Universidad de SanAndrés

I302 - Aprendizaje Automático y Aprendizaje Profundo

1er Semestre 2025

Trabajo Práctico 4

Fecha de entrega: Viernes 30 de mayo, 23:59 hs.

Formato de entrega: Los archivos desarrollados deberán entregarse en un archivo comprimido (.zip) a través del Campus Virtual, utilizando el siguiente formato de nombre: *Apellido_Nombre_TP4.zip*. Se aceptará únicamente un archivo por estudiante y debe contener por lo menos los siguientes elementos:

```
Apellido_Nombre_TP4.zip/
|- data/
|- Apellido_Nombre_Informe_TP4.pdf
|- Apellido_Nombre_Notebook_TP4.ipynb
```

- **Informe:** Debe incluir todos los aspectos teóricos, decisiones metodológicas, visualizaciones, análisis y conclusiones. El objetivo es que el informe contenga toda la explicación principal del trabajo. Se puede hacer referencia al notebook con frases como "Ver sección X del notebook para la implementación". El informe debe entregarse utilizando el archivo `template_informe.tex` provisto y no debe exceder las 10 páginas.
- **Notebook:** Debe contener el código utilizado, experimentos, análisis exploratorio, gráficos y el proceso completo de procesamiento y modelado. Sirve como respaldo técnico del informe y debe estar ordenado y bien documentado. Se recomienda modularizar el código en archivos `.py` cuando sea posible.

Trabajo Práctico 4: Aprendizaje No-Supervisado

1. **Clustering de datos.** Para el dataset *clustering.csv* realizar los siguientes análisis:

   a) Implementar el algoritmo K-means y determinar la cantidad de clusters con el método de "ganancias decrecientes" (graficar L vs. K, y elegir un valor K donde al aumentar K deje de reducir significativamente L, donde L es la suma de las distancias). Graficar el conjunto de datos x_i mostrando a qué cluster pertenece cada dato (usando colores/marcadores distintos para cada cluster) y también mostrar el centroide de cada cluster.

   b) Implementar el algoritmo Gaussian Mixture Model (GMM) y realizar la misma tarea que en el inciso anterior. Recuerde que puede inicializar la optimización de GMM con una corrida de K-means.

   c) Implementar el algoritmo DBSCAN y aplicarlo al conjunto de datos. Explorar el efecto de variar los parámetros ϵ (radio de la vecindad) y K (mínimo número de puntos en una zona densa). Luego, elegir una combinación razonable de ϵ y K y graficar los datos mostrando a qué cluster pertenece cada uno, utilizando colores/marcadores distintos para cada cluster/ruido.

2. **Reducción de dimensionalidad.** Este problema se basará en el dataset *MNIST_ dataset.csv*, que contiene representaciones tabulares de imágenes de dígitos del 0 al 9. Originalmente, cada imagen tiene una resolución de 28x28 píxeles en escala de grises. En este conjunto de datos, cada imagen se representa como una fila de 784 (28x28) valores, donde cada valor representa la intensidad de gris de un píxel en la imagen.

   a) Implementar Principal Component Analysis (PCA) y aplicarlo al conjunto de datos. Graficar cómo varía el error cuadrático medio de reconstrucción sobre el conjunto de datos en función de la cantidad de componentes principales utilizadas.

   b) Seleccionar la cantidad de componentes principales que considere adecuada y justifique la elección. Usando dicha cantidad de componentes, graficar las imagenes de los digitos originales y reconstruidos para las primeras 10 muestras del dataset.

   c) OPCIONAL: Entrenar un modelo de autoencoder variacional (VAE) utilizando la librería PyTorch para armar y entrenar las redes neuronales involucradas (la red de encoder y la de decoder). Recuerde dividir el conjunto de datos en dos subconjuntos: entrenamiento y validación. El subconjunto de entrenamiento se empleará para entrenar el VAE, mientras que el de validación servirá para ajustar los hiperparámetros y evaluar el error de reconstrucción. Una vez desarrollado el VAE, compare la calidad de las imágenes reconstruidas con las obtenidas mediante PCA en el inciso anterior, utilizando 10 imagenes tomadas aleatoriamente del conjunto de validación del VAE.