R Notebook

Code

Lectura de datos

Hide

```
library(readxl)
setwd("C:\\Users\\Adal\\Dropbox\\Magister Control de Gestión\\20181\\BI 20181\\Ayudantia 2")
titanic_train<-read_excel("Titanic_Train.xlsx",col_names = TRUE)
titanic_test<-read_excel("Titanic_Test.xlsx",col_names = TRUE)
```

Estructura de los datos

Hide

```
str(titanic_train)
```

Hide

```
str(titanic_test)
```

Analisis de datos - presencia de missing values

Como podemos ver, hay presencia de missing values tanto en la base de entrenamiento como en la de test.

Hide

```
pMiss<-function(x){
  sum(is.na(x))
}

apply(titanic_train,2,pMiss)
#Age, cabin and embarked with missing values

apply(titanic_test,2,pMiss)
#Age, fae and cabin with missing values
```

Analisis de datos - Estadistica descriptiva

Revisión descriptiva de la base utilizando la biblioteca de manejo de datos dplyr.

Hide

```
library(dplyr)

total_datos <- nrow(titanic_train)

titanic_train%>%
  group_by(Survived)%>%
  summarize(n = n(), percent = (n()/total_datos)*100)

titanic_train%>%
  group_by(Survived)%>%
  summarize(avg=mean(Age,na.rm = TRUE),desv = sd(Age,na.rm = TRUE))

titanic_train%>%
  filter(Survived == 1)%>%
  group_by(Pclass)%>%
  summarize(n = n(),percent = (n()/total_datos)*100)

titanic_train%>%
  filter(Survived == 1)%>%
  group_by(Sex)%>%
  summarize(n = n(),percent = (n()/total_datos)*100)

titanic_train%>%
  summarize(correlacion = cor(Survived,Age, use = "complete.obs"))

#install.packages("corrr")
#library(corrr)

titanic_train %>%
  select(Age, Pclass,Fare,Parch, Survived) %>%
  correlate() %>%
  focus(Survived)
```

Preprocesamiento de datos

Para preprocesar los datos, podemos encontrar missing values y outliers. En particular revisaremos como solucionar el problema de los missing values. En este caso, existen tres opciones
1) Eliminar todas las tuplas que contienen datos vacios na.omit()
2) Eliminar las variables (o columnas) completas con los datos vacios, usando subset()
3) Rellenar los datos faltantes. Para esto tenemos ciertas opciones:
3.1 Reemplazar por la media o moda, generaremos un vector para probar con age.
3.2 Reemplazar por la moda, no existe una función, por lo tanto hay que crearla, utilizaremos la variable Embarked.
3.3 Utilizaremos modelos estadisticos y de ML mas avanzados. Para esto utilizaremos la libreria mice

IMPORTANTE: DEBEMOS INTEGRAR TANTO LA BASE DE TEST Y LA DE ENTRENAMIENTO PARA IMPUTAR LOS DATOS. MISMO PROCEDIMIENTO SE DEBE HACER PARA LA TRANSFORMACION DE DATOS.

Hide

```
#UNION DE BASES PARA PROCESAMIENTO#
#Debemos añadir la variable de prediccion que nos falta en la base de test

titanic_test <- titanic_test%>%
                  mutate(Survived = NA)
titanic_test
titanic_full<-rbind(titanic_train,titanic_test)

titanic_full_preproces <- subset(titanic_full, select = -Survived )

#3.1#

attach(titanic_full_preproces)
titanic_full_preproces.Age<-titanic_full_preproces$Age
titanic_full_preproces.Age[is.na(titanic_full_preproces.Age)]<-mean(titanic_full_preproces.Ag
e, na.rm=TRUE)

#3.2#

titanic_train.Embarked<-titanic_train$Embarked
getmode<-function(v) {
  uniqv<-unique(v)
  uniqv[which.max(tabulate(match(v, uniqv)))]
}

titanic_train.Embarked[is.na(titanic_train.Embarked)]<-getmode(titanic_train.Embarked)

#3.2#

library(mice)

mod_mice<-mice(data = titanic_full_preproces, m = 5,meth='cart')

titanic_full_preproces <- complete(mod_mice)

titanic_full_preproces
```

Transformacion de los datos

En transformacion de datos tambien tenemos distintos elementos con los cuales podemos modificar la estructura de nuestros datos, en particular
1) Normalizacion de datos: nos sirve para tener una misma medida y comparar posteriormente impactos de las variables con respecto a nuestro label. Ademas en modelos de ML como redes neuronales tiene un mejor rendimiento.
2) Creación de variables sintéticas: a partir de la conjugacion logica entre dos variables (y,o) creamos una nueva variable. El beneficio es que contiene informacion de dos variables en una sola, reduciendo la dimensionalidad del modelo y mejorando el accuracy de esa variable. El problema es que se puede incurrir en un sobreajuste del modelo, ya que pierde generalidad.
3) Variables categoricas a variables dummy: Para modelos matemáticos que utilizan solo variables numericas utilizamos la transformacion de categorias a 1 y 0.

Hide

```
#1#
#1.1
scale(titanic_train_dummy$Age)

#1.2
library(clusterSim)
data.Normalization(titanic_train_dummy$Age,type = "n3",normalization = "column")
  #n0 - without normalization
  #n1 - standardization ((x-mean)/sd)
  #n2 - positional standardization ((x-median)/mad)
  #n3 - unitization ((x-mean)/range)
  #n3a - positional unitization ((x-median)/range)
  #n4 - unitization with zero minimum ((x-min)/range)
  #n5 - normalization in range <-1,1> ((x-mean)/max(abs(x-mean)))
  #n5a - positional normalization in range <-1,1> ((x-median)/max(abs(x-median)))
  #n6 - quotient transformation (x/sd)
  #n6a - positional quotient transformation (x/mad)
  #n7 - quotient transformation (x/range)
  #n8 - quotient transformation (x/max)
  #n9 - quotient transformation (x/mean)
  #n9a - positional quotient transformation (x/median)
  #n10 - quotient transformation (x/sum)
  #n11 - quotient transformation (x/sqrt(SSQ))
  #n12 - normalization ((x-mean)/sqrt(sum((x-mean)^2)))
  #n12a - positional normalization ((x-median)/sqrt(sum((x-median)^2)))
  #n13 - normalization with zero being the central point ((x-midrange)/(range/2))


#1.3
scalling <- function(x){(x-min(x))/(max(x)-min(x))}

titanic_full_preproces$Fare <- scalling(titanic_full_preproces$Fare)
titanic_full_preproces$Age <- scalling(titanic_full_preproces$Age)

titanic_full_transform<-titanic_full_preproces
```

Preparacion de la data para ML

Hide

```
Survived <- titanic_full$Survived
titanic_ml<-cbind(titanic_full_transform,Survived)

titanic_ml <- titanic_ml%>%
                select(-PassengerId,-Name,-Ticket,-Cabin,)

titanic_ml_train <- titanic_ml%>%
                      filter(is.na(Survived)==FALSE)

titanic_ml_train

titanic_ml_test <- titanic_ml%>%
                      filter(is.na(Survived)==TRUE)

titanic_ml_test
```

ML

Finalmente para la predicción, utilizaremos modelos de Machine Learning. En la predicción, revisaremos el uso de cross-validation.

Hide

```
library(caret)
install.packages("h2o")

library(h2o)

model <- train(
              Survived ~ ., titanic_ml_train,
              method = "gbm",
              trControl = trainControl(
                method = "cv", number = 10,
                verboseIter = TRUE
              )
            )
```

Revisión de los resultados del modelo segun los parametros de optimizacion.

Hide

```
model
```

Desarrollamos la prediccion y revisamos la matriz de confusion.

Hide

```
p<-predict(model,titanic_ml_train)

p

# Calculate class probabilities: p_class
p_class <-
  ifelse(p > 0.50,
         1,
         0
  )

# Create confusion matrix
p_class

confusionMatrix(p_class, titanic_ml_train$Survived)
```