TFG
Beyond Traditional Frameworks for Modelling Learning Domains

Sergio Sancho Asensio
Director: Xavier Solé Beteta

Enginyeria i Arquitectura La Salle – Universitat Ramon Llull
Quatre camins 2, 08022 Barcelona
ls24029@salleurl.edu

Julio 2015

Motivaciones

- Hoy en día el personal docente debe invertir mucho tiempo y esfuerzo modelando el rendimiento académico del alumnado.
  - Realizar este proceso de forma **manual** es poco efectivo.

- Gracias a los nuevos avances en el diseño de los procesadores y las altas prestaciones ofrecidas por éstos, se ha incrementado el interés en el **Aprendizaje Automático** [Mitchell, 1997].
  - Se han aplicado técnicas tradicionales de Aprendizaje Automático en el campo docente y los resultados **no son muy precisos**.

Objetivos

Propósito

- El propósito del presente TFG es revisar el **estado del arte** en técnicas de Aprendizaje Automático con la finalidad de ayudar al cuerpo docente a **modelar** y **predecir** el rendimiento académico del alumnado.

Objetivos

- **Revisar** las principales técnicas del Aprendizaje Automático.
- **Desarrollar** un algoritmo *Gradient Boosting* y **compararlo** con otras técnicas.
- Aplicar *Gradient Boosting* a datos docentes con la finalidad de **modelar** al alumnado.

Tabla de contenidos

Introducción

Árboles de decisión

Sistemas clasificadores avanzados

Experimentación

Modelado del rendimiento académico de los alumnos

Tiempo dedicado

Conclusiones

Líneas de futuro

Aprendizaje Automático

- El Aprendizaje Automático es una rama de la inteligencia artificial cuyo objetivo es desarrollar técnicas que permitan a las computadoras **aprender** con la finalidad de construir **modelos descriptivos** a partir de los datos, realizar **predicciones**, etcétera.

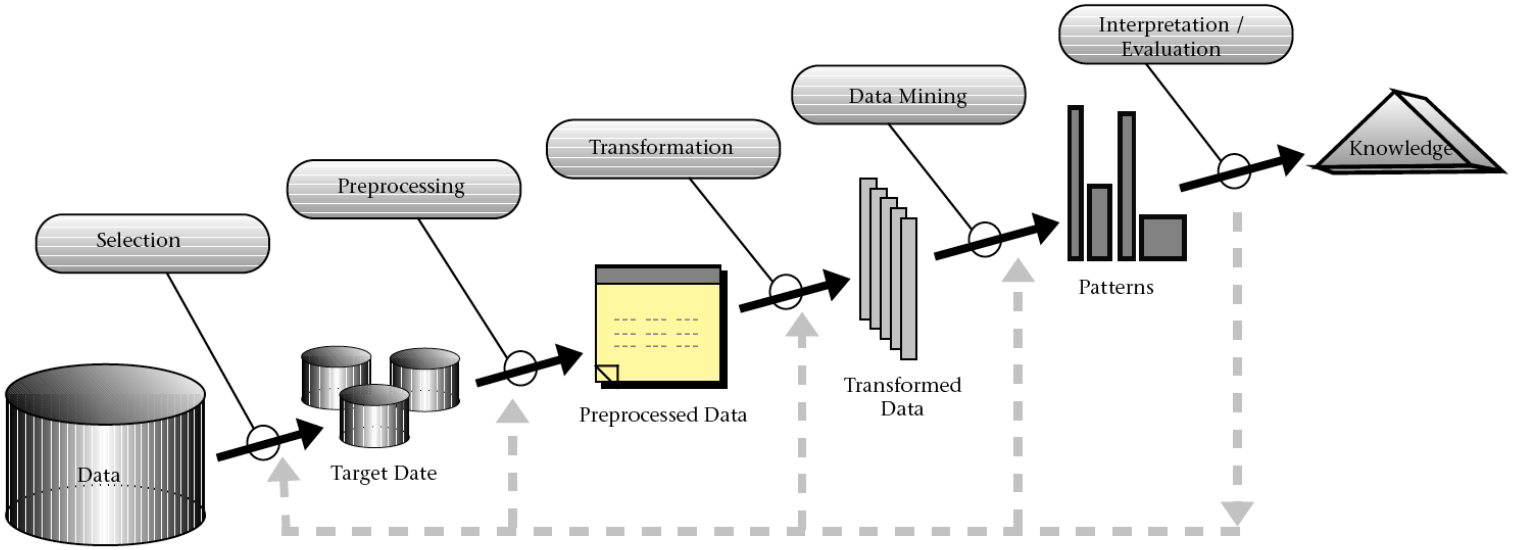
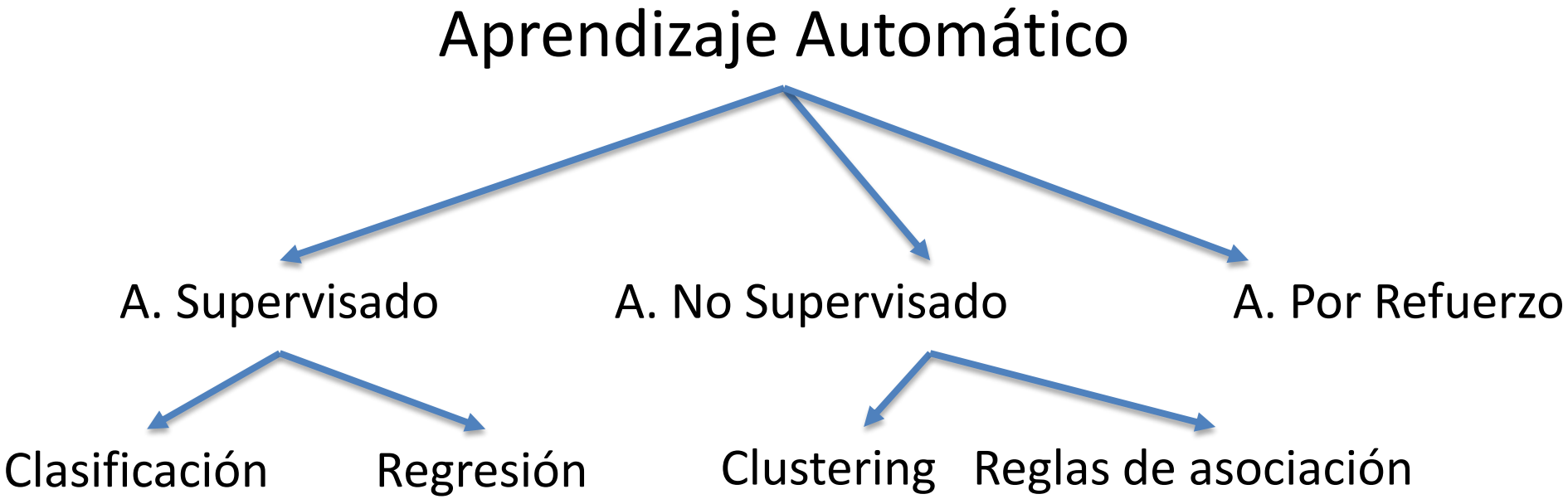


Figura 1.1: Ciclo de vida *Knowledge Discovery Database* según Fayyad et al. [1996].

Paradigmas



A. Perezoso (*IBK*)

Redes Neuronales (*Back Propagation*)

Modelos probabilísticos (*Naive Bayes*)

Máquinas de Soporte Vectorial (*SMO*)

Árboles de decisión (*J48*)

Técnicas más comunes supervisadas [Wu et al.,2007].

Introducción

Árboles de decisión

Sistemas clasificadores avanzados

Experimentación

Modelado del rendimiento académico de los alumnos

Tiempo dedicado

Conclusiones

Líneas de futuro

Introducción a los árboles de decisión

- Un árbol es una estructura **recursiva** formada por un nodo raíz, nodos internos y nodos hoja.

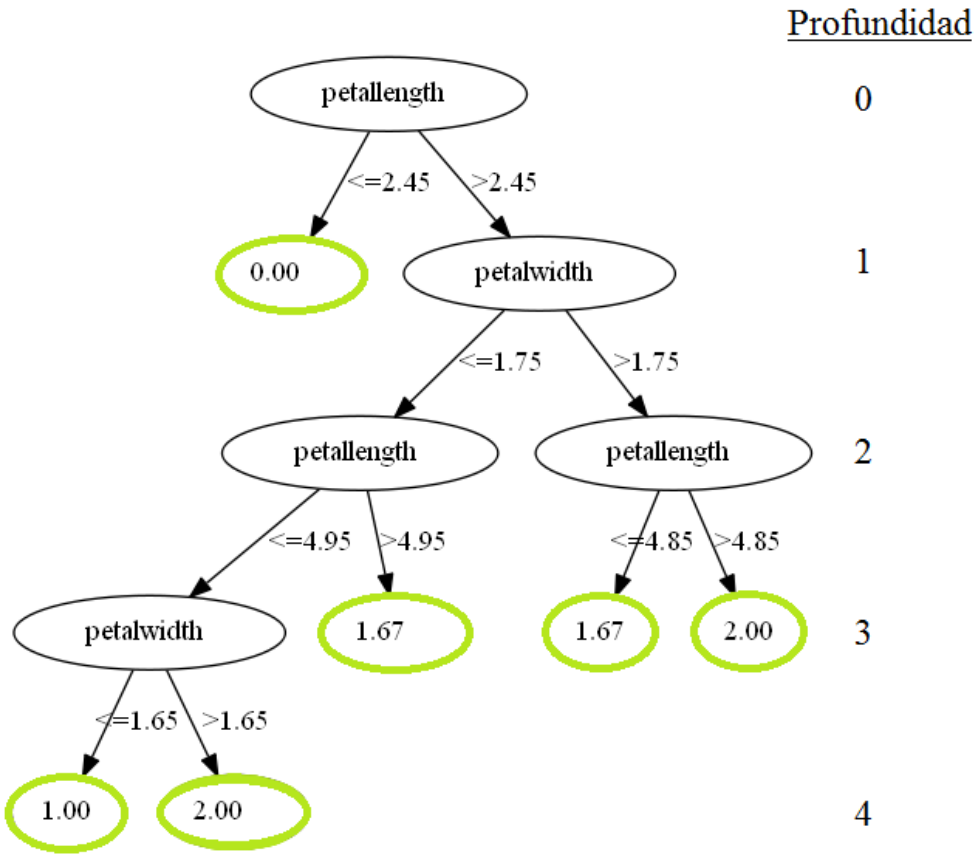


Figura 2.1: Árbol binario de regresión que modela el *data set Iris* [Bache and Lichman, 2013].

Fase de entrenamiento

Ejemplo de construcción de un árbol binario de decisión (fase de entrenamiento).

Atributo X_1	Atributo X_2	Atributo Y
1	1	△
2	3	△
3	1	●
4	2	●
5	1	△

Tabla 2.1: *Data set* de ejemplo con 5 instancias. Formado por los atributos X_1, X_2 y la clase Y.

Fase de entrenamiento

1. Ordenar el primer atributo de menor a mayor.

Atributo X_1	Atributo X_2	Atributo Y
1	1	△
2	3	△
3	1	●
4	2	●
5	1	△

Tabla 2.1: *Data set* de ejemplo con 5 instancias. Formado por los atributos X_1, X_2 y la clase *Y*.

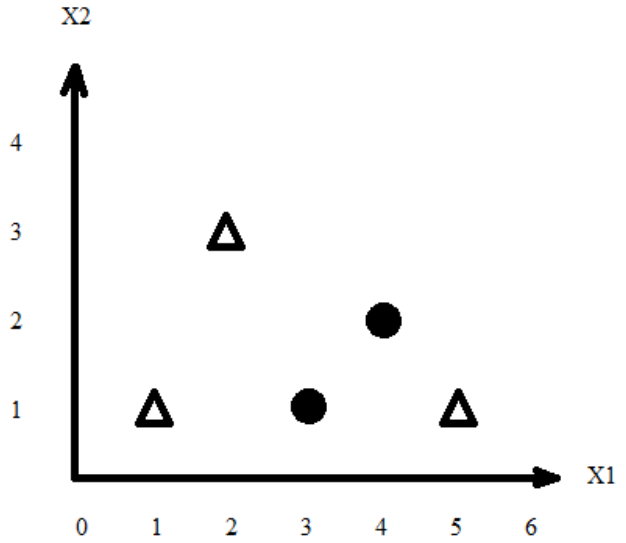


Figura 2.2: Representación gráfica del *data set* de ejemplo en función de X_1 y X_2.

Fase de entrenamiento

- Seleccionar criterio que defina el modo en que el modelo ha particionado los datos hasta el nodo hoja actual. Éste se denomina **impureza** y puede ser la ganancia en varianza, la entropía, el índice *Gini*, etcétera.
  - La **ganancia en varianza** es la heurística más común en regresión [Breiman et al., 1984]

Atributo X_1	Atributo X_2	Atributo Y
1	1	△
2	3	△
3	1	●
4	2	●
5	1	△

Tabla 2.1: *Data set* de ejemplo con 5 instancias. Formado por los atributos X_1, X_2 y la clase *Y*.

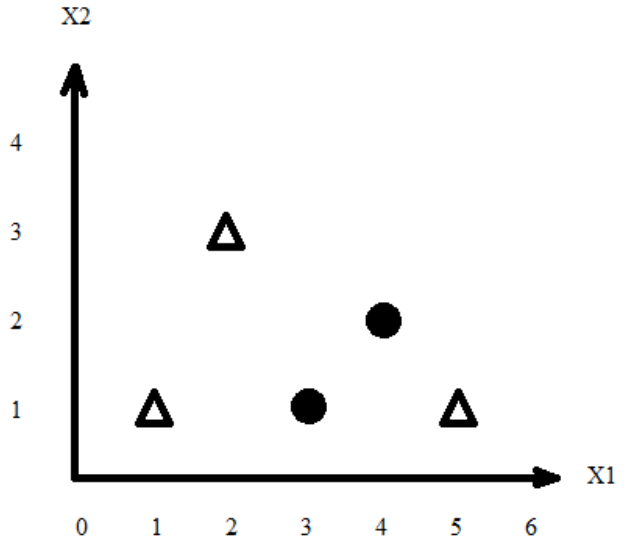


Figura 2.2: Representación gráfica del *data set* de ejemplo en función de X_1 y X_2.

Fase de entrenamiento

2. Calcular la varianza del atributo actual utilizando todo el *data set* (sin hacer partición).

Atributo X_1	Atributo X_2	Atributo Y
1	1	△
2	3	△
3	1	●
4	2	●
5	1	△

Tabla 2.1: *Data set* de ejemplo con 5 instancias. Formado por los atributos X_1, X_2 y la clase *Y*.

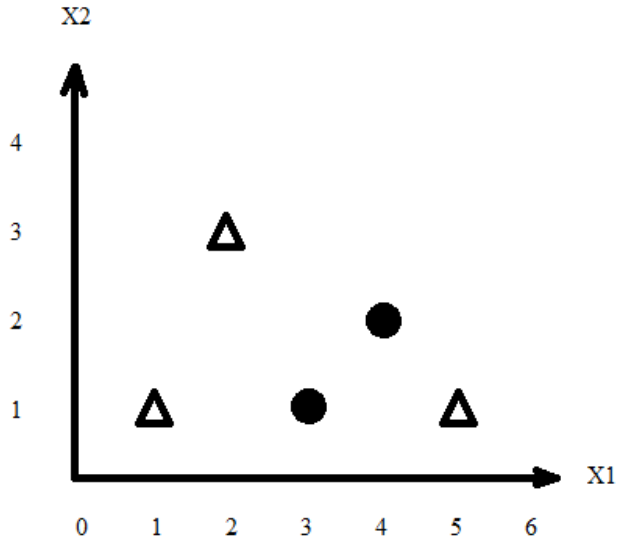


Figura 2.2: Representación gráfica del *data set* de ejemplo en función de X_1 y X_2.

Fase de entrenamiento

3. Calcular la varianza ejemplo a ejemplo del atributo actual guardando la mejor partición hasta el momento (la que contiene menos varianza). La heurística que se utiliza en el ejemplo es el promedio.

Atributo X_1	Atributo X_2	Atributo Y	
1	1	△	
2	3	△	2.5
3	1	●	
4	2	●	4.5
5	1	△	

Tabla 2.1: *Data set* de ejemplo con 5 instancias. Formado por los atributos X_1, X_2 y la clase *Y*.

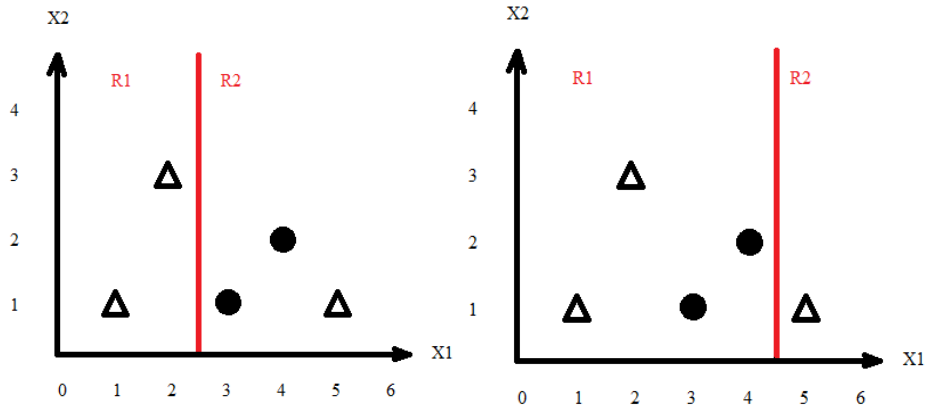


Figura 2.3: Candidatos al particionamiento de la variable.

Fase de entrenamiento

4. Ordenar el siguiente atributo en orden ascendente.

5. calcular la varianza de todo el atributo sin hacer partición utilizando todo el *data set*.

Atributo X_1	Atributo X_2	Atributo Y
1	1	△
3	1	•
5	1	△
4	2	•
2	3	△

Tabla 2.2: *Data set* formado por los atributos X_1, X_2 y la clase Y. Se han ordenado las instancias según el atributo X_2.

Fase de entrenamiento

6. Calcular varianza ejemplo a ejemplo utilizando el segundo atributo. En el ejemplo se ve claramente que se comete un error insalvable en la tercera región donde hay dos tipos de respuesta distintos.

Atributo X_1	Atributo X_2	Atributo Y
1	1	△
3	1	●
5	1	△
4	2	●
2	3	△

1.5
2.5

Tabla 2.2: *Data set* formado por los atributos X_1, X_2 y la clase Y. Se han ordenado las instancias según el atributo X_2.

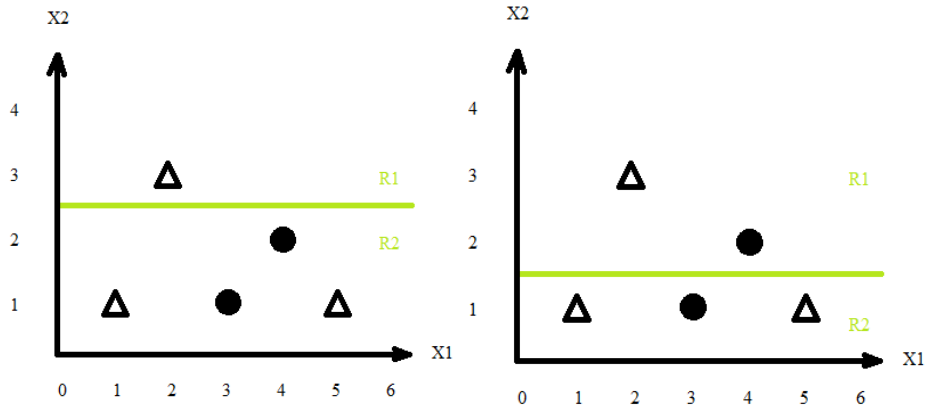


Figura 2.4: Candidatos al particionamiento de la variable.

Fase de entrenamiento

7. Elegir aquella variable con menor varianza y generar dos particiones del *data set* inicial.

Atributo X_1	Atributo X_2	Atributo Y
1	1	△
2	3	△

Tabla 2.3: Primera partición del *data set*, no hay impureza. Se generará un nodo hoja directamente con una predicción del 100%.

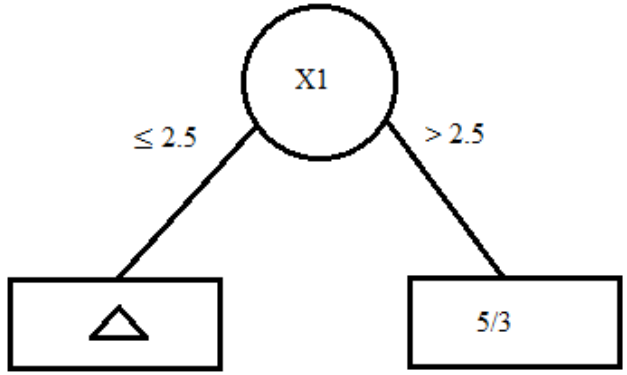


Figura 2.5: Primera etapa del árbol generado. Se trata de un árbol con profundidad 1 y con un nodo **hoja puro** (sin varianza). El valor de la partición es 2.5 y 5/3 es un valor numérico temporal e implica que de los cinco casos posibles quedan tres por diferenciar.

Fase de entrenamiento

- A medida que se van generando **nodos internos**, se va particionando el espacio de búsqueda hasta que la **impureza** es mínima.
  - Impureza nula significa que la partición es perfecta

Atributo X_1	Atributo X_2	Atributo Y
3	1	●
4	2	●
5	1	△

Tabla 2.4: Segunda partición del *data set*, en este caso hay impureza.

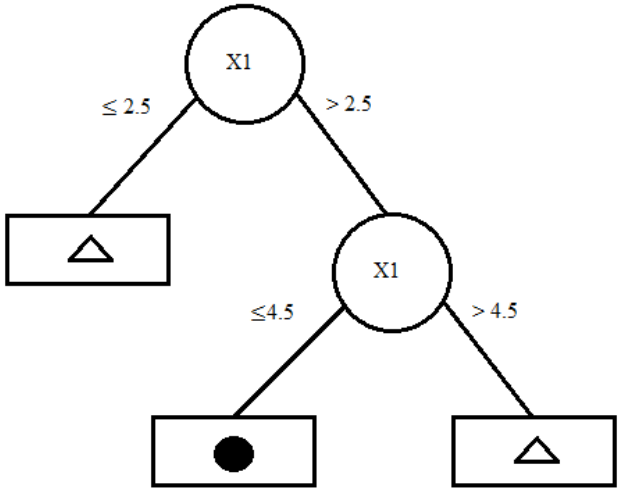


Figura 2.6: Segunda etapa (y final) del árbol generado. El árbol es **puro**, es decir, el acierto es del 100% y **ha descartado por completo el segundo atributo**, significa que éste no aporta nada.

Fase de entrenamiento

- Se utilizan métodos para finalizar el proceso de la construcción del árbol basados en limitar:
  - El número de nodos
  - La profundidad del árbol
  - **La pureza del árbol**

Fase de predicción

Fase de predicción utilizando el *data set* del ejemplo anterior (ver *tabla 2.1*).

Atributo X_1	Atributo X_2	Atributo Y
3	3	?

Tabla 2.5: Predicción de la clase *Y según* X_1 y X_2.

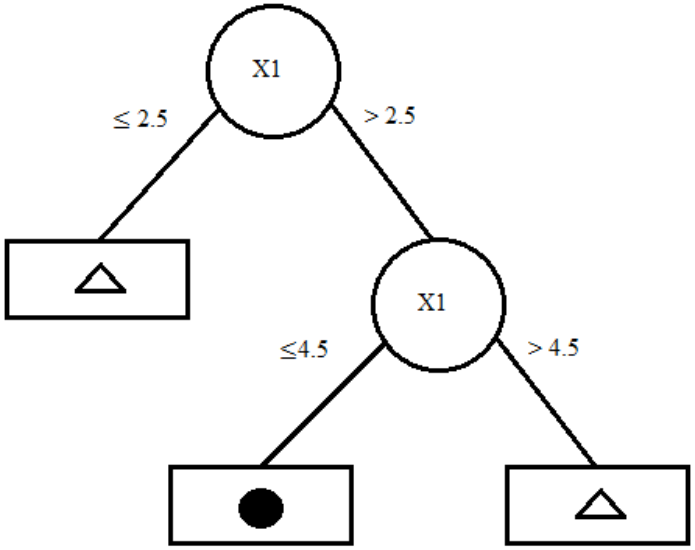


Figura 2.6: Etapa final del árbol generado.

Fase de predicción

Fase de predicción utilizando el *data set* del ejemplo anterior (ver *tabla 2.1*).

Atributo X_1	Atributo X_2	Atributo Y
3	3	?

Tabla 2.5: Predicción de la clase *Y según* X_1 y X_2.

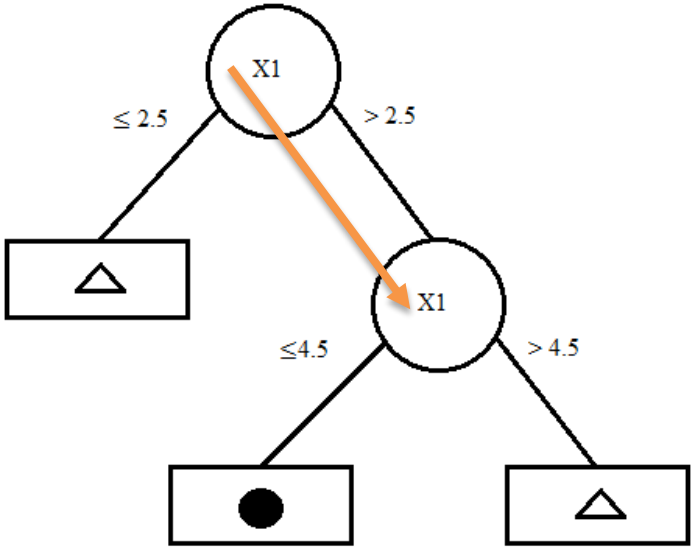


Figura 2.6: Etapa final del árbol generado.

Fase de predicción

Fase de predicción utilizando el *data set* del ejemplo anterior (ver *tabla 2.1*).

Atributo X_1	Atributo X_2	Atributo Y
3	3	●

Tabla 2.5: Predicción de la clase *Y según* X_1 y X_2.

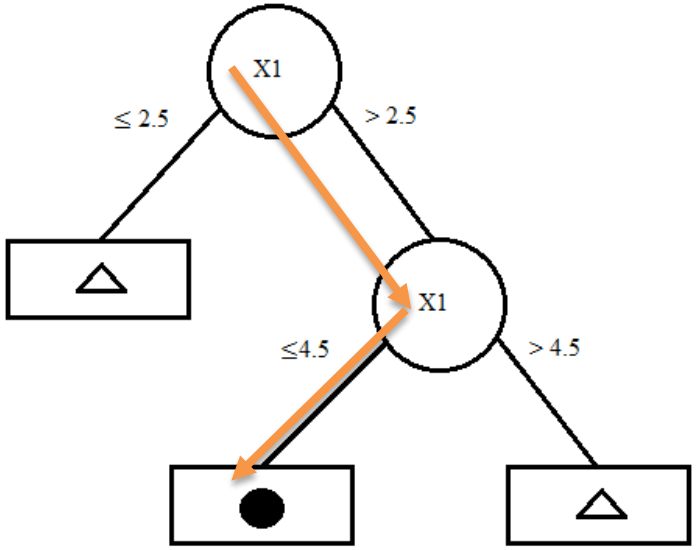


Figura 2.6: Etapa final del árbol generado.

Introducción

Árboles de decisión

Sistemas clasificadores avanzados

Experimentación

Modelado del rendimiento académico de los alumnos

Tiempo dedicado

Conclusiones

Líneas de futuro

Contexto

- Continuamente se están realizando investigaciones sobre nuevas técnicas que permitan obtener clasificadores más certeros, unas de las más utilizadas recientemente se denomina ***Ensemble*** [James et al. 2013]

- Los métodos ***Ensemble*** se caracterizan por combinar las predicciones de múltiples clasificadores obteniendo así una mayor precisión a uno individual, las dos familias más populares son *Bagging* y ***Boosting***.

- ***Gradient Boosting*** es una técnica que pertenece a la familia *Boosting* de los métodos *Ensemble.* Es especialmente efectiva con **árboles binarios de regresión** como base [Friedman, 2000].

Introducción al *Gradient Boosting*

- Combinar múltiples *weak learners* (ligeramente mejores que una elección al azar) para producir un *strong learner*.
- Mejorar continuamente mediante los **residuos** (la salida de un *weak learner* se verá mejorada por el *weak learner* generado en la iteración siguiente).

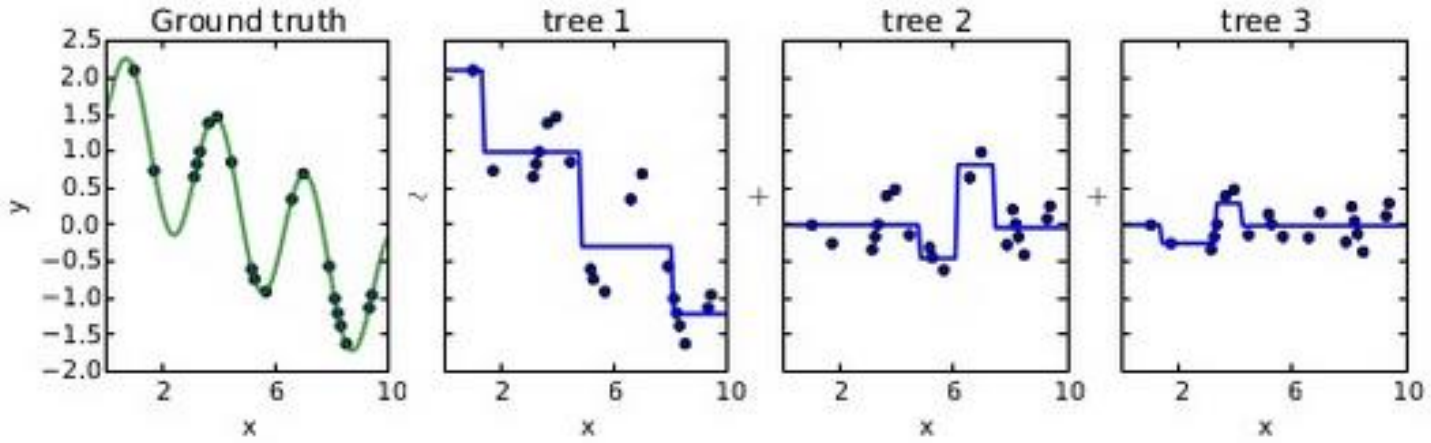


Figura 3.1: Técnica *Gradient Boosting* [Friedman, 2000]. La curva verde representa la función que se desea aproximar. Los puntos del primer árbol representan la salida real menos la obtenida, en el segundo representan la diferencia de la salida del primer árbol con la obtenida y en el tercero la diferencia entre la salida del segundo árbol y la obtenida.

Introducción al *Gradient Boosting*

- El residuo es la diferencia entre el valor real y el aproximado *(loss-function).*

$$L(y, F(x)) = \frac{1}{2}\sum_{k=1}^{\text{ins}} (y_k - \hat{y}_k)^2$$

Donde ins es el número de instancias.

- Los residuos se minimizan utilizando un gradiente. La idea es que estos residuos tiendan a cero (no haya error).

$$r_{im} = -\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}$$

Donde instancia $i = 1, \ldots, n$ y árbol $m = 1, \ldots, M$.

- A la predicción se le denomina *gamma*.

$$\gamma = \eta \frac{c-1}{c} E$$

Donde η es el ratio de aprendizaje, c es el número de clases y E el valor esperado.

Fase de entrenamiento

1. Mapear la clase a residuos.

Sepal length	Sepal width	Petal length	Petal width	Clase	Residuo
5.1	3.5	1.4	0.2	(1) Iris-Setosa	[1,0,0]
7	3.2	4.7	1.4	(2) Iris-Versicolor	[0,1,0]
6.3	3.3	6	2.5	(3) Iris-Virginica	[0,0,1]
4.9	3	1.4	0.2	(1) Iris-Setosa	[1,0,0]

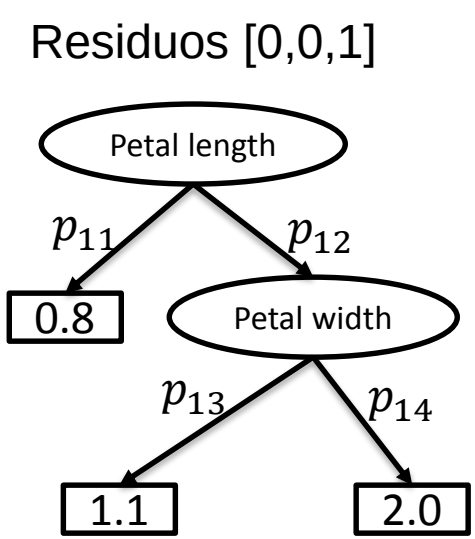
Tabla 3.1: Fragmento del *data set Iris* formado por cuatro instancias. Este *data set* está formado por los atributos *Sepal length, Sepal width, Petal length, Petal width* y la clase (*Iris-Setosa, Iris-Versicolor y Iris-Virginica*). La columna residuo mapea cada una de las clases (***one versus all***), convierte un problema de *n* clases en *n* problemas de una clase.

2. Generar un *weak learner* para cada clase, para cada *weak learner*:

- utilizar los residuos de la iteración anterior.
- calcular la *gamma* para cada nodo hoja.
- restar residuos para cada una de las clases.

Fase de entrenamiento

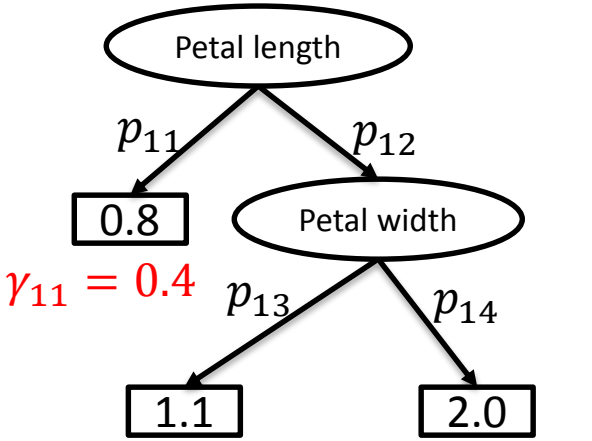
Ejemplo con tres árboles por clase, en este caso se modela la clase *Iris-Virginica*.



Fase de entrenamiento

Se calculan las *gammas* para cada nodo hoja y los residuos según estas.

Residuos [0,0,1]

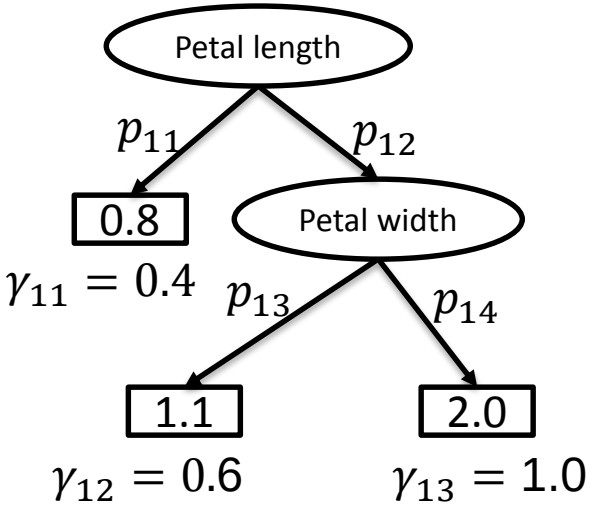


$$r_1 = \gamma_{11}p_{11} + \gamma_{12}p_{12}p_{13} + \gamma_{13}p_{12}p_{14} = 0.4$$

Fase de entrenamiento

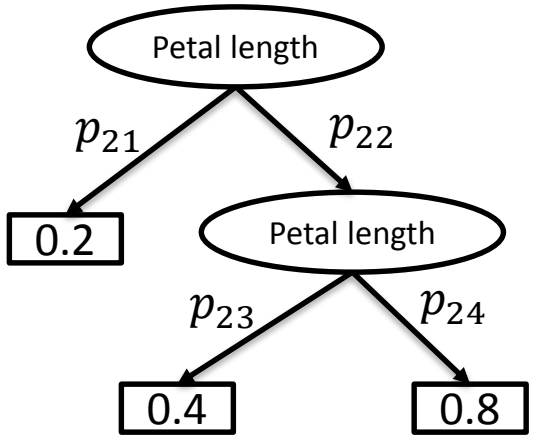
Se genera el segundo árbol según el nuevo residuo.

Residuos [0,0,1]



$r_1 = 0.4$

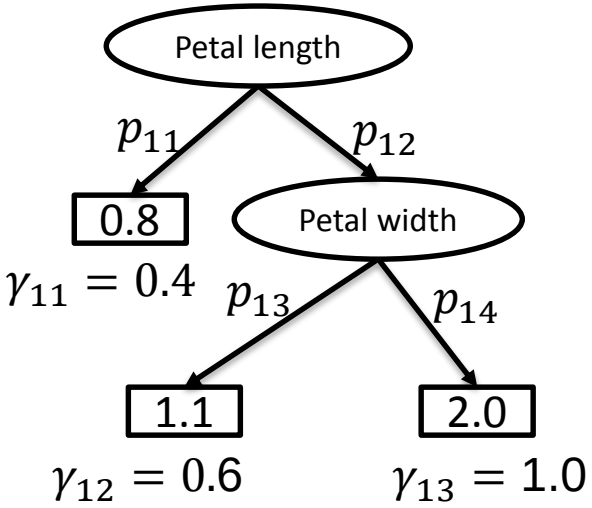
Residuos [0,0,0.6]



Fase de entrenamiento

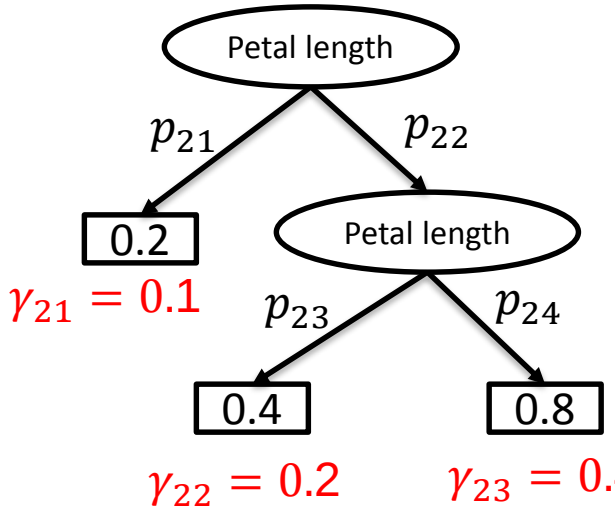
Se calculan las *gammas* para cada nodo hoja y los residuos según estas.

Residuos [0,0,1]



$r_1 = 0.4$

Residuos [0,0,0.6]

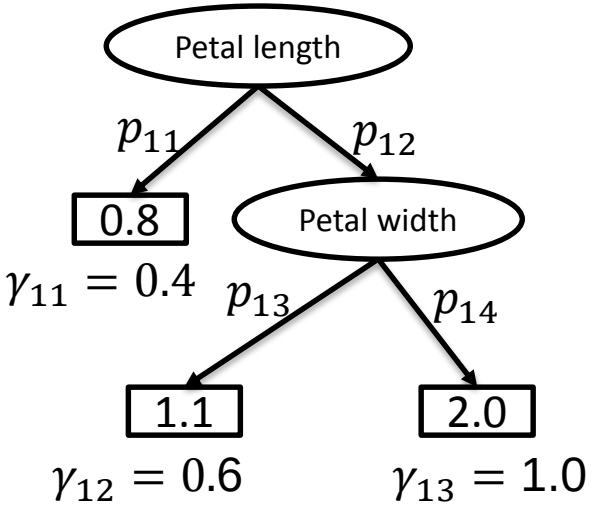


$r_2 = \gamma_{21}p_{21} + \gamma_{22}p_{22}p_{23} + \gamma_{23}p_{22}p_{24} = 0.5$

Fase de entrenamiento

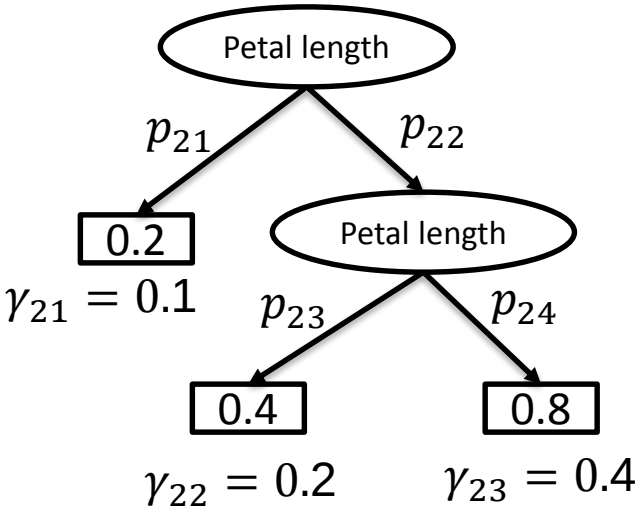
Se genera el tercer árbol según el nuevo residuo.

Residuos [0,0,1]



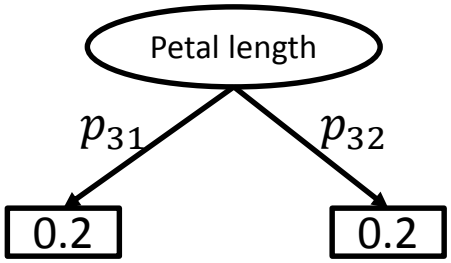
$r_1 = 0.4$

Residuos [0,0,0.6]



$r_2 = 0.5$

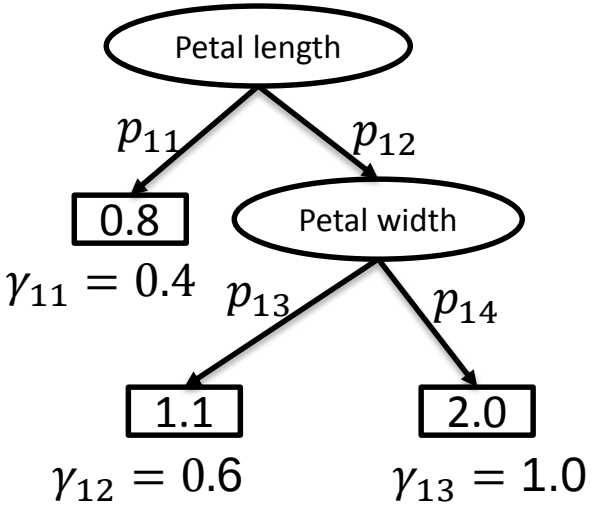
Residuos [0,0,0.1]



Fase de entrenamiento

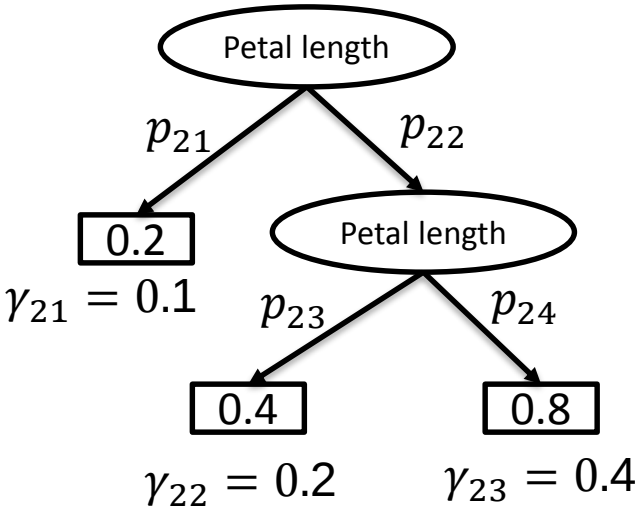
Se genera el tercer árbol según el nuevo residuo.

Residuos [0,0,1]



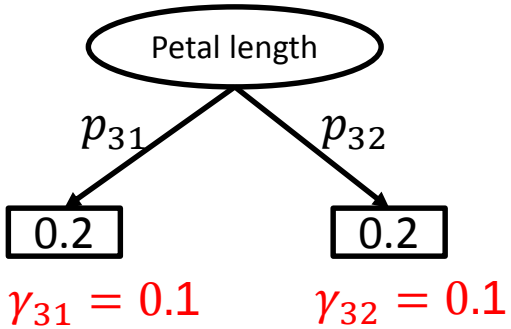
$r_1 = 0.4$

Residuos [0,0,0.6]



$r_2 = 0.5$

Residuos [0,0,0.1]



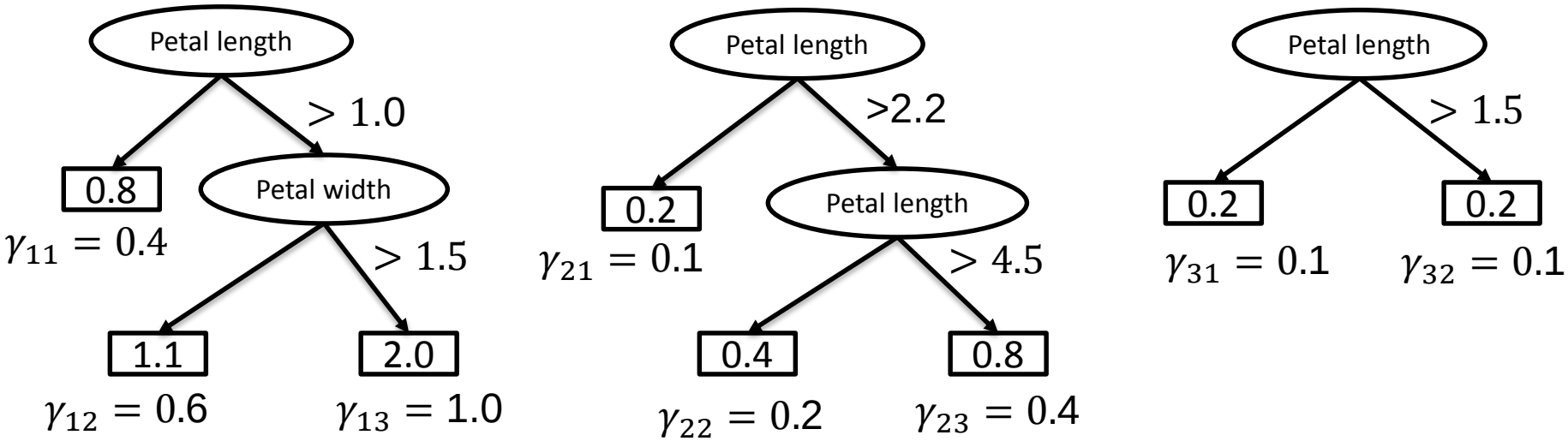
$r_3 = 0.001$

Fase de predicción

Sepal length	Sepal width	Petal length	Petal width	Clase
3	2	3	3	?

Tabla 3.2: Fase de predicción de *GBM*: Nueva instancia desconocido.

Ejemplo de predicción (en este ejemplo hay tres árboles por clase).



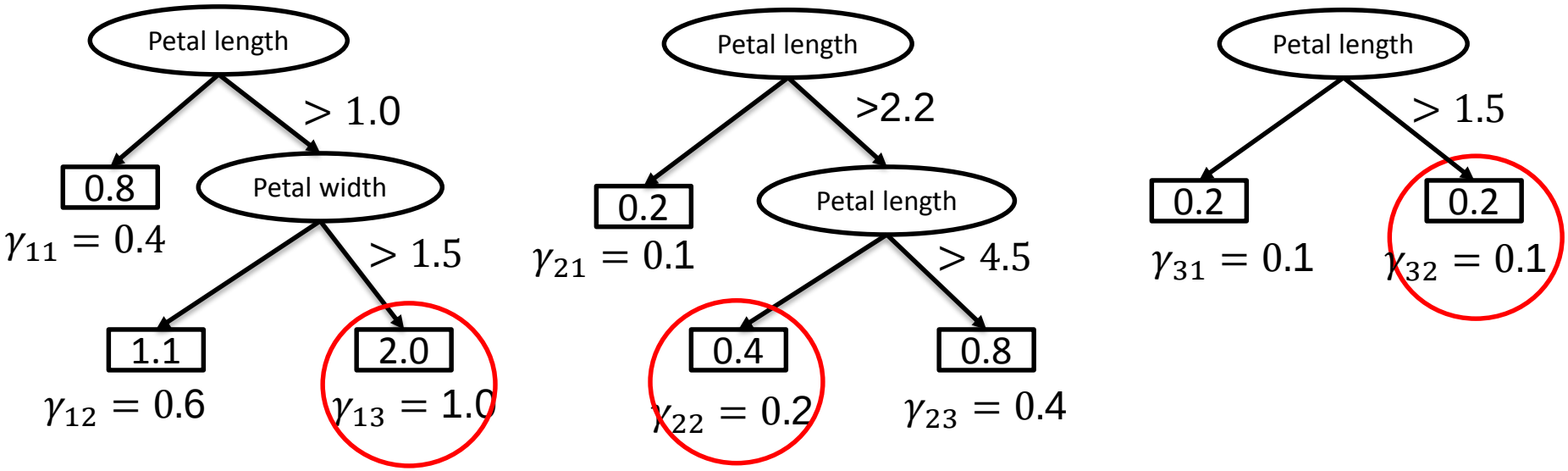
Árboles generados para la clase *Iris-Virginica*.

Fase de predicción

Sepal length	Sepal width	Petal length	Petal width	Clase
3	2	3	3	?

Tabla 3.2: Fase de predicción de GBM.

Se calcula la predicción para cada uno de los tres árboles por clase.



$$\gamma_{acumulada} = \gamma_{13} + \gamma_{22} + \gamma_{32} = 1.3$$

Fase de predicción

Softmax es una función matemática que convierte un número arbitrario a una probabilidad.

Clase	$\gamma_{acumulada}$	*Softmax*
Iris-Setosa	1.1	0.1
Iris-Versicolor	12.2	0.7
Iris-Virginica	1.3	0.2

Tabla 3.3: Resultados de las predicciones para las tres clases.

Sepal length	Sepal width	Petal length	Petal width	Clase
3	2	3	3	*Iris-Versicolor*

Tabla 3.4: Resultado de la predicción.

Introducción

Árboles de decisión

Sistemas clasificadores avanzados

Experimentación

Modelado del rendimiento académico de los alumnos

Tiempo dedicado

Conclusiones

Líneas de futuro

Matriz de confusión

- La **matriz de confusión** es una herramienta que indica que clases se están confundiendo.

	Iris-Setosa (Clasificado)	Iris-Versicolor (Clasificado)
Iris-Setosa (Pertenece) (P)	49 (VP)	1 (FP)
Iris-Versicolor (Pertenece) (N)	6 (FN)	44 (VN)

Tabla 4.1: Matriz de confusión, fragmento del *data set* de *Iris* con 100 instancias. Cada columna representa el número de predicciones de cada clase, mientras que cada fila representa a las instancias en la clase real.

- Existen muchas medidas estadísticas para conocer el acierto del clasificador. La más común es la ***puntuación F*** o *F Measure (F1)* [García et al., 2008].

$$F1 = \frac{2VP}{2VP + FP + FN}$$

Validación cruzada y estadística

- La **validación cruzada** es una técnica estadística que permite evaluar la validez de un modelo, herramienta ***DCoL***.
  - ***stratified 10 fold cross validation***: parte un *data set* de entrada en 20 sub *data sets* donde la mitad se usan para entrenar y la otra mitad para predecir. Cada *fold* mantiene la misma proporción de instancias que el fichero original.

- La **validación estadística** es una técnica que genera un *ranking* y mide si hay diferencias significativas entre los algoritmos, herramienta ***SCI2S multiple test***.

Comparativa con las técnicas tradicionales

- Para llevar a cabo la experimentación:
  - Se ha hallado una configuración que en general maximiza la precisión de los algoritmos de la comparativa.
  - Se han elegido varios problemas de los repositorios *KEEL* [Alcalá-Fdez et al., 2011] y *UCI* [Bache and Lichman, 2013]. Se ha hallado la configuración que maximiza la precisión para cada uno de los algoritmos de la comparativa.

Data set	**GBM**	**IBK**	**SMO**	**J48**	**NB**
Auto Imports Database	**84.48**	66.95	68.95	84.05	55.45
BUPA Liver Disorders	**69.66**	62.50	42.85	65.85	54.25
Glass Identification Database	**70.16**	69.40	52.45	66.45	45.55
Heart Disease Data Set	77.51	81.85	82.90	74.05	**83.15**
Statlog (Heart) Data Set	79.15	78.35	83.45	78.80	**83.85**
Microcalcifications	**66.37**	65.60	65.40	61.85	64.85
The Monk's Problems	79.42	80.30	53.90	**88.40**	53.85
Pima Indians Diabetes Data Set	**75.98**	73.55	75.80	73.25	75.80
Lymphography Domain	84.51	81.10	**87.05**	77.95	83.65
Sonar, Mines vs. Rocks	**84.26**	83.05	76.70	71.90	67.40
Tao	88.83	**96.40**	83.95	95.45	80.80
Vehicle Silhouettes	70.37	70.35	**72.90**	71.50	42.85
Deterding Vowel Recognition Data	80.81	**96.75**	71.00	79.75	64.00

Tabla 4.2: Comparativa de algoritmos. Muestra los *data sets* en los que se hace la comparación y los resultados de la **puntuación *F*** en los algoritmos *GBM*, *IBK*, *SMO*, *J48* y *NB*.

Introducción

Árboles de decisión

Sistemas clasificadores avanzados

Experimentación

Modelado del rendimiento académico de los alumnos

Tiempo dedicado

Conclusiones

Líneas de futuro

Procesado de los datos

- Los problemas del *UCI repository*, pese a ser problemas del mundo real, son utilizados como *benchmarks* y por ende el modelado de los datos de los datos no resulta excesivamente problemático.

Columna	Valores [Rango]
EXPEDIENTE	Numeral
EJERCICIO EC1	Numeral [0,10] y NP
EJERCICIO EC2	Numeral [0,10] y NP
MIDTERM	Numeral [0,10] y NP
EJERCICIO EC3	Numeral [0,10] y NP
EXAMEN FEBRERO	Numeral [0,10] y NP

Tabla 5.1: Atributos del *data set* original. Se trata de un *data set* **real** con las notas del alumnado en una asignatura. Muestra los atributos del *data set* y su rango de valores.

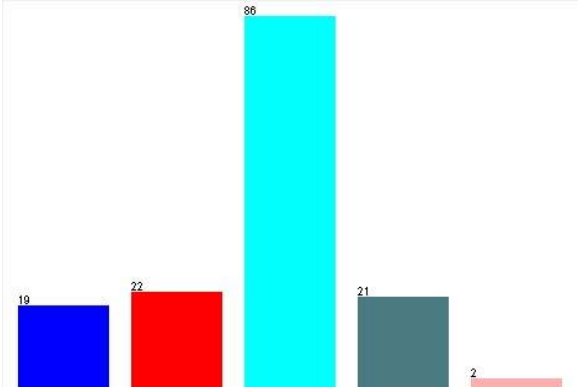


Figura 5.1: Diagrama de barras de la clase nominal del *data set* procesado. (Cyan: suspendido, azul marino: suficiente, rojo: bien, verde: notable y rosa: excelente)

- Observaciones
  - No tiene definido el atributo de salida.
  - Es necesario un procesado para las notas *NP* (problema de representación, solución: añadir ***dummies***).
  - Las clases están desbalanceadas.

Procesado de los datos

Columna	Valores [Rango]
EJERCICIO EC1	Numeral [0,10]
EJERCICIO EC2	Numeral [0,10]
MIDTERM	Numeral [0,10]
EJERCICIO EC3	Numeral [0,10]
EXAMEN FEBRERO	Numeral [0,10]
NP EC1	Numeral [0,1]
NP EC2	Numeral [0,1]
MIDTERM	Numeral [0,1]
NP EC3	Numeral [0,1]
NP EXAMEN FEBRERO	Numeral [0,1]
CLASS	Nominal [SUSPENDIDO, SUFICIENTE, BIEN, NOTABLE, EXCELENTE]

Tabla 5.2: *Data set* procesado (primer experimento). Muestra los atributos del *data set* procesado y su rango de valores. Se ha eliminado el atributo *Expediente*, se ha modificado el rango de valores de cada atributo, se ha añadido la clase (la nota final de la asignatura) y por último los atributos que mapean los valores *NP* (*dummies*).

Primer experimento

- Objetivos
  - **Configurar** *Gradient Boosting.*
  - **Comprobar la eficacia** de los *dummies*.

- Clase a predecir
  - Nota final de la asignatura, se trata de una combinación lineal de los otros atributos.
  - $C = EF \cdot 0.6 + M \cdot 0.2 + EC \cdot 0.2$

Parámetro	Valor
Mínimo número de instancias para generar nodos hoja	4
Máximo número de nodos hoja	0
Profundidad máxima	1
Varianza mínima	0.000001
Número de árboles	10
Ratio de aprendizaje	0.1

Tabla 5.3: Configuración *GBM* para modelar el rendimiento del alumnado.

Data set	GBM	IBK	SMO	J48	NB
Notas alumnos sin *dummies*	**70.19**	57.00	57.60	41.80	56.50
Notas alumnos con *dummies*	**86.98**	75.20	72.80	79.00	74.80

Tabla 5.4: Comparativa entre *GBM*, *IBK*, *SMO*, *J48* y *NB* en el *data set* docente con y sin *dummies*.

Segundo experimento

- Objetivo
  - **Modelar** el rendimiento académico del alumnado.

- Clase a predecir
  - Nota del examen de febrero.
  - ¡Problema muy complejo!

Data set	GBM	IBK	SMO	J48	NB
Examen Febrero con *dummies*	**75.99**	43.50	47.40	45.80	52.00

Tabla 5.5: Comparativa de precisiones según la *medida F* de los distintos algoritmos (*GBM*, *IBK*, *SMO*, *J48* y *NB*) en el segundo experimento.

Introducción

Árboles de decisión

Sistemas clasificadores avanzados

Experimentación

Modelado del rendimiento académico de los alumnos

Tiempo dedicado

Conclusiones

Líneas de futuro

Coste en horas

Fase	Tiempo dedicado (h)	Porcentaje (%)
Investigación	272	34.21
Diseño	131	16.48
Implementación	117	14.72
Documentación	205	25.79
Experimentación	70	8.80
Total	795	100

Tabla 6.1: Coste en horas del trabajo. Muestra la fase, el tiempo dedicado en horas y el porcentaje del proyecto en tanto por ciento.

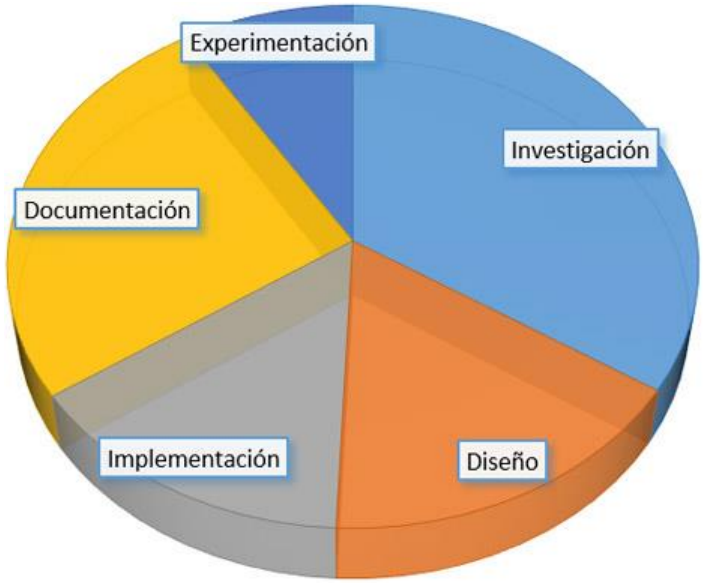


Figura 6.1: Desglose del tiempo dedicado en el presente Trabajo Final de Grado.

Introducción

Árboles de decisión

Sistemas clasificadores avanzados

Experimentación

Modelado del rendimiento académico de los alumnos

Tiempo dedicado

Conclusiones

Líneas de futuro

Conclusiones

- **Revisar las principales técnicas del Aprendizaje Automático.**
  - Existe un gran número de técnicas en el campo del Aprendizaje Automático y para cada una de ellas muchas variantes.
  - El estado del arte son los métodos *Ensemble*.
  - No hay una técnica mejor que otra para cualquier problema aunque los métodos *Ensemble* en general son más precisos.

- **Desarrollar un algoritmo *Gradient Boosting* y compararlo con otras técnicas.**
  - Elaborar una especificación de requisitos software y utilizar una metodología en espiral han resultado muy útiles de cara a diseñar el sistema.
  - Combinar *C++* con un sistema operativo *Debian* y las herramientas *GDB, Valgrind* y *Gprof* es clave para obtener un rendimiento elevado.
  - Automatización con Python ha resultado muy productiva.
  - En la experimentación realizada *Gradient Boosting* resulta el algoritmo con mejor *ranking*.

- **Aplicar *Gradient Boosting* a datos docentes con la finalidad de modelar al alumnado.**
  - En general el procesado de los datos es la parte principal del proceso *KDD*.
  - *Gradient Boosting* es una técnica muy robusta.
  - Se ha logrado modelar al alumnado con precision.

Conclusiones

Fortalezas	**Debilidades**
- Precisión elevada. - Rapidez en generar modelos. - Rapidez en predecir. - Relativamente interpretable. - Pocos parámetros configurables. - Paralelizable (para cada clase).	- Facilidad en obtener sobre ajuste. - Puede ocupar mucho espacio en memoria. debido a la gran cantidad de modelos que se pueden utilizar.
Oportunidades	**Amenazas**
- El algoritmo de particionado de variables se puede mejorar investigando otras técnicas - Se puede tener en cuenta el peso de cada atributo de manera simple. - La función *gamma* puede ser calculada con mayor precisión.	- Es un algoritmo complejo que puede resultar difícil de implementar y esto puede ahuyentar a los practicantes. - Existen algoritmos más simples que dan un resultado similar.

Tabla 7.1: Análisis *DAFO* de *Gradient Boosting*.

Introducción

Árboles de decisión

Sistemas clasificadores avanzados

Experimentación

Modelado del rendimiento académico de los alumnos

Tiempo dedicado

Conclusiones

Líneas de futuro

Líneas de futuro

- **Paralelizar** el algoritmo
  - librería *OpenMP*.
- Estudio de distintas técnicas para **particionar** las variables.
- Estudio de mecanismos de identificación de **pesos** distintos.
- Cálculo de la función ***gamma*** de una forma más eficaz.

Bibliografía

Tom Mitchell. *Machine Learning*. Prentice Hall, Pittsburgh, 1997. ISBN 0070428077

Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. *The kdd process for extracting useful knowledge from volumes of data*. Commun. ACM, 39(11):27–34, November 1996. ISSN 0001-0782. doi: 10.1145/240455.240464

Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey McLachlan, Angus Ng, Bing Liu, Philip Yu, Zhi-Hua Zhou, Michael Steinbach, David Hand, and Dan Steinberg. *Top 10 algorithms in data mining. Knowledge and Information Systems*, 14:1–37, January 2007. ISSN 0219-1377

L. Breiman, J. Friedman, R. Olshen, and C. Stone. Classification and Regression Trees. Wadsworth and Brooks, Monterey, CA, 1984. ISBN 0534980538

Bibliografía

G. James, D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning: with Applications in R. Springer Texts in Statistics*. Springer, 2013. ISBN 9781461471387

Jerome H. Friedman. *Greedy function approximation: A gradient boosting machine*. Annals of Statistics, 29:1189–1232, 2000.

S. García, A. Fernández, J. Luengo y F. Herrera. *A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability*. Soft Computing, 13(10): 959-977, 2009.

Stephen K. Bache and Moshe Lichman. *UCI machine learning repository*, 2013. URL http://archive.ics.uci.edu/ml.

TFG
Beyond Traditional Frameworks for Modelling Learning Domains

Sergio Sancho Asensio
Director: Xavier Solé Beteta

Enginyeria i Arquitectura La Salle – Universitat Ramon Llull
Quatre camins 2, 08022 Barcelona
ls24029@salleurl.edu

Julio 2015