Universidad Autónoma de Yucatán

Ingeniería en Mecatrónica

Aprendizaje Automático

Edson G. Estrada López

Proyecto #2

Clasificación multiclase

Alumnos:

Burgos Paz Emir Eduardo

Domínguez Mendoza Jonathan Ariel

Marín Pech Adiel Sait

Fecha:

21 de mayo de 2021

Contenido

Objetivos

Objetivo general

- Aplicar diversas variantes de algoritmos de clasificación multiclase en la solución de un problema.

Objetivos específicos

- Implementar los siguientes métodos de regresión logística OvA, OvO y Softmax
- Realizar una comparación entre los métodos de regresión logística y la clasificación SGD para el dataset del $FASHION - MINST$
- Realizar una comparación entre los métodos de regresión logística y la regresión lineal para el dataset de los vinos.

Contenido

Introducción

En este proyecto se van a implementar y conocer diferentes estilos de regresión logística, además se van a comparar esta metodología en contra de otras implementaciones de regresión basada en librerías, como la producida por la clasificación SGD y la clasificación linear.

Para ello primeros implementaremos tres modelos de regresión logística, $One\ vs\ all, One\ vs\ One\ y\ Softmax$ con el dataset de Iris, luego implementaremos una comparación del modelo $LogisticRegression()$ y lo comparemos con el modelo de $SGDClasifier()$ en el dataset de $FASHION - MINST$ y finalmente implemntaremos una comparación entre los modelos de regresión logística y el modelo lineal redondeado para el dataset de los vinos.

Con ello podemos observar como este tipo de regresión puede trabajar en diferentes tipos de dataset, con diferentes características y respuestas.

Tabla de implementaciones

Tabla 1 Tabla de implementaciones

Se pidió…	Se realizo…
Implementación de OvA	Se implementó el Ova
Implementación de OvO	Se implementó el OvO
implementación de Softmax	Se implementó el Softmax
Comparación de $LogisticRegression()$ y $SGDClasifier()$ en $FASHION - MINST$	Se realizo la comparación
Comparación de la regresión lineal y la logística para el dataset de vinos	Se realizo la comparación

Marco teórico

Los siguientes métodos que vamos a explorar están basados en la implementación de la regresión logística que esta basada en las siguientes funciones.

La primera función que debemos de conocer es la idea de la binarización, la cual se puede representada gráficamente por un sigmoide, esta función tiene como respuesta la probabilidad de que lo observado sea una categoría del dataset.

$$\pi(X) = \frac{1}{1 + \exp\,(-X\,\beta)}$$

Igualmente la regresión logística tiene una ecuación propia de costo, la cual es:

$$J(\theta) = \sum_{i=0}^{n} [y_i X_i \beta - \log\,(1 + \exp(X_i \beta))]$$

$$One\ vs\ all$$

$One\ vs\ all$ (OvA) es un tipo de clasificación binaria basada en la cantidad de clases de un dataset. OvA produce la misma cantidad de modelos aprendidos con el número de clases. Así, si el número de clases es 10, el número de modelos aprendidos es también 10. (Abdul Raziff, Sulaiman, & Perumal, 2017)

En este método, cada clase se empareja con las restantes. Sin embargo, este método tiene la posibilidad de sufrir un desequilibrio en la clasificación si el número de clases es elevado. Esto ocurre porque el número del conjunto de datos de datos de entrenamiento difiere enormemente para cada modelo de aprendizaje. (Abdul Raziff, Sulaiman, & Perumal, 2017)

En la fase de validación, se presenta un patrón a cada uno de los clasificadores binarios y, a continuación, el clasificador que da una salida positiva indica la clase de salida. En muchos casos, la salida positiva no es única y se requieren algunas técnicas de desempate. El enfoque más común utiliza la confianza de los clasificadores para decidir la salida final, prediciendo la clase del clasificador con mayor confianza. (Galar, Fernández, Barrenechea, Bustince, & Herrera, 2011)

Estrategias

Galar, Fernández, Barrenechea, Bustince, & Herrera (2011) hacen mencion que una de las estratrgias para decidir la validez de la predicción en OvA, es tomar el clasificador con la mayor respuesta positiva, donde:

$$Class = argmax(x_{i=1,\dots,m}\, r_{ij})$$

$One\ vs\ One$

$One\ vs\ One$ es una clasificación por pares para un mapeo multiclase en el que todos los conjuntos de datos que pertenecen a una clase determinada se emparejan con otros conjuntos de datos de otra clase para el aprendizaje. (Abdul Raziff, Sulaiman, & Perumal, 2017)

El número de modelos generados depende del número de clases:

$$\frac{n(n-1)}{2}$$

donde n es el número de clases.

Si el n, equivale a 10, entonces el total del modelo aprendido es de 45 según la fórmula mencionada.

En la fase de validación, se presenta un patrón a cada uno de los clasificadores binarios. La salida de un clasificador dada por r_{ij} en [0,1] es la confianza del clasificador binario que discrimina las clases i y j a favor de la primera. La confianza del clasificador para la segunda se calcula mediante $r_{ji} = 1 - r_{ij}$ si el clasificador no la proporciona (la clase con mayor confianza es la clase de salida de un clasificador). (Galar, Fernández, Barrenechea, Bustince, & Herrera, 2011)

Galar, Fernández, Barrenechea, Bustince, & Herrera (2011) hacen mencion que una de las estratrgias para decidir la validez de la predicción en OvO, es que cada clasificador binario de un voto a la clase predicha. Se cuentan los votos recibidos por cada clase y se predice la clase con mayor número de votos

$$Class = argmax\left(x_{i=1,\dots,m}\right) \sum_{1 \le j\ !=i \le m} s_{ij}$$

Liu, et al., (2021) exponen las sigueintes ventajas y desventjas al momento de usar los metodos de clasificacion OvA y OvO:

- Basándose en los métodos OVA y OVO, la precisión de la clasificación mejora significativamente en comparación con un algoritmo convencional.
- En todos los tipos de estrategias de decisión, la estrategia de votación basada en los resultados de la clasificación obtiene la precisión óptima, y la estrategia de votación basada en cierto coeficiente de reglas obtiene la precisión subóptima
- El método de clasificación binaria basado en OVO requiere construir más clasificadores que el método de clasificación binaria basado en OVA y, por tanto, suele costar más tiempo de cálculo.

- El número de atributos y reglas obtenidos por el método de supresión de interferencias entre clases OVO es significativamente menor que el del algoritmo convencional
- El método OVO obtiene la mejor precisión de clasificación entre OVO, OVA y algoritmos convencionales

$Softmax$

La regresión Softmax es una generalización de la regresión logística que podemos utilizar para la clasificación multiclase (bajo el supuesto de que las clases son mutuamente excluyentes). En cambio, utilizamos el modelo de regresión logística (estándar) en tareas de clasificación binaria. (Raschka, 2015)

La Regresión Softmax fue diseñada para su uso con múltiples clases que son mutuamente excluyentes entre sí. No evalúa la pertenencia a una clase independientemente de otras clases. (Wolfe, Jin, Bahr, & Holzer, 2017)

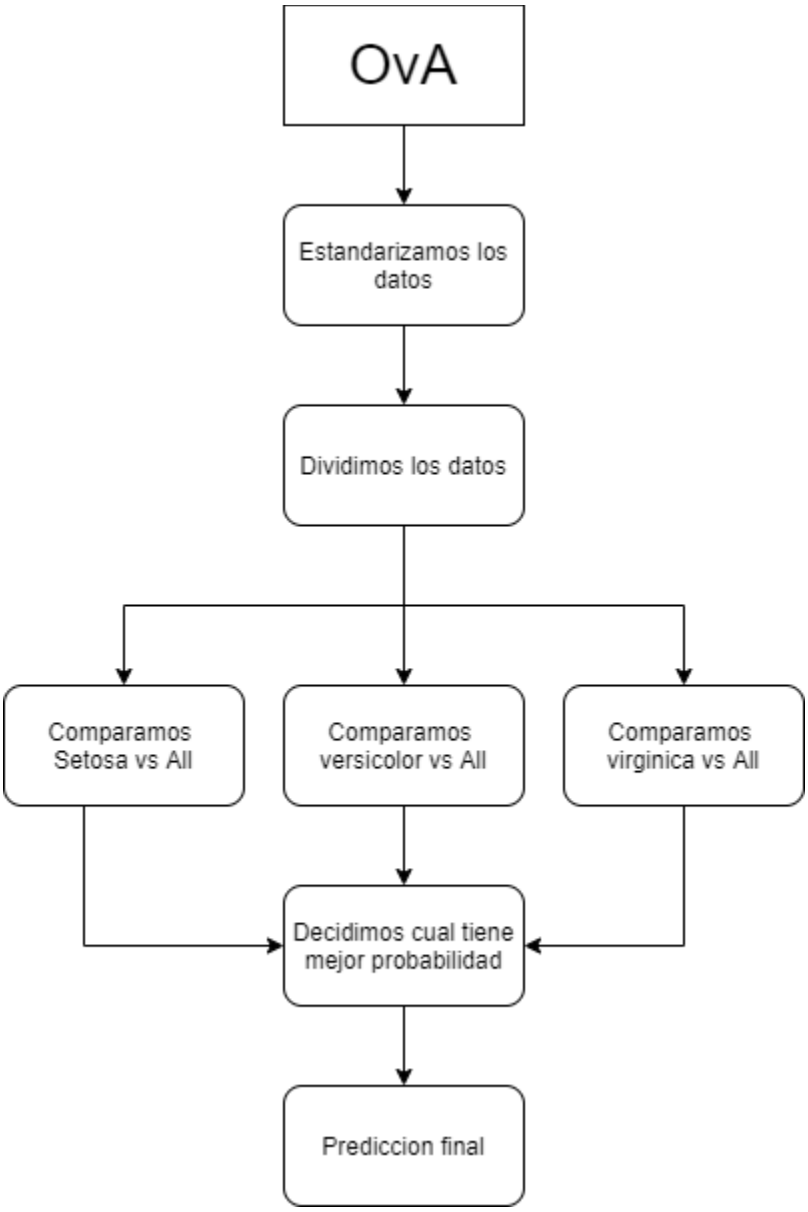
Para cada dato, la regresión Softmax calcula los valores de activación para cada clase de salida. Normaliza los valores para obtener un conjunto de conjunto de probabilidades que suman 1. La clase con la mayor probabilidad gana, y el elemento de datos se asigna a esa clase. Un escenario ideal es que una clase tenga una probabilidad cercana a 1 y las otras se acercan a 0. (Wolfe, Jin, Bahr, & Holzer, 2017)

Zhang, Lipton, Li, & Smola (2020) menciona que:

- La operación Softmax toma un vector y lo convierte en probabilidades.
- La regresión Softmax se aplica a los problemas de clasificación. Utiliza la distribución de probabilidad de la clase de salida en la operación Softmax.
- La entropía cruzada es una buena medida de la diferencia entre dos distribuciones de probabilidad. Mide el número de bits necesarios para codificar los datos dado nuestro modelo.

Descripción de cada implementación

One vs all



El método de OvA se basa en la comparación de una clase contra las demás clases, por ello en este caso debemos de hacer tres comparaciones, que representan a tres modelos. En base a estos tres modelos podemos hacer una predicción para conseguir las probabilidades y escoger el valor que tenga una mayor probabilidad de ser la clase.

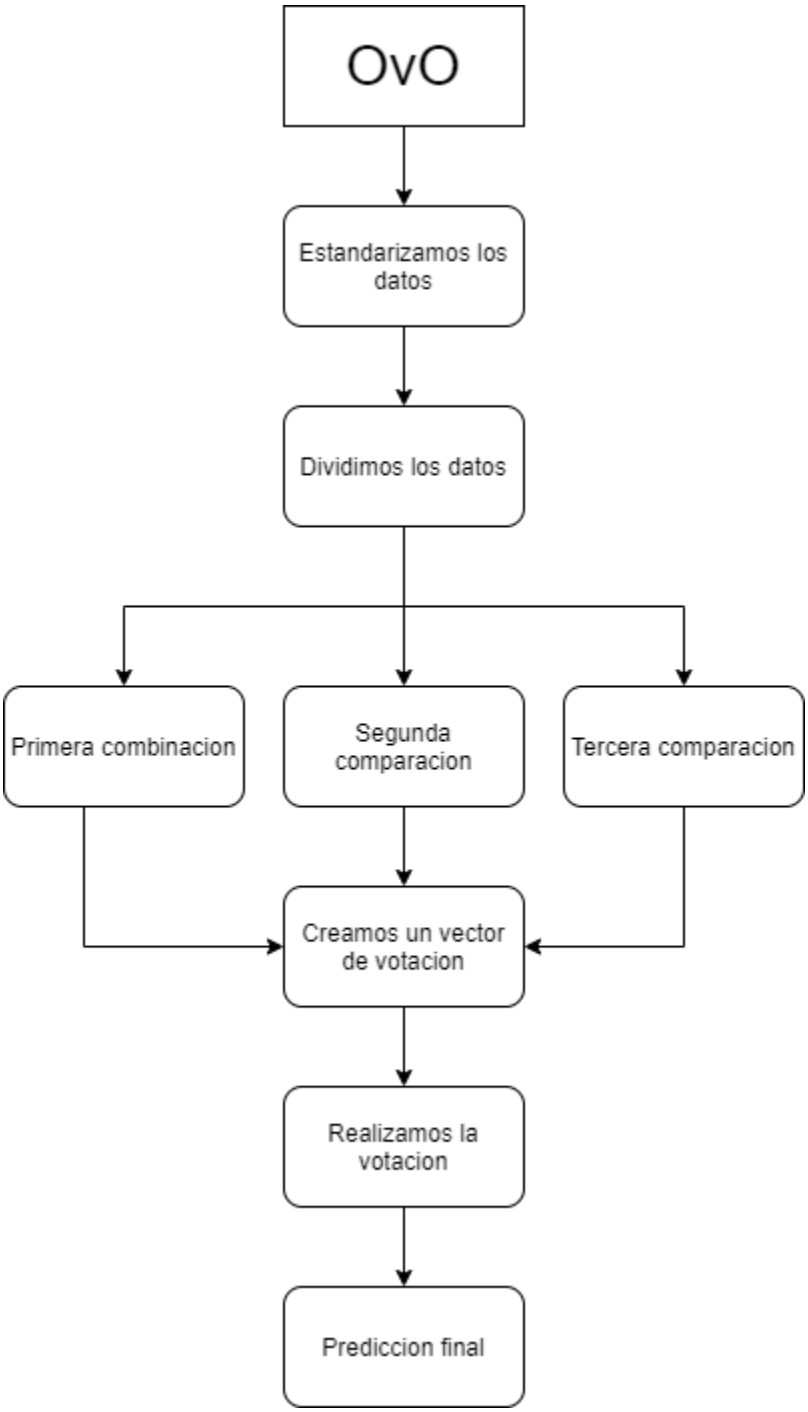
En la imagen siguiente podemos observar de manera grafica las diferentes respuestas de esta metodología.

```
array([[0.  , 0.59, 0.93, 2.  , 2.  ],
       [1.  , 0.27, 0.  , 0.  , 0.  ],
       [0.  , 0.37, 0.  , 1.  , 1.  ],
       [0.  , 0.78, 0.  , 1.  , 1.  ],
       [0.  , 0.25, 0.  , 1.  , 1.  ],
       [0.  , 0.73, 0.14, 1.  , 1.  ],
       [0.  , 0.92, 0.  , 1.  , 1.  ],
       [0.  , 0.75, 0.  , 1.  , 1.  ],
       [0.  , 0.46, 0.  , 1.  , 1.  ],
       [0.  , 0.45, 0.98, 2.  , 2.  ],
       [1.  , 0.02, 0.  , 0.  , 0.  ],
       [0.  , 0.31, 1.  , 2.  , 2.  ],
       [0.  , 0.65, 0.  , 1.  , 1.  ],
       [0.  , 0.56, 0.99, 2.  , 2.  ],
       [0.  , 0.37, 0.  , 1.  , 1.  ],
       [1.  , 0.23, 0.  , 0.  , 0.  ],
       [0.  , 0.07, 1.  , 2.  , 2.  ],
       [1.  , 0.31, 0.  , 0.  , 0.  ],
       [0.  , 0.67, 1.  , 2.  , 2.  ],
       [0.  , 0.34, 1.  , 2.  , 2.  ],
       [1.  , 0.03, 0.  , 0.  , 0.  ],
       [0.  , 0.39, 0.02, 1.  , 1.  ],
       [0.  , 0.2 , 1.  , 2.  , 2.  ],
       [0.  , 0.63, 1.  , 2.  , 2.  ],
       [0.  , 0.76, 0.71, 1.  , 2.  ],
       [1.  , 0.11, 0.  , 0.  , 0.  ],
       [1.  , 0.12, 0.  , 0.  , 0.  ],
       [1.  , 0.29, 0.  , 0.  , 0.  ],
       [0.  , 0.84, 0.  , 1.  , 1.  ],
       [0.  , 0.47, 0.  , 1.  , 1.  ],
       [0.  , 0.27, 0.  , 1.  , 1.  ],
       [0.  , 0.77, 0.  , 1.  , 1.  ],
       [0.  , 0.38, 0.99, 2.  , 2.  ],
       [0.  , 0.9 , 0.79, 1.  , 2.  ],
       [0.  , 0.57, 0.01, 1.  , 1.  ],
       [1.  , 0.29, 0.  , 0.  , 0.  ],
       [1.  , 0.06, 0.  , 0.  , 0.  ],
       [1.  , 0.17, 0.  , 0.  , 0.  ]])
```

Esta matriz representa los valores de probabilidad dependiendo de cada posición de testeo, la cuarta columna son los valores de predicción que obtenemos cuando realizamos nuestra predicción y la quinta columna son los valores reales.

Igualmente obtenemos una matriz de confusión, para nuestros datos.

One vs One

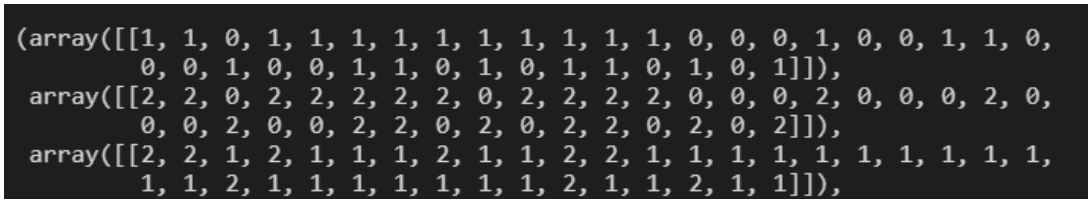


Al igual que el OvO en este método se deben de hacer una comparación entre clases, solo que estas comparaciones van a ser de manera individual, como tenemos un dataset de 3 clases, se debe de aplicar la siguiente formula:

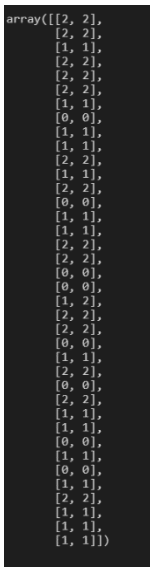
$$\frac{n(n-1)}{2} = \frac{3 * 2}{2} = 3$$

Esta función nos menciona que debemos de hacer tres comparaciones, las cuales van a ser entre setos vs versicolor, setos vs virginica y versicolor vs virginica. Estos modelos realizan una predicción de probabilidades se clasifican con respecto a sus clases.

La imagen siguiente representa a los vectores de predicción de cada uno de las comparaciones:



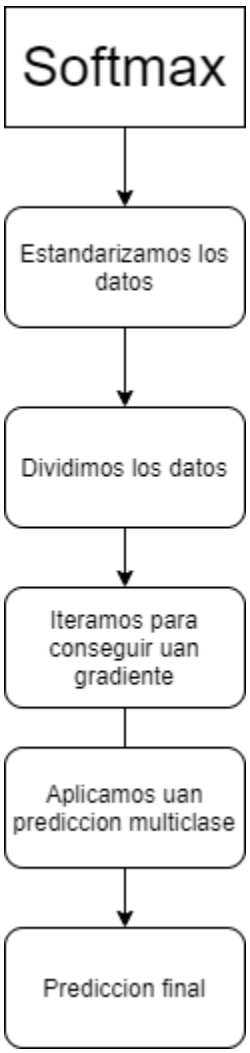
Con el cual al momento de realizar las votaciones obtenemos un vector de predicción final, como el que se ve en la siguiente imagen:



Este vector en la primera columna tiene a los valores de las votaciones y en la segunda columna los valores de las predicciones. Finalmente obtenemos una matriz de confusión para nuestro sistema:


setosa
versicolor
virginica
8
0
0
0
15
0
0
1
14
setosa_pred
versicolor_pred
virginica_pred
14
12
10
8
6
4
2
0

$Softmax$


Softmax
Estandarizamos los datos
Dividimos los datos
Iteramos para conseguir uan gradiente
Aplicamos uan prediccion multiclase
Prediccion final

La regresión de Softmax esta conducida por las siguientes formulas:

Softmax

$$\phi_{softmax}(z^i) = \frac{\exp^{z^{(i)}}}{\sum_{j=0}^{k} \exp^{z_k^{(i)}}}$$

Costo

$$J(W) = \frac{1}{n} \sum_{i=0}^{n} H(T_i, O_i)$$

Cross entropy

$$H(T_i, O_i) = -\sum_{m} T_i * log(O_i)$$

Gradiente

$$\nabla w_j J(W) = -\frac{1}{n} \sum_{i=0}^{n} [x^{(i)}(T_i - O_i)]$$

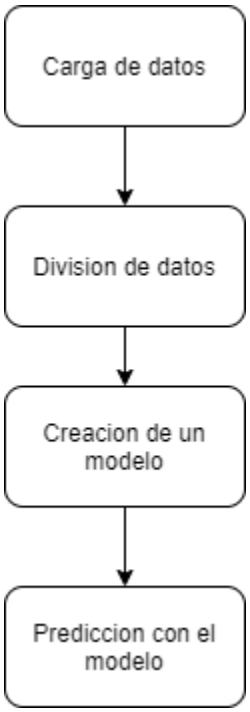
Las cuales al ser aplicadas al momento de generar nuestro ciclado de iteraciones pueden trabajar con un dataset multiclase, por ello este modelo es el más rápido de programar y entender ya que solo se necesita crear un modelo general para todo el sistema. Y al momento de aplicar la predicción de nuestro sistema, podemos conseguir de manera directa una predicción multiclase, como la observada en la siguiente imagen.

```
array([1, 0, 2, 0, 2, 0, 2, 1, 1, 2, 0, 0, 1, 2, 1, 1, 2, 2, 0, 2, 0, 1,
       1, 1, 2, 0, 2, 0, 1, 2, 2, 1, 1, 0, 1, 2, 2, 1], dtype=int64)
```

Finalmente podemos observar en matriz de confusión, los resultados que obtuvimos en una de nuestras iteraciones del sistema.

$$LogisticRegression()\ vs\ SGDclassifier()$$

Para la comparación entre estos dos métodos se utilizaron las implementaciones que ya existen en sklearn, y se siguió el modelo estándar de creación de un modelo, donde se sigue el siguiente diagrama:



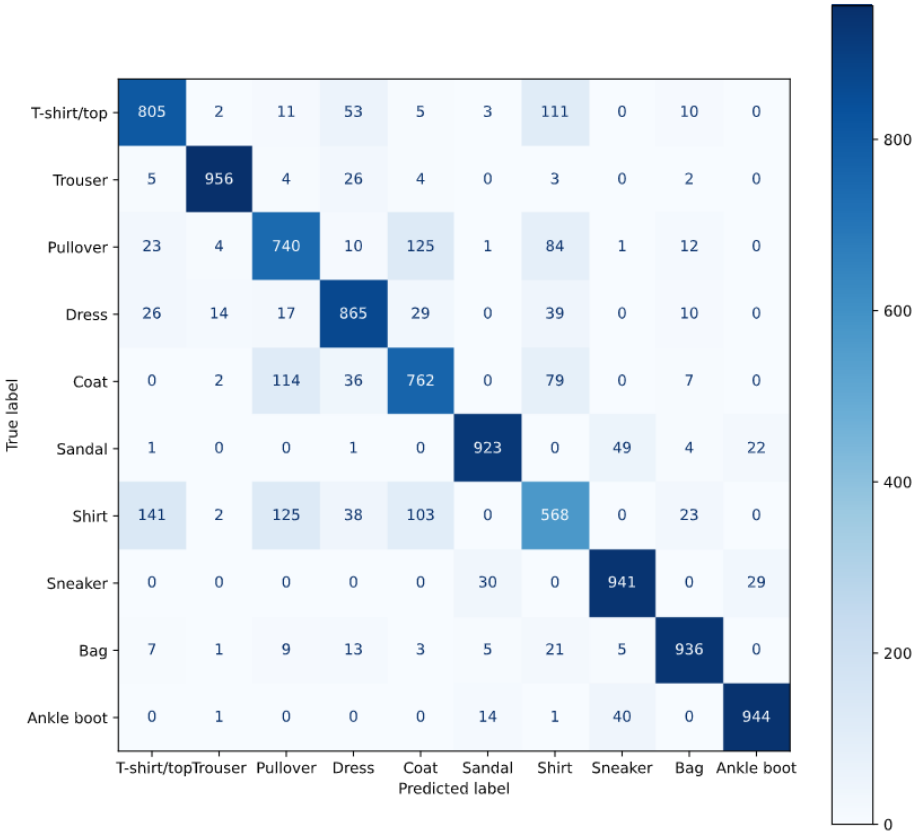
Solo que en este caso se tuvo que parametrizar los datos en los modelos de regresión logística, y se debe de parametrizar los modelos del clasificador SGD.

Para ello se usaron los siguientes parámetros:

$$lgr = LogisticRegression(solver = 'saga', random\_state = 42)$$
$$sgd = SGDClassifier(loss = 'log', alpha = 0.01, max\_iter = 3000)$$

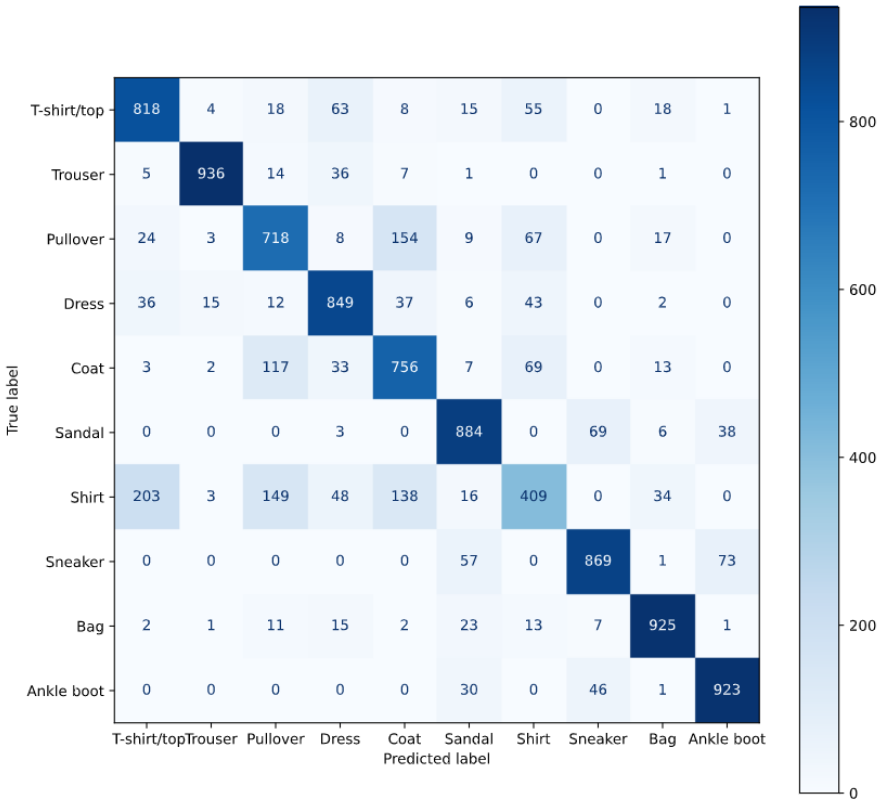
Finalmente los dos modelos tuvieron altos niveles de precisión, la principal de diferencia que hubo entre ellos fue el tiempo de procesado.

Resultados de la regresión logística



En general la precios del sistema fue 0.84 la cual es igual a la precisión que tienen estos modelos en el benchmark de la página de GitHub del dataset.

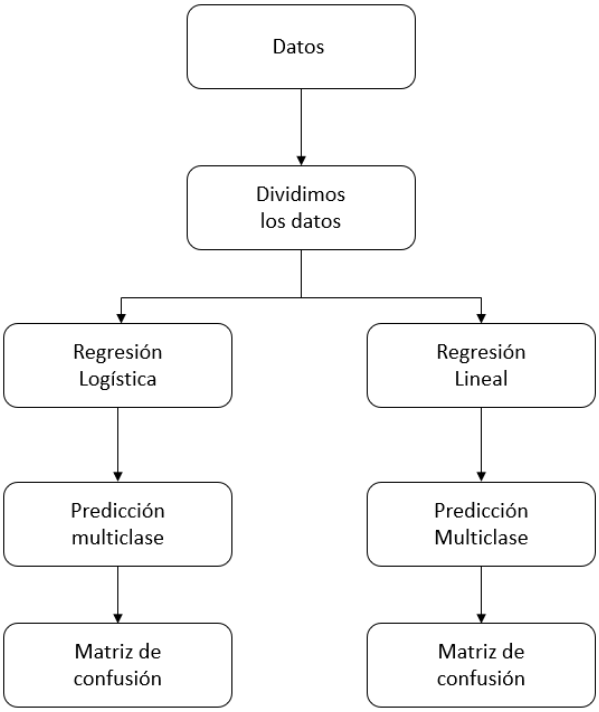
Resultados de la regresión SGD



La precisión de este sistema fue 0.82 el cual es parecido a la precisión que tienen los modelos de este tipo en el benchmark del GitHub del dataset.

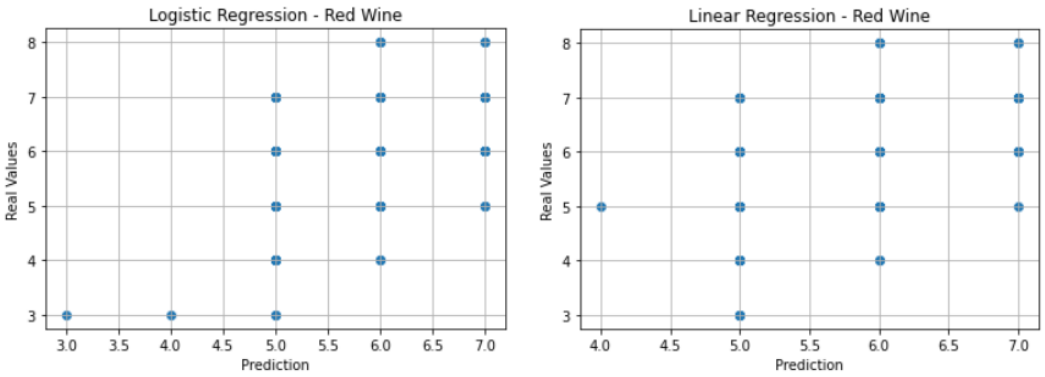
$$LogisticRegression()\ vs\ LinearRegression()$$

Para comparar los modelos de la regresión lineal y la regresión logística utilizamos las librerías proporcionadas por Phyton. En este caso utilizamos las librerías Seaborn, Pandas, Matplotlib y Sklearn.

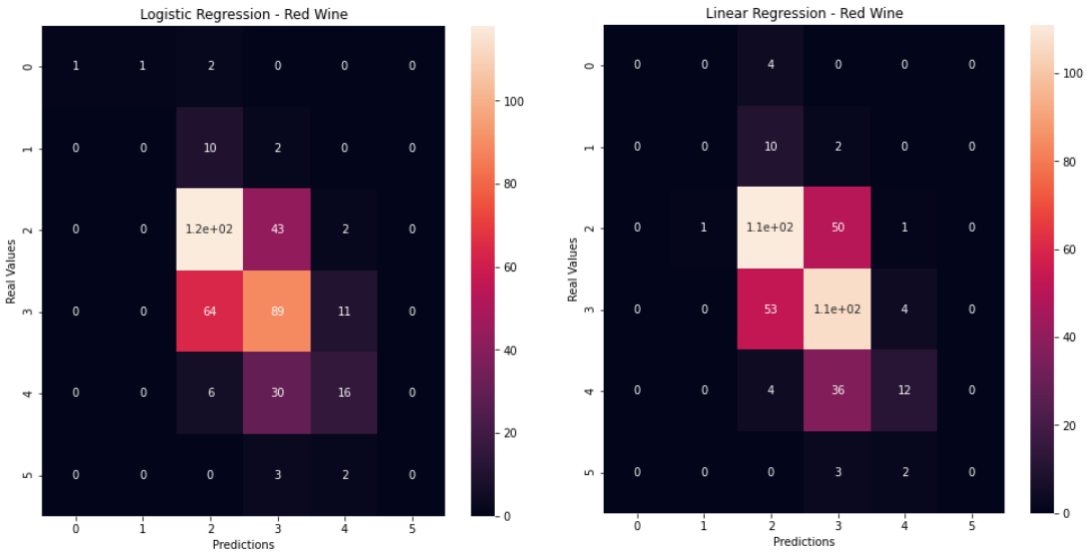


Una vez que tenemos creados los modelos, podemos comparar el resultado de la regresión logística contra el de la regresión lineal mediante una gráfica de predicciones contra valores reales.

Vino rojo



Podemos observar que ambos modelos pudieron predecir algunos valores deforma correcta y que también cuentan con una dispersión de predicciones comparados a los valores reales.



Podemos comparar las matrices de confusión y podemos observar que este *dataset* no contenía muchos valores para las calidades exteriores (3, 4 y 8). Por lo que la predicción de estos valores se espera que sea baja comparada las clases de mayor concentración.

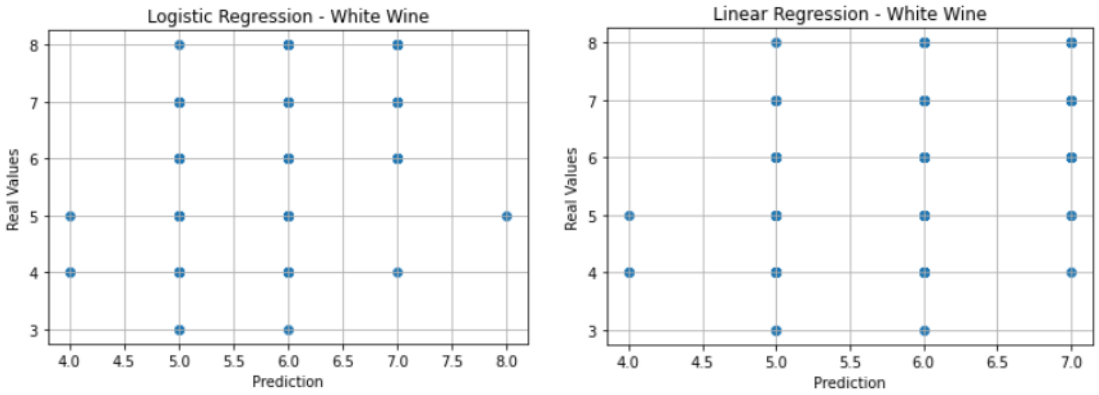
La mayoría de las predicciones se concentran en los valores intermedios (5 y 6) y podemos ver que las predicciones se concentran en la diagonal para estos valores, lo que significa que, por lo menos para estos valores, el modelo es capaz de predecir con cierta fiabilidad.

En la siguiente tabla nos podemos dar una idea de la precisión y el *recall* del modelo, se uso el parámetro '*weighted*' para poder comparar de una manera correcta los modelos debido a la gran variación de la cantidad de datos.

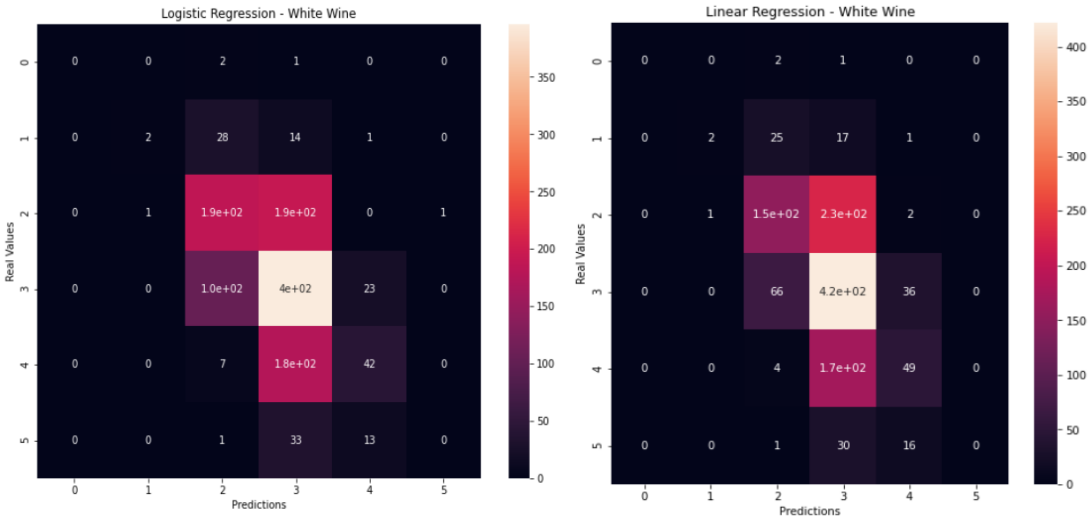
Vino Rojo	Precisión	Recall
Logístico	0.536	0.560
Lineal	0.552	0.575

Los modelos parecen muy similares dados los resultados, pero cabe recalcar que el modelo de regresión lineal redondeado no fue capaz de predecir ninguna muestra de los valores exteriores, mientras que el logístico fue capaz de dar un resultado para las muestras más alejadas. Faltaría una mayor cantidad de datos para poder comparar mejor los modelos en los casos exteriores.

Vino blanco



En este *dataset* podemos encontrar también que el modelo tiene algo de dispersión en sus predicciones.



Por la cantidad de datos se puede notar que el modelo no fue capaz de producir valores correctos para las calidades extremas (3 y 8). Sin embargo, también podemos ver que hubo una concentración de los datos en la diagonal, para los valores intermedios, igual que con el *dataset* del vino rojo.

Vino Blanco	Precisión	Recall
Logístico	0.507	0.511
Lineal	0.509	0.511

En este modelo se puede observar que tanto la precisión como el recall son equivalentes para ambos modelos. Lo que quiere decir que realmente no hay casi diferencia en usar un modelo sobre otro.

Esto no necesariamente significa que los modelos sean incorrectos, simplemente que por la cantidad de datos, tal vez no fueron la mejor opción, o los mismos *datasets* fueron relativamente bajos en datos para el modelo propuesto en el sentido que podrían estar mejor equilibrados.

Listado de bugs conocidos

En la comparación de la regresión logística contra la regresión lineal, al realizar el cálculo de la precisión y el recall para los valores se puede observar un aviso; esto sucede debido a que las predicciones de los valores no contienen algunas clases debido a la baja población de estas mismas.

```
C:\Users\Jonathan\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1245: UndefinedMetricWarning: Precision is ill
-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

Dependiendo en la forma como cargues los datos el sistema puede que no detecte los valores adecuados, pues si se cargan los datos, de manera no estandarizada y no normalizada, la precisión disminuye y el tiempo de procesado aumenta.

Conclusión

La metodología de la regresión logística nos permite trabajar con dataset que tengan más de dos salidas, de tal manera que los datos que estamos prediciendo sean más exactos o tengo mayor sentido, pues la regresión lineal, aunque nos permite identificar de manera muy clara la tendencia de un dataset, no nos devuelve los valores reales de la predicción. A comparación la regresión logística nos permite conocer de manera exacta que clase se predice en base a una serie de probabilidades.

Por lo tanto entender cómo funcionan estas metodologías desde cero y cómo podemos trabajar en ellas con librerías de alto nivel, nos permite conocer de manera más específica como podríamos trabajar con diferentes tipo de datos de entrada y salida.

Comentarios individuales

Emir Eduardo Burgos Paz

Por mi parte al aplicar los métodos de regresión logística y tener que investigar sobre ellos, pude entender algunas dudas que tuve después de la clase. Además, la realización de este proyecto me permitió conocer como la regresión logística funciona, pues nunca había escuchado acerca de este tipo de regresión antes de la clase.

Adiel

Con la realización del proyecto pude entender mejor como se usa la regresión logística y como funciona a la hora de clasificar problemas multiclase, así como diferenciar las características de los diferentes modelos de algoritmos one vs all, one vs one y softmax.

También al utilizar las funciones de biblioteca para comparar el Logistic Regression con el SGD Classifier fue más sencillo ver como funcionaban estos dos modelos.

Jonathan

En la realización de este proyecto me familiaricé con el uso de la regresión logística y como utilizarla para poder obtener modelos multiclase a través de la comparación de modelos uno a uno, uno contra todos o una comparación de probabilidades a través del algoritmo softmax.

También fue muy útil implementar su uso con algoritmos de la biblioteca sklearn y obtener una comparación con resultados de distintos algoritmos. Aunque el resultado de la comparación no fue el esperado contra el algoritmo de regresión lineal redondeado (Dataset de vinos), ayudo a identificar que el modelo necesita estar mejor equilibrado para poder desempeñarse de una mejor forma.

Bibliografía

Abdul Raziff, A. R., Sulaiman, M., & Perumal, T. (10 de 2017). Single classifier, OvO, OvA and RCC multiclass classification method in handheld based smartphone gait identification. *AIP Conference Proceedings, 1891*, 020009. doi:10.1063/1.5005342

Galar, M., Fernández, A., Barrenechea, E., Bustince, H., & Herrera, F. (2011). An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. *Pattern Recognition, 44*, 1761-1776. doi:https://doi.org/10.1016/j.patcog.2011.01.017

Liu, J., Bai, M., Jiang, N., Cheng, R., Li, X., Wang, Y., & Yu, D. (1 de 2021). Interclass Interference Suppression in Multi-Class Problems. *Applied Sciences, 11*. doi:10.3390/app11010450

Raschka, S. (2015). *Python Machine Learning.* Packt Publishing - ebooks Account. Obtenido de http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/1783555130

Wolfe, J., Jin, X., Bahr, T., & Holzer, N. (5 de 2017). APPLICATION OF SOFTMAX REGRESSION AND ITS VALIDATION FOR SPECTRAL-BASED LAND COVER MAPPING. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLII-1/W1*, 455-459. doi:10.5194/isprs-archives-XLII-1-W1-455-2017

Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2020). *Dive into Deep Learning.*