Universidad Tecnológica Centroamericana
Facultad de Ingeniería

CC414 - Sistemas Inteligentes
Docente: Kenny Dávila, PhD

Tarea #4 (3% Puntos Oro)

Para completar esta tarea es requerido usar **Python 3** y la librería **Scikit-Learn**. También se recomienda el uso de las librerías **Numpy** para manejar los datos y Matplotlib para la creación de plots con los datos resultantes.

Parte 1. Clustering (2.0)

Se le entregan 3 diferentes datasets (datos_1.csv, datos_2.csv, datos_3.csv). El objetivo de este ejercicio es experimentar con diferentes algoritmos de clustering y comparar sus resultados en cada uno de ellos. En particular se le pide utilizar los algoritmos K-means, Agglomerative Clustering ("Ward"), y DBScan. Nótese que diferentes algoritmos de clustering requieren el uso de diferentes parámetros para obtener mejores resultados. En este ejercicio se le pide experimentar con estos valores e intentar encontrar los clusters ideales en cada dataset:

a) Para K-means, se le pide experimentar con diferentes valores de K. Como mínimo, debería probar con $1 \leq K \leq 5$. En total son **al menos 5 posibles configuraciones** que debe probar con este algoritmo. Puede probar otros valores si considera que los resultados no son satisfactorios.
b) Para Clustering jerárquico o aglomerativo, se le pide usar "Ward", y deberá experimentar con valores de K y con diferentes valores en el umbral de distancia. Nótese que si usa umbral de distancia entonces el valor K debe ser None y viceversa ya que solamente se puede usar un número fijo de clústeres o una distancia máxima, pero no ambos criterios al mismo tiempo. Similar a K-means, se le recomienda experimentar con valores K ($1 \leq K \leq 5$), al menos 5 valores. En el caso de la distancia, es importante notar que **NO todos los datasets están en la misma escala**. Por lo tanto, los valores que funcionen bien para un dataset, no necesariamente producirán buenos resultados en otros datasets. Se le pide considerar **al menos** 5 posibles valores por dataset. Posiblemente necesite explorar aún más de 5. En total son **al menos 10 posibles configuraciones** que debe probar con este algoritmo (**5+ valores K y 5+ umbrales de distancia**).
c) Para DBScan, debe probar con diferentes valores de distancia entre vecinos (eps) y diferentes valores del mínimo de muestras por vecindario (min_samples={5,10,15}). Igual que en el caso del Clustering jerárquico, el valor eps ideal cambia según la escala del dataset y los mejores valores eps para un dataset no necesariamente serán buenos en otros datasets. Debe experimentar con al menos 5 valores para eps. En total son **al menos 15 posibles configuraciones** ({5+ valores eps} X {3+ valores min_samples}) que debe probar con este algoritmo.

Para cada configuración de cada algoritmo, se le pide generar un scatter plot (puede usar matplotlib), donde se pueda observar los resultados del algoritmo de clustering. Su objetivo es **analizar** dichos resultados visualmente y **determinar cuál fue la combinación de parámetros que produjo los**

resultados más satisfactorios. **Si ninguna de las configuraciones propuestas aquí parece dar resultados satisfactorios, debe intentar algunas configuraciones adicionales**. Para obtener el puntaje completo, **debe seleccionar solamente un plot por cada algoritmo por cada uno de los datasets**. En Total, deberá proveer exactamente **9 scatter plots** (3 algoritmos x 3 datasets) **en su reporte**.

Posteriormente, se le pide proveer un breve análisis sobre cada uno de los datasets provistos:

1. ¿Qué tipo de clustering considera que funciona mejor en cada dataset y por qué?
2. ¿Cuántas clases reales cree que se usaron para generar los datos en cada dataset?

Parte 2. K-NN (1.0)

En esta parte se desea utilizar el clasificador de K vecinos mas cercanos (K-NN) para resolver el problema de clasificación planteado en la tarea 3. Dicho problema intenta determinar de manera automática la cadena de hamburguesas preferida por una persona en base a un perfil de gustos. Se consideran 4 tipos cadenas de restaurantes: Mendys, Wac Ronalds, Burger Queen, y Rigos.

Nótese que será necesario que convierta los atributos binarios en números (0 y 1s) antes de poder usar el K-NN sobre estos datos. Otros atributos con más de 2 valores necesitan codificarse usando múltiples atributos binarios.

Debe utilizar los mismos datos de entrenamiento (4 conjuntos diferentes) para entrenar diferentes modelos del K-NN, con los valores de K = {1, 3, 5, 7, 9, 11, 13, 15}. Se espera que se explore al menos **un total de 32 configuraciones** de manera sistemática (4 datasets de entrenamiento x 8 valores k). Otros parámetros deben dejarse en sus valores por defecto. Deberá utilizar los datos de validación ("validation_data.csv") para calcular y reportar las mismas métricas solicitadas para los árboles de decisión y los bosques aleatorios. Debe usar una sola tabla para reportar dichos resultados. Debe utilizar **al menos 4 cifras significativas** por cada valor provisto.

Tabla de resultados de ejemplo (K-NN)

Train Dataset	K	Train Acc.	Val. Acc.	Val. Avg Rec.	Val. Avg Prec.	Val. Avg F1	Time Train	Time Val
Small	1	XX.XX%	XX.XX%	XX.XX%	XX.XX%	XX.XX%	T s	T s
…	….	…	…	…	…	…	…	…
Very Large	15	XX.XX%	XX.XX%	XX.XX%	XX.XX%	XX.XX%	T s	T s

Requisitos

1. Implementación de un script basado en Python que recibe como entrada el nombre del archivo de datos y otros parámetros del clustering K-means y entrene un K-means y grafique los resultados del clustering.

2. Implementación de un script basado en Python que recibe como entrada el nombre del archivo de datos y otros parámetros del clustering Jerárquico y entrene dicho algoritmo y grafique los resultados del clustering.

3. Implementación de un script basado en Python que recibe como entrada el nombre del archivo de datos y otros parámetros del clustering DB-SCAN y entrene dicho algoritmo y grafique los resultados del clustering.

4. Implementación de un script basado en Python que recibe como entrada el nombre del archivo de datos de entrenamiento y otros parámetros del K-NN y entrene clasificador K-NN que deberá ser guardado en un archivo de salida.

5. Implementación de un script basado en Python que recibe como entrada el nombre del archivo donde se guardó el clasificador (K-NN), y el nombre del archivo con los datos de prueba, y que produzca como salida una matriz de confusión y las métricas correspondientes sobre los datos de prueba. Note que gracias al "duck typing" de Python, no necesitara hacer nada especial para que el mismo código funcione bien con un tipo de clasificador o con el otro. **Es válido re-utilizar el mismo script de evaluación de la tarea 3**.

6. Un reporte que detalle lo siguiente:
    a. **Introducción**. Debe incluir una breve motivación del problema en sus propias palabras
    b. **Implementación.** Descripción de su implementación incluyendo comandos específicos para ejecutar su código fuente y librerías específicas de Python que son requeridas por su programa.
    c. **Resultados Clustering.**
        i. Debe proveer las ilustraciones de las mejores configuraciones por cada clasificador por cada dataset.
        ii. Contestar a las preguntas mencionadas anteriormente.
    d. **Resultados K-NN.** Debe proveer las tablas con todos los resultados que se obtuvieron de sus experimentos con diferentes datos de entrenamiento y configuraciones de cada clasificador.
        i. Análisis del efecto del valor de K en cuanto a rendimiento y overfitting.
        ii. Análisis del efecto del tamaño del dataset en cuanto a rendimiento y overfitting.
        iii. Comparación contra los resultados de Arboles de Decision y Random Forest.
        iv. Sorpresas. Detalle si los resultados anteriores fueron consistentes o no con sus propias expectativas sobre el tema.
    e. **Dificultades encontradas.** Describa en sus propias palabras cuales fueron los retos más difíciles al momento de implementar el algoritmo.
    f. **Planteamiento**. Debe presentar un problema distinto donde usted aplicaría K-NN, con atributos que podría colectar sobre ese problema y una corta justificación (Mínimo 300-Máximo 500 palabras).
    g. **Conclusiones**

Otras políticas

1. Esta tarea deberá trabajarse y entregarse **individual** o en **parejas.**
2. La entrega será **un solo archivo comprimido (.zip o .rar)**. Dentro de dicho archivo debe contener la guía completada **en formato PDF**. También **debe incluir los scripts de Python** que se usaron para contestar cada punto.
3. Favor **NO incluir entornos virtuales ni datasets dentro** de su archivo Zip. La entrega final no debería pasar de 10MB.
4. No se permite el uso de repositorios públicos.
5. Si se hace en parejas, **ambas personas deben subir el mismo archivo**. Si solamente una persona lo sube, **es posible que solamente esta persona reciba una calificación.**
6. El **plagio** será penalizado de manera severa.
7. Los estudiantes que entreguen una tarea 100% original recibirán una nota parcial a pesar de errores existentes. En cambio, los estudiantes que presenten tareas que contenga material plagiado recibirán 0% automáticamente independientemente de la calidad.
8. Tareas entregadas después de la fecha indicada solamente podrán recibir la mitad de la calificación final. Por esta razón, es posible que **un trabajo incompleto pero entregado a tiempo termine recibiendo mejor calificación que uno completo entregado un minuto tarde**.