1. ETL Extract Transform Load

* Obtener los datos en un Excel y analizar cuáles sirven, depurar valores, ver si hay algo extra para calcular a priori.
  + Preguntas sobre los valores de los datos:
    - Se cambiaron los -- y vacíos por “no”, donde correspondía
    - Se normalizaron todos los valores de Plan 1988 y 88 a 88, ya que son equivalentes.
    - Los valores de Plan S95 y 1995 non son equivalentes.
    - Se marcaron como “usables = 0” los registros incompletos para no usarlos hasta tener la información necesaria. Faltan datos de Plan y adeuda trab final con ?
    - Una vez realizados estos cambios, se cambiaron los valores de sí y no por si=1 y no=0, para utilizar booleanos en la BD.
  + Se agregó el dato de Alumnos Recibido al final del 1er año de Reválidas (SI/NO)
* Importar los datos a la BD a partir de un Excel
  + Definir DBMS: PostgreSQL- listo
  + Copy con archivo CSV: cambiar permisos de lectura en el disco o mover el archivo a donde el usuario pueda abrirlo.

COPY public.prueba FROM 'D:\prueba.csv' USING DELIMITERS ';' with csv

* Agregar Store Procedures si es necesario calcular datos derivados a partir de las variables originales

1.2 Variables Involucradas

2. Calcular la Clasificación de Base para ver la precisión mínima a alcanzar

Total de alumnos inscriptos=123

Graduados al finalizar R1=50

Clasificación de Base = 40,65% de presición

3. Calcular con los distintos métodos sobre las distintas variables para ver cuáles podrían ser índices viables (según la precisión). Incluyendo el dato “Alumno Recibido” (S/N).

* OneR

**Script**

# query the data from postgreSQL

df\_postgres <- dbGetQuery(con, "SELECT anio\_ingreso,finales\_adeudados,cursadas\_adeudadas,optativas\_adeudadas,adeuda\_trab\_final\_inicio,plan,tutor\_r1, recibido\_fin\_r1 from revalidas WHERE usable=TRUE")

#df\_postgres <- dbGetQuery(con, "SELECT finales\_adeudados,cursadas\_adeudadas,optativas\_adeudadas,adeuda\_trab\_final\_inicio,plan,tutor\_r1, recibido\_fin\_r1 from revalidas WHERE usable=TRUE")

df\_postgres [ , ]

str(df\_postgres)

#install.packages(OneR)

require("OneR")

model <- OneR(df\_postgres, verbose = TRUE)

summary(model)

plot(model)

prediction <- predict(model, df\_postgres)

eval\_model(prediction, df\_postgres)

**Resultados grales:**

Attribute Accuracy

1 \* anio\_ingreso 64.75%

2 plan 62.3%

3 finales\_adeudados 59.84%

4 cursadas\_adeudadas 59.02%

4 optativas\_adeudadas 59.02%

4 adeuda\_trab\_final\_inicio 59.02%

4 tutor\_r1 59.02%

[podriamos agregar un one rule con una formula -ind ponderado x sentido común- ]

* J48

Script

## install.packages("RWeka")

require("RWeka")

df\_postgres <- dbGetQuery(con, "SELECT finales\_adeudados, cursadas\_adeudadas, optativas\_adeudadas, adeuda\_trab\_final\_inicio , plan, tutor\_r1, tutor\_r2, recibido\_fin\_r1 from revalidas")

## Identify a decision tree.

df\_postgres <- lapply(df\_postgres, as.factor)

m <- J48(recibido\_fin\_r1 ~., data = df\_postgres, control= Weka\_control(M=1,U=TRUE))

m

## Use 10 fold cross-validation.

e <- evaluate\_Weka\_classifier(m, cost = matrix(c(0,2,1,0), ncol = 2),numFolds = 10, complexity = TRUE,seed = 123, class = TRUE)

e

summary(e)

e$details

**Resultados:**

J48 unpruned tree

------------------

plan = 1995: FALSE (10.0)

plan = 88: FALSE (11.0/1.0)

plan = s95

| finales\_adeudados = 0: TRUE (2.0)

| finales\_adeudados = 1: FALSE (1.0)

| finales\_adeudados = 2: FALSE (2.0)

| finales\_adeudados = 3: FALSE (1.0)

| finales\_adeudados = 4: FALSE (0.0)

| finales\_adeudados = 5: FALSE (0.0)

plan = S95

| finales\_adeudados = 0

| | adeuda\_trab\_final\_inicio = FALSE: TRUE (6.0/1.0)

| | adeuda\_trab\_final\_inicio = TRUE

| | | tutor\_r1 = FALSE: TRUE (30.0/14.0)

| | | tutor\_r1 = TRUE

| | | | optativas\_adeudadas = 0: TRUE (24.0/9.0)

| | | | optativas\_adeudadas = 1: FALSE (1.0)

| | | | optativas\_adeudadas = 2: FALSE (2.0/1.0)

| | | | optativas\_adeudadas = 3: TRUE (0.0)

| | | | optativas\_adeudadas = 4: TRUE (0.0)

| | | | optativas\_adeudadas = 5: TRUE (0.0)

| finales\_adeudados = 1

| | optativas\_adeudadas = 0

| | | adeuda\_trab\_final\_inicio = FALSE: TRUE (1.0)

| | | adeuda\_trab\_final\_inicio = TRUE: FALSE (3.0/1.0)

| | optativas\_adeudadas = 1: FALSE (1.0)

| | optativas\_adeudadas = 2: FALSE (2.0)

| | optativas\_adeudadas = 3: FALSE (2.0)

| | optativas\_adeudadas = 4: TRUE (1.0)

| | optativas\_adeudadas = 5: FALSE (0.0)

| finales\_adeudados = 2

| | adeuda\_trab\_final\_inicio = FALSE: TRUE (1.0)

| | adeuda\_trab\_final\_inicio = TRUE

| | | tutor\_r1 = FALSE

| | | | optativas\_adeudadas = 0: TRUE (1.0)

| | | | optativas\_adeudadas = 1: FALSE (1.0)

| | | | optativas\_adeudadas = 2: FALSE (0.0)

| | | | optativas\_adeudadas = 3: FALSE (0.0)

| | | | optativas\_adeudadas = 4: FALSE (0.0)

| | | | optativas\_adeudadas = 5: FALSE (0.0)

| | | tutor\_r1 = TRUE: FALSE (6.0)

| finales\_adeudados = 3: FALSE (4.0)

| finales\_adeudados = 4: FALSE (5.0)

| finales\_adeudados = 5: TRUE (1.0)

plan = S96: TRUE (2.0)

plan = S97: TRUE (1.0)

plan = S98: TRUE (1.0)

Number of Leaves : 37

Size of the tree : 48

>

> ## Use 10 fold cross-validation.

> e <- evaluate\_Weka\_classifier(m, cost = matrix(c(0,2,1,0), ncol = 2),numFolds = 10, complexity = TRUE,seed = 123, class = TRUE)

> e

=== 10 Fold Cross Validation ===

=== Summary ===

Correctly Classified Instances 77 62.6016 %

Incorrectly Classified Instances 46 37.3984 %

Kappa statistic 0.2532

Total Cost 63

Average Cost 0.5122

K&B Relative Info Score 2760.8453 %

K&B Information Score 26.9166 bits 0.2188 bits/instance

Class complexity | order 0 119.8999 bits 0.9748 bits/instance

Class complexity | scheme 10815.6883 bits 87.9324 bits/instance

Complexity improvement (Sf) -10695.7883 bits -86.9576 bits/instance

Mean absolute error 0.371

Root mean squared error 0.4846

Relative absolute error 76.8339 %

Root relative squared error 98.6476 %

Total Number of Instances 123

=== Detailed Accuracy By Class ===

TP Rate FP Rate Precision Recall F-Measure MCC ROC Area PRC Area Class

0.603 0.340 0.721 0.603 0.657 0.258 0.675 0.748 FALSE

0.660 0.397 0.532 0.660 0.589 0.258 0.675 0.533 TRUE

Weighted Avg. 0.626 0.363 0.644 0.626 0.629 0.258 0.675 0.661

=== Cost Matrix ===

0 1

2 0

=== Confusion Matrix ===

a b <-- classified as

44 29 | a = FALSE

17 33 | b = TRUE

> summary(e)

Length Class Mode

string 1 -none- character

details 8 -none- numeric

detailsCost 1 -none- numeric

detailsComplexity 4 -none- numeric

detailsClass 12 -none- numeric

confusionMatrix 4 -none- numeric

> e$details

pctCorrect pctIncorrect pctUnclassified

62.6016260 37.3983740 0.0000000

kappa meanAbsoluteError rootMeanSquaredError

0.2531679 0.3710235 0.4845945

relativeAbsoluteError rootRelativeSquaredError

76.8338570 98.6475973

[pegar resultados con las variables como estan]

[armar atributos derivados para rangos en las var de finales, optativas y cursadas adeudados y calcular con eso]

* Otros algoritmos…?

Analizar los resultados para cada combinación método-variable/s (anotar).

4. Automatizar el proceso de carga de datos en BD, cálculo del/los indicador/es y presentación de resultados.