FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Trabajo Práctico 2
Profundizando la capacidad de predicción
Cuidado! Lluvia de Hamburgesas

[75.06/95.58] Organización de Datos
Cátedra Collinet
Segundo cuatrimestre de 2021

Integrantes:	
106004	Primerano Lomba, Franco Alejandro
105555	Montecalvo, Ignacio

Índice

1. Introducción

En dicho trabajo se realizará una continuación al análisis de datos ofrecidos por Flint Lockwood. Gracias al éxito logrado en la primera campaña él tiene más confianza en los "algoritmos" y está ansioso por probar las avanzadas técnicas de inteligencia artificial de las que todo el mundo habla.

Se propone buscar algún modelo que prediga con mayor acierto que el obtenido en el baselide si al siguiente día lloverá o no hamburguesas.

2. Objetivos

- Poner en práctica distintas técnicas de preprocesado de datos que permitan mejorar la performance
- Aplicar modelos de machine learning vistos en la materia
- Utilizar técnicas de selección de mejores hiperparámetros de un tipo de modelo.
- Utilizar diversas métricas para evaluar los distintos modelos y establecer una comparación entre ellos.

3. Archivos Presentados

3.1. Notebooks de modelos

- 01. Arbol de Decision.ipynb
- 02. KNN.ipynb
- 03. Support Vector Machines.ipynb
- 04. Random Forest.ipynb
- 05. Naive Bayes.ipynb
- 06. Boosting.ipynb
- 07. Red Neuronal.ipynb

3.2. Otros archivos

- preprocessing.py: Contiene funciones de preprocesamiento.
- auxiliares.py: Contiene funciones auxiliares usadas en el resto de notebooks.
- requirements.txt: Requerimientos necesarios para ejecutar sin errores los notebooks.
- Baseline.ipynb: Evalúa el baseline del TP 1 utilizando las diversas métricas empleadas en esta segunda parte.
- Obtencion datasets.ipynb: Se obtienen los subsets de training, val-dev y holdout a partir del dataset original y se aplica preprocesamiento básico tratando los missing values.
- Predicciones: Carpeta que contiene las predicciones realizadas por cada modelo.
- Datasets: Carpeta que contiene todos los distintos set de datos.

4. Preprocesamientos y Modelos

4.1. Tabla de preprocesamientos

Nombre	Explicación	Nombre de la función
Manejo de missing values	Se manejan los valores faltantes usando la misma logica empleada en el TP 1	manejo_missing_values()
Estandarización	A cada valor de un feature se le restará el promedio de ese feature y se lo dividirá por el desvio estandar	estandarizar()
One Hot Encoding	Se eliminará el dia por tener una cardinalidad alta. Se dejará una columna para los nans y una implicita.	aplicarOneHot()
PCA	Se devolverá uno nuevo con features según la cantidad de componentes.	aplicarPCA()
Normalización	Se aplicará un MinMaxEscaler, dejando cada feature con valores entre 0 y 1	normalizar_entre_0_y_1()
Normalización según valor absoluto	Se aplicará un MaxAbsEscaler, dejando cada feature con valores en un rango equivalente a su máximo valor absoluto	normalizar_segun_maximo _valor_absoluto()
Agregado de feature 'Estacion'	A partir del feature 'dia', se crea un nuevo feature que indica la estación del año en que se tomó el registro	agregar_feature_estacion()
Filtrado de features continuos	Se obtienen los features del dataset que toman valores continuos	obtener_features_continuos()
Filtrado de features categóricos	Se obtienen los features del dataset que toman valores categóricos	obtener_features_categoricos()
Filtrado de features discretos	Se obtienen los features del dataset que toman valores numericos discretos	obtener_features_discretos()
Filtrado de features por varianza	Se filtran los features que poseen una varianza menor a un umbral dado	filtrar_features_por_varianza()

4.2. Tabla de modelos

Nombre	Preprocesamiento	AUC-Score	Accuracy	Precision	Recall	F1
Boosting	Manejo de missing values, Agregado de feature 'Estacion', One Hot Encoding	0,8872	0,857	0,74	0,55	0,63
Random Forest	Manejo de missing values, One Hot Encoding	0,8615	0,841	0,79	0,39	0,52
KNN	Manejo de missing values, One Hot Encoding, PCA, Normalización	0,8530	0,839	0,77	0,39	0,52
Árbol de decision	Manejo de missing values, One Hot Encoding	0,8469	0,838	0,71	0,47	0,56
SVM	Manejo de missing values, One Hot Encoding, PCA, Normalización	0,8444	0,837	0,73	0,42	0,54
Naive Bayes	Manejo de missing values, Agregado de feature 'Estacion', One Hot Encoding, Filtrado de features	0.8301	0.8312	0.68	0.44	0.54
Redes Neuronales	Manejo de missing values, One Hot Encoding, PCA, Normalización	0,7168	0,847	0,75	0,47	0,58

5. Conclusiones

A partir de este trabajo, podemos concluir que en general todos los modelos (excepto la red neuronal) presentaron un AUC score y un Accuracy superior a 0.80, superando así lo pedido en la primera parte del TP. Sin embargo, el objetivo ahora constaba en mejorar estas predicciones, lo cual fue logrado notablemente por ciertos modelos.

El modelo que seleccionamos como 'modelo final' y recomendamos es Boosting. La razón de dicha decisión es que este modelo es el que presenta el AUC score más alto, el Accuracy más alto y también el F1 Score más alto, sobresaliendo notablemente del resto, siguiéndolo por debajo el modelo Random Forest. El resultado era esperado, pues ambos son ensambles de modelos más sencillos que al combinarlos nos permiten reducir la varianza del modelo final sin necesidad de aumentar el sesgo, por lo tanto, es esperable que tengan un mejor resultado.

A su vez, resulta interesante hacer una comparación entre estos modelos y el baseline inicial del TP 1, dado que éste ultimo presenta un nivel de simplicidad mucho mayor al de los modelos utilizados. Si observamos los scores correspondientes al baseline que se presentan en el respectivo notebook, observamos que los puntajes no están nada mal (excepto el AUC score) y que incluso puede ponerse a la altura de los modelos que peores scores obtuvieron, de modo tal que si analizamos la relación simplicidad-performance puede ser incluso más beneficioso utilizar el baseline que una red neuronal, SVM o Naive Bayes para este problema. Igualmente, no hay que dejar de tener en cuenta que para la construcción de dicho baseline se usó todos los datos, incluso con los que se está evaluando.

Por último, para el caso en que busquemos tener la menor cantidad posible de falsos positivos, debemos seleccionar el modelo que mayor precisión nos aporte, siendo éste Random Forest con una precisión de 0.79 y seguido por KNN con una precisión de 0.77. En cambio, si buscamos obtener todos los días que potencialmente lloverán hamburguesas al día siguiente sin preocuparse por considerar días que realmente no llovieron hamburguesas al día siguiente, seleccionaremos el modelo que nos aporte un mayor recall, siendo éste el Boosting con recall de 0.55, seguido por el de arbol de decisión y redes neuronales con 0.47. De todas formas, si buscamos un equilibrio, el

modelo más recomendable sería Boosting ya que es el que posee valores más aceptables tanto para recall como para precisión, haciendo que sea el modelo con el F1 score más alto.