Lab 3 Write-Up

Kurtis Potier, Nick Hiller, Zane Shango

2/11/2020

Exercise 1

1. Load the Ames data. Drop the variables OverallCond and OverallQual.

```
library(ggplot2)
library(reshape2)
#functions
rmse = function(model){
  sqrt(mean(resid(model)^2))
}

get_complexity = function(model) {
  length(coef(model)) - 1
}


ameslist <- as.data.frame(read.table("https://msudataanalytics.github.io/SSC442/Labs/data/ames.csv",hea

#identifies columns with NA values
na_cols <- names(which(colSums(is.na(ameslist))>0))

#fixes for these columns
ameslist$Alley <- addNA(ameslist$Alley) #lot is not on an alley
ameslist$MasVnrType[is.na(ameslist$MasVnrType)] <- "None" #replace NA with None
ameslist$MasVnrArea[is.na(ameslist$MasVnrArea)] <- 0 #replace NA with 0
ameslist$Electrical <- addNA(ameslist$Electrical)
ameslist$GarageType <- addNA(ameslist$GarageType)
ameslist$GarageFinish <- addNA(ameslist$GarageFinish)
ameslist$PoolQC <- addNA(ameslist$PoolQC)
ameslist$Fence <- addNA(ameslist$Fence)
ameslist$MiscFeature <- addNA(ameslist$MiscFeature)


#delete overallcond and overallqual
#additionally, delete variables with unuseable NA (continuous variables that describe factor variables
ameslist <- within(ameslist,rm("OverallCond","OverallQual","LotFrontage","BsmtQual",
                               "BsmtCond","BsmtExposure","BsmtFinType1","BsmtFinType2","FireplaceQu",
                               "GarageYrBlt","GarageQual","GarageCond"))
head(ameslist)
```

2. Using forward selection—see Lecture 6 for details—create a series of models up to complexity length 15. You may use all variables, including categorical variables.

```
#forward selection
#null model
null_model <- lm(SalePrice~NULL,data=ameslist)

#one variable
rmse_lm1 <- sort(sapply(ameslist,function(column) rmse(lm(SalePrice~column,data=ameslist))))
rmse1<-rmse_lm1[2]
lm1 <-lm(SalePrice~Alley,data=ameslist)

#two variable
rmse_lm2 <- sort(sapply(ameslist,function(column) rmse(lm(SalePrice~Alley+column,data=ameslist))))
rmse2<-rmse_lm2[2]
lm2 <- lm(SalePrice~Alley+Neighborhood,data=ameslist)

#three variable
rmse_lm3 <- sort(sapply(ameslist,function(column) rmse(lm(SalePrice~Alley+Neighborhood+column,data=amesl
rmse3<-rmse_lm3[2]
lm3 <-lm(SalePrice~Alley+Neighborhood+GrLivArea,data=ameslist)

#four variable
rmse_lm4 <- sort(sapply(ameslist,function(column) rmse(lm(SalePrice~Alley+Neighborhood+GrLivArea+
                                                          column,data=ameslist))))
rmse4<-rmse_lm4[2]
lm4 <-lm(SalePrice~Alley+Neighborhood+GrLivArea+KitchenQual,data=ameslist)

#five variable
rmse_lm5 <- sort(sapply(ameslist,function(column) rmse(lm(SalePrice~Alley+Neighborhood+GrLivArea+
                                                          KitchenQual+column,data=ameslist))))
rmse5<-rmse_lm5[2]
lm5 <- lm(SalePrice~Alley+Neighborhood+GrLivArea+KitchenQual+RoofMatl,data=ameslist)

#six variable
rmse_lm6 <- sort(sapply(ameslist,function(column) rmse(lm(SalePrice~Alley+Neighborhood+GrLivArea+
                                                          KitchenQual+RoofMatl+column,data=ameslist)))
rmse6<-rmse_lm6[2]
lm6 <-lm(SalePrice~Alley+Neighborhood+GrLivArea+KitchenQual+RoofMatl+TotalBsmtSF,data=ameslist)

#seven variable
rmse_lm7 <- sort(sapply(ameslist,function(column) rmse(lm(SalePrice~Alley+Neighborhood+GrLivArea+
                                                          KitchenQual+RoofMatl+TotalBsmtSF+
                                                          column,data=ameslist))))
rmse7<-rmse_lm7[2]
lm7<-lm(SalePrice~Alley+Neighborhood+GrLivArea+KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF1,data=ameslis

#eight variable
rmse_lm8 <- sort(sapply(ameslist,function(column) rmse(lm(SalePrice~Alley+Neighborhood+GrLivArea+
                                                          KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF1
                                                          column,data=ameslist))))
rmse8<-rmse_lm8[2]
```

```
lm8<- lm(SalePrice~Alley+Neighborhood+GrLivArea+KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF1+BldgType,da

#nine variable
rmse_lm9 <- sort(sapply(ameslist,function(column) rmse(lm(SalePrice~Alley+Neighborhood+GrLivArea+
                                                          KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF1
                                                          BldgType+
                                                          column,data=ameslist))))
rmse9<-rmse_lm9[2]
lm9<-lm(SalePrice~Alley+Neighborhood+GrLivArea+KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF1+BldgType+
          ExterQual,data=ameslist)

#ten variable
rmse_lm10 <- sort(sapply(ameslist,function(column) rmse(lm(SalePrice~Alley+Neighborhood+GrLivArea+
                                                          KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF1
                                                          BldgType+ExterQual+
                                                          column,data=ameslist))))
rmse10<-rmse_lm10[2]
lm10<-lm(SalePrice~Alley+Neighborhood+GrLivArea+KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF1+
           BldgType+ExterQual+Condition2,data=ameslist)

#elven variable
rmse_lm11 <- sort(sapply(ameslist,function(column) rmse(lm(SalePrice~Alley+Neighborhood+GrLivArea+
                                                           KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF
                                                           BldgType+ExterQual+Condition2+
                                                           column,data=ameslist))))
rmse11<-rmse_lm11[2]
lm11<-lm(SalePrice~Alley+Neighborhood+GrLivArea+KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF1+
           BldgType+ExterQual+Condition2+YearBuilt,data=ameslist)

#twelve variable
rmse_lm12 <- sort(sapply(ameslist,function(column) rmse(lm(SalePrice~Alley+Neighborhood+GrLivArea+
                                                          KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF
                                                          BldgType+ExterQual+Condition2+YearBuilt+
                                                          column,data=ameslist))))
rmse12<-rmse_lm12[2]
lm12<- lm(SalePrice~Alley+Neighborhood+GrLivArea+KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF1+
            BldgType+ExterQual+Condition2+YearBuilt+Functional,data=ameslist)

#thirteen variable
rmse_lm13 <- sort(sapply(ameslist,function(column) rmse(lm(SalePrice~Alley+Neighborhood+GrLivArea+
                                                          KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF
                                                          BldgType+ExterQual+Condition2+YearBuilt+Fu
                                                          column,data=ameslist))))
rmse13<-rmse_lm13[2]
lm13 <- lm(SalePrice~Alley+Neighborhood+GrLivArea+KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF1+
             BldgType+ExterQual+Condition2+YearBuilt+Functional+SaleCondition,data=ameslist)

#fourteen variable
rmse_lm14 <- sort(sapply(ameslist,function(column) rmse(lm(SalePrice~Alley+Neighborhood+GrLivArea+
                                                          KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF
                                                          BldgType+ExterQual+Condition2+YearBuilt+Fu
                                                          SaleCondition+
                                                          column,data=ameslist))))
```

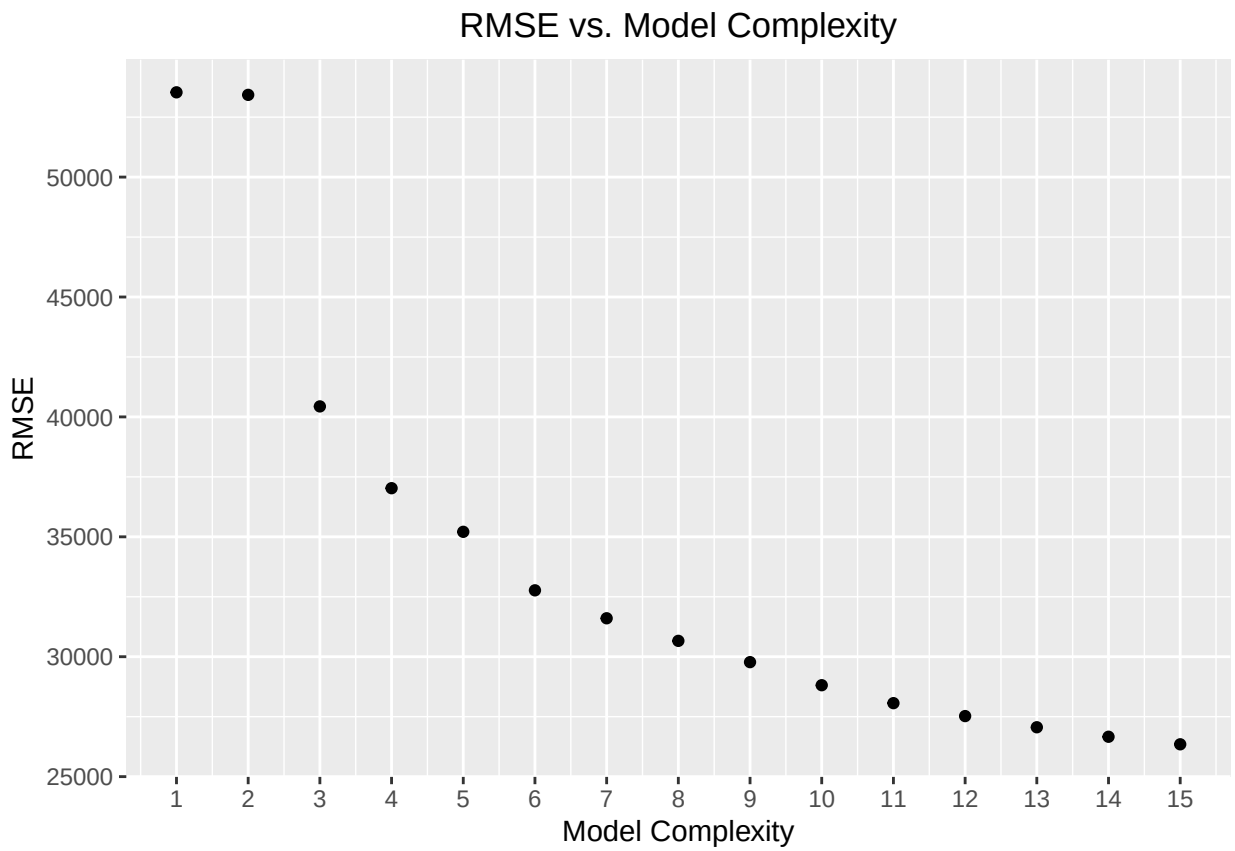
```
rmse14<-rmse_lm14[2]
lm14 <- lm(SalePrice~Alley+Neighborhood+GrLivArea+KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF1+
             BldgType+ExterQual+Condition2+YearBuilt+Functional+SaleCondition+LotArea,data=ameslist)

#fifteen variable
rmse_lm15 <- sort(sapply(ameslist,function(column) rmse(lm(SalePrice~Alley+Neighborhood+GrLivArea+
                                                           KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF
                                                           BldgType+ExterQual+Condition2+YearBuilt+Fu
                                                           SaleCondition+LotArea+
                                                           column,data=ameslist))))
rmse15<-rmse_lm15[2]
lm15 <- lm(SalePrice~Alley+Neighborhood+GrLivArea+KitchenQual+RoofMatl+TotalBsmtSF+BsmtFinSF1+
             BldgType+ExterQual+Condition2+YearBuilt+Functional+SaleCondition+LotArea+GarageCars,data=a

rmse_all <- c(rmse1,rmse2,rmse3,rmse4,rmse5,rmse6,rmse7,rmse8,rmse9,rmse10,rmse11,rmse12,rmse13,rmse14,
lm_all <- c(lm1,lm2,lm3,lm4,lm5,lm6,lm7,lm8,lm9,lm10,lm11,lm12,lm13,lm14,lm15)
rmse_complex <- c(1:15)
rmse_df <-data.frame(rmse_all,rmse_complex)
```

3. Create a chart plotting the model complexity as the xx-axis variable and RMSE as the yy-axis variable.

```
ggplot(rmse_df,aes(x=rmse_complex,y=rmse_all))+ylab("RMSE")+ggtitle("RMSE vs. Model Complexity")+
  geom_point()+scale_x_continuous(("Model Complexity"),labels=rmse_complex,breaks=rmse_complex)+
  theme(plot.title=element_text(hjust=0.5))
```



As the complexity of the model increases the RMSE value seems to decrease, with a weird result for complexity level 2. We think we should use the full size model, because it minimizes the RMSE.

Exercise 2

1. Plot the Train and Test RMSE for the 15 models you fit in Exercise 1.

```
rmse_pred <- function(actual, predicted) {
  sqrt(mean((actual - predicted) ^ 2))
}

set.seed(9)
num_obs <- nrow(ameslist)

train_index <- sample(num_obs, size = trunc(0.50 * num_obs))
train_data <- ameslist[train_index, ]
test_data <- ameslist[-train_index, ]

fit_0 = lm(SalePrice ~ 1, data = train_data)
get_complexity(fit_0)
```

```
## [1] 0
```

```r
#simplified function as SalePrice is the only response variable for this exercise
get_rmse = function(model, data) {
  rmse_pred(actual = subset(data, select = "SalePrice", drop = TRUE),
       predicted = predict(model, data))
}

get_rmse(model = fit_0, data = train_data) # train RMSE
```

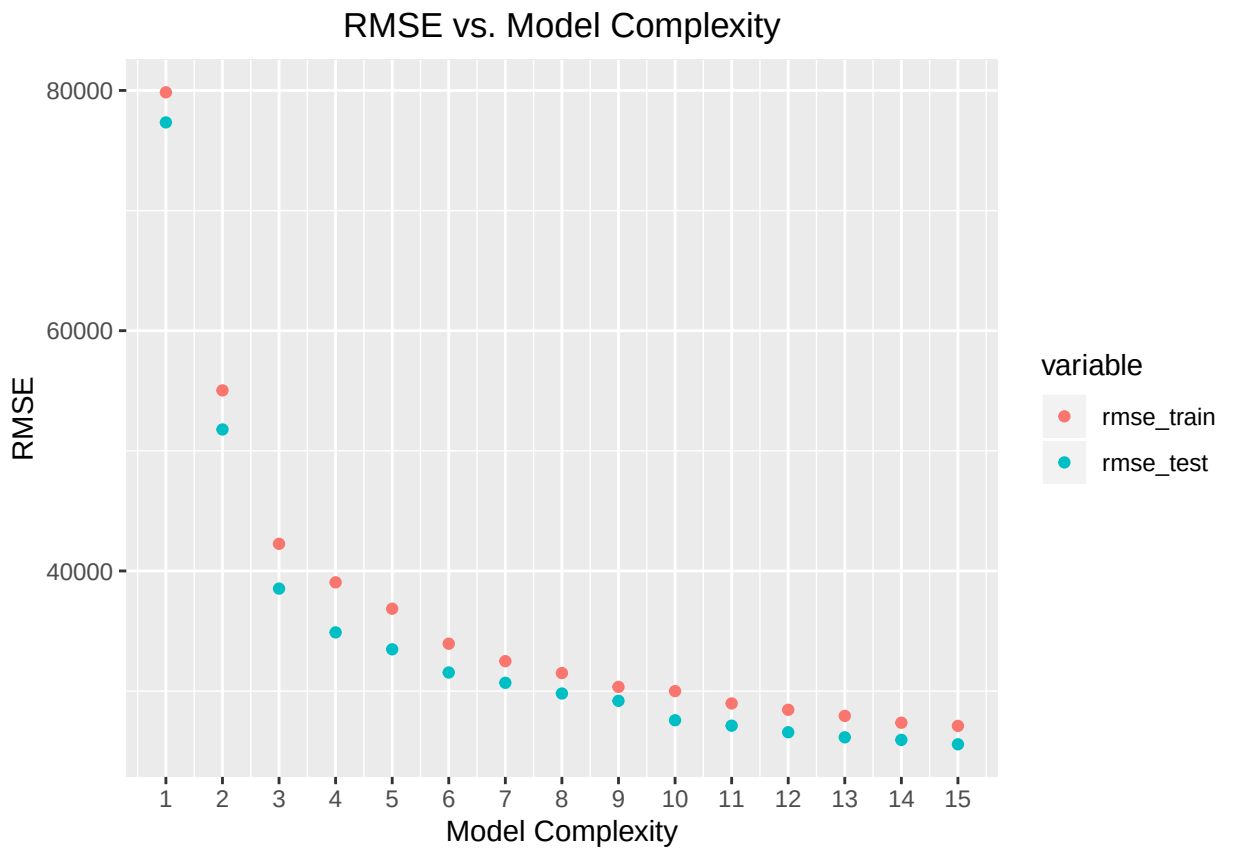
```
## [1] 80875.98
```

```r
get_rmse(model = fit_0, data = test_data) # test RMSE
```

```
## [1] 77928.62
```

```r
rmse_train <-c(get_rmse(lm1,train_data),get_rmse(lm2,train_data),get_rmse(lm3,train_data),
               get_rmse(lm4,train_data),get_rmse(lm5,train_data),get_rmse(lm6,train_data),
               get_rmse(lm7,train_data),get_rmse(lm8,train_data),get_rmse(lm9,train_data),
               get_rmse(lm10,train_data),get_rmse(lm11,train_data),get_rmse(lm12,train_data),
               get_rmse(lm13,train_data),get_rmse(lm14,train_data),get_rmse(lm15,train_data))
rmse_test <-c(get_rmse(lm1,test_data),get_rmse(lm2,test_data),get_rmse(lm3,test_data),
              get_rmse(lm4,test_data),get_rmse(lm5,test_data),get_rmse(lm6,test_data),
              get_rmse(lm7,test_data),get_rmse(lm8,test_data),get_rmse(lm9,test_data),
              get_rmse(lm10,test_data),get_rmse(lm11,test_data),get_rmse(lm12,test_data),
              get_rmse(lm13,test_data),get_rmse(lm14,test_data),get_rmse(lm15,test_data))
rmse_df_2 <- data.frame(rmse_train,rmse_test,rmse_complex)
rmse_df_2_melted <- melt(rmse_df_2,id="rmse_complex")


ggplot(rmse_df_2_melted,aes(x=rmse_complex,y=value,color=variable))+
  ylab("RMSE")+
  ggtitle("RMSE vs. Model Complexity")+
  geom_point()+
  scale_x_continuous(("Model Complexity"),labels=rmse_complex,breaks=rmse_complex)+
  theme(plot.title=element_text(hjust=0.5))
```



2. This question is the most time-consuming question. Using any method you choose and any number of regressors, predict SalePrice. Calculate the Train and Test RMSE.

```
optimizer <- function(var){
  if (class(var) == "integer"){
    linear_rmse<-get_rmse(lm(SalePrice~var,data=train_data),train_data)
    x2_rmse<-get_rmse(lm(SalePrice~(var^2),data=train_data),train_data)
    ifelse((min(var)>0),ln_rmse<-get_rmse(lm(SalePrice~log(var),data=train_data),train_data),ln_rmse<-99
    recip_rmse<-get_rmse(lm(SalePrice~(1/var),data=train_data),train_data)
    power_.2_rmse<-get_rmse(lm(SalePrice~I(var^.2),data=train_data),train_data)
    power_.4_rmse<-get_rmse(lm(SalePrice~I(var^.4),data=train_data),train_data)
    power_.5_rmse<-get_rmse(lm(SalePrice~I(var^.5),data=train_data),train_data)
    power_.6_rmse<-get_rmse(lm(SalePrice~I(var^.6),data=train_data),train_data)
    power_.8_rmse<-get_rmse(lm(SalePrice~I(var^.8),data=train_data),train_data)
    #vector of results
    rmse_comp<-c(linear_rmse,x2_rmse,ln_rmse,recip_rmse,power_.2_rmse,
                 power_.4_rmse,power_.5_rmse,power_.6_rmse,power_.8_rmse)
    names(rmse_comp) <- c("linear","x^2","ln","reciprocal","power_.2_rmse","power_.4_rmse",
                          "power_.5_rmse","power_.6_rmse","power_.8_rmse")
    rmse_comp[which.min(rmse_comp)]
  }
}
```

```r
#applies the optimization function to determine the best transformation
sapply(train_data,optimizer)
```

```
## $Id
## power_.2_rmse
##      80815.88
##
## $MSSubClass
##   linear
## 80682.55
##
## $MSZoning
## NULL
##
## $LotArea
## power_.2_rmse
##      74575.93
##
## $Street
## NULL
##
## $Alley
## NULL
##
## $LotShape
## NULL
##
## $LandContour
## NULL
##
## $Utilities
## NULL
##
## $LotConfig
## NULL
##
## $LandSlope
## NULL
##
## $Neighborhood
## NULL
##
## $Condition1
## NULL
##
## $Condition2
## NULL
##
## $BldgType
## NULL
##
## $HouseStyle
## NULL
```

```
##
## $YearBuilt
##   linear
## 68699.42
##
## $YearRemodAdd
##  linear
## 69196.8
##
## $RoofStyle
## NULL
##
## $RoofMatl
## NULL
##
## $Exterior1st
## NULL
##
## $Exterior2nd
## NULL
##
## $MasVnrType
## NULL
##
## $MasVnrArea
## NULL
##
## $ExterQual
## NULL
##
## $ExterCond
## NULL
##
## $Foundation
## NULL
##
## $BsmtFinSF1
##   linear
## 75023.28
##
## $BsmtFinSF2
## power_.2_rmse
##       80758.9
##
## $BsmtUnfSF
##  linear
## 79401.7
##
## $TotalBsmtSF
## power_.8_rmse
##      65914.75
##
## $Heating
## NULL
```

```
##
## $HeatingQC
## NULL
##
## $CentralAir
## NULL
##
## $Electrical
## NULL
##
## $X1stFlrSF
## power_.6_rmse
##      64880.14
##
## $X2ndFlrSF
##   linear
## 76366.74
##
## $LowQualFinSF
## power_.2_rmse
##      80673.22
##
## $GrLivArea
## power_.6_rmse
##      57968.43
##
## $BsmtFullBath
## power_.2_rmse
##      78517.82
##
## $BsmtHalfBath
## power_.2_rmse
##      80871.28
##
## $FullBath
##   linear
## 64897.25
##
## $HalfBath
## power_.2_rmse
##       77091.4
##
## $BedroomAbvGr
##  linear
## 79573.2
##
## $KitchenAbvGr
##       ln
## 80061.97
##
## $KitchenQual
## NULL
##
## $TotRmsAbvGrd
```

```
##   linear
## 68743.86
##
## $Functional
## NULL
##
## $Fireplaces
## power_.6_rmse
##      71137.62
##
## $GarageType
## NULL
##
## $GarageFinish
## NULL
##
## $GarageCars
##   linear
## 62451.04
##
## $GarageArea
##   linear
## 64535.09
##
## $PavedDrive
## NULL
##
## $WoodDeckSF
## power_.6_rmse
##      76568.33
##
## $OpenPorchSF
## power_.2_rmse
##      73726.23
##
## $EnclosedPorch
## power_.2_rmse
##      79301.58
##
## $X3SsnPorch
##   linear
## 80674.51
##
## $ScreenPorch
## power_.6_rmse
##      80169.54
##
## $PoolArea
## power_.2_rmse
##      80267.39
##
## $PoolQC
## NULL
##
```

```
## $Fence
## NULL
##
## $MiscFeature
## NULL
##
## $MiscVal
## power_.2_rmse
##      80786.35
##
## $MoSold
## power_.8_rmse
##      80752.77
##
## $YrSold
##       ln
## 80872.48
##
## $SaleType
## NULL
##
## $SaleCondition
## NULL
##
## $SalePrice
##       linear
## 2.190679e-10
```

```
#all the variables in a linear regression
benchmark_lm <- lm(SalePrice~.,data=train_data)
get_rmse(benchmark_lm,train_data)
```

```
## Warning in predict.lm(model, data): prediction from a rank-deficient fit
## may be misleading
```

```
## [1] 20314.17
```

```
#knocked out variables that cause errors, linear, used as a benchmark
benchmark_simple <- lm(SalePrice~(MSSubClass)+(MSZoning)+log(LotArea)+ (Street)+(Alley)+(LotShape)+(Lan
    (Utilities)+(LotConfig)+(LandSlope)+(Condition1)+(BldgType)+(HouseStyle)+
    (YearBuilt)+(YearRemodAdd)+(RoofStyle)+(MasVnrType)+(MasVnrArea)+
    (ExterQual)+(Foundation)+(BsmtFinSF1)+I(BsmtFinSF2)+(BsmtUnfSF)+(TotalBsmtSF)+
    (HeatingQC)+(CentralAir)+(X1stFlrSF)+(X2ndFlrSF)+(LowQualFinSF)+(GrLivArea)+(BsmtFullBath)+
    (BsmtHalfBath)+(FullBath)+(HalfBath)+(BedroomAbvGr)+(KitchenAbvGr)+(KitchenQual)+(TotRmsAbvGrd)+(Fu
    (Fireplaces)+(GarageType)+(GarageCars)+(GarageArea)+(PavedDrive)+(WoodDeckSF)+(OpenPorchSF)+
    (EnclosedPorch)+(X3SsnPorch)+(ScreenPorch)+(PoolArea)+(PoolQC)+(Fence)+(MiscVal)+
    (MoSold)+(YrSold)+(SaleType),data=train_data)

get_rmse(benchmark_simple,train_data)
```

```
## Warning in predict.lm(model, data): prediction from a rank-deficient fit
## may be misleading
```

```
## [1] 24179.2
```

```
get_rmse(benchmark_simple,test_data)
```

```
## Warning in predict.lm(model, data): prediction from a rank-deficient fit
## may be misleading
```

```
## [1] 66616.17
```

```
#knocked out droplevels variables, KitchenAbvGr had a 0 in the test data set,
#so log so was not used, although it was optimal
#many variables with added levels in the test data set had to be removed due to errors
master1_lm <- lm(SalePrice~(MSSubClass)+(MSZoning)+I(LotArea^.2)+ (Street)+(Alley)+(LotShape)+(LandCont
    (LotConfig)+(LandSlope)+(Condition1)+(BldgType)+(HouseStyle)+
    (YearBuilt)+(YearRemodAdd)+(RoofStyle)+(MasVnrType)+(MasVnrArea)+
    (ExterQual)+(Foundation)+(BsmtFinSF1)+I(BsmtFinSF2^.2)+(BsmtUnfSF)+I(TotalBsmtSF^.8)+
    (HeatingQC)+(CentralAir)+I(X1stFlrSF^.6)+poly(X2ndFlrSF,2)+I(LowQualFinSF^.2)+I(GrLivArea^.6)+I(Bsm
    I(BsmtHalfBath^.2)+(FullBath)+I(HalfBath^.2)+(BedroomAbvGr)+(KitchenAbvGr)+(KitchenQual)+(TotRmsAbv
    I(Fireplaces^.6)+(GarageType)+(GarageCars)+(GarageArea)+(PavedDrive)+I(WoodDeckSF^.6)+I(OpenPorchSF
    I(EnclosedPorch^.2)+(X3SsnPorch)+I(ScreenPorch^.6)+I(PoolArea^.2)+(PoolQC)+(Fence)+I(MiscVal^.2)+
    I(MoSold^.8)+log(YrSold)+(SaleType),data=train_data)

get_rmse(master1_lm,train_data)
```

```
## [1] 24259.3
```

```
get_rmse(master1_lm,test_data)
```

```
## [1] 53720.31
```

```
which(summary(master1_lm)$coefficients[,4]<.1)
```

```
##          (Intercept)           MSZoningFV           MSZoningRH
##                    1                    3                    4
##           MSZoningRL           MSZoningRM       I(LotArea^0.2)
##                    5                    6                    7
##       LandContourLvl       Condition1Norm     HouseStyle1Story
##                   16                   24                   36
##         YearRemodAdd    MasVnrTypeBrkFace       MasVnrTypeNone
##                   43                   49                   50
##      MasVnrTypeStone          MasVnrArea          ExterQualFa
##                   51                   52                   53
##          ExterQualGd          ExterQualTA      FoundationSlab
##                   54                   55                   58
##   I(TotalBsmtSF^0.8)          CentralAirY     I(X1stFlrSF^0.6)
##                   64                   69                   70
##  poly(X2ndFlrSF, 2)1  poly(X2ndFlrSF, 2)2         BedroomAbvGr
##                   71                   72                   79
##         KitchenAbvGr        KitchenQualFa        KitchenQualGd
##                   80                   81                   82
##        KitchenQualTA    I(Fireplaces^0.6)     GarageTypeAttchd
```

```
##                   83                   91                   92
##    GarageTypeBuiltIn     GarageTypeDetchd         GarageTypeNA
##                   94                   96                   97
## I(EnclosedPorch^0.2)   I(ScreenPorch^0.6)      I(PoolArea^0.2)
##                  104                  106                  107
##             PoolQCFa             PoolQCGd             PoolQCNA
##                  108                  109                  110
##          SaleTypeCon          SaleTypeCWD          SaleTypeNew
##                  118                  122                  123
##           SaleTypeWD
##                  125
```

```
#restricted model to variable with p-values <0.1
master2_lm <- lm(SalePrice~(MSZoning)+I(LotArea^.2)+(LandContour)+
    (Condition1)+(HouseStyle)+(YearRemodAdd)+(MasVnrType)+(ExterQual)+
    (Foundation)+I(TotalBsmtSF^.8)+(CentralAir)+I(X1stFlrSF^.6)+
    poly(X2ndFlrSF,2)+I(GrLivArea^.6)+(BedroomAbvGr)+(KitchenAbvGr)+
    (KitchenQual)+I(Fireplaces^.6)+(GarageType)+I(EnclosedPorch^.2)+
    I(ScreenPorch^.6)+I(PoolArea^.2)+(PoolQC)+(SaleType),data=train_data)
#final results
get_rmse(master2_lm,train_data)
```

```
## [1] 27302.5
```

```
get_rmse(master2_lm,test_data)
```

```
## [1] 53044.87
```

3. In a PDF write-up, describe the resulting model. Discuss how you arrived at this model, what interactions you're using (if any) and how confident you are that your group's prediction will perform well, relative to other groups.

We created an optimizer function that calculated the rmse for each integer type variable and told us the optimal transformation for each variable. We applied that function to the train_data and found which transformation was best for each variable. After seeing the rmse for the benchmark regression (with all variables regressed linearly), we applied the optimal transformation to each variable and regressed again. The transformations we used include: linear, power_.2, power_.6, power_.8, poly of the second degree, and log. We initially intended to to multiple degrees of poly, but it ended up overfitting out data by quite a bit so we had to cut it back. After doing this regression (master1), we restricted the model to only the variables with p-values less than .1 (master2). This returned our final rmse results. We believe we fit the model fairly well, but there is room to improve; however we did to quite a bit better than the benchmark of a strictly linear regression. Overall we are happy with the results and we believe our rmse will compare pretty well with the results from the rest of the class, considering we were able to improve on the benchmark while avoiding the overfitting problem we ran into initially.