

Examen - Parcial 1 - Aprendizaje de máquina - 6CV1

Torres Abonce Luis Miguel

1. ¿Qué estudia el Aprendizaje Automático o de Máquina (Machine Learning)?

¿Cuáles son sus paradigmas (supervisado, no supervisado y por refuerzo)? Puedes responder a las preguntas anteriores realizando un mapa mental o un resumen min. 1 cuartilla.

• El aprendizaje Automático es una rama de la inteligencia artificial que se centra en la construcción de sistemas que pueden aprender de los datos. En lugar de ser programados explícitamente, estos sistemas se diseñan para aprender y mejorar con la experiencia.

El aprendizaje automático estudia diversos aspectos:

• Modelado y optimización. Se enfoca en desarrollar modelos matemáticos y algoritmos que pueden aprender patrones.

• Generalización. Busca modelos que aprendan de manera efectiva a partir de conjuntos de datos específicos.

• Evaluación y validación. Evalúa la eficacia y la generalización de los modelos.

Paradigmas principales del aprendizaje automático

1. Aprendizaje supervisado. Este paradigma implica el entrenamiento de un modelo en un conjunto de datos etiquetados. Cada ejemplo en el conjunto de datos tiene una entrada y una salida correspondiente. El objetivo del modelo es que aprenda a mapear las entradas a las salidas correctas.

2. Aprendizaje no supervisado. En este paradigma, el modelo se entrena en un conjunto de datos sin etiquetas. El objetivo del modelo es que descubra estructuras interesantes en los datos por sí mismo.

3. Aprendizaje por esfuerzo. Este paradigma implica que un agente aprenda a tomar decisiones realizando acciones en un entorno. El agente recibe recompensas o penalizaciones en función de la calidad de sus acciones y su objetivo es maximizar la suma total de recompensas.

2. Explica el modelo de espacio vectorial el cual es ampliamente utilizado en el aprendizaje Automático Machine Learning

Elegir esta pregunta libre

3. ¿Cómo se calculan las medidas de precisión, especificidad (recall) y F1 (F-score) que permiten evaluar el rendimiento de un sistema de aprendizaje automático?

Se basan en la matriz de confusión

	Predicho +	Predicho -
Verdadero positivo	Casos + correctamente identificados	Casos + incorrectamente identificados
Verdadero negativo	Casos - correctamente identificados	Casos - incorrectamente identificados

Precisión

$$\frac{VP}{VP + FP} = \text{Precisión}$$

Mide la proporción de predicciones positivas que son correctas. Un valor Alto indica que el modelo tiene un bajo número de falsos negativos.

Especificidad (Recall)

$$\frac{VN}{VN + VP} = \text{Recall}$$

Mide la proporción de casos negativos que son correctamente identificados. Un valor alto indica que el modelo tiene un bajo número de falsos

F1 (F-score)

~~$\frac{VP + VN}{VP + VN + FP + FN}$~~

$$2 \left(\frac{\text{Precisión} \cdot \text{Especificidad}}{\text{Precisión} + \text{Especificidad}} \right)$$

Un valor alto indica un buen equilibrio entre ambas métricas

4. ¿En qué consiste la tarea de clasificación?
Menciona tanto las características principales como las diferencias entre las técnicas de clasificación supervisadas y no supervisadas.

No se trata de asignar una instancia de datos a una categoría o clase específica.

Características.

- Problema de aprendizaje supervisado. El modelo aprende a partir de un conjunto de datos con ejemplos etiquetados.
- Objetivo. Encontrar un modelo que pueda predecir la clase de nuevas instancias de datos no vistos anteriormente.
- Tipo de variables. La variable de entrada pueden ser numéricas, categóricas o textuales. La variable objetivo es categórica.
- Diversidad de aplicaciones. La clasificación se utiliza en: detección de correo no deseado, diagnóstico médico, reconocimiento facial, etc.

~~Características~~

Técnicas de clasificación:

Características	Supervisadas	No Supervisadas
Etiquetado de datos	Requiere datos con etiquetas de clase	No requiere datos etiquetados
Objetivo	Predecir la clase de nuevas instancias	Encontrar patrones y estructuras en datos
Ejemplo de algoritmos	Regresión logística, KNN, SVM, Redes neuronales	K-means clustering, agrupamiento jerárquico

5. Explica la técnica de validación cruzada de 10 iteraciones la cual sirve para medir la eficiencia de los clasificadores.

- Dividir el conjunto de datos en diez partes iguales o folds.
- Repetir este proceso diez veces:
 - Utilizar 9 de los folds como conjuntos de entrenamiento.
 - Utilizar el fold restante como conjunto de prueba.
 - Entrenar el clasificador en el conjunto de entrenamiento.
 - Evaluar el rendimiento del clasificador en el conjunto de prueba.
- Calcular la métrica de rendimiento promedio (precisión, F1-score) de las 10 iteraciones

6. ¿Cuál es el teorema de Bayes?

El teorema de Bayes es una fórmula fundamental en la teoría de la probabilidad que nos permite actualizar la probabilidad de un evento a la luz de nueva información.

$$P(A|B) = \frac{P(B|A) P(A)}{P(B)}$$

Donde:

- $P(A|B)$ es la probabilidad de ~~observar la evidencia B~~ la hipótesis A dado que observamos la evidencia B si la hipótesis A es verdadera.
- $P(B|A)$ es la probabilidad de observar la evidencia B si la hipótesis A es verdad

• $P(A)$ es la probabilidad de observar la evidencia A

• $P(B)$ es la probabilidad de observar la evidencia B

El teorema de Bayes nos permite calcular la probabilidad de una hipótesis después de haber observado alguna evidencia.

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

Librería scikit-learn

Make-classification

La función `make_classification` es una herramienta poderosa en el arsenal de `scikit-learn` para la generación de conjuntos de datos simulados destinados a problemas de clasificación.

Esta función es particularmente útil en escenarios donde se requieren crear datos de entrenamiento y prueba rápidamente para probar algoritmos de aprendizaje automático, validar modelos o realizar experimentos.

Parámetros importantes.

"n - samples"

• Este parámetro define la cantidad de muestras (instancias de datos) que se generarán en el conjunto de datos simulados. Es esencial para controlar el tamaño del conjunto de datos y puede ajustarse según el escenario particular del problema.

"n - features"

• Indica el número de características (o atributos) que tendrán las muestras generadas. Cada muestra es el conjunto de datos simulado, estará representado por un vector de características con este número de dimensiones. Controlar el número de características es crucial para simular conjuntos de datos realistas y representativos del problema.

"n - classes"

• Especifica el número de clases distintas en las que se pueden clasificar las muestras generadas. Este parámetro está determinando la complejidad del problema de clasificación simulado.

"n_clusters_per_class"

• Este parámetro controla la cantidad de grupos (o clusters) de puntos por clase. Si se establece `n_clusters_per_class=1`, las muestras se distribuirán uniformemente entre todas las clases, pero si se aumenta este valor, las clases pueden contener subgrupos.

"random_state"

• Si se proporciona un valor entero, se utilizará como semilla para el generador de números pseudoaleatorios. Esto asegura que el conjunto de datos generado sea reproducible.

Código Python.

```
from sklearn.datasets import make_classification
import matplotlib.pyplot as plt

X, y = make_classification(n_samples=1000, n_features=20,
                           n_classes=3, n_clusters_per_class=2, random_state=42)

plt.figure(figsize=(8, 6))
plt.scatter(X[:, 0], X[:, 1], c=y, cmap='viridis', s=50, alpha=0.7)
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.title('Clasificación Sintética generada')
plt.colorbar(label='Clase')
plt.show
```

Train_test_split

Esta función se utiliza para dividir un conjunto de datos en dos subconjuntos: uno para entrenar el modelo (conjunto de entrenamiento) y otro para evaluar su rendimiento (conjunto de prueba). Este enfoque permite estimar cómo se generalizará el modelo a nuevos datos, lo que es fundamental para evaluar su capacidad predictiva.

Parámetros importantes

• Arrays.

- Este parámetro representa los arrays (o matrices) que se dividirán en conjuntos de entrenamiento y prueba. Por lo general, se proporcionan dos arrays: uno para las características (X) y otro para las etiquetas (y).

test_size

- Indica el tamaño del conjunto de prueba con el conjunto de datos original. Puede especificarse como un número Flotante entre 0 y 1, lo que representa el porcentaje del conjunto de datos original.

train_size

- Define el tamaño del conjunto de entrenamiento. Si no se especifica, se calcula automáticamente como complemento del conjunto de prueba.

random_state

- Se proporciona un valor entero, se utilizará como semilla para el generador de números pseudoaleatorios. Esto garantiza que la división del conjunto de datos sea reproducible.

Shuffle

- Un Booleano que indica si se deben barajar aleatoriamente los datos antes de dividirlos. Por defecto está establecido en "True".

Ejemplo python

```
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_iris

iris = load_iris()
X = iris.data
y = iris.target

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.2, random_state=42)

print('Forma del conjunto de entrenamiento (X):', X_train.shape)
print('Forma del conjunto de prueba (X):', X_test.shape)
print('Forma del conjunto de entrenamiento (y):', y_train.shape)
print('Forma del conjunto de prueba (y):', y_test.shape)
```

accuracy_score

Es una herramienta comúnmente utilizada en scikit-learn para evaluar el rendimiento de un modelo de clasificación. Proporciona una medida simple pero importante de la precisión del modelo al calcular la fracción de muestras clasificadas correctamente.

Parámetros importantes

y_true

- Este parámetro representa las etiquetas verdaderas (o valores reales) de las muestras. Debe ser un array o una lista que contenga las etiquetas verdaderas para cada muestra en el conjunto de datos.

y_pred

- Representa las etiquetas predichas por el modelo para las muestras. Debe ser un array o una lista que contenga las etiquetas predichas para cada muestra en el conjunto de datos.

normalize

- Un booleano que indica si se debe devolver la precisión como un valor normalizado o como un recuento de muestras correctas.

Ejemplo python

```
from sklearn.metrics import accuracy_score
```

```
y_true = [0, 1, 1, 0, 1, 1, 0, 1]
```

```
y_pred = [0, 1, 1, 0, 0, 1, 0, 1]
```

```
accuracy = accuracy_score(y_true, y_pred)
```

```
print("precisión del modelo:", accuracy)
```


8. Implementación de un clasificador Naive Bayes en Python

Explica ¿para que sirven `fit()` y `predict()`?
Anota una descripción de sus parámetros más importantes
Muestra un ejemplo que los utilice en python

- `fit()`: Entrena el modelo con un conjunto de datos de entrenamiento. Este conjunto de datos debe contener las variables independientes (atributos) y la variable dependiente (clase).

- `predict()`: Utiliza el modelo entrenado para predecir la clase de nuevas instancias de datos.

Parámetros importantes

- `data`: El conjunto de datos utilizado para entrenar el modelo o para realizar las predicciones.
- `target`: La variable dependiente (clase) del conjunto de datos.
- `alpha`: Un parámetro de regularización que ayuda a evitar el sobreajuste.
- `priors`: Las probabilidades a priori de las diferentes clases.

Ejemplo de uso:

```
from sklearn.naive_bayes import GaussianNB
data = pd.read_csv("data.csv")
X = data.drop('clase', axis=1)
y = data['clase']
modelo = GaussianNB()
modelo.fit(X, y)
predicciones = modelo.predict(X)
print(classification_report(y, predicciones))
```