Universidad de
SanAndrés

I302 - Aprendizaje Automático y Aprendizaje Profundo

2do Semestre 2024

Trabajo Práctico 5

Fecha de entrega: Viernes 8 de noviembre, 23:59 hs.

Formato de entrega: Los archivos desarrollados deben ser entregados en un archivo comprimido .zip a través del Campus Virtual, utilizando el siguiente formato de nombre de archivo: *Apellido_Nombre_TP5.zip*. Se aceptará únicamente 1 archivo por estudiante. En caso de que el nombre del archivo no cumpla con la nomenclatura especificada, el trabajo no será corregido.

La carpeta comprimida deberá constar de N sub-carpetas, una por cada problema del TP (es decir, cada problema tiene su sub-carpeta denominada "Problema N"). Dentro de cada sub-carpeta deberá incluir un Jupyter Notebook llamado *Entrega_Problema_N.ipynb* en el cual se den las respuestas a los incisos del problema y se muestren los gráficos resultantes. Puede agregar resultados o análisis adicionales si lo considera necesario. Se recomienda fuertemente no realizar todo el desarrollo dentro del Jupyter Notebook; en su lugar, se sugiere usar archivos .py para desarrollar el código, siguiendo las buenas prácticas de programación y modularización vistas en clase.

---

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.