.UBAfiuba
FACULTAD DE INGENIERÍA

Trabajo Práctico N°2

[75.06/95.58] Organización de Datos
Primer cuatrimestre de 2021

Alumno 1:	KELMAN, Axel
Número de padrón:	104379
Email:	akelman@fi.uba.ar
Alumno 2:	PACHECO, Federico Jose
Número de padrón:	104541
Email:	fpacheco@fi.uba.ar

Índice

1. Introducción

El objetivo del siguiente informe será mostrar los preprocesamientos utilizados durante la realización del trabajo práctico así como también las métricas obtenidas. Finalmente, estableceremos algunas conclusiones.

2. Preprocesamientos utilizados

Preprocesamiento	Explicación simple	Función en preprocessing.py
conversión numérica	convierte las features categóricas en numéricas, con noción de orden o no	conversion_numerica()
conversión numérica generalizado	ídem a la conversión numérica pero para el dataset generalizado	conversion_numerica_generalizado()
expansión polinomial númericas	expande solo las variables numéricas del dataset según combinaciones polinomiales	get_dataframe_polynomial()
expansión polinomial todas	expande todo el dataset según combinaciones polinomiales	get_dataframe_polynomial$_all$()
StandardScaler	escala los datos de forma estándar	get_dataframe_scaled(dataset, scaled = StandardScaler())
MinMaxScaler	escala los datos en un determinado rango (default 0,1)	get_dataframe_scaled(dataset, scaled = MinMaxScaler())
Normalizer	normaliza todas las instancias	get_dataframe_scaled(dataset, scaled = Normalizer())
PowerTransformer	aplica una transformación a los datos para que se vuelvan mas Gaussianos	get_dataframe_scaled(dataset, scaled = PowerTransformer())
RobustScaler	escala los datos utilizando métodos estadísticos, volviéndolos robusto a outliers	get_dataframe_scaled(dataset, scaled = RobustScaler())
obtención discreta	obtiene el dataset con todas las features discretas	obtener_features_discretas()
obtención continuos	obtiene el dataset con todas las features continuas	obtener_features_continuas()
reducción rfecv	reduce dataset según el algoritmo de RFECV	reduccion_rfecv()
reducción numérica	reduce dataset numérico según un porcentaje de varianza deseada	reduccion_numerica()

3. Modelos y métricas

Modelo	Preprocesamiento	AUC-ROC	Accuracy	Precision	Recall	F1 score
Decision Tree	conversión numérica	0.9014	0.8563	0.79	0.54	0.65
KNN	conversión numérica + StandardScaler	0.8840	0.8413	0.72	0.56	0.63
Naive Bayes	conversión numérica + expansión polinomial todas	0.8752	0.7798	0.53	0.80	0.64
SVM	conversión numérica + MinMaxScaler	0.8959	0.8455	0.74	0.55	0.63
Red Neuronal	conversión numérica + StandardScaler	0.9054	0.8507	0.72	0.62	0.67
Boosting	conversión numérica	0.9296	0.8719	0.78	0.65	0.71
Stacking	conversión numérica + StandardScaler	0.9062	0.8512	0.76	0.56	0.64

Las métricas precision, recall y F1 score corresponden a la clase de altos ingresos

4. Conclusiones

- El modelo que recomendamos a nivel general es el número 6, boosting con árboles de decisión. En particular, es el que mejor métrica AUC-ROC tiene y además posee mejor balance de precisión y recall
- El modelo que mejor serviría en caso de querer tener la menor cantidad de falsos positivos se tratá del que tiene mejor precisión para la clase de altos ingresos. En este caso se trata de el Decision Tree que figura en primer lugar en la tabla, que posee 0.79 de score en la precisión para la clase de altos ingresos
- El modelo que mejor serviría en caso de querer atrapar la mayor cantidad de casos de la clase de altos ingresos sin preocuparse por los falsos positivos, se tratá del modelo con mayor recall para esta clase. En particular, el modelo es el SVM que se puede encontrar en el notebook hacia el final previo el holdout (5.2.2). El mismo utiliza SVM con kernel polinomial y llega a un valor de 0.89 de recall para la clase de altos ingresos. Cabe destacar, que esta es la única métrica en la que destaca este modelo ya que a nivel score se encuentra muy por debajo de los obtenidos en la tabla