.UBAfiuba
FACULTAD DE INGENIERÍA

Trabajo Práctico 2
Machine Learning

[7506] Organización de datos
Curso Collinet
Segundo cuatrimestre de 2020

Alumno	Padrón	Email
Grassano, Bruno	103855	bgrassano@fi.uba.ar
Romero, Adrián	103371	adromero@fi.uba.ar

Índice

1. Introducción

En el presente trabajo se propone realizar una expansión del análisis realizado a los datos recolectados para la primera fase. Gracias al éxito logrado en la primera campaña de la empresa, éstos decidieron probar las avanzadas técnicas de inteligencia artificial. La empresa espera con esto poder profundizar aún más su campaña de marketing mediante el uso de un modelo de machine learning.

2. Objetivos

Los objetivos de este trabajo práctico son:

- Entender los diferentes modelos de machine learning vistos a lo largo de la cursada.
- Familiarizarse con las diversas métricas presentadas y aprender a compararlas entre si.
- Entender y aprender distintas formas de preprocesar los datos.
- Aprender los caminos que hay para obtener las mejores combinaciones de hiperparámetros en los diferentes modelos.

3. Archivos presentados

Se dejan a continuación la lista de archivos presentados junto con una breve explicación.

3.1. Notebooks

Cada notebook contiene el análisis realizado por su respectivo modelo.

- 01 - NaiveBayes.ipynb
- 02 - ArbolDeDecision.ipynb
- 03 - Support Vector Machines.ipynb
- 04 - KNN.ipynb
- 05 - RegresionLogistica.ipynb
- 06 - RandomForest.ipynb
- 07 - Boosting.ipynb
- 08 - Voting.ipynb
- 09 - RedesNeuronales.ipynb
- 10 - Stacking.ipynb

3.2. Auxiliares

- preprocessing.py - Contiene los diferentes preprocesamientos utilizados a lo largo de los notebooks.
- funcionesAuxiliares.py - Contienen funciones que se utilizan en varios notebooks. Fueron puestas en este archivo para evitar estar repitiendo su código en los notebooks.
- requirements.txt - Contiene la lista de requerimientos para correr los distintos notebooks.
- Carpeta PrediccionesHoldout - Contiene los archivos csv con los resultados de los modelos sobre el Holdout entregado

4. Tablas

4.1. Tabla 1: Preprocesamientos

Nombre	Explicación	Nombre de la función
PrepararValidacion	Convierte la variable target para ser usada en los modelos.	prepararSetDeValidacion
PrepararDatos	Aplica el feature engineering relizado en el TP1.	prepararSetDeDatos
PrepararHoldout	Llama a PrepararDatos y cambia 'atrás' por 'No responde'.	prepararSetDeHoldout
Expansión	Creara nuevas features a partir del dataset.	expansionDelDataset
Numéricas	Convierte a números las variables, aplica OneHot en las que corresponda.	conversionAVariablesNumericas
Normalizadas	Llama a Numéricas y normaliza el resultado.	conversionAVariablesNormalizadas
Codificadas	Se queda con las variables indicadas y las codifica de forma ordenal.	conversionAVariablesCodificadas
Continuas	Se queda con las variables continuas.	conversionAVariablesContinuas

4.2. Tabla 2: Métricas de los modelos

Presentamos la tabla ordenada según la métrica AUC-ROC. Para la toma de los valores de Precision, Recall, y F1 Score se tomaron los de los promedios ponderados.

Todos los modelos pasaron primero por el preprocesamiento básico llamado 'PrepararDatos', por lo que no se agrega esto en la columna indicada. Se agregan las conversiones realizadas y si corresponde el llamado a otro preprocesamiento (Expansion). Algo a tener en cuenta, es que algunos preprocesamientos aceptan parámetros para modificar el resultado. Estas combinaciones tampoco se incluyen en la tabla.

Nombre	Preprocesamiento	AUC-ROC	Accuracy	Precision	Recall	F1 Score
08 - Voting	Normalizadas	0.890	0.840	0.840	0.840	0.830
10 - Stacking	Normalizadas	0.887	0.840	0.840	0.840	0.830
07 - Boosting	Expansion & Numéricas	0.878	0.820	0.820	0.820	0.820
09 - Redes Neuronales	Normalizadas	0.877	0.820	0.820	0.820	0.820
06 - Random Forest	Normalizadas	0.876	0.840	0.850	0.840	0.840
05 - Regresión Logística	Expansion & Normalizadas	0.871	0.820	0.810	0.820	0.810
02 - Árbol de decisión	Expansion & Numéricas	0.868	0.840	0.840	0.840	0.840
01 - CategoricalNB	Expansion & Codificadas	0.867	0.810	0.810	0.810	0.810
03 - SVM	Normalizadas	0.862	0.790	0.780	0.770	0.780
01 - 'Ensamble' Gaussiano	Codificadas	0.856	0.790	0.790	0.790	0.790
01 - MultinomialNB	Codificadas	0.851	0.780	0.770	0.780	0.770
04 - KNN	Normalizadas	0.811	0.770	0.760	0.770	0.760
01 - GaussianNB	Continuas	0.657	0.660	0.670	0.660	0.610

5. Conclusiones

Como conclusiones del trabajo práctico, destacamos que casi todos los modelos implementados tuvieron un buen rendimiento, estando este en el rango de 0.8 a 0.9 según la métrica AUC-ROC (salvo por GaussianNB debido a lo ya explicado en su Notebook). Debido a esto todos son candidatos validos a la hora de elegir el modelo que se utilizará en la siguiente campaña de marketing.

Habiendo visto los resultados de los diferentes modelos presentados y considerando que debemos elegir solo un modelo, llegamos a la conclusión que recomendamos utilizar el modelo de Voting, esto se debe a que presenta el mejor resultado de AUC-ROC, junto a un muy buen rendimiento en Accuracy, Precision, Recall, y F1 Score estando en 0.84.

Que Voting sea el modelo recomendado resulta esperable, debido a que es un ensamble que une los mejores modelos que se fueron realizando a lo largo del trabajo práctico. Al realizar esto, y estar trabajando en conjunto, se obtuvo un muy buen desempeño.

Si comparamos este modelo con el baseline implementado anteriormente, podemos observar que el Voting tiene un mejor resultado para Accuracy, la métrica presentada en el baseline. (0.81 a 0.84). Sin embargo, cabe recordar que el accuracy que se obtuvo en el baseline fue obtenido a partir de evaluar la clasificación sobre todo el set de datos, y no sobre una partición de los mismos.

La soluciones a los problemas recién mencionados se obtuvieron realizando *grid search cross validation*, tanto usando la función de *sklearn* como con implementaciones nuestras mediante ciclos for, con esto obtuvimos las mejores combinaciones de hiperparámetros para cada modelo en particular (probando en diferentes casos distintos preprocesamientos).

Una vez obtenidos, hicimos una división del set de datos en *train* y *test* (test del tamaño 0.25, aproximadamente 200 casos de evaluación). Con el set de datos, entrenamos el modelo y con el set de test evaluamos la predicciones realizadas y las métricas para el modelo. Por ultimo, realizamos las predicciones finales con el set de *holdout* entregado, siendo estos datos que el modelo nunca vio. Cabe destacar que la partición de los datos es pseudo-aleatoria, manejada por un parámetro de *random state* (igualado a 0 a lo largo del trabajo). Por lo tanto, al cambiar este parámetro es posible que se obtengan otros resultados para las métricas obtenidas. En muchos casos, para reducir las probabilidades de obtener una partición atípica se partieron los datos en *K-folds*.

Es destacable también el modelo de árbol de decisión conseguido. Si bien utilizando este modelo obtuvimos un AUC-ROC score promedio (0.868), las métricas de Accuracy, Precision, Recall y F1 Score se encuentran entre las más altas encontradas (0.84 todas). Esto, junto con la simplicidad y buena interpretabilidad lo hace un segundo modelo candidato a ser utilizado en la próxima campaña de marketing de la empresa, sobre todo considerando que es sencillo de explicar y que otorga también una idea de qué *features* son los más relevantes a la hora de predecir si alguien volvería o no.

Un tercer modelo destacado es el Random Forest, modelo con el cuál se obtuvo la mayor Precision. En caso de que a la empresa le interese que de las predicciones realizadas, la mayor parte de ellas este correctamente clasificada, este es un buen modelo. Sin embargo, consideramos que en general, a la empresa le interesará tener un buen Recall. Esto se debe a que esta métrica indica qué proporción de las personas que volverían se logran detectar. Si se logra detectar a una gran cantidad de personas que volverían, la publicidad que se envíe llegará a una mayor cantidad de personas a las que le pueda resultar interesante.

En cuanto a conclusiones personales y finales del trabajo, estuvo bueno tener la oportunidad de ir probando y familiarizarse con los distintos modelos presentados a lo largo de la cursada. Ver a lo largo del desarrollo del trabajo los diferentes caminos que hay para obtener mejores resultados e ir empujando mas arriba la meta, ya sea probando combinaciones de los hiperparámetros, o intentando diferentes preprocesamientos.