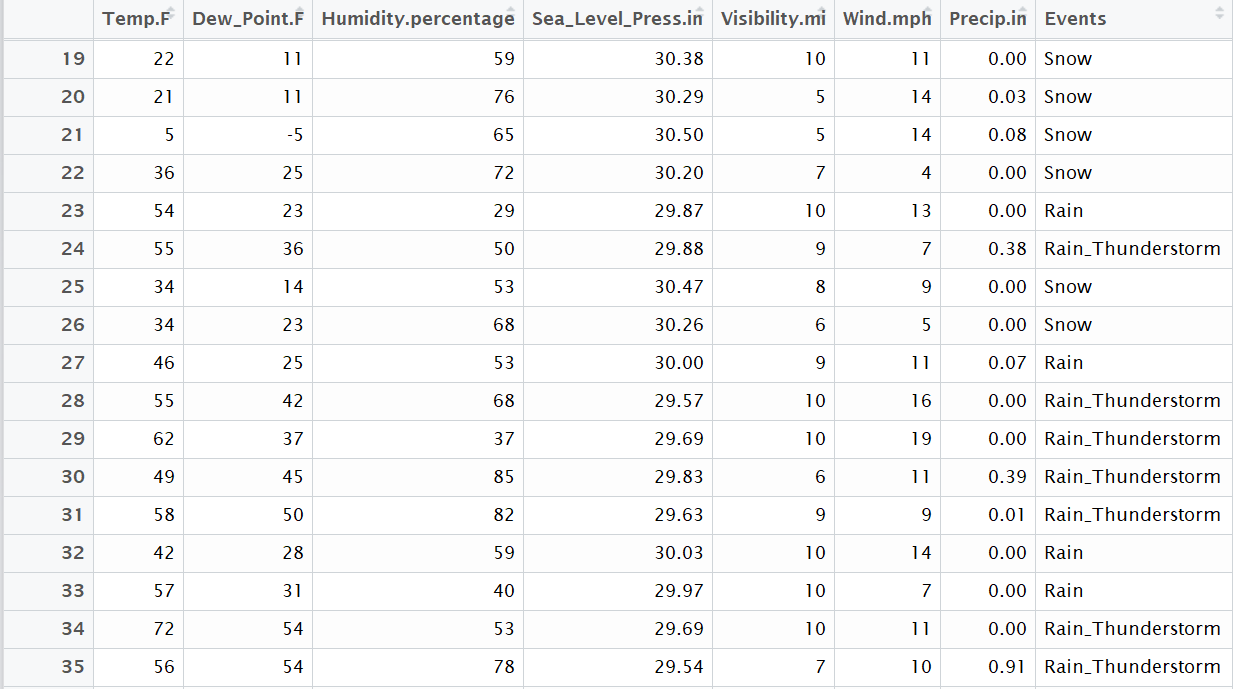


*#import the dataset and make some changes*

library(readr)

kc\_weather\_srt <- read\_csv("C:/Users/bvkka/Desktop/ISL-Deep Medhi/kc\_weather\_srt.csv")

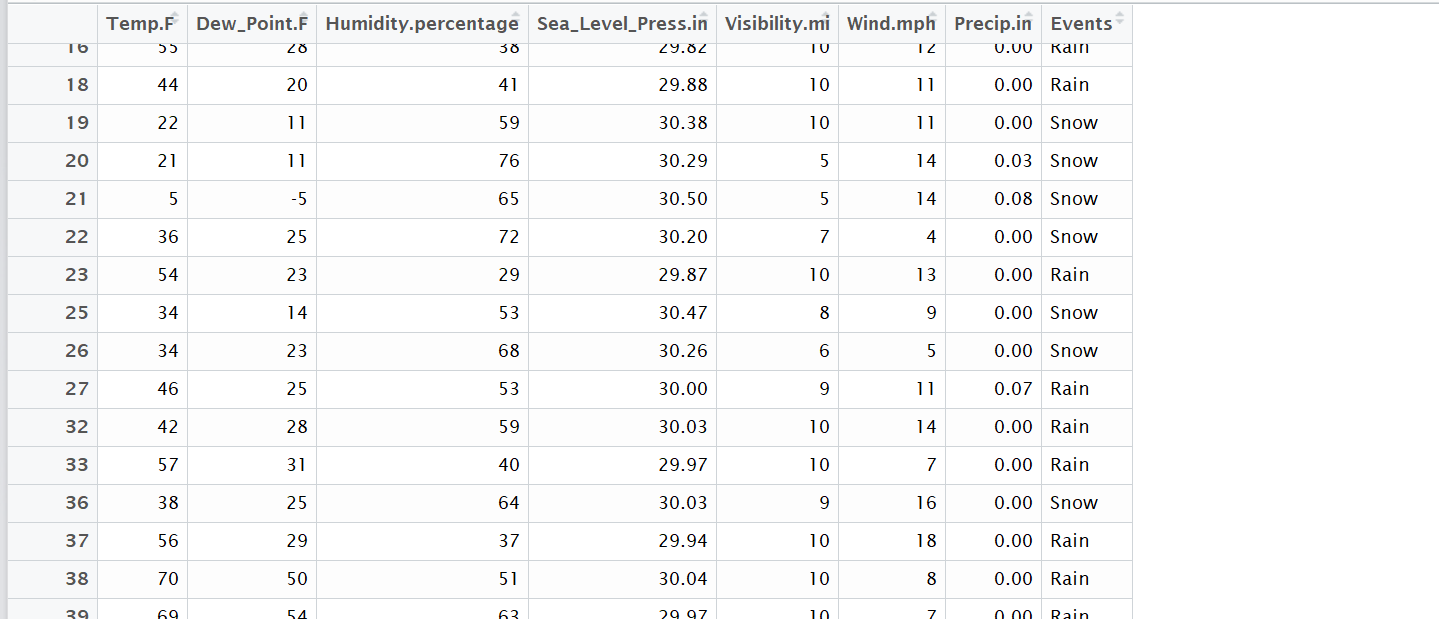
kc\_weather\_srt=kc\_weather\_srt[,2:9] *#selecting all the predictors and response column*



*#subset that consists of only rain and snow*

kc\_weather\_srt\_without\_rainthunderstorm<-kc\_weather\_srt[!grepl("Rain\_Thunderstorm",kc\_weather\_srt$Events),]

dim(kc\_weather\_srt\_without\_rainthunderstorm) *#dimensions*



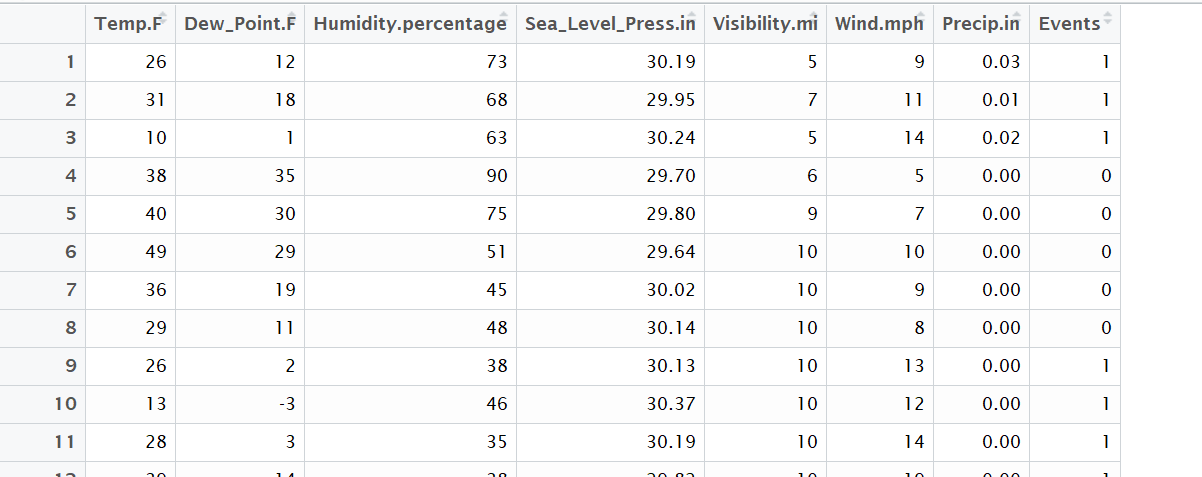
*#first make the response column to 0 and 1 (qualitative)*

*#install.packages("plyr")*

library(plyr)

kc\_weather\_srt\_without\_rainthunderstorm$Events <- revalue(kc\_weather\_srt\_without\_rainthunderstorm$Events,c("Snow"=1))

kc\_weather\_srt\_without\_rainthunderstorm$Events <- revalue(kc\_weather\_srt\_without\_rainthunderstorm$Events,c("Rain"=0))



*#character to numeric*

kc\_weather\_srt\_without\_rainthunderstorm$Events<-as.numeric(as.character(kc\_weather\_srt\_without\_rainthunderstorm$Events))

*# number of replications*

rep=100

*# newly added*

*#snow=1 rain=0*

accuracy=dim(rep)

accuracy1=dim(rep)

accuracy2=dim(rep)

accuracy3=dim(rep)

precision\_snow=dim(rep)

precision\_rain=dim(rep)

recall\_snow=dim(rep)

recall\_rain=dim(rep)

precision\_snow1=dim(rep)

precision\_rain1=dim(rep)

recall\_snow1=dim(rep)

recall\_rain1=dim(rep)

precision\_snow2=dim(rep)

precision\_rain2=dim(rep)

recall\_snow2=dim(rep)

recall\_rain2=dim(rep)

precision\_snow3=dim(rep)

precision\_rain3=dim(rep)

recall\_snow3=dim(rep)

recall\_rain3=dim(rep)

*#splitting the dataset into training and test sets, also install caTools packages*

*#install.packages('caTools')*

library(caTools)

set.seed(123)

for(k in 1:rep)

{

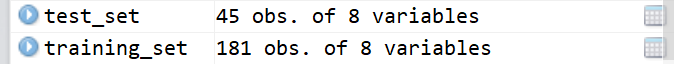
split=sample.split(kc\_weather\_srt\_without\_rainthunderstorm$Events,SplitRatio = 0.8) *#80% split ratio*

training\_set=subset(kc\_weather\_srt\_without\_rainthunderstorm,split==TRUE) *#80% split into training set*

test\_set=subset(kc\_weather\_srt\_without\_rainthunderstorm,split==FALSE) *#20% split into testing set*

table(split)

dim(training\_set)



*#\*\*\*\*Logistic Regression\*\*\*#*

*#Fitting Logistic Regression to the Training set*

model1<-glm(formula = Events ~ ., binomial(link="logit"),data =training\_set)

*#predicting the test set results*

prob\_pred=predict(model1,type='response',newdata = test\_set[-8]) *#for predicitng we only need predictors , but not response*

summary(prob\_pred)

y\_pred=ifelse(prob\_pred>0.5,1,0) *#vector of predictions #Threshold value of 0.5*

y\_pred

*#Making the confusion Matrix*

cm=table(y\_pred,test\_set[,8])

cm

*#accuracy*

accuracy[k]=mean(y\_pred==test\_set[,8])

accuracy

*#precision*

precision=precision<-diag(cm)/colSums(cm)

precision\_snow[k]=precision[2]

precision\_rain[k]=precision[1]

*#recall*

recall=recall<-diag(cm)/rowSums(cm)

recall\_snow[k]=recall[2]

recall\_rain[k]=recall[1]

*#\*\*\*LDA\*\*\*#*

*#install.packages("MASS")*

library(MASS)

lda=lda(formula=Events~.,data=training\_set)

y\_pred1=predict(lda,test\_set)$class

cm1=table(y\_pred1,test\_set[,8])

cm1

accuracy1[k]=mean(y\_pred1==test\_set[,8])

precision1=precison1<-diag(cm1/colSums(cm1))

precision\_snow1[k]=precison1[2]

precision\_rain1[k]=precision1[1]

recall1=recall1<-diag(cm1/rowSums(cm1))

recall\_snow1[k]=recall1[2]

recall\_rain1[k]=recall1[1]

#### or use SVM (but not for this project)###

*#lda=lda(formula=Events~.,data=training\_set)*

*#training\_set1=as.data.frame(predict(lda,training\_set))*

*#training\_set1=training\_set1[c(4,1)]*

*#plot(lda)*

*#test\_set1=as.data.frame(predict(lda,test\_set))*

*#test\_set1=test\_set1[c(4,1)]*

*#fitting SVM to the training set*

*#install.packages('e1071')*

*#library(e1071)*

*#classifier=svm(formula=class~.,data=training\_set1,type='C-classification',kernel='linear')*

*#predicting the test set results*

*#y\_pred1=predict(classifier,newdata = test\_set1[-2])*

*#making the confusion matrix*

*#cm1=table(y\_pred1,test\_set1[,2])*

*#cm1 #we see TP+TN=33+12=45 true predictions and FP+FN=0+0=0 False predictions, therefore it is 100% accurate*

*#accuracy1[k]=mean(y\_pred1==test\_set1[,2])*

*#accuracy1*

*#\*\*\*QDA\*\*\*#*

qda=qda(formula=Events~.,data=training\_set)

y\_pred2=predict(qda,test\_set)$class

cm2=table(y\_pred2,test\_set[,8])

cm2

accuracy2[k]=mean(y\_pred2==test\_set[,8])

precision2=precison2<-diag(cm2/colSums(cm2))

precision\_snow2[k]=precison2[2]

precision\_rain2[k]=precision2[1]

recall2=recall2<-diag(cm2/rowSums(cm2))

recall\_snow2[k]=recall2[2]

recall\_rain2[k]=recall2[1]

*#\*\*\*KNN\*\*\*#*

*#install.packages('class')*

*#fitting KNN to the training set and predicting the test set results*

library(class)

y\_pred3=knn(train=training\_set[,-8],test=test\_set[,-8],cl=training\_set[,8],k=5)

*#making the cm*

cm3=table(y\_pred3,test\_set[,8])

cm3

accuracy3[k]=mean(y\_pred3==test\_set[,8])

precision3=precision3<-diag(cm3/colSums(cm3))

precision\_snow3[k]=precision3[2]

precision\_rain3[k]=precision3[1]

recall3=recall3<-recall3<-diag(cm3/rowSums(cm3))

recall\_snow3[k]=recall3[2]

recall\_rain3[k]=recall3[1]

}

*###\*\*\*\*\*\*\*MEAN VALUES\*\*\*\*\*#####*

*#\*\*\*\*Logestic regression\*\*\*#*

mean(accuracy)*###0.9531111*

mean(precision\_snow)*###0.889*

mean(precision\_rain)*###0.9714286*

mean(recall\_snow) *### 0.9075805*

mean(recall\_rain) *### 0.9688803*

*#\*\*\*\*LDA\*\*\*#*

mean(accuracy1)*###0.936222*

mean(precision\_snow1)*###0.865*

mean(precision\_rain1)*###0.9565714*

mean(recall\_snow1) *### 0.8595878*

mean(recall\_rain1) *### 0.9618026*

*#\*\*\*\*QDA\*\*\*#*

mean(accuracy2)*###0.9346667*

mean(precision\_snow2)*###0.952*

mean(precision\_rain2)*###0.9297143*

mean(recall\_snow2) *### 0.8043408*

mean(recall\_rain2) *### 0.985939*

*#\*\*\*\*KNN\*\*\*#*

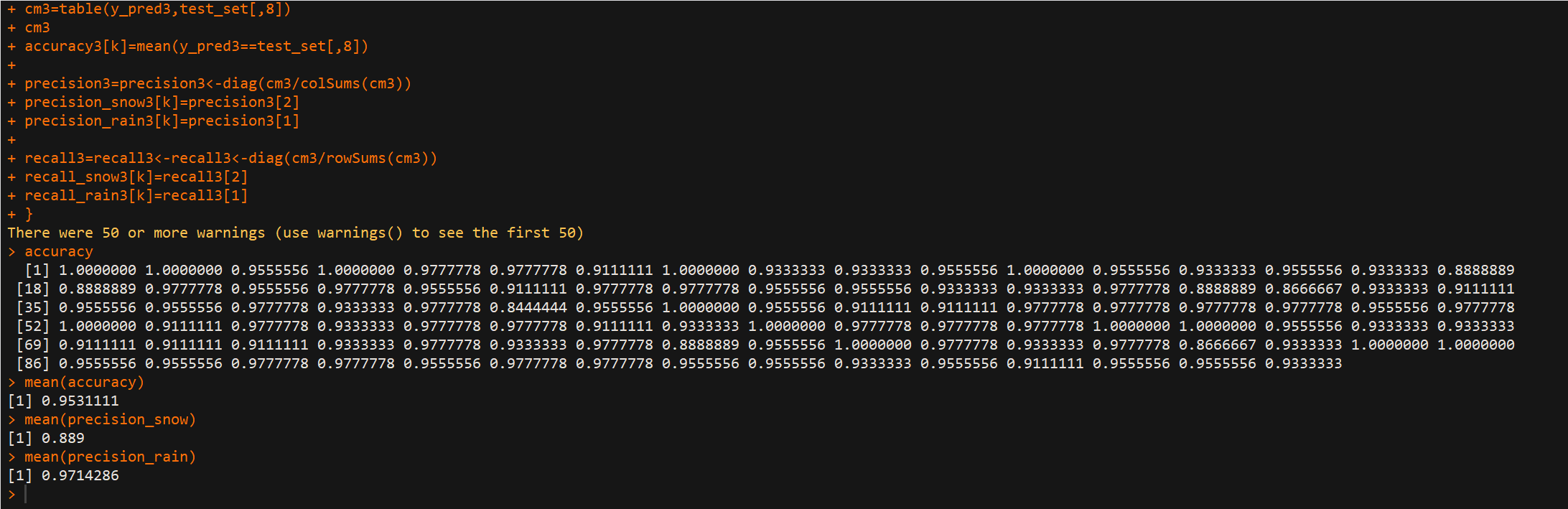
mean(accuracy3)*###0.9486667*

mean(precision\_snow3)*###0.878*

mean(precision\_rain3)*###0.9688571*

mean(recall\_snow3) *### 0.89958469*

mean(recall\_rain3) *### 0.9658621*



**Summary:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model | Accuracy | Precision Snow | Precision Rain | Recall Snow | Recall Rain |
| Logistic Regression | 0.9531111 | 0.889 | 0.9714286 | 0.9075805 | 0.9688803 |
| LDA | 0.936222 | 0.865 | 0.9565714 | 0.8595878 | 0.9618026 |
| QDA | 0.9346667 | 0.952 | 0.9297143 | 0.8043408 | 0.985939 |
| KNN (K=5) | 0.9486667 | 0.878 | 0.9688571 | 0.89958469 | 0.9658621 |

**Text Summarization:**

*#As you can see accuracy of LR>KNN>LDA>QDA, we should opt for Logistic Regression*

*#As you can see Precision of Snow, QDA>LR>KNN>LDA*

*#As you can see Precision of rain, LR>KNN>LDA>QDA*

*#As you can see Recall of Snow, LR>KNN>LDA>QDA*

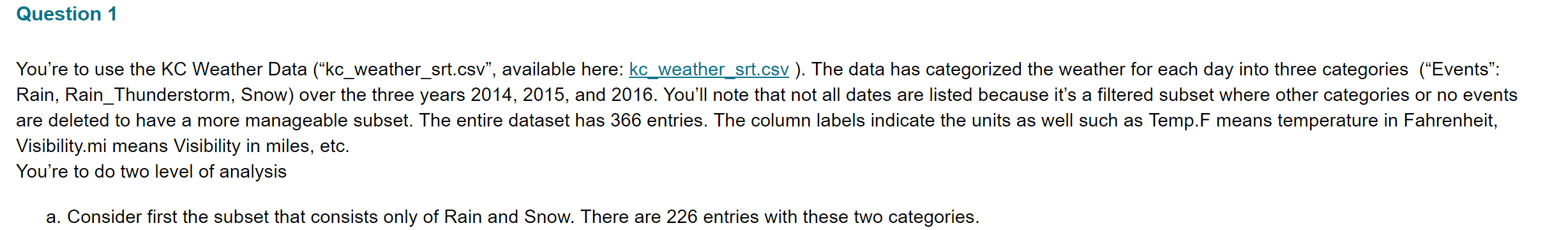
*#As you can see Recall of rain, LR>KNN>LDA>QDA*

*###decision boundary is not very highly non-linear in this case, so we see Logistic Regression dominating KNN and LDA.*

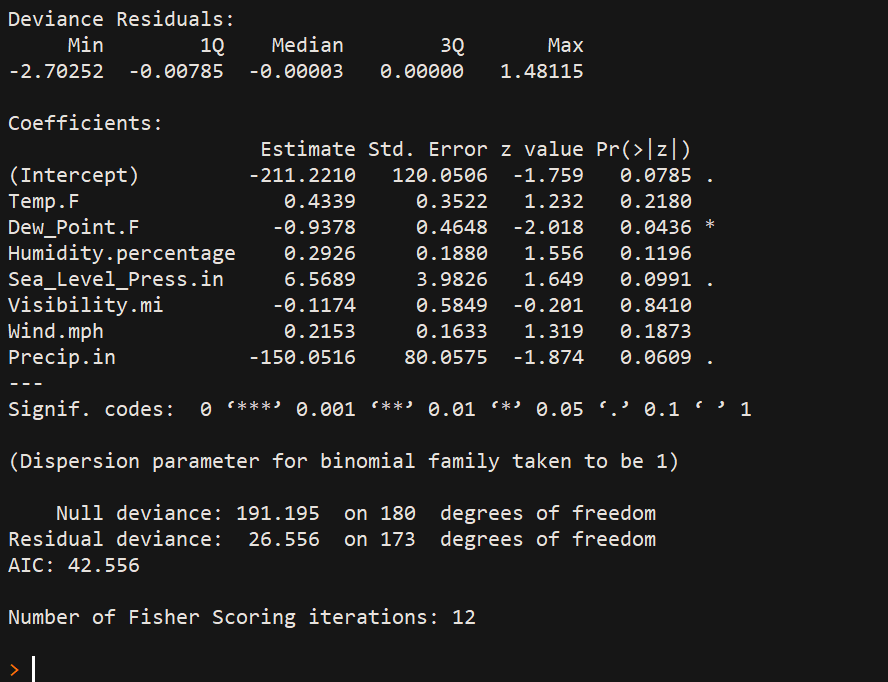
*###KNN is a completely non-parametric approach: no assumptions are made about the shape of the decision boundary. Therefore, we can expect this approach to dominate LDA and logistic regression when the decision boundary is highly non-linear. On the other hand, KNN does not tell us which predictors are important; we don’t get a table of coefficients with p-values*

*###QDA serves as a compromise between KNN, LDA and LR, In this case there are more number of training observations, so QDA doesnt perform well*

*###\*\*\*So based on results, Logestic regression performs better in accuracy, precision and recall, so we choose Logestic Regression model on this dataset\*\*\*\*###*







*From the first problem, when I performed Summary of Logistic Regression, I found that p values of Temp.f, humidity%, visibility and wind are high. So, I removed these predictors and just took Dewpoint, SealevelPress, and Precip.in predictors since their p values are considerably lower.*

*###Removing some predictors after noticing p-values from summary of Logistic Regression table###*

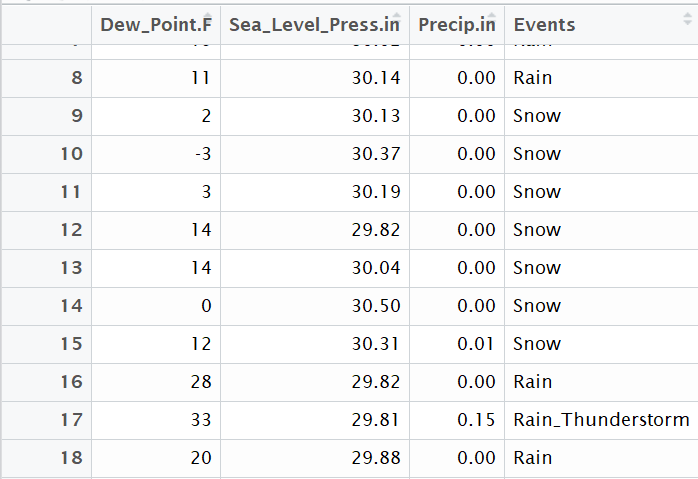
*###Taking only Dewpoint, SeaLevel and Precipitation predictors, removing rest. Let's hope for the best results.*

*#import the dataset and make some changes*

library(readr)

kc\_weather\_srt <- read\_csv("C:/Users/bvkka/Desktop/ISL-Deep Medhi/kc\_weather\_srt.csv")

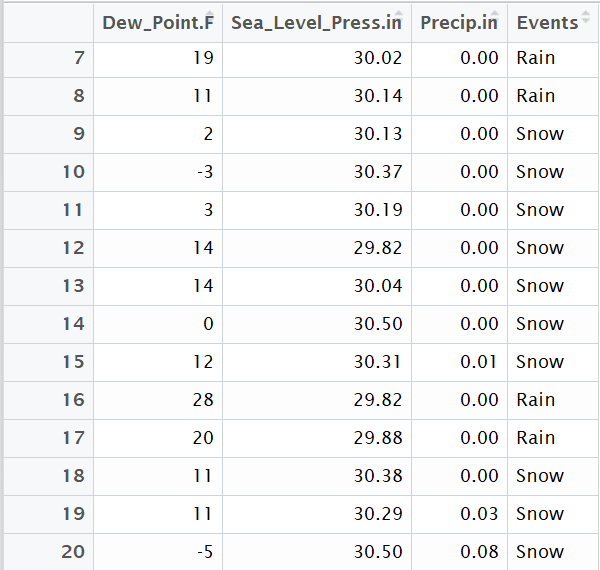
kc\_weather\_srt=kc\_weather\_srt[c(2,4,7,8,9)]



*#subset that consists of only rain and snow*

kc\_weather\_srt\_without\_rainthunderstorm<-kc\_weather\_srt[!grepl("Rain\_Thunderstorm",kc\_weather\_srt$Events),]

dim(kc\_weather\_srt\_without\_rainthunderstorm)



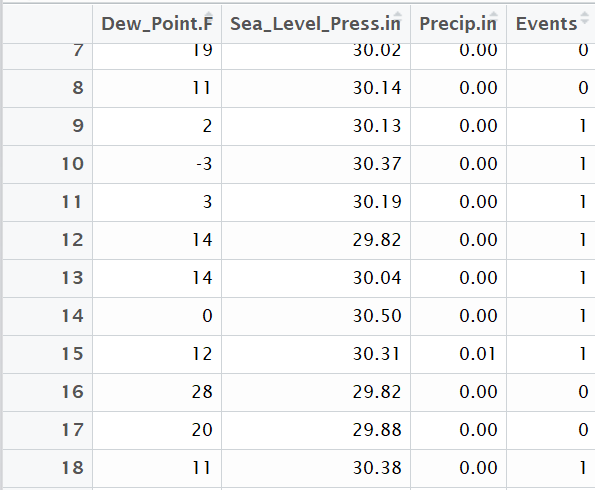
*#first make the response column to 0 and 1*

*#install.packages("plyr")*

library(plyr)

kc\_weather\_srt\_without\_rainthunderstorm$Events <- revalue(kc\_weather\_srt\_without\_rainthunderstorm$Events,c("Snow"=1))

kc\_weather\_srt\_without\_rainthunderstorm$Events <- revalue(kc\_weather\_srt\_without\_rainthunderstorm$Events,c("Rain"=0))



*#character to numeric*

kc\_weather\_srt\_without\_rainthunderstorm$Events<-as.numeric(as.character(kc\_weather\_srt\_without\_rainthunderstorm$Events))

*# number of replications*

rep=100

*# newly added*

*#snow=1 rain=0*

accuracy=dim(rep)

accuracy1=dim(rep)

accuracy2=dim(rep)

accuracy3=dim(rep)

precision\_snow=dim(rep)

precision\_rain=dim(rep)

recall\_snow=dim(rep)

recall\_rain=dim(rep)

precision\_snow1=dim(rep)

precision\_rain1=dim(rep)

recall\_snow1=dim(rep)

recall\_rain1=dim(rep)

precision\_snow2=dim(rep)

precision\_rain2=dim(rep)

recall\_snow2=dim(rep)

recall\_rain2=dim(rep)

precision\_snow3=dim(rep)

precision\_rain3=dim(rep)

recall\_snow3=dim(rep)

recall\_rain3=dim(rep)

*#splitting the dataset into training and test sets, also install caTools packages*

*#install.packages('caTools')*

library(caTools)

set.seed(123)

for(k in 1:rep)

{

split=sample.split(kc\_weather\_srt\_without\_rainthunderstorm$Events,SplitRatio = 0.8) *#80% split ratio*

training\_set=subset(kc\_weather\_srt\_without\_rainthunderstorm,split==TRUE) *#80% split into training set*

test\_set=subset(kc\_weather\_srt\_without\_rainthunderstorm,split==FALSE) *#20% split into testing set*

table(split)

dim(training\_set)

*#\*\*\*\*Logistic Regression\*\*\*#*

*#Fitting Logistic Regression to the Training set*

model1<-glm(formula = Events ~ ., binomial(link="logit"),data =training\_set)

*#predicting the test set results*

prob\_pred=predict(model1,type='response',newdata = test\_set[-4]) *#for predicitng we only need predictors , but not response*

summary(prob\_pred)

y\_pred=ifelse(prob\_pred>0.5,1,0) *#vector of predictions*

y\_pred

*#Making the confusion Matrix*

cm=table(y\_pred,test\_set[,4])

cm

*#accuracy*

accuracy[k]=mean(y\_pred==test\_set[,4])

accuracy

*#precision*

precision=precision<-diag(cm)/colSums(cm)

precision\_snow[k]=precision[2]

precision\_rain[k]=precision[1]

*#recall*

recall=recall<-diag(cm)/rowSums(cm)

recall\_snow[k]=recall[2]

recall\_rain[k]=recall[1]

*#\*\*\*LDA\*\*\*#*

*#install.packages("MASS")*

library(MASS)

lda=lda(formula=Events~.,data=training\_set)

y\_pred1=predict(lda,test\_set)$class

cm1=table(y\_pred1,test\_set[,4])

cm1

accuracy1[k]=mean(y\_pred1==test\_set[,4])

precision1=precison1<-diag(cm1/colSums(cm1))

precision\_snow1[k]=precison1[2]

precision\_rain1[k]=precision1[1]

recall1=recall1<-diag(cm1/rowSums(cm1))

recall\_snow1[k]=recall1[2]

recall\_rain1[k]=recall1[1]

*#a=lda(formula=Events~.,data=training\_set)*

*#training\_set1=as.data.frame(predict(lda,training\_set))*

*#training\_set1=training\_set1[c(4,1)]*

*#plot(lda)*

*#test\_set1=as.data.frame(predict(lda,test\_set))*

*#test\_set1=test\_set1[c(4,1)]*

*#fitting SVM to the training set*

*#install.packages('e1071')*

*#library(e1071)*

*#classifier=svm(formula=class~.,data=training\_set1,type='C-classification',kernel='linear')*

*#predicting the test set results*

*#y\_pred1=predict(classifier,newdata = test\_set1[-2])*

*#making the confusion matrix*

*#cm1=table(y\_pred1,test\_set1[,2])*

*#cm1 #we see TP+TN=33+12=45 true predictions and FP+FN=0+0=0 False predictions, therefore it is 100% accurate*

*#accuracy1[k]=mean(y\_pred1==test\_set1[,2])*

*#accuracy1*

*#\*\*\*QDA\*\*\*#*

qda=qda(formula=Events~.,data=training\_set)

y\_pred2=predict(qda,test\_set)$class

cm2=table(y\_pred2,test\_set[,4])

cm2

accuracy2[k]=mean(y\_pred2==test\_set[,4])

precision2=precison2<-diag(cm2/colSums(cm2))

precision\_snow2[k]=precison2[2]

precision\_rain2[k]=precision2[1]

recall2=recall2<-diag(cm2/rowSums(cm2))

recall\_snow2[k]=recall2[2]

recall\_rain2[k]=recall2[1]

*#\*\*\*KNN\*\*\*#*

*#install.packages('class')*

*#fitting KNN to the training set and predicting the test set results*

library(class)

y\_pred3=knn(train=training\_set[,-4],test=test\_set[,-4],cl=training\_set[,4],k=5)

*#making the cm*

cm3=table(y\_pred3,test\_set[,4])

cm3

accuracy3[k]=mean(y\_pred3==test\_set[,4])

precision3=precision3<-diag(cm3/colSums(cm3))

precision\_snow3[k]=precision3[2]

precision\_rain3[k]=precision3[1]

recall3=recall3<-recall3<-diag(cm3/rowSums(cm3))

recall\_snow3[k]=recall3[2]

recall\_rain3[k]=recall3[1]

}

*###\*\*\*\*\*\*\*MEAN VALUES\*\*\*\*\*#####*

*#\*\*\*\*Logestic regression\*\*\*#*

mean(accuracy)*###0.95022*

mean(precision\_snow)*###0.891*

mean(precision\_rain)*###0.967*

mean(recall\_snow) *### 0.8958132*

mean(recall\_rain) *### 0.9697611*

*#\*\*\*\*LDA\*\*\*#*

mean(accuracy1)*###0.95*

mean(precision\_snow1)*###0.9*

mean(precision\_rain1)*###0.9642857*

mean(recall\_snow1) *### 0.866313*

mean(recall\_rain1) *### 0.971967*

*#\*\*\*\*QDA\*\*\*#*

mean(accuracy2)*###0.9088889*

mean(precision\_snow2)*###0.938*

mean(precision\_rain2)*###0.9005714*

mean(recall\_snow2) *### 0.7419847*

mean(recall\_rain2) *### 0.9813024*

*#\*\*\*\*KNN\*\*\*#*

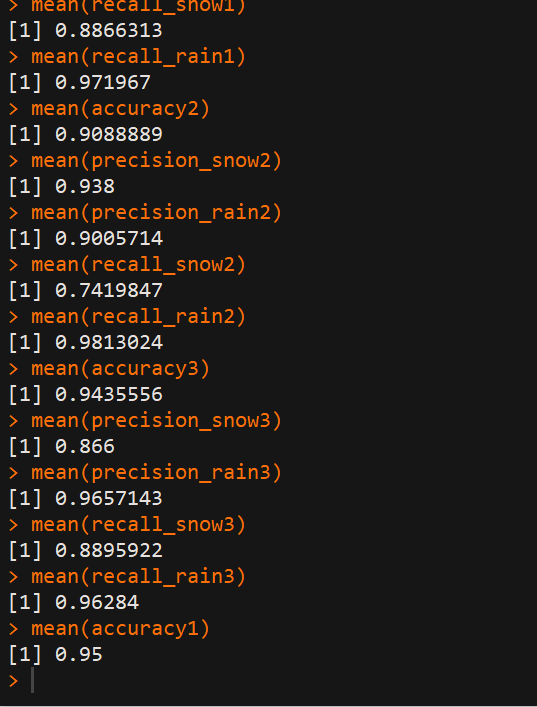
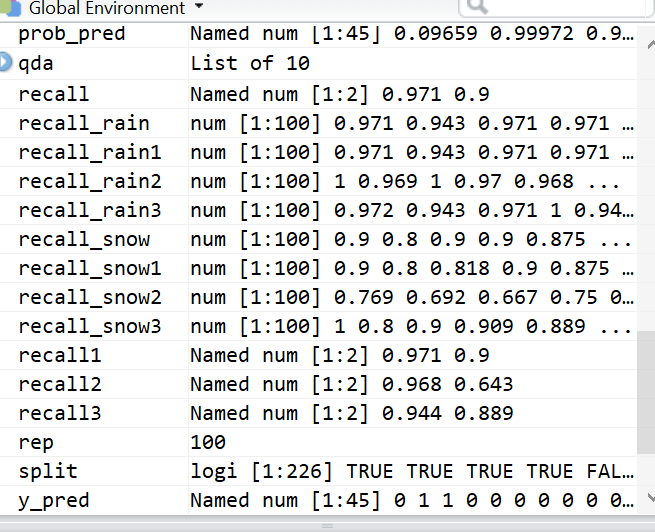
mean(accuracy3)*###0.94355556*

mean(precision\_snow3)*###0.8666*

mean(precision\_rain3)*###0.9657143*

mean(recall\_snow3) *### 0.8895922*

mean(recall\_rain3) *### 0.96284*

**Summary:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model | Accuracy | Precision Snow | Precision Rain | Recall Snow | Recall Rain |
| Logistic Regression | 0.95022 | 0.891 | 0.967 | 0.8958132 | 0.9697611 |
| LDA | 0.95 | 0.9 | 0.9642857 | 0.866313 | 0.971967 |
| QDA | 0.9088889 | 0.938 | 0.9005714 | 0.7419847 | 0.9813024 |
| KNN (K=5) | 0.94355556 | 0.86666 | 0.9657143 | 0.8895922 | 0.96284 |

**Text Summarization:**

*#As you can see accuracy of LDA=LR>KNN>QDA, we should opt for LR*

*#As you can see Precision of Snow , QDA>LDA>LR>KNN QDA gives highest*

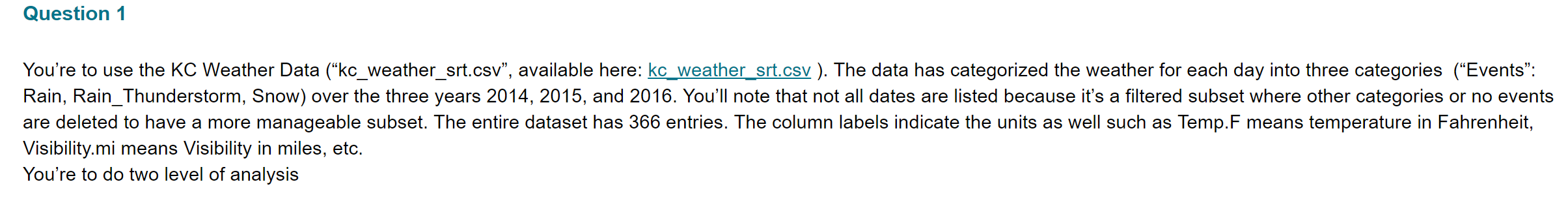
*#As you can see Precision of rain , KNN=LR>LDA >QDA LR gives highest*

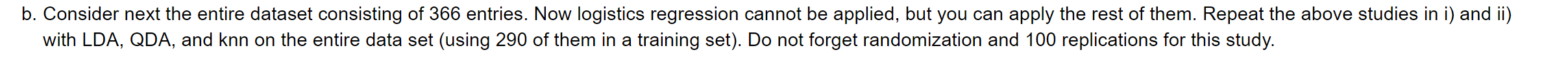
*#As you can see Recall of Snow , KNN~=LR>LDA>QDA LR gives highest*

*#As you can see Recall of rain , QDA>LDA>LR>KNN ,,,QDA gives highest, outperforms LR, KNN and LR*

*## QDA can perform better in the presence of a limited number of training observations because it does make some assumptions about the form of the decision boundary*

*###\*\*\*So based on results, QDA AND LR are close but we choose QDA model on this dataset\*\*\*\*###*



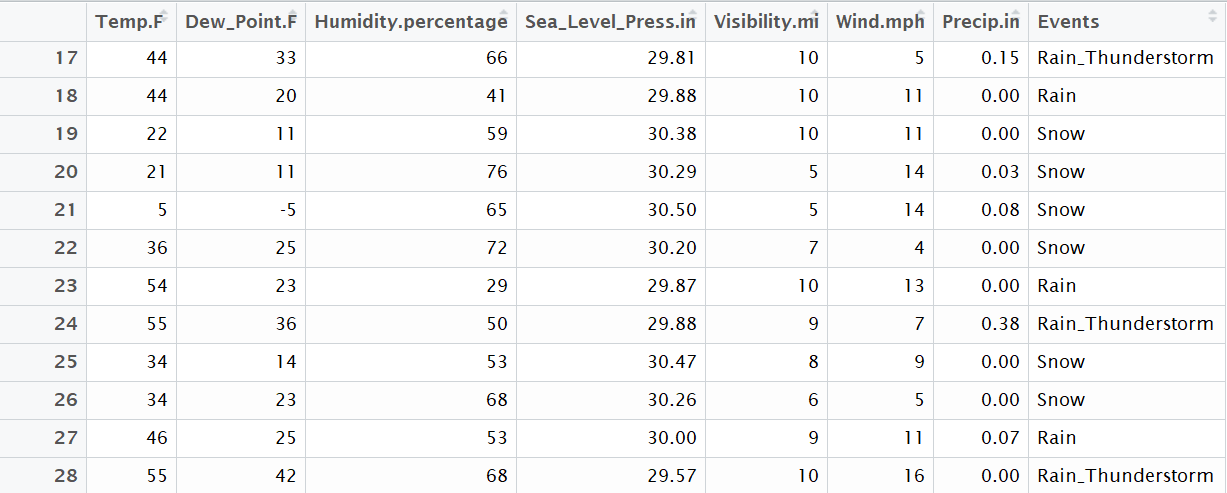


*#import the dataset and make some changes*

library(readr)

kc\_weather\_srt <- read\_csv("C:/Users/bvkka/Desktop/ISL-Deep Medhi/kc\_weather\_srt.csv")

kc\_weather\_srt=kc\_weather\_srt[,2:9]



*#first make the response column to 0-snow, 1-rain and 2-rain\_thunderstorm*

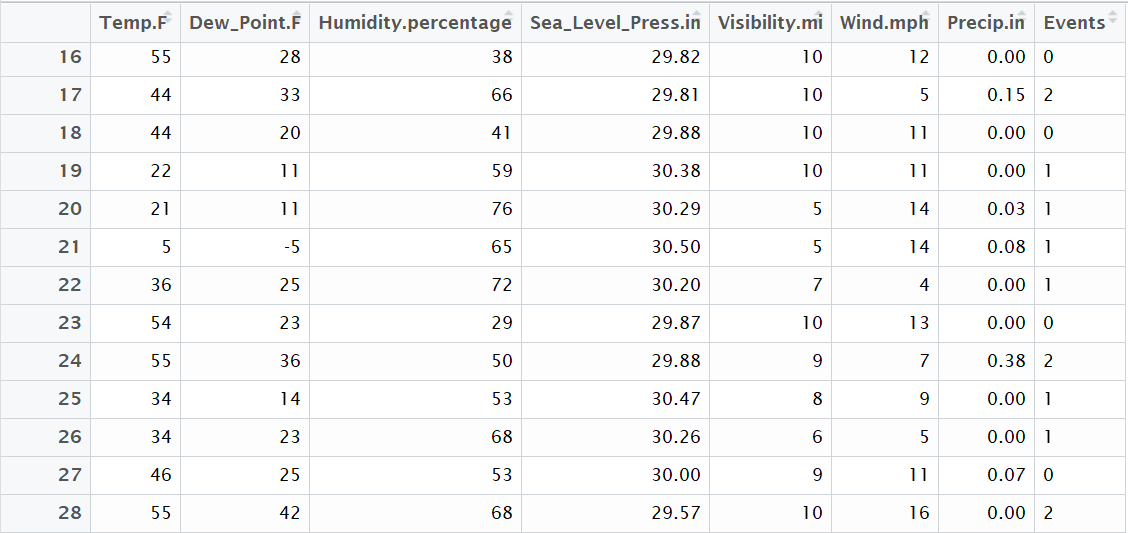
#install.packages("plyr")

library(plyr)

kc\_weather\_srt$Events <- revalue(kc\_weather\_srt$Events,c("Snow"=1))

kc\_weather\_srt$Events <- revalue(kc\_weather\_srt$Events,c("Rain"=0))

kc\_weather\_srt$Events <- revalue(kc\_weather\_srt$Events,c("Rain\_Thunderstorm"=2))



*#small changes to Events column , making it to numeric from character*

kc\_weather\_srt$Events<-as.numeric(as.character(kc\_weather\_srt$Events))

*#replications*

rep=100

*# newly added*

*#snow=1 rain=0 thunderstorm=2*

accuracy1=dim(rep)

accuracy2=dim(rep)

accuracy3=dim(rep)

precision\_snow1=dim(rep)

precision\_rain1=dim(rep)

precision\_rainThunderstorm1=dim(rep)

recall\_snow1=dim(rep)

recall\_rain1=dim(rep)

recall\_rainThunderstorm1=dim(rep)

precision\_snow2=dim(rep)

precision\_rain2=dim(rep)

precision\_rainThunderstorm2=dim(rep)

recall\_snow2=dim(rep)

recall\_rain2=dim(rep)

recall\_rainThunderstorm2=dim(rep)

precision\_snow3=dim(rep)

precision\_rain3=dim(rep)

precision\_rainThunderstorm3=dim(rep)

recall\_snow3=dim(rep)

recall\_rain3=dim(rep)

recall\_rainThunderstorm3=dim(rep)

*#splitting the dataset into training and test sets, also install caTools packages*

install.packages('caTools')

library(caTools)

set.seed(123)

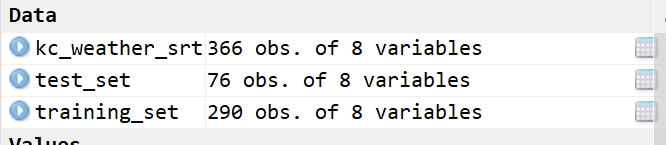
for(k in 1:rep)

{

split=sample.split(kc\_weather\_srt$Events,SplitRatio = 0.7923)

training\_set=subset(kc\_weather\_srt,split==TRUE)

test\_set=subset(kc\_weather\_srt,split==FALSE)



*#\*\*\*LDA\*\*\*#*

*#install.packages("MASS")*

library(MASS)

lda=lda(formula=Events~.,data=training\_set)

training\_set1=as.data.frame(predict(lda,training\_set))

training\_set1=training\_set1[c(4,1)]

plot(lda)

test\_set1=as.data.frame(predict(lda,test\_set))

test\_set1=test\_set1[c(4,1)]

*#fitting SVM to the training set*

*#install.packages('e1071')*

library(e1071)

classifier=svm(formula=class~.,data=training\_set1,type='C-classification',kernel='linear')

*#predicting the test set results*

y\_pred1=predict(classifier,newdata = test\_set1[-2])

*#making the confusion matrix*

cm1=table(y\_pred1,test\_set1[,2])

accuracy1[k]=mean(y\_pred1==test\_set1[,2])

precision1=precision1<-diag(cm1)/colSums(cm1)

precision\_rainThunderstorm1[k]=precision1[3]

precision\_snow1[k]=precision1[2]

precision\_rain1[k]=precision1[1]

recall1=recall1<-diag(cm1/rowSums(cm1))

recall\_rainThunderstorm1[k]=recall1[3]

recall\_snow1[k]=recall1[2]

recall\_rain1[k]=recall1[1]

*#\*\*\*QDA\*\*\*#*

qda=qda(formula=Events~.,data=training\_set)

y\_pred2=predict(qda,test\_set)$class

cm2=table(y\_pred2,test\_set[,8])

accuracy2[k]=mean(y\_pred2==test\_set[,8])

precision2=precison2<-diag(cm2/colSums(cm2))

precision\_rainThunderstorm2[k]=precison2[3]

precision\_snow2[k]=precison2[2]

precision\_rain2[k]=precision2[1]

recall2=recall2<-diag(cm2/rowSums(cm2))

recall\_rainThunderstorm2[k]=recall2[3]

recall\_snow2[k]=recall2[2]

recall\_rain2[k]=recall2[1]

*#\*\*\*KNN\*\*\*#*

*#install.packages('class')*

*#fitting KNN to the training set and predicting the test set results*

library(class)

y\_pred3=knn(train=training\_set[,-8],test=test\_set[,-8],cl=training\_set[,8],k=5)

*#making the cm*

cm3=table(y\_pred3,test\_set[,8])

accuracy3[k]=mean(y\_pred3==test\_set[,8])

precision3=precision3<-diag(cm3/colSums(cm3))

precision\_rainThunderstorm3[k]=precision3[3]

precision\_snow3[k]=precision3[2]

precision\_rain3[k]=precision3[1]

recall3=recall3<-recall3<-diag(cm3/rowSums(cm3))

recall\_rainThunderstorm3[k]=recall3[3]

recall\_snow3[k]=recall3[2]

recall\_rain3[k]=recall3[1]

}

*###\*\*\*\*\*\*\*MEAN VALUES\*\*\*\*\*#####*

*#\*\*\*\*LDA\*\*\*#*

mean(accuracy1)*###0.9026316*

mean(precision\_snow1)*###0.6407459*

mean(precision\_rain1)*###0.9115871*

mean(precision\_rainThunderstorm1)*###0.9906168*

mean(recall\_snow1) *### 0.911675*

mean(recall\_rain1) *### 0.902705*

mean(recall\_rainThunderstorm1) *### 0.9906152*

*#\*\*\*\*QDA\*\*\*#*

mean(accuracy2)*###0.7389474*

mean(precision\_snow2)*###0.945*

mean(precision\_rain2)*###0.697027*

mean(precision\_rainThunderstorm2)*###0.7213793*

mean(recall\_snow2) *### 0.7950919*

mean(recall\_rain2) *### 0.7514844*

mean(recall\_rainThunderstorm2) *### 0.7115056*

*#\*\*\*\*KNN\*\*\*#*

mean(accuracy3)*###0.745*

mean(precision\_snow3)*###0.895*

mean(precision\_rain3)*###0.7305405*

mean(precision\_rainThunderstorm3)*###0.7117241*

mean(recall\_snow3) *### 0.9098042*

mean(recall\_rain3) *### 0.7444542*

mean(recall\_rainThunderstorm3) *### 0.701065*

**Summary:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Model | Accuracy | Precision Snow | Precision Rain | Precision Rain Thunderstorm | Recall Snow | Recall Rain | Recall ThunderStorm |
| LDA | 0.9026316 | 0.6407459 | 0.9115871 | 0.9906168 | 0.911675 | 0.902705 | 0.9906152 |
| QDA | 0.7389474 | 0.945 | 0.697027 | 0.7213793 | 0.7950919 | 0.7514844 | 0.7115056 |
| KNN (K=5) | 0.745 | 0.895 | 0.7305405 | 0.7117241 | 0.9098042 | 0.7444542 | 0.701065 |

**Text Summarization:**

*#As you can see accuracy of LDA>KNN>QDA , we should opt for LDA*

*#As you can see Precision of Snow , LDA gives highest*

*#As you can see Precision of rain , LDA gives highest, outperforms KNN and QDA*

*#As you can see Precision of rainthunderstorm , LDA gives highest, outperforms KNN and QDA*

*#As you can see Recall of Snow , LDA gives highest*

*#As you can see Recall of rain , LDA gives highest, outperforms KNN and QDA*

*#As you can see Recall of rainthunderstorm , LDA gives highest, outperforms KNN and QDA*

*###\*\*\*So based on results, we choose LDA model on this dataset\*\*\*\*###*