** Logo

Description automatically generated**

**COMPUTER SCIENCE AND DATA ANALYTICS**

Course: Intro to Big Data Analytics

# Project 2

Project title: ***“Concrete Compressive Strength Analysis”***

Student(s):David Ahmadov

Instructor: **Dr. Abzatdin Adamov**

## Baku 2021

1. **For this data set, plot the data using pairwise plotting** to get a sense of the relationships between the attributes.

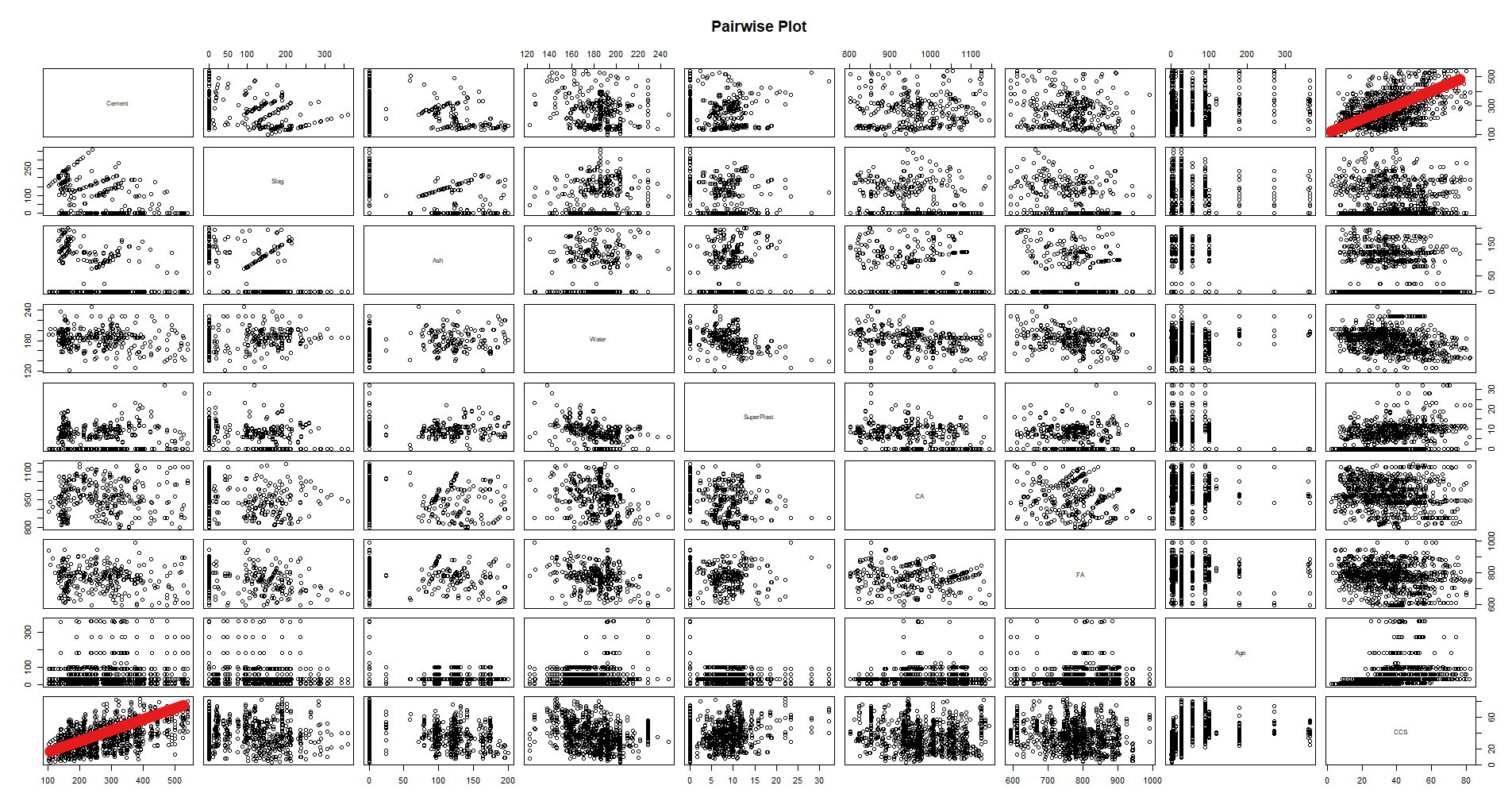
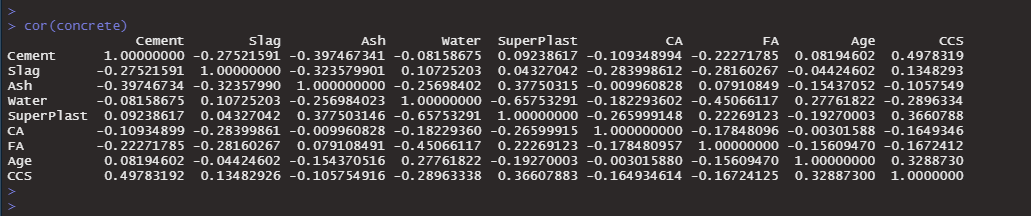


Fig1. Pairwise plot of dataset

From first glance only linear relation I can observe is between Cement and CCS. Otherwise, it is hard to tell if there are more linear relationships between other predictor variables and observed (CCS) variable. In order to explore more, we can look into correlation matrix and plot.



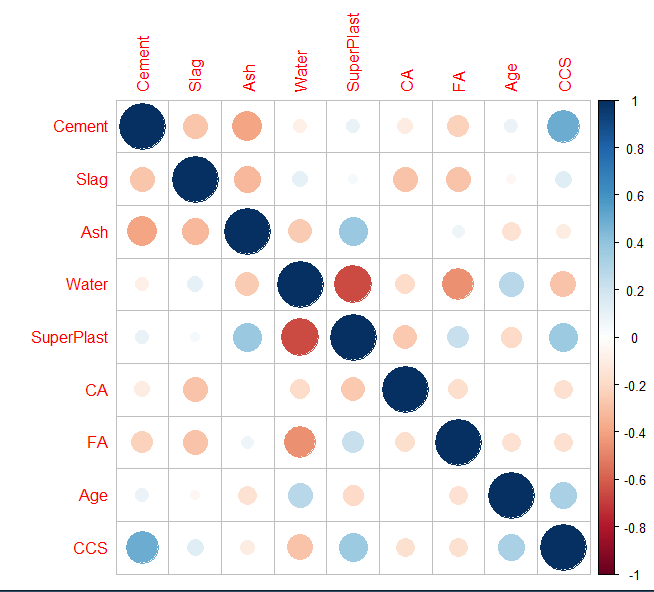


Fig2. Correlation Plot

a.There is some positive linear correlation between Cement and CCS

b.There is a weak inverse linear relationship between Water and CCS

c. There is some positive linear correlation between Superplast and CCS

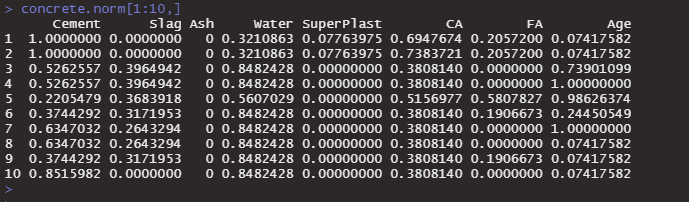
d. There is some positive linear correlation between Age and CCS

e.There is some significant inverse relationship between Water and Superplast

1. **Scaling and Normalization**

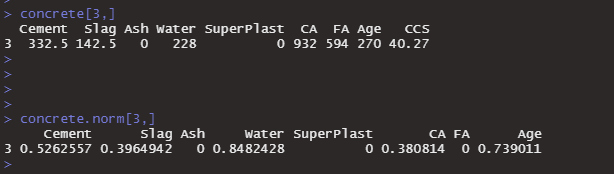
normalize <- function(x) { ((x-min(x)) / (max(x) - min(x))) }

concrete.norm <- as.data.frame(lapply(concrete.df,normalize))



Test going back from normal to real value.

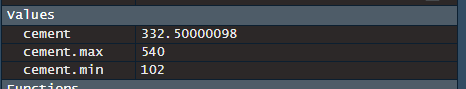
x = norm(x) \* (max(x)-min(x) ) + min(x)



cement.min <- min(concrete.df$Cement)

cement.max <- max(concrete.df$Cement)

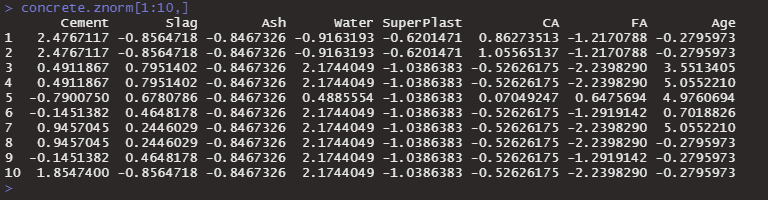
cement <- 0.52625571 \* (cement.max - cement.min) + cement.min



Zscore Normalization

zscore <- function(x) { (x - mean(x))/ sd(x) }

concrete.znorm <- as.data.frame(lapply(concrete.df,scale))



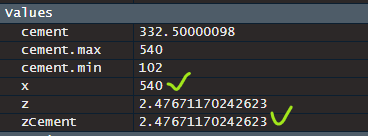
Test going back from Zscore to original value

zscore = (x - mean(x))/sd(x)

x = zscore \* sd(x) + mean(x)

zCement <- concrete.znorm[1,1]

x <- zCement \* sd(concrete.df$Cement) + mean(concrete.df$Cement)



Cement, Water, Superplasticizer and Age attributes seem to contribute most of the CCS value. But we can that Water and Superplast are inversely correlated. When we have attributes that correlated between each other we have to address this. Having correlated attributes can introduce problems with accuracy of model.

1. **Clustering**

In order to proceed with clustering, we need to normalize our data.

concrete.df.norm <- as.data.frame(lapply(concrete,normalize))

concrete.df.znorm <- as.data.frame(lapply(concrete,scale))

wssplot <- function(data, nc=15,seed=1234) {

wss <- (nrow(data)-1)\* sum(apply(data,2,var))

for(i in 2:nc) {

set.seed(seed)

wss[i] <- sum(kmeans(data,centers = i)$withinss)

}

plot(1:nc,wss,type="b",xlab = "Number of clusters", ylab = "Within groups sum of squares")

}

wssplot(concrete.df.norm,nc=10,seed=12324)

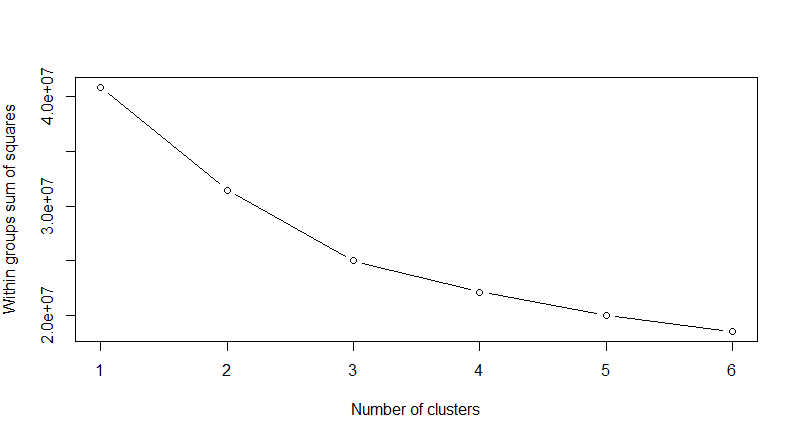


Fig3. Using wss plot to try to find an elbow for best cluster number.

factoextra::fviz\_nbclust(concrete.df, FUNcluster=kmeans,print.summary=TRUE)

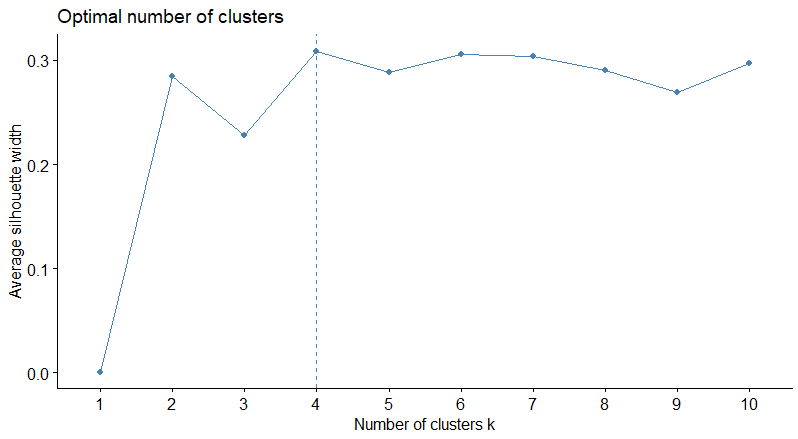


Fig4. Estimate of optimal number of clusters.

Generate clusters n : 2-6

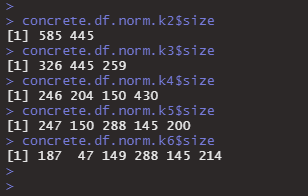
concrete.df.norm.k2 = kmeans(concrete.df.norm,centers=2,nstart=25)

concrete.df.norm.k3 = kmeans(concrete.df.norm,centers=3,nstart=25)

concrete.df.norm.k4 = kmeans(concrete.df.norm,centers=4,nstart=25)

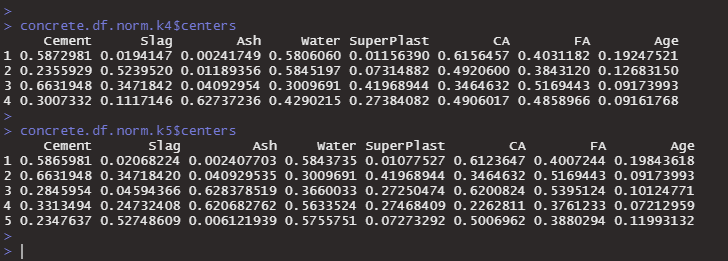
concrete.df.norm.k5 = kmeans(concrete.df.norm,centers=5,nstart=25)

concrete.df.norm.k6 = kmeans(concrete.df.norm,centers=6,nstart=25)



I choose k = 4.

Examine Centroids



concrete.df.norm.k2 = kmeans(concrete.df.norm,centers=2,nstart=25)

concrete.df.norm.k3 = kmeans(concrete.df.norm,centers=3,nstart=25)

concrete.df.norm.k4 = kmeans(concrete.df.norm,centers=4,nstart=25)

concrete.df.norm.k5 = kmeans(concrete.df.norm,centers=5,nstart=25)

concrete.df.norm.k6 = kmeans(concrete.df.norm,centers=6,nstart=25)

# Plot k2

factoextra::fviz\_cluster(concrete.df.norm.k2,concrete.df.norm)

# Plot k3

factoextra::fviz\_cluster(concrete.df.norm.k3,concrete.df.norm)

# Plot k4

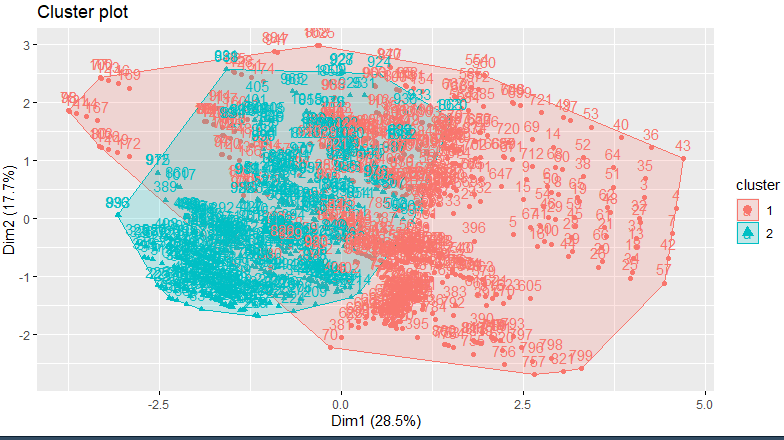
factoextra::fviz\_cluster(concrete.df.norm.k4,concrete.df.norm)

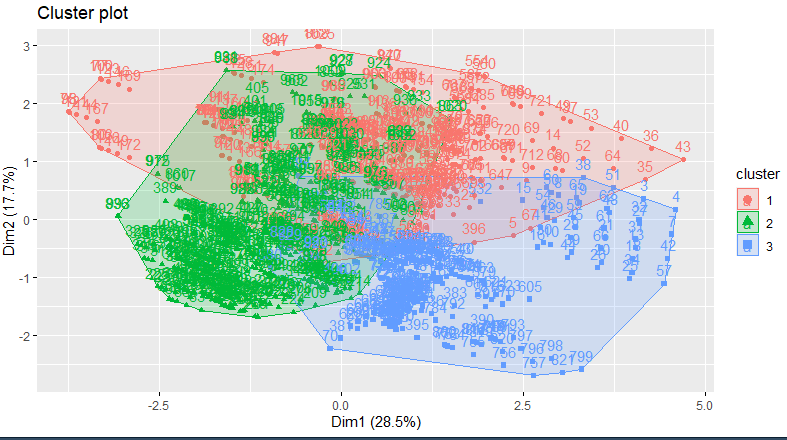
# Plot k5

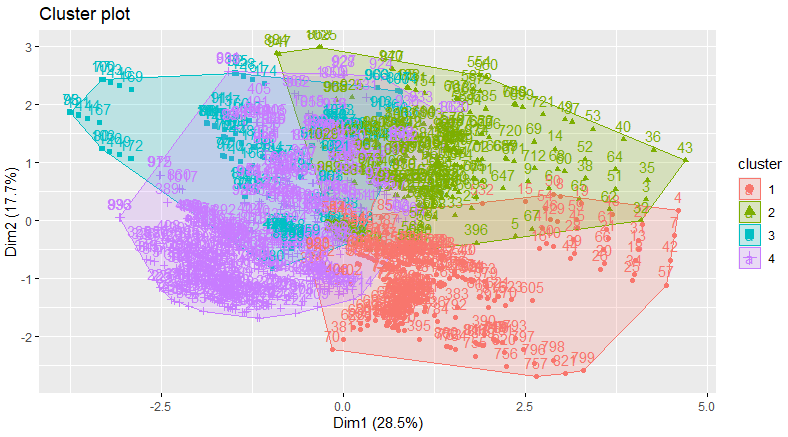
factoextra::fviz\_cluster(concrete.df.norm.k5,concrete.df.norm)

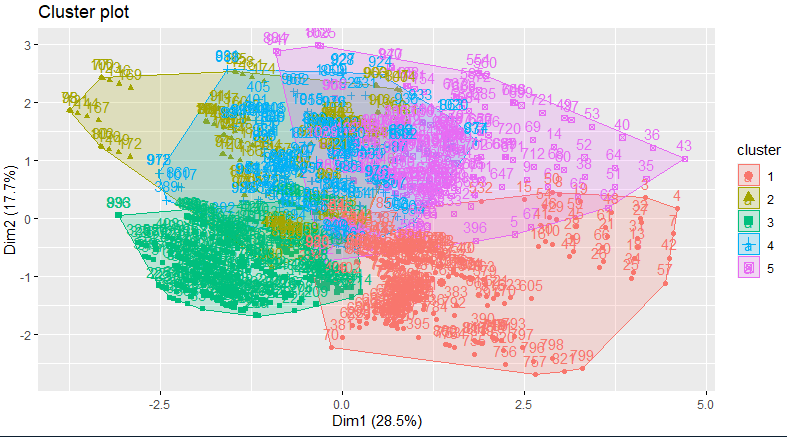
# Plot k6

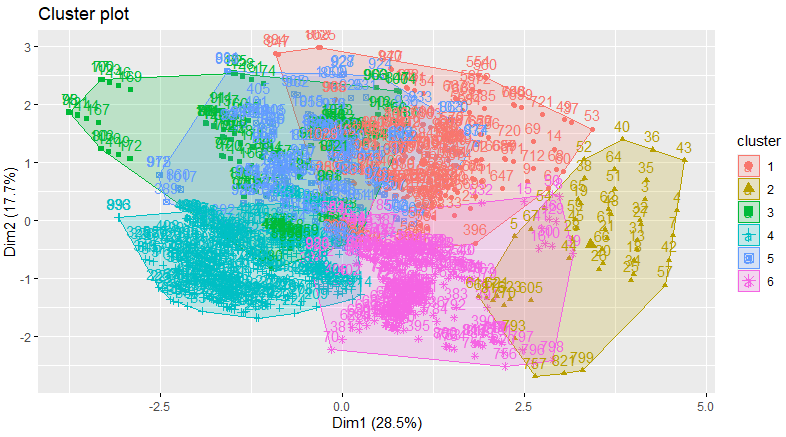
factoextra::fviz\_cluster(concrete.df.norm.k6,concrete.df.norm)











KNN

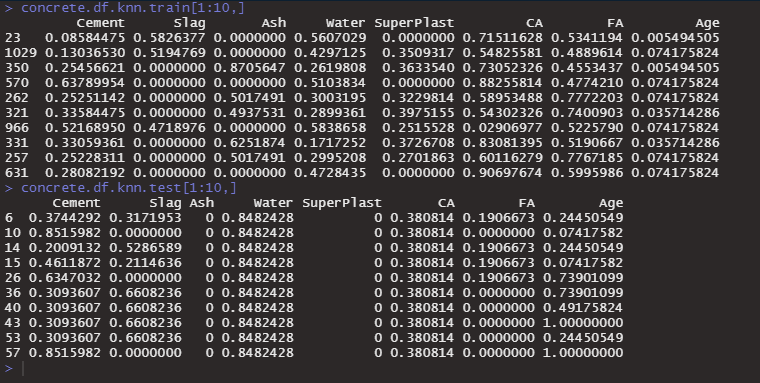
concrete.df.knn.nrows <- nrow(concrete.df.norm)

concrete.df.knn.sample <- 0.7

concrete.df.knn.train.index <- sample(concrete.df.knn.nrows,concrete.df.knn.sample\*concrete.df.knn.nrows)

concrete.df.knn.train <- concrete.df.norm[concrete.df.knn.train.index,]

concrete.df.knn.test <- concrete.df.norm[-concrete.df.knn.train.index,]



Create training labels for k4

concrete.df.knn.train.k4 <- kmeans(concrete.df.knn.train,centers = 4)

concrete.df.knn.test.k4 <- knn(concrete.df.knn.train,concrete.df.knn.test, concrete.df.knn.train.k4$cluster, k = 4)

# Generate Labels

concrete.df.knn.train.labels <- concrete.df.knn.train.k4$cluster

# Create test labels via kmeans

concrete.df.knn.test.k4 <- kmeans(concrete.df.knn.test, centers = 4)

concrete.df.knn.test.labels <- concrete.df.knn.test.k4$cluster

Evaluating kNN

concrete.df.knn.predict <- knn(concrete.df.knn.train,concrete.df.knn.test,concrete.df.knn.train.k4$cluster, k=4)

concrete.df.knn.ct <- CrossTable(concrete.df.knn.test.labels, concrete.df.knn.predict,prop.chisq = FALSE)

