Práctica 1

Adrián Cobo Merino y Arturo Lara Martínez

8 de diciembre de 2024

1. Introducción al problema

El proyecto tiene como objetivo clasificar tipos de cultivo (maíz, arroz, algodón, soja y trigo de invierno) a partir de datos hiperespectrales, representados por arrays de 131 longitudes de onda. Para abordar el problema, se emplearon varios modelos de machine learning y técnicas de transformación de datos. La solución busca equilibrar precisión, generalización y facilidad de implementación en un sistema final listo para producción.

2. Desarrollo del modelo

El enfoque de nuestro trabajo consistió en ir probando distintos modelos y ver cuáles obtenían mejores resultados. Los que mostraban un mejor rendimiento los juntábamos en un sistema de votación para ver si el rendimiento mejoraba. Esto se repitió en cada fase de la competición hasta que obtuvimos el mejor modelo que pudimos obtener. Como algunos de nuestros modelos se veían afectados mucho si usábamos todos los datos de entrada en crudo debido a la alta dimensionalidad de sus características, decidimos comenzar reduciéndolas usando ingeniería de características.

2.1. Ingeniería de características

Para mejorar la eficiencia al entrenar los modelos probamos:

- **Eliminación de características con baja varianza**: Se descartaron aquellas que aportaban poca información al modelo.
- **Estandarización (StandardScaler)**: Se normalizaron las características para mejorar el rendimiento de los modelos sensibles a escalas, como SVM y MLP. También se probó con "MinMaxScaler" y a pesar de que los resultados fueron también buenos, "StandardScaler", dio mejores resultados frente a "MinMaxScaler"
- **SMOTE (Synthetic Minority Over-sampling Technique)**: Se equilibraron las clases mediante la generación sintética de datos, mejorando la capacidad de los modelos para manejar datos desbalanceados. De esta forma se pudo entrenar la clasificación de clases con bajo número de ejemplos en el dataset.

2.2. Evaluación independiente de modelos

- **Random Forest**: Se destacó por su capacidad para manejar datos de alta dimensionalidad y correlaciones entre características. Se usaron en su entrenamiento final un total de 1000 estimadores.
- **Máquinas de Soporte Vectorial (SVM)**: Utilizando un kernel polinómico de grado 12, mostró buen desempeño en la separación de clases, especialmente tras aplicar técnicas de reducción de dimensionalidad de las características de los datos de entrada. Se decidió el uso de este clasificador dado que se le atribuye uno de los mejores modelos para problemas de clasificación en "machine learning" al ser capaz de realizar separaciones no lineales de los datos con "el truco del kernel".

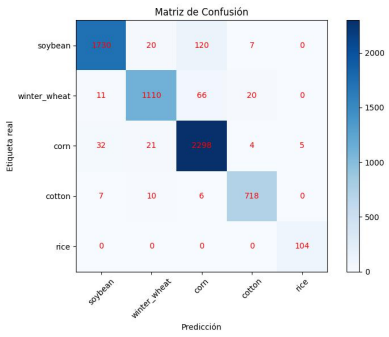
- **KNN con Análisis de Componentes Vecinales (NCA)**: Mejoró la eficacia al seleccionar las características más relevantes para cada vecino, reduciendo ruido y mejorando la clasificación. Se utilizó en su entrenamiento el uso de 3 vecinos ya que fueron los mejores resultados obtenidos para el tipo de datos tabulados de los que se disponen en este problema. Este modelo es especialmente útil cuando se cuenta con pocos datos como era nuestro caso ya que el modelo son los propios datos en sí.
- **Red Neuronal (MLPClassifier)**: Proporcionó buenos resultados al clasificar clases mayoritarias, con una arquitectura de 30-40 neuronas por capa y función de activación tangente hiperbólica.

2.3. Combinación mediante votación

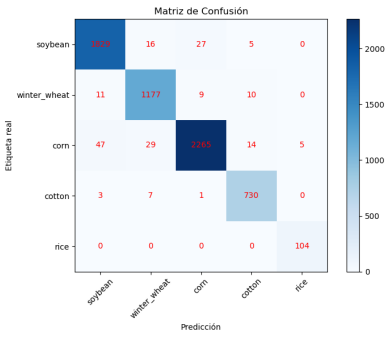
Los modelos con mejor rendimiento se integraron en un votador suave (soft voting), permitiendo combinar predicciones basadas en la confianza de cada modelo en cada predicción realizada. Esta técnica no solo mejoró la precisión global, sino que también incrementó la estabilidad frente a las clases menos representadas.

Los modelos se analizaron y obtuvieron métricas por separado, buscando los mejores parámetros. Se obtuvieron accuracies de entre 0.84(RF) y 0.93(MLP) por separado. Tras combinar los modelos con un sistema de votación "soft"se mejoró alrededor de un 0.02 sobre el mayor score obtenido de los modelos de forma individual, llegando a un score global del sistema de clasificación del 0.95. Esto sumado a un ajuste del tamaño del conjunto de entrenamiento y test mejoró considerablemente la clasificación final y los resultados obtenidos.

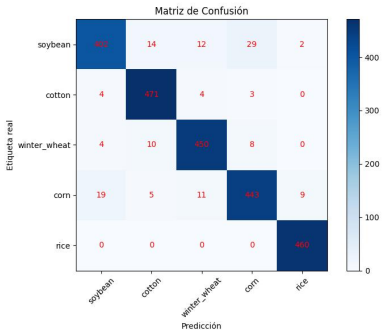
Las matrices de confusión obtenidas durante el desarrollo del proyecto fueron las siguientes:



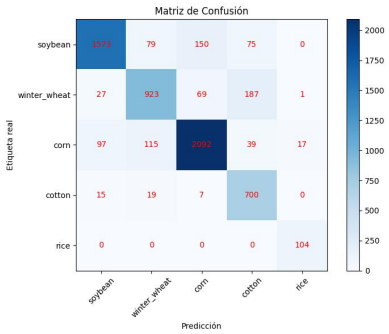
(a) Matriz de confusión SVM



(b) Matriz de confusión Random Forest



(c) Matriz de confusión KNN



(d) Matriz de confusión MLP

Figura 1: Matrices de confusión de cada modelo

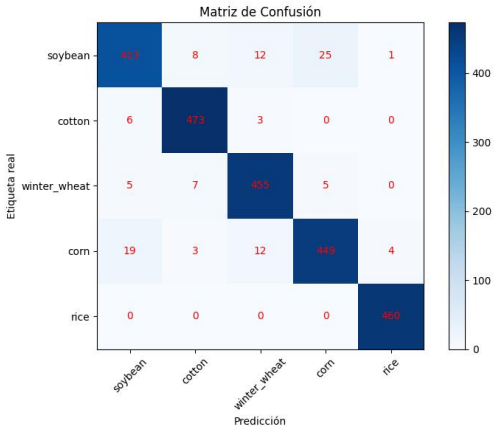


Figura 2: Matriz de confusión usando votación

Aclaración: cada modelo fue entrenado con el 80 % de los datos para entrenamiento y el 20 % restante para validación

3. Conclusiones

El sistema final integra lo mejor de cada modelo individual en un esquema de votación, obteniendo un balance óptimo entre precisión y robustez. La ingeniería de características resultó esencial para mitigar el impacto de la alta dimensionalidad y el desbalance en las clases, haciendo posible que los modelos trabajaran de manera eficiente y precisa.

Las decisiones en el diseño del modelo, así como del preprocesado de características, permitieron reducir los tiempos de entrenamiento a menos de 15 minutos utilizando una GPU Nvidia RTX 4050.