Informe Final - Grupo 32

Introducción

Para el Decision Tree realizamos varios modelos: un modelo simple con hiperparametros elegidos a mano, un modelo con cross validation con poda y un modelo también con cross validation pero sin poda. El mejor modelo lo obtuvimos a partir del cross validation sin poda con un f1-score de 0.851 en test y una puntuación de 0.843 en Kaggle.
En KNN buscamos hiperparametros a través de cross validation y con el mejor modelo obtuvimos un f1-score de 0.8399 en test y una puntuación de 0.8369 en Kaggle.
En la creación del modelo SVM al poder variar el kernel con los que se entrenaba el modelo decidimos utilizar el radial y el polinómico y variar, según el kernel utilizado, sus diferentes hiper parámetros. Con el SVM kernel radial obtuvimos un f1-score de 0.844 en test y una puntuación de 0.8466 en Kaggle, muy similar entre sí; con el SVM kernel polinómico obtuvimos un f1-score de 0.845 en test y 0.8435 en Kaggle, también muy similar. Antes de buscar los mejores hiperparametros para este modelo realizamos una pequeña prueba utilizando los parámetros por default y entrenamos un modelo con datos normalizados y otro con los datos sin normalizar y al probarlo con los datos de test se notaba una clara mejoría en las diferentes métricas al utilizar los datos normalizados.
Comparando la puntuación de f1-score en test de los modelos Random Forest, Stacking y XGBoost obtuvimos una puntuación de 0.8789, 0.8788 y 0.8778 respectivamente, las cuales son las más altas de todo el trabajo y además, a parte de tener muy poca diferencia entre sí, no tuvieron una diferencia mayor a ±0.01 respecto a la puntuación obtenida en la competencia de Kaggle (Véase en “Cuadro de Resultados”).
En Voting obtuvimos un f1-score de 0.8657 en test y una puntuación de 0.8681 en Kaggle
Para el ensamble de Voting y Stacking utilizamos los modelos de Random Forest, KNN, SVM kernel radial, SVM kernel polinómico y XGBoost. Para el metamodelo de Stacking utilizamos una regresión logística.
Para redes neuronales buscamos optimizar los hiper parámetros de la arquitectura a utilizar mediante k-fold Cross Validation, logramos obtener así un modelo con un f1-score de 0.844 en Kaggle. Una observación es que usamos conexiones densas únicamente ya que según entendimos, por la dimensionalidad de los datos, no

pudimos usar otro tipo. De igual manera, logramos encontrar valores que optimizan el número de neuronas y capas, inclusive armando una función que indica el mejor umbral de decisión final según el conjunto de test.

Cuadro de Resultados

Modelo	CHPN	F1-Test	Precision Test	Recall Test	Accuracy Test	Kaggle
DecissionTree (sin poda)	2	0.851	0.880	0.824	0.846	0.843
Random Forest	3	0.8789	0.875	0.875	0.88	**0.8721**
Stacking	3	0.8788	0.88	0.88	0.88	0.8707
XGBoost	3	0.8778	0.875	0.875	0.88	0.8690
Red Neuronal	4	0.853	0.82	0.887	0.845	0.845

- **Decision Tree:** Se basa en la idea de elegir la característica que mejor divide al conjunto de datos en subconjuntos tan puros como se indique con los diferentes hiperparametros.
- **Random Forest:** Algoritmo de conjunto que combina múltiples árboles de decisión. Cada árbol se entrena con una muestra aleatoria del conjunto de datos, y luego se combinan la salida de los árboles para tomar decisiones. **Fue el modelo que mejor resultados nos dio, con un score 0.87208 en Kaggle.**
- **XGBoost:** Es un método de aprendizaje automático supervisado para clasificación y regresión basado en el Boosting, en este caso usado para clasificación. Esto es una forma de aprendizaje de los modelos basada en generar predicciones y asignar un mayor peso a aquellas mal clasificadas, así iterativamente hasta un límite determinado por los hiper parámetros.
- **Red Neuronal:** Consiste en un conjunto de neuronas artificiales conectadas entre sí para transmitir diferentes tipos de información, la cual ingresa por las neuronas de entrada y atraviesa la red neuronal donde se le realizan distintas operaciones (métodos de regularización, backpropagation) y finalmente produciendo valores de salida según, entre otras cosas, una función de activación.

Conclusiones generales

El hecho de realizar feature engineering nos resultó útil ya que nos ayudó a comprender con detalle el significado y que tipo de variables diferentes teníamos para trabajar en la creación de modelos. En esta misma etapa al momento de tener que eliminar algunas variables, en principio irrelevantes, llegamos a la conclusión de que era muy prematuro como para eliminar “definitivamente” alguna variable y preferimos realizarlo habiendo “jugado” un poco más con estas al momento de la creación de modelos.
Al momento comenzar crear nuestro primer modelo para predecir no realizamos muchas modificaciones sobre el dataset en sí, lo cual se tradujo en una calificación de 0.737 en Kaggle, la cual no era baja para un primer modelo pero no era lo esperable teniendo en cuenta que habíamos obtenido alrededor de 0.80 de f1-score en test. Debido a esto generamos nuevas variables, realizando operaciones aritméticas y comparativas sobre otras variables para que estas sean más determinantes en general al momento de predecir. Como consecuencia mejoró el puntaje de Kaggle significativamente y hubo menos diferencia respecto a las métricas obtenidas en test, además encontramos que al generar estas nuevas variables algunas de los originales resultaban ser redundantes y hasta podían generar overfitting por lo que decidimos eliminarlas.
Nuestro modelo con mejor desempeño tanto en test como en Kaggle fue el Random Forest con una puntuación de 0.8789 y 0.8721, respectivamente. Y el modelo con peor desempeño tanto en test como en Kaggle fue el KNN con una puntuación de 0.8399 y 0.8369, respectivamente.
Uno de los modelos más lentos a la hora de buscar hiperparametros a través de cross validation fue KNN, sin tener en cuenta que con las SVM que aun habiéndolas dejado horas, no llegamos a obtener un resultado, y contrario a como uno esperaría tuvo de los desempeños más pobre de todo el trabajo.
Nos hubiera gustado tener la posibilidad de experimentar más con variables continuas para así poder aplicar técnicas vistas en clase principalmente referidas a la correlación y podes buscar también agrupamiento de datos, nos notamos recurriendo mucho a variables booleanas y categóricas por falta de variables de rango continuo (tuvimos por ejemplo bebés y adultos como continuas pero de rangos muy acotados). Por la naturaleza del problema también, no pudimos hallar lugar para tratar de aplicar técnicas de reducción de dimensionalidad como visto en clase.

Tareas Realizadas

Teniendo en cuenta que el trabajo práctico tuvo una duración de 9 (nueve) semanas, le pedimos a cada integrante que indique cuántas horas (en promedio) considera que dedicó semanalmente al TP

Integrante	Promedio Semanal (hs)
Daniel Agustin Marianetti	12
Ezequiel Lazarte	15
Franco Ezequiel Rodriguez	10