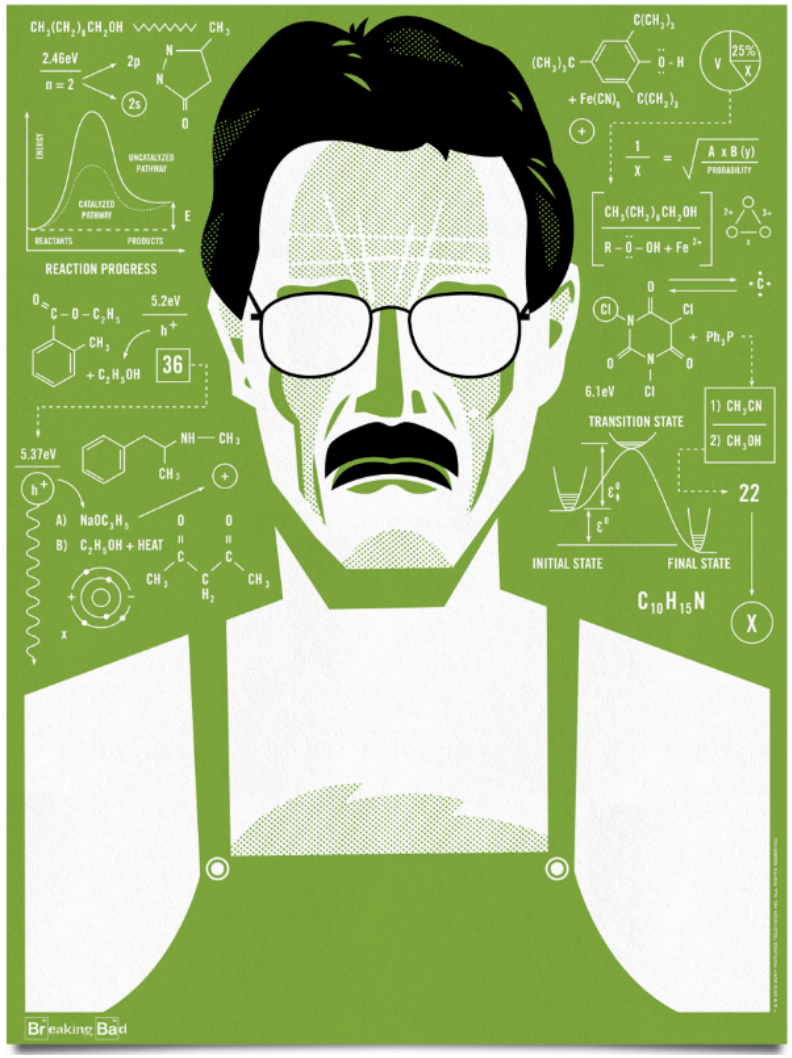


Breaking Bias

Housing prices of Ames

The Great Mystery

- Which model/models
- With what features/predictors
- Best predict the housing prices of Ames, Iowa

Peek on Data Set

- **Data size**

	row	column
train	1460	81
test	1459	80

- **Data types**

object	43
int64	26
float64	12

- **Variable Names**

['MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street', ...]

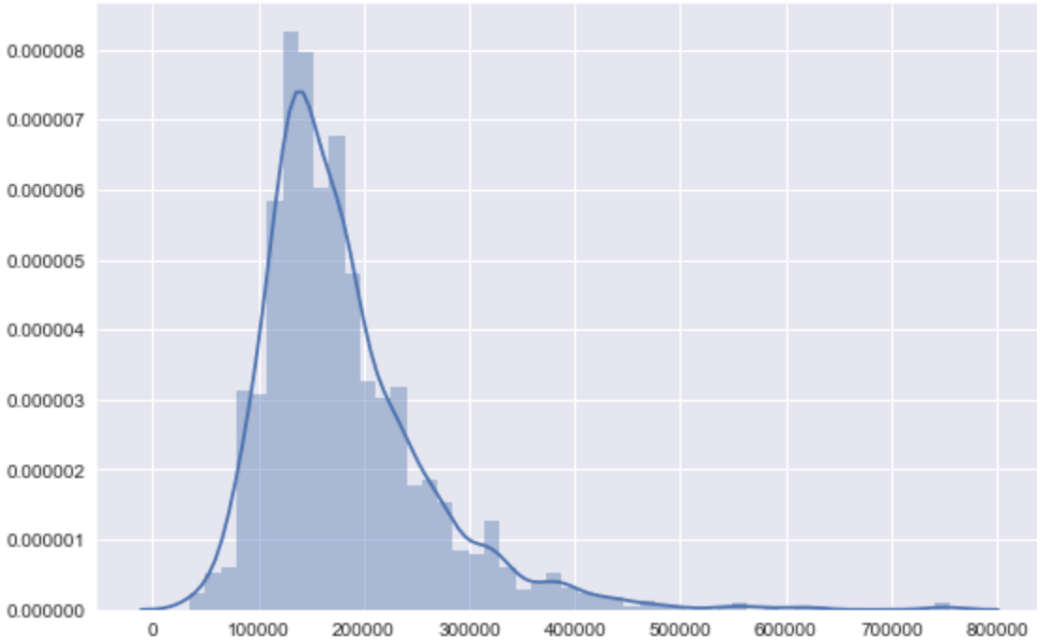
- **Data head**

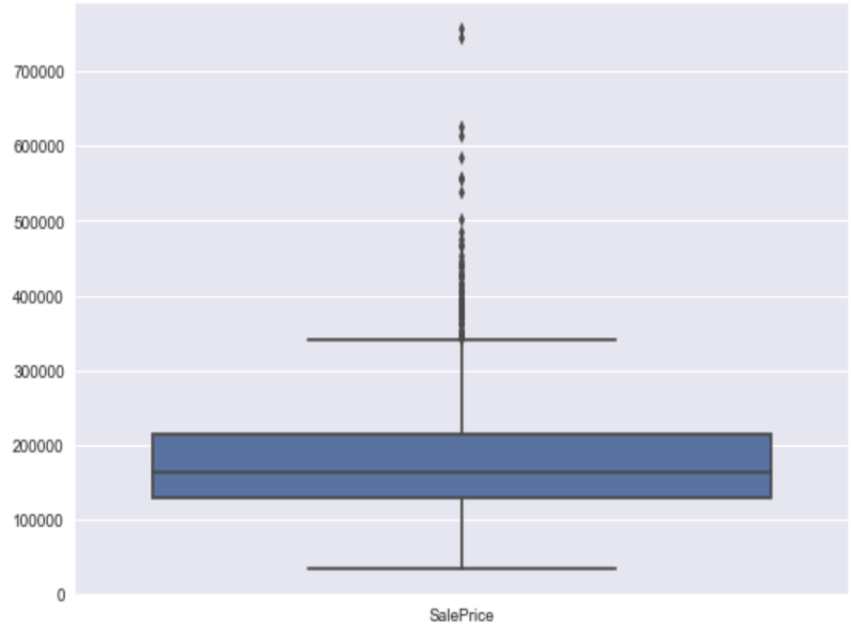
	Id	MSSubClass	MSZoning	LotFrontage	LotArea
0	1	60	RL	65.0	8450
1	2	20	RL	80.0	9600
2	3	60	RL	68.0	11250

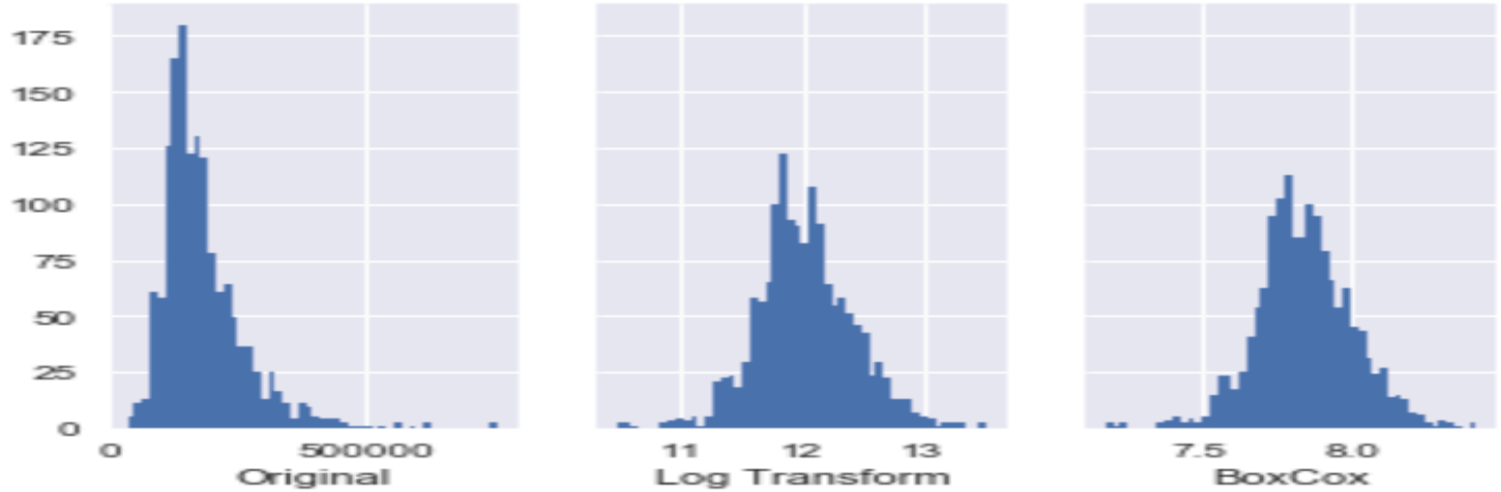
- **Data summary**

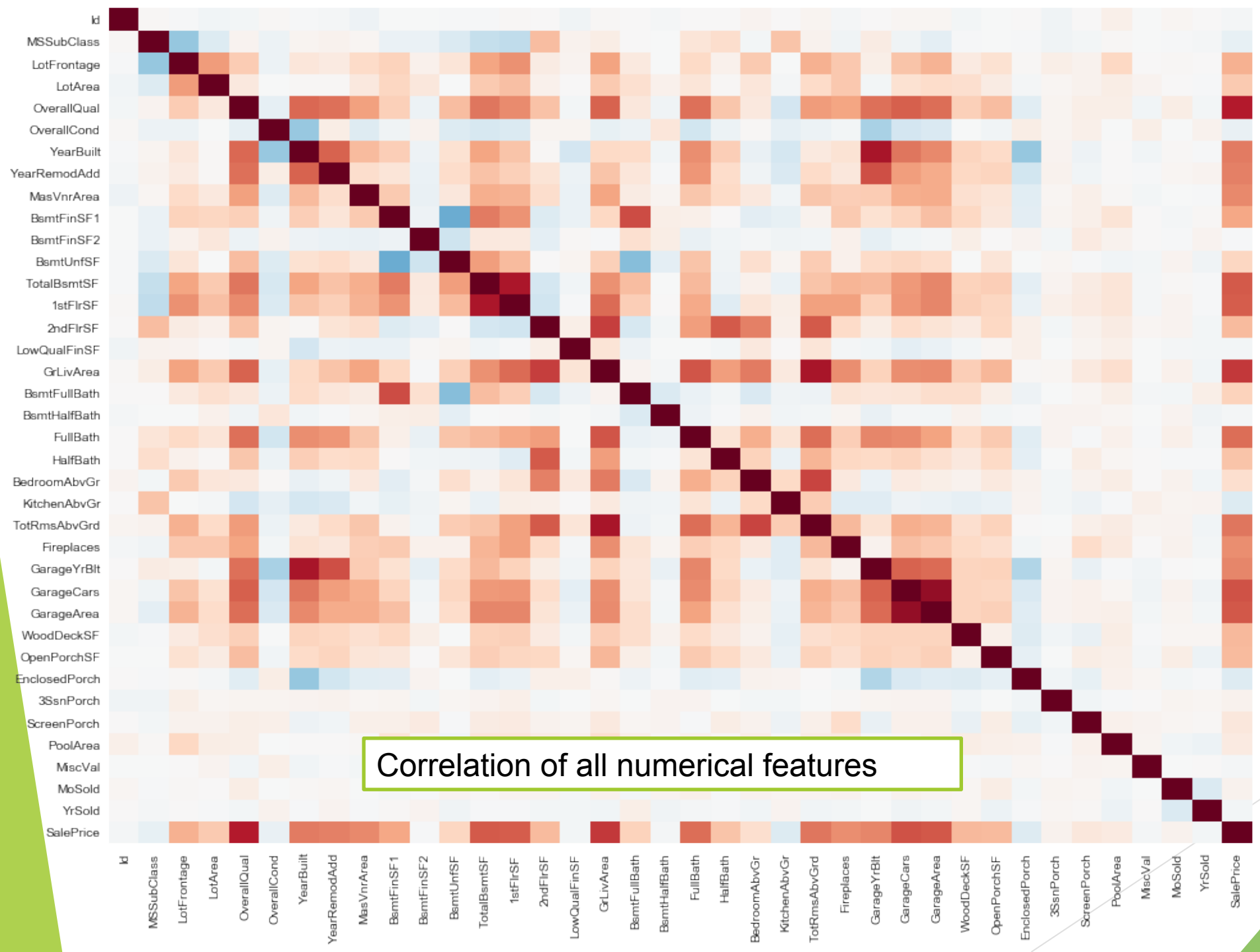
	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond
count	1460.000000	1460.000000	1201.000000	1460.000000	1460.000000	1460.000000
mean	730.500000	56.897260	70.049958	10516.828082	6.099315	5.575342
std	421.610009	42.300571	24.284752	9981.264932	1.382997	1.112799
min	1.000000	20.000000	21.000000	1300.000000	1.000000	1.000000
25%	365.750000	20.000000	59.000000	7553.500000	5.000000	5.000000
50%	730.500000	50.000000	69.000000	9478.500000	6.000000	5.000000
75%	1095.250000	70.000000	80.000000	11601.500000	7.000000	6.000000
max	1460.000000	190.000000	313.000000	215245.000000	10.000000	9.000000

What does Y(SalePrice) look like?









Correlation of all numerical features

Most Related Features

Garage Cars
~
Garage Area

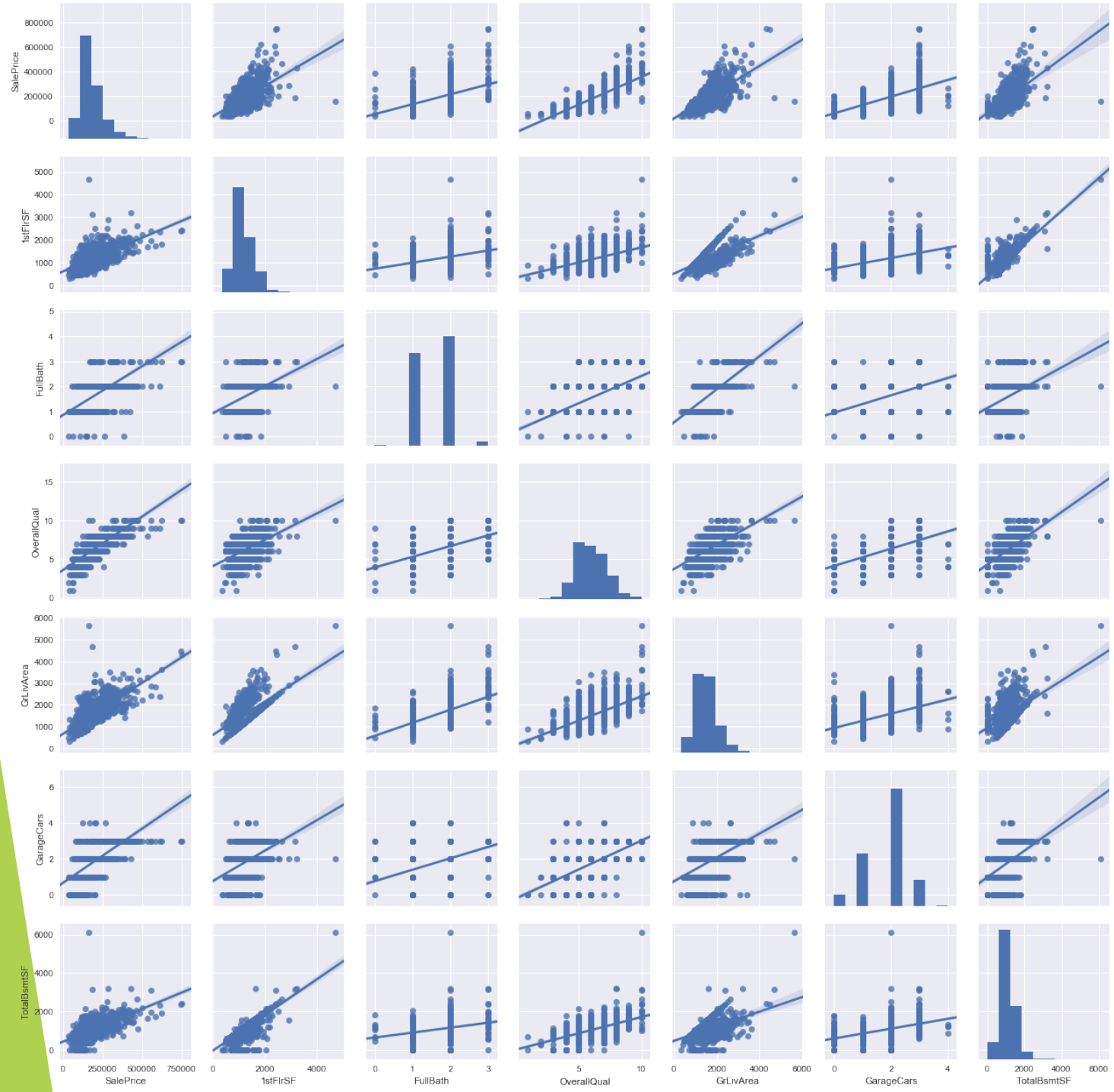
YearBuilt
~
GarageYrBlt

1stFlrSF
~
TotalBsmtSF

Most related to Y

SalePrice
~
OverallQual

SalePrice
~
GrLivArea



More explore on most related features

-- Linear

-- Variance

-- Normality / Skewness (need to log?)

What we found?

-- strong linear between some features

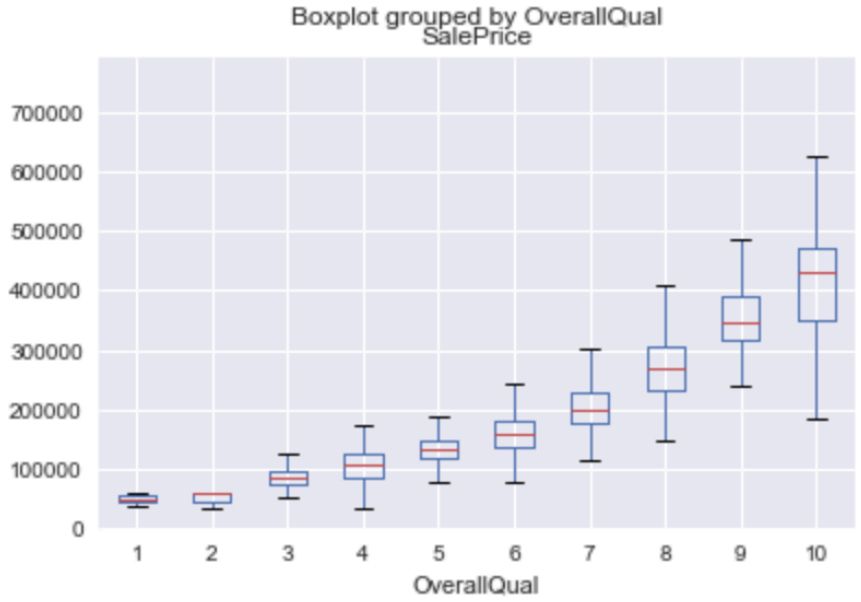
Have questions?

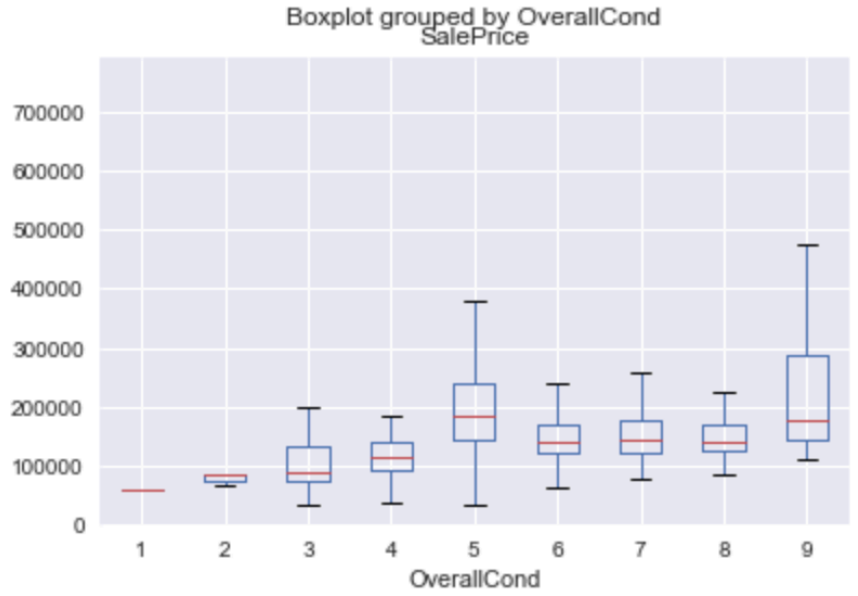
-- 4 cars garage prices?

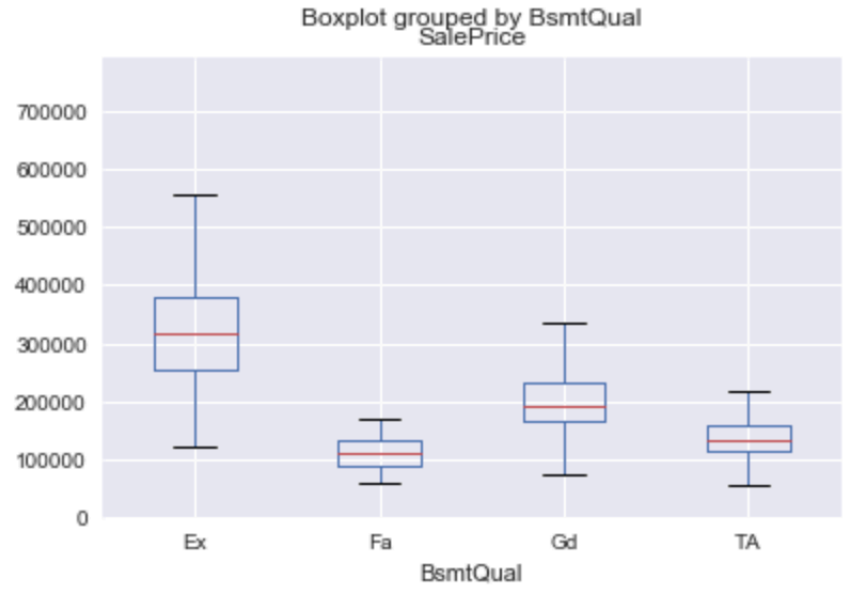
GarageCars	GarageArea	SalePrice
4	784	206300
4	864	265979
4	1356	168000
4	480	123000
4	968	200000

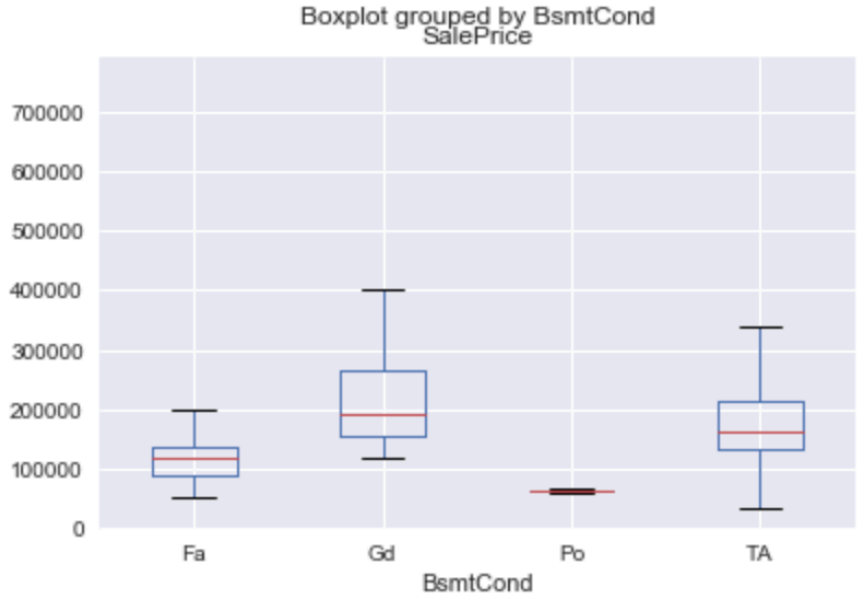
GarageCars	mean
0	0.000000
1	300.517615
2	519.797330
3	811.574586
4	890.400000

Some related features (boxplot)

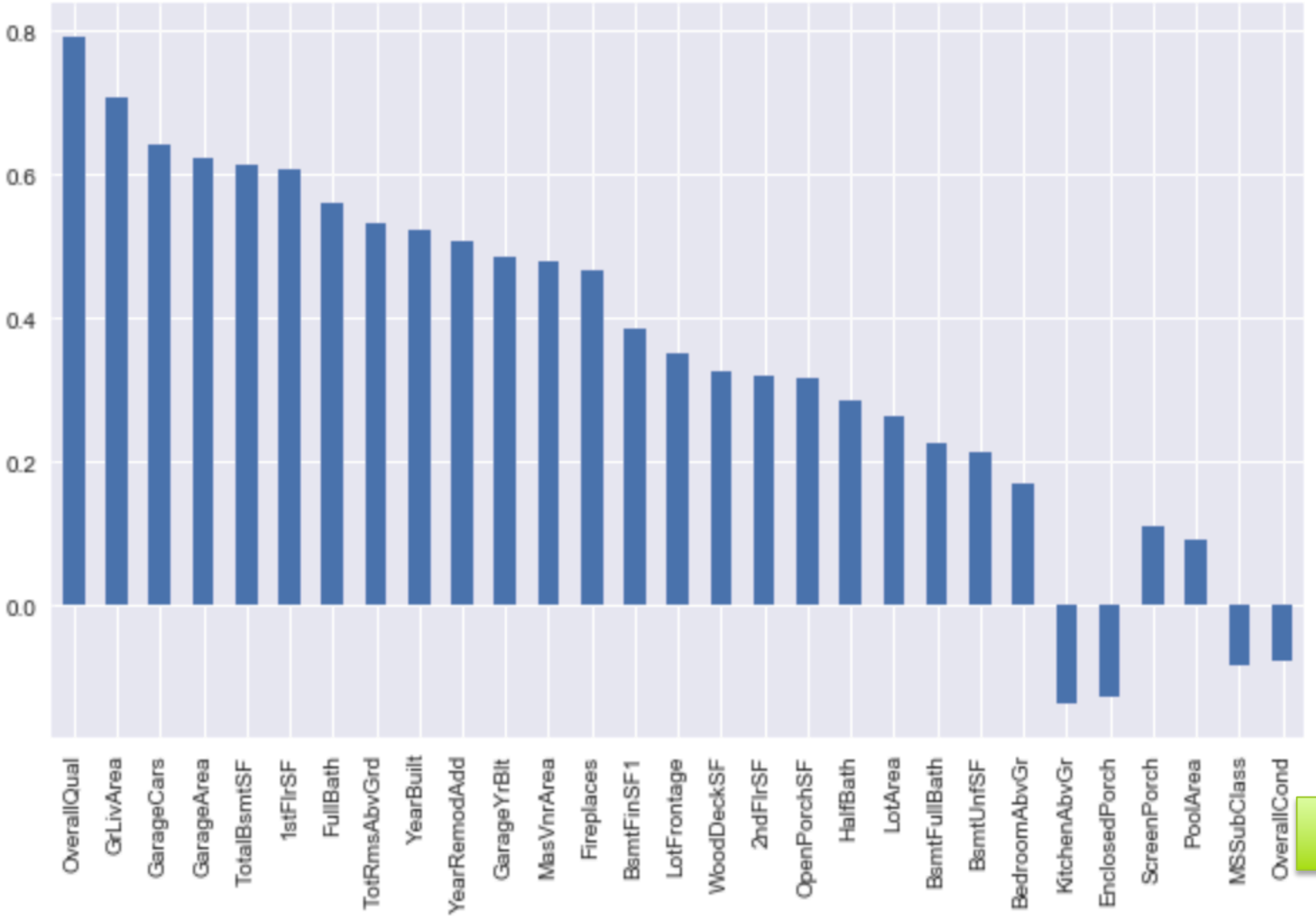




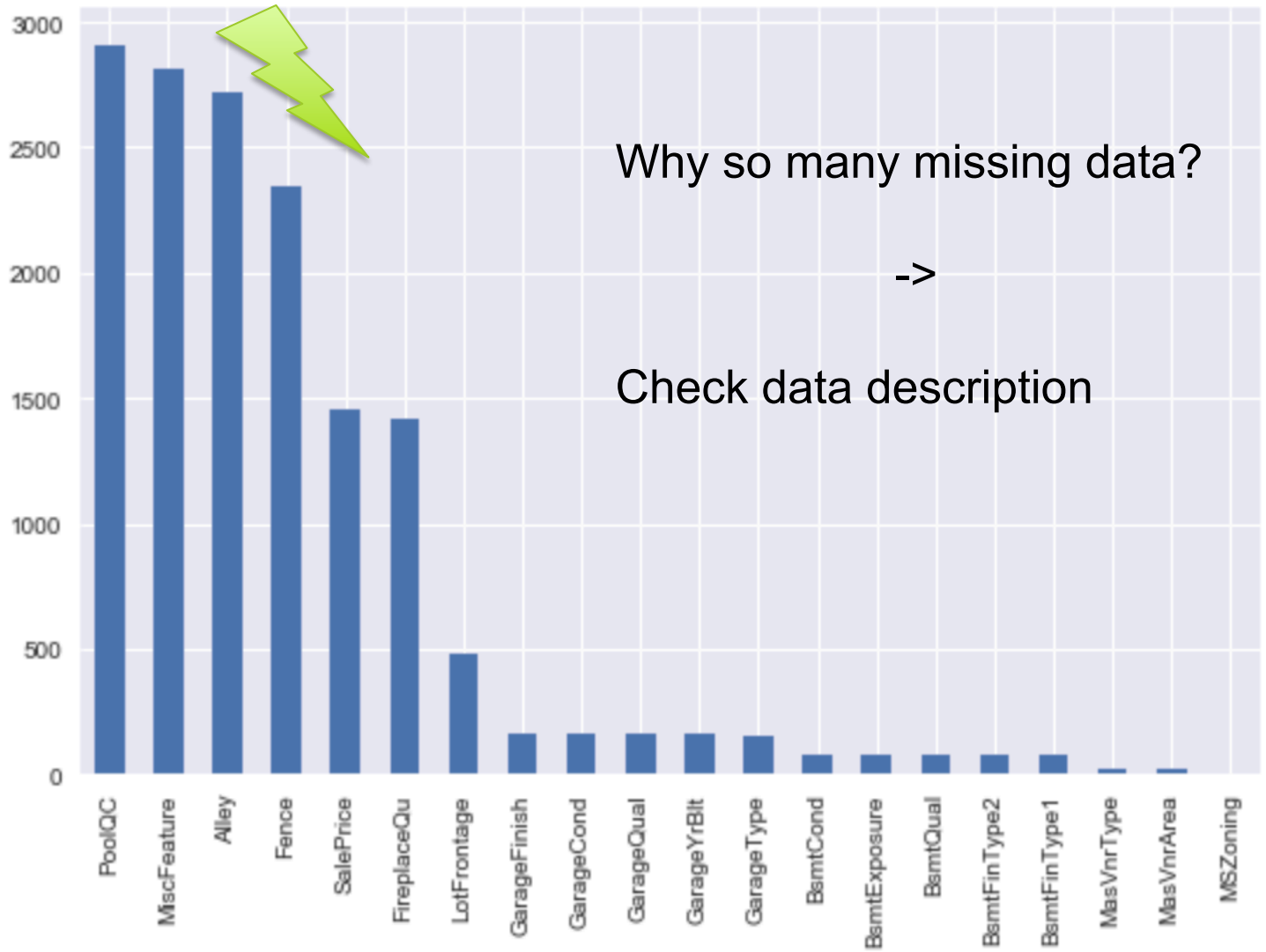




Sorted features by Corr with Y(HousePrice)



Missing Data Visualization



Why so many missing data?

->

Check data description

More about missing data!

Attention!

SCALE

Initial Cleaning / Imputing

- Many columns have missing values, and not all can be treated the same way
- Some are missing because they are just not applicable to the house in question (e.g. columns giving info about fireplaces/pools/garages for houses which do not have those things)
- Some just seem not to have been recorded, and it makes sense to impute them with the mean or median of the column
- Treat years as numerical, but this is an interesting question to think about

Initial Cleaning / Imputing

- A number of columns have a high skewness, including the important SalePrice column from the training data
- We transformed these columns by the transformation $f(x) = \log(x + 1)$ to reduce skew, transforming SalePrice back at the end to get the predicted price
- Skew was lessened by this transformation so the assumptions for regression were closer to being fulfilled.

Feature Engineering

- TotalSF = GrLivArea + TotalBsmtSF
- Took out Pool Quality
- Combining 1stFlrSF, 2ndFlrSF, LowQualFinSF, and GrLivArea all into **Total_Liv_AreaSF**.

  -more just an attempt to eliminate some colinearity
- Unfortunately, our KaggleScore (.12340) was worse than our Kaggle Score (.12290) without feature engineering the data in this way.
- "YrSold" and "MoSold"

Ridge Model

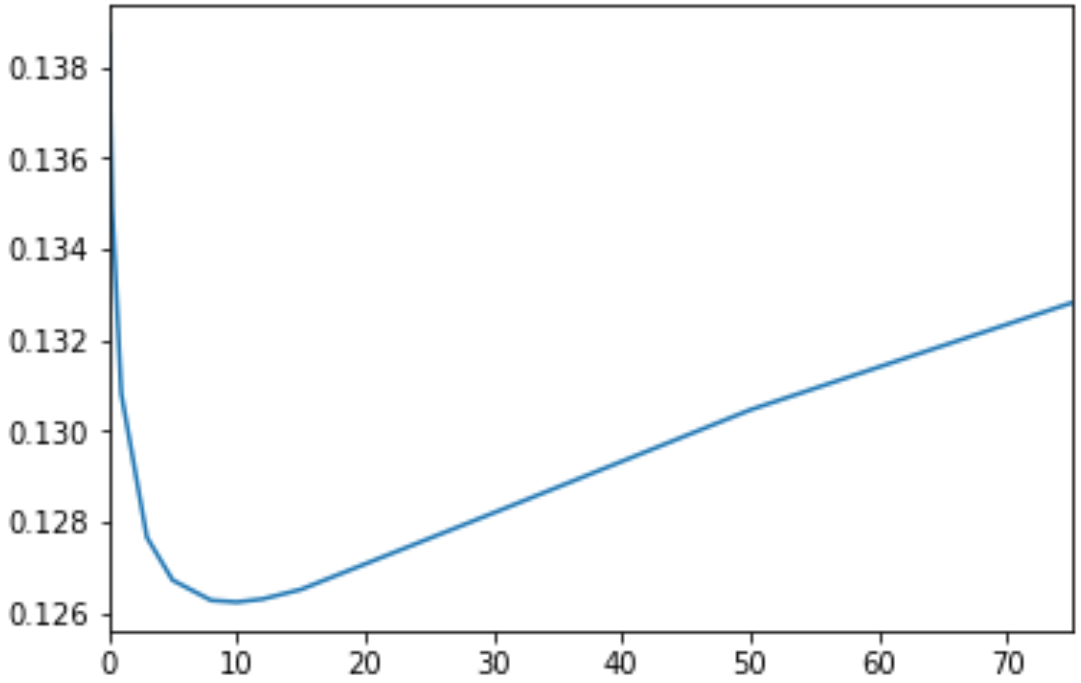
- Why Ridge

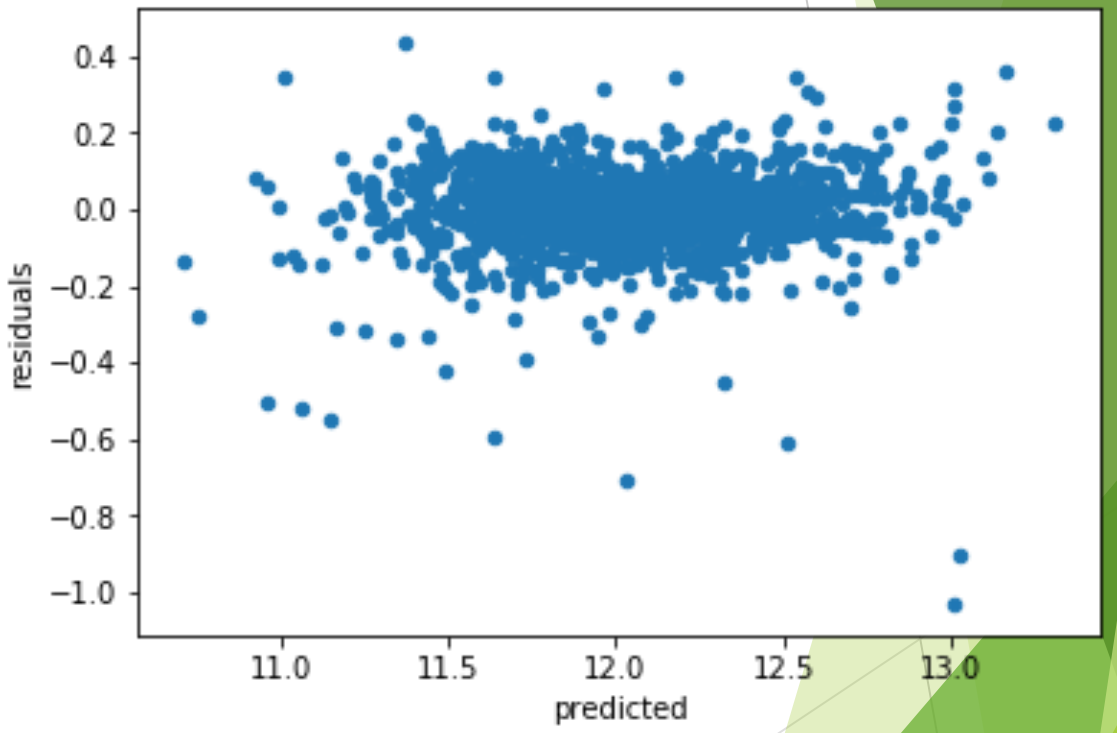
-The main idea behind our group using Ridge Regression was because of the model's ability to alleviate Multiple Collinearity.

-Adds a small bias factor to the variables in order to alleviate this problem.

- Alpha = 10 is best
- RMSE = 0.12623
- Kaggle Score = 0.12479

Ridge model

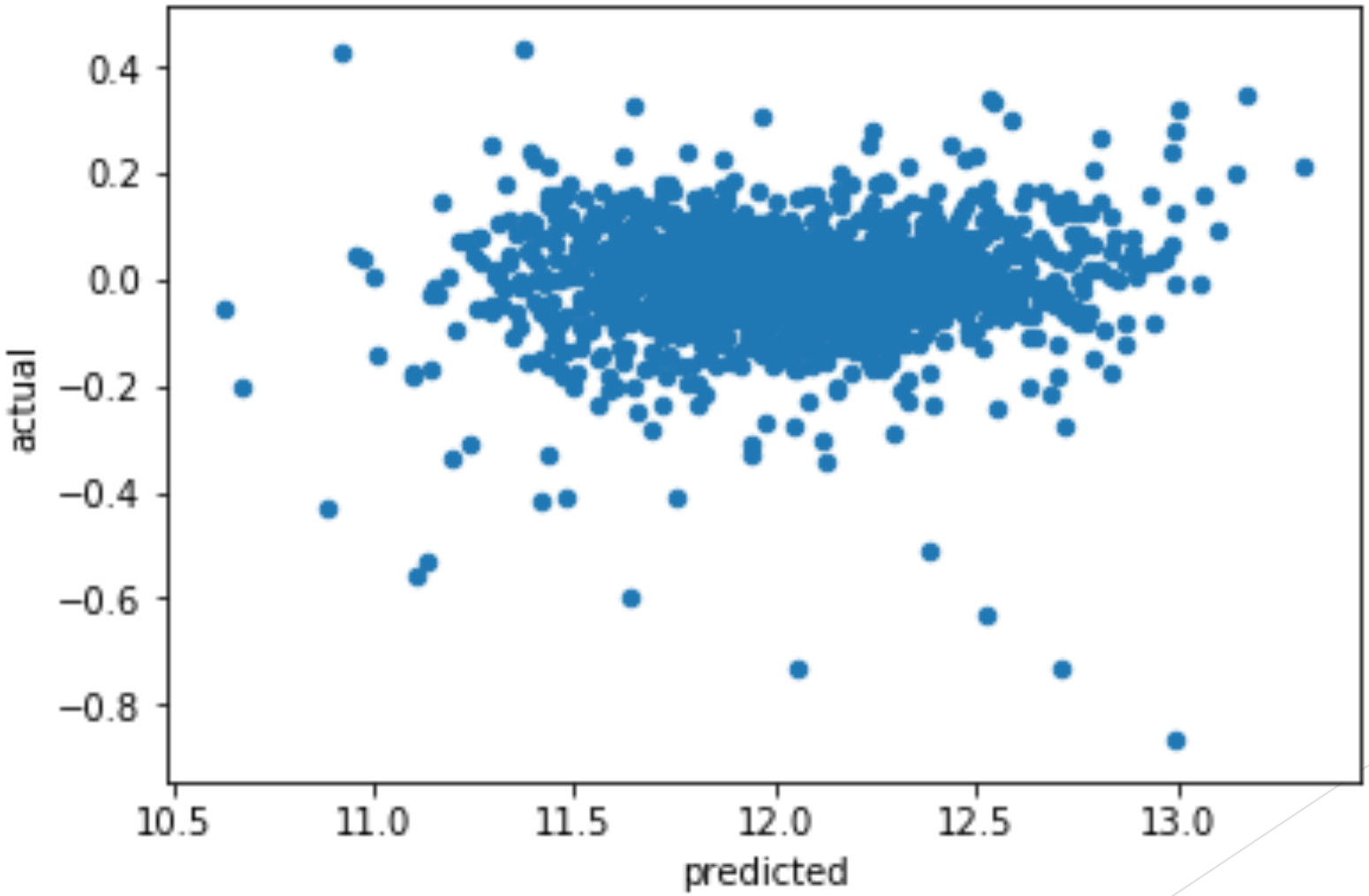

0.138
0.136
0.134
0.132
0.130
0.128
0.126
0
10
20
30
40
50
60
70


residuals
predicted
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
−1.0
11.0
11.5
12.0
12.5
13.0

Lasso Model

- The limits of Ridge regression
- Lasso model is useful because of its tendency to prefer solutions with fewer parameters.
- The solutions are dependent on much fewer variables because of Lasso Model's ability to have large enough lambda values forcing some parameter estimates all the way to 0.
- Our RMSE for our Lasso Model was 0.12310
- Kaggle CV Score = 0.12290 (Our Best Score)

Lasso Model



Random Forest

- Why Random Forest
  -It is functional for dataset with large number of features (303 in our case).
  -It automatically selects the most important features.
- Optimal Parameters

```
grid_search_forest.best_params_
```

```
{'max_features': 67,
 'min_samples_leaf': 1,
 'min_samples_split': 4,
 'random_state': 0}
```

- Output:

  default model RMSE: 0.15369

  tuning model RMSE: 0.14198

Kaggle Score = 0.14540

Random Forest

- Top 10 important features

	feature	importance
28	OverallQual	0.184119
15	GrLivArea	0.130236
34	YearBuilt	0.071004
106	ExterQual_TA	0.056580
32	TotalBsmtSF	0.047466
13	GarageCars	0.047145
12	GarageArea	0.043041
0	1stFlrSF	0.038725
11	FullBath	0.037914
14	GarageYrBlt	0.027026

- Random Forest seems to be the model with highest RMSE even after tuning some parameters.
- To reach a better result, it may requires additional work on parameter tuning.

Gradient Boosting

- Why gradient boosting model?
  -Frequently outperforms random forest
  -The learning rate can be used to improve each tree by building upon the previous tree.
- Optimal Parameters:

```
{'learning_rate': 0.05,
 'max_depth': 5,
 'max_features': 7,
 'min_samples_split': 4,
 'n_estimators': 1000,
 'random_state': 1}
```

- Output:

default model RMSE = 0.13082

tuning model RMSE = 0.12916

Kaggle Score = 0.13733

Gradient Boosting

- Top 10 important features
- feature selections method could be further implemented

	feature	importance
15	GrLivArea	0.094779
28	OverallQual	0.062939
34	YearBuilt	0.048330
27	OverallCond	0.046223
19	LotArea	0.043830
32	TotalBsmtSF	0.032298
26	OpenPorchSF	0.032239
12	GarageArea	0.031918
4	BsmtFinSF1	0.029367
35	YearRemodAdd	0.028967

	feature	importance
28	OverallQual	0.184119
15	GrLivArea	0.130236
34	YearBuilt	0.071004
106	ExterQual_TA	0.056580
32	TotalBsmtSF	0.047466
13	GarageCars	0.047145
12	GarageArea	0.043041
0	1stFlrSF	0.038725
11	FullBath	0.037914
14	GarageYrBlt	0.027026

Conclusions

- What worked:

- Our best model and score we received was with the Lasso Model. (0.12290)

- What did not work

-Our random forest and Gradient Boosting models did not work as well as our Lasso Model and returned values much worse than our Lasso RMSE.

-Feature Engineering was harder than perceived and many times did not improve our RMSE

- Why?

-This possibly happened because variables weren't altered in ways that could have been more beneficial. Cost complexity may have contributed to the higher score. Also data could have been cleaned or imputed differently as well to give better results.

Future Research

Limit the number of features to reduce complexity in tree based models

Also potentially forge Principal Components to reduce complexity by reducing the number of dimensions

Try using Support Vector Machines, XGBoost, and other potentially useful models
Try stacking, bayesian optimization

Find more effective ways to feature engineer. Better understand Real Estate.