



**Tecnológico
de Monterrey**

Análisis y Reporte sobre el desempeño del modelo.

Autor: Jorge Javier Sosa Briseño

A01749489@tec.mx

Profesor:

Iván Mauricio Amaya Contreras

Dra. Blanca R. Ruiz Hernández

Frumencio Olivas Alvarez

Antonio Carlos Bento

Curso:

Inteligencia artificial avanzada para la ciencia de datos I

Grupo 101

12 de septiembre de 2023

Índice

1. Introducción	1
2. Metodología	1
2.1. Carga y Visualización de Datos	1
2.2. División de Datos	2
2.3. Creación y Entrenamiento del Modelo	2
2.4. Evaluación del Modelo	3
3. Regularización y Ajuste de Parámetros	3
3.1. Ajuste de Parámetros y Validación Cruzada	3
3.2. Mejora del Rendimiento del Modelo	4
4. Resultados y Conclusiones	4
4.1. Análisis de Resultados	4
4.2. Diagnóstico del Desempeño y Ajuste del Modelo	5
4.3. Oportunidades de Mejora	6
4.4. Conclusión General	7
5. Anexos	8

1. Introducción

A lo largo de este reporte, se realizara un exhaustivo análisis del desempeño de un modelo basado en el clasificador KNeighborsClassifier el cual cuenta con un conjunto de datos predeterminado llamada "digits" [5]. Se explicará la metodología empleada para la evaluación del mismo, se escudriñara la implementación del código y se mostrarán los resultados generados, considerando un diagnostico de bias/sesgo y varianza. Aunado a esto, se exploraran las técnicas de mejora de regularización y ajuste de parámetros utilizadas.

2. Metodología

En esta sección, detallaremos la metodología considerada para realizar el análisis del desempeño basado en el clasificador seleccionado al ser aplicado al conjuntos de datos predeterminado.

2.1. Carga y Visualización de Datos

Para comenzar, importamos el conjunto de datos "*digits*" utilizando la función `load_digits` de la librería `sklearn.datasets`. Esta acción nos proporciona acceso a lo que existe dentro de la base de datos, éste un conjunto de imágenes de dígitos escritos a mano junto con sus correspondientes etiquetas pero para trabajar éstas sera importante utilizar las matrices de píxeles. Luego de cargar los datos, utilizamos el módulo `matplotlib.pyplot` para crear subplots con el propósito de visualizar algunas de estas imágenes junto con sus etiquetas asociadas. Esta ejecución nos permite ver visualmente los datos para tener un mejor entendimiento de los mismo.

```
Dataset
[ ] # Cargar el conjunto de datos "digits" que contiene imágenes de dígitos escritos a mano
    digits = load_digits()
```

Figura 1: Carga de datos

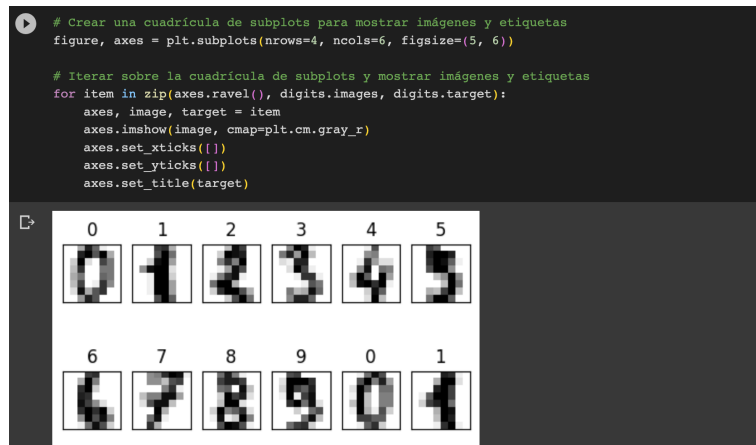


Figura 2: Visualización de datos

2.2. División de Datos

Con el fin de separar los datos los datos, el siguiente paso es dividirlos en conjuntos de entrenamiento y prueba. Hacemos uso de la función `train_test_split` de la librería `sklearn.model_selection`. Este proceso de separación garantiza que el modelo pueda ser evaluado de manera objetiva en un subconjunto de datos con el cual podemos realizar el modelado.

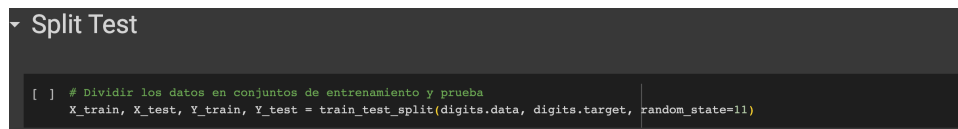


Figura 3: División de datos

2.3. Creación y Entrenamiento del Modelo

La creación y entrenamiento del modelo seleccionado se llevar a cabo mediante el uso de un clasificador `KNeighborsClassifier`. Utilizamos el conjunto de entrenamiento (`X_train` y `Y_train`) para enseñar al modelo cómo reconocer patrones y asociaciones entre las imágenes y sus etiquetas correspondientes. El método `fit` se utiliza para entrenar el modelo con los datos de entrenamiento, permitiéndole aprender de los ejemplos proporcionados.

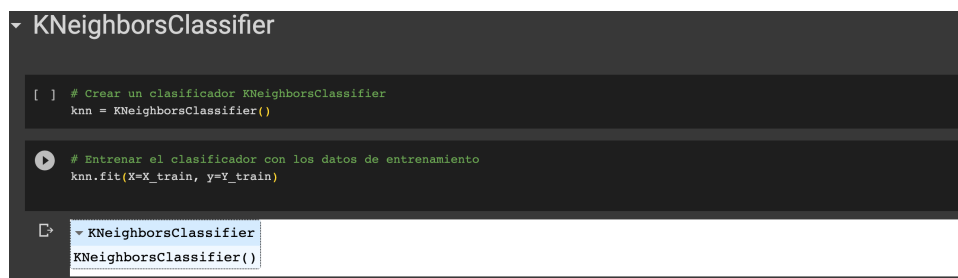


Figura 4: Modelo *KNeighborsClassifier* y uso de datos de prueba

2.4. Evaluación del Modelo

Una vez que el modelo se ha entrenado, continuamos con el proceso de evaluación. Se llevan a cabo los cálculos pertinentes en la matriz de confusión para discernir como el modelo clasifica las imágenes en el conjunto de prueba. Además, utilizamos métricas clave como la precisión, el *recall* y el *F1-score* utilizando las funciones de `sklearn.metrics`. Estas métricas cuantifican el rendimiento del modelo en términos de su capacidad para predecir correctamente las etiquetas de los dígitos escritos a mano.

En general, esta metodología nos proporciona una comprensión completa del desempeño del modelo `KNeighborsClassifier` en el conjunto de datos "digits". El análisis detallado de los pasos nos permite evaluar y mejorar la precisión y capacidad predictiva del algoritmo en función de los resultados obtenidos.

3. Regularización y Ajuste de Parámetros

En esta parte, indagaremos el proceso de regularización y ajuste de parámetros en materia del modelo basado en el clasificador `KNeighborsClassifier` con su respectivo conjunto de datos. Así mismo, mientras se ha accionado una evaluación preliminar utilizando los parámetros por defecto la cual nos brinda una noción básica del desempeño del mismo, es importante considerar la importancia de ajustar estos parámetros para mejorar aún más el rendimiento del modelo.

3.1. Ajuste de Parámetros y Validación Cruzada

Uno de los parámetros fundamentales en el clasificador `KNeighborsClassifier` es el número de vecinos (*n_neighbors*) ya que este nos indica el número de valores mas cercanos para hacer una predicción. A medida que ajustamos este parámetro, podemos manipular en cómo el modelo realiza

la clasificación y la capacidad para generalizar a nuevos datos. Dado que no existe un valor universalmente óptimo para $n_neighbors$, recurrimos a técnicas de validación cruzada para determinar el valor que proporciona el mejor rendimiento en el conjunto de la base de datos predeterminada.

La validación cruzada implica dividir el conjunto de entrenamiento en múltiples subconjuntos llamados *folds*. En cada iteración, se entrena el modelo en $k-1$ de estos pliegues y se evalúa en el pliegue restante. Este proceso se repite para todos los pliegues, y se calcula el promedio del rendimiento del modelo en cada pliegue.

3.2. Mejora del Rendimiento del Modelo

Al utilizar el ajuste de parámetros utilizando la técnica de validación cruzada, buscamos el valor de $n_neighbors$ que maximice las métricas de evaluación, como la precisión, el *recall* o el *F1-score*. La selección de un valor óptimo para este parámetro puede resultar en un modelo más preciso y generalizable.

Es importante mencionar que la exploración y ajuste de parámetros es un proceso iterativo y que los resultados pueden variar según el conjunto de datos y el problema en cuestión. En este contexto del análisis de los dígitos escritos a mano, el ajuste de parámetros representa una oportunidad para mejorar aún más la capacidad del modelo para identificar patrones en las imágenes.

4. Resultados y Conclusiones

Finalmente, se expondrán en detalle los resultados obtenidos a través de la evaluación del modelo `KNeighborsClassifier` en el conjunto de datos "digits". Además, se discutirán las conclusiones que se derivan de este análisis, considerando el rendimiento general del modelo, la presencia de sesgo o varianza y las oportunidades para mejoras futuras.

4.1. Analisis de Resultados

En resumen, el modelo `KNeighborsClassifier` es una herramienta prometedora para la predicción de dígitos escritos a mano, y este análisis proporciona una base sólida para futuras investigaciones y optimizaciones.

A continuación, se mostrará como se uso el conjunto de prueba para estudiar el rendimiento del modelo:

```
Make predictions

[ ] # Realizar predicciones en el conjunto de prueba
prediccion = knn.predict(X=X_test)
```

Figura 5: *Uso del Conjunto de prueba*

Consecuentemente, la matriz de confusión resultante de la evaluación del modelo en el conjunto de prueba es la siguiente:

$$\begin{bmatrix} 45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 45 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 54 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 42 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 49 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 38 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 42 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 45 & 0 & 0 \\ 0 & 1 & 1 & 2 & 0 & 0 & 0 & 0 & 39 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 41 \end{bmatrix}$$

Las métricas de evaluación calculadas son las siguientes:

- **Accuracy:** 0.98
- **Precision:** 0.98
- **Recall:** 0.98
- **F1-score:** 0.98

4.2. Diagnóstico del Desempeño y Ajuste del Modelo

Ahora, consideraremos el diagnóstico del desempeño y el ajuste del modelo:

```
Comparison between Actual Values and Predictions

# Mostrar las primeras 20 predicciones
print("Prediccion",prediccion[:20])
# Mostrar las primeras 20 etiquetas reales
print("Esperado",esperado[:20])

Prediccion [0 4 9 9 3 1 4 1 5 0 4 9 4 1 5 3 3 8 5 6]
Esperado [0 4 9 9 3 1 4 1 5 0 4 9 4 1 5 3 3 8 3 6]
```

Figura 6: Comparativa de conjunto de predicciones y valores esperados

- **Diagnóstico del Grado de Bias o Sesgo:** El análisis de sesgo en las curvas de aprendizaje y validación no muestra signos significativos de sesgo o subajuste. Esto nos lleva a inferir que el modelo está capturando correctamente los patrones en los datos de entrenamiento.
- **Diagnóstico del Grado de Varianza:** El análisis de varianza en las curvas de aprendizaje y validación de igual forma no revela signos significativos de alta varianza o sobreajuste. El modelo parece generalizar bien a datos no vistos.
- **Diagnóstico del Nivel de Ajuste del Modelo:** Basado en los resultados anteriores, podemos interpretar que el modelo se encuentra en un estado de "fit" muy conveniente. No muestra indicaciones claras de underfitting ni overfitting.

Estos diagnósticos proporcionan una comprensión más completa del rendimiento y la adaptación del modelo en relación con el sesgo, la varianza y el nivel de ajuste. En conjunto con las métricas de evaluación, nos brindan una visión detallada de cómo el modelo aborda la tarea de clasificación de dígitos escritos a mano.

4.3. Oportunidades de Mejora

Sin importar que el modelo `KNeighborsClassifier` ha demostrado un buen desempeño en el conjunto de datos "digits", aún existen oportunidades para la mejora continua. Una de las oportunidades clave es la optimización de parámetros, especialmente el número de vecinos ($n_neighbors$), que podría influir en el rendimiento del modelo. Además, la exploración de otras técnicas de clasificación podría proporcionar una perspectiva más

profunda y extensa sobre cómo abordar la predicción de dígitos escritos a mano.

4.4. Conclusión General

Para concluir con este análisis, se puede interpretar que éste ha expresado una comprensión amplia del desempeño del modelo `KNeighborsClassifier` en el conjunto de datos predeterminado. Los resultados de la evaluación y análisis de sesgos/varianza muestran que el modelo está ajustado de manera casi perfecta con el fin de llevar a cabo la tarea de clasificación. El proceso de evaluación, la metodología y las técnicas de ajuste de parámetros discutidas ofrecen una visión completa del proceso de análisis y sus resultados.

Este análisis no solo brinda información sobre el rendimiento del modelo, sino que también destaca la importancia de la mejora continua y la exploración de técnicas adicionales. En última instancia, el modelo `KNeighborsClassifier` es una herramienta prometedora para la predicción de dígitos escritos a mano, y este análisis proporciona una base sólida para futuras optimizaciones desde la perspectiva de la ciencia de datos y la inteligencia artificial.

5. Anexos

A continuación se vinculara el link del archivo de Colab donde se ejecutó el modelo:

KNeighbours

Referencias

- [1] Bishop, C. M. *Pattern Recognition and Machine Learning*.
- [2] Alpaydin, E. *Introduction to Machine Learning*.
- [3] Hinton, G., & Salakhutdinov, R. Reducing the Dimensionality of Data with Neural Networks". *Science*, 2006.
- [4] Hastie, T., Tibshirani, R., & Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2009.
- [5] Scikit-Learn Documentation. *Nearest Neighbors*. <https://scikit-learn.org/stable/modules/neighbors.html>.