Instituto Tecnológico de Costa Rica
Escuela de Ingeniería Electrónica
Profesor: Dr. Pablo Alvarado Moya
EL 5857 Aprendizaje Automático
I Semestre 2024

Entrega: **14 de mayo, 2024**

Tarea 6: Reducción de dimensiones, aglomeración y modelos generativos

I. Introducción

En esta tarea se utilizarán las bibliotecas Scikit-Learn, y UMAP/PaCMAP para explorar la estructura de conjuntos de datos médicos. En particular se usará el conjunto de diagnósitico de cancer de mama de Wisconsin (UCI Machine Learning), disponible originalmente en este sitio, pero también disponible en kaggle, y el conjunto de datos de predicción de insuficiencia cardiaca, disponible en kaggle también.

En el conjunto de diagnóstico de cancer de mama las 30 características han sido calculadas para 569 imágenes digitales de un aspirado con aguja fina (FNA) de una masa mamaria. Estas describen aspectos de los núcleos celulares presentes en imágenes similares a la mostrada en la figura 1.

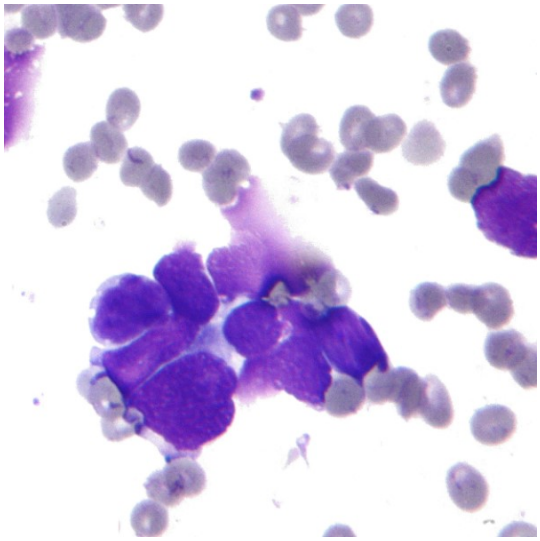

Figura 1: Imagen muestra del set de datos de diagnóstico de cancer de mama [tomado de kaggle].

En la matriz de datos, la primera columna corresponde a un número de identificación de la paciente. La segunda columna corresponde al diagnóstico, es decir, a si el tejido es benigno o maligno. Las columnas de la 3 a la 12 corresponden a la media de atributos de los núcleos celulares (como radio, desviación estándar de los niveles de gris, perímetro, área, etc.). Adicionalmente se agregan otras 10 columnas con las desviaciones estándares y otras 10 con las medias de los 3 valores más grandes de esos atributos, por imagen.

De la descripción del segundo conjunto de datos se extrae: las enfermedades cardiovasculares (ECV) son la primera causa de muerte en el mundo, y se calcula que se cobran 17,9 millones de vidas al año, lo que supone el 31 % de todas las muertes en el mundo. Cuatro de cada cinco muertes por ECV se deben a infartos de miocardio y accidentes cerebrovasculares, y un tercio de estas muertes se producen prematuramente en personas menores de 70 años. La insuficiencia cardíaca es un evento común causado por las ECV. Este conjunto de datos contiene 918 entradas de 11 características,

que pueden utilizarse para predecir una posible enfermedad cardíaca. Puesto que algunas de las columnas corresponden a caracterísiticas categóricas, estas deben convertirse a codificación *one-hot* para no afectar su interpretación numérica. Eso ya se hace en el código provisto, por lo que la dimensión final de los vectores es mayor a 11.

Las personas con enfermedades cardiovasculares o que tienen un alto riesgo cardiovascular (debido a la presencia de uno o más factores de riesgo como la hipertensión, la diabetes, la hiperlipidemia o una enfermedad ya establecida) necesitan una detección y una gestión tempranas en las que un modelo de aprendizaje automático puede ser de gran ayuda.

En esta tarea intentaremos visualizar qué tan factible es la clasificación para ambos casos.

La invitación al repositorio del GitHub Classroom con el código base para la tarea es esta. Note que allí ya se encuentran los datos. En esta ocasión utilizaremos cuadernos de Jupyter, como alternativa interactiva de Python.

II. Procedimiento

Los siguientes pasos deben hacerse para los dos conjuntos de datos por separado.

1. Debido a que la distribucion de las características es muy variada, el primer paso es, como siempre, normalizar los datos.

2. Aplique PCA para proyectar los datos al subespacio de dos dimensiones con los dos componentes principales de mayor varianza.

3. Visualice los datos proyectados, utilizando las etiquetas en la segunda columna para identificar las dos clases. Observe que el PCA, como tal, es un método no supervisado, pero en la visualización queremos ver estructuras asociadas a las clases de interés, y por eso, solo para la visualización, usamos la etiqueta de clases.

4. Revise los primeros dos componentes principales (es decir, los dos eigenvectores con los mayores eigenvalores). ¿Es posible identificar si hay ejes en el espacio original más importantes que otros? ¿Es posible visualizar conglomerados en el subespacio bidimensional encontrado con PCA?

5. Repita lo anterior con UMAP, en su versión no supervisada (ver sitio de autores para instrucciones de cómo instalarlo). Experimente con los parámetros de UMAP para lograr una visible separación de las clases.

   Como alternativa más rápida y configurable, puede utilizar el código de Andrew Draganov.

6. Repita lo anterior con la versión supervisada de UMAP, para ayudar al método a encontrar estructuras más afines a un problema de clasificación como tal.

7. De los experimentos anteriores, analice si se pueden observar conglomerados aparentes, y qué número de ellos podría ser interesante evaluar.

8. Utilice k-means para encontrar clústers en el espacio original de características, donde k es el número de clústers que, en el punto anterior, usted estima que hay.

9. Visualice los clúster encontrados con la técnica de su preferencia. Es decir, repita los experimentos anteriores, pero en vez de usar la clase benigno/maligno o insuficiencia/saludable, use el número de clúster para determinar el color de los puntos.

   Se recomienda implementar la función de visualización en un archivo de Python (`.py`) que usted entonces importa en cada cuaderno que lo necesite. Esa función se usaría, entonces, ya sea con las etiquetas del conjunto de datos o con las etiquetas de los clúster encontrados. Así evita replicar el código y tener que dar mantenimiento a múltiples versiones.

10. Como último punto, construya un clasificador generativo, utilizando una mezcla de gaussianas (`GaussianMixture`) para representar la distribución de densidad de cada una de las clases $p(\underline{\boldsymbol{x}}|y)$. Ese método utiliza el algoritmo de esperanza-maximización para encontrar las distribuciones requeridas.

    Con los modelos de cada clase, seleccione como clase ganadora aquella que conduce a la mayor probabilidad *a posteriori* $p(y|\underline{\boldsymbol{x}})$.

    Debe utilizar algún método de selección de modelos (como por ejemplo el descrito aquí) para encontrar cuántas gaussianas aproximan bien la distribución de densidad probabilística de cada clase.

    Debe partir sus datos en: datos de entrenamiento, de validación y de prueba, para esta parte, y utilizarlos como corresponde.

III. Entregables

1. Archivos de Python.
2. Archivo README con instrucciones de cómo ejecutar el código.
3. Archivo PDF con las gráficas generadas y el análisis de resultados.
4. Favor subir la última versión del código al tecDigital.
5. El trabajo en Git se tomará en cuenta y la versión en la rama `master` o `main` deberá corresponder con la versión entregada. Marque esa versión con un `tag` adecuado.

IV. Notas

- Esta tarea se resuelve con los mismos grupos de las otras tareas. Si existe modificación en la integración los grupos, favor reportarlo al profesor para indicarlo en los grupos del tecDigital.