Contents

1. Project Overview

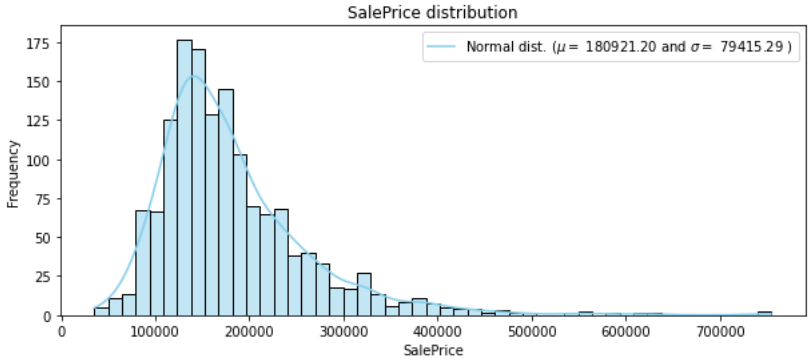
This is a Kaggle competition which is to use Advanced Regression technique to model house price based on 79 explanatory variables describing every aspect of residential homes in Ames, Iowa. In the project, I include the following approaches and techniques:

- EDA with Pandas and Seaborn
- Find features with strong correlation to target
- Creative feature engineering
- Data Wrangling and categorical features encoding
- Regression-based models and tree-based models
- Hyperparameter tuning by GridsearchCV or RandomizedSearchCV
- Model selection based on model performance

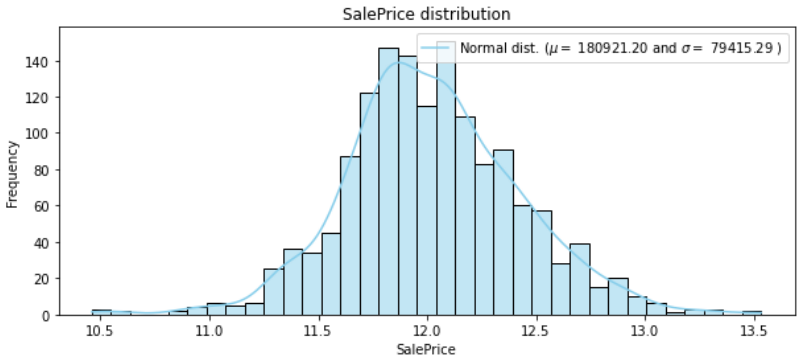
2. Data Exploration and Data Wrangling

2.1 Data Inspection

In my dataset, there are 79 predictor variables and 1 response variable at 1460 observations. The predictor variables are features of the house, for example space, location or physical characteristics of the house while the response variable is the house price.
There is no duplicated house in the data set. Among 79 predictor variables, 36 are numerical variables and 43 are categorical variables.
The target variable (house price) is not normally distributed and is skewed to the right with mean around $180,921 and standard deviation around $79,415.



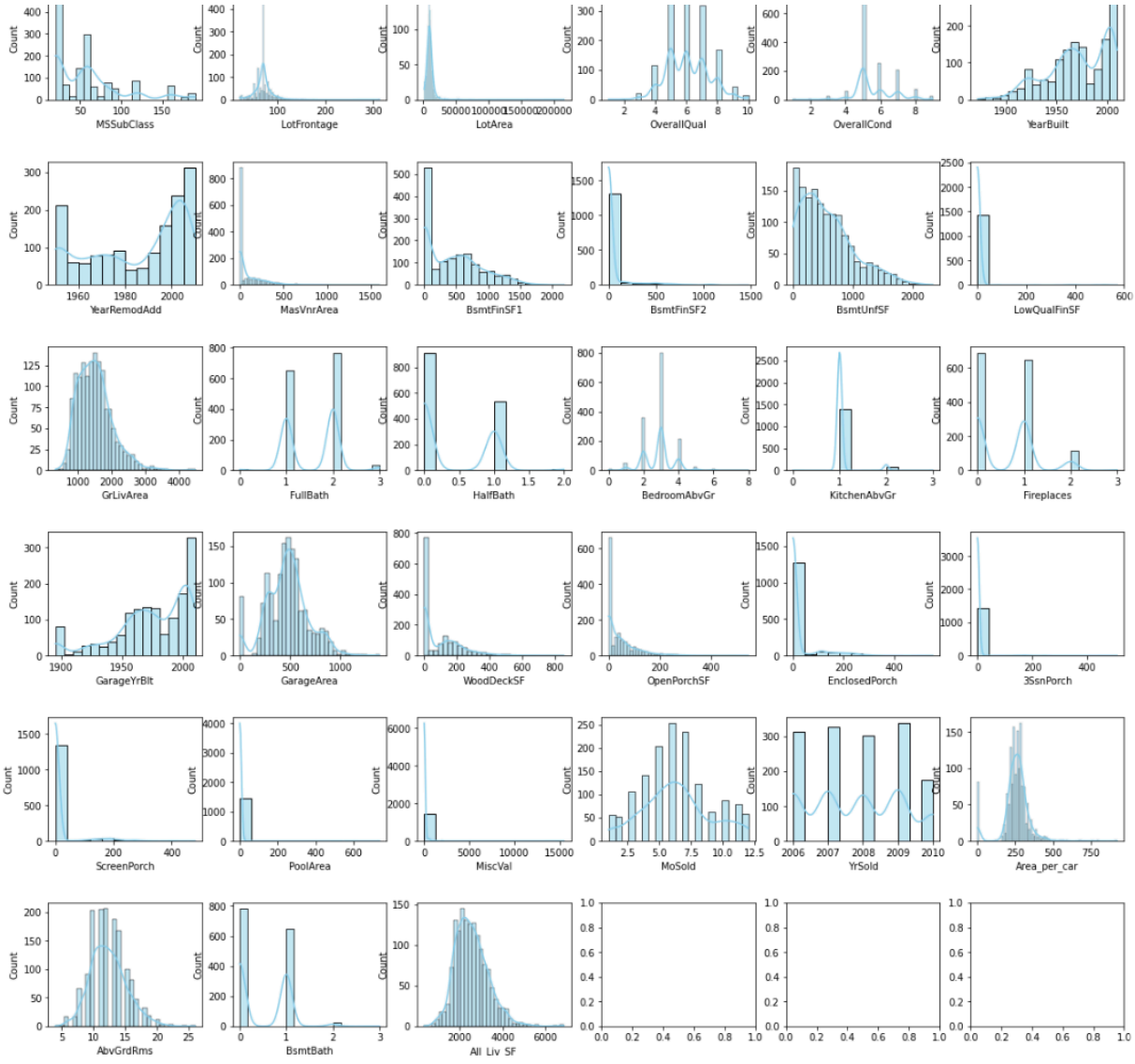
This may reduce the performance of the ML regression model since it assumes normal distribution. Therefore, I make log transformation and the resulting distribution looks much better.



2.2 Data Exploration

Distribution:

Predictor variables distribution: Most of the features are not normally distributed and they are measured in different scales, which suggest I need to standard scale them before modeling.



- Normality: Running a normality test we see that all features are not following a normal distribution.

	Feature	Test-statistics	P_value	Decision
0	MSSubClass	336.330345	9.263895e-74	Not Normal
1	LotFrontage	792.900632	6.665234e-173	Not Normal
2	LotArea	2625.627776	0.000000e+00	Not Normal
3	OverallQual	10.083987	6.460855e-03	Not Normal
4	OverallCond	130.442912	4.728045e-29	Not Normal
5	YearBuilt	97.969727	5.322849e-22	Not Normal
6	YearRemodAdd	2480.188685	0.000000e+00	Not Normal
7	MasVnrArea	872.241582	3.936834e-190	Not Normal
8	BsmtFinSF1	115.222570	9.544140e-26	Not Normal
9	BsmtFinSF2	1284.874812	9.839649e-280	Not Normal
10	BsmtUnfSF	163.553199	3.054036e-36	Not Normal
11	LowQualFinSF	2157.708554	0.000000e+00	Not Normal
12	GrLivArea	250.423512	4.180472e-55	Not Normal
13	FullBath	168.225766	2.952835e-37	Not Normal
14	HalfBath	563.092254	5.321949e-123	Not Normal
15	BedroomAbvGr	89.972034	2.902827e-20	Not Normal
16	KitchenAbvGr	1336.601177	5.764241e-291	Not Normal
17	Fireplaces	89.903893	3.003431e-20	Not Normal
18	GarageYrBlt	175.586065	7.446811e-39	Not Normal
19	GarageArea	24.223390	5.494873e-06	Not Normal
20	WoodDeckSF	427.307869	1.626579e-93	Not Normal
21	OpenPorchSF	759.312018	1.310661e-165	Not Normal
22	EnclosedPorch	957.211862	1.393430e-208	Not Normal
23	3SsnPorch	2345.891478	0.000000e+00	Not Normal
24	ScreenPorch	1247.387243	1.359258e-271	Not Normal
25	PoolArea	2923.539431	0.000000e+00	Not Normal
26	MiscVal	3558.683888	0.000000e+00	Not Normal
27	MoSold	26.240526	2.004205e-06	Not Normal
28	YrSold	1064.791781	6.072990e-232	Not Normal
29	Area_per_car	226.122707	7.908206e-50	Not Normal
30	AbvGrdRms	66.181059	4.255648e-15	Not Normal

- Skewedness: The majority of the features are skewed to the right, which is confirmed by calculating the skewness.

Imputation:

- There are 5 features with missing values more than 40% and 28 features with less than 40%. Most of the missing value is due to the fact that the house doesn't have this specific feature. Therefore, imputing them by either median or mode is not appropriate. Instead, I walk through all these missing values and impute them case by case.

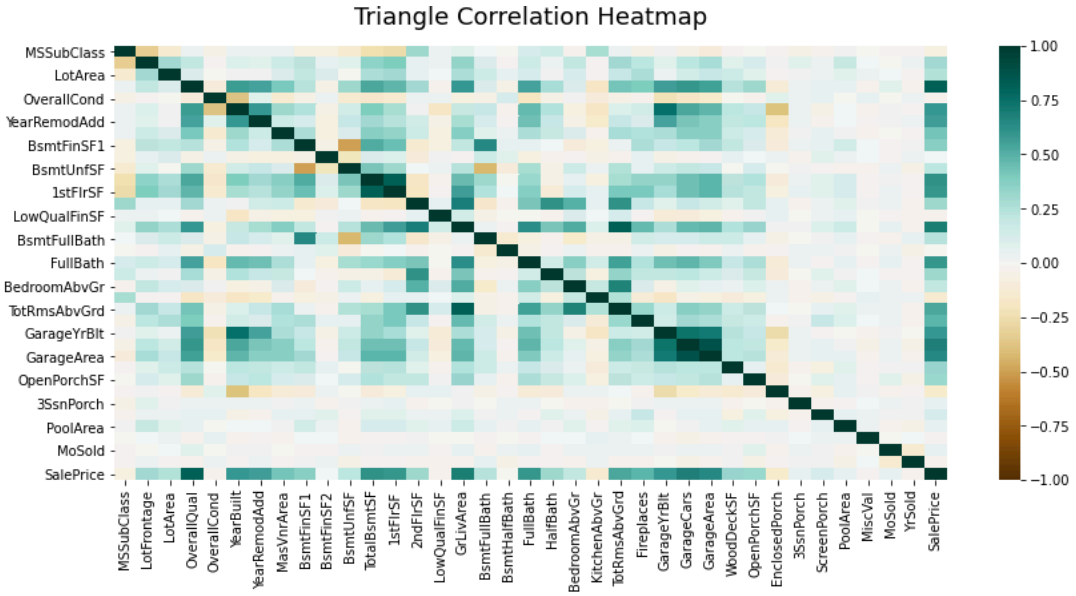
	Feature	number_of_missing	Percentage_of_missing_values
30	PoolQC	2909	85.558824
32	MiscFeature	2814	82.764706
2	Alley	2721	80.029412
31	Fence	2348	69.058824
22	FireplaceQu	1420	41.764706
1	LotFrontage	486	14.294118
25	GarageFinish	159	4.676471
28	GarageQual	159	4.676471
29	GarageCond	159	4.676471
24	GarageYrBlt	159	4.676471
23	GarageType	157	4.617647
10	BsmtExposure	82	2.411765
9	BsmtCond	82	2.411765
8	BsmtQual	81	2.382353
13	BsmtFinType2	80	2.352941
11	BsmtFinType1	79	2.323529
6	MasVnrType	24	0.705882
7	MasVnrArea	23	0.676471
0	MSZoning	4	0.117647
18	BsmtFullBath	2	0.058824
19	BsmtHalfBath	2	0.058824
21	Functional	2	0.058824
3	Utilities	2	0.058824
27	GarageArea	1	0.029412
26	GarageCars	1	0.029412
17	Electrical	1	0.029412
20	KitchenQual	1	0.029412
16	TotalBsmtSF	1	0.029412
15	BsmtUnfSF	1	0.029412
14	BsmtFinSF2	1	0.029412
12	BsmtFinSF1	1	0.029412
5	Exterior2nd	1	0.029412

- Impute categorical features: After walking through the data description, I find out the reason for these missing information is because some house don't have this specific feature. Thus, I create a new category called "None" to capture them.
- Impute numerical features: Similarly, some missing numerical features is due to the fact that the house doesn't have this feature. For example, some house don't have garage and thus some garage related numerical features are missing. Thus, following a similar fashion, I impute these kind of missing information by imputing a zero value. Other than that, there is feature called "Lot Frontage", which is highly correlated with lot area. Thus I used the feature "Lot area" to extrapolate "Lot frontage" by fitted them into a regression line.

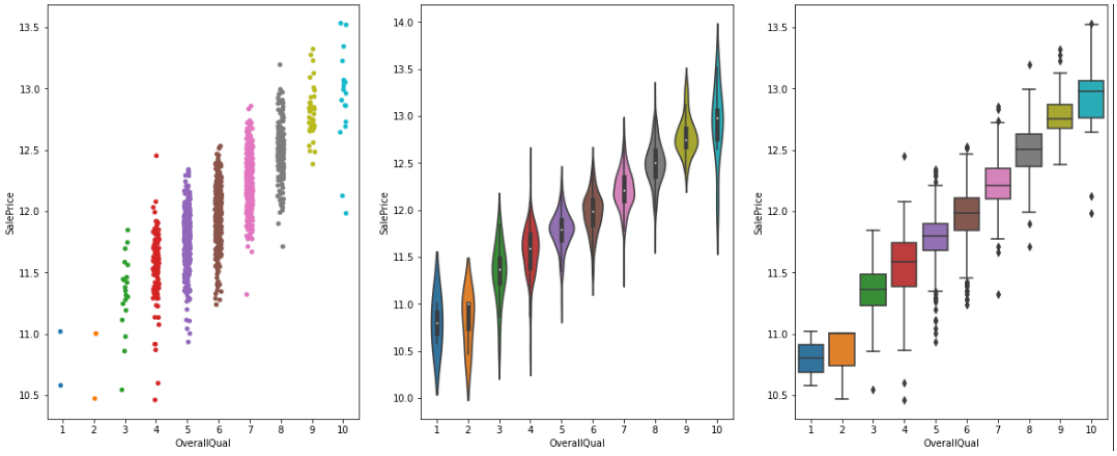
Correlations:

- In general, there are some predictors are highly correlated such that I need to either transform them or delete some of them in order to avoid multilinearity. More will be discussed in data transformation section.

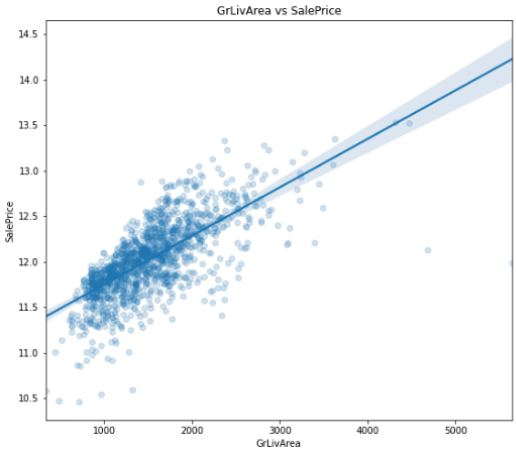
- Year remodel and Year built are highly correlated. I guess this is because many of the houses haven't been remodeled ever since they were built.
- Basement unfinished area and 1st floor area are highly correlated. This makes sense since in most cases, the basement area and 1st floor area are the same
- All garage related features are highly correlated. This may suggest to either combine them together or drop the duplicated features.

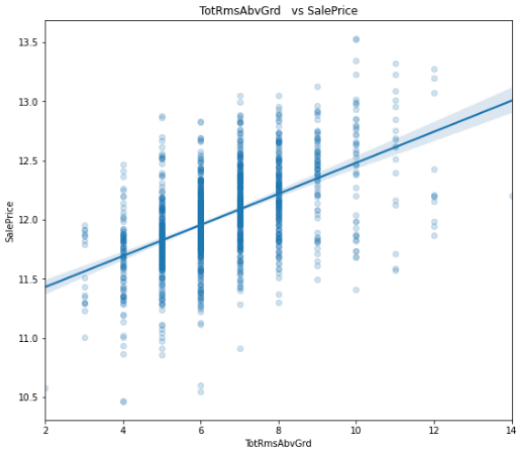


- Deeper dive of the relationship between the top important features vs. target variable:
  - "Overall Qual" vs. sales price: It is pretty obvious that overall quality and sales price have a strong positive relationship.

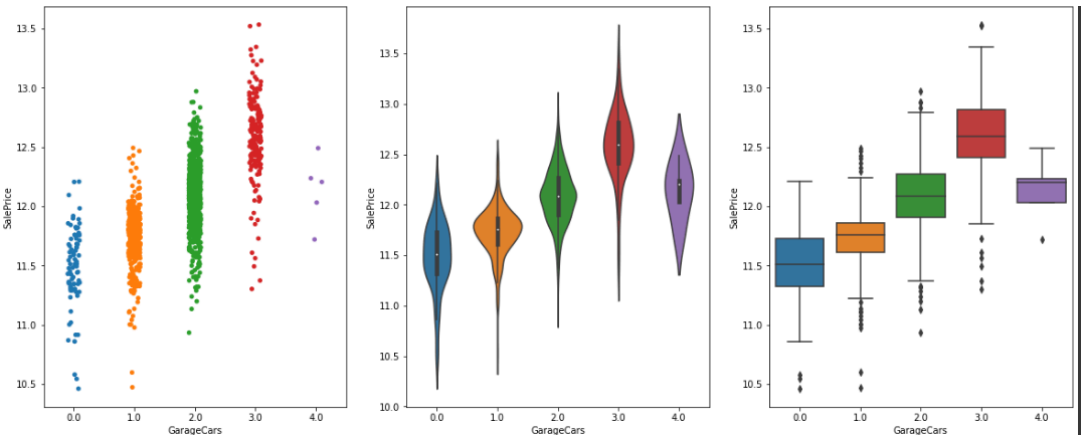


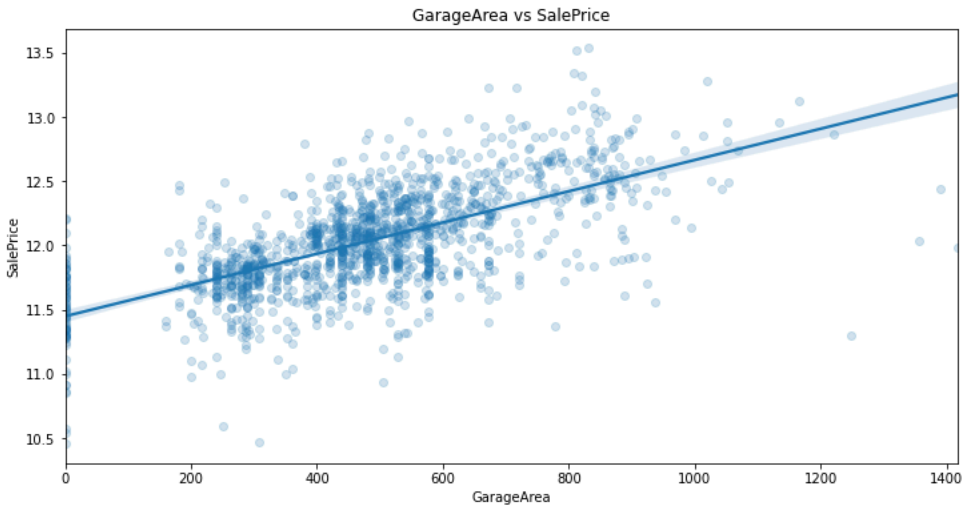
  - "Ground live area" and "Total rooms above ground" vs. sales price: Both of them have a strong positive relationship with sales price. However, they are also correlated with each other, which suggests I need to combine them or drop one of them to avoid multilinearity. More in data transformation section.





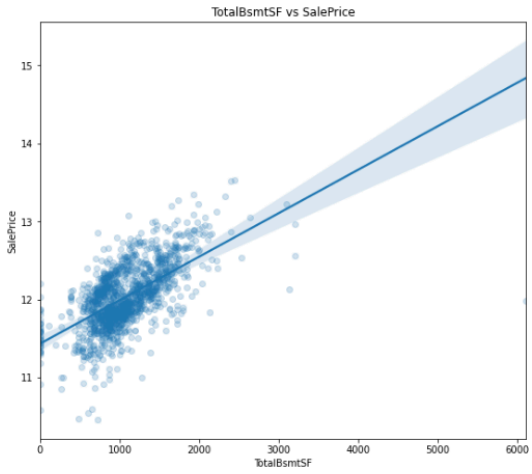
- Garage related features vs. Sales Price: Overall they have a strong relationship with sales price. But one interesting fact is when the garage has four-car capacity, the corresponding house price doesn't seem to rise as much. I guess these may be the houses in the farm.

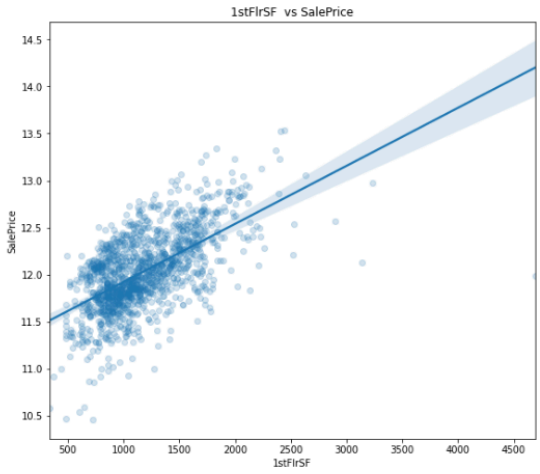




- Space related features(ex.Total basement square feet & 1st Floor square feet) vs. Sales price: Both of them have a strong positive relationship with sales price. However,

they are also correlated with each other, which suggests I need to combine them or drop one of them to avoid multilinearity. More in data transformation section



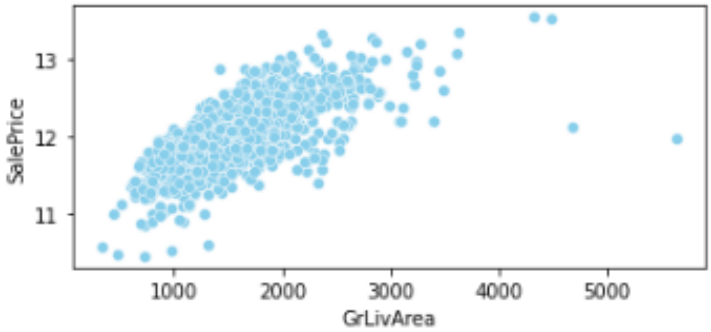


2.3 Data Transformation

- Based on the correlation analysis above and data description, I combine some of the useful features into one and drop some redundant features
- New features:
  - Space related features:
    - Combine above ground rooms into one variable called “AbvGrdRms”
    - Combine all living area into one variable called “All_Liv_SF”
    - Combine all basement bathrooms into one variable called “BsmtBath”
  - Garage related features:
    - Calculate “Area_per_car” using garage related features
  - Year related features:
    - Calculate elapse time for year related features
- Drop features:
  - Space related features: Total rooms above ground, 1st floor square feet, 2nd floor square feet, Basement full bath, Basement half bath, Total basement square feet
  - Garage related features: Garage cars
  - Year related features: Year built, Year remodeled, Garage Year built, Year sold

2.4 Outliers

The two values with bigger “GrLivArea” seem strange and they are not following the crowd. I guess maybe they refer to agricultural area and that could explain the low price. But since most of the houses in the data set are not this case. Thus, I delete these two points.



2.4 Data Preprocessing Pipeline

For categorical features, I use one hot encoder to encode the 10 most frequent categories of each categorical variable. Given the different scales of the continuous predictor variables and most of them do not follow a normal distribution, I normalize them by standard scaler. Considering relatively small size of the training data set, I use cross validate to assess the model performance.

3. Models

3.1 Evaluation metrics

Since the models are regression-based, I choose RMSE as my main evaluation metric.

3.2 Baseline models

I create three baseline regression models (See table for a detailed summary) with all of them predict the log house price. Based on the performance table, it is obvious that the linear regression model suffers from overfit as the sample size is not big, but there are a lot features with many of them as noise.

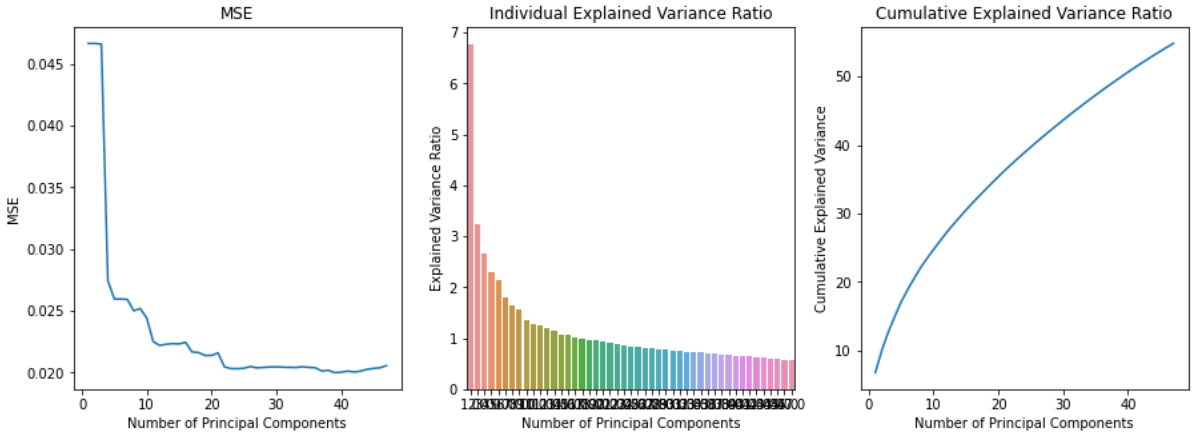
1. The first baseline model is a linear regression model with cross validate grid search on number of predictors.
2. The second baseline model is a Ridge regression model with L2 penalty with cross validate grid search on alpha. The best performed one is when alpha is equal to 40, which suggests there are a lot noise features.
3. The third baseline model is a Lasso regression model with L1 penalty with cross validate grid search on alpha. The best performed one is when alpha is around 0.0028.

Models	Cross-validate RMSE	Degree of penalization
Linear Regression	0.3418	
Ridge Regression	0.1233	40
Lasso Regression	0.1161	0.0028

3.3 Other Regression-based models

I then fit the data with Principal Component Regression. In order to find the number of principal components, I plot the MSE, individual explained variance ratio and cumulative explained

variance ratio. I fit the data using 37 principal components, which improves the performance than Linear regression.



To make the features more explainable than Principal Components, I then fit the data by Partial Least Squared Regression and grid search on the number of components. This gives me a better performance than Principal Component.

Models	Cross-validate RMSE	# Components
Principal Component Regression	0.1424	37
Partial Least Squared Regression	0.1246	7

3.4 Tree-based models

3.4.1 Decision Tree Regression

I fit the data with Decision Tree Regression and find the best *max_depth* for the tree based on 5-fold cross validation. The best model chooses a *max_depth* = 10, which is not so complex to overfit while also includes a satisfying number of features to avoid underfitting. The cross-validated RMSE is worse for Decision Tree than for Regression-based model (except Linear Regression). This might be the result of underfitting, as I built a really basic tree which may fail to fir the underlying complexities in the predictor-response. Enhancement will be made in other tree-based models in later sections.

3.4.2 Random Forest Regression

I pick Random Forest Regression since it tends to reduce the impact of the strongest feature through decorrelation of trees. Random Forest is pretty robust to overfitting in that the model randomly chooses one feature from sqrt(number_of_features) for every split in every tree. Thus I construct deep trees and ensemble them into a forest. By random search, the best model chooses a *max_depth* = 20, which is way deeper than the optimal depth *max_depth* = 10 in Decision Tree Classifier. Considering the run time, I fit the model with only 50 trees. Finally, I use cross validation for hyperparameter tuning and assess model performance.

3.4.3 Gradient Boost Regression

Continue my journey by fitting my model with Gradient Boosted Regression with a default learning rate of 0.1. As Gradient Boost Regression is more powerful algorithm in that it can learn to boost the performance by learning the previous tree. Thus, I have a better cross validate RMSE than the two tree-based regressions mentioned above with the optimal *max_depth* = 5. Again, considering the run time, I fit the model with only 50 trees.

3.4.4 XGBoost Regression

Continue my journey by fitting my model with XGBoost Regression which improves model generalization by including a regularization term. In this case, by 5-fold cross validation, the best degree of penalty is choose when *alpha* = 0.4. The optimal *max_depth* and learning rate is identical as Gradient Boost Regression but gives a slightly better performance.

3.5 Model Selection

Based on cross-validate RMSE, the best model is Lasso Regression with the cross-validate RSME of 0.1161. There is not much of an improvement using Tree-based models compared with regression-based models. One hypothesis is that the relationship between the response and the predictors is close to linear, which is why Lasso Regression achieves solid performance. Another hypothesis is that some key features could explain much of the variance of the model, so a simple model like Lasso regression with regularization term is sufficient. Thus to make thing simple, I choose Lasso Regression as my final model.

Models	Cross-validate RMSE
Linear Regression	0.3418
Ridge Regression	0.1233
Lasso Regression	0.1161
Principal Component Regression	0.1424
Partial Least Squared Regression	0.1246
Decision Tree Regression	0.1832
Random Forest Regression	0.1398
Gradient Boost Regression	0.1294
XGBoost Regression	0.1292

4. Feature Interpretation for the Best Model

More are coming soon