BITTIGER

BITTIGER

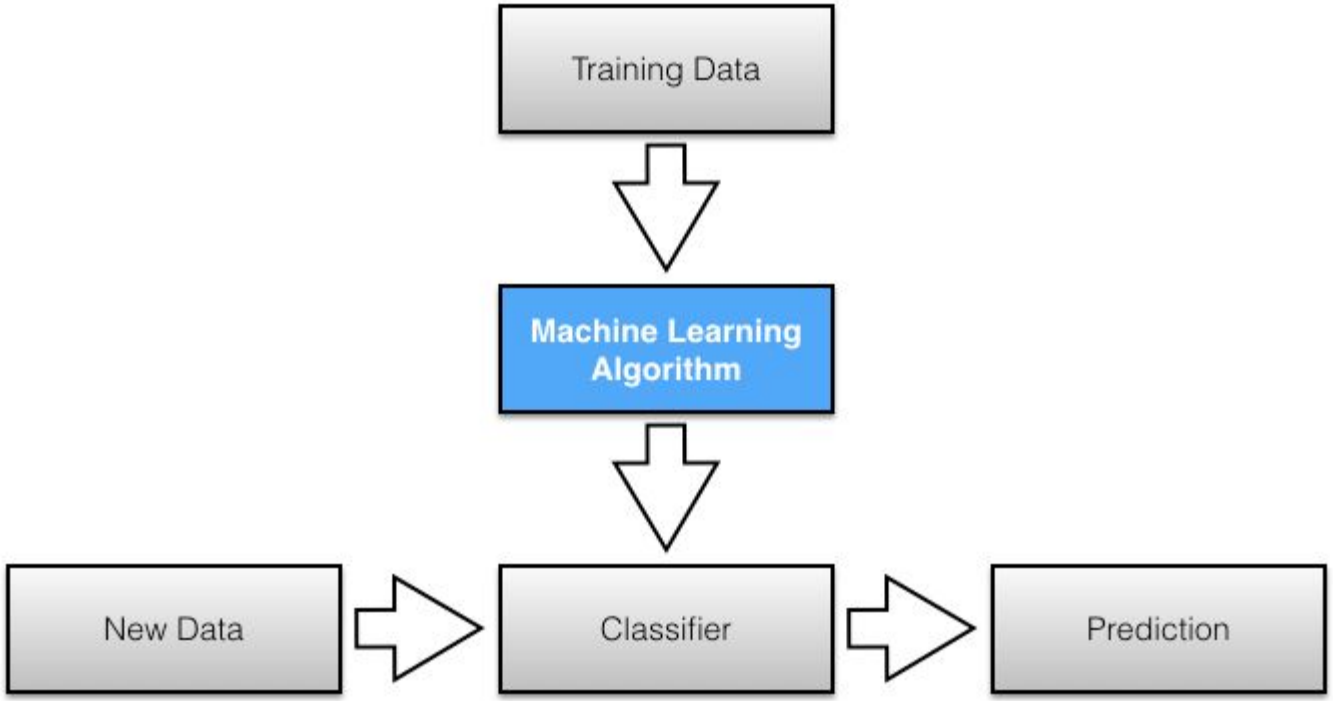
2/10 Predictive Modeling for House Price & Analytics in R

Joanne

Predictive Modeling

Supervised Learning:

build a model that makes predictions based on evidence in the presence of uncertainty.



Agenda

Regression Models & Prediction

EDA & Imputation

Linear Regression (线性回归模型)

- Multiple Linear Regression(多元线性回归)
- Model Diagnostics for Linear Regression(模型诊断)
- Interaction Terms (交互项)
- Non-linear Transformations(非线性转换)

Linear Model Selection and Regularization (变量选择和正则化)

-Best Subset/Stepwise

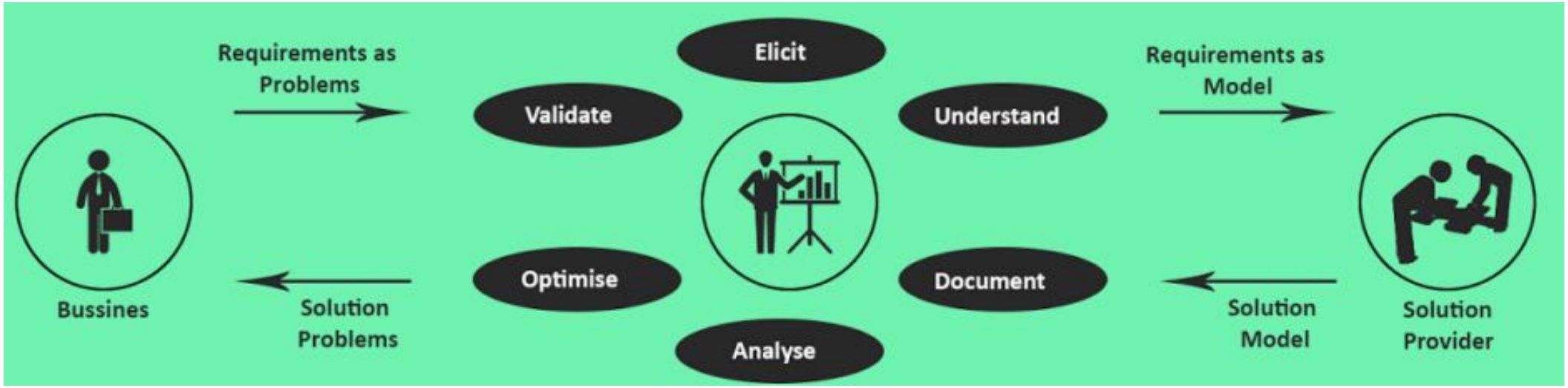
-LASSO

-RIDGE

***Regression Tree**

***Binary Logistic Regression** (逻辑回归)

Project-Process Flow



Glance through Data

“There are no routine statistical questions, only questionable statistical routines.”

SalePrice Prediction-Ames,Iowa

Knowledge • 2,057 teams

House Prices: Advanced Regression Techniques

Tue 30 Aug 2016 — Wed 1 Mar 2017 (3 months to go)

Dashboard

Home
- Data
- Make a submission

Information

Competition Details » Get the Data » Make a submission

Sold! How do home features add up to its price tag?

Import Data

```
setwd("D:/.../model")
data=read.csv("train.csv",header=T,na.strings = "NA")
data2=read.csv("test.csv",header=T,na.strings = "NA")

# remove ID
data=data[,-c(1)]

summary(data)
str(data)
```

```
> summary(data)
   MSSubClass       MSZoning     LotFrontage      LotArea        Street      Alley
 Min.   : 20.0   C (all):  10   Min.   : 21.00   Min.   :  1300   Grvl:   6   Grvl:  50
 1st Qu.: 20.0   FV     :  65   1st Qu.: 59.00   1st Qu.:  7554   Pave:1454   Pave:  41
 Median : 50.0   RH     :  16   Median : 69.00   Median :  9478               NA's:1369
 Mean   : 56.9   RL     :1151   Mean   : 70.05   Mean   : 10517
 3rd Qu.: 70.0   RM     : 218   3rd Qu.: 80.00   3rd Qu.: 11602
 Max.   :190.0                  Max.   :313.00   Max.   :215245
                                NA's   :259
```

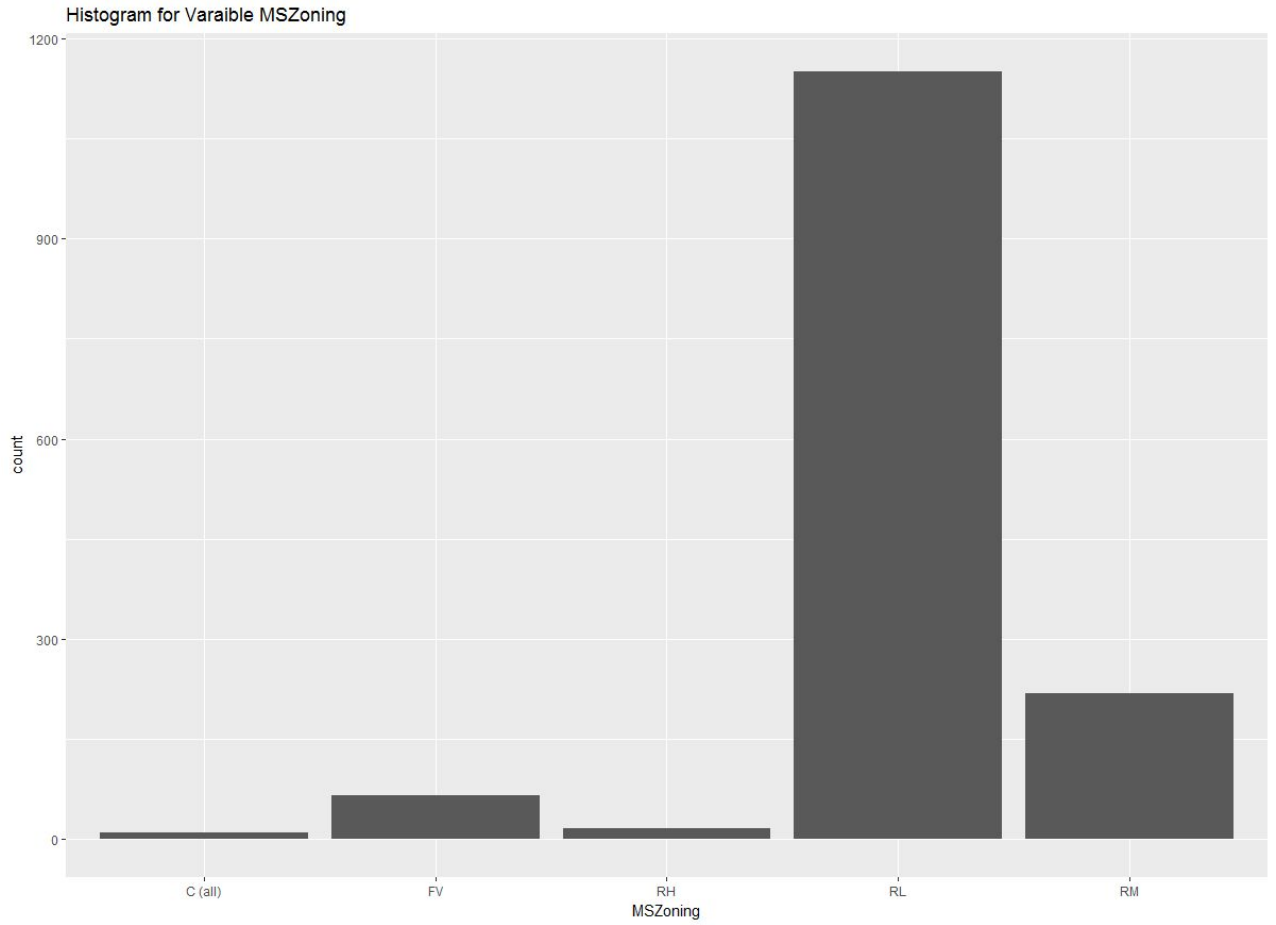
Exploratory Data Analysis

```
                                                                    (other):    9
> str(data)
'data.frame':   1460 obs. of  80 variables:
 $ MSSubClass   : int  60 20 60 70 60 50 20 60 50 190 ...
 $ MSZoning     : Factor w/ 5 levels "C (all)","FV",..: 4 4 4 4 4 4 4 4 5 4 ...
 $ LotFrontage  : int  65 80 68 60 84 85 75 NA 51 50 ...
 $ LotArea      : int  8450 9600 11250 9550 14260 14115 10084 10382 6120 7420 ...
 $ Street       : Factor w/ 2 levels "Grvl","Pave": 2 2 2 2 2 2 2 2 2 2 ...
 $ Alley        : Factor w/ 2 levels "Grvl","Pave": NA NA NA NA NA NA NA NA NA NA ...
 $ LotShape     : Factor w/ 4 levels "IR1","IR2","IR3",..: 4 4 1 1 1 1 4 1 4 4 ...
 $ LandContour  : Factor w/ 4 levels "Bnk","HLS","Low",..: 4 4 4 4 4 4 4 4 4 4 ...
 $ Utilities    : Factor w/ 2 levels "AllPub","NoSeWa": 1 1 1 1 1 1 1 1 1 1 ...
 $ LotConfig    : Factor w/ 5 levels "Corner","CulDSac",..: 5 3 5 1 3 5 5 1 5 1 ...
 $ LandSlope    : Factor w/ 3 levels "Gtl","Mod","Sev": 1 1 1 1 1 1 1 1 1 1 ...
 $ Neighborhood : Factor w/ 25 levels "Blmngtn","Blueste",..: 6 25 6 7 14 12 21 17 18 4 ...
 $ Condition1   : Factor w/ 9 levels "Artery","Feedr",..: 3 2 3 3 3 3 3 5 1 1 ...
 $ Condition2   : Factor w/ 8 levels "Artery","Feedr",..: 3 3 3 3 3 3 3 3 3 1 ...
 $ BldgType     : Factor w/ 5 levels "1Fam","2fmCon",..: 1 1 1 1 1 1 1 1 1 2 ...
```

optional: data$MSSubClass=as.factor(data$MSSubClass)

Exploratory Data Analysis

```
> ggplot(data = data) +
  geom_bar(mapping = aes(x = MSZoning )) # bar for
categorical
+   ggtitle("Histogram for Varaible MSZoning")
```

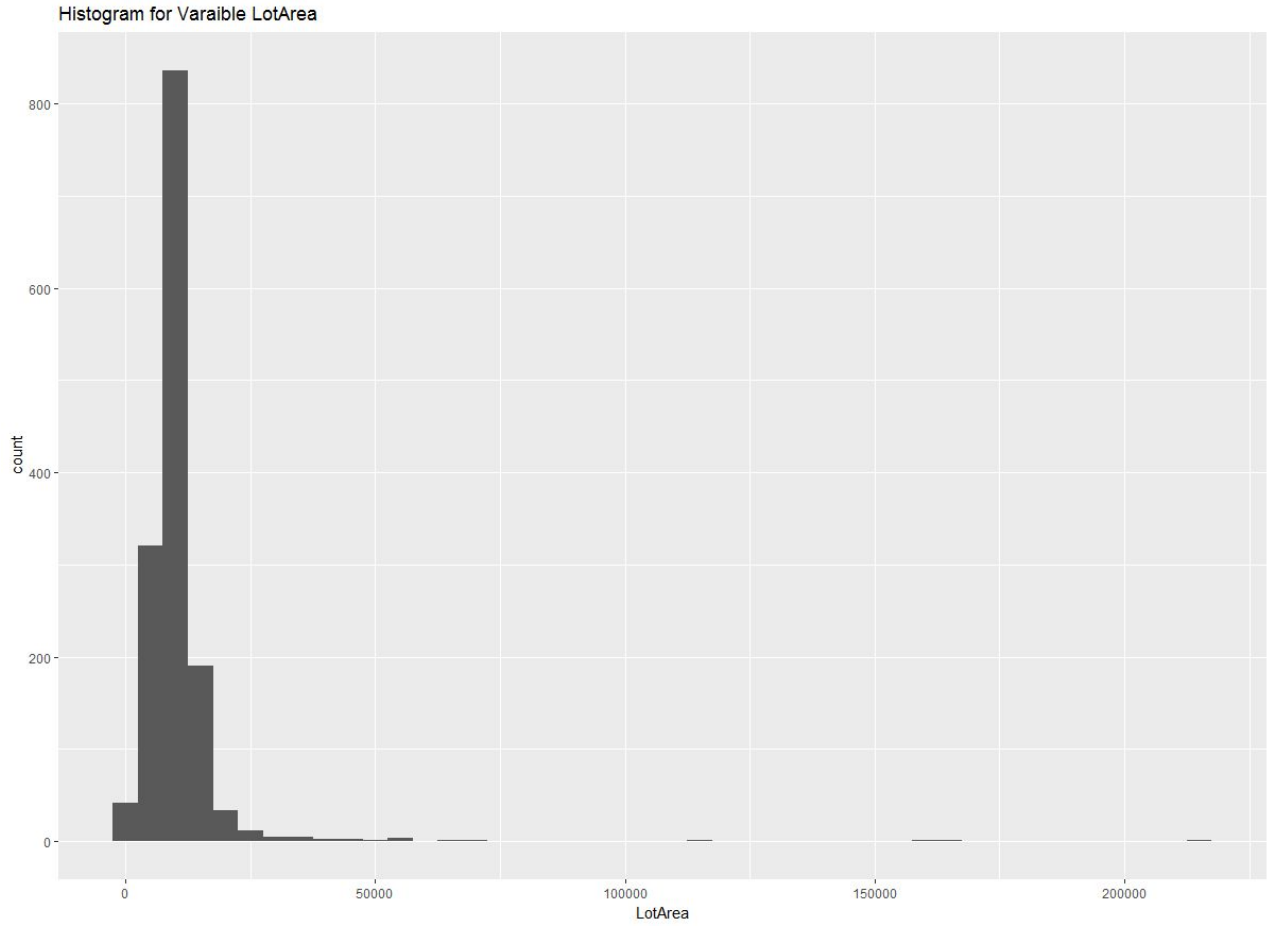



```
MSZoning: Identifies the general zoning classification of the sale.

       A      Agriculture
       C      Commercial
       FV     Floating Village Residential
       I      Industrial
       RH     Residential High Density
       RL     Residential Low Density
       RP     Residential Low Density Park
       RM     Residential Medium Density
```

Exploratory Data Analysis

```
> summary(data$LotArea) # to determine binwidth
# LotArea: Lot size in square feet

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1300    7554    9478   10520   11600  215200
```

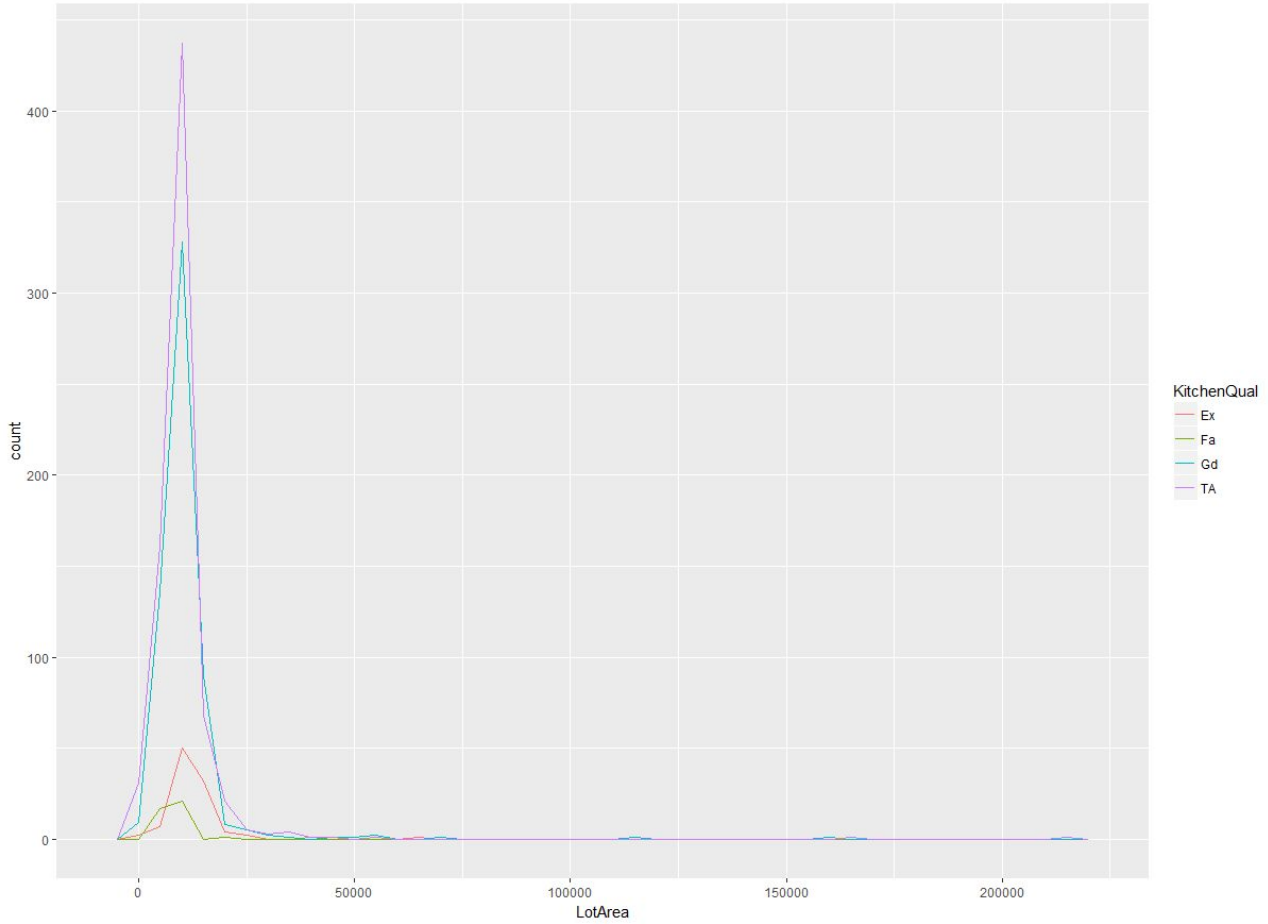
```
> ggplot(data = data) +
geom_histogram(mapping = aes(x = LotArea), binwidth =5000)
# histogram for continuous
+    ggtitle("Histogram for Varaible LotArea")
```



Exploratory Data Analysis

```
# visualize a categorical and a continuous variable
ggplot(data = data, mapping = aes(x = LotArea, colour =
KitchenQual)) +
  geom_freqpoly(binwidth = 5000)
```

Ex Excellent Gd Good TA Average/Typical Fa Fair



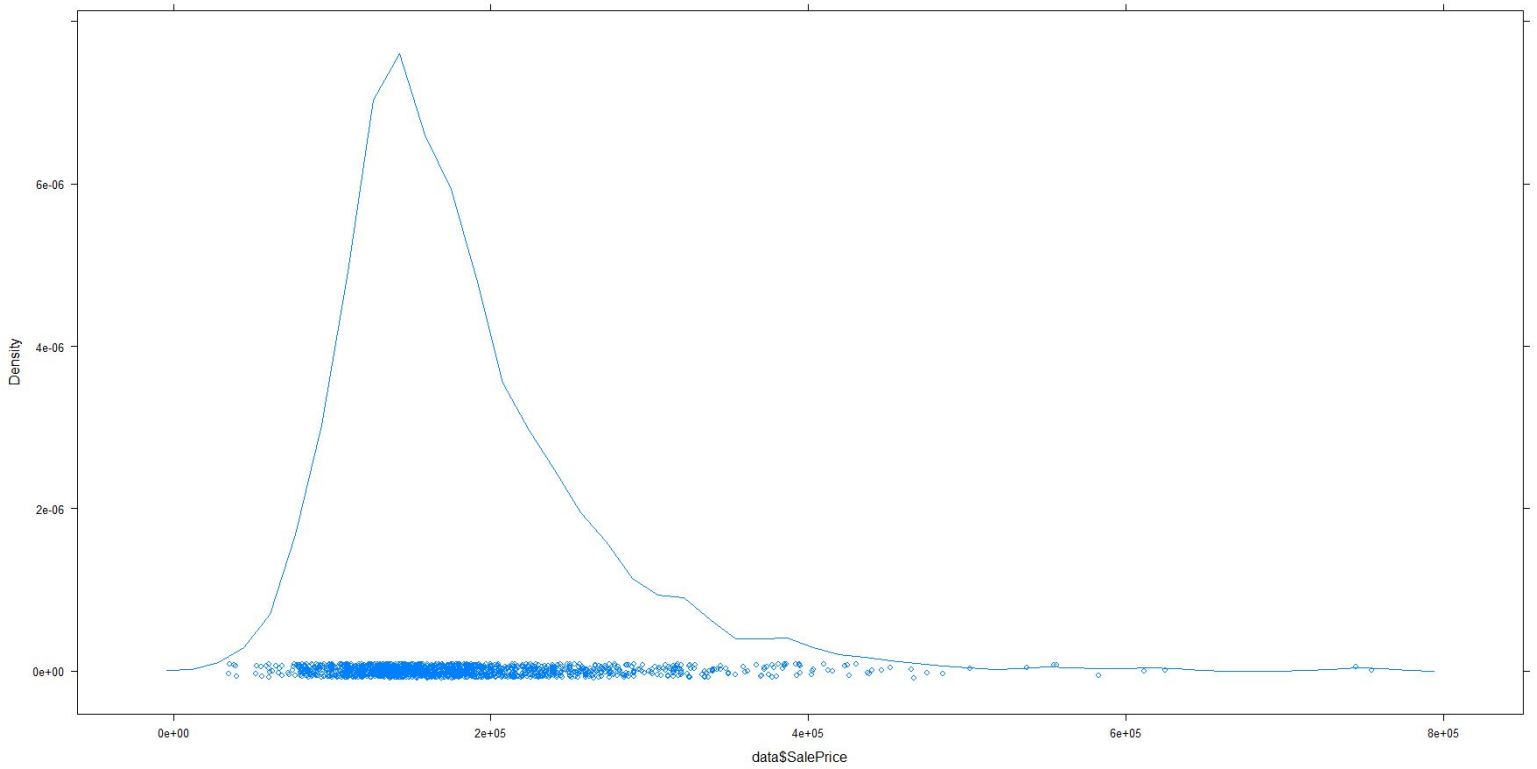
Exploratory Data Analysis- dplyr

```
> library(dplyr)
> data %>% count(MSZoning)
# A tibble: 5 × 2
  MSZoning     n
    <fctr> <int>
1  C (all)    10
2       FV    65
3       RH    16
4       RL  1151
5       RM   218
```

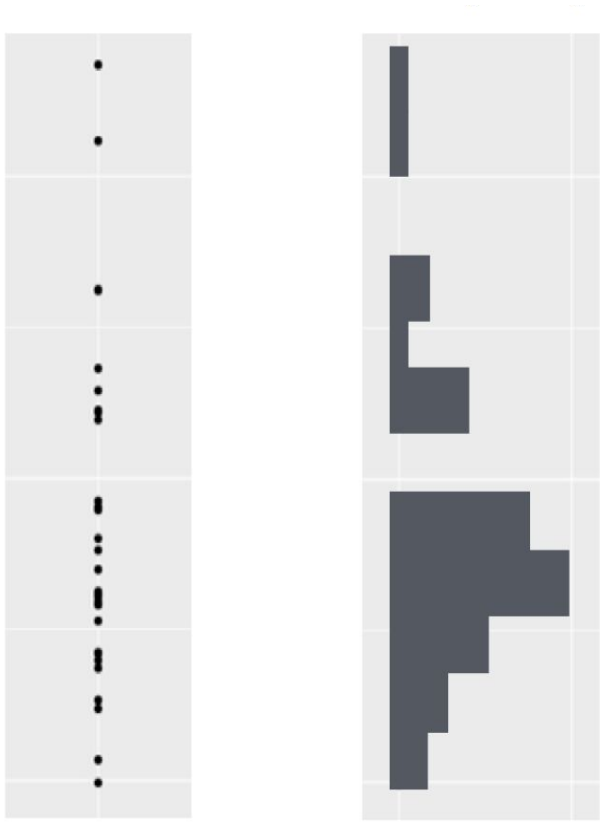
```
> data %>% count(cut_width(LotArea, 5000))
# A tibble: 18 × 2
   `cut_width(LotArea, 5000)`     n
                        <fctr> <int>
1          [-2.5e+03,2.5e+03]    42
2           (2.5e+03,7.5e+03]   321
3           (7.5e+03,1.25e+04]   836
4          (1.25e+04,1.75e+04]   190
5          (1.75e+04,2.25e+04]    34
6          (2.25e+04,2.75e+04]    12
7          (2.75e+04,3.25e+04]     5
8          (3.25e+04,3.75e+04]     5
9          (3.75e+04,4.25e+04]     2
10         (4.25e+04,4.75e+04]     2
11         (4.75e+04,5.25e+04]     1
12         (5.25e+04,5.75e+04]     4
13         (6.25e+04,6.75e+04]     1
14         (6.75e+04,7.25e+04]     1
15         (1.12e+05,1.18e+05]     1
16         (1.58e+05,1.62e+05]     1
17         (1.62e+05,1.68e+05]     1
18         (2.12e+05,2.18e+05]     1
```

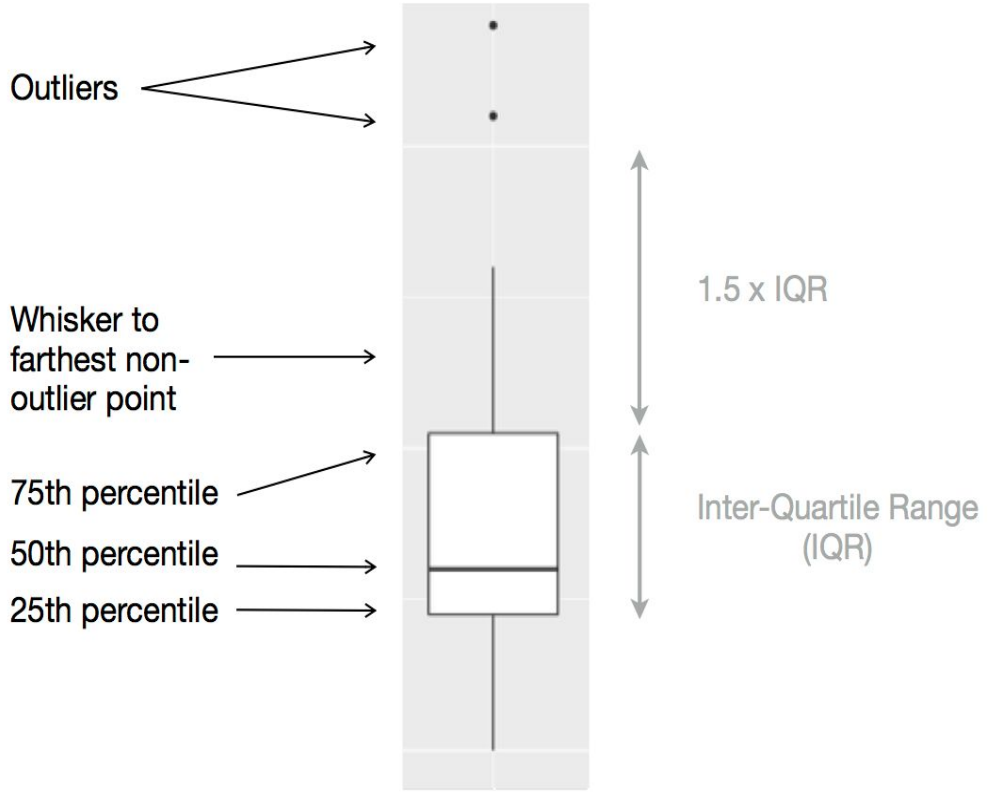
Better Understand Y variable

```
library(lattice)
densityplot(data$SalePrice)
```



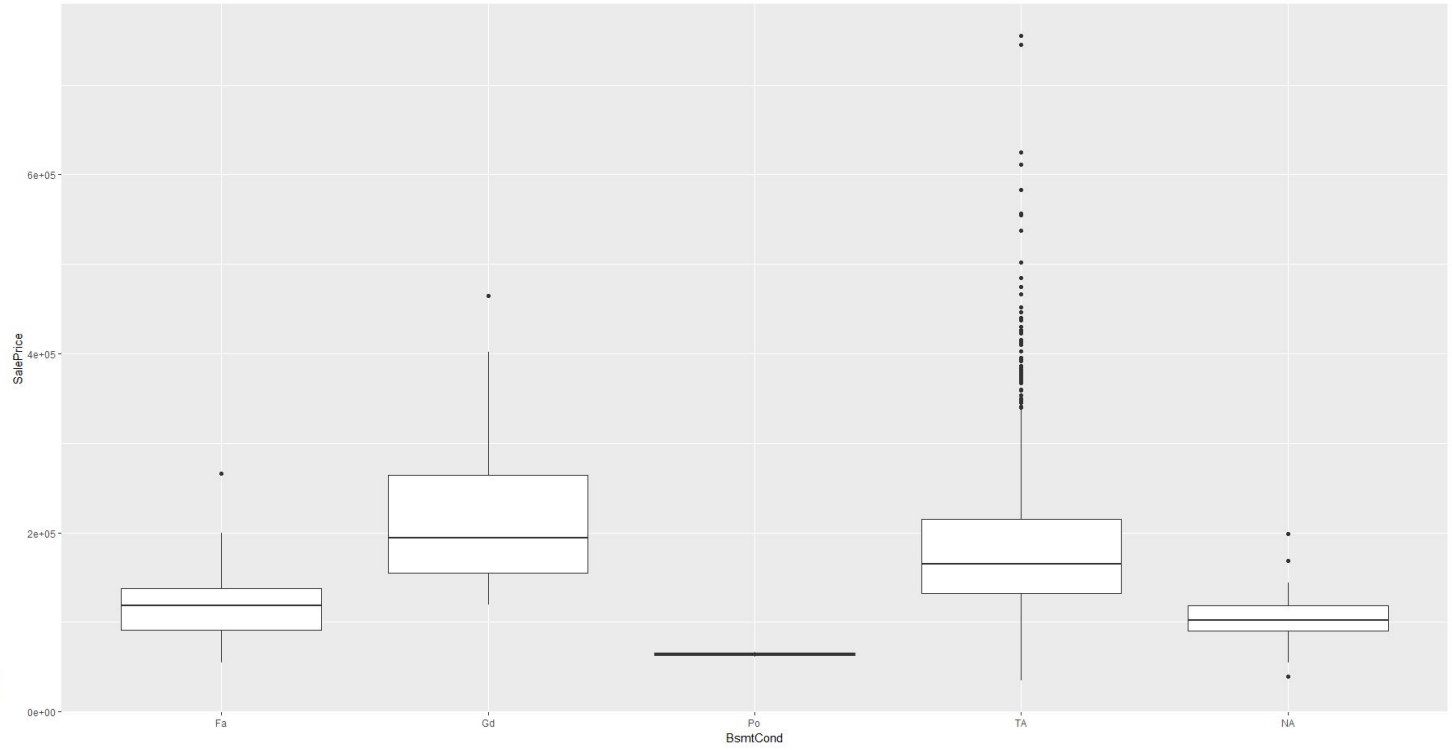
BoxPlot




Outliers
Whisker to farthest non-outlier point
75th percentile
50th percentile
25th percentile
1.5 x IQR
Inter-Quartile Range (IQR)

Exploratory Data Analysis

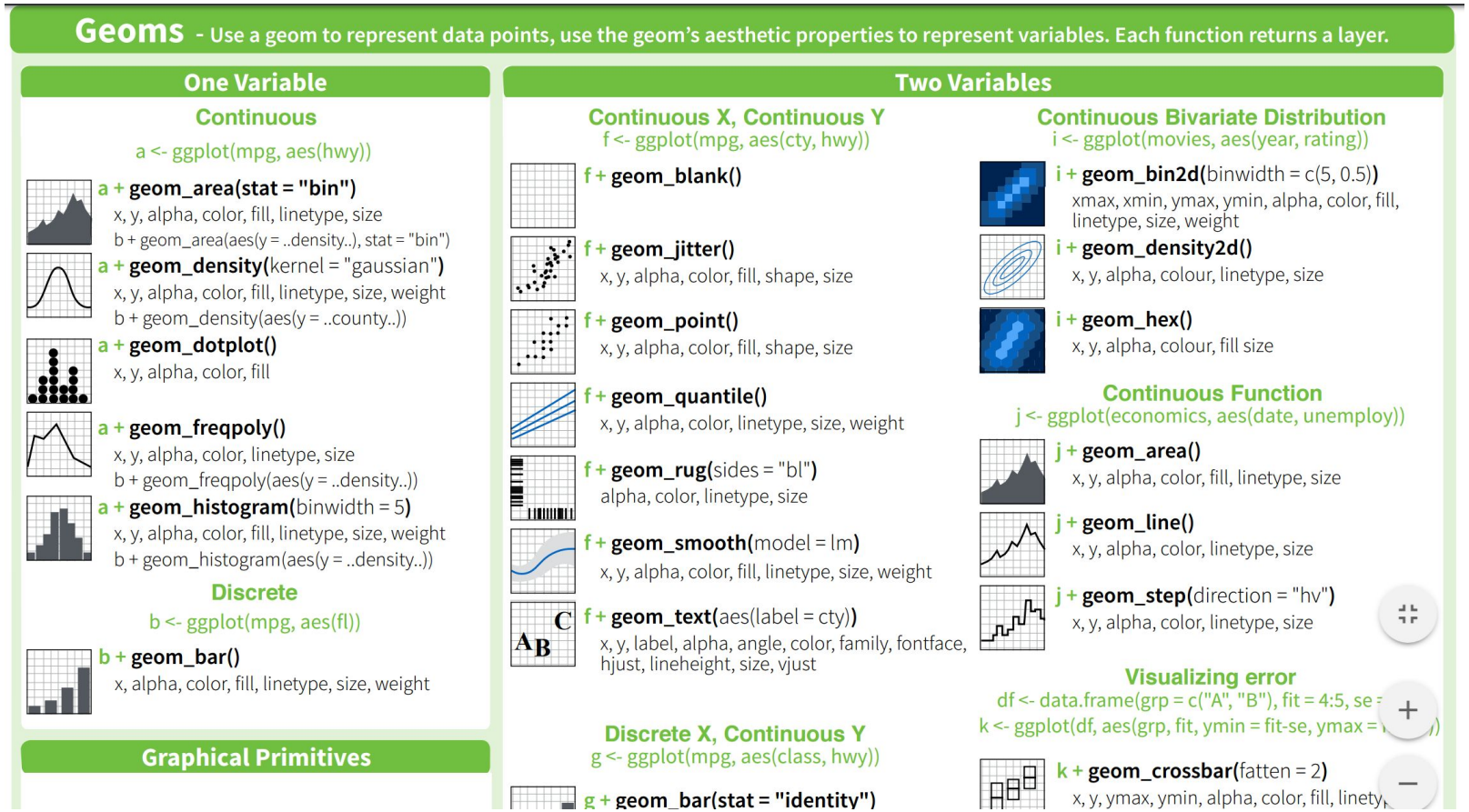
```
ggplot(data = data, mapping = aes(x =
BsmtCond , y = SalePrice)) +
  geom_boxplot()
```



BsmtCond: Evaluates the general condition of the basement

Ex	Excellent
Gd	Good
TA	Typical - slight dampness allowed
Fa	Fair - dampness or some cracking or settling
Po	Poor - Severe cracking, settling, or wetness
NA	No Basement

R for Data Visualization: ggplot2



Geoms - Use a geom to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

One Variable

Continuous

a <- ggplot(mpg, aes(hwy))

a + geom_area(stat = "bin")
x, y, alpha, color, fill, linetype, size
b + geom_area(aes(y = ..density..), stat = "bin")

a + geom_density(kernel = "gaussian"**)**
x, y, alpha, color, fill, linetype, size, weight
b + geom_density(aes(y = ..county..))

a + geom_dotplot()
x, y, alpha, color, fill

a + geom_freqpoly()
x, y, alpha, color, linetype, size
b + geom_freqpoly(aes(y = ..density..))

a + geom_histogram(binwidth = 5**)**
x, y, alpha, color, fill, linetype, size, weight
b + geom_histogram(aes(y = ..density..))

Discrete

b <- ggplot(mpg, aes(fl))

b + geom_bar()
x, alpha, color, fill, linetype, size, weight

Graphical Primitives

Two Variables

Continuous X, Continuous Y

f <- ggplot(mpg, aes(cty, hwy))

f + geom_blank()

f + geom_jitter()
x, y, alpha, color, fill, shape, size

f + geom_point()
x, y, alpha, color, fill, shape, size

f + geom_quantile()
x, y, alpha, color, linetype, size, weight

f + geom_rug(sides = "bl"**)**
alpha, color, linetype, size

f + geom_smooth(model = lm**)**
x, y, alpha, color, fill, linetype, size, weight

f + geom_text(aes(label = cty)**)**
x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

Discrete X, Continuous Y

g <- ggplot(mpg, aes(class, hwy))

g + geom_bar(stat = "identity")

Continuous Bivariate Distribution

i <- ggplot(movies, aes(year, rating))

i + geom_bin2d(binwidth = c(5, 0.5)**)**
xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size, weight

i + geom_density2d()
x, y, alpha, colour, linetype, size

i + geom_hex()
x, y, alpha, colour, fill size

Continuous Function

j <- ggplot(economics, aes(date, unemploy))

j + geom_area()
x, y, alpha, color, fill, linetype, size

j + geom_line()
x, y, alpha, color, linetype, size

j + geom_step(direction = "hv"**)**
x, y, alpha, color, linetype, size

Visualizing error

df <- data.frame(grp = c("A", "B"), fit = 4:5, se =
k <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax =

k + geom_crossbar(fatten = 2**)**
x, y, ymax, ymin, alpha, color, fill, linety

Cheat Sheet:
https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf

Clean Data: Imputation

Check % of Missing Data

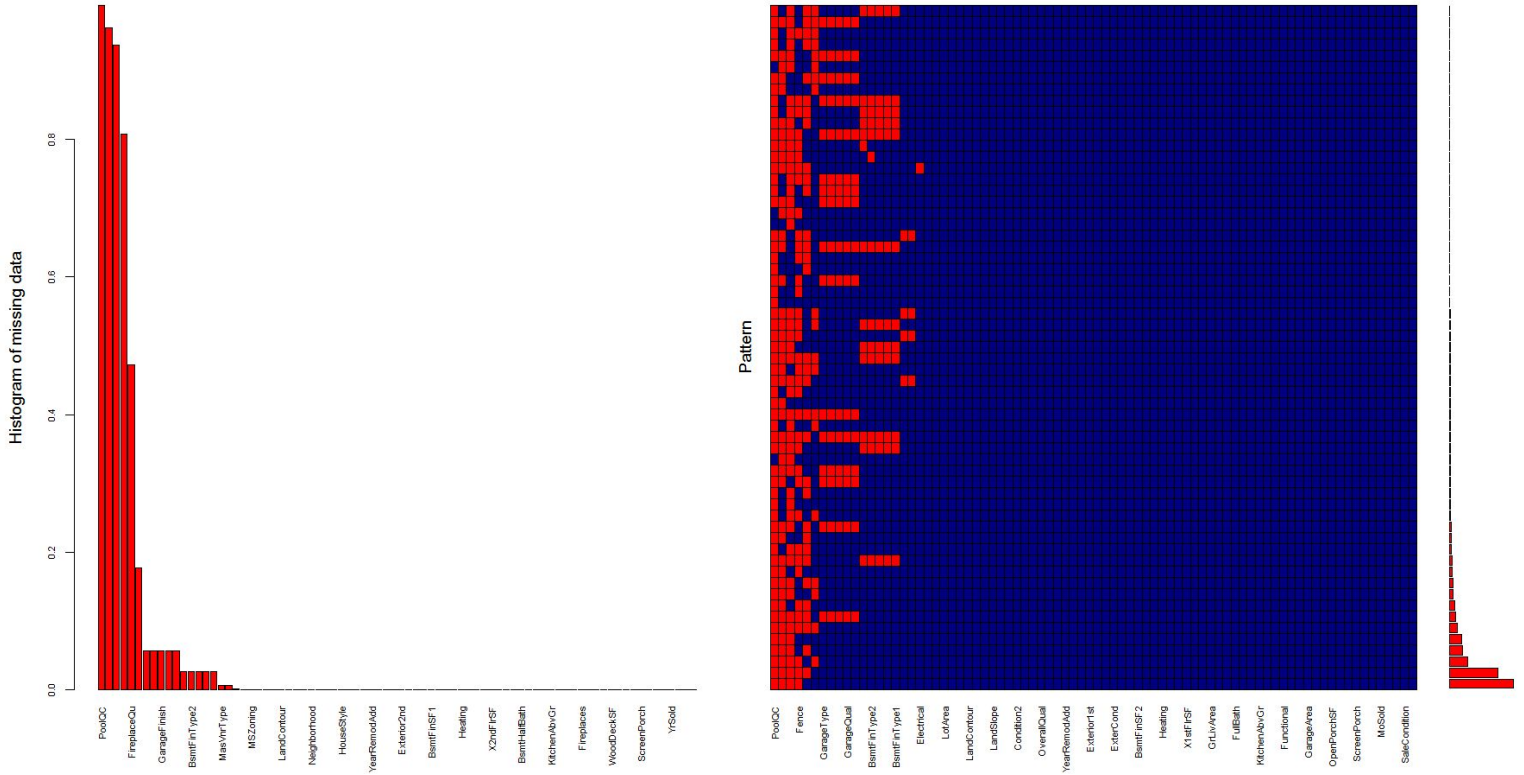
```
> MissingPercentage <- function(x){sum(is.na(x))/length(x)*100}
> sort(apply(data,2,MissingPercentage),decreasing=TRUE)
      PoolQC   MiscFeature         Alley         Fence   FireplaceQu   LotFrontage    GarageType   GarageYrBlt  GarageFinish
99.52054795   96.30136986   93.76712329   80.75342466   47.26027397   17.73972603    5.54794521    5.54794521    5.54794521
  GarageQual    GarageCond  BsmtExposure  BsmtFinType2      BsmtQual      BsmtCond  BsmtFinType1    MasVnrType    MasVnrArea
   5.54794521    5.54794521    2.60273973    2.60273973    2.53424658    2.53424658    2.53424658    0.54794521    0.54794521
  Electrical    MSSubClass      MSZoning       LotArea        Street      LotShape   LandContour     Utilities     LotConfig
   0.06849315    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000
   LandSlope  Neighborhood    Condition1    Condition2      BldgType    HouseStyle   OverallQual   OverallCond     YearBuilt
   0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000
YearRemodAdd     RoofStyle       RoofMatl   Exterior1st   Exterior2nd     ExterQual     ExterCond    Foundation    BsmtFinSF1
   0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000
  BsmtFinSF2     BsmtUnfSF   TotalBsmtSF       Heating     HeatingQC     CentralAir      X1stFlrSF     X2ndFlrSF  LowQualFinSF
   0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000
   GrLivArea  BsmtFullBath  BsmtHalfBath      FullBath      HalfBath  BedroomAbvGr  KitchenAbvGr   KitchenQual  TotRmsAbvGrd
   0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000
  Functional    Fireplaces    GarageCars    GarageArea    PavedDrive    WoodDeckSF   OpenPorchSF EnclosedPorch    X3SsnPorch
   0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000
```

Check # of Missing Data

```
> # check # of NA
> sort(sapply(data, function(x) sum(is.na(x))),decreasing=TRUE)
       PoolQC   MiscFeature         Alley         Fence   FireplaceQu   LotFrontage    GarageType   GarageYrBlt  GarageFinish
         1453          1406          1369          1179           690           259            81            81            81
   GarageQual    GarageCond  BsmtExposure  BsmtFinType2      BsmtQual      BsmtCond  BsmtFinType1    MasVnrType    MasVnrArea
           81            81            38            38            37            37            37             8             8
   Electrical   MSSubClass      MSZoning       LotArea        Street      LotShape   LandContour     Utilities     LotConfig
            1             0             0             0             0             0             0             0             0
    LandSlope  Neighborhood    Condition1    Condition2      BldgType    HouseStyle   OverallQual   OverallCond     YearBuilt
            0             0             0             0             0             0             0             0             0
 YearRemodAdd     RoofStyle      RoofMatl   Exterior1st   Exterior2nd     ExterQual     ExterCond    Foundation    BsmtFinSF1
            0             0             0             0             0             0             0             0             0
   BsmtFinSF2     BsmtUnfSF   TotalBsmtSF       Heating     HeatingQC    CentralAir     X1stFlrSF     X2ndFlrSF  LowQualFinSF
            0             0             0             0             0             0             0             0             0
    GrLivArea  BsmtFullBath  BsmtHalfBath      FullBath      HalfBath  BedroomAbvGr  KitchenAbvGr   KitchenQual  TotRmsAbvGrd
            0             0             0             0             0             0             0             0             0
   Functional    Fireplaces    GarageCars    GarageArea    PavedDrive    WoodDeckSF   OpenPorchSF EnclosedPorch    X3SsnPorch
            0             0             0             0             0             0             0             0             0
  ScreenPorch      PoolArea       MiscVal        MoSold        YrSold      SaleType SaleCondition     SalePrice
            0             0             0             0             0             0             0             0
```

Visualizing Missing Data and Delete

```
library(VIM)
aggr_plot <- aggr(data,
                  col=c('navyblue','red'),
                  numbers=TRUE,
                  sortVars=TRUE,
                  labels=names(data),
                  cex.axis=.7,
                  gap=3,
                  ylab=c("Histogram of missing data","Pattern"))
```



Delete Columns with more than 5% Missig Data and Imputing the Rest

Assumption:

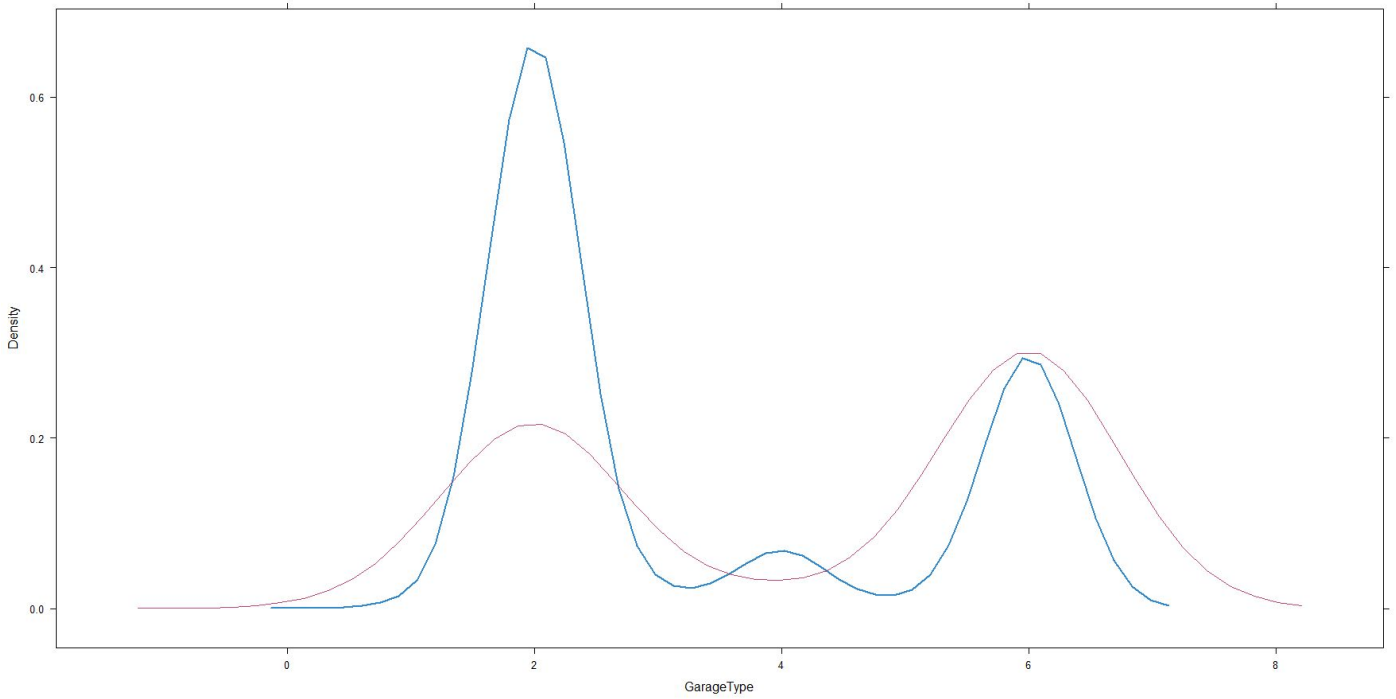
MCAR: missing completely at random.

```
# Delete columns with more than 5% missing data
library(dplyr)
data=select(data,-c(PoolQC,MiscFeature,Alley,Fence,FireplaceQu,LotFrontage))
```

```
# CART: classification and regression trees
library(mice)
imp_data<- mice(data, m=1, method='cart', printFlag=FALSE)
```

Test Result

```
> # Test Original and Imputed
> table(data$GarageType)

 2Types  Attchd Basment BuiltIn CarPort  Detchd
      6     870      19      88       9     387
> table(imp_data$imp$GarageType)

 2Types  Attchd BuiltIn  Detchd
      1      32       3      45
> # vasualize density blue:actual; red:imputed
> densityplot(imp_data, ~GarageType)
```



Imputing Done! Double Check!

```
# Merge to Original Data
data_complete <- complete(imp_data)

#Confirm no NAs
sum(sapply(data_complete, function(x) { sum(is.na(x)) }))

write.csv(data_complete, file = "data_complete.csv")
data_complete=read.csv("data_complete.csv",header=T)
```

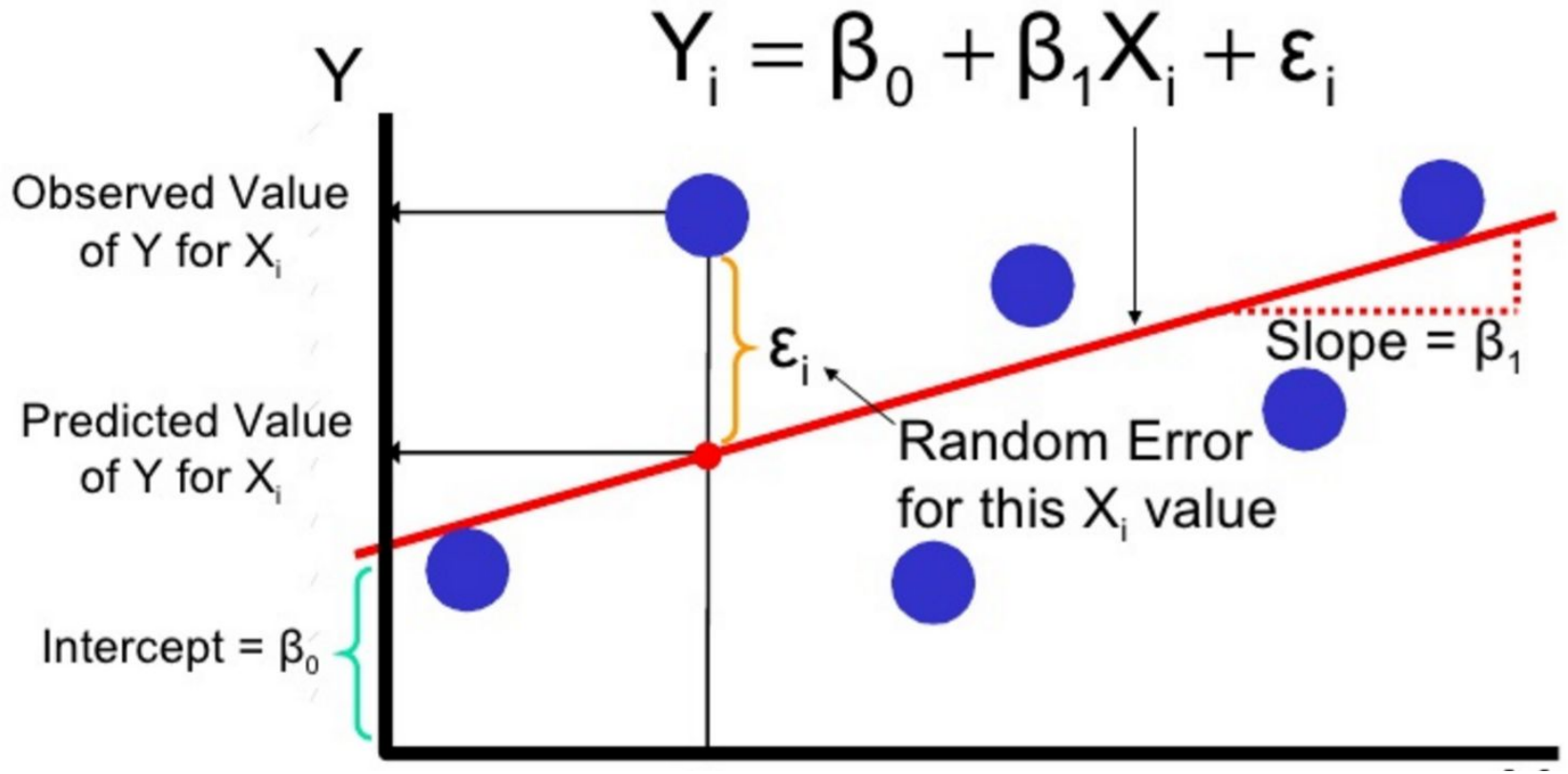
Multiple Linear Regression Model

Training and Testing Set

```
> set.seed(1)
> train = sample(1:nrow(data_complete),nrow(data_complete)/2)
> test = -train
> traindata = data_complete[train,]
> testdata = data_complete[test,]
```

testdata	730 obs. of 74 variables
traindata	730 obs. of 74 variables

Linear Regression



- *x*, is regarded as the **predictor**, **explanatory**, or **independent** variable.
- *y*, is regarded as the **response**, **outcome**, or **dependent** variable.
- Residual: The difference between an observed value of the dependent variable and the value of the dependent variable predicted from the regression line.

Fit A Model...Oops

```
> model=lm(SalePrice~.,data=traindata)
Warning messages:
1: contrasts dropped from factor Condition1 due to missing levels
2: contrasts dropped from factor Condition2 due to missing levels
3: contrasts dropped from factor RoofStyle due to missing levels
4: contrasts dropped from factor RoofMatl due to missing levels
5: contrasts dropped from factor Exterior1st due to missing levels
6: contrasts dropped from factor Exterior2nd due to missing levels
7: contrasts dropped from factor Heating due to missing levels
8: contrasts dropped from factor Functional due to missing levels
> distinct(data,RoofStyle)
  RoofStyle
1     Gable
2       Hip
3   Gambrel
4   Mansard
5      Flat
6      Shed
> distinct(traindata,RoofStyle)
  RoofStyle
1       Hip
2     Gable
3   Mansard
4      Flat
5   Gambrel
```

Understand the Model

```
> summary(model)

Call:
lm(formula = SalePrice ~ ., data = traindata)

Residuals:
    Min      1Q  Median      3Q     Max
-107928   -8614       0    9346  133886

Coefficients: (4 not defined because of singularities)
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)    -1.515e+06  1.534e+06  -0.988 0.323807
MSSubClass     -1.612e+02  1.095e+02  -1.473 0.141443
MSZoning2       3.730e+04  1.775e+04   2.101 0.036150 *
MSZoning3       3.173e+04  1.883e+04   1.685 0.092606 .
MSZoning4       1.821e+04  1.539e+04   1.183 0.237430
MSZoning5       1.388e+04  1.465e+04   0.947 0.343922
LotArea         1.041e+00  1.612e-01   6.457 2.47e-10 ***
Street2         4.276e+04  1.991e+04   2.148 0.032188 *
LotShape2      -5.435e+03  6.616e+03  -0.821 0.411796
LotShape3      -1.367e+04  1.260e+04  -1.085 0.278631
LotShape4       3.276e+02  2.267e+03   0.145 0.885157
GarageCond5            NA         NA      NA       NA
PavedDrive2    -4.131e+03  9.145e+03  -0.452 0.651675
PavedDrive3    -3.067e+03  4.865e+03  -0.630 0.528707
WoodDeckSF      1.125e+01  8.841e+00   1.272 0.203790
OpenPorchSF    -1.118e+01  1.867e+01  -0.599 0.549485
EnclosedPorch  -1.328e+01  1.674e+01  -0.793 0.427997
X3SsnPorch      4.678e+01  3.813e+01   1.227 0.220410
ScreenPorch     1.709e+01  1.893e+01   0.903 0.367039
PoolArea        9.805e+01  2.284e+01   4.294 2.10e-05 ***
MiscVal         1.012e+00  1.452e+00   0.697 0.486105
MoSold          5.118e+02  3.653e+02   1.401 0.161853
YrSold          4.140e+02  7.557e+02   0.548 0.584041
SaleType2       3.966e+04  2.589e+04   1.532 0.126195
SaleType3       3.849e+04  2.053e+04   1.875 0.061373 .
SaleType4      -3.455e+03  1.616e+04  -0.214 0.830725
SaleType5      -8.056e+03  1.657e+04  -0.486 0.627019
SaleType6      -5.967e+04  3.107e+04  -1.920 0.055374 .
SaleType7      -2.875e+03  2.489e+04  -0.115 0.908098
SaleType8       1.414e+04  1.504e+04   0.940 0.347463
SaleType9      -2.739e+03  6.421e+03  -0.427 0.669828
SaleCondition2  3.815e+03  1.980e+04   0.193 0.847301
SaleCondition3  9.637e+03  1.307e+04   0.738 0.461083
SaleCondition4  2.132e+04  8.512e+03   2.504 0.012577 *
SaleCondition5  8.947e+03  4.326e+03   2.068 0.039127 *
SaleCondition6  2.114e+04  2.395e+04   0.883 0.377846
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 21430 on 513 degrees of freedom
Multiple R-squared:  0.9518,    Adjusted R-squared:  0.9314
F-statistic: 46.86 on 216 and 513 DF,  p-value: < 2.2e-16
```

Calculate RMSE

```
> # Calculate Root Mean Squared Error
> RMSE <- sqrt(mean(model$residuals^2))
> RMSE
[1] 17965.74
```

Root Mean Squared Error (RMSE)

The square root of the mean/average of the square of all of the error.

The use of RMSE is very common and it makes an excellent general purpose error metric for numerical predictions.

Compared to the similar Mean Absolute Error, RMSE amplifies and severely punishes large errors.

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

Manually Select Variables (subset)

```
> #NA as a coefficient in a regression indicates that the variable in question is linearly related to the other variables.
> #a.k.a collinearity
> model=lm(SalePrice~
+            LotArea+OverallQual+OverallCond+YearBuilt+BsmtQual+BsmtFinSF1+
+            BsmtFinSF2+BsmtUnfSF+X1stFlrSF+X2ndFlrSF+BedroomAbvGr+
+            KitchenAbvGr+KitchenQual+GarageCars+PoolArea,
+          data=traindata)
> summary(model)

Call:
lm(formula = SalePrice ~ LotArea + OverallQual + OverallCond +
    YearBuilt + BsmtQual + BsmtFinSF1 + BsmtFinSF2 + BsmtUnfSF +
    X1stFlrSF + X2ndFlrSF + BedroomAbvGr + KitchenAbvGr + KitchenQual +
    GarageCars + PoolArea, data = traindata)

Residuals:
    Min      1Q  Median      3Q     Max
-139861  -13659      97   13871  165297
```

RMSE INCREASED!

```
Call:
lm(formula = SalePrice ~ LotArea + OverallQual + OverallCond +
    YearBuilt + BsmtQual + BsmtFinSF1 + BsmtFinSF2 + BsmtUnfSF +
    X1stFlrSF + X2ndFlrSF + BedroomAbvGr + KitchenAbvGr + KitchenQual +
    GarageCars + PoolArea, data = traindata)

Residuals:
    Min      1Q  Median      3Q     Max
-139861  -13659      97   13871  165297

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)  -8.501e+05  1.161e+05  -7.323 6.62e-13 ***
LotArea       6.085e-01  9.170e-02   6.636 6.42e-11 ***
OverallQual   1.070e+04  1.328e+03   8.054 3.38e-15 ***
OverallCond   5.608e+03  9.965e+02   5.628 2.62e-08 ***
YearBuilt     4.422e+02  5.781e+01   7.649 6.59e-14 ***
BsmtQual2    -2.068e+04  8.152e+03  -2.537 0.011396 *
BsmtQual3    -3.138e+04  4.374e+03  -7.175 1.83e-12 ***
BsmtQual4    -2.898e+04  5.574e+03  -5.199 2.62e-07 ***
BsmtFinSF1    5.046e+01  5.062e+00   9.969  < 2e-16 ***
BsmtFinSF2    3.020e+01  7.761e+00   3.891 0.000109 ***
BsmtUnfSF     2.568e+01  4.855e+00   5.289 1.64e-07 ***
X1stFlrSF     7.561e+01  5.492e+00  13.767  < 2e-16 ***
X2ndFlrSF     7.466e+01  3.611e+00  20.676  < 2e-16 ***
BedroomAbvGr -9.486e+03  1.619e+03  -5.859 7.14e-09 ***
KitchenAbvGr -1.764e+04  4.578e+03  -3.854 0.000127 ***
KitchenQual2 -2.775e+04  8.217e+03  -3.378 0.000771 ***
KitchenQual3 -3.469e+04  4.610e+03  -7.526 1.59e-13 ***
KitchenQual4 -3.580e+04  5.134e+03  -6.974 7.08e-12 ***
GarageCars    6.566e+03  1.814e+03   3.620 0.000316 ***
PoolArea      5.960e+01  2.368e+01   2.517 0.012045 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 26570 on 710 degrees of freedom
Multiple R-squared:  0.8974,    Adjusted R-squared:  0.8946
F-statistic: 326.8 on 19 and 710 DF,  p-value: < 2.2e-16

> RMSE <- sqrt(mean(model$residuals^2))
> RMSE
[1] 26202.13
```

Make Prediction & RMSE for Test Set

```
# make prediction based on test set
predict_model= predict(model,testdata)
head(predict_model) #prediction results
head(testdata$SalePrice) # vs. actual Saleprice
```

```
> head(predict_model) #prediction results
       1        3        5        6        7        9
212486.2 215267.0 265279.9 181560.2 293183.8 163363.9
> head(testdata$SalePrice) # vs. actual Saleprice
[1] 208500 223500 250000 143000 307000 129900
```

```
# calculate the value of R-squared for the prediction model on the test
data set as follows:
SSE <- sum((testdata$SalePrice - predict_model) ^ 2)
SST <- sum((testdata$SalePrice - mean(testdata$SalePrice)) ^ 2)
1 - SSE/SST

[1] 0.6781038
```

```
> # testset RMSE compare to traindata
> testRMSE <- sqrt(mean((predict_model - testdata$SalePrice)^2))
> testRMSE
[1] 43648.79
> trainRMSE <- sqrt(mean(model$residuals^2))
> trainRMSE
[1] 26230.96
```

Diagnostic Plots & Linear Regression Assumptions

```
par(mfrow=c(2,2))
plot(model)
```

Assumptions:

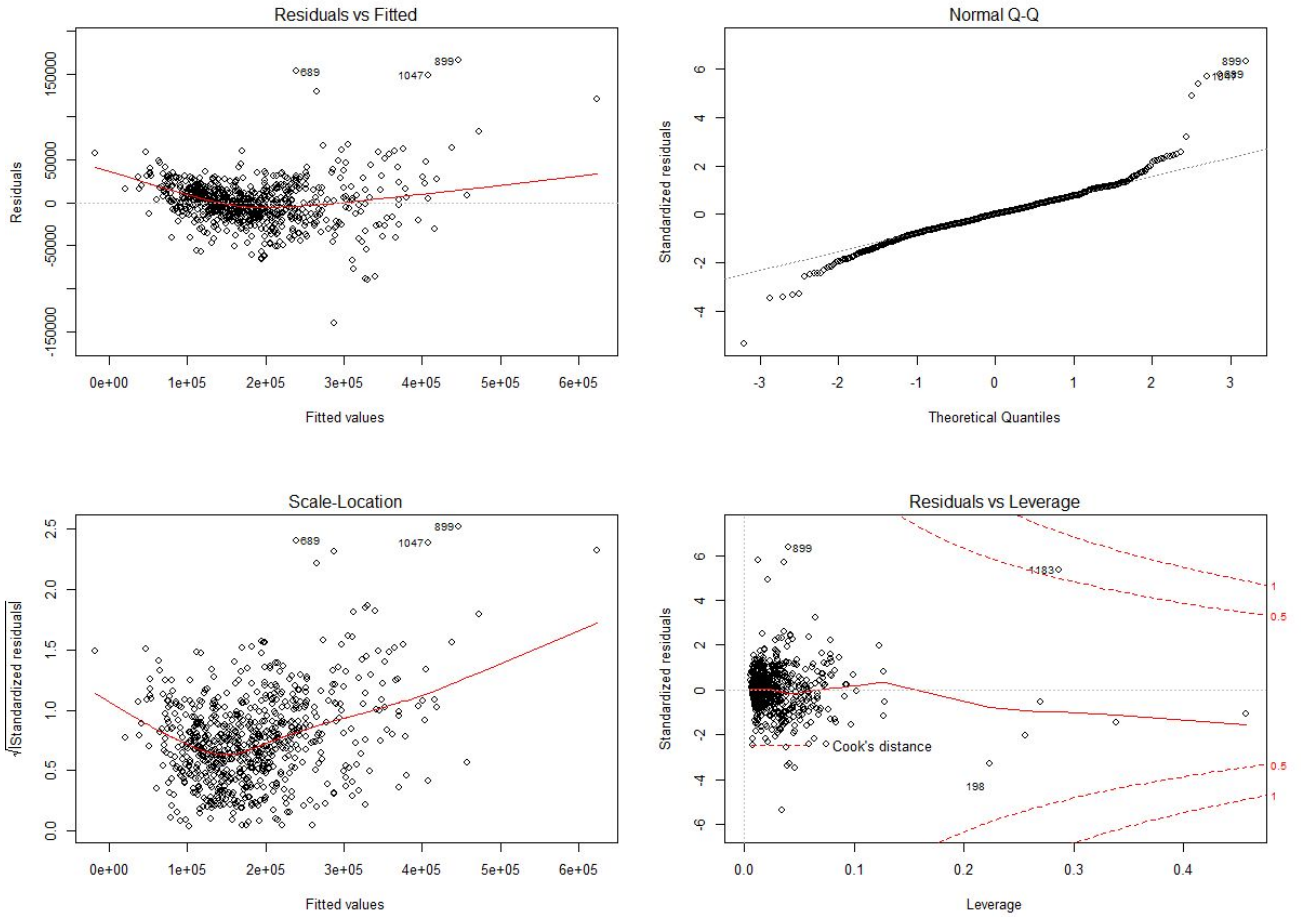
(i) linearity

(ii) Normality of the error distribution

(iii) statistical independence of the errors (No or little Autocorrelation)

(iv) homoscedasticity (constant variance) of the errors

+ No or litter **Multicollinearity**



Linear Regression - multicollinearity

Variance Inflation Factors: If all values are less than 5, there are no issues of multicollinearity.

```
> #multicollinearity
> library(car)
> vif(model)
                GVIF Df GVIF^(1/(2*Df))
LotArea     1.159642  1         1.076867
OverallQual 3.581770  1         1.892556
OverallCond 1.345979  1         1.160163
YearBuilt   3.260906  1         1.805798
BsmtQual    4.533062  3         1.286467
BsmtFinSF1  5.115702  1         2.261792
BsmtFinSF2  1.535658  1         1.239217
BsmtUnfSF   4.898007  1         2.213144
X1stFlrSF   4.395743  1         2.096603
X2ndFlrSF   2.536661  1         1.592690
BedroomAbvGr 1.788910  1         1.337501
KitchenAbvGr 1.221112  1         1.105039
KitchenQual 2.750669  3         1.183695
GarageCars  2.020974  1         1.421610
PoolArea    1.044017  1         1.021772
```

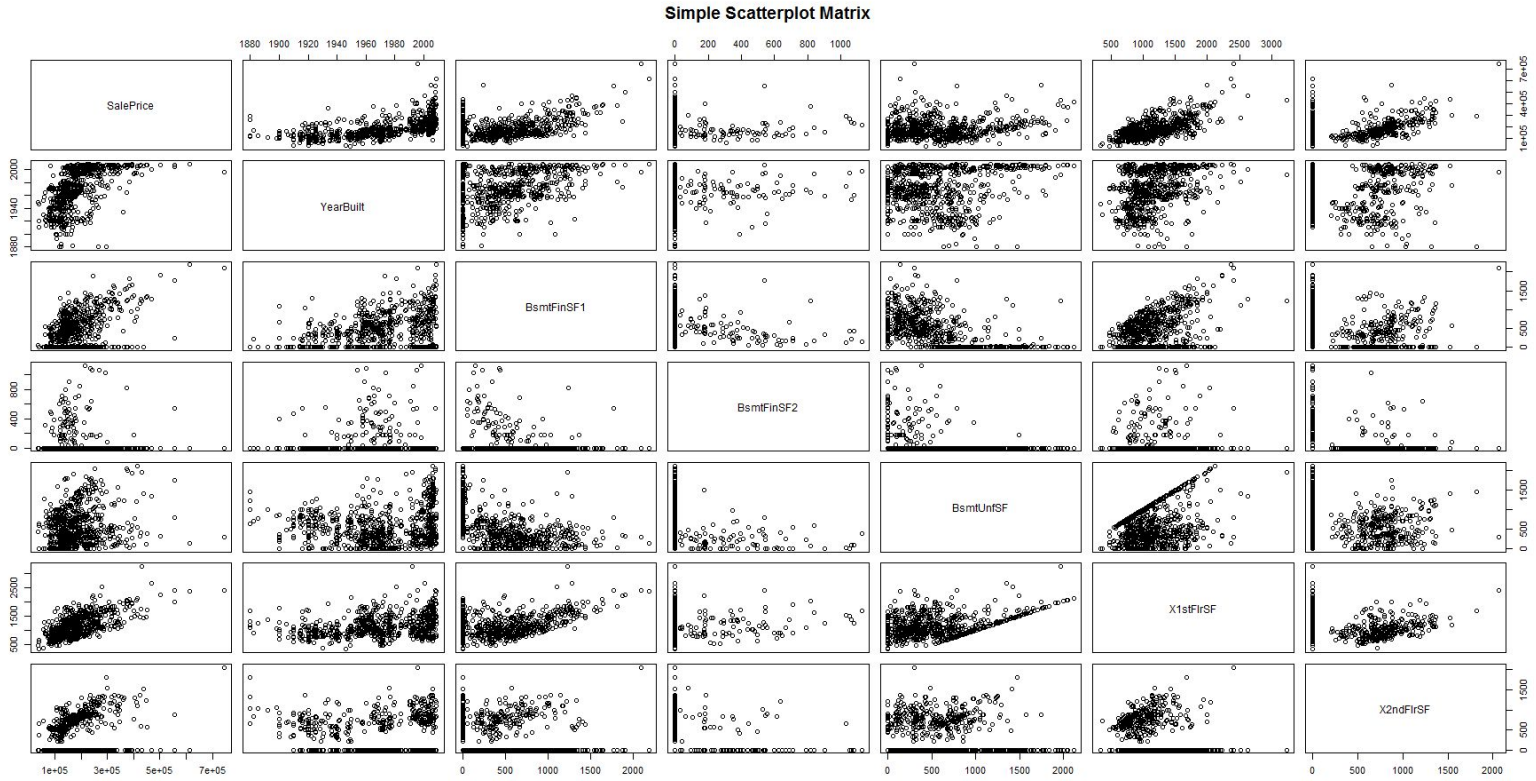
VIF	Status of predictors
VIF = 1	Not correlated
1 < VIF < 5	Moderately correlated
VIF > 5 to 10	Highly correlated

Solution: Keep one of the two independent variables.

Linear Regression - Correlation Plot

```
# Basic Scatterplot Matrix
pairs(~SalePrice+YearBuilt+
      BsmtFinSF1+BsmtFinSF2+BsmtUnfSF+X1stFlrSF+X2ndFlrSF,
      data=traindata,main="Simple Scatterplot Matrix")
```

Pearson Correlation Test won't work here due to Categorical Variables!
We directly dropped BsmFinSF1 from our model.



Multicollinearity Solution

```
# dropped the "bad" independent variable that cause multicollinearity, and insignificant variables
model=lm(SalePrice~
          LotArea+OverallQual+OverallCond+YearBuilt+BsmtQual+
          X1stFlrSF+X2ndFlrSF+BedroomAbvGr+
          KitchenAbvGr+KitchenQual+GarageCars+PoolArea,
        data=traindata)
summary(model)
RMSE <- sqrt(mean(model$residuals^2))
RMSE
library(car)
vif(model) # result looks good now:)
```

Linear Regression Assumptions-Independence

- The Durbin-Watson test has the null hypothesis that the autocorrelation is 0. Since the p-value of durbin-watson test is greater than 0.05, we reject null hypothesis.
- Therefore, there is autocorrelation and the regression assumption of independence is not satisfied.

```
> library("lmtest", lib.loc="~/R/win-library/3.3")
> dwtest(model)

        Durbin-Watson test

data:  model
DW = 2.1278, p-value = 0.9576
alternative hypothesis: true autocorrelation is greater than 0
```

Interaction Term- : vs. *

```
# Interaction Term
model=lm(SalePrice~

LotArea+OverallQual+OverallCond+YearBuilt+BsmtQual+
BsmtFinSF1*X1stFlrSF+X2ndFlrSF+BedroomAbvGr+
KitchenAbvGr+KitchenQual+GarageCars+PoolArea,data=tra
indata)
summary(model)
```

: Only interaction

*** All**

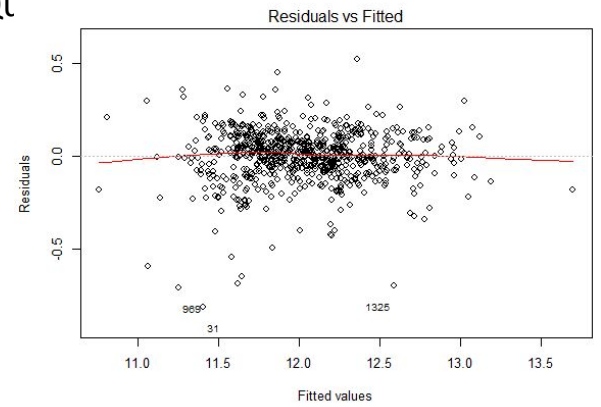
```
Call:
lm(formula = SalePrice ~ LotArea + OverallQual + OverallCond +
    YearBuilt + BsmtQual + BsmtFinSF1 * X1stFlrSF + X2ndFlrSF +
    BedroomAbvGr + KitchenAbvGr + KitchenQual + GarageCars +
    PoolArea, data = traindata)

Residuals:
    Min      1Q  Median      3Q     Max
-125825  -14162    -238   13125  167161

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)        -9.058e+05  1.155e+05  -7.842 1.63e-14 ***
LotArea             6.003e-01  8.972e-02   6.692 4.48e-11 ***
OverallQual         1.115e+04  1.297e+03   8.602  < 2e-16 ***
OverallCond         5.753e+03  9.839e+02   5.847 7.61e-09 ***
YearBuilt           4.807e+02  5.765e+01   8.337 3.93e-16 ***
BsmtQual2          -2.934e+04  8.106e+03  -3.620 0.000316 ***
BsmtQual3          -3.044e+04  4.333e+03  -7.025 5.02e-12 ***
BsmtQual4          -2.934e+04  5.474e+03  -5.359 1.13e-07 ***
BsmtFinSF1         -2.171e+01  7.752e+00  -2.800 0.005249 **
X1stFlrSF           7.445e+01  5.165e+00  14.415  < 2e-16 ***
X2ndFlrSF           6.811e+01  3.561e+00  19.126  < 2e-16 ***
BedroomAbvGr       -7.614e+03  1.613e+03  -4.720 2.84e-06 ***
KitchenAbvGr       -1.948e+04  4.491e+03  -4.339 1.64e-05 ***
KitchenQual2       -3.072e+04  8.110e+03  -3.789 0.000164 ***
KitchenQual3       -3.161e+04  4.582e+03  -6.900 1.15e-11 ***
KitchenQual4       -3.401e+04  5.086e+03  -6.688 4.59e-11 ***
GarageCars          7.705e+03  1.794e+03   4.296 1.98e-05 ***
PoolArea            4.896e+01  2.348e+01   2.085 0.037397 *
BsmtFinSF1:X1stFlrSF  3.508e-02  5.236e-03   6.699 4.26e-11 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 26290 on 711 degrees of freedom
Multiple R-squared:  0.8994,    Adjusted R-squared:  0.8969
F-statistic: 353.1 on 18 and 711 DF,  p-value: < 2.2e-16
```

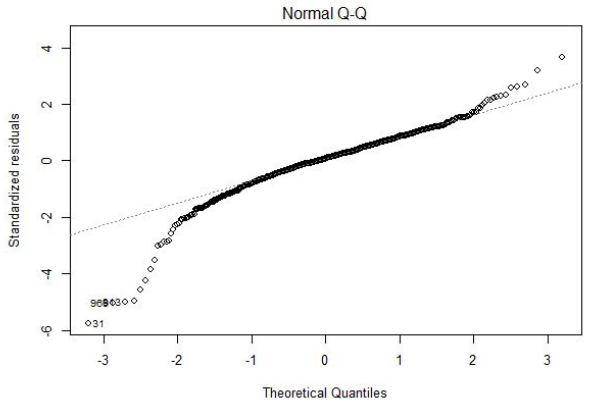
Comparison between models-ANOVA table

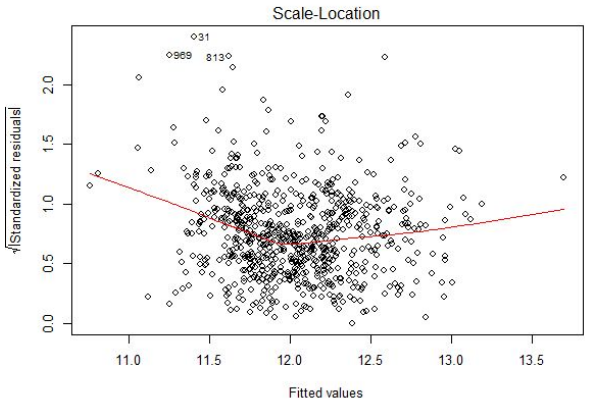
```
> # test the hypothesis that the 2nd model adds explanatory value over the 1st model
> anova(model,model_int,test = "F")
Analysis of Variance Table

Model 1: SalePrice ~ LotArea + OverallQual + OverallCond + YearBuilt +
    BsmtQual + X1stFlrSF + X2ndFlrSF + BedroomAbvGr + KitchenAbvGr +
    KitchenQual + GarageCars + PoolArea
Model 2: SalePrice ~ LotArea + OverallQual + OverallCond + YearBuilt +
    BsmtQual + BsmtFinSF1 * X1stFlrSF + X2ndFlrSF + BedroomAbvGr +
    KitchenAbvGr + KitchenQual + GarageCars + PoolArea
  Res.Df        RSS Df  Sum of Sq      F    Pr(>F)
1    713 5.9710e+11
2    711 4.9135e+11  2 1.0574e+11 76.507 < 2.2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

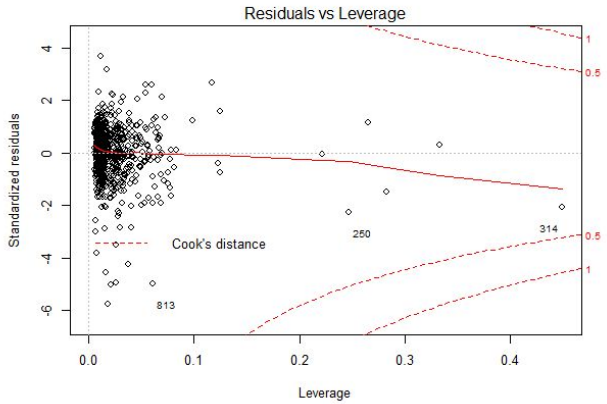
Non-Linear Transformation-Log;poly(,n)

```
model=lm(log(SalePrice)~LotArea+OverallQual+OverallCond+YearBuilt+BsmtQu
                        X1stFlrSF+X2ndFlrSF+BedroomAbvGr+
                        KitchenAbvGr+KitchenQual+GarageCars+PoolArea,
                       data=traindata)
par(mfrow=c(2,2))
plot(model)

model2=lm(log(SalePrice)~
       poly(LotArea,2)+OverallQual+OverallCond+YearBuilt+BsmtQual+
       X1stFlrSF+X2ndFlrSF+BedroomAbvGr+
       KitchenAbvGr+KitchenQual+GarageCars,data=traindata)

summary(model2)
anova(model1,model2)
```










```
Analysis of Variance Table

Model 1: log(SalePrice) ~ LotArea + OverallQual + OverallCond + YearBuilt +
    BsmtQual + X1stFlrSF + X2ndFlrSF + BedroomAbvGr + KitchenAbvGr +
    KitchenQual + GarageCars
Model 2: log(SalePrice) ~ poly(LotArea, 2) + OverallQual + OverallCond +
    YearBuilt + BsmtQual + X1stFlrSF + X2ndFlrSF + BedroomAbvGr +
    KitchenAbvGr + KitchenQual + GarageCars
  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1    714 14.366
2    713 13.847  1   0.51939 26.744 3.021e-07 ***
---
Signif. codes:  0 ?**?0.001 ?*?0.01 ??0.05 ??0.1 ??1
```

Linear Regression-log vs. poly

```
> trainRMSE <- sqrt(mean((exp(modellog$fitted.values)-traindata$SalePrice)^2))
> trainRMSE
[1] 25436.5
> predict_model= exp(predict(modellog,testdata))
> testRMSE <- sqrt(mean((predict_model - testdata$SalePrice)^2))
> testRMSE
[1] 85866.79
> trainRMSE <- sqrt(mean((exp(modellog2$fitted.values)-traindata$SalePrice)^2))
> trainRMSE #25K
[1] 24874.04
> predict_model= exp(predict(modellog2,testdata))
> testRMSE <- sqrt(mean((predict_model - testdata$SalePrice)^2))
> testRMSE #85K
[1] 93996.32
```

Models Comparison

Model	Train RMSE	Test RMSE
Multi Linear Regression (~.)	18K	?
Multi Linear Regression (sig subset)	26K	43K
Muti Linear Regression (drop x)	28K	41K
Muti Linear Regression (interaction term)	26K	59K
Muti Linear Regression (log)	25K	85K
Muti Linear Regression (log poly)	24K	93K

How to select variables?

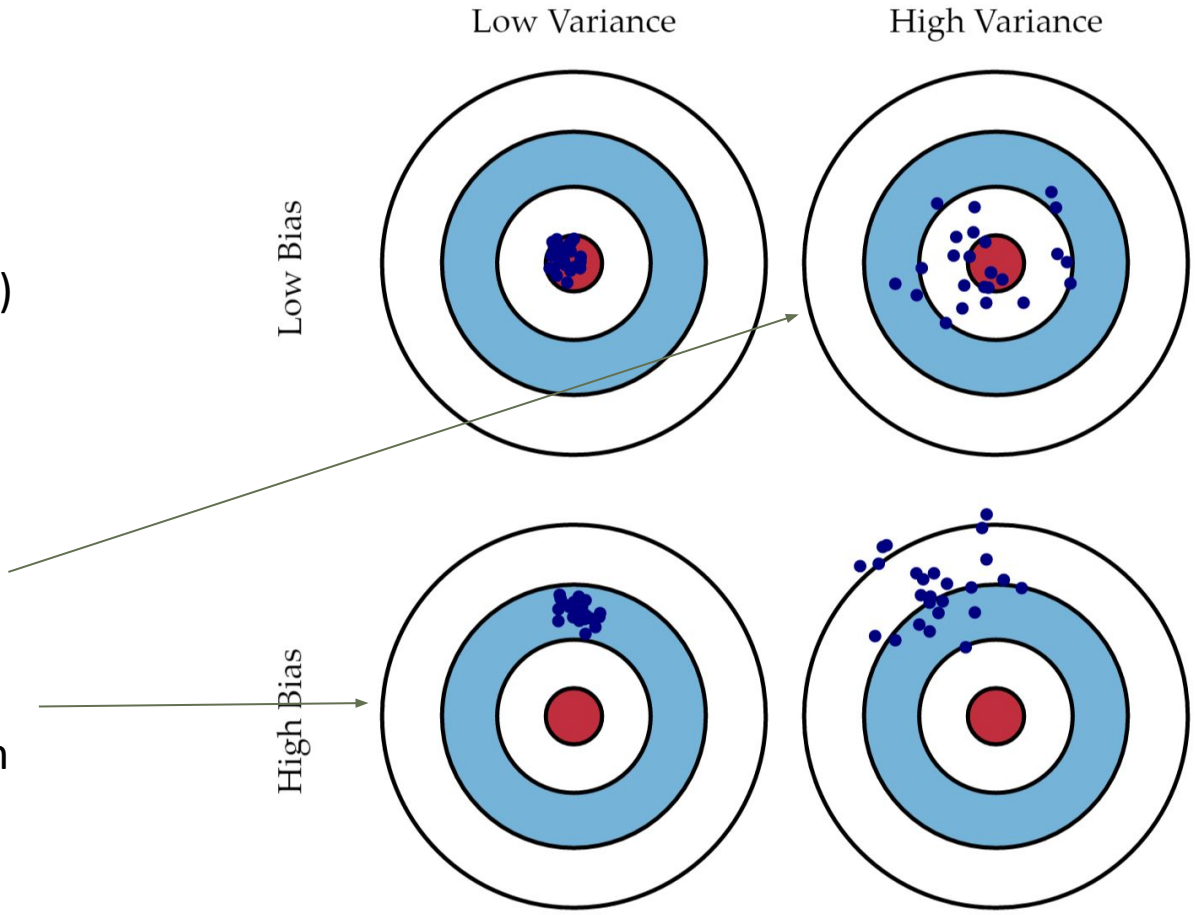
Goal of Prediction Model & Variance Bias Trade-off

Two aspects:

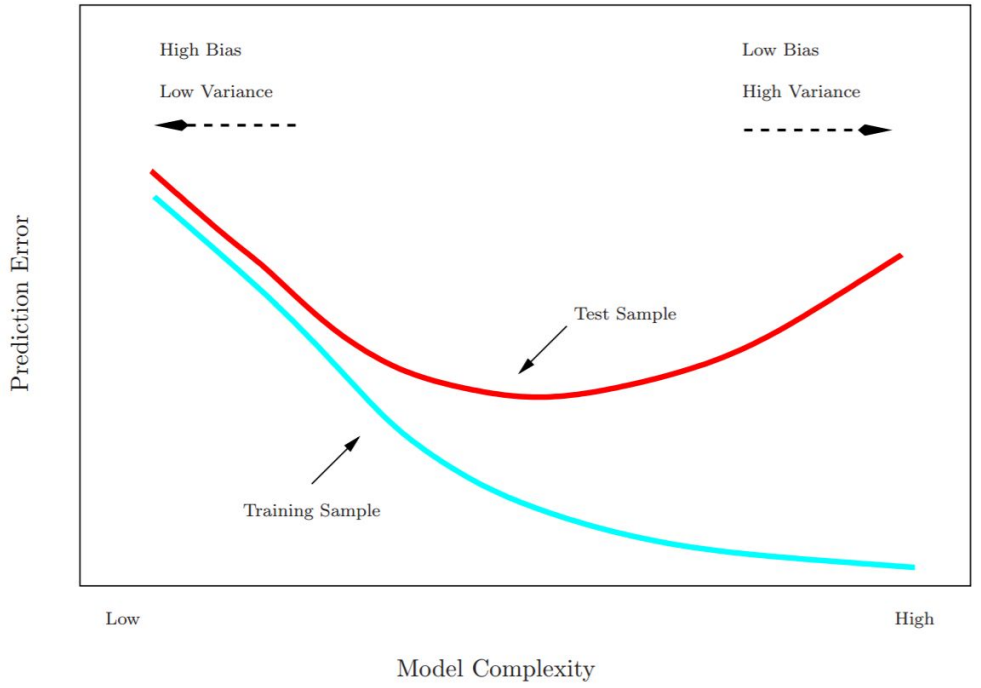
- model fit training data well requires a more complex model
- behavior of model on test data should match that on training data requires a less complex (more stable) model

Model complexity:

- **more complex model**: smaller training error but larger difference between test and training error
- **less complex model**: larger training error but smaller difference between test and training error



Linear Regression-Overfitting



Model Selection and Regularization

- Subset Selection

best subset selection

stepwise selection

- Shrinkage

Ridge regression

Lasso

Best subset and Stepwise

best subset selection :

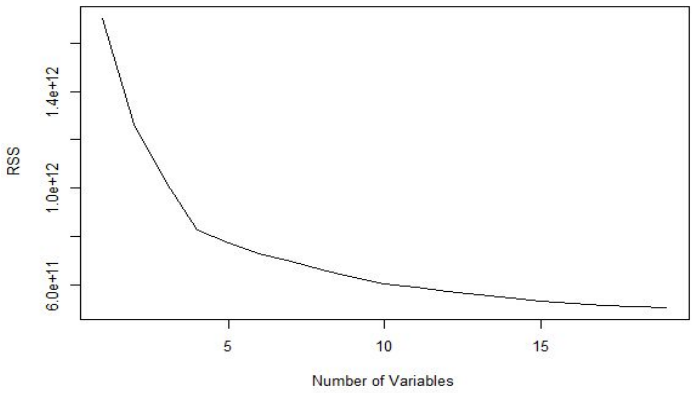
1. Let $\mathcal{M}_0$ denote the *null model*, which contains no predictors. This model simply predicts the sample mean for each observation.
2. For $k = 1, 2, \ldots p$:
   (a) Fit all $\binom{p}{k}$ models that contain exactly k predictors.
   (b) Pick the best among these $\binom{p}{k}$ models, and call it $\mathcal{M}_k$. Here *best* is defined as having the smallest RSS, or equivalently largest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

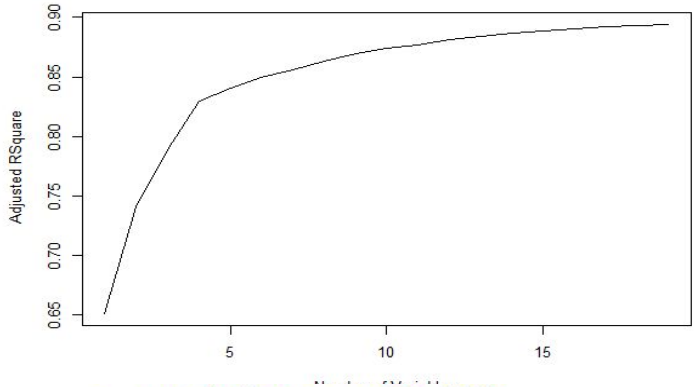
What if P is extremely large?

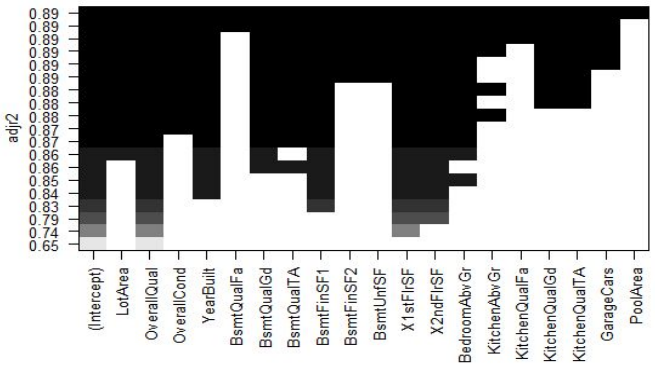
Best subset and Stepwise

```
> library(leaps)
> regfit.full=regsubsets(SalePrice~
+                          LotArea+OverallQual+OverallCond+YearBuilt+BsmtQual+BsmtFinSF1+
+                          BsmtFinSF2+BsmtUnfSF+X1stFlrSF+X2ndFlrSF+BedroomAbvGr+
+                          KitchenAbvGr+KitchenQual+GarageCars+PoolArea,data=traindata,nvmax=19)
> summary_reg = summary(regfit.full)
> names(summary_reg)
[1] "which"  "rsq"    "rss"    "adjr2"  "cp"     "bic"    "outmat" "obj"
> summary_reg$rsq # r square
 [1] 0.6510052 0.7410025 0.7898505 0.8308585 0.8414218 0.8507300 0.8570136 0.8644758 0.8712373 0.8761133
[11] 0.8790201 0.8831726 0.8859439 0.8887921 0.8912377 0.8931206 0.8948691 0.8961717 0.8971194
```

Best subset and Stepwise








```
> par(mfrow =c(2,2))
> plot(summary_reg$rss ,xlab=" Number of Variables ",ylab=" RSS",
+        type="l") # type l connect dot with line
> plot(summary_reg$adjr2 ,xlab =" Number of Variables ",
+        ylab=" Adjusted RSquare",type="l")
> which.max(summary_reg$adjr2)
[1] 19
> plot(regfit.full,scale="adjr2") #visualize model selection
> which.min(summary_reg$rss)
[1] 19
> which.min(summary_reg$bic)
[1] 19
```

Best subset and Stepwise

Stepwise Selection :

-Forward

-Backward

- Forward stepwise selection begins with a model containing no predictors, and then adds predictors to the model, one-at-a-time, until all of the predictors are in the model.
- In particular, at each step the variable that gives the greatest *additional* improvement to the fit is added to the model.

But…

1. possible to miss optimal model
2. Rely too much on P value

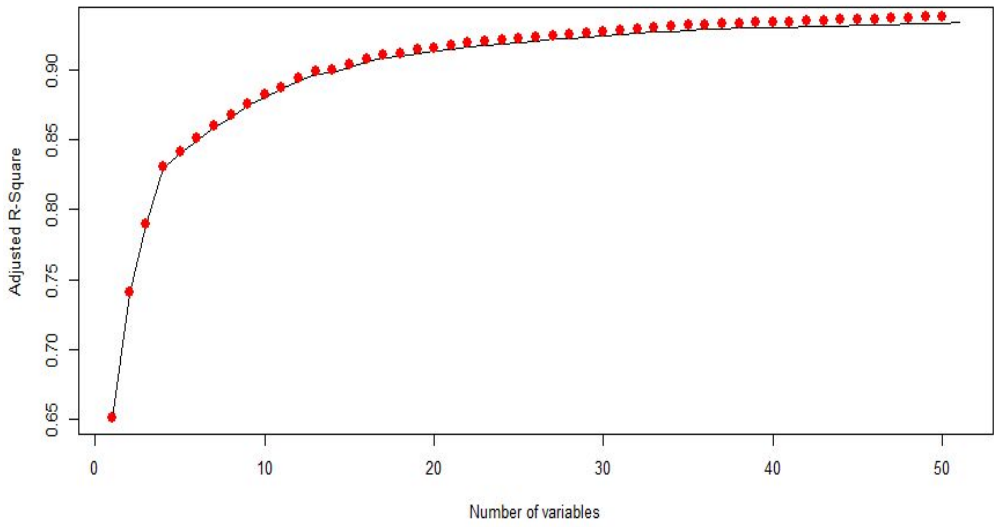
Best subset and Stepwise

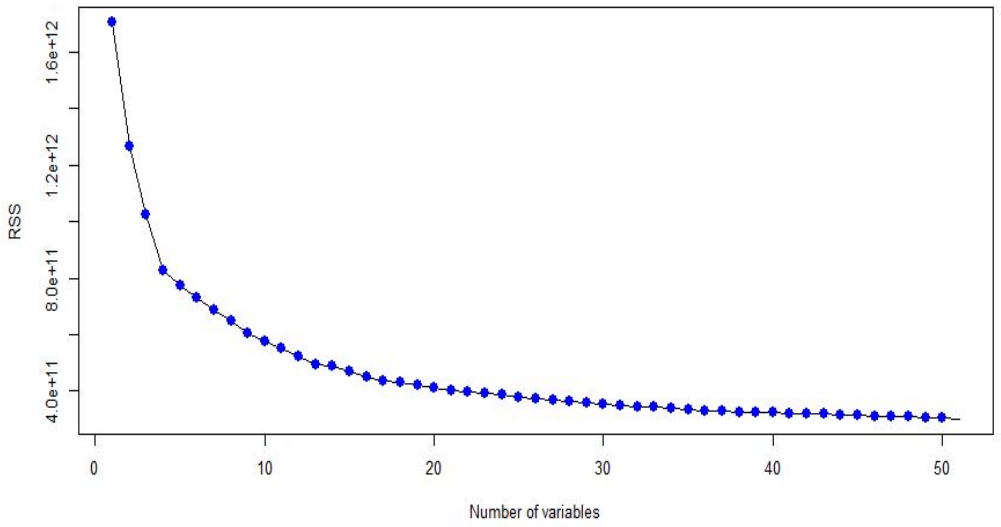
```
regfit.bwd=regsubsets (SalePrice~.,nvmax =50,really.big = TRUE,
                       method="backward",data=traindata)
summary(regfit.fwd)
```

```
         Exterior1stImStucc Exterior1stMetalSd Exterior1stPlywood Exterior1stStone Exterior1stStucco
1  ( 1 ) " "                " "                " "                " "             " "
2  ( 1 ) " "                " "                " "                " "             " "
3  ( 1 ) " "                " "                " "                " "             " "
4  ( 1 ) " "                " "                " "                " "             " "
5  ( 1 ) " "                " "                " "                " "             " "
6  ( 1 ) " "                " "                " "                " "             " "
```

```
> summary_reg2= summary(regfit.bwd)
> which.min(summary_reg2$rss)
[1] 51
>
> par(mfrow=c(2,2))
> plot(summary_reg2$adjr2,xlab="Number of variables", ylab="Adjusted R-Square", type="l")
> points(1:50,summary_reg2$rsq[1:50], col="red",cex=2,pch=20)
> plot(summary_reg2$rss,xlab="Number of variables", ylab="RSS", type="l")
> points(1:50,summary_reg2$rss[1:50], col="blue",cex=2,pch=20)
```

Best subset and Stepwise





$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n} \cdot RSS}$$

Best subset and Stepwise

```
> coef(regfit.bwd,20)
       (Intercept)          MSSubClass             LotArea NeighborhoodCrawfor NeighborhoodNoRidge
     -1.677768e+06       -1.756417e+02        1.175840e+00        5.027642e+04        1.149452e+05
NeighborhoodNridgHt NeighborhoodStoneBr      RoofStyleGable    RoofStyleGambrel        RoofStyleHip
      9.055233e+04        1.163651e+05       -3.192320e+04       -8.428360e+03       -1.223199e+04
   BsmtFinType2GLQ         HeatingGrav          CentralAirY     ElectricalFuseF        TotRmsAbvGrd
      2.304392e+04       -6.862932e+03        2.142506e+04       -1.543427e+03        1.681050e+04
    FunctionalMaj2   GarageTypeBasment   GarageTypeCarPort    GarageTypeDetchd         GarageYrBlt
     -1.509332e+04       -8.880001e+03       -7.692010e+04       -2.066454e+04        8.814708e+02
      GarageCondTA
      8.625512e+03
```

```
> predict.regsubsets =function (object ,newdata ,id ,...){
+   form=as.formula (object$call[[2]])
+   mat=model.matrix (form,newdata)
+   coefi =coef(object,id=id)
+   xvars =names (coefi)
+   mat[,xvars ]%*% coefi
+ }
>
> best_subset_pred = predict.regsubsets (regfit.bwd,testdata,20)
>
> testRMSE <- sqrt(mean((best_subset_pred- testdata$SalePrice)^2))
> testRMSE
[1] 46843.79
```

Ridge = OLS + Penalty

- Ordinary Least Squares (OLS) estimates $\beta's$ by minimizing

$$\text{RSS} = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 .$$

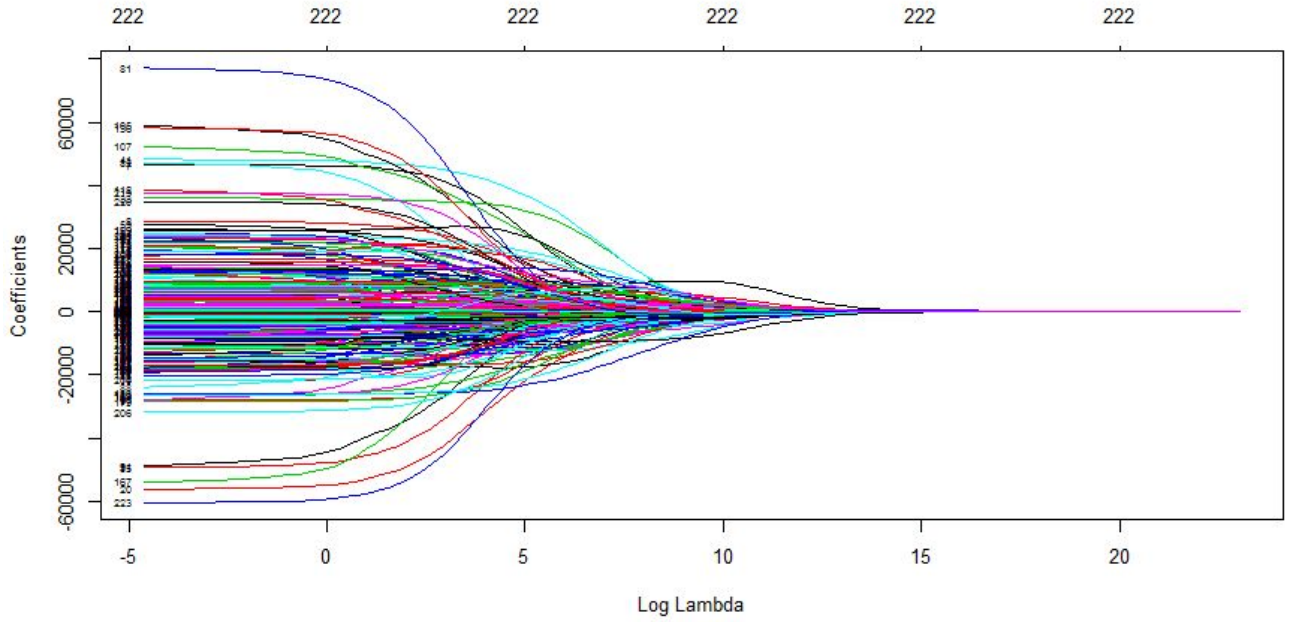
- Ridge Regression uses a slightly different equation

$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 = \text{RSS} + \lambda \sum_{j=1}^{p} \beta_j^2,$$

the **residual sum of squares (RSS)** = **sum of squared residuals (SSR)** = **sum of squared errors of prediction (SSE)**:
the sum of the squares of residuals

Regularization: ridge

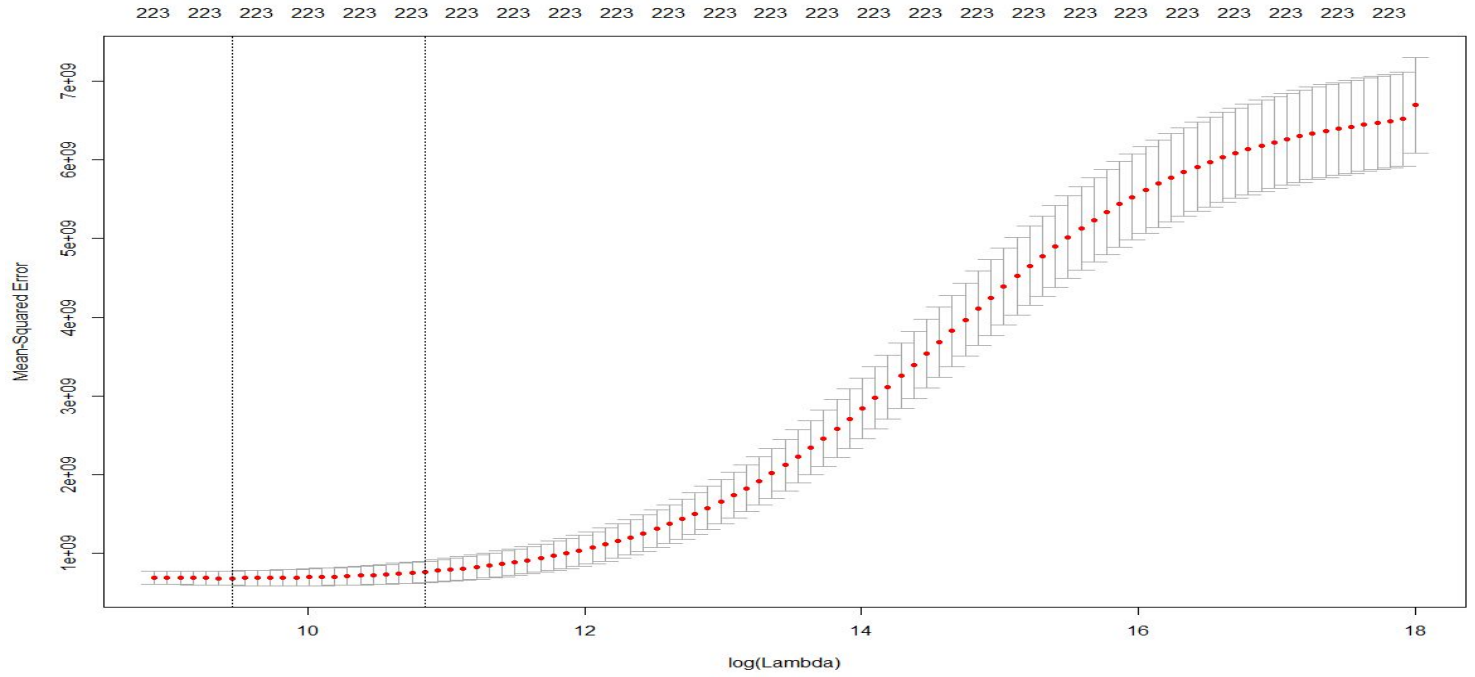
```
# convert any qualitative variables to dummy variables
x=model.matrix(SalePrice~.,traindata)[,-1] # get rid of
intercept column
head(x)
y=traindata$SalePrice

# glmnet package to perform ridge and lasso
library(glmnet)
#lambda 10^10 to 10^-2
grid = 10^seq (10,-2,length =100)
ridge_model = glmnet(x,y,alpha = 0,lambda = grid,
standardize = FALSE)
plot(ridge_model, xvar = "lambda", label = TRUE)
```



Cross Validation for best lambda

```
> ### choose the best value of lambda that would minimize the error. Run cross validation
> set.seed(2)
> cv_error = cv.glmnet(training_x,
+                      training_y,
+                      alpha = 0) #default 10 fold cv
> plot(cv_error)
> best_lambda = cv_error$lambda.min
> best_lambda
[1] 12657.65
```

Cross Validation for best lambda



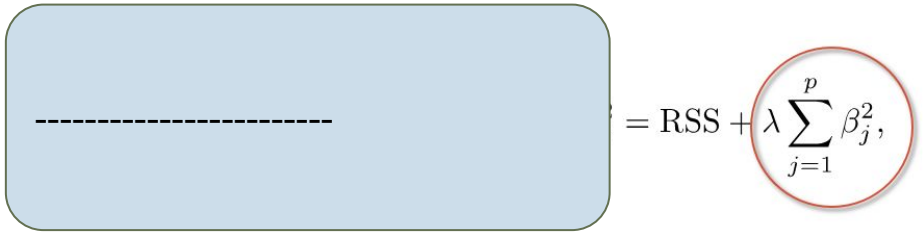
At the start, lambda is big, coefficients are restricted to be very small, MSE is big.
Later and towards the end, when lambda is very small and coefficients are big, MSE becomes small and stays flat, indicating the full model is probably the best.

Fit Ridge model and Calculate RMSE

```
> # model with best lambda
> model_coef = predict(ridge_model,
+                      type = "coefficients",
+                      s= best_lambda)
>
> ### test the model
> predicted_y = predict(ridge_model,
+                       s= best_lambda,
+                       newx = testing_x)
> ### RMSE
> sqrt(mean((predicted_y - testing_y)^2))
[1] 43236.35
```

LASSO's Penalty Term

- Ridge Regression minimizes



- The LASSO estimates the $\beta's$ by minimizing the

Benefit:

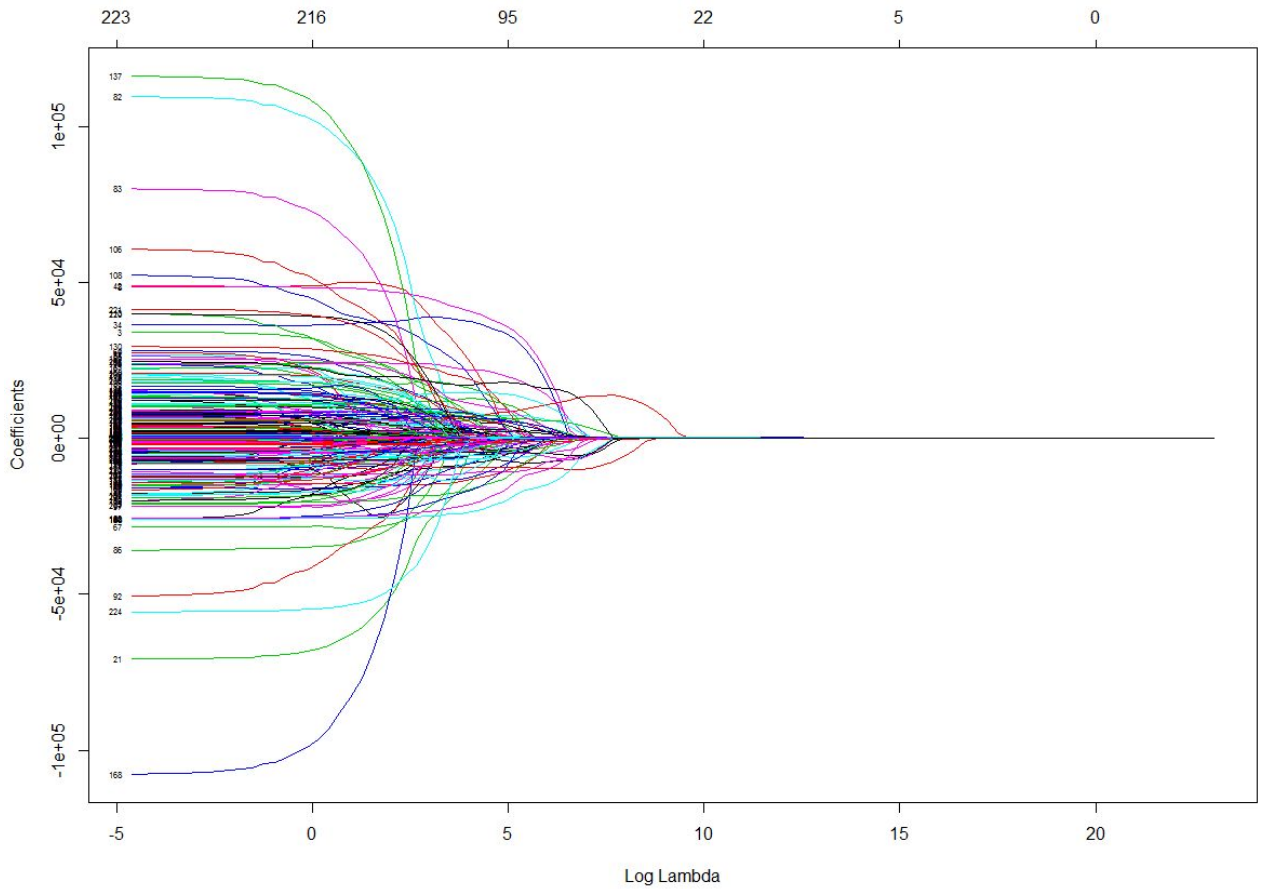
some coefficients end up being set to exactly zero!

(does both parameter shrinkage and variable selection automatically)

```
lasso_model = glmnet(training_x,
                     training_y,
                     alpha =1)
```

Fit LASSO model and Calculate RMSE

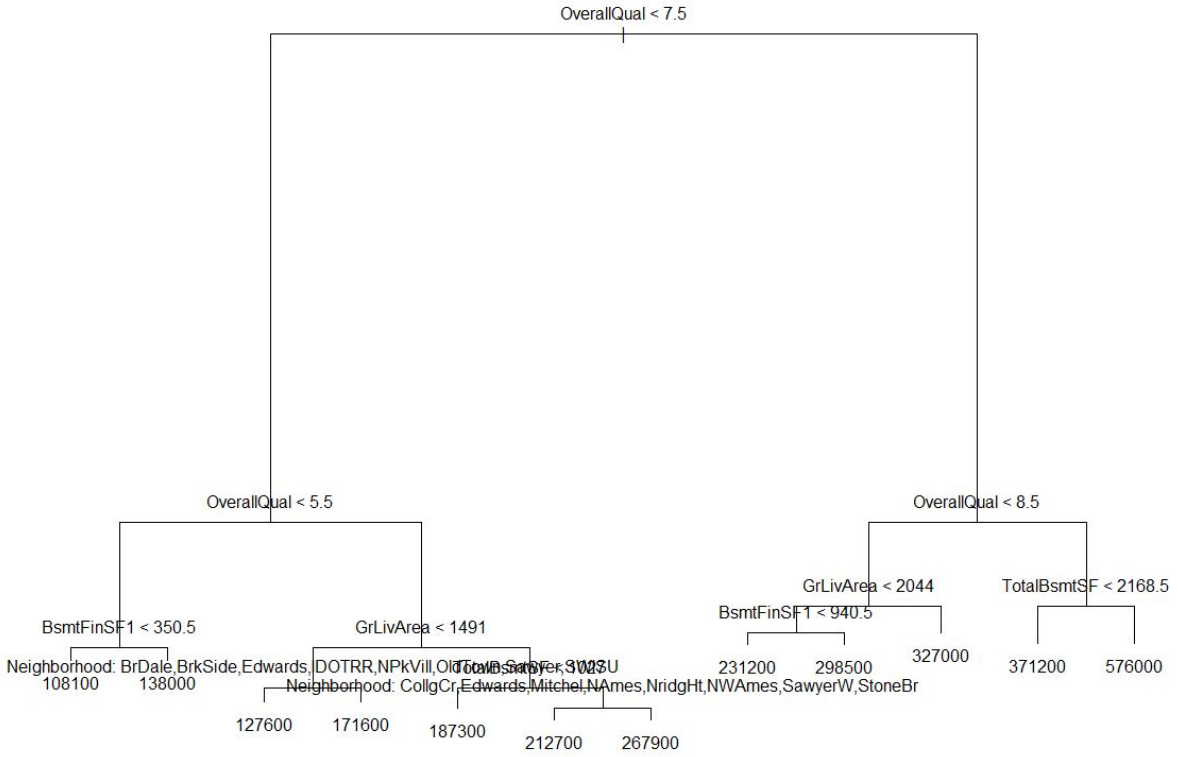
```
> ##### LASSO
> lasso_model = glmnet(training_x,
+                      training_y,
+                      alpha =1,
+                      lambda=grid,
+                      standardize=FALSE)
>
> plot(lasso_model, xvar = "lambda",label = TRUE)
>
> set.seed(2)
> cv_error = cv.glmnet(training_x,
+                      training_y,
+                      alpha = 1)
> best_lambda = cv_error$lambda.min
> best_lambda
[1] 523.0157
>
> plot(cv_error)
>
> ### OUR FINAL LASSO
> model_coef = predict(lasso_model,
+                      type = "coefficients",
+                      s= best_lambda)
>
> ### test the model
> predicted_y = predict(lasso_model,
+                       s= best_lambda,
+                       newx = testing_x)
> ### RMSE
> sqrt(mean((predicted_y - testing_y)^2))
[1] 42685.33
```



Regression Tree

Regression Tree

```
tg=tree(SalePrice~.,data=traindata)
tg
plot(tg)
text(tg,pretty=0)
```



Regression Tree

```
tg=tree(SalePrice~.,data=traindata)
tg
plot(tg)
text(tg,pretty=0)
```

```
> tg
node), split, n, deviance, yval
      * denotes terminal node

 1) root 730 4.884e+12 182400
   2) OverallQual < 7.5 607 1.523e+12 157300
     4) OverallQual < 5.5 267 2.571e+11 122300
       8) BsmtFinSF1 < 350.5 140 1.036e+11 108100 *
       9) BsmtFinSF1 > 350.5 127 9.397e+10 138000 *
     5) OverallQual > 5.5 340 6.844e+11 184700
      10) GrLivArea < 1491 158 1.542e+11 159300
        20) Neighborhood: BrDale,BrkSide,Edwards,IDOTRR,NPkVill,OldTown,Sawyer,SWISU 44 2.534e+10 127600 *
        21) Neighborhood: Blmngtn,ClearCr,CollgCr,Crawfor,Gilbert,Mitchel,NAmes,NridgHt,NWAmes,SawyerW,Somerst,Timb
er,Veenker 114 6.739e+10 171600 *
      11) GrLivArea > 1491 182 3.403e+11 206700
        22) TotalBsmtSF < 1027 104 1.097e+11 187300 *
        23) TotalBsmtSF > 1027 78 1.395e+11 232500
          46) Neighborhood: CollgCr,Edwards,Mitchel,NAmes,NridgHt,NWAmes,SawyerW,StoneBr 50 4.316e+10 212700 *
          47) Neighborhood: ClearCr,Crawfor,Gilbert,NoRidge,OldTown,Somerst,Timber,Veenker 28 4.164e+10 267900 *
   3) OverallQual > 7.5 123 1.087e+12 306400
     6) OverallQual < 8.5 93 3.740e+11 274400
      12) GrLivArea < 2044 63 1.654e+11 249400
        24) BsmtFinSF1 < 940.5 46 8.092e+10 231200 *
        25) BsmtFinSF1 > 940.5 17 2.823e+10 298500 *
      13) GrLivArea > 2044 30 8.607e+10 327000 *
     7) OverallQual > 8.5 30 3.246e+11 405300
      14) TotalBsmtSF < 2168.5 25 1.021e+11 371200 *
      15) TotalBsmtSF > 2168.5 5 4.776e+10 576000 *
```

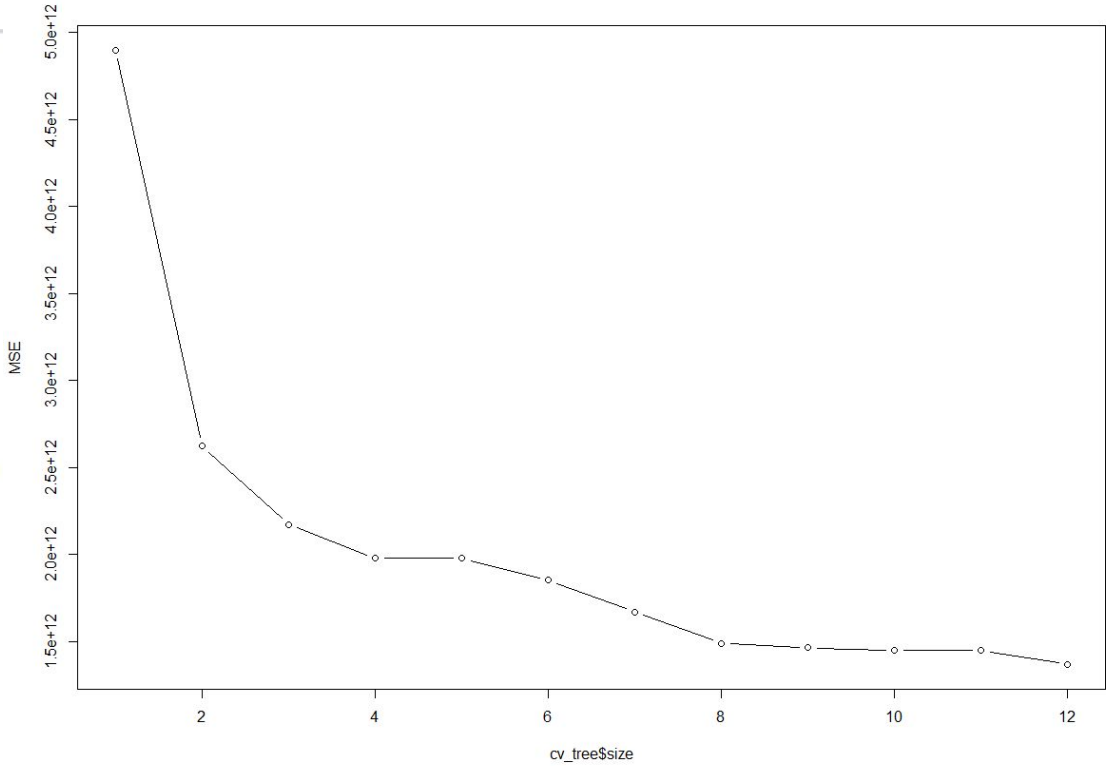
Regression Tree

```
tg=tree(SalePrice~.,data=traindata)
tg
plot(tg)
text(tg,pretty=0)
```

```
> tg
node), split, n, deviance, yval
      * denotes terminal node

 1) root 730 4.884e+12 182400
   2) OverallQual < 7.5 607 1.523e+12 157300
     4) OverallQual < 5.5 267 2.571e+11 122300
       8) BsmtFinSF1 < 350.5 140 1.036e+11 108100 *
       9) BsmtFinSF1 > 350.5 127 9.397e+10 138000 *
     5) OverallQual > 5.5 340 6.844e+11 184700
      10) GrLivArea < 1491 158 1.542e+11 159300
        20) Neighborhood: BrDale,BrkSide,Edwards,IDOTRR,NPkVill,OldTown,Sawyer,SWISU 44 2.534e+10 127600 *
        21) Neighborhood: Blmngtn,ClearCr,CollgCr,Crawfor,Gilbert,Mitchel,NAmes,NridgHt,NWAmes,SawyerW,Somerst,Timb
er,Veenker 114 6.739e+10 171600 *
      11) GrLivArea > 1491 182 3.403e+11 206700
        22) TotalBsmtSF < 1027 104 1.097e+11 187300 *
        23) TotalBsmtSF > 1027 78 1.395e+11 232500
          46) Neighborhood: CollgCr,Edwards,Mitchel,NAmes,NridgHt,NWAmes,SawyerW,StoneBr 50 4.316e+10 212700 *
          47) Neighborhood: ClearCr,Crawfor,Gilbert,NoRidge,OldTown,Somerst,Timber,Veenker 28 4.164e+10 267900 *
   3) OverallQual > 7.5 123 1.087e+12 306400
     6) OverallQual < 8.5 93 3.740e+11 274400
      12) GrLivArea < 2044 63 1.654e+11 249400
        24) BsmtFinSF1 < 940.5 46 8.092e+10 231200 *
        25) BsmtFinSF1 > 940.5 17 2.823e+10 298500 *
      13) GrLivArea > 2044 30 8.607e+10 327000 *
     7) OverallQual > 8.5 30 3.246e+11 405300
      14) TotalBsmtSF < 2168.5 25 1.021e+11 371200 *
      15) TotalBsmtSF > 2168.5 5 4.776e+10 576000 *
```

Regression Tree

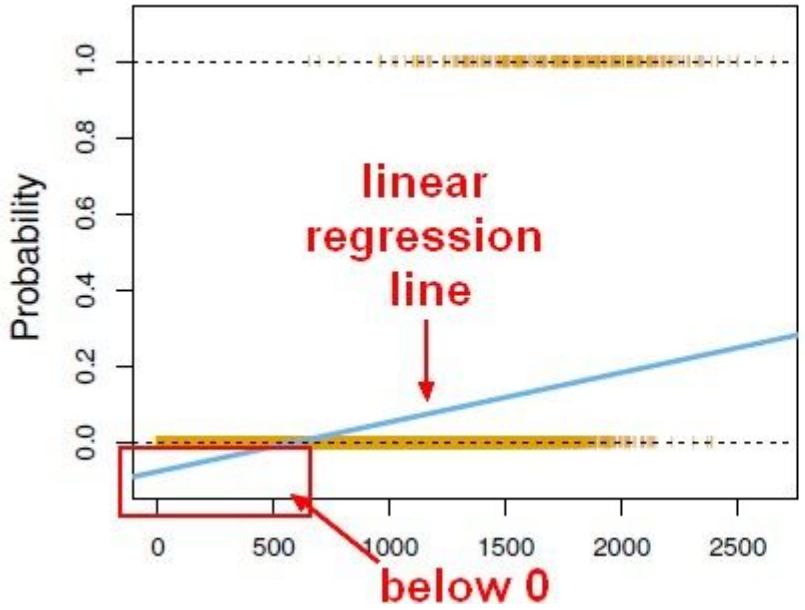
```
> plot(tg)
> text(tg,pretty=0)
> tpred=predict(tg,testdata)
> sqrt(mean((tpred-testdata$SalePrice)^2))
[1] 46920.61
> cv_tree=cv.tree(tg)
> names(cv_tree)
[1] "size"   "dev"    "k"      "method"
> cv_tree$size
 [1] 12 11 10  9  8  7  6  5  4  3  2  1
> plot(cv_tree$size,cv_tree$dev, type="b",ylab="MSE")
> which.min(cv_tree$dev)
[1] 1
> cv_tree$size[1]# size 12
[1] 12
```

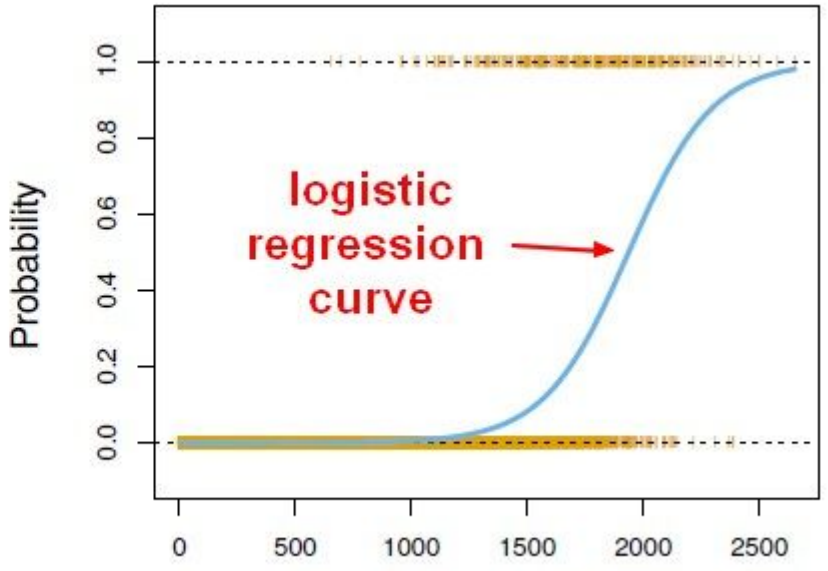


Binary Logistic Regression

Binary Logistic Regression

```
glm(y ~ ., data = mydata, family = "binomial")
```





$$Ln\left(\frac{P}{1-P}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k$$

Binary Logistic Regression

$$Ln\left(\frac{P}{1-P}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k$$

- ln is the natural logarithm, $\log_{exp}$, where exp=2.71828…
- p is the probability that the event Y occurs, p(Y=1)
- p/(1-p) is the "odds ratio"
- ln[p/(1-p)] is the log odds ratio, or "logit"

What does odds ratio=1 means?

Binary Logistic Regression

```
# create a binary variable "sale_above_avg"
traindata$sale_above_avg =ifelse(traindata$SalePrice>=mean(traindata$SalePrice),1,0)
head(traindata)

# fit the model
glm.fit=glm(sale_above_avg~
          LotArea+OverallQual+OverallCond+YearBuilt+BsmtQual+
          X1stFlrSF+X2ndFlrSF+BedroomAbvGr+
          KitchenAbvGr+KitchenQual+GarageCars,
        data=traindata,family =binomial)
summary(glm.fit) # but have a problem of perfect separation
anova(glm.fit, test="Chisq")
```

```
Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.060e+02  2.324e+01  -4.562 5.06e-06 ***
LotArea      1.800e-04  4.665e-05   3.857 0.000115 ***
OverallQual  1.084e+00  2.449e-01   4.425 9.63e-06 ***
OverallCond  5.384e-01  2.235e-01   2.409 0.016000 *
YearBuilt    4.384e-02  1.109e-02   3.951 7.77e-05 ***
BsmtQual2    2.179e+00  1.640e+00   1.328 0.184164
BsmtQual3    6.917e-01  7.205e-01   0.960 0.337057
BsmtQual4    2.819e-01  9.106e-01   0.310 0.756854
X1stFlrSF    7.628e-03  1.006e-03   7.583 3.39e-14 ***
X2ndFlrSF    5.814e-03  8.327e-04   6.982 2.90e-12 ***
BedroomAbvGr -6.785e-01  3.446e-01  -1.969 0.048979 *
KitchenAbvGr -3.066e+00  1.645e+00  -1.864 0.062299 .
KitchenQual2 -1.513e+01  9.920e+02  -0.015 0.987834
KitchenQual3 -7.601e-01  1.038e+00  -0.733 0.463823
KitchenQual4 -1.700e+00  1.062e+00  -1.600 0.109496
GarageCars    7.717e-01  5.616e-01   1.374 0.169419
```

```
             Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
NULL                          729     971.09
LotArea       1   127.99       728     843.09 < 2.2e-16 ***
OverallQual   1   424.74       727     418.35 < 2.2e-16 ***
OverallCond   1     0.08       726     418.27  0.778712
YearBuilt     1    28.21       725     390.06 1.087e-07 ***
BsmtQual      3     1.68       722     388.38  0.642331
X1stFlrSF     1    28.46       721     359.93 9.581e-08 ***
X2ndFlrSF     1    88.00       720     271.93 < 2.2e-16 ***
BedroomAbvGr  1     8.76       719     263.17  0.003073 **
KitchenAbvGr  1     5.43       718     257.73  0.019750 *
KitchenQual   3     7.28       715     250.46  0.063517 .
GarageCars    1     2.11       714     248.34  0.146025
```

Prediction and Accuracy test

AIC: balance the goodness of fit and a penalty for model complexity

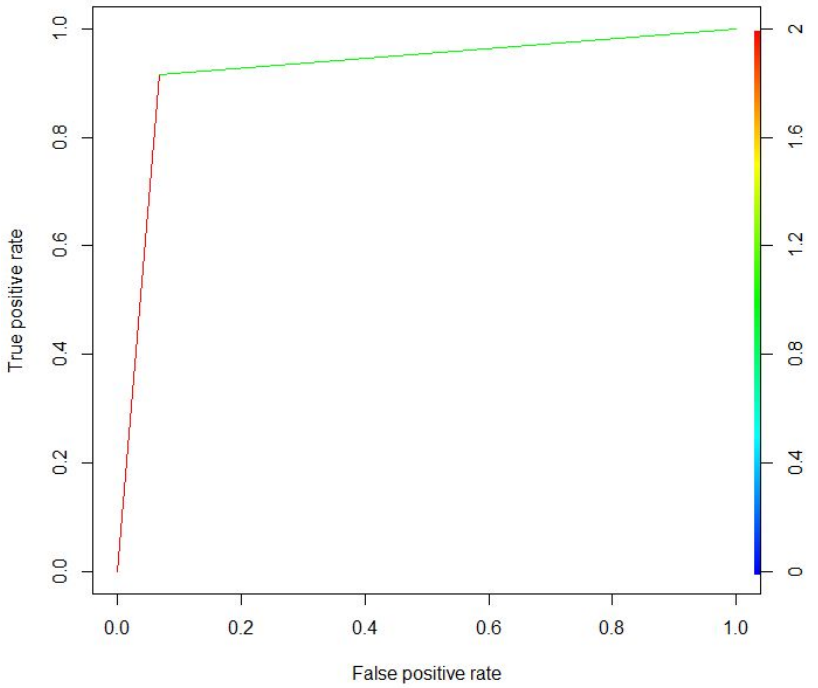
```
fitted.results <- predict(glm.fit,newdata=testdata,type='response')
fitted.results <- ifelse(fitted.results > 0.5,1,0)

misClasificError <- mean(fitted.results != testdata$sale_above_avg)
print(paste('Accuracy',1-misClasificError))
```

```
[1] "Accuracy 0.926027397260274"
```

```
> table=table(fitted.results,testdata$sale_above_avg)
> table

fitted.results   0   1
             0 425  23
             1  31 251
> error_rate <- (table[1,2]+table[2,1])/sum(table)
> error_rate
[1] 0.0739726
```

ROC Curve

```
library(ROCR)
rocpred <-
prediction(fitted.results,testdata$sale_above_avg)
rocperf <- performance(rocpred,'tpr','fpr')
plot(rocperf,colorize=TRUE, text.adj=c(-0.2,1.7))
```



Recap

Modeling Process & Model Selection

- **EDA Imputation**
- **Multiple Linear Regression & Diagnostics**
- **Regularization:Subset, Stepwise, Ridge & Lasso**
- **Binary Logistic Regression**