Inteligencia de Negocio: Práctica 3

Aplicación Real: Competición en DrivenData

José Antonio Álvarez Ocete - 77553417Q
joseantonioao@correo.ugr.es

7 de enero de 2020

Índice

1. Introducción

En esta última práctica participamos en una competición DrivenData - Richter's Predictor: Modeling Earthquake Damage (`https://www.drivendata.org/competitions/57/nepal-earthquake/submissions/`). Nuestro objetivo será predecir la severidad de los daños tras un terremoto en Nepal.

2. Intentos realizados

Presentamos a continuación los intentos realizados para la competición.

SUBMISSIONS

Score	Submitted by	Timestamp
0.6883	Jose_Antonio_Alvarez_UGR	2019-12-03 10:08:00 UTC
0.7181	Jose_Antonio_Alvarez_UGR	2019-12-26 17:32:35 UTC
0.7469	Jose_Antonio_Alvarez_UGR	2019-12-28 15:55:33 UTC
0.7169	Jose_Antonio_Alvarez_UGR	2019-12-28 16:32:23 UTC
0.7169	Jose_Antonio_Alvarez_UGR	2019-12-28 19:41:14 UTC
0.7440	Jose_Antonio_Alvarez_UGR	2019-12-29 01:36:41 UTC
!	Jose_Antonio_Alvarez_UGR	2019-12-29 22:25:21 UTC
0.7445	Jose_Antonio_Alvarez_UGR	2019-12-29 22:26:46 UTC
0.7476	Jose_Antonio_Alvarez_UGR	2019-12-30 18:18:29 UTC

Figura 1: Submissions - resumen de los intentos realizados

2.1. Intento 1: Ejemplo de Jorge

Primera subida realizada en clase.

- **Fecha y hora**: 2019-12-03 10:08:00 UTC
- **F1-score obtenido**: 0.6883
- **Preprocesado**: *Category to number* aplicado a las variables categóricas.
- **Algorimo empleado**: LGBM
- **Parámetros de los algoritmos**: *objective='regression_l1', n_estimators=200, n_jobs=4*

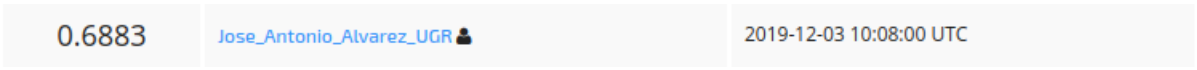

Figura 2: Intento 1

2.2. Intento 2: Get Dummies

Primer día de trabajo. Trasteando un poco el preprocesado sin tocar el algoritmo en si, cambió *Category to number* por *Get Dummies*. Resulta dar unos resultados mucho mejores. Sin embargo, el número de atributos aumenta a mas de seis decenas. Esto tendrá repercusiones posteriormente en el tiempo de ejecución de los algoritmos, lo que acaba siendo un factor decisivo en esta competición. Aún así, ni si quiera se plantéa quitarlos debido a la increible mejora que proporcionan.

- **Fecha y hora**: 2019-12-26 17:32:35 UTC
- **F1-score obtenido**: 0.7181
- **Preprocesado**: *Get Dummies*.
- **Algorimo empleado**: LGBM
- **Parámetros de los algoritmos**: *objective='regression_l1', n_estimators=200, n_jobs=4*

Figura 3: Intento 2

2.3. Intento 3: Noise reduction, feature selection y XGBoosting

Sigo trabajando el primer día. Intento reducir ruido utilizando los algoritmos vistos en clase. Tras no conseguir que funcione nada en Python intento trabajar en R. Me dispongo a usarlo pero los algoritmos me dan problemas de dependencias que no se resolver. Deshecho la idea de limpiar ruido con algoritmo conocidos con perspectiva de quizás implementar SMOTE a mano.

Comienza el segundo día. Intento aplicar selección de características. Utilizo los algoritmos de Mlxtend. Los algoritmos SFS, SBS, SFFS, SBFS tienen problemas de dependencias (les falta un import!) y no puedo ejecutarlos. Pruebo con ExhaustiveForwardSearch (EFS) pero no acaba, como es natural. Tambien pruebo a estudiar una matriz de correlaciones. Como en principio no acaba de ejecutar utilizo undersampling para ejecutarlo en un subgrupo de las instancias. Tampoco acaba.

Tercer día de trabajo. Empiezo a usar SelectKBest (un *wrapper* que selecciona los k mejores atributos). Para ello le vamos dando distintos valores a k y realizamos un estudio con cuál da mejor resultado. Puesto que toma mucho tiempo de ejecución, mientras tantoconfiguro el algoritmo XGBoosting, ya que mis ataques al preprocesado no han tenido absolutamente ningún resultado.

Subimos la ejecución de XGBoosting con asombrosamente buenos resultados. Las conclusiones del SelectKBest se estudian en el siguiente intento.

- **Fecha y hora**: 2019-12-28 15:55:33 UTC

- **F1-score obtenido**: 0.7469
- **Preprocesado**: *Get Dummies*.
- **Algorimo empleado**: XGBoosting
- **Parámetros de los algoritmos**: *(n_estimators=600 reg_alpha=0.3, seed=123456, eta=0.1, max_depth=10*

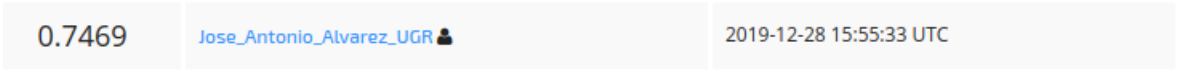

Figura 4: Intento 3

2.4. Intento 4: Estudio de SelectKBest

Tras completar el estudio de Feature Selection con SelectKBest subo el mejor valor de SelectKBest (k=33) con lgbm. Obtengo 0.7169, peor que el resultado original (T2 - 0.7181) sin preprocesado.

- **Fecha y hora**: 2019-12-28 16:32:23 UTC
- **F1-score obtenido**: 0.7169
- **Preprocesado**: *Get Dummies* y *SelectKBest* con *k=33*.
- **Algorimo empleado**: LGBM
- **Parámetros de los algoritmos**: *objective='regression_l1', n_estimators=200, n_jobs=4*

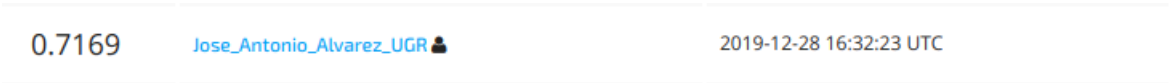

Figura 5: Intento 4

2.5. Intento 5: Subida errónea

En este intento subí de forma equivocada los resultados del intento 4.

2.6. Intento 6: SelectKBest con XGBoosting

Unimos a continuación los resultados de los dos intentos anteriores, combinando la mejor configuración obtenida de XGBoosting con el preprocesado de SelectKBest con k=33. Obtenemos un F1-score de 0.7440, ligeramente peor que el 0.7469 del intento con XGBoosting sin preprocesado pero en la mitad de tiempo.

- **Fecha y hora**: 2019-12-29 01:36:41 UTC
- **F1-score obtenido**: 0.7440
- **Preprocesado**: *Get Dummies* y *SelectKBest* con *k=33*.
- **Algorimo empleado**: XGBoosting

- **Parámetros de los algoritmos**: *(n_estimators=600 reg_alpha=0.3, seed=123456, eta=0.1, max_depth=10*

Figura 6: Intento 6

2.7. Intentos 7 y 8: Selección de instancias y salto a stacking

En esta etapa final de la competición comenté mis resultados con mis compañeros. Dado que ha ellos les estaba yendo bien con la técnica de Stacking me propuse a usarla combinándola con el preprocesado que había estado usando hasta ahora. Si bien se ha demostrado en dos ocasiones que este preprocesado reduce la calidad del predictor obtenido a posteriori, si que reduce significativamente el tiempo de ejecución. Utilizo por tanto el preprocesado buscando lanzar algoritmos mucho más pesados.

Me temo que por alguna razón que desconozco no tengo guardados los parámetros de la subida número 7, pero ambas son exactamente iguales: aplicación de stacking con preprocesado de SelectKBest, aumentando los valores de los parámetros del 7 al 8. La última subida llevo un tiempo de ejecución de entre 10 y 17 horas (lo dejé por la noche ejecutando) y dado el día que era y que la alternativa que me quedaba era ejecutar stacking con mayores valores de los parámetros (añadir otros estimadores tampoco me daba mejores resultados), di por terminada la competición. A día 7 de enero estoy en la posición 88 con un F1-score final de 0.7476.

- **Fecha y hora**: 2019-12-29 22:26:46 UTC y 2019-12-30 18:18:29 UTC.
- **F1-score obtenido**: 0.7445 y 0.7476.
- **Preprocesado**: *Get Dummies* y *SelectKBest* con *k=33*.
- **Algorimo empleado**: Stacking con estimadores RandomForest y XGBoosting.
- **Parámetros de los algoritmos**: XGBoosting: *(n_estimators=600, seed=323232, eta=0.1, max_depth=10*. RandomForest: *n_estimators=200, n_jobs=-1, max_depth=50, warm_start="True"*

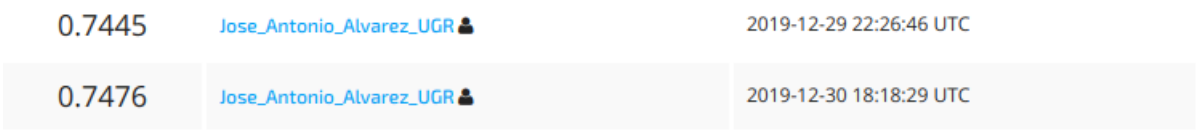

Figura 7: Intento 7 y 8