Proyecto final: Clasificación de minifiguras LEGO.

Carolina Maria Burgos Anillo, Mateo Gómez Abril, William David Moreno Rendón

Repositorio GitHub: https://github.com/cmba-alt/lego_classification

Video Youtube: https://youtu.be/gRPajPmXRQg

Introducción

En el presente proyecto se comparan tres diferentes clasificadores sobre un *dataset* que describe minifuguras de LEGO a partir de catacterísticas de su distribución de color. La imágenes se extrajeron del *dataset* de 'LEGO Minifigures', disponible en Kaggle [1], y con ellas se creó un nuevo *dataset* obtenido en el proyecto realizado en la asignatura de procesamiento de imágenes y visión, contenido en el repositorio como *lego_data.csv* y compuesto por 121 muestras, cada una con 64 caraceterísticas y con etiquetas correspondientes a 4 clases.

Objetivo general

Clasificar imágenes de figuras de LEGO por serie, a partir de sus características de color**.**

Objetivos específicos

1. Entrenar a partir de histogramas bidimensionales obtenidos de imágenes de figuras de LEGO un conjunto de por lo menos 2 clasificadores.
2. Evaluar los clasificadores a partir de una metodología vista en clase.

Desarrollo y resultados

El *dataset* de *lego_data.csv* se compone de 65 columnas y 121 filas. 64 de estas columnas representan las características de Hue y saturación de un histograma de color 2D extraído de imágenes recortadas de figuras LEGO y una columna se compone de 4 clases de series. El histograma se extrae en el espacio de color HSV, donde se utilizó la librería de OpenCV (función calcHist) para obtener distribución del color en las imágenes a partir de histogramas bidimensionales. Si se toma la salida completa de la función los rangos de valores de Hue irían entre 0 y 179 grados, mientras que los de saturación entre 0-255. Cabe resaltar que si tomamos todos estos datos tendríamos 46080 características por cada fila lo que sería computacionalmente ineficiente, lo que se hace es escalizar estás características en un histograma que se compone de 16 valores para Hue y 4 para saturación, obteniendo las 64 características descritas al comienzo. A continuación se muestra la distribución de imágenes para cada clase:

Tabla 1. Número de imágenes por clase.

Clase	Número de imágenes
Marvel (1)	31
star-wars (2)	36
jurassic-world (3)	28
harry-potter (4)	26

Teniendo el *.csv* con las 64 características y las 4 clases, se dispone a normalizar los datos de las características tanto para Train como para Test , teniendo un 70% para train y 30% para test.

1. Clasificación con regresión logística

Un clasificador por regresión logística se utiliza comúnmente en casos de clasificación binaria como alternativa a la regresión lineal pues este tiene la ventaja de que se puede determinar un valor umbral a partir del cual se clasifican las muestras entre una clase u otra. Sin embargo, es posible implementar una regresión logísitica para un problema con múltiples clases. Para la aplicación presente, se utilizó el LogisticRegressor de sklearn. En la figura 1 se presentan los resultados obtenidos.

El LogisticRegressor de sklearn utiliza una configuración llamada 'multinomial' en lugar de 'OvR' para implementar el clasisificador multiclase. Para 'multinomial', la pérdida minimizada es el fit fit de la función de pérdida, a través de toda la distribución de probabilidad.

Los pasos seguidos para realizar la implementación de este clasificador se presentan en el diagrama de flujo 1.

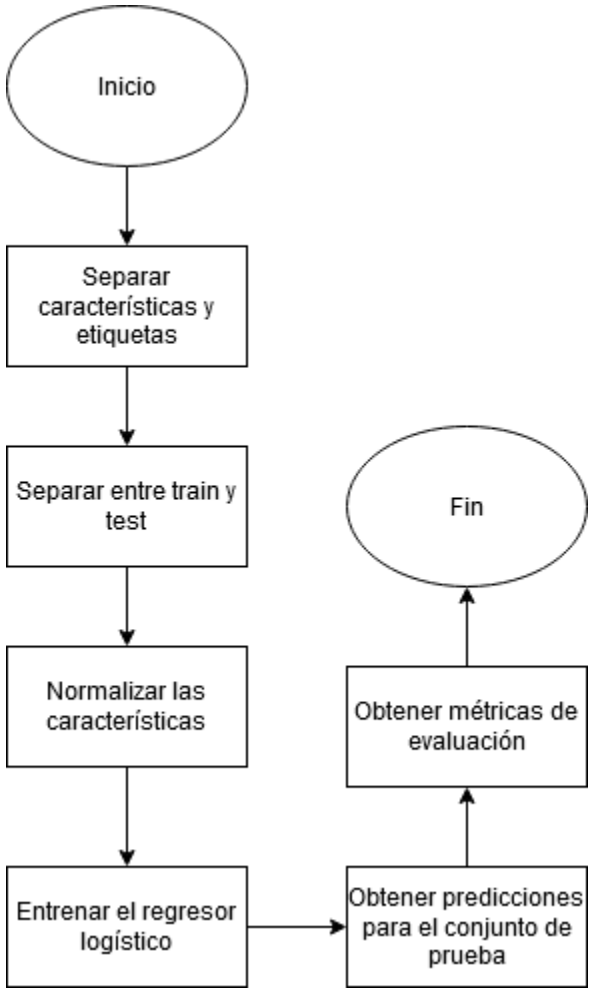


Diagrama de flujo 1. Implementación regresión logística.

```
matthews_corrcoef 0.8848935914888149
Accuracy 0.918918918918919
              precision    recall  f1-score   support

         1.0       1.00      1.00      1.00         3
         2.0       1.00      0.87      0.93        15
         3.0       0.71      0.83      0.77         6
         4.0       0.93      1.00      0.96        13

    accuracy                           0.92        37
   macro avg       0.91      0.93      0.92        37
weighted avg       0.93      0.92      0.92        37
```

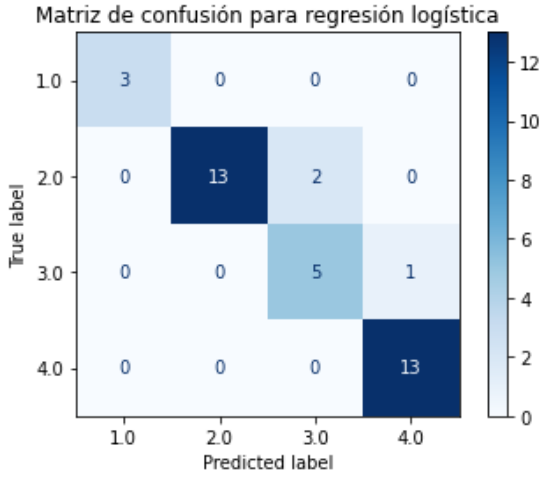


Figura 1. Evaluación modelo regresión logística

A partir de los resultados, se puede ver que coeficiente de correlación de Matthews obtenido fue 0.88, lo cual respresentaría un buen resultado que no indica un sobreentrenamiento. Con la matriz de confusión generada sobre los datos de test se puede ver que los errores se presentan sobre muestras de las clases 2 y 3 pero no en grandes cantidades.

2. Clasificación con Máquina de soporte vectorial

Una máquina de soporte vectorial (SVM) es un clasificador que se define por un hiperplano separador. Dados los datos de entrenamiento con sus respectivas etiquetas se halla un hiperplano óptimo para los datos en cuestión. Para este caso se clasifica en 4 clases diferentes y se realiza el mismo procedimiento con diferentes kernels para evaluar su comportamiento.

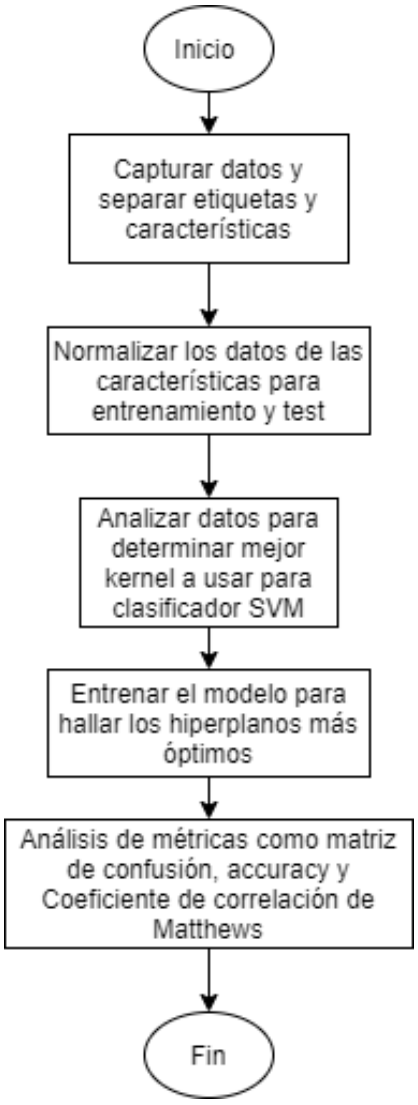
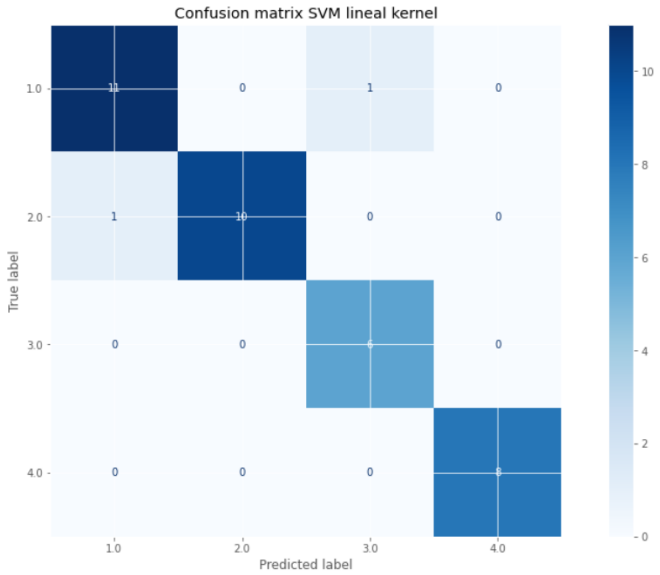


Diagrama de flujo 2. Solución implementada SVM.

2.1 Kernel lineal

Se utiliza si los datos son linealmente separables, es así que se obtienen resultados deseables con un accuracy de 0.95 y f1 score para cada clase muy cercano a 1.

	precision	recall	f1-score	support
1.0	0.92	0.92	0.92	12
2.0	1.00	0.91	0.95	11
3.0	0.86	1.00	0.92	6
4.0	1.00	1.00	1.00	8
accuracy			0.95	37
macro avg	0.94	0.96	0.95	37
weighted avg	0.95	0.95	0.95	37




```
Accuracy para entrenamiento: 1.0
Accuracy para test: 0.9459459459459459
El coeficiente de correlación de Matthews es: 0.927586
```

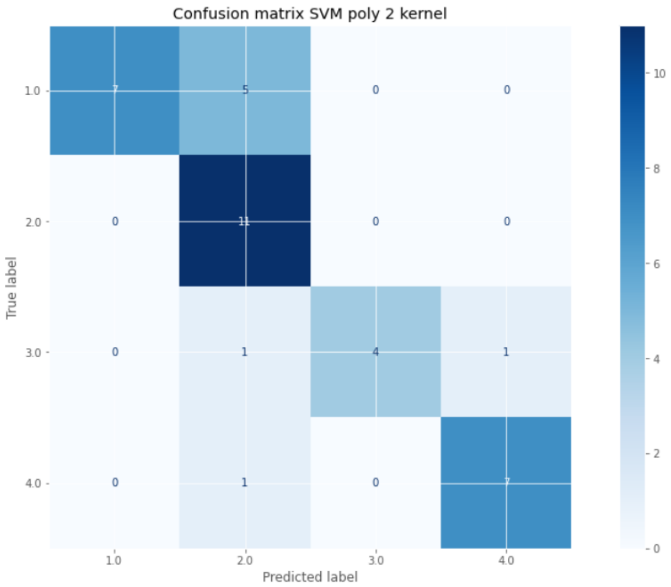
Figura 2. Evaluación modelo SVM kernel lineal

2.2 Kernel polinómico

Se da a conocer que el kernel polinómico es de mayor grado que el lineal y sirve para separar tipos de datos que no son linealmente separables, es así que tiene un menor grado de precisión.

2 grados

	precision	recall	f1-score	support
1.0	1.00	0.58	0.74	12
2.0	0.61	1.00	0.76	11
3.0	1.00	0.67	0.80	6
4.0	0.88	0.88	0.88	8
accuracy			0.78	37
macro avg	0.87	0.78	0.79	37
weighted avg	0.86	0.78	0.78	37

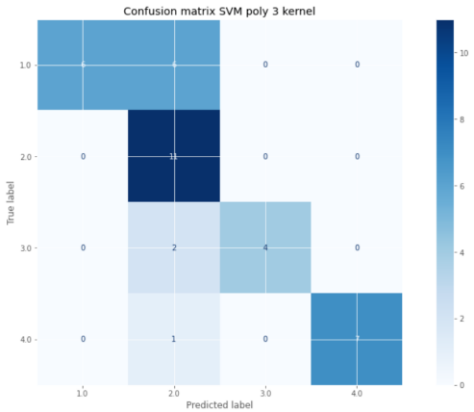



```
Accuracy para entrenamiento: 0.7738095238095238
Accuracy para test: 0.7837837837837838
El coeficiente de correlación de Matthews es: 0.733062
```

Figura 3. Evaluación modelo SVM kernel polinómico 2 grados

3 grados

	precision	recall	f1-score	support
1.0	1.00	0.50	0.67	12
2.0	0.55	1.00	0.71	11
3.0	1.00	0.67	0.80	6
4.0	1.00	0.88	0.93	8
accuracy			0.76	37
macro avg	0.89	0.76	0.78	37
weighted avg	0.87	0.76	0.76	37




```
Accuracy para entrenamiento: 0.6547619047619048
Accuracy para test: 0.7567567567567568
El coeficiente de correlación de Matthews es: 0.7112812498549702
```

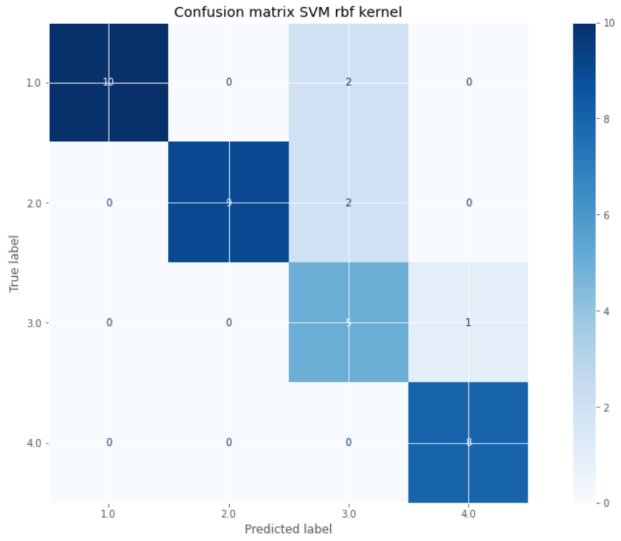
Figura 4. Evaluación modelo SVM kernel polinómico 3 grados

Se puede observar que los coeficientes de matthews para este caso son los que arrojan peor resultado de los hasta ahora analizado porque se está tomando como si los datos fueran no lineales cuando son lineales, complicando mucho más el cálculo de un modelo preciso.

2.3 Kernel rbf

Este es el kernel más utilizado por su facilidad de implementación y arroja un coeficiente de Matthews de 0.826.

	precision	recall	f1-score	support
1.0	1.00	0.83	0.91	12
2.0	1.00	0.82	0.90	11
3.0	0.56	0.83	0.67	6
4.0	0.89	1.00	0.94	8
accuracy			0.86	37
macro avg	0.86	0.87	0.85	37
weighted avg	0.90	0.86	0.87	37



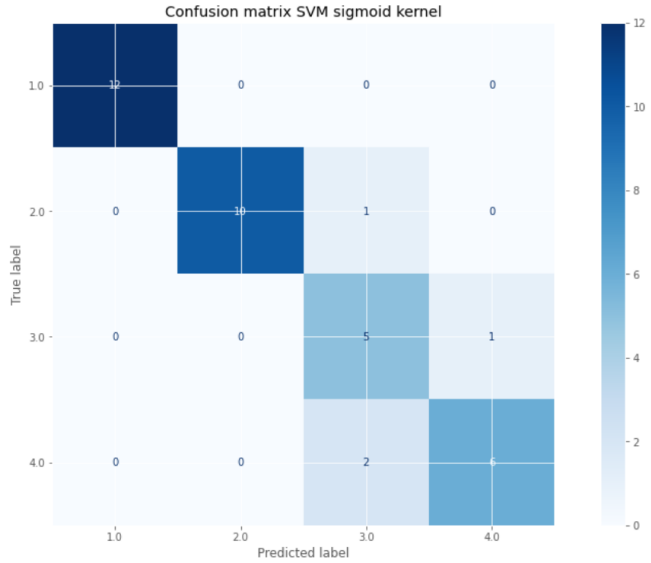

```
Accuracy para entrenamiento: 0.9880952380952381
Accuracy para test: 0.8648648648648649
El coeficiente de correlación de Matthews es: 0.8266495316704985
```

Figura 5. Evaluación modelo SVM kernel rbf

2.4 Kernel sigmoide

Tiene relación con la regresión logística dada su función característica sigmoide y se obtiene un coeficiente de Matthews de 0.856.

	precision	recall	f1-score	support
1.0	1.00	1.00	1.00	12
2.0	1.00	0.91	0.95	11
3.0	0.62	0.83	0.71	6
4.0	0.86	0.75	0.80	8
accuracy			0.89	37
macro avg	0.87	0.87	0.87	37
weighted avg	0.91	0.89	0.90	37




```
Accuracy para entrenamiento: 0.9642857142857143
Accuracy para test: 0.8918918918918919
El coeficiente de correlación de Matthews es: 0.8561575346506138
```

Figura 6. Evaluación modelo SVM kernel sigmoide

3. Clasificación con KNN

El método de clasificación KNN es un método supervisado, por lo que puede ser fácilmente aplicado al problema que se está tratando en este proyecto donde se tienen 4 posibles etiquetas para cada una de las imágenes analizadas. En general el método de KNN funciona de forma diferente al resto de clasificadores supervisados ya que no trabaja buscando una función que describa los datos sino que funciona con los propios datos y sus etiquetas. Este mide la distancia entre la muestra que se quiere clasificar y el resto de las muestras del dataset, así, dependiendo del k seleccionado se establece la etiqueta de la muestra. El k es la cantidad de muestras más cercanas que definen la clasificacion de la muestra en cuestion.

Los resultados se muestran a continuación, en este oportunidad el datset se partió en 0.7 train y 0.3 test, a partir de entrenar este clasificador con diferentes valores de K y al medir su accuracy se llegó a la conclusión de que el valor de k más apropiado es k = 2.

Diagrama de flujo 3. Solución implementada KNN

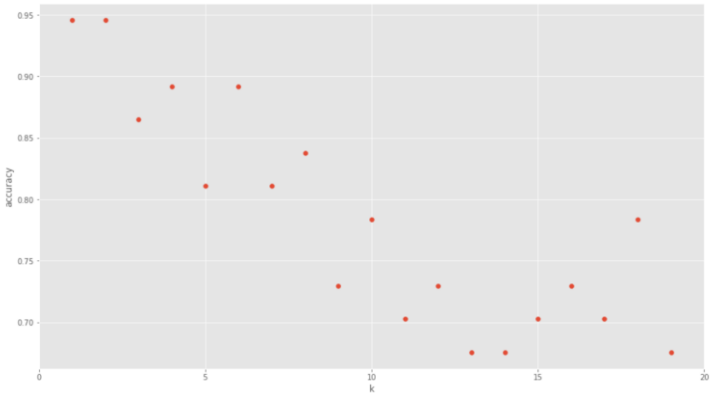


Figura 7. Mejor K para KNN

Posteriormente tras establecer K=2 en el clasificador se obtuvieron las siguientes métricas.

Accuracy of K-NN classifier on training set: 1.00

Accuracy of K-NN classifier on test set: 0.95

matthews_corrcoef: 0.9274372204904507

Accuracy: 0.9459459459459459

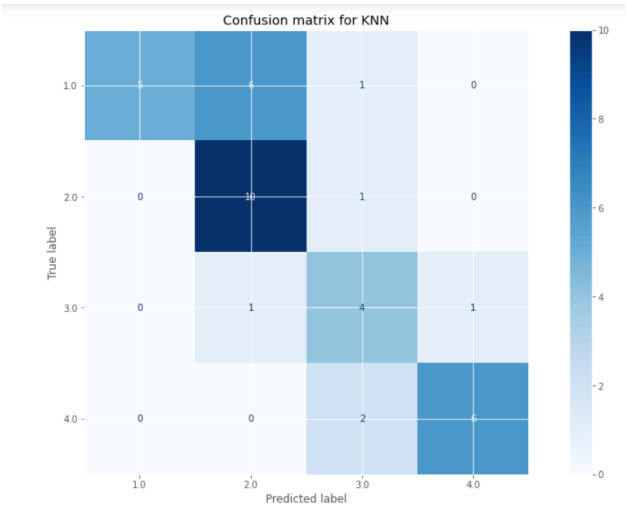


Figura 8. Evaluación modelo KNN

A partir de los datos previamente mencionados se observa que puede haber cierto sobreentrenamiento en el clasificador ya que el accuracy es más alto en el conjunto de entrenamiento que en el conjunto de test. Sin embargo, se observa que el clasificador tiene un buen desempeño a partir de su resultado del coeficiente de Matthews, que es una de las métricas más estrictas donde se obtiene un excelente resultado.

Conclusiones

1. Con respecto a la evaluación de estos clasificadores se marca una diferencia con el coeficiente de correlación de Matthews, donde el KNN fue el mejor de todos con 0.927, aún así, los demás clasificadores fueron bastante precisos ya que el peor caso fue el SVM con kernel polinómico de grado 3 donde se establece un valor de 0.71. Se concluye que KNN obtiene los mejores resultados debido a su funcionamiento, es decir, a diferenica de la regresión logística donde para realizar una clasificación es necesario establecer un umbral de decisión o las SVM donde se requiere generar un hiperplano de decisión, KNN clasifica a partir de la distribución de los datos del dataset, calculando la distancia de las K muestras mas cercanas a la muestra que se quiere clasificar. Si bien este proceso permite que el clasificador no dependa como tal de un umbral sino de la seleccion de un buen k, si hace que sea mas costoso computacionalmente. Si bien este proceso permite que el clasificador no dependa como tal de un umbral sino de la selección de un buen k, si hace que sea más costoso computacionalmente.

2. La matriz de confusión y el accuracy también fueron de gran ayuda para evaluar el modelo, junto con F1 score, precision and recall, con todos estos resultados se obtiene un modelo confiable para todos los clasificadores y se relaciona que este problema es de índole lineal ya que con el kernel lineal en SVM se estableció un mejor resultado a diferencia de los kernels polinómicos con un coeficiente de correlación de Matthews de 0.92.

Referencias

[1] Y. Isaienkov, K. Isaienkov y A. Zhuravel, «LEGO Minifigures Classification,» [En línea]. Available: https://www.kaggle.com/ihelon/lego-minifigures-classification. [Último acceso: 2 12 2020].