Lab 3:

Exercise One

```
amesmapp <- mapply(is.integer,ameslist)
Ames <- subset(ameslist, select = amesmapp)
```

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

```
Ames <- subset(Ames, select = -c(OverallCond, OverallQual))
```

2. Using forward selection, create a series of models up to complexity length 15. You may use all variables, including categorical variables.

```
AmesAll <- lm(SalePrice ~ .,data = Ames) #model with all predictors
AmesStart <- lm(SalePrice ~ 1,data = Ames) #model with y variable and no predictors
step(AmesStart, direction = "forward",scope = formula(AmesAll)) #shows which
variables to include
```

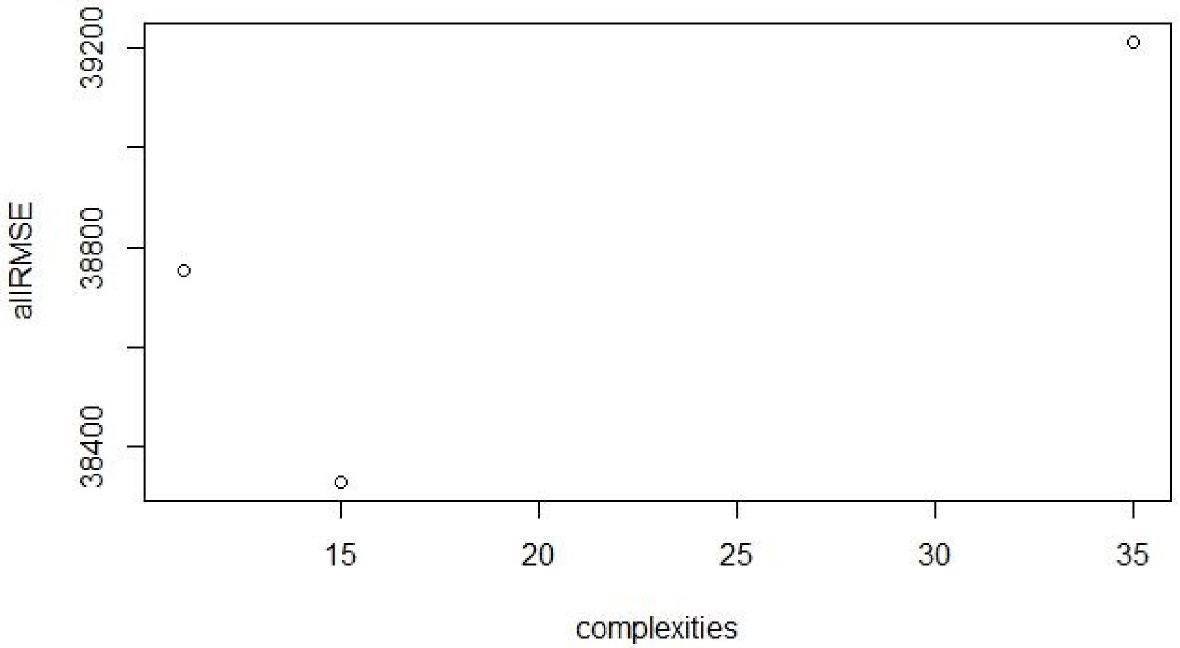
returns AIC of 8529.25

```
AmesFixed <- lm(SalePrice ~ GrLivArea + GarageArea + TotalBsmtSF +
YearRemodAdd +
                BedroomAbvGr + YearBuilt + LotArea + KitchenAbvGr + TotRmsAbvGrd +
                BsmtUnfSF + MasVnrArea, data = Ames )
step(AmesFixed, direction = "forward",scope = formula(AmesAll))
AmesFinal <- lm(formula = SalePrice ~ GrLivArea + GarageArea + TotalBsmtSF +
    YearRemodAdd + BedroomAbvGr + YearBuilt + LotArea + KitchenAbvGr +
    TotRmsAbvGrd + BsmtUnfSF + MasVnrArea + PoolArea + OpenPorchSF +
    ScreenPorch + MSSubClass, data = Ames)
AIC(AmesAll) # 9108.402
RSSA <- c(crossprod(AmesAll$residuals))
MSEA <- RSSA/length(AmesAll$residuals)
RMSEA <- sqrt(MSEA) # 28600.14
AIC(AmesStart) # 10442.25
RSSS <- c(crossprod(AmesStart$residuals))
MSES <- RSSS/length(AmesStart$residuals)
RMSES <- sqrt(MSES) # 76790.71
AIC(AmesFixed) #9669.106
RSSF <- c(crossprod(AmesFixed$residuals))
MSEF <- RSSF/length(AmesFixed$residuals)
```

```
RMSEF <- sqrt(MSEF) # 30101.62
AIC(AmesFinal) # 9655.619
RSME <- c(crossprod(AmesFinal$residuals))
MSEE <- RSME/length(AmesFinal$residuals)
RMSEE <- sqrt(MSEE) # 29324.94
```

3. **Create a chart plotting the model complexity as the x-axis variable and RMSE as the**
y-axis variable.

```
rmse = function(actual, predicted) {sqrt(mean((actual - predicted) ^ 2))}
get_complexity = function(model) {length(coef(model)) - 1}
complexAmesStart <- get_complexity (AmesStart)
complexAmesFixed <- get_complexity (AmesFixed)
complexAmesFinal<- get_complexity (AmesFinal)
complexAmesAll <- get_complexity (AmesAll)
complexities <-
c(complexAmesStart,complexAmesFixed,complexAmesFinal,complexAmesAll)
complexities <- c(complexAmesFixed,complexAmesFinal,complexAmesAll)
allRMSE <- c(RMSES,RMSEF, RMSEE,RMSEA)
allRMSE <- c(RMSEF, RMSEE,RMSEA)
plot(complexities, allRMSE)
```



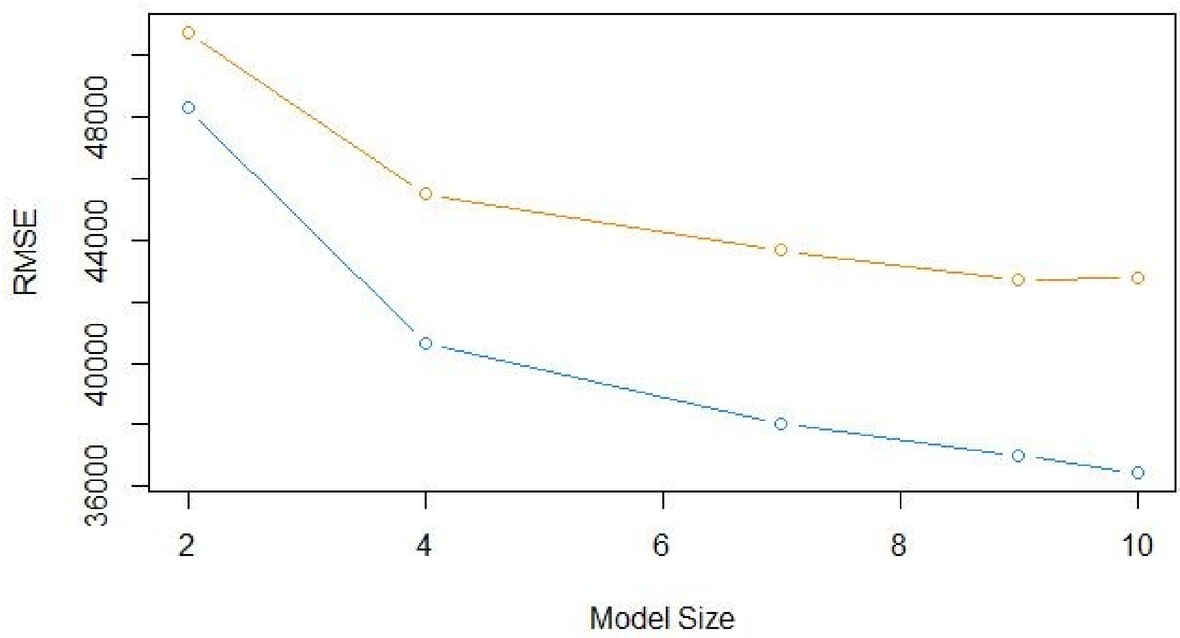
When looking at the plot, the rmse of a point with a complexity of 1 is much higher than a point with a larger complexity. For this reason, it does not make sense to use a full-sized model, since it may not be as accurate as data with limited variables. However, a higher complexity doesn't necessarily conclude that the point has a low rmse. Adding more variables can also make the model less true. This is seen in the plot, since the point with 15 variables has the lowest rmse, while those with 35 and 12 are slightly higher.

Exercise Two

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

```
set.seed(9)
num_obs = nrow(Ames)
train_index = sample(num_obs, size = trunc(0.50 * num_obs))
train_data = Ames[train_index, ]
test_data = Ames[-train_index, ]
fit_0 = lm(SalePrice ~ 1, data = train_data)
get_complexity(fit_0)
```

```
# train RMSE
sqrt(mean((train_data$SalePrice - predict(fit_0, train_data)) ^ 2))

# test RMSE
sqrt(mean((test_data$SalePrice - predict(fit_0, test_data)) ^ 2))

# train RMSE
rmse(actual = train_data$SalePrice, predicted = predict(fit_0, train_data))

# test RMSE
rmse(actual = test_data$SalePrice, predicted = predict(fit_0, test_data))
get_rmse = function(model, data, response) {
  rmse(actual = subset(data, select = response, drop = TRUE),
        predicted = predict(model, data))}
get_rmse(model = fit_0, data = train_data, response = "SalePrice") # train RMSE
get_rmse(model = fit_0, data = test_data, response = "SalePrice") # test RMSE

fit_1 <- lm(SalePrice ~ GrLivArea + GarageArea, data = train_data)
fit_2 <- lm(SalePrice ~ GrLivArea + GarageArea + TotalBsmtSF +
        YearRemodAdd, data = train_data )
fit_3 <- lm(SalePrice ~ GrLivArea + GarageArea + TotalBsmtSF + YearRemodAdd +
        BedroomAbvGr + YearBuilt + LotArea, data = train_data)
fit_4 <- lm(SalePrice ~ GrLivArea + GarageArea + TotalBsmtSF + YearRemodAdd +
        BedroomAbvGr + YearBuilt + LotArea + KitchenAbvGr + TotRmsAbvGrd,
        data = train_data)
fit_5 <- lm(SalePrice ~ GrLivArea + GarageArea + TotalBsmtSF + YearRemodAdd +
        BedroomAbvGr + YearBuilt + LotArea + KitchenAbvGr + TotRmsAbvGrd +
        BsmtUnfSF, data = train_data)

model_list = list(fit_1, fit_2, fit_3, fit_4, fit_5)
train_rmse = sapply(model_list, get_rmse, data = train_data, response = "SalePrice")
test_rmse = sapply(model_list, get_rmse, data = test_data, response = "SalePrice")
model_complexity = sapply(model_list, get_complexity)

 This is the same as the apply command above
test_rmse = c(get_rmse(fit_1, test_data, "SalePrice"),
        get_rmse(fit_2, test_data, "SalePrice"),
        get_rmse(fit_3, test_data, "SalePrice"),
        get_rmse(fit_4, test_data, "SalePrice"),
        get_rmse(fit_5, test_data, "SalePrice"))
 train data
plot(model_complexity, train_rmse, type = "b",
        ylim = c(min(c(train_rmse, test_rmse)) - 0.02,
        max(c(train_rmse, test_rmse)) + 0.02),
        col = "dodgerblue",
        xlab = "Model Size",
        ylab = "RMSE")
 test data
lines(model_complexity, test_rmse, type = "b", col = "darkorange")
```



2. Predict SalePrice. Calculate the Train and Test RMSE

 Backward Selection step by step

```
StartAmes <- lm(SalePrice ~ .,data = Ames)
step(StartAmes, direction = "backward",scope = formula(AmesAll))
Ames1 <- lm(SalePrice ~ . - OpenPorchSF, data = Ames)
Ames2 <- lm(SalePrice ~ . - OpenPorchSF - YrSold, data = Ames)
Ames3 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold, data = Ames)
```

```
Ames4 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath, data =
Ames)
Ames5 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea,
     data = Ames)
Ames6 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea -
     MiscVal, data = Ames)
Ames7 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea -
     MiscVal - HalfBath, data = Ames)
Ames8 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea -
     MiscVal - HalfBath - X3SsnPorch, data = Ames)
Ames9 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea -
     MiscVal - HalfBath - X3SsnPorch - GarageYrBlt, data = Ames)
Ames10 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea -
     MiscVal - HalfBath - X3SsnPorch - GarageYrBlt - EnclosedPorch, data = Ames)
Ames11 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea -
          MiscVal - HalfBath - X3SsnPorch - GarageYrBlt - EnclosedPorch - Id,
     data = Ames)
Ames12 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea -
          MiscVal - HalfBath - X3SsnPorch - GarageYrBlt - EnclosedPorch - Id,
     data = Ames)
Ames13 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea -
          MiscVal - HalfBath - X3SsnPorch - GarageYrBlt - EnclosedPorch - Id -
          LowQualFinSF, data = Ames)
Ames14 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea -
          MiscVal - HalfBath - X3SsnPorch - GarageYrBlt - EnclosedPorch - Id -
          LowQualFinSF - X1stFlrSF, data = Ames)
Ames15 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea -
          MiscVal - HalfBath - X3SsnPorch - GarageYrBlt - EnclosedPorch - Id -
          LowQualFinSF - X1stFlrSF - X2ndFlrSF, data = Ames)
Ames16 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea -
          MiscVal - HalfBath - X3SsnPorch - GarageYrBlt - EnclosedPorch - Id -
          LowQualFinSF - X1stFlrSF - X2ndFlrSF - LotFrontage, data = Ames)
Ames17 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea -
          MiscVal - HalfBath - X3SsnPorch - GarageYrBlt - EnclosedPorch - Id -
          LowQualFinSF - X1stFlrSF - X2ndFlrSF - LotFrontage - BsmtFinSF2, data
= Ames)
Ames18 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea -
          MiscVal - HalfBath - X3SsnPorch - GarageYrBlt - EnclosedPorch - Id -
          LowQualFinSF - X1stFlrSF - X2ndFlrSF - LotFrontage - BsmtFinSF2 -
BsmtUnfSF,
          data = Ames)
Ames19 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea -
          MiscVal - HalfBath - X3SsnPorch - GarageYrBlt - EnclosedPorch - Id -
          LowQualFinSF - X1stFlrSF - X2ndFlrSF - LotFrontage - BsmtFinSF2 -
BsmtUnfSF -
          LotArea, data = Ames)
Ames20 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea -
          MiscVal - HalfBath - X3SsnPorch - GarageYrBlt - EnclosedPorch - Id -
          LowQualFinSF - X1stFlrSF - X2ndFlrSF - LotFrontage - BsmtFinSF2 -
BsmtUnfSF -
          LotArea - BsmtFinSF1, data = Ames)
Ames21 <- lm(SalePrice ~ . - OpenPorchSF - YrSold - MoSold - BsmtHalfBath -
GarageArea -
          MiscVal - HalfBath - X3SsnPorch - GarageYrBlt - EnclosedPorch - Id -
          LowQualFinSF - X1stFlrSF - X2ndFlrSF - LotFrontage - BsmtFinSF2 -
BsmtUnfSF -
          LotArea - BsmtFinSF1 - PoolArea, data = Ames)

model_list = list(Ames1, Ames2, Ames3, Ames4, Ames5, Ames6, Ames7, Ames8,
Ames9, Ames10, Ames11,
          Ames12, Ames13, Ames14, Ames15, Ames16, Ames17, Ames18,
Ames19, Ames20, Ames21)
```

```
RSMEAmes1 <- c(crossprod(Ames1$residuals))
MSEE1 <- RSMEAmes1/length(Ames1$residuals)
RMSE1 <- sqrt(MSEE1)
RSMEAmes21 <- c(crossprod(Ames21$residuals))
MSEE21 <- RSMEAmes21/length(Ames21$residuals)
RMSE21 <- sqrt(MSEE21)
complexAmes1 <- get_complexity (Ames1)
complexAmes21 <- get_complexity (Ames21)
complexAmesFinal<- get_complexity (AmesFinal)
complexAmesAll <- get_complexity (AmesAll)
complexities <- c(complexAmes1,complexAmes21)
allrmses <- c(RMSE1, RMSE21)
plot(complexities, allrmses)
```

```
train_rmse = sapply(model_list, get_rmse, data = train_data, response = "SalePrice")
test_rmse = sapply(model_list, get_rmse, data = test_data, response = "SalePrice")
model_complexity = sapply(model_list, get_complexity)
# train data
plot(model_complexity, train_rmse, type = "b",
        ylim = c(min(c(train_rmse, test_rmse)) - 0.02,
        max(c(train_rmse, test_rmse)) + 0.02),
        col = "dodgerblue",
        xlab = "Model Size",
        ylab = "RMSE")
# test data
lines(model_complexity, test_rmse, type = "b", col = "darkorange")
```

Backward Selection with formula

```
StartAmes <- lm(SalePrice ~ .,data = Ames)
step(StartAmes, direction = "backward",scope = formula(AmesAll))
Fixed1 <- lm(SalePrice ~ Id + MSSubClass + LotFrontage + LotArea + YearBuilt +
  YearRemodAdd + MasVnrArea + BsmtFinSF1 + BsmtFinSF2 + BsmtUnfSF +
  X1stFlrSF + X2ndFlrSF + LowQualFinSF + BsmtFullBath + FullBath +
  HalfBath + BedroomAbvGr + KitchenAbvGr + TotRmsAbvGrd + Fireplaces +
  GarageCars + WoodDeckSF + EnclosedPorch + X3SsnPorch +
  ScreenPorch + PoolArea + MiscVal, data = Ames)
step(Fixed1, direction = "backward",scope = formula(AmesAll))
Fixed2 <- lm(SalePrice ~ Id + MSSubClass + LotFrontage + LotArea +
                YearBuilt + YearRemodAdd + MasVnrArea + BsmtFinSF1 + BsmtFinSF2
+
                BsmtUnfSF + X1stFlrSF + X2ndFlrSF + LowQualFinSF + BsmtFullBath +
                FullBath + BedroomAbvGr + KitchenAbvGr + TotRmsAbvGrd + Fireplaces
+
                GarageCars + WoodDeckSF + ScreenPorch + PoolArea, data = Ames)
```

3. In a PDF write-up, describe the resulting model.

Our model is a backwards selection model that does not use any interaction variables. The model relies on the factor variables both our first and last models from the previous section. The first containing only the porch size as a factor variable and the last model containing 21 factor variables. The increase in complexity has reduced our RMSE and we believe that this has made our model quite accurate in the prediction of sale price.

To be specific, the process of backward selection is that select all independent variables into the model at the very first beginning and remove the variable which has minimum F value smaller than critical value at significance level F0, then repeat the above steps until all independent variables in the model cannot be eliminated. Based on this process, we keep the explanatory variable as much as we can, so we avoided the omitted variable bias. Also, according to the RMSE formula, we know increasing the number of variables will reduce the RMSE. In sum, we think our prediction will perform well.