Centro de Investigación en Matemáticas (CIMAT).
Unidad Monterrey

Ciencia de Datos Tarea 5

Diego Godinez Bravo

2 de junio de 2024

1. Problema 1

Considera de nuevo los datos MNIST que hemos usado en la **Tarea 3**.

a) Aplica métodos de clasificación basados en regresión logística, redes neuronales, SVM, CART, boosting, bagging y random forests. Haz una comparación de las métricas que obtuviste con todos los clasificadores que has usado en este conjunto de datos, especifica los parámetros que usaste en cada método e incluye gráficos informativos.

   Tu reporte debe contener una comparación cuantitativa y cualitativa sobre el desempeño de cada método y finalmente, una conclusión donde indiques cuál, o cuáles métodos preferirías para este conjunto de datos y por qué.

b) Obtén la medida de importancia de las variables según los métodos basados en CART, AdaBoost, Bagging y RF. Usa una visualización adecuada de estos pesos y discute tus resultados.

Análisis Comparativo de Métodos de Clasificación

Centro de Investigación en Matemáticas
Maestría en Cómputo Estadístico

Junio 2024

Aprendizaje Supervisado: Modelos de Clasificación

La clasificación de datos es un problema fundamental en el aprendizaje automático supervisado, donde el objetivo es asignar cada punto de datos a una o más categorías predeterminadas.

El análisis comparativo de métodos de clasificación es una tarea esencial en el área de Ciencia de Datos y Machine Learning (ML). Este análisis permite evaluar el desempeño de diferentes algoritmos y seleccionar el más adecuado para un conjunto de datos específico.

Evaluación de Modelos de Clasificación

A lo largo de la literatura, se han descrito diversas técnicas para abordar el problema de clasificación de manera eficaz. Este reporte presenta una evaluación comparativa de siete modelos de clasificación aplicados al conjunto de datos MNIST, que consiste en imágenes de dígitos escritos a mano. El conjunto de datos contiene $60,000$ instancias con 784 variables, las cuales representan los píxeles de cada imagen. Los modelos evaluados son: Regresión Logística, Perceptrón Multicapa (MLP), Máquina de Soporte Vectorial (SVM), Árboles de Decisión, Adaboost, Bagging y Random Forest.

Basándonos en la documentación oficial de Scikit-Learn, se establecieron los parámetros relevantes para cada uno de los modelos evaluados. Esta elección de parámetros se realizó para optimizar el rendimiento y asegurar la estabilidad y generalización de los modelos.

Regresión Logística

El modelo de Regresión Logística es un método eficiente y ampliamente utilizado para problemas de clasificación, especialmente cuando las clases son linealmente separables. Este modelo estima las probabilidades de pertenencia a una clase utilizando la función sigmoide, lo que permite transformar una combinación lineal de las características en una probabilidad que se sitúa entre 0 y 1.

Dentro de los parámetros relevantes considerados para ajustar el modelo se incluyen:

- 'solver = saga' - Optimizador adecuado para trabajar con grandes conjuntos de datos y que soporta la regularización de tipo *l2*.
- 'penalty = l2' - Regularización para evitar el sobreajuste, mantiene el equilibrio entre la complejidad del modelo y su capacidad de generalización.

Basándonos en la matriz de confusión, se observa que el número de instancias mal clasificadas es mínimo. El número máximo de dígitos mal clasificados entre pares de clases es de 10, como ocurre en el caso de las clases '9' y '4'. Esto es coherente, ya que la similitud entre estos números es alta (**Figura 1**).

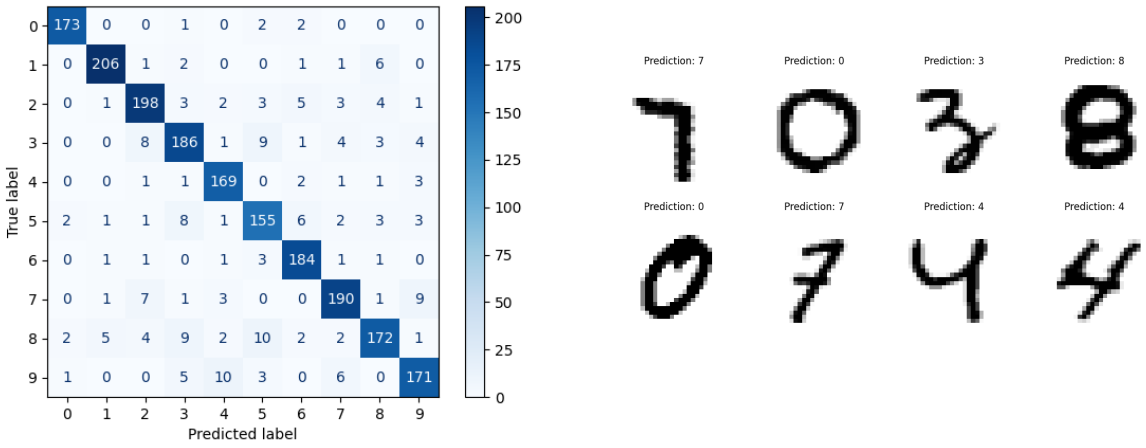


Figura 1. Rendimiento del modelo **Regresión Logística**. Matriz de confusión: predicciones de clasificación (*gráfico izquierdo*). Primeras 6 etiquetas con las predicciones correspondientes (*gráfico derecho*).

Redes Neuronales

El modelo Perceptrón Multiclase (MLP) es una clase de red neuronal que consta de múltiples capas de nodos, cada una conectada en su totalidad con la siguiente. A diferencia de las redes neuronales simples, el modelo MLP es capaz de aprender y modelar relaciones no lineales en los datos debido a la presencia de múltiples capas ocultas y funciones de activación no lineales

Entre los parámetros relevantes considerados para ajustar el modelo se encuentran:

- 'activation = tanh' - Función de activación hiperbólica para las capas ocultas utilizada por su capacidad para modelar relaciones no lineales.
- 'solver = adam' - El optimizador 'adam' se selecciona por su eficiencia en el entrenamiento de grandes conjuntos de datos.
- 'alpha = 0.32' - El término de regularización *alpha* se utiliza para controlar el equilibrio entre el sesgo y la varianza del mpodelo. A través de validación cruzada se encontró que un valor de 0,32 proporciona buenos resultados en términos de precisión y generalización en el conjunto de datos MNIST. Esto ayuda a reducir el riesgo de sobreajuste del modelo al mismo tiempo que se mantiene una buena capacidad predictiva del modelo.

De acuerdo con la matriz de confusión, al igual que en el caso del modelo de Regresión Logística, el modelo MLP muestra un número pequeño de instancias mal clasificadas. Sin embargo, algunas clases siguen presentando dificultades para distinguirse entre sí (**Figura 2**).

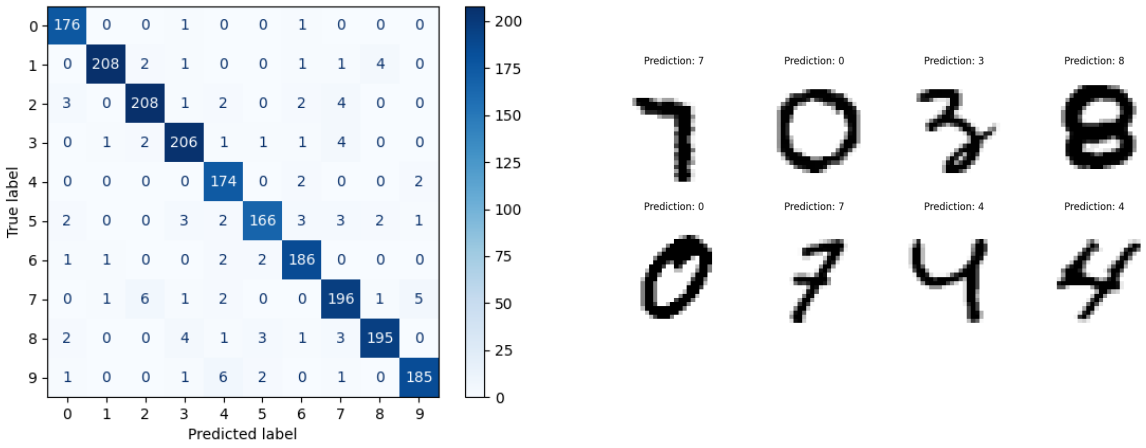


Figura 2. Rendimiento del modelo **MLP**. Matriz de confusión: predicciones de clasificación (*gráfico izquierdo*). Primeras 6 etiquetas con las predicciones correspondientes (*gráfico derecho*).

Máquina de Soporte Vectorial (SVM)

El objetivo principal del algoritmo SVM es encontrar un hiperplano óptimo (también conocido como superficie de decisión) para patrones que sean linealmente separables. Este hiperplano está determinado por puntos específicos denominados 'vectores de soporte', que representan los más difíciles de clasificar debido a su proximidad al hiperplano.

El problema de clasificación SVM multiclase es una extensión del algoritmo SVM estándar para problemas donde el número de clases es mayor a dos. Dado un conjunto de datos de entrenamiento, el objetivo es encontrar un hiperplano en un espacio de alta dimensión que separe de manera óptima los datos en una de las múltiples clases posibles.

Se detallan a continuación los parámetros relevantes considerados para ajustar el modelo:

- 'C = 1.0' - Controla el equilibrio entre la complejidad del clasificador y el número de muestras no separables. Un valor grande de C minimiza de manera significativa los errores de entrenamiento maximizando en medida de lo posible el margen entre las clases. Mientras que un valor bajo de C permite algunos errores de clasificación para encontrar un margen más amplio, lo que ayuda a mejorar la capacidad de generalización del modelo.

- 'kernel = rfb ' - Kernel radial para manejar datos no linealmente separables.

- 'decision_function_shape = ovo' - Diversos enfoques se han propuesto para la resolución de problemas de clasificación multiclase. Los enfoques más mencionados en la literatura son conocidos como 'one-against-all' y 'one-against-one'. En el enfoque 'one-against-all', se considera un clasificador SVM estándar para cada clase, donde cada uno se entrena para distinguir las muestras de su respectiva clase de las muestras de las clases restantes. Por otro lado, la estrategia 'one-against-one' implica la construcción de un clasificador binario para cada par de clases posible. Diversos estudios argumentan que el enfoque 'one-against-one' es más rápido y preciso, y ayuda a mitigar el desequilibrio de clases al considerar más modelos binarios SVM.

El número de instancias bien clasificadas aumentó en comparación con los modelos anteriores. En este caso, el número máximo de dígitos mal clasificados entre pares de clases sigue siendo de 6, sin embargo, la diagonal indica un aumento en la proporción de aciertos (**Figura 3**).

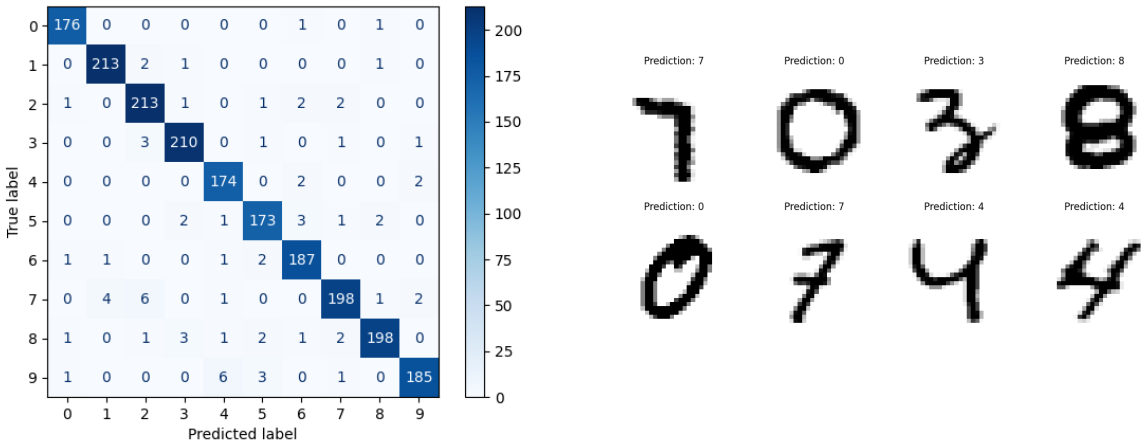


Figura 3. Rendimiento del modelo **SVM**. Matriz de confusión: predicciones de clasificación (*gráfico izquierdo*). Primeras 6 etiquetas con las predicciones correspondientes (*gráfico derecho*).

Árboles de Decisión

Métodos de aprendizaje supervisado no paramétricos utilizados para clasificación o regresión. Una de sus principales ventajas es su facilidad para ser interpretados y analizados. En términos de su arquitectura, a medida que el árbol se profundiza, las reglas de decisión se vuelven más complejas y el modelo se ajusta más.

La principal desventaja de estos métodos es la alta probabilidad de sobreajuste, es decir, el aumento excesivo de la complejidad del modelo. Para disminuir este problema, se establece la profundidad máxima del árbol.

Dentro de los parámetros relevantes considerados para ajustar el modelo se incluyen:

- 'criterion = entropy' - Se optó por el criterio basado en la entropía ya que, aunque es computacionalmente más exigente que otros criterios, ofrece una mayor precisión en términos de las métricas de evaluación.
- 'max_depth = 40' - Se asignó un valor de profundiad máxima del árbol para evitar el sobreajuste del modelo.

A diferencia de los modelos evaluados con anterioridad, el número máximo de instancias mal clasificadas entre pares de clases aumentó de manera significativa. Esto afecta claramente el rendimiento y desempeño del modelo (**Figura 4**).

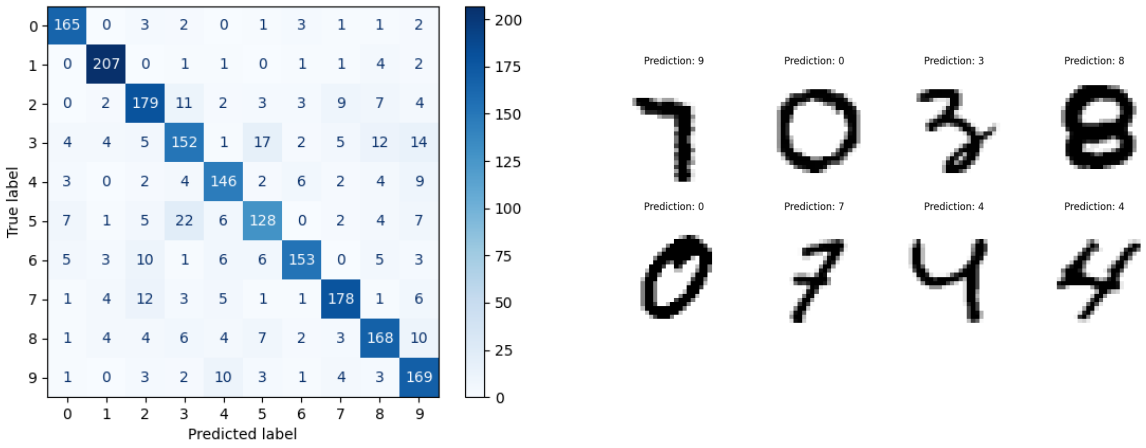


Figura 4. Rendimiento del modelo **CART**. Matriz de confusión: predicciones de clasificación (*gráfico izquierdo*). Primeras 6 etiquetas con las predicciones correspondientes (*gráfico derecho*).

Ensemble Learning Methods

Métodos de aprendizaje que combinan una serie de clasificadores débiles en un único modelo predictivo para aumentar el rendimiento. Estos métodos son especialmente efectivos en la mejora de la capacidad de generalización y la reducción del sobreajuste.

Adaboost

El método de Adaboost se basa en la combinación de múltiples clasificadores débiles para la construcción de un clasificador fuerte, mejorando así la capacidad de predicción del modelo.

Entre los parámetros relevantes considerados para ajustar el modelo se incluyen:

- 'algorithm = SAMME' - El algoritmo SAMME es robusto y flexible, permite ajustar los parámetros como el número de estimadores y la tasa de aprendizaje para optimizar el rendimiento del modelo. Este algoritmo extiende la capacidad del modelo para trabajar con problemas de clasificación multiclase.
- 'n_estimator = 100' - Utilizar 100 clasificadores proporciona un equilibrio adecuado entre la complejidad del modelo y su capacidad de generalización. Además, al entrenar un número razonable de estimadores permite optimizar el tiempo de cómputo y los recursos disponibles.

Al igual que el modelo CART, se observa un aumento significativo en el número de instancias mal clasificadas. Siendo 47 el número máximo de dígitos mal clasificados y por lo tanto, el modelo con el mayor número de errores entre un par de clases dado (**Figura 5**).

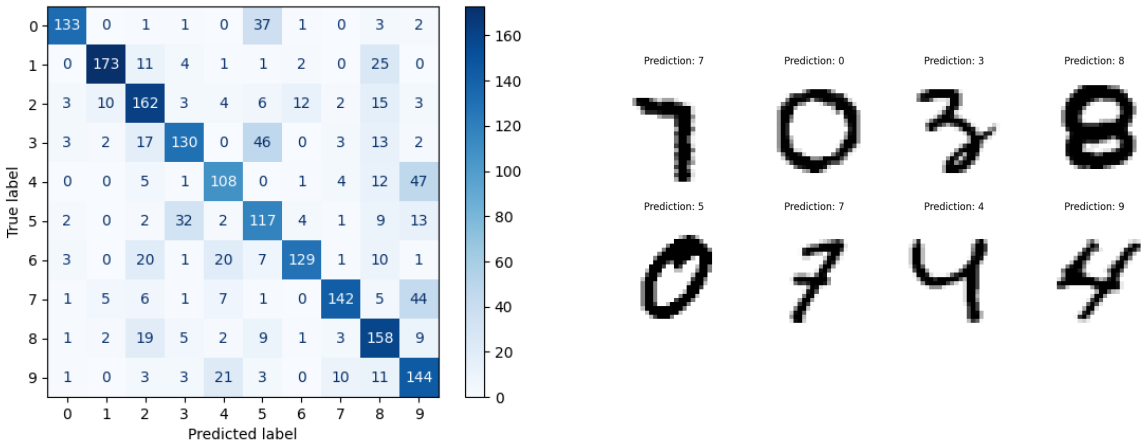


Figura 5. Rendimiento del modelo **Adaboost**. Matriz de confusión: predicciones de clasificación (*gráfico izquierdo*). Primeras 6 etiquetas con las predicciones correspondientes (*gráfico derecho*).

Bagging

La idea general del modelo es la generación de múltiples versiones del mismo modelo en diferentes subconjuntos de datos para posteriormente promediar sus predicciones. Este enfoque permite mejorar la precisión y la estabilidad del modelo al reducir el riesgo de sobreajuste a datos específicos. Cada clasificador débil se entrena en un subconjunto de datos generado aleatoriamente y, posteriormente, se combinan para formar un modelo robusto y generalizable.

Dentro de los parámetros clave considerados para ajustar el modelo se incluyen:

- 'estimator = SVC()' - Se optó por utilizar el algoritmo SVC (*Support Vector Classifier*) como estimador base debido a su robustez y eficiencia en la clasificación.
- 'n_estimator = 100' - Utilizar 100 clasificadores permite alcanzar un equilibrio adecuado entre la complejidad del modelo y su capacidad de generalización. Además, entrenar un número razonable de estimadores contribuye a optimizar el tiempo de cálculo y los recursos utilizados.

Al igual que el algoritmo SVM, el modelo de clasificación Bagging aumenta significativamente el número de predicciones correctas. A diferencia de Adaboost, que también pertenece a los métodos de ensamblaje, Bagging muestra resultados eficientes para este conjunto de datos en particular (**Figura 6**).

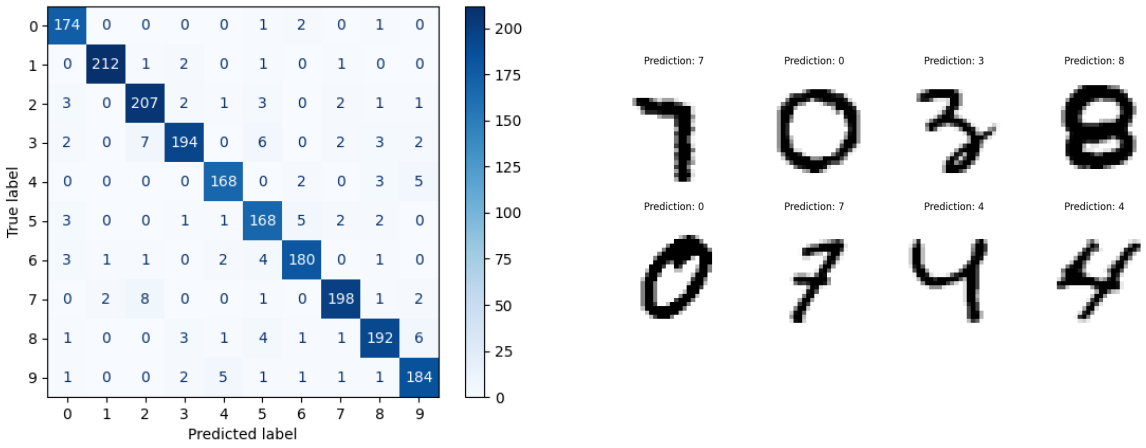


Figura 6. Rendimiento del modelo **Bagging**. Matriz de confusión: predicciones de clasificación (*gráfico izquierdo*). Primeras 6 etiquetas con las predicciones correspondientes (*gráfico derecho*).

Random Forest

Random Forest es un algoritmo de aprendizaje supervisado que se basa en el método de ensamble de árboles de decisión. La idea principal es construir múltiples árboles de decisión durante el entrenamiento y combinar sus resultados para obtener una predicción más robusta y precisa. Cada árbol se construye utilizando una muestra aleatoria de los datos de entrenamiento y una selección aleatoria de características, lo que ayuda a reducir el sobreajuste y mejorar la generalización del modelo.

Algunos de los parámetros clave considerados para ajustar el modelo incluyen:

- 'n_estimators = 150' - De acuerdo con Oshiro *et. al* (2012) no se observa una mejora significativa despúes de utilizar 128 árboles. Por lo que la elección del número de estimadores se basó en este hallazgo.
- 'criterion = gini' - En este caso se prefirió el índice de gini por su eficiencia en términos de cómputo. Lo que permitió hacer mejor uso del tiempo y de los recursos computacionales disponibles.

bullet 'max_depth = None' - A pesar de que los árboles profundos pueden causar sobreajuste, la agregación de múltiples árboles ayuda a mitigar este problema, por lo que se optó por no limitar la profundidad máxima de los árboles.

Entre los métodos de ensamble, se observó que tanto los modelos de clasificación Bagging como Random Forest mostraron resultados eficaces para este conjunto de datos en particular (**Figura 7**).

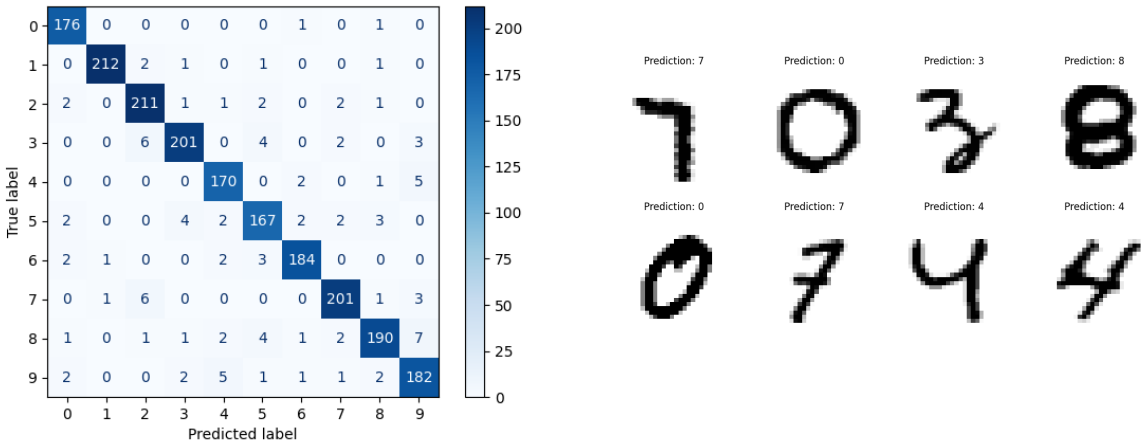


Figura 7. Rendimiento del modelo **Random Forest**. Matriz de confusión: predicciones de clasificación (*gráfico izquierdo*). Primeras 6 etiquetas con las predicciones correspondientes (*gráfico derecho*).

Rendimiento de los Modelos

Se construyeron representaciones gráficas del rendimiento de los modelos utilizando técnicas de validación cruzada para obtener los valores promedio de las métricas *accuracy* y *precision*.

Obsrvamos con claridad que los modelos de clasificación Adaboost y Árboles de Decisión muestran valores de *accuracy* y *precision* promedios relativamente bajos, en el rango del 60 % al 75 %. Estos valores sugieren que estos modelos pueden no ser los más adecuados para este conjunto de datos específico.

Por otro lado, los modelos SVM y Bagging muestran los valores más altos de *accuracy* y *precision* promedios, superando el 90 %. Esto indica que estos enfoques son altamente efectivos para la tarea de clasificación en el conjunto de datos MNIST, posiblemente debido a su capacidad para manejar eficazmente conjuntos de datos de alta dimensionalidad.

Los modelos restantes, como MLP, Regresión Logística y Random Forest, también muestran resultados aceptables, con valores de *accuracy* y *precision* promedios superiores al 90 %. Estos modelos demuestran ser competitivos en términos de rendimiento, por lo que pueden ser considerados como alternativas viables dependiendo de otros criterios de rendimiento y requisitos específicos del problema (**Figura 8**, **Figura 9**).

Adicional a esto, se calculó el tiempo promedio de entrenamiento de cada modelo. Esto proporciona información adicional sobre la eficiencia computacional de los diferentes enfoques de clasificación. Estos datos se presentan en la **Tabla 1**, junto con los valores promedio del *accuracy* obtenidos por cada modelo. Esta combinación de información permite una evaluación más completa y equilibrada de los modelos, considerando tanto su rendimiento predictivo como su eficiencia en el uso de los recursos computacionales.

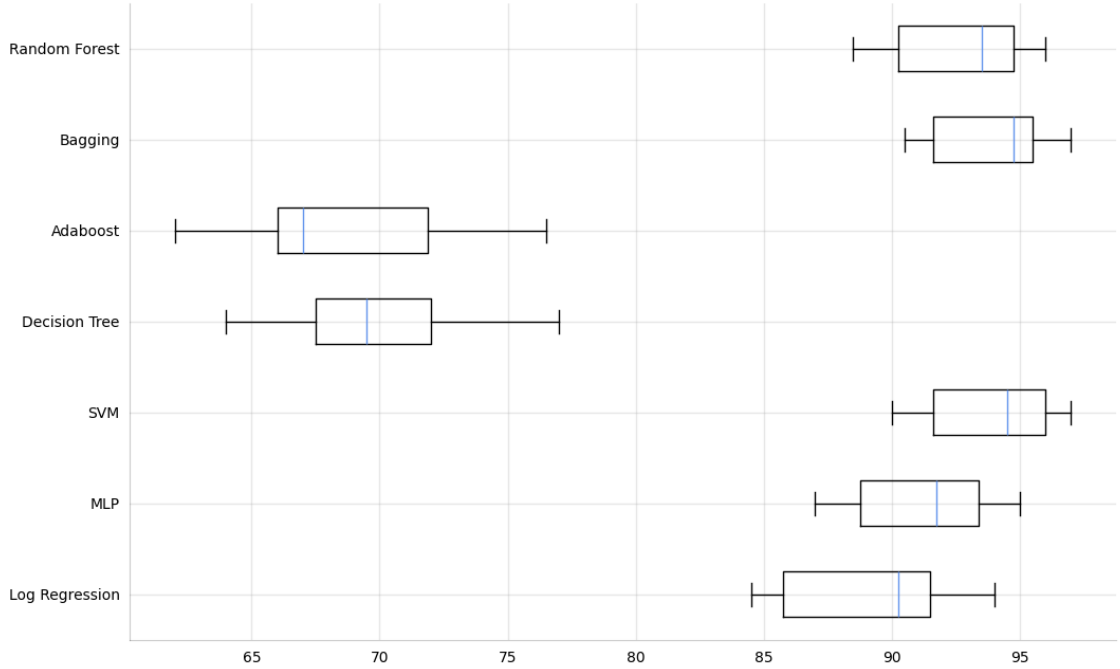


Figura 8. Rendimiento del Modelo de Referencia: *accuracy*.

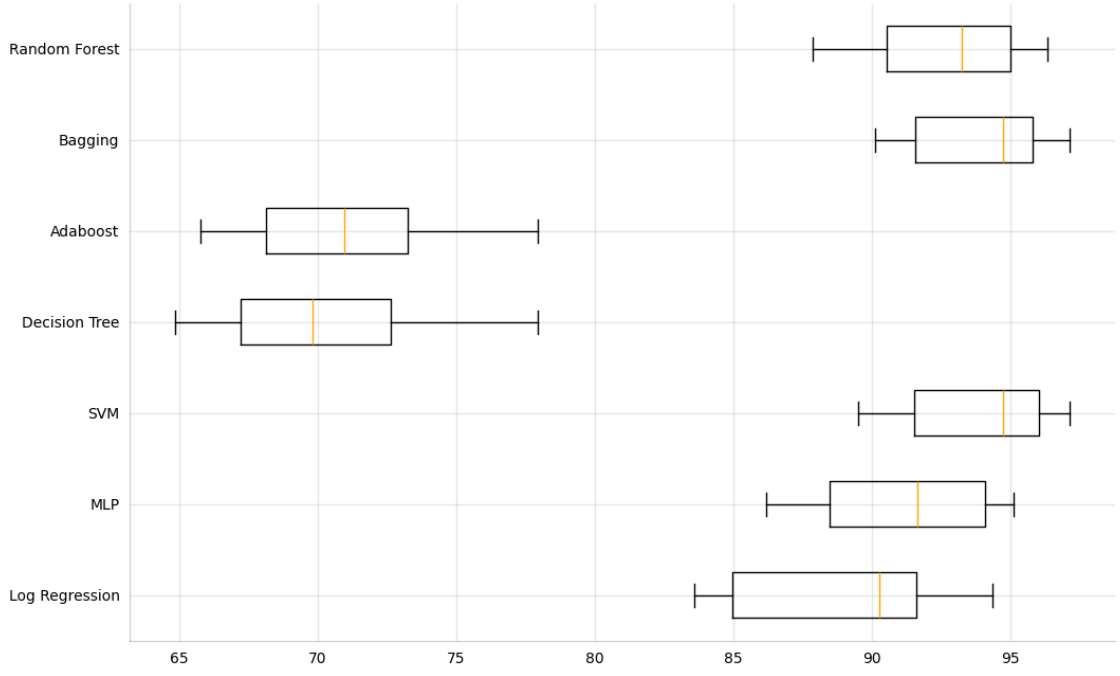


Figura 9. Rendimiento del Modelo de Referencia: *precision*.

La evaluación de los distintos modelos de clasificación en el conjunto de datos MNIST revela que el algoritmo **Máquina de Soporte Vectorial (SVM)** y **Bagging** son los mejores en términos de *accuracy* y *precision*, alcanzando el 94 %. Sin embargo, SVM tiene la ventaja adicional de alcanzar un tiempo de entrenamiento menor.

El modelo **Random Forest** también es un modelo adecuado para el conjunto de datos, con un *accuracy* promedio del 93 % y un tiempo de entrenamiento rápido (0.856 segundos).

Entre los modelos de clasificación que presentaron rendimiento bajo en términos del *accuracy* se encuentran el modelo **Árboles de decisión** y el modelo de ensamble **Adaboost**, los cuales alcanzan valores menores del 70 % (**Tabla 1**).

Tabla 1. Comparación del valor *accuracy* obtenido de la evaluación de 7 modelos de clasificación.

Model	**Accuracy**	**Training Time (s)**
Logistic Regression	0.89	32.095
Multilayer Perceptron	0.91	4.109
Support Vector Machine	0.94	0.374
Decision Tree	0.70	0.264
Adaboost	0.69	2.354
Bagging	0.94	13.039
Random Forest	0.93	0.856

En resumen, para el conjunto de datos MNIST, los modelos **SVM** y **Random Forest** son altamente recomendados, ya que ofrecen una alta precisión, accuracy y tiempos de entrenamiento razonables. Bagging y MLP también son opciones válidas, dependiendo de la prioridad entre precisión y tiempo de entrenamiento. En situaciones donde el tiempo de entrenamiento es crítico, **SVM parece ser el modelo más adecuado**.

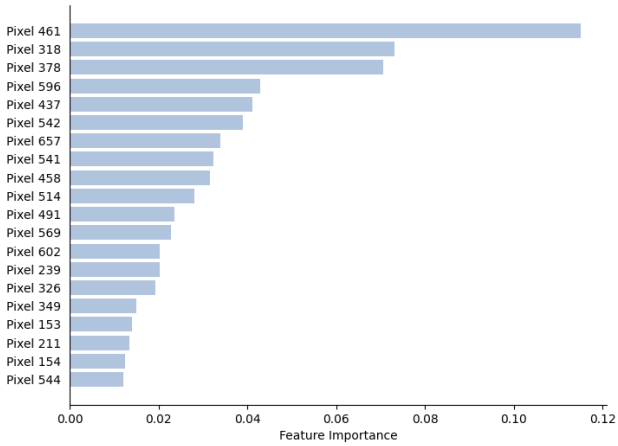
Importancia de las Características

La importancia de las variables para el conjunto de datos MNIST nos puede brindar un panorama general sobre qué características o en este caso, píxeles de las imágenes, son más relevantes para los algoritmos de clasificación a la hora de reconocer las imágenes.

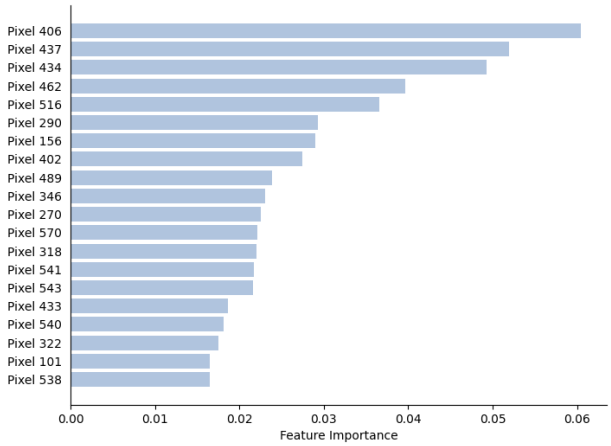
Para evaluar la importancia de las variables asignada por cada modelo de interés, se generaron múltiples visualizaciones que muestran la importancia relativa de cada característica. En este caso, al trabajar con un conjunto de datos con 784 variables, se filtró el gráfico para enfocarse únicamente en las primeras 20 variables con el mayor grado de importancia.

Los resultados revelan que cada método de clasificación otorga un peso específico a cada variable. Mientras que Random Forest tiende a distribuir la importancia de manera uniforme entre las variables, CART y Adaboost tienden a asignar mayor peso a píxeles específicos. Esto resulta en una diferencia significativa entre las variables en cuanto a la importancia relativa que cada una aporta (**Figura 10**).

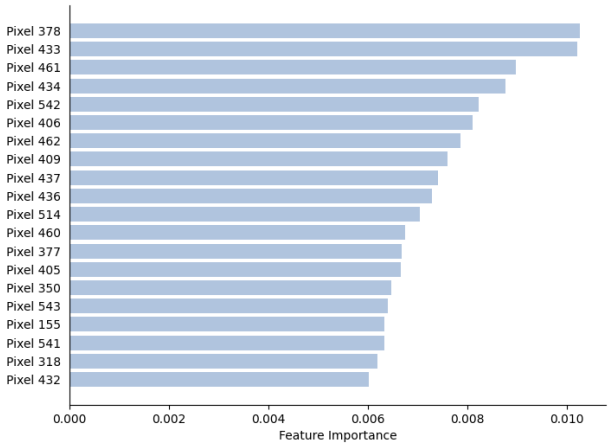
La metodología seguida por cada modelo al asignar los pesos a las variables puede influir en el rendimiento y el desempeño general del método de clasificación. En este sentido, Random Forest, al distribuir los pesos de manera uniforme, puede mejorar la precisión de las predicciones, mientras que CART y Adaboost pueden no capturar información relevante debido a cómo asignan los pesos a cada variable del conjunto de datos. Estos resultados son consistentes con lo presentado en la sección anterior (**Figura 8**, **Tabla 1**).



(a) CART.



(b) Adaboost.



(c) Random Forest.

Figura 10. Importancia de las variables de acuerdo a distintos método de clasificación.