Algoritmos

```
tabla_final <- read_csv("~/tabla_final.csv")
```

Eliminar columnas innecesarias para los algoritmos

```
tabla_final[,1]<-NULL
tabla_final[,2:3]<- NULL
```

Crear fechas de duracion Extraer el año de fundacion de la compañia y transformarlo a numero

```
library(lubridate)
tabla_final$founded_at<- year(tabla_final$founded_at)
tabla_final$founded_at<- as.numeric(tabla_final$founded_at)
```

Crear columna año cierre. Si la fecha es Open ponerle 2013 sino poner la fecha de cierre

```
tabla_final$año_cierre <- ifelse(tabla_final$closed_at =="Open",2013,tabla_final$cl
osed_at)
```

Parsear la columna de año de cierre

```
tabla_final$año_cierre <- ymd(tabla_final$año_cierre)
```

Extraer el año de cierre

```
tabla_final$año_cierre <- year(tabla_final$año_cierre)
```

Reemplazar los “NA” por 2013

```
tabla_final$año_cierre[is.na(tabla_final$año_cierre)]<- 2013
tabla_final$año_cierre <-as.numeric(tabla_final$año_cierre)
```

Restar el año de cierre menos la fecha de duracion

```
tabla_final$vida_empresa <- tabla_final$año_cierre - tabla_final$founded_at
class(tabla_final$vida_empresa)
```

Escribir la tabla nueva en csv

```
write.csv(tabla_final, file="tabla_final2.csv")
```

Transformaciom de datos para poder trabajar los algoritmos

Funding total como double

```
tabla_final$funding_total_usd <- as.double(tabla_final$funding_total_usd)
class(tabla_final$funding_total_usd)
```

Universidad como factor

```
tabla_final$funding_total_usd<- as.factor(tabla_final$universidad)
```

Variables numericas como numero

```
tabla_final$oficinas <- as.numeric(tabla_final$oficinas)
tabla_final$directivos <- as.numeric(tabla_final$directivos)
summary(tabla_final)
```

Algoritmos RF Crear una tabla exclusivamente para trabajar los algoritmos

```
tabla_algoritmos <- tabla_final[,c("entity_type","status","funding_rounds","funding
_total_usd","directivos","universidad","inversores","oficinas","directivos","vida_e
mpresa")]
```

Ahora voy a verificar la clase de cada una de las columnas

```
lapply(tabla_algoritmos,class)
```

Transformar el status acquired a ipo

```
tabla_algoritmos$status<- ifelse(tabla_algoritmos$status=="ipo","acquired",tabla_al
goritmos$status)
```

Luego transformar a factor el status

```
tabla_algoritmos$status <- as.factor(tabla_algoritmos$status)
```

Transformar directivos y oficinas en numero y universidad en factor

```
tabla_algoritmos$directivos<- as.numeric(tabla_algoritmos$directivos)
tabla_algoritmos$oficinas <- as.numeric(tabla_algoritmos$oficinas)
tabla_algoritmos[,9]<- NULL
```

Eliminar el entity_tipe

```
tabla_algoritmos[,1]<- NULL
```

Tabla con 3 clases y 18000 observaciones

Crear particion de datos

```
in_train <- createDataPartition(y=tabla_algoritmos$status, p= 0.7, list = FALSE)
trainSet <- tabla_algoritmos[in_train,]
testSet <- tabla_algoritmos[-in_train,]

train= sample(1:nrow(trainSet),12970)

rfModel1 <- randomForest(status ~ .,data=tabla_algoritmos,subset=train)
```

Revisar las variables mas importantes

```
varImpPlot(rfModel,
           sort = T,
           main="Variable Importance",
           n.var=5)
predictionsrf1 <-  predict(rfModel1,testSet)
confusionMatrix(data= predictionsrf1, reference = testSet$status)
```

Reference Prediction acquired closed operating acquired 276 6 27 closed 2 58 25 operating 434 278 4451

Overall Statistics

Accuracy : 0.8611
95% CI : (0.8517, 0.8701) No Information Rate : 0.8103
P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.4189
Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

Class: acquired Class: closed Class: operating Sensitivity 0.38764 0.16959 0.9885 Specificity 0.99319 0.99482 0.3245 Pos Pred Value 0.89320 0.68235 0.8621 Neg Pred Value 0.91692 0.94810 0.8680 Prevalence 0.12813 0.06154 0.8103 Detection Rate 0.04967 0.01044 0.8010 Detection Prevalence 0.05561 0.01530 0.9291 Balanced Accuracy 0.69041 0.58221 0.6565

Libreria MLR

Esta libreria no funciona con que el target sea multiclase Asi que primero saque el status operating para dejar las clases aqcired y closed

library(dplyr)

Prueba sacando el operating de la tabla algoritmos Se decidio dejarlo en dos clases para mejorar el accuracy del algoritmo

```
tabla_algoritmos2 <- tabla_algoritmos %>% filter(status!="operating")

in_train <- createDataPartition(y=tabla_algoritmos2$status, p= 0.7, list = FALSE)
trainSet <- tabla_algoritmos2[in_train,]
testSet <- tabla_algoritmos2[-in_train,]

train= sample(1:nrow(trainSet),2463)

rfModel2 <- randomForest(status ~ .,data=tabla_algoritmos2,subset=train)
```

Revisar las variables mas importantes

```
varImpPlot(rfModel,
           sort = T,
           main="Variable Importance",
           n.var=5)

predictionsrf2 <-  predict(rfModel2,testSet)

confusionMatrix(data= predictionsrf2, reference = testSet$status)
```

Confusion Matrix and Statistics

Reference Prediction acquired closed acquired 657 77 closed 55 265

Accuracy : 0.8748
95% CI : (0.8533, 0.8942) No Information Rate : 0.6755
P-Value [Acc > NIR] : < 2e-16

Kappa : 0.7095
Mcnemar's Test P-Value : 0.06758

Sensitivity : 0.9228
Specificity : 0.7749
Pos Pred Value : 0.8951
Neg Pred Value : 0.8281
Prevalence : 0.6755
Detection Rate : 0.6233
Detection Prevalence : 0.6964
Balanced Accuracy : 0.8488

'Positive' Class : acquired

El resultado fue de 77% por ende se decidio realizar un tuning de parametros para poder mejorar el accuracy

```
library(e1071)
modelTuningParams <- list(ntree = c(500, 1000, 1500, 2000), mtry = 3:8)


modelTuning <- tune(randomForest, status ~ ., data = trainSet, ranges = modelTuning
Params)

modelTuning$best.parameters
```

ntree mtry 500 3

El resultado del tuning fue de 500 y 3 variables por nodo

```
rfModel_tuned <- randomForest(status ~ .,data=tabla_algoritmos2, subset= train, ntr
ee = 500, mtry= 3)

predictionsrf <-  predict(rfModel_tuned,testSet)



confusionMatrix(data= predictionsrf, reference = testSet$status)

varImpPlot(rfModel_tuned,
           sort = T,
           main="Variable Importance",
           n.var=5)
```

Confusion Matrix and Statistics

Reference Prediction acquired closed acquired 667 58 closed 45 284 # Accuracy : 0.9023
95% CI : (0.8827, 0.9195) No Information Rate : 0.6755
P-Value [Acc > NIR] : <2e-16

Kappa : 0.7749
Mcnemar’s Test P-Value : 0.237

Sensitivity : 0.9368
Specificity : 0.8304
Pos Pred Value : 0.9200
Neg Pred Value : 0.8632
Prevalence : 0.6755
Detection Rate : 0.6328
Detection Prevalence : 0.6879
Balanced Accuracy : 0.8836

‘Positive’ Class : acquired

El mejor modelo para nuestros datos fue el random forrest con el tuning. Este modelo predice 90% de los datos. La sensibilidad un 93% y la especificidad un 83%