ALGORITMOS DE APRENDIZAJE SUPERVISADO

- **K-NN**
- **SVM**
- **NAIVE BAYES**
- **RANDOM FORESTS**

ENSEMBLE

En este proyecto aprenderás a generar modelos entrenados mediante aprendizaje supervisado que puedan predecir información. En particular, vamos a trabajar con:

- K-NN
- SVM
- Naive Bayes
- Random forests

También aprenderás a crear un método de conjunto (ensemble), que estará basado en los modelos anteriormente mencionados.

Vamos a trabajar con las distintas versiones del conjunto de datos Iris que se crearon en la práctica anterior (original, estandarizado, normalizado, y las distintas versiones de PCA (originalPCA95, originalPCA80, estandarizadoPCA95, estandarizadoPCA80, normalizadoPCA95 y normalizadoPCA80)), y se realizará una validación cruzada de 5 iteraciones para evaluar el rendimiento de los modelos que se generen. Concretamente, los pasos de los que consta la validación cruzada son los siguientes:

- En la primera iteración de la validación cruzada, primero hay que cargar los conjuntos de datos de entrenamiento y test correspondientes a dicha primera iteración. Una vez cargados estos datos, entrena un modelo, guárdalo, testéalo con las diferentes métricas y guarda los resultados obtenidos; si tienes varios métodos, haz lo mismo.
- En la segunda iteración de la validación cruzada, primero hay que cargar los conjuntos de datos de entrenamiento y test correspondientes a dicha segunda iteración. Una vez cargados estos datos, entrena un modelo, guárdalo, testéalo con las diferentes métricas y guarda los resultados obtenidos; si tienes varios métodos, haz lo mismo.
- …
- Hacer lo mismo hasta la iteración k.
- Una vez ejecutadas las k iteraciones de la validación cruzada, para cada métrica se obtiene la media y la desviación típica de los resultados obtenidos. De esta forma, se puede decir que "usando una validación cruzada utilizando k pliegues, el método ha tenido un rendimiento medio de F-medida con un valor de XXX y una desviación típica de YYY".

CONSIDERACIONES

Cada iteración de la validación cruzada es cerrada en el sentido de que no comparte información con las otras iteraciones. Es decir, esto consiste en lo siguiente:
Para la primera iteración:
- Se forman los conjuntos de entrenamiento y test para la primera iteración a partir de las particiones.

- Entrenas los modelos desde cero con los conjuntos de entrenamiento de la primera iteración.
- Testeas los modelos de la primera iteración con el conjunto de test de la primera iteración y obtienes métricas.
De esta forma, los modelos no han visto las imágenes de test para su entrenamiento.
Pasamos ahora a la segunda iteración:
- Se forman los conjuntos de entrenamiento y test para la segunda iteración a partir de las particiones.
- Entrenas los modelos desde cero con los conjuntos de entrenamiento y validación de la segunda iteración.
- Testeas los modelos de la segunda iteración con el conjunto de test de la segunda iteración y obtienes métricas.
Nuevamente, estos modelos no han visto las imágenes de test para su entrenamiento.
Y así hasta completar el número de iteraciones, tras lo cual, cada muestra del conjunto de datos habrá formado parte del conjunto de test en una única iteración.

¿MÉTODO O MODELO?

¿Qué diferencia hay entre método (o algoritmo) y modelo? Por método se refiere, por ejemplo, a "K-NN", mientras que modelo de "K-NN" hay muchos, para ser exactos, caso de que hagas una validación cruzada, vas a generar un modelo por cada iteración de la validación cruzada. Así, el rendimiento del método "K-NN" será la media de los rendimientos de los diferentes modelos entrenados del método "K-NN".

PARAMETRIZACIÓN

Te recomiendo que parametrices muy bien tu código. Por ejemplo, incluye el parámetro --fold=XXX –method=XXX. Así, si se hace --fold=3 –method=knn, se estará ejecutando la tercera iteración de la validación cruzada con k pliegues que hayas estructurado previamente para el método K-NN. De igual forma, guarda de manera parametrizada toda la información que vayas generando. Por ejemplo, el archivo el archivo pred_3_norm_PCA95_knn contendrá las predicciones del modelo K-NN del conjunto de datos normalizado PCA95 correspondientes a la tercera iteración de la validación cruzada.

MÉTRICAS

Implementa todas las siguientes métricas:

Métrica	**Definición**
F1-score (Fm)	$Fm = 2 \times (PR \times RC) / (PR + RC)$
Sensibilidad (S)	$S = TP / (TP + FN)$
Exactitud (Acc)	$Acc = (TP + TN) / (TP + FP + FN + TN)$
Especificidad (SP)	$SP = TN / (FP + TN)$
Recall (RC)	$RC = TP / (TP + FN)$
Precisión (PR)	$PR = TP / (TP + FP)$
Tasa de falsos negativos (FNR)	$FNR = FN / (TP + FN)$
Tasa de falsos positivos (FPR)	$FPR = FP / (FP + TN)$

Obtén también la curva ROC y el AUC (área bajo la curva):
https://es.wikipedia.org/wiki/Curva_ROC

Aunque el accuracy es una medida que ofrece una visión general sobre el rendimiento del modelo, esta medida se ve distorsionada cuando el conjunto de datos está desbalanceado. Cuando el conjunto de datos es desbalanceado se debe utilizar otra medida de rendimiento general como referencia, como pueden ser el spatial_accuracy (cuando acertar la clase negativa no es importante) o, sobre todo, el f1-score.

Se pide desarrollar un código (archivo train.ipynb) que genere los modelos entrenados para cada iteración de la validación cruzada de cada uno de los algoritmos de aprendizaje supervisado considerados: K-NN, SVM, Naive Bayes y Random forests.

Se pide desarrollar un código (archivo eval.ipynb) que, para cada modelo, realice las predicciones de los tests en cada iteración de la validación cruzada y genere su rendimiento. A la hora de hacer las predicciones de cada muestra de test se tienen que obtener las probabilidades estimadas de pertenencia a cada clase y almacenarlas.

Se pide desarrollar un código (archivo results.ipynb) que cargue el rendimiento de cada modelo en cada iteración de la validación cruzada de cada conjunto de datos considerado (original, estandarizado, normalizado, y las distintas versiones de PCA (originalPCA95, originalPCA80, estandarizadoPCA95, estandarizadoPCA80, normalizadoPCA95 y normalizadoPCA80)). Tras esto, el código deberá crear:

- Para cada método, una tabla resumen forma que en cada fila aparezca el rendimiento del método con cada conjunto de datos considerado y en cada columna aparezca cada métrica. Cada celda tendrá el formato “media +- desviaciónTípica” (en Latex sería "media $\pm$ desviaciónTípica").
- Para la métrica f1-score, una tabla resumen forma que en cada fila aparezca el rendimiento del método con cada conjunto de datos considerado y en cada columna aparezca cada conjunto de datos considerado. Cada celda tendrá el formato “media +- desviaciónTípica” (en Latex sería "media $\pm$ desviaciónTípica").
- Genera las siguientes figuras:
    - FN contra FP
    - PR contra RC
    - ACC contra Fm

ENSEMBLE

Los métodos de conjunto (ensemble) combinan varios algoritmos de aprendizaje con el objetivo de lograr una capacidad predictiva superior a la que cada uno de ellos alcanzaría individualmente. Esa combinación se realiza mediante una función de agregación sencilla como la media, mediana, etc. o mediante una función más compleja.

En este caso, se piden implementar varios ensembles diferentes:

1) Votación: dada la clase predicha por cada modelo para una muestra de test, la predicción del ensemble será la clase predicha que tenga más votos. Por ejemplo, si K-NN, SVM y Naive Bayes predicen una muestra como clase Positiva, y Random forests predice dicha muestra como clase Negativa, la salida del ensemble será predecir clase Positiva, ya que esta clase cuenta con más votos (3) que la Negativa (1).
2) Media: dada la probabilidad de pertenencia a cada clase predicha por cada modelo para una muestra de test, la predicción del ensemble será la media de dichas probabilidades. Por ejemplo, si para la clase Positiva, K-NN, SVM, Naive Bayes y

Random forests predicen una probabilidad de pertenencia de 0.90, 0.80, 0.75 y 0.40, la salida del ensemble será predecir la probabilidad de pertenencia a clase Positiva de dicha muestra con un valor de (0.90 + 0.80 + 0.75 + 0.40) / 4 = 0.7125.

3) Mediana: igual que el ensemble Media, pero usando la función mediana en lugar de la media.

Finalmente, se tiene que redactar un informe con los resultados obtenidos y las conclusiones que se pueden extraer. El informe tiene que estar escrito en Latex usando la plantilla LNCS.

Se tiene que entregar un único archivo comprimido que contenga los siguientes archivos:

- El informe en formato pdf generado por el proyecto Latex.
- Archivo comprimido (.rar o .zip) con todos los archivos del proyecto Latex.
- El código (en formato ipynb) que se ha desarrollado.
- Un video (en formato mp4) con la ejecución del código.
- Declaración explícita en la que se asuma la originalidad del trabajo entregado.