Profesor: Dr. Oldemar Rodríguez Rojas
Análisis de Datos 2
Fecha de Entrega: Domingo 20 de noviembre a las 12 media noche
Instrucciones:

- Las tareas son estrictamente individuales.
- Tareas idénticas se les asignará cero puntos.
- Todas las tareas tienen el mismo valor en la nota final del curso.
- Cada día de entrega tardía tendrá un rebajo de 20 puntos.

Tarea Número 12

1. [**50 puntos**] La tabla de datos `novatosNBA.csv` contiene diferentes métricas de desempeño de novatos de la NBA en su primera temporada. Para esta tabla, las 21 primeras columnas corresponden a las variables predictoras y la variable Permanencia es la variable a predecir, la cual indica si el jugador permanece en la NBA luego de 5 años. La tabla contiene 1340 filas (individuos) y 21 columnas (variables), con la tabla realice lo siguiente:

   a) Usando el paquete `MLPClassifier` en `Python` genere modelos predictivos usando un 75 % de los datos para tabla aprendizaje y un 25 % para la tabla testing. Genere al menos 2 modelos con configuraciones diferentes en los parámetros vistos en clase(hidden_layer_sizes, activation, solver). Realice lo anterior sin estandarizar los datos y luego con los datos estandarizados, es decir, al menos 4 modelos. Para estandarizar los datos utilice la clase `StandardScaler` de `sklearn.preprocessing`

   b) Para cada modelo obtenga los índices de precisión, compare e interprete los resultados y las diferencias entre los modelos con datos estandarizados y los que no.

2. [20 puntos] Este conjunto de datos es originalmente del Instituto Nacional de Diabetes y Enfermedades Digestivas y Renales. El objetivo del conjunto de datos es predecir de forma diagnóstica si un paciente tiene diabetes o no, basándose en determinadas medidas de diagnóstico incluidas en el conjunto de datos..

   El conjunto de datos tiene 390 filas y 16 columnas.

   - `X`: Id del paciente.
   - `colesterol`: Colesterol en mg/dL.
   - `glucosa`: Glucosa en mg/dL.
   - `hdl_col`: Lipoproteínas (colesterol bueno).
   - `prop_col_hdl`: Proporción del colesterol entre el hdl.
   - `edad`: Edad del paciente.
   - `genero`: Género del paciente.
   - `altura`: Altura en pulgadas del paciente.
   - `peso`: Peso en libras del paciente.

- `IMC`: índice de masa corporal.
- `ps_sistolica`: Presión arterial sistólica.
- `ps_diastolica`: Presión arterial diastólica.
- `cintura`: Longitud de la cintura en pulgadas.
- `cadera`: Longitud de la cadera en pulgadas.
- `prop_cin_cad`: Proporción de la longitud de la cintura entre la longitud de la cadera.
- `diabetes`: Diagnóstico de la diabetes (variable a predecir).

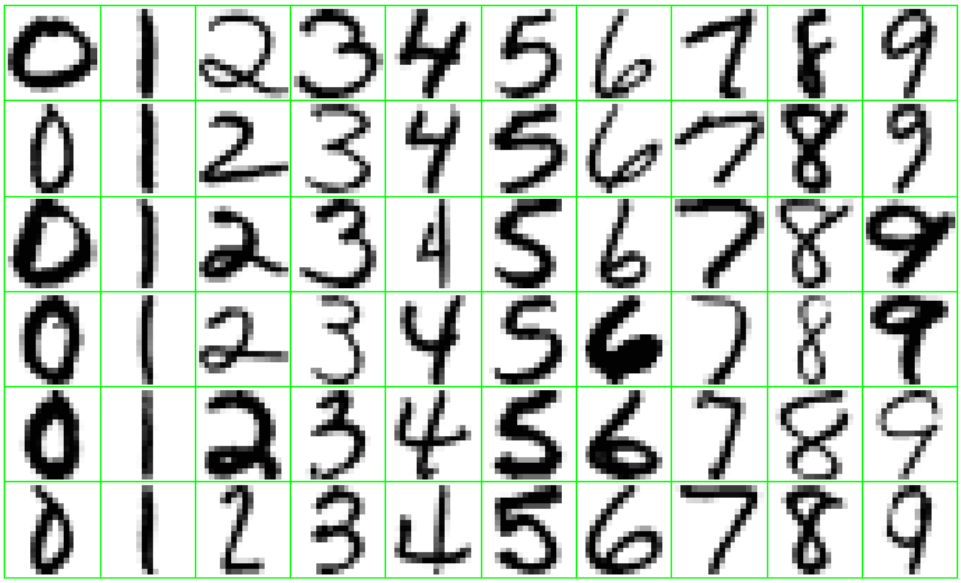
Realice lo siguiente:

a) Cargue en **Python** la tabla de datos `diabetes.csv`.

b) Usando el paquete `MLPClassifier` en `Python` genere modelos predictivos usando un 75 % de los datos para tabla aprendizaje y un 25 % para la tabla testing. Genere al menos 2 modelos con configuraciones diferentes en los parámetros vistos en clase(hidden_layer_sizes, activation, solver). Realice lo anterior sin estandarizar los datos y luego con los datos estandarizados, es decir, al menos 4 modelos. Para estandarizar los datos utilice la clase `StandardScaler` de `sklearn.preprocessing`

c) Para cada modelo obtenga los índices de precisión, compare e interprete los resultados y las diferencias entre los modelos con datos estandarizados y los que no.

3. [20 puntos] En este ejercicio vamos a predecir números escritos a mano (Hand Written Digit Recognition), la tabla de aprendizaje está en el archivo `ZipDataTrainCod.csv` y la tabla de testing está en el archivo `ZipDataTestCod.csv`. En la figura siguiente se ilustran los datos:



Los datos de este ejemplo vienen de los códigos postales escritos a mano en sobres del correo postal de EE.UU. Las imágenes son de 16×16 en escala de grises, cada pixel va de intensidad de -1 a 1 (de blanco a negro). Las imágenes se han normalizado para tener aproximadamente

el mismo tamaño y orientación. La tarea consiste en predecir, a partir de la matriz de 16×16 de intensidades de cada pixel, la identidad de cada imagen $(0, 1, \ldots, 9)$ de forma rápida y precisa. Si es lo suficientemente precisa, el algoritmo resultante se utiliza como parte de un procedimiento de selección automática para sobres. Este es un problema de clasificación para el cual la tasa de error debe mantenerse muy baja para evitar la mala dirección de correo. La columna 1 tiene la variable a predecir `Número` codificada como sigue: 0=‘cero’; 1=‘uno’; 2=‘dos’; 3=‘tres’; 4=‘cuatro’; 5=‘cinco’;6=‘seis’; 7=‘siete’; 8=‘ocho’ y 9=‘nueve’, las demás columnas son las variables predictivas, además cada fila de la tabla representa un bloque 16×16 por lo que la matriz tiene 256 variables predictoras.

a) Usando el paquete `MLPClassifier` en `Python` genere un modelo predictivo de redes neuronales para estos datos. Utilice los siguientes parámetros: `hidden_layer_sizes = (250,100,50,25), max_iter = 50000, activation = ‘‘relu’’, solver = ‘‘adam’’, random_state=0`.
Interprete los resultados.

b) Genere un modelo de redes neuronales con los mismos parámetros del ítem anterior, pero esta vez reemplace cada bloque 4×4 de píxeles por su promedio. ¿Mejora la predicción? ¿Qué ventaja tiene estos datos respecto a los anteriores? Recuerde que cada bloque 16×16 está representado por una fila en las matrices de aprendizaje y testing. **Despliegue la matriz de confusión resultante**. La matriz de confusión obtenida debería ser igual o muy similar a ésta:

	cero	uno	dos	tres	cuatro	cinco	seis	siete	ocho	nueve
cero	343	6	2	0	0	0	3	1	4	0
uno	1	250	1	0	5	1	1	3	2	0
dos	5	3	180	1	0	1	0	1	6	1
tres	2	1	3	138	0	15	0	1	4	2
cuatro	0	4	4	0	166	1	2	0	1	22
cinco	10	0	1	17	0	123	0	1	5	3
seis	6	1	2	0	2	2	157	0	0	0
siete	0	0	2	0	7	0	0	130	1	7
ocho	9	19	2	2	1	5	2	0	124	2
nueve	1	0	0	0	2	0	0	1	1	172

No es necesario que las categorías se muestren en orden.

c) Repita el item anterior pero esta vez reemplace cada bloque 4×4 de píxeles por el máximo. ¿Mejoran resultados respecto a usar el promedio de cada bloque?

4. [20 puntos] Represente en un grafo dirigido la Red Neuronal que tiene la siguiente entrada:

$$\overrightarrow{x} = \begin{pmatrix} -2 \\ 1 \\ 1 \\ 3 \end{pmatrix}.$$

Tiene las siguientes matrices de pesos:

$$W_1 = \begin{pmatrix} 1 & 2 & 0 & 4 \\ 5 & 3 & 1 & 2 \\ 2 & 3 & 0 & 2 \end{pmatrix}, \quad W_2 = \begin{pmatrix} 4 & 2 & 3 \\ 1 & 3 & 6 \end{pmatrix}.$$

Tiene el siguiente bías:

$$\overrightarrow{b} = \begin{pmatrix} -6 \\ -2 \end{pmatrix}.$$

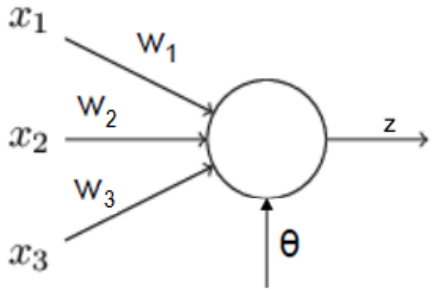
Además, use una función de activación tipo Tangente Hiperbólica, es decir:

$$f(x) = \frac{2}{1 + e^{-2x}} - 1.$$

5. [20 puntos] [**no usar** `MLPClassifier`] Para la Tabla de Datos que se muestra seguidamente donde x^j para $j = 1, 2, 3$ son las variables predictoras y la variable a predecir es z diseñe y programe a pie una Red Neuronal de una capa (Perceptron):

x^1	x^2	x^3	z
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

Es decir, encuentre todos los posibles pesos w_1, w_2, w_3 y umbrales θ para la Red Neuronal que se muestra en el siguiente gráfico:



Use una función de activación tipo Sigmoidea, es decir:

$$f(x) = \frac{1}{1 + e^{-x}}.$$

Para esto escriba una Clase en `Python` que incluya los métodos necesarios pra implementar esta Red Neuronal.

Se deben hacer variar los pesos w_j con $j = 1, 2, 3$ en los siguientes valores `v=(-1,-0.9,-0.8,...,0,...,0.8,0.9,1)` y haga variar θ en `u=(0,0.1,...,0.8,0.9,1)`. Escoja los pesos w_j con $j = 1, 2, 3$ y el umbral θ de manera que se minimiza el error cuadrático medio:

$$E(w_1, w_2, w_3) = \frac{1}{4} \sum_{i=1}^{4} \left[I \left[f \left(\sum_{j=1}^{3} w_j \cdot x_i^j - \theta \right) \right] - z_i \right]^2,$$

donde x_i^j es la entrada en la fila i de la variable x^j e $I(z)$ se define como sigue:

$$I(t) = \begin{cases} 1 & \text{si } t \geq \frac{1}{2} \\ 0 & \text{si } t < \frac{1}{2}. \end{cases}$$