Machine Learning: Fundamentals and Applications

House Price Prediction

Group Members

Harsh Agrawal
Jayraj Rameshbhai Kanani
Nidhi Singh
Sophia Roper

TABLE OF CONTENTS

Dataset

- This dataset comprises 79 explanatory variables that cover practically every facet of Ames, Iowa's residential dwellings.
- Has 1460 rows and 79 columns related to residential homes
- Target value is Sales Price to predict final price of each home

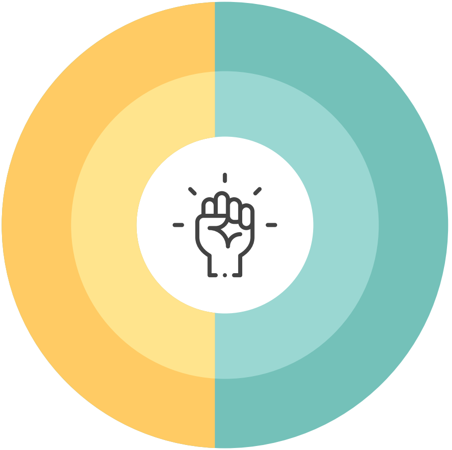
House price prediction

Problem Statement

When you ask a home buyer to describe their ideal home, they are unlikely to start with the basement ceiling height or the closeness to an east-west railroad. However, the data from this playground competition shows that far more factors influence price negotiations than the number of bedrooms or the presence of a white-picket fence. However, the data in this thesis shows that the number of bedrooms and floors have a greater impact on the price of a home. I also want to use this dataset to anticipate an acceptable home price based on these attributes of the properties.

GOALS

The goal of our Project is to train machine learning models to predict the housing prices and find which aspects of the house influence the housing prices mostly.



EDA Analysis

Correlation Matrix

Obtained correlation between two variable to check the highly correlated variable with its dependent variable

Dropping Columns

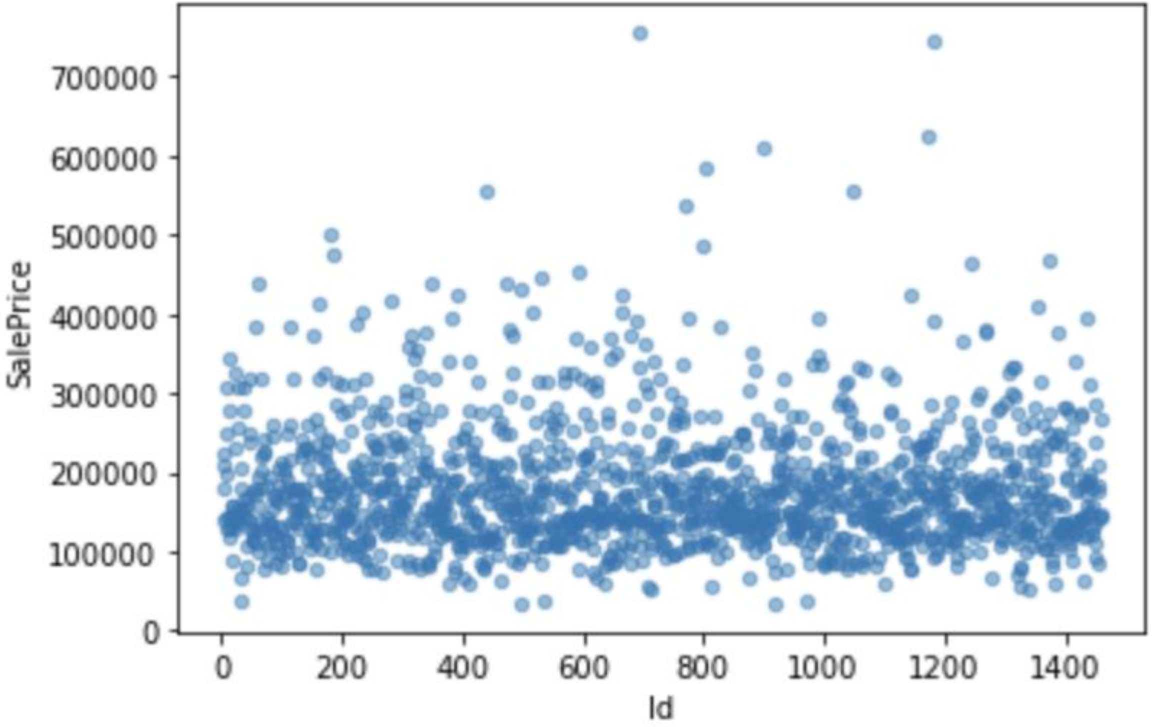
Columns with mostly NaN values were dropped, including FireplaceQu, Alley, Id, PoolQC, Fence, and MiscFeature

Using only necessary columns

Eliminated all the least important columns from the dataset

Scatter plot of the sale price

Most of the points are assembled on the bottom. And there seems to be no large outliers in the sale price variable



Heatmap

Observation

- No direct patterns are observed front heatmap.
- Continuous numeric values seem to affect correlation most.
- The column “OverallQual” seems to be directly correlated to the SalePrice

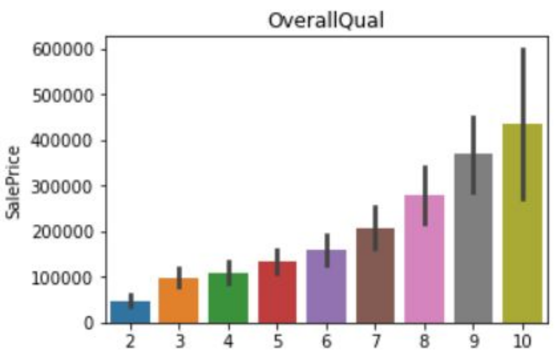
	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt
Id	1.000000	0.015540	-0.014479	-0.042315	-0.058371	0.008627	-0.022610
MSSubClass	0.015540	1.000000	-0.389466	-0.197903	0.031639	-0.085553	0.021605
LotFrontage	-0.014479	-0.389466	1.000000	0.419714	0.241169	-0.047132	0.107958
LotArea	-0.042315	-0.197903	0.419714	1.000000	0.169876	-0.033113	0.028954
OverallQual	-0.058371	0.031639	0.241169	0.169876	1.000000	-0.189587	0.590761
OverallCond	0.008627	-0.085553	-0.047132	-0.033113	-0.189587	1.000000	-0.437647
YearBuilt	-0.022610	0.021605	0.107958	0.028954	0.590761	-0.437647	1.000000
YearRemodAdd	-0.030239	0.010178	0.082938	0.024308	0.568582	0.024427	0.625905
MasVnrArea	-0.072344	0.040009	0.189769	0.106600	0.419756	-0.174581	0.328897
BsmtFinSF1	-0.013234	-0.069439	0.239734	0.232341	0.230438	-0.068285	0.234207
BsmtFinSF2	0.014964	-0.073834	0.046928	0.138615	-0.081342	0.040598	-0.058987
BsmtUnfSF	-0.014316	-0.147155	0.111368	0.008924	0.297384	-0.169743	0.170077
TotalBsmtSF	-0.024541	-0.264277	0.407566	0.324476	0.547448	-0.243419	0.423763
1stFlrSF	-0.007492	-0.258207	0.453035	0.331295	0.527908	-0.166191	0.311928
2ndFlrSF	-0.005997	0.319176	0.074953	0.075311	0.265906	0.004047	-0.021327
LowQualFinSF	-0.040553	0.024935	0.010748	0.019956	-0.011186	0.047865	-0.165742
GrLivArea	-0.013772	0.078213	0.397260	0.308590	0.610102	-0.115250	0.198778

Snippet of heatmap. Complete heatmap can be found in the notebook

EDA Analysis

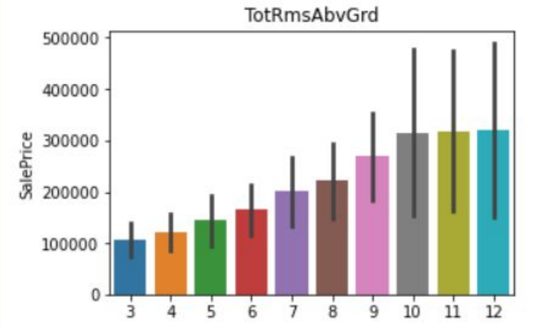
Overall Qual

High direct correlation
is observed



TotRmsAbvGrd

High direct correlation
is observed



RMSE -root mean square error

$$RMSE(\hat{\theta}) = \sqrt{MSE(\hat{\theta})} = \sqrt{E((\hat{\theta} - \theta)^2)}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}}$$

Linear Regression

```
lreg = LinearRegression()
lreg.fit(X_train,y_train)
mse = mean_squared_error(y, lreg.predict(X))
RMSE = math.sqrt(mse/num_data)
print(RMSE)
```

```
939.6790554728881
```

Random Forest

```
RF = RandomForestRegressor(n_estimators=500,bootstrap=True,random_state=1,oob_score=True)
RF = RF.fit(X_train,y_train)
y_pred = RF.predict(X)
num_data = X.shape[0]
mse = mean_squared_error(y, y_pred)
RMSE = math.sqrt(mse/num_data)
print(RMSE)
```

```
403.8107826066123
```

Gaussian Process

```
kernel = DotProduct() + WhiteKernel()
gp = GaussianProcessRegressor(kernel=kernel,random_state=42)
gp.fit(X_train, y_train)
mse = mean_squared_error(y, gp.predict(X))
RMSE = math.sqrt(mse/num_data)
print(RMSE)
```

```
1054.74356190732
```

Xgboost

```
xg_reg = XGBRegressor(max_depth=5, n_estimators=10)
xg_reg.fit(X_train,y_train)
mse = mean_squared_error(y, xg_reg.predict(X))
RMSE = math.sqrt(mse/num_data)
print(RMSE)
```

```
[14:19:45] WARNING: /workspace/src/objective/regression
1984.8613427728646
```

Stacking

```
def get_stacking():
    # define the base models
    level0 = list()
    #level0.append(('lr', LinearRegression()))
    level0.append(('gp', GaussianProcessRegressor(kernel=kernel,random_state=42)))
    level0.append(('rc', RandomForestRegressor(n_estimators=500,bootstrap=True,random_state=1,oob_score=True)))
    level0.append(('xgb', XGBRegressor(max_depth=5, n_estimators=10)))
    # define meta learner model
    level1 = LinearRegression()
    # define the stacking ensemble
    model = StackingRegressor(estimators=level0, final_estimator=level1, cv=5)
    return model
```

```
stacked = get_stacking()
stacked.fit(X_test, y_test)
y_pred = stacked.predict(X)
```

```
num_data = X.shape[0]
mse = mean_squared_error(y, y_pred)
RMSE = math.sqrt(mse/num_data)
print(RMSE)
```

```
823.7815744511447
```

Models used

Linear Regression
RMSE: 939.68

Gaussian Process
RMSE: 1054.74

Stacked
RMSE: 823.78

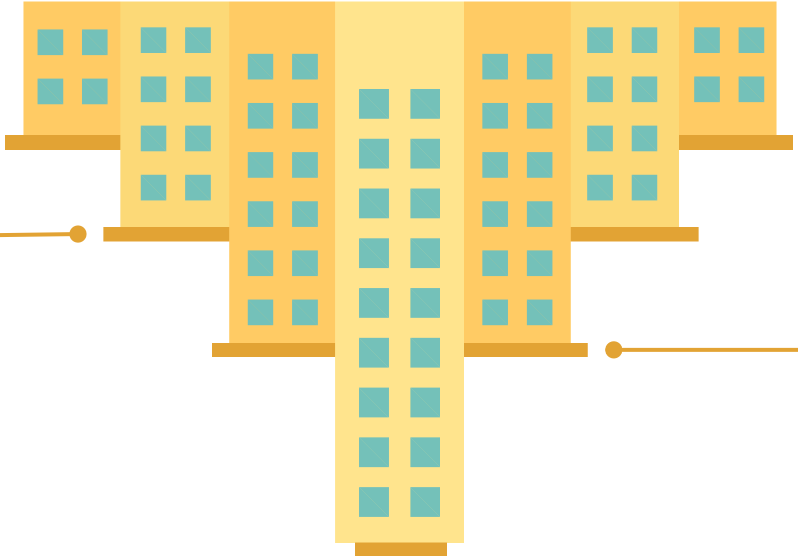
Random Forest
RMSE: 403.81

XGBoost
RMSE: 1984.86

Conclusion

Best Model

Using a Random Forest Regressor to train the data achieved the lowest RMSE



Stacking

The Stacking model, with Linear Regression as its final classifier, achieved the next to lowest RMSE