COMPLEJIDAD DE DATOS

Rodrigo Quiñones Mayorga

Uno de los problemas mas grandes a los que nos enfrentaremos será el trabajar con datasets desbalanceados, donde una o más clases tienen significativamente más instancias que otras. Esto es problemático porque los algoritmos de aprendizaje automático tienden a estar sesgados hacia la clase mayoritaria, lo que puede llevar a un mal desempeño en la predicción de las clases minoritarias.

Además, muchos datasets pueden contener **valores perdidos (missing values)**, lo que también puede afectar negativamente el desempeño del modelo. Estos valores pueden surgir por errores de registro, falta de datos o fallas técnicas, y deben ser manejados adecuadamente antes de entrenar el modelo.

El dataset utilizado es el Breast Cancer Dataset proporcionado por scikit-learn. Este dataset contiene características extraídas de imágenes digitales de tumores de mama y su clasificación como benignos o malignos. Las características son medidas físicas como el tamaño del tumor, la textura, y otros atributos relacionados.

1. Cargar el Dataset

Se cargó el dataset utilizando la función load_breast_cancer de la biblioteca scikit-learn. El dataset consta de las siguientes partes:

- **Características (X)**: Atributos del tumor como radio, textura, área, entre otros.
- **Etiquetas (y)**: 0 para tumores benignos y 1 para tumores malignos.

```
data = load_breast_cancer()
X = pd.DataFrame(data.data, columns=data.feature_names)
y = pd.Series(data.target, name='target')
```

2. Exploración Inicial del Dataset

Se realizó una exploración inicial para visualizar las primeras filas de las características del dataset y la distribución de las clases en las etiquetas.

Resultado: Se observó que el dataset está desbalanceado. Hay una mayor cantidad de casos de tumores benignos (clase 0) en comparación con tumores malignos (clase 1).

```
print(X.head())
print(y.value_counts())


print(X.isnull().sum())
```

3. Verificación de Valores Perdidos

Para asegurar la calidad de los datos, se verificó si había valores faltantes en las características del dataset. Se utilizó la función isnull().sum() para contar los valores perdidos en cada columna.

```
print(X.isnull().sum())
```

4. División del Dataset en Entrenamiento y Prueba

Se dividió el dataset en dos subconjuntos: entrenamiento (80%) y prueba (20%). Esta separación asegura que el modelo sea evaluado en datos que no ha visto durante su entrenamiento, proporcionando una medición precisa de su rendimiento.

```
from imblearn.over_sampling import SMOTE
from sklearn.model_selection import train_test_split


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, ranc
```

5. Aplicación de SMOTE para Sobremuestreo

Dado que el dataset original estaba desbalanceado, se aplicó la técnica de SMOTE para balancear las clases en el conjunto de entrenamiento. SMOTE genera nuevas muestras sintéticas de la clase minoritaria, creando un conjunto de datos más balanceado sin duplicar simplemente las instancias existentes.

```
smote = SMOTE(random_state=42)
X_train_res, y_train_res = smote.fit_resample(X_train, y_train)
```

Se verificó la nueva distribución de clases. Ahora ambas clases (benigno y maligno) tienen el mismo número de muestras, lo cual es óptimo para evitar el sesgo del modelo hacia la clase mayoritaria.

6. Verificación de Valores Perdidos en el Dataset Balanceado

Como paso adicional, se verificó nuevamente si el dataset balanceado contenía valores perdidos. Esto asegura que SMOTE no haya introducido ningún valor nulo durante el proceso de sobremuestreo.

7. Guardar el Dataset Transformado

Finalmente, se guardó el dataset transformado (con las clases balanceadas) en un archivo CSV. Este archivo contiene tanto las características como las etiquetas balanceadas, lo cual es esencial para su posterior uso en el entrenamiento de modelos de clasificación.

```
X_train_res['target'] = y_train_res
X_train_res.to_csv('dataset_transformado.csv', index=False)  # Guardar el dataset como CSV

print("Dataset transformado guardado como 'dataset_transformado.csv'")
```

Conclusión

El trabajo abordó dos problemas comunes en los datasets de clasificación: el desbalance de clases y la presencia de valores perdidos. Utilizando la técnica SMOTE, logramos balancear las clases de manera efectiva, generando ejemplos sintéticos de la clase minoritaria. Además, se verificó que no existen valores faltantes en el dataset, tanto antes como después del proceso de balanceo. El dataset transformado fue guardado en un archivo CSV, listo para ser utilizado en el entrenamiento de modelos.

De esta manera podremos tener más certeza de que cualquier modelo que entrenemos usando esta dataset tenga mejor calidad al predecir clases de manera equitativa.