Universidad Central

“Marta Abreu” de Las Villas

Facultad de Matemática, Física y Computación

Práctica Laboral Investigativa II

2do Año de Lic. en Ciencia de la Computación

Nombre del Proyecto: Primeros pasos en Machine Learning

Autores:

* Ernesto Camilo Molina Ortega
* Fabián Guerra Carrazana

Índice

[Introducción 3](#_Toc178821329)

[Desarrollo 4](#_Toc178821330)

[**Epígrafe 1:** Principales conceptos dentro del Machine Learning 4](#_Toc178821331)

[**1.1** ¿Qué es el aprendizaje automático? 4](#_Toc178821332)

[**1.2** ¿Qué tipos de aprendizaje existen? 4](#_Toc178821333)

[**1.3** ¿En qué tareas se usa el aprendizaje automático? 4](#_Toc178821334)

[**1.4** ¿Cómo aprende un programa? 4](#_Toc178821335)

[**1.5** Estrategias de entrenamiento 5](#_Toc178821336)

[1.5.1 Entrenamiento y prueba: 5](#_Toc178821337)

[1.5.2 Sobreajuste, ajuste ineficiente e hiperparámetros: 5](#_Toc178821338)

[1.5.3 Ajuste de hiperparámetros y validación: 5](#_Toc178821339)

[1.5.4 Validaciones cruzadas: 5](#_Toc178821340)

[**1.6** Algoritmos principales usados en el Machine Learning 6](#_Toc178821341)

[**1.7** Métricas: 7](#_Toc178821342)

[1.7.1 ¿Qué es y cómo funciona una métrica? 7](#_Toc178821343)

[1.7.2 Métricas más comunes en tareas de regresión: 7](#_Toc178821344)

[1.7.3 Métricas más comunes en tareas de clasificación: 7](#_Toc178821345)

[1.7.4 Métricas más comunes en tareas de clustering: 7](#_Toc178821346)

[**Epígrafe 2:** Flujo de trabajo en el Machine Learning mediate la solución a los problemas propuestos 7](#_Toc178821347)

[**2.1** Clasificación sobre el Iris Dataset 8](#_Toc178821348)

[2.1.1 Descripción del dataset: 8](#_Toc178821349)

[2.1.2 Análisis exploratorio de los datos: 8](#_Toc178821350)

[2.1.3 División del dataset: 8](#_Toc178821351)

[2.1.4 Elección y entrenamiento de los modelos: 9](#_Toc178821352)

[2.1.5 Predicción sobre el conjunto de prueba: 10](#_Toc178821353)

[**2.2** Regresión sobre el Boston Housing Dataset 10](#_Toc178821354)

[2.2.1 Descripción del dataset: 10](#_Toc178821355)

[2.2.2 Análisis exploratorio de los datos: 10](#_Toc178821356)

[2.2.3 División del dataset: 11](#_Toc178821357)

[2.2.4 Elección y entrenamiento de los modelos: 11](#_Toc178821358)

[2.2.5 Predicción sobre el conjunto de prueba y análisis de error: 12](#_Toc178821359)

[**2.3** Clustering 13](#_Toc178821360)

[2.3.1 Descripción de la tarea: 13](#_Toc178821361)

[2.3.2 Generación de los puntos: 13](#_Toc178821362)

[2.3.3 Aplicación de los algoritmos y visualización: 13](#_Toc178821363)

[**Epígrafe 3** Análisis y comparación de los resultados obtenidos 14](#_Toc178821364)

[Conclusiones 15](#_Toc178821365)

[Bibliografía 16](#_Toc178821366)

# Introducción

Este trabajo tuvo como objetivo general la familiarización con los conceptos principales y el flujo de trabajo en el área del aprendizaje automático (Machine Learning), con la finalidad de adquirir habilidades y conocimiento para la inserción en el Grupo de Trabajo Científico. Para ello el trabajo fue estructurado en 3 partes.

Durante la primera parte los autores investigaron sobre conceptos básicos de esta disciplina como Aprendizaje, Tipos de Aprendizaje, tareas en las q se utilizan, estrategias de entrenamiento, algoritmos principales, métricas, entre otros.

La segunda parte se centra en el flujo de trabajo del Machine Learning. Para esto se resolvieron 3 tareas en las cuales se realizaron las siguientes labores.

* Análisis exploratorio de los datos mediante estadística, matrices de correlación, valoración usando gráficos, entre otros.
* Pre-procesamiento de los datos (Análisis de valores faltantes, selección de características).
* Elección de un modelo según la tarea a tratar (Clasificación, Regresión, Clusterización).
* División de los datasets en conjuntos de entrenamiento, validación y de prueba.
* Entrenamientos de los modelos
* Evaluación de los modelos sobre el conjunto de prueba mediante las métricas.

La tercera parte del trabajo se enfocará en el análisis y la comparación de los resultados obtenidos en cada una de las tareas orientadas.

# Desarrollo

## **Epígrafe 1:** Principales conceptos dentro del Machine Learning

### **1.1** ¿Qué es el aprendizaje automático?

El aprendizaje automático es la ciencia de programar computadoras para que puedan aprender de los datos. Este puede entenderse desde muchos aspectos, en 1997 el científico de la computación americano Tom Mitchell proporcionó una definición que ha sido ampliamente citada: “Se dice que un programa informático aprende de la experiencia E con respecto a una tarea T y alguna medida de desempeño P, si su desempeño en T, medido por P, mejora con la experiencia E”.

### **1.2** ¿Qué tipos de aprendizaje existen?

Los sistemas de ML se pueden clasificar en dependencia de la cantidad y tipo de supervisión que reciben durante la formación. Entre las principales categorías se encuentran el aprendizaje supervisado, el aprendizaje no supervisado, el aprendizaje autosupervisado, aprendizaje semisupervisado y aprendizaje por refuerzo, aprendizaje por lotes, aprendizaje en línea, entre otros.

### **1.3** ¿En qué tareas se usa el aprendizaje automático?

Dentro de las tareas más comunes dentro del ML se encuentran la creación de sistemas automáticos que resuelvan problemas de regresión (Pronosticar el ingreso de una empresa el siguiente año, determinar el precio aproximado de una propiedad dentro de una región específica, etc.), clasificación (Filtro de spam, determinar un tipo de vino dadas sus características, clasificación de radiografías para determinar si una persona está enferma o sana, etc.), clusterización (Detección de anomalías,, identificación de grupos de usuarios con intereses o conexiones similares en redes sociales, análisis de clientes en función de sus compras, etc.), análisis de imágenes (Analizar las imágenes de productos para clasificarlos, detección de tumores, etc.), procesamiento de lenguaje natural(Clasificación de texto, detección de comentarios ofensivos en redes, creación de un chatbot o un asistente personal, etc.) entre otros.

### **1.4** ¿Cómo aprende un programa?

En el epígrafe 2.1 se obtiene un programa capaz de determinar la clase (iris setosa, iris virgínica o iris versicolor) a la que pertenece una flor dada sus características. Para poder determinar la clasificación de una nueva flor (instancia) este programa debe ser previamente entrenado con un conjunto de datos (instancias de entrenamiento) la parte de un sistema de ML que aprende de los datos y hace predicciones es llamada modelo

### **1.5** Estrategias de entrenamiento

#### 1.5.1 Entrenamiento y prueba:

La única forma de saber que tan bien se generalizará un modelo a casos nuevos es probarlo en casos nuevos. La forma más extendida de hacerlo es dividir los datasets en dos conjuntos: conjunto de entrenamiento y conjunto de prueba. De esta manera se entrena el modelo usando los datos del conjunto de entrenamiento y luego se prueba usando los datos que este modelo aún no ha visto, ósea el conjunto de prueba. La tasa de error en los casos nuevos se denomina error de generalización y se obtiene una estimación de este al evaluar el modelo en los casos del conjunto de prueba. Este error indica que tan bien se adapta un modelo a casos que nunca ha visto (Es común utilizar el 80% de los datos para entrenar y reservar el resto para las pruebas, sin embargo, esto depende del tamaño del conjunto).

#### 1.5.2 Sobreajuste, ajuste ineficiente e hiperparámetros:

Supongamos que está visitando un país extranjero y un taxista le estafa. Podría sentirse tentado a pensar que en ese país todos los taxistas son ladrones, estaría cayendo en la trampa de generalizar demasiado basándose en su experiencia (En este caso se tiene muy poca experiencia para generalizar bien). En aprendizaje automático a esto se le llama sobreajuste, significa que el modelo se ajusta fuertemente con los datos de entrenamiento pero que no se generaliza bien. El sobreajuste ocurre cuando un modelo es demasiado complejo en relación con la cantidad de ruido (Datos no representativos, como puntos aislados o como resultados del azar) de los datos, si el conjunto de entrenamiento es demasiado pequeño o ruidoso, entonces es probable que el modelo detecte patrones en el ruido mismo (como en el ejemplo del taxista), obviamente estos patrones no se generalizarán bien a nuevos casos.

Restringir un modelo para hacerlo menos complejo y reducir el riesgo de sobreajuste se llama regularización, la cantidad de regularización que se le aplicará durante el aprendizaje se puede controlar medite un hiperparámetro. Un hiperparámetro es un parámetro de un algoritmo de aprendizaje (no del modelo), debe establecerse antes del entrenamiento y su valor permanecerá constante durante este.

El ajuste insuficiente es el problema opuesto al sobreajuste, este ocurre cuando un modelo es demasiado simple para aprender de la estructura subyacente de los datos. Por ejemplo, un modelo lineal de satisfacción con la vida tiende a no adaptarse, la realidad es simplemente mucho más compleja que el modelo, por lo que sus predicciones seguramente serán inexactas incluso en los ejemplos de entrenamiento.

#### 1.5.3 Ajuste de hiperparámetros y validación:

Supongamos que tiene un modelo lineal que se generaliza bien, pero desea aplicar cierta regularización para evitar el sobreajuste, ¿Cómo elegir el valor del hiperparámetro de regularización? Una opción es entrenar varios modelos con valores diferente y escoger el que produce un menor error de generalización, pero al someter el modelo a casos nuevos obtiene resultados muy por debajo de los esperados. Lo que ocurrió es que midió el error de generalización varias veces en el conjunto de prueba y adaptó el modelo y los hiperparámetros para producir el mejor modelo para ese conjunto en particular. Por lo tanto, es poco probable que ante un conjunto de datos nuevos el modelo se mantenga funcionando tan bien. Una solución común a este problema es la validación exclusión, simplemente se reservan una parte de los datos de entrenamiento para evaluar varios modelos candidatos y seleccionar el mejor, a este nuevo conjunto se le llama conjunto de validación (o Dev Set)

#### 1.5.4 Validaciones cruzadas:

Es una técnica que mejora la evaluación de modelos al utilizar múltiples divisiones del conjunto de entrenamiento. El conjunto de datos se divide en K conjuntos (pliegues o folds) y se realizan K rondas de entrenamiento. En cada ronda se utiliza uno de los folds (siempre uno distinto) para validación y los K-1 folds restantes para entrenar. Los resultados de las K rondas se promedian para obtener una mejor medida del rendimiento del modelo.

### **1.6** Algoritmos principales usados en el Machine Learning

**Regresión Lineal (Linear Regression):** Es un algoritmo de aprendizaje supervisado que busca una relación de dependencia lineal entre una variable dependiente y una o varias variables independientes. Se basa en una ecuación lineal

Donde Y es la variable dependiente, es el vector de coeficientes y son las características (predictores). Es decir, el algoritmo de regresión lineal encuentra la recta que mejor se ajusta a un conjunto de puntos n-dimensionales buscando los valores de vector de coeficientes que minimicen la diferencia entre las predicciones del modelo y los datos reales de la variable Y, esto se optimizando la función de error cuadrático

**Regresión Logística (Logistic Regression):** Es un algoritmo de aprendizaje supervisado utilizado para la clasificación binaria, es decir, predice la probabilidad de que una instancia pertenezca a una clase específica. Se basa en una función sigmoidea

Donde P es la probabilidad de pertenecer a la clase positiva y Z es una función lineal de las variables predictoras, esta vez para encontrar el vector de coeficientes que mejor se ajuste a los datos se utiliza el algoritmo de Descenso de Gradiente.

**Máquinas de Vectores de Soporte (Support Vector Machines o SVM):** Es un algoritmo de aprendizaje supervisado ampliamente utilizado para tareas de regresión y clasificación. La idea básica detrás de este algoritmo es tratar de separar dos conjuntos de datos que representan dos clases distintas. Para ello se encuentra el hiperplano que maximice el márgen (distancia entre el hiperplano y los puntos de datos más cercano, llamados vectores de soporte) ente las dos clases. En casos de que las clases no sean linealmente separables, las SVM utilizan funciones kernel para proyectar los datos a un espacio de mayor dimensión donde si lo sean.

**K Vecinos más cercanos (K Nearest Neighbors o KNN):** Es un algoritmo de aprendizaje supervisado usado en tareas de clasificación y regresión que utiliza la distancia entre los datos para clasificar o predecir un nuevo punto de datos. Se basa en la premisa de que los datos similares se encuentran cercanos entre sí.

**Árboles de decisión (Decision Trees):** Es un algoritmo de aprendizaje supervisado usado en tareas de clasificación y regresión. Se basan en una estructura ramificada que representan una serie de reglas para clasificar o predecir un resultado. Cada nodo interno del árbol representa una prueba en un atributo y cada rama representa un posible resultado de la prueba, las hojas del árbol representan las predicciones o las clases.

**Bosques aleatorios (Random Forest):** Este es un algoritmo de aprendizaje de conjuntos que combina múltiples árboles de decisión para mejorar la precisión y la estabilidad del modelo. Cada árbol de decisión se entrena con un subconjunto aleatorio de los datos y un subconjunto aleatorio de las características, para clasificar un nuevo punto de datos se obtienen los resultados de cada árbol y se selecciona la clase más frecuente.

**K-Means:** Es un algoritmo de aprendizaje no supervisado que pertenece a los algoritmos de agrupamiento (clustering), está basado en particiones que intentan dividir un conjunto de datos en K bloques (cluster), donde K es un parámetro especificado por el usuario. Se seleccionan aleatoriamente K centroides como punto de partida, cada punto de datos se asigna al grupo cuyo centroide está más cerca y luego se recalculan los centroides como la media de los puntos que ha sido asignados a cada grupo. Se repite este proceso hasta que los centroides dejen de moverse. Se garantiza que el algoritmo convergerá en un numero finito de pasos (generalmente pequeños), esto se debe a que las distancias medias cuadráticas entre las instancias y los centroides solo puede disminuir con cada paso y al no ser negativa esta convergerá.

**DBSCAN (Density-Based Spatial Clustering of Applications with Noise):** Es un algoritmo de agrupamiento basado en densidad que agrupa puntos de datos que están en zonas densamente pobladas, mientras trata los puntos aislados como ruido. Se establecen dos parámetros Eps (Radio máximo para considerar vecinos) y MinPts (Mínimo de puntos necesarios en un radio Eps para que se considere un punto denso), los puntos que tienen al menos MinPts vecinos dentro de un radio Eps se consideran puntos núcleos. Se encuentran cluster expandiendo los puntos núcleos a través de sus vecinos hasta que se llega a un punto no núcleo o el cluster ya no se expande, los puntos que no están conectados a ningún cluster son clasificados como ruido.

### **1.7** Métricas:

#### 1.7.1 ¿Qué es y cómo funciona una métrica?

Las métricas son herramientas cruciales dentro del ML para evaluar el rendimiento de los modelos, permiten comparar diferentes modelos y determinar la eficiencia de un modelo para resolver un problema específico. Una métrica no es más que una función matemática que cuantifica el rendimiento de un modelo en relación con un conjunto de datos de prueba, generalmente se basa en las comparaciones entre las predicciones del modelo y los valores reales.

#### 1.7.2 Métricas más comunes en tareas de regresión:

* **Error Absoluto Medio (Mean Absolute Error o MAE):** mide la diferencia absoluta promedio entre las predicciones del modelo y los valores reales. Es poco sensible a valores atípicos.
* **Error Cuadrático Medio (Mean Squared Error o MSE):** Mide la diferencia cuadrática promedio entre las predicciones del modelo y los valores reales. Es sensible a valores atípicos.
* **RMSE:** Es la raíz cuadrada el MSE, ofrece una interpretación más intuitiva en la misma escala que la variable objetivo.
* **r2:** Mide la variación en la variable objetivo que puede ser explicada por el modelo. Un r2 cercano a 1 indica que el modelo explica mejor la varianza.

#### 1.7.3 Métricas más comunes en tareas de clasificación:

* **Accuracy (Precisión total):** Mide la proporción de predicciones correctas entre todas las predicciones.
* **Precision (Precisión):** Mide la proporción de predicciones correctas dentro de las instancias predichas para una clase.
* **Recall (Sensibilidad):** Mide la proporción de casos positivos correctamente identificados.
* **F1:** Es la media armónica entre la precisión y la sensibilidad.
* **AUC (Área bajo la curva ROC):** Mide la capacidad de un modelo para distinguir entre clases positivas y negativas.

#### 1.7.4 Métricas más comunes en tareas de clustering:

* **Silhouette Score:** Mide la Calidad de la agrupación en función de las distancias entre los puntos dentro del mismo cluster y la distancia entre los puntos de cluster diferentes.
* **Davies-Bouldin Index:** Mide la calidad de agrupación en función de la relación entre la distancia intra-cluster y la distancia inter-cluster

## **Epígrafe 2:** Flujo de trabajo en el Machine Learning mediate la solución a los problemas propuestos

Fueron propuestos para este trabajo tres tareas de distinto tipo, clasificación, regresión y clusterización. Los problemas fueron resueltos usando la tecnología de Jupyter Notebook de Python y todo el código se encentra en GitHub en el siguiente enlace <https://github.com/FguerraCarrazana/Pr-ctica-laboral-ML>.

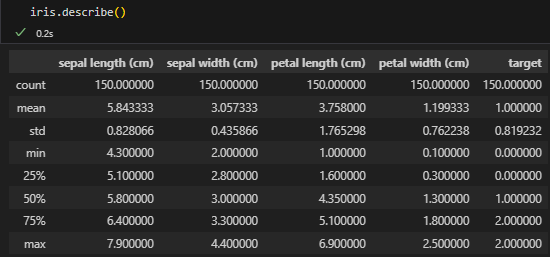
### **2.1** Clasificación sobre el Iris Dataset

#### 2.1.1 Descripción del dataset:

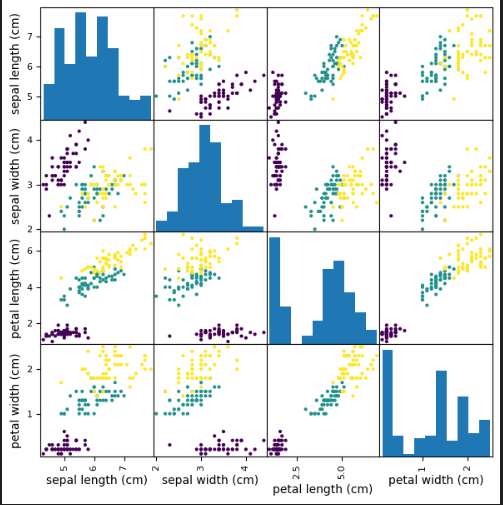
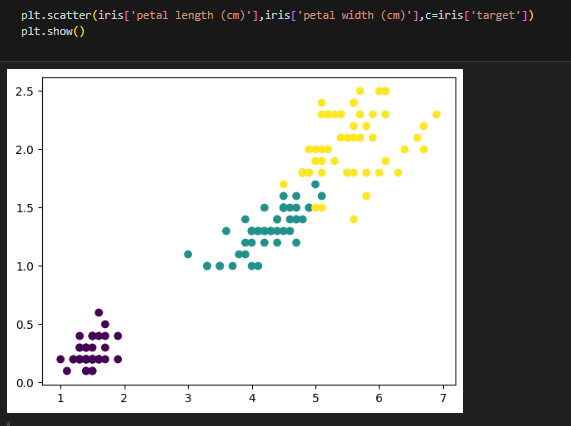
La librería scikit-learning provee un conjunto datasets clásicos entre ellos el Iris dataset. Este dataset está formado por datos de flores de tres clases distintas Iris setosa (0), Iris versicolor (1) e Iris virgínica (2). Las características o predictores de cada instancia son **sepal length (cm), sepal width (cm), petal length (cm), petal width (cm).** Todas las características son valores numéricos reales y a cada clase de flor se le asigna un numero identificador. En este conjunto encontramos un total de 150 instancias, 50 de cada una de las clases. No existen dentro del conjunto valores nulos.

#### 2.1.2 Análisis exploratorio de los datos:

Utilizando el método decribe() de la librería pandas obtenemos algunas estadísticas representativas de los datos

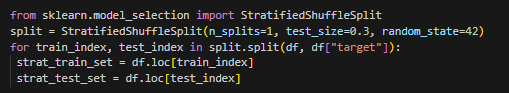


Para observar la correlación entre las características se utilizó la función scatter\_matrix() de Pandas que relaciona todos los atributos dos a dos. La siguiente figura (izquierda) muestra el gráfico, en ella se puede observar que los últimos dos atributos presentan una correlación bastante interesante que se muestra mejor en la figura de la derecha. Como puede observarse el grupo de iris setosa está claramente aislado de los demás, mientras que las clases de versicolor y virgínica no son tan claramente separables pero la mayoría de los puntos están distribuidos en grupos marcados.

#### 2.1.3 División del dataset:

Antes de continuar, se deben dividir el conjunto de datos en entrenamiento y prueba, dado que las clases de las instancias están balanceadas (en proporción [1,1,1]) se dividió el conjunto de manera que ambos conjuntos resultantes mantuvieran la proporción, para esto se usó la función StratifiedShuffleSplit() de scikit-learning.

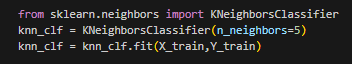


Posteriormente fueron seleccionadas las características más prometedoras (**petal length (cm), petal width (cm)**). Antes de pasar a la elección de los modelos se agrandó el conjunto de entrenamiento usando la función SMOTE() de la librería imblearn.

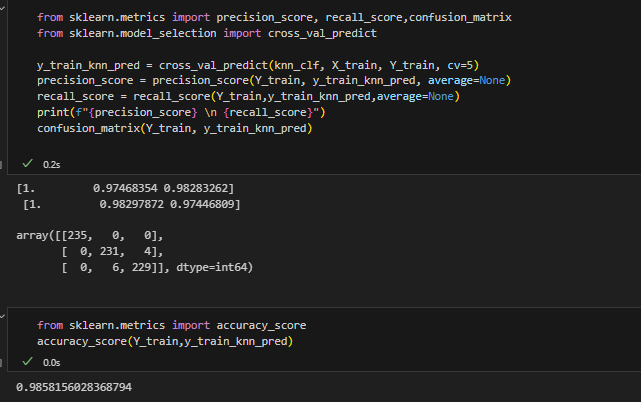
#### 2.1.4 Elección y entrenamiento de los modelos:

Para esta tarea fueron usados los modelos de los clasificadores de KNN (**KNeighborsClassifier()**) y Decision Tree (**DecisionTreeClassifier()**)

**KNeighborsClassifier:** Creación y entrenamiento del modelo



Se obtienen las métricas de precisión (precision) y sensibilidad (recall) usando validación cruzada en el conjunto de entrenamiento y se arrojan los resultados usando la matriz de confusión

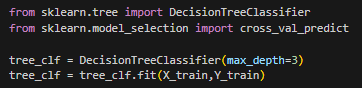


En cuanto a la precisión para la clase 0 (setosa) todas las flores que el modelo predijo dentro de esa clase pertenecían realmente a ella, por lo que se obtuvo un 100% de precisión. En la clase 1 (versicolor) el modelo predijo un total de 237 flores de las cuales 231 pertenecían verdaderamente a esa clase, por lo cual la precisión obtenida es de 97.46%. El modelo clasificó a un total de 233 flores como pertenecientes a la clase 2 (virgínica), de ella 229 fueron correctas arrojando una precisión de 98.28%.

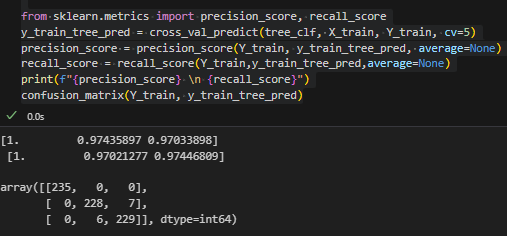
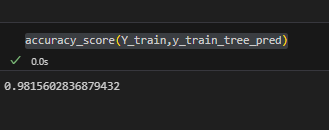
En cuanto a la sensibilidad se puede observar que las flores de la clase 0 fueron clasificadas correctamente en su totalidad los por lo cual la sensibilidad es de 100% para la clase 0. En cuanto las flores de la clase 1 fueron reconocidas correctamente 231 y las 4 restantes fueron confundidas por el modelo y clasificadas como flores de la clase 2 para una sensibilidad del 98.29% para la clase 1. Por último 229 flores de la clase 2 fueron clasificadas de manera correcta y otras 6 fueron incorrectamente clasificadas como flores de la clase 1, arrojando así una sensibilidad del 97.44% para la clase 2.

Finalmente se obtiene un puntaje de accuracy de 98.58% para la validación cruzada.

**DecisionTreeClassifier():** Creación y entrenamiento del modelo



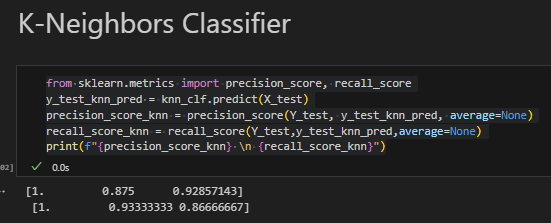
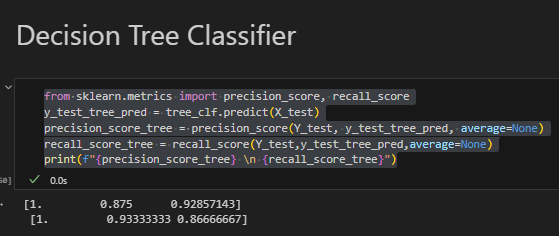
Se evalúa el modelo con validación cruzada y se obtienen los resultados mostrados en las siguientes imágenes (El análisis de estos resultados es el mismo realizado con el modelo anterior)

Estos resultados son ligeramente inferiores a los obtenidos en el modelo anterior.

#### 2.1.5 Predicción sobre el conjunto de prueba:

En las siguientes imágenes se muestra los resultados del modelo sobre el conjunto de prueba

Frente a los datos del conjunto de prueba ambos modelos se han comportado de la misma manera con respecto a las métricas de precisión y sensibilidad, aunque estos resultados parecen bastante inferiores a los obtenidos durante los entrenamientos, esta prueba no es realmente concluyente por la baja cantidad de datos en el conjunto de prueba (15 instancias de cada clase).,

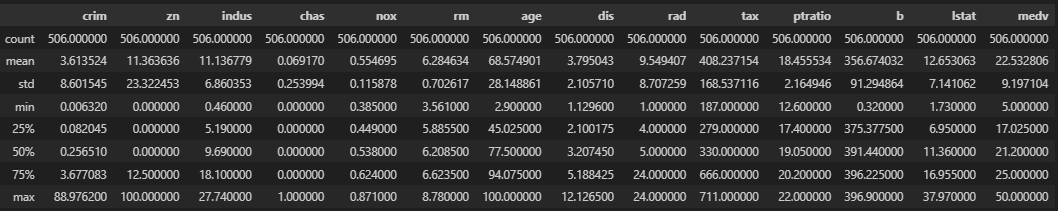
### **2.2** Regresión sobre el Boston Housing Dataset

#### 2.2.1 Descripción del dataset:

El dataset utilizado para esta tarea fue obtenido en el sitio web de Kaggle. Los datos de este dataset presentan un total de 13 características, todas numéricas cuyos nombres son [**crim, zn, indus, chas, nox, rm, age, dis, rad, tax, ptratio, b, lstat]** y la columna **medv** representa el valor medio de la vivienda (En el orden de 10000 USD) y es la variable a predecir. El conjunto contiene un total de 506 instancias de las cuales ninguna tiene valores nulos

#### 2.2.2 Análisis exploratorio de los datos:

Utilizando el método decribe() de la librería pandas obtenemos algunas estadísticas representativas de los datos. Observamos que el valor medio de la vivienda se encuentra entre 5 y 50 unidades, como promedio se tiene 22.53 unidades y como mediana 21.2 pero el 75% .de las instancias tienen un valor medio igual o menor a las 25 unidades por lo cual se puede intuir que los datos con valores cercanos al precio máximo son más atípicos en este conjunto.



Para visualizar la correlación entre los datos se utilizó las función scatter\_matrix() y se utilizó la función heatmap() de la librería seaborn para revisar los valores de la matriz de correlación (no se mostrarán dichos gráficos en este informe debido a su tamaño)

#### 2.2.3 División del dataset:

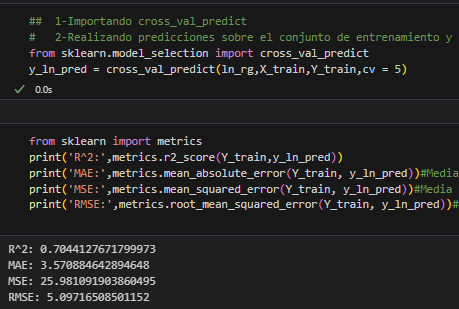
Se dividió el conjunto de datos en conjunto de entrenamiento y de prueba usando la función train\_test\_split()de scikit-learning



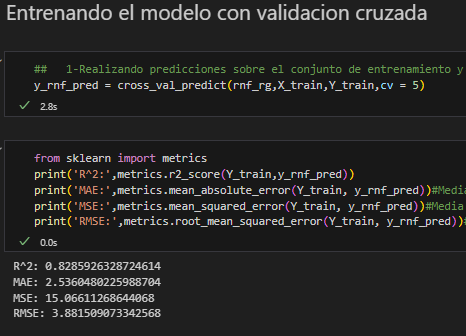
#### 2.2.4 Elección y entrenamiento de los modelos:

Para esta tarea fueron usados los modelos de los clasificadores de Regresión Lineal (**LinearRegression()**), RandomForest (**RandomForestRegressor()**) y SVM

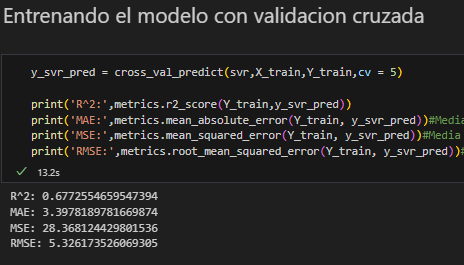
**LinearRegression:** Entrenamiento del modelo y midiendo los resultados de las predicciones con validación cruzada

****

**RandomForestRegressor:** Entrenamiento del modelo y midiendo los resultados de las predicciones con validación cruzada



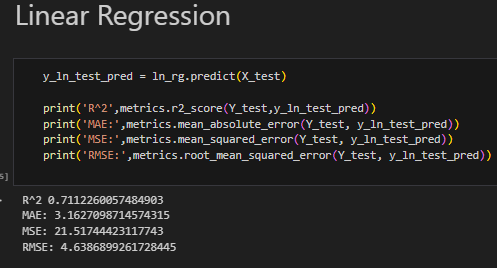
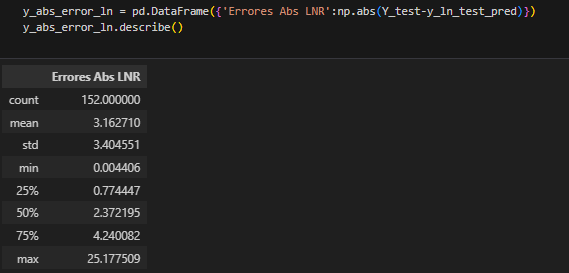
**SVR():** Entrenamiento del modelo y midiendo los resultados de las predicciones con validación cruzada



A priori el segundo modelo parece mucho más prometedor, supera claramente a los demás en todas las métricas.

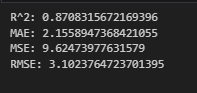
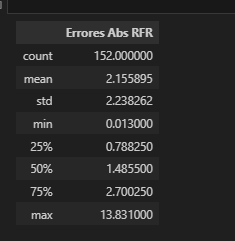
#### 2.2.5 Predicción sobre el conjunto de prueba y análisis de error:

**LinearRegression:**

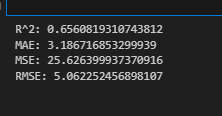
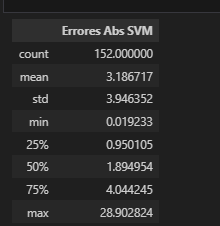
La predicción arroja un RMSE de 4.638, lo cual mejora el 5.097 obtenido en el entrenamiento, el error absoluto medio también mejoró, por lo tanto, parece que el modelo se generalizó bien. Para analizar más a detalle el error en la predicción se creó un DataFrame con el valor de los errores absolutos. Analizando este DataFrame se descubre que la predicción más desacertada tuvo un error de más de 25 unidades, bastante alto, pero parecen ser mayormente casos aislados ya que analizando los percentiles se puede observar que la mitad de las predicciones (76 instancias) tuvieron un error inferior a 2.38 unidades y el 75% de estas (114 instancias) mantienen un error por debajo de las 4.25 unidades.

**RandomForestRegressor:**

La predicción con este modelo arroja un RMSE de 3.102 mucho mejor que el 3.881 obtenido en el entrenamiento y un MAE de 2.155 que también es mejor que el 2.536 del entrenamiento. El modelo parece generalizarse muy bien y se comporta mucho mejor que la regresión lineal en cuanto a las métricas. Haciendo el análisis más profundo del error absoluto de las predicciones, esta vez se obtuvo un error máximo de 13.831 unidades, también es mucho mejor que las más de 25 unidades en el modelo anterior. También la mediana de los errores baja considerablemente con un buen 1.485 y el 75% de las predicciones tuvieron un error que fue menor o igual que aproximadamente 2.7 unidades.

**SVR:**

Con este modelo las predicciones presentaron un RMSE de 5.062 y un MAE de 3.186, los cuales también son mejores que los resultados obtenidos en el entrenamiento pero que lo convierten en el modelo con las peores métricas de los tres. El máximo error en las predicciones fue de casi 29 unidades, casi 4 más que en el modelo de Regresión Lineal y más de 15 unidades que el modelo del Bosque Aleatorio. Los valores de los percentiles son mejores que los de la Regresión Lineal, pero al parecer el modelo se vuelve muy susceptible a los datos atípicos.

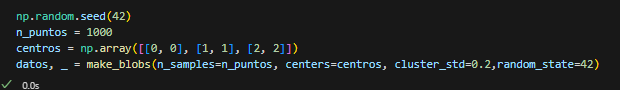
### **2.3** Clustering

#### 2.3.1 Descripción de la tarea:

Generar de forma artificial y aleatoria un conjunto de datos bidimensionales (al menos 1000 puntos) que se encuentren distribuidos de forma uniforme alrededor de los puntos (0,0), (1,1) y (2,2) en un radio de 1. Los grupos deben poseer un tamaño similar. Aplicar dos algoritmos de clusterización y aprovechar la bidimensionalidad de los datos para visualizarlos.

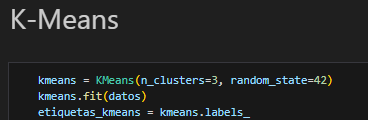
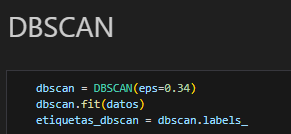
#### 2.3.2 Generación de los puntos:

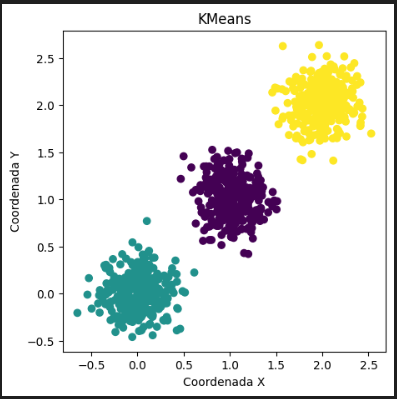
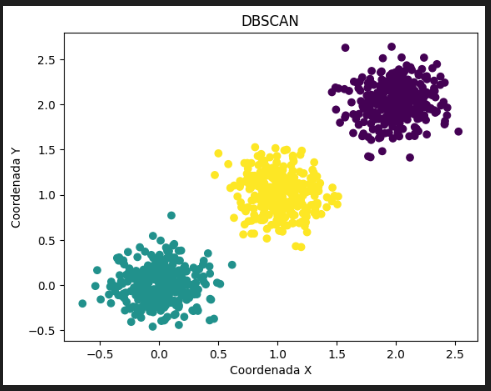
Se seleccionan los centros y la cantidad de puntos y se generan los puntos usando la función **make\_blobs()** de scikit-learning.



#### 2.3.3 Aplicación de los algoritmos y visualización:

Para agrupar los puntos generados se seleccionaron los algoritmos de K-Means y DBSCAN, el funcionamiento de ambos algoritmos fue explicado en el epígrafe 1.6. Al algoritmo de K-Means se le pasa como parámetro la cantidad de cluster. En el caso del DBSCAN se le especifica el valor del parámetro eps, mientras que el valor de MinPts es por defecto igual a 5. Finalmente se visualizan los datos generados.

 ****

## **Epígrafe 3** Análisis y comparación de los resultados obtenidos

En la tarea de clasificación sobre el Iris Dataset no hubo diferencia entre los modelos a la hora de la predicción, solo el modelo de K vecinos más cercanos se mostró ligeramente superior en el entrenamiento, ambos modelos tuvieron tiempos de ejecución similares y se usaron las métricas más comunes y sencillas de comprender en las tareas de clasificación (precision, recall, accuracy).

En la tarea de regresión sobre el Boston Housing Dataset el modelo del Bosque Aleatorio se mostró claramente superior, con respecto a las mediciones de todas las métricas, a los modelos de Regresión Lineal y de la Máquina de Vectores de Soporte, lo cual podría justificarse por la falta de linealidad que presentan los datos y la alta dimensionalidad de los mismos. Se comprobó que las predicciones del Bosque Aleatorio tenían errores mucho más pequeños que los demás modelos y se entrenó en un tiempo notablemente inferior al tiempo requerido por la Máquina de Vectores de Soporte, y aunque fue superior al tiempo requerido por la Regresión Lineal este tiempo puede ser considerado aceptable y dada la superioridad mostrada en las métricas este es, sin dudas, el mejor modelo para esta tarea. Las métricas utilizadas en esta tarea fueron las más representativas y comunes en las tareas de regresión (MAE, MSE, RMSE, r2).

En la tarea de Clusterización ambos algoritmos mostraron igual rendimiento en las mediciones realizadas, el algoritmo de K-means puede preferirse al DBSCAN debido a que tiene una mayor sencillez a la hora de seleccionar los parámetros, pero realmente ambos modelos presentan los resultados similares por lo cual ambos se consideran óptimos para el cumplimiento de la tarea.

# Conclusiones

El presente trabajo ha permitido profundizar en los fundamentos y las aplicaciones del aprendizaje automático, abordando de manera sistemática los aspectos teóricos y prácticos que sustentan esta disciplina en constante evolución. A través del uso de datasets clásicos, se ha podido implementar un flujo de trabajo que sigue las prácticas más comunes del ámbito de Machine Learning, utilizando tecnologías y herramientas estándar de la industria, como Python, las librerías NumPy, Pandas, Scikit-learning y Matplotlib así como el trabajo con Jupyter Notebook. Esto ha facilitado no solo la comprensión de los conceptos, sino también la aplicación de los mismos en contextos de clasificación, regresión y clusterización. Se implementó un flujo de trabajo estructurado que incluyó análisis exploratorio de datos, preprocesamiento de datos, elección de modelos y evaluación. La división del dataset en conjuntos de entrenamiento, validación y prueba garantizó la robustez del proceso de evaluación, permitiendo ajustar los modelos mediante técnicas de afinación efectiva y se compararon y analizaron los resultados obtenidos. En conclusión, este trabajo ha proporcionado una base sólida tanto en teoría como en práctica del aprendizaje automático, destacando la importancia de seguir un enfoque sistemático y crítico al tratar con datos y modelos, lo cual es esencial para el éxito en proyectos futuros que involucren Machine Learning.

# Bibliografía

1. Developer Team. (2007). *Scikit-learn.* Retrieved from <https://sickit-learn.org/>
2. Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learning, Keras and TensorFlow.* O'Relly Media.
3. Goldbloom, A., & Howard, J. (2010). *Kaggle.*

Principio del formulario