Práctica 1: KNN y selección de atributos

Aprendizaje Automático II, 2025-2026

17 de septiembre de 2025

Como entrega de esta práctica deberá subirse a la plataforma Moodle un archivo comprimido (zip) que contenga los archivos *KNNClassifier.py* y *mRMR.py* con las implementaciones solicitadas, así como un *jupyter notebook* (.ipynb) con la solución de los ejercicios. En cada uno de estos ficheros debe incluirse un comentario con el nombre y apellidos de los integrantes del grupo.

La entrega será **antes de las 10:00** del día **2 de octubre para el grupo de los jueves**, y del día **3 de octubre para el grupo de los viernes**. Ese mismo día, durante la clase de prácticas, se realizará una prueba práctica de evaluación relacionada con la entrega.

1. Objetivos

Los objetivos de esta práctica son los siguientes:

- Implementar un algoritmo de KNN.
- Usar el KNN para realizar predicciones.
- Optimizar los hiperparámetros del KNN.
- Buscar los atributos más relevantes con mRMR y otros métodos.

Nota: Puede verificar su código ejecutando los tests proporcionados. Si tiene instalado *pytest*, basta con ejecutarlo desde su terminal. Estos tests no son exhaustivos, por lo que es recomendable que realice sus propias comprobaciones.

2. Descripción de la práctica

1. Implemente el algoritmo KNN completando la clase *KNNClassifier* en el archivo *KNNClassifier.py*.

   Tras realizar la implementación deberá resolver los siguientes problemas.

   a) Descargue los datos del siguiente problema relacionado con el cáncer de mama:

   ```
   from sklearn.datasets import load_breast_cancer
   data = load_breast_cancer()
   ```

   b) Preprocese el dataset siguiendo estos pasos: (1) separe los atributos de la etiquetas; (2) divida los datos en una partición con el 70 % de los puntos para *training* y el 30 % de los puntos para *test*; (3) normalice los datos.

   c) Si hubiera datos ausentes (*missing values*) y éstos se completaran, ¿cómo cree que influiría el orden en el que se realizan las operaciones de normalizar y completar? ¿Qué pasaría si primero se completan los datos ausentes y luego se normaliza? ¿Y si primero se normaliza y luego se realiza la partición *training-test*?

   d) A continuación, complete la clase *KNNClassifier*, cuyos atributos son el número de vecinos y una función distancia (una función cuyas entradas son dos vectores de la misma dimensión, y cuya salida es un número real positivo). Complete el constructor y los métodos *fit* y *predict*.

   e) Utilice la clase anterior para predecir las etiquetas de los datos de test, con un número de vecinos k, fijo pero arbitrario.

   f) Responda a la siguiente pregunta: ¿Qué ocurriría si hubiera un desbalanceo de clases en el conjunto de entrenamiento? Si esto supone un problema, ¿podría proporcionar una solución?

   g) Responda a la siguiente pregunta: ¿Cuál es el coste en memoria del algoritmo KNN? ¿Se le ocurre alguna forma de reducirlo?

   h) Utilice la función *KNeighborsClassifier* de la biblioteca de *sklearn* y, para el mismo número de vecinos k prediga las etiquetas del conjunto de test y_{predsk}. Si $y_{predcustom}$ son las prediciones de su modelo, ¿cuál es el error medio entre las predicciones y_{predsk} e $y_{predcustom}$? ¿Por qué?

   i) Se encuentra usted a un individuo que afirma que, en vez de utilizar KNN, él prefiere usar la siguiente alternativa. Para predecir la etiqueta del punto x, toma los tres puntos más cercanos a x en el conjunto de *training*. Si las distancias de estos tres puntos al punto x son d_1, d_2 y

d_3, y sus respectivas etiquetas son y_1, y_2 e y_3, la predicción vendrá dada por

$$f(x) = \text{sign}\left(\sum_{i=1}^{3} \frac{y_i}{d_i}\right). \tag{1}$$

¿Considera que este método es mejor que KNN con $k = 3$? ¿Por qué?

2. Optimización de KNN

   a) Encuentre, utilizando validación cruzada, el número de vecinos óptimos, k_{opt}.

   b) Dé la métrica de *accuracy* sobre el conjunto de *test* del clasificador KNN usando el valor k_{opt} obtenido.

   c) ¿Cree que el valor k_{opt} encontrado es el que proporciona mejor *accuracy* en el conjunto de *test*?

   d) Muestre en una gráfica el *accuracy* frente al número de vecinos, tanto para el conjunto de *training* como para el de *test*.

   e) Repita los experimentos anteriores utilizando la distancia de Minkowski para $p \in \{1, 2, 10\}$.

   f) ¿Cómo afecta el valor de p a los resultados? ¿Qué p cree que es mejor?

3. Selección de atributos. A continuación procederemos a la reducción de la dimensión de los datos.

   a) Usando el método *VarianceThreshold* de *sklearn.feature_selection* para cierto umbral fijo u, elimine los atributos que no superen dicho umbral.

   b) Analice cómo afecta la selección de atributos al accuracy del modelo. Para ello, fijado un valor de k para KNN, calcule *accuracy* en *test*. El conjunto de entrenamiento tendrá los atributos seleccionados para un umbral $u \in [0, 1]$ concreto. El resultado será una gráfica con el *accuracy* frente al umbral u.

   c) ¿Tienen sentido los casos $u = 0$ y $u = 1$?

   d) Ahora seleccione los mejores atributos del conjunto de datos utilizando `SelectKBest` de la librería `scikit-learn`. Siga los siguientes pasos:

      1) Importe el método `SelectKBest`:

         ```
         from sklearn.feature_selection import SelectKBest
         ```

2) Utilice una estadística univariada como `f_classif` para la selección de atributos:

```
from sklearn.feature_selection import f_classif
```

3) Seleccione los mejores `K` atributos:

```
selector = SelectKBest(score_func=f_classif, k=K)
selector.fit(X, y)
X_selected = selector.transform(X)
```

e) Combinando el método de selección de atributos con el clasificador KNN para un valor de k fijo, determinte cuál es el mejor valor de `K`. Nótese que k hace referencia al número de vecinos en KNN y `K` es el número de atributos seleccionado.

f) A continuación implemente el método de selección de atributos mRMR. Para ello, complete el archivo *mRMR.py*.

g) ¿Cuál es su mejor valor de k?

h) ¿Cuál es el papel de la información mutua en el método mRMR? ¿Se podría sustituir por otra métrica?

i) ¿Qué método de selección de atributos, de los dos utilizados, considera que es mejor?