



Tecnológico de Monterrey

Instituto Tecnológico y de Estudios Superiores de Monterrey

Campus Estado de México

Análisis y Reporte sobre el desempeño del modelo

TC3006C

Grupo: 101

Profesorado:

Jorge Uresti

Fecha de entrega:

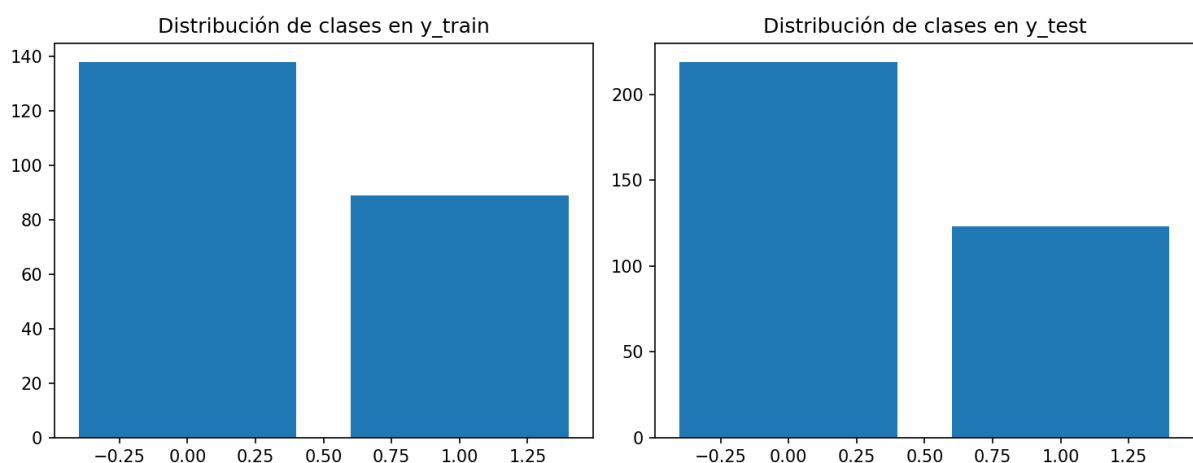
11 de septiembre 2023

Justificación del Dataset:

En la primera entrega, empleamos un conjunto de datos similar al conjunto Iris, conocido como Penguins, debido a la similitud en las medidas y valores entre ambos. Estos conjuntos de datos son especialmente adecuados para el algoritmo de vecinos más cercanos (KNN) debido a su baja dimensionalidad. Con un número reducido de columnas, datos numéricos y equilibrio en la distribución de las clases, el conjunto de datos Penguins es ampliamente recomendado en la comunidad de ciencia de datos, lo que lo convierte en una fuente confiable y de fácil clasificación.

Sin embargo, al utilizar las bibliotecas de sklearn, se presentaron complicaciones en el rendimiento del algoritmo KNN al intentar predecir las clases. En respuesta a esto, decidí buscar conjuntos de datos específicamente diseñados para KNN en Kaggle, lo que resultó en una mejora significativa en los resultados. Uno de los aspectos distintivos de este nuevo conjunto de datos es que, a diferencia de Penguins, que tiene 4-5 columnas funcionales, este nuevo conjunto de datos cuenta con 31 columnas. Esta mayor dimensionalidad permite que el modelo aprenda de manera más específica y logre predicciones de “diagnosis” (valor de "y") más precisas.

Distribución de las muestras:

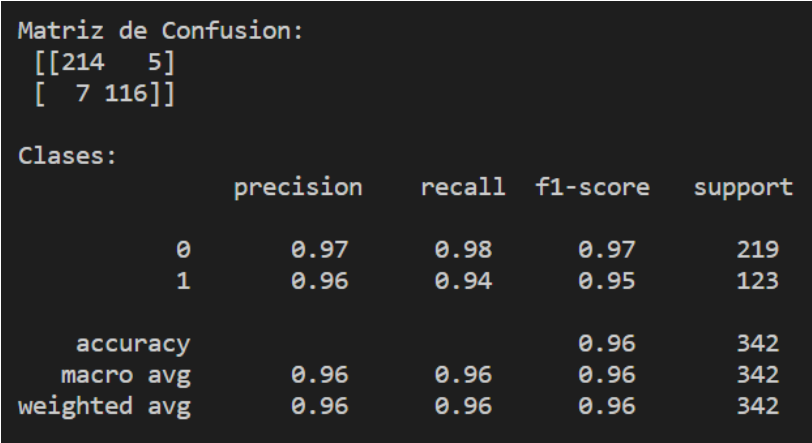


1.0 gráfica mostrando la distribución de las muestras.

Para garantizar una selección imparcial de los datos de prueba y entrenamiento, se empleó el parámetro `random_state`. Al fijar su valor en 42, se asegura que la división de datos se mantenga constante en cada ejecución del código, lo que facilita realizar ajustes visuales en el modelo. Además, se utilizó el parámetro `test_size` para asignar un porcentaje de los datos al conjunto de entrenamiento de acuerdo con el tamaño del conjunto de datos. Aunque comúnmente se utiliza un valor entre 0.2 y 0.3, en este caso se optó por un valor más alto, 0.6, ya que se observó que proporcionaba el mayor porcentaje de precisión (accuracy) en el modelo.

Se optó únicamente por utilizar `train` y `test`, excluyendo `validation`, ya que con las adaptaciones y las librerías, no se tomó en cuenta. Sin embargo, para una mejor optimización del modelo, se implementará en la parte final del reporte.

Bias o Sesgo:



```
Matriz de Confusion:
[[214   5]
 [  7 116]]

Clases:
```

| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0 | 0.97 | 0.98 | 0.97 | 219 |
| 1 | 0.96 | 0.94 | 0.95 | 123 |
| accuracy | | | 0.96 | 342 |
| macro avg | 0.96 | 0.96 | 0.96 | 342 |
| weighted avg | 0.96 | 0.96 | 0.96 | 342 |

1.0 Screenshot mostrando el comportamiento del modelo KNN

Al analizar la matriz de confusión, se evidencia que tanto los valores verdaderos como los falsos positivos son notablemente altos, lo que constituye un indicativo inicial de la precisión del modelo.

A continuación, al examinar la tabla de precisión para las clases benignas y malignas (1, 0), se observa que alcanzan valores de 0.97 y 0.96, respectivamente, lo que denota una precisión muy elevada. Esto se confirma con el valor de `recall`, que representa la proporción de aciertos del modelo y muestra cifras de 0.98 y 0.94 para las clases benigna y maligna, respectivamente. Además, el `f1-score`, que combina estas métricas, también refleja un

rendimiento positivo. No obstante, se nota una ligera dificultad en la predicción de la clase maligna (0).

Es importante destacar que la diferencia en el soporte (cantidad de muestras) entre las clases benignas (219 muestras) y malignas (123 muestras) es considerable, siendo el doble de ejemplos disponibles para aprender en el caso benigno.

En resumen, los resultados indican un bajo sesgo en el modelo, ya que los datos han sido previamente escalados, ajustados y limpiados para optimizar el procesamiento. Los indicadores de precisión, recall y f1-score respaldan la validez del modelo en general, a pesar de la ligera dificultad en la predicción de la clase maligna debido a la menor cantidad de ejemplos disponibles para aprender.

Grado de Varianza y Nivel de ajuste del modelo:

```
validación cruzada: [0.93478261 1.          0.95555556 0.93333333 0.93333333]  
Media validación cruzada: 0.9514009661835751  
Desviación estándar 0.025722267473760774
```

2.0 Screenshot mostrando validación cruzada, la media de la validación cruzada y la desviación estándar del modelo

Entre las diversas formas de determinar si el modelo está correctamente ajustado, se optó por la validación cruzada debido a su simplicidad y rapidez de implementación. En el gráfico se presentan cinco valores, que representan las diferentes particiones de los datos para evaluar el rendimiento del modelo. Es notable que no existe una variación significativa entre estos resultados, y destaca una muestra que logra una precisión del 100%. Esta mínima variación indica que el modelo está bien ajustado, ya que si hubiera valores extremadamente altos o bajos, sugeriría problemas de underfitting o overfitting. Luego, se calculó la media de estos cinco valores, obteniendo un resultado de 0.9514, lo cual es considerado bastante bueno, reforzando la idea de que el modelo está bien ajustado y no sufre de problemas de sesgo o varianza.

Para evaluar el grado de varianza del modelo, se calculó la desviación estándar de los resultados de la validación cruzada, que dio como resultado una variación de 0.0257. Esta cifra demuestra que el modelo exhibe una varianza muy baja y es coherente con los resultados previamente mencionados.

Uso de técnicas para mejorar el modelo:

```
PS C:\Users\sebas\Documents\GitHub\Modulo2_dataframe>  
/Documents/GitHub/Modulo2_dataframe/kk.py  
Mejor valor de K encontrado: 18  
Precisión con el mejor valor de K: 0.9766163742690058
```

3.0 Screenshot mostrando el resultado de K más óptimo y el accuracy

| | |
|-----------------|-------------|
| accuracy | 0.96 |
|-----------------|-------------|

4.0 Screenshot del accuracy cuando k es igual a 7 y no se ha implementado ninguna técnica

Para mejorar el modelo KNN, la elección del valor de K es un factor crítico que puede tener un impacto significativo en su rendimiento. Después de investigar, descubrí que se puede utilizar una función de la biblioteca scikit-learn llamada GridSearchCV. Esta función permite explorar una lista de valores de K, en este caso del 1 al 20, y ejecuta el modelo KNN en diferentes divisiones del conjunto de datos para encontrar el valor de K que produce el mejor rendimiento.

La función GridSearchCV realiza una búsqueda exhaustiva de hiperparámetros, y al final, selecciona el valor óptimo de K que maximiza el rendimiento. Una vez que se encuentra el valor óptimo, el modelo KNN se entrena nuevamente con ese valor de K ajustado.

Los resultados de esta optimización mostraron que el valor de K óptimo es 18, y al utilizar este valor en el modelo, se logró mejorar la precisión en un aumento de 0.013 en comparación con el valor de K anteriormente utilizado. Esto demuestra la importancia de una selección adecuada de K en el rendimiento del modelo KNN.