Detector de Cáncer de Mama de Wisconsin

Chacón Estévez Ivon Elyana, Poma Quispe Gustavo Andres

Abstract

This study explores the classification of breast cancer tumors using machine learning with the Wisconsin Diagnostic Breast Cancer dataset. A neural network model was trained to predict tumor malignancy, and PCA was applied for dimensionality reduction. Results demonstrate that PCA reduces computational complexity without compromising model accuracy, emphasizing the importance of data preprocessing and feature reduction in medical diagnosis.

Introducción

El dataset de Cáncer de Mama de Wisconsin (WDBC) es un conjunto de datos ampliamente utilizado en investigaciones sobre la clasificación y diagnóstico del cáncer de mama. Este dataset ha sido clave en el desarrollo de modelos predictivos que ayudan a identificar si un tumor es maligno (M) o benigno (B), basándose en características extraídas de aspiraciones con aguja fina (AAF) de células mamarias. Estas características son medidas geométricas y estadísticas de los núcleos celulares, como el radio, textura, perímetro, área, suavidad, compacidad, entre otras, que describen cómo se ven las células del tumor. El objetivo principal de este artículo es construir y optimizar un modelo de clasificación utilizando redes neuronales artificiales para diferenciar entre tumores malignos y benignos, basándose en las 30 características calculadas de los núcleos celulares.

Este problema de clasificación binaria no solo es crucial para la detección temprana del cáncer de mama, sino que también tiene un gran potencial para asistir a los profesionales de la salud en el diagnóstico, mejorando así las tasas de supervivencia mediante intervenciones médicas tempranas y eficaces. En este contexto, el artículo realiza un análisis exploratorio y preprocesamiento exhaustivo del dataset, con el fin de garantizar la calidad de los datos antes de entrenar el modelo. Las etapas iniciales incluyen la limpieza del dataset, donde se identifican y eliminan columnas irrelevantes o con valores nulos, como la columna Unnamed: 32, y se clasifica el tipo de variables (continuas, discretas y categóricas). Además, se emplean técnicas de transformación de datos, como la codificación de etiquetas para la variable dependiente diagnosis y la normalización de las variables continuas para asegurar que todas las características están en la misma escala, un paso crítico para las redes neuronales.

Objetivo de Investigación

El principal objetivo de este dataset es clasificar tumores mamarios como malignos (M) o benignos (B) basándose en características extraídas de las aspiraciones con aguja fina (AAF) de células de una masa mamaria. El propósito es ayudar a desarrollar modelos predictivos que puedan asistir en la detección temprana del cáncer de mama y en la diferenciación entre tumores malignos y benignos. El diagnóstico temprano puede ser crucial para la intervención médica efectiva y mejorar las tasas de supervivencia.

El dataset contiene características de núcleos celulares extraídas de las imágenes digitalizadas de aspiraciones con aguja fina, las cuales son representadas mediante 10 características numéricas. Estas características

describen propiedades geométricas y estadísticas de los núcleos celulares. Las imágenes de los núcleos celulares de la masa mamaria se procesan para calcular estas características que pueden ser utilizadas para entrenar un modelo de clasificación.

Utilizar este dataset para entrenar un modelo de clasificación que pueda predecir el diagnóstico (benigno o maligno) a partir de las 30 características calculadas de los núcleos celulares. Este es un problema típico de clasificación binaria. La investigación busca utilizar la información extraída de las imágenes digitalizadas de los núcleos celulares en biopsias de cáncer de mama para entrenar algoritmos de clasificación que ayuden a predecir la naturaleza del tumor (benigno o maligno). Esto se logra a partir de características geométricas y estadística de las células, que son accesibles sin la necesidad de procedimientos invasivos.

Las columnas del dataset:

1. *Número de identificación:* Es como un código único para cada tumor. Sirve para identificar cada caso sin confusión.

2. *Diagnóstico:* Aquí se nos dice si el tumor es maligno (M) o benigno (B).

    1. M (maligno) significa que el tumor es peligroso y puede crecer y extenderse por el cuerpo.
    2. B (benigno) significa que el tumor es no peligroso y no se esparce a otras partes del cuerpo.
3. *Las características de las células del tumor (columnas 3 a 32):* Estas son medidas que describen cómo se ven las células del tumor. Los médicos no miran directamente las células, sino que las miden usando una máquina y los resultados de esas mediciones se convierten en números que podemos usar para hacer predicciones. Estas son las características:

    1. *Radio:* Es como medir la distancia desde el centro de la célula hasta los bordes (como si fueras a medir el radio de una rueda). Cuanto más grande sea el radio, más grande es la célula.

    2. *Textura:* Mide la rugosidad de la superficie de la célula. Si la célula es más suave o más áspera, puede indicar un tipo diferente de tumor.

    3. *Perímetro:* Es como medir cuánta longitud tiene el borde de la célula (como medir el borde de una pizza). Si la célula tiene un perímetro más largo, eso podría indicar algo importante sobre su forma.

    4. *Área*: Mide cuánta superficie ocupa la célula. Si el área es más grande, puede ser un indicio de un tumor más grande.

    5. *Suavidad:* Es como medir cuánto cambia la longitud de las líneas de la célula en diferentes partes. Si hay mucha variación, significa que la célula no es tan regular.

    6. *Compacidad:* Compara el perímetro de la célula con su área. Si el perímetro es mucho más grande que el área, podría ser un signo de que la célula

tiene una forma extraña.

7. *Concavidad:* Mide si el borde de la célula tiene huecos o curvas hacia adentro (como si el borde fuera dentado). Si hay más huecos, es más probable que el tumor sea maligno.

8. *Puntos cóncavos:* Cuenta cuántos huecos tiene el borde de la célula. Si tiene muchos, puede ser una señal de que el tumor es más peligroso.

9. *Simetría:* Mide cuán simétrica es la célula. Si la célula tiene una forma más simétrica, es más probable que sea un tumor benigno (no peligroso).

10. *Dimensión fractal:* Es una medida matemática que muestra lo compleja que es la forma del borde de la célula. Si la forma es más compleja (como una figura irregular o fractal), podría ser un signo de que el tumor es maligno.

Análisis Exploratorio y Preprocesamiento de Datos para la Clasificación con Redes Neuronales Artificiales

- **Carga y Exploración Inicial del Dataset**

  El dataset utilizado contiene información sobre el diagnóstico de cáncer de mama, incluyendo tanto variables continuas como categóricas. La exploración inicial reveló la presencia de columnas con valores nulos, las cuales fueron eliminadas para garantizar la integridad de los datos. Además, se observó que una de las columnas, 'Unnamed: 32', no aportaba información útil y fue eliminada.

  Por ejemplo, la verificación de valores nulos se realizó de la siguiente manera:

```
df.isnull().sum()
  # Eliminación de la columna irrelevante
  df = df.drop(columns=['Unnamed: 32'])
```

- **Clasificación de Variables**

  El siguiente paso fue clasificar las variables del dataset en continuas, discretas y categóricas. Este análisis permitió identificar que la mayoría de las variables eran continuas, mientras que la variable categórica principal era 'diagnosis', que indica si el tumor es benigno o maligno.

  Un ejemplo del proceso de clasificación es el siguiente:

```
def getColumnTypes(dataset):
        continuas = []
        discretas = []
        categoricas = []
        for col in dataset.columns:
        if dataset[col].dtype == 'object':
        categoricas.append(col)
        elif dataset[col].nunique() < 10:
        discretas.append(col)
        else:
        continuas.append(col)
        return continuas, discretas, categoricas
```

- **Detección y Tratamiento de Outliers**

Se emplearon gráficos como distribuciones y boxplots para identificar outliers en las variables continuas. Estos valores atípicos fueron tratados utilizando el rango intercuartílico (IQR), lo que permitió establecer límites inferiores y superiores para cada variable. Los datos que excedían estos límites fueron eliminados del análisis para mejorar la calidad del modelo.

Por ejemplo, la eliminación de outliers se realizó así:

```
def outl(dataset, col):
    IQR = dataset[col].quantile(0.75) - dataset[col].quantile(0.25)
    LI = dataset[col].quantile(0.25) - (IQR * 1.5)
    LS = dataset[col].quantile(0.75) + (IQR * 1.5)
    dataset[col] = dataset[col].clip(LI, LS)
```

- **Preparación de los Datos para el Modelado**

Con los datos limpios, se procedió a separarlos en variables independientes (características) y dependientes (etiquetas). La variable dependiente, 'diagnosis', fue codificada en valores numéricos mediante técnicas como el uso de `LabelEncoder`. Además, se normalizaron las variables continuas para garantizar que todas estuvieran en la misma escala, un paso crítico para mejorar el rendimiento del modelo de red neuronal.

Ejemplo del proceso de codificación y normalización:

```
from sklearn.preprocessing import LabelEncoder, StandardScaler
lb = LabelEncoder()
y = lb.fit_transform(df['diagnosis'])
scaler = StandardScaler()
x = scaler.fit_transform(df.drop('diagnosis', axis=1))
```

En este caso, nosotros solamente normalizamos un dato, que es *diagnosis*, ya que los demás datos son numéricos y el hecho de llegar a cambiarlos puede generar problemas al momento de llegar a entrenar la red neuronal.

- **Entrenamiento Inicial de la Red Neuronal**

Se implementó una red neuronal artificial con una estructura de dos capas ocultas, utilizando una función de activación logística y un límite de iteraciones de 20,000. Los datos fueron divididos en conjuntos de entrenamiento y prueba en una proporción de 80/20. El modelo fue evaluado mediante métricas como la precisión, lo que permitió establecer una línea base de rendimiento antes de aplicar técnicas avanzadas como el análisis de Componentes Principales (PCA).

- **Configuración básica del modelo:**

```
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=40)

redNeuronal = MLPClassifier(hidden_layer_sizes=(10, 4), max_iter=20000, activation='logistic', tol=1e-7)
redNeuronal.fit(x_train, y_train)
```

Análisis de Componentes Principales (PCA) y Optimización del Modelo

- **Aplicación de PCA**

  El PCA transforma el conjunto de datos original en un espacio de menor dimensión, definiendo nuevos ejes que maximizan la varianza de los datos. En esta implementación, se evalúan diferentes números de componentes principales para determinar el número óptimo que maximiza el rendimiento del modelo.

  Ejemplo de aplicación de PCA:

  ```python
  from sklearn.decomposition import PCA
  from sklearn.metrics import accuracy_score

  # Normalización previa
  scaler = StandardScaler()
  x_scaled = scaler.fit_transform(x)

  # PCA con diferentes componentes
  componentes_a_probar = [12, 10, 9, 5, 3]
  for n_componentes in componentes_a_probar:
      pca = PCA(n_components=n_componentes)
      x_pca = pca.fit_transform(x_scaled)

      # Dividir y entrenar
      x_train, x_test, y_train, y_test = train_test_split(x_pca, y, test_size=0.2, random_state=40)
      redNeuronal = MLPClassifier(hidden_layer_sizes=(10, 4), max_iter=20000, activation='logistic')
      redNeuronal.fit(x_train, y_train)

      # Evaluar
      y_pred = redNeuronal.predict(x_test)
      print(f'Precisión con {n_componentes} componentes: {accuracy_score(y_test, y_pred):.2%}')
  ```

- **Selección del Número Óptimo de Componentes**

  El número de componentes principales se selecciona en función de la varianza explicada acumulada. Esto asegura que las dimensiones reducidas conserven la mayor cantidad de información posible sin comprometer significativamente la precisión del modelo.

  Cálculo de la varianza explicada:

  ```python
  pca = PCA(n_components=x_scaled.shape[1])
  pca.fit(x_scaled)
  varianza_acumulada = np.cumsum(pca.explained_variance_ratio_)
  plt.plot(range(1, len(varianza_acumulada)+1), varianza_acumulada)
  plt.title('Varianza explicada acumulada')
  plt.xlabel('Número de componentes')
  plt.ylabel('Varianza acumulada')
  plt.show()
  ```

- **Evaluación del Modelo con PCA**

  Una vez seleccionado el número óptimo de componentes, se evalúa el modelo con métricas de rendimiento como la precisión, la matriz de confusión y gráficos de desempeño. Esto permite verificar que la reducción de dimensionalidad no afecta negativamente la capacidad del modelo para clasificar correctamente.

  Ejemplo de evaluación:

  ```python
  from sklearn.metrics import
  ```

```
confusion_matrix,
classification_report

# Evaluación de desempeño
conf_matrix = confusion_matrix(y_test, y_pred)
print('Matriz de confusión:')
print(conf_matrix)
print('Reporte de clasificación:')
print(classification_report(y_test, y_pred))
```

- **Impacto del PCA en el Rendimiento del Modelo**

  El análisis demuestra que el uso de PCA reduce significativamente el tiempo de cómputo al trabajar con dimensiones menores. Sin embargo, la selección adecuada del número de componentes es crucial para evitar pérdida de información importante que pueda afectar negativamente la precisión del modelo.

Conclusión

El análisis y preprocesamiento de datos son pasos fundamentales en la construcción de modelos predictivos efectivos, especialmente en aplicaciones de alta sensibilidad como el diagnóstico del cáncer de mama. A lo largo de este estudio, se realizó una exhaustiva limpieza y transformación del dataset de cáncer de mama de Wisconsin (WDBC), abordando problemas comunes como valores nulos y outliers. Las técnicas de codificación y normalización garantizaron que las variables estuvieran en la misma escala, optimizando así el rendimiento del modelo.

La implementación de un modelo de red neuronal artificial para la clasificación de tumores malignos y benignos mostró resultados prometedores en términos de precisión, proporcionando una línea base sólida antes de aplicar técnicas adicionales como la reducción de dimensionalidad mediante el análisis de componentes principales (PCA). Este paso resultó ser clave para mejorar la eficiencia computacional y reducir el tiempo de procesamiento sin sacrificar la precisión del modelo. La elección del número óptimo de componentes mediante PCA permitió conservar la mayor cantidad de información relevante, asegurando la robustez del modelo.

Bibliografía

1. Geron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly Media.
2. Morales, V. (2020). Análisis de Componentes Principales. En Machine Learning: Teoría y Aplicaciones.