Data exploration and transformations

Data observation, transformation, and preparation of data for model development have been carried out in python. HTML file of Jupiter notebook for some data observation is attached here: [Jupyter_sujal_housepredict]

Some more data analysis other than what I did on Jupiter is carried out in a local python environment. Which is discussed below:

Removed the following columns. ‘Id’ column is simply the index of the sample, and of no use in further analysis. It has been found that ‘Alley’,’FireplaceQu’, ‘PoolQC’, ‘Fence’, ‘MiscFeature’ are the parameters which have more than 25% missing values. Hence, rather than imputing it, I found it appropriate to drop that out.

```
train_copy=train_org.copy()
test_copy=test_org.copy()
print(test_copy.shape)
train_pre=train_copy.drop(['Id','Alley','FireplaceQu','PoolQC','Fence','MiscFeature'],axis=1)
test_pre=test_copy.drop(['Id','Alley','FireplaceQu','PoolQC','Fence','MiscFeature'],axis=1)
```

Imputation of missing values. Numerical data with column means and non-numeric data with the highest frequency.

```
train=DataFrameImputer().fit_transform(train_pre)
test=DataFrameImputer().fit_transform(test_pre)
```

It was found that ‘GarageYrBlt’ is a numerical data type in training set but is categorical in nature. Hence, after making an imputation on it, it is necessary to make sure that imputed values are integers (and not fractions). This concern was handled both on training and testing datasets.

```
train['GarageYrBlt']=train['GarageYrBlt'].astype(int) # This is the only numerical quantity which
test['GarageYrBlt']=test[['GarageYrBlt', 'GarageCars','BsmtHalfBath','BsmtFullBath']].astype(int)
```

The column names which include numbers were renamed to avoid further confusion.

```
train=train.rename(columns={"1stFlrSF":"firstFlrSf","2ndFlrSF": "secondFlrSf","3SsnPorch":"threeSsnPorch"}, errors="raise") #The provide
test=test.rename(columns={"1stFlrSF":"firstFlrSf","2ndFlrSF": "secondFlrSf","3SsnPorch":"threeSsnPorch"}, errors="raise")
```

After observing the scatter plot, I found it appropriate to remove ‘GrLivArea’>4500, and ‘LotFrontage’>300. They seem to be an outlier. (It has also been removed from the testing set)

```
train.drop(train[train['LotFrontage']>300].index, inplace=True)
train.reset_index(drop=True, inplace=True)
```

```
train.drop(train[train['GrLivArea']>4500].index, inplace=True)
train.reset_index(drop=True, inplace=True)
```

As discussed in Jupyter notebook (Jupyter_sujal_housepredict), I found it reasonable to work with log(y) scale. Hence, converted ‘SalePrice’ to log scale.

```
train['logSalePrice']=np.log(train['SalePrice'])
train=train.drop('SalePrice',axis=1)
print(train.shape)
print(train.columns)
```

After observing the histogram of the categorical variable. It seems that these parameters 'Street','Utilities','Condition2','RoofMatl' have no variation in the training sample, and would provide no information on possible contribution on final prediction. Hence, removed.

```
train=train.drop(['Street','Utilities','Condition2','RoofMatl'],axis=1)
test=test.drop(['Street','Utilities','Condition2','RoofMatl'],axis=1)
print(train.shape)
print(test.shape)
```

The following numerical variables are highly skewed. Logarithm transformation makes sense. However, since some of the samples have zero values, I had instead chosen log(1+x) transformation.

```
train['LotFrontage']=np.log(train['LotFrontage']+1)
train['LotArea']=np.log(train['LotArea']+1)
train['BsmtFinSF1']=np.log(train['BsmtFinSF1']+1)
train['firstFlrSf']=np.log(train['firstFlrSf']+1)
train['MiscVal']=np.log(train['MiscVal']+1)

test['LotFrontage']=np.log(test['LotFrontage']+1)
test['LotArea']=np.log(test['LotArea']+1)
test['BsmtFinSF1']=np.log(test['BsmtFinSF1']+1)
test['firstFlrSf']=np.log(test['firstFlrSf']+1)
test['MiscVal']=np.log(test['MiscVal']+1)
```

Model Development Python code here

1. Full Regression

Linearly regressed the ‘logSalePrice’ with all other parameters. A total summary can be seen at Jupyter notebook (Jupyter_sujal_housepredict)

```
                            OLS Regression Results
==============================================================================
Dep. Variable:            logSalePrice   R-squared:                       0.965
Model:                             OLS   Adj. R-squared:                  0.944
Method:                  Least Squares   F-statistic:                     46.44
Date:                 Thu, 12 Dec 2019   Prob (F-statistic):               0.00
Time:                         14:51:57   Log-Likelihood:                 1706.4
No. Observations:                 1457   AIC:                            -2331.
Df Residuals:                      916   BIC:                             528.0
Df Model:                          540
Covariance Type:             nonrobust
==============================================================================
```

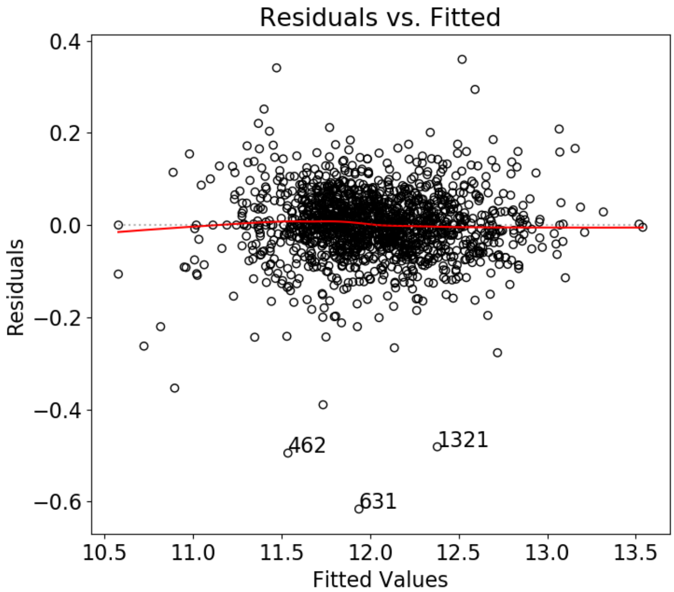
As seen, the R-squared adjusted comes out to be 0.944. By looking at the whole summary, it is evident that very few parameters have shown statistical significance (by observing P values).

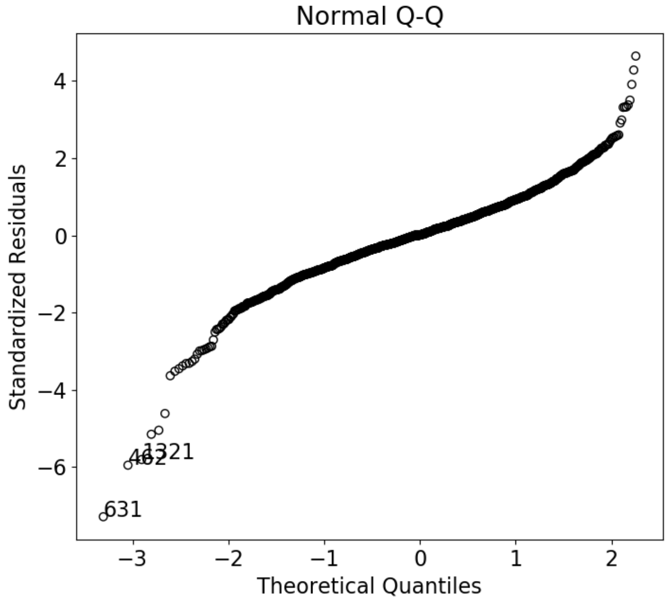
The snapshot of the first few parameters is presented here:

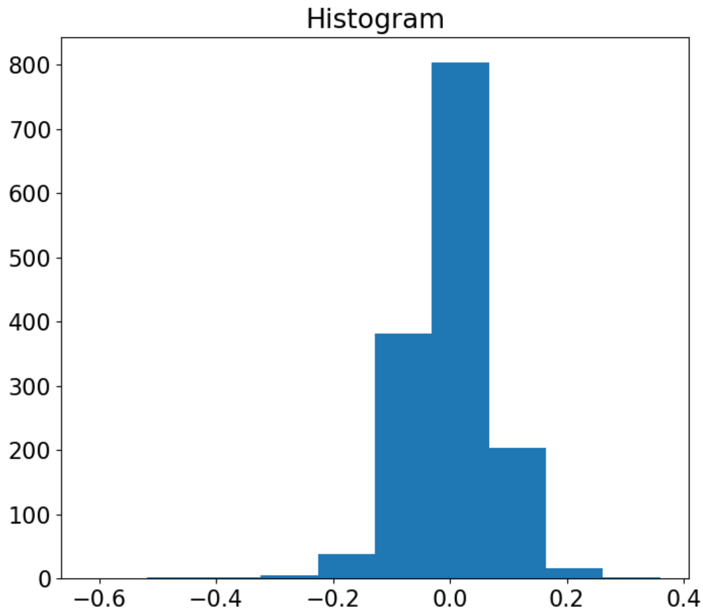
	coef	std err	t	P>|t|	[0.025	0.975]
ntercept	7.9411	0.495	16.027	0.000	6.969	8.914
(MSSubClass)[T.30]	-0.0527	0.029	-1.812	0.070	-0.110	0.004
(MSSubClass)[T.40]	-0.0131	0.071	-0.185	0.853	-0.152	0.126
(MSSubClass)[T.45]	-0.0494	0.111	-0.444	0.657	-0.268	0.169
(MSSubClass)[T.50]	-0.0229	0.047	-0.491	0.623	-0.114	0.068
(MSSubClass)[T.60]	-0.0436	0.040	-1.088	0.277	-0.122	0.035
(MSSubClass)[T.70]	-0.0538	0.046	-1.158	0.247	-0.145	0.037
(MSSubClass)[T.75]	-0.0953	0.106	-0.897	0.370	-0.304	0.113
(MSSubClass)[T.80]	0.0071	0.055	0.127	0.899	-0.102	0.116
(MSSubClass)[T.85]	-0.0026	0.058	-0.045	0.964	-0.117	0.111
(MSSubClass)[T.90]	-0.0239	0.022	-1.074	0.283	-0.067	0.020
(MSSubClass)[T.120]	-0.0754	0.073	-1.032	0.302	-0.219	0.068
(MSSubClass)[T.160]	-0.1309	0.088	-1.493	0.136	-0.303	0.041
(MSSubClass)[T.180]	-0.0450	0.097	-0.464	0.643	-0.235	0.145
(MSSubClass)[T.190]	0.4118	0.220	1.870	0.062	-0.020	0.844
SZoning[T.FV]	0.5772	0.065	8.840	0.000	0.449	0.705
SZoning[T.RH]	0.5093	0.070	7.318	0.000	0.373	0.646
SZoning[T.RL]	0.5260	0.058	9.017	0.000	0.411	0.640
SZoning[T.RM]	0.4657	0.056	8.363	0.000	0.356	0.575
otShape[T.IR2]	0.0026	0.020	0.130	0.896	-0.037	0.042
otShape[T.IR3]	0.0482	0.039	1.226	0.221	-0.029	0.125
otShape[T.Reg]	0.0104	0.008	1.344	0.179	-0.005	0.026
andContour[T.HLS]	0.0060	0.027	0.223	0.823	-0.047	0.058
andContour[T.Low]	0.0171	0.033	0.512	0.609	-0.049	0.083
andContour[T.Lvl]	-0.0048	0.021	-0.234	0.815	-0.045	0.036
otConfig[T.CulDSac]	0.0384	0.016	2.406	0.016	0.007	0.070
otConfig[T.FR2]	-0.0349	0.019	-1.873	0.061	-0.072	0.002
otConfig[T.FR3]	-0.0471	0.055	-0.853	0.394	-0.155	0.061
otConfig[T.Inside]	-0.0075	0.009	-0.851	0.395	-0.025	0.010
andSlope[T.Mod]	0.0138	0.020	0.679	0.498	-0.026	0.054
andSlope[T.Sev]	-0.0824	0.049	-1.690	0.091	-0.178	0.013
eighborhood[T.Blueste]	0.0089	0.097	0.092	0.927	-0.182	0.200
eighborhood[T.BrDale]	0.0297	0.058	0.510	0.610	-0.085	0.144
eighborhood[T.BrkSide]	0.0147	0.047	0.312	0.755	-0.078	0.108
eighborhood[T.ClearCr]	0.0009	0.046	0.018	0.985	-0.090	0.092
eighborhood[T.CollgCr]	-0.0467	0.036	-1.292	0.197	-0.118	0.024
eighborhood[T.Crawfor]	0.0683	0.044	1.567	0.117	-0.017	0.154
eighborhood[T.Edwards]	-0.0841	0.039	-2.133	0.033	-0.161	-0.007
eighborhood[T.Gilbert]	-0.0484	0.038	-1.267	0.205	-0.123	0.027
eighborhood[T.IDOTRR]	-0.0035	0.056	-0.062	0.951	-0.114	0.107
eighborhood[T.MeadowV]	-0.0972	0.062	-1.567	0.117	-0.219	0.025
eighborhood[T.Mitchel]	-0.0836	0.041	-2.023	0.043	-0.165	-0.003
eighborhood[T.NAmes]	-0.0791	0.039	-2.036	0.042	-0.155	-0.003
eighborhood[T.NPkVill]	0.0100	0.066	0.152	0.879	-0.119	0.139
eighborhood[T.NWAmes]	-0.0669	0.041	-1.646	0.100	-0.147	0.013
eighborhood[T.NoRidge]	-0.0003	0.043	-0.008	0.994	-0.084	0.083
eighborhood[T.NridgHt]	0.0128	0.036	0.354	0.724	-0.058	0.084
eighborhood[T.OldTown]	-0.0449	0.049	-0.924	0.356	-0.140	0.050
eighborhood[T.SWISU]	-0.0668	0.051	-1.311	0.190	-0.167	0.033
eighborhood[T.Sawyer]	-0.0656	0.041	-1.602	0.109	-0.146	0.015
eighborhood[T.SawyerW]	-0.0436	0.040	-1.098	0.273	-0.122	0.034
eighborhood[T.Somerst]	-0.0225	0.043	-0.522	0.602	-0.107	0.062

Residual diagnostic on full model:

Three plots were made to check the underline assumption of residuals, i.e to observe the normality of errors, and uniformity in variance. Nothing seems much concerning about he residual behavior. We are good to proceed ahead with the developed model for further improvements.


Residuals vs. Fitted
Residuals
Fitted Values
0.4
0.2
0.0
−0.2
−0.4
−0.6
10.5
11.0
11.5
12.0
12.5
13.0
13.5
462
1321
631


Normal Q-Q
Standardized Residuals
Theoretical Quantiles
4
2
0
−2
−4
−6
−3
−2
−1
0
1
2
462
1321
631


Histogram
800
700
600
500
400
300
200
100
0
−0.6
−0.4
−0.2
0.0
0.2
0.4

2. Providing interaction in three pairs of parameters

It is observed from the correlation graph that the following pairs are highly correlated 1stFlrSf:TotalBsmtSF TotalRmAbvGrnd:GrLivArea GarageArea:GarageCar. I have provided interaction between those in my full model to check the performance. However, it has provided no improvement over the existing model.

P values

LotFrontage	0.0198	0.018	1.126	0.260	-0.015	0.054
LotArea	0.0742	0.014	5.307	0.000	0.047	0.102
MasVnrArea	1.567e-05	2.66e-05	0.589	0.556	-3.66e-05	6.79e-05
BsmtFinSF1	0.0055	0.009	0.615	0.539	-0.012	0.023
BsmtFinSF2	-1.402e-05	4.37e-05	-0.321	0.748	-9.98e-05	7.18e-05
TotalBsmtSF	0.0001	2.98e-05	4.066	0.000	6.26e-05	0.000
firstFlrSf:BsmtUnfSF	-6.351e-06	3.4e-06	-1.866	0.062	-1.3e-05	3.29e-07
secondFlrSf	2.274e-07	3.58e-05	0.006	0.995	-7e-05	7.05e-05
LowQualFinSF	-9.547e-05	0.000	-0.812	0.417	-0.000	0.000
C(TotRmsAbvGrd)[2]:GrLivArea	0.0009	0.001	1.718	0.086	-0.000	0.002
C(TotRmsAbvGrd)[3]:GrLivArea	0.0003	6.02e-05	4.802	0.000	0.000	0.000
C(TotRmsAbvGrd)[4]:GrLivArea	0.0003	3.55e-05	7.659	0.000	0.000	0.000
C(TotRmsAbvGrd)[5]:GrLivArea	0.0003	3.03e-05	8.887	0.000	0.000	0.000
C(TotRmsAbvGrd)[6]:GrLivArea	0.0003	2.84e-05	9.916	0.000	0.000	0.000
C(TotRmsAbvGrd)[7]:GrLivArea	0.0003	2.69e-05	10.248	0.000	0.000	0.000
C(TotRmsAbvGrd)[8]:GrLivArea	0.0003	2.66e-05	10.531	0.000	0.000	0.000
C(TotRmsAbvGrd)[9]:GrLivArea	0.0003	2.66e-05	10.137	0.000	0.000	0.000
C(TotRmsAbvGrd)[10]:GrLivArea	0.0003	2.62e-05	10.322	0.000	0.000	0.000
C(TotRmsAbvGrd)[11]:GrLivArea	0.0003	2.77e-05	9.637	0.000	0.000	0.000
C(TotRmsAbvGrd)[12]:GrLivArea	0.0003	3.21e-05	8.746	0.000	0.000	0.000
C(TotRmsAbvGrd)[14]:GrLivArea	0.0003	6.28e-05	4.414	0.000	0.000	0.000
C(GarageCars)[0]:GarageArea	2.067e-17	1.2e-17	1.716	0.087	-2.97e-18	4.43e-17
C(GarageCars)[1]:GarageArea	9.863e-06	5.66e-05	0.174	0.862	-0.000	0.000
C(GarageCars)[2]:GarageArea	5.947e-05	3.6e-05	1.652	0.099	-1.12e-05	0.000
C(GarageCars)[3]:GarageArea	9.748e-05	2.95e-05	3.301	0.001	3.95e-05	0.000
C(GarageCars)[4]:GarageArea	0.0002	7.21e-05	2.458	0.014	3.57e-05	0.000
WoodDeckSF	0.0001	2.82e-05	3.632	0.000	4.71e-05	0.000
OpenPorchSF	0.0001	5.6e-05	2.016	0.044	2.97e-06	0.000
EnclosedPorch	7.412e-05	6.42e-05	1.154	0.249	-5.19e-05	0.000
threeSsnPorch	0.0002	0.000	1.635	0.102	-3.45e-05	0.000
ScreenPorch	0.0002	6e-05	3.804	0.000	0.000	0.000
PoolArea	0.0002	8.99e-05	1.734	0.083	-2.06e-05	0.000
MiscVal	-0.0024	0.003	-0.870	0.385	-0.008	0.003

Though p values of some of the interactions are significant. There is no improvement over R-squared.

```
                            OLS Regression Results
==============================================================================
Dep. Variable:            logSalePrice   R-squared:                       0.965
Model:                             OLS   Adj. R-squared:                  0.944
Method:                  Least Squares   F-statistic:                     46.42
Date:                 Thu, 12 Dec 2019   Prob (F-statistic):               0.00
Time:                         15:07:26   Log-Likelihood:                 1701.9
No. Observations:                 1457   AIC:                            -2326.
Df Residuals:                      918   BIC:                             522.3
Df Model:                          538
Covariance Type:             nonrobust
==============================================================================================
```

[I did the rest of the modeling in R. The prepared dataset in python is exported as a CSV and imported in R for further model development: train, test]

R script: here

3. Further development in R

```r
train<-read.csv('trainsujal.csv')
test<-read.csv('testsujal.csv')
train
train<- subset(train,select=-X)
test<- subset(test,select=-X)
train
ncol(train)
ncol(test)

combi <- rbind(subset(train,select=-logSalePrice),test)
combi
nrow(combi)

combi$MSSubClass <- as.factor(combi$MSSubClass)
combi$OverallQual <- as.factor(combi$OverallQual)
combi$YearBuilt <- as.factor(combi$YearBuilt)
combi$YearRemodAdd <- as.factor(combi$YearRemodAdd)
combi$BsmtFullBath <- as.factor(combi$BsmtFullBath)
combi$BsmtHalfBath <- as.factor(combi$BsmtHalfBath)
combi$BedroomAbvGr <- as.factor(combi$BedroomAbvGr)
combi$KitchenAbvGr <- as.factor(combi$KitchenAbvGr)
combi$TotRmsAbvGrd <- as.factor(combi$TotRmsAbvGrd)
combi$Fireplaces <- as.factor(combi$Fireplaces)
combi$GarageYrBlt <- as.factor(combi$GarageYrBlt)
combi$GarageCars <- as.factor(combi$GarageCars)
combi$MoSold <- as.factor(combi$MoSold)
combi$YrSold <- as.factor(combi$YrSold)
```

Training and testing sets are combined to deal with some of the parameter transformations. Such as, numerical categorical variables are passed with as.factor.

After the required transformation, the combined data is again split into training and testing data based on their sample size.

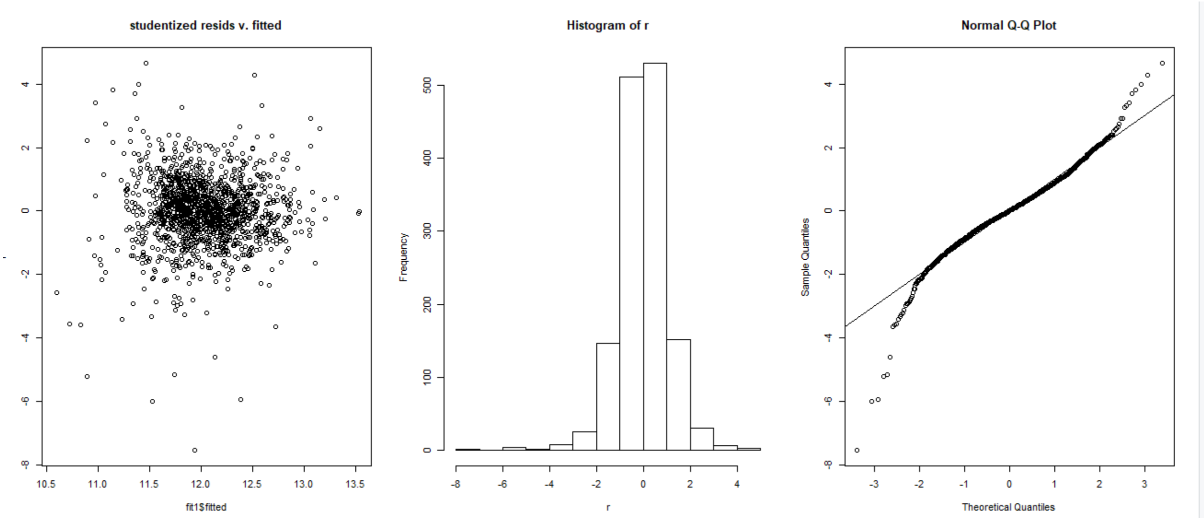
```r
train_nw<-combi[1:1457,]
test_nw<- combi[1458:2916,]
train_nw
train['logSalePrice']

train_nw['logSalePrice']<-train['logSalePrice']
train_nw
test_nw$MSSubClass
train_nw$MSSubClass
```

Full Model check. in R

```
#MSSubClass<- as.factor(MSSubClass)
fit1<- lm(logSalePrice~.,data=train_nw)
summary(fit1)
```

```
r <- rstudent(fit1)
par(mfrow=c(1,3))
plot(fit1$fitted, r, main="studentized resids v. fitted")
hist(r)
qqnorm(r); abline(0,1)
```

Residual diagnostic for full model form R

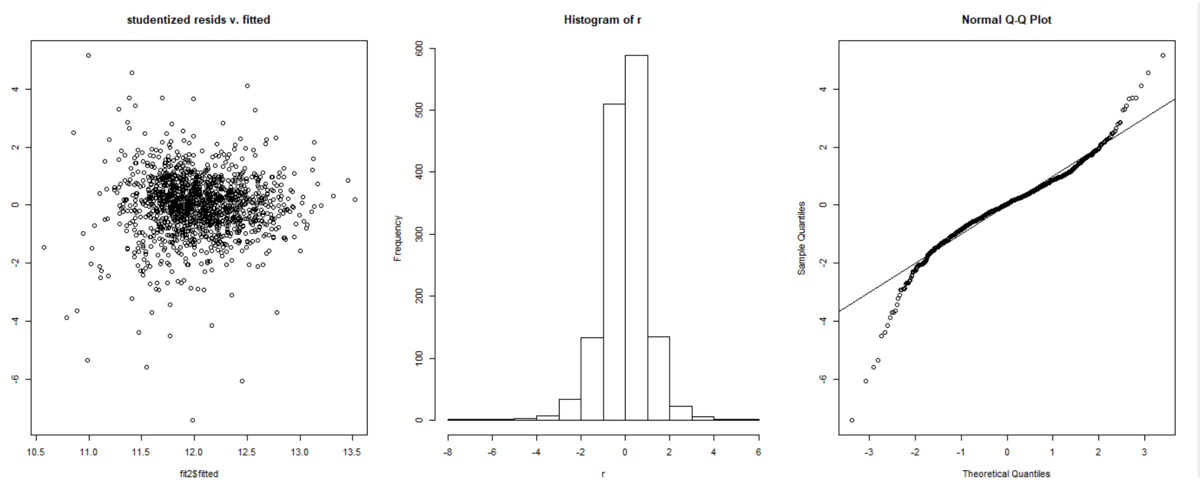


Some levels of certain parameters are different in testing and training sets. Also from the full model summary, it reveals that those columns had no statistical significance. Hence, I found it appropriate to drop that out for making a prediction on testing data.

```
## Some of the levels are missing from testing set. also from the summary it reveals that those column had no statistical significance

train_nw1<- subset(train_nw,select=-c(MSSubClass,YearBuilt,BsmtFullBath,BsmtHalfBath,TotRmsAbvGrd,Fireplaces, GarageYrBlt, GarageCars))
test_nw1<- subset(test_nw,select=-c(MSSubClass,YearBuilt,BsmtFullBath,BsmtHalfBath, TotRmsAbvGrd, Fireplaces, GarageYrBlt, GarageCars))
fit2<- lm(logSalePrice~.,data=train_nw1)
summary(fit2)
r <- rstudent(fit2)
par(mfrow=c(1,3))
plot(fit2$fitted, r, main="studentized resids v. fitted")
hist(r)
qqnorm(r); abline(0,1)
```

Adjusted R-squared for fit2 is `0.9377`

Residual diagnostic from fit 2



Quite similar to fit1. Nothing much concerning.

Prediction through fit2:

```
# par(mfrow=c(1,1))
# plot(logSalePrice)
p1<-predict(fit2,test_nw1,interval='prediction')
# abline(fit2)
# x<-seq(1,1457,by=1)
# lines(x, p1[,2], lty=2)
#
```

Got RMSE value of 0.13619 on the testing set.

Sujal_sb.csv 0.13619
2 days ago by Sujal Bhavsar

This is a simple linear regression. With consideration on few variable transformation. Some of the features have been dropped.

4. Used step with BIC to reduce the model

```
## use BIC with step to reduce the model

null<-lm(logSalePrice~1,data=train_nw)
full<-lm(logSalePrice~.,data=train_nw)
n <- nrow(train_nw)
fit3<- step(null,scope=formula(full),direction='forward',k=log(n),trace=0)
summary(fit3)
```

Obtained reduced model through step is:

```
Call:
lm(formula = logSalePrice ~ OverallQual + GrLivArea + Neighborhood +
    TotalBsmtSF + OverallCond + BsmtUnfSF + GarageArea + SaleCondition +
    CentralAir + LotArea + Foundation + MSZoning + GarageFinish +
    KitchenQual + ScreenPorch + Functional + BsmtQual + HalfBath +
    firstFlrSF + secondFlrSF + BsmtExposure + KitchenAbvGr +
    FullBath, data = train_nw)
```

Got an adjustd R-squred of `0.9271`

Model probability check.

```
# model probabilities

BIC<- data.frame(fit2=extractAIC(fit2,k=log(n))[2],fit3=extractAIC(fit3,k=log(n))[2])
eBIC<-exp(-.5*(BIC-min(BIC)))
round(probs<-eBIC/sum(eBIC),4)

#Obvious to see that fit3 is better
```

```
> BIC<- data.frame(fit2=extractAIC(fit2,k=log(n))[2],fit3=extractAIC(fit3,k=log(n))[2])
> eBIC<-exp(-.5*(BIC-min(BIC)))
> round(probs<-eBIC/sum(eBIC),4)
  fit2 fit3
1    0    1
```

Hence, the probability of the fit3 to be better than fit 2 is 100%.

Making prediction with fit3

```
p2<-predict(fit3,test_nw2,interval='prediction')
write.csv(exp(p2[,1]),'p2.csv')
```

Got RMSE value of 0.12681 on testing data

Sujal_sb2.csv 2 days ago by Sujal Bhavsar After applying BIC. Reduced model	0.12681

5. Interaction with step

```
fit4<- step(fit3, scope=~.+.^2, direction="forward", k=log(n), trace=0)
summary(fit4)
```

Interaction with step was checked considering fit3 as a base model. However, fit4 shows no improvements with consideration on interaction. It was found that non of the interaction except few is statistically significant.

```
KitchenAbvGr3                  1.985e-01  1.622e-01   1.224 0.221119    
FullBath                       3.290e-02  8.984e-03   3.662 0.000260 ***
CentralAirY:firstFlrSF        -1.593e-01  4.041e-02  -3.941 8.52e-05 ***
OverallCond:BsmtExposureGd    -5.620e-02  1.306e-02  -4.302 1.81e-05 ***
OverallCond:BsmtExposureMn     4.590e-03  1.165e-02   0.394 0.693518    
OverallCond:BsmtExposureNo    -2.120e-03  8.671e-03  -0.245 0.806877    
TotalBsmtSF:MSZoningFV        -8.764e-04  1.732e-04  -5.060 4.76e-07 ***
TotalBsmtSF:MSZoningRH        -8.993e-04  1.871e-04  -4.806 1.71e-06 ***
TotalBsmtSF:MSZoningRL        -9.248e-04  1.674e-04  -5.525 3.94e-08 ***
TotalBsmtSF:MSZoningRM        -8.420e-04  1.683e-04  -5.004 6.36e-07 ***
firstFlrSF:secondFlrSF        -8.781e-05  2.665e-05  -3.295 0.001010 ** 
BsmtUnfSF:ScreenPorch          4.186e-07  1.245e-07   3.361 0.000797 ***
GarageArea:KitchenAbvGr1       1.079e-03  4.159e-04   2.595 0.009559 ** 
GarageArea:KitchenAbvGr2       9.220e-04  4.189e-04   2.201 0.027910 *  
GarageArea:KitchenAbvGr3              NA         NA      NA       NA    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1037 on 1365 degrees of freedom
Multiple R-squared:  0.9369,	Adjusted R-squared:  0.9327 
F-statistic: 222.8 on 91 and 1365 DF,  p-value: < 2.2e-16
```

```
Call:
lm(formula = logSalePrice ~ OverallQual + GrLivArea + Neighborhood + 
    TotalBsmtSF + OverallCond + BsmtUnfSF + GarageArea + SaleCondition + 
    CentralAir + LotArea + Foundation + MSZoning + GarageFinish + 
    KitchenQual + ScreenPorch + Functional + BsmtQual + HalfBath + 
    firstFlrSF + secondFlrSF + BsmtExposure + KitchenAbvGr + 
    FullBath + CentralAir:firstFlrSF + OverallCond:BsmtExposure + 
    TotalBsmtSF:MSZoning + firstFlrSF:secondFlrSF + BsmtUnfSF:ScreenPorch + 
    GarageArea:KitchenAbvGr, data = train_nw)
```

```
R-squared:  0.9327
```

Model probability check for fit4

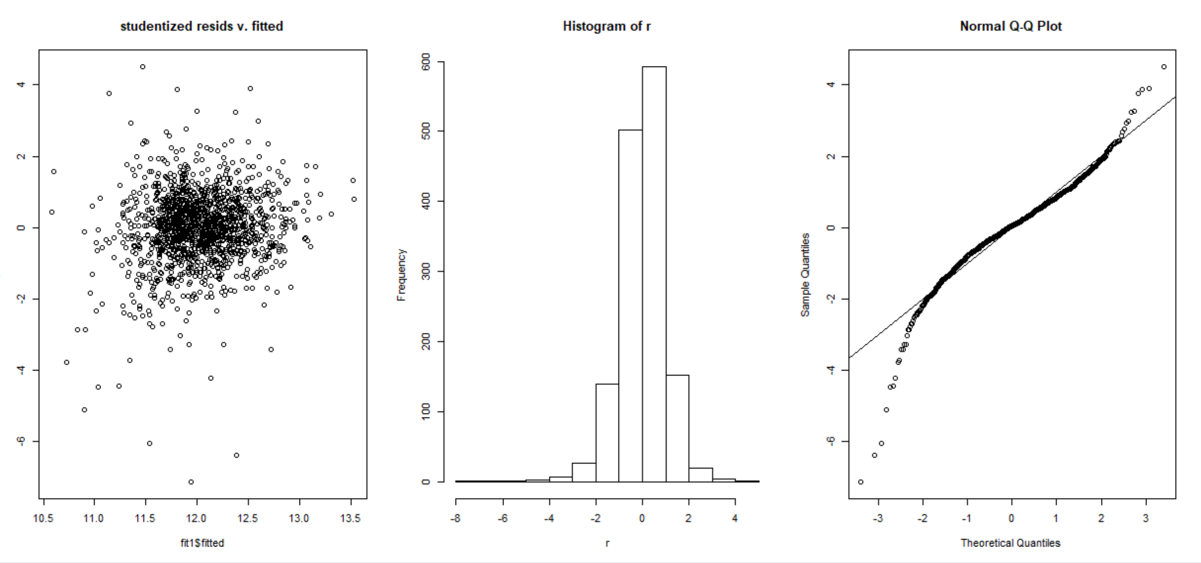
```
# Model Probabilities
BIC<- data.frame(fit3=extractAIC(fit3,k=log(n))[2],fit4=extractAIC(fit4,k=log(n))[2])
eBIC<-exp(-.5*(BIC-min(BIC)))
round(probs<-eBIC/sum(eBIC),4)
```

```
> round(probs<-eBIC/sum(eBIC),4)
  fit3 fit4
1    0    1
> 
```

It is seen that fit4 is better than fit3. However, its performance on the testing set is not improved over fit3.

Performing residual diagnosis

```
# again fit4 seems promising let us check its residual diagnostic

r <- rstudent(fit4)
par(mfrow=c(1,3))
plot(fit1$fitted, r, main="studentized resids v. fitted")
hist(r)
qqnorm(r); abline(0,1)
# Nothing much concerning on residual diagnostic
```

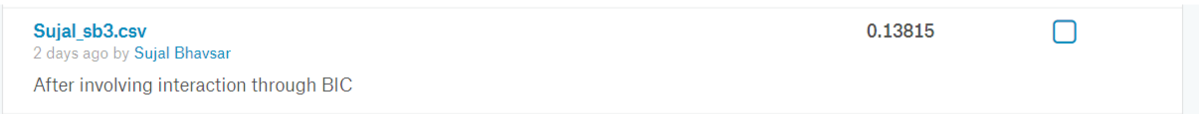


Nothing to worry about.

Prediction on testing set:

```
p4<-predict(fit4,test_nw2,interval='prediction')
write.csv(exp(p4[,1]),'p4.csv')
```

Got RMSE of 0.13815



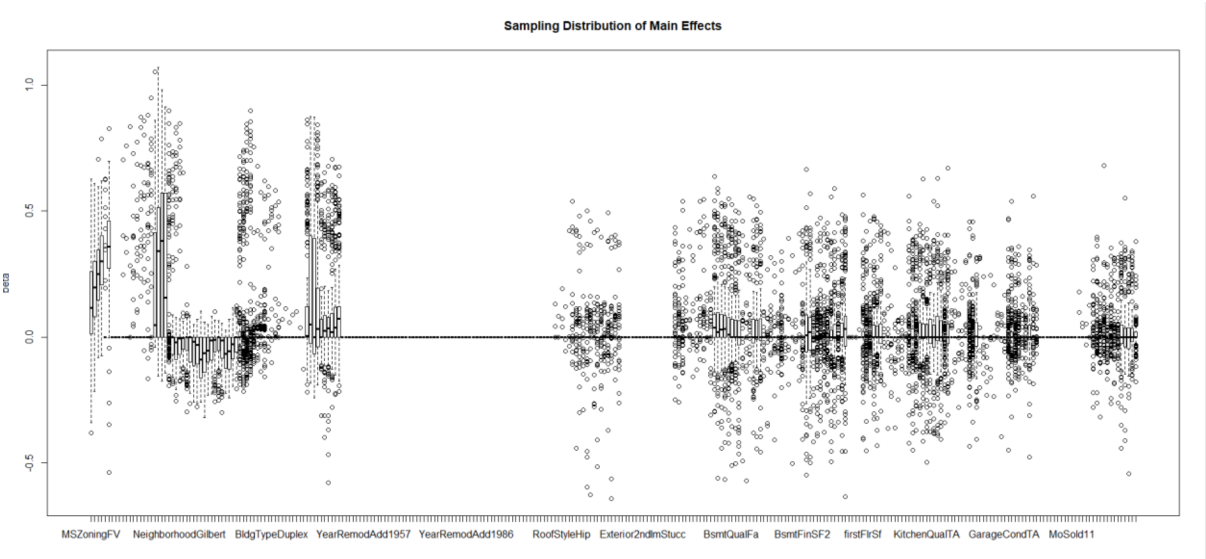
6. Step withing Bootstrap

For storing the values of particular parameters in the beta matrix, I need to have all dummy variables assigned as a column name to my beta matrix. To handle this, I have made a separate CSV file that contains the name of all parameters including dummies. [Columns name with dummy considerations]

```
bb<-read.csv('xx.csv')
B<-199
beta <- matrix(0,296, nrow=B)
colnames(beta)<-bb[,2]
# train_nw3<-as.matrix(train_nw1)
# fit5 <- lars(train_nw3[,1:62], train_nw3[,63], type="lasso")
# lassofit

for(b in 1:B) {
  index <- sample(1:nrow(train_nw1), nrow(train_nw1), replace=TRUE)
  XYT <- data.frame(SalePrice=train_nw1[index,63], train_nw1[index,1:62])
  null <- lm(SalePrice~1, data=XYT)
  full <- lm(SalePrice~., data=XYT)
  fwdbak <- step(null, scope=formula(full), trace=0, k=log(nrow(XYT)))
  beta[b,which(colnames(beta) %in% names(coef(fwdbak)[-1]))] <- coef(fwdbak)[-1]
}

par(mfrow=c(1,1))
boxplot(beta, xaxt="n", ylab="beta", main="Sampling Distribution of Main Effects")
axis(1, at=1:ncol(beta), labels=colnames(beta))
```

Since there are around ~300 parameters after including dummy one, It is extremely difficult to observe the boxplot of betas.

But after looking at the boxplot, it felt that 25% of the betas are centered around zero.



```
##IT is very difficult to observe this variation

ps <- apply(beta, 2, function(x) { mean(x != 0)} )
ps
#overqual selected 100% of the time
```

It was observed that the ‘Overqual’ parameter was selected 100% of the time.

7. Lasso and Ridge Regression

Converted all non-numerical category to a numerical one (working with combined testing and training dataset):

```
# converting to numeric categoy

mapping = c('Ex'=5,'Gd'=4,'TA'=3,'Fa'=2,'Po'=1,'None'=0)
ordinal_var = c('BsmtQual', 'BsmtCond', 'GarageQual', 'GarageCond', 'ExterQual', 'ExterCond', 'HeatingQC', 'KitchenQual')
combi_2<-combi_1
for (v in ordinal_var){
  combi_2[v] = c('Ex'=5,'Gd'=4,'TA'=3,'Fa'=2,'Po'=1,'None'=0)[combi_1[v][[1]]]
}
combi_2['BsmtFinType1'] = c('GLQ'=6,'ALQ'=5,'BLQ'=4,'Rec'=3,'LwQ'=2,'Unf'=1,'None'=0)[combi_1['BsmtFinType1'][[1]]]
combi_2['BsmtFinType2'] = c('GLQ'=6,'ALQ'=5,'BLQ'=4,'Rec'=3,'LwQ'=2,'Unf'=1,'None'=0)[combi_1['BsmtFinType2'][[1]]]
combi_2['Functional'] = c('Typ'=7,'Min1'=6,'Min2'=5,'Mod'=4,'Maj1'=3,'Maj2'=2,'Sev'=1,'Sal'=0)[combi_1['Functional'][[1]]]
combi_2['BsmtExposure'] = c('Gd'=4,'Av'=3,'Mn'=2,'No'=1,'None'=0)[combi_1['BsmtExposure'][[1]]]
combi_2['GarageFinish'] = c('None'=0, 'Unf'=1, 'RFn'=2, 'Fin'=3)[combi_1['GarageFinish'][[1]]]
combi_2['LandSlope'] = c('Sev'=1, 'Mod'=2, 'Gtl'=3)[combi_1['LandSlope'][[1]]]
combi_2['LotShape'] = c('IR3'=1, 'IR2'=2, 'IR1'=3, 'Reg'=4)[combi_1['LotShape'][[1]]]
combi_2['PavedDrive'] = c('N'=1, 'P'=2, 'Y'=3)[combi_1['PavedDrive'][[1]]]
combi_2['CentralAir'] = c('N'=0, 'Y'=1)[combi_1['CentralAir'][[1]]]

combi_2[combi_2 == 'None'] = 0
```

Some of the parameters which were not possible to make numeric are made a dummy variable.

```
# # dummy variables
combi_2 = combi_2 %>%
  as.data.frame() %>%
  fastDummies::dummy_cols() %>%
  .[colnames(.)[sapply(., class) != "character"]]
```

Standardization of values

```
for (v in colnames(combi_2)){
  combi_2[v] = (combi_2[v] - mean(combi_2[v][[1]])) / sd(combi_2[v][[1]])
}
```

Converting NaN to zero

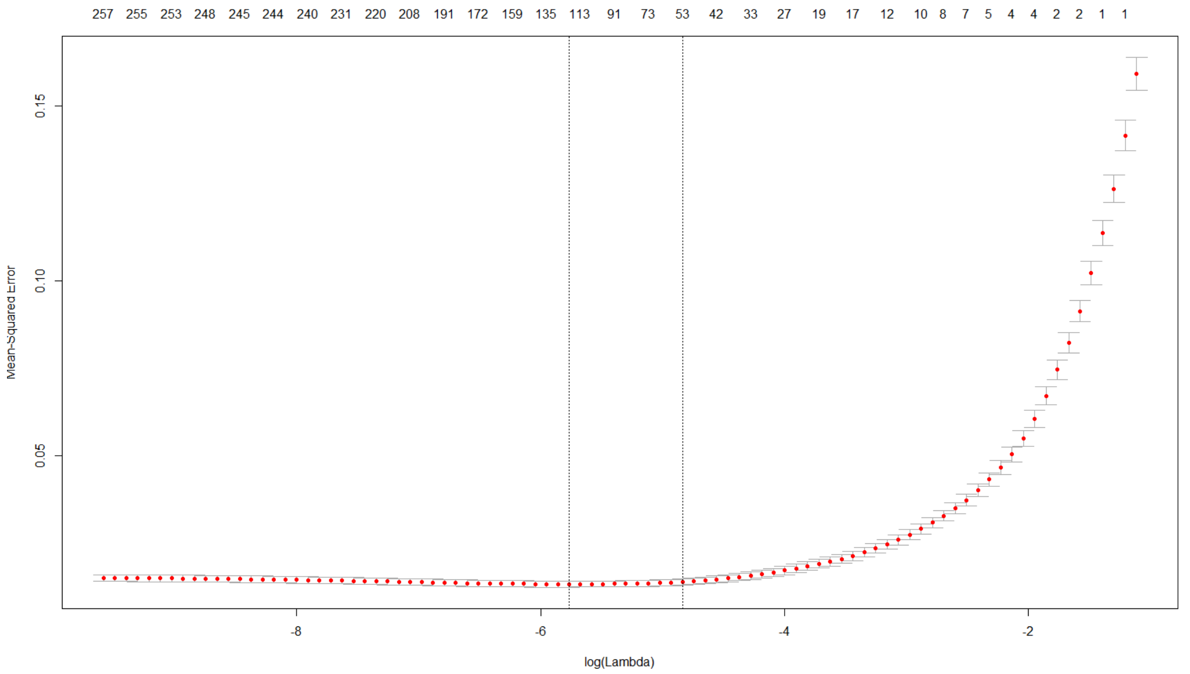
```
combi_2[is.na(combi_2)]<-0
```

Splitting into training and testing set

```
train_nw4<-combi_2[1:1457,]   #(this is only parameters)
test_nw4<- combi_2[1458:2916,]
```

Min. value so λ is selected using the cv.glmnet function. And lasso model is trained using that λ value

```
cv.lasso1=cv.glmnet(as.matrix(train_nw4),train_nw1[,63],alpha=1,family='gaussian')
plot(cv.lasso1)
```



Best lambda for LASSO: 0.003133

Model training with min_lambda value

```
#FInal model with lasso lambda min
lasso.model1 = glmnet(as.matrix(train_nw4),train_nw1[,63], family = "gaussian", lambda = cv.lasso1$lambda.min)
```

Training RMSE of LASSO: 0.101391.

Making prediction from lasso model

```
lasso_pre_tr = lasso.model %>% predict(newx = as.matrix(train_nw3[,-63]))
```

```
lasso_pre_te = lasso.model1 %>% predict(newx = as.matrix(test_nw4))
#write.csv(exp(lasso_pre_te),'p5.csv')
```

Got RMSE on testing set: 0.12740

Sujal_sb4.csv 0.12740

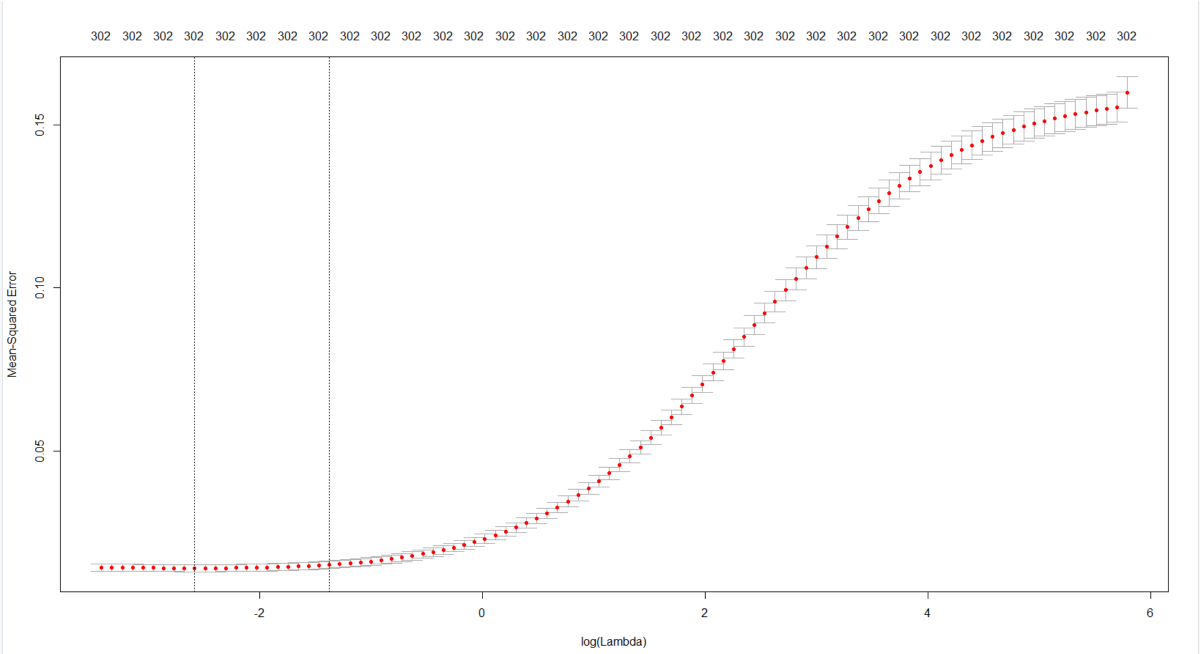
21 hours ago by Sujal Bhavsar

Lasso implementation

Ridge Regression:

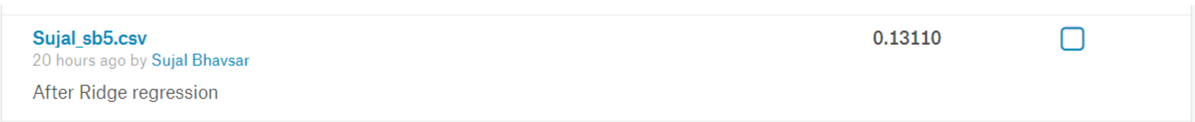
The same procedure is followed for ridge regression with required modifications.

```
cv.ridge=cv.glmnet(as.matrix(train_nw4),train_nw1[,63],alpha=0,family='gaussian')
plot(cv.ridge)
sprintf('Best lambda for LASSO: %f.', cv.ridge$lambda.min)

coef_ridge = coef(cv.ridge, cv.ridge$lambda.min) %>%
  as.matrix() %>%
  as.data.frame()
coef_ridge$abs = abs(coef_ridge[,1])

coef_ridge$abs %>%
  order(coef_ridge$abs, decreasing = TRUE) %>%
  coef_ridge[., ] %>%
  head(20) %>%
  dplyr::select(-'abs')


#FInal model with lasso lambda min
ridge.model = glmnet(as.matrix(train_nw4),train_nw1[,63],alpha = 0, family = "gaussian", lambda = cv.ridge$lambda.min)

ridge_pre_tr = ridge.model %>% predict(newx = as.matrix(train_nw4))
sprintf('Training RMSE of LASSO: %f.', sqrt(mean((train_nw1[,63] - ridge_pre_tr)^2))) # training RMSE

ridge_pre_te = ridge.model %>% predict(newx = as.matrix(test_nw4))
#write.csv(exp(ridge_pre_te),'p6.csv')
```



Best lambda for LASSO: 0.075820

Got RMSE on testing set: 0.13110



Conclusion

Fit 3, which is a reduced model using step with BIC, provided higher predictability than any other model. It has the lowest RMSE amongst the selected model (0.1268). Lasso seems to have improved the predictability over full model but not as much as fit 3. The selection of the right features and making a suitable transformation on those features can have a higher chance of making our model accurate than merely applying regularization on the full model.