sambu064-report

Assignment-01 Spam classification with nearest neighbors

1.1 Data

We have devided our actual data into training and testing. We will training data in other tasks to train our model and testing data to test our model complexity as well as generalization for the new data.

1.2-3 Logistic Regression

```
##
## y_pred_fifyt   0   1
##            0 808  92
##            1 143 327
```

```
##
## y_pred_eighty   0   1
##             0 931 314
##             1  20 105
```

1.4 kKnn classification for K=30

```
##    knfit
##       0   1
##   0 702 249
##   1 180 239
```

1.5 Kknn classification for k=1

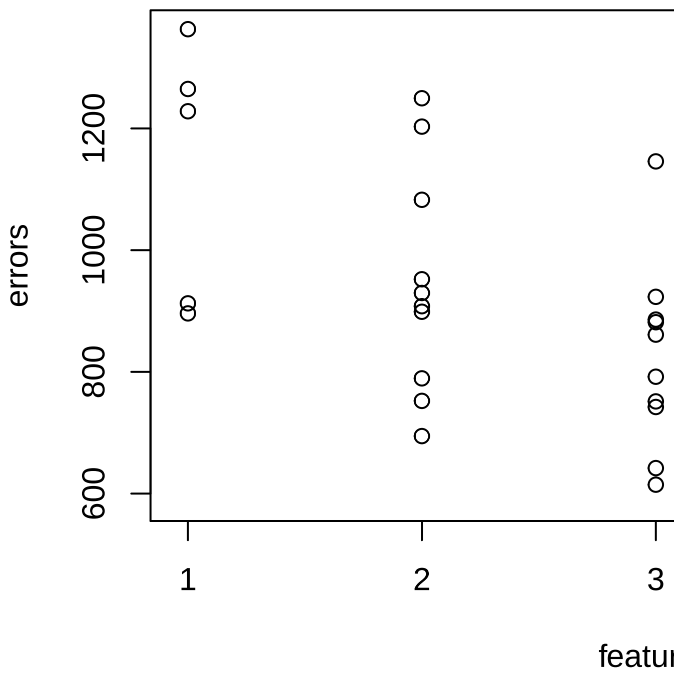
```
##    kknfit2
##       0   1
##   0 644 307
##   1 185 234
```

By analyzing task 1.4 and 1.5 for different values of K we have concluded that, when decreasing the K, Type I Error (FP) and Type II Error (FN) has increased. In our case kknn model underfits for the value of k=1.

Assignment-03 Feature selection by cross-validation in a linear model.

3.1-2 Manual Cross Validation

In this task we have to choose the best features using cross valudation technique by considering Fertility as our reponse variable and other features as predictors.



Below Graph depicts the dependency of errors on the basis of fesures.

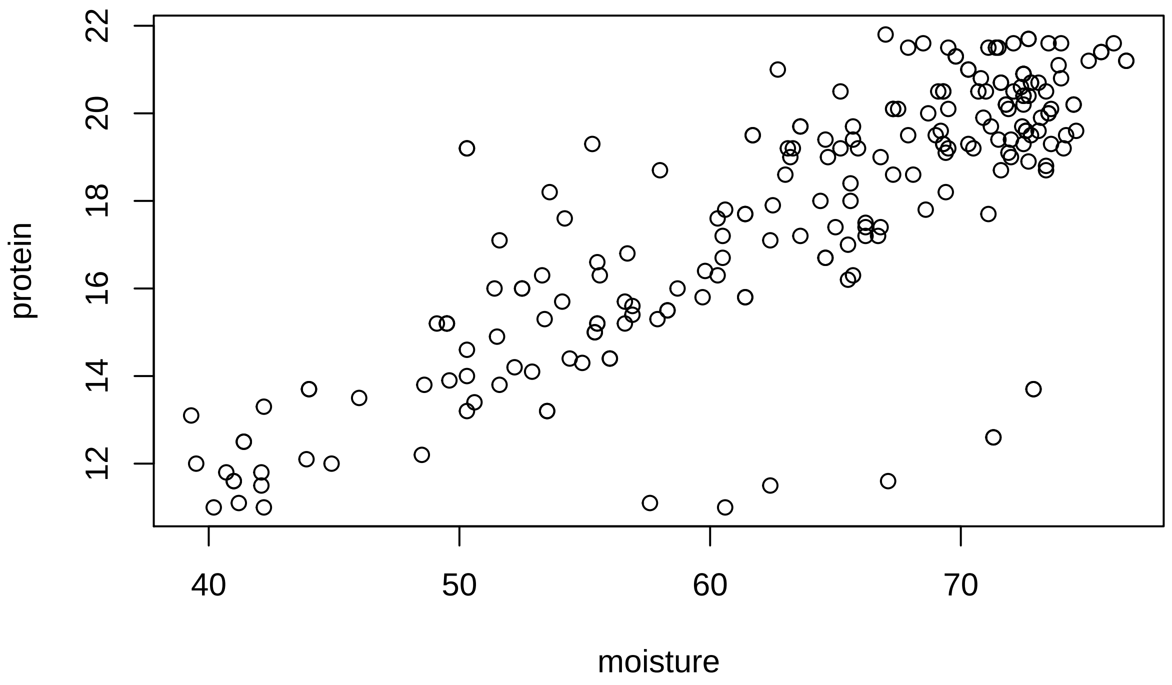
```
## $MinError
## [1] 586.0551
##
## $Feature
## [1] 1 3 4 5
```

As we have to choose the feature which has highest impact and we can analyze the impact of features by mean squred error. We will choose the list of features with lowest mean sequared error. In our case the best feature is `$Feature` with minimum mean squared error `$MinError`

Assignment-04 Linear regression and regularization

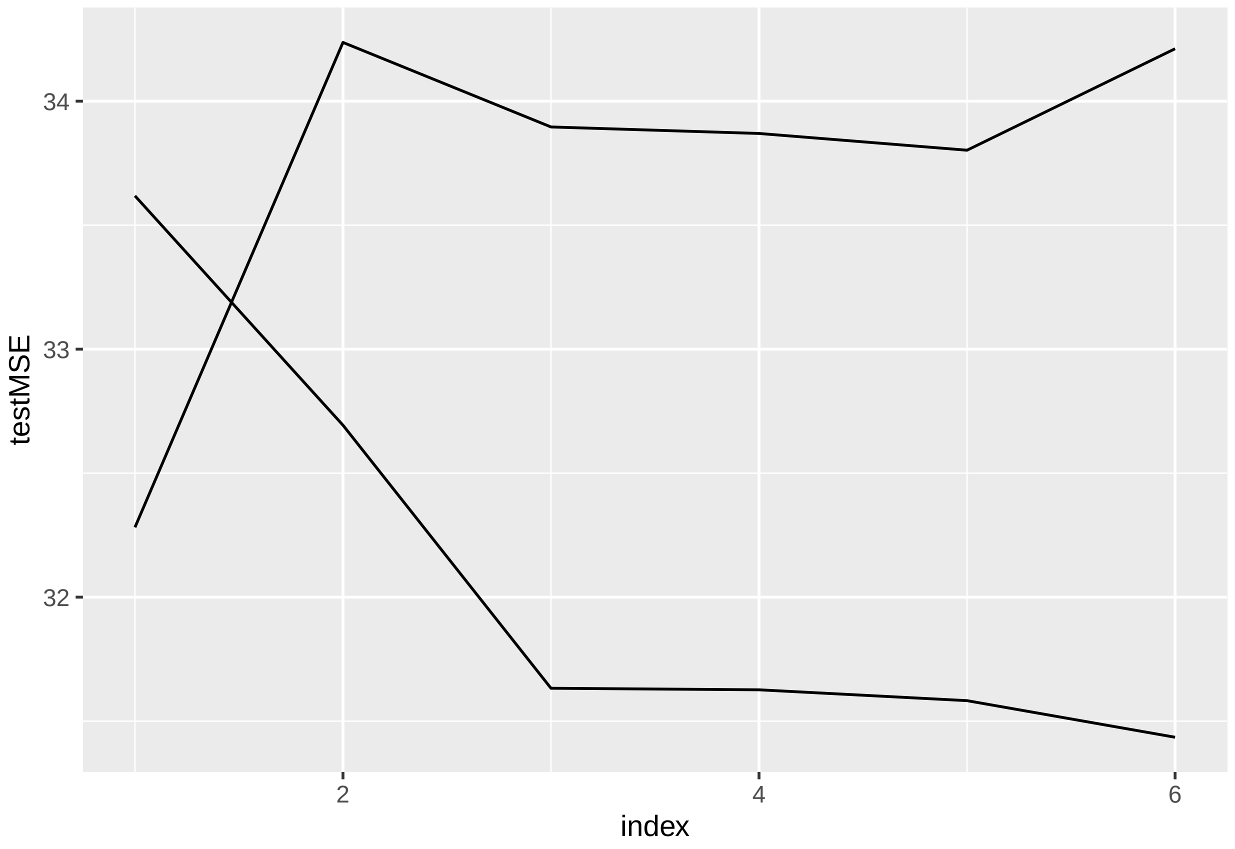
4.1 Linear Relationship

Below graph depicts the linear relatioship between moisture and protein.



According to the above graph, as the moisture increases, protein is also increasing. Hence we can conclude that there is a linear relationship between moisture and protein.

4.2-3 polynomial Model



### 4.4 SetAIC

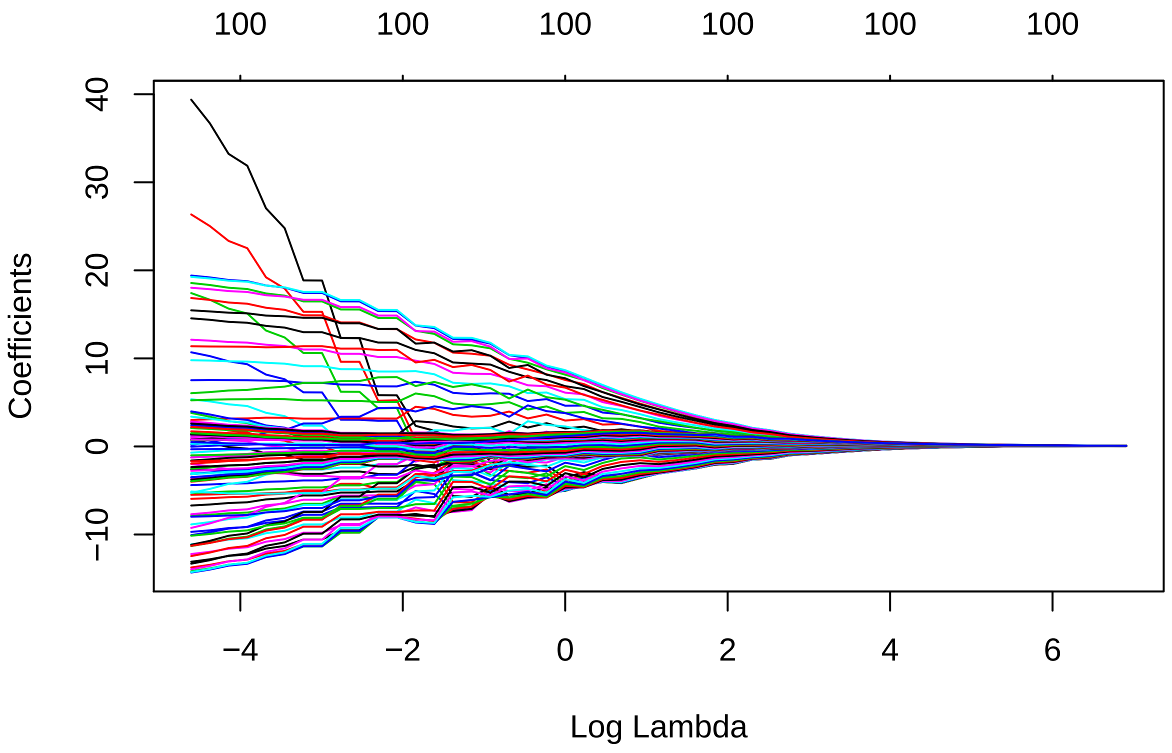
In this task we are selecting features with high imapct by using stepAIC.

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

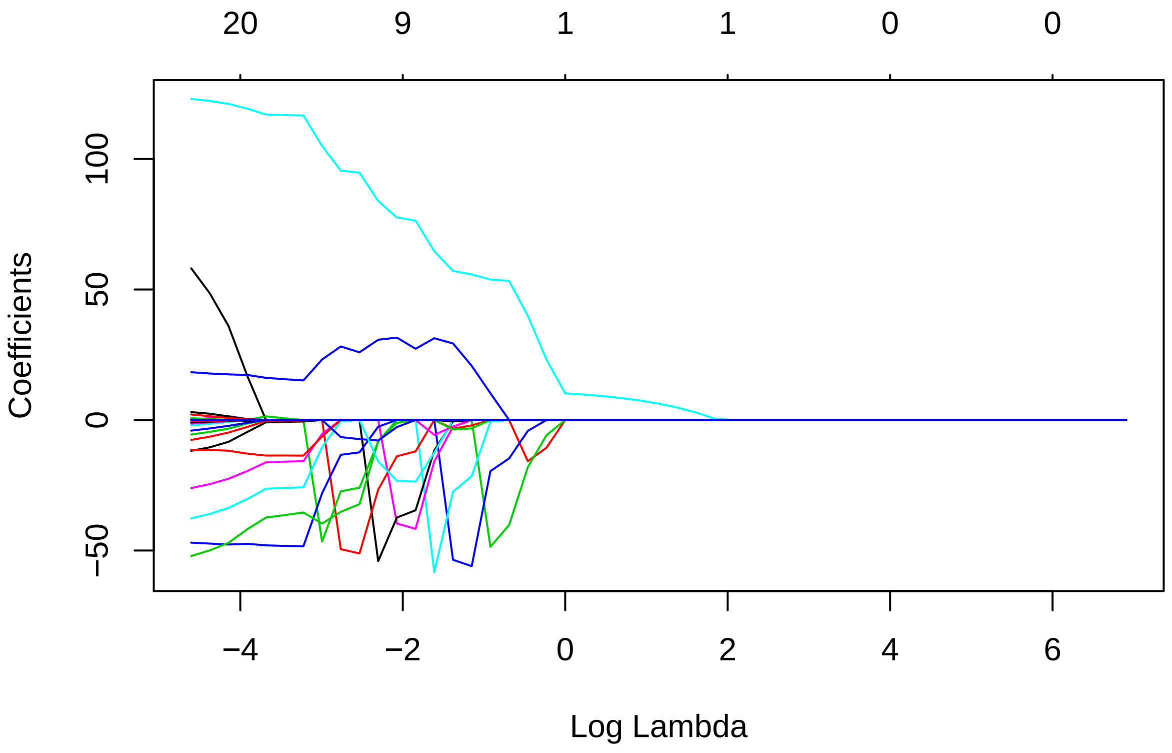
4.5 Ridge Regression

```
## Loading required package: Matrix
```

```
## Loaded glmnet 3.0-1
```



4.6 Lasso Regression



4.7 CV Lasso

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

The best lasso model is when 100 features are selected, and where lambda is equal to zero, and this is where we get the least MSE.

```
## [1] "selected features are  100"
```

When using AIC, 63 features are selected, whereas, when we used lasso all 100 features are selected.

Appendix

```
knitr::opts_chunk$set(echo = FALSE)
library(readxl)
options(warn=-1)

data<-read_excel(path = "spambase.xlsx")
data$Spam<-as.factor(data$Spam)

n=dim(data)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.5))
train=data[id,]
test=data[-id,]

logmodel<-glm(Spam ~ . , family = "binomial",data = train)

pred<-predict(logmodel,test,type = "response")
```

```
y_pred_fifyt <- ifelse(pred > 0.5, 1, 0)

y_pred_eighty <- ifelse(pred > 0.8, 1, 0)

table(y_pred_fifyt,test$Spam)

table(y_pred_eighty,test$Spam)

library(kknn)
knmodel <- kknn (Spam ~ .,train = train,test = test, k=30)

 knfit <- fitted(knmodel)

 table(test$Spam,knfit)


 knmodel2 <- kknn (Spam ~ .,train = train,test = test, k=1)

 kknfit2 <- fitted(knmodel2)

 table(test$Spam,kknfit2)



mylin=function(xtd,y, Xpred){

  xtd <- as.matrix(xtd)
  xtd <- cbind(1,xtd)
  y<-as.matrix(y)
  Xpred1=cbind(1,Xpred)
  beta <- as.matrix((solve((t(xtd) %*% xtd)) %*% t(xtd) %*% y))
  #print(beta)
  #betalm <- coef(lm(y~xtd))
  #print(Xpred1)
  Res=as.matrix(Xpred1) %*% beta
  return(Res)
}

MyCV<- function(X, Y, kfolds){
  k=kfolds
  n=length(Y)
  ind=sample(n,n)
  x=X[ind,]
  y=Y[ind]
  nr <- nrow(x)
  nc <- ncol(x)
  feature <- list()
  MSE=list()
  folds <- cut(seq(1,nrow(X)),breaks=k,labels=FALSE)


  colcomb <- list()
  for (i in 1:5){
```

```r
    colcomb <- c(colcomb,combn(5,i,simplify = FALSE))
  }

 returnse<- sapply(1:length(colcomb), function(j){

   currFeature = colcomb[[j]]

    appresult <- sapply(1:k, function(i){
      indexes <- which(folds==i,arr.ind = TRUE)

      xtrain <- x[-indexes,currFeature]
      xtest <- x[indexes,currFeature]

      Ypred = mylin(xtrain, y[-indexes], xtest)

      SSE=sum((Ypred-y[indexes])^2)

    })

    MSE[j] <- list(mean(appresult),length(currFeature),currFeature)

  })

features = unlist(returnse[2,])
errors = unlist(returnse[1,])
plot(features,errors)

minmse<-which.min(returnse[1,])

return(list(MinError=returnse[,minmse][[1]],Feature=returnse[,minmse][[3]]))
}
RNGkind(sample.kind = "Rounding")
MyCV(swiss[,-1],swiss[,1],5)
X = swiss[,-1]
Y = swiss[,1]
kfolds= 5

#load(file = "tecator_data.rda")
data = read_excel("tecator.xlsx")
n=dim(data)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.5))
train=data[id,]
test=data[-id,]

lnmodel = lm(Moisture~Protein, data=train)

ypred = predict(lnmodel,newdata = test)

moisture = data$Moisture
protein = data$Protein

plot(moisture,protein)
```

```r
library(ggplot2)
resultMSE <- sapply(1:6, function(i){
 lmresult = lm(Moisture~poly(Protein,i) , data = train)

 ypredicttest = predict(lmresult,newdata = test )
 ypredicttrain = predict(lmresult,newdata = train )

 MSETest = mean((ypredicttest-test$Moisture)^2)
 MSETrain = mean((ypredicttrain-train$Moisture)^2)
  combList = list(testMSE = MSETest, trainMSE = MSETrain, index = i)
})

resultgraph = as.data.frame(t(resultMSE))
resultgraph = as.data.frame(lapply(resultgraph, unlist))

ggplot(fortify(resultgraph))+
  geom_line(aes(x = index , y = testMSE ))+
  geom_line(aes(x = index , y = trainMSE ))

library(MASS)
subdata = data[,-c(1,103,104)]
lnmodelaic = lm(Fat~., data=subdata)
resuaic = stepAIC(lnmodelaic,direction = "backward", trace = FALSE)

length(resuaic$coefficients)-1
library(glmnet)
lambdas = 10^seq(3, -2, by = -0.1)
ridgreg = glmnet(x = as.matrix(subdata[,-101]),y=  as.matrix(subdata$Fat),alpha = 0,lambda = lambdas)
plot(ridgreg,xvar = "lambda")
lassoreg = glmnet(as.matrix(subdata[,-101]) ,subdata$Fat,alpha = 1,lambda = lambdas)
plot(lassoreg,xvar = "lambda")
lambdascv = 10^seq(3, -2, by = -0.1)
lambdascv[length(lambdascv)+1] = 0
calLambda = cv.glmnet(as.matrix(subdata[,-101]) ,subdata$Fat,alpha = 1,lambda = lambdascv)

#plot(calLambda)

optimalLambda = calLambda$lambda.min
optimalLambda
resulglmopti = glmnet(as.matrix(subdata[,-101]) ,subdata$Fat,alpha = 1,lambda = optimalLambda)
coefficients = coef(resulglmopti)
paste("selected features are ",length(coefficients)-1)
```