

EARTHQUAKE DAMAGE PREDICTION

Minería de Datos: Preprocesamiento y Clasificación (2020/2021)

Andrea Morales Garzón
andreamgmq@correo.ugr.es

Ithiel Piñero Darías
ithiel@correo.ugr.es

Paula Villa Martín
pvilla@correo.ugr.es

Antonio Manjavacas Lucas
manjavacas@correo.ugr.es

ÍNDICE DE CONTENIDOS

1. Definición del problema
2. Análisis y preprocesamiento de datos
3. Clasificación
 - 3.1. Regresión logística - [Andrea Morales Garzón \(andreamorgar\)](#)
 - 3.2. Árboles de decisión - [Ithiel Piñero Darías \(IthielPD\)](#)
 - 3.3. Reglas - [Antonio Manjavacas Lucas \(manjavacas\)](#)
 - 3.4. SVM - [Paula Villa Martín \(PaulaVillaMartin\)](#)
4. Conclusiones



1. Definición del problema

Predicción del nivel de daño provocado por el terremoto Gorkha de 2015 sobre edificios en Nepal.

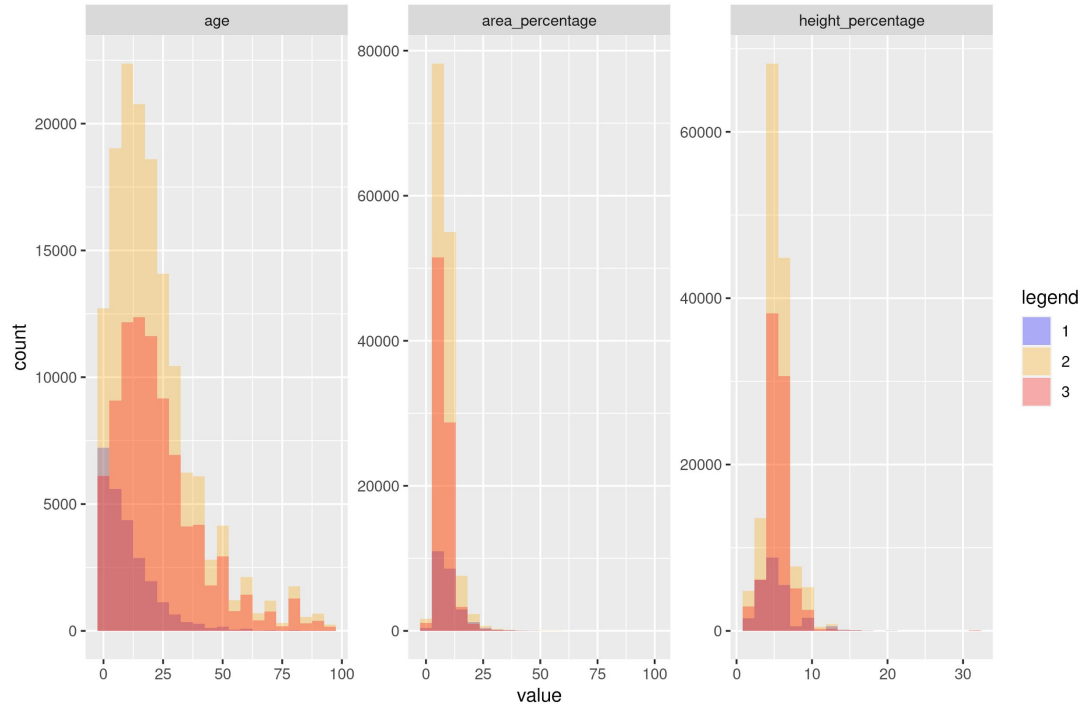
Trataremos de predecir la variable ordinal **damage_grade**, que representa el nivel de daño provocado sobre los edificios afectados por el terremoto:

- **damage_grade = 1** representa un daño bajo;
- **damage_grade = 2** representa un daño medio;
- **damage_grade = 3** representa una destrucción del edificio casi completa.

<https://www.drivendata.org/competitions/57/nepal-earthquake/page/136/>

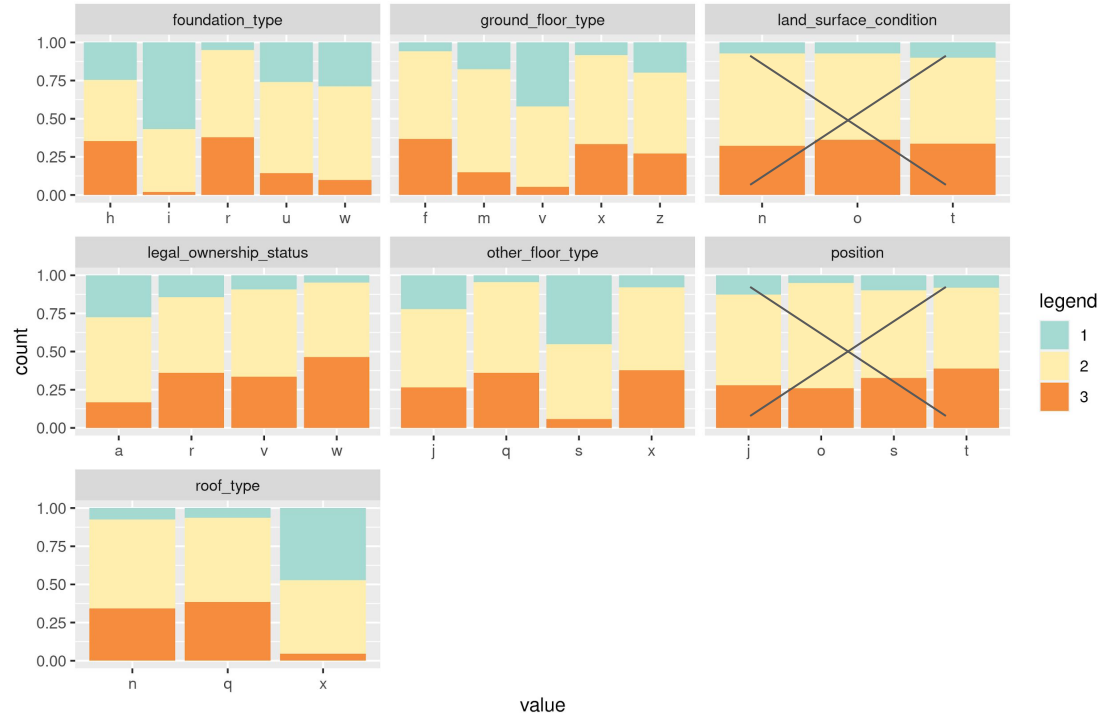


2. Análisis y preprocesamiento de datos



Las **distribuciones son diferentes** según el grado de daño
(Test de Kolmogorov-Smirnov)

2. Análisis y preprocesamiento de datos

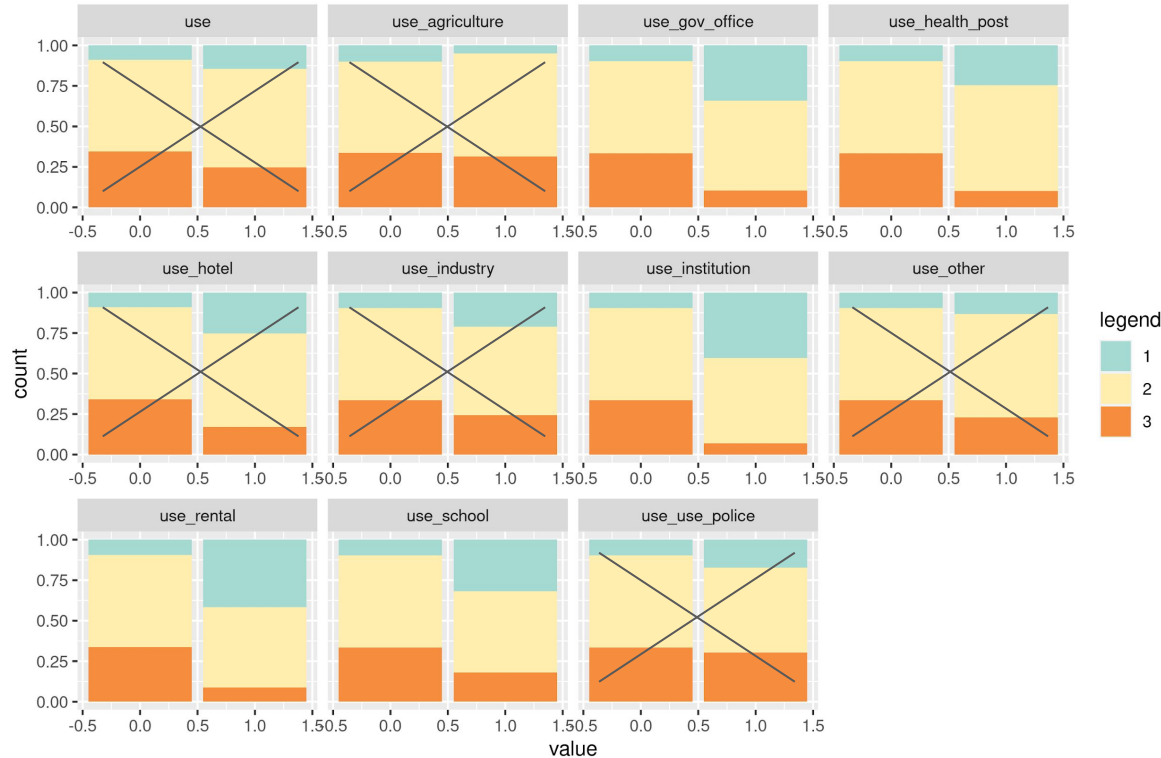


Se pueden **agrupar categorías** con patrones **similares** en grado de daño.

Se pueden **eliminar variables** con patrones iguales para todas las categorías:

land_surface_condition,
position

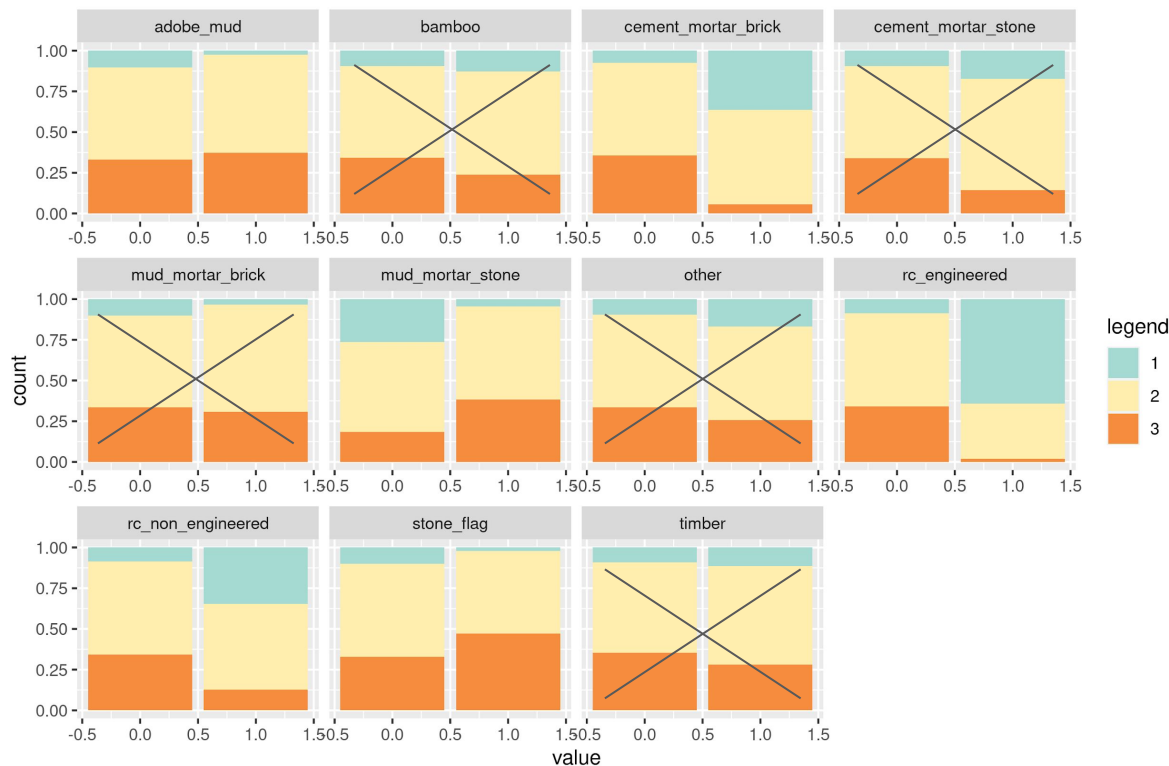
2. Análisis y preprocesamiento de datos



Se pueden
eliminar categorías
con patrones similares
en grado de daño:

**use,
police,
other,
agriculture,
hotel,
industry**

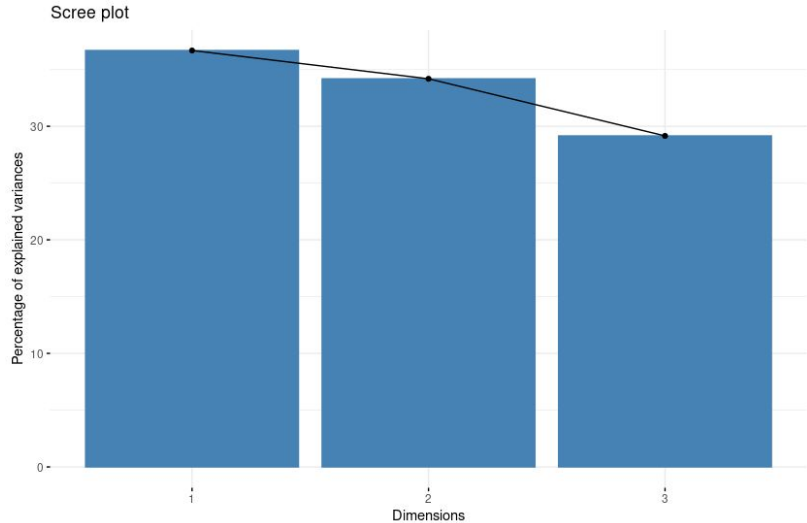
2. Análisis y preprocesamiento de datos



Se pueden
eliminar categorías
con patrones similares
en grado de daño:
timber,
bamboo,
other,
cement_mortar_stone,
mud_mortar_brick

Se pueden
agrupar categorías
en “robust” y “weak”

2. Análisis y preprocesamiento de datos



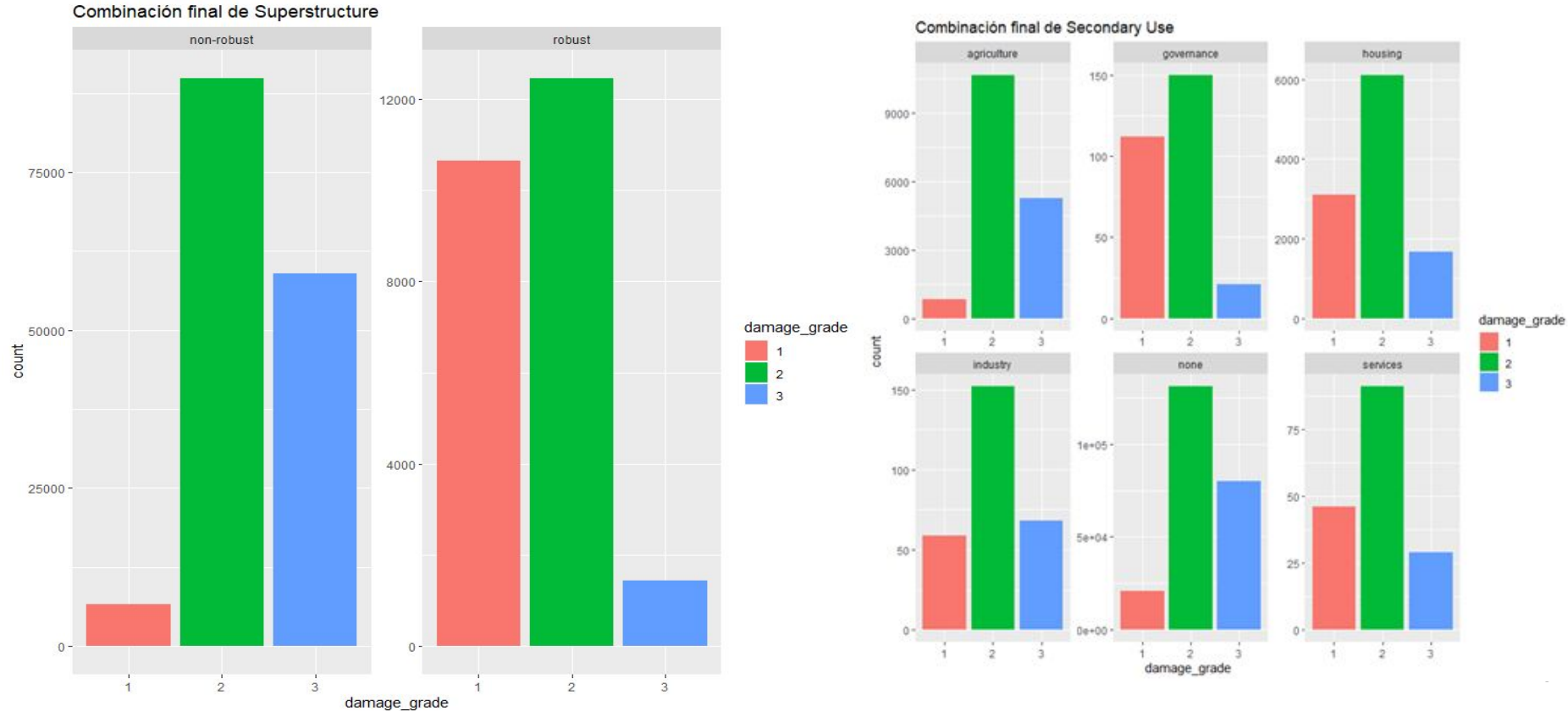
El análisis de PCA indica que:

Las componentes principales (constituídas por geolevel 1,2, y 3) explican entorno a un 30% de la varianza cada una



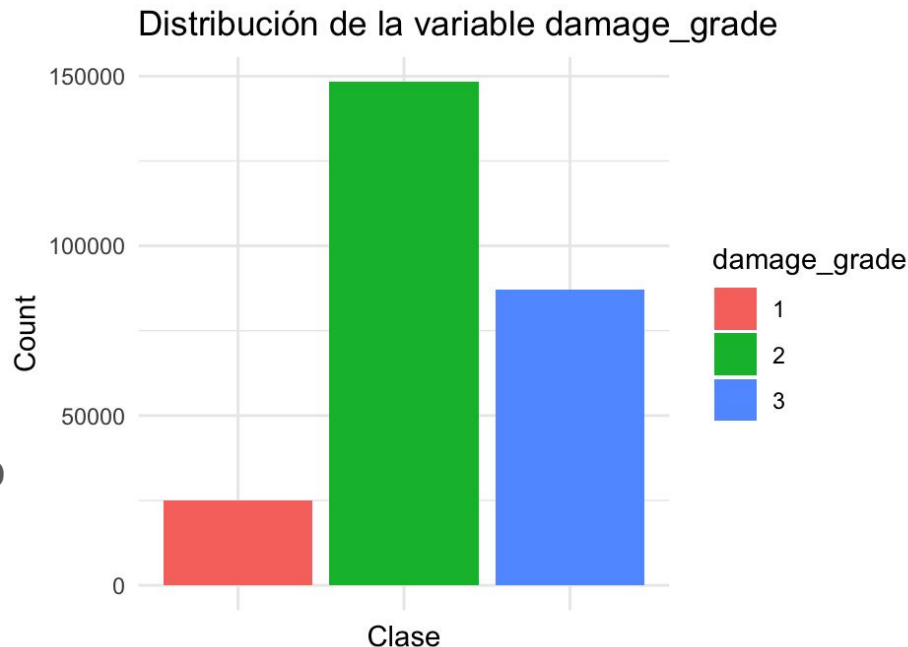
No parece haber ningún patrón o clustering entre estas componentes principales

2. Análisis y preprocesamiento de datos



3. Clasificación

- Problema de clasificación con las clases desbalanceadas.
- Técnicas para balanceo de clases
 - **Random Undersampling**
 - **SMOTE** (con variables categóricas)
- Mejora/Empeoramiento de los resultados dependiendo del algoritmo de clasificación utilizado



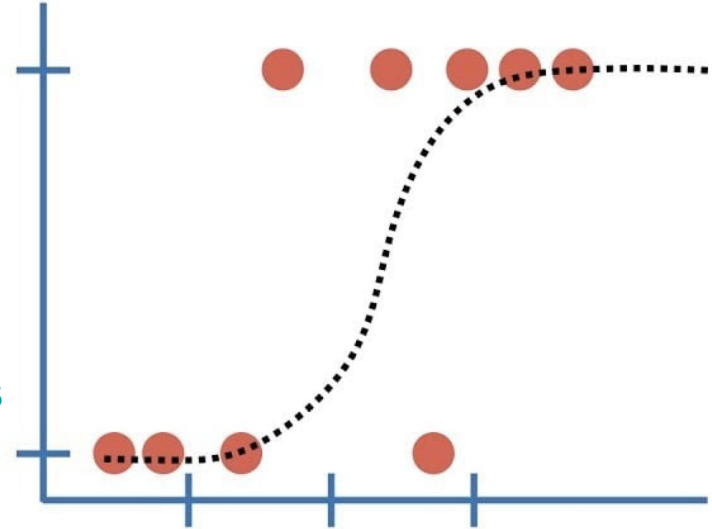
3. Problema detectado

- **Problema general detectado**: la dificultad está principalmente al clasificar la clase 2
- **Solución propuesta**: entrenar de forma **personalizada** la clase 2 contra el resto, y la clase 1 contra la clase 3 → **descomposición en dos problemas binarios**



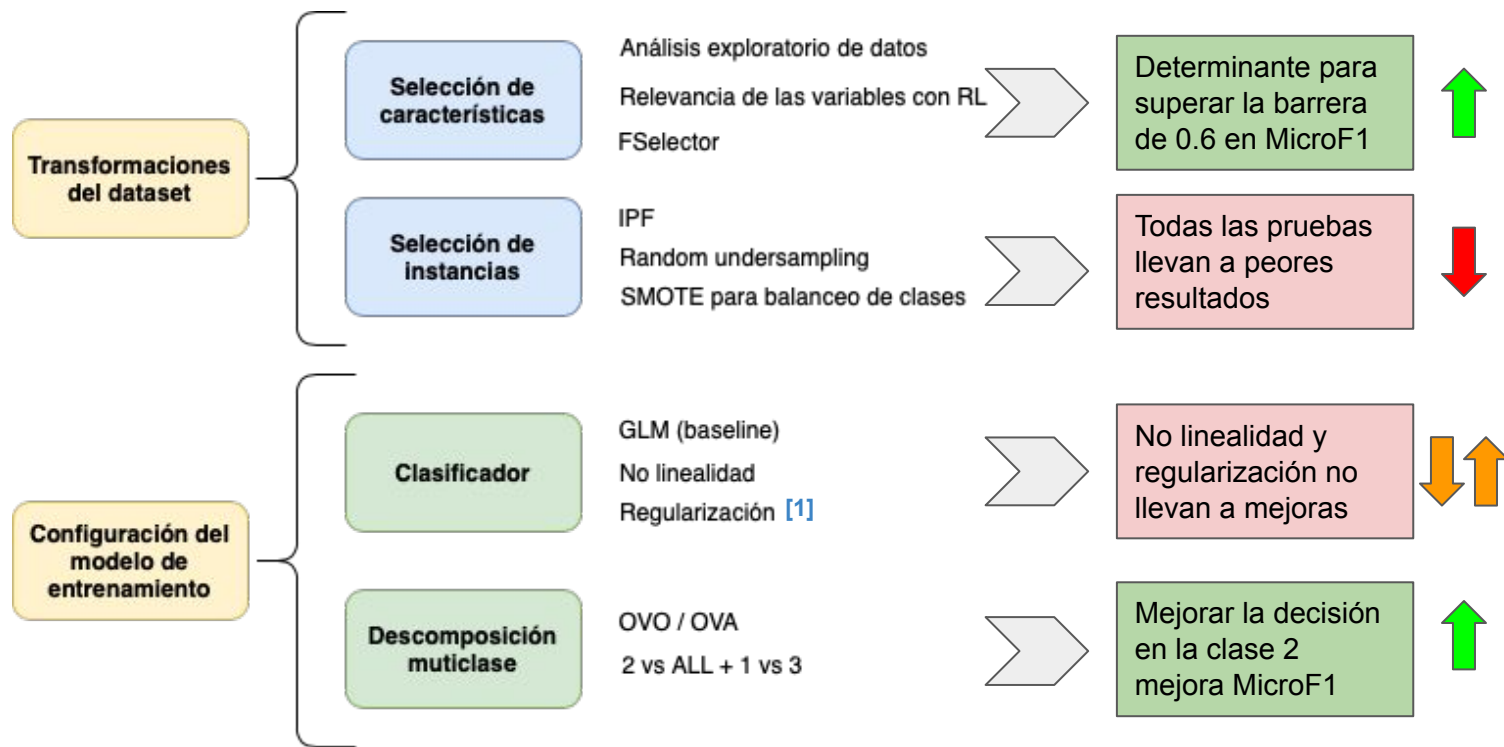
3.1. Clasificación: Regresión Logística

- La regresión logística es un algoritmo de clasificación sencillo e interpretable
- **Ventaja:** nos permite ver qué factores tienen más peso en la clasificación.
- Para cada muestra proporciona un valor entre 0 y 1, que se interpreta como probabilidades. **En datasets desbalanceados 0.5 no suele ser un buen discriminador entre clases [2].**



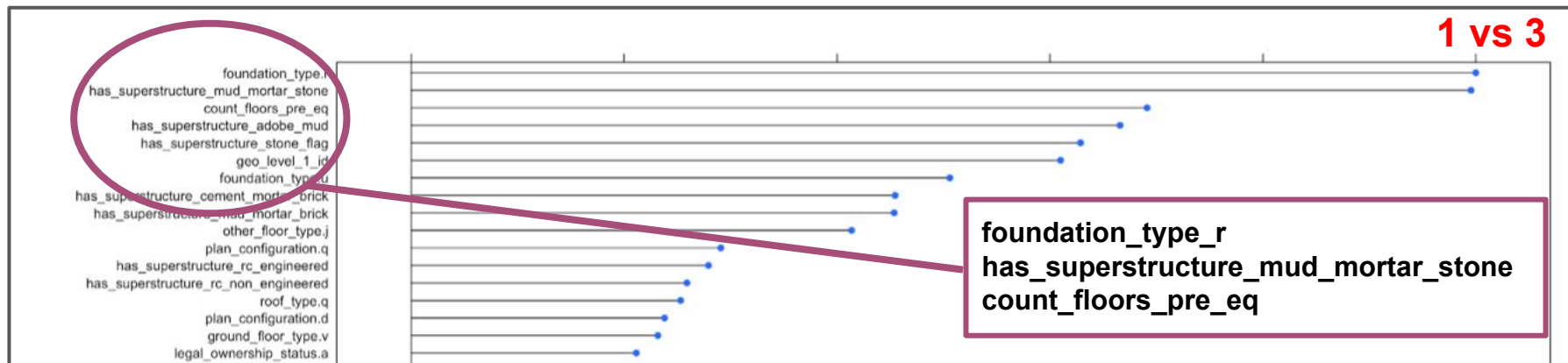
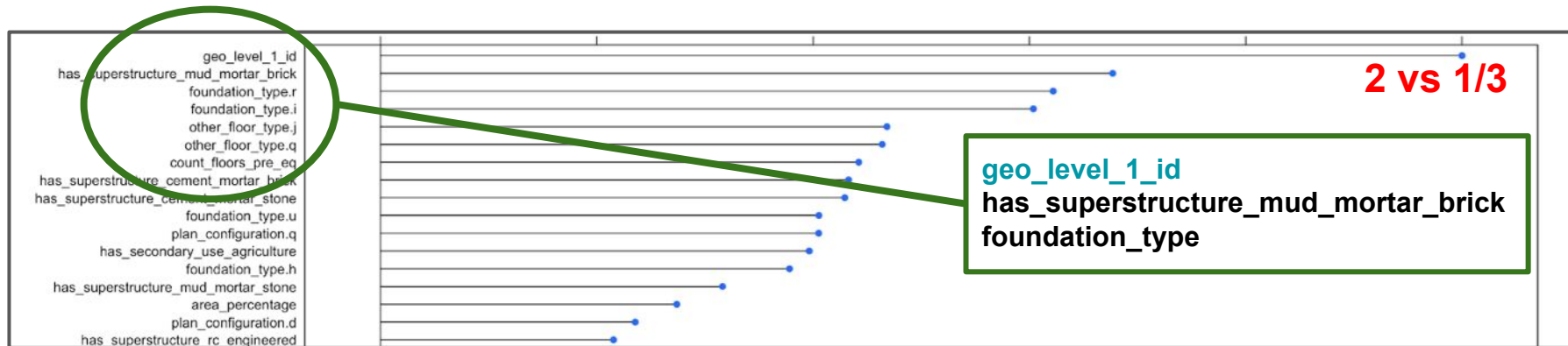
[2] Provost, F. (2000, July). Machine learning from imbalanced data sets 101. In *Proceedings of the AAAI'2000 workshop on imbalanced data sets* (Vol. 68, No. 2000, pp. 1-3). AAAI Press.

3.1. Clasificación: Regresión Logística **Experimentación**



3.1. Clasificación: Regresión Logística

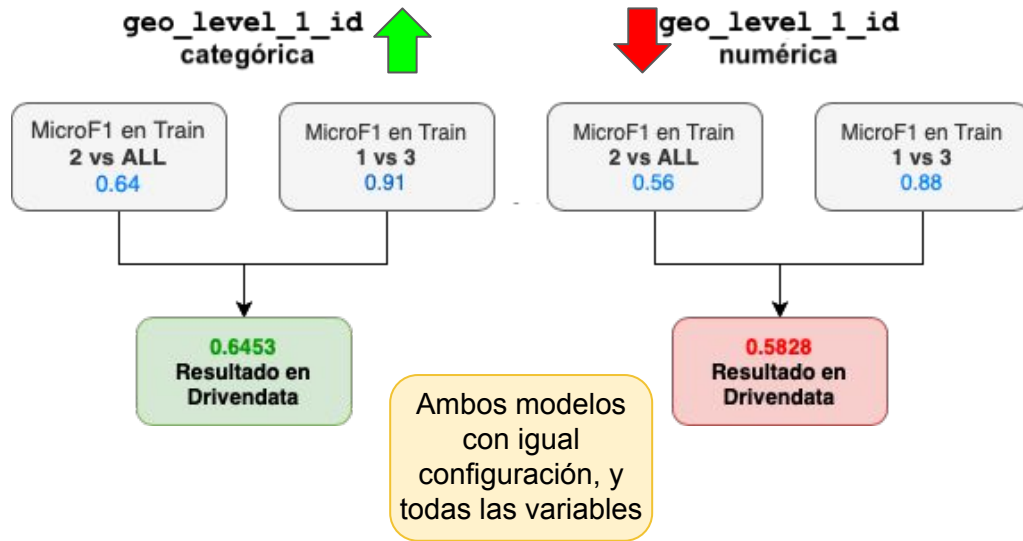
Relevancia de las variables



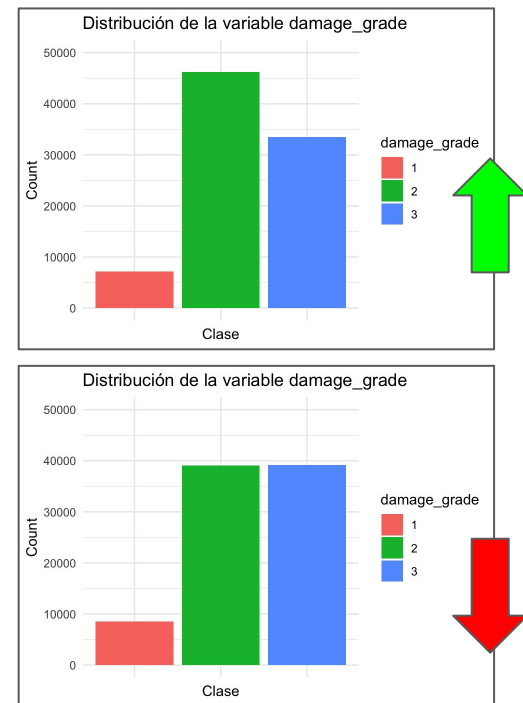
3.1. Clasificación: Regresión Logística

Pruebas

Interpretación de la variable
geo_level_1




Ajuste del umbral de decisión para tratar con desbalanceo



3.1. Clasificación: Regresión Logística **Modelos OVA-OVO**

	Características	MicroF1
1	A. 2vsALL : Seleccionar variables con FSelector B. 1vs3 : Todas las variables	0.6336
2	A. 2vsALL : Seleccionar variables con FSelector B. 1vs3 : Seleccionar variables con FSelector	0.6337
3	Selección de variables resultante del EDA (mejor modelo)	0.6453
4	Modelo anterior añadiendo no linealidad	0.6449



En regresión, cada variable utilizada tiene un peso con su importancia en el modelo final → si quitamos variables, no tenemos por qué mejorar. **Solución: mejorar la calidad de las variables**

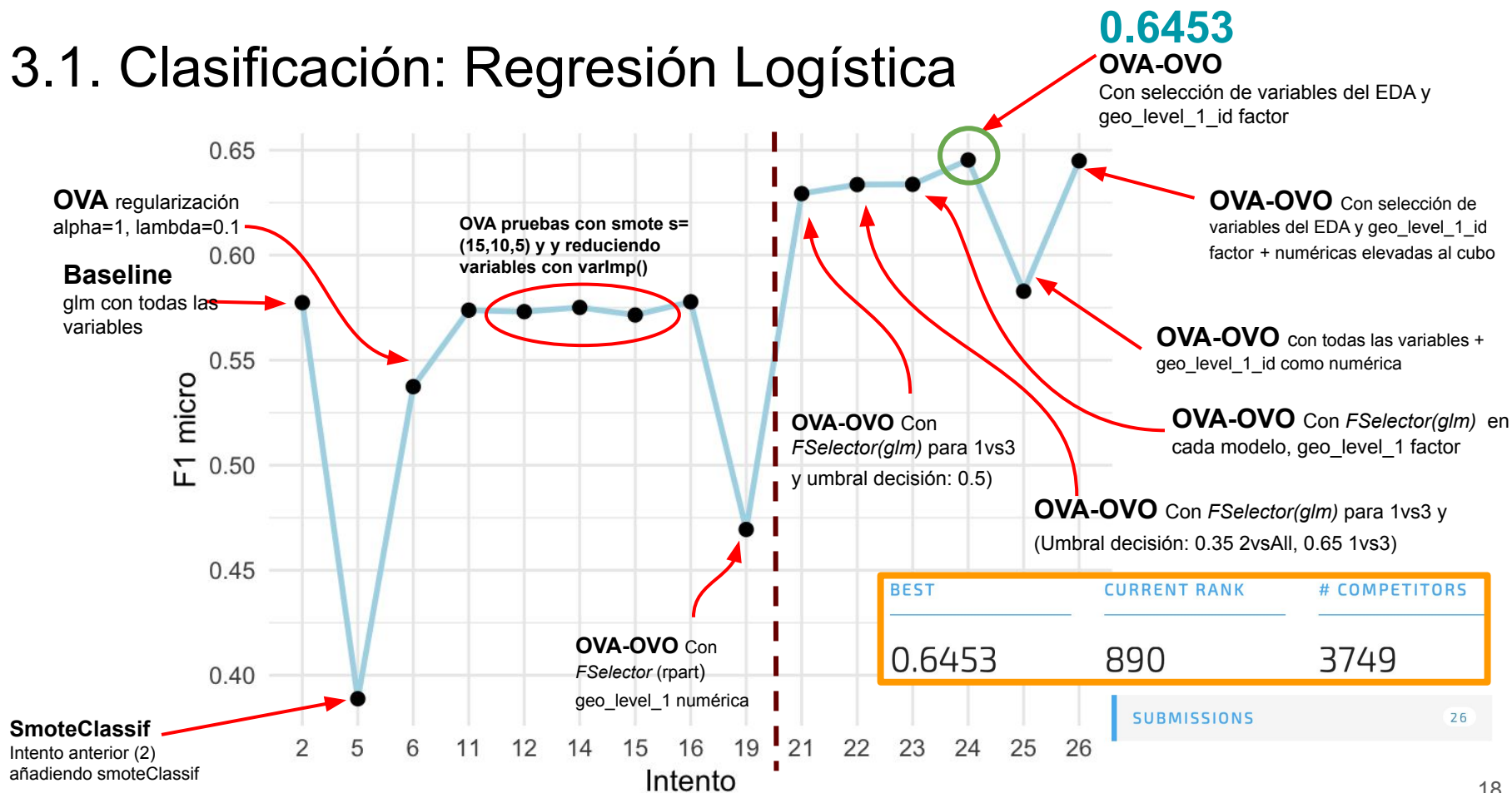
3.1. Clasificación: Regresión Logística Mejor modelo

Las variables que mejor funcionan en el modelo se obtienen a partir del EDA:

- `geo_level_1_id` → como factor
 - `geo_level_2_id` → como numérica
 - `geo_level_3_id` → como numérica

 - `foundation_type`
 - `roof_type`
 - `ground_floor_type`
 - `plan_configuration`
 - `secondary_use`
- } agrupación de los factores de las variables
- `count_floors_pre_eq`
 - `age`
 - `area_percentage`
 - `height_percentage`
 - `land_surface_condition`
 - `position`
 - `legal_ownership_status`
 - `count_families`
-
- `superstructure` → convertida en una variable binaria
 - **no-robust:** `adobe_mud`, `mud_mortar_brick`, `mud_mortar_stone`, `stone_flag`
 - **robust:** resto

3.1. Clasificación: Regresión Logística



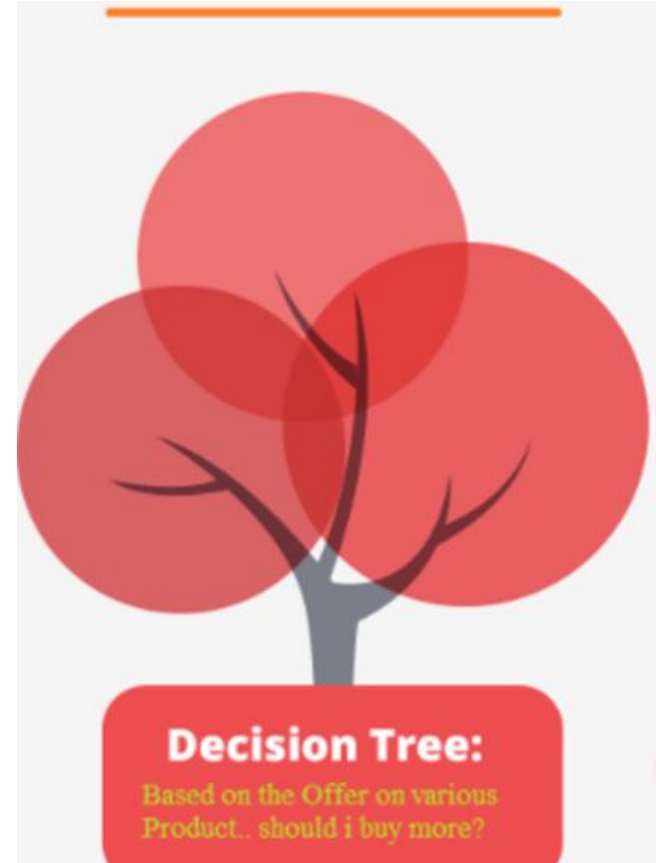
3.1. Clasificación: Regresión Logística

Conclusiones

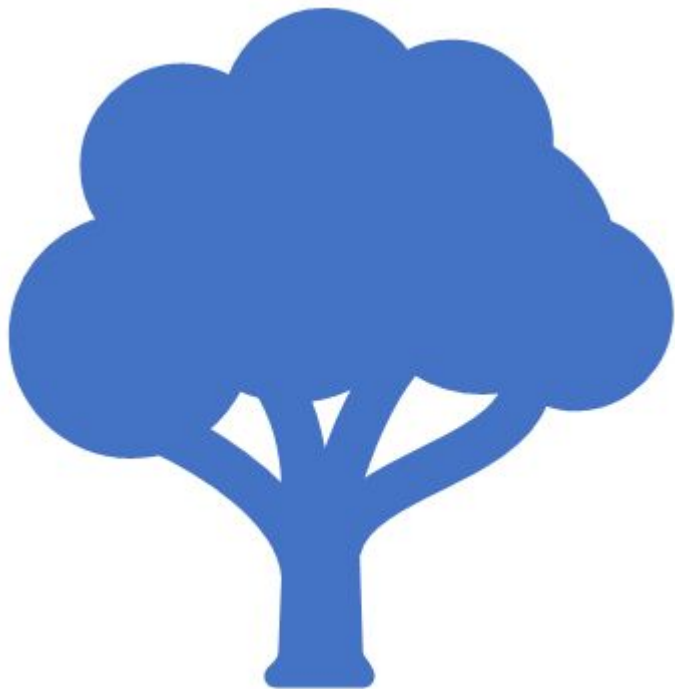
- Utilizar `geo_level_1_id` como variable categórica permite mejorar resultados.
- El modelo **2 vs ALL** es el más complejo de definir y conseguimos mejorar con su ajuste.
- El modelo selecciona internamente variables (con los pesos), por lo que **la mejor solución se alcanza al mejorar las variables con agrupaciones en base al EDA.**
- Modificar el umbral de decisión de los modelos mejora los resultados.
- No se consigue mejorar resultados con regularización, no-linealidad, undersampling
- A pesar de ello, los resultados con regresión logística no son los mejores:
 - No funciona bien con espacio de características muy grande.
 - No es capaz de obtener relaciones complejas entre variables.
- **Posibles mejoras:** usar regresión logística con kernel.

3.2. Clasificación: Árboles de decisión

- **C4.5**, extensión de ID3 (Iterative Dichotomiser 3).
- Algoritmo de generación de árboles de decisión.
- Propuesto por Ross Quinlan en 1993.
- J48 es la implementación opensource.
- Disponible tanto en RWeka como en Caret.
- Disponible mejora con C5.0



3.2. Clasificación: Árboles de decisión



- **Selección de características** según la ganancia de información del árbol (pérdida de entropía).
- **Selección de instancias**
 - SMOTE
 - Undersampling
 - Oversampling
- **Mejor resultado** obtenido con Random Undersampling hasta 180.000 instancias.
- **Selección de hiperparámetros:**
 - C, confidence factor.
 - M, minimum number of instances per leaf.

3.2. Clasificación: Árboles de decisión

	1 VS 2	1 VS 3	2 VS 3	1 VS ALL	2 VS ALL	3 VS ALL
<u>Accuracy</u>	0.8851	0.9262	0.7717	0.9221	0.7185	0.7919
MicroF1	0.5066	0.8337	0.8283	0.9579	0.6348	0.8515

Se comprueba:

- 1vsA + 3vsA
- 2vsA + 1vs3 (Mejor resultado obtenido entre todas)
- 1vsA + 2vs3
- 1vs3 + 1vs2 + 2vs3

3.2. Clasificación: Árboles de decisión

Variables más relevantes modelo único vs OVA+OVO (umbral 0.1 en Gain Information):

Variables modelo único
Geo_level_1_id
Geo_level_2_id
Geo_level_3_id
Foundation_type
Superstructure

2 vs All	1 vs 3
geo_level_1_id	geo_level_1_id
geo_level_2_id	geo_level_2_id
geo_level_3_id	geo_level_3_id
cluster	cluster
	foundation_type
	roof_type
	ground_floor_type
	other_floor_type
	superstructure

3.2. Clasificación: Árboles de decisión

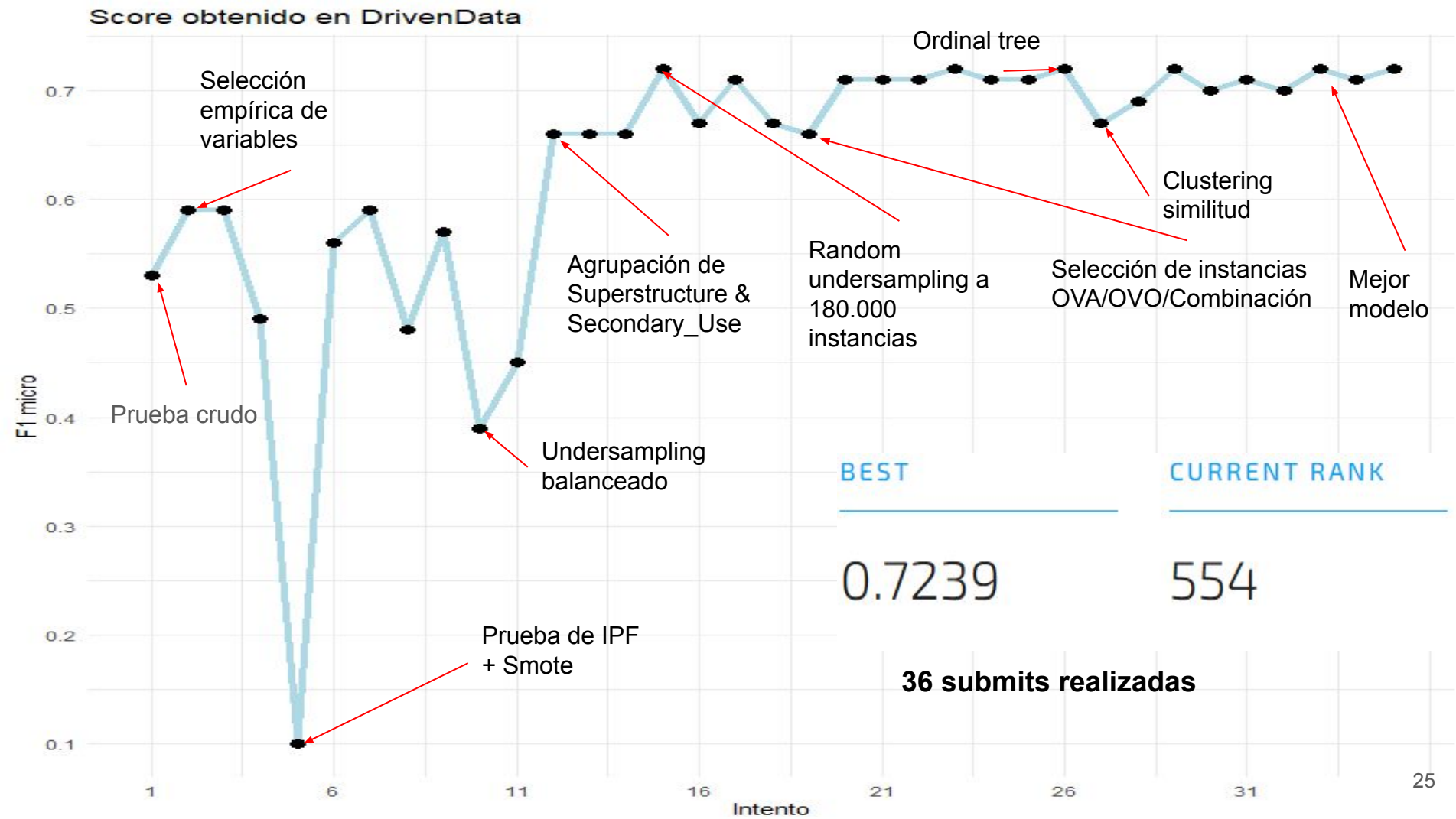
Mejor resultado obtenido con Random Undersampling a 180.000 instancias con todas las variables. $C = 0.1$ & $M = 25$

0.7239	lthielPD 	2021-01-29 00:03:35 UTC
--------	--	-------------------------

Segundo mejor modelo obtenido 2VAll+1vs3+Clustering por similitud. $C = 0.1$ & $M = 25$

0.7194	lthielPD 	2021-01-26 12:02:07 UTC
--------	--	-------------------------

CV con 10 folds en ambos modelos.



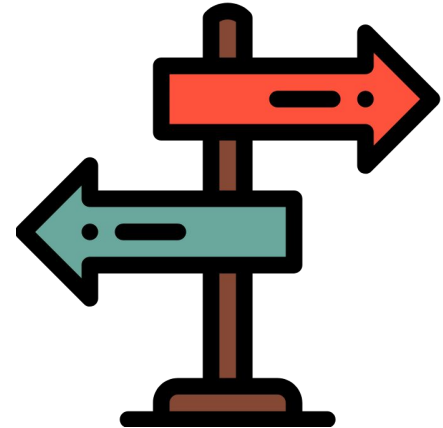
3.2. Clasificación: Árboles de decisión

•**Conclusiones:**

- Mejora con división del dataset mediante estrategias de OVA y OVO.
- No existe presencia de ruido dentro de cada clase.
- Se producen mejores resultados sin aplicar técnicas de corrección de balanceo, conllevan overfitting.
- Problemas con variables con muchos levels (desbordamiento de pila en ejecución).
- Mayor dificultad: diferenciar entre la clase 1 y 2 (0.5 en microF1).
- Posibles mejoras: exploración de nuevas features selections en ajustadas a divisiones OVA u OVO.

3.3. Clasificación: Reglas

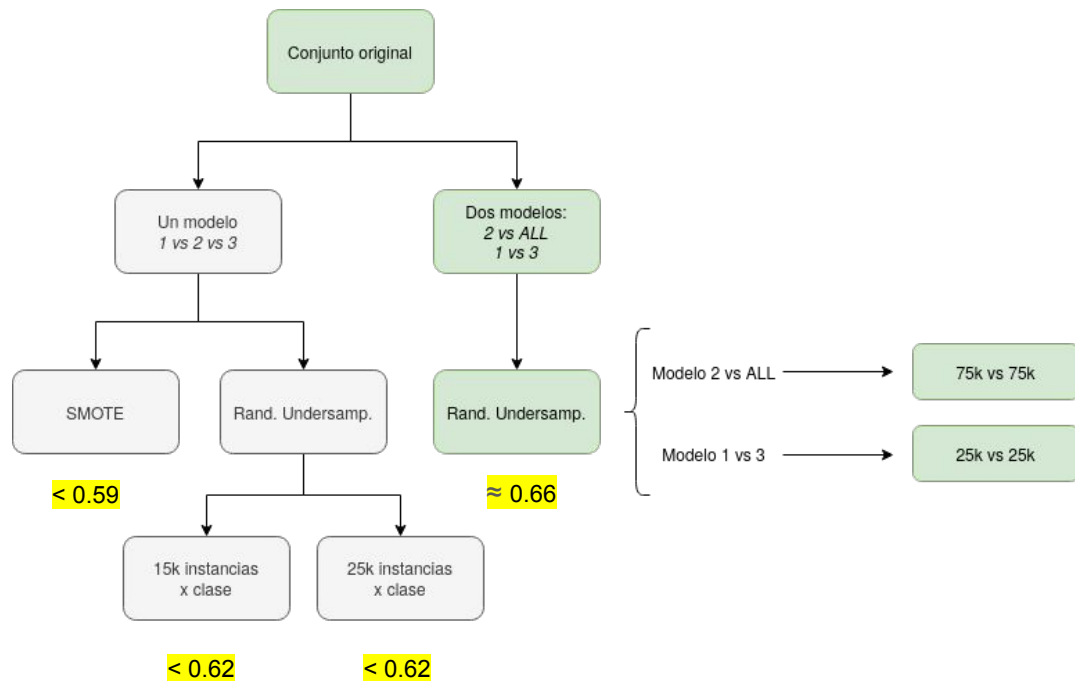
- **RIPPER** (Repeated Incremental Pruning to Produce Error Reduction).
- Algoritmo de clasificación basado en reglas.
- Propuesto por W. Cohen en 1995.
- Mejora el rendimiento con respecto a las propuestas que lo precedieron: *REP* e *IREP*.
- Utilizaremos la implementación disponible en la librería *RWeka*: **JRip**.
 - Menor tiempo de ejecución que *RKEEL*.



3.3. Clasificación: Reglas

SELECCIÓN DE INSTANCIAS

- Limitación: instancias permitidas por JRip (no más de $\approx 140k$ instancias por modelo)
- Selección de instancias:
 - **SMOTE**
 - **Random Undersampling**



3.3. Clasificación: Reglas

SELECCIÓN DE CARACTERÍSTICAS

Modelo 2 vs ALL	Modelo 1 vs 3
<pre>geo_level_1_id geo_level_2_id geo_level_3_id count_floors_pre_eq age area_percentage height_percentage land_surface_condition foundation_type roof_type ground_floor_type other_floor_type position plan_configuration legal_ownership_status count_families superstructure secondary_use</pre>	<p>Variables con <i>InfoGain</i> ≥ 0.1</p> <pre>geo_level_1_id geo_level_2_id geo_level_3_id foundation_type roof_type ground_floor_type other_floor_type superstructure</pre>

3.3. Clasificación: Reglas

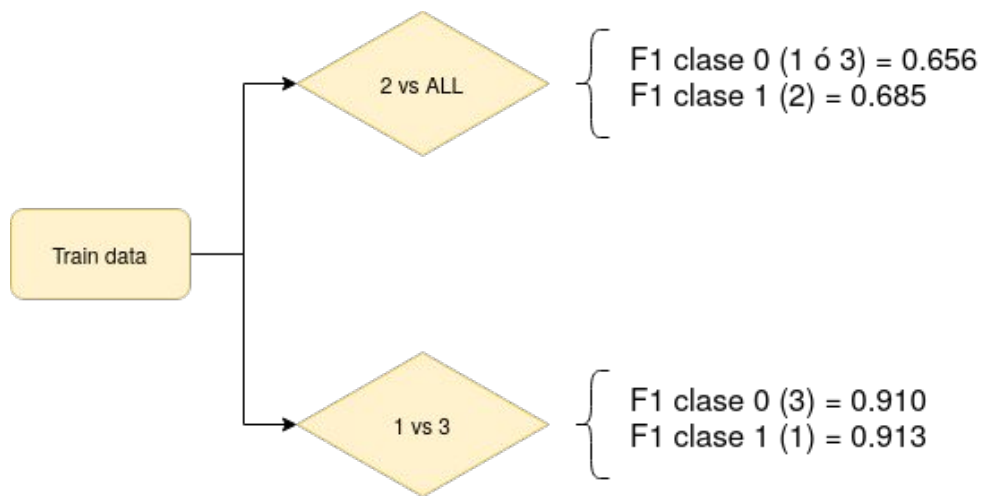
Mejor modelo: **0.6657**

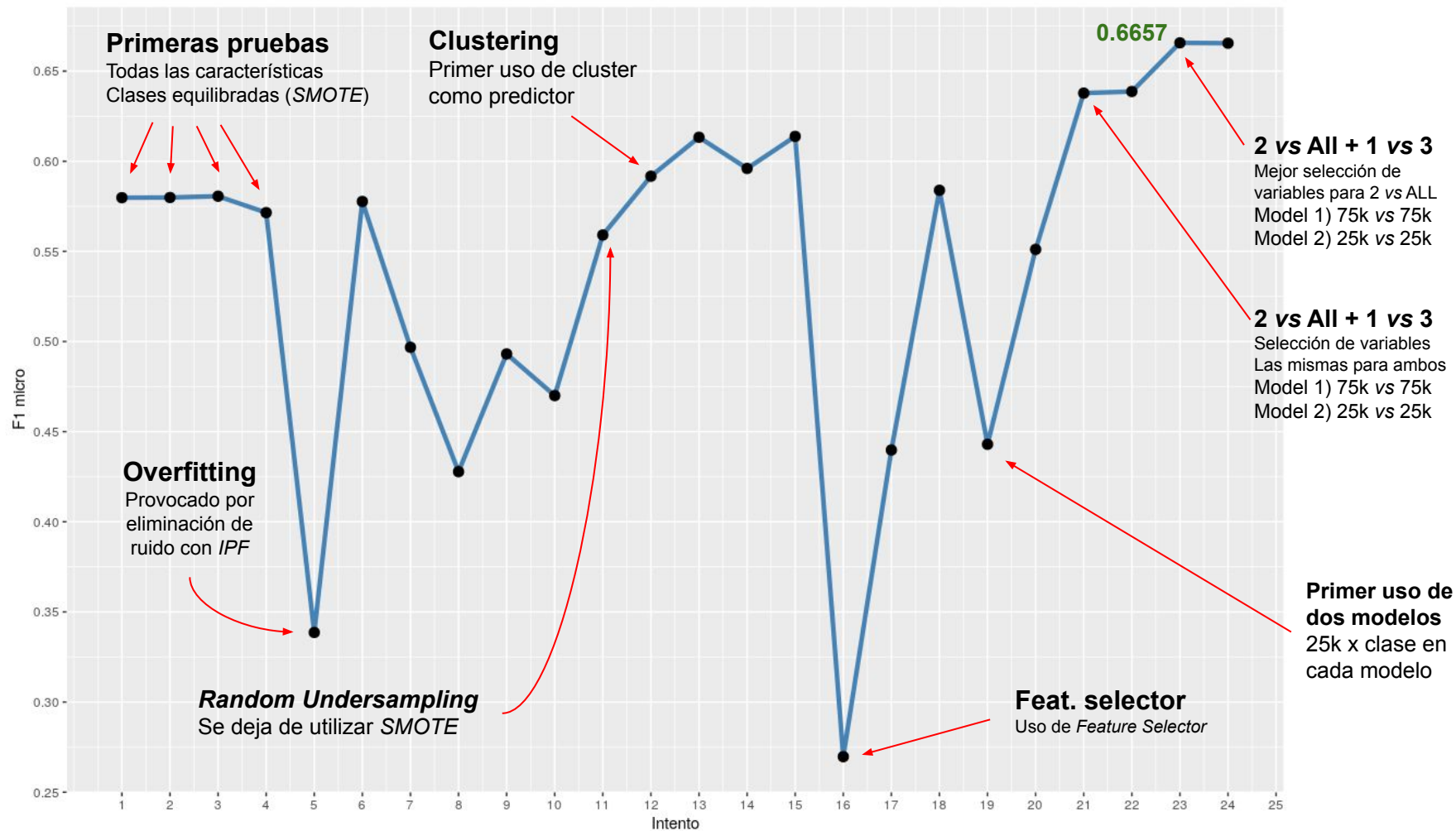
Ranking: **848**

24 submissions

SUBMISSIONS

Score	Submitted by	Timestamp
0.6657	manjavacas	2021-01-28 08:57:12 UTC





3.3. Clasificación: Reglas

CLUSTERING

- El uso de *clusters* como predictores dio buenos resultados
 - Algoritmo de *MacQueen* mejor que k-means estándar
 - Mejora de *InfoGain* del 17%
 - Número de clusters = $\text{sqrt}(N)$ → seguramente mejorable
 - Mejores resultados → clustering `train` y `test` por separado

3.3. Clasificación: Reglas

AJUSTE DE HIPERPARÁMETROS

-F <number of folds>

Set number of folds for REP. One fold is used as pruning set.

-N <min. weights>

Set the minimal weights of instances within a split.

-O <number of runs>

Set the number of runs of optimizations.

- **5-fcv** utilizando `caret`
- Sin mejoras notables

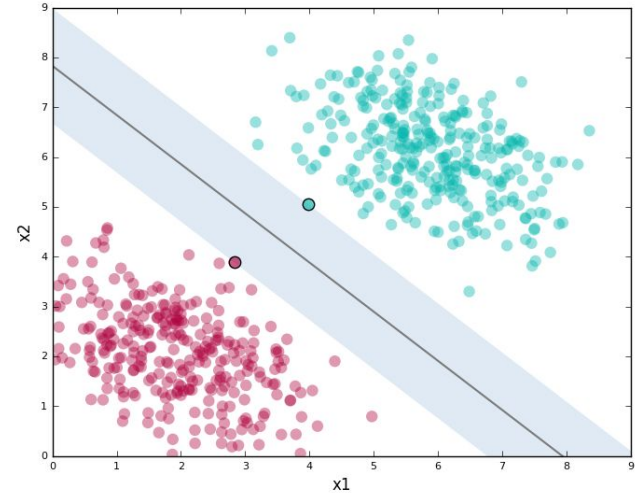
3.3. Clasificación: Reglas

CONCLUSIONES

- La `clase 2` presentó más dificultades ($F1 \approx 0.6$ vs 0.9 para `clases 1 y 3`)
- La división en 2 modelos supuso mejoras considerables
- El uso de datos sintéticos (`SMOTE`) hizo que el modelo tendiese a sobreajustarse
- Limitaciones técnicas: número de instancias permitidas por `RWeka::JRip`
- Poca influencia de los hiperparámetros
- La ganancia de información es una buena guía para elegir predictores
- El *overfitting* se debió más al balanceo de clases que a las variables elegidas

3.4. Clasificación: SVM

- **SVM** (Support Vector Machine).
- Algoritmo de clasificación basado en vectores de soporte.
- Propuesto por Vladimir Vapnik en 1992.
- Utilizaremos la implementación disponible en la librería *LiquidSVM*: `svm`.
 - Menor tiempo de ejecución que *e1071*.



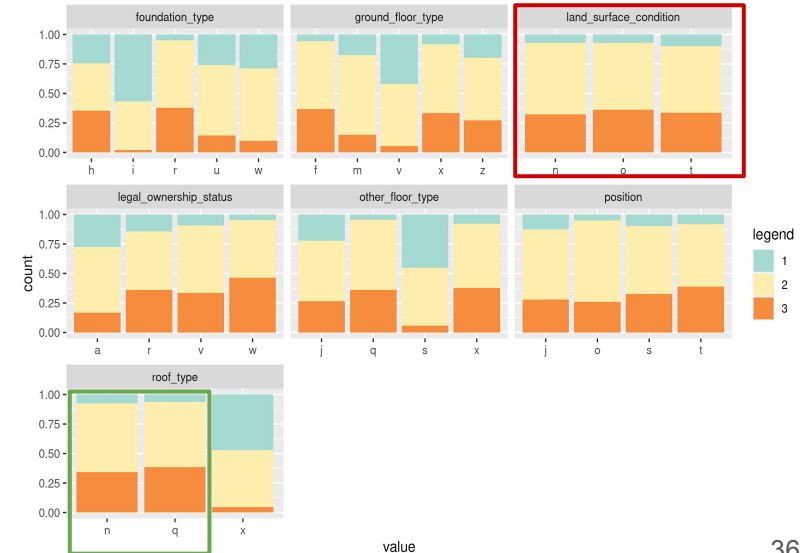
Preprocesamiento

- Variables:

- **Eliminación de categorías con patrones iguales en grado de daño**
- **Agrupación de categorías con patrones iguales en grado de daño**
- Importancia (InfoGain): Mejores 8,6, y 4 variables
- One-hot encoding
- Estandarización de variables numéricas

- Instancias:

- Random Undersampling
- Todas (Imbalance ratio 1:6:3 aproximadamente)



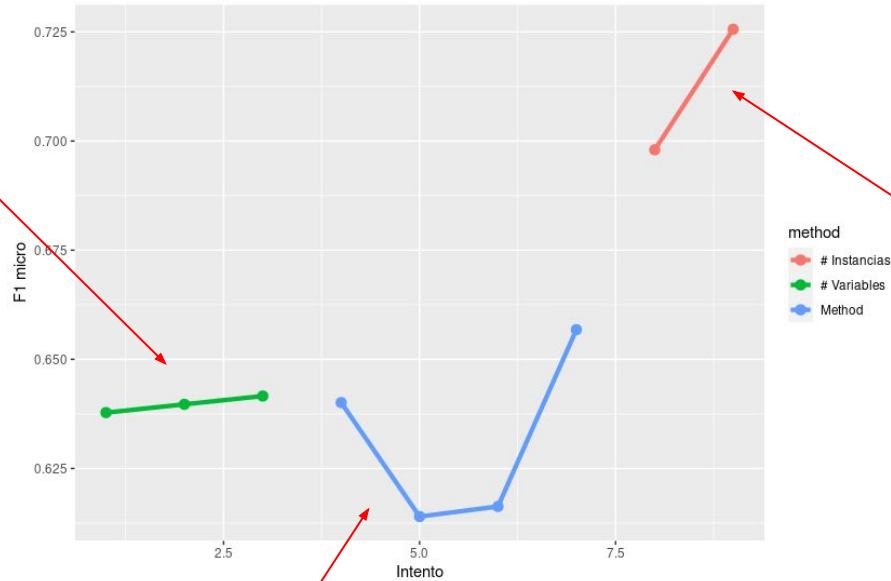
Configuraciones empleadas (más importantes)

Disminución de variables:

1. 8 variables
2. 6 variables
3. 4 variables

Manteniendo fijo:

- 75000 Instancias
- Ova (least squares)



Aumento del número de instancias:

1. 180000 instancias
2. Todas

Diferentes métodos:

1. Ova (hinge),
2. 2vsA+1vs3 (4 variables),
3. 2vsA+1vs3 (4 y 8 variables respectivamente),
4. Ova ordinal

Manteniendo fijo:

- 4 variables
- 75000 Instancias

Manteniendo fijo:

- 4 variables
- Ova ordinal

Estrategia con el mejor ranking

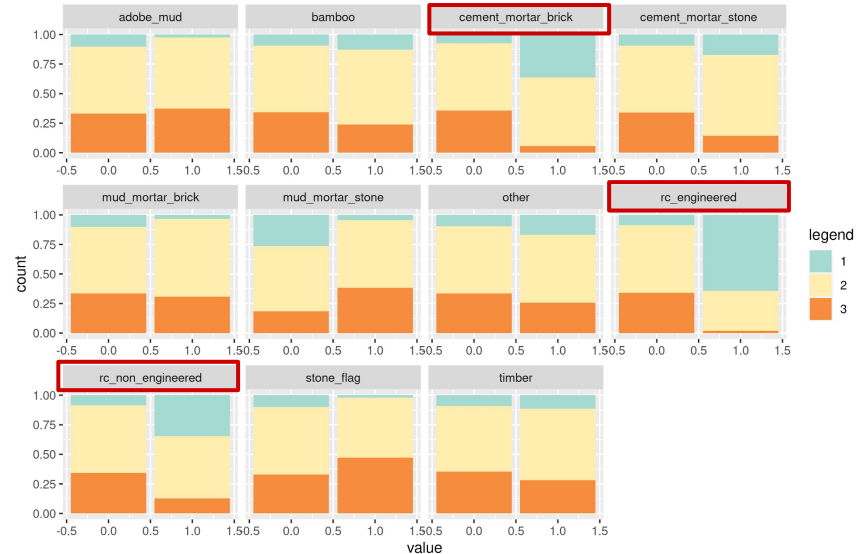
Mejor modelo:

- F1 micro: **0.7256**
- Ranking: **529**

BEST	CURRENT RANK	# COMPETITORS	SUBS. MADE
0.7256	529	3748	3 of 3

Características:

- Todas las instancias
- 4 variables más relevantes:
 - Geo levels 1,2,3
 - “Robust” superstructure



Estrategia con el mejor ranking

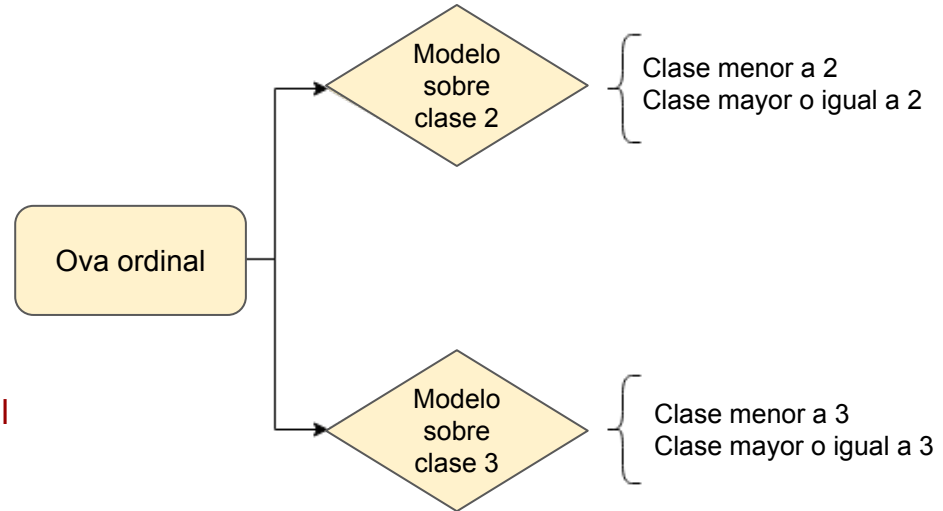
Mejor modelo:

- F1 micro: **0.7256**
- Ranking: **529**

BEST	CURRENT RANK	# COMPETITORS	SUBS. MADE
0.7256	529	3748	3 of 3

Características:

- Todas las instancias
- 4 variables más relevantes:
 - Geo levels 1,2,3
 - “Robust” superstructure
- **PROBLEMA ORDINAL** → Ova ordinal



Conclusiones

- Las variables más relevantes en mi modelo son:
 - La **localización** del edificio.
 - La **robustez** del material empleado en la construcción.
- Aumentar el **número de variables** empeora las predicciones del modelo.
- Es muy importante tener en cuenta que es un problema **ordinal**.
- Es más importante entrenar con el **mayor número de instancias** ya que
- Considerar **clases desbalanceadas** no parece empeorar demasiado la predicción.

Evolución completa

0.5591	PaulaVillaMartin
0.6329	PaulaVillaMartin
0.6378	PaulaVillaMartin
0.6397	PaulaVillaMartin
0.6416	PaulaVillaMartin
0.6401	PaulaVillaMartin
0.6396	PaulaVillaMartin
0.6155	PaulaVillaMartin
0.6140	PaulaVillaMartin
0.6568	PaulaVillaMartin
0.6544	PaulaVillaMartin
0.6980	PaulaVillaMartin
0.7256	PaulaVillaMartin
0.6163	PaulaVillaMartin

BEST

0.7256

CURRENT RANK

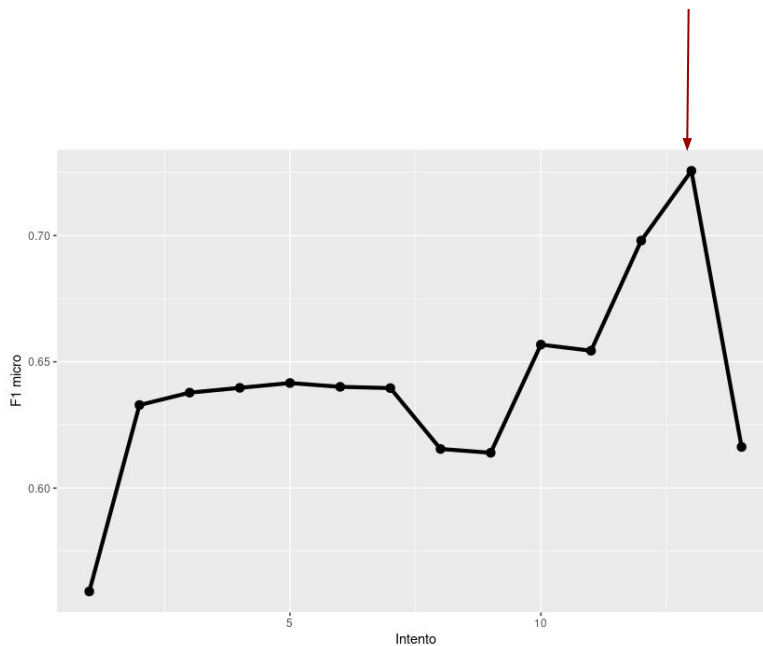
529

COMPETITORS

3748

SUBS. MADE

3 of 3



14 Intentos

4. Conclusiones

Modelo	Random undersampling	SMOTE	Eliminación de ruido	Clustering	One-hot encoding (categóricas)	Centrado y escalado (numéricas)	Selección de variables	Max. score
Regresión logística	✗	✗	✗	✗	✓	😐	✓	0.6453
Árboles	✗	✗	✗	✗	✗	😐	✓	0.7239
Reglas	✓	✗	✗	✓	✗	😐	✓	0.6657
SVM	✗	✗	✗	✗	✓	✓	✓	0.7256