

Inteligencia de Negocio (2019-2020)
GRADO EN INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE GRANADA

Memoria Práctica 3

Competición en DrivenData

Daniel Terol Guerrero
DNI: 09076204J
Correo: danielterol@correo.ugr.es

1 de enero de 2020

1. POSICIÓN FINAL EN LA COMPETICIÓN Y *Submissions*

1.1. DATOS DE MI CUENTA PERSONAL *Daniel_Terol_UGR*

BEST	CURRENT RANK	# COMPETITORS	SUBS. MADE
0.7486	46	1582	0 of 3

Figura 1.1: Mejor puntuación alcanzada en la competición.

Score	Submitted by	Timestamp ⓘ
0.6776	Daniel_Terol_UGR	2019-12-03 09:58:38 UTC
0.6883	Daniel_Terol_UGR	2019-12-03 10:07:07 UTC
0.6038	Daniel_Terol_UGR	2019-12-06 11:15:34 UTC
0.5763	Daniel_Terol_UGR	2019-12-06 12:13:21 UTC
0.6951	Daniel_Terol_UGR	2019-12-06 12:29:57 UTC
0.6998	Daniel_Terol_UGR	2019-12-07 14:19:42 UTC
0.4791	Daniel_Terol_UGR	2019-12-07 21:37:44 UTC
0.6810	Daniel_Terol_UGR	2019-12-07 22:37:04 UTC
0.6660	Daniel_Terol_UGR	2019-12-08 09:38:28 UTC
0.6868	Daniel_Terol_UGR	2019-12-08 10:38:10 UTC
0.6868	Daniel_Terol_UGR	2019-12-08 11:02:59 UTC
0.5625	Daniel_Terol_UGR	2019-12-10 09:57:16 UTC
0.4210	Daniel_Terol_UGR	2019-12-10 10:11:18 UTC
0.6868	Daniel_Terol_UGR	2019-12-10 11:03:03 UTC

Figura 1.2: Lista de *submissions* (1).











0.6660	Daniel_TeroLUGR 	2019-12-12 22:23:18 UTC
0.6951	Daniel_TeroLUGR 	2019-12-12 22:38:53 UTC
0.7001	Daniel_TeroLUGR 	2019-12-13 06:38:55 UTC
0.6843	Daniel_TeroLUGR 	2019-12-17 17:31:02 UTC
0.6843	Daniel_TeroLUGR 	2019-12-17 17:55:21 UTC
0.7219	Daniel_TeroLUGR 	2019-12-17 18:53:17 UTC
0.7440	Daniel_TeroLUGR 	2019-12-18 11:03:43 UTC
0.7451	Daniel_TeroLUGR 	2019-12-18 18:17:39 UTC
0.7456	Daniel_TeroLUGR 	2019-12-18 19:00:47 UTC
0.7462	Daniel_TeroLUGR 	2019-12-19 14:17:47 UTC
0.7461	Daniel_TeroLUGR 	2019-12-19 17:57:15 UTC
0.7450	Daniel_TeroLUGR 	2019-12-19 20:21:38 UTC
0.7458	Daniel_TeroLUGR 	2019-12-20 08:20:11 UTC
0.7457	Daniel_TeroLUGR 	2019-12-20 09:12:51 UTC
0.7462	Daniel_TeroLUGR 	2019-12-20 20:33:52 UTC

Figura 1.3: Lista de *submissions* (2).

0.7442	Daniel_TeroLUGR 	2019-12-21 14:37:09 UTC
0.7472	Daniel_TeroLUGR 	2019-12-21 21:38:42 UTC
0.7481	Daniel_TeroLUGR 	2019-12-21 23:25:30 UTC
0.7476	Daniel_TeroLUGR 	2019-12-22 00:08:32 UTC
0.7485	Daniel_TeroLUGR 	2019-12-22 00:39:30 UTC
0.7477	Daniel_TeroLUGR 	2019-12-22 01:45:29 UTC
0.7479	Daniel_TeroLUGR 	2019-12-23 08:51:15 UTC
0.7473	Daniel_TeroLUGR 	2019-12-23 10:05:19 UTC
0.7478	Daniel_TeroLUGR 	2019-12-23 19:00:30 UTC
0.7470	Daniel_TeroLUGR 	2019-12-24 13:02:13 UTC
0.7486	Daniel_TeroLUGR 	2019-12-24 13:26:21 UTC

Figura 1.4: Lista de *submissions* (3).

1.2. DATOS DE MI CUENTA SECUNDARIA *Ihore*

BEST	CURRENT RANK	# COMPETITORS	SUBS. MADE
0.7493	42	1659	0 of 3

Figura 1.5: Mejor puntuación alcanzada en la competición con la cuenta *Ihore*.

Score	Submitted by	Timestamp
0.7181	Ihore	2019-12-13 09:36:47 UTC
0.7219	Ihore	2019-12-17 21:31:04 UTC
0.7219	Ihore	2019-12-17 21:53:59 UTC
0.7226	Ihore	2019-12-17 23:02:20 UTC
0.7440	Ihore	2019-12-18 10:44:16 UTC
0.7456	Ihore	2019-12-18 12:51:17 UTC
0.7445	Ihore	2019-12-18 14:51:11 UTC
0.7457	Ihore	2019-12-19 10:00:52 UTC
0.7461	Ihore	2019-12-19 12:34:31 UTC
0.7461	Ihore	2019-12-19 13:33:27 UTC

Figura 1.6: Lista de *submissions* (1) con la cuenta *Ihore*.











0.7462	Ihore 	2019-12-20 19:06:57 UTC
0.7462	Ihore 	2019-12-21 17:29:14 UTC
0.7474	Ihore 	2019-12-21 18:34:23 UTC
0.7471	Ihore 	2019-12-21 20:14:50 UTC
0.7486	Ihore 	2019-12-23 11:14:44 UTC
0.7477	Ihore 	2019-12-23 12:16:26 UTC
0.7487	Ihore 	2019-12-28 23:35:08 UTC
0.7490	Ihore 	2019-12-29 10:00:38 UTC
0.7491	Ihore 	2019-12-29 11:43:05 UTC
0.7485	Ihore 	2019-12-29 13:47:22 UTC
0.7489	Ihore 	2019-12-30 10:56:40 UTC

Figura 1.7: Lista de *submissions* (2) con la cuenta *Ihore*.

0.7493	Ihore 	2019-12-30 12:07:46 UTC
0.7493	Ihore 	2019-12-30 14:44:36 UTC
0.7483	Ihore 	2019-12-31 12:33:36 UTC
0.7467	Ihore 	2019-12-31 14:53:14 UTC

Figura 1.8: Lista de *submissions* (3) con la cuenta *Ihore*.

ÍNDICE

1. Posición final en la competición y <i>Submissions</i>	
1.1. Datos de mi cuenta personal <i>Daniel_Terol_UGR</i>	
1.2. Datos de mi cuenta secundaria <i>Ihore</i>	
2. Introducción	1
3. Visualización de los datos	2
3.1. Balanceo de clases	2
3.2. Correlación	2
4. Registro de subidas	4
5. El camino hasta mis mejores puntuaciones	6
5.1. Selección de características inicial	7
5.2. Toma de contacto con diferentes modelos	7
5.2.1. Random Forest	7
5.2.2. KNN	7
5.2.3. Gradient Boosted	8
5.2.4. Árboles de decisión	8
5.2.5. LightGBM	8
5.3. Intentando paliar el desbalanceo de clases	8
5.3.1. Aplicación incorrecta de SMOTE	9
5.3.2. Ejecuciones con SMOTE	9
5.4. Cambio en el tratamiento de las variables categóricas	9
5.4.1. Explorando la mejor configuración con LightGBM	11
5.4.2. Obteniendo mi mejor puntuación y entrando en el TOP50	13
5.4.3. Últimos días de competición	13
5.4.4. Suspensión de mi cuenta principal	14
6. Conclusión	14
7. Bibliografía	15

2. INTRODUCCIÓN

En esta tercera práctica se ha abordado una competición real alojada en [DrivenData](#). El objetivo de la competición es predecir la gravedad del daño a los edificios causado por el terremoto Gorkha de 2015 en Nepal.

El *dataset* de entrenamiento consta de 260.601 instancias y 39 atributos. La variable a predecir es una variable ordinal, *damage_grade*, que representa un nivel de daño al edificio que fue golpeado por el terremoto. Hay 3 grados de daño.

- 1 representa un daño bajo.
- 2 representa un daño medio.
- 3 representa la destrucción casi completa.

La medida de evaluación para nuestros algoritmos, se utiliza **F1-score** que equilibra la precisión y el *recall* de un clasificador.

Como aspecto a destacar es que la búsqueda de parámetros óptimos con *GridSearch* no me funcionaba del todo bien puesto que me devolvía puntuaciones peores que con otras configuraciones. Por tanto, la mayoría de subidas son explorando diferentes configuraciones de los modelos.

3. VISUALIZACIÓN DE LOS DATOS

3.1. BALANCEO DE CLASES

Lo primero que hay que analizar es el balanceo de las diferentes clases del problema:

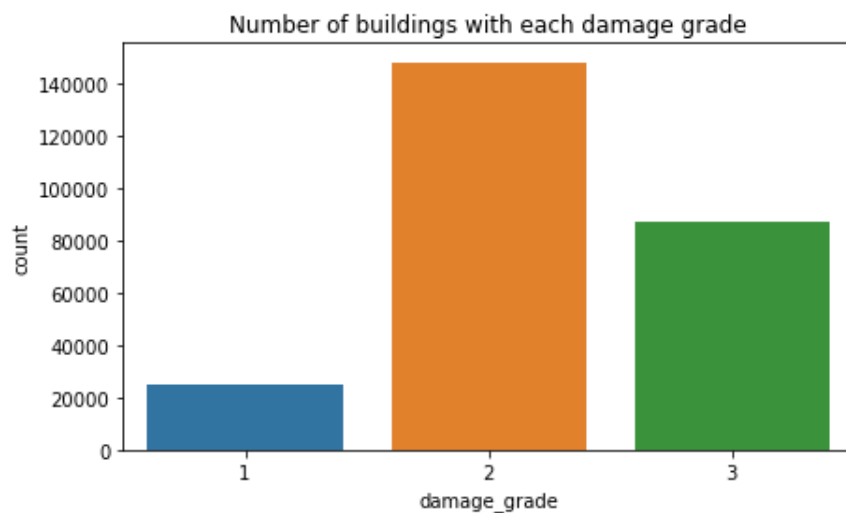


Figura 3.1: Cardinalidad de las diferentes clases del problema.

Como se puede ver, la clase 2 es la que mayor cardinalidad tiene. Mientras que la clase 3 y 1 están por detrás, siendo la clase 1 la clase minoritaria.

3.2. CORRELACIÓN

En un punto de la competición, decidí estudiar la correlación de todas las variables respecto a la variable a predecir, *damage_grade*:

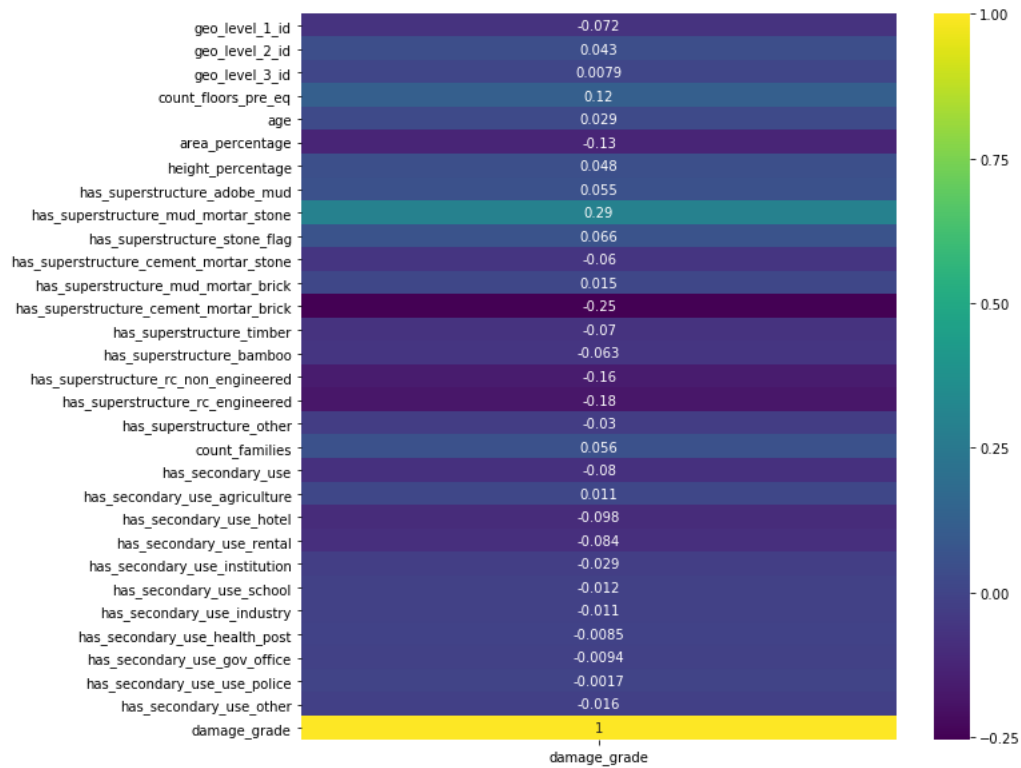


Figura 3.2: Correlación de todas las variables respecto a *damage_grade*

Observando la [figura 3.2](#), se puede ver que las características que más correlación tienen con la clase son:

- *has_superstructure_mud_mortar_stone* refleja si el edificio está hecho de barro.
- Mientras que la *count_floors_pre_eq* refleja el número de pisos que tenía el edificio antes del terremoto.
- Ninguna variable más refleja una correlación a destacar.

4. REGISTRO DE SUBIDAS

La mayoría de entradas de la tabla, excepto la última, son de mi cuenta personal. No he podido realizar un seguimiento de versiones con las subidas de la cuenta secundaria pues no sé las configuraciones establecidas, excepto la última puesto que es la mejor puntuación conseguida y la última configuración, prácticamente, utilizada.

Versión	Fecha y hora	Posición	Score Training	Score DrivenData	Preprocesado	Descripción algoritmos	Configuración	Comentarios
1	03/12/2019 10:58	305	0'6886	0'6776	Archivo de prueba de Jorge Casillas	XGBOOST	Archivo de prueba de Jorge Casillas	
2	03/12/2019 11:07	301	0'7264	0'6883	Archivo de prueba de Jorge Casillas	LightGBM	Archivo de prueba de Jorge Casillas	
3	06/12/2019 12:15	306	0'6192	0'6038	Eliminación de las características 'geo_level_x_id'	LightGBM	Configuración básica (original de Jorge Casillas)	En la columna 'Preprocesado' se va a dar por hecho la parte de pasar variables categóricas a numéricas realizado por Jorge Casillas.
4	06/12/2019 13:13	306	0.8249	0'5763	Eliminación de las características 'geo_level_x_id'	RandomForest	n_estimators = 100 max_depth=None	
5	06/12/2019 13:29	294	0.9843	0'6951	Ninguno	RandomForest	n_estimators = 100 max_depth=None validación cruzada	
6	07/12/2019 15:19	287	0.9764	0'6998	Ninguno	RandomForest	n_estimators = 500 max_depth=None validación cruzada	
7	07/12/2019 22:37	288	0.7855	0'4791	Ninguno	KNN	n_neighbors = 5 validación cruzada	
8	07/12/2019 23:37	288	0.6944	0'6810	Ninguno	Gradient Boosted	n_estimators = 200 validación cruzada	
9	08/12/2019 10:39	290	0.7369	0'6660	Ninguno	Decision Tree	min_samples_split = 100 min_samples_leaf = 50 criterion = Entropy	

Figura 4.1: Seguimiento de las diferentes versiones (1).

10	08/12/2019 11:38	290	0.7738	0'6868	Ninguno	LightGBM	n_estimators = 1000 validación cruzada	
11	08/12/2019 11:02	290	0.7738	0'6868	Balanceo de clases (SMOTE)	LightGBM	n_estimators = 1000 validación cruzada	
12	10/12/2019 10:57	292	0.7718	0'5625	Balanceo de clases (SMOTE)	Decision Tree	min_samples_split = 100 min_samples_leaf = 50 criterion = Entropy	
13	10/12/2019 11:11	292	0.8279	0'4210	Balanceo de clases (SMOTE)	LightGBM	n_estimators = 1500 validación cruzada	Sobreaprendizaje
14	10/12/2019 11:03	292	0.7738	0'6868	Balanceo de clases (SMOTE)	LightGBM	n_estimators = 1000 validación cruzada	Configuración similar a la anterior pero aplicando de forma correcta SMOTE.
15	12/12/2019 23:23	292	0.7369	0'6660	Balanceo de clases (SMOTE)	Decision Tree	min_samples_split = 100 min_samples_leaf = 50 criterion = Entropy	Se aplica de forma correcta SMOTE.
16	12/12/2019 23:38	294	0.9843	0'6951	Balanceo de clases (SMOTE)	RandomForest	n_estimators = 100 max_depth=None validación cruzada	
17	13/12/2019 07:40	295	0.9843	0'7001	Balanceo de clases (SMOTE)	RandomForest	n_estimators = 1000 max_depth=None validación cruzada	
18	17/12/2019 18:31	310	0'8019	0'6843	Balanceo de clases (SMOTE)	LightGBM	n_estimators = 2000 validación cruzada	
19	17/12/2019 18:56	310	0'8019	0'6843	Ninguno	LightGBM	n_estimators = 2000 validación cruzada	

Figura 4.2: Seguimiento de las diferentes versiones (2).

20	17/12/2019 19:53	221	0'9868	0'7219	Forma diferente de pasar las variables categóricas a numéricas.	RandomForest	n_estimators = 500 max_depth=None	La forma diferente para pasar las variables categóricas a variables numéricas es usando métodos propios de pandas (se transforma la columna a "category" y luego se codifica)
21	18/12/2019 11:03	99	0'8532	0'7440	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 2000 validación cruzada	Pese haber probado anteriormente con RF porque era el modelo que mejor me funcionaba anteriormente, Jorge dio a entender que LightGBM debería dar mejores resultados que RF, por eso lo he usado aquí.
22	18/12/2019 19:18	87	0'8593	0'7451	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 5500 validación cruzada	
23	18/12/2019 20:00	80	0'8758	0'7456	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 5000 validación cruzada	
24	19/12/2019 15:17	74	0'8313	0'7462	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 3500 validación cruzada	Estoy reduciendo el número de modelos de LightGBM porque, a vista de los resultados, estaba sobreaprendiendo.
25	19/12/2019 18:57	74	0'8221	0'7461	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 3000 validación cruzada	
26	19/12/2019 21:21	74	0'8983	0'7450	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 3500 validación cruzada max_bin=750 num_leaves=75	Sobreaprendizaje

Figura 4.3: Seguimiento de las diferentes versiones (3).

27	20/12/2019 09:20	74	0'8495	0'7458	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 3500 validación cruzada max_bin=500 num_leaves=40
28	20/12/2019 10:12	74	0'8492	0'7457	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 3500 validación cruzada max_bin=350 num_leaves=40
29	20/12/2019 21:33	78	0'8313	0'7462	Forma diferente de pasar las variables categóricas a numéricas. SMOTE con polynom_fit (smote_variants).	LightGBM	n_estimators = 3500 validación cruzada
30	21/12/2019 15:37	81	0'8272	0'7442	Forma diferente de pasar las variables categóricas a numéricas. SMOTE con polynom_fit (smote_variants). Selección de características.	LightGBM	n_estimators = 3500 validación cruzada
31	21/12/2019 22:38	68	0'8509	0'7472	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 3500 max_bin = 1000 num_leaves = 40 validación cruzada
32	22/12/2019 00:25	52	0'8323	0'7481	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 3500 max_bin = 950 validación cruzada

Figura 4.4: Seguimiento de las diferentes versiones (4).

33	22/12/2019 01:08	52	0'8325	0'7476	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 3500 max_bin = 940 validación cruzada
34	22/12/2019 01:39	44	0'8322	0'7485	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 3500 max_bin = 960 validación cruzada
35	22/12/2019 02:45	44	0'8324	0'7477	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 3500 max_bin = 965 validación cruzada
36	23/12/2019 09:51	46	0'8326	0'7479	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 3500 max_bin = 955 validación cruzada
37	23/12/2019 11:05	46	0'8339	0'7473	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 3500 max_bin = 960 num_leaves = 32 validación cruzada
38	23/12/2019 20:00	47	0'8255	0'7478	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 3500 max_bin = 960 num_leaves = 28 validación cruzada
39	24/12/2019 14:02	47	0'8296	0'7470	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 3500 max_bin = 960 num_leaves = 30

Figura 4.5: Seguimiento de las diferentes versiones (5).

40	24/12/2019 14:26	46	0'8276	0'7486	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 3500 max_bin = 960 num_leaves = 29 validación cruzada
41	30/12/2019 13:07	36	0'8369	0'7493	Forma diferente de pasar las variables categóricas a numéricas.	LightGBM	n_estimators = 8250 max_bin = 960 num_leaves = 29 learning_rate = 0.05 validación cruzada

Figura 4.6: Seguimiento de las diferentes versiones (6).

5. EL CAMINO HASTA MIS MEJORES PUNTUACIONES

La primera toma de contacto con *DrivenData* fueron dos subidas, [figura 4.1](#), haciendo uso de los modelos proporcionados por Jorge Casillas para poder saber cuál era mi punto de partida en la competición, donde se puede ver que era la posición 301 y una puntuación de 0'6883.

5.1. SELECCIÓN DE CARACTERÍSTICAS INICIAL

Analizando un poco el *dataset* vi tres variables que eran identificadores y, además, con una gran cardinalidad cada uno de ellos:

- `geo_level_1_id`
- `geo_level_2_id`
- `geo_level_3_id`

Me imaginé que eliminando estas características incrementaría, aunque fuera un poco, la puntuación conseguida. Para mi sorpresa, tanto con *LightGBM* como con *Random Forest*, empeoró drásticamente.

5.2. TOMA DE CONTACTO CON DIFERENTES MODELOS

Después de este fracaso, me propuse probar diferentes modelos¹ para saber cuáles podrían ofrecer mejor rendimiento en el problema, mientras se me ocurría algún preprocesado o limpieza de ruido.

5.2.1. RANDOM FOREST

Probé con *Random Forest*² con $n_estimators = 100$ y vi que me producía una puntuación de 0'6951, que ya era una pequeña mejora frente al 0'6883 inicial de *LightGBM*.

Puesto que vi una mejora con $n_estimators = 100$, probé con $n_estimators = 500$. El resultado fue una mejora considerable, 0'6998. Ya me encontraba bastante cerca de superar la barrera de 0'70.

5.2.2. KNN

Pese a haber obtenido buenos resultados con *Random Forest* y pudiendo incrementar el número de modelos, decidí seguir probando con diferentes modelos, en este caso *KNN*. El resultado no fue muy esperanzador, quizá por elegir solo los cinco mejores vecinos.

¹Quise probar con SVM pero la ejecución se alargaba a más de 12 horas, cosa que lo hacía inviable.

²Como curiosidad, y sin saber el motivo, *Random Forest* sobreaprendía muchísimo siendo el *score* en el training de 0'9843.

5.2.3. GRADIENT BOOSTED

Los resultados fueron buenos pero no lo suficientemente para poder explorar diferentes configuraciones de este algoritmo. Algo de destacar de *Gradient Boosted* es que la puntuación obtenida en el *training* y en *DrivenData* es la que menos se diferencia entre ellas.

5.2.4. ÁRBOLES DE DECISIÓN

Estableciendo los parámetros $min_samples_split = 100$ y $min_samples_leaf = 50$, los resultados son decentes pero no suficientes para esta competición. Cabe destacar que la ejecución fue tremendamente rápida.

5.2.5. LIGHTGBM

LightGBM ya lo ejecuté, ya que era uno de los algoritmos disponibles en el archivo original de Jorge Casillas. Puesto que el resultado obtenido con $n_estimators = 200$ fue bastante bien, pensé que podría conseguir una puntuación mejor con $n_estimators = 1000$ que la obtenida por *Random Forest* pero el resultado no fue el esperado. Conseguí una puntuación de 0'6868.

5.3. INTENTANDO PALIAR EL DESBALANCEO DE CLASES

Con un intento de paliar el desbalanceo de clases, se hizo uso de la técnica *SMOTE*.

```
def do_smote (X, y):
    #SMOTE
    #print('Resampled dataset shape %s' % Counter(y))
    sm = SMOTE(random_state=123456)
    X_res, y_res = sm.fit_resample(X, y)
    #print('Resampled dataset shape %s' % Counter(y_res))

    return (X_res, y_res)

def validacion_cruzada(modelo, X, y, cv):
    y_test_all = []

    for train, test in cv.split(X, y):
        t = time.time()
        x_train, y_train = do_smote(X[train], y[train])
        modelo = modelo.fit(x_train, y_train)
        tiempo = time.time() - t
        y_pred = modelo.predict(X[test])
```

Figura 5.1: Aplicación de la técnica *SMOTE*.

5.3.1. APLICACIÓN INCORRECTA DE SMOTE

Después de probar varios algoritmos, me di cuenta que había problemas de sobreaprendizaje ³ puesto que obtuve un *score* en el *training* de 0'8279 y en *DrivenData* una puntuación de 0'4210.

5.3.2. EJECUCIONES CON SMOTE

Una vez arreglado el problema de sobreaprendizaje, volví a probar con diferentes algoritmos para ver si el balanceo de clases producía algún impacto en las puntuaciones obtenidas pero no era así. Las ideas empezaban a escasear puesto que había probado selección de características, aunque no lo subiera a *DrivenData* ya que los resultados eran nefastos.

Entonces, ejecuté el problema con *Random Forest* con el número de modelos a 1000. La diferencia en la puntuación obtenida entre el número de modelos igual a 500 y a 1000 es 0'0003, diferencia suficiente para poder sobrepasar la línea de los 0'70, posicionándome el 295 con una puntuación de 0'7001.

5.4. CAMBIO EN EL TRATAMIENTO DE LAS VARIABLES CATEGÓRICAS

Si nos fijamos en la [figura 4.2](#), se puede ver como después de obtener 0'7001 hubo varios días donde no subí nada a *DrivenData*. En esos días empecé a usar métodos de selección de características y limpieza de ruido puesto que la ejecución de *Random Forest* con el número de modelos a 1000 duró varias horas, lo que empezaba a ser un inconveniente para poder seguir ajustando el número de modelos.

En ejecuciones anteriores, *LightGBM* no me había dado resultados excepcionales pero el día 17 de Diciembre, Jorge Casillas en clase de prácticas le pareció extraño que *Random Forest* proporcionase mejores resultados que *LightGBM*. Como consecuencia de esto, el día 17 de Diciembre hay reflejadas dos ejecuciones con *LightGBM* con el número de modelos a 2000, puesto que *LightGBM* es muy

³Pese a no tener captura de la mala aplicación de SMOTE, aplicaba el balanceo de clases tanto en el *training* como en el *test*.

ligero y rápido. Los resultados eran decepcionantes.

Las ideas se me agotaban y los resultados no eran muy buenos, entonces decidí modificar la forma en la que se codificaban las variables categóricas puesto que, desde el principio de la práctica, me parecía confuso y para nada intuitivo la manera en la que se hacía.

```
columnas_categoricas=['land_surface_condition', 'foundation_type', 'roof_type',
                      'ground_floor_type', 'other_floor_type', 'position',
                      'plan_configuration', 'legal_ownership_status']

# Por cada variable categóricas, se codifica haciendo uso de '.cat.codes'.
for i in columnas_categoricas:
    data_x[i]=data_x[i].astype("category")
    data_x[i]=data_x[i].cat.codes

    data_x_tst[i]=data_x_tst[i].astype("category")
    data_x_tst[i]=data_x_tst[i].cat.codes

X = data_x.values
X_tst = data_x_tst.values
y = np.ravel(data_y.values)
```

Figura 5.2: Modificación a la conversión de categórica a numérica.

La nueva forma de pasar las variables categóricas a numéricas consiste en los siguientes pasos:

1. Se guardan todas las variables categóricas en un vector.
2. Tanto en el *training* como en el *test* se recorren las columnas correspondientes:
 - a) Se hace uso del método *astype* que se encarga de convertir un vector al tipo que se le especifique, en este caso *category*.
 - b) Una vez hecha la conversión, se usa *cat.codes* que realiza la codificación correspondiente.

El método de conversión de Jorge Casillas y este método consiguen el mismo resultado. Decidí modificarlo puesto que pensé que si se hace uso de métodos propios de *pandas* para la codificación sería más eficiente y efectivo.

Debido a la limitación horaria, no pude probar ese día con el algoritmo y configuración que mejor resultado me devolvieron. Es decir, *Random Forest* con el número de modelos con 1000 pero probé con

$n_estimators = 500$ y, sorprendentemente, obtuve una puntuación en *DrivenData* de 0'7219 posicionándome en el puesto 221.

5.4.1. EXPLORANDO LA MEJOR CONFIGURACIÓN CON LIGHTGBM

Como comenté anteriormente, a Jorge le extrañó que *LightGBM* funcionase mejor que *Random Forest* por tanto probé con *LightGBM* con $n_estimators = 2000$ obteniendo una puntuación de 0'7440 posicionándome el 99.

A partir de este momento me dediqué a buscar el número más cercano al número óptimo de $n_estimators$. Después de varias configuraciones, el mejor resultado que obtuve fue con $n_estimators = 3500$ obteniendo una puntuación de 0'7462 posicionándome en el puesto 74.

Investigando por internet, y como se puede ver en [1], encontré que para obtener mejor precisión, había que aumentar el parámetro max_bin y num_leaves . El inconveniente es que si num_leaves aumentaba mucho había riesgo de que el modelo sobreprendiese. Después de varios intentos, desistí de modificar estos parámetros.

Explorando el [Github de gykovacs](#), encontré que el SMOTE que ofrecía mejor rendimiento, siendo evaluado con *F1-score*, era el *polynom fit SMOTE*:

```
import smote_variants as sv
```

Figura 5.3: Adición del módulo `smote_variants`.

```
def do_smote (X, y):
    #SMOTE
    #print('Resampled dataset shape %s' % Counter(y))
    sm = sv.polynom_fit_SMOTE()
    X_res, y_res = sm.sample(X, y)
    #print('Resampled dataset shape %s' % Counter(y_res))

    return (X_res, y_res)

def validacion_cruzada(modelo, X, y, cv):
    y_test_all = []

    for train, test in cv.split(X, y):
        t = time.time()
        x_train, y_train = do_smote(X[train], y[train])
        modelo = modelo.fit(x_train, y_train)
        tiempo = time.time() - t
        y_pred = modelo.predict(X[test])
```

Figura 5.4: Aplicación de polynom fit SMOTE.

Los resultados no mejoraron la puntuación ya obtenida. Aún así, realicé un tratamiento sobre los datos para quedarme con las mejores características y ver si con este tratamiento y el SMOTE mejoraba la puntuación obtenida.

```
#Se seleccionan las mejores características.
modelo = lgb.LGBMClassifier(objective='regression_l1', n_estimators=3500)
modelo.fit(data_x, data_y)

feat_importances = pd.Series(modelo.feature_importances_, index= data_x.columns)
feat_importances = feat_importances.nlargest(18)

import_features = list(feat_importances.index)

data_x_new = data_x[import_features]
data_x_tst_new = data_x_tst[import_features]

X = data_x_new.values
X_tst = data_x_tst_new.values
y = np.ravel(data_y.values)
```

Figura 5.5: Selección de características.

El método sigue los siguientes pasos:

1. Se entrena un modelo, en este caso *LightGBM* con la mejor configuración hasta el momento.
2. Se escogen las 18 mejores características del conjunto de datos y se crean dos nuevos conjuntos de datos con las características más importantes del *dataset*.

Los resultados no fueron los mejores.

5.4.2. OBTENIENDO MI MEJOR PUNTUACIÓN Y ENTRANDO EN EL TOP50

Al no haber ninguna mejora, retomé mi búsqueda de parámetros óptimos de *LightGBM*. Decidí aumentar mucho *max_bin*, puesto que había riesgo menor de que el modelo sobreaprendiera, y aumentar poco *num_leaves*. Para mi sorpresa, y como se puede ver en la [figura 4.4](#) (versión 31), obtuve una puntuación de 0'7472 posicionándome en el puesto 68.

Después de varios intentos, la configuración de *max_bin* = 960 es la que mejor resultado me produjo, puntuándome con 0'7485 en la posición 44, entrando así en el TOP 50.

Una vez tenía la configuración "óptima" de *n_estimators* y *max_bin* me faltaba encontrar el parámetro óptimo de *num_leaves*. Después de varias pruebas, [figura 4.5](#), mi mejor resultado lo logré, [figura 4.6](#), con *num_leaves* = 29. Dicha puntuación fue 0'7486 posicionándome en el puesto 46.

5.4.3. ÚLTIMOS DÍAS DE COMPETICIÓN

Pese a haber estado en el TOP50 siendo la única cuenta *UGR* varios días, el día 28-29 de Diciembre compañeros de la asignatura empezaron a entrar en el TOP50 y empezaron a adelantarme. Antes de ese momento llevaba varios días sin tocar la práctica, entrando simplemente a consultar la posición en *DrivenData* pero cuando me adelantaron, tuve que empezar a buscar alternativa para rascar centésimas en la puntuación.

De los diferentes consejos para mejorar la precisión en [1] había uno que había pasado por alto:

- Disminuir el *learning_rate* y aumentar el número de modelos, *n_estimators*.

Puesto que había que rascar lo que se pudiera, decidí disminuir el *learning_rate* hasta 0'05 e ir aumentando progresivamente el núme-

ro de modelos para ver con cuál me devolvía mejor valor. Después de varias pruebas, como se puede ver en la [figura 4.6](#), la mejor configuración que encontré fue establecer el número de modelos a 8250 obteniendo una puntuación de 0'7493 posicionándome en el puesto 36 en la cuenta secundaria.

5.4.4. SUSPENSIÓN DE MI CUENTA PRINCIPAL

El día 31 por la mañana me encontré que me habían cerrado la cuenta por múltiples cuentas y no pude realizar la mejor subida sobre mi cuenta personal. Sabía el riesgo de baneo, ya que las reglas de *DrivenData* están bastante claras y creí tomar las medidas convenientes al respecto, es decir:

- Un compañero mío se creó una cuenta, en una IP diferente a la mía (él en su casa), y yo nunca entré en esa cuenta desde mi ordenador/móvil, simplemente le pasaba por *Whatsapp* los submissions para poder ver el resultado que devolvía dicha configuración. Posteriormente, subía en mi cuenta la mejor configuración que obtenía, imagino que fue en esos momentos cuando subía el mejor submission ya que obtenía la misma puntuación que otra cuenta y era sospechoso y me tumbaron la cuenta.
- Con estas medidas y Jorge Casillas permitiéndonos cuenta secundaria creía que no pasaría nada pero, para mi sorpresa, sí pasó.

6. CONCLUSIÓN

Habiendo finalizado la competición, mi mejor posición es el puesto 42 con 0'7493, pese a ser con la cuenta secundaria.

Como conclusión de la competición, voy a comentar mis sensaciones respecto a la competición:

- Empecé muy temprano a probar diferentes modelos, técnicas, visualizando datos. En definitiva, con ganas de aprender diferentes técnicas.

- Logré colarme en el TOP50, después de varios intentos y configuraciones, modificando la codificación de las variables categóricas.
- Los últimos días de competición fueron agobiantes puesto que tu nota depende de la posición obtenida por ti respecto a tus compañeros. Por tanto, cuando tus compañeros te alcanzaban, era agobiante y te hacía estar pendiente de la clasificación todo el día.
- Pese al agobio, la experiencia de participar en una competición real ha sido interesante aunque quizá para un ambiente académico propondría un modelo de evaluación diferente en el que tu nota fuese aumentando según diferentes umbrales de puntuación que consigas, y no tanto por la posición que tengas.

7. BIBLIOGRAFÍA

REFERENCIAS

- [1] <https://lightgbm.readthedocs.io/en/latest/Parameters-Tuning.html>, Consultado el 19 de Diciembre.
- [2] <https://effectiveml.com/using-grid-search-to-optimise-catboost-parameters.html>, Consultado el 17 de Diciembre.
- [3] <https://towardsdatascience.com/a-feature-selection-tool-for-machine-learning-in-python-b64dd23710>, Consultado el 16 de Diciembre.