Instituto Tecnológico y de Estudios Superiores de Monterrey

TC3006C.101

Inteligencia Artificial Avanzada para la ciencia de datos. (Gpo. 101)

ENLACE AL PORTAFOLIO

Análisis y Reporte sobre el desempeño del modelo.

Profesores:
Dr. Ivan Mauricio Amaya Contreras

Realizado por : Javier de Golferichs - A01139500

16 de septiembre de 2022

1. Introducción

La implementación seleccionada para esta entrega es la segunda entrega, que usa el framework de redes neuronales. La librería usada para el modelo son sklearn.neural_network.

Los parámetros que se cambiaron para evaluar su exactitud es el número de capas ocultas y neuronas por capa.

El código inicia con la conexión a drive, la importación del dataset (wines.data) y la definición de los datos de entrada (todas las 13 características) y datos de salida (la clase a la que corresponde).

El link al repositorio Github del portafolio se encuentra en este enlace

2. Análisis de la implementación

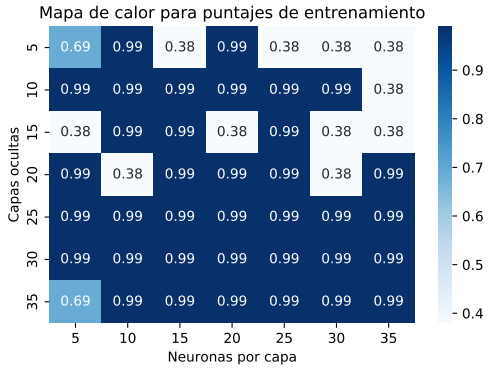
En el listado 1 se presenta el código con el que se obtienen y almacenan los puntajes y matrices de confusión para los subconjuntos de entrenamiento, validación y prueba. Las iteraciones llevan a matrices de puntajes de 5 a 35 capas ocultas y 5 a 35 neuronas por capa.

```
scores_train = []
scores_valid= []
scores_test = []
almacenamiento_conf_mat_train = []
almacenamiento_conf_mat_valid = []
almacenamiento_conf_mat_test = []

cont = 0

for ii in range (5, 45, 5): # el número de ii (ii,jj) es el que modifica el número de capas
    ocultas
  if cont == 1:
    scores_train.append(jj_scores_train)
    scores_valid.append(jj_scores_valid) # aqui estaba el error
    scores_test.append(jj_scores_test)
    almacenamiento_conf_mat_train.append(jj_mc_train)
    almacenamiento_conf_mat_valid.append(jj_mc_valid) # aqui estaba el error
    almacenamiento_conf_mat_test.append(jj_mc_test)

  cont = 1

  jj_scores_train = []
  jj_scores_valid = []
  jj_scores_test = []
  jj_mc_train = []
  jj_mc_valid = []
  jj_mc_test = []

  for jj in range(5,40,5):
    cross_val_predict,
    nnRE = MLPClassifier(hidden_layer_sizes=(ii,jj), ## cambiar estos parámetros
                         activation='logistic', verbose=False, solver='adam',
                         learning_rate='adaptive', max_iter=2000)
```

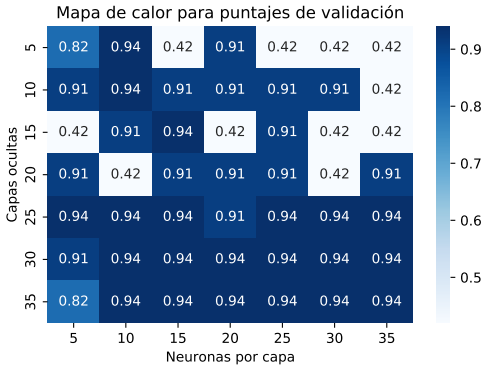
```
nnRE.fit(X_train,y_train)
jj_scores_train.append(nnRE.score(X_train, y_train))
jj_scores_valid.append(nnRE.score(X_valid, y_valid))
jj_scores_test.append(nnRE.score(X_test, y_test))

jj_mc_train.append(confusion_matrix(y_train,cross_val_predict(nnRE,X_train,y_train, cv =
    10)))
jj_mc_valid.append(confusion_matrix(y_valid,cross_val_predict(nnRE,X_valid,y_valid, cv =
    10)))
jj_mc_test.append(confusion_matrix(y_test,cross_val_predict(nnRE,X_test,y_test, cv = 5)))
```

Listado 1: Obtención y almacenamientos de puntajes y matrices de confusión. Nótese que el rango utilizado va desde 5 capas hasta 35 capas, y jj va desde 5 unidades por capa hasta 35.

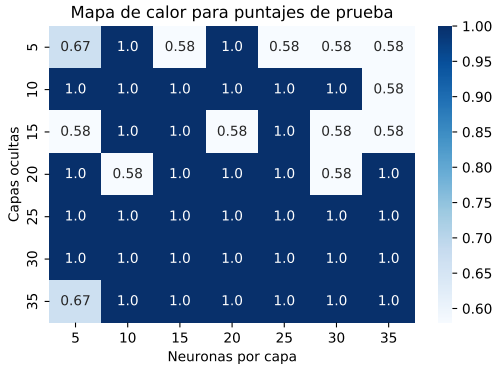
En la figura 1, se observa el cambio en los puntajes de los tres subconjuntos de datos, haciendo las iteraciones de número de capas ocultas y neuronas por capa. Las tablas de datos de estas figuras se presentan en los anexos (cuadros 2 a 4).



(a)



(b)



(c)

Figura 1: 1a Puntajes de subconjunto de entrenamiento. 1b Puntajes de subconjunto de validación. 1c. Puntajes de subconjunto de prueba.

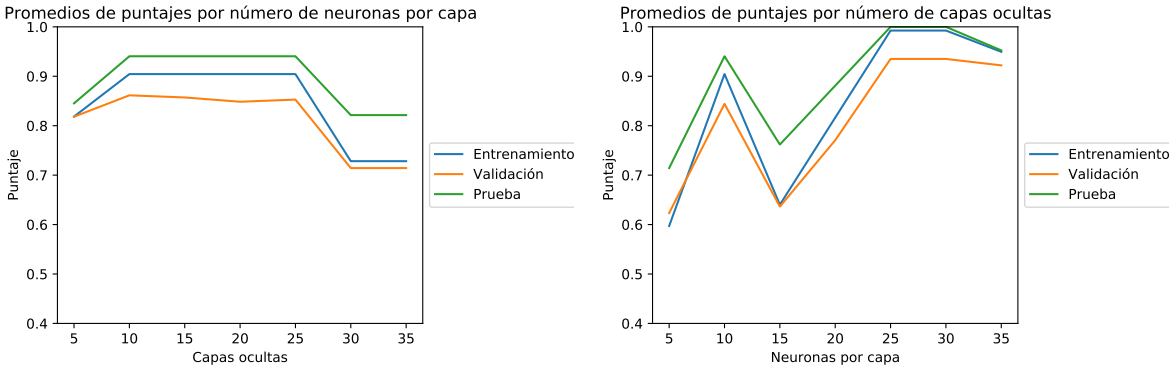
Nótese de las figuras 1a a 1c, que tienen patrones similares, lo cual es relevante para la evaluación de underfitting y overfitting. Los valores superiores a 0.90 en figuras 1a y 1b suben a 1.0 en la figura 1c.

2.1. Análisis de sesgo, varianza y nivel de ajuste del modelo

La varianza y el sesgo son inversamente proporcionales, tal que si uno aumenta, por el otro disminuye. Si hay underfitting se encuentra que hay alto sesgo, y overfitting, se tiene alta varianza.

Para la evaluación de estos conceptos se hace una comparación de los puntajes de los tres subconjuntos (entrenamiento, validación y prueba).

En la figura 2a los promedios de los puntajes con respecto al número de neuronas por capa y en figura 2b se presentan los promedios de los puntajes con respecto al número de capas ocultas.



(a) (b)

Figura 2: 2a Promedios de los puntajes con respecto al número de neuronas por capa. 2b Promedios de los puntajes con respecto al número de capas ocultas..

En la figura 2b se encuentra que en general aumentar el número de neuronas por capa aumenta el puntaje encuentra que claramente aumentar el número de capas implica un mayor puntaje, excepto cuando se tienen 15 neuronas por capa. Cuando se usan 25 neuronas por capa, se encuentra que se llega al máximo puntaje para los tres subconjuntos.

En la figura 2a, sucede algo similar, pues se encuentra que el máximo está cuando se usan 25 capas oculta, y disminuye cuando se usan 30 y 35 capas ocultas.

El modelo seleccionado como técnica de regularización una red neuronal de 25 capas ocultas y 25 neuronas por capa.

De este se considera que se tiene **sesgo medio**, pues los puntajes obtenidos para los subconjuntos de entrenamiento y validación superan el puntaje de .90, lo cual se considera aceptable.

Con respecto a la varianza se encuentra que tiene **nivel de varianza medio**, aceptable, debido a que el sesgo no es bajo ni alto.

Por estas razones **se considera que este modelo no tiene underfitting ni overfitting, sino que se tiene un balance**.

3. Técnica de regularización

3.1. Modelo final seleccionado y nivel de ajuste.

El modelo seleccionado como técnica de regularización una red neuronal de 25 capas ocultas y 25 neuronas por capa.

Como mencionado en la sección anterior, se considera que el modelo tiene sesgo medio, varianza media y un balance entre underfitting y overfitting.

El subconjunto de prueba, dan un puntaje de 1 para el caso de este modelo.

3.2. Comparación de modelo simple y modelo seleccionado.

Se retoma como **modelo inicial** la red neuronal con 5 capas ocultas y 5 unidades por capa.

Se considera que el modelo inicial de 5 capas ocultas y 5 unidades por capa sufre de un poco de baja exactitud y bajos niveles de underfitting, puesto que el puntaje para el conjunto de entrenamiento es de 0.69, que es menor que el de validación de validación de 0.82. Además de no tener un buen desempeño en el subconjunto de prueba (0.67).

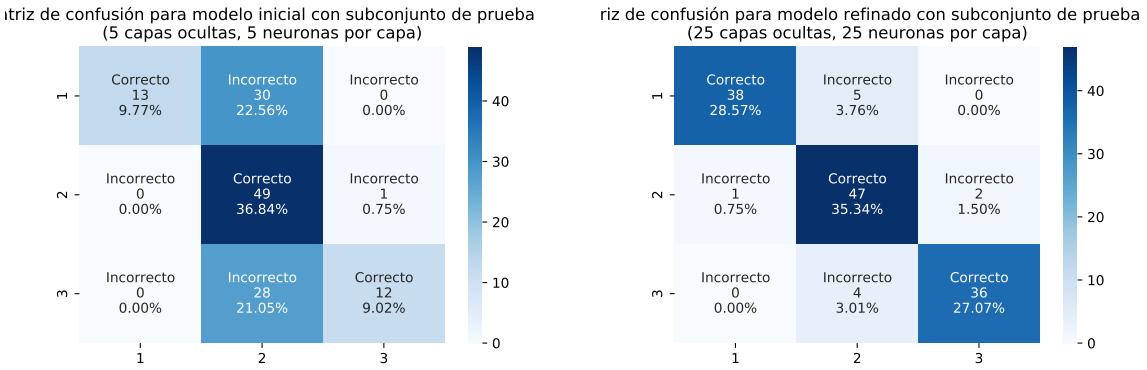
En la cuadro 1 se comparan los puntajes para los subconjuntos de datos de entrenamiento, validación y prueba para el modelo inicial y el modelo refinado (25 capas ocultas y 25 neuronas por capa).

	Modelo inicial	Modelo refinado
Entrenamiento	0.69	0.99
Validación	0.82	0.94
Prueba	0.67	1.00

Tabla 1: Comparación de modelo inicial y modelo refinado

En el cuadro 1, se observa que el desempeño del modelo refinado es superior para los tres subconjuntos.

A continuación se presentan las matrices de confusión del subconjunto de prueba para ambos modelos.



(a) (b)

Figura 3: 3a Matriz de confusión para modelo inicial con subconjunto de prueba (5 capas ocultas, 5 neuronas por capa). 3b Matriz de confusión para modelo refinado con subconjunto de prueba (25 capas ocultas, 25 neuronas por capa).

La figura 3 muestra el modelo inicial tiene problemas para identificar correctamente las clases 1 y 3. El modelo refinado también, pero en menor medida.

De esto se concluye que el modelo refinado ofrece más ventajas que el modelo inicial, lo cual se relaciona con la mayor complejidad del modelo. El modelo refinado cuenta con sesgo medio y varianza media, así como que no sufre de underfitting ni overfitting, y tiene buen desempeño.

A. Portafolio de implementación

Link a Portafolio de implementación con códigos utilizados: Oprima aquí

B. Portafolio de análisis

Link a Portafolio de análisis con reporte: Oprima aquí

C. Tablas con puntajes

	5	10	15	20	25	30	35
0	0.691729	0.992481	0.375940	0.992481	0.375940	0.375940	0.375940
1	0.992481	0.992481	0.992481	0.992481	0.992481	0.992481	0.375940
2	0.375940	0.992481	0.992481	0.375940	0.992481	0.375940	0.375940
3	0.992481	0.375940	0.992481	0.992481	0.992481	0.375940	0.992481
4	0.992481	0.992481	0.992481	0.992481	0.992481	0.992481	0.992481
5	0.992481	0.992481	0.992481	0.992481	0.992481	0.992481	0.992481
6	0.691729	0.992481	0.992481	0.992481	0.992481	0.992481	0.992481

Tabla 2: Puntajes para datos de entrenamiento

	5	10	15	20	25	30	35
0	0.818182	0.939394	0.424242	0.909091	0.424242	0.424242	0.424242
1	0.909091	0.939394	0.909091	0.909091	0.909091	0.909091	0.424242
2	0.424242	0.909091	0.939394	0.424242	0.909091	0.424242	0.424242
3	0.909091	0.424242	0.909091	0.909091	0.909091	0.424242	0.909091
4	0.939394	0.939394	0.939394	0.909091	0.939394	0.939394	0.939394
5	0.909091	0.939394	0.939394	0.939394	0.939394	0.939394	0.939394
6	0.818182	0.939394	0.939394	0.939394	0.939394	0.939394	0.939394

Tabla 3: Puntajes para datos de validación

	5	10	15	20	25	30	35
0	0.666667	1.000000	0.583333	1.000000	0.583333	0.583333	0.583333
1	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	0.583333
2	0.583333	1.000000	1.000000	0.583333	1.000000	0.583333	0.583333
3	1.000000	0.583333	1.000000	1.000000	1.000000	0.583333	1.000000
4	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
5	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
6	0.666667	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

Tabla 4: Puntajes para datos de prueba