Analisis y Reporte Sobre el Desempeño del Modelo Desarrollado

TC3006 Concentración de Inteligencia Artificial

Docente del Módulo: Cesar Javier Guerra
Alumno: José Pablo Cruz Ramos
Matrícula: A01138740

17 de Septiembre del 2022

Para este reporte se ha seleccionado analizar el modelo que se creó en la segunda parte del módulo, el cual fue el modelo de regresión logística con el uso de un framework, en este caso SciKit Learn.

Separación y evaluación del modelo con un conjunto de prueba y un conjunto de validación (Train/Test/Validation).

Para la creación de un modelo de regresión logística, debemos utilizar un set de datos que nos permita crear dos muestras, una siendo la prueba y otra para el entrenamiento del modelo como tal.

Antes de construir el modelo debemos asegurarnos de que los datos con los que estamos trabajando sean adecuados y que no presenten anomalías como datos duplicados, nulos o incluso outliers.

En este caso tenemos identificado nuestra variable objetivo como la especie de las flores, la cual viene en la columna "specie", en este caso debido a que la columna presenta ser una variable categórica, utilizaremos una técnica de variable dummy para transformarla en una variable numérica cuantitativa. Para esto utilizamos el módulo de scikit learn llamado "Label Encoder" capaz de transformar nuestros datos a enumeraciones de variables categóricas. En este caso obtenemos la enumeración de 0 a 2, debido a que solo existen 3 especies en nuestro dataset.

```
   Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Species
0   1            5.1           3.5            1.4           0.2  Iris-setosa
1   2            4.9           3.0            1.4           0.2  Iris-setosa
2   3            4.7           3.2            1.3           0.2  Iris-setosa
3   4            4.6           3.1            1.5           0.2  Iris-setosa
4   5            5.0           3.6            1.4           0.2  Iris-setosa

Limpieza del data frame de IRIS

Valores nulos encontrados: 0
Valores duplicados encontrados: 0

Pasando a variable dummy la especia de las plantas

   Id  SepalLengthCm  SepalWidthCm  PetalLengthCm  PetalWidthCm      Species  \
0   1            5.1           3.5            1.4           0.2  Iris-setosa
1   2            4.9           3.0            1.4           0.2  Iris-setosa
2   3            4.7           3.2            1.3           0.2  Iris-setosa
3   4            4.6           3.1            1.5           0.2  Iris-setosa
4   5            5.0           3.6            1.4           0.2  Iris-setosa

   species
0        0
1        0
2        0
3        0
4        0
```

Ilustración de la terminal mostrando la variable dummy generada y la inspección de los datos con anomalía

Ahora que nuestros datos están listos, podemos proceder a la generación de los grupos de entrenamiento y prueba del dataset para probarlos con el modelo a generar.

Se dio uso de la función de train_test_split la cual se encarga de dividir el dataset en uso en dos agrupaciones, una para el entrenamiento de los datos y otra para la prueba, esto para ambas variables, el conjunto de x y la variable objetivo y. En esta misma función se puede cambiar el grado de aleatoriedad y el tamaño de la muestra de prueba. En este caso hemos decidido usar un 33% para la prueba y un 67% para el entrenamiento.

Se procede a generar el modelo con la librería de scikit learn, y utilizamos los grupos de entrenamiento para entrenar nuestro modelo, posterior a este se le indica al modelo entrenado que genere la predicción de la variable de “Species” con base en las demás variables con la función de predict().

Se utilizaron las métricas de matriz de confusión, accuracy score y el reporte de clasificación para medir la precisión del modelo y se obtienen los siguientes resultados.

```
Acurracy score: 0.98
Matriz de confusion

[[16  0  0]
 [ 0 17  0]
 [ 0  1 16]]


Report de clasificacion de especies:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        16
           1       0.94      1.00      0.97        17
           2       1.00      0.94      0.97        17

    accuracy                           0.98        50
   macro avg       0.98      0.98      0.98        50
weighted avg       0.98      0.98      0.98        50
```

Ilustración de los resultados de las métricas utilizadas, accuracy score, matriz de confusión y reporte de clasificación.

Diagnóstico y explicación el grado de bias o sesgo: bajo medio alto

Para poder realizar el diagnóstico debemos revisar el sesgo de los datos con los que estamos tratando, en este caso del dataset de iris, para eso usaremos la función de Skew la cual nos genera el analisis del sesgo de cada variable.

```
Medida del sesgo en los datos presentes
SepalLengthCm    0.314911
SepalWidthCm     0.334053
PetalLengthCm   -0.274464
PetalWidthCm    -0.104997
species          0.000000
dtype: float64
```

Ilustración del sesgo por variable del dataset de iris.csv

Como podemos observar el valor de sesgo de cada variable se encuentran dentro de un rango de sesgo aceptable ya que se encuentran dentro del rango 0.5 y -0.5, incluso son muy cercanos al valor de 0, por lo que entendemos que los datos son los adecuados.

Utilizando la función de Kurt() la cual se importó al programa de python, podemos obtener la curtosis de los datos, los resultados son los siguientes.

```
Id              -1.200000
SepalLengthCm   -0.552064
SepalWidthCm     0.290781
PetalLengthCm   -1.401921
PetalWidthCm    -1.339754
species         -1.510135
dtype: float64
```

Ilustración de la curtosis por variable del dataset de iris.csv

Como podemos observar existe una gran curtosis en las variables de petalwidth y length, mientras que en las variables de sepalwidth y length la curtosis es normal o baja ya que tienen valores cercanos al 0. Lo que nos hace entender esto es que los datos para los pétalos no se distribuyen siguiendo un modelo ideal o normal. Por lo que habrá que generar una transformación o manipulación de dichos datos para mejorar el resultado del modelo.

Diagnóstico y explicación el grado de varianza: bajo medio alto

Utilizando la función del módulo de metrics de scikit learn podemos usar la función de explained_variance el cual se encarga de otorgarnos la varianza del modelo con base en la comparación entre los datos reales y aquellos que fueron estimados o creados con predicción. En este caso la interpretación del valor es que mientras más cercano a uno mejor es el modelo porque obtiene menor varianza y cuando es cercano al cero entonces la varianza es alta.

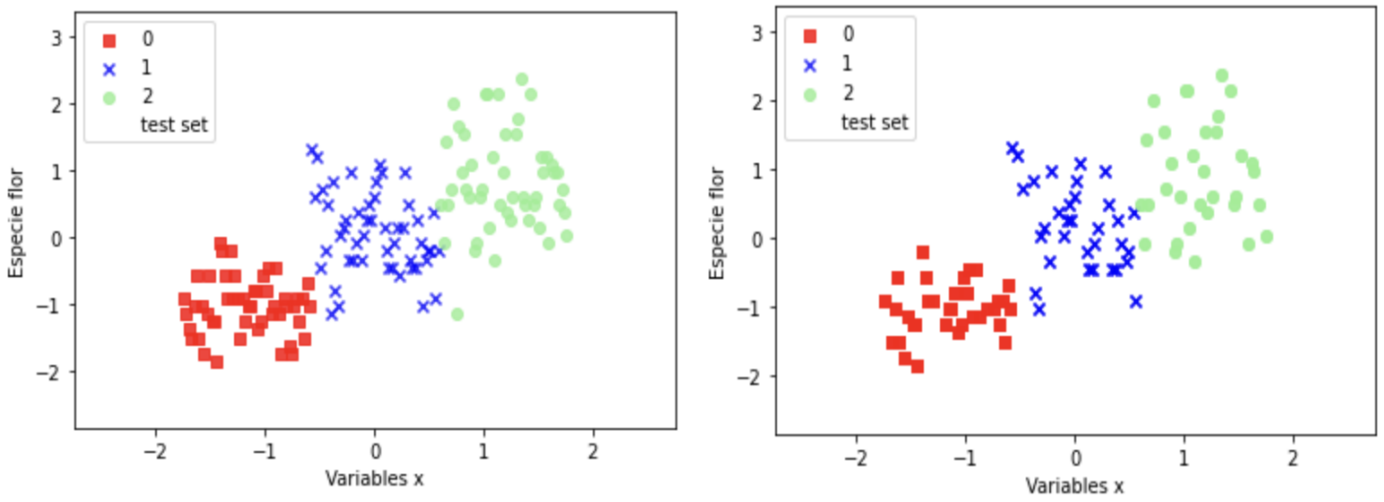
```
1 #Utilizamos los resultados creados por el modelo
2 explained_variance_score(y_prueba, predicciones)

0.9702850212249848
```

Resultado de la varianza del modelo logístico generado con base en los datos estimados y los datos reales del dataset iris.csv

Como podemos observar en este caso, obtuvimos con la función de explained_variance_score un valor de 0.97 o 97%, el cual es muy cercano al 1, con esto podemos entender que el modelo hace predicciones acertadas debido a que la varianza que tiene es baja, con base en esto entendemos que la distancia entre las datos o predicciones que realiza el modelo es corta, y las predicciones se encuentran en un rango de datos concentrado o un rango corto.

Esto es útil para nuestro modelo, siempre y cuando no se tenga un sesgo o una curtosis alta la cual haga que los valores no se estén concentrando en el lugar adecuado.

Diagnóstico y explicación el nivel de ajuste del modelo: underfitt fitt overfitt



Gráficas que muestran la clasificación de especie de flor la derecha muestra las predicciones hechas por el modelo, la izquierda muestra los datos reales.

En las gráficas que se presentan podemos visualizar que para ambos casos tanto en los datos reales como en los datos que se estiman, la diferencia de distancias entre los puntos de ambas gráficas son muy pequeñas, esto nos hace entender que el modelo de regresión logística hace buen trabajo estimando las especies de las flores.

Otra forma en la que podemos evaluar si el modelo tiene un buen ajuste es con base en las métricas de precisión que seleccionamos para nuestro modelo, como la matriz de confusión, el accuracy score y el reporte de clasificación.

```
Acurracy score: 0.98
Matriz de confusion

[[16  0  0]
 [ 0 17  0]
 [ 0  1 16]]


Report de clasificacion de especies:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        16
           1       0.94      1.00      0.97        17
           2       1.00      0.94      0.97        17

    accuracy                           0.98        50
   macro avg       0.98      0.98      0.98        50
weighted avg       0.98      0.98      0.98        50
```

Como se observa de las métricas, se obtuvo un valor de precisión del 98%, lo cual nos demuestra que el modelo está agrupando las flores por especie con base en las variables que la caracterizan de manera acertada. Igualmente la matriz de confusión, la cual demuestra que los datos que se predicen con el modelo encajan correctamente con los datos reales del dataset. Con base en esto podemos concluir que el modelo no tiene un sobre ajuste sino más bien está en un ajuste indicado para poder recibir datos nuevos y predecir en qué grupo de flor pertenece.

Técnicas de regularización o ajuste de parámetros para mejorar el desempeño de tu modelo

Durante el desarrollo de este modelo se realizaron regularizaciones y ajustes que nos permitieron disminuir las peculiaridades del modelo, de esta forma permitiremos que el modelo fuera capaz de crear estimaciones más acertadas con los datos que se le presentaban, sin las regularizaciones el modelo en promedio otorgaba una precisión del 96-97%, sin embargo utilizando los hiperparametros y otro ajuste se logró aumentar a un 98% en algunos casos.

Se dio uso del módulo de scikit learn llamado standard scaler, para poder cambiar el comportamiento de los datos del dataset de iris.csv. Se sabe que los datos de pétalos obtuvieron una curtosis más alta que las demás columnas, por lo que esto podría afectar la media y la mediana de nuestro conjunto de datos, esto como resultado afectaría el aprendizaje de nuestro modelo. Es por eso que con el módulo de estándar scaler, se logra transformar los datos presentes para que tengan una distribución más ideal para nuestro modelo.

Así mismo utilizamos hiperparametros dentro de nuestro modelo de regresión logística, para buscar la posibilidad de mejorar la precisión de nuestro modelo, con base en esto decidimos agregar un iterador máximo dentro de la regresión en este caso tiene un valor de 600 iteraciones, por otro lado también se le agregó un "penalty" al modelo de tipo "l2" y un solver llamado "bfgs" el cual significa "limited memory broyden fletcher goldfarb shanno", todo estos hiperparametros llevaron a la precisión del modelo que se obtuvo al final, el cual tiene una precisión del 98% por lo que podemos concluir que los hiperparametros nos ayudaron a aumentar la precisión del modelo a un 1-2%.

Repositorio del modelo realizado:

https://github.com/jpablocruz/ModelosImplementados_ConcentracionIA