Assignment 5

79546 - Stephen K. Ng'etich

Contents

1 Pre-requisite

1.1 Load Packages

```
# Clear variables
rm(list=ls())

library(readxl)
library(glmnet)
library(dplyr)
library(ggplot2)
library(caret)
library(tidyverse)
library(pls)
```

1.2 Load Dataset

```
set.seed(475)

dataset <- read_excel("dataset/Dataset7.xlsx")
```

2 Questions

2.1 Split the data set into a training set and a test set.

The dataset is split into training and test set in the ratio of 7:3

```
index <- sample(x=nrow(dataset), size=.70*nrow(dataset))
train <- dataset[index,]
test <-  dataset[-index,]
```

2.2 Fit a linear model using least squares on the training set, and report the test error obtained.

2.2.1 Fit the model

```
# fit the regression model
lm_model = lm(Profit ~ ., data = train)

# get model summary
lm_model_summary = summary(lm_model)

print(lm_model_summary)
```

```
##
## Call:
## lm(formula = Profit ~ ., data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.617  -3.288  -0.218   2.960  88.830
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2764.9589   110.7827  24.958  < 2e-16 ***
## Expenses      -4.8434     0.7798  -6.211 8.14e-10 ***
## Adverts        6.1042     0.5563  10.974  < 2e-16 ***
## System         0.7306     0.4493   1.626    0.104
## Furniture     13.5361     0.2417  56.001  < 2e-16 ***
## Remittance     0.8179     0.6099   1.341    0.180
## Debts          0.2867     0.1959   1.464    0.144
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 5.281 on 867 degrees of freedom
## Multiple R-squared:  0.9998, Adjusted R-squared:  0.9998
## F-statistic: 9.432e+05 on 6 and 867 DF,  p-value: < 2.2e-16
```

From the fitted regression model `Expenses`,`Adverts`,`system` and `Furniture` are significant predictors of Profit at 95% confidence interval.The estimated model has an adjusted error of 99.9%.

The linear regression can be summarized as:

$$Profit = 2729.4862 - 5.3853\,\text{Expenses} + 6.2612\,\text{Adverts} + 0.9028\,\text{System} + 13.6387\,\text{Furniture}$$

2.2.2 Calculate the Mean Squared Error

```r
lm_model_pred <- predict(lm_model, test)

#Model performance metrics
ml_performance.lse=data.frame(
MODEL = "Least Squares",
R2 = caret::R2(lm_model_pred, test$Profit),
RMSE = RMSE(lm_model_pred, test$Profit),
MAE = MAE(lm_model_pred, test$Profit))

ml_performance.lse
```

```
##           MODEL       R2     RMSE      MAE
## 1 Least Squares 0.999888 4.666283 3.873894
```

2.3 Fit a ridge regression model on the training set, with λ chosen by cross-validation.Report the test error obtained.

```r
#All values of x without the profit
x_train = data.matrix(train[-1])

#Values of Y only
y_train = train$Profit

#Find the optimal lambda value via cross validation
cv.out=cv.glmnet(x_train,y_train,alpha=0)
bestlam=cv.out$lambda.min

cat("Optimal lambda value for cross validation",bestlam, "  \n")
```

```
## Optimal lambda value for cross validation 42.49277
```

```r
#Define lambda grid to be used through out analysis
grid=10^seq(10,-2,length=100)

#Fit a ridge regression model
```

```r
ridge.mod=glmnet(x_train,y_train,alpha = 0, lambda=grid)

x_test = data.matrix(test[-1])
y_test = test$Profit

#Compute the test error w/ lambda chosen by cross validation
ridge.pred=predict(ridge.mod,s=bestlam,newx=x_test)

#Store ridge coefficients
ridge.coef=predict(ridge.mod,type="coefficients",s=bestlam)


#Model performance metrics
ml_performance.ridge = data.frame(
MODEL = "Ridge regression",
"R2" = caret::R2(ridge.pred, y_test),
RMSE = RMSE(ridge.pred, y_test),
MAE = MAE(ridge.pred, y_test))

print(ml_performance.ridge)
```

```
##              MODEL        s1     RMSE     MAE
## 1 Ridge regression 0.9980068 20.95244 12.0597
```

2.4 Fit a lasso model on the training set, with λ chosen by crossvalidation. Report the test error obtained, along with the number of non-zero coefficient estimates.

```r
#Find the optimal lambda value via cross validation
cv.out=cv.glmnet(x_train,y_train,alpha=1)
bestlam=cv.out$lambda.min
cat("Optimal lambda value for cross validation",bestlam, "  \n")
```

```
## Optimal lambda value for cross validation 11.28645
```

```r
#Train the model
lasso.mod=glmnet(x_train,y_train,alpha = 1, lambda=grid)

#Compute the test error
lasso.pred=predict(lasso.mod,s=bestlam,newx=x_test)

#Store lasso coefficients
lasso.coef=predict(lasso.mod,type="coefficients",s=bestlam)
lasso.coef
```

```
## 7 x 1 sparse Matrix of class "dgCMatrix"
##                      s1
## (Intercept) 2888.91323
## Expenses        .
## Adverts        7.31185
```

```
## System          .
## Furniture     10.96995
## Remittance      .
## Debts           .
```

```
#Model performance metrics
ml_performance.lasso = data.frame(
MODEL = "Lasso regression",
R2 = caret::R2(lasso.pred, y_test),
RMSE = RMSE(lasso.pred,y_test),
MAE = MAE(lasso.pred, y_test))

ml_performance.lasso
```

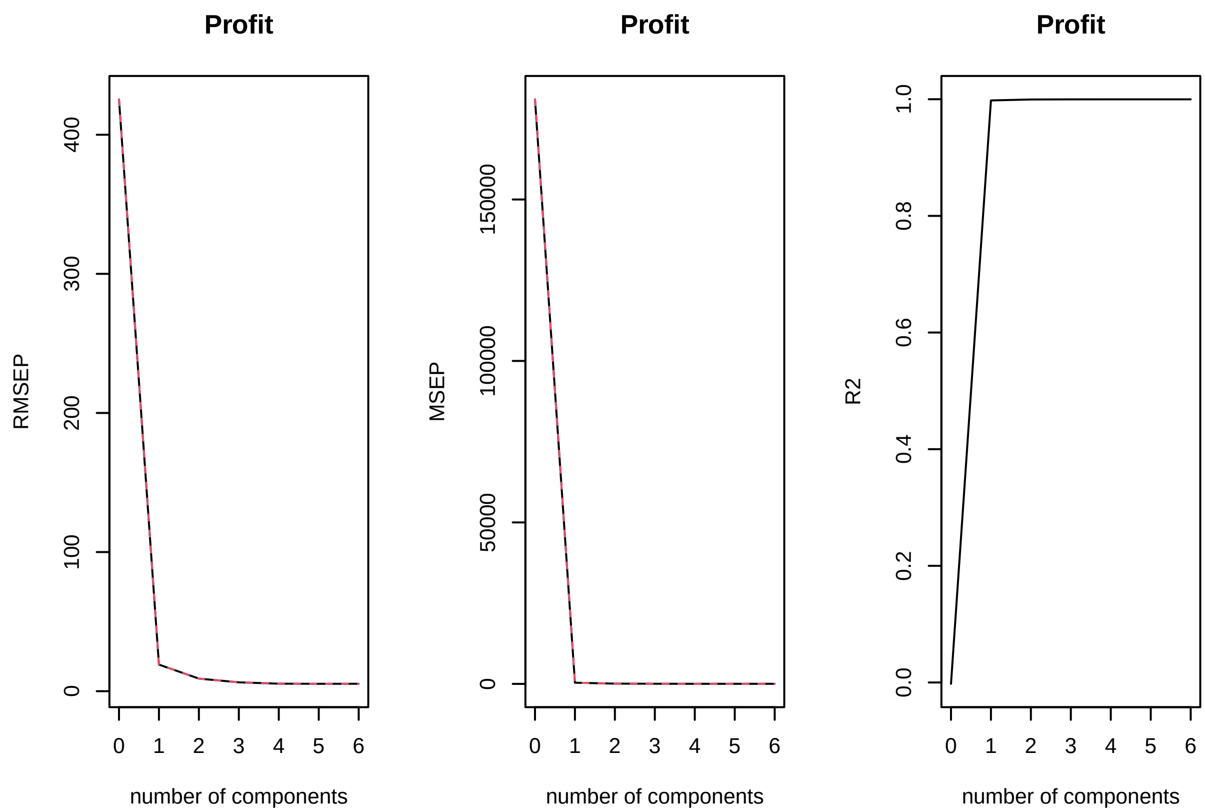
```
##              MODEL       s1    RMSE      MAE
## 1 Lasso regression 0.999778 13.9886 11.35732
```

2.5 Fit a PCR model on the training set, with M chosen by crossvalidation. Report the test error obtained, along with the value of M selected by cross-validation.

```
set.seed(45)
#Fit and determine M based on CV results
pcr.fit=pcr(Profit~., data=train, scale=TRUE, validation="CV")
summary(pcr.fit)
```

```
## Data:    X dimension: 874 6
##  Y dimension: 874 1
## Fit method: svdpc
## Number of components considered: 6
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           425.5    19.12    9.053    6.355    5.410    5.314    5.321
## adjCV        425.5    19.12    9.047    6.351    5.407    5.311    5.318
##
## TRAINING: % variance explained
##         1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## X         99.89    99.95    99.98    99.99   100.00   100.00
## Profit    99.80    99.96    99.98    99.98    99.98    99.98
```

```
#visualize cross-validation plots
par(mfrow=c(1,3))
validationplot(pcr.fit)
validationplot(pcr.fit, val.type="MSEP")
validationplot(pcr.fit, val.type="R2")
```



The following is noted:
1. if the intercept term is only used, the test RMSE is **425**
2. if the first PLS component is added, the test RMSE drops to **19.12**
3. if the second PLS component is added, the test RMSE drops to **9.053**
4. if the third PLS component is added, the test RMSE drops to **6.355**
5. if the forth PLS component is added, the test RMSE drops to **5.410**
5. if the fifth PLS component is added, the test RMSE drops to **5.314**

adding PLS components add the test RMSE hence it would be optimal to only use 5 PLS components

```
pcr.pred = predict(pcr.fit,test,ncomp = 5 )

#Model performance metrics
ml_performance.pcr = data.frame(
MODEL = "PCR regression",
R2 = caret::R2(pcr.pred, y_test),
RMSE = RMSE(pcr.pred,y_test),
MAE = MAE(pcr.pred, y_test))

ml_performance.pcr
```

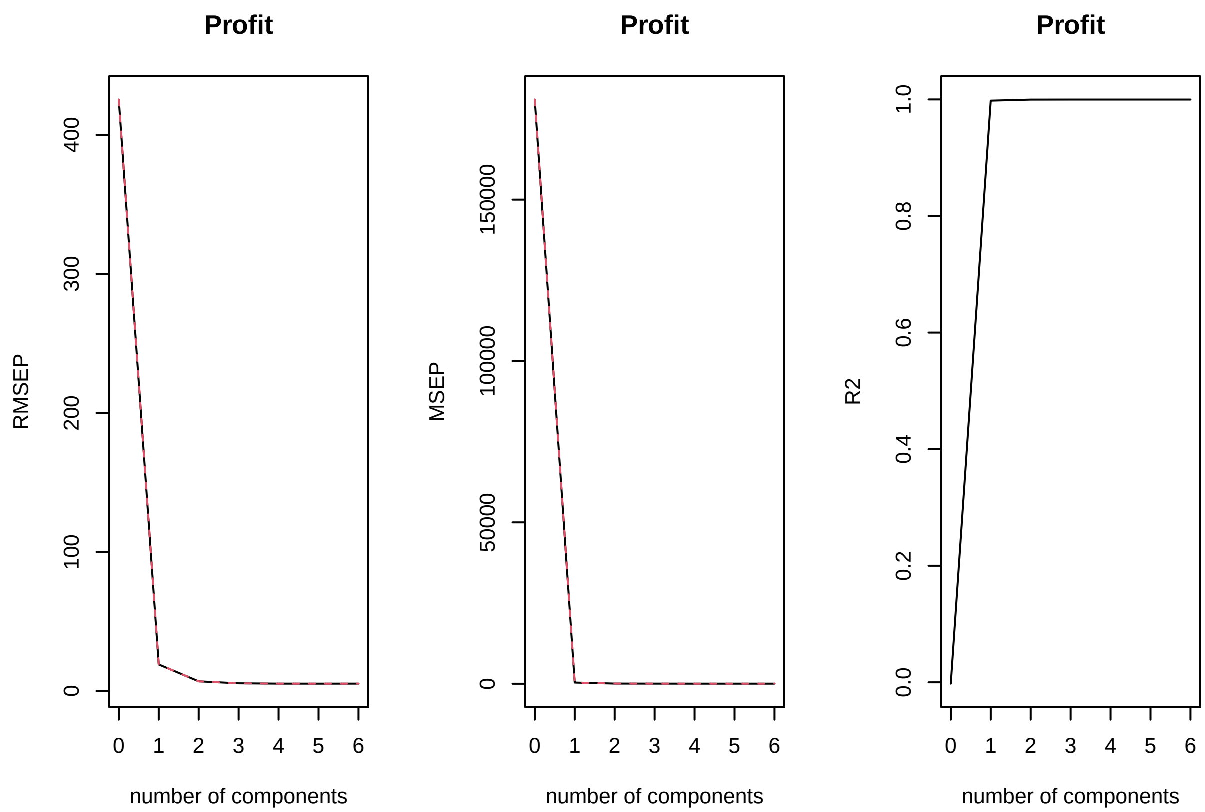
```
##              MODEL        R2     RMSE      MAE
## 1 PCR regression 0.9998877 4.668872 3.879486
```

2.6 Fit a PLS model on the training set, with M chosen by cross validation. Report the test error obtained, along with the value of M selected by cross-validation.

```r
set.seed(4)
#Fit and determine M based on CV results
pls.fit=plsr(Profit~., data=train, scale=TRUE, validation="CV")
summary(pls.fit)
```

```
## Data:    X dimension: 874 6
##  Y dimension: 874 1
## Fit method: kernelpls
## Number of components considered: 6
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           425.5    19.11    6.969    5.536    5.323    5.301    5.304
## adjCV        425.5    19.11    6.968    5.534    5.321    5.299    5.302
##
## TRAINING: % variance explained
##         1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## X         99.89    99.95    99.98    99.99   100.00   100.00
## Profit    99.80    99.97    99.98    99.98    99.98    99.98
```

```r
#visualize cross-validation plots
par(mfrow=c(1,3))
validationplot(pls.fit)
validationplot(pls.fit, val.type="MSEP")
validationplot(pls.fit, val.type="R2")
```



The lowest cross-validation error occurs when only M=5 partial least squares squares is used.

```
pls.pred = predict(pls.fit,test,ncomp = 5 )

#Model performance metrics
ml_performance.pls = data.frame(
MODEL = "PLS regression",
R2 = caret::R2(pls.pred, y_test),
RMSE = RMSE(pls.pred,y_test),
MAE = MAE(pls.pred, y_test))

ml_performance.pcr
```

```
##            MODEL        R2     RMSE      MAE
## 1 PCR regression 0.9998877 4.668872 3.879486
```

2.7 Comment on the results obtained. How accurately can we predict the profits of the organisation? Is there much difference among the test errors resulting from these five approaches?

```
colnames(ml_performance.lasso)[2] = "R2"
colnames(ml_performance.ridge)[2] = "R2"

 r= rbind(
```

```r
  ml_performance.lse,
  ml_performance.lasso,
  ml_performance.ridge,
  ml_performance.pcr,
  ml_performance.pls)
# Sort by R2
sorted_ml = dplyr::arrange(r,desc(R2))


knitr::kable(sorted_ml,"pipe")
```

MODEL	R2	RMSE	MAE
Least Squares	0.9998880	4.666283	3.873894
PLS regression	0.9998879	4.667205	3.876443
PCR regression	0.9998877	4.668872	3.879486
Lasso regression	0.9997780	13.988597	11.357322
Ridge regression	0.9980068	20.952441	12.059696

Least Square is the best model to predict profit since it has the least `RMSE` while Ridge is the worst in predicting Profit since it has the highest `RMSE`