Universidad de Buenos Aires - Facultad de Ingeniería
Organización de Datos 75.06/95.58

Informe trabajo práctico Nº2

Lucas Risaro - Javier Nuñez Leyes

94335 - 94455

Índice

Informe TP2

Introducción

A continuación se enuncian los detalles de los modelos utilizados para realizar las predicciones solicitadas sobre el set de datos entregado.

Se listan los nombres, métodos utilizados y resultados obtenidos de cada uno junto con las métricas que se tuvieron en cuenta para decidir cuál de los modelos fue el mejor.

La información se mostrará en dos tablas, una que listara los métodos de preprocesamiento y otra con la información pertinente de cada modelo.

Tabla de métodos de preprocesamiento

Nombre del preprocesamiento	Explicación	Función de python
Preprocesamiento General de los modelos XGBOOST y Random Forest	Función donde se realiza todo el feature engineering sobre el set de datos. Se unifican y eliminan columnas y se aplica One Hot Encoding a las definitivas.	feature_engineering_xg_rf()
One Hot Encoding	Aplica One Hot Encoding a las columnas y set que se le especifique	apply_one_hot_encoding()
Setear rango de edad	Crea una columna si el usuario pertenece a un cierto rango de edad	set_age_range()
Codificación de combinación de valores de "trabajo" y "rol_familiar_registrado".	Función que devuelve un 1 si esa row posee la combinación rol_familiar_registrado = "casado" y (trabajo = "profesional_especializado" o trabajo = "directivo_gerente")	set_value_row_casado_trabajo()
Unifica valores Casado y Casada	Unifica los valores casado y casada de la columna rol_familiar_registrado en casado	unificar_values_casado_casada()

Obtener columnas por índice	Devuelve las columnas del set de datos indicadas por sus indices	get_columns_by_index
Preprocesamiento de todas las columnas para KNN, Naive Bayes y SVM	Convierte las columnas a valores numéricos, unifica valores categóricos y aplica one hot encoding.	feature_engineering_KNN_SVM_Naive_Bayes
Preprocesamiento usado en la primera parte del TP	Unifica valores, aplica one hot encoding y elimina las columnas innecesarias (según análisis de la primera parte)	feature_engineering_TP_primera_parte

Tabla de modelos

Nombre de modelo	**Preprocesamiento**	**AUC-ROC**	**Accuracy**	**Precisión**	**Recall**	**F1-Score**
Random Forest	Preprocesamiento General de los modelos XGBOOST y Random Forest	0.657	0.819	0.782	0.969	0.479
1-XGBoost	Preprocesamiento General de los modelos XGBOOST y Random Forest	0.881	0.881	0.767	0.956	0.564
2-XGBoost	Preprocesamiento General de los modelos XGBOOST y Random Forest Setear rango de edad	0.888	0.888	0.752	0.949	0.581
3-XGBoost	Preprocesamiento General de los modelos XGBOOST y Random Forest Obtener columnas por índice	0.710	0.710	0.737	0.995	0.070

1-KNN	Preprocesamiento de todas las columnas para KNN, Naive Bayes y SVM	0.736	0.817	0.617	0.888	0.600
2-KNN	Preprocesamiento usado en la primera parte del TP	0.738	0.765	0.512	0.790	0.586
1-Naive Bayes	Preprocesamiento de todas las columnas para KNN, Naive Bayes y SVM	0.754	0.734	0.463	0.716	0.584
2-Naive Bayes	Preprocesamiento usado en la primera parte del TP	0.754	0.734	0.463	0.716	0.584
1-SVM	Preprocesamiento de todas las columnas para KNN, Naive Bayes y SVM	0.703	0.831	0.729	0.947	0.563
2-SVM	Preprocesamiento usado en la primera parte del TP	0.584	0.777	0.609	0.956	0.315

Conclusión

Mejor modelo

En base al área bajo la curva AUC-ROC (0.888), el mejor modelo es XGBoost version 2. Este número nos indica que tan bueno es nuestro modelo para distinguir entre las clases. Cuanto más alto es el valor de AUC, mejor el modelo está prediciendo los 0 como 0 y los 1 como 1.

Con respecto al baseline del TP 1 este está fuertemente ligado a los datos de “entrenamiento”, es decir que el “modelo” con “IFs” tiene problemas de overfitting, no es capaz de predecir de manera correcta nuevos sets de datos ya que memoriza los datos que uso para su entrenamiento.

Análisis de falsos positivos

Para recomendar un modelo basado en el manejo de los falsos positivos analizaremos el recall y la precisión de cada uno.
Si no nos importa tener entre nuestras predicciones un número alto de falsos positivos nos fijaremos en el recall del modelo en cambio si lo que queremos es un modelo que posea pocos falsos positivos entre sus predicciones nos concentraremos en la precisión del modelo.

Al tener una alta precision nos aseguramos que la gran mayoría de resultados que el modelo dice que son positivos efectivamente lo son, quiere decir que el modelo devuelve pocos falsos positivos. Pero esto hace que en el esfuerzo de asegurar esos resultados como positivos se le escapen otros y lo predigan como negativos, es decir tendría más falsos negativos. Lo que podría ocasionar que su efectividad en la predicción general disminuya.

Al tener un alto recall el modelo devuelve mucho más resultados positivos pero esto hace que tenga muchos falsos positivos entre ellos.

Entonces viendo los resultados detallados en las tablas anteriores si lo que el usuario necesita es un modelo que posea la menor cantidad de falsos positivos posibles, corriendo el riesgo de tener muchos falsos negativos, entre sus predicciones le recomendamos usar Random Forest, que de los modelos probados es el que mayor precisión tiene.

En cambio si lo que queremos es obtener la mayor cantidad de predicciones positivas sin preocuparnos por que un gran número de ellas en realidad no seas positivas recomendamos el modelo con mayor Recall, en nuestro caso XGBoost version 3 que posee un recall de 0,995.