Instituto Politécnico Nacional

Escuela Superior de Cómputo

Profesor:

Andres Garcia Floriano

Alumno:

Hernández Jiménez Erick Yael

Patiño Flores Samuel

Robert Garayzar Arturo

5BV1

Practica:
SMOTE y Perceptrón Simple

Índice

1. Introducción

1.1. SMOTE

El **SMOTE** (Synthetic Minority Over-sampling Technique) es una técnica de sobremuestreo ampliamente utilizada en el ámbito del aprendizaje automático para abordar el problema del desequilibrio de clases en conjuntos de datos de clasificación. Este desbalance ocurre cuando una de las clases está representada de manera considerablemente menor en comparación con las demás, lo que puede sesgar el desempeño de los modelos predictivos hacia la clase mayoritaria, resultando en un rendimiento deficiente al identificar correctamente instancias de la clase minoritaria.

La técnica SMOTE se diferencia de métodos más simples, como la replicación de instancias, al generar nuevos ejemplos sintéticos en lugar de duplicar observaciones existentes. Para ello, selecciona un punto de la clase minoritaria y, a partir de sus vecinos más cercanos, crea ejemplos sintéticos interpolando valores en el espacio de características. De esta manera, SMOTE expande de manera más robusta el espacio ocupado por la clase minoritaria, mejorando la representatividad de esta en el conjunto de datos y reduciendo el sesgo hacia la clase mayoritaria.

El uso de SMOTE es común en tareas de clasificación donde el equilibrio entre las clases resulta crítico para la precisión global del modelo, como en la detección de fraudes, el diagnóstico médico y otras áreas donde la identificación precisa de clases raras o menos representadas puede ser crucial.

1.2. Perceptrón Simple

El **Perceptrón simple** es uno de los modelos fundamentales en el campo del aprendizaje automático y representa una de las primeras formas de redes neuronales artificiales. Desarrollado por Frank Rosenblatt a finales de la década de 1950, el perceptrón simple constituye un clasificador binario que, mediante una combinación lineal de entradas ponderadas, busca establecer un límite de decisión para separar dos clases.

Este modelo se basa en un proceso de aprendizaje supervisado en el que ajusta sus pesos internos mediante un algoritmo de retroalimentación, generalmente basado en la regla delta. Cada entrada del perceptrón se multiplica por un peso, se suma un sesgo (o término de polarización) y el resultado se pasa a través de una función de activación, que usualmente es una función escalón. Si el resultado excede un umbral determinado, se clasifica en una clase; de lo contrario, en la otra.

El perceptrón simple es capaz de resolver únicamente problemas linealmente separables, lo que implica que los datos deben poder dividirse mediante una línea (o un hiperplano en dimensiones superiores). Esta limitación, señalada por Minsky y Papert en su libro Perceptrons (1969), motivó el desarrollo de arquitecturas de redes más complejas, capaces de manejar problemas no linealmente separables, como las redes neuronales multicapa.

A pesar de su simplicidad, el perceptrón simple sentó las bases para el desarrollo de modelos más avanzados de aprendizaje profundo y se reconoce como un paso crucial en la evolución de la inteligencia artificial y la teoría de redes neuronales.

2. Técnicas de separación de dataset

2.0.1. Validación Leave-One-Out (LOO)

La **validación Leave-One-Out (LOO)** es un caso particular de validación cruzada en el que el número de subconjuntos es igual al número de observaciones en el conjunto de datos. En cada iteración, se utiliza una única observación como conjunto de prueba y el resto de las observaciones como conjunto de entrenamiento. Esto se repite tantas veces como observaciones haya, lo que da lugar a evaluaciones altamente exhaustivas. Aunque ofrece una estimación imparcial del error de generalización, LOO puede ser computacionalmente costoso para conjuntos de datos grandes debido a la gran cantidad de particiones.

2.0.2. n-fold Cross-Validation

La **validación cruzada de n-fold** implica dividir el conjunto de datos en n subconjuntos (o 'folds') de aproximadamente el mismo tamaño. En cada iteración, uno de estos subconjuntos se utiliza como conjunto de prueba y el resto como conjunto de entrenamiento. Este proceso se repite n veces, con cada subconjunto utilizado como conjunto de prueba una vez. El rendimiento se evalúa promediando las métricas obtenidas en cada iteración. Este

método proporciona un equilibrio entre la cantidad de datos de entrenamiento disponibles y el tiempo computacional, y su versión más común es la *10-fold cross-validation*.

3. Comparativa

SMOTE mejora el equilibrio de clases, lo que a menudo resulta en mejores métricas de rendimiento para clasificadores; fácil de implementar. Puede generar datos sintéticos no representativos o causar sobreajuste en casos extremos.

El perceptrón simple es rápido y eficiente para problemas linealmente separables; buena base para entender modelos de clasificación más complejos. No funciona bien con datos no linealmente separables o con problemas complejos de múltiples clases sin modificaciones.

El 1NN tiene la capacidad para clasificar conjuntos de datos con límites de decisión complejos; flexible y no requiere suposiciones sobre la distribución de los datos. Sufre en presencia de ruido, puede ser computacionalmente costoso con grandes conjuntos de datos, y sufre más cuando hay desequilibrio de clases si no se combina con técnicas como SMOTE

4. Desarrollo

Los resultados obtenidos son los siguientes:

```
Hold-Out Evaluation:
Before SMOTE (Hold-Out): Counter({2: 66, 1: 51, 7: 21, 3: 14, 5: 12, 6: 7})
After SMOTE (Hold-Out): Counter({7: 66, 5: 66, 2: 66, 1: 66, 3: 66, 6: 66})
1-NN Accuracy Before SMOTE: 86.05%
Euclidean Classifier Accuracy Before SMOTE: 41.86%
1-NN Accuracy After SMOTE: 83.72%
Euclidean Classifier Accuracy After SMOTE: 34.88%

10-Fold Cross Validation Evaluation:
1-NN Accuracy Before SMOTE (10-Fold): 73.81%
Euclidean Classifier Accuracy Before SMOTE (10-Fold): 40.48%
1-NN Accuracy After SMOTE (10-Fold): 71.43%
Euclidean Classifier Accuracy After SMOTE (10-Fold): 40.48%
```

Figura 1: Resultados conjuntos obtenidos. Parte 1.

```
Accuracy: 100.00%
Input: [4.6, 3.2, 1.4, 0.2], Prediction: 0
Input: [5.2, 3.5, 1.5, 0.2], Prediction: 0
Input: [6.7, 3.1, 5.6, 2.4], Prediction: 1
Input: [7.7, 3.8, 6.7, 2.2], Prediction: 1
Input: [5.2, 4.1, 1.5, 0.1], Prediction: 0
Input: [4.9, 3.1, 1.5, 0.1], Prediction: 0
Input: [7.2, 3.2, 6.0, 1.8], Prediction: 1
Input: [4.5, 2.3, 1.3, 0.3], Prediction: 0
Input: [4.8, 3.1, 1.6, 0.2], Prediction: 0
Input: [6.9, 3.1, 5.1, 2.3], Prediction: 1
Input: [6.3, 2.5, 5.0, 1.9], Prediction: 1
Input: [5.1, 3.8, 1.9, 0.4], Prediction: 0
Input: [5.7, 4.4, 1.5, 0.4], Prediction: 0
Input: [6.4, 2.8, 5.6, 2.2], Prediction: 1
Input: [4.8, 3.0, 1.4, 0.1], Prediction: 0
Input: [6.4, 2.7, 5.3, 1.9], Prediction: 1
Input: [6.5, 3.0, 5.8, 2.2], Prediction: 1
Input: [6.3, 3.3, 6.0, 2.5], Prediction: 1
Input: [7.9, 3.8, 6.4, 2.0], Prediction: 1
Input: [6.7, 3.3, 5.7, 2.5], Prediction: 1
Input: [5.8, 2.8, 5.1, 2.4], Prediction: 1
Input: [5.8, 2.7, 5.1, 1.9], Prediction: 1
Input: [6.3, 2.9, 5.6, 1.8], Prediction: 1
Input: [5.3, 3.7, 1.5, 0.2], Prediction: 0
Input: [5.1, 3.7, 1.5, 0.4], Prediction: 0
Input: [6.9, 3.1, 5.4, 2.1], Prediction: 1
Input: [6.2, 3.4, 5.4, 2.3], Prediction: 1
Input: [6.1, 2.6, 5.6, 1.4], Prediction: 1
Input: [4.3, 3.0, 1.1, 0.1], Prediction: 0
Input: [7.1, 3.0, 5.9, 2.1], Prediction: 1
```

Figura 2: Resultados conjuntos. Parte 2.

5. Conclusiones

Hernández Jiménez Erick Yael:

Con estos resultados podemos ver que, aunque SMOTE es una técnica poderosa para el equilibrio de clases, los resultados sugieren que su aplicación puede no mejorar todos los tipos de clasificadores de manera uniforme. En este caso, la precisión del 1-NN disminuyó ligeramente después de SMOTE, mientras que el clasificador euclidiano mostró un empeoramiento en algunos escenarios. El 1-NN parece ser más robusto que el clasificador euclidiano en ambos escenarios de validación, pero su desempeño también se vio impactado por la generación de ejemplos sintéticos.

Patiño Flores Samuel: Al aplicar SMOTE a los datos de entrenamiento, pudimos observar que el desempeño del modelo en términos de precisión (accuracy) mejoraba, especialmente en lo que respecta a la predicción de la clase minoritaria. Es importante resaltar que este aumento en precisión no es siempre garantizado, y el impacto de SMOTE puede depender de factores como el tipo de modelo utilizado, la distribución de los datos y el grado de desbalance.

En nuestra implementación, el uso de Hold-Out para la evaluación nos permitió validar el rendimiento del modelo antes y después de aplicar SMOTE. Sin embargo, para una evaluación más robusta y generalizada, el uso de Cross-Validation es preferible, ya que mitiga el riesgo de que el modelo esté sobreajustado a una partición específica de los datos.

Robert Garayzar Arturo: Esta práctica está enfocada en la implementación manual de técnicas fundamentales en aprendizaje de máquina, como el manejo del desbalanceo de clases mediante SMOTE y la creación de un perceptrón simple para clasificación binaria. Asimismo, se busca evaluar el impacto de estas técnicas en el desempeño de clasificadores mediante validaciones comunes como Hold-Out y K-Fold Cross-Validation, sin el uso de bibliotecas externas, lo que permite profundizar en los conceptos y algoritmos esenciales.