HE2: IA Aplicada a la Economía – Período 2025-10

Profesor principal: Juan Camilo Vega, jc.vega611@uniandes.edu.co

Profesor asistente: Sergio Julian Zona Moreno, sj.zona@uniandes.edu.co

Parcial 2

- **Fecha de entrega: viernes 14 de marzo antes de las 12:20 PM (cerrando la clase).**
- **El parcial deberá subirse a *Bloque Neón* en el enlace que designará el equipo docente.**
- **El entregable final es el *link al repositorio (documentado) y el cuadernillo de Jupyter Notebook* en formato .ipynb o .html. Allí deberá quedar todo el análisis y las celdas ejecutadas de código.**

Contexto:

En un mundo cada vez más interconectado, comprender el panorama económico global es esencial para formuladores de políticas, investigadores y empresas. Este parcial permitirá aplicar los conceptos vistos en los laboratorios de Python, utilizando el *World GDP Dataset* extraído del World Bank Group. Este *dataset* incluye información sobre el Producto Interno Bruto (PIB) de múltiples países desde 1960 hasta 2022.

Estructura de evaluación:

- **Procesamiento de datos y preparación:** 20%
- **Construcción de modelos de redes neuronales:** 60%
- **Relatoría del equipo:** 20%

Aclaración:

Es importante que **todas las preguntas** sean contestadas. En caso contrario se penaliza la respuesta incluso si el código funciona correctamente.

Instrucciones:

(1 punto) Procesamiento de datos y preparación: 20%

1. **(0.25 puntos)** Descargue el *dataset* del siguiente enlace:

   https://www.kaggle.com/datasets/sazidthe1/world-gdp-data

   Cargue en su entorno de trabajo los siguientes archivos CSV:

   a. *gdp_data.csv*: Contiene valores del PIB por país y año.

   b. *country_codes.csv*: Contiene la clasificación de países por región y grupo de ingreso.

   Cree un diccionario de las variables (ver el cuadernillo *Monitoria_Parcial1.ipynb*), explique brevemente cada una de ellas, realice un análisis descriptivo y procese los datos. Este *dataset*

es de regresión, pero deberán transformarlo en clasificación. Revise los anexos para instrucciones/consejos adicionales.

(3 puntos) Construcción de modelos de redes neuronales: 60%

2. A continuación, construya tres redes neuronales sobre el *dataset*:
   a. **(1.5 puntos) Red neuronal tradicional:** construya la mejor red neuronal MLP posible con *Scikit-Learn*, con la cantidad de capas (mínimo 1 capa oculta), neuronas y número de épocas que considere necesario. Realice búsqueda de hiperparámetros, optimice la arquitectura, ajuste la regularización y aplique otras estrategias para mejorar su desempeño.
   b. **(1.5 puntos) Red neuronal profunda:** construya la mejor red neuronal MLP posible con *Tensorflow*, con la cantidad de capas (mínimo 2 capas ocultas), neuronas y número de épocas que considere necesario. Realice búsqueda de hiperparámetros, optimice la arquitectura, ajuste la regularización y aplique otras estrategias para mejorar su desempeño.
   c. **(0.25 puntos) Bono: Red neuronal con mala función de pérdida:** construya una red neuronal con *Tensorflow*, con una función de pérdida incorrecta (por ejemplo, MSE en clasificación) y un *learning rate* demasiado alto. Entrene la red y analice su desempeño. Evalúe cómo afecta esta configuración a la convergencia y al resultado final del modelo. Identifique los errores en la configuración, explique por qué afectan el rendimiento y ajuste los parámetros para corregirlos, justificando cada cambio realizado.

      Nota: también pueden crear su propia función de pérdida personalizada.

```python
def custom_classification_loss(y_true, y_pred):
    loss_ce = tf.keras.losses.CategoricalCrossentropy()(y_true, y_pred)
    loss_kl = tf.keras.losses.KLDivergence()(y_true, y_pred)
    return 0.6 * loss_ce + 0.4 * loss_kl
```

   d. **(0.25 puntos) Bono:** exporte sus modelos (consulte cómo hacerlo) y cárguelos para los siguientes pasos.

No olvide graficar sus resultados para cada modelo:

- Grafique la función de pérdida a medida que aumenta el número de iteraciones.
- Grafique la tasa de aprendizaje a medida que aumenta el número de iteraciones.
- Grafique la precisión del modelo (u otra métrica que considere) a medida que aumenta el número de iteraciones.
- Grafique la curva ROC.
- Grafique la matriz de confusión tanto en entrenamiento como en prueba.

Compare los resultados de las redes neuronales tanto en entrenamiento como en prueba. ¿Cuál dio mejor resultado? ¿Qué cree que falta hacer en este ejercicio? ¿Qué podría mejorar los resultados?

3. **(0.25 puntos) Bono:** haga un análisis con SHAP Values y revise la importancia de las variables en la predicción. ¿Cuáles influyen más? ¿Cómo se comportan? ¿Puede argumentar causalidad?

(1 punto) Relatoría del equipo: 20%

- Explique las preguntas que se hicieron y cómo resolvieron los problemas encontrados.
- Exponga las soluciones y conclusiones a las que llegaron.
- Explique cómo se dividieron el trabajo. En qué aportó cada integrante.

Consejos:

- Se permite el uso de herramientas basadas en inteligencia artificial (IA) para apoyar el desarrollo del parcial. Se sugiere utilizar plataformas como ChatGPT, Copilot, Claude, BlackBox, entre otras, según lo considere conveniente.
- Es recomendable revisar los materiales y cuadernillos trabajados durante los laboratorios, ya que contienen ejemplos y procedimientos relevantes para la resolución de este parcial.
- Le recomendamos utilizar como base el cuadernillo *Monitoria_Parcial1.ipynb* para el procesamiento de datos.

Anexo*:*

Recomendaciones para procesar los datos:

1. **(0.25 puntos) Bono:** Creación de una nueva variable y construcción del *dataset*.
    - Seleccione una variable adicional de interés (por ejemplo, expectativa de vida, años de escolaridad, precios del petróleo, inflación, etc.).
    - Busque un *dataset* externo con información relevante sobre la variable elegida.
    - Integre la nueva variable asegurando que coincida con las columnas claves: países, años, etc.
    - Explique la fuente y justifique su elección.
    - Al final debería tener un solo *DataFrame* con las tres fuentes de datos.
2. Como este es un ejercicio de aprendizaje supervisado, divida desde el inicio el *train set* y el *test set* en 80% y 20% de los datos respectivamente. Al final debe tener cuatro *DataFrames*: *X_train*, *X_test*, *y_train*, y *y_test*. ¿Debe tener cuidado al dividir el *dataset*? ¿Por qué?
3. Realice un análisis descriptivo de las variables. Se recomienda utilizar una librería como *ydata_profiling* para esta tarea. No olvide responder:
    - ¿Cuál es la variable objetivo?

- ¿Qué teorías puede plantear a partir de este análisis sobre la variable objetivo?
- ¿Existen variables con alta correlación?
- ¿Hay variables que se podrían eliminar? ¿Por qué?
- ¿Qué ocurre con la serie de tiempo? ¿Cuál es el desafío?

4. Procesamiento:
    a. Elimine/corrija aquellas filas que considere dañadas o no útiles para el análisis. Justifique.
    b. Elimine/corrija aquellas columnas que considere dañadas o no útiles para el análisis. Justifique.
    c. Transforme las variables categóricas en variables numéricas mediante *OneHotEncoding* o *LabelEncoding*. ¿Cuál es la diferencia entre ambos métodos? ¿Cuál debería utilizar?
    d. Estandarice/normalice las variables numéricas (recuerde que este paso es opcional en la *y*, pero obligatorio en las *X*).
5. Del *y_train* y *y_test* que tienen valores numéricos, cree dos copias que serán *y_train_class* y *y_test_class*. Plantee una regla/estrategia para dividir el GDP en *Low, Medium* y *High*. Ejemplo:
    - Si el GDP promedio es menor a 57000, será "*Low GDP*".
    - Si el GDP promedio es menor a 70000 y mayor de 60000, será "*Medium GDP*".
    - Si el GDP promedio es mayor a 70000, será "*High GDP*".

    La regla utilizada solo fue un ejemplo (y los valores fueron inventados), puede utilizar alguna otra forma de dividir estas tres categorías (percentiles, promedios ponderados, etc). Justifique dicha elección.

    Consejo: aplique *OneHotEncoder* o *LabelEncoder* a *y_train_class* y *y_test_class* luego de crear la variable categórica. El resultado final debería verse así:

y_train

GDP
57000
63000
78000
66000
58000

→

y_train_class
OneHotEncoder

Low GDP	Medium GDP	High GDP
1	0	0
0	1	0
0	0	1
0	1	0
1	0	0

V

y_train_class
LabelEncoder

GDP Level
1
2
3
2
1