Escuela de Ciencias Físicas y Matemática
Ciencia de Datos • Aprendizaje Automático Inicial
Clase 03: Conjuntos de Entrenamiento y Validación
Andrés Merino • Semestre 2024-2

Resultado de Aprendizaje

RdA de la asignatura:

- Plantear los conceptos fundamentales del aprendizaje automático, incluyendo los principios básicos, técnicas de preprocesado de datos, métodos de evaluación y ajuste de modelos, destacando su importancia en el análisis y resolución de problemas de datos.

RdA de la actividad:

- Comprender las técnicas principales de preprocesado de datos, como limpieza, normalización, discretización y reducción de dimensionalidad.
- Aplicar métodos de partición de conjuntos de datos, como leave-one-out, leave-p-out y k-fold cross-validation.
- Identificar la importancia de los conjuntos de validación en el proceso de ajuste y evaluación de modelos.

Introducción

Pregunta inicial: ¿Cómo podemos asegurar que nuestro modelo generalice bien y no simplemente memorice los datos?

Desarrollo

Actividad 1: Preprocesado de Datos (30 minutos)

En esta actividad, los estudiantes aprenderán las técnicas básicas de preprocesado de datos, incluyendo limpieza, normalización, discretización y reducción de dimensionalidad. Estas técnicas son fundamentales para preparar los datos para modelos de aprendizaje automático, asegurando su calidad y optimizando su representación.

Recursos:

- Presentación teórica sobre preprocesado de datos.
- Jupyter Notebook con ejemplos de preprocesado utilizando `scikit-learn`.
- Conjunto de datos Iris (disponible en `scikit-learn`).

Ejercicio práctico:

```python
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.preprocessing import StandardScaler, KBinsDiscretizer
from sklearn.decomposition import PCA

# Cargar datos
data = load_iris()
X, y = data.data, data.target

# Normalización
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Discretización
discretizer = KBinsDiscretizer(n_bins=3, encode='ordinal', strategy='uniform')
X_discretized = discretizer.fit_transform(X_scaled)

# Reducción de dimensionalidad
pca = PCA(n_components=2)
X_reduced = pca.fit_transform(X_discretized)

print("Datos preprocesados:")
print(X_reduced[:5])
```

Verificación de aprendizaje: Con un conjunto de datos con al menos 5 características numéricas y 1 categórica (puede ser proporcionado o descargado de una fuente como `Kaggle` o `UCI Machine Learning Repository`). Deben realizar las siguientes tareas:

1. Identificar valores faltantes, valores atípicos y datos redundantes, y proponer una estrategia para manejarlos.
2. Estandarizar las variables numéricas y codificar las categóricas (por ejemplo, con `One-Hot Encoding` o `Label Encoding`).
3. Aplicar reducción de dimensionalidad utilizando `PCA` o seleccionando un subconjunto de características basado en correlaciones.

Actividad 2: Conjuntos de Entrenamiento y Test (40 minutos)

En esta actividad, se abordará la importancia de dividir un conjunto de datos en entrenamiento, prueba y validación. Se explicarán los métodos más comunes: `train_test_split`, k-fold cross-validation, leave-one-out y leave-p-out.

Recursos:

- Presentación teórica sobre partición de datos.
- Jupyter Notebook: 03-Conjuntos-Entrenamiento-Prueba.

Ejercicio práctico:

```python
from sklearn.model_selection import train_test_split, KFold, cross_val_score
from sklearn.ensemble import RandomForestClassifier

# Dividir datos en entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X_reduced, y, test_size=0.2, random_state=42)

# Validación cruzada
kfold = KFold(n_splits=5, shuffle=True, random_state=42)
model = RandomForestClassifier()
scores = cross_val_score(model, X_train, y_train, cv=kfold)

print("Scores de validación cruzada:", scores)
print("Promedio de scores:", scores.mean())
```

Verificación de aprendizaje: Con un conjunto de datos (por ejemplo, el conjunto *Titanic*), realice las siguientes actividades:

1. Implementar al menos tres métodos de partición:
   - División simple (`train_test_split`).
   - Validación cruzada estratificada.
   - k-fold cross-validation (sin entrenar modelos).
2. Dividir el conjunto de datos utilizando cada método y observar las proporciones de clases en cada subconjunto.

CIERRE

Tarea: Aplicar todo lo aprendido en un conjunto nuevo de datos.

Pregunta de investigación:

1. ¿Qué estrategias existen para manejar conjuntos de datos desbalanceados al realizar particiones de entrenamiento y prueba?
2. ¿Cómo afecta el tamaño relativo de los conjuntos de entrenamiento y prueba al desempeño de un modelo?
3. ¿Qué tipos de problemas se pueden enfrentar al evaluar modelos con conjuntos de datos extremadamente pequeños? ¿Cómo puede abordarse este desafío?