



**Título:** CLASIFICACIÓN SUPERVISADA

Máster Universitario en Informática Industrial y Robótica

Aprendizaje Automático II

**Alumnos:**

Domingo Alberto Capelo Luces

Iraisy Carolina Figueroa Silva

Fecha: 19 abril del 2021

Tabla de contenido

[Descripción de los datos empleados 3](#_Toc72173494)

[Tratamiento entradas y salidas 3](#_Toc72173495)

[Estrategia de partición de datos 7](#_Toc72173496)

[Métricas de evaluación 8](#_Toc72173497)

[Experimentos realizados 8](#_Toc72173498)

[Random Forest 8](#_Toc72173499)

[K-nearest neighbour 8](#_Toc72173500)

[Decision Tree Classifier 9](#_Toc72173501)

[Redes Neuronales Convolucionales 9](#_Toc72173502)

[Análisis de resultados 10](#_Toc72173503)

[Reducción de la dimensionalidad y el clustering 10](#_Toc72173504)

[Random Forest 11](#_Toc72173505)

[K-nearest neighbour 12](#_Toc72173506)

[Decision Tree Classifier - Árboles de decisión 13](#_Toc72173507)

[Redes Neuronales Convolucionales 14](#_Toc72173508)

[Tiempos de ejecución de los modelos entrenados con un único proceso o con multiproceso. 16](#_Toc72173509)

[Conclusiones 16](#_Toc72173510)

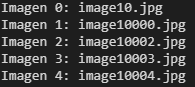
# Descripción de los datos empleados

El punto de partida consta de un banco de 5983 imágenes y un archivo Data.csv.

En lo referente a las imágenes se cumple que:

* Nombre: image**#N**.jpg-🡪 **#N**: número de imagen.
* Imagen a color RGB.
* Dimensiones distintas.

Ejemplo:



En lo referente al archivo Data.csv, se clasifican las imágenes en 4 grupos:

'Food', 'Attire', 'Decorationandsignage', 'misc'.

Ejemplo:

|  |  |
| --- | --- |
| **Image** | **Class** |
| image7042.jpg | Food |
| image3327.jpg | Misc |
| image10335.jpg | Attire |
| image8019.jpg | Food |
| image2128.jpg | Attire |
| image1106.jpg | Misc |
| image6750.jpg | Food |

Partiendo de estos datos y con el uso del lenguaje de programación Python, se realizará el entrenamiento y posterior comparación de modelos de aprendizaje automático aplicados a la clasificación de imágenes.

El primer paso, antes de entrenar cualquier modelo, es preparar las variables que vamos a introducir como entrada.

## Tratamiento entradas y salidas

Objetivo:

**x:** entrada del modelo, matriz de bits de una imagen.

**t:** salida del modelo, clasificación entrada.

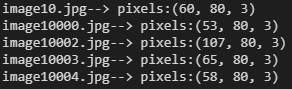
x 🡪 🡪t

MODELO

PASO 1: Redimensión imagenes

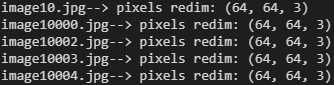
Las imágenes no tienen la misma dimensión, por lo que se redimensionan para que todas tengan la misma: 64x64 Pixeles.

Imágenes originales:





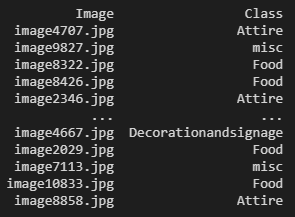
Imágenes redimensionadas:



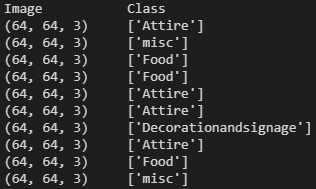
PASO 2 : Asociacion clase imagen

La clasificación de la imagen que se representa en el archivo Data.csv se asocia el nombre de una imagen con una clase. En nuestro caso nos interesa asociar la matriz de pixeles de cada imagen con su clase.

imagen.jpg 🡪clase

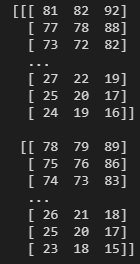


Matriz píxeles 🡪clase



PASO 3 : Aplanamiento matriz

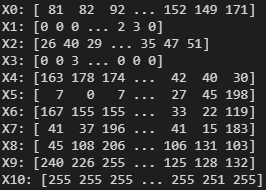
Las matrices de pixeles no se pueden introducir como entradas a los modelos directamente, hay que realizar un proceso de aplanamiento de la matriz de forma que se representen todos los pixeles de la imagen en un único vector.

(64,64,3) (1, 12288)

🡪 

De esta forma para las 5983 imágenes tendremos todos los vectores de entrada de nuestro modelo.

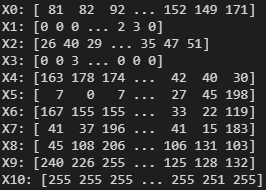
En el siguiente ejemplo se muestran las primeras 11 entradas:



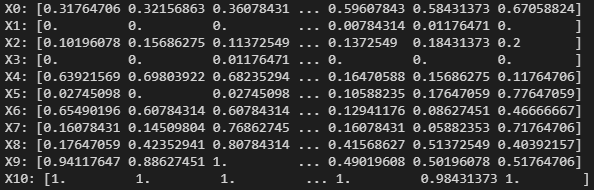
PASO 4 : Normalización entradas

Para miminizar el tiempo de entrenamiento y debido a que los datos tienen una variación de 0 a 255 ya que son de 8 bits, se opta por normalizar las entradas de forma que varien entre 0 y 1 y de esta forma lograr que el resultado del entrenamiento sea mas preciso.

Sin normalizar



Normalizado



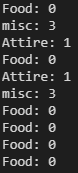
PASO 5: Normalización salidas

Para facilitar el entrenamiento de los modelos, las clases que tienen un formato del tipo “string” se asocian con un número.

'Food': 0, 'Attire': 1, 'Decorationandsignage':2, 'misc':3

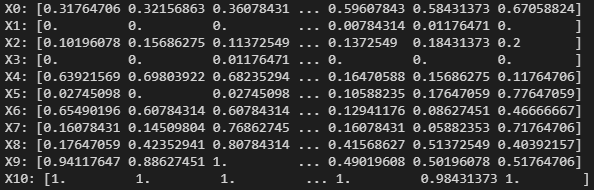
Ejemplo:





Finalmente, ya podemos realizar los entrenamientos de los modelos con el uso de nuestras entradas salidas previamente tratadas:

X normalizado t numérica

MODELO

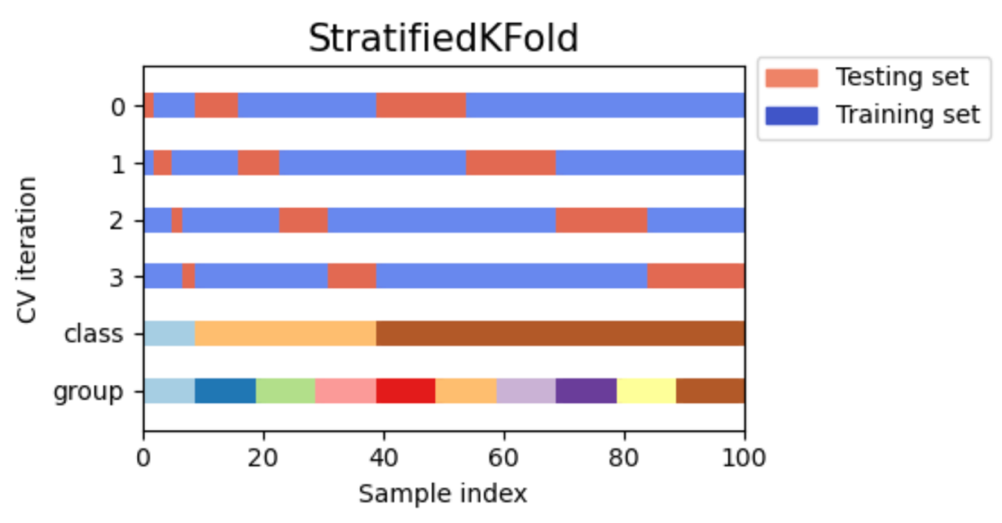
En el siguiente estudio se analizaran los resultados de distintos modelos de aprendizaje automático supervisado aplicados a la clasificación de imágenes.

Cabe destacar que el aprendizaje es supervisado porque además de las entradas del modelo se indican las salidas del mismo.

# Estrategia de partición de datos

En el enfoque básico, *k* -fold, el conjunto de entrenamiento se divide en *k* conjuntos más pequeños

K - Fold Estratificado es una variación de k-fold que devuelve particiones estratificadas, donde cada conjunto contiene aproximadamente el mismo porcentaje de muestras de cada clase objetivo que el conjunto completo. Este ha sido el método seleccionado por tener un conjunto de imágenes desbalanceado por cada clase



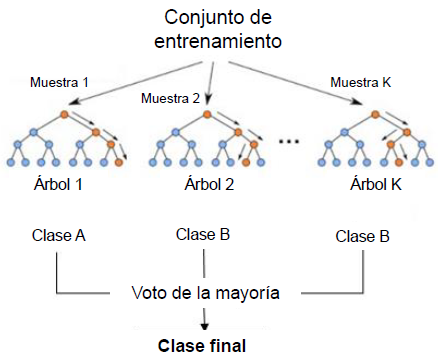
# Métricas de evaluación

* **Exactitud (accuracy):** tasa de acierto global del sistema.
* **Roc Auc (roc\_auc\_ovo):** El AUC calcula el área bajo la curva ROC y, por lo tanto, su valor varía entre 0 (peor caso) y 1 (mejor caso). El AUC indica cómo de bien se separan las probabilidades de las clases positivas de las negativas.
* **F1 macro:** es la media armónica entre la precisión y la sensibilidad, tiene la ventaja que su métrica es más relevante si las clases son desbalanceadas.
* **Presición (precision\_macro):** es la capacidad del clasificador de no etiquetar como positiva una muestra negativa. De todos los ejemplos que se clasificaron como positivos, ¿cuántos son realmente positivos?
* **Recall\_macro:** de todos los ejemplos positivos, ¿cuántos se pronostican como positivos?

# Experimentos realizados

## Random Forest

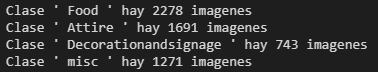
Formado por un conjunto de árboles de decisión individuales, cada árbol del conjunto se construye a partir de una muestra extraída con reemplazo del conjunto de entrenamiento. Los principales parámetros a ajustar cuando se emplea este método es n\_estimators concerniente al número de árboles. Cuanto más grande sea este valor mejor serán los resultados del estudio, influyendo también en más tiempo de calculo, sin embárgo Los resultados dejan de mejorar significativamente más allá de un número crítico de árboles.

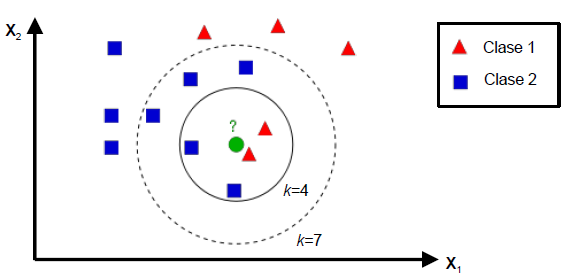


## K-nearest neighbour

Clasifica los datos tomando como referencia la clase mayoritaria de entre los k vecinos más cercanos de entre todos los datos de entrenamiento.

En nuestro caso, en base a las siguientes imágenes, las clasificara en base a la clase que tengan las imágenes más cercanas a ella.





Los métodos de este tipo emplean una métrica (medida disimilitud o distancia) entre los distintos datos.

Nuestro conjunto de datos de entrenamiento consta de 5983 imágenes por lo que según la teoría el valor de k tiene que ser menor a la raíz cuadrada de nuestro número de datos

Como estrategia de evaluación del modelo al igual que en el modelo de regresión logística se opta por realizar validación cruzada evaluando los parámetros comentados anteriormente.

## Decision Tree Classifier

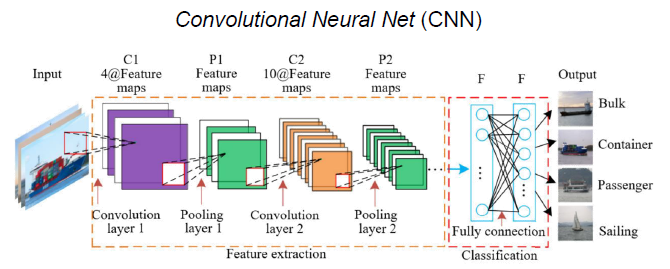
Es un método de aprendizaje supervisado no paramétrico que se utiliza para clasificación y regresión. El objetivo es crear un modelo que prediga el valor de una variable objetivo mediante el aprendizaje de reglas de decisión simples inferidas de las características de los datos. Una de las ventajas de los árboles de decisión es que son capaces de manejar problemas de múltiples salidas.

Los arboles de decisión toman como entrada dos matrices: una matriz X que contiene las muestras de entrenamiento, y una matriz Y de valores enteros, que contiene las etiquetas de clase para esas mismas muestras.

## Redes Neuronales Convolucionales

Modelo de red profunda que es capaz de capturar con éxito las dependencias espaciales y temporales en una imagen mediante la aplicación de filtros relevantes.

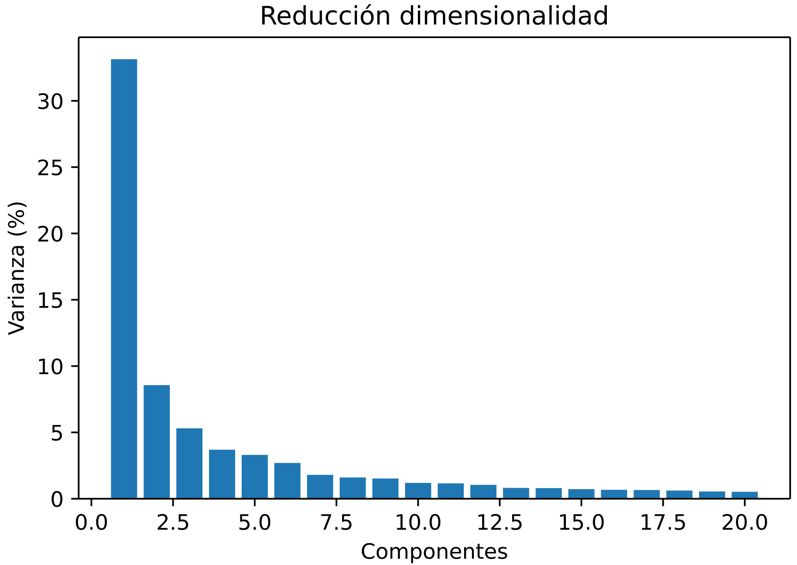
La arquitectura se adapta mejor a los datos de imágenes debido a la reducción en el número de parámetros involucrados y la reutilización de pesos.



# Análisis de resultados

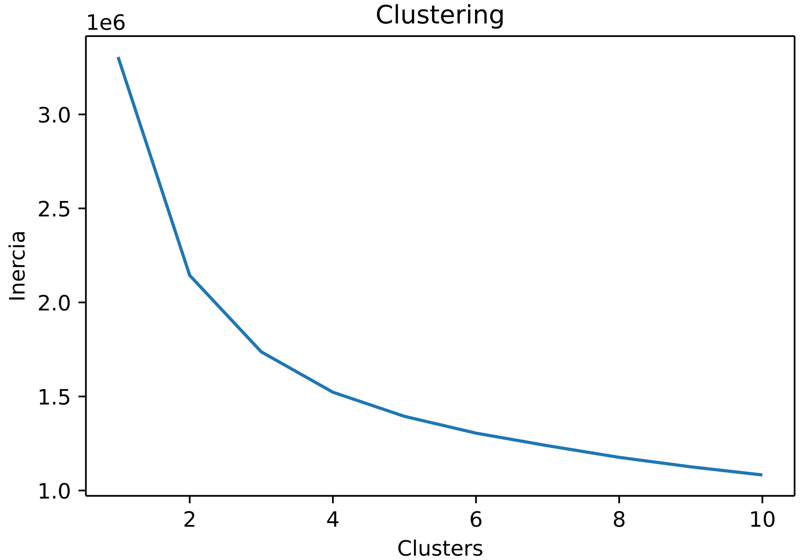
## Reducción de la dimensionalidad

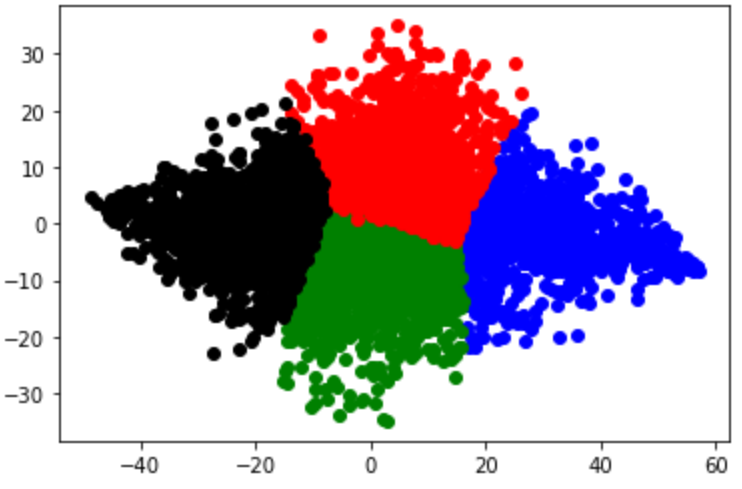
Se ha empleado el algoritmo Análisis de Componentes Principales (PCA) usando 20 componentes, con lo que se ha podido realizar la eliminación de los componentes menos relevantes, fijando la reducción en hasta 6 componentes.



## Clustering

Se agruparon los resultados en cluster y se ha comprobado la inercia fijando un resultado aceptable con 4 cluster, coincidiendo con el numero de clases estudiadas

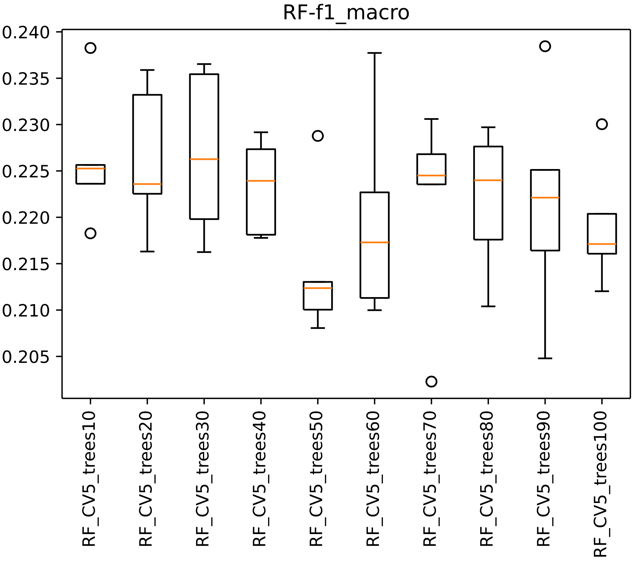
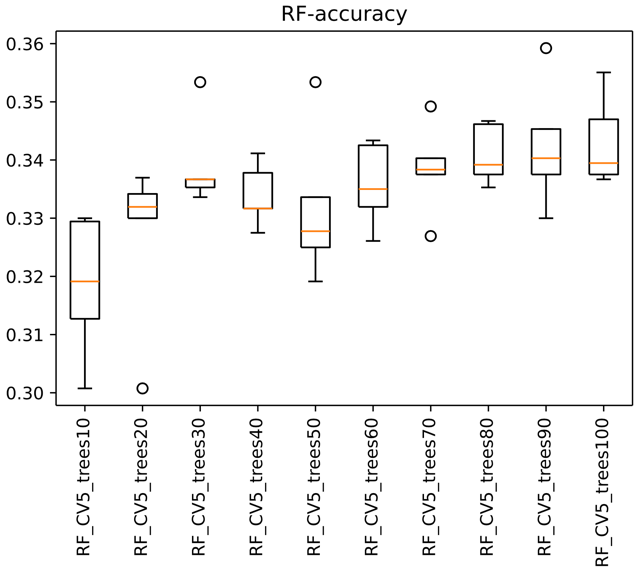


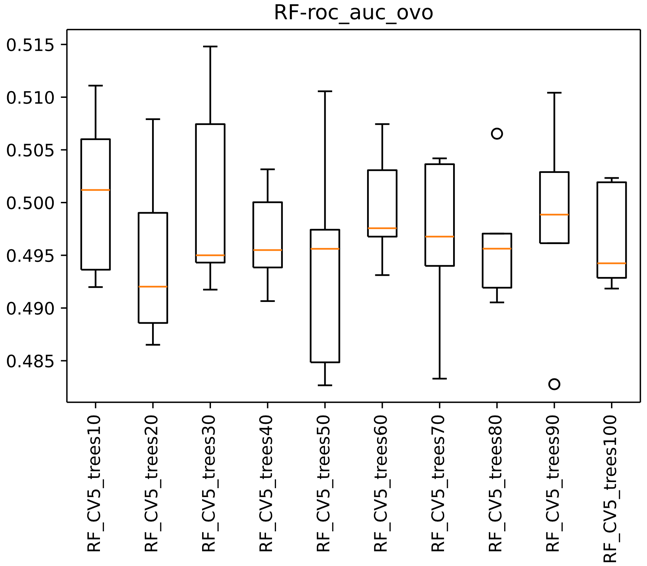
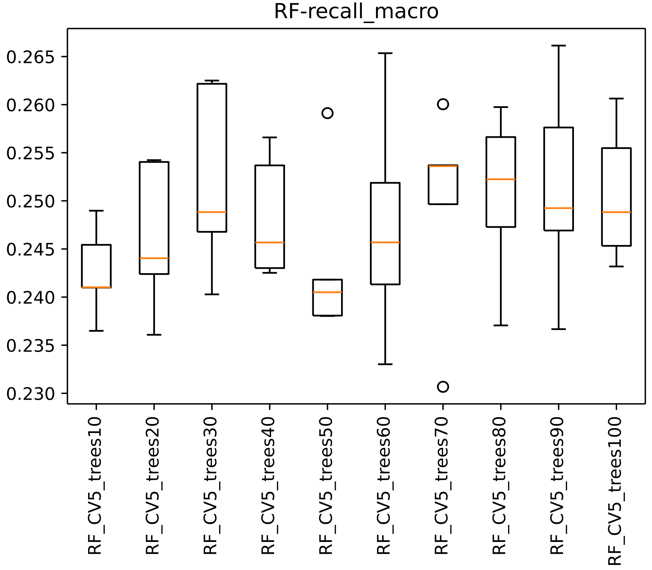


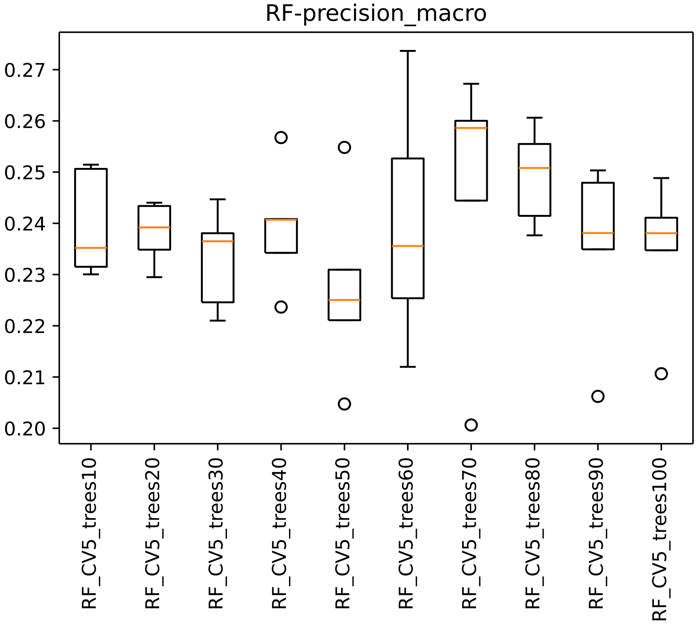
## 

## Random Forest

En el estudio del random forest se usaron diferentes conjuntos de entrenamiento y validación con una k-fold estratificada = 5 y un intervalo de 10 a 100 árboles. Obteniendo con 90 árboles una mejor *accuracy y auc.*

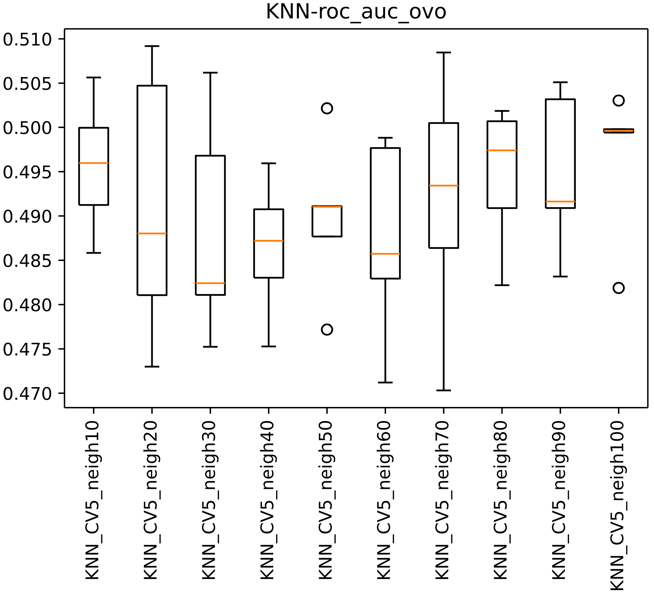
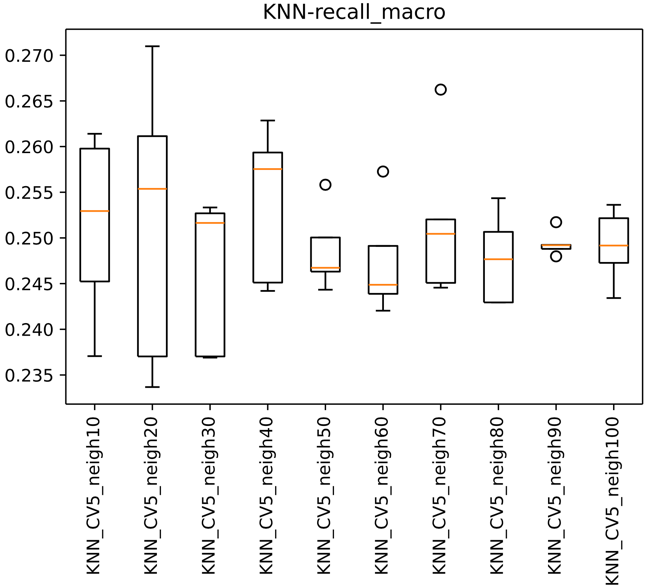
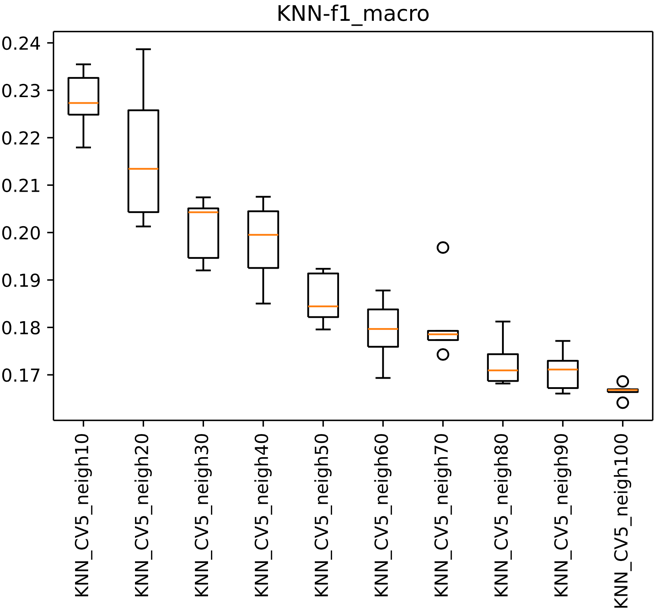
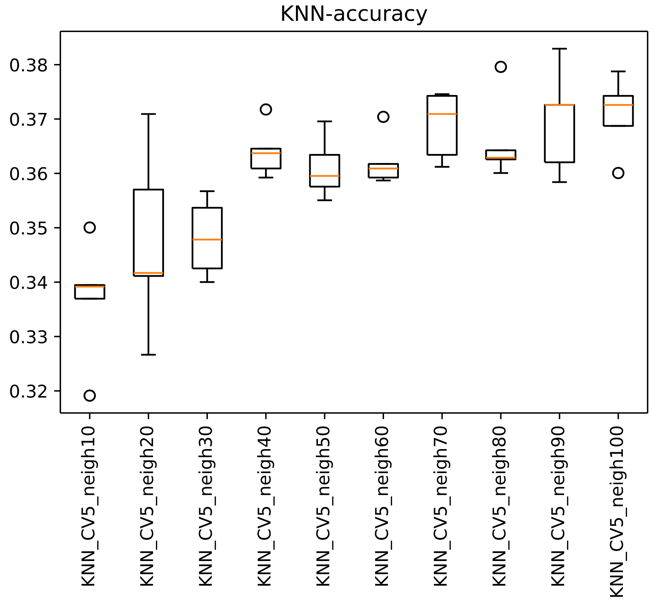


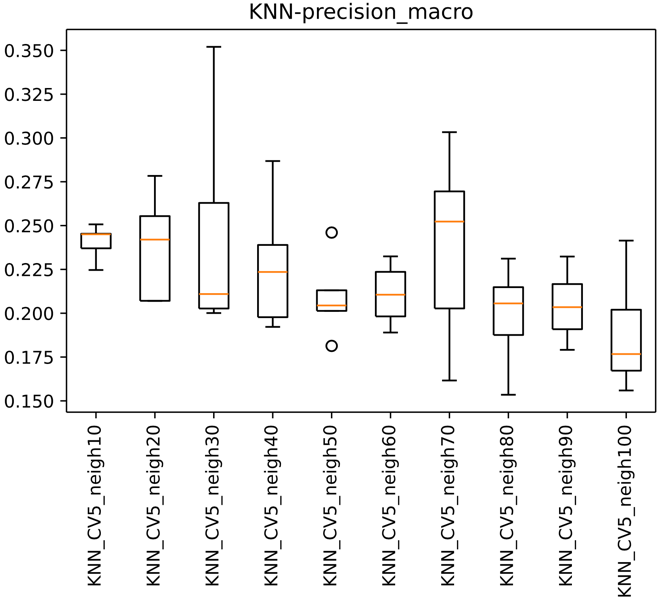




## K-nearest neighbours

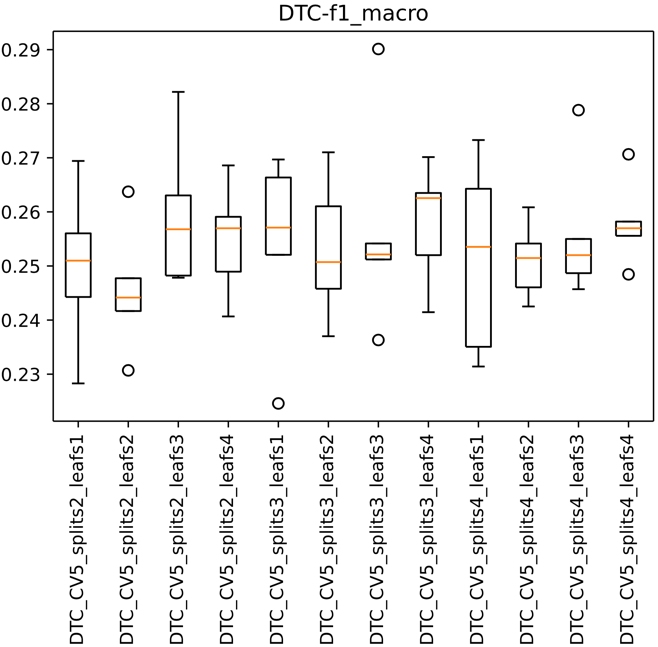
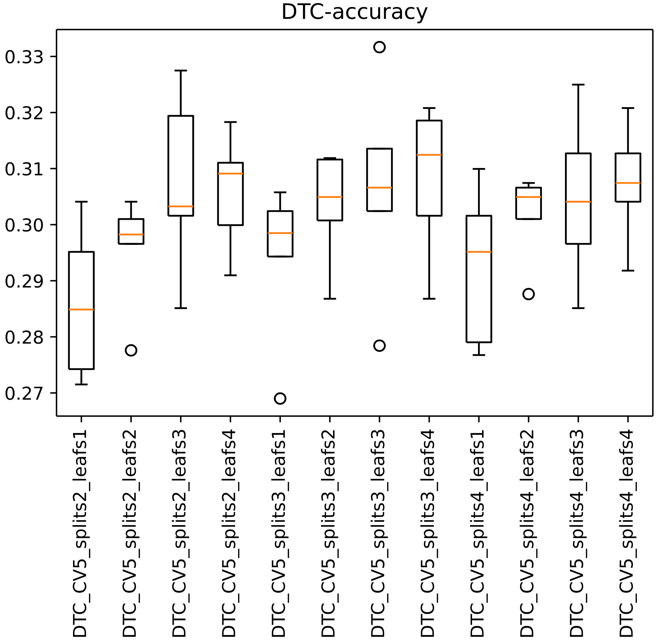
En el estudio de K-nearest neighbours se empleó una validación con una k-fold estratificada = 5 y un intervalo de 10 a 100 neighbours. Obteniendo con 70 a 80 vecinos los mejores valores de *accuracy, auc y precision\_macro.*

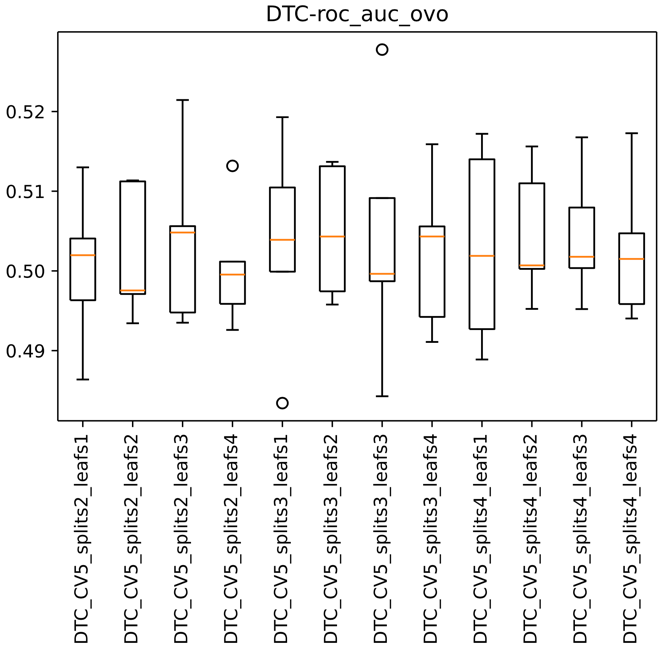
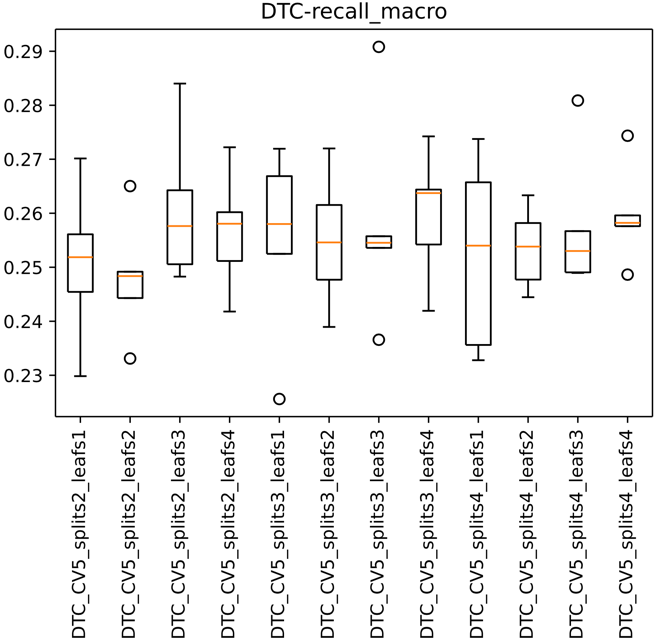


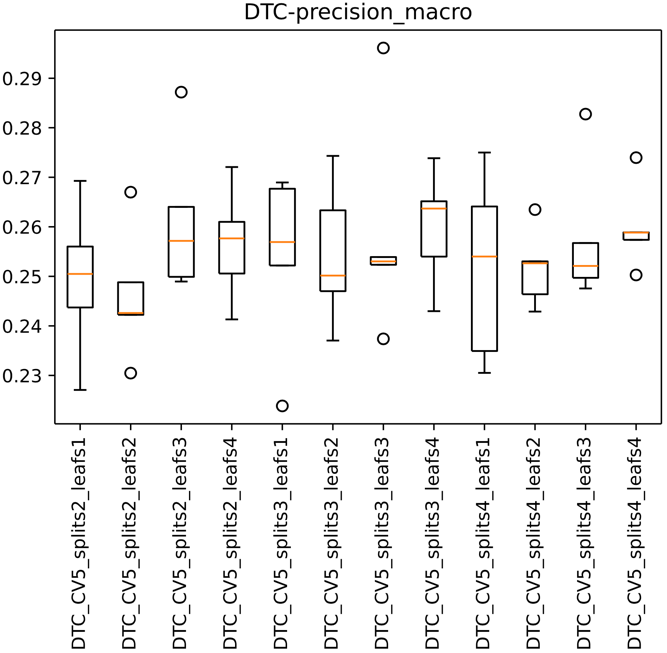


## Decision Tree Classifier

En este estudio con el Decision Tree Classifier, se ha escogido la opción donde cada nodo debe tener 3 elementos para permitirle hacer un split, y cada hoja debe tener al menos 4 elementos para que se le permita ser nodo-hoja.

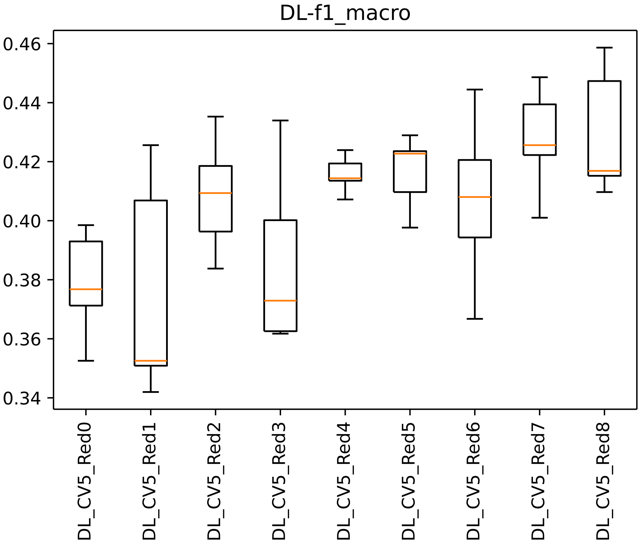
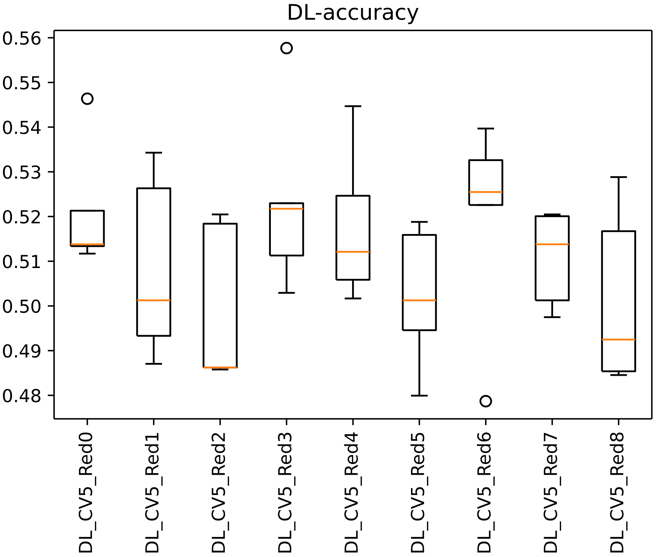


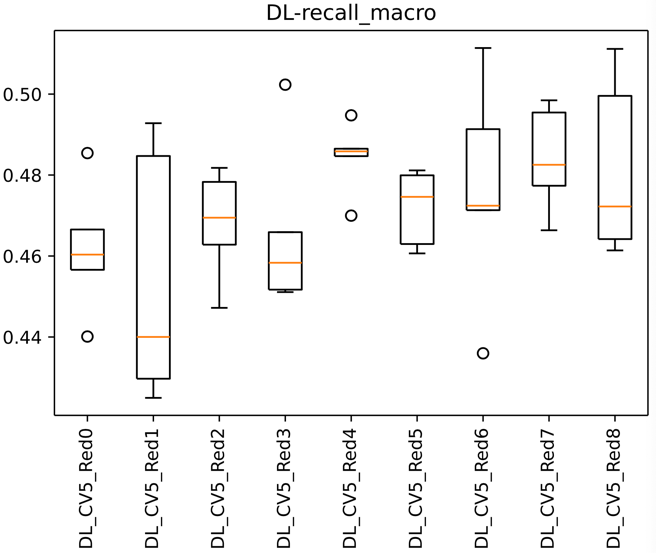
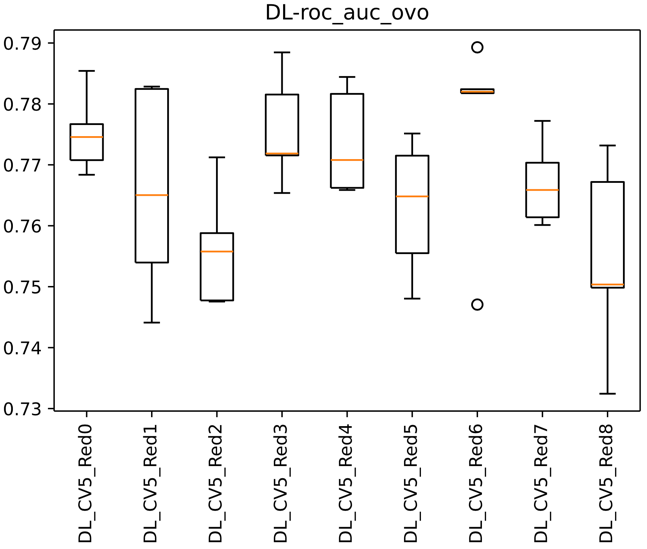


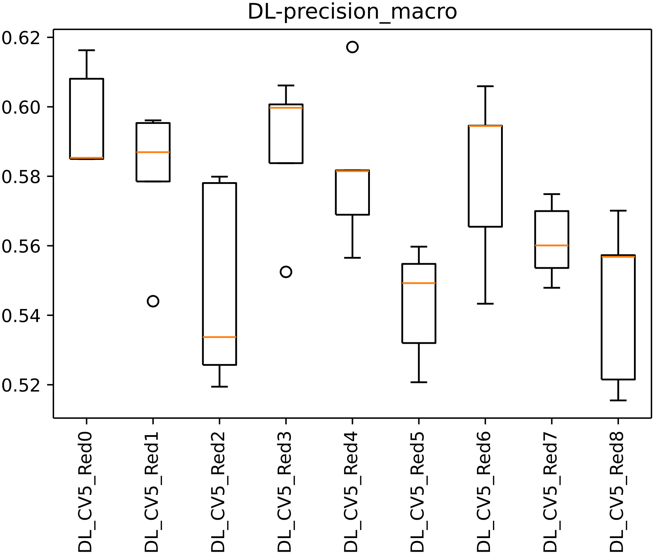


## Redes Neuronales Convolucionales

Como podemos observar en las gráficas, los valores que pueden mejorar los resultados son los obtenidos en DL\_CV5\_Read6, por ser la combinación que arrojó mayor precisión, tiene 3 capas de neuronas convolucionales de tamaño [16, 32, 32] y dos capas de neuronas densas de tamaño [32, 4].



****

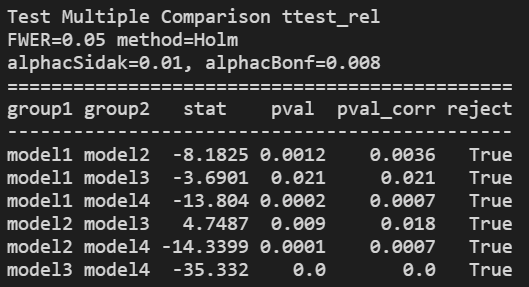
****

**Contraste de Hipótesis**

Primero se muestran los resultados obtenidos aplicando el test de Kruskall-Wallis y el test de Anova. Si se rechaza la hipótesis, entonces se puede afirmar que los modelos son diferentes.

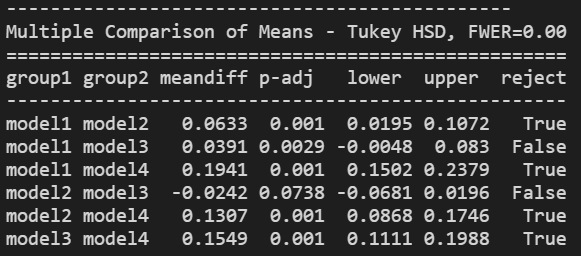


MultiComparison: Estos métodos comprueban si hay diferencias significativas con un p<0.05, corrigiendo el hecho de que se están haciendo múltiples comparaciones que normalmente aumentarían la probabilidad de que se identifique una diferencia significativa. Un resultado de "reject = true" significa que se ha observado una diferencia significativa.

****

Se puede observar que en el test de *multiple comparison* se rechaza todas las hipótesis con lo que todos los modelos son diferentes.

Método de Tukey: Se emplea para ello la función tukeyhsd.

****

Este estudio arroja que el modelo 1. DTC y el modelo 3. RF son iguales y el modelo 2.KNN con el modelo 3. RF son iguales

## Tiempos de ejecución de los modelos entrenados con un único proceso o con multiproceso.

# Conclusiones

Podemos observar un descenso de la exactitud con la reducción de los componentes