

**Apellidos y Nombres:** Huamani Feria Claudio

**ID:** 001597435

**Dirección Zonal/CFP:** Arequipa/Arequipa/Puno **Carrera:** Ingeniería de software con Inteligencia artificial **Semestre:** 4to Semestre

**Instructor:** Jesús Romero Villanueva

**Curso:** seminario de complementación práctica I

# TAREAS Y OPERACIONES COVERTURADAS

**-TH1\_Realiza operaciones con las Librerías Pandas y Numpy**

* Estudia los fundamentos de vectores y matrices.
* Define la librería Pandas y Numpy.
* Manipula y analiza estructuras de datos.
* Lee archivos CSV con Numpy y Pandas.

# -TH2\_Estudia el uso de las Librerías Scikit-learn y Pytorch

* Define el concepto de Machine Learning
* Define la librería Scikit-Learn y Pytorch.
* Identifica principales aplicaciones.

# -TH3\_Estudia el uso de las librerías SciPy y Nltk

* Define que el procesamiento de lenguaje natural (NLP)
* Define la librería SciPy y Nltk.
* Identifica principales aplicaciones del NLP

# -TH4\_Estudia el uso de las Librerías Tensorflow y Keras

* Define el concepto de Deep Learning
* Define la librería Tensorflow y Keras.
* Identifica principales aplicaciones

# -TH5\_Realiza operaciones con las librerías Matplotlib y Seaborn

* Define los histogramas y la importancia de la visualización de datos.
* Define la librería Matplotlib y Seaborn.
* Crea ejemplos de aplicación

# -TH6\_Crea programas con algoritmos de aprendizaje supervisado

* Describe los tipos de algoritmos del aprendizaje supervisado.
* Define la regresión lineal simple y múltiple.
* Implementa algoritmo de regresión lineal  
  simple con Python.

# -TH6\_Crea programas con algoritmos de no aprendizaje supervisado

* Describe los tipos de algoritmos del aprendizaje supervisado.
* Define la regresión lineal simple y múltiple.
* Implementa algoritmo K-Means con  
  Python.

# \_TH7\_ Crea programas con algoritmos de aprendizaje supervisado

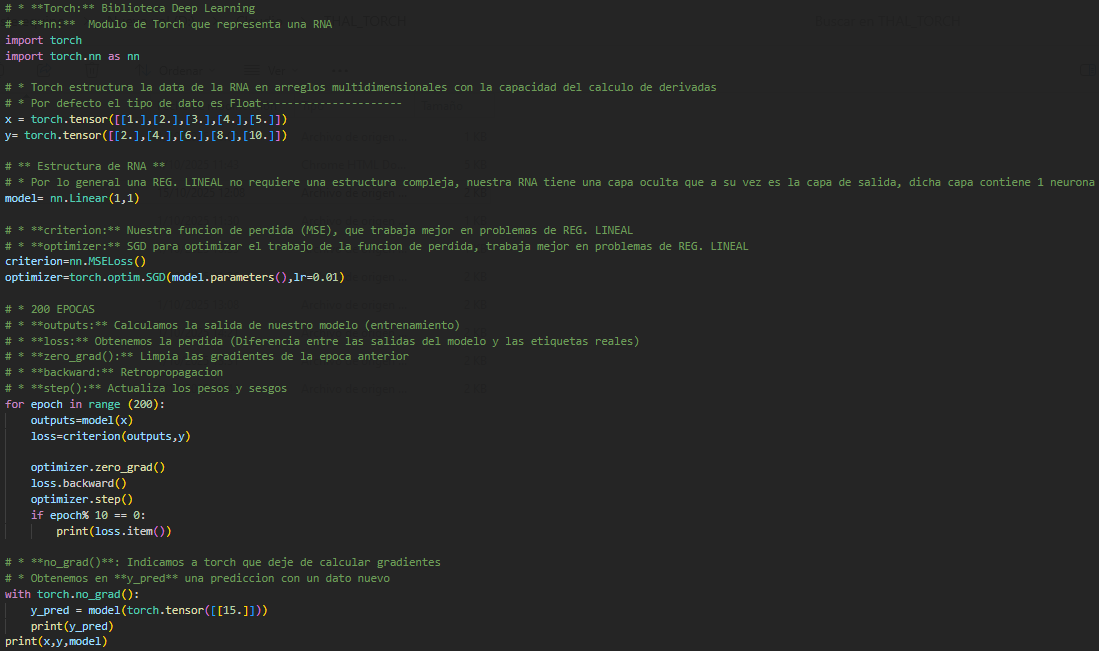
* Describe los tipos de algoritmos del aprendizaje supervisado.
* Define la regresión lineal simple y múltiple.
* Implementa algoritmo de regresión lineal simple con Python.

# \_TH8\_ Define la estructura y crea una red neuronal artificial

* Define la red neuronal artificial y su importancia en al IA.
* Describe la estructura de una red neuronal artificial.
* Identifica los tipos de redes neuronales artificiales.
* Crea una red neuronal con Tensorflow y Keras

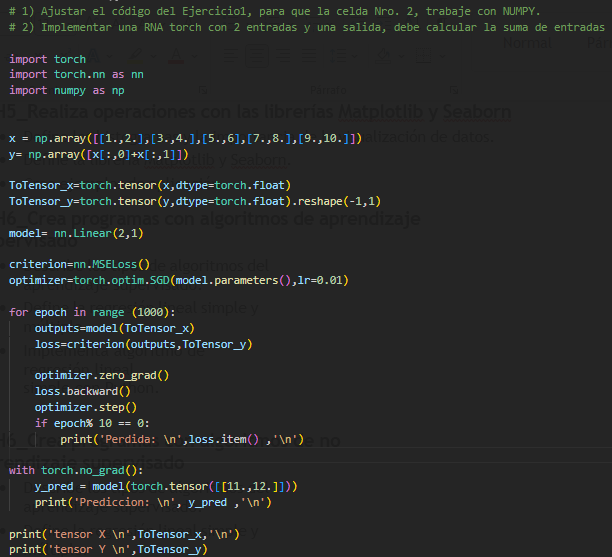
Tarea 1

Se crea un modelo de regresión lineal simple con PyTorch usando una capa lineal.  
Se entrena con datos de entrada y salida, optimizando el error cuadrático medio con SGD durante 200 épocas.



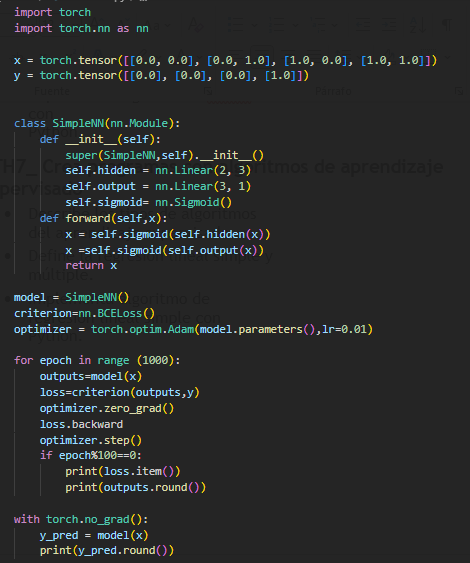
Tarea 2

Se generan datos con dos entradas usando NumPy, luego se convierten a tensores de PyTorch para entrenar una red simple con una capa lineal que aprende a sumar las dos entradas.



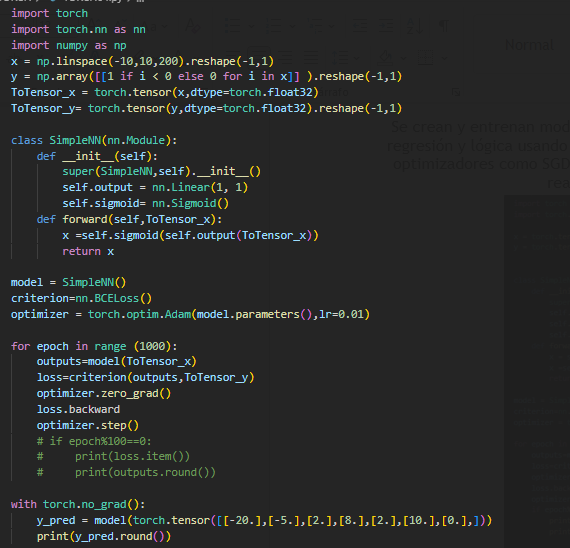
Tarea 3

Se crean y entrenan modelos de redes neuronales en PyTorch para tareas de regresión y lógica usando funciones de pérdida como MSE y BCELoss junto con optimizadores como SGD y Adam, preparando datos con NumPy o tensores y realizando predicciones precisas.



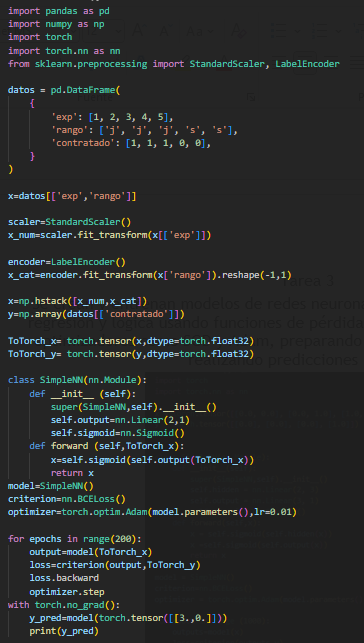
Tarea 4

Se crea y entrena una red neuronal simple con una sola neurona y activación sigmoide para clasificar valores negativos y positivos usando BCELoss y Adam.



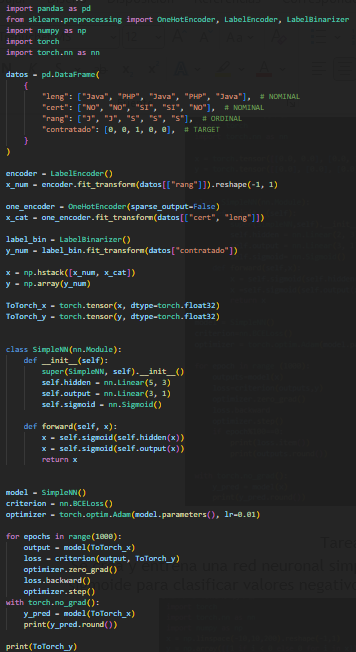
Tarea 5

El código prepara datos con experiencia y rango codificados luego crea una red neuronal simple con PyTorch que usa una capa lineal y sigmoid para predecir la variable contratado optimiza con Adam y calcula la pérdida con BCELoss y finalmente predice un ejemplo



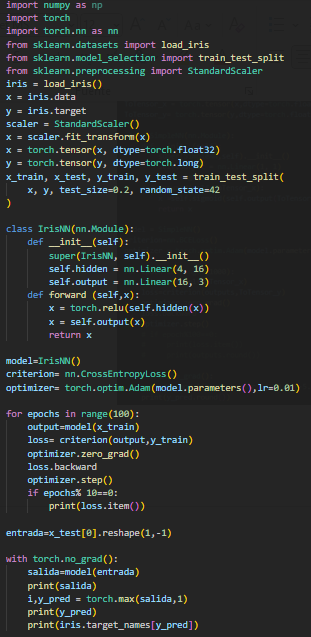
Tarea 6

El código codifica variables categóricas y ordinales prepara datos para PyTorch crea una red neuronal con capa oculta y salida con sigmoid usa BCELoss y Adam para entrenar predice probabilidades y muestra resultados redondeados junto con etiquetas reales.



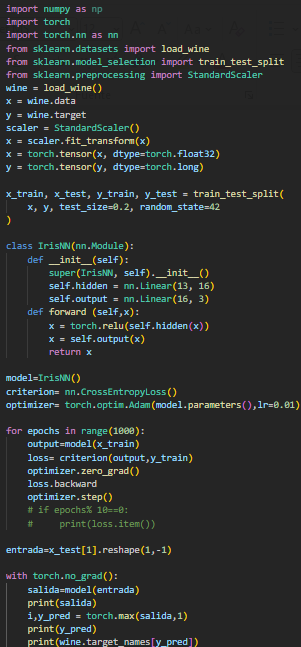
Tarea 7

El código carga Iris, escala datos, crea red neuronal con capa oculta usa ReLU y CrossEntropyLoss entrena con Adam durante 100 épocas imprime pérdida cada 10 épocas y predice clase para una muestra mostrando su nombre corregir faltan paréntesis en backward y step.



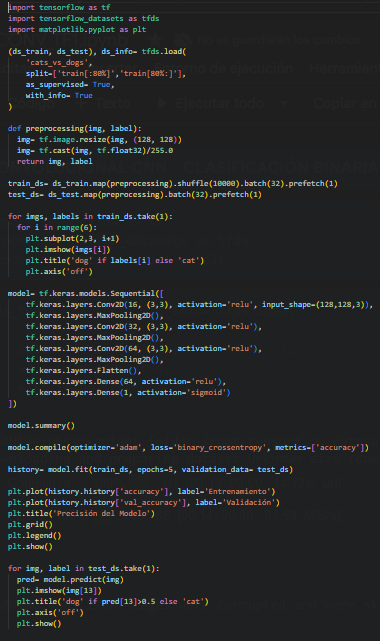
Tarea 8

El código carga y escala datos de vino crea red neuronal con capa oculta usa ReLU y CrossEntropyLoss entrena con Adam durante 1000 épocas pero faltan paréntesis en loss.backward() y optimizer.step() impidiendo entrenamiento correcto y predicción final.



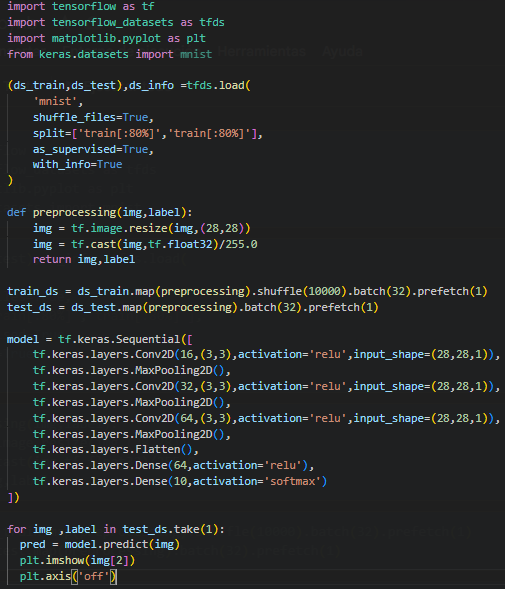
Tarea 9

El código carga y preprocesa cats\_vs\_dogs crea CNN con capas convolucionales y pooling entrena con Adam y binary crossentropy durante cinco épocas grafica precisión y muestra predicción para una imagen etiquetando como perro o gato según el resultado.



Tarea 10

El código carga MNIST mal divide datos usa mismo split sin canal extra en imágenes no compila ni entrena modelo y predice sin ajustar capas Conv2D ni mostrar imagen con escala de grises faltan pasos para entrenamiento y evaluación correcta.



Tarea 11

El código carga imágenes en escala de grises de FER\_2013 desde carpetas prepara datasets normalizando crea CNN con dos capas convolucionales y pooling capa densa y salida softmax usa sparse\_categorical\_crossentropy entrena 15 épocas predice y muestra una imagen con su etiqueta y grafica precisión entrenamiento y validación.

