STAT 4360 (Introduction to Statistical Learning, Spring 2023)
Mini Project 4
Name: Sayema Rahman

---

1. (a) I fitted a linear regression model using all the predictors to create a full model of the data set. I then made a function to compute the MSE. MSE is also known as the mean squared error. This measures the mean of the squared errors. I got an error of approximately 1.796. This is a relatively good MSE to get from a model. This indicates that the model's predictions are more accurate.

```
> # make the full model of the data
> full <- lm(Quality ~ ., data = wine)
> # compute the loocv
> compute_loocv <- function(full){
+   mse <- numeric(length = nrow(wine))
+   for(i in 1:nrow(wine)) {
+     model <- lm(Quality ~ ., data = wine[-i, ])
+     mse[i] <- (wine[i, "Quality"] - predict(model, newdata = wine[i,]))^2
+   }
+   return(mean(mse))
+ }
> answer <- compute_loocv(full)
> answer
[1] 1.796343
>
```

(b) Best-subset selection regarding a linear regression helps us find the best predictors for the response variable. The regsubsets function helps us to consider some combinations of predictors. Then the model with the highest adjusted R-squared value gets chosen. Now we can compute the test MSE using LOOCV. This is the same MSE as we get in the full model. The test MSE was 1.796343.

```
> model <- regsubsets(Quality ~ ., data = wine, method = "exhaustive")
> best_subset_model <- summary(model)$which[which.max(summary(model)$adjr2),]
> best_subset_predictors <- names(best_subset_model)[-1]
> best_subset_lm <- lm(Quality ~ ., data = wine[, c(best_subset_predictors,
+                                                   "Quality")])
> test_mse <- compute_loocv(best_subset_lm)
> test_mse
[1] 1.796343
```

(c) Forward Stepwise selection process begins with the null model. Then gradually, you start adding more predictors into the model one at a time. In this case, the previous variables may not be considered important after adding certain variables. To calculate the test MSE variable, we used the compute_loocv function that was made previously with the new model. The test MSE was 1.796343.

```
>
> model <- regsubsets(Quality ~ ., data = wine, method = "forward")
> best_forward_model <- summary(model)$which[which.max(summary(model)$adjr2), ]
> best_forward_predictors <- names(best_forward_model)[-1]
> best_forward_lm <- lm(Quality ~ ., data = wine[, c(best_forward_predictors,
+                                                    "Quality")])
> test_mse <- compute_loocv(best_forward_lm)
> test_mse
[1] 1.796343
```

(d) Backward stepwise selection is when the model starts with all its respective predictors. Then as you progress through the process, you remove one predictor at a time. Do this until you are just left with the null model. We compute the test MSE the same way as we did in the forward stepwise function, using the LOOCV function. The test MSE was 1.796343.

```
> model <- regsubsets(Quality ~ ., data = wine, method = "backward")
> best_backward_model <- summary(model)$which[which.max(summary(model)$adjr2),]
> best_backward_predictors <- names(best_backward_model)[-1]
> best_backward_lm <- lm(Quality ~ ., data = wine[, c(best_backward_predictors,
+                                                     "Quality")])
> test_mse <- compute_loocv(best_backward_lm)
> test_mse
[1] 1.796343
```

(e) Ridge regression is a regularization technique used in linear regression to help with multicollinearity and overfitting. It helps to decrease the coefficient values to make it closer to 0. The model does this by leaving one out cross-validation which fits the model, n times. Then using the left-out observation we can evaluate the model's performance. The ridge regression MSE was 0.09991706.

```
> ridge <- cv.glmnet(as.matrix(wine[, -1]), wine$Quality, alpha = 0)
> ridge_mse <- min(ridge$cvm)
> ridge_mse
[1] 0.09991706
```

(f) Lasso regression uses shrinkage. You should be using simple models in this type of regression. This helps reduce the parameters and the number of variables in the solution. The lasso regression MSE was 0.00389008.

```
> lasso <- cv.glmnet(as.matrix(wine[, -1]), wine$Quality, alpha = 1)
> lasso_mse <- min(lasso$cvm)
> lasso_mse
[1] 0.00389008
```

(g) The summary table shows us that the best models are the best subset, forward selection, and backward selection models. According to the table, all the models have the same test MSE. This shows that we can't rely on the test MSE to help us make decisions on which model is the best model. Since all the models use the same data set, seeing the difference between all the different models becomes tricky.

Model <chr>	Parameter_Estimates <dbl>	Test_MSE <dbl>
All Predictors	2.34751214	1.796343
Best Subset	0.49730728	1.796343
Forward Selection	0.27841003	1.796343
Backward Selection	1.16987265	1.796343
All Predictors	-0.69229034	1.796343
Best Subset	-0.03381226	1.796343
Forward Selection	2.34751214	1.796343
Backward Selection	0.49730728	1.796343
All Predictors	0.27841003	1.796343
Best Subset	1.16987265	1.796343
Forward Selection	-0.69229034	1.796343
Backward Selection	-0.03381226	1.796343
All Predictors	2.34751214	1.796343
Best Subset	0.49730728	1.796343
Forward Selection	0.27841003	1.796343
Backward Selection	1.16987265	1.796343
All Predictors	-0.69229034	1.796343
Best Subset	-0.03381226	1.796343
Forward Selection	2.34751214	1.796343
Backward Selection	0.49730728	1.796343
All Predictors	0.27841003	1.796343
Best Subset	1.16987265	1.796343
Forward Selection	-0.69229034	1.796343
Backward Selection	-0.03381226	1.796343

2) a) $\min \left(\frac{1}{2} \sum_{i=1}^{n} (Y_i - X_i^T \beta)^2 + \lambda \|\beta\|^2 \right)$

$$\frac{d}{d\beta} \left(\frac{1}{2} \sum_{i=1}^{n} (Y_i - X_i^T \beta)^2 + \lambda \|\beta\|^2 \right)$$

$$= \left(-\sum_{i=1}^{n} X_i (Y_i - X_i^T \beta)^2 + \lambda \beta \right)$$

$$\hat{\beta} = (X X^T + \lambda I)^{-1} X^T Y$$

b) $E(\varepsilon_i) = 0$

$E(\varepsilon_j^2) = \sigma^2$

The covariant between Y_i and Y_j for $\frac{I}{j}$ comes from their respective error terms so the covariance of both the error terms would equal to 0.

c) $\hat{Y} = X\hat{\beta}$

$= X (X^T X + 2 \lambda I)^{-1} X^T Y$

The fitted value for each observation is obtained by plugging the estimates into the linear model

d) $\text{cov}(\hat{Y}_i, Y_i) = E([\hat{Y}_i - E[\hat{Y}_i]])(Y_i - E[Y_i])$

The sum of all covariances in the relationship captures the model's alignment with the given data. It also gives numerical answers for X and λ.

Python Code (or R Code)

```r
library(ggplot2)
library(leaps)
library(glmnet)
library(MASS)
wine <- read.table("~/wine.txt", header = T, sep = '')
View(wine)
```

1(a) Fit a linear regression model using all predictors and compute its
test MSE

```r
# make the full model of the data
full <- lm(Quality ~ ., data = wine)
# compute the loocv
compute_loocv <- function(full){
  mse <- numeric(length = nrow(wine))
  for(i in 1:nrow(wine)) {
    model <- lm(Quality ~ ., data = wine[-i, ])
    mse[i] <- (wine[i, "Quality"] - predict(model, newdata =
wine[i,]))^2
  }
  return(mean(mse))
}
full_mse <- compute_loocv(full)
full_mse
```

1(b) Use best-subset selection based on adjusted R2 to find the best linear
regression model. Compute the test MSE of the best model.

```r
model <- regsubsets(Quality ~ ., data = wine, method = "exhaustive")
best_subset_model <-
summary(model)$which[which.max(summary(model)$adjr2),]
best_subset_predictors <- names(best_subset_model)[-1]
best_subset_lm <- lm(Quality ~ ., data = wine[,
c(best_subset_predictors,
                                                "Quality")])
best_subset_mse <- compute_loocv(best_subset_lm)
best_subset_mse
```

1(c) Use forward stepwise selection based on adjusted R2 to find the best linear
regression model.Compute the test MSE of the best model.

```r
model <- regsubsets(Quality ~ ., data = wine, method = "forward")
```

```
best_forward_model <-
summary(model)$which[which.max(summary(model)$adjr2), ]
best_forward_predictors <- names(best_forward_model)[-1]
best_forward_lm <- lm(Quality ~ ., data = wine[,
c(best_forward_predictors,
                                                    "Quality")])
best_forward_mse <- compute_loocv(best_forward_lm)
best_forward_mse
```
1(d) Use backward stepwise selection based on adjusted R2 to find
the best
linear regression model. Compute the test MSE of the best model.
```{r}
model <- regsubsets(Quality ~ ., data = wine, method = "backward")
best_backward_model <-
summary(model)$which[which.max(summary(model)$adjr2),]
best_backward_predictors <- names(best_backward_model)[-1]
best_backward_lm <- lm(Quality ~ ., data = wine[,
c(best_backward_predictors,
                                                    "Quality")])
best_backward_mse <- compute_loocv(best_backward_lm)
best_backward_mse
```
1(e) Use ridge regression with penalty parameter chosen optimally
via LOOCV to
fit a linear regression model. Compute the test MSE of the model.
```{r}
ridge <- cv.glmnet(as.matrix(wine[, -1]), wine$Quality, alpha = 0)
ridge_mse <- min(ridge$cvm)
ridge_mse
```
1(f) Use lasso with penalty parameter chosen optimally via LOOCV to
fit a
linear regression model. Compute the test MSE of the model
fit a linear regression model. Compute the test MSE of the model.
```{r}
lasso <- cv.glmnet(as.matrix(wine[, -1]), wine$Quality, alpha = 1)
lasso_mse <- min(lasso$cvm)
lasso_mse
```
1(g) Make a tabular summary of the parameter estimates and test MSEs
from
(a) - (c). Compare the results. Which model(s) would you recommend?
```{r}
parameter_estimates <- c(coefficients(full)[-1],
                         coefficients(best_subset_lm)[-1],
                         coefficients(best_forward_lm)[-1],
                         coefficients(best_backward_lm)[-1])
test_mses <- c(full_mse, best_subset_mse, best_forward_mse,
best_backward_mse)

summary <- data.frame(Model = c("All Predictors", "Best Subset",
```

```
                                    "Forward Selection",
                                    "Backward Selection"),
                          Parameter_Estimates =
parameter_estimates,
                          Test_MSE = test_mses)

print(summary)
```