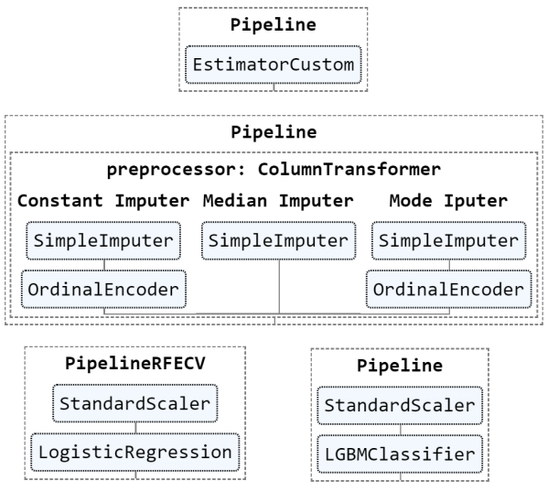


Figure 7: Pipeline Processing

4 Feature Selection

4.1 Feature Importance - Initial Assessment

For introductory feature selection I run random forest. This approach attempts to assess which features contribute most to explain variance in the target variable or in the case of classification Gini impurity. The most important features appear at the top of each tree. This approach has some caveats and needs cautious - I control the depth of the trees and take feature cardinality into account.

- 75% features account for 99% variability in the target variable.
- 25% features account only for 1% and we drop these features from the selection.

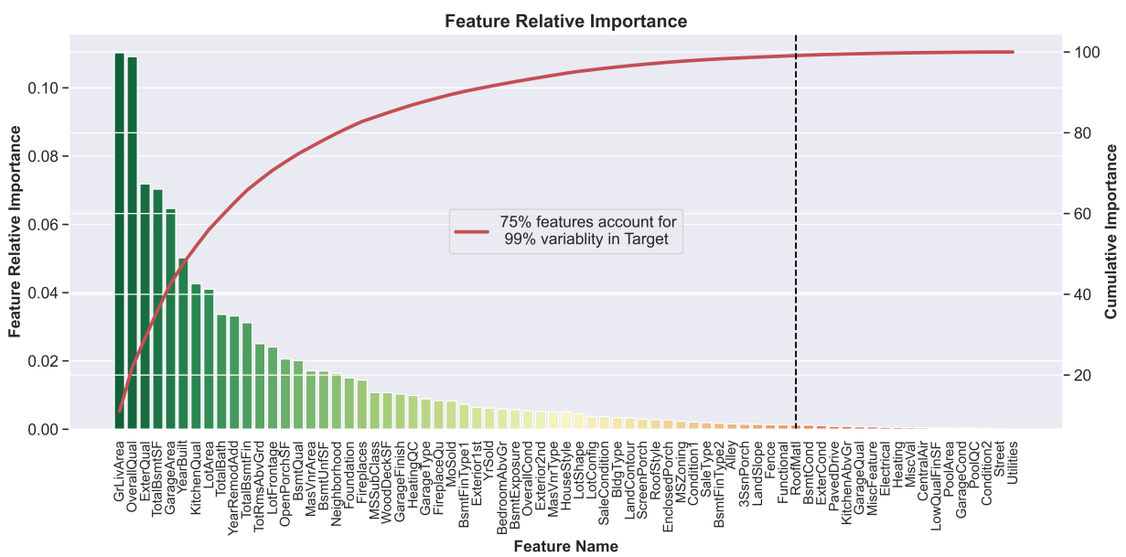


Figure 8: Feature Selection

4.2 Recursive Feature Elimination with Cross-Validation

More methodical aapproach to feature selection is to run optimization for specific metrics like area under the curve AUC or balanced accuracy by search in the independent features domain.

- AUC and Balanced Accuracy reach optimum for 40+ features, depending on model
- Set of models considered: GBM, Logistic Regression, Random Forest, MLP

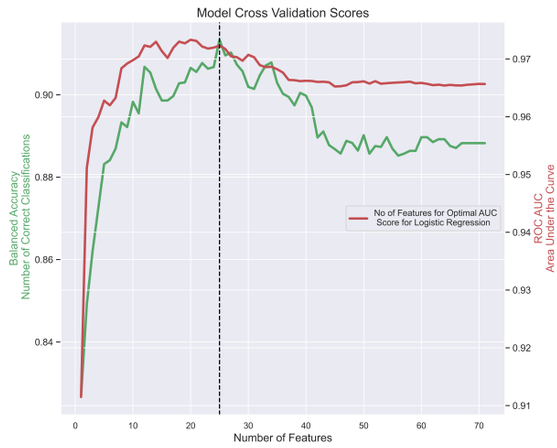


Figure 9: Cross Validation Scores

Finally, I consider a separate set of features for each model. Feature selection and scaling play bigger role for logistic regression more than for the tree methods.

4.3 New Features

Based on high correlation or low variance findings in the data exploration section and problem domain I create new or drop some features, for example

- Introduce new features:
  TotalBsmtFin = BsmtFinSF1 + BsmtFinSF2
- Arbitrarily drop features: *BsmtFinSF1, BsmtFinSF2, FullBath, HalfBath, BsmtFullBath, BsmtHalfBath, 1stFlrSF, 2ndFlrSF, 1stFlrSF, 2ndFlrSF, GarageCars, GarageYrBlt*

5 Modeling

Build model ensemble with a set of different models.

5.1 GBM with Grid Search and Cross Validation

The solution implements GBM model and estimates its parameters with Grid Search and Cross Validation.

5.1.1 ROC Characteristics

The model has a good ability to separate classes wiht AUC 0.98 ± 0.01

5.1.2 Cross Validation Confusion Matrix

The matrix is build in an additive manner by enumerating over stratified kfold.

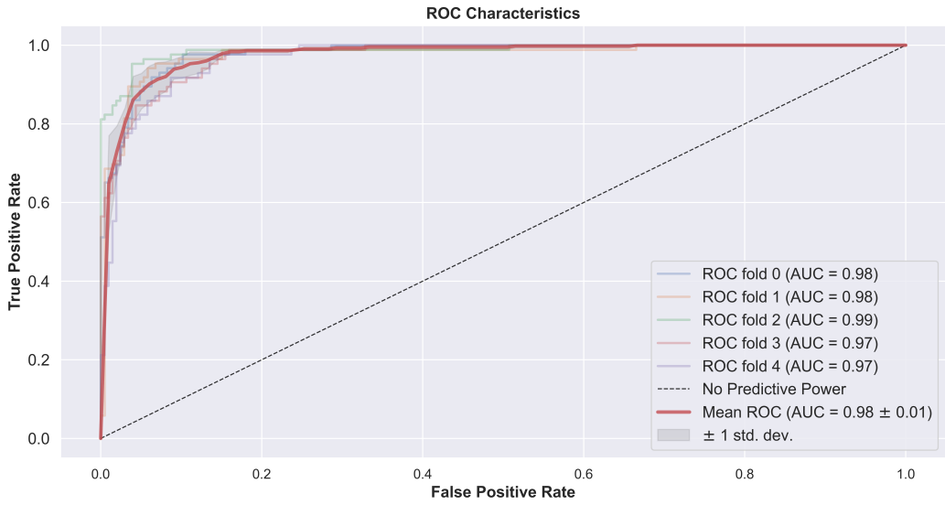


Figure 10: ROC Characteristics with KFolds

- Confusion matrix is calculated in each CV KFold in additive manner.
- Green and Yellow Squares are True Predictions
- Red Squares are False Predictions

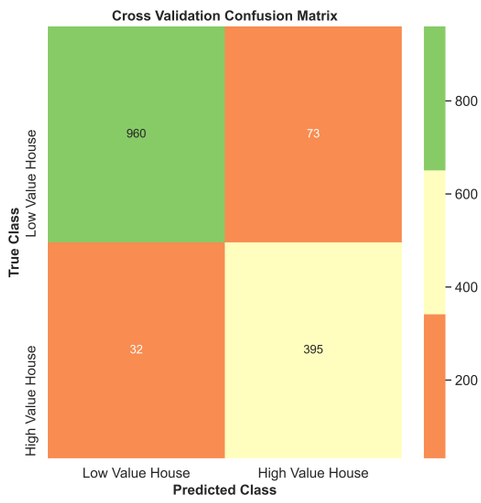


Figure 11: Cross Validation Confusion Matrix

5.1.3 Classification Report

```
from sklearn.metrics import classification_report
target_names = ['Low Value House', 'High Value House']
print(classification_report(cv_true_y, cv_predicted_y, target_names=target_names))
```

```
                precision    recall  f1-score   support

 Low Value House       0.97      0.93      0.95      1033
High Value House       0.84      0.93      0.88       427

        accuracy                           0.93      1460
       macro avg       0.91      0.93      0.92      1460
    weighted avg       0.93      0.93      0.93      1460
```

```
from sklearn.metrics import balanced_accuracy_score,accuracy_score

accuracy_score(cv_true_y, cv_predicted_y), balanced_accuracy_score(cv_true_y, cv_predicted_y)
```

```
(0.928082191780822, 0.9271952953018765)
```

Figure 12: Classification Report