Midterm Project

Sam Bales

10/16/2019

```
# Reading Data
train1 <- read.csv("~/Documents/Graduate School/Fall 2019/MATH 624/Midterm project/train_reg_features.cs
test1_features <- read.csv("~/Documents/Graduate School/Fall 2019/MATH 624/Midterm project/test_reg_feat
test_target <- read.csv("~/Documents/Graduate School/Fall 2019/MATH 624/Midterm project/response_id.csv
```

Exploratory Data Analsis

NOTE: I have commented out some of the function calls that add unnecessary length to the paper

```
# Examining Dataset structure
#head(train1)
#tail(train1)
#head(test1_features)
#dim(train1)
dim(test1_features)
```

```
## [1] 1459   80
```

```
str(train1)
```

```
## 'data.frame':    1460 obs. of  81 variables:
##  $ Id           : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ 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 ...
##  $ HouseStyle   : Factor w/ 8 levels "1.5Fin","1.5Unf",..: 6 3 6 6 6 1 3 6 1 2 ...
##  $ OverallQual  : int  7 6 7 7 8 5 8 7 7 5 ...
##  $ OverallCond  : int  5 8 5 5 5 5 5 6 5 6 ...
##  $ YearBuilt    : int  2003 1976 2001 1915 2000 1993 2004 1973 1931 1939 ...
##  $ YearRemodAdd : int  2003 1976 2002 1970 2000 1995 2005 1973 1950 1950 ...
##  $ RoofStyle    : Factor w/ 6 levels "Flat","Gable",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ RoofMatl     : Factor w/ 8 levels "ClyTile","CompShg",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Exterior1st  : Factor w/ 15 levels "AsbShng","AsphShn",..: 13 9 13 14 13 13 13 7 4 9 ...
##  $ Exterior2nd  : Factor w/ 16 levels "AsbShng","AsphShn",..: 14 9 14 16 14 14 14 7 16 9 ...
```

```
##  $ MasVnrType   : Factor w/ 4 levels "BrkCmn","BrkFace",..: 2 3 2 3 2 3 4 4 3 3 ...
##  $ MasVnrArea   : int  196 0 162 0 350 0 186 240 0 0 ...
##  $ ExterQual    : Factor w/ 4 levels "Ex","Fa","Gd",..: 3 4 3 4 3 4 3 4 4 4 ...
##  $ ExterCond    : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ Foundation   : Factor w/ 6 levels "BrkTil","CBlock",..: 3 2 3 1 3 6 3 2 1 1 ...
##  $ BsmtQual     : Factor w/ 4 levels "Ex","Fa","Gd",..: 3 3 3 4 3 3 1 3 4 4 ...
##  $ BsmtCond     : Factor w/ 4 levels "Fa","Gd","Po",..: 4 4 4 2 4 4 4 4 4 4 ...
##  $ BsmtExposure : Factor w/ 4 levels "Av","Gd","Mn",..: 4 2 3 4 1 4 1 3 4 4 ...
##  $ BsmtFinType1 : Factor w/ 6 levels "ALQ","BLQ","GLQ",..: 3 1 3 1 3 3 3 1 6 3 ...
##  $ BsmtFinSF1   : int  706 978 486 216 655 732 1369 859 0 851 ...
##  $ BsmtFinType2 : Factor w/ 6 levels "ALQ","BLQ","GLQ",..: 6 6 6 6 6 6 6 2 6 6 ...
##  $ BsmtFinSF2   : int  0 0 0 0 0 0 0 32 0 0 ...
##  $ BsmtUnfSF    : int  150 284 434 540 490 64 317 216 952 140 ...
##  $ TotalBsmtSF  : int  856 1262 920 756 1145 796 1686 1107 952 991 ...
##  $ Heating      : Factor w/ 6 levels "Floor","GasA",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ HeatingQC    : Factor w/ 5 levels "Ex","Fa","Gd",..: 1 1 1 3 1 1 1 1 3 1 ...
##  $ CentralAir   : Factor w/ 2 levels "N","Y": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Electrical   : Factor w/ 5 levels "FuseA","FuseF",..: 5 5 5 5 5 5 5 5 2 5 ...
##  $ X1stFlrSF    : int  856 1262 920 961 1145 796 1694 1107 1022 1077 ...
##  $ X2ndFlrSF    : int  854 0 866 756 1053 566 0 983 752 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GrLivArea    : int  1710 1262 1786 1717 2198 1362 1694 2090 1774 1077 ...
##  $ BsmtFullBath : int  1 0 1 1 1 1 1 1 0 1 ...
##  $ BsmtHalfBath : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ FullBath     : int  2 2 2 1 2 1 2 2 2 1 ...
##  $ HalfBath     : int  1 0 1 0 1 1 0 1 0 0 ...
##  $ BedroomAbvGr : int  3 3 3 3 4 1 3 3 2 2 ...
##  $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 2 2 ...
##  $ KitchenQual  : Factor w/ 4 levels "Ex","Fa","Gd",..: 3 4 3 3 3 4 3 4 4 4 ...
##  $ TotRmsAbvGrd : int  8 6 6 7 9 5 7 7 8 5 ...
##  $ Functional   : Factor w/ 7 levels "Maj1","Maj2",..: 7 7 7 7 7 7 7 7 3 7 ...
##  $ Fireplaces   : int  0 1 1 1 1 0 1 2 2 2 ...
##  $ FireplaceQu  : Factor w/ 5 levels "Ex","Fa","Gd",..: NA 5 5 3 5 NA 3 5 5 5 ...
##  $ GarageType   : Factor w/ 6 levels "2Types","Attchd",..: 2 2 2 6 2 2 2 2 6 2 ...
##  $ GarageYrBlt  : int  2003 1976 2001 1998 2000 1993 2004 1973 1931 1939 ...
##  $ GarageFinish : Factor w/ 3 levels "Fin","RFn","Unf": 2 2 2 3 2 3 2 2 3 2 ...
##  $ GarageCars   : int  2 2 2 3 3 2 2 2 2 1 ...
##  $ GarageArea   : int  548 460 608 642 836 480 636 484 468 205 ...
##  $ GarageQual   : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 5 5 5 5 5 5 2 3 ...
##  $ GarageCond   : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ PavedDrive   : Factor w/ 3 levels "N","P","Y": 3 3 3 3 3 3 3 3 3 3 ...
##  $ WoodDeckSF   : int  0 298 0 0 192 40 255 235 90 0 ...
##  $ OpenPorchSF  : int  61 0 42 35 84 30 57 204 0 4 ...
##  $ EnclosedPorch: int  0 0 0 272 0 0 0 228 205 0 ...
##  $ X3SsnPorch   : int  0 0 0 0 0 320 0 0 0 0 ...
##  $ ScreenPorch  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolQC       : Factor w/ 3 levels "Ex","Fa","Gd": NA NA NA NA NA NA NA NA NA NA ...
##  $ Fence        : Factor w/ 4 levels "GdPrv","GdWo",..: NA NA NA NA NA 3 NA NA NA NA ...
##  $ MiscFeature  : Factor w/ 4 levels "Gar2","Othr",..: NA NA NA NA NA 3 NA 3 NA NA ...
##  $ MiscVal      : int  0 0 0 0 0 700 0 350 0 0 ...
##  $ MoSold       : int  2 5 9 2 12 10 8 11 4 1 ...
##  $ YrSold       : int  2008 2007 2008 2006 2008 2009 2007 2009 2008 2008 ...
##  $ SaleType     : Factor w/ 9 levels "COD","Con","ConLD",..: 9 9 9 9 9 9 9 9 9 9 ...
```

```
##  $ SaleCondition: Factor w/ 6 levels "Abnorml","AdjLand",..: 5 5 5 1 5 5 5 5 1 5 ...
##  $ SalePrice    : int  208500 181500 223500 140000 250000 143000 307000 200000 129900 118000 ...
```

```
# Exploration
#summary(train1)
```

Looking at the summary of the data, there are several columns with missing data. After consulting the data description file, many of these missing values indicate the house did not have the attribute measured by the feature. If this is not corrected, R will exclude the observations from the dataset when fitting the model, which will substantially decrease the sample size. I am going to create new factor levels based on the data description to indicate the absence of features. These factors will be releveled to embed the absence of an attribute in the intercept of the regression equation.

```
fct.relevel <- function(x){
  x = relevel(fct_explicit_na(x, "None"), "None")
}

# Fixing factor levels
train1_rl <- train1 %>%
  mutate(MSSubClass = factor(MSSubClass),
         Alley = fct.relevel(Alley),
         BsmtQual = fct.relevel(BsmtQual),
         BsmtCond = fct.relevel(BsmtCond),
         BsmtExposure = fct.relevel(BsmtExposure),
         BsmtFinType1 = fct.relevel(BsmtFinType1),
         BsmtFinType2 = fct.relevel(BsmtFinType2),
         FireplaceQu = fct.relevel(FireplaceQu),
         GarageType = fct.relevel(GarageType),
         GarageFinish = fct.relevel(GarageFinish),
         GarageCond = fct.relevel(GarageCond),
         GarageQual = fct.relevel(GarageQual),
         PoolQC = fct.relevel(PoolQC),
         Fence = fct.relevel(Fence),
         MiscFeature = fct.relevel(MiscFeature)) %>%
  dplyr::select(-Id)

#summary(train1_rl)
str(train1_rl)
```

```
## 'data.frame':    1460 obs. of  80 variables:
##  $ MSSubClass   : Factor w/ 15 levels "20","30","40",..: 6 1 6 7 6 5 1 6 5 15 ...
##  $ 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/ 3 levels "None","Grvl",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ 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 ...
```

```
##  $ HouseStyle   : Factor w/ 8 levels "1.5Fin","1.5Unf",..: 6 3 6 6 6 1 3 6 1 2 ...
##  $ OverallQual  : int  7 6 7 7 8 5 8 7 7 5 ...
##  $ OverallCond  : int  5 8 5 5 5 5 5 6 5 6 ...
##  $ YearBuilt    : int  2003 1976 2001 1915 2000 1993 2004 1973 1931 1939 ...
##  $ YearRemodAdd : int  2003 1976 2002 1970 2000 1995 2005 1973 1950 1950 ...
##  $ RoofStyle    : Factor w/ 6 levels "Flat","Gable",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ RoofMatl     : Factor w/ 8 levels "ClyTile","CompShg",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Exterior1st  : Factor w/ 15 levels "AsbShng","AsphShn",..: 13 9 13 14 13 13 13 7 4 9 ...
##  $ Exterior2nd  : Factor w/ 16 levels "AsbShng","AsphShn",..: 14 9 14 16 14 14 14 7 16 9 ...
##  $ MasVnrType   : Factor w/ 4 levels "BrkCmn","BrkFace",..: 2 3 2 3 2 3 4 4 3 3 ...
##  $ MasVnrArea   : int  196 0 162 0 350 0 186 240 0 0 ...
##  $ ExterQual    : Factor w/ 4 levels "Ex","Fa","Gd",..: 3 4 3 4 3 4 3 4 4 4 ...
##  $ ExterCond    : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ Foundation   : Factor w/ 6 levels "BrkTil","CBlock",..: 3 2 3 1 3 6 3 2 1 1 ...
##  $ BsmtQual     : Factor w/ 5 levels "None","Ex","Fa",..: 4 4 4 5 4 4 2 4 5 5 ...
##  $ BsmtCond     : Factor w/ 5 levels "None","Fa","Gd",..: 5 5 5 3 5 5 5 5 5 5 ...
##  $ BsmtExposure : Factor w/ 5 levels "None","Av","Gd",..: 5 3 4 5 2 5 2 4 5 5 ...
##  $ BsmtFinType1 : Factor w/ 7 levels "None","ALQ","BLQ",..: 4 2 4 2 4 4 4 2 7 4 ...
##  $ BsmtFinSF1   : int  706 978 486 216 655 732 1369 859 0 851 ...
##  $ BsmtFinType2 : Factor w/ 7 levels "None","ALQ","BLQ",..: 7 7 7 7 7 7 7 3 7 7 ...
##  $ BsmtFinSF2   : int  0 0 0 0 0 0 0 32 0 0 ...
##  $ BsmtUnfSF    : int  150 284 434 540 490 64 317 216 952 140 ...
##  $ TotalBsmtSF  : int  856 1262 920 756 1145 796 1686 1107 952 991 ...
##  $ Heating      : Factor w/ 6 levels "Floor","GasA",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ HeatingQC    : Factor w/ 5 levels "Ex","Fa","Gd",..: 1 1 1 3 1 1 1 1 3 1 ...
##  $ CentralAir   : Factor w/ 2 levels "N","Y": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Electrical   : Factor w/ 5 levels "FuseA","FuseF",..: 5 5 5 5 5 5 5 5 2 5 ...
##  $ X1stFlrSF    : int  856 1262 920 961 1145 796 1694 1107 1022 1077 ...
##  $ X2ndFlrSF    : int  854 0 866 756 1053 566 0 983 752 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GrLivArea    : int  1710 1262 1786 1717 2198 1362 1694 2090 1774 1077 ...
##  $ BsmtFullBath : int  1 0 1 1 1 1 1 1 0 1 ...
##  $ BsmtHalfBath : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ FullBath     : int  2 2 2 1 2 1 2 2 2 1 ...
##  $ HalfBath     : int  1 0 1 0 1 1 0 1 0 0 ...
##  $ BedroomAbvGr : int  3 3 3 3 4 1 3 3 2 2 ...
##  $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 2 2 ...
##  $ KitchenQual  : Factor w/ 4 levels "Ex","Fa","Gd",..: 3 4 3 3 3 4 3 4 4 4 ...
##  $ TotRmsAbvGrd : int  8 6 6 7 9 5 7 7 8 5 ...
##  $ Functional   : Factor w/ 7 levels "Maj1","Maj2",..: 7 7 7 7 7 7 7 7 3 7 ...
##  $ Fireplaces   : int  0 1 1 1 1 0 1 2 2 2 ...
##  $ FireplaceQu  : Factor w/ 6 levels "None","Ex","Fa",..: 1 6 6 4 6 1 4 6 6 6 ...
##  $ GarageType   : Factor w/ 7 levels "None","2Types",..: 3 3 3 7 3 3 3 3 7 3 ...
##  $ GarageYrBlt  : int  2003 1976 2001 1998 2000 1993 2004 1973 1931 1939 ...
##  $ GarageFinish : Factor w/ 4 levels "None","Fin","RFn",..: 3 3 3 4 3 4 3 3 4 3 ...
##  $ GarageCars   : int  2 2 2 3 3 2 2 2 2 1 ...
##  $ GarageArea   : int  548 460 608 642 836 480 636 484 468 205 ...
##  $ GarageQual   : Factor w/ 6 levels "None","Ex","Fa",..: 6 6 6 6 6 6 6 6 3 4 ...
##  $ GarageCond   : Factor w/ 6 levels "None","Ex","Fa",..: 6 6 6 6 6 6 6 6 6 6 ...
##  $ PavedDrive   : Factor w/ 3 levels "N","P","Y": 3 3 3 3 3 3 3 3 3 3 ...
##  $ WoodDeckSF   : int  0 298 0 0 192 40 255 235 90 0 ...
##  $ OpenPorchSF  : int  61 0 42 35 84 30 57 204 0 4 ...
##  $ EnclosedPorch: int  0 0 0 272 0 0 0 228 205 0 ...
##  $ X3SsnPorch   : int  0 0 0 0 0 320 0 0 0 0 ...
```

```
##  $ ScreenPorch  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolQC       : Factor w/ 4 levels "None","Ex","Fa",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ Fence        : Factor w/ 5 levels "None","GdPrv",..: 1 1 1 1 1 4 1 1 1 1 ...
##  $ MiscFeature  : Factor w/ 5 levels "None","Gar2",..: 1 1 1 1 1 4 1 4 1 1 ...
##  $ MiscVal      : int  0 0 0 0 0 700 0 350 0 0 ...
##  $ MoSold       : int  2 5 9 2 12 10 8 11 4 1 ...
##  $ YrSold       : int  2008 2007 2008 2006 2008 2009 2007 2009 2008 2008 ...
##  $ SaleType     : Factor w/ 9 levels "COD","Con","ConLD",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ SaleCondition: Factor w/ 6 levels "Abnorml","AdjLand",..: 5 5 5 1 5 5 5 5 1 5 ...
##  $ SalePrice    : int  208500 181500 223500 140000 250000 143000 307000 200000 129900 118000 ...
```

```
train1_rl %>%
  dplyr::select(LotFrontage) %>%
  filter_all(any_vars(is.na(.))) %>%
  summarize(n())
```

```
##   n()
## 1 259
```

```
train1_rl %>%
  group_by(MSSubClass) %>%
  summarize(sum(is.na(LotFrontage)))
```

```
## # A tibble: 15 x 2
##    MSSubClass `sum(is.na(LotFrontage))`
##    <fct>                           <int>
##  1 20                                 99
##  2 30                                  6
##  3 40                                  1
##  4 45                                  0
##  5 50                                 16
##  6 60                                 69
##  7 70                                  5
##  8 75                                  1
##  9 80                                 20
## 10 85                                  6
## 11 90                                  5
## 12 120                                20
## 13 160                                 8
## 14 180                                 0
## 15 190                                 3
```

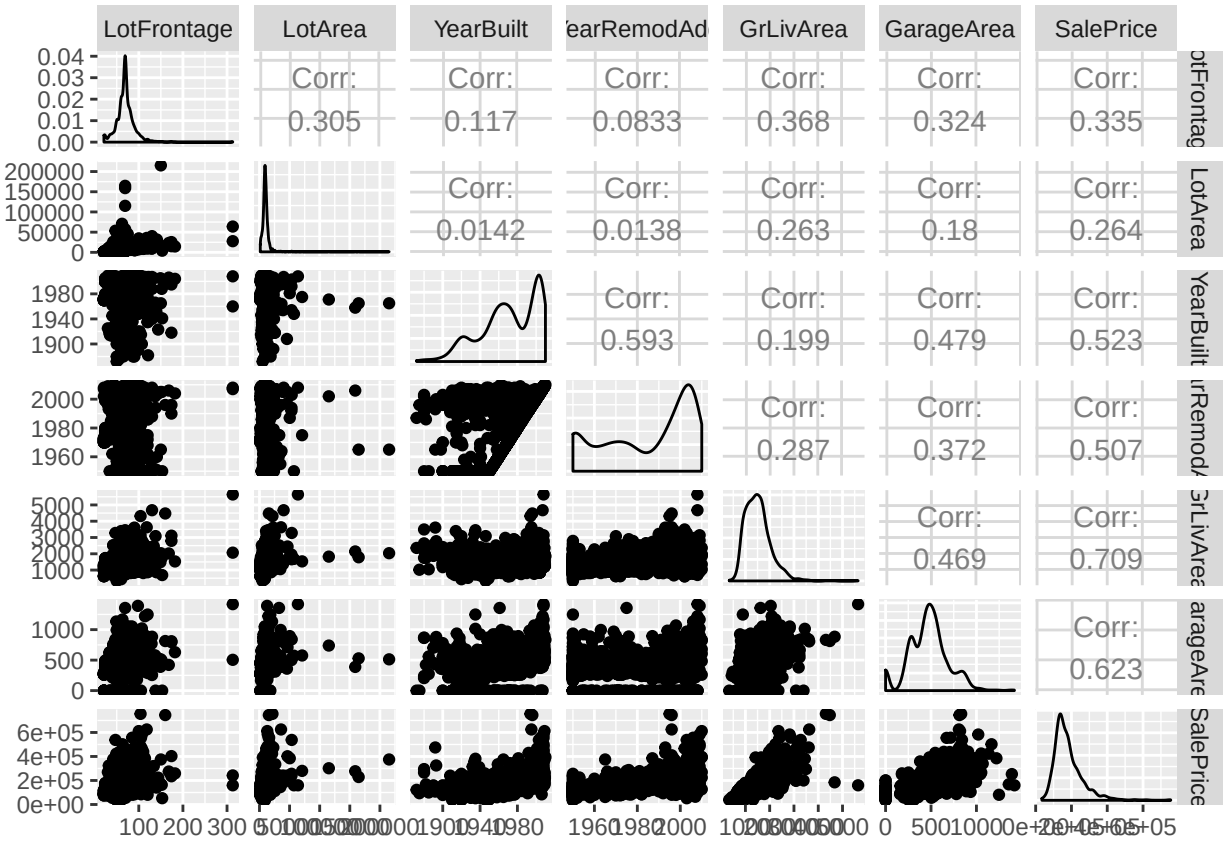
Looking at the summary above, LotFrontage has several missing values. It does not appear NA's represent the absence of Lot Frontage, so NA's can't be replaced with 0's. There are several approaches to deal with this, but I am going to employ median imputation for its simplicity and robust qualities.

```
train1_rl <- train1_rl %>%
  mutate(LotFrontage = ifelse(is.na(LotFrontage), median(LotFrontage, na.rm = T), LotFrontage))

summary(train1_rl$LotFrontage)
```

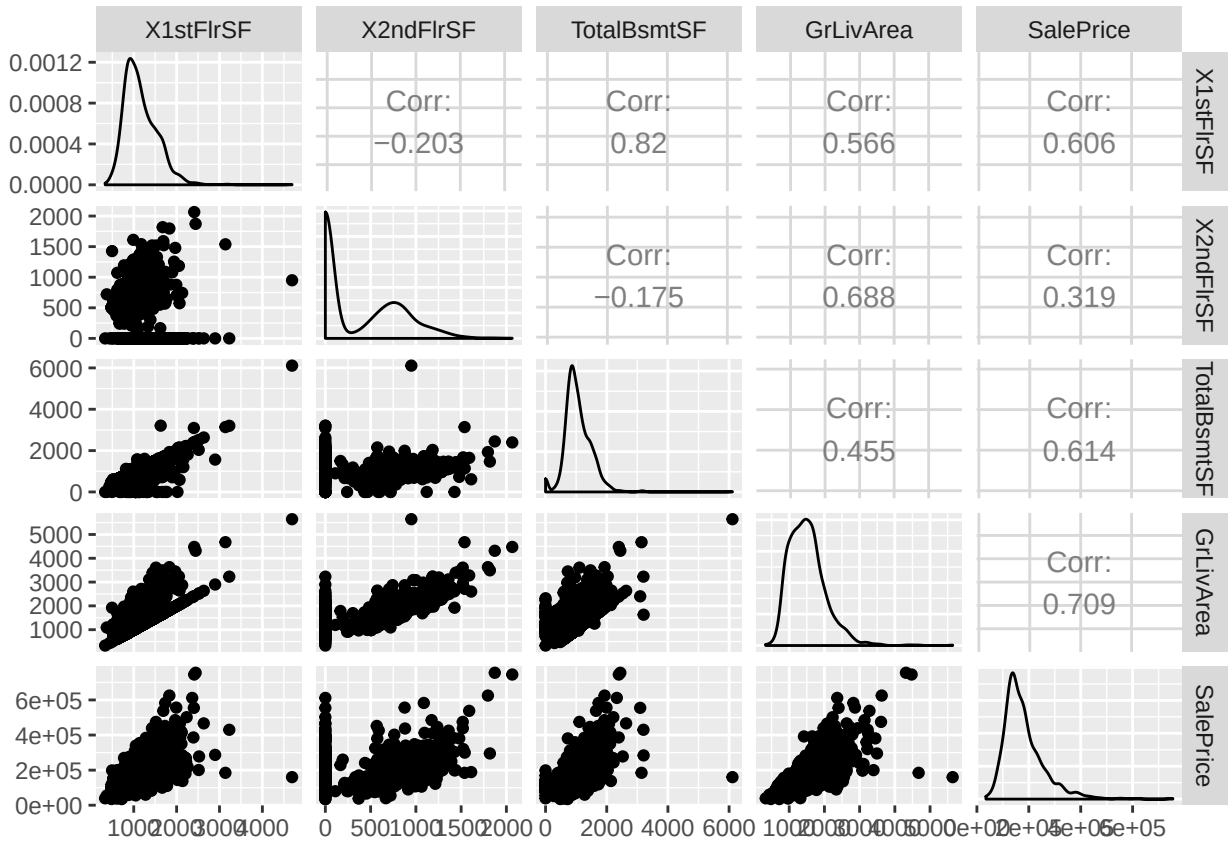
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   21.00   60.00   69.00   69.86   79.00  313.00
```

Gaphical and Numerical Analysis

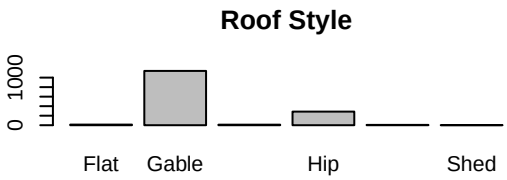
```
#Plotting Continuous Data
select_cont_df <- train1_rl %>%
  dplyr::select(LotFrontage, LotArea, YearBuilt, YearRemodAdd,
         GrLivArea, GarageArea, SalePrice)

ggpairs(select_cont_df)
```

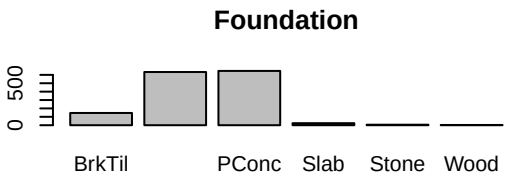


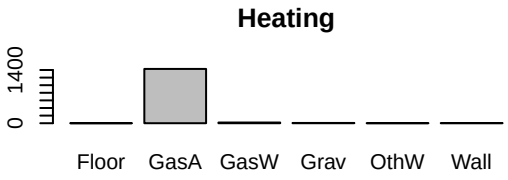

```
select_cont_df2 <- train1_rl %>%
  dplyr::select(X1stFlrSF, X2ndFlrSF, TotalBsmtSF, GrLivArea, SalePrice)

ggpairs(select_cont_df2)
```

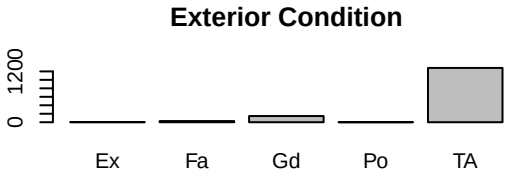


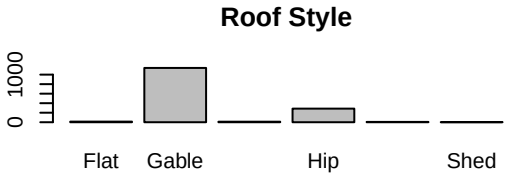

```
# Plotting Discrete Data
par(mfrow = c(3,2))
barplot(table(train1_rl$RoofStyle), main = "Roof Style")
barplot(table(train1_rl$Foundation), main = "Foundation")
barplot(table(train1_rl$BsmtFinType1), main = "Basement Type")
barplot(table(train1_rl$Heating), main = "Heating")
barplot(table(train1_rl$ExterCond), main = "Exterior Condition")
barplot(table(train1_rl$RoofStyle), main = "Roof Style")
```



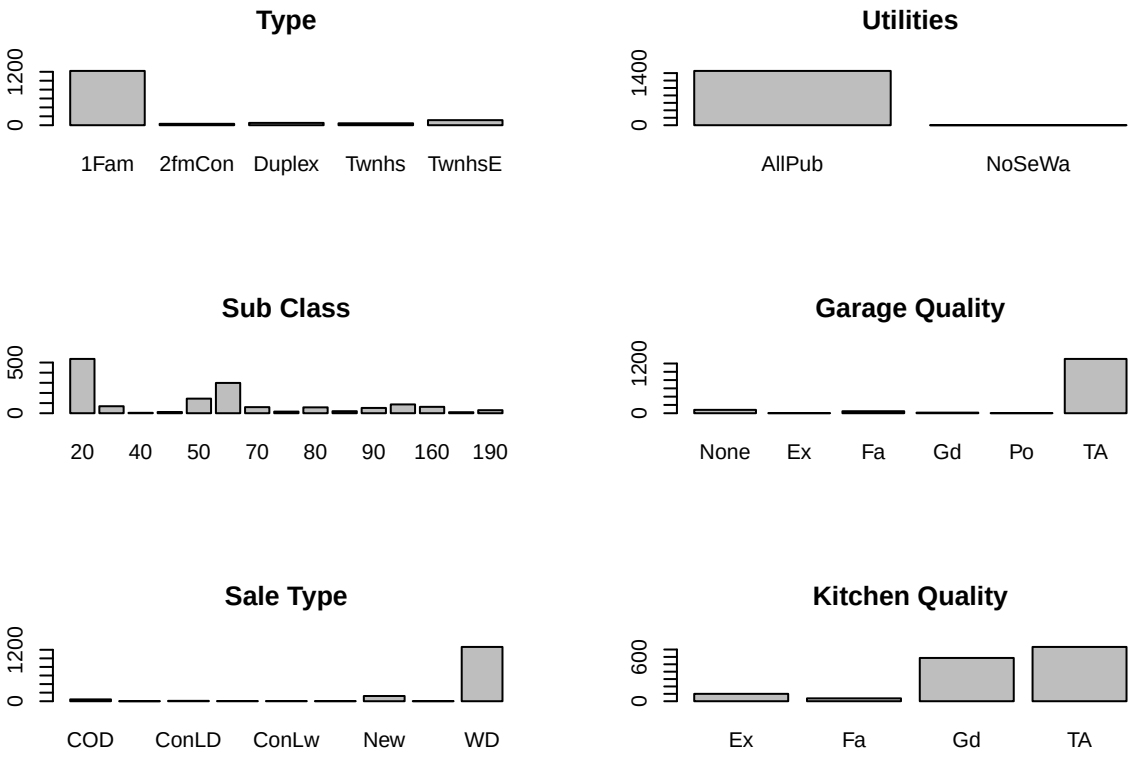








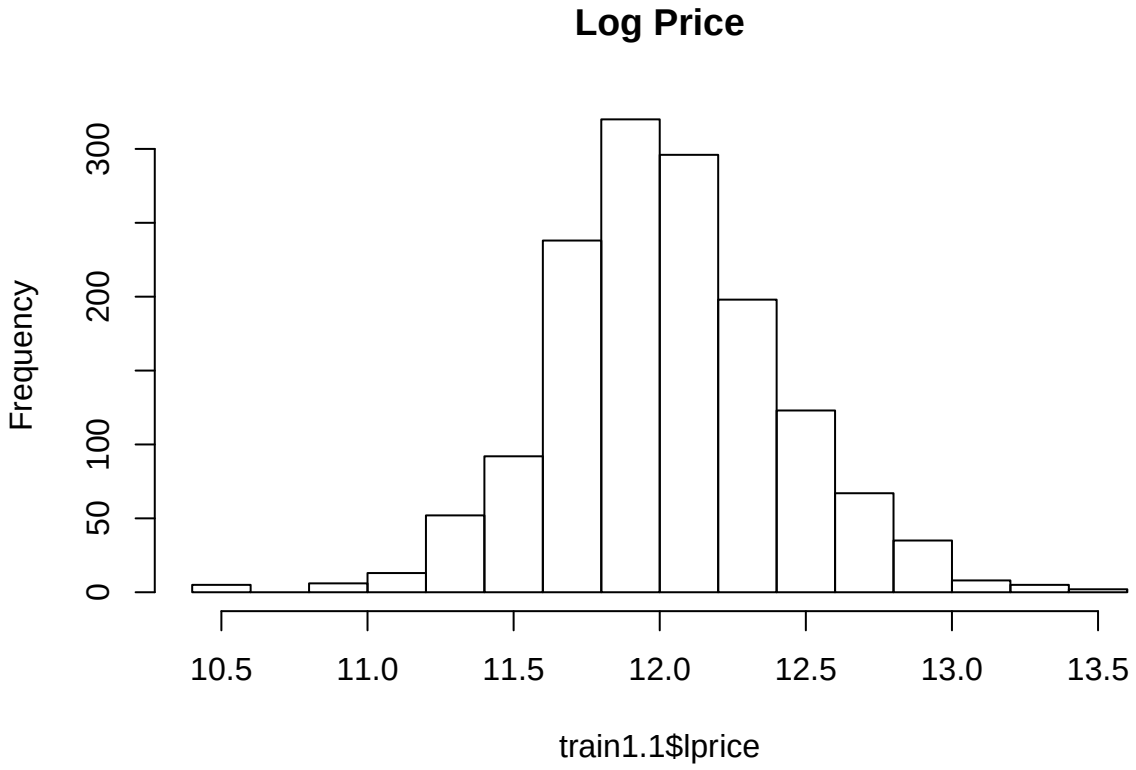

```
## Second Round
par(mfrow = c(3,2))
barplot(table(train1_rl$BldgType), main = "Type")
barplot(table(train1_rl$Utilities), main = "Utilities")
barplot(table(train1_rl$MSSubClass), main = "Sub Class")
barplot(table(train1_rl$GarageQual), main = "Garage Quality")
barplot(table(train1_rl$SaleType), main = "Sale Type")
barplot(table(train1_rl$KitchenQual), main = "Kitchen Quality")
```



There are a few points that stand out to me from the EDA. First, some of the predictors are highly collinear. Collinearity isn't a significant issue for prediction purposes, but will disrupt interpretation of the coefficients after the model is built. Secondly, the histograms in the ggpairs plots suggest some of the predictors could benefit from transformations. In the Ames paper, the author suggested a square-root transformation on sale price to improve accuracy. I will explore transformations of the predictors later. Lastly, some of the categorical variables have levels with few observations. I may compress some of the factors in the model building phase if the levels are insignificant. Many categorical variables also poses issues for degrees of freedom, but that is not a main concern here since our objective is not inference.

Modeling

```
# Creating a dataframe for regression
train1.1 <- train1_rl %>%
  mutate(lprice = log(SalePrice)) %>%
  dplyr::select(-SalePrice)

hist(train1.1$lprice, main = "Log Price")
```




```
## First Model
model1.1 <- lm(lprice ~., data = train1.1)
summary(model1.1)
```

```
##
## Call:
## lm(formula = lprice ~ ., data = train1.1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.68992 -0.04440  0.00000  0.04568  0.68992
##
## Coefficients: (7 not defined because of singularities)
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)           9.147e+00  4.649e+00   1.968 0.049363 *
## MSSubClass30         -6.947e-02  2.276e-02  -3.053 0.002322 **
## MSSubClass40         -7.872e-02  8.213e-02  -0.958 0.338056
## MSSubClass45         -1.729e-01  1.189e-01  -1.454 0.146132
## MSSubClass50         -3.502e-02  4.278e-02  -0.819 0.413223
## MSSubClass60         -6.009e-02  3.727e-02  -1.612 0.107156
## MSSubClass70         -1.781e-02  3.953e-02  -0.450 0.652440
## MSSubClass75         -4.639e-02  7.766e-02  -0.597 0.550412
## MSSubClass80         -1.089e-01  5.725e-02  -1.903 0.057275 .
## MSSubClass85         -6.410e-02  5.002e-02  -1.282 0.200276
## MSSubClass90         -7.904e-02  3.893e-02  -2.030 0.042583 *
## MSSubClass120        -5.784e-02  6.289e-02  -0.920 0.357905
## MSSubClass160        -1.824e-01  7.784e-02  -2.344 0.019273 *
## MSSubClass180        -1.219e-01  8.991e-02  -1.356 0.175311
## MSSubClass190         6.126e-02  1.335e-01   0.459 0.646366
## MSZoningFV            5.213e-01  5.708e-02   9.133  < 2e-16 ***
```

```
## MSZoningRH            4.636e-01  5.920e-02   7.832 1.12e-14 ***
## MSZoningRL            4.733e-01  5.010e-02   9.446  < 2e-16 ***
## MSZoningRM            4.261e-01  4.735e-02   9.000  < 2e-16 ***
## LotFrontage           3.927e-04  1.975e-04   1.988 0.047010 *
## LotArea               3.070e-06  4.868e-07   6.306 4.13e-10 ***
## StreetPave            1.742e-01  7.159e-02   2.433 0.015118 *
## AlleyGrvl            -4.541e-03  2.024e-02  -0.224 0.822511
## AlleyPave             2.526e-02  2.222e-02   1.137 0.255857
## LotShapeIR2           1.749e-02  1.816e-02   0.963 0.335696
## LotShapeIR3          -6.540e-04  3.869e-02  -0.017 0.986515
## LotShapeReg           4.274e-03  7.116e-03   0.601 0.548196
## LandContourHLS        2.063e-02  2.376e-02   0.868 0.385430
## LandContourLow       -2.654e-02  2.940e-02  -0.903 0.366846
## LandContourLvl        8.290e-03  1.754e-02   0.473 0.636571
## UtilitiesNoSeWa      -3.665e-01  1.240e-01  -2.955 0.003190 **
## LotConfigCulDSac      2.923e-02  1.437e-02   2.034 0.042205 *
## LotConfigFR2         -3.965e-02  1.775e-02  -2.234 0.025669 *
## LotConfigFR3         -6.783e-02  5.422e-02  -1.251 0.211236
## LotConfigInside      -1.386e-02  7.920e-03  -1.750 0.080327 .
## LandSlopeMod          3.511e-02  1.825e-02   1.924 0.054637 .
## LandSlopeSev         -2.004e-01  4.965e-02  -4.037 5.78e-05 ***
## NeighborhoodBlueste   3.796e-02  8.658e-02   0.438 0.661136
## NeighborhoodBrDale    2.265e-03  5.189e-02   0.044 0.965184
## NeighborhoodBrkSide   3.966e-02  4.373e-02   0.907 0.364576
## NeighborhoodClearCr   3.006e-02  4.079e-02   0.737 0.461359
## NeighborhoodCollgCr  -1.738e-02  3.162e-02  -0.550 0.582615
## NeighborhoodCrawfor   1.027e-01  3.790e-02   2.710 0.006832 **
## NeighborhoodEdwards  -9.554e-02  3.547e-02  -2.693 0.007184 **
## NeighborhoodGilbert  -9.630e-03  3.334e-02  -0.289 0.772723
## NeighborhoodIDOTRR    2.766e-02  4.981e-02   0.555 0.578799
## NeighborhoodMeadowV  -1.585e-01  5.477e-02  -2.893 0.003886 **
## NeighborhoodMitchel  -3.246e-02  3.604e-02  -0.901 0.367863
## NeighborhoodNAmes    -4.862e-02  3.435e-02  -1.415 0.157277
## NeighborhoodNoRidge   4.382e-02  3.671e-02   1.194 0.232825
## NeighborhoodNPkVill  -4.223e-03  6.033e-02  -0.070 0.944208
## NeighborhoodNridgHt   8.150e-02  3.250e-02   2.508 0.012299 *
## NeighborhoodNWAmes   -4.030e-02  3.501e-02  -1.151 0.249952
## NeighborhoodOldTown  -1.618e-02  4.404e-02  -0.368 0.713293
## NeighborhoodSawyer   -2.785e-02  3.560e-02  -0.782 0.434171
## NeighborhoodSawyerW   4.936e-03  3.419e-02   0.144 0.885232
## NeighborhoodSomerst   1.853e-02  3.923e-02   0.472 0.636799
## NeighborhoodStoneBr   1.442e-01  3.594e-02   4.013 6.39e-05 ***
## NeighborhoodSWISU    -1.827e-02  4.435e-02  -0.412 0.680460
## NeighborhoodTimber    9.316e-03  3.538e-02   0.263 0.792380
## NeighborhoodVeenker   5.887e-02  4.527e-02   1.300 0.193765
## Condition1Feedr       4.720e-02  2.343e-02   2.014 0.044212 *
## Condition1Norm        9.509e-02  1.937e-02   4.909 1.05e-06 ***
## Condition1PosA        5.164e-02  4.376e-02   1.180 0.238200
## Condition1PosN        1.152e-01  3.307e-02   3.484 0.000513 ***
## Condition1RRAe       -2.573e-02  4.002e-02  -0.643 0.520368
## Condition1RRAn        7.889e-02  3.095e-02   2.549 0.010932 *
## Condition1RRNe        1.534e-02  7.520e-02   0.204 0.838450
## Condition1RRNn        1.084e-01  5.630e-02   1.925 0.054458 .
## Condition2Feedr       5.663e-02  1.222e-01   0.464 0.643070
```

```
## Condition2Norm        1.287e-01  1.095e-01   1.176 0.240037
## Condition2PosA        3.379e-01  1.926e-01   1.754 0.079735 .
## Condition2PosN       -7.650e-01  1.355e-01  -5.647 2.07e-08 ***
## Condition2RRAe       -2.642e-01  3.322e-01  -0.795 0.426510
## Condition2RRAn        6.804e-02  1.519e-01   0.448 0.654334
## Condition2RRNn        8.710e-02  1.353e-01   0.644 0.519792
## BldgType2fmCon       -1.256e-01  1.304e-01  -0.964 0.335366
## BldgTypeDuplex               NA         NA      NA       NA
## BldgTypeTwnhs        -9.536e-03  6.755e-02  -0.141 0.887771
## BldgTypeTwnhsE        1.868e-02  6.389e-02   0.292 0.770012
## HouseStyle1.5Unf      9.622e-02  1.151e-01   0.836 0.403428
## HouseStyle1Story     -3.711e-02  4.342e-02  -0.855 0.392852
## HouseStyle2.5Fin     -4.866e-02  8.967e-02  -0.543 0.587492
## HouseStyle2.5Unf      5.159e-02  8.143e-02   0.634 0.526496
## HouseStyle2Story      5.814e-03  4.138e-02   0.140 0.888292
## HouseStyleSFoyer      9.528e-03  5.930e-02   0.161 0.872380
## HouseStyleSLvl        6.387e-02  6.512e-02   0.981 0.326848
## OverallQual           3.703e-02  4.665e-03   7.938 5.01e-15 ***
## OverallCond           3.658e-02  3.981e-03   9.189  < 2e-16 ***
## YearBuilt             1.638e-03  4.070e-04   4.025 6.08e-05 ***
## YearRemodAdd          6.227e-04  2.605e-04   2.390 0.017014 *
## RoofStyleGable       -1.455e-02  8.001e-02  -0.182 0.855780
## RoofStyleGambrel     -3.159e-02  8.900e-02  -0.355 0.722710
## RoofStyleHip         -1.683e-02  8.023e-02  -0.210 0.833914
## RoofStyleMansard      4.035e-02  9.596e-02   0.420 0.674230
## RoofStyleShed         4.884e-01  1.680e-01   2.907 0.003725 **
## RoofMatlCompShg       2.636e+00  2.300e-01  11.465  < 2e-16 ***
## RoofMatlMembran       3.009e+00  2.715e-01  11.082  < 2e-16 ***
## RoofMatlMetal         2.894e+00  2.702e-01  10.711  < 2e-16 ***
## RoofMatlRoll          2.721e+00  2.554e-01  10.653  < 2e-16 ***
## RoofMatlTar&Grv       2.648e+00  2.464e-01  10.744  < 2e-16 ***
## RoofMatlWdShake       2.592e+00  2.411e-01  10.754  < 2e-16 ***
## RoofMatlWdShngl       2.695e+00  2.338e-01  11.526  < 2e-16 ***
## Exterior1stBrkComm   -3.279e-01  1.247e-01  -2.629 0.008670 **
## Exterior1stBrkFace    6.778e-02  5.952e-02   1.139 0.255068
## Exterior1stCBlock    -8.556e-02  1.324e-01  -0.646 0.518171
## Exterior1stCemntBd   -1.021e-01  8.519e-02  -1.198 0.231117
## Exterior1stHdBoard   -1.550e-02  6.045e-02  -0.256 0.797683
## Exterior1stImStucc    1.702e-02  1.236e-01   0.138 0.890467
## Exterior1stMetalSd    6.238e-02  7.012e-02   0.890 0.373896
## Exterior1stPlywood   -1.718e-02  6.009e-02  -0.286 0.774933
## Exterior1stStone     -9.832e-03  1.095e-01  -0.090 0.928441
## Exterior1stStucco     2.503e-02  6.457e-02   0.388 0.698301
## Exterior1stVinylSd    2.181e-02  6.234e-02   0.350 0.726554
## Exterior1stWd Sdng   -3.708e-02  5.878e-02  -0.631 0.528291
## Exterior1stWdShing    1.315e-02  6.323e-02   0.208 0.835359
## Exterior2ndAsphShn    9.890e-02  1.007e-01   0.982 0.326469
## Exterior2ndBrk Cmn    1.810e-01  9.154e-02   1.978 0.048219 *
## Exterior2ndBrkFace    2.769e-02  6.062e-02   0.457 0.647965
## Exterior2ndCBlock            NA         NA      NA       NA
## Exterior2ndCmentBd    2.088e-01  8.333e-02   2.505 0.012386 *
## Exterior2ndHdBoard    7.464e-02  5.770e-02   1.294 0.196045
## Exterior2ndImStucc    6.966e-02  6.585e-02   1.058 0.290383
## Exterior2ndMetalSd    3.367e-02  6.763e-02   0.498 0.618665
```

```
## Exterior2ndOther     -5.020e-02  1.180e-01  -0.425 0.670576
## Exterior2ndPlywood    7.619e-02  5.641e-02   1.351 0.177064
## Exterior2ndStone      1.015e-01  8.194e-02   1.239 0.215763
## Exterior2ndStucco     5.788e-02  6.207e-02   0.933 0.351237
## Exterior2ndVinylSd    6.823e-02  5.939e-02   1.149 0.250849
## Exterior2ndWd Sdng    1.103e-01  5.592e-02   1.972 0.048860 *
## Exterior2ndWd Shng    6.296e-02  5.830e-02   1.080 0.280383
## MasVnrTypeBrkFace     6.729e-02  2.966e-02   2.268 0.023502 *
## MasVnrTypeNone        5.820e-02  2.992e-02   1.945 0.052049 .
## MasVnrTypeStone       7.917e-02  3.139e-02   2.522 0.011804 *
## MasVnrArea            2.766e-05  2.507e-05   1.103 0.270251
## ExterQualFa           5.623e-02  7.790e-02   0.722 0.470526
## ExterQualGd          -4.252e-03  2.067e-02  -0.206 0.837048
## ExterQualTA          -5.103e-03  2.314e-02  -0.221 0.825459
## ExterCondFa          -2.593e-02  1.120e-01  -0.232 0.816930
## ExterCondGd          -1.046e-02  1.085e-01  -0.096 0.923173
## ExterCondTA           1.920e-02  1.084e-01   0.177 0.859463
## FoundationCBlock      5.463e-03  1.565e-02   0.349 0.727151
## FoundationPConc       2.828e-02  1.643e-02   1.721 0.085569 .
## FoundationSlab        3.441e-02  4.794e-02   0.718 0.473135
## FoundationStone       6.724e-02  5.113e-02   1.315 0.188791
## FoundationWood       -1.498e-01  6.426e-02  -2.332 0.019905 *
## BsmtQualEx           -1.313e-01  1.571e-01  -0.836 0.403204
## BsmtQualFa           -1.403e-01  1.563e-01  -0.898 0.369580
## BsmtQualGd           -1.539e-01  1.561e-01  -0.986 0.324485
## BsmtQualTA           -1.533e-01  1.556e-01  -0.985 0.324988
## BsmtCondFa           -4.078e-02  1.970e-02  -2.070 0.038681 *
## BsmtCondGd           -3.847e-03  1.419e-02  -0.271 0.786265
## BsmtCondPo           -1.094e-01  1.663e-01  -0.658 0.510910
## BsmtCondTA                   NA         NA      NA       NA
## BsmtExposureAv        5.510e-02  9.811e-02   0.562 0.574467
## BsmtExposureGd        8.386e-02  9.857e-02   0.851 0.395073
## BsmtExposureMn        5.249e-02  9.844e-02   0.533 0.594010
## BsmtExposureNo        4.559e-02  9.791e-02   0.466 0.641583
## BsmtFinType1ALQ       9.857e-04  1.313e-02   0.075 0.940160
## BsmtFinType1BLQ       1.294e-02  1.414e-02   0.915 0.360405
## BsmtFinType1GLQ       2.059e-02  1.191e-02   1.729 0.084168 .
## BsmtFinType1LwQ      -4.713e-03  1.702e-02  -0.277 0.781926
## BsmtFinType1Rec       8.845e-03  1.443e-02   0.613 0.539889
## BsmtFinType1Unf              NA         NA      NA       NA
## BsmtFinSF1            1.463e-04  2.420e-05   6.045 2.03e-09 ***
## BsmtFinType2ALQ       1.358e-01  1.074e-01   1.264 0.206471
## BsmtFinType2BLQ       7.886e-02  1.065e-01   0.741 0.459002
## BsmtFinType2GLQ       1.326e-01  1.107e-01   1.198 0.231157
## BsmtFinType2LwQ       1.092e-01  1.065e-01   1.025 0.305546
## BsmtFinType2Rec       1.097e-01  1.063e-01   1.032 0.302319
## BsmtFinType2Unf       1.164e-01  1.060e-01   1.098 0.272330
## BsmtFinSF2            1.284e-04  3.982e-05   3.225 0.001297 **
## BsmtUnfSF             7.982e-05  2.263e-05   3.527 0.000437 ***
## TotalBsmtSF                  NA         NA      NA       NA
## HeatingGasA           1.580e-01  1.115e-01   1.417 0.156707
## HeatingGasW           2.289e-01  1.150e-01   1.991 0.046721 *
## HeatingGrav           8.802e-02  1.448e-01   0.608 0.543466
## HeatingOthW           2.061e-01  1.601e-01   1.287 0.198344
```

```
## HeatingWall           2.129e-01  1.301e-01   1.637 0.102018
## HeatingQCFa           2.642e-03  2.165e-02   0.122 0.902927
## HeatingQCGd          -1.747e-02  9.244e-03  -1.890 0.059080 .
## HeatingQCPo          -7.684e-02  1.175e-01  -0.654 0.513434
## HeatingQCTA          -2.974e-02  9.360e-03  -3.178 0.001526 **
## CentralAirY           8.241e-02  1.959e-02   4.206 2.81e-05 ***
## ElectricalFuseF      -7.187e-03  2.863e-02  -0.251 0.801848
## ElectricalFuseP      -2.040e-02  1.027e-01  -0.199 0.842573
## ElectricalMix                NA         NA      NA       NA
## ElectricalSBrkr      -1.026e-02  1.388e-02  -0.739 0.459810
## X1stFlrSF             2.206e-04  2.560e-05   8.619  < 2e-16 ***
## X2ndFlrSF             2.102e-04  2.600e-05   8.087 1.59e-15 ***
## LowQualFinSF          3.799e-05  1.045e-04   0.364 0.716209
## GrLivArea                    NA         NA      NA       NA
## BsmtFullBath          1.653e-02  8.859e-03   1.866 0.062369 .
## BsmtHalfBath          1.189e-02  1.331e-02   0.893 0.371961
## FullBath              2.239e-02  1.012e-02   2.212 0.027202 *
## HalfBath              2.342e-02  9.558e-03   2.450 0.014431 *
## BedroomAbvGr          3.111e-03  6.426e-03   0.484 0.628371
## KitchenAbvGr         -5.739e-02  3.403e-02  -1.687 0.091928 .
## KitchenQualFa        -6.890e-02  2.984e-02  -2.309 0.021137 *
## KitchenQualGd        -6.651e-02  1.522e-02  -4.370 1.36e-05 ***
## KitchenQualTA        -6.902e-02  1.739e-02  -3.968 7.70e-05 ***
## TotRmsAbvGrd          4.636e-03  4.276e-03   1.084 0.278543
## FunctionalMaj2       -1.749e-01  7.454e-02  -2.347 0.019113 *
## FunctionalMin1       -1.362e-02  4.054e-02  -0.336 0.736935
## FunctionalMin2       -2.407e-02  4.167e-02  -0.578 0.563664
## FunctionalMod        -8.222e-02  4.856e-02  -1.693 0.090755 .
## FunctionalSev        -2.629e-01  1.303e-01  -2.018 0.043864 *
## FunctionalTyp         1.781e-02  3.639e-02   0.489 0.624728
## Fireplaces            1.667e-02  1.118e-02   1.491 0.136235
## FireplaceQuEx        -1.298e-02  2.694e-02  -0.482 0.629964
## FireplaceQuFa        -1.313e-02  2.302e-02  -0.570 0.568476
## FireplaceQuGd         8.223e-03  1.506e-02   0.546 0.585134
## FireplaceQuPo        -1.234e-02  2.872e-02  -0.429 0.667685
## FireplaceQuTA         1.074e-02  1.558e-02   0.689 0.490771
## GarageTypeAttchd      9.541e-02  4.862e-02   1.962 0.049969 *
## GarageTypeBasment     1.168e-01  5.671e-02   2.059 0.039734 *
## GarageTypeBuiltIn     9.051e-02  5.066e-02   1.787 0.074255 .
## GarageTypeCarPort     1.182e-01  6.657e-02   1.775 0.076141 .
## GarageTypeDetchd      9.440e-02  4.850e-02   1.947 0.051828 .
## GarageYrBlt          -1.552e-04  2.774e-04  -0.559 0.575938
## GarageFinishRFn      -8.826e-04  8.494e-03  -0.104 0.917257
## GarageFinishUnf      -1.111e-02  1.053e-02  -1.055 0.291593
## GarageCars            2.117e-02  9.932e-03   2.132 0.033244 *
## GarageArea            1.309e-04  3.413e-05   3.834 0.000133 ***
## GarageQualFa         -4.170e-01  1.344e-01  -3.103 0.001965 **
## GarageQualGd         -3.384e-01  1.384e-01  -2.445 0.014643 *
## GarageQualPo         -4.120e-01  1.811e-01  -2.275 0.023097 *
## GarageQualTA         -3.651e-01  1.332e-01  -2.741 0.006216 **
## GarageCondFa          3.107e-01  1.538e-01   2.020 0.043574 *
## GarageCondGd          3.367e-01  1.607e-01   2.095 0.036434 *
## GarageCondPo          4.215e-01  1.679e-01   2.511 0.012184 *
## GarageCondTA          3.300e-01  1.527e-01   2.161 0.030879 *
```

```
## PavedDriveP          -1.907e-02  2.668e-02  -0.715 0.474777
## PavedDriveY           1.156e-02  1.743e-02   0.663 0.507170
## WoodDeckSF            9.600e-05  2.568e-05   3.739 0.000194 ***
## OpenPorchSF           7.723e-05  5.363e-05   1.440 0.150135
## EnclosedPorch         9.612e-05  5.667e-05   1.696 0.090131 .
## X3SsnPorch            1.493e-04  9.573e-05   1.559 0.119227
## ScreenPorch           2.604e-04  5.405e-05   4.817 1.66e-06 ***
## PoolArea              1.117e-03  1.001e-03   1.116 0.264565
## PoolQCEx             -5.119e-01  5.481e-01  -0.934 0.350545
## PoolQCFa             -6.556e-01  6.587e-01  -0.995 0.319816
## PoolQCGd             -4.612e-01  6.460e-01  -0.714 0.475361
## FenceGdPrv           -6.304e-03  1.604e-02  -0.393 0.694341
## FenceGdWo            -2.377e-02  1.592e-02  -1.493 0.135847
## FenceMnPrv           -3.617e-03  1.024e-02  -0.353 0.723911
## FenceMnWw            -2.249e-02  3.216e-02  -0.699 0.484520
## MiscFeatureGar2      -4.219e-01  5.145e-01  -0.820 0.412412
## MiscFeatureOthr       1.435e-01  1.304e-01   1.101 0.271144
## MiscFeatureShed      -2.370e-02  2.914e-02  -0.813 0.416338
## MiscFeatureTenC      -1.341e-01  2.081e-01  -0.645 0.519251
## MiscVal               3.027e-05  3.265e-05   0.927 0.354065
## MoSold               -1.042e-03  1.090e-03  -0.956 0.339380
## YrSold               -3.111e-03  2.281e-03  -1.364 0.172986
## SaleTypeCon           9.767e-02  7.523e-02   1.298 0.194453
## SaleTypeConLD         1.386e-01  4.874e-02   2.844 0.004543 **
## SaleTypeConLI        -5.406e-02  5.529e-02  -0.978 0.328402
## SaleTypeConLw        -3.119e-02  5.600e-02  -0.557 0.577632
## SaleTypeCWD           5.888e-02  5.525e-02   1.066 0.286775
## SaleTypeNew           9.045e-02  6.717e-02   1.347 0.178378
## SaleTypeOth           1.240e-01  1.039e-01   1.193 0.233248
## SaleTypeWD           -1.728e-02  1.821e-02  -0.949 0.343025
## SaleConditionAdjLand  2.192e-01  1.087e-01   2.017 0.043902 *
## SaleConditionAlloca   2.953e-02  4.485e-02   0.659 0.510338
## SaleConditionFamily   1.150e-02  2.657e-02   0.433 0.665266
## SaleConditionNormal   5.466e-02  1.328e-02   4.115 4.16e-05 ***
## SaleConditionPartial -1.617e-02  6.468e-02  -0.250 0.802645
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.09635 on 1110 degrees of freedom
##   (90 observations deleted due to missingness)
## Multiple R-squared:  0.9479, Adjusted R-squared:  0.9358
## F-statistic: 78.03 on 259 and 1110 DF,  p-value: < 2.2e-16
```

Most of the variables in this model are factors, which will make subset selection techniques hard to use. Additionally, the data are highly collinear, and the predictor space is high dimensional. Because of this, shrinkage techniques will perform well on this problem. Thus, I am going to compare the efficacy of Lasso and Ridge regression. I will use the caret package to perform cross-validation to select the best model and tune hyperparameters.

Shrinkage Methods

In this first code chunk, I am going to use cross-validation to determine a good starting value for lambda for lasso and ridge regression.

```r
set.seed(123)

# Reformatting Data
train1.2 <- train1.1 %>% drop_na()
X <- model.matrix(model1.1)[,-1]
Y <- train1.2 %>%
  dplyr::select(lprice) %>%
  as.matrix()

# Determining starting values for lambda
cv.glmnet(X, Y, alpha = 1)$lambda.min # Lasso
```

```
## [1] 0.006207057
```

```r
cv.glmnet(X, Y, alpha = 0)$lambda.min # Ridge
```

```
## [1] 0.2391907
```

Lasso

I am using 10 fold cross-validation to tune the hyperparameter lambda. This tuning process performs the variable selection–selecting the correct value of lambda will "turn off" some of the coefficients and leave us with a sparse model. The caret::train performs this through a grid-search prcedure and cross-validation. For each potential value in the grid, caret performs 10-fold cross validation, and then reports the best model and estimated coefficients. All of this is abstracted from the user and makes for some very clean and readable code.

```r
# Lasso
lasso_lambda <- seq(0, 1, length = 100)
train_control <- trainControl(method = "cv", number = 10)

lasso_caret <- train(
  lprice ~.,
  data = train1.2,
  method = "glmnet",
  trControl = train_control,
  tuneGrid = expand.grid(alpha = 1, lambda = lasso_lambda)
)
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info =
## trainInfo, : There were missing values in resampled performance measures.
```

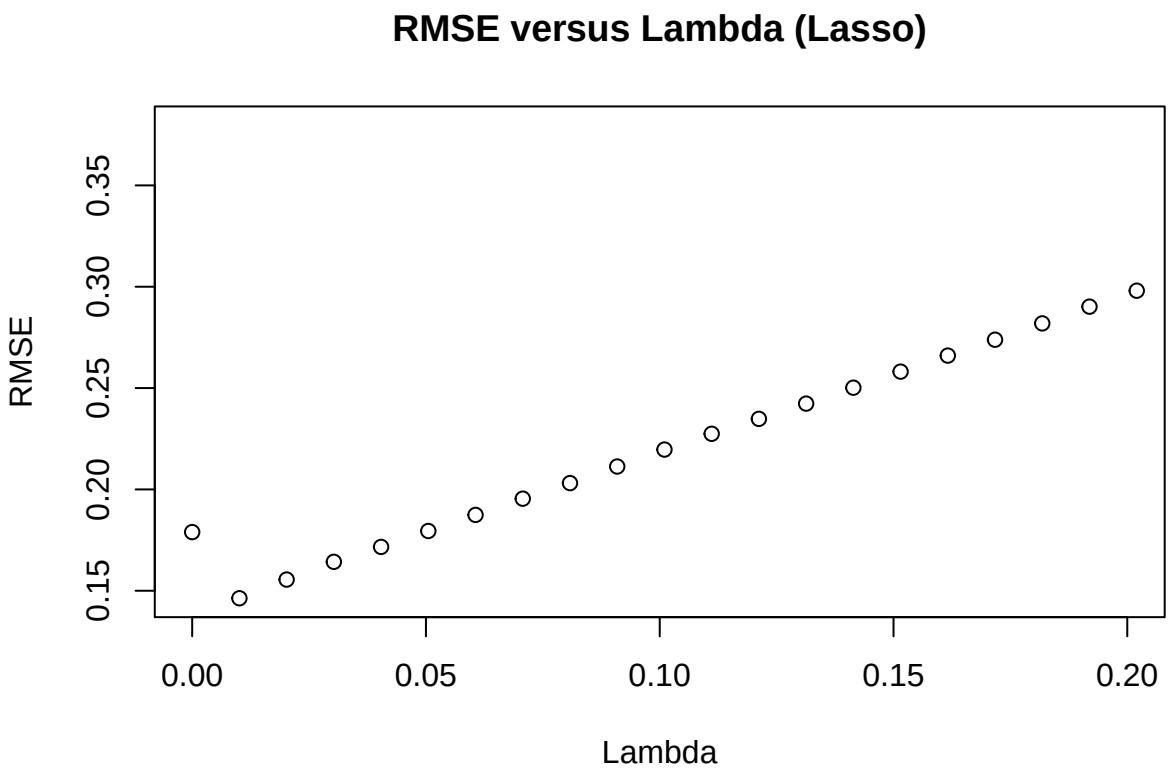
```r
# Best model summary information
lasso_caret$bestTune$lambda
```

```
## [1] 0.01010101
```

```r
sum(coef(lasso_caret$finalModel, lasso_caret$bestTune$lambda) != 0)
```

```
## [1] 63
```

```r
## Lambda versus RMSE
plot(lasso_caret$results$lambda, lasso_caret$results$RMSE,
     xlab = "Lambda", ylab = "RMSE", main = "RMSE versus Lambda (Lasso)",
     xlim = c(0, 0.2))
```



As you can see from the above output, Lasso reduced the number of parameters from 273 to 63! This is a substantial reduciton in the number of features. Now on to Ridge.

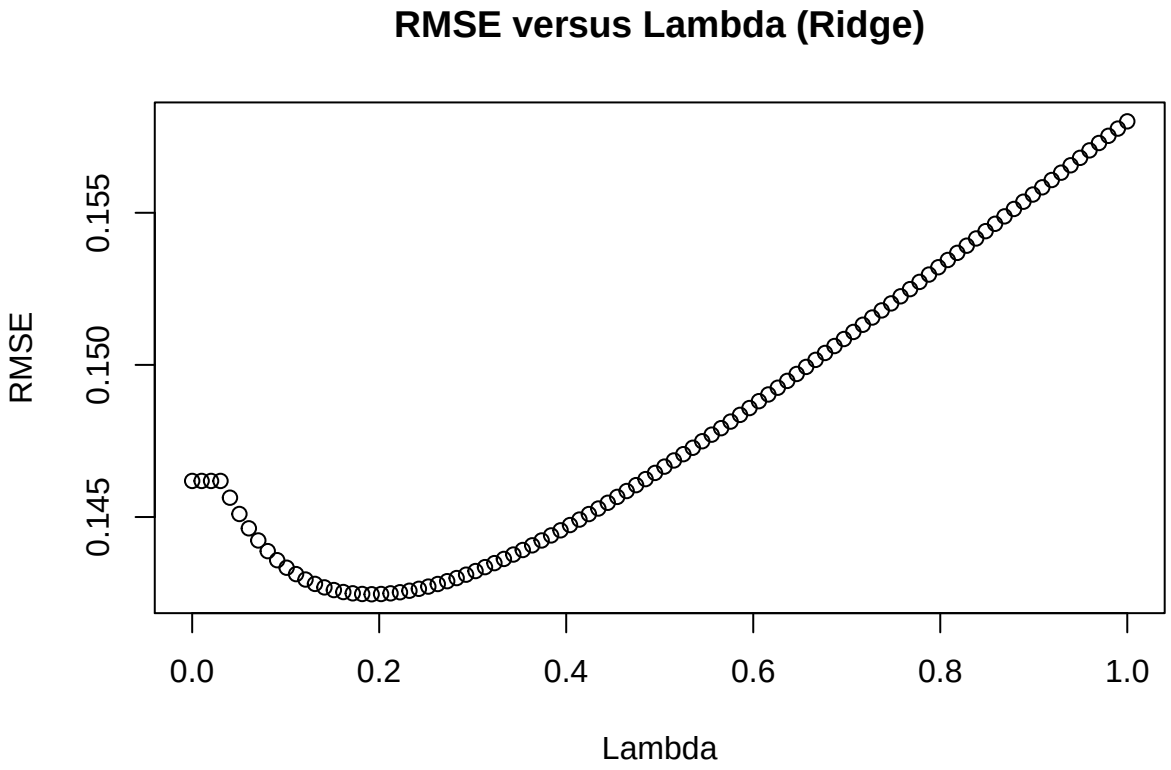
Ridge

Ridge is very similar to lasso, except it will not shrink coefficients to zero. I am going to employ the same selection process as above.

```
# Ridge Regression
ridge_lambda <- seq(0, 1, length = 100)

ridge_caret <- train(
  lprice ~.,
  data = train1.2,
  method = "glmnet",
  trControl = train_control,
  tuneGrid = expand.grid(alpha = 0, lambda = ridge_lambda)
)

# Best model summary information
## Best Lambda
ridge_caret$bestTune$lambda
```

```
## [1] 0.1919192
```

```
## Lambda versus RMSE
plot(ridge_caret$results$lambda, ridge_caret$results$RMSE,
     xlab = "Lambda", ylab = "RMSE", main = "RMSE versus Lambda (Ridge)")
```




```
## Best Ridge Model Coefficients
sum(coef(ridge_caret$finalModel, ridge_caret$bestTune$lambda) != 0)
```

```
## [1] 271
```

Testing Lasso and Ridge

Now that I have estimated the coefficients and selected the best model for the shrinkage methods, it is time to calculate test error.

```
# Cleaning up the test dataframe
test1_features <- test1_features %>%
  mutate(MSSubClass = factor(MSSubClass),
         Alley = fct.relevel(Alley),
         BsmtQual = fct.relevel(BsmtQual),
         BsmtCond = fct.relevel(BsmtCond),
         BsmtExposure = fct.relevel(BsmtExposure),
         BsmtFinType1 = fct.relevel(BsmtFinType1),
         BsmtFinType2 = fct.relevel(BsmtFinType2),
         FireplaceQu = fct.relevel(FireplaceQu),
         GarageType = fct.relevel(GarageType),
         GarageFinish = fct.relevel(GarageFinish),
         GarageCond = fct.relevel(GarageCond),
         GarageQual = fct.relevel(GarageQual),
         PoolQC = fct.relevel(PoolQC),
         Fence = fct.relevel(Fence),
         MiscFeature = fct.relevel(MiscFeature),
         LotFrontage = ifelse(is.na(LotFrontage), median(LotFrontage, na.rm = T),
                              LotFrontage)) %>%
  filter(MSSubClass != "150")
```

```
test_df <- left_join(test_target, test1_features, by = "Id") %>%
  drop_na()

# Making Predictions
lasso_preds <- predict(lasso_caret, test_df)
ridge_preds <- predict(ridge_caret, test_df)

# Calculating MSE
rmse_lasso <- sqrt(mean((exp(lasso_preds) - test_df$SalePrice)^2))
rmse_ridge <- sqrt(mean((exp(ridge_preds) - test_df$SalePrice)^2))
rmse1 <- matrix(cbind(rmse_lasso, rmse_ridge), dimnames = list(c("Lasso", "Ridge"), "RMSE"))

rmse1
```

```
##           RMSE
## Lasso 67357.66
## Ridge 68850.31
```

Lasso regression has a slightly higher RMSE than ridge. However, Lasso produces a more parsimonious model, shrinking the predictor space down to only 63 features. It is important to note that ordinary least squares was already evaluated in the training process since the grid search started with a lambda value of 0. If OLS produced the model with the lowest training MSE, it would have been selected at that stage.

Lambda Tuning

After determining Lasso to be the best methodology for this problem, I wanted to tune lambda to minimize the test error.

```
# Lasso
lasso_lambda_tune <- seq(0.75, 1, length = 10)
train_control_tune <- trainControl(method = "cv", number = 10)

lasso_caret_tune <- train(
  lprice ~.,
  data = train1.2,
  method = "glmnet",
  trControl = train_control_tune,
  tuneGrid = expand.grid(alpha = 1, lambda = lasso_lambda_tune)
)
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info =
## trainInfo, : There were missing values in resampled performance measures.
```

```
lasso_preds_tune <- predict(lasso_caret_tune, test_df)

# RMSE
sqrt(mean((exp(lasso_preds_tune) - test_df$SalePrice)^2))
```

```
## [1] 17475
```

```
# Number of non-zero coefficients
sum(coef(lasso_caret_tune$finalModel, lasso_caret_tune$bestTune$lambda) != 0)
```

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

I noticed larger values of lambda decreased the test error (see above output). So much so that nearly all of

the coefficients were being "turned off", and there was only one remaining predictor. I then build a naive model of just the average house price in the train dataset and have reported the RMSE below.

```
train_price_mean <- mean(train1$SalePrice)
mean_diffs_squared <- (train_price_mean - test_df$SalePrice)^2
MSE_mean <- mean(mean_diffs_squared)
sqrt(MSE_mean)
```

```
## [1] 16020.81
```

Amazingly, just using the mean value to predict the SalePrice in the test dataset produces the lowest RMSE. I think this is a result of differences in the correlation structure of the variables in the training and test sets. This is an extreme form of variable selection where no covariates are retained in the final model.

Problem 2

```
# Reading Data
train2 <- read.delim("~/Documents/Graduate School/Fall 2019/MATH 624/Midterm Project/train_GE_LW.txt")
test2 <- read.delim("~/Documents/Graduate School/Fall 2019/MATH 624/Midterm Project/test_GE_LW.txt")

dim(train2)
```

```
## [1] 4313   29
```

```
dim(test2)
```

```
## [1] 4313   29
```

```
#head(train2)
#head(test2)

str(train2$COS.Intensity)
```

```
##  Factor w/ 4 levels "LPA","MPA","SED",..: 3 3 3 3 3 3 3 3 3 3 ...
```

```
#str(test2)

train2 <- train2 %>%
  mutate(COS.Intensity = fct_relevel(COS.Intensity, c("SED", "LPA", "MPA", "VPA")))
test2 <- test2 %>%
  mutate(COS.Intensity = fct_relevel(COS.Intensity, c("SED", "LPA", "MPA", "VPA")))
```

For this part of the project, I am assuming the user of the model has an interest in classifying physical activity into the four levels. Thus, I am not going to group levels of COS.Intensity.

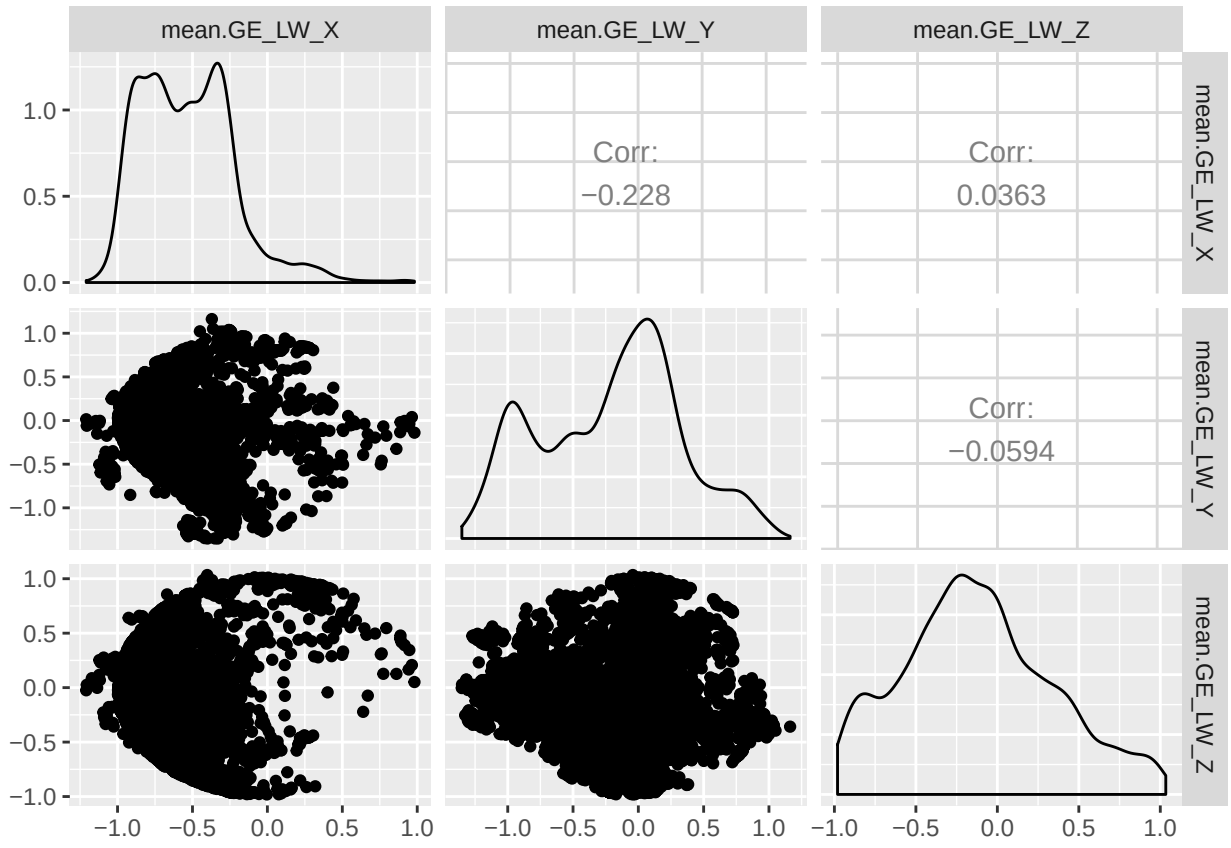
Exploratory Data Analysis

```
summary(test2)
```

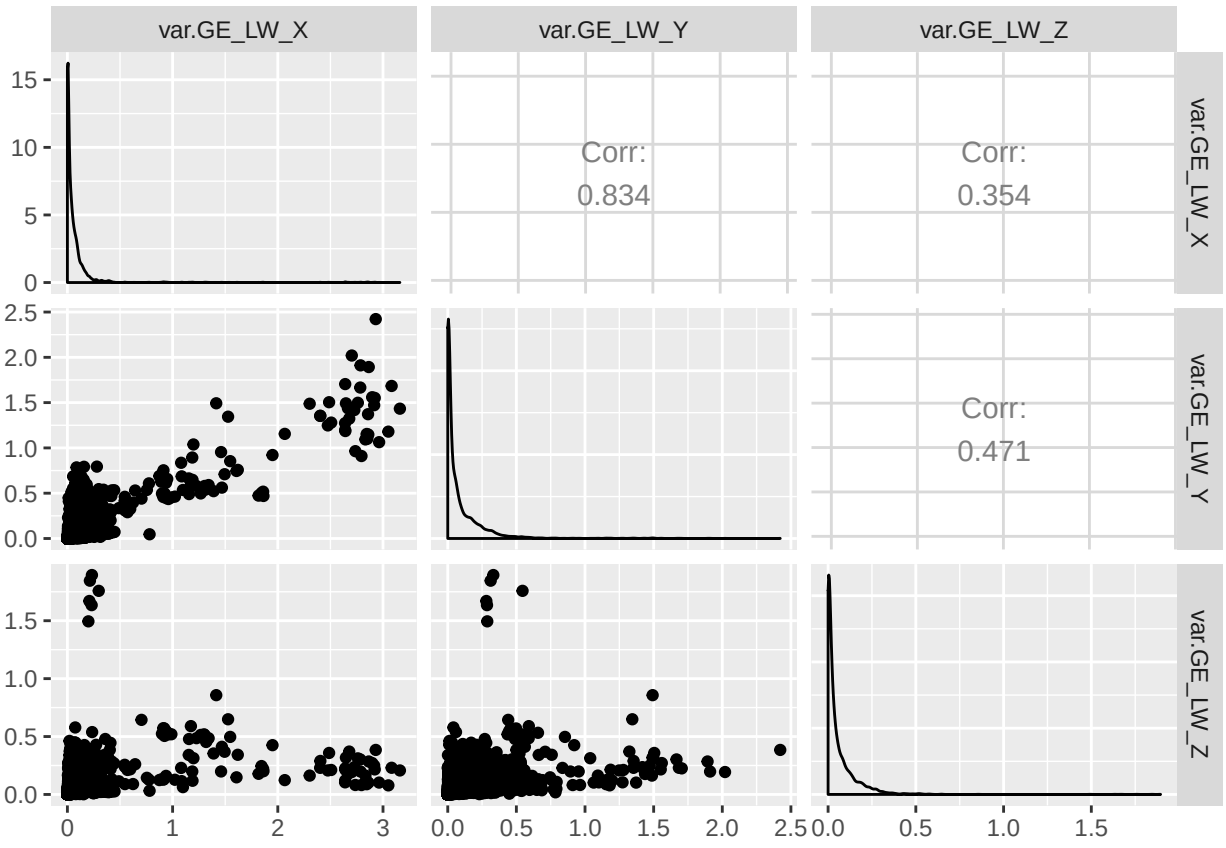
```
##   mean.GE_LW_X       var.GE_LW_X          max.GE_LW_X       min.GE_LW_X
##  Min.   :-1.2092   Min.   :0.0000288   Min.   :-0.9752   Min.   :-8.2102
##  1st Qu.:-0.7838   1st Qu.:0.0024913   1st Qu.:-0.3730   1st Qu.:-1.8153
##  Median :-0.5550   Median :0.0258686   Median : 0.1463   Median :-1.2166
##  Mean   :-0.5363   Mean   :0.0855144   Mean   : 0.2484   Mean   :-1.5298
##  3rd Qu.:-0.3326   3rd Qu.:0.0756567   3rd Qu.: 0.6767   3rd Qu.:-0.9090
```

```
##  Max.   : 0.9814   Max.   :3.1604705   Max.   : 8.1456   Max.   : 0.9163
##  quarlite.GE_LW_X70 quarlite.GE_LW_X80 quarlite.GE_LW_X90
##  Min.   :-0.9950    Min.   :-0.9910    Min.   :-0.9910
##  1st Qu.:-0.7255    1st Qu.:-0.6899    1st Qu.:-0.6387
##  Median :-0.4325    Median :-0.3698    Median :-0.2927
##  Mean   :-0.4429    Mean   :-0.3916    Mean   :-0.3139
##  3rd Qu.:-0.2061    3rd Qu.:-0.1450    3rd Qu.:-0.0583
##  Max.   : 0.9962    Max.   : 0.9962    Max.   : 1.0721
##   mean.GE_LW_Y      var.GE_LW_Y         max.GE_LW_Y       min.GE_LW_Y
##  Min.   :-1.3511   Min.   :0.0000277   Min.   :-0.9525   Min.   :-8.1845
##  1st Qu.:-0.6480   1st Qu.:0.0013239   1st Qu.: 0.0286   1st Qu.:-1.7720
##  Median :-0.1308   Median :0.0261734   Median : 0.4373   Median :-0.7700
##  Mean   :-0.2108   Mean   :0.0909575   Mean   : 0.5244   Mean   :-1.0592
##  3rd Qu.: 0.1567   3rd Qu.:0.1110885   3rd Qu.: 0.8946   3rd Qu.:-0.1026
##  Max.   : 1.1621   Max.   :2.4221250   Max.   : 8.1257   Max.   : 0.8686
##  quarlite.GE_LW_Y70 quarlite.GE_LW_Y80 quarlite.GE_LW_Y90
##  Min.   :-1.2223    Min.   :-1.11400   Min.   :-0.99130
##  1st Qu.:-0.5532    1st Qu.:-0.44100   1st Qu.:-0.29560
##  Median :-0.0274    Median : 0.01640   Median : 0.07960
##  Mean   :-0.1093    Mean   :-0.04342   Mean   : 0.05291
##  3rd Qu.: 0.2366    3rd Qu.: 0.29220   3rd Qu.: 0.41370
##  Max.   : 1.3550    Max.   : 1.46880   Max.   : 1.62970
##   mean.GE_LW_Z      var.GE_LW_Z         max.GE_LW_Z       min.GE_LW_Z
##  Min.   :-0.9807   Min.   :0.0000567   Min.   :-0.9466   Min.   :-8.0669
##  1st Qu.:-0.4451   1st Qu.:0.0026452   1st Qu.: 0.0523   1st Qu.:-1.2390
##  Median :-0.1508   Median :0.0236952   Median : 0.5412   Median :-0.8025
##  Mean   :-0.1200   Mean   :0.0597044   Mean   : 0.6068   Mean   :-0.9054
##  3rd Qu.: 0.1876   3rd Qu.:0.0786449   3rd Qu.: 1.0294   3rd Qu.:-0.3919
##  Max.   : 1.0337   Max.   :1.8945059   Max.   : 7.9844   Max.   : 0.9636
##  quarlite.GE_LW_Z70 quarlite.GE_LW_Z80 quarlite.GE_LW_Z90
##  Min.   :-0.97820   Min.   :-0.97430   Min.   :-0.9703
##  1st Qu.:-0.37210   1st Qu.:-0.32480   1st Qu.:-0.2353
##  Median :-0.03840   Median : 0.02430   Median : 0.1046
##  Mean   :-0.03465   Mean   : 0.02057   Mean   : 0.1019
##  3rd Qu.: 0.29430   3rd Qu.: 0.36140   3rd Qu.: 0.4731
##  Max.   : 1.40077   Max.   : 2.17048   Max.   : 3.5071
##  corri.GE_LW_XY     corri.GE_LW_XZ     corri.GE_LW_YZ         Sex
##  Min.   :-0.98946   Min.   :-0.97733   Min.   :-0.96738   Female:2254
##  1st Qu.:-0.28372   1st Qu.:-0.33620   1st Qu.:-0.22459   Male  :2059
##  Median :-0.03814   Median :-0.03039   Median : 0.01262
##  Mean   :-0.02102   Mean   :-0.01920   Mean   : 0.02506
##  3rd Qu.: 0.24358   3rd Qu.: 0.31996   3rd Qu.: 0.27928
##  Max.   : 0.98122   Max.   : 0.99390   Max.   : 0.98555
##       Age              Ht              Wt        COS.Intensity
##  Min.   :18.00   Min.   :62.80   Min.   : 94.8   SED:1687
##  1st Qu.:29.00   1st Qu.:64.90   1st Qu.:150.4   LPA:1108
##  Median :51.00   Median :68.40   Median :167.8   MPA:1256
##  Mean   :48.34   Mean   :68.31   Mean   :174.6   VPA: 262
##  3rd Qu.:62.00   3rd Qu.:71.50   3rd Qu.:207.6
##  Max.   :79.00   Max.   :74.50   Max.   :238.6
```

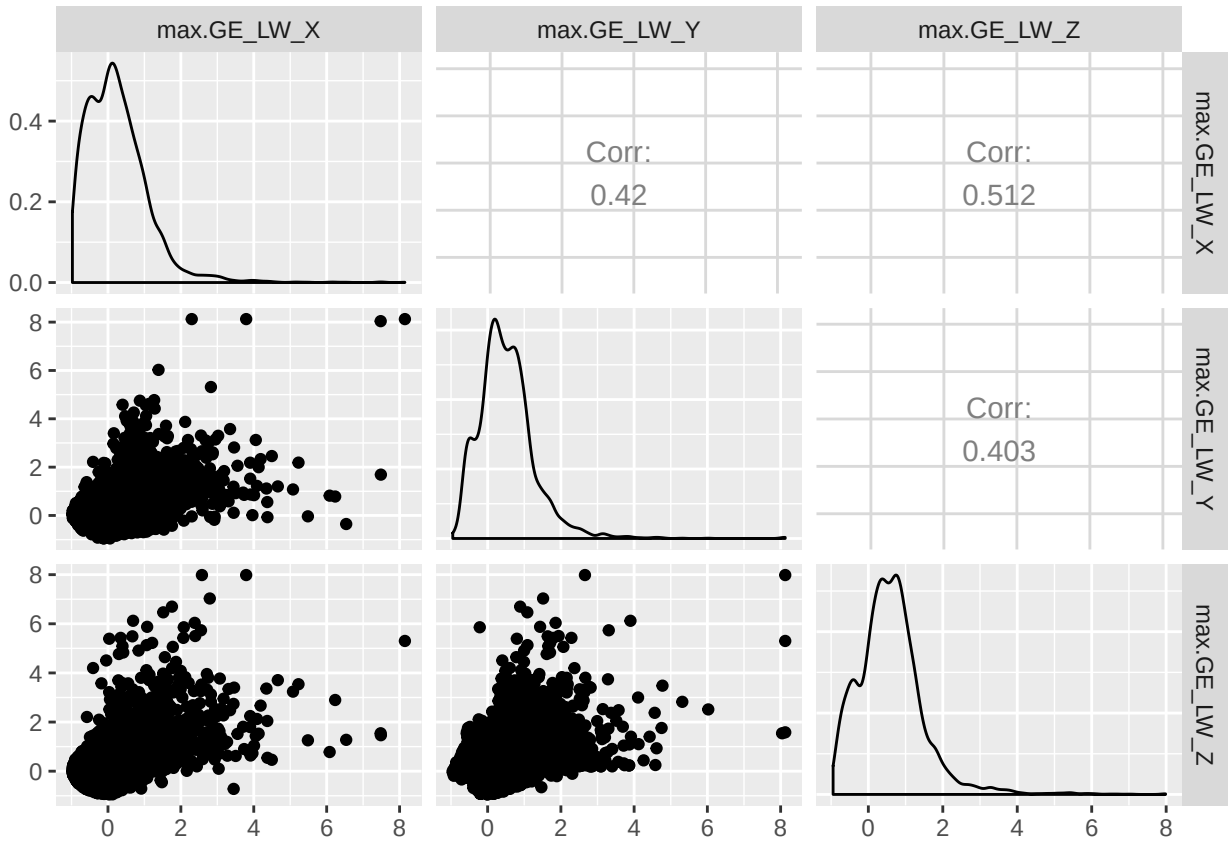
```
test2 %>% dplyr::select(mean.GE_LW_X, mean.GE_LW_Y, mean.GE_LW_Z) %>% ggpairs()
```



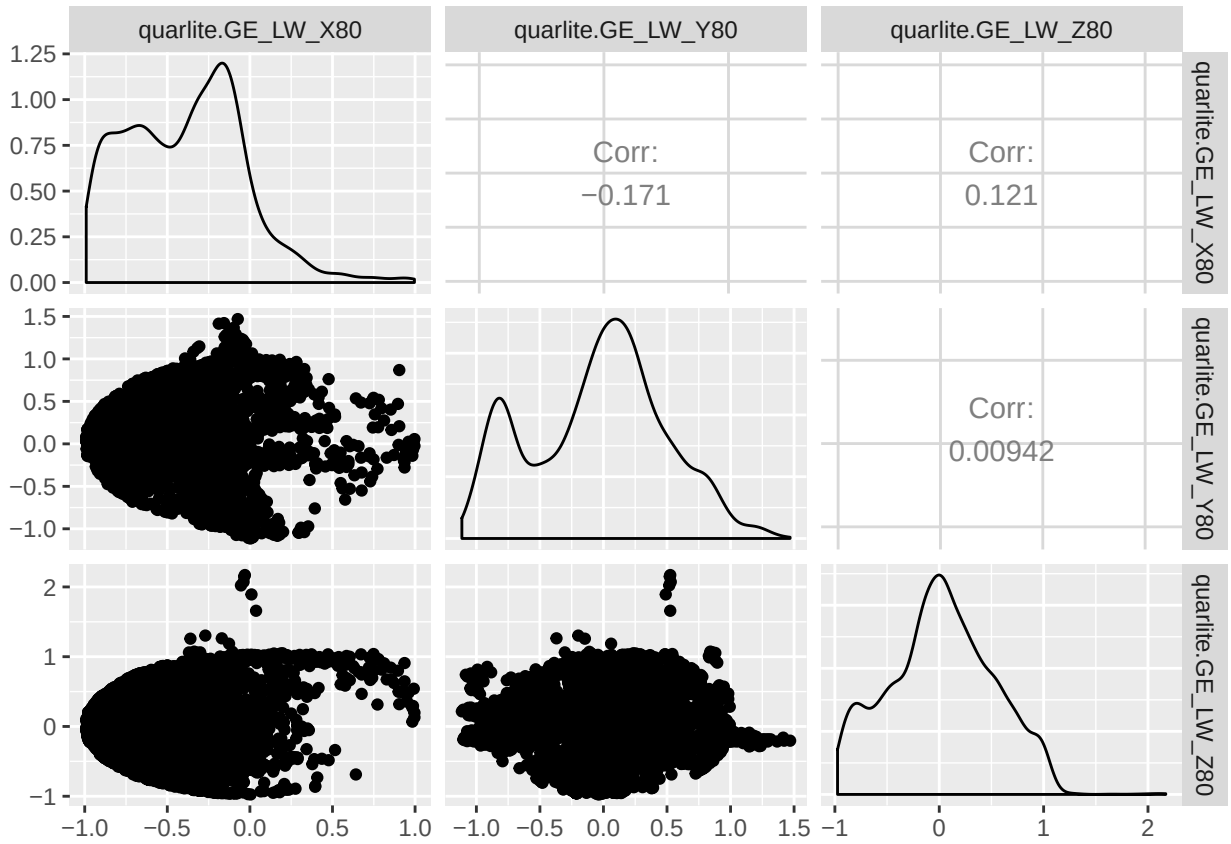

```
test2 %>% dplyr::select(var.GE_LW_X, var.GE_LW_Y, var.GE_LW_Z) %>% ggpairs()
```




```
test2 %>% dplyr::select(max.GE_LW_X, max.GE_LW_Y, max.GE_LW_Z) %>% ggpairs()
```




```
test2 %>% dplyr::select(quarlite.GE_LW_X80, quarlite.GE_LW_Y80, quarlite.GE_LW_Z80) %>% ggpairs()
```




```
test2 %>% dplyr::select(-25, -29) %>% cor()
```

```
##                    mean.GE_LW_X var.GE_LW_X  max.GE_LW_X min.GE_LW_X
## mean.GE_LW_X        1.000000000 -0.14969452  0.421746149  0.28107036
## var.GE_LW_X        -0.149694521  1.00000000  0.360225498 -0.59575452
## max.GE_LW_X         0.421746149  0.36022550  1.000000000 -0.49507015
## min.GE_LW_X         0.281070360 -0.59575452 -0.495070154  1.00000000
## quarlite.GE_LW_X70  0.916269309  0.20726271  0.579604614  0.02039913
## quarlite.GE_LW_X80  0.865736606  0.29824115  0.638796525 -0.06973686
## quarlite.GE_LW_X90  0.784729740  0.38436605  0.705388074 -0.18087239
## mean.GE_LW_Y       -0.227571530 -0.11577775 -0.311409209  0.15199486
## var.GE_LW_Y        -0.060112256  0.83433021  0.499752820 -0.64318849
## max.GE_LW_Y        -0.035292371  0.32677009  0.419707746 -0.47852923
## min.GE_LW_Y        -0.139646495 -0.45302379 -0.675945038  0.61690649
## quarlite.GE_LW_Y70 -0.230972554  0.05010209 -0.181843263  0.00681198
## quarlite.GE_LW_Y80 -0.227541903  0.13190768 -0.100532911 -0.07730833
## quarlite.GE_LW_Y90 -0.211354387  0.21288282  0.009734338 -0.18690680
## mean.GE_LW_Z        0.036313672  0.02575325  0.018296652 -0.01221167
## var.GE_LW_Z        -0.058999049  0.35417279  0.418115482 -0.49571833
## max.GE_LW_Z         0.058321910  0.34936700  0.512006091 -0.53470288
## min.GE_LW_Z        -0.004667681 -0.21391123 -0.539255866  0.53272055
## quarlite.GE_LW_Z70  0.025464943  0.07385559  0.100138866 -0.09504877
## quarlite.GE_LW_Z80  0.020571866  0.11118971  0.152591755 -0.15279165
## quarlite.GE_LW_Z90  0.012281892  0.16497105  0.220158524 -0.23515339
## corri.GE_LW_XY     -0.055202585  0.12618633  0.026058841 -0.10640323
## corri.GE_LW_XZ      0.072682635 -0.04928924 -0.011031047  0.04095728
## corri.GE_LW_YZ     -0.003119719 -0.04562095  0.028277219 -0.05259895
```

```
## Age                -0.003349502 -0.12678373 -0.116000895  0.08938301
## Ht                 -0.154066678  0.05810194 -0.110769769 -0.03643448
## Wt                 -0.087357863  0.01501143 -0.065353397 -0.05052955
##                    quarlite.GE_LW_X70 quarlite.GE_LW_X80
## mean.GE_LW_X              0.916269309        0.865736606
## var.GE_LW_X               0.207262712        0.298241151
## max.GE_LW_X               0.579604614        0.638796525
## min.GE_LW_X               0.020399130       -0.069736856
## quarlite.GE_LW_X70        1.000000000        0.984027942
## quarlite.GE_LW_X80        0.984027942        1.000000000
## quarlite.GE_LW_X90        0.931783694        0.971796710
## mean.GE_LW_Y             -0.306443722       -0.322573010
## var.GE_LW_Y               0.261176195        0.353911378
## max.GE_LW_Y               0.105157187        0.162220928
## min.GE_LW_Y              -0.350250089       -0.417232098
## quarlite.GE_LW_Y70       -0.231324516       -0.225741621
## quarlite.GE_LW_Y80       -0.192135362       -0.171401077
## quarlite.GE_LW_Y90       -0.139653116       -0.105340209
## mean.GE_LW_Z              0.050900873        0.057127401
## var.GE_LW_Z               0.131914421        0.201481103
## max.GE_LW_Z               0.225105693        0.288190661
## min.GE_LW_Z              -0.120076356       -0.171882393
## quarlite.GE_LW_Z70        0.074836499        0.095125284
## quarlite.GE_LW_Z80        0.090951543        0.121270956
## quarlite.GE_LW_Z90        0.109567078        0.150327222
## corri.GE_LW_XY           -0.022701132       -0.004319119
## corri.GE_LW_XZ            0.057817022        0.045137851
## corri.GE_LW_YZ           -0.006117944       -0.008708129
## Age                      -0.031498018       -0.043756442
## Ht                       -0.145093377       -0.144968119
## Wt                       -0.088752533       -0.094969629
##                    quarlite.GE_LW_X90 mean.GE_LW_Y var.GE_LW_Y
## mean.GE_LW_X              0.784729740  -0.22757153 -0.06011226
## var.GE_LW_X               0.384366048  -0.11577775  0.83433021
## max.GE_LW_X               0.705388074  -0.31140921  0.49975282
## min.GE_LW_X              -0.180872386   0.15199486 -0.64318849
## quarlite.GE_LW_X70        0.931783694  -0.30644372  0.26117619
## quarlite.GE_LW_X80        0.971796710  -0.32257301  0.35391138
## quarlite.GE_LW_X90        1.000000000  -0.33433195  0.44674079
## mean.GE_LW_Y             -0.334331945   1.00000000 -0.17379502
## var.GE_LW_Y               0.446740793  -0.17379502  1.00000000
## max.GE_LW_Y               0.234033466   0.40230077  0.49634186
## min.GE_LW_Y              -0.492885102   0.62556179 -0.61967843
## quarlite.GE_LW_Y70       -0.217910804   0.96295053  0.03594281
## quarlite.GE_LW_Y80       -0.147873467   0.92502250  0.16006318
## quarlite.GE_LW_Y90       -0.058442390   0.85083548  0.29739973
## mean.GE_LW_Z              0.058204434  -0.05943874  0.05977825
## var.GE_LW_Z               0.276126264  -0.04180215  0.47101655
## max.GE_LW_Z               0.357268536  -0.20654398  0.49027105
## min.GE_LW_Z              -0.244433753   0.09765448 -0.35308890
## quarlite.GE_LW_Z70        0.111612881  -0.07496984  0.13304387
## quarlite.GE_LW_Z80        0.148070654  -0.08799975  0.18701113
## quarlite.GE_LW_Z90        0.191900314  -0.10624594  0.26024489
## corri.GE_LW_XY            0.025800235   0.19279492  0.08491196
```

```
## corri.GE_LW_XZ            0.032842613  -0.13423543 -0.04457852
## corri.GE_LW_YZ           -0.008792169   0.04937356 -0.04409385
## Age                      -0.063840859   0.05139319 -0.15339346
## Ht                       -0.142284941  -0.02765875  0.03111620
## Wt                       -0.097857111   0.01762458 -0.01141742
##                       max.GE_LW_Y  min.GE_LW_Y quarlite.GE_LW_Y70
## mean.GE_LW_X       -0.035292371 -0.139646495       -0.230972554
## var.GE_LW_X         0.326770088 -0.453023785        0.050102088
## max.GE_LW_X         0.419707746 -0.675945038       -0.181843263
## min.GE_LW_X        -0.478529233  0.616906489        0.006811980
## quarlite.GE_LW_X70  0.105157187 -0.350250089       -0.231324516
## quarlite.GE_LW_X80  0.162220928 -0.417232098       -0.225741621
## quarlite.GE_LW_X90  0.234033466 -0.492885102       -0.217910804
## mean.GE_LW_Y        0.402300774  0.625561793        0.962950530
## var.GE_LW_Y         0.496341860 -0.619678432        0.035942807
## max.GE_LW_Y         1.000000000 -0.196218173        0.506631340
## min.GE_LW_Y        -0.196218173  1.000000000        0.477564931
## quarlite.GE_LW_Y70  0.506631340  0.477564931        1.000000000
## quarlite.GE_LW_Y80  0.578234854  0.389263544        0.980637628
## quarlite.GE_LW_Y90  0.676340784  0.266182229        0.917506057
## mean.GE_LW_Z        0.001396161 -0.052456818       -0.039136999
## var.GE_LW_Z         0.413822706 -0.403333707        0.093785300
## max.GE_LW_Z         0.402604318 -0.526190946       -0.082555003
## min.GE_LW_Z        -0.407750246  0.519387636        0.007420875
## quarlite.GE_LW_Z70  0.068751618 -0.127642662       -0.026966940
## quarlite.GE_LW_Z80  0.117067216 -0.178405711       -0.025145011
## quarlite.GE_LW_Z90  0.183166099 -0.246945425       -0.026511076
## corri.GE_LW_XY      0.136776102  0.041970159        0.210067659
## corri.GE_LW_XZ     -0.109624380 -0.053406903       -0.130866666
## corri.GE_LW_YZ      0.064910104 -0.000511067        0.040865222
## Age                -0.061749100  0.112051066        0.029643440
## Ht                 -0.073712436  0.006648272       -0.023084296
## Wt                 -0.031537905  0.017955965        0.014548882
##                    quarlite.GE_LW_Y80 quarlite.GE_LW_Y90 mean.GE_LW_Z
## mean.GE_LW_X             -0.227541903       -0.211354387  0.036313672
## var.GE_LW_X               0.131907682        0.212882819  0.025753246
## max.GE_LW_X              -0.100532911        0.009734338  0.018296652
## min.GE_LW_X              -0.077308329       -0.186906804 -0.012211670
## quarlite.GE_LW_X70       -0.192135362       -0.139653116  0.050900873
## quarlite.GE_LW_X80       -0.171401077       -0.105340209  0.057127401
## quarlite.GE_LW_X90       -0.147873467       -0.058442390  0.058204434
## mean.GE_LW_Y              0.925022502        0.850835476 -0.059438736
## var.GE_LW_Y               0.160063175        0.297399728  0.059778245
## max.GE_LW_Y               0.578234854        0.676340784  0.001396161
## min.GE_LW_Y               0.389263544        0.266182229 -0.052456818
## quarlite.GE_LW_Y70        0.980637628        0.917506057 -0.039136999
## quarlite.GE_LW_Y80        1.000000000        0.961830933 -0.027012976
## quarlite.GE_LW_Y90        0.961830933        1.000000000 -0.015318865
## mean.GE_LW_Z             -0.027012976       -0.015318865  1.000000000
## var.GE_LW_Z               0.163317505        0.243198858  0.104114624
## max.GE_LW_Z              -0.007004663        0.089408673  0.594012262
## min.GE_LW_Z              -0.049627415       -0.135191053  0.446398989
## quarlite.GE_LW_Z70       -0.002773865        0.021543473  0.978539362
## quarlite.GE_LW_Z80        0.009417586        0.045057619  0.957713290
```

```
## quarlite.GE_LW_Z90        0.020247786        0.075727386  0.911211595
## corri.GE_LW_XY            0.214799550        0.223200682  0.096580044
## corri.GE_LW_XZ           -0.136518979       -0.140294240  0.322482688
## corri.GE_LW_YZ            0.038915741        0.043088880 -0.029601911
## Age                       0.011773380       -0.012512670  0.127773822
## Ht                       -0.029025557       -0.037804835  0.017539287
## Wt                        0.004185702       -0.006280385 -0.049396624
##                     var.GE_LW_Z  max.GE_LW_Z  min.GE_LW_Z
## mean.GE_LW_X       -0.05899905  0.058321910 -0.004667681
## var.GE_LW_X         0.35417279  0.349366998 -0.213911230
## max.GE_LW_X         0.41811548  0.512006091 -0.539255866
## min.GE_LW_X        -0.49571833 -0.534702883  0.532720549
## quarlite.GE_LW_X70  0.13191442  0.225105693 -0.120076356
## quarlite.GE_LW_X80  0.20148110  0.288190661 -0.171882393
## quarlite.GE_LW_X90  0.27612626  0.357268536 -0.244433753
## mean.GE_LW_Y       -0.04180215 -0.206543981  0.097654485
## var.GE_LW_Y         0.47101655  0.490271045 -0.353088899
## max.GE_LW_Y         0.41382271  0.402604318 -0.407750246
## min.GE_LW_Y        -0.40333371 -0.526190946  0.519387636
## quarlite.GE_LW_Y70  0.09378530 -0.082555003  0.007420875
## quarlite.GE_LW_Y80  0.16331751 -0.007004663 -0.049627415
## quarlite.GE_LW_Y90  0.24319886  0.089408673 -0.135191053
## mean.GE_LW_Z        0.10411462  0.594012262  0.446398989
## var.GE_LW_Z         1.00000000  0.545621526 -0.390134160
## max.GE_LW_Z         0.54562153  1.000000000 -0.137253153
## min.GE_LW_Z        -0.39013416 -0.137253153  1.000000000
## quarlite.GE_LW_Z70  0.23889175  0.649606572  0.354700211
## quarlite.GE_LW_Z80  0.33289610  0.699616246  0.296826347
## quarlite.GE_LW_Z90  0.45236355  0.760954121  0.210445842
## corri.GE_LW_XY      0.02331628  0.074153036  0.018069129
## corri.GE_LW_XZ     -0.09450305  0.097830752  0.151337355
## corri.GE_LW_YZ      0.06924180  0.039975000 -0.092088935
## Age                -0.05780997 -0.004874301  0.117931603
## Ht                 -0.03284038 -0.026250726  0.052330883
## Wt                 -0.02061826 -0.058742369 -0.019939538
##                    quarlite.GE_LW_Z70 quarlite.GE_LW_Z80
## mean.GE_LW_X              0.025464943       0.0205718665
## var.GE_LW_X               0.073855594       0.1111897056
## max.GE_LW_X               0.100138866       0.1525917546
## min.GE_LW_X              -0.095048773      -0.1527916480
## quarlite.GE_LW_X70        0.074836499       0.0909515435
## quarlite.GE_LW_X80        0.095125284       0.1212709563
## quarlite.GE_LW_X90        0.111612881       0.1480706537
## mean.GE_LW_Y             -0.074969838      -0.0879997455
## var.GE_LW_Y               0.133043866       0.1870111257
## max.GE_LW_Y               0.068751618       0.1170672157
## min.GE_LW_Y              -0.127642662      -0.1784057106
## quarlite.GE_LW_Y70       -0.026966940      -0.0251450106
## quarlite.GE_LW_Y80       -0.002773865       0.0094175858
## quarlite.GE_LW_Y90        0.021543473       0.0450576190
## mean.GE_LW_Z              0.978539362       0.9577132897
## var.GE_LW_Z               0.238891750       0.3328960956
## max.GE_LW_Z               0.649606572       0.6996162457
## min.GE_LW_Z               0.354700211       0.2968263468
```

```
## quarlite.GE_LW_Z70        1.000000000       0.9885550411
## quarlite.GE_LW_Z80        0.988555041       1.0000000000
## quarlite.GE_LW_Z90        0.948029157       0.9780592628
## corri.GE_LW_XY            0.095904914       0.0954921595
## corri.GE_LW_XZ            0.306729823       0.2828961504
## corri.GE_LW_YZ           -0.022629461      -0.0109345665
## Age                       0.112535529       0.1022402300
## Ht                        0.005751639      -0.0002167297
## Wt                       -0.053383537      -0.0579945354
##                     quarlite.GE_LW_Z90 corri.GE_LW_XY corri.GE_LW_XZ
## mean.GE_LW_X             0.0122818922   -0.055202585     0.07268263
## var.GE_LW_X              0.1649710529    0.126186332    -0.04928924
## max.GE_LW_X              0.2201585245    0.026058841    -0.01103105
## min.GE_LW_X             -0.2351533865   -0.106403232     0.04095728
## quarlite.GE_LW_X70       0.1095670776   -0.022701132     0.05781702
## quarlite.GE_LW_X80       0.1503272216   -0.004319119     0.04513785
## quarlite.GE_LW_X90       0.1919003138    0.025800235     0.03284261
## mean.GE_LW_Y            -0.1062459372    0.192794915    -0.13423543
## var.GE_LW_Y              0.2602448936    0.084911964    -0.04457852
## max.GE_LW_Y              0.1831660993    0.136776102    -0.10962438
## min.GE_LW_Y             -0.2469454248    0.041970159    -0.05340690
## quarlite.GE_LW_Y70      -0.0265110762    0.210067659    -0.13086667
## quarlite.GE_LW_Y80       0.0202477857    0.214799550    -0.13651898
## quarlite.GE_LW_Y90       0.0757273860    0.223200682    -0.14029424
## mean.GE_LW_Z             0.9112115952    0.096580044     0.32248269
## var.GE_LW_Z              0.4523635498    0.023316276    -0.09450305
## max.GE_LW_Z              0.7609541210    0.074153036     0.09783075
## min.GE_LW_Z              0.2104458418    0.018069129     0.15133735
## quarlite.GE_LW_Z70       0.9480291572    0.095904914     0.30672982
## quarlite.GE_LW_Z80       0.9780592628    0.095492159     0.28289615
## quarlite.GE_LW_Z90       1.0000000000    0.095752943     0.24434183
## corri.GE_LW_XY           0.0957529430    1.000000000     0.12541667
## corri.GE_LW_XZ           0.2443418258    0.125416668     1.00000000
## corri.GE_LW_YZ           0.0091674581    0.017994470     0.01837627
## Age                      0.0866146366   -0.043264744     0.06624651
## Ht                      -0.0006133082   -0.026698577     0.07323011
## Wt                      -0.0572070307    0.026306819     0.03696791
##                     corri.GE_LW_YZ          Age           Ht           Wt
## mean.GE_LW_X        -0.003119719 -0.003349502 -0.1540666782 -0.087357863
## var.GE_LW_X         -0.045620954 -0.126783729  0.0581019444  0.015011428
## max.GE_LW_X          0.028277219 -0.116000895 -0.1107697691 -0.065353397
## min.GE_LW_X         -0.052598949  0.089383005 -0.0364344838 -0.050529548
## quarlite.GE_LW_X70  -0.006117944 -0.031498018 -0.1450933767 -0.088752533
## quarlite.GE_LW_X80  -0.008708129 -0.043756442 -0.1449681195 -0.094969629
## quarlite.GE_LW_X90  -0.008792169 -0.063840859 -0.1422849405 -0.097857111
## mean.GE_LW_Y         0.049373559  0.051393194 -0.0276587511  0.017624579
## var.GE_LW_Y         -0.044093846 -0.153393458  0.0311162048 -0.011417417
## max.GE_LW_Y          0.064910104 -0.061749100 -0.0737124358 -0.031537905
## min.GE_LW_Y         -0.000511067  0.112051066  0.0066482715  0.017955965
## quarlite.GE_LW_Y70   0.040865222  0.029643440 -0.0230842961  0.014548882
## quarlite.GE_LW_Y80   0.038915741  0.011773380 -0.0290255575  0.004185702
## quarlite.GE_LW_Y90   0.043088880 -0.012512670 -0.0378048350 -0.006280385
## mean.GE_LW_Z        -0.029601911  0.127773822  0.0175392872 -0.049396624
## var.GE_LW_Z          0.069241803 -0.057809970 -0.0328403848 -0.020618259
```

```
## max.GE_LW_Z           0.039975000 -0.004874301 -0.0262507261 -0.058742369
## min.GE_LW_Z          -0.092088935  0.117931603  0.0523308831 -0.019939538
## quarlite.GE_LW_Z70   -0.022629461  0.112535529  0.0057516394 -0.053383537
## quarlite.GE_LW_Z80   -0.010934566  0.102240230 -0.0002167297 -0.057994535
## quarlite.GE_LW_Z90    0.009167458  0.086614637 -0.0006133082 -0.057207031
## corri.GE_LW_XY        0.017994470 -0.043264744 -0.0266985774  0.026306819
## corri.GE_LW_XZ        0.018376271  0.066246506  0.0732301097  0.036967909
## corri.GE_LW_YZ        1.000000000 -0.001360113 -0.0185001606  0.048340773
## Age                  -0.001360113  1.000000000 -0.0999158829  0.017950552
## Ht                   -0.018500161 -0.099915883  1.0000000000  0.606922629
## Wt                    0.048340773  0.017950552  0.6069226289  1.000000000
```

```
table(test2$COS.Intensity)
```

```
##
##  SED  LPA  MPA  VPA
## 1687 1108 1256  262
```

```
table(test2$Sex)
```

```
##
## Female   Male
##   2254   2059
```

```
train2 %>%
  group_by(COS.Intensity) %>%
  drop_na() %>%
  dplyr::select(-Sex) %>%
  summarize_all(mean)
```

```
## Warning: Factor `COS.Intensity` contains implicit NA, consider using
## `forcats::fct_explicit_na`
```

```
## # A tibble: 4 x 28
##   COS.Intensity mean.GE_LW_X var.GE_LW_X max.GE_LW_X min.GE_LW_X
##   <fct>                <dbl>       <dbl>       <dbl>       <dbl>
## 1 SED                 -0.519      0.0242      -0.187      -0.948
## 2 LPA                 -0.434      0.0676       0.559      -1.64
## 3 MPA                 -0.398      0.106        0.585      -1.61
## 4 VPA                 -0.605      0.779        1.14       -3.29
## # ... with 23 more variables: quarlite.GE_LW_X70 <dbl>,
## #   quarlite.GE_LW_X80 <dbl>, quarlite.GE_LW_X90 <dbl>,
## #   mean.GE_LW_Y <dbl>, var.GE_LW_Y <dbl>, max.GE_LW_Y <dbl>,
## #   min.GE_LW_Y <dbl>, quarlite.GE_LW_Y70 <dbl>, quarlite.GE_LW_Y80 <dbl>,
## #   quarlite.GE_LW_Y90 <dbl>, mean.GE_LW_Z <dbl>, var.GE_LW_Z <dbl>,
## #   max.GE_LW_Z <dbl>, min.GE_LW_Z <dbl>, quarlite.GE_LW_Z70 <dbl>,
## #   quarlite.GE_LW_Z80 <dbl>, quarlite.GE_LW_Z90 <dbl>,
## #   corri.GE_LW_XY <dbl>, corri.GE_LW_XZ <dbl>, corri.GE_LW_YZ <dbl>,
## #   Age <dbl>, Ht <dbl>, Wt <dbl>
```

```
train2 %>%
  group_by(COS.Intensity) %>%
  drop_na() %>%
  dplyr::select(-Sex) %>%
  summarize_all(sd)
```

```
## Warning: Factor `COS.Intensity` contains implicit NA, consider using
## `forcats::fct_explicit_na`
```

```
## # A tibble: 4 x 28
##   COS.Intensity mean.GE_LW_X var.GE_LW_X max.GE_LW_X min.GE_LW_X
##   <fct>                <dbl>       <dbl>       <dbl>       <dbl>
## 1 SED                  0.426      0.120        0.684       0.834
## 2 LPA                  0.362      0.0831       0.900       1.10
## 3 MPA                  0.300      0.232        0.748       1.06
## 4 VPA                  0.417      0.829        0.932       1.71
## # ... with 23 more variables: quarlite.GE_LW_X70 <dbl>,
## #   quarlite.GE_LW_X80 <dbl>, quarlite.GE_LW_X90 <dbl>,
## #   mean.GE_LW_Y <dbl>, var.GE_LW_Y <dbl>, max.GE_LW_Y <dbl>,
## #   min.GE_LW_Y <dbl>, quarlite.GE_LW_Y70 <dbl>, quarlite.GE_LW_Y80 <dbl>,
## #   quarlite.GE_LW_Y90 <dbl>, mean.GE_LW_Z <dbl>, var.GE_LW_Z <dbl>,
## #   max.GE_LW_Z <dbl>, min.GE_LW_Z <dbl>, quarlite.GE_LW_Z70 <dbl>,
## #   quarlite.GE_LW_Z80 <dbl>, quarlite.GE_LW_Z90 <dbl>,
## #   corri.GE_LW_XY <dbl>, corri.GE_LW_XZ <dbl>, corri.GE_LW_YZ <dbl>,
## #   Age <dbl>, Ht <dbl>, Wt <dbl>
```

Looking at the results of the EDA, the main problem with this data is collinearity. This won't be an issue for KNN, LDA, and QDA, but will pose substantial challenges for logistic regression.

KNN

There are two methods of scaling that can be used for KNN. I am going to first try scaling and then normalization.

```
# Creating the training and test dataframes
train2.1 <- scale(train2[, c(-25, -29)]) %>%
  cbind(train2[, c(25, 29)]) %>%
  mutate(Sex = ifelse(as.character(Sex) == "Female", 0, 1))  %>%
  drop_na()


test2.1 <- scale(test2[, c(-25, -29)]) %>%
  cbind(test2[, c(25, 29)]) %>%
  mutate(Sex = ifelse(as.character(Sex) == "Female", 0, 1))

#summary(train2.1)
#summary(test2.1)
#dim(train2.1)
#dim(test2.1)

# Separating features and targets
train2.1_features <- train2.1[, -29]
train2.1_target <- train2.1[, 29]

test2.1_features <- test2.1[,-29]
test2.1_target <- test2.1[,29]

# Clustering
k <- round(sqrt(nrow(train2.1))) - 1

model2.1 <- knn(train2.1_features, test2.1_features, train2.1_target, k)

# Examining results
```

```r
confusionMatrix(model2.1, test2.1_target)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  SED  LPA  MPA  VPA
##        SED 1222  392  132    5
##        LPA  425  554  398   28
##        MPA   39  162  716  139
##        VPA    1    0   10   90
##
## Overall Statistics
##
##                Accuracy : 0.5987
##                  95% CI : (0.5838, 0.6133)
##     No Information Rate : 0.3911
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4139
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: SED Class: LPA Class: MPA Class: VPA
## Sensitivity              0.7244     0.5000     0.5701    0.34351
## Specificity              0.7986     0.7345     0.8888    0.99728
## Pos Pred Value           0.6979     0.3943     0.6780    0.89109
## Neg Pred Value           0.8185     0.8095     0.8342    0.95916
## Prevalence               0.3911     0.2569     0.2912    0.06075
## Detection Rate           0.2833     0.1284     0.1660    0.02087
## Detection Prevalence     0.4060     0.3258     0.2448    0.02342
## Balanced Accuracy        0.7615     0.6172     0.7294    0.67040
```

Looking at the accuracy rates and kappa statistic, this model has moderate performance. I am going to employ regularization to see if it improves accuracy.

```r
normalize <- function(x){
  return((x - min(x)) / (max(x) - min(x)))
}

# Creating the training and test dataframes
train2.2 <- lapply(train2[, c(-25, -29)], normalize) %>%
  cbind(train2[, c(25, 29)]) %>%
  mutate(Sex = ifelse(as.character(Sex) == "Female", 0, 1)) %>%
  drop_na()


test2.2 <- lapply(test2[, c(-25, -29)], normalize) %>%
  cbind(test2[, c(25, 29)]) %>%
  mutate(Sex = ifelse(as.character(Sex) == "Female", 0, 1))

#summary(train2.2)
#summary(test2.2)
#dim(train2.2)
```

```r
#dim(test2.2)

# Separating features and targets
train2.2_features <- train2.2[, -29]
train2.2_target <- train2.2[, 29]

test2.2_features <- test2.2[,-29]
test2.2_target <- test2.2[,29]

# Clustering
k <- round(sqrt(nrow(train2.2))) - 1

model2.2 <- knn(train2.2_features, test2.2_features, train2.2_target, k)

# Examining results
confusionMatrix(model2.2, test2.2_target)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  SED  LPA  MPA  VPA
##        SED 1025  329  137    5
##        LPA  591  577  362   32
##        MPA   71  202  749  141
##        VPA    0    0    8   84
##
## Overall Statistics
##
##                Accuracy : 0.5646
##                  95% CI : (0.5496, 0.5794)
##     No Information Rate : 0.3911
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3703
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: SED Class: LPA Class: MPA Class: VPA
## Sensitivity              0.6076     0.5208     0.5963    0.32061
## Specificity              0.8206     0.6927     0.8646    0.99803
## Pos Pred Value           0.6852     0.3694     0.6440    0.91304
## Neg Pred Value           0.7650     0.8070     0.8390    0.95783
## Prevalence               0.3911     0.2569     0.2912    0.06075
## Detection Rate           0.2377     0.1338     0.1737    0.01948
## Detection Prevalence     0.3469     0.3622     0.2696    0.02133
## Balanced Accuracy        0.7141     0.6067     0.7305    0.65932
```

The normalized data performed worse in this case; I will use the centered and scaled data. One additional consideration is tuning the k parameter. The models are currently built using a heuristic for k, so I am going to perform a search.

Tuning k

```
# Centered and Scaled Data
k_tune <- seq(from = 3, to = 199, by = 2)
sum((k_tune %/% 2) == 0)
```

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

```
tune2.1_kappa <- rep(NA, length(k_tune))
tune2.1_acc <- rep(NA, length(k_tune))

for(i in 1:length(k_tune)){
  model <- knn(train2.1_features, test2.1_features, train2.1_target, k_tune[i])
  tune2.1_acc[i] <- confusionMatrix(model, test2.1_target)$overall[1]
  tune2.1_kappa[i] <- confusionMatrix(model, test2.1_target)$overall[2]
}

# Normalized data
tune2.2_kappa <- rep(NA, length(k_tune))
tune2.2_acc <- rep(NA, length(k_tune))

for(i in 1:length(k_tune)){
  model <- knn(train2.2_features, test2.2_features, train2.2_target, k_tune[i])
  tune2.2_acc[i] <- confusionMatrix(model, test2.2_target)$overall[1]
  tune2.2_kappa[i] <- confusionMatrix(model, test2.2_target)$overall[2]
}

tune2.1_kappa[which.max(tune2.1_kappa)]
```

```
## [1] 0.4216087
```

```
# Best K's
max_2.1 <- which.max(tune2.1_kappa)
max_2.2 <- which.max(tune2.2_kappa)

# Alignment between Kappa and Accuracy
which.max(tune2.1_acc) == max_2.1
```

```
## [1] FALSE
```

```
k_tune[max_2.1]; k_tune[max_2.2]
```

```
## [1] 13
```

```
## [1] 31
```

```
tune2.1_kappa[max_2.1]; tune2.2_kappa[max_2.2]
```

```
## [1] 0.4216087
```

```
## [1] 0.3770527
```

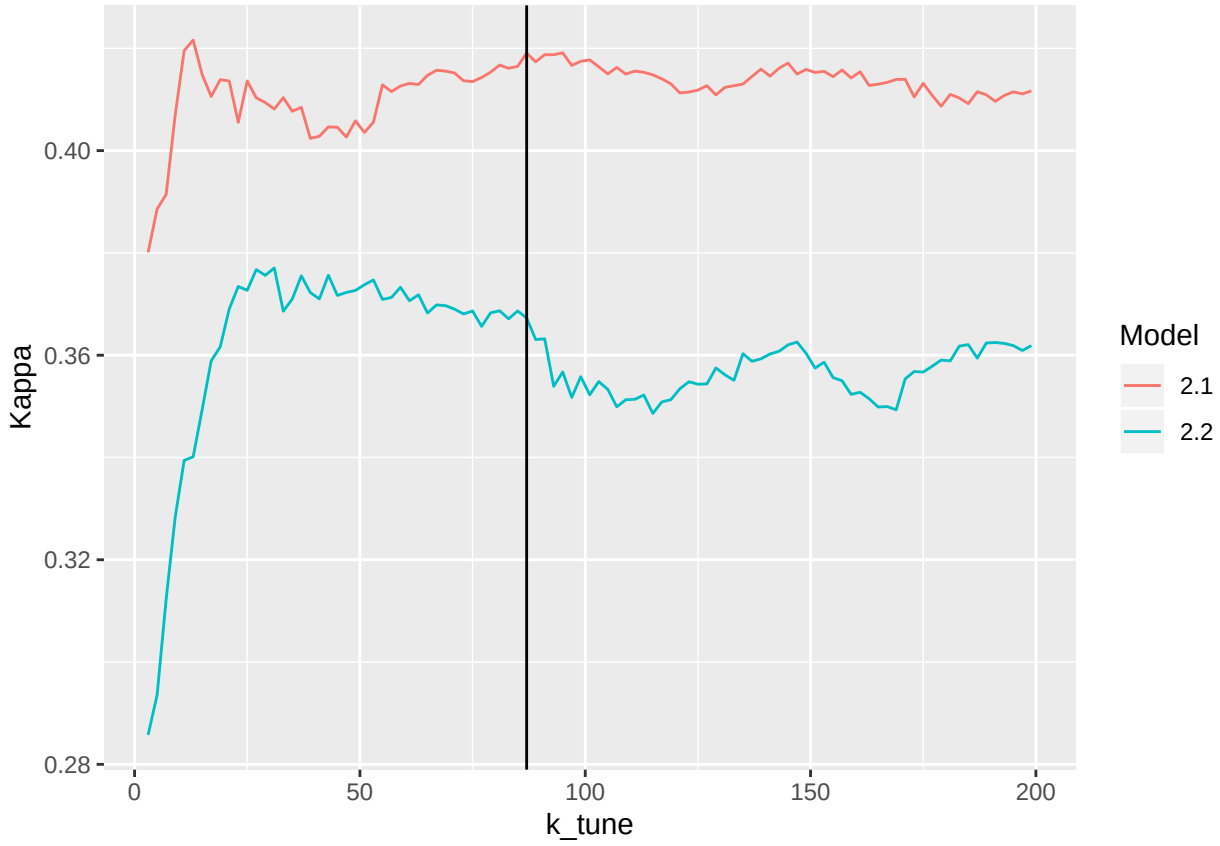
Looking at the above output, the centered and scaled data systematically performs better. I am going to plot the statistics to understand the tuning process better.

```
# Plotting Kappa
plot_df_kappa <- cbind(k_tune, tune2.1_kappa, tune2.2_kappa) %>%
  as.tibble() %>%
```

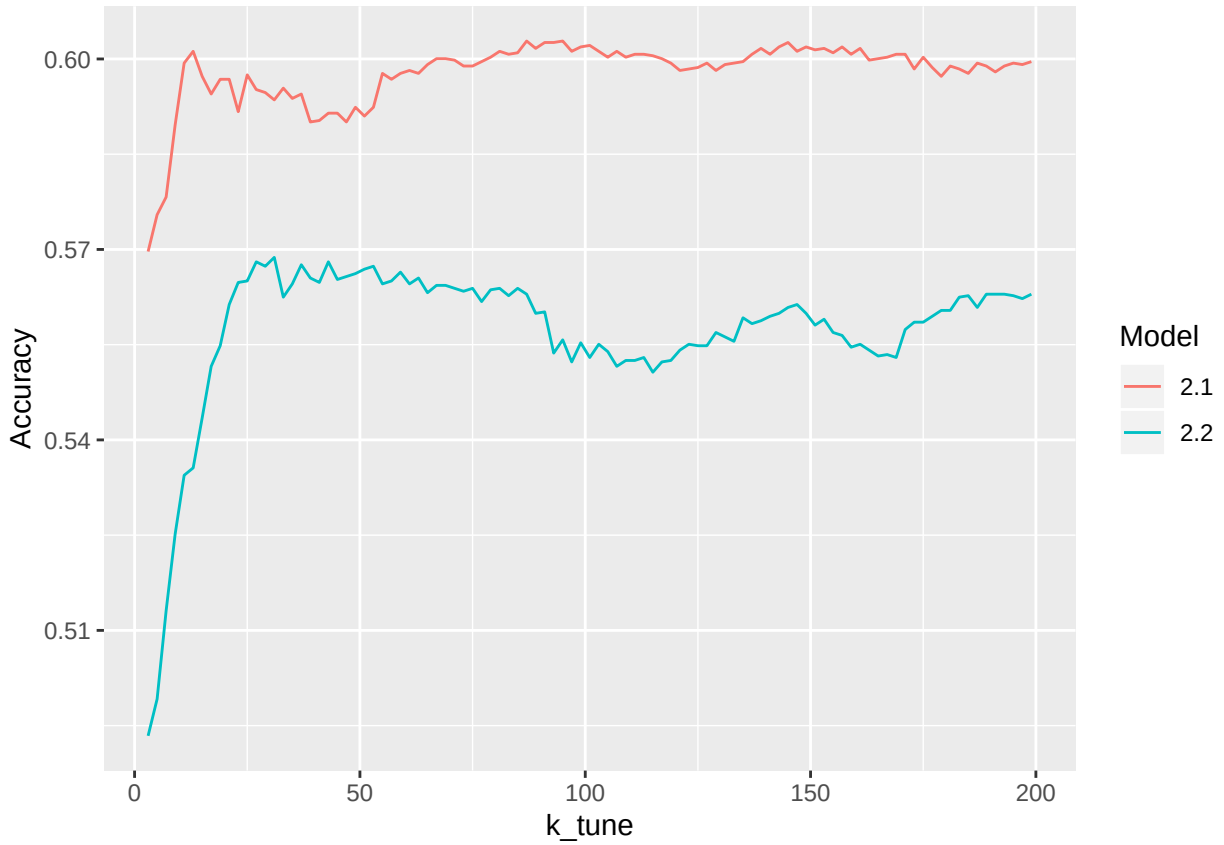
```
  gather(key = "Model", value = "Kappa", tune2.1_kappa, tune2.2_kappa) %>%
  mutate(Model = factor(ifelse(Model == "tune2.1_kappa", "2.1", "2.2")))
```

```
## Warning: `as.tibble()` is deprecated, use `as_tibble()` (but mind the new semantics).
## This warning is displayed once per session.
```

```
ggplot(plot_df_kappa, aes(x = k_tune, y = Kappa, color = Model)) +
  geom_line() +
  geom_vline(xintercept = 87)
```




```
# Plotting Accuracy
plot_df_acc <- cbind(k_tune, tune2.1_acc, tune2.2_acc) %>%
  as.tibble() %>%
  gather(key = "Model", value = "Accuracy", tune2.1_acc, tune2.2_acc) %>%
  mutate(Model = factor(ifelse(Model == "tune2.1_acc", "2.1", "2.2")))

ggplot(plot_df_acc, aes(x = k_tune, y = Accuracy, color = Model)) +
  geom_line()
```



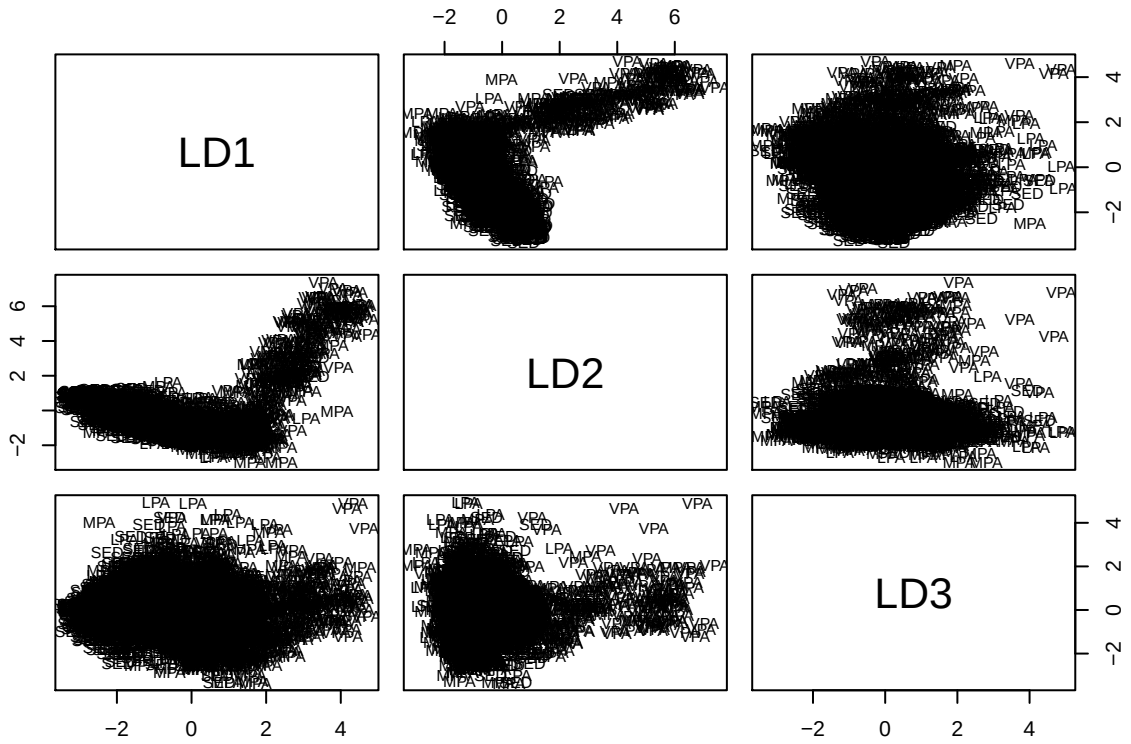

```
left_join(plot_df_kappa, plot_df_acc, by = c("k_tune", "Model"))[c(6, 43),]
```

```
## # A tibble: 2 x 4
##   k_tune Model Kappa Accuracy
##    <dbl> <fct> <dbl>    <dbl>
## 1     13 2.1   0.422    0.601
## 2     87 2.1   0.419    0.603
```

Looking at the plot, centered and scaled data systematically performs better. This model performs best when the algorithm considers 87 of its nearest neighbors (Kappa = 0.4205). However, considering only 13 nearest neighbors, one is able to achieve a similar level of accuracy (Kappa = 0.4183). Additionally, the heuristic suggests a value of 65. The performance of the model at 65 is not terrible, but definitely not the optimal. This demonstrates the importance of model tuning and the downfalls of heuristic guides.

Linear Discriminant Analysis

```
# Training the LDA model
lda2 <- lda(COS.Intensity ~., data = train2)
plot(lda2)
```




```
lda2_preds <- predict(lda2, test2)

confusionMatrix(lda2_preds$class, test2$COS.Intensity)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  SED  LPA  MPA  VPA
##        SED 1389  472  203   11
##        LPA  253  461  385   30
##        MPA   44  174  660  136
##        VPA    1    1    8   85
##
## Overall Statistics
##
##                Accuracy : 0.6017
##                  95% CI : (0.5869, 0.6163)
##     No Information Rate : 0.3911
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4097
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: SED Class: LPA Class: MPA Class: VPA
## Sensitivity              0.8234     0.4161     0.5255    0.32443
## Specificity              0.7388     0.7916     0.8842    0.99753
## Pos Pred Value           0.6694     0.4083     0.6509    0.89474
## Neg Pred Value           0.8668     0.7968     0.8193    0.95804
```

```
## Prevalence                0.3911     0.2569     0.2912    0.06075
## Detection Rate            0.3220     0.1069     0.1530    0.01971
## Detection Prevalence      0.4811     0.2618     0.2351    0.02203
## Balanced Accuracy         0.7811     0.6038     0.7048    0.66098
```

The LDA model performs similarly to the KNN model. LDA doesn't require much tuning aside from data pre-processing. So, I am taking the output essentially as-is.

Quadratic Discriminant Analysis

```
qda2 <- qda(COS.Intensity ~., data = test2)

qda2_preds <- predict(qda2, data = test2)

confusionMatrix(qda2_preds$class, test2$COS.Intensity)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  SED  LPA  MPA  VPA
##        SED 1466  518  223    4
##        LPA  173  357  151   10
##        MPA   46  231  854   96
##        VPA    2    2   28  152
##
## Overall Statistics
##
##                Accuracy : 0.6559
##                  95% CI : (0.6415, 0.6701)
##     No Information Rate : 0.3911
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4887
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: SED Class: LPA Class: MPA Class: VPA
## Sensitivity              0.8690    0.32220     0.6799    0.58015
## Specificity              0.7163    0.89579     0.8780    0.99210
## Pos Pred Value           0.6630    0.51664     0.6960    0.82609
## Neg Pred Value           0.8949    0.79266     0.8697    0.97336
## Prevalence               0.3911    0.25690     0.2912    0.06075
## Detection Rate           0.3399    0.08277     0.1980    0.03524
## Detection Prevalence     0.5126    0.16021     0.2845    0.04266
## Balanced Accuracy        0.7926    0.60899     0.7790    0.78613
```

QDA performs the best out of all available models up to this point. It is a pretty accurate model (Acc = 0.6559), and it has fair discretion over discerning cases from just chance (Kappa = 0.4887). Given the output of this model, I would put this model into production.

Multinomial Regression

I am favoring multinomial regression in this case because it allows comparison to the previous models.

```
mnr2 <- multinom(COS.Intensity ~., data = train2)
```

```
## # weights:  120 (87 variable)
## initial  value 5973.542402
## iter  10 value 4655.731152
## iter  20 value 4409.923942
## iter  30 value 4038.410721
## iter  40 value 3855.854905
## iter  50 value 3743.081237
## iter  60 value 3713.365877
## iter  70 value 3706.929307
## iter  80 value 3706.693938
## final  value 3706.692386
## converged
```

```
summary(mnr2)
```

```
## Call:
## multinom(formula = COS.Intensity ~ ., data = train2)
##
## Coefficients:
##     (Intercept) mean.GE_LW_X var.GE_LW_X max.GE_LW_X min.GE_LW_X
## LPA  -0.1600136    -5.017826   -5.854607   0.5945613   0.1167122
## MPA  -1.5149876    -7.504751   -4.390664   0.3720283   0.2109719
## VPA  -4.6198798    -8.651805   -3.876111   0.3604176  -0.2500586
##     quarlite.GE_LW_X70 quarlite.GE_LW_X80 quarlite.GE_LW_X90 mean.GE_LW_Y
## LPA           3.693086          -2.048802           2.972307    -3.648456
## MPA           5.339683          -1.837091           3.946683    -7.914414
## VPA           7.434367          -4.012695           5.960604    -7.653323
##     var.GE_LW_Y max.GE_LW_Y min.GE_LW_Y quarlite.GE_LW_Y70
## LPA   -4.758777   0.8103339  -0.9357264          0.4250478
## MPA   -9.477948   1.0761165  -1.1584847          2.7523898
## VPA   -5.253412   1.0237662  -1.2214892          4.2441078
##     quarlite.GE_LW_Y80 quarlite.GE_LW_Y90 mean.GE_LW_Z var.GE_LW_Z
## LPA          1.2168911           1.119826    -1.341651   -6.150706
## MPA          0.5147999           2.849782     1.182101   -2.428197
## VPA         -0.1067413           1.572294     5.213322    3.793721
##     max.GE_LW_Z min.GE_LW_Z quarlite.GE_LW_Z70 quarlite.GE_LW_Z80
## LPA   0.2805299  0.09951224          0.4241684           0.568117
## MPA   0.1796032  0.44629707         -1.8609987           1.687996
## VPA   0.1268990  0.57895268         -3.4917065          -1.531876
##     quarlite.GE_LW_Z90 corri.GE_LW_XY corri.GE_LW_XZ corri.GE_LW_YZ
## LPA          1.0861401     -0.4564892    -0.08455861     -0.3991164
## MPA         -0.9403728      0.2794868     0.14003414     -0.2158352
## VPA         -1.2004027     -0.7880179     1.44224790      0.7027602
##       SexMale          Age           Ht           Wt
## LPA 0.6751221 0.0004983662 -0.009176426 -0.008025185
## MPA 0.5026070 0.0002603943  0.013653262 -0.012716998
## VPA 1.2823995 0.0004908512  0.033607173 -0.023099519
##
## Std. Errors:
```

```
##     (Intercept) mean.GE_LW_X var.GE_LW_X max.GE_LW_X min.GE_LW_X
## LPA  0.65267735    0.3928872   0.5622463   0.1620911   0.1012952
## MPA  0.57048218    0.4221667   0.3779191   0.1701613   0.1110505
## VPA  0.05610244    0.3884827   0.3608166   0.2037105   0.1317833
##     quarlite.GE_LW_X70 quarlite.GE_LW_X80 quarlite.GE_LW_X90 mean.GE_LW_Y
## LPA          0.3850560          0.5223263          0.3858035    0.3594101
## MPA          0.3404316          0.4936803          0.3869617    0.3719589
## VPA          0.2977964          0.2390281          0.3914052    0.4474546
##     var.GE_LW_Y max.GE_LW_Y min.GE_LW_Y quarlite.GE_LW_Y70
## LPA   0.5860494   0.1586262   0.1431044          0.4106490
## MPA   0.5055461   0.1653885   0.1473119          0.4465736
## VPA   0.4540516   0.2048782   0.1674075          0.4863564
##     quarlite.GE_LW_Y80 quarlite.GE_LW_Y90 mean.GE_LW_Z var.GE_LW_Z
## LPA          0.5283633          0.3995392    0.3815803   0.4367762
## MPA          0.5649928          0.3906863    0.3910087   0.4968511
## VPA          0.3241556          0.4674974    0.3367545   0.4612559
##     max.GE_LW_Z min.GE_LW_Z quarlite.GE_LW_Z70 quarlite.GE_LW_Z80
## LPA   0.1625437   0.1082010          0.4840718          0.5764187
## MPA   0.1721376   0.1233512          0.4434478          0.5348722
## VPA   0.2163818   0.1597249          0.2858053          0.1723536
##     quarlite.GE_LW_Z90 corri.GE_LW_XY corri.GE_LW_XZ corri.GE_LW_YZ
## LPA          0.4146395      0.1290057      0.1288372      0.1246766
## MPA          0.4404205      0.1403825      0.1401750      0.1396092
## VPA          0.3694408      0.2635776      0.2571779      0.2590671
##       SexMale          Age         Ht          Wt
## LPA 0.1208160 0.0004118091 0.01130451 0.002035267
## MPA 0.1305866 0.0004251704 0.01052191 0.002242561
## VPA 0.2183831 0.0005770961 0.01077111 0.003929057
##
## Residual Deviance: 7413.385
## AIC: 7587.385
```

```
mnr2_preds <- predict(mnr2, type = "class", newdata = test2)

confusionMatrix(mnr2_preds, test2$COS.Intensity)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  SED  LPA  MPA  VPA
##        SED 1413  464  164    7
##        LPA  233  471  452   35
##        MPA   38  166  625  136
##        VPA    3    7   15   84
##
## Overall Statistics
##
##                Accuracy : 0.6012
##                  95% CI : (0.5864, 0.6159)
##     No Information Rate : 0.3911
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4106
##
##  Mcnemar's Test P-Value : < 2.2e-16
```

```
##
## Statistics by Class:
##
##                      Class: SED Class: LPA Class: MPA Class: VPA
## Sensitivity              0.8376     0.4251     0.4976    0.32061
## Specificity              0.7582     0.7754     0.8888    0.99383
## Pos Pred Value           0.6899     0.3955     0.6477    0.77064
## Neg Pred Value           0.8790     0.7960     0.8115    0.95766
## Prevalence               0.3911     0.2569     0.2912    0.06075
## Detection Rate           0.3276     0.1092     0.1449    0.01948
## Detection Prevalence     0.4748     0.2761     0.2237    0.02527
## Balanced Accuracy        0.7979     0.6002     0.6932    0.65722
```

From the above output, it appears multinomial regression does not offer and advantages over the other models being considered.

Conclusion

The best model for classifying physical activity into 4 categories is Quadratic Discriminant Analysis since it has the highest accuracy and largest Kappa statistic. QDA assumes different means and variances across the classes of interest. It makes intuitive sense as to why it would perform the best: the means for activity levels are clearly different, and the variance is also likely to be different. To illustrate this point, consider jogging at a moderate pace and sprinting. Jogging would require an average level of movement less than sprinting, but the average may not be substantially so. However, the variation of movement will be far less during jogging than during sprinting (think of how much more your legs move), which will clearly separate the categories. The summary values below support this claim.

```
# Means
train2 %>%
  drop_na() %>%
  group_by(COS.Intensity) %>%
  dplyr::select(COS.Intensity, var.GE_LW_X, var.GE_LW_Y, var.GE_LW_Z) %>%
  summarize_all(mean)
```

```
## # A tibble: 4 x 4
##   COS.Intensity var.GE_LW_X var.GE_LW_Y var.GE_LW_Z
##   <fct>               <dbl>       <dbl>       <dbl>
## 1 SED                0.0242      0.0298      0.0251
## 2 LPA                0.0676      0.122       0.0782
## 3 MPA                0.106       0.131       0.0754
## 4 VPA                0.779       0.445       0.302
```

```
# Standard deviations
train2 %>%
  drop_na() %>%
  group_by(COS.Intensity) %>%
  dplyr::select(COS.Intensity, var.GE_LW_X, var.GE_LW_Y, var.GE_LW_Z) %>%
  summarize_all(sd)
```

```
## # A tibble: 4 x 4
##   COS.Intensity var.GE_LW_X var.GE_LW_Y var.GE_LW_Z
##   <fct>               <dbl>       <dbl>       <dbl>
## 1 SED                0.120       0.0865      0.0615
## 2 LPA                0.0831      0.125       0.0809
## 3 MPA                0.232       0.146       0.110
```

```
## 4 VPA                0.829       0.328       0.516
```