Machine Learning Engineer Nanodegree

Capstone ProjectHouse Prices Prediction

Hesham Shabana December 18th, 2016

I. Definition Project Overview

House prices is a topic that always attract a lot of attention due to the daily demand and the wide range of interested party's investors, real estate agents, tax estimators, house owners, and house buyers with this the demand for a reliable model to assist a house price is needed. Traditionally house price prediction does not take into consideration the wide range of available parameters and it also assume independence between these parameters which does not hold in practice.

In Egypt buying a house is very important decision especially for young people who are looking to start a family and this decision becomes even harder with this increase in population, Egypt population estimated to be 93,383,574 with almost 2% growth in the last 4 years, and there is very little research in this area as well as a lack of data. Not to mention that people are following only one rule what is the square feet price in specific area? Therefore, using a machine learning algorithm to learn from the past purchasing history will help us to determine and predict the price of the house taking into consideration the attributes that matter the most and this will support any new buyer or a seller to set the right expectation.

The dataset describing the sales for houses in Ames, Iowa from 2006 to 2010 which contains 2930 observations and 80 variables, the variable distribution is 23 nominal, 23 ordinals, 14 discrete and 20 continuous.

This project inspired by: https://www.kaggle.com/c/house-prices-advanced-regression-techniques

Problem Statement

The goal is to analyze historical sales of house prices features and through feature selection, feature engineering, machine learning finds the most relevant set of features that affect the price and which will allow us to perform an accurate prediction for new houses.

To avoid curse of dimensionality given the large number of features included in our database (80 feature) first we need to reduce our vector space by applying feature selection methods, as well as feature transformation. The goal here is to reduce our input space as much as possible to avoid overfitting and to achieve better prediction results.

Workflow

1. ***Data Exploration:*** Imputation of missing values Understand the dataFeature analysis and visualization
2. ***Data Preprocessing:*** Feature ScalingFeature Scaling for continues variables so they can be compared on compound ground, scale down all the features between 0 and 1. Feature NormalizationFeature Normalization: rescale features so they have the properties of normal distribution, as PCA may perform better after standardization. One-Hot Encoding for categorical features Split the data to training and validation sets
3. ***Feature selection/transformation:*** Feature selection methods (SelectPercentile, SelectKBest, Lasso regression) Feature transformation (PCA algorithm)Feature engineering
4. ***Training:*** Train our model using the below algorithms: Regularized regression Boosted Random forest regressor
5. ***Model tuning:*** Tune our model parameters by using GridSearchCV
6. ***Evaluation:*** Evaluate the model performance using RMSE
7. ***Result: Apply*** our model to the testing data

Metrics

To assists the model performance, the below matric will be used:

Root Mean squared error: Measures the root average of the squares error, it is the difference between the estimator and what is estimated variable

The Mean Squared Error (MSE) is a measure of how close a fitted line is to data points.

For every data point, you take the distance vertically from the point to the corresponding y value on the curve fit (the error), and square the value. Then you add up all those values for all data points, and divide by the number of points minus two. The squaring is done so negative values do not cancel positive values also it amplifies the impact of errors. The smaller the Mean Squared Error, the closer the fit is to the data. The MSE has the units squared of whatever is plotted on the vertical axis.

Another quantity that we calculate is the Root Mean Squared Error (**RMSE**) The one that we are using in this project. It is just the square root of the mean square error it represent the distance on average of a data point from the fitted line.

The RMSE is directly interpretable in terms of measurement units, and so is a better measure of goodness of fit than a correlation coefficient. One can compare the RMSE to observed variation in measurements of a typical point.

Therefore, RMSE is a perfect choice for our problem.

Reference: https://www.vernier.com/til/1014/

II. AnalysisData Exploration

This dataset describing the sales for houses in Ames, Iowa from 2006 to 2010 which contains 2930 observations.

Understand the data

The dataset contains large number of features 80 in total, the feature distribution is 23 nominal, 23 ordinals, 14 discrete and 20 continuous, there are 1460 observations in this database for training and 1459 for testing.

Every dataset has some null or missing value as well as our dataset, below are the features that has null/missing values and their count and how we will deal with each case.

LotFrontage feature has 259 Null value, LotFrontage represent linear feet of street connected to property and it's a numerical feature. Null value does not mean that the property is not connected to a street, therefore, replacing the Null values for this feature with the mean will give us reasonable approximation.

Alley A categorical feature that represent type of alley access to property, it has 1369 Null values, the Null in this feature represent No alley access, therefore, this is not considered as a Null value and we could leave it and consider the NA as a 3rd category along with Gravel and Paved or we could replace it with no_alley_access explicitly.

MasVnrType A categorical feature that represent masonry veneer type, the null here means the absence of this value and ***MasVnrArea*** is a numerical feature that represent Masonry veneer area in square feet if it has a type, both features are related and have only 8 Null values, therefore, setting the type to No and the Area to be zero.

BsmtQual, BsmtCond, BsmtExposure, BsmtFinType1, BsmtFinType2 are categorical features describe the basement and the Na is a category value that represent No Basement therefore, we could replace both features Na values with no_basement

Electrical A categorical feature that represent Electrical system and has only 1 null value, therefore, we will drop this row.

FireplaceQu A categorical feature describe Fireplace quality and the Na is a category value that represent No Fireplace, therefore, we could replace both features Na values with no_fireplace

GarageType, GarageYrBlt, GarageFinish, GarageQual, GarageCond are categorical features describe the garage and the Na is a category value that represent No Garage therefore, we could replace both features Na values with no_garage, however, we will leave ***GarageYrBlt*** feature to be Null as it's a numerical feature that has no value when there is no garage, however, we will set this value to be same as YearBuilt of the house.

PoolQC A categorical feature describe Pool quality and the Na is a category value that represent No Pool, therefore, we could replace both features Na values with no_pool

Fence A categorical feature describe Fence quality and the Na is a category value that represent No Fence, therefore, we could replace both features Na values with no_fence

MiscFeature A categorical feature describe Miscellaneous feature not covered in other categories and the Na is a category value that represent None, therefore, we could replace both features Na values with None

Analyze target variable (House price)

House prices distribution

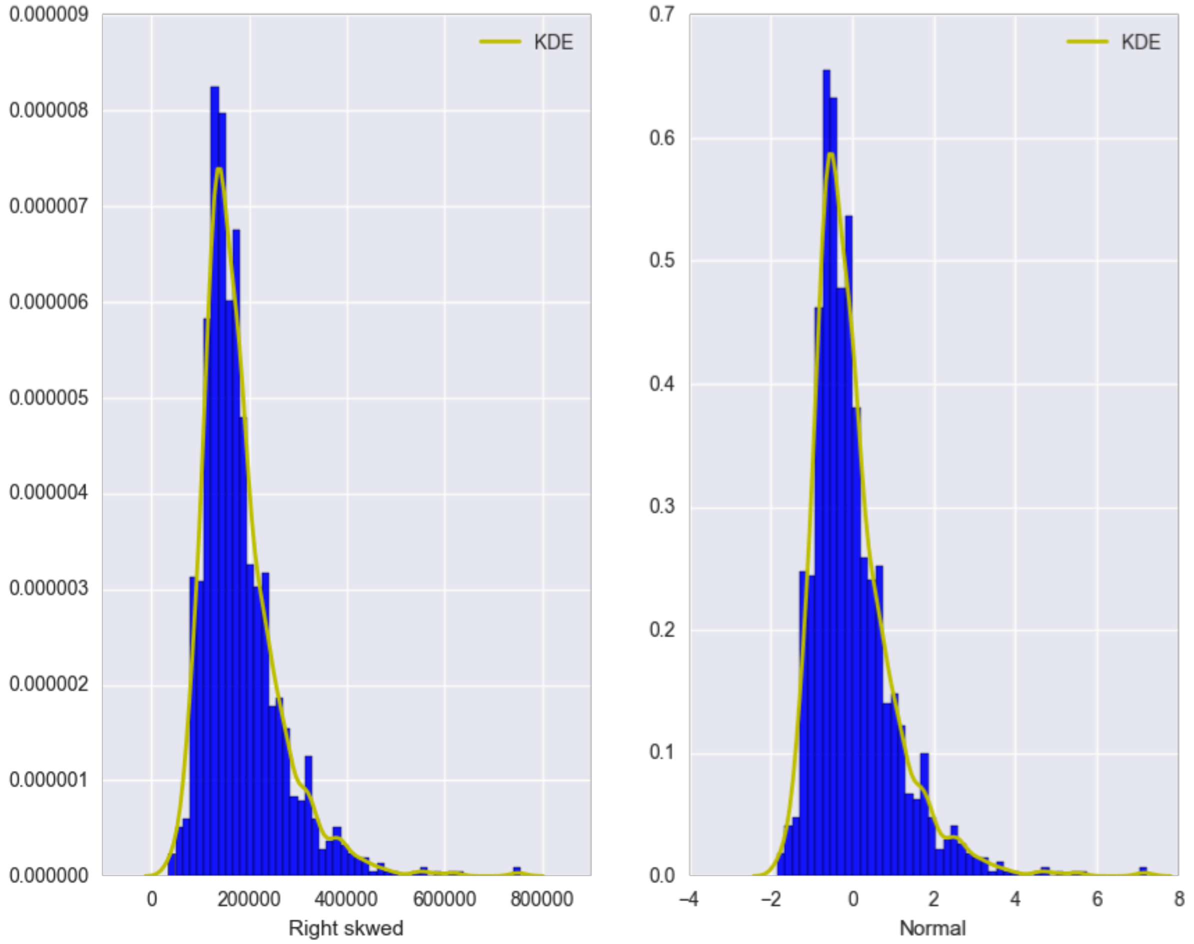


Figure 1: House price distribution transformation

Many statistical tests and intervals are based on the assumption of normality.

The assumption of normality often leads to tests that are simple, mathematically tractable, and powerful compared to tests that do not make the normality assumption.

Unfortunately, our dataset is in fact not approximately normal.

However, an appropriate transformation of a data set can often yield a data set that does follow approximately a normal distribution.

This increases the applicability and usefulness of statistical techniques based on the normality assumption.

Exploratory Visualization

Reference: https://www.kaggle.com/xchmiao/house-prices-advanced-regression-techniques/detailed-data-exploration-in-python

Numerical features

Given that the independent variables are dichotomous or continuous and the dependent variable (house price) is continuous, therefore, we will conduct a correlation test to answer the below question.

How strongly and in what direction (i.e., +, -) are the independent variables and dependent variable related?

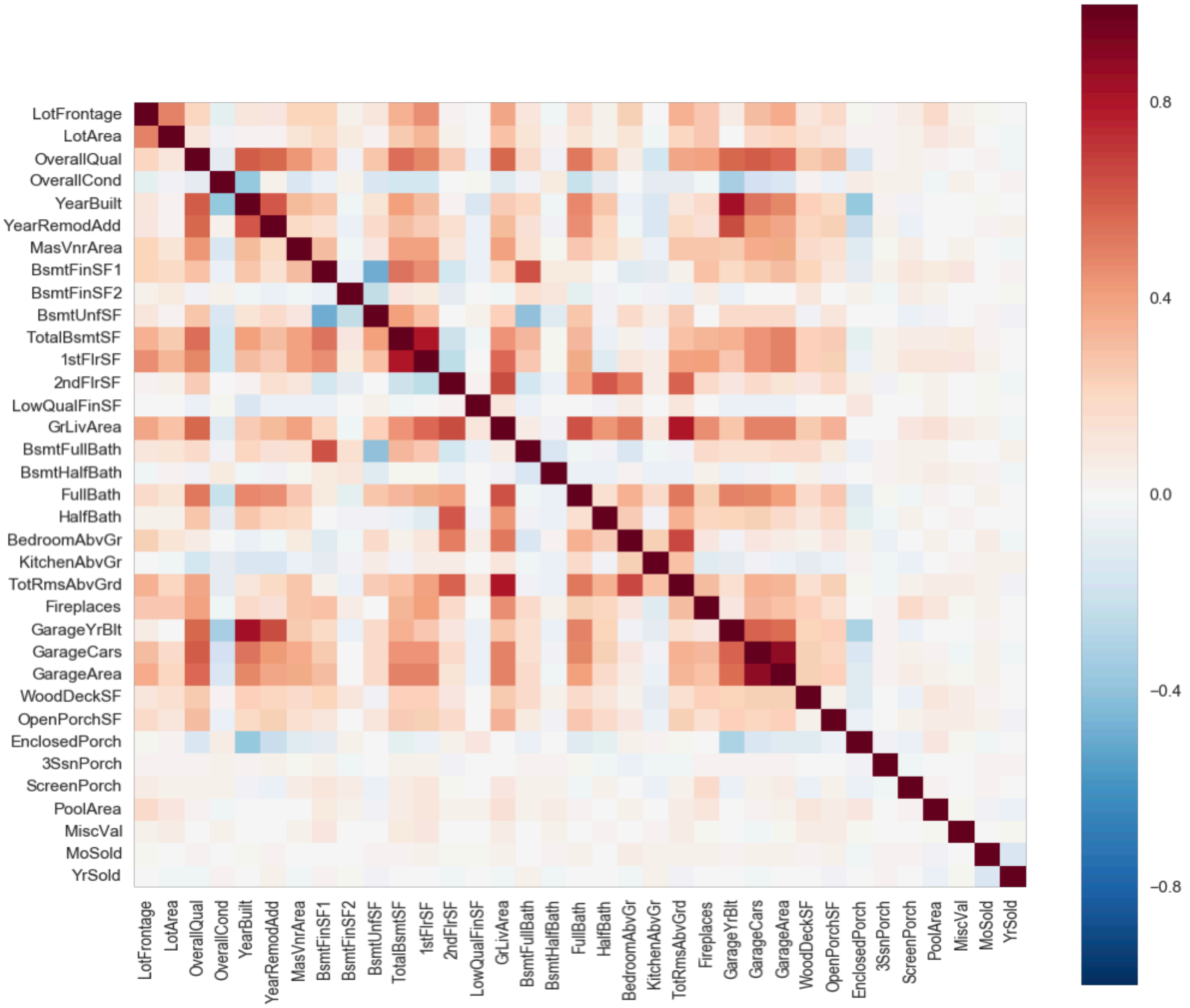


Figure 2: Numerical variables correlation

List of all numerical features ordered by their correlation with Sale Price

After measuring the correlation between each feature and the sale price, the below table shows that the housing price correlates strongly with *OverallQual, GrLivArea, GarageArea, TotalBsmtSF, 1stFlrSF, FullBath, TotRmsAbvGrd, YearBuilt, YearRemodAdd, GarageYrBlt, MasVnrArea and Fireplaces.* However, some of those features are highly correlated among each other's, which could be a valuable information that may help to reduce dataset dimension since we could find a two variable that are highly correlated with each other and therefore using only one of them will be sufficient.

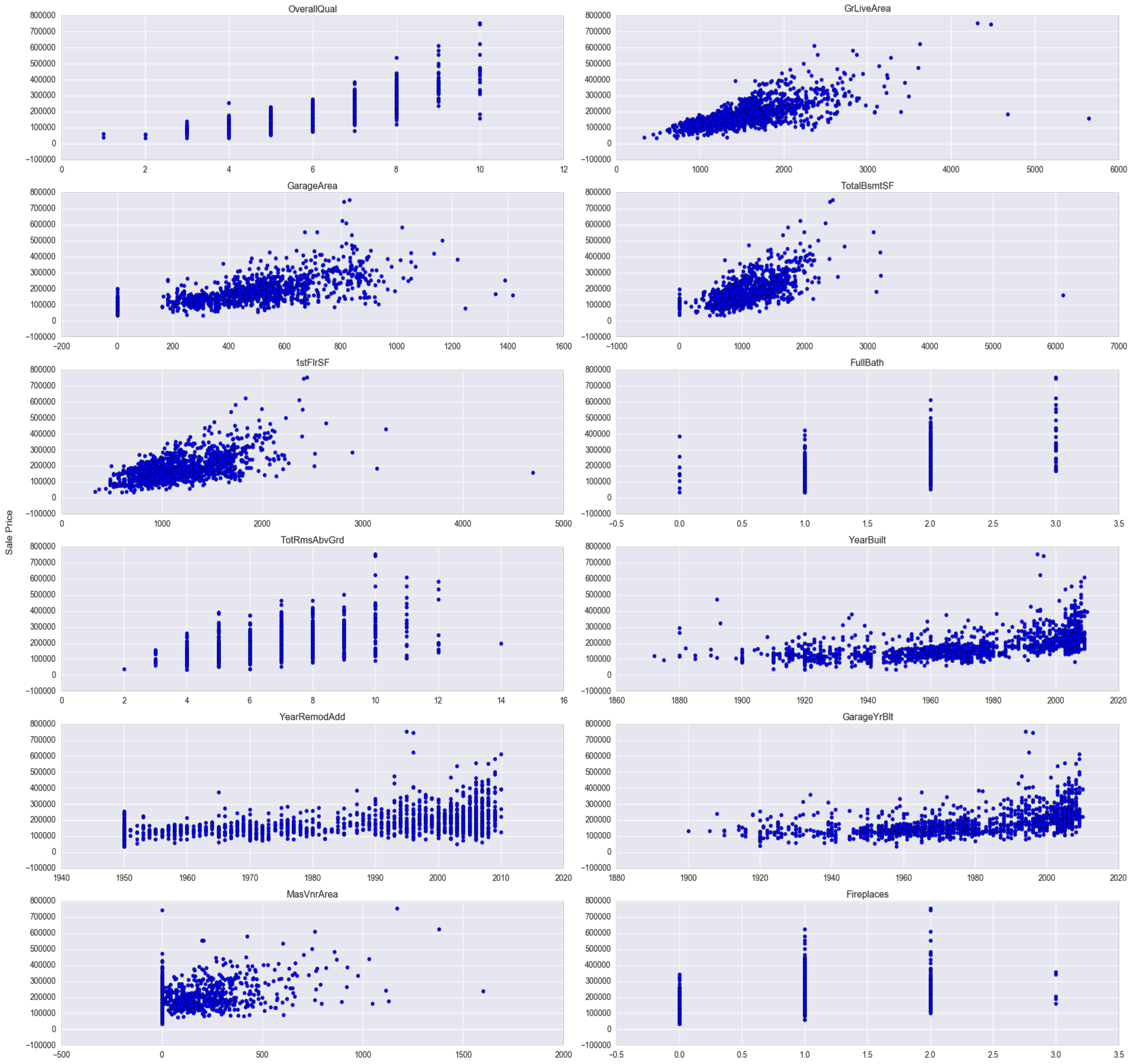


Figure 3: Strongly correlated numerical features correlation with Sale Price

Categorical Features

Given that the independent variables categorical and the dependent variable (house price) is continuous, therefore, we will try to answer the below question.

Do differences exist between 2 or more groups on one dependent variable?

Neighborhood

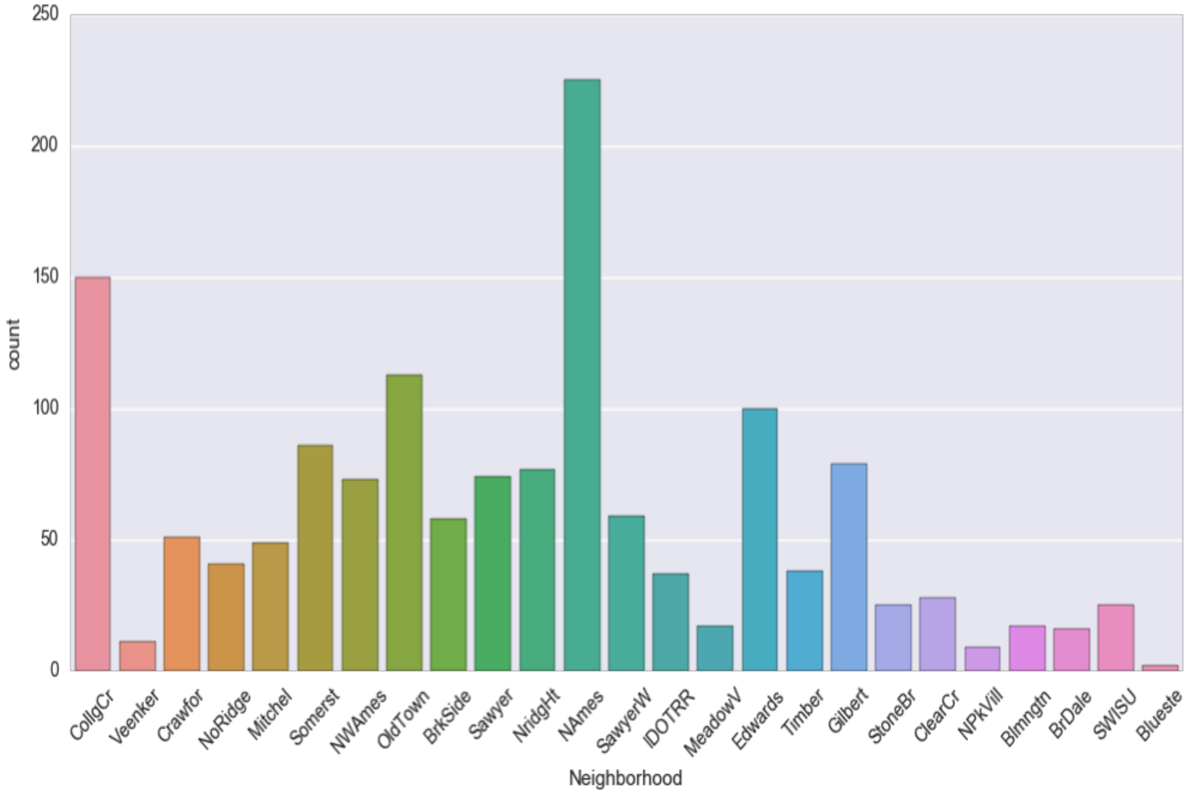


Figure 4: Neighborhood.

As we can see in *Figure 4* some groups are very different and some have some similarity and we could add similar groups into a same bucket for dimension-reduction.

House Sale

Sale Type

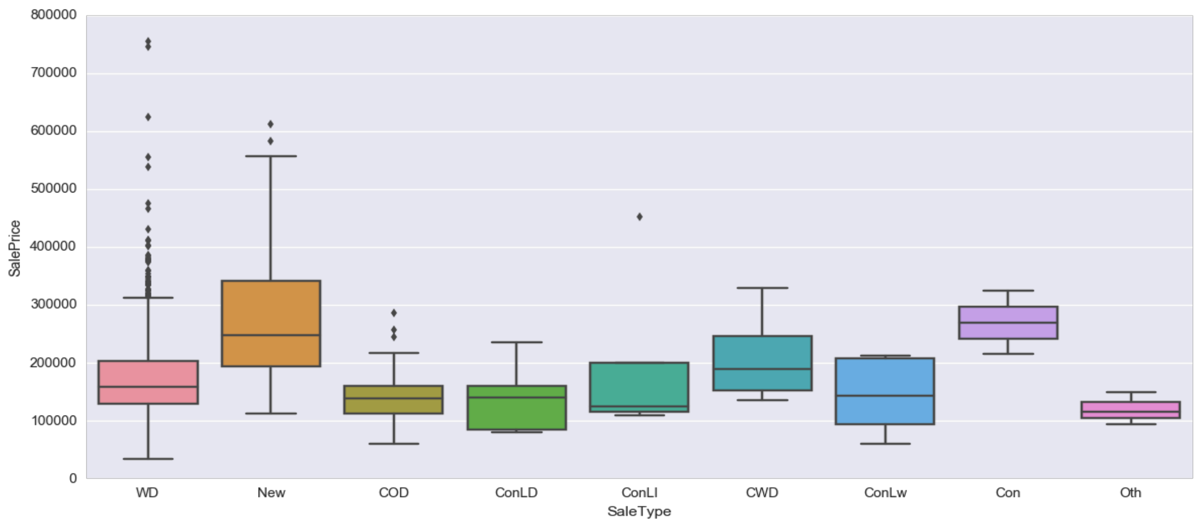


Figure 5: Sale Type.

From *Figure 5* we could indicate that the new houses are the most expensive, another observation is that all sales that offer low down payment, low interest like ConLw, ConLi, ConLD are very similar and could be grouped same for sales with warranty like WD, CWD, VWD also we could create a new feature that has New, Low Down payment, Warranty, Other which will be more representative.

Sale Condition

Condition of sale

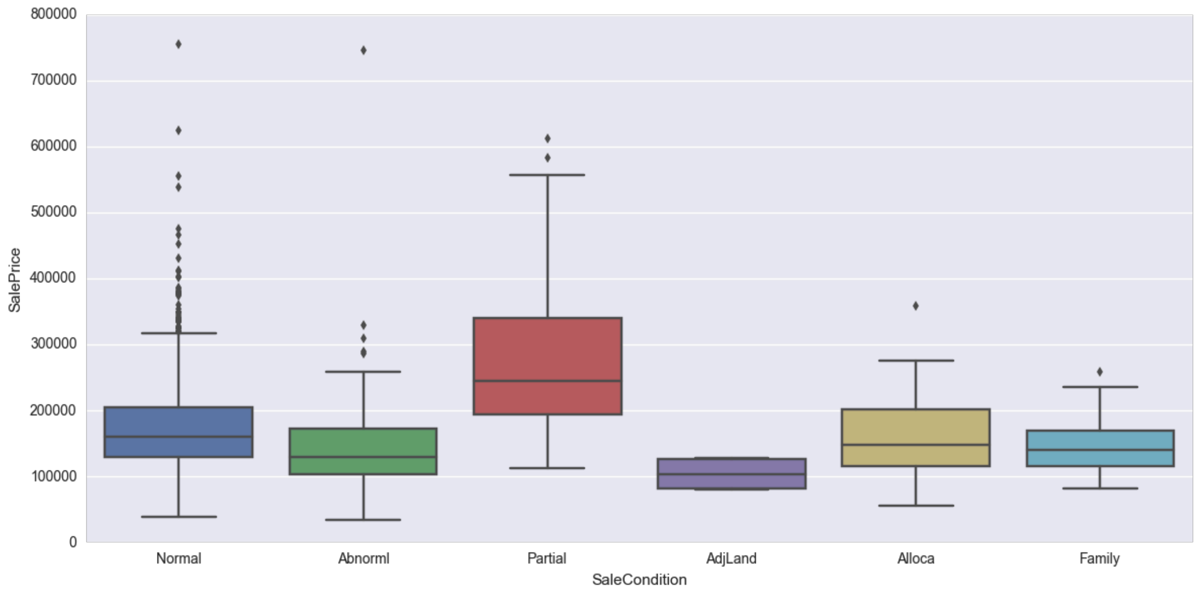


Figure 6: Condition of the sale.

Figure 6 above Indicate that the partial finished (Home was not completed when last assessed) has the heighted prices it could indicate that people prefer to buy houses that was never occupied before, Normal and Alloca are similar, Abnormal and Family have also the same sale price rate and could be grouped together.

Year and Month of the Sale

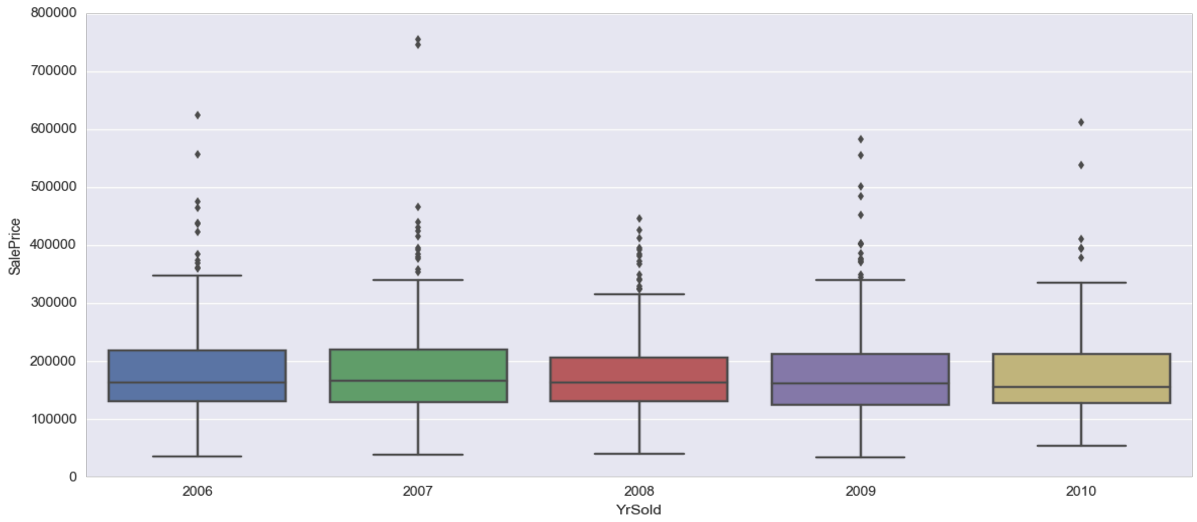


Figure 7: Year of the sale with house price.

Figure 7 above Indicate that the year of the sale doesn't impact on the sale price.

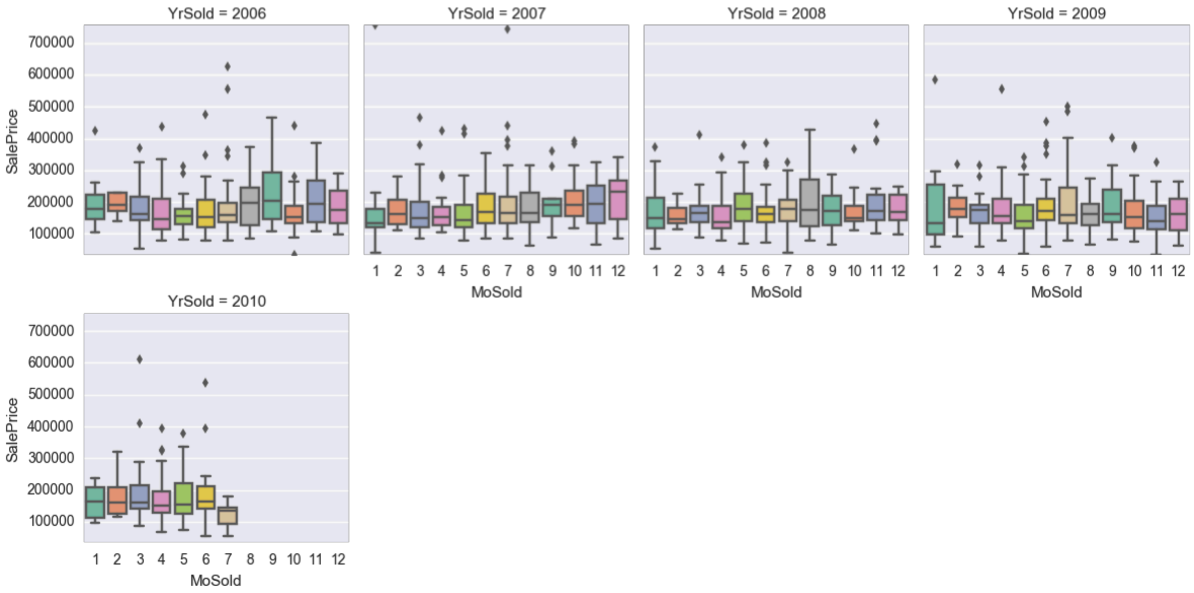


Figure 8: Month of the sale vs sale price.

Figure 8 above Indicate that year and month of the sale doesn't impact the price much.

Housing Style

Type of dwelling

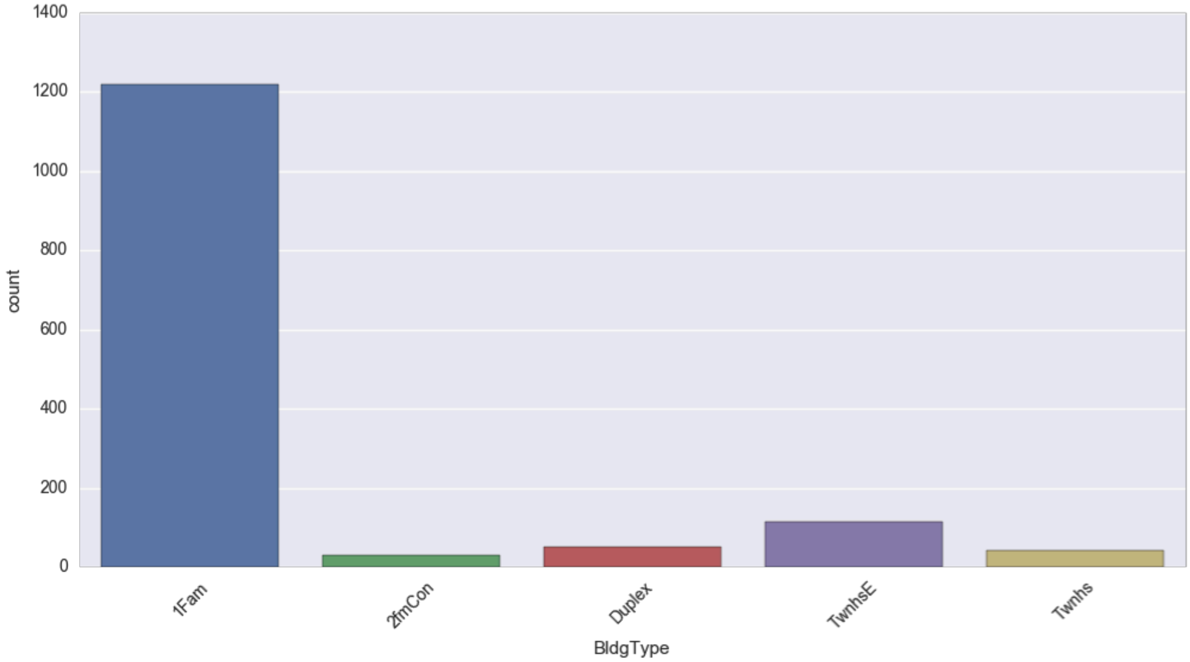


Figure 9: Type of dwelling.

From *Figure 9* we could conclude that single-family detached (1Fam) dwelling type is the most sold type and the rest of groups are not sold that much.

Style of dwelling

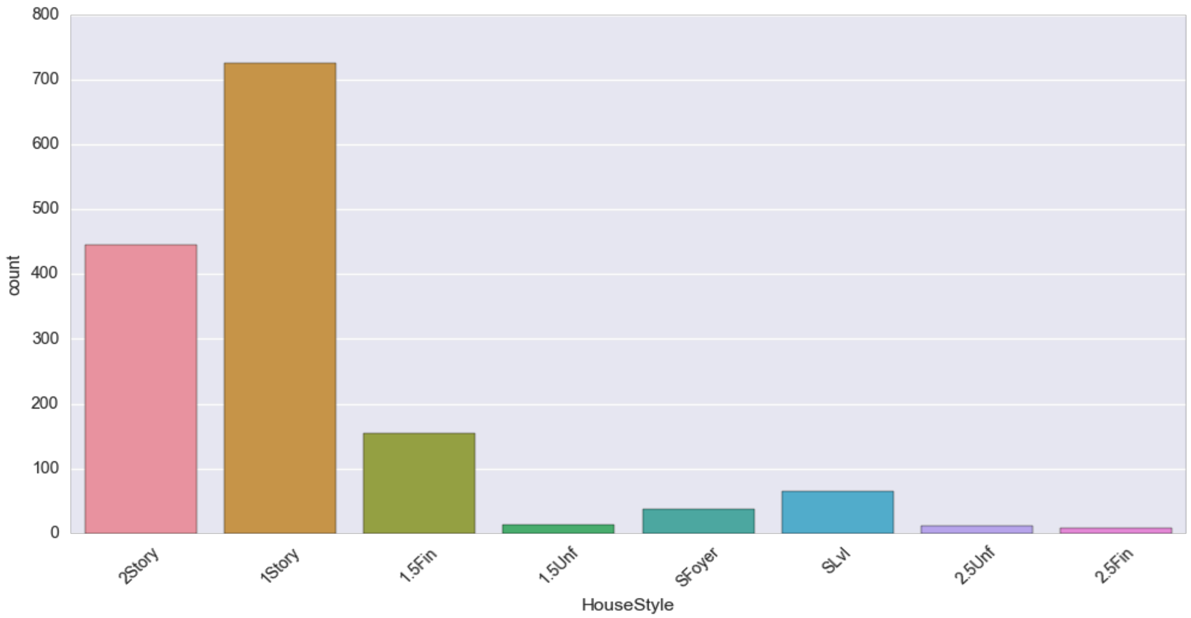


Figure 10: House style.

The above figure Indicate that 1story and 2story are the most type sold, 1.5Unf, 2.5Unf, 2.5Fin have a low sale rate.

Housing Condition

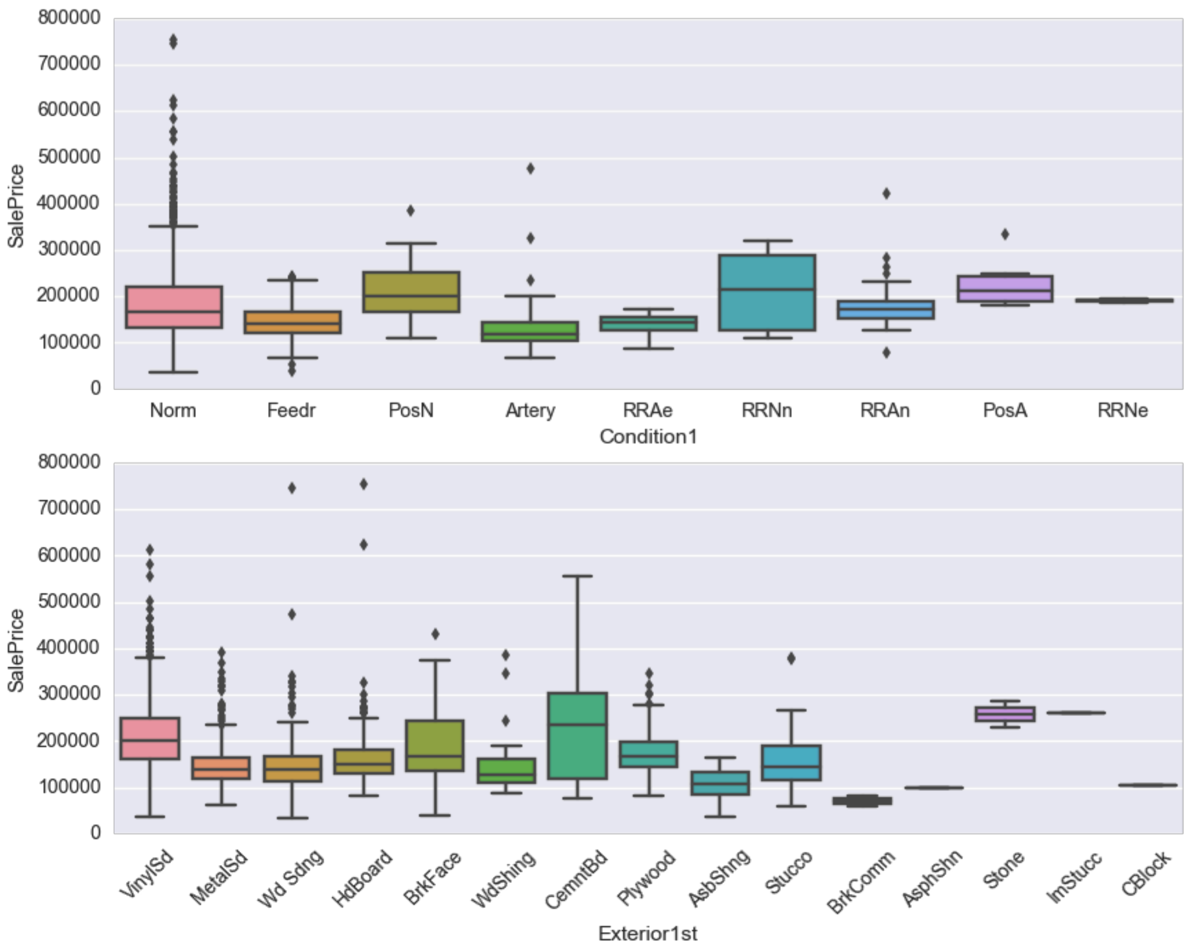


Figure 11: House condition.

Figure 11 above shows that house condition contributes a lot in the sale price and there is a large variance between the groups, for example, having a Cement Board will increase the sale price Hard Board … etc.

Basement Conditions

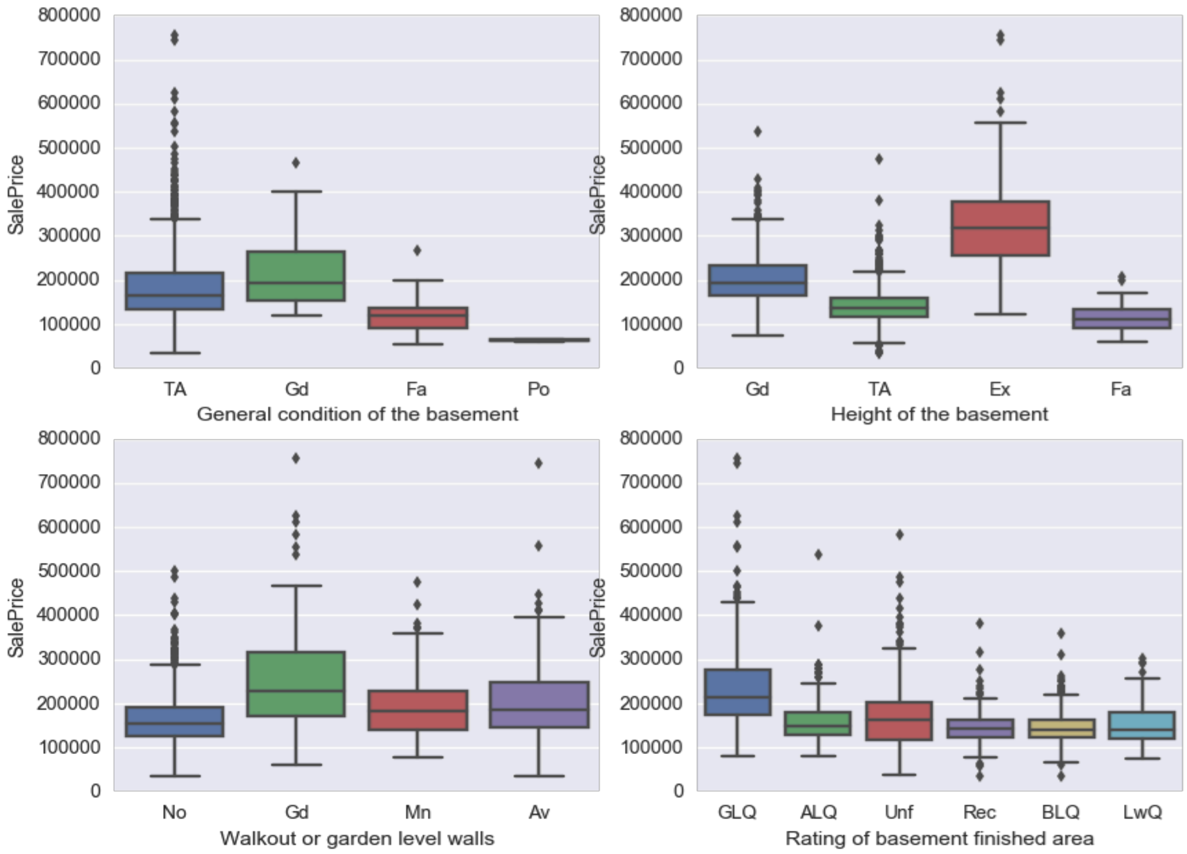


Figure 12: Basement.

Indicates that having no basement or having a basement with poor condition decrease the house price

Fireplace quality

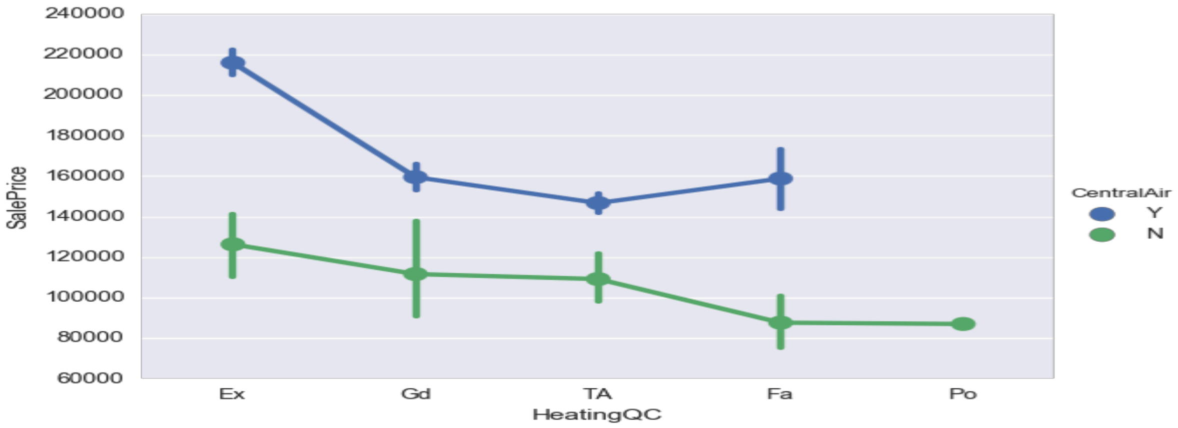


Figure 13: fire place.

Having a fireplace/heating is a feature that increase the house price, and having none decrease house sale price, apparently Ames city is a cold place.

Kitchen quality

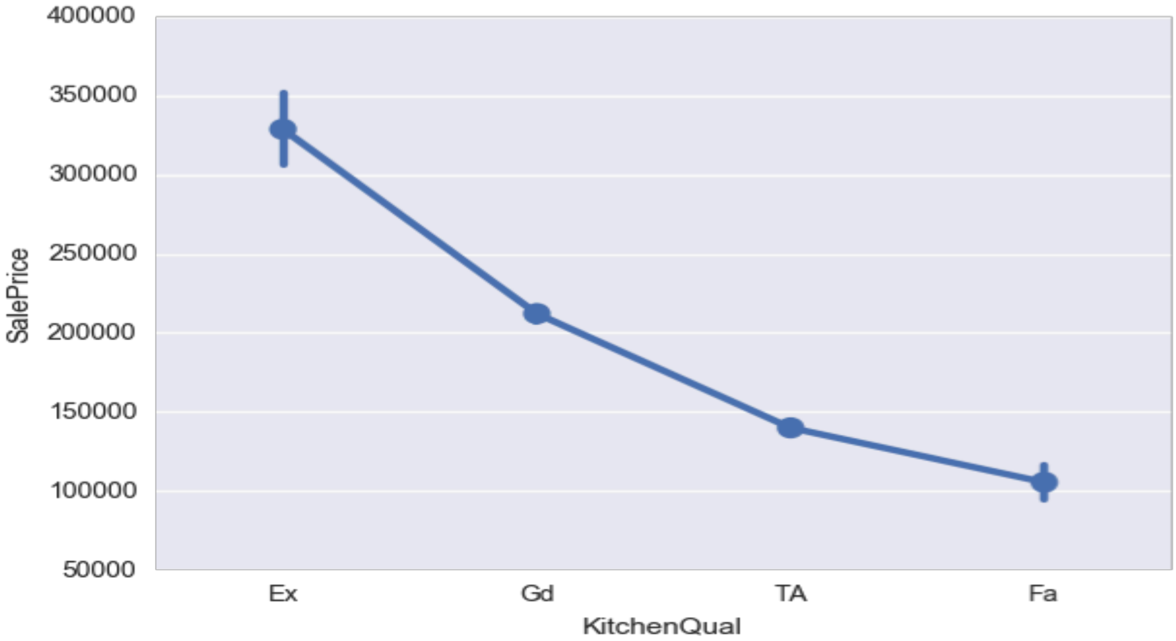


Figure 14: kitchen

Indicate that having an excellent kitchen condition has a great impact on the sale price.

Street & Alley Access

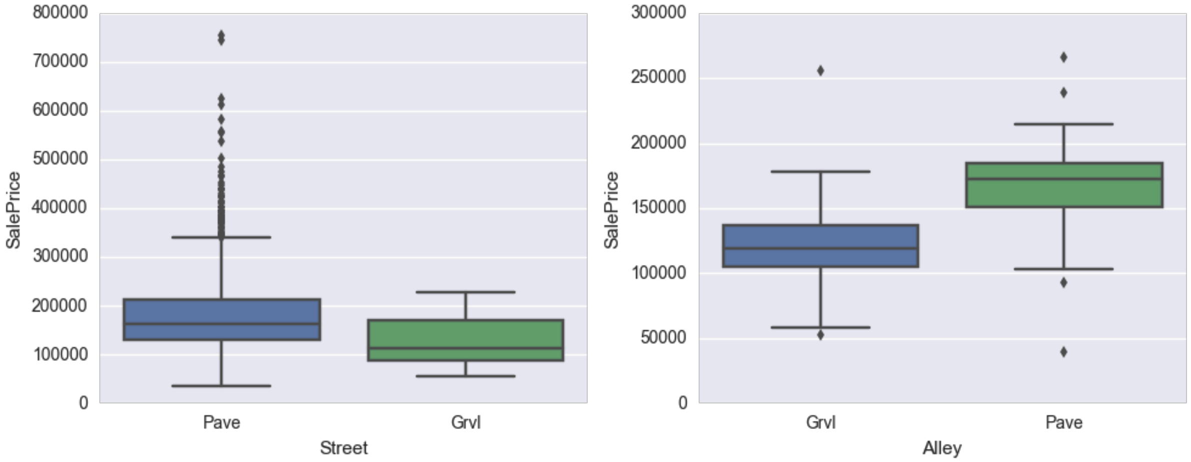


Figure 15: Street and Ally access.

Indicate that Ally access have an impact on the house price**.**

Algorithms and Techniques

Before we train our model first we need to preprocess our data, starting from dealing with null or missing data, scale our numerical feature to be between 0 and 1 so that comparison performed on the same ground as well as transform the numerical data to be normally distribute, encode the categorical features and finally split out data to training, validation and testing sets.

Feature Scaling: (Xnorm = X – Xmin\ Xmax – Xmin)

It is a method used to standardize the range of features of the data it's usually performed during the data preprocessing step. Feature scaling is very important for continues variables so they can be compared on compound ground also when features being on different scales, certain weights may update faster than others since the feature values play a role in the weight updates.

Feature Normalization: (z = x – u\ sigma)

Many statistical tests are based on the assumption of normality which often leads to tests that are simple mathematically compared to tests that do not make the normality assumption. Therefore, we will identify the skewed features and normalize them.

One-Hot Encoding:

As many machine learning algorithms does not work with string variables we need a way to convert the categorical feature with number, if the categorical features don't have an order information between them replacing categories names with number may give one category unintentionally weight, therefore, in this case it's better to use One-Hot-encoding which will group of bits which the legal combinations of values are only those with a single high (1) bit and all the others low (0).

Features:

As we saw in the data exploration part above this dataset has many features some are relevant to the sale price feature (our target value) and many are not, therefore, to build a model that is able to predict the sale price of new houses we need to reduce our dimension space to avoid curse of dimensionality, overfitting and to get better results, starting from feature selection methods like SelectPercentile, SelectKBest followed with Feature transformation methods like PCA we should be able to select the most relevant features for our model.

Principal component analysis (PCA): Is a method for feature selection that transform a set of correlated features into underlying set of orthogonal variables, PCA finds new coordinate system that's obtained by transformation and rotation only and moves the center of the data by moving the x-axis into the principal axis of variation and the y-axis into the orthogonal axis? PCA try to come up with a direction in the data to project our data to it while losing the minimum amount of information.

The principal component is an unsupervised algorithm that try to make a composite feature (latent features) that probes the underlying phenomenon, which will help against dimensionality reduction.

Simply train a PCA model to find the best component analysis returned and combine them with the above technique as the best set of features.

Univariate feature selection:

Univariate feature selection examines each feature individually to determine the strength of the relationship of the feature with the response variable. These methods are simple to run and understand and are in general particularly good for gaining a better understanding of data. There are lot of different options for univariate selection we will be using:

- **SelectPercentile**: Select features according to a percentile of the highest scores
- **SelectKBest**: Select features according to the k highest scores.

Now we are ready to train our model we will be using multiple algorithms and compare them also as we need to select the best hyper-parameter for each method we will use GridSearchCV or RandomizedSearchCV to tune our model parameters, below are the list of algorithms that we will be using: Linear regression, PCA and Random forest

Finally, we will evaluate our models, select the best model and report the final result on the testing data.

Reference: http://blog.datadive.net/selecting-good-features-part-i-univariate-selection/

Benchmark

"Traditionally house price prediction does not take into consideration the wide range of available parameters." Therefore, our benchmark model could be a linear regression model that predict the price based on some of the traditional parameters like square_feed, num_of_rooms, neighbor.

Create our benchmark dataset

Generate a dataset to include these variables BedroomAbvGr, Neighborhood, LotArea and the target variable SalePrice

	BedroomAbvGr	**Neighborhood**	**LotArea**	**SalePrice**
0	3	CollgCr	8450	208500
1	3	Veenker	9600	181500

Benchmark model results

- `Benchmark RMSE on Training set: 0.245443865925`
- `Benchmark RMSE on testing set: 0.247736170388`

III. Methodology

Data Preprocessing

Spot outliers

Outliers in data could mislead the training process which could result in longer training times, less accurate and poorer results.

Machine learning algorithms like Linear regression or PCA which we will be using are prone to outliers, therefore, we will spot the outliers in the dataset and remove them.

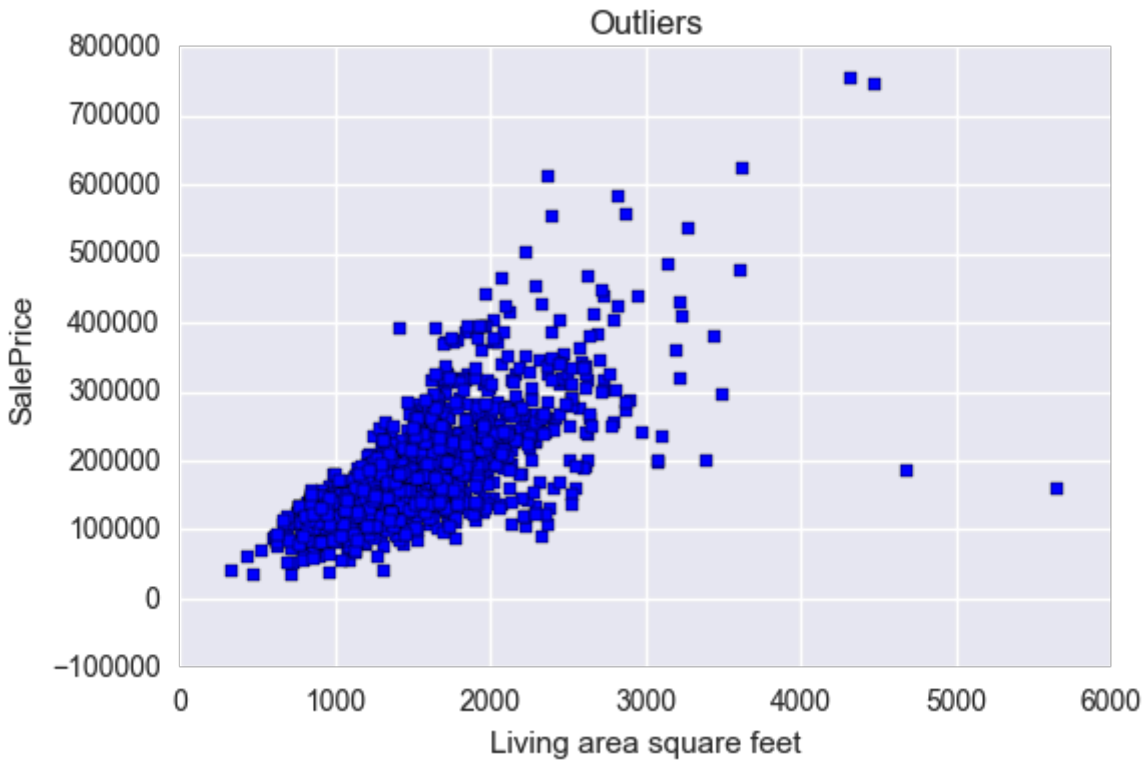


Figure: OutliersThere are some extreme outliers on the dataset, large houses sold for low price and very expensive houses.

The author of the dataset recommends removing *any houses with more than 4000 square feet*.

https://ww2.amstat.org/publications/jse/v19n3/decock.pdf

Handel Null data

Null or missing data points should not be removed or deleted blindly, as it could represent an interesting pattern on the data or it could be a value on its own, therefore, next we will report the features and the number of null values in each one and assist them one by one to determine the best approach that we should follow.

Some numerical features are actually really categories

MSSubClass: Identifies the type of dwelling involved in the sale.

MoSold: Month Sold (MM) Therefore, we will replace there numerical values with categorical ones, for example, the Month Sold value 1 will be Jan ... etc.

Now that we handled all Null/Missing values we will start to analyze our dataset.

Encoding categorical feature

Many machine learning algorithms do not deal with string data, therefore, we need to transform our numerical data.

Manually encode categorical features as ordered numbers when there is information in the order.

One-Hot-Encoding for the rest of categorical features

Feature Engineering

Transforming raw data into features that better represent the underlying problem to the predictive models should result in improved model accuracy on unseen data.

In this part we will manually analyze some features and create new features that could be more representative.

http://machinelearningmastery.com/discover-feature-engineering-how-to-engineer-features-and-how-to-get- good-at-it/

1. Floors square feet Combine the below features to indicate the overall floor square feet. 1stFlrSF: First Floor square feet 2ndFlrSF: Second floor square feet

2. All Living Area square feet Combine the below features to represent the hole living area: 1stFlrSF: First Floor square feet2ndFlrSF: Second floor square feetLowQualFinSF: Low quality finished square feet (all floors) GrLivArea: Above grade (ground) living area square feet

3. Overall Quality Create a new feature that describe the overall house condition OverallQual: Rates the overall material and finish of the house OverallCond: Rates the overall condition of the house

4. Overall garage quality Create a new feature that describe the overall garage of the house condition, GarageQual: Garage quality GarageCond: Garage condition.The new feature *OverallGarageQuality = GarageQual * GarageCond* represent the overall garage condition.

5. Overall exterior quality ExterQual: Evaluates the quality of the material on the exterior ExterCond: Evaluates the present condition of the material on the exterior. The new feature *OverallExteriorQuality = ExterQual * ExterCond* represent the overall quality of material on the exterior.

7. Overall kitchen score Create a new feature that describe the overall kitchen condition/scoreKitchen: Kitchens above grade KitchenQual. The new feature *OverallKitchenScore* represent the overall kitchen score.

8. Overall fireplace score Fireplaces: Number of fireplaces FireplaceQu: Fireplace quality. The new feature *OverallFireplaceScore* represent the overall fireplace score.

9. Overall garage score GarageArea: Size of garage in square feet, GarageQual: Garage quality. The new feature *OverallGarageScore* represent the overall garage score.

10. Overall pool score PoolArea: Pool area in square feet, PoolQC: Pool quality. The new feature *OverallPoolScore* represent the overall pool score.

11. Total number of bathrooms BsmtFullBath: Basement full bathrooms, BsmtHalfBath: Basement half bathrooms FullBath: Full bathrooms above grade HalfBath: Half baths above grade. The new feature *OverallBathroomsNumber = BsmtFullBath + (0.5 BsmtHalfBath) + FullBath + (0.5 * HalfBath)* represent the overall number of bathrooms.

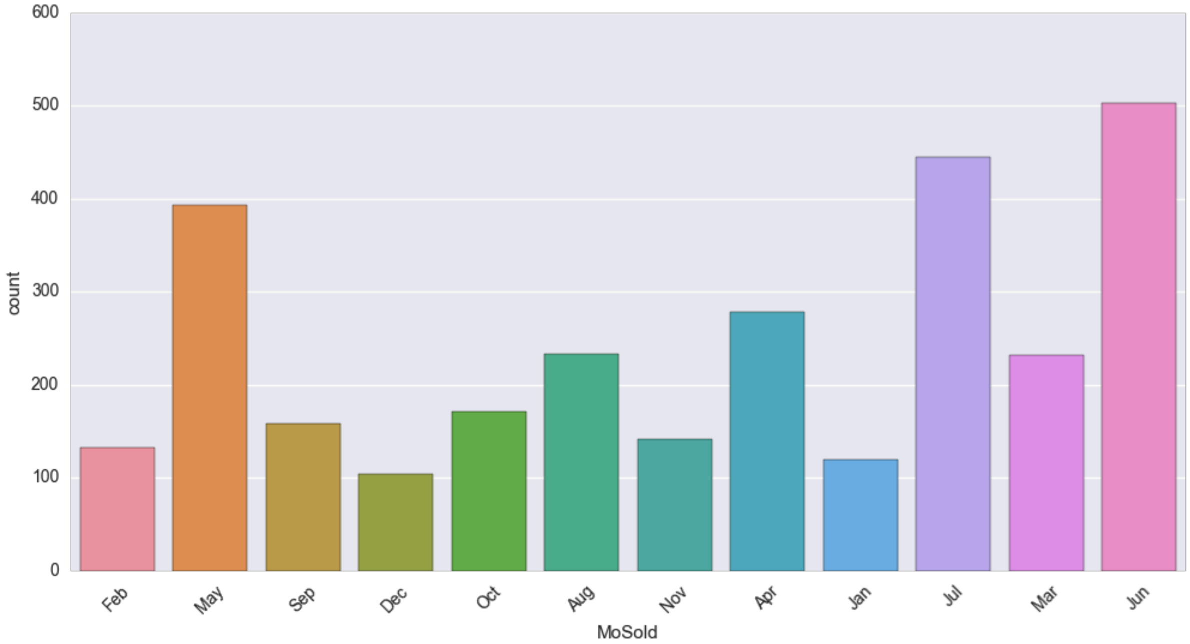
12. Total square feet for porch OpenPorchSF: Open porch area in square feet EnclosedPorch: Enclosed porch area in square feet 3SsnPorch: Three season porch area in square feet ScreenPorch: Screen porch area in square feet. The new feature *OverallPorchSF* represent the overall SF for proch.

13. House completed before sale or not The new feature *IsCompletedBeforeSale* will represent weather the house was completed before sale or not

14. Dwelling features Extracting new set of features: Newer dwelling, Older dwelling, Duplex dwelling, SplitFoyer dwelling

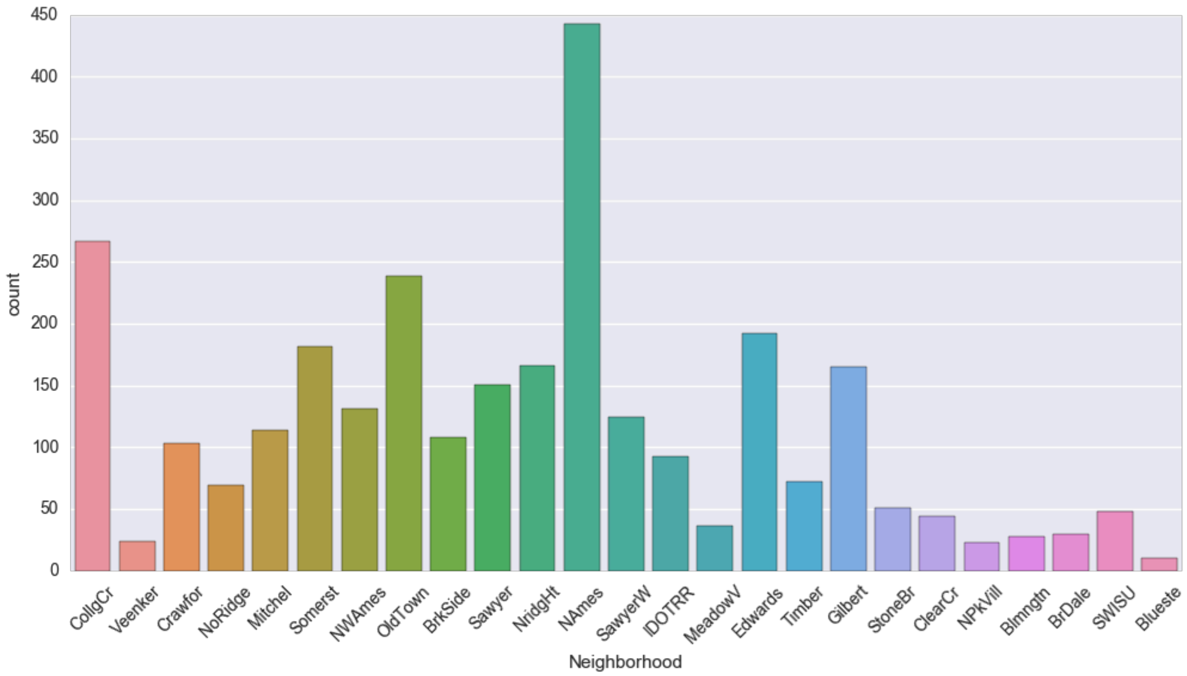
14. Bad heating Extract a new feature from HeatingQC to represent the existence of bad heating.

15. Season months Extract a new feature to represent the most selling months. From the below figure we could conclude that these months are May, Aug, Mar, Apr, Jul, Jun, therefore, a new feature will be added to capture this observation.



16. High selling Neighborhood

Extract new feature that represent high selling neighborhood, from the below figure we could conclude that CollgCr, OldTown, Somerst, NAmes, Edwards as a high selling neighborhood.



17. Feature Polynomials

Find most correlated features relative to sale price and create Polynomials features for the top selected features.

- `Numerical features: 95`
- `Categorical features: 19`

Feature Standardization for numerical features

With features being on different scales, certain weights may update faster than others since the feature values play a role in the weight updates. It is also a general requirement for many machine learning algorithms like Linea regression, PCA which will be using to train our model.

http://sebastianraschka.com/Articles/2014_about_feature_scaling.html

Identify and transform Skewed numerical features

Log transform of the skewed numerical features to lessen impact of outliers and to normalize the dataset. Skewness with |value| > 0.5 is considered at least moderately skewed. https://www.kaggle.com/apapiu/house-prices-advanced-regression-techniques/regularized-linear-models, skewed_feature = log(skewed_feature + 1)

```
We have transformed 74 skewed numerical features to log
transform.
```

One-Hot-Encoding Perform One-Hot-Encoding for the rest of categorical features to transform to one larger vector with only 0 and 1 to determine the absence or existence of specific feature.

Summary of the dataset features Encoded categorical features that has some order information Added 20 new features Apply one-hot-encoding to the rest of categorical features

Implementation

Feature selection: The aim of this part is to try to height the features that have the highest impact on the sale price

https://www.kaggle.com/juliencs/house-prices-advanced-regression-techniques/a-study-on-regression- applied-to-the-ames-dataset/

Lasso and Ridge Regression: First Linear Regression as a start point to be able to determine the impact of applying Lass and Ridge regression, in addition Lasso regression will highlight the most important features based on the correlation.

1. Linear Regression

```
RMSE on Training set: 0.250033598343, RMSE on Test set:
0.697361191992
```

The Root-mean-square (RMSE) error on the training set is high and higher on testing set which is a sign for over-fit problem, therefore, next we will apply regularization to solve the overfitting problem as well as highlight the most important feature.

2. Linear Regression with regularization Regularization is a very useful method to handle: Collinearity, Filter out noise from data, Prevent overfitting, The concept behind regularization is to introduce additional information (bias) to penalize extreme parameter weights.

2.1 Linear Regression with Ridge regularization (L2 penalty) L2 regularization adds penalty equivalent to square of the magnitude of coefficients.

2.1.1 Choosing Alpha Alpha is the bias term that will be used to penalize the extreme coefficient: if Alpha is zero: we will end up with simple linear regression, in other words, we'll get the same coefficients as simple linear regression.if Alpha is infinity: the model will not be able to capture anything as The coefficients will be zero therefore, our goal is choosing the best Alpha between zero and infinity.

Starting with very small to large Alpha values and by applying cross validation we will be able to determine the best Alpha that we should use to train our model.
l2_penalty_values = [0.05, 0.1, 0.3, 1, 3, 5, 6, 7, 8, 9, 10, 15, 30, 50, 75]
https://www.analyticsvidhya.com/blog/2016/01/complete-tutorial-ridge-lasso-regression-python/

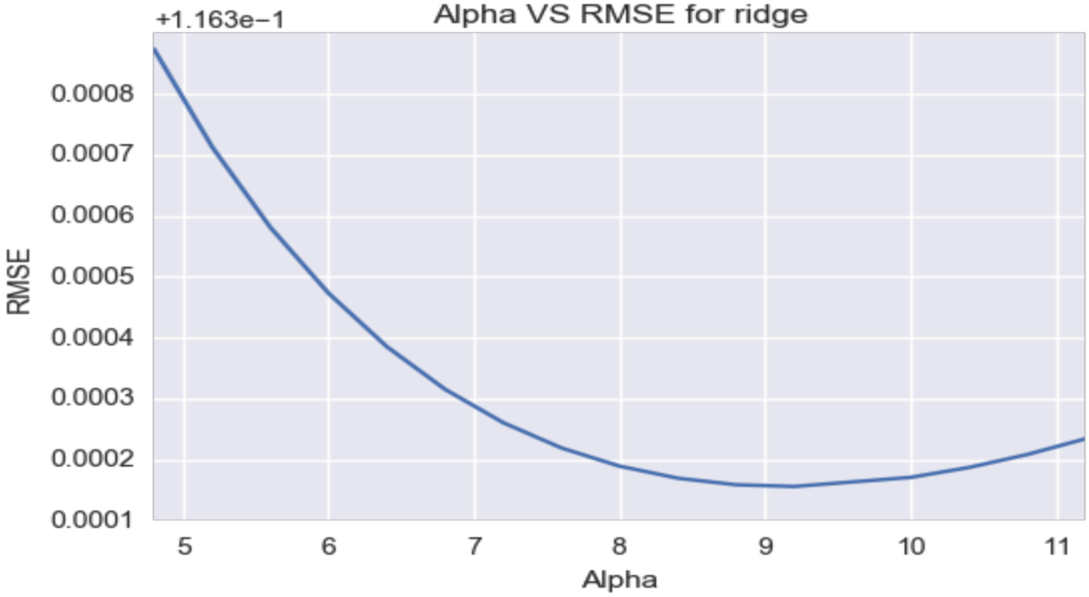


Figure: RMSE vs Alpha

When alpha is large the regularization is strong and the model cannot capture all the complexities in the data, however, when alpha is small (flexible) the model begins to over-fit.

A value of alpha = 8 is the right choice based on the above figure.

2.1.2 Train and report RMSE

- `Ridge RMSE on Training set: 0.116396272233`
- `Ridge RMSE on Test set: 0.121408532411`

With Ridge regression our RMSE error has improved a lot, the difference between training and test results indicate that we eliminated most of the overfitting.

2.1.2 Best and Worst Coefficients `Ridge picked 267 features and eliminated 6 features.` Ridge used almost all of the existing features. Thus, the major advantage of ridge regression is coefficient shrinkage and reducing model complexity.

2.2 Linear Regression with Lasso regularization (L1 penalty) Least Absolute Shrinkage and Selection Operator (Lasso) It is an alternative regularization method, by replacing the square of the weights by the sum of the absolute value of the weights. Lasso adds penalty equivalent to absolute value of the magnitude of coefficients, Minimization objective = LS Obj + α * (sum of absolute value of coefficients). L1 regularization yields sparse feature vectors as most feature weights will be helpful for features selection.

2.2.1 Choosing Alpha Same as ridge regression, however, here we try smaller range of l1_penalty_values = [0.0001, 0.0003, 0.0004, 0.0005, 0.0006, 0.001, 0.003, 0.006, 0.01, 0.03, 0.06, 0.1, 0.3, 0.6, 1]

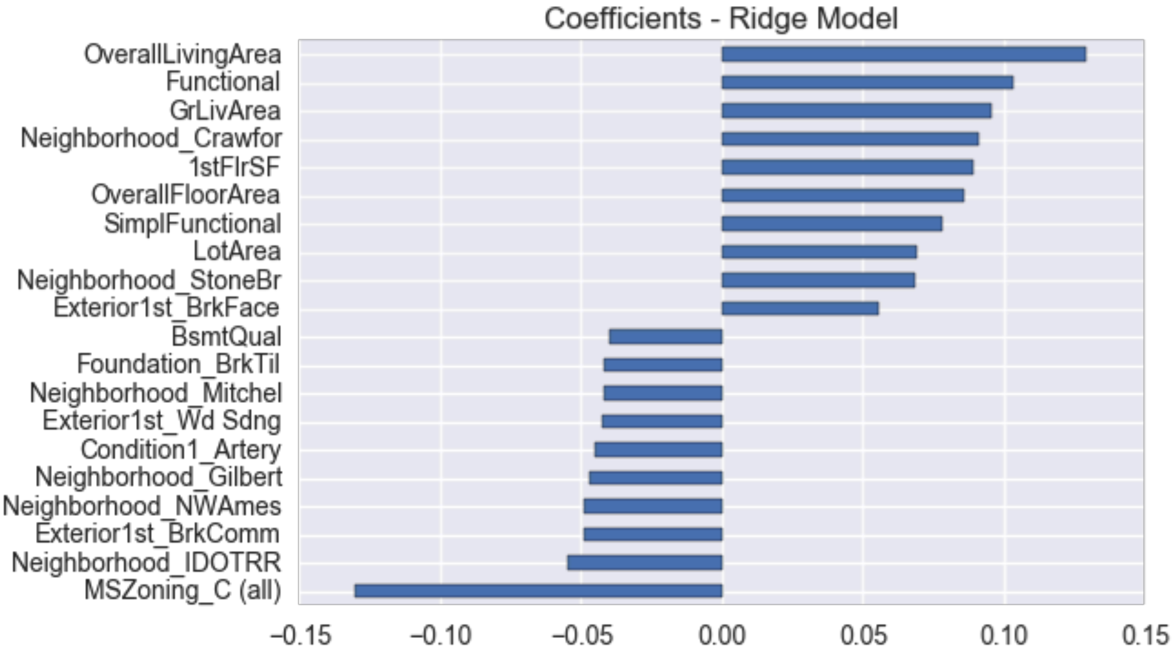


Figure: Coefficients in Ridge Model

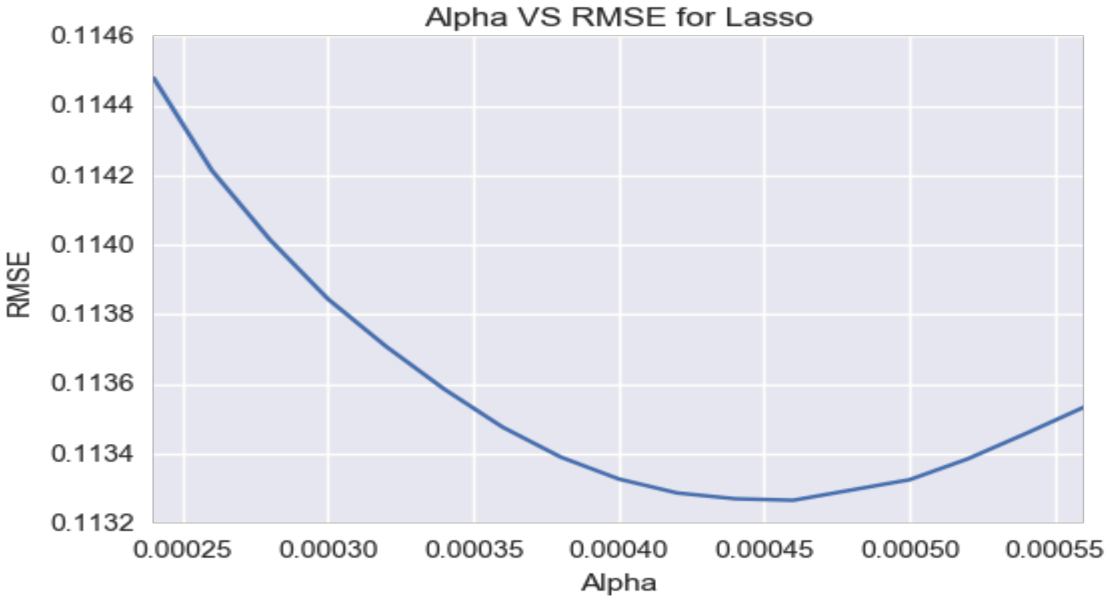
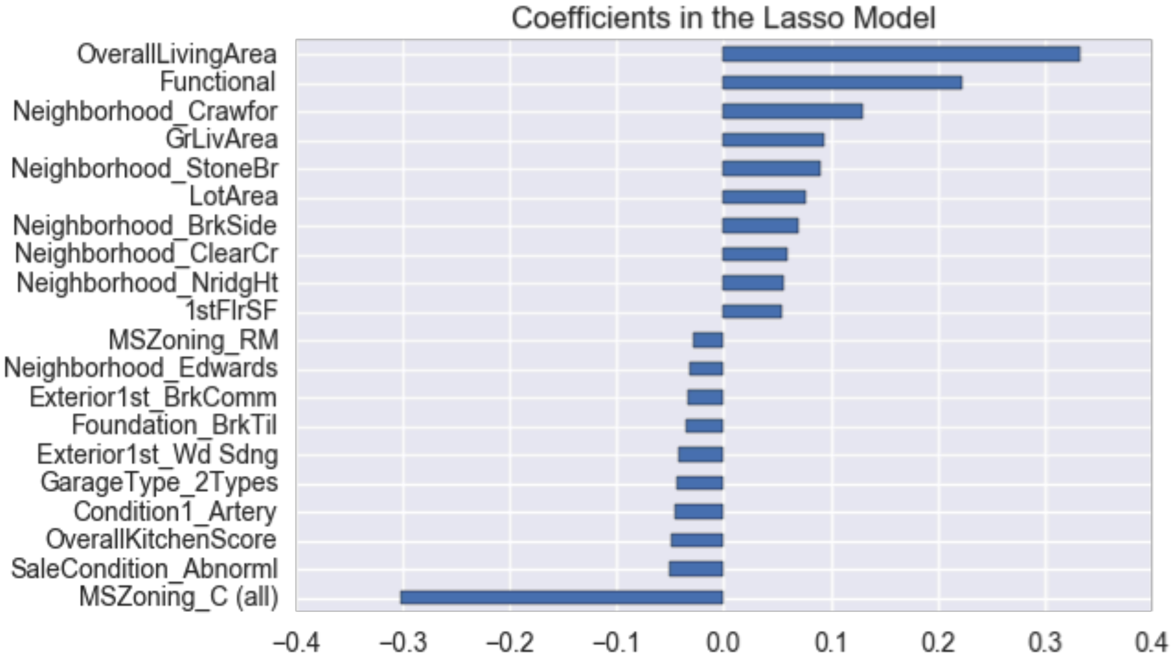


Figure: RMSE vs Alpha. Best alpha: 0.00045

2.2.2 Train and report RMSE

- `Lasso RMSE on Training set: 0.11350695539842474`
- `Lasso RMSE on Test set: 0.11600723086643834`

2.2.2 Best and Worst Coefficients Lasso picked 108 features and eliminated the other 165 features



The Lasso model allow us to select 112 features and ignored 176 features. As we could see along with shrinking coefficients, lasso performs feature selection as well, some of the coefficients become exactly zero, which is equivalent to the particular feature being excluded from the model.

Univariate Selection It's a statistical test can be used to select those features that have the strongest relationship with the output variable. http://scikit-learn.org/stable/modules/feature_selection.html#univariate-feature-selection

3.1 SelectKBest Select features according to the k highest scores. We will be using f_regression from scikit-learn: It's a linear model for testing the effect of a single regressor, sequentially for many regressors.

This is done in 2 steps:

1. The cross correlation between each and the target is computed.
2. It is converted to an F score then to a p-value.

```
[BsmtCond, BsmtFinSF1, BsmtFinSF2, BsmtFullBath, Condition1_RRNe, Condition2_Feedr,
Condition2_PosN, DuplexDwelling, Electrical_Mix, Electrical_SBrkr, EnclosedPorch,
ExterCond, ExterQual, Exterior1st_VinylSd, Exterior2nd_CBlock, Exterior2nd_Other,
Exterior2nd_Stone, Exterior2nd_Stucco, Exterior2nd_VinylSd, Exterior2nd_Wd,
FireplaceQ, Foundation_BrkTil,Foundation_CBlock, Foundation_Slab, PoolQC]
```

Above is the list of features that SelectKBest method returns.

3.2 SelectPercentile Select features according to a percentile of the highest scores

```
[3SsnPorch, BsmtCond, BsmtFullBath, Condition2_Feedr, DuplexDwelling, ExterCond,
Exterior1st_VinylSd, Exterior2nd_Stone, Exterior2nd_Stucco, Exterior2nd_VinylSd,
Foundation_CBlock, Foundation_Slab, Foundation_Wood, MSSubClass_SC190,
MSSubClass_SC20, MSZoning_RM, MiscFeature_Shed, MiscVal, MoSold_Jan,
Neighborhood_NridgHt, OverallGarageScore, RoofMatl_Roll, SaleType_Con,
SimplFunctional, SimplGarageCond, SimplOverallCond, TotRmsAbvGrd, WoodDeckSF]
```

Above is the list of features that SelectPercentile method returns.

Feature transformationPrincipal Component Analysis (PCA)

From sklearn library we will use GridSearchCV to select the best parameters from PCA, the parameter that we will search through are number of component and svd_solver.

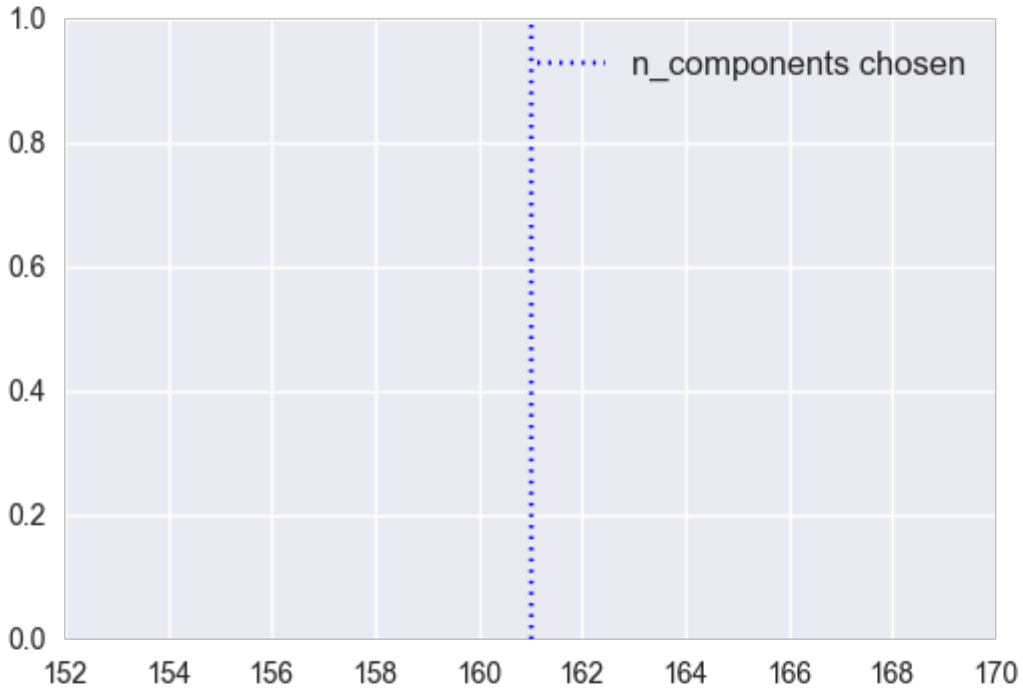


Figure: PCA number of components chosen

From the above figure we could see that PCA created 160 transformed features that could be used.

Training

Random Forest A random forest fits a number of classifying decision trees on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control over-fitting. http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

- `Random Forest RMSE on Training set: 0.140613028916`
- `Random Forest RMSE on Test set: 0.137831765785`

Tune Random Forest Hyper-parameters by using Gridsearch

Hyper-parameters are parameters that are not directly learnt within estimators they are passed as arguments to the constructor of the estimator classes. The best parameters that we reached for RandomForestRegressor are: `bootstrap: True, criterion: mse, max_features: 50, n_estimators: 700`.

- `Tuned Random Forest RMSE on Training set: 0.131269766413`
- `Tuned Random Forest RMSE on Test set: 0.130984158588`

The RMSE for the tuned Random Forest is better than the un-tuned Random Forest, however, we got better scores with Lasso and Ridge regression.

Combine Tune Random Forest with Lasso features

- `Tuned Random Forest with Lasso features RMSE on Training set: 0.131325177679`
- `Tuned Random Forest with Lasso features RMSE on Test set: 0. 130886344633`

The RMSE for the tuned Random Forest with features from lasso are very similar.

Combine Tune Random Forest with PCA features

- `Tuned, Combine (PCA) Random Forest RMSE on Training set: 0.207585810796`
- `Tuned, Combine (PCA) Random Forest RMSE on Test set: 0.233587678663`

The RMSE for the tuned Random Forest with features from PCA is higher than the rest of all models.

Combine Tune Random Forest with combined features from PCA and SelectKBest

- `Tuned, Combine Random Forest RMSE on Training set: 0.144987114561`
- `Tuned, Combine Random Forest RMSE on Test set: 0.147744570882`

The RMSE for the tuned Random Forest with features from PCA and Kbest is better than features from PCA alone, however, it's still higher that the tuned parameter alone.

Tune Random Forest and combine features from SelectPercentile and

SelectKBest

- Tuned, Combine (Kbest, Percentile) Random Forest RMSE on Training set: 0.145523129516
- Tuned, Combine (Kbest, Percentile) Random Forest RMSE on Test set: 0.144164712044

The RMSE for the tuned Random Forest with features from SelectPercentile and SelectKBest is better than features from PCA alone and PCA with SelectKBest, however, it's still higher that the tuned parameter alone.

Visualization summary of best random forest model

As a conclusion the tuned random forest model produced the better results, however, in general Regularized regression model yield better scores.

Below is a visualization summary for random forest model.

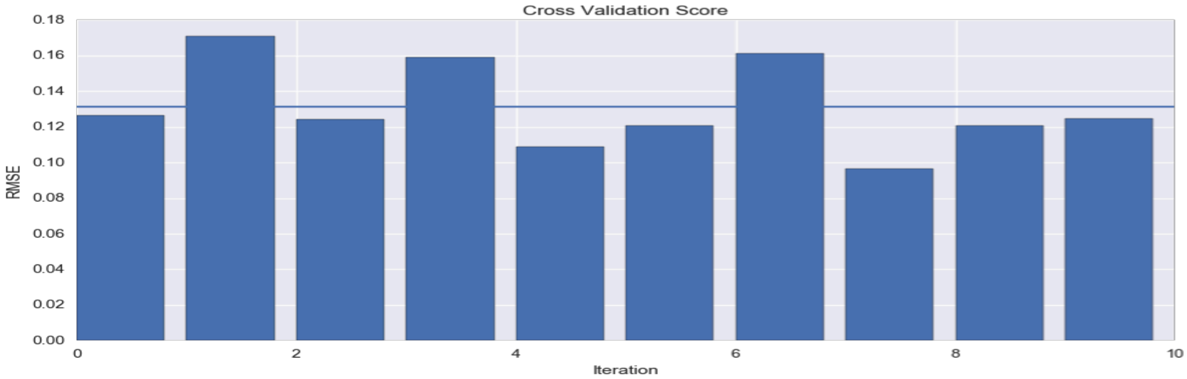


Figure: the cross validation errors summary for RandomForestRegressor.

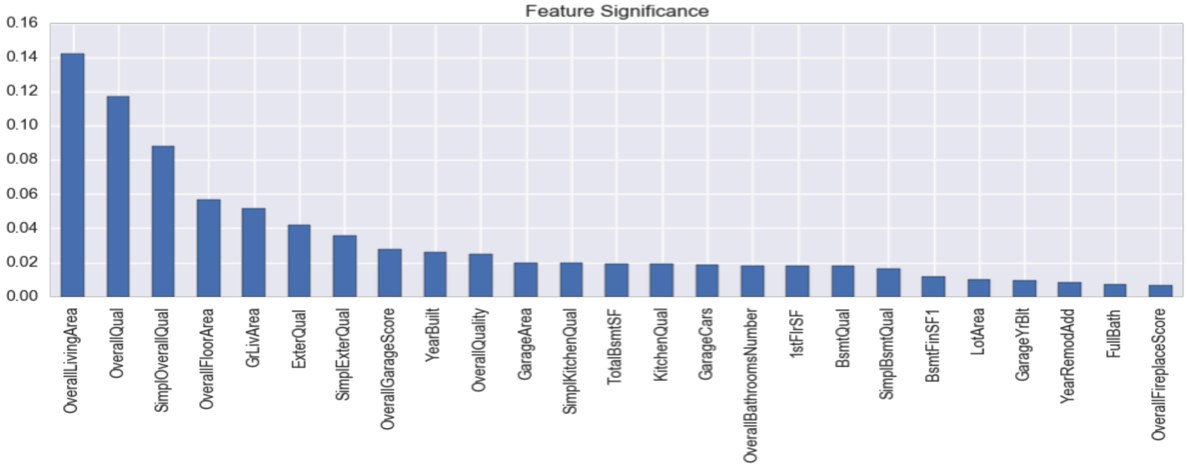


Figure: RandomForestRegressor feature importance.

Best Model as lasso model is the model that reported the lowest RMSE score for both training and testing sets, we will try to use the same model and features from SelectKBest, SelectPercintile and PCA.

- Combine (Kbest, Percentile) with lasso RMSE on Training set: 0.137002205605
- Combine (Kbest, Percentile) with lasso RMSE on Test set: 0.128995129598
- Combine (Kbest) with lasso RMSE on Training set: 0.140483763831
- Combine (Kbest) with lasso RMSE on Test set: 0.133560380862
- Combine (Percentile) with lasso RMSE on Training set: 0.156094254311
- Combine (Percentile) with lasso RMSE on Test set: 0.149342309673
- Combine (PCA) with lasso RMSE on Training set: 0.116792311047
- Combine (PCA) with lasso RMSE on Test set: 0.123500844157

Refinement

Regression Models Applied linear regression then used regularized our linear model using Ridge and Lasso regression to deal with over-fit problem that we got from linear regression and highlighted the the most relevant features.

Feature selection/transformation Models Used SelectPercentile and SelectKBest for feature selection and PCA for feature transformation.

Random Forest Models Used a random forest algorithm without any tuning, then tune it's hyper-parameter. Train random forest, with features from PCA, with features from PCA and SelectKBest and finally with features from SelectKBest and SelectPercentile

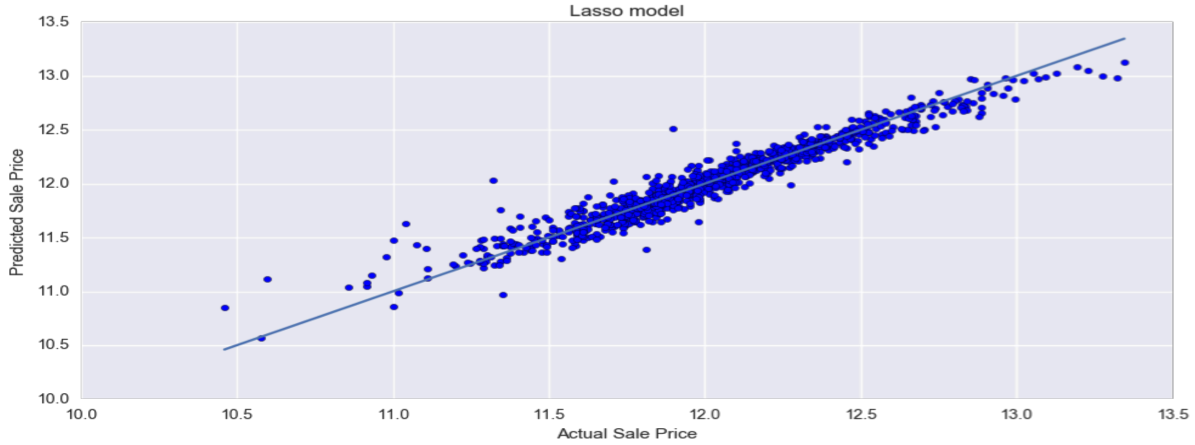
IV. Results

From the above we could conclude that regularized regression with lasso reported the lowest RMSE error. The best Alpha value is 0.00045 which we reached after trying different Alpha values by using cross validation over 10 folds. The reported RMSE for training data is 0.1135 and the reported RMSE for testing data is 0.1160, as we could see that the scores are very close to each other which means that our model was able to generalize very well to new or unseen data. We also tried Lasso model with the features yield from the feature selection and transformation models, we have combine SelectKBest features, SelectPercientile features and PCA transformed features. However, all of them reported a little higher RMSE which mean that selecting the best features really has an impact on the learning model. Changing the parameters for Lasso model yield different results, however, in general the variation between the results is very small 0.233 to 0.116 which an indication that the result from the model could be trusted.

Justification

Now we will compare our best model (regularized regression with lasso) to the benchmark model. Least Absolute Shrinkage and Selection Operator (Lasso) It is an alternative regularization method, by replacing the square of the weights by the sum of the absolute value of the weights. Lasso adds penalty equivalent to absolute value of the magnitude of coefficients, Minimization objective = LS Obj + α * (sum of absolute value of coefficients) L1 regularization yields sparse feature vectors as most feature weights will be helpful for features selection. As stated above our bench mark model yield RMSE of 0.245 on training dataset 0.247 on testing dataset. On the other hand, our best model yield 0.113 on training set and 0.116 which clearly shows that our model performs better than the benchmark model. With error rate 10% our model is significant enough to have solved the problem.

V. Conclusion



The above figure shows the how lass was able to capture most of sale price, by comparing the actual sale price and the predicted sale price the prediction line shows how the predicted and actual sale price are highly correlated.

Reflection

Summarization of the entire process

1. Data Exploration: Understand the dataFeatures distribution and description House prices distribution
2. Exploratory Visualization: Numerical features correlation Categorical Features relevance
3. Benchmark: A linear regression model that predict the price based on some of the traditional parameters like square_feed, num_of_rooms, neighbour.
4. Data Preprocessing: Spot outliers, Handel Null data, Encoding categorical feature, Feature Engineering, Feature Standardization for numerical features
5. Implementation
    a. Lasso and Ridge Regression Feature selection:
    b. Univariate Selection SelectKBest, SelectPercentile Feature transformation
    c. PCA
    d. Random Forest Regressor
    e. Tune Hyper-parameters by using Gridsearch
    f. Combine features selected and transformed features with Lasso and Random Forest models
    g. Sated the results for each model and reported the Best Model

Interesting aspects It was very interesting for me that a linear regression regularization model could performs better than a boosted random forest regressor model, as a result, there are no good or bad models however, each model may perform better or worse than other models with respect to the dataset or the problem that we are trying to solve.

Difficult aspects The most difficult aspect was data preprocessing as the dataset contains many features some numerical, some categorical and some are categorical that has some order, and each type need different approach to make the dataset ready for machine learning algorithms. Another observation is how by just make the skewed data more normal has significant improvement over the model results.

The final solution meets my expectations and i think the process taking could be applied to different datasets that try to predict any continues variable like sale price.

Improvement

As an improvement I believe that we need to work more on the features selection part as there many features that has less impact, also emphasize more the important features, for example, the neighborhood has a great impact on the sale price if we could do some research on the Ames city and create a features that point out or order the most important neighborhood. I would also consider using xgboost algorithm that I researched and compare its results. I believe if my solution could be as a benchmark it will be very hard to find better solution, however, it's a challenge and that I wish I could overcome or see other that reported a better solution as learn from.