

Proyecto Final

André Rodas

Aplicación

- Este es un proyecto de clasificación binaria. El **objetivo** es generar y evaluar diferentes modelos de probabilidad para saber si los billetes son fraudulentos o no.



Data set

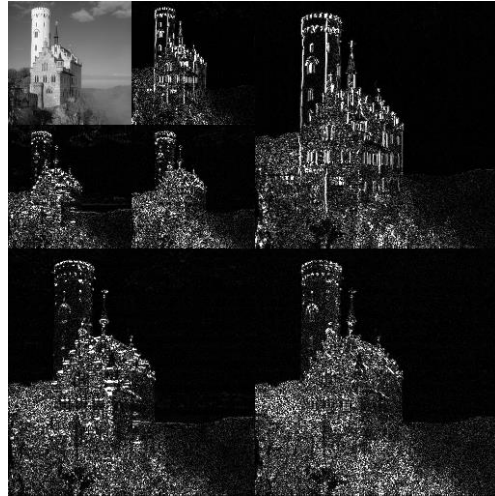
- El archivo banknote.csv es la fuente de información para el problema de clasificación. El número de instancias (filas) en el conjunto de datos es 1372 y el número de variables (columnas) es 5.
- El archivo contiene las siguientes variables:
 - **Variance**: utilizado como entrada. Continua. Wavalet Trasform aplicado.
 - **Skewness**: utilizado como entrada. Continua. Wavalet Trasform aplicado.
 - **Curtosis**: utilizado como entrada. Continua. Wavalet Trasform aplicado.
 - **Entropy**: usado como entrada. Continua. Wavalet Trasform aplicado.
 - **Class**: utilizado como objetivo. Solo puede tener dos valores: 0 (no falsificado) o 1 (falsificado).

Fuente del data set

- Los datos se extrajeron de imágenes que se tomaron de ejemplares similares a **billetes genuinos y falsificados**. Para la digitalización se utilizó una **cámara industrial** que se suele utilizar para la inspección de impresiones. Las imágenes finales tienen **400x 400 píxeles**. Debido a la lente del objeto y la distancia al objeto investigado, se obtuvieron imágenes en escala de grises con una resolución de aproximadamente **660 dpi**. El método **Wavelet Transform** se utilizó para extraer características de las imágenes.
- Walvelet Transform (slide hidden)

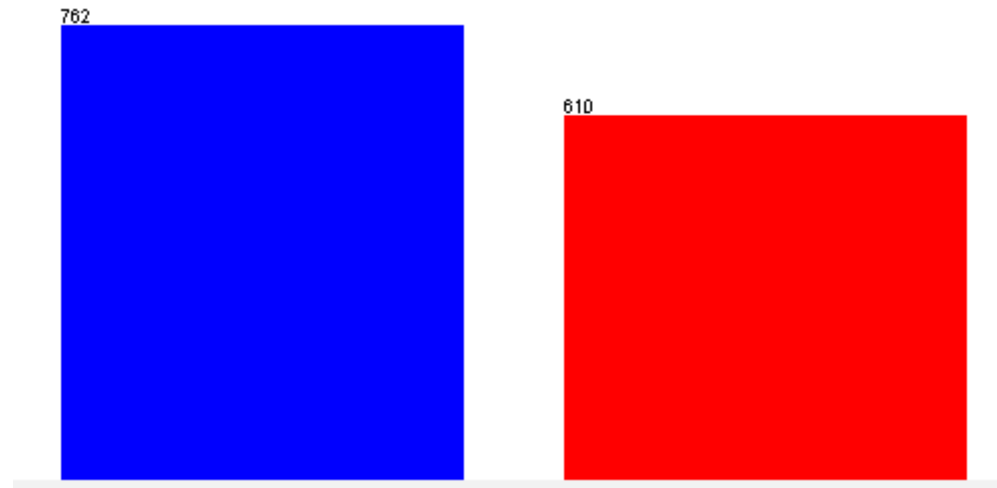
Wavelet Transform

“The basic idea behind wavelet transform is, a new basis(window) function is introduced which can be enlarged or compressed to capture both low frequency and high frequency component of the signal (which relates to scale).”



Fuentes: <https://www.intechopen.com/chapters/61705>

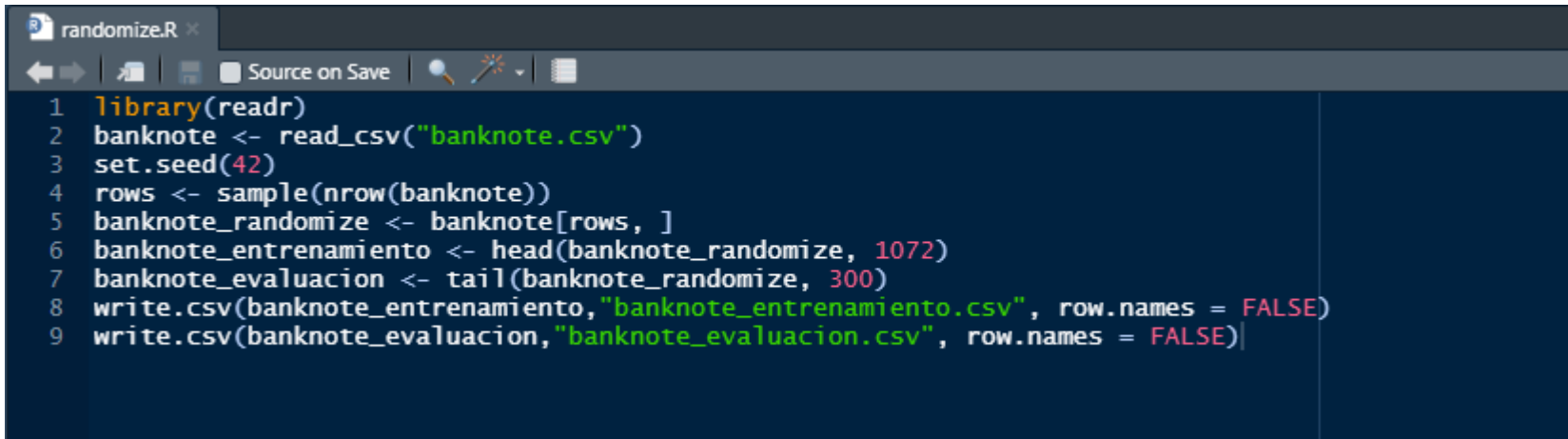
Proporción de datos



- 0 - (azul) no falsificado
- 1 - (rojo) falsificado

Datos de entrenamiento y evaluación

- De lo enseñado en clase se aplicó un shuffle de los datos, necesario por si llevan cierto orden que pueda ocasionar un sesgo en el entrenamiento. Para ello, se desarrolló un script en R. Adicional, se separó entre los datos de entrenamiento (1072) y evaluación (300).

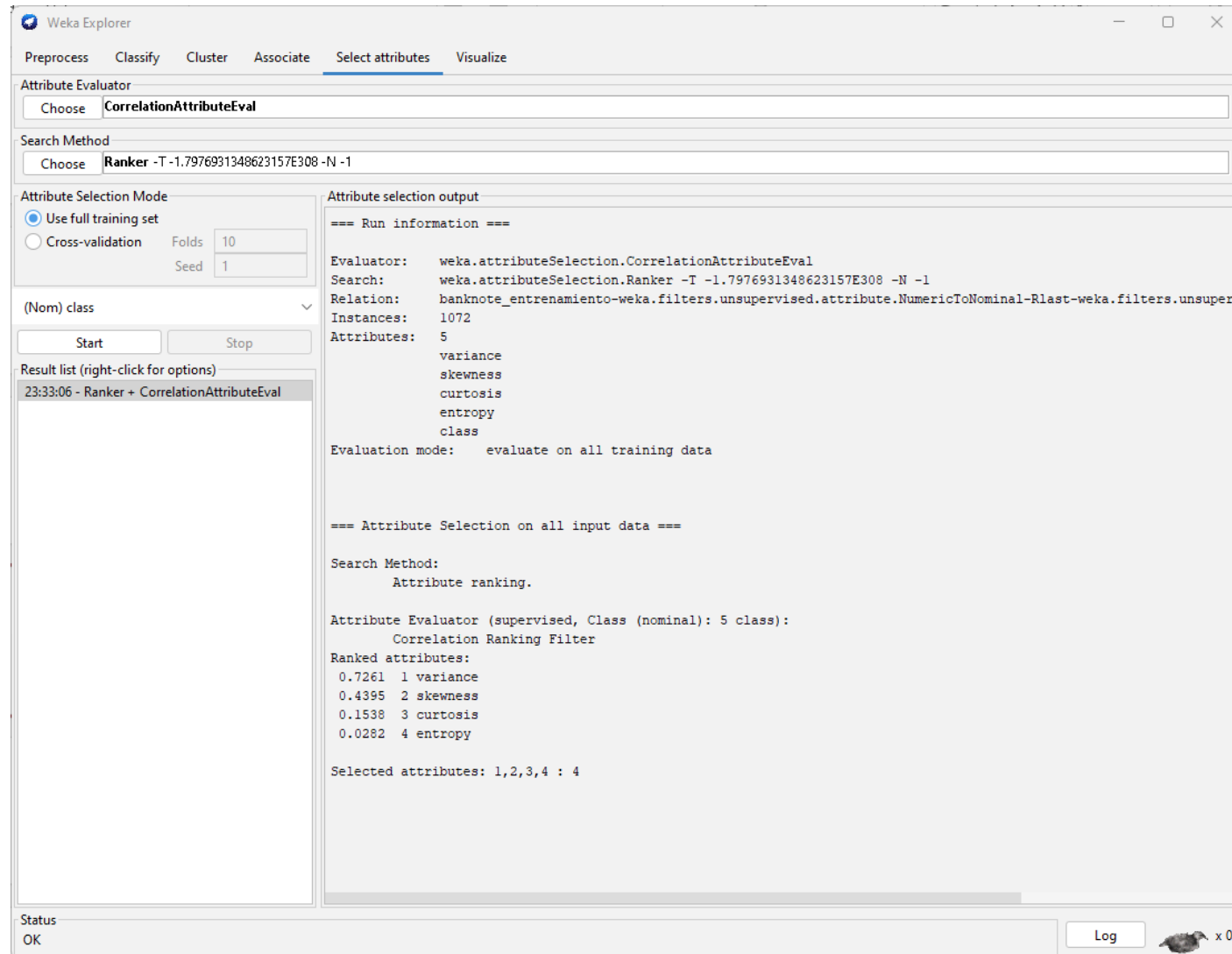


```
1 library(readr)
2 banknote <- read_csv("banknote.csv")
3 set.seed(42)
4 rows <- sample(nrow(banknote))
5 banknote_randomize <- banknote[rows, ]
6 banknote_entrenamiento <- head(banknote_randomize, 1072)
7 banknote_evaluacion <- tail(banknote_randomize, 300)
8 write_csv(banknote_entrenamiento, "banknote_entrenamiento.csv", row.names = FALSE)
9 write_csv(banknote_evaluacion, "banknote_evaluacion.csv", row.names = FALSE)
```

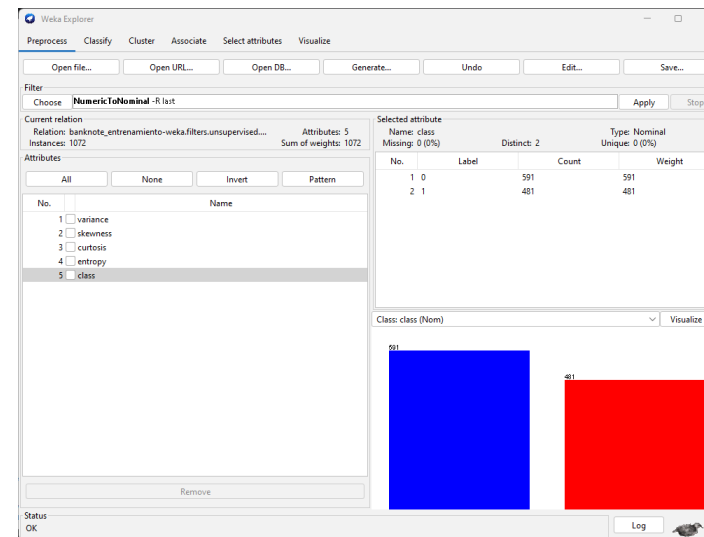
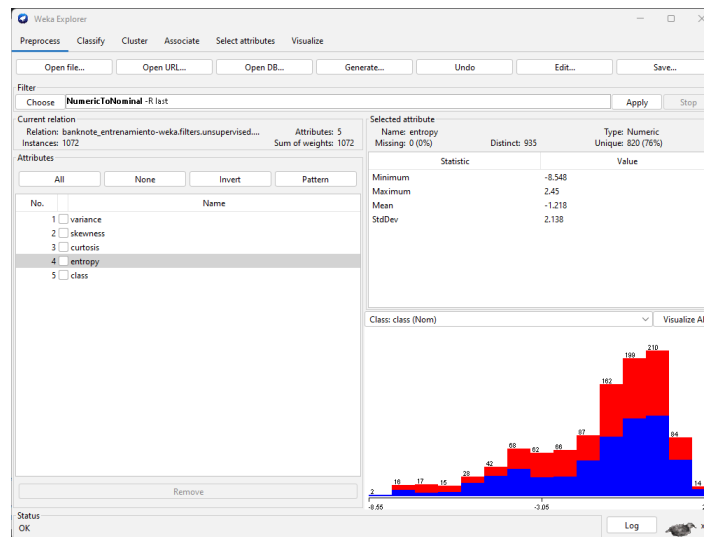
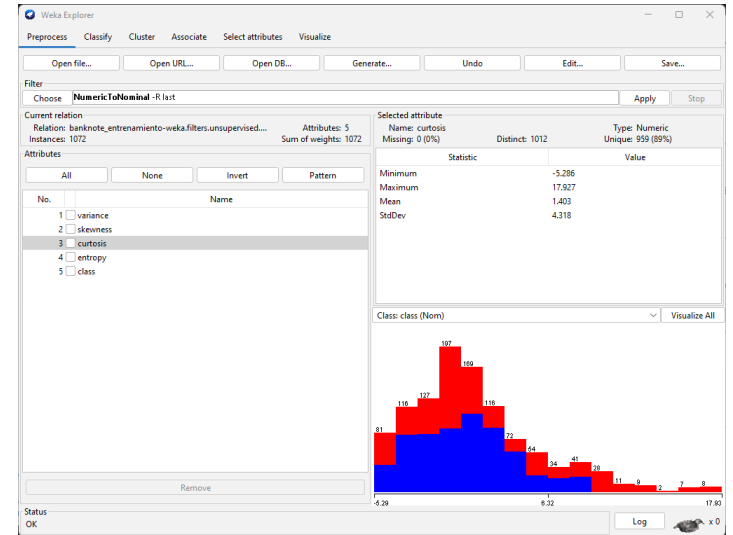
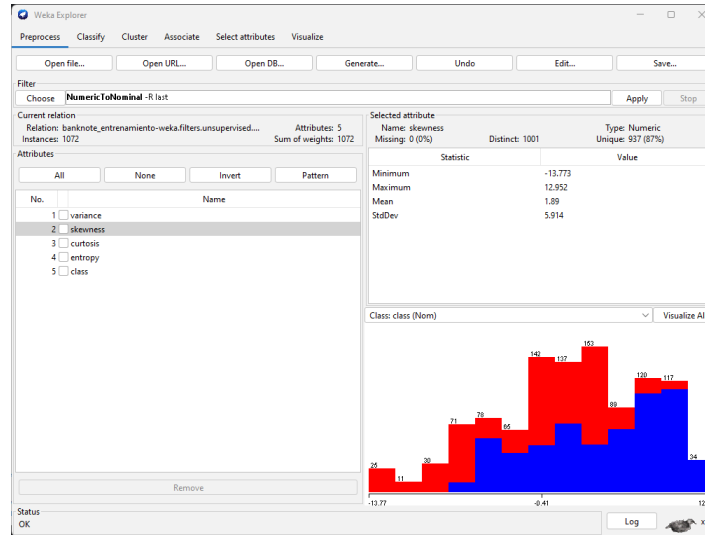
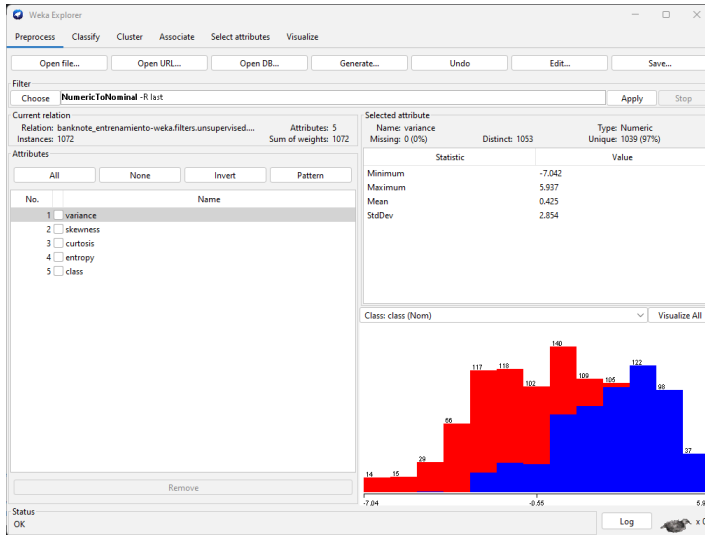
Feature Selection [1]

- Se tomaron las todas las features (4 en total).
 - Esto debido a que el data set contiene muy pocas features. Adicional, estuve consultando proyectos en Python y usuarios han logrado una precisión alta (1 o cercana) tomando en cuenta todas las features.
 - Al ser medidas de estadísticas de distribución se prefirió no hacer ningún tipo de operación entre estas.
 - El desconocimiento de Wavelet Transform aplicado en las métricas hace que no realice más allá que una normalización en los datos.
 - De quitar una a lo mejor sería la entropía, ya que no aporta tanto al modelo de una forma significativa versus las otras (ver el Correlation Attribute de Weka en la siguiente diapositiva), aún así se conserva.

Feature Selection [2]



Distribuciones



Modelo Logistic Regression

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier: Choose RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1

Test options

- ☐ Use training set
- ☐ Supplied test set (Set...)
- ☒ Cross-validation Folds: 10
- ☐ Percentage split %: 66

More options...

(Nom) class

Start Stop

Result list (right-click for options)

- 22:20:55 - functions.Logistic
- 22:21:25 - meta.AdaBoostM1
- 22:21:37 - trees.RandomForest

Classifier output

Odds Ratios...

Variable	Class
variance	1.6722602368742152E42
skewness	6.804450082935159E47
curtosis	4.684087829114988E51
entropy	1908.7655

Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	1061	98.9739 %
Incorrectly Classified Instances	11	1.0261 %
Kappa statistic	0.9793	
Mean absolute error	0.0117	
Root mean squared error	0.0847	
Relative absolute error	2.3649 %	
Root relative squared error	17.0379 %	
Total Number of Instances	1072	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.990	0.010	0.992	0.990	0.991	0.979	1.000	1.000	0
	0.990	0.010	0.988	0.990	0.989	0.979	1.000	1.000	1
Weighted Avg.	0.990	0.010	0.990	0.990	0.990	0.979	1.000	1.000	

=== Confusion Matrix ===

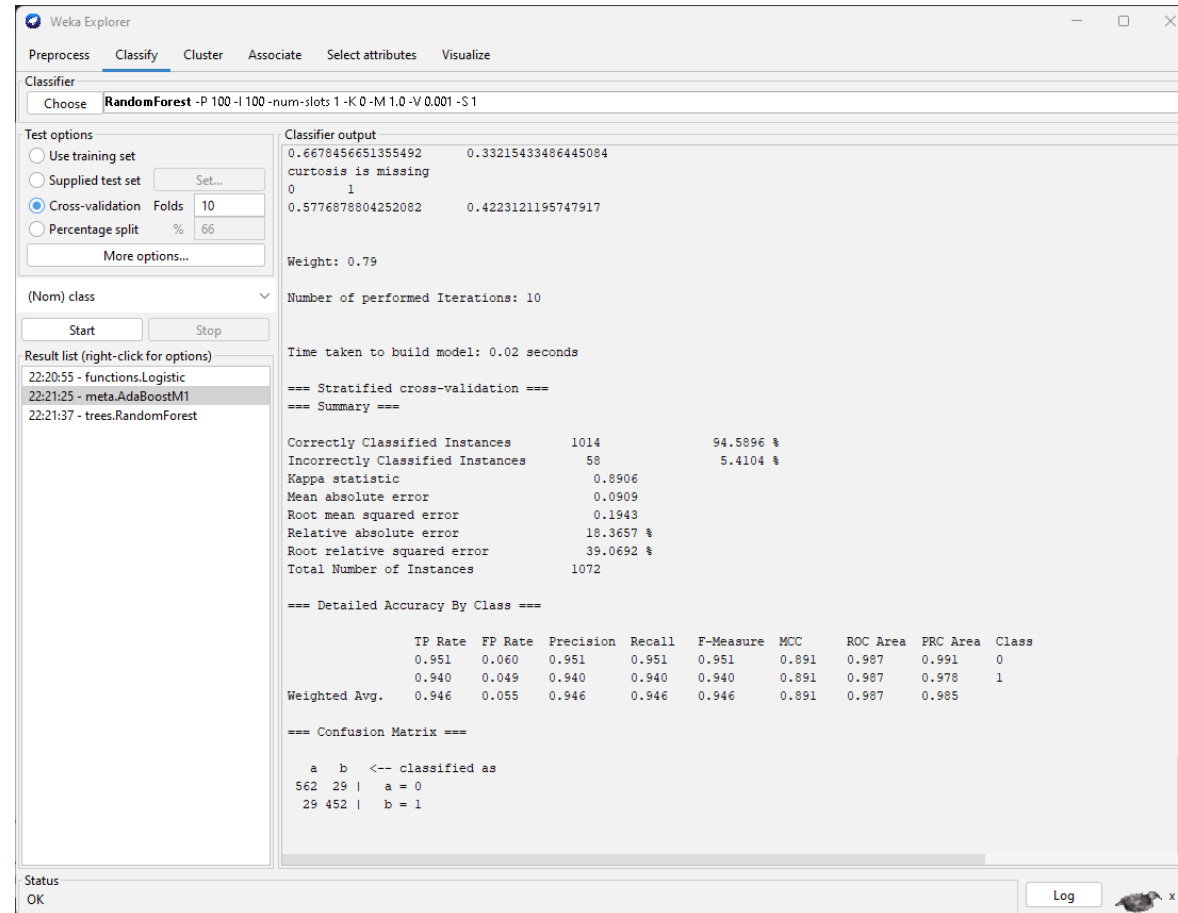
a	b	<-- Classified as
585	6	a = 0
5	476	b = 1

Status: OK

Log x 0

- Se aplicó una normalización en el set de entrenamiento

Modelo AdaBoostM1



The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. The 'Classifier' dropdown is set to 'RandomForest'. The 'Test options' section shows 'Cross-validation' selected with 'Folds' set to 10. The 'Result list' on the left shows three entries: '22:20:55 - functions.Logistic', '22:21:25 - meta.AdaBoostM1' (which is highlighted), and '22:21:37 - trees.RandomForest'. The 'Classifier output' pane on the right displays the results for the AdaBoostM1 model.

Classifier output

```
0.6678456651355492    0.33215433486445084
curtosis is missing
0      1
0.5776878804252082    0.4223121195747917
```

Weight: 0.79

Number of performed Iterations: 10

Time taken to build model: 0.02 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	1014	94.5896 %
Incorrectly Classified Instances	58	5.4104 %
Kappa statistic	0.8906	
Mean absolute error	0.0909	
Root mean squared error	0.1943	
Relative absolute error	18.3657 %	
Root relative squared error	39.0692 %	
Total Number of Instances	1072	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	FRC Area	Class
	0.951	0.060	0.951	0.951	0.951	0.891	0.987	0.991	0
	0.940	0.049	0.940	0.940	0.940	0.891	0.987	0.978	1
Weighted Avg.	0.946	0.055	0.946	0.946	0.946	0.891	0.987	0.985	

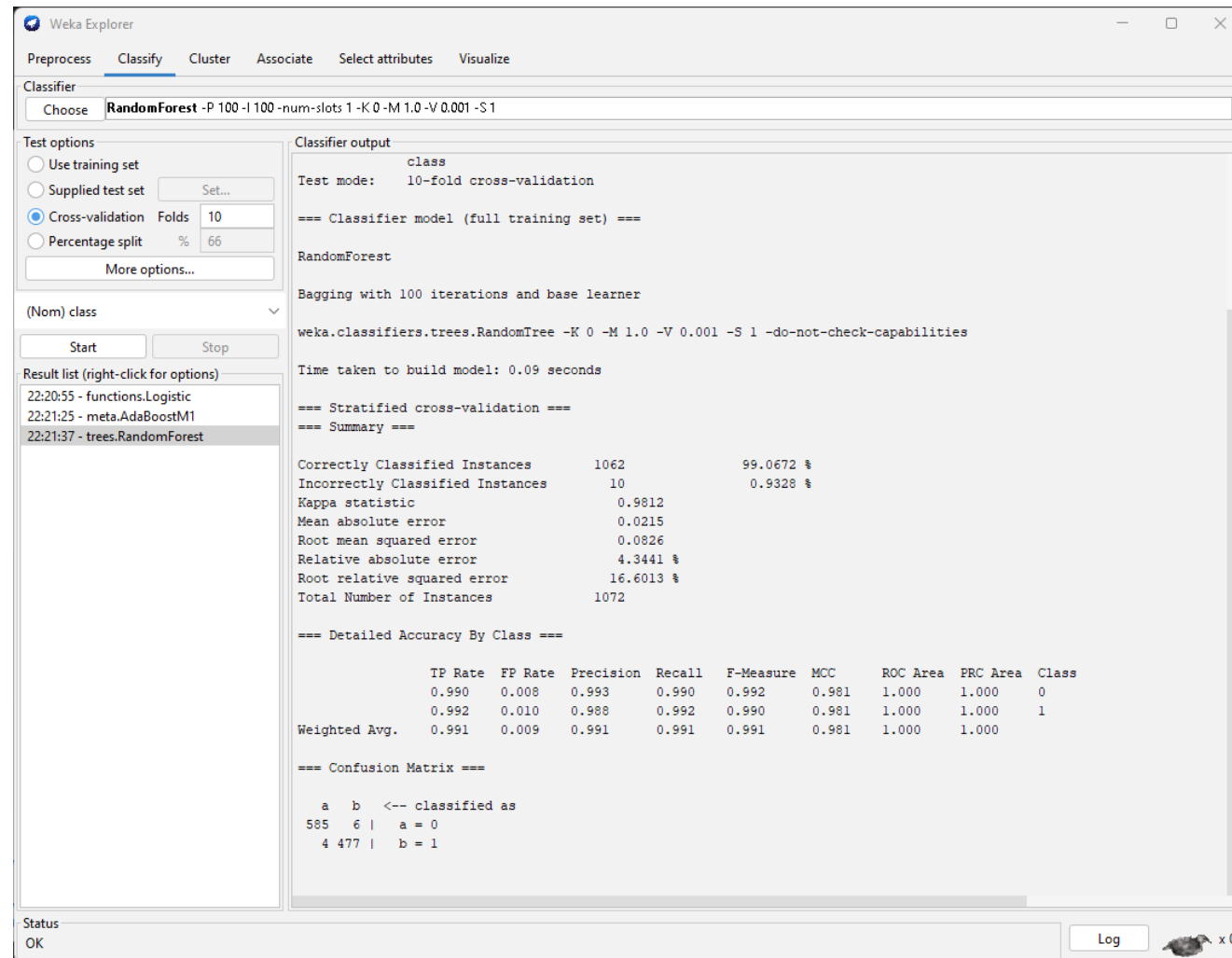
=== Confusion Matrix ===

a	b	<-- classified as
562	29	a = 0
29	452	b = 1

Status: OK

Log x 0

Modelo RandomForest



The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. The classifier chosen is 'RandomForest' with parameters: -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1. The test options are set to 'Cross-validation' with 10 folds. The result list on the left shows three entries: '22:20:55 - functions.Logistic', '22:21:25 - meta.AdaBoostM1', and '22:21:37 - trees.RandomForest', with the last one selected. The classifier output on the right displays the following information:

Classifier output

class
Test mode: 10-fold cross-validation

=== Classifier model (full training set) ===

RandomForest

Bagging with 100 iterations and base learner

weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities

Time taken to build model: 0.09 seconds

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	1062	99.0672 %
Incorrectly Classified Instances	10	0.9328 %
Kappa statistic	0.9812	
Mean absolute error	0.0215	
Root mean squared error	0.0826	
Relative absolute error	4.3441 %	
Root relative squared error	16.6013 %	
Total Number of Instances	1072	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.990	0.008	0.993	0.990	0.992	0.981	1.000	1.000	0
	0.992	0.010	0.988	0.992	0.990	0.981	1.000	1.000	1
Weighted Avg.	0.991	0.009	0.991	0.991	0.991	0.981	1.000	1.000	

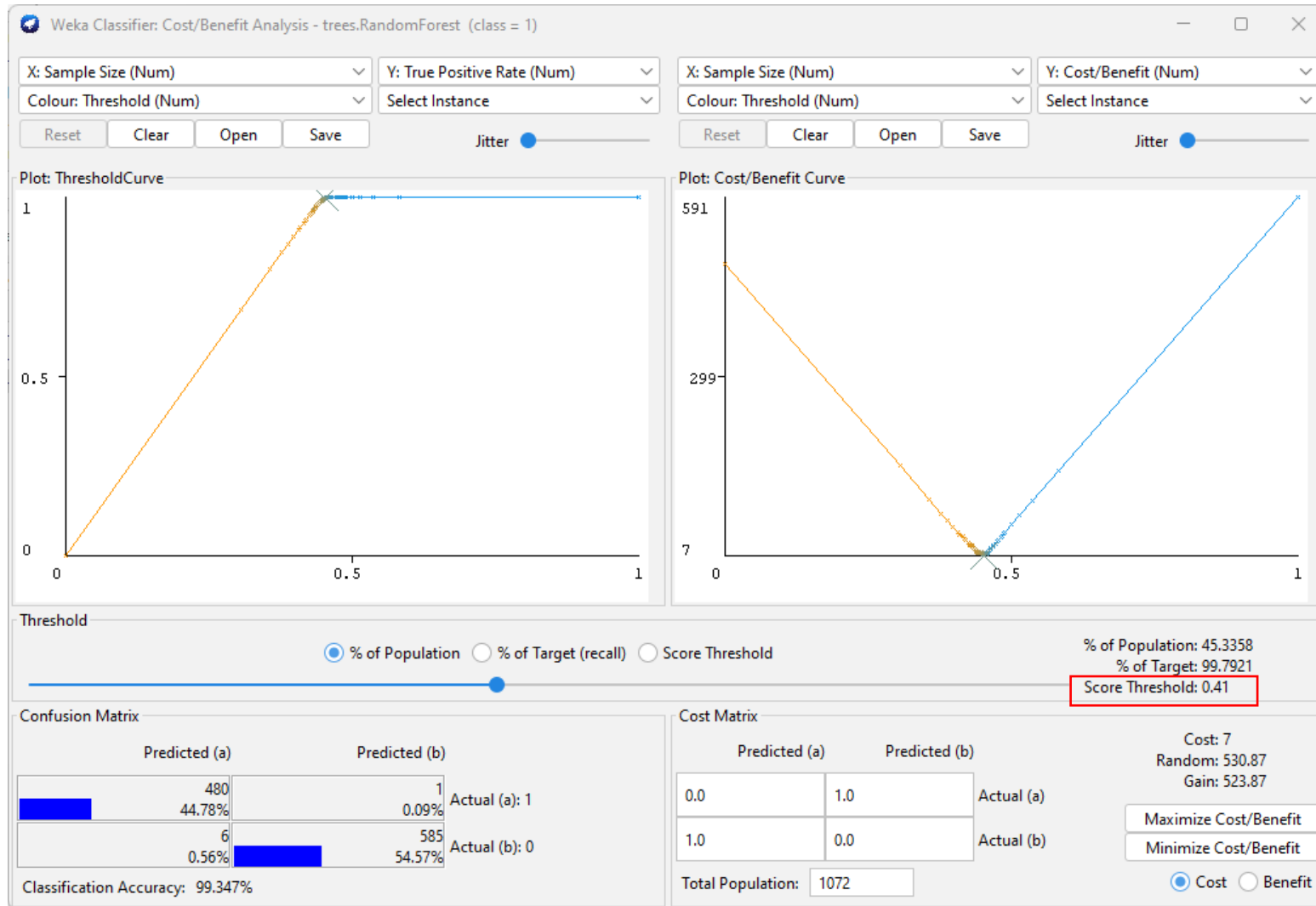
=== Confusion Matrix ===

a	b	<-- classified as
585	6	a = 0
4	477	b = 1

Status: OK

Log x 0

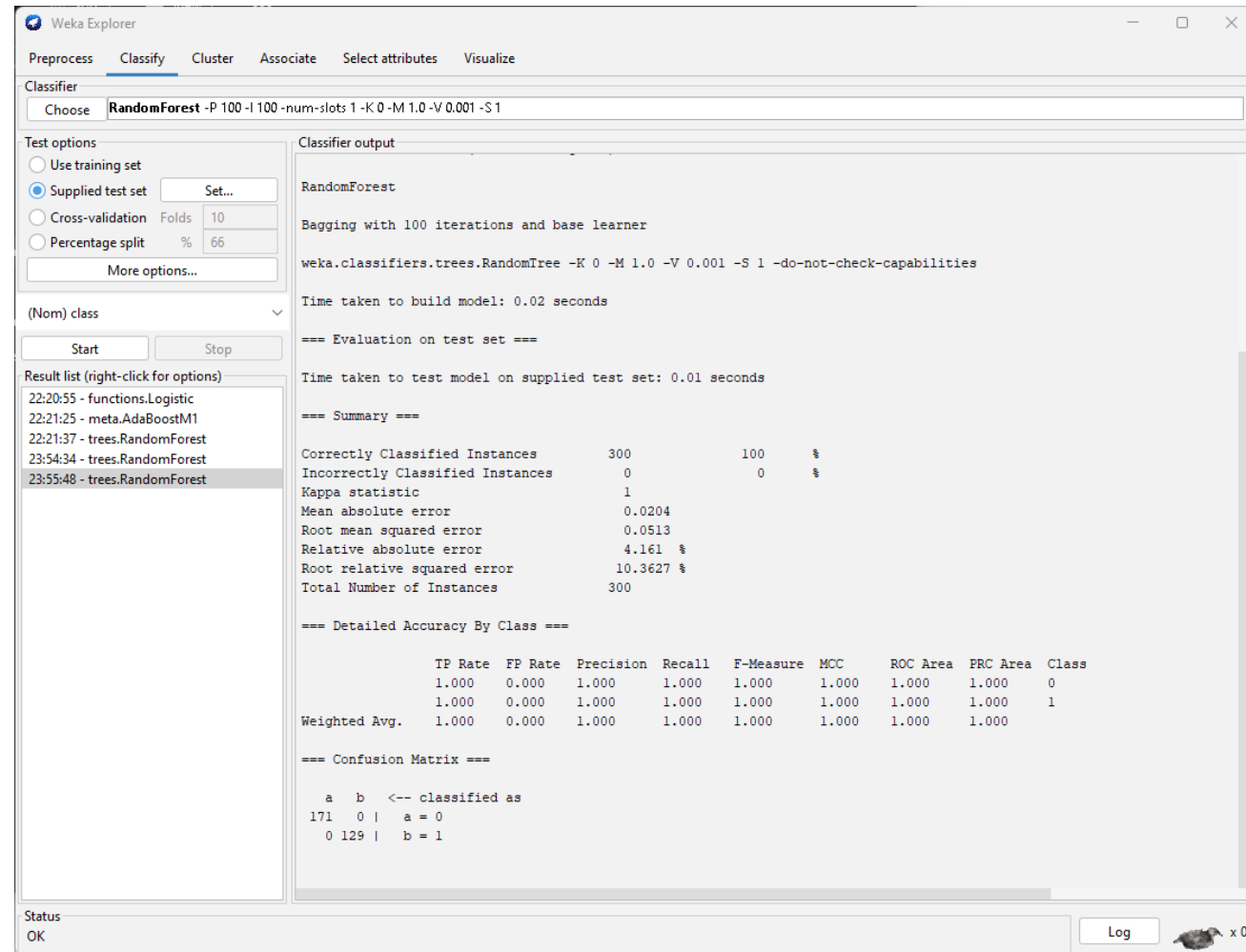
Modelo RandomForest – Cost/Benefit (class = 1)



Selección de modelo

- En este caso quiero seleccionar el modelo con mejor Precision para la Class = 1 (falsificado). Mi razón de negocio es la siguiente: **prefiero detectar los billetes falsificados** y no recibirlos, que tener falsos positivos en billetes no falsificados; en todo caso **es preferible que el cliente me pague con otro billete a que yo reciba uno falsificado**.
- Los modelos de Logistic Regression y RandomForest presentan el mismo precisión y recall para el Class 1. Se investigó cuales eran los pros y contras de estos modelos ([Ver 6.5 Summary of Results](#)).
- **Se selecciona el modelo de RandomForest** ya que posee el mejor true positive rate (según el artículo) y en los resultados tiene una ligera mejora en recall.

Evaluación de modelo RandomForest



The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. The classifier chosen is 'RandomForest' with parameters: -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1. The test options are set to 'Supplied test set' with 10 folds. The result list on the left shows several entries, with the most recent 'trees.RandomForest' entry selected. The classifier output on the right displays the following information:

RandomForest

Bagging with 100 iterations and base learner

weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities

Time taken to build model: 0.02 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

=== Summary ===

Metric	Value	Percentage
Correctly Classified Instances	300	100 %
Incorrectly Classified Instances	0	0 %
Kappa statistic	1	
Mean absolute error	0.0204	
Root mean squared error	0.0513	
Relative absolute error	4.161 %	
Root relative squared error	10.3627 %	
Total Number of Instances	300	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
1.000	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	0
1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1
Weighted Avg.	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	

=== Confusion Matrix ===

a	b	<-- classified as
171	0	a = 0
0	129	b = 1

Status: OK

Log x0

- Se tomó el archivo banknote_evaluacion_transformado.arff

Discusión de resultados

- Obtuvimos un **Precision** y **Recall** de **1** para las dos Class. Estamos con un **modelo robusto** sin estar realizando overfitting. Obviamente, ayuda que los datos que ya están depurados y analizados anteriormente para liberarlos en la web.
- A lo mejor el modelo se podría mejorar realizando un PCA, pero perderíamos interpretabilidad de qué significan esas variables.
- Está fuera de mi rango de conocimiento métodos cómo Wavelet Transform, por lo que no tengo alguna propuesta de que variables adicionales incluir en el modelo que nos ayude a mejorar el Precision desde la toma de datos.