Untitled

2023-09-12

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

summary(cars)

## speed dist   
## Min. : 4.0 Min. : 2.00   
## 1st Qu.:12.0 1st Qu.: 26.00   
## Median :15.0 Median : 36.00   
## Mean :15.4 Mean : 42.98   
## 3rd Qu.:19.0 3rd Qu.: 56.00   
## Max. :25.0 Max. :120.00

## Including Plots

You can also embed plots, for example:



Note that the echo = FALSE parameter was added to the code chunk to prevent printing of the R code that generated the plot. ### Data Import and Cleaning

library(class) library(caret) library(e1071) ub.df <- read.csv(“C:/Users/samyu/Downloads/UniversalBank.csv”) dim(ub.df) t(t(names(ub.df)))# transposes the columns as rows # removing unwanted variables ub.df = ub.df[,-c(1,5)] ub.df # categorical variables ub.dfEducation) # dummy variables groups = dummyVars(~. , data = ub.df) # this creates dummy groups # converting factor to a dummy variable ub\_m.df = as.data.frame(predict(groups,ub.df)) # splitting the data # to ensure that we get the same sample if we return the code

set.seed(1) train.index = sample(row.names(ub\_m.df),0.6\*dim(ub\_m.df)[1]) valid.index = setdiff(row.names(ub\_m.df),train.index) train\_df = ub\_m.df[train.index ,] valid\_df = ub\_m.df[valid.index ,]

#normalise the data train.norm.df = train\_df[,-10] valid.norm.df = valid\_df[,-10]

norm.values = preProcess(train\_df[,-10], method = c(“center”, “scale”)) train.norm.df = predict(norm.values, train\_df[,-10]) valid.norm.df = predict(norm.values, valid\_df[-10])

new\_customer = data.frame() new.cust.norm = new\_customer new.cust.norm = predict(norm.values, new.cust.norm)

# lets predict using knn

knn.predict = class::knn(train = train.norm.df, test = new.cust.norm, c1 =train.df$Personal.Loan, k=1) knn.pred

#calculate the accuracy for each value of k accuracy.df = data.frame(k= 1,15,1), overallaccuracy = rep(0,15)) for(i in 1:15) { knn.pred = class::knn(train = train.norm.df, test = new.cust.norm, c1 =train.df$Personal.Loan, k=i) accuracy.df[i,2] = confusionMatrix(knn.pred,

as.factor(valid.df$Personal.Loan) }