1 CODE

```
#EDA
#load all libraries
library(dummies)
library(corrplot)
library(DataExplorer)

#upload the dataset
data <- read.csv(file.choose())

# grouping output variable
for (i in 1:395){
  if (data$G3[i]<5){
    data$G3[i] = 5
  }else{
    if (data$G3[i]<9){
      data$G3[i] = 4
    }else{
      if (data$G3[i]<13){
        data$G3[i] = 3
      }else{
        if (data$G3[i]<17){
          data$G3[i] = 2
        }else{
          if (data$G3[i]<21){
            data$G3[i] = 1
          }
        }
      }
    }
  }
}

# exploring data set
plot_str(data)
plot_missing(data)

## continuous data
plot_histogram(data)
plot_density(data)
plot_correlation(data, type = "continuous")

## categorical data
plot_bar(data)
plot_correlation(data, type = "all")
plot_correlation(data[,-33], type = "discrete")

# converting categorical data into numeric data
new_data <- dummy.data.frame(data, names = c("school", "sex", "
    address", "famsize", "Pstatus","Mjob", "Fjob",  "reason", "
    guardian", "schoolsup", "famsup", "paid", "activities", "
    nursery", "higher", "internet", "romantic"))
new_data <- new_data[,-c(2,4,7,9,11,18,23,27,30,35,37,39,
                         41,43,45,47,49, 59)]
# ensuring we have k-1 dummy variables to avoid multicollinearity

```

```r
# correlation plot
library(corrplot)
corrplot(cor(new_data), type = "upper")

# finding training set
set.seed(2020)
train <- sample(1:395,0.70*395)
table(data[,33])
table(data[train,33])

# checking for outliers in training set
Sx <- cov(new_data[train,])
D2 <- mahalanobis(new_data[train,], colMeans(new_data[train,]), Sx)

plot(density(D2, bw = 0.5),
     main="Squared Mahalanobis distances") ; rug(D2)
qqplot(qchisq(ppoints(395), df = 40), D2,
       main = expression("Q-Q plot of Mahalanobis" * ~D^2 *
                           " vs. quantiles of" * ~ chi[3]^2))
abline(0, 1, col = 'gray')

library(ggplot2)
df <- data.frame(Index = 1:276, y = D2)
ggplot(df, aes(Index,sqrt(y))) + geom_point(colour = "black") + labs(y = "Squared Mahalanobis Distances")

# converting classes in target variable into categorical variable
data$G3 <- as.factor(data$G3)
```

Listing 1: EDA code

```r
#loading libraries
library(caret)
library(class)
library(ggplot2)
library(scales)
library(dummies)

# read data into r
data <- read.csv(file.choose())

# grouping output variable
for (i in 1:395){
  if (data$G3[i]<5){
    data$G3[i] = 5
  }else{
    if (data$G3[i]<9){
      data$G3[i] = 4
    }else{
      if (data$G3[i]<13){
        data$G3[i] = 3
      }else{
        if (data$G3[i]<17){
          data$G3[i] = 2
        }else{
          if (data$G3[i]<21){
            data$G3[i] = 1
          }
```

```
        }
      }
    }
  }
}
data$G3 <- as.factor(data$G3)

# converting categorical data into numeric data
new_data <- dummy.data.frame(data, names = c("school", "sex", "
    address", "famsize", "Pstatus","Mjob", "Fjob","reason", "
    guardian", "schoolsup", "famsup", "paid", "activities","nursery
    ", "higher",
"internet", "romantic"))

# dividing new data
set.seed(2020)
train <- sample(1:395,0.7*395)
pca.train <- new_data[train,-c(2,4,7,9,11,18,23,27,30,
                               35,37,39,41,43,45,47,49,59)]
# ensuring we have k-1 dummy variables to avoid collinearity
pca.test <- new_data[-train,-c(2,4,7,9,11,18,23,27,30,
                              35,37,39, 41,43,45,47,49,59)]

# principal component analysis
pca2 <- prcomp(pca.train, scale=T, center=T)
pca1 <- prcomp(pca.train, scale=F, center=T)
pca <- prcomp(pca.train, scale=F, center=F)

# plotting resultatnt principal components
biplot(pca2, scale=0)
biplot(pca1, scale=0)
biplot(pca, scale=0)

# standard deviation of each principal component
std_dev2 <- pca2$sdev
std_dev1 <- pca1$sdev
std_dev <- pca$sdev

# compute variance
pca_var2 <- std_dev2^2
pca_var1 <- std_dev1^2
pca_var <- std_dev^2

# proportion of variance explained by each component
prop_var2 <- pca_var2/sum(pca_var2)
prop_var1 <- pca_var1/sum(pca_var1)
prop_var <- pca_var/sum(pca_var)

# scree plot
plot(prop_var[1:5], xlab="Principal Component",
     ylab="Proportion of Variance Explained",
     type="b", col="red")
lines(prop_var2[1:5], col="blue", type="b")
lines(prop_var1[1:5], col="green", type="b")

# plotting pca
dataset = data.frame(G3 = data[train,"G3"],
```

```
                       pca = pca$x)

ggplot(dataset) + geom_point(aes(pca.PC1, pca.PC2, colour = G3,
    shape = G3), size = 2.5) +
  labs(x = paste("First Component (", percent(prop_var[1]), ")",
    sep = ""),
       y = paste("Second Component (", percent(prop_var[2]), ")",
    sep = ""))

# predictive modelling with PCA components
train_data <- data.frame(G3 = new_data$G3[train], pca$x)
train_data <- train_data[,1:3]
test_data <- predict(pca, newdata = pca.test)
test_data <- as.data.frame(test_data)
test_data <- test_data[,1:2]

#predicting using k-nearest neighbours
prediction <- knn(train_data[,2:3], test_data, train_data[,1], k=5)

#evaluating performance
confusionMatrix(prediction,new_data$G3[-train])
```

Listing 2: PCA code

```
# installing pcaMethods
if (!requireNamespace("BiocManager", quietly = TRUE))
  install.packages("BiocManager")
BiocManager::install("pcaMethods")

#load packages
library(pcaMethods)
library(tidyverse)
library(caret)
library(class)

# read data into r
data <- read.csv(file.choose())

# grouping output variable
for (i in 1:395){
  if (data$G3[i]<5){
    data$G3[i] = 5
  }else{
    if (data$G3[i]<9){
      data$G3[i] = 4
    }else{
      if (data$G3[i]<13){
        data$G3[i] = 3
      }else{
        if (data$G3[i]<17){
          data$G3[i] = 2
        }else{
          if (data$G3[i]<21){
            data$G3[i] = 1
          }
        }
      }
    }
```

```
  }
}

data$G3 <- as.factor(data$G3)

# converting categorical data into numeric data
new_data <- dummy.data.frame(data, names = c("school", "sex", "address", "famsize", "Pstatus","Mjob", "Fjob", "reason","guardian", "schoolsup", "famsup", "paid", "activities", "nursery", "higher", "internet", "romantic"))

# dividing new data
set.seed(2020)
train <- sample(1:395,0.7*395)
rpca.train <- new_data[train,-c(2,4,7,9,11,18,23,27,30,
                                35,37,39,41,43,45,47,49,59)]
# ensuring we have k-1 dummy variables to avoid collinearity
rpca.test <- new_data[-train,-c(2,4,7,9,11,18,23,27,30,
                                35,37,39,41,43,45,47,49,59)]


rpca <- pca(rpca.train, nPcs = 10, method = "robustPca", center = FALSE)  # unstandardized data
variation <- rpca@R2[1] + rpca@R2[3]


# plotting unstandardized components
require(ggplot2)
require(scales)
dataset = data.frame(G3 = data[train,"G3"],
                     rpca = rpca@scores)

ggplot(dataset) + geom_point(aes(rpca.PC1, rpca.PC3, colour = G3, shape = G3), size = 2.5) +
  labs(x = paste("First Principal Component (", percent(rpca@R2[1]), ")", sep = ""),
       y = paste("Third Principal Component (", percent(rpca@R2[3]), ")", sep = ""))

# predictive modelling with PCA components
train_data <- data.frame(G3 = new_data$G3[train], rpca@scores)
train_data <- train_data[,c(1,2,4)]
test_data <- predict(rpca, newdata = rpca.test)
test_data <- as.data.frame(test_data$scores)
test_data <- test_data[,c(1,3)]

#predicting using k-nearest neigbours
prediction <- knn(train_data[,2:3], test_data, train_data[,1], k=5)

#evaluating performance
confusionMatrix(prediction,new_data$G3[-train])
```

Listing 3: RPCA code

```
#load packages
library(pcaMethods)
library(class)
library(caret)
```

```
# read data into r
data <- read.csv(file.choose())

# grouping output variable
for (i in 1:395){
  if (data$G3[i]<5){
    data$G3[i] = 5
  }else{
    if (data$G3[i]<9){
      data$G3[i] = 4
    }else{
      if (data$G3[i]<13){
        data$G3[i] = 3
      }else{
        if (data$G3[i]<17){
          data$G3[i] = 2
        }else{
          if (data$G3[i]<21){
            data$G3[i] = 1
          }
        }
      }
    }
  }
}

data$G3 <- as.factor(data$G3)

# converting categorical data into numeric data
new_data <- dummy.data.frame(data, names = c("school", "sex", "
    address", "famsize", "Pstatus","Mjob", "Fjob", "reason", "
    guardian", "schoolsup", "famsup", "paid", "activities", "
    nursery", "higher", "internet", "romantic"))

# ensuring we have k-1 dummy variables to avoid collinearity
new_data <- new_data[, -c(2,4,7,9,11,18,23,27,30,
                          35,37,39,41,43,45,47,49)]

# creating missing values
set.seed(2020)
cols <- sample(1:41, 39, replace = T)
rows <- sample(1:395, 39)
miss <- cbind(rows,cols)
new_data[miss] <- NA


# dividing new data
set.seed(2020)
train <- sample(1:395,0.7*395)
ppca.train <- new_data[train,-42]
ppca.test <- new_data[-train,-42]


# conducting ppca
ppca <- pca(as.matrix(ppca.train), method = "ppca", nPcs = 2) #
    unstandardized data
```

```
ppca@R2cum

# plotting unstandardized components
require(ggplot2)
require(scales)
dataset = data.frame(G3 = data[train,"G3"],
                     ppca = ppca@scores)

ggplot(dataset) + geom_point(aes(ppca.PC1, ppca.PC2, colour = G3, shape = G3), size = 2.5) +
  labs(x = paste("First Principal Component (", percent(ppca@R2[1]), ")", sep = ""),
       y = paste("Second Principal Component (", percent(ppca@R2[2]), ")", sep = ""))


# predictive modelling with PCA components
train_data <- data.frame(G3 = new_data$G3[train], ppca@scores)
test_data <- predict(ppca, newdata = ppca.test)
test_data <- as.data.frame(test_data$scores)

#predicting using k-nearest neigbours
prediction <- knn(train_data[,2:3], test_data, train_data[,1], k=5)

#evaluating performance
confusionMatrix(prediction,new_data$G3[-train])
```

Listing 4: PPCA code

```
#load packages
library(kernlab)
library(dummies)

# read data into r
data <- read.csv(file.choose())

# grouping output variable
for (i in 1:395){
  if (data$G3[i]<5){
    data$G3[i] = 5
  }else{
    if (data$G3[i]<9){
      data$G3[i] = 4
    }else{
      if (data$G3[i]<13){
        data$G3[i] = 3
      }else{
        if (data$G3[i]<17){
          data$G3[i] = 2
        }else{
          if (data$G3[i]<21){
            data$G3[i] = 1
          }
        }
      }
    }
  }
}
```

```r

data$G3 <- as.factor(data$G3)

# converting categorical data into numeric data
new_data <- dummy.data.frame(data, names = c("school", "sex", "
    address", "famsize", "Pstatus","Mjob", "Fjob","reason", "
    guardian", "schoolsup", "famsup", "paid", "activities", "
    nursery", "higher", "internet", "romantic"))

# dividing new data
set.seed(2020)
train <- sample(1:395,0.7*395)
kpca.train <- new_data[train,-c(2,4,7,9,11,18,23,27,30,
                                35,37,39,41,43,45,47,49,59)]
# ensuring we have k-1 dummy variables to avoid collinearity
kpca.test <- new_data[-train,-c(2,4,7,9,11,18,23,27,30,
                                35,37,39,41,43,45,47,49,59)]

# kernel principal component analysis
kpca <- kpca(as.matrix(kpca.train), kernel = "rbfdot", kpar = list(
    sigma =c(100, 10, 1, 0.1, 0.001, 0.0001, 0.00001, 0.0000001)))
kpca@eig

# plotting kpca
dataset = data.frame(G3 = data[train,"G3"],
                     kpca = kpca@pcv)
require(ggplot2)
require(scales)

prop_var <- kpca@eig/sum(kpca@eig)
sum(prop_var[1:2])

ggplot(dataset) + geom_point(aes(kpca.1, kpca.2, colour = G3, shape
     = G3), size = 2.5) +
  labs(x = paste("First Component (", percent(prop_var[1]), ")",
    sep = ""),
       y = paste("Second Component (", percent(prop_var[2]), ")",
    sep = ""))
```

Listing 5: KPCA code

```r
#load libraries
library(kernlab)
library(caret)
library(dummies)
library(class)

# read data into r
data <- read.csv(file.choose())

# grouping output variable
for (i in 1:395){
  if (data$G3[i]<5){
    data$G3[i] = 5
  }else{
    if (data$G3[i]<9){
      data$G3[i] = 4
    }else{
```

```
      if (data$G3[i]<13){
        data$G3[i] = 3
      }else{
        if (data$G3[i]<17){
          data$G3[i] = 2
        }else{
          if (data$G3[i]<21){
            data$G3[i] = 1
          }
        }
      }
    }
  }
}
data$G3 <- as.factor(data$G3)

# converting categorical data into numeric data
new_data <- dummy.data.frame(data, names = c("school", "sex", "
    address", "famsize", "Pstatus","Mjob", "Fjob", "reason","
    guardian", "schoolsup", "famsup", "paid", "activities",  "
    nursery", "higher", "internet", "romantic"))

# dividing new data
set.seed(2020)
train <- sample(1:395,0.7*395)
kpca.data <- new_data[,-c(2,4,7,9,11,18,23,27,30,
                          35,37,39,41,43,45,47,49,59)]
# ensuring we have k-1 dummy variables to avoid collinearity

# kernel principal component analysis
kpca <- kpca(as.matrix(kpca.data), kernel = "rbfdot", kpar = list(
    sigma =c(100, 10, 1, 0.1, 0.001, 0.0001, 0.00001, 0.0000001)))
kpca@eig

# predictive modelling with PCA components
Z <- kpca@rotated[,1:2]
train_data <- as.data.frame(Z[train,])
test_data <- as.data.frame(Z[-train,])

#using knn for predicting
prediction <- knn(train_data, test_data, new_data$G3[train], k=5)

#evaluating performance
confusionMatrix(prediction,new_data$G3[-train])
```

Listing 6: KPCA predicting code