STAT 4360 (Introduction to Statistical Learning, Spring 2023)
Mini Project 5
Name: Sayema Rahman

1. (a) Standardizing the variable before performing PCA is a good idea. It makes sure that the analysis will be interpretable, and robust. Since PCA is a method that uis generally used to display data. it converts correlated data to uncorrelated data. It is an unsupervised method to display the relationship among variables. It is used to keep important pieces of data while reducing the dimensionality of said data.

   (b) The first principal component, PC1, explain 45.31% of the variance. The first two principle components together show about 71% of the variance, The first fice components explains about 91.6% of the variance. This is a great portion of the variance showing that it captured a majority of the data set.

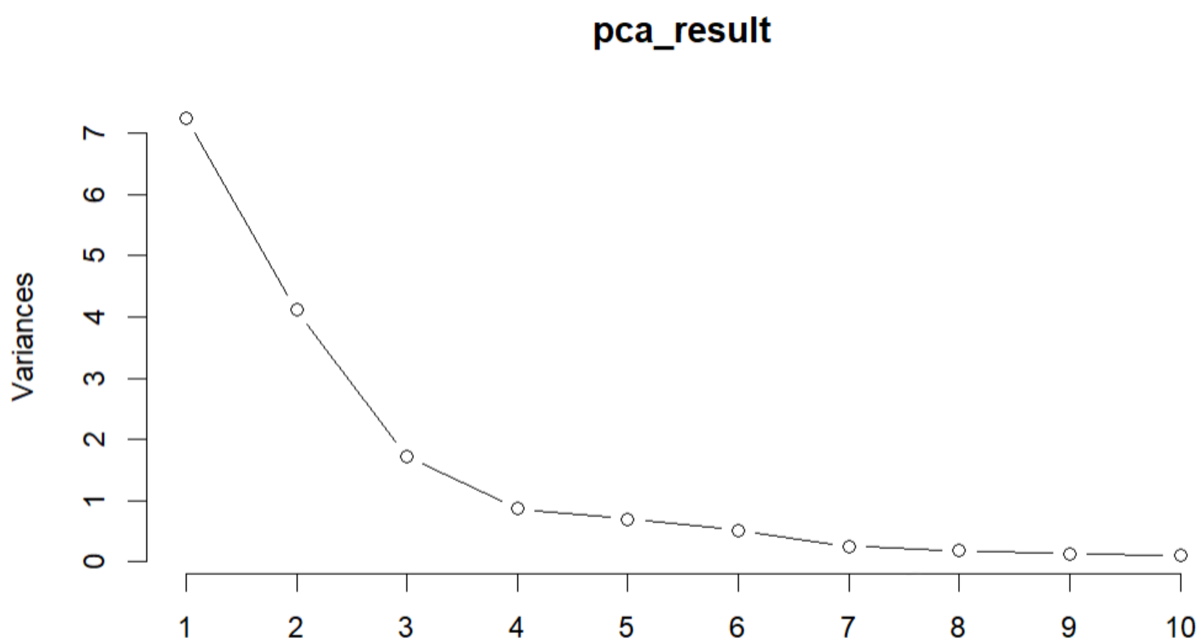
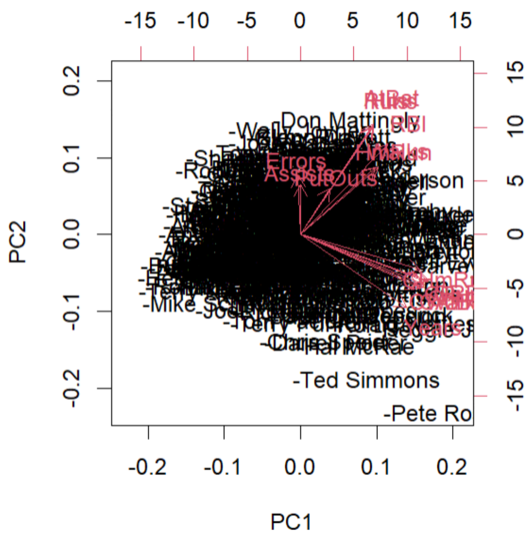


Figure 1: PCA graph

```
Importance of components:
                          PC1    PC2   PC3    PC4     PC5     PC6     PC7     PC8     PC9
Standard deviation     2.6926 2.0273 1.315 0.9329 0.83509 0.71580 0.50168 0.43005 0.36390
Proportion of Variance 0.4531 0.2569 0.108 0.0544 0.04359 0.03202 0.01573 0.01156 0.00828
Cumulative Proportion  0.4531 0.7100 0.818 0.8724 0.91597 0.94799 0.96372 0.97528 0.98356
                          PC10    PC11    PC12    PC13   PC14    PC15    PC16
Standard deviation     0.31300 0.24770 0.23460 0.16788 0.1202 0.06996 0.03467
Proportion of Variance 0.00612 0.00383 0.00344 0.00176 0.0009 0.00031 0.00008
Cumulative Proportion  0.98968 0.99351 0.99695 0.99872 0.9996 0.99992 1.00000
```

Figure 2: Summary of results from PCA

(c) The matrix represents the correlation between the variable and principle component. The higher the value is, the strong the correlation is between the variable and the principle component. A positive value will indicate a positive correlation between the two and a negative value will indicate a negative correlation between the two. In PC1 the player's entire carrer is greatly influenced. In PC2, a player's season performance is greatly influenced.



	PC1	PC2
AtBat	0.196209479	0.3911182
Hits	0.193644439	0.3839217
HmRun	0.201534072	0.2359434
Runs	0.195225544	0.3811792
RBI	0.232768724	0.3181685
Walks	0.208482683	0.2373117
Years	0.285182942	-0.2597158
CAtBat	0.333319239	-0.1867080
CHits	0.333576637	-0.1764398
CHmRun	0.320087213	-0.1245505
CRuns	0.340501234	-0.1672463
CRBI	0.342675322	-0.1630780
CWalks	0.319261481	-0.1875815
PutOuts	0.078226958	0.1634037
Assists	-0.000530817	0.1769527
Errors	-0.007283347	0.2106993

Figure 2: biplot of PCA
Figure 3: Correlation matrix

2. (a) I fitted a linear regression model. I used log(Salary) as the response variable and left all the other variables as predictors. The MSE is 0.1166741.

Question 2
Part A

```{r}
library(pls)
# creating new variable
hitters_cleaned$LogSalary <- log(hitters_cleaned$Salary)
library(boot)
linearreg.fit <- glm(LogSalary ~., data = hitters_cleaned)
cv.linearreg <- cv.glm(hitters_cleaned, linearreg.fit, K = nrow(hitters_cleaned))
cv.linearreg$delta[1]
```

```
[1] 0.1166741
```

Figure 4: MSE result and code

(b) The model is fitted with the PCR model and the data is scaled as needed. The test MSE is 0.8908877. The RMSEP is approximately 0.891 for one component.

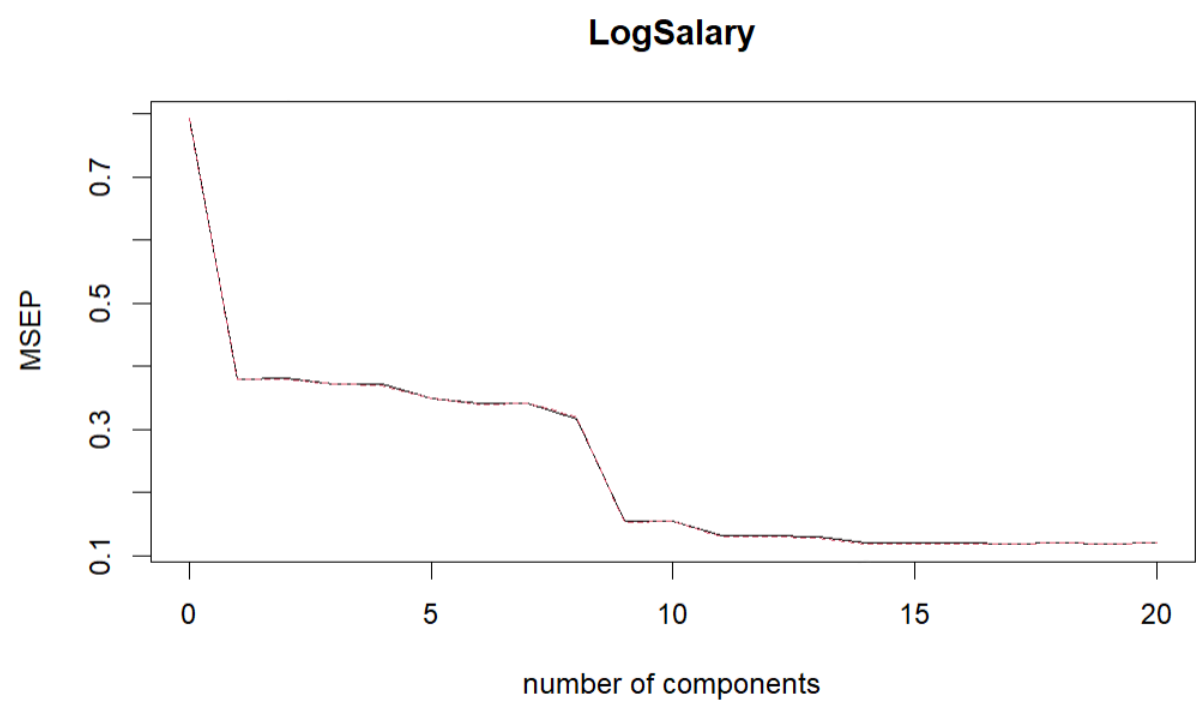


Figure 5: plot for LogSalary

```
> summary(pcr.fit, ncomp = m_pcr)
Data: 	X dimension: 263 20 
	Y dimension: 263 1
Fit method: svdpc
Number of components considered: 20

VALIDATION: RMSEP
Cross-validated using 10 random segments.
       (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
CV          0.8909   0.6200   0.6218   0.6197    0.616   0.5993   0.5927   0.5865   0.5589
adjCV       0.8909   0.6194   0.6210   0.6187    0.615   0.5982   0.5911   0.5854   0.5653
       9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps  16 comps  17 comps
CV      0.3947    0.3952    0.3553    0.3561    0.3552    0.3434    0.3442    0.3450    0.3460
adjCV   0.3931    0.3940    0.3541    0.3551    0.3540    0.3421    0.3427    0.3436    0.3444
       18 comps  19 comps  20 comps
CV       0.3482    0.3466    0.3475
adjCV    0.3465    0.3446    0.3454

TRAINING: % variance explained
           1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps  9 comps
X            38.61    59.40    69.59    77.45    82.62    86.89    90.35    92.94    95.27
LogSalary    52.99    53.22    54.44    55.07    58.58    60.15    61.47    66.24    82.24
           10 comps  11 comps  12 comps  13 comps  14 comps  15 comps  16 comps  17 comps
X             96.52     97.42     98.11     98.74     99.22     99.51     99.77     99.90
LogSalary     82.25     85.66     85.67     86.02     86.98     87.09     87.20     87.29
           18 comps  19 comps  20 comps
X             99.97     99.99    100.00
LogSalary     87.31     87.88     87.92
> sqrt(MSEP(pcr.fit)$val[1, m_pcr,1])
[1] 0.8908877
```

Figure 6: Summary of pcr.fit

(c) This is the code for the LOOCV with the PLS mode. the test MSE is 0.8875003. The model is fitted with the PLDS model and the data is scaled as needed. The graph changes in comparison to the PCR model.

```
> # make validation plot
> validationplot(pls.fit, val.type = "MSEP")
> m_pls <- which.min(MSEP(pls.fit)$val[1,,1])
> print(m_pls)
[1] 1
>
> # computing the test MSE
> summary(pls.fit, ncomp = m_pls)
Data:    X dimension: 263 20
        Y dimension: 263 1
Fit method: kernelpls
Number of components considered: 20
TRAINING: % variance explained
           1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps  9 comps
X            38.41    44.98    59.30    70.87    76.76    81.52    85.71    89.36    91.35
LogSalary    58.18    78.05    82.76    84.98    86.04    86.70    87.14    87.24    87.35
           10 comps  11 comps  12 comps  13 comps  14 comps  15 comps  16 comps  17 comps
X             93.75     96.22     96.89     97.55     98.31     98.79     98.98     99.55
LogSalary     87.42     87.46     87.61     87.72     87.80     87.86     87.90     87.90
           18 comps  19 comps  20 comps
X             99.84     99.99    100.00
LogSalary     87.91     87.91     87.92
> sqrt(MSEP(pls.fit)$val[1, m_pls,1])
[1] 0.8875003
```

Figure 7: summary of the fitted model

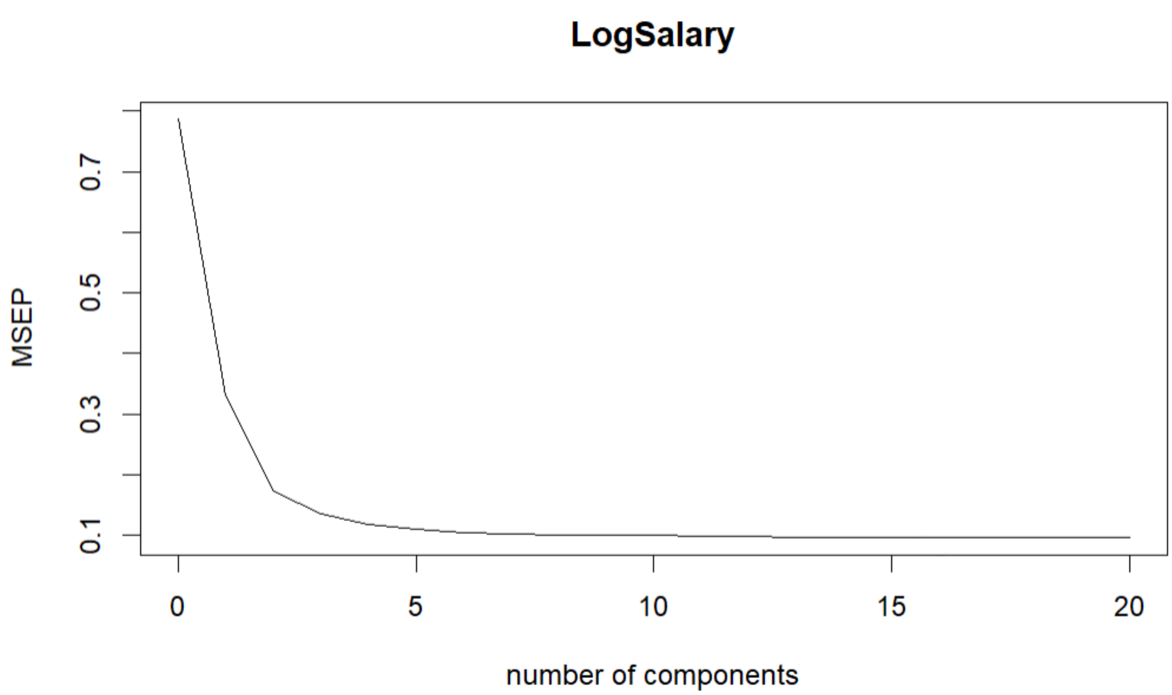
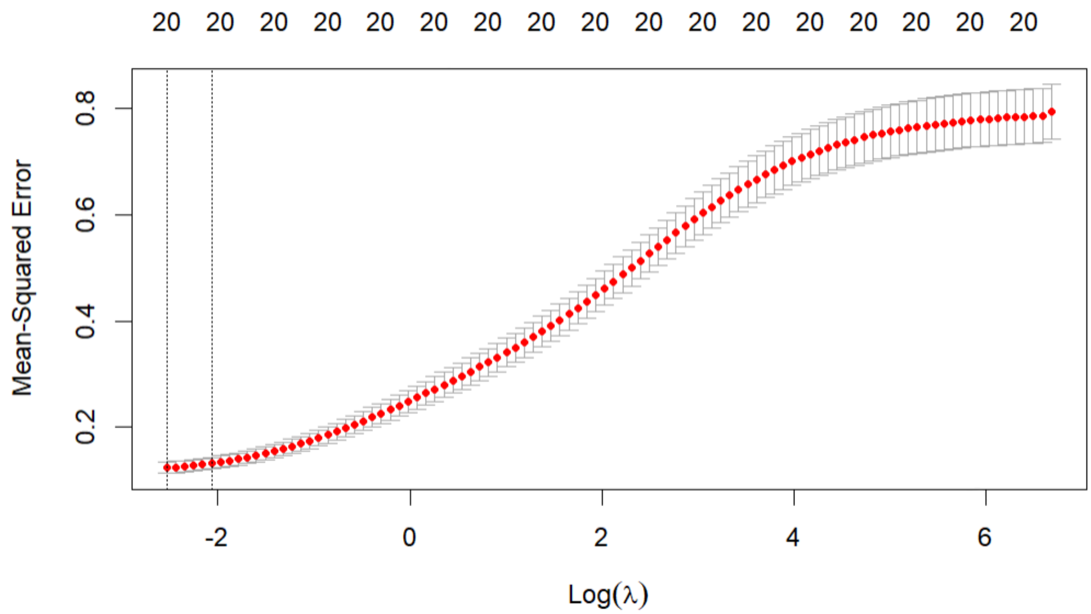


Figure 8: Graph of the LogSalary

(d) Ridge regression is generally used to analyze multiple regression datas. The data is affected by multicolinearity. The ridge regression introduces a penalty term to the model. This is useful when you may expect multicollinearity of overfitting in your model. The test MSE of the model is 0.1104553.




```
> sqrt(MSEP(pls.fit)$val[1, m_pls,1])
[1] 0.8875003
> library(glmnet)
> # matrix
> x <- model.matrix(LogSalary ~., data = hitters_cleaned)[,-1]
> y <- hitters_cleaned$LogSalary
>
> # ridge regression using lambda chosen by LOOCV and glmnet
> ridge.fit <- cv.glmnet(x, y, alpha = 0, nfolds = nrow(hitters_cleaned))
Warning: Option grouped=FALSE enforced in cv.glmnet, since < 3 observations per fold
> # make plot to find best lambda
> plot(ridge.fit)
> lamda <- ridge.fit$lambda.min
>
> # doing test MSE for optimal lambda
> ridge.predict <- predict(ridge.fit, s = lamda, newx = x)
> mean((ridge.predict - y)^2)
[1] 0.1104553
```

(e) The best is chosen based on the smallest MSE. The ridge regression with an MSE of 0.1104553 and the linear model's MSE is 0.1166741.

3) a) The mode important predictors are Years, Salary, and Log Salary.

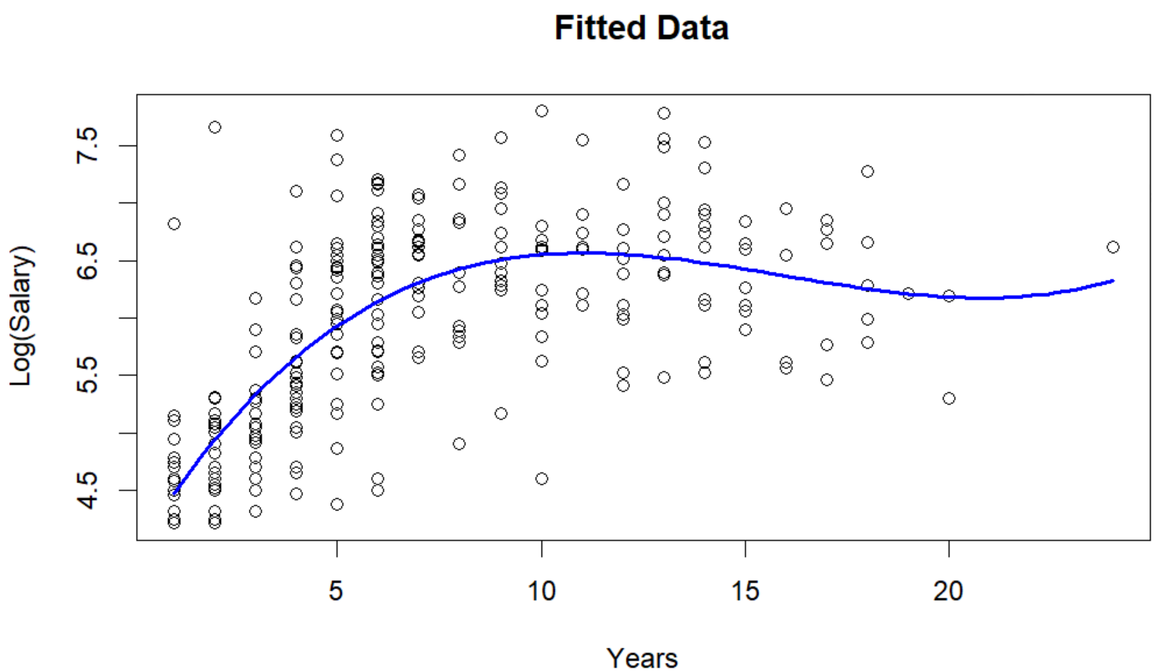
```
Call:
lm(formula = LogSalary ~ ., data = hitters_cleaned)

Residuals:
    Min      1Q  Median      3Q     Max 
-1.0905 -0.1880  0.0549  0.2230  0.8036 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  4.348e+00  9.310e-02  46.700  < 2e-16 ***
AtBat        2.963e-04  6.588e-04   0.450  0.65324    
Hits         6.576e-04  2.472e-03   0.266  0.79042    
HmRun        4.618e-03  6.325e-03   0.730  0.46599    
Runs         2.518e-03  3.041e-03   0.828  0.40857    
RBI          5.579e-05  2.651e-03   0.021  0.98323    
Walks        6.313e-04  1.907e-03   0.331  0.74088    
Years        6.274e-02  1.265e-02   4.961 1.32e-06 ***
CAtBat       4.122e-04  1.382e-04   2.981  0.00316 ** 
CHits       -6.634e-04  6.873e-04  -0.965  0.33544    
CHmRun       2.083e-04  1.648e-03   0.126  0.89951    
CRuns       -8.965e-04  7.705e-04  -1.164  0.24576    
CRBI        -1.207e-03  7.077e-04  -1.706  0.08936 .  
CWalks      -1.213e-04  3.385e-04  -0.358  0.72042    
LeagueN      1.788e-01  8.086e-02   2.211  0.02799 *  
DivisionW    2.795e-02  4.183e-02   0.668  0.50472    
PutOuts     -1.281e-04  8.103e-05  -1.581  0.11512    
Assists      6.620e-06  2.267e-04   0.029  0.97673    
Errors      -6.399e-03  4.480e-03  -1.428  0.15448    
Salary       1.657e-03  6.536e-05  25.348  < 2e-16 ***
NewLeagueN  -1.331e-01  8.051e-02  -1.654  0.09949 .  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3215 on 242 degrees of freedom
Multiple R-squared:  0.8792,	Adjusted R-squared:  0.8692 
F-statistic: 88.09 on 20 and 242 DF,  p-value: < 2.2e-16
```

(b) The greatest MSE is greater than 0.425 for 10 knots. The lowest MSE is less than 0.410. This shows that since both the MSE values were close to 0 that the mode’s predictions are close to accurate. The overall performance of the model and the data is average.


Fitted Data
Log(Salary)
4.5
5.5
6.5
7.5
5
10
15
20
Years

Python Code (or R Code)

````
Part A answered in report above.
Part B
```{r}
library(ISLR2)
data("Hitters")
View(Hitters)
```
# clean the data set
hitters_cleaned <- na.omit(Hitters)

# extract the predictor variables
# keping response variable Salary to keep dataframe intact
predictors <- hitters_cleaned[,-which(names(hitters_cleaned) == "Salary"),
                              drop = FALSE]
columns <- sapply(predictors, is.numeric)
numeric_predictors <- predictors[, columns]
standardized_predictors <- scale(numeric_predictors)
pca_result <- prcomp(standardized_predictors)
summary(pca_result)

# plot the scree plot
screeplot(pca_result, type = "lines")

Part C
correlations <- pca_result$rotation[,1:2]
print(correlations)
biplot(pca_result)

Question 2
Part A
```{r}
library(pls)
# creating new variable
hitters_cleaned$LogSalary <- log(hitters_cleaned$Salary)
library(boot)
linearreg.fit <- glm(LogSalary ~., data = hitters_cleaned)
cv.linearreg <- cv.glm(hitters_cleaned, linearreg.fit, K = nrow(hitters_cleaned))
cv.linearreg$delta[1]
```
Part B
```{r}
# fitting the pcr with loocv
pcr.fit <- pcr(LogSalary ~., data = hitters_cleaned, scale = TRUE,
````
````

                validation = "CV", segments = 10)
validationplot(pcr.fit, val.type = "MSEP")
m_pcr <- which.min(MSEP(pcr.fit)$val[1, , 1])
print(m_pcr)

# computing the test MSE
summary(pcr.fit, ncomp = m_pcr)
sqrt(MSEP(pcr.fit)$val[1, m_pcr,1])
```
Part C
```{r}
pls.fit <- plsr(LogSalary ~ ., data = hitters_cleaned, scale = TRUE,
                validate = "CV", segments = 10)
# make validation plot
validationplot(pls.fit, val.type = "MSEP")
m_pls <- which.min(MSEP(pls.fit)$val[1,,1])
print(m_pls)

# computing the test MSE
summary(pls.fit, ncomp = m_pls)
sqrt(MSEP(pls.fit)$val[1, m_pls,1])
```

Part D
```{r}
library(glmnet)
# matrix
x <- model.matrix(LogSalary ~., data = hitters_cleaned)[,-1]
y <- hitters_cleaned$LogSalary

# ridge regression using lambda chosen by LOOCV and glmnet
ridge.fit <- cv.glmnet(x, y, alpha = 0, nfolds = nrow(hitters_cleaned))
# make plot to find best lambda
plot(ridge.fit)
lamda <- ridge.fit$lambda.min

# doing test MSE for optimal lambda
ridge.predict <- predict(ridge.fit, s = lamda, newx = x)
mean((ridge.predict - y)^2)
```
Question 3
Part A
```{r}
hitters_cleaned$LogSalary <- log(hitters_cleaned$Salary)
model = lm(LogSalary ~., data = hitters_cleaned)
summary(model)

```r
hitters_cleaned$LogSalary <- log(hitters_cleaned$Salary)

# fitting a polynomila regression mode with degree 3
fitted_model <- lm(LogSalary ~ poly(Years, 3), data = hitters_cleaned)
summary(fitted_model)
# plot raw data
plot(LogSalary ~ Years, data = hitters_cleaned, main = "Fitted Data", xlab = "Years",
    ylab = "Log(Salary)")
# generate predictions
years <- seq(min(hitters_cleaned$Years), max(hitters_cleaned$Years),
             length.out = 100)
predictions <- predict(fitted_model, newdata = data.frame(Years = years))

# plot the curve
lines(years, predictions, col = "blue", lwd = 2)
```