|  |
| --- |
| Título |
| Clasificación de mangos mediante técnicas de aprendizaje |
| Resumen |
| TBD |

|  |
| --- |
| Autores |
| Daniel Vidal Soroa |
| Juan Diego Peña N |
| 19/03/2022 |

|  |  |  |  |
| --- | --- | --- | --- |
| Rev. | Descripción | Elaborado | Aprobado |
| 01 | Primera edición |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Índice

[1. Introducción 3](#_Toc99121580)

[1.1. Objeto 3](#_Toc99121581)

[1.2. Alcance 3](#_Toc99121582)

[Normas y referencias 3](#_Toc99121583)

[1.3. Definiciones y abreviaturas 3](#_Toc99121584)

[2. Descripción general 4](#_Toc99121585)

[3. Diseño e implementación 4](#_Toc99121586)

[3.1. Preprocesamiento de datos 5](#_Toc99121587)

[3.2. Evaluación de modelos y optimización de hiperparametros 5](#_Toc99121588)

[3.3. Comparación de modelos y selección 10](#_Toc99121589)

[4. Conclusiones 10](#_Toc99121590)

[5. ANEXOS: 12](#_Toc99121591)

[5.1. ANEXO I : ANALISIS DE HIPER PARAMETROS K-NN 12](#_Toc99121592)

# Introducción

## Objeto

El objetivo de este proyecto es implementar técnicas de aprendizaje automático supervisado sobre un dataset definido. Cada método se evaluará escogiendo las métricas de error más adecuadas, para luego optimizar sus hiperparametros y determinar el de mayor rendimiento.

## Alcance

El proyecto incluye la aplicabilidad de modelos ya existentes en las librerías scikitlearn [1] y keras [2]. Así como el uso datos de dominio público, el proyecto no contempla el uso de técnicas de aprendizaje no supervisado y se ceñirá únicamente a las técnicas: LR, KNN, RF y CNN.

## Normas y referencias

|  |  |
| --- | --- |
| [1] | «Scikit learn,» [En línea]. Available: https://scikit-learn.org/stable/. |
| [2] | «TensorFlow,» [En línea]. Available: https://www.tensorflow.org/guide/keras?hl=es-419. |

## Definiciones y abreviaturas

**ML** Machine learning (*Aprendizaje automatico)*

**LDA** Linear Discriminant Analysis (*Discriminante lineal)*

***KNN*** K-Nearest Neighbor (K-Vecinos)

***RF*** *Random Forest (Arbol aleatorio)*

***CNN*** Convolutional neural networks (Redes neuronales convucionales)

# Descripción general

El proyecto se motiva en servir como una herramienta base para países productores de mango al eliminar un control de calidad manual que posibilita el daño de la fruta y abarata el costo de otros métodos de control como el uso resonancias magnéticas. Además, debido a la migración rural negativa y el corto período de conservación de esta fruta, cada año se pueden pierden millones de toneladas debido a falta de mano de obra y errores de clasificación.

Los datos se seleccionaron de la plataforma kaggle [3] el cual cuenta con tres clases de mango y 200 imágenes por clase:

* ***Clase extra:*** Usados generalmente para la exportación, no contienen defectos o son lo suficientemente leves para no afectar su aspecto, calidad o conservación en general.
* ***Clase I:*** Destinados al consumo local, de buena calidad con algunos defectos como quemaduras por el sol, rozaduras, etc.
* ***Clase II:*** Reservados normalmente para el procesamiento industrial, cuentan con los mismos defectos de la Clase I a una escala mayor.

A picture containing fruit

Description automatically generated

Figura 1. Clases de mango disponible

# Diseño e implementación

El proceso de implementación de las técnicas de aprendizaje se basa en el propuesto en la guía de usuario de scikit-learn. Este se basa en un flujo de trabajo típico para este tipo de aplicaciones.

En primer lugar, se dividen los datos en dos grupos, uno de validación y otro de entrenamiento. Posteriormente, se optimizan los hiperarametros de cada modelo para consecutivamente elegir el modelo más adecuado. Tanto la selección de hiperparametros como de modelo se hace mediante un test estadístico. Se muestra un diagrama del flujo en el que se basa el proyecto:

Diagram

Description automatically generated

Figura 2. Flujo de trabajo típico

## Preprocesamiento de datos

Primeramente, se definió un tamaño de imagen de 32x32 pixeles con el fin de reducir el tiempo de procesamiento y extraer de manera más eficiente características relevantes de las imágenes. Seguidamente, utilizando la librería de tensor Flow, se generaron 1800 imágenes nuevas para cada clase. Estas imágenes generadas se usan como entrenamiento y las 200 imágenes originales se dejan como test de validación. Los datos también fueron reorganizados aleatoriamente para que el orden no sea un factor determinante para los clasificadores.

A picture containing shape

Description automatically generated

Figura 3. Ejemplo de imágenes generadas das.

## Evaluación de modelos y optimización de hiperparametros

Las métricas utilizadas para la evaluación de todos los modelos serán, la exactitud, sensibilidad, precisión, f1-score, especificidad y área bajo la curva ROC. Con estas métricas y en cojuncion con el método de validación cruzada se hará una evaluación objetiva de cada clasificador. Ya que el conjunto de datos es relativamente mediano (6000 imágenes), los datos de entrenamiento se dividirán en 5 grupos diferentes.

### Discriminante lineal (LDA)

Esta técnica se basa en encontrar el hiperplano que mejor separe los datos proporcionados. Ya que este método cuenta con pocos hiperparametros, se realizó un análisis de los tipos de *solvers:*

|  |  |
| --- | --- |
| Parámetros | Parámetros propuestos |
| Solver | [svd, lsqr] |

Tabla . Hiperparametros analizados.

#### Resultado hiperparametros

A continuación, se muestran los resultados obtenidos para cada métrica de error propuesta evaluando los dos tipos de solver:

A picture containing graphical user interface

Description automatically generated

Figura . Resultados recall.

Chart, box and whisker chart

Description automatically generated

Figura . Resultados precisión.

A picture containing graphical user interface

Description automatically generated

Figura . Resultados F1.

A picture containing graphical user interface

Description automatically generated

Figura Resultados exactitud.

Graphical user interface

Description automatically generated with low confidence

Figura . Resultados, área bajo la curva ROC.

Aunque en los resultados se evidencia un claro mejor desempeño por parte del solver *svd,* se realizó una prueba estadística de wilcoxon que comprueba que los modelos son diferentes estadísticamente, por lo que el entrenamiento final se realizó con el solver *svd.*

#### Evaluación final del modelo

Luego de determinar la mejor combinación de hiperparametros, mediante validación cruzada, se hace una validación final con los datos dispuestos para este propósito. A continuación, los resultados:

Chart, treemap chart

Description automatically generated

Figura . Matriz de confusión para el clasificador LDA.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Sensibilidad* | Precisión | F1 | Exactitud | Area bajo la curva ROC | Especificidad |
| 0.945 | 0.945 | 0.945 | 0.945 | 0.972 | 0.972 |

Tabla . Resultados en datos de validación.

### K-Vecinos (KNN)

El clasificador KNN se encarga de almacenar los datos de entrenamiento y realiza la clasificación basándose en una votación de vecinos. Cada vez que se realiza una predicción, se le asigna una clase a la entrada basado en la clase preasignada de los vecinos más cercanos.

Uno de los algoritmos usados por el clasificador para escoger el vecino más cercano es *KD Tree*, que usa como uno de sus parámetros el *Leaf size*  y será tomado en cuenta para el análisis.

Otro hiperparametro a tener en cuenta es el número de vecinos empleado para realizar la clasificación, tomando en cuenta la recomendación de que este número sea menor a la raíz cuadrada del número de datos, se definió un rango de vecinos a evaluar.

Finalmente se evalúa también el modelo con dos definiciones de distancia mínima diferente, la euclidiana y la de Minkowski. A continuación, se muestra u resumen de los parámetros utilizados en la optimización:

|  |  |
| --- | --- |
| Parámetros | Parámetros propuestos |
| Leaf size | [1, 11, 21, 31, 41] |
| K-vecinos | [5, 13, 21, 29, 37] |
| Tipo de distancia[[1]](#footnote-2) | [Manhattan, Euclidiana] |

Tabla 3. Hiperparametros a optimizar.

#### Resultados:

Ya que se tomaron en cuenta 250 combinaciones de hiperparametros, se mostrará solamente los 5 mejores resultados de cada métrica, en el anexo I se muestran las gráficas completas.

Chart, box and whisker chart

Description automatically generated

Figura 10. Resultados *Recall*

Chart, box and whisker chart

Description automatically generated

Figura . Resultados precisión.

Chart, box and whisker chart

Description automatically generated

Figura . Resultados f1.

Chart, box and whisker chart

Description automatically generated

Figura . Resultados exactitud.

Chart, box and whisker chart

Description automatically generated

Figura . Resultados, área bajo la curva roc.

Ya que el comportamiento de todas las métricas de error exhibe un comportamiento similar, se hace uso del *accuracy* o exactitud como métrica para realizar un análisis de hipótesis y decidir el mejor modelo. Cabe destacar por lo que la exactitud es un buen indicador para el análisis de hipótesis.

Usando un test de Wilconxon, se compara la exactitud de cada una de las cinco divisiones de la validación cruzada. Dando como resultado que los modelos son iguales, de manera que se entrena el modelo más eficiente computacionalmente con los siguientes parámetros:

|  |  |  |
| --- | --- | --- |
| *Leaf size* | *# Vecinos* | *Tipo de distancia* |
| 41 | 5 | manhattan |

Tabla . Parámetros finales del modelo.

#### Evaluación final del modelo

La prueba final del modelo se realiza con los datos de validación, a continuación, se muestran los resultados:

Chart

Description automatically generated

Figura . Matriz de confusión, modelo KNN.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Sensibilidad* | Precisión | F1 | Exactitud | Área bajo la curva ROC | Especificidad |
| 0.892 | 0.895 | 0.891 | 0.892 | 0.980 | 0.946 |

Tabla . Métricas finales en datos de validación.

### Bosque aleatorio (RF)

Random forest o bosque aleatorio, realiza la clasificación a través de una serie de pruebas a cada variable hasta alcanzar su clasificación, en cada prueba el conjunto de datos se divide en un cierto número, la cantidad de divisiones son controladas por el hiperparametro *min\_samples\_split* el cual se usará para el análisis.

El número máximo de variables que se evalúa en cada prueba es otro hiperparametro escogido para el análisis, así como el *boostrap* y *mean\_samples\_leaf*

|  |  |
| --- | --- |
| Parámetros | Parámetros propuestos |
| Mínimo de divisiones | [2,5,10] |
| Mínimo de hojas | [1,2,4] |
| Bootstrap | [Falso, Verdadero] |
| Máximo de variables | [Automatico,raíz cuadrada] |

Tabla . Combinación de hiperparametros a evaluar.

#### Resultados

Ya que se entrenaron mas de 180 combinaciones, se muestran solo los mejores dos resultados de cada métrica, los resultados completos se encuentran en el anexo I

A picture containing text

Description automatically generated

Figura . Resultados recall.

A picture containing diagram

Description automatically generated

Figura . Resultados precisión.

Text

Description automatically generated with medium confidence

Figura . Resultados f1.

Text

Description automatically generated with medium confidence

Figura . Resultados exactitud.

A picture containing text

Description automatically generated

Figura . Resultados área bajo la curva ROC.

Aplicando de nuevo el test estadístico de Wilcoxon, se llega al resultado que los modelos son iguales, por lo que se usa los parámetros que agilicen la velocidad de cómputo:

|  |  |  |  |
| --- | --- | --- | --- |
| *Boostrap* | *Máximo de variables* | *Mínimo de hojas* | *Mínimo de divisiones* |
| Falso | Automático | 2 | 5 |

Tabla . Modelo final.

#### Evaluación final del modelo

Luego de escoger el mejor modelo, se evalúa el desempeño con los datos de validación:

Chart

Description automatically generated

Figura . Matriz de confusión con los datos de validación.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Sensibilidad* | Precisión | F1 | Exactitud | Área bajo la curva ROC | Especificidad |
| 0.982 | 0.982 | 0.982 | 0.982 | 0.999 | 0.991 |

Tabla . Métricas en datos de validación.

### Redes neuronales convencionales (CNN)

Los clasificadores de redes artificiales convolucionales se basan en la aplicación de filtros sucesivos a imágenes, que facilitan la extracción de características que serán usadas luego para realizar la clasificación.

Para el análisis de hiperparametros se tomaron en cuenta diferentes arquitecturas predefinidas en la librería keras. Estas redes cuentan con una configuración de capas distinta por lo que se evalúan distintos hiperparametros en cada arquitectura.

* ResNet50: A la salida de esta arquitectura se añade una red neuronal de clasificación con tres capas, la primera se encarga de aplanar los datos de salida de la ResNet50, la segunda capa cuenta con 64 salidas y finalmente la última con las probabilidades de cada clase.
* VGG16: Cuenta con 5 capas a la salida de la extracción de parámetros, la primera aplana las imágenes para posteriormente pasar por una capa densa con 100 salidas. Finalmente se aplica una capa de *dropout* con una ratio de 0.5, seguido la capa final de salida con el numero de clases.

#### Resultados:

Los dos modelos propuestos se optimizaron usando la métrica de exactitud. A continuación, se muestra el resultado de las dos arquitecturas con diferentes hiperparametros:

Chart, scatter chart

Description automatically generated

Figura . Arquitectura VGG16.

Chart, line chart, scatter chart

Description automatically generated

Figura . Arquitectura ResNet50

Finalmente se escoge el modelo que hace uso de la arquitectura VGG16.

#### Evaluación final del modelo

A continuación se muestran las métricas finales con los datos de validación:

Chart

Description automatically generated

Figura . Matriz de confusión modelo CNN

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Sensibilidad* | Precisión | F1 | Exactitud | Área bajo la curva ROC | Especificidad |
| 0.85 | 0.87 | 0.86 | 0.86 | 0.97 | 0.93 |

Tabla . Métricas finales modelo CNN.

## Comparación de modelos y selección

Luego de optimizar cada clasificador, se debe seleccionar el mejor. La métrica encargada realizar la comparación es la exactitud. Por cada modelo se realiza un test estadístico para validar si los datos son iguales o no.

Chart, box and whisker chart

Description automatically generated

Luego de comprar los cuatro modelos con el contraste de hipótesis de Kruskall-Wallis con un Alpha de 0.05 se determino que los modelos son distintos. De manera que el modelo con mayor exactitud es el ***random forest***.

# Conclusiones

# ANEXOS:

## ANEXO I : ANALISIS DE HIPER PARAMETROS K-NN

Chart

Description automatically generated with medium confidence

Chart, box and whisker chart

Description automatically generated

Chart

Description automatically generated

Chart

Description automatically generated

Chart, box and whisker chart

Description automatically generated

## ANEXO I : ANALISIS DE HIPER PARAMETROS RF

Chart

Description automatically generated

Chart, box and whisker chart

Description automatically generated

Chart

Description automatically generated

Chart, box and whisker chart

Description automatically generated

Chart

Description automatically generated

1. Representado en scikit-learn como p ( 1 =Manhattan 2 = Euclidiana ) [↑](#footnote-ref-2)