2025/2026 1Q

Lista de Problemas 4

APA

Javier Béjar

Departament de Ciències de la Computació

Grau en Enginyeria Informàtica - UPC

UPC FIB Facultat d'Informàtica de Barcelona

UNIVERSITAT POLITÈCNICA DE CATALUNYA

DEPARTAMENT DE CIÉNCIES DE LA COMPUTACIÓ

FACULTAT D'INFORMÁTICA DE BARCELONA

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Primera edición, septiembre 2021

Esta edición, Septiembre 2025

Instrucciones:

Para la entrega de grupo debéis elegir un problema del capítulo de problemas de grupo.

Para la entrega individual debéis elegir un problema del capítulo de problemas individuales. **Cada miembro del grupo debe elegir un problema diferente**.

Debéis hacer la entrega subiendo la solución al racó.

Evaluación:

La nota de esta entrega se calculará como 1/3 de la nota del problema de grupo más 2/3 de la nota del problema individual.

Puntuación:

En esta segunda lista, los errores leves se penalizarán con 2 puntos, los errores graves, sobre todo metodológicos, se penalizarán con 8 puntos. Explicaciones demasiado breves sobre lo que habéis hecho y los resultados también reducirán la nota. No hagáis una resolución mecánica de los problemas.

Al realizar el informe correspondiente a los problemas explicad los resultados y las respuestas a las preguntas de la manera que os parezca necesaria. Se valorará más que uséis gráficas u otros elementos para ser más ilustrativos.

Entregad los resultados como un notebook (Colab/Jupyter). Podéis poner las respuestas a las preguntas en el notebook, este os permite insertar texto en markdown y en latex.

Aseguraos de que los notebooks mantienen la solución que habéis obtenido, no los entreguéis sin ejecutar.

Objetivos de aprendizaje:

1. Conocer el análisis de problemas usando máquinas de soporte vectorial, árboles de decisión y conjuntos de clasificadores
2. Interpretar modelos de árboles de decisión
3. Usar técnicas de relevancia de atributos sobre clasificadores no lineales

CAPÍTULO 1

Problemas de Grupo

Al resolver el problema explicad bien los que hacéis, no hacer ningún comentario o hacer comentarios superficiales tendrán una nota más baja.

Tenéis que mostrar que habéis entendido los métodos que estáis aplicando, así que un corta y pega de problemas similares no es suficiente.

Para obtener los datos para algunos de estos problemas necesitaréis instalaros la última versión de la librería `apafib`. La podéis instalar localmente haciendo:

```
pip install -user -upgrade apafib
```

Para usar las funciones de carga de datos solo tenéis que añadir su importación desde la librería, en vuestro script o notebook, por ejemplo

```
from apafib import load_fraude
```

La función os retornará un `DataFrame` de Pandas o arrays de numpy con los datos y la variable a predecir, cada problema os indicará que es lo que obtendréis.

1. **Fraude detectado**

   Una aplicación compleja en el área de comercio electrónico es la detección de usos fraudulentos de tarjetas robadas. El volumen de transacciones fraudulentas suele ser pequeño comparado con el volumen total de transacciones y es obviamente más importante que se escape el menor número de ellas. Vamos a trabajar con una parte de conjunto de datos Credit Card Fraud data[1] Este conjunto de datos tiene diferentes atributos correspondientes a la transacción, la persona que ha hecho la compra y el lugar de compra.

   Podéis obtener estos datos mediante la función `load_fraude` de la librería `apafib`. Resolved los siguientes apartados ilustrando los resultados de la manera que os parezca adecuada.

[1] Podéis encontrar los datos originales aquí https://www.kaggle.com/datasets/neharoychoudhury/credit-card-fraud-data.

a) El primer paso es preprocesar y preparar los datos antes de ajustar cualquier modelo. Hay algunas variables que no son útiles para el problema o que no tiene sentido usar. Eliminadlas del conjunto de datos, pero eliminad solo las más obvias, si quitáis alguna más lo deberéis justificar. El conjunto de datos tiene una mezcla de atributos fecha, categóricos y numéricos, transformad los categóricos a numéricos. Podéis calcular algunas variables derivadas de las variables fecha que puedan tener sentido para los datos. Partid los datos en conjunto de entrenamiento y test (70 %/30 %).

b) Comenzaremos con un modelo de predicción base ajustando una regresión logística. Fijaos que los que nos interesa es que la mejor predicción sea para la clase *fraude*. Evaluad el modelo. Este modelo puede penalizar los errores en las clases minoritarias usando el parámetro `class_weight`. Usad el valor `balanced` y comprobad si eso mejora el modelo. Comprobad las curvas ROC.

c) Nos interesará también entender qué delata una transacción como fraudulenta. Podemos ajustar un árbol de decisión para obtener un modelo interpretable. Ajustad este modelo explorando adecuadamente sus hiperparámetros. Evaluad la calidad del modelo y comparadla con el anterior. Representad el árbol de decisión, ¿es suficientemente simple para saber cuándo una transacción es fraudulenta?

d) Buscamos la máxima precisión en la clasificación de transacciones fraudulentas. Ajustad un random forest y un gradient boosting explorando adecuadamente sus hiperparámetros. Comparad los resultados con los modelos anteriores.

e) Las SVM se benefician de datos con alta dimensionalidad, ajustad una SVM lineal, una con kernel cuadrático y otra con kernel RBF explorando adecuadamente sus hiperparámetros. Tendréis que hacer que los modelos predigan probabilidades para poder ver la curva ROC. Comparad los resultados con los modelos anteriores.

f) Como hemos visto en teoría, cuando tenemos varios modelos podemos combinarlos usando diferentes estrategias. Entrenad un `StackedRegressor` y un `VotingRegressor` combinando los tres mejores modelos que habéis encontrado en los apartados anteriores con sus mejores hiperparámetros. ¿Es mejor alguno de estos modelos combinados?

g) Calculad la *permutation importance* de los atributos sobre el test para el mejor modelo que habéis encontrado excepto el árbol de decisión ¿coincide la importancia de los atributos con los que considera el árbol en sus decisiones?

2. **Diagnostico del Alzheimer**

Con el envejecimiento de la población, la enfermedad de Alzheimer es una de las enfermedades del envejecimiento que ha cobrado mayor relevancia. Es por eso que un diagnóstico a partir de las características del paciente, pruebas diagnósticas y test de evaluación es una manera simple de obtener una aproximación del diagnóstico. Vamos a usar una parte del conjunto de datos Alzheimer's Disease Dataset[2].

Podéis obtener estos datos mediante la función `load_alzheimer` de la librería `apafib`. Resolved los siguientes apartados ilustrando los resultados de la manera que os parezca adecuada.

a) Los datos están ya preprocesados, veréis que hay variables binarias/categóricas y variables continuas, podéis eliminar las variables identificadoras y las que no tienen información útil. Estamos interesados en la capacidad para diagnosticar de los diferentes tipos de variables, así que analizaremos tres conjuntos de datos, uno solo con las variables binarias/categóricas, otro con las continuas y otro con todas las variables. Partid los tres

[2]Podéis encontrar los datos originales aquí https://www.kaggle.com/datasets/rabieelkharoua/alzheimers-disease-dataset.

conjuntos de datos en entrenamiento y test (70 %/30 %) de manera que cada uno tenga los mismos ejemplos.

b) Empezaremos usando un modelo lineal. Ajustad una regresión logística a cada conjunto de datos explorando adecuadamente sus hiperparámetros. Comparad los modelos. Mirad el peso que les asigna cada modelo a los atributos y comprobad que el orden de importancia que tienen las variables en los modelos con los subconjuntos de variables corresponde con el orden que tienen en el modelo con todos los datos.

c) Para poder entender bien la relación entre las variables y cómo permiten el diagnóstico podemos usar un árbol de decisión. Ajustad este modelo a los tres subconjuntos de datos y visualizad los árboles resultantes. Comparad los atributos que aparecen en los primeros niveles en los tres árboles.

d) La combinación de clasificadores permite tener una mejor precisión. Ajustad un random forest y un gradient boosting a los tres conjuntos de datos explorando adecuadamente sus hiperparámetros. Comparad sus resultados.

e) Calculad la *permutation importance* de los atributos sobre el test para el mejor modelo de combinación de clasificadores para todos los datos que habéis encontrado. Seleccionad un 25 %, 50 % y 75 % de los mejores atributos (siguiendo del orden de importancia) y ajustad al subconjunto de datos un árbol de decisión. Analizad qué atributos son los que usan los árboles en sus primeros niveles.

f) Como hemos visto en teoría, cuando tenemos varios modelos podemos combinarlos usando diferentes estrategias. Entrenad un `StackedClassifier` y un `VotingClassifier` usando los tres árboles de decisión que habéis ajustado en el apartado anterior. Comparad este modelo con el árbol de decisión con todos los datos.

3. **Bicicletas de nuevo**

Este problema continúa la exploración de los datos de bicicletas compartidas del repositorio de UCI que está en la primera lista de problems, si hicisteis ese problema igual tenéis curiosidad sobre si modelos más complejos pueden obtener un mejor resultado. Este conjunto de datos recopila estadísticas agregadas de uso de bicicletas junto con otra información adicional relevante. El objetivo es obtener predicciones del uso del servicio de bicicletas compartidas en una ciudad. Vamos a trabajar con el conjunto de datos Se pueden descargar los datos desde aquí `https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset`.

El objetivo de este problema es predecir cuántas bicicletas se usarán al día siguiente a partir de los datos de días anteriores (usaremos el archivo `day.csv`). Podéis leer en el `Readme.txt` los detalles sobre las variables.

a) El primer paso es preprocesar y preparar los datos antes de ajustar cualquier modelo. Hay algunas variables que no son útiles para el problema o que no tiene sentido usar. Eliminadlas del conjunto de datos y explicad por qué las eliminái s.

Necesitaremos obtener variables que nos permitan predecir a partir de la historia del sistema. Tal como están los datos no podemos hacer eso, por lo que necesitaremos un poco de preproceso. La librería `pandas` permite generar una copia de una tabla de datos desplazada una serie de instantes temporales usando el método `shift`. Generad una copia de los datos desplazada un día, dos y tres días y añadidlas como nuevas columnas (eliminad los datos perdidos que se generan). Partid la tabla de datos en las columnas que usaremos para predecir el número de bicicletas que se usarán en un día concreto y añadid las variables del día actual que sabremos

Dado que tenemos que predecir el futuro vamos a seleccionar los primeros 500 ejemplos para entrenamiento y el resto para test. Estandarizad los datos antes de generar los modelos. Eliminad las ventanas del conjunto de test que comparten datos con el conjunto de entrenamiento.

b) Para tener un modelo base usaremos una regresion LASSO. Ajustad este modelo adecuadamente y calculad la *calidad* del modelo empleando validación cruzada y con los datos de test. Para hacer la validación cruzada tenéis que usar el método `TimeSeriesSplit` con 5 particiones en el parámetro `cv` de la validación. Calculad el error de validación cruzada y el del test con el mean absolute error. Representad los valores reales del test contra las predicciones.

c) Empezaremos por el modelo no lineal más sencillo. Ajustad un árbol de regresión a los datos explorando sus hiperparámetros de manera adecuada. Compara los resultados con los del modelo anterior. ¿Son las variables con mayor importancia en el modelo LASSO las que el árbol de decisión usa en sus primeros niveles?

d) La combinación de clasificadores puede permitir obtener un mejor ajuste. Entrenad un random forest y un gradient boosting explorando adecuadamente sus parámetros. Comparad la calidad de estos modelos con los anteriores.

e) Las máquinas de soporte vectorial de regresión permiten trabajar en un espacio de alta dimensionalidad que tiene en cuenta interacciones entre variables. Ajustad primero una SVM lineal y comparad sus pesos con los de la regresión LASSO ¿es el orden de la importancia asignada por los pesos similar? Ajustad SVM con kernel polinómico y kernel RBF explorando adecuadamente los parámetros. Comparad la calidad de estos modelos con los otros.

f) Como hemos visto en teoría, cuando tenemos varios modelos podemos combinarlos usando diferentes estrategias. Entrenad un `StackedRegressor` y un `VotingRegressor` usando el mejor modelo de cada tipo que habéis ajustado en los apartados anteriores (LASSO, combinación de clasificadores, SVM). Comparad los resultados.

CAPÍTULO 2

Problemas Individuales

Al resolver el problema explicad bien los que hacéis, no hacer ningún comentario o hacer comentarios superficiales tendrán una nota más baja.

Tenéis que mostrar que habéis entendido los métodos que estáis aplicando, así que un corta y pega de problemas similares no es suficiente.

Para obtener los datos para estos problemas necesitaréis instalaros la última versión de la librería `apafib`. La podéis instalar localmente haciendo:

```
pip install -user -upgrade apafib
```

Para usar las funciones de carga de datos solo tenéis que añadir su importación desde la librería, en vuestro script o notebook, por ejemplo

```
from apafib import load_BCN_vuelos
```

La función os retornará un `DataFrame` de Pandas o arrays de numpy con los datos y la variable a predecir, cada problema os indicará que es lo que obtendréis.

1. **De la IA no se fia**

   El centro de investigaciones sociológicas (CIS) es un organismo que tiene por finalidad el estudio científico de la sociedad y es una gran fuente de datos. Entre los diferentes estudios que se realizaron durante este año hubo uno sobre la percepción de la sociedad de la inteligencia artificial preguntando sobre su conocimiento sobre el tema, sus efectos y consecuencias en la sociedad y en el mercado de trabajo y en los dilemas éticos que supone. La clave de las variables y su significado la puedes encontrar en la página web de los problemas.

   Trabajaremos con una versión reducida de este conjunto de datos que puedes obtener mediante la función `load_CIS_IA` de la librería `apafib` esta retornará un dataframe de Pandas. Resuelve los siguientes apartados ilustrando los resultados de la manera que te parezca más adecuada.

   a) Divide el conjunto de datos en entrenamiento y test (80 %/20 %). Explorando los datos en-

contrarás que hay bastantes valores perdidos. Unas pocas variables son categóricas, que se pueden imputar usando la moda de la variable y transformar mediante *one hot encoding*. La mayoría de las variables corresponden a una valoración numérica discreta, o son directamente valores continuos que se puede imputar usando `KNNImputer`

Una vez tenemos el conjunto de datos preprocesado puedes aplicar adecuadamente PCA para ver su comportamiento. Analiza sus resultados y representa la variable objetivo sobre el PCA en dos dimensiones. Explica lo que has observado. ¿Podrían ser las clases linealmente separables?

b) Vamos a establecer un nivel base de predicción, dado que las variables ahora son todas numéricas, ajusta un naive bayes gausiano a los datos. Evalúa adecuadamente la calidad del modelo.

c) Un modelo lineal es directamente interpretable a partir de sus pesos. Ajusta una regresión logística y una SVM lineal explorando adecuadamente sus hiperparámetros. Evalúa la calidad de los modelos y compara la importancia que asignan los modelos a los atributos para cada clase.

d) Los conjuntos de clasificadores permiten adaptarse mejor a la complejidad de los datos. Ajusta un Gradient Boosting y un Exta Trees classifier a los datos explorando adecuadamente sus hiperparámetros. Compara la calidad de los modelos con los anteriores. Calcula la *permutation importance* sobre el test para determinar qué atributos son más importantes para predecir. Compara los atributos más importantes con los de los apartados anteriores. ¿Hay cierto acuerdo entre los modelos respecto a cuáles son los atributos más importantes?

e) Como hemos visto en teoría, cuando tenemos varios modelos podemos combinarlos usando diferentes estrategias. Entrena un `StackedClassifer` usando los tres mejores modelos de ente los dos modelos lineales y los dos de combinación de árboles de decisión usando sus mejores hiperparámetros ¿Es mejor este modelo combinado? Calcula la *permutation importance* sobre el test para determinar qué atributos son más importantes. Compara los atributos más importantes con los de los apartados anteriores.

2. **Esa música me hace sentir ...**

   El poder clasificar la música de manera automática a partir de las caracterísiticas de la onda sonora tiene aplicaciones tanto para organizarla de manera coherente respecto a ciertos atributos subjetivos difíciles de medir, como para personalización en aplicaciones de streaming. Este conjunto de datos de UCI[1] plantea el clasificar canciones a partir de caracterísiticas espectrales del sonido en cuatro emociones (relajado, triste, feliz, exaltado). El objetivo es tener un modelo que pueda clasificar canciones de acuerdo con estas emociones.

   Puedes obtener mediante la función `load_musica` de la librería `apafib` esta retornará un dataframe de Pandas. Resuelve los siguientes apartados ilustrando los resultados de la manera que te parezca más adecuada.

   a) Divide el conjunto de datos en entrenamiento y test (80 %/20 %). El conjunto de datos solo tiene variables continuas y no hay valores perdidos. Aplica reducción de dimensionalidad con PCA y t-SNE y representa la variable respuesta sobre la transformación, ¿parece haber alguna relación entre las variables con la variable respuesta? ¿Podrían ser las clases linealmente separables?

   b) Vamos a entrenar primero un par de modelos como base de la calidad que se puede obtener con estos datos. Ajusta una regresión logística y una SVM lineal explorando adecuadamente sus hiperparámetros. Evalúa la calidad de los modelos.

[1] Puedes encontrar la descripción de las variables aquí https://archive.ics.uci.edu/dataset/862/turkish+music+emotion.

c) Quizás introduciendo no linearidades en el modelo podremos obtener un mejor resultado. Entrena modelos SVM con kernel RBF ajustando adecuadamente sus hiperparámetros. Calcula la *permutation importance* sobre el test para determinar qué atributos son más importantes en el modelo para predecir. Compara los atributos más importantes con los de los apartados anteriores. ¿Hay cierto acuerdo entre los modelos respecto a cuáles son los atributos más importantes?

d) Los conjuntos de clasificadores permiten adaptarse mejor a la complejidad de los datos. Ajusta un random forest a los datos explorando adecuadamente sus hiperparámetros. Compara la calidad del modelo con los anteriores. Calcula la *permutation importance* sobre el test para determinar qué atributos son más importantes en el modelo para predecir. Compara los atributos más importantes con los de los apartados anteriores. ¿Hay cierto acuerdo entre los modelos respecto a cuáles son los atributos más importantes?

e) Como hemos visto en teoría, cuando tenemos varios modelos podemos combinarlos usando diferentes estrategias. Entrena un `StackedClassifer` usando la SVM RBF y el random forest combinado alternativamente con la regresión logística y la SVM linear usando sus mejores hiperparámetros ¿Es mejor alguno de estos modelos combinados? Calcula la *permutation importance* sobre el test para determinar qué atributos son más importantes en la mejor combinación. Compara los atributos más importantes con los de los apartados anteriores.

3. **La red es una jungla**

El centro de investigaciones sociológicas (CIS) es un organismo que tiene por finalidad el estudio científico de la sociedad y es una gran fuente de datos. Entre los diferentes estudios que se realizaron durante el año pasado, hubo uno sobre los riesgos de internet, hábitos de prevención y opiniones sobre la regulación sobre desinformación, delitos y contol de acceso a menores. La clave de las variables y su significado la puedes encontrar en la página web de los problemas.

Trabajaremos con una versión reducida de este conjunto de datos que puedes obtener mediante la función `load_CIS_Red` de la librería `apafib` esta retornará un dataframe de Pandas. Resuelve los siguientes apartados ilustrando los resultados de la manera que te parezca más adecuada.

a) Divide el conjunto de datos en entrenamiento y test (80 %/20 %). Comprobarás que la mayoría de las variables del conjunto de datos son categóricas, en su mayoría binarias. En este problema consideraremos que todas las variables categóricas son no ordenadas. Explorando los datos también encontrarás que hay bastantes valores perdidos. En este caso los valores perdidos corresponden a valores *no sabe/no contesta* y en el caso de los valores categóricos esto puede ser información interesante. Haremos la imputación de manera diferente para categóricos y para numéricos. Para los categóricos, la función de Pandas que la calcula las columnas de la codificación *One Hot* permite indicar que se quiere una columna más para el valor perdido, lo haremos así. Para los numéricos, haremos una imputación usando el `KNNImputer` después de imputar los categóricos.

Una vez tienes el conjunto de datos preprocesado puedes aplicar adecuadamente PCA para ver su comportamiento. Analiza sus resultados y representa la variable objetivo sobre el PCA en dos dimensiones. Explica lo que has observado. ¿Podrían ser las clases linealmente separables?

b) Vamos a establecer un nivel base de predicción, dado que las variables ahora son todas numéricas, ajusta una regresión logística a los datos. Evalúa adecuadamente la calidad del modelo.

c) Quizás con un modelo de aprendizaje más complejo podremos obtener un mejor resultado. Entrena un modelo SVM lineal y un modelo SVM con kernel RBF ajustando adecuadamente sus hiperparámetros. Calcula la *permutation importance* sobre el test para determinar qué atributos son más importantes en estos modelos para predecir. Compara los atributos más importantes de los dos modelos. ¿Hay cierto acuerdo entre los modelos respecto a cuáles son los atributos más importantes?

d) Los conjuntos de clasificadores permiten adaptarse mejor a la complejidad de los datos. Ajusta un random forest y un Exta Trees classifier a los datos explorando adecuadamente sus hiperparámetros. Compara la calidad de los modelos con los anteriores. Calcula la *permutation importance* sobre el test para determinar qué atributos son más importantes para predecir en estos modelos. Compara los atributos más importantes con los del apartado anterior. ¿Hay cierto acuerdo entre los modelos respecto a cuáles son los atributos más importantes?

e) Como hemos visto en teoría, cuando tenemos varios modelos podemos combinarlos usando diferentes estrategias. Entrena un `StackedClassifer` usando los tres modelos de los apartados (b) y (c) con sus mejores hiperparámetros ¿Es mejor este modelo que los modelos individuales? Añade ahora el modelo Extra Trees a la combinación ¿Ha mejorado este modelo al mejor modelo individual? Calcula la *permutation importance* sobre el test para esta última combinación para determinar qué atributos son más importantes. Compara los atributos más importantes con los de los apartados anteriores.

4. **Esto se ha roto**

   El mantenimiento predictivo es una técnica importante en la industria. El funcionamiento de las máquinas es monitorizado de manera continuada para predecir si funcionarán adecuadamente durante las próximas horas o presentarán alguna de las posibles averías que puede tener. Este conjunto de datos utiliza diferentes parámetros de una máquina, el proceso que está realizando y valores de sus sensores para determinar si su estado es normal o cuál es el tipo de avería que se presentará en un futuro cercano. El problema tiene ocho clases, la clase mayoritaria que es el estado normal y siete posibles fallos.

   Puedes obtener estos datos mediante la función `load_maintenance` de la librería `apafib`. Resuelve los siguientes apartados ilustrando los resultados de la manera que te parezca adecuada.

   a) Divide el conjunto de datos en entrenamiento y test (80 %/20 %). El conjunto de datos solo tiene variables continuas y tiene valores perdidos, haz una imputación usando el `KNNImputer`. Aplica reducción de dimensionalidad con PCA y t-SNE y representa la variable respuesta sobre la transformación, ¿parece haber alguna relación entre las variables con la variable respuesta? ¿Podrían ser las clases linealmente separables?

   b) Vamos a establecer un nivel base de predicción, dado que las variables ahora son todas numéricas, ajusta un naive bayes gausiano a los datos. Evalúa adecuadamente la calidad del modelo. ¿Te parece un modelo adecuado para esta tarea?

   c) Quizás usando un tipo de modelo mejor podremos obtener un mejor resultado. Entrena un modelos SVM lineal y otro con kernel RBF ajustando adecuadamente sus hiperparámetros. ¿Son estos modelos de mejor calidad? Calcula la *permutation importance* sobre el test para determinar qué atributos son más importantes en el modelo para predecir. ´

   d) Los conjuntos de clasificadores permiten adaptarse mejor a la complejidad de los datos. Ajusta un random forest y un Exta Trees classifier a los datos explorando adecuadamente sus hiperparámetros. Compara la calidad de los modelos con los anteriores. Calcula la *permutation importance* sobre el test para determinar qué atributos son más importantes

para predecir. Compara los atributos más importantes con los de los apartados anteriores. ¿Hay cierto acuerdo entre los modelos respecto a cuáles son los atributos más importantes?

e) Como hemos visto en teoría, cuando tenemos varios modelos podemos combinarlos usando diferentes estrategias. Entrena un `StackedClassifer` usando todos los modelos no lineales que has entrenado en los dos apartados anteriores con sus mejores hiperparámetros ¿Es mejor este modelo combinado que los modelos individuales? Calcula la *permutation importance* sobre el test para determinar qué atributos son más importantes. Compara los atributos más importantes con los de los apartados anteriores.

5. **Duerme bien**

Los problemas del sueño son algo común en las sociedades industrializadas que pueden derivar en problemas graves de salud. Por eso es importante detectarlos a tiempo. Vamos a trabajar con una versión de un conjunto de datos que recoge diferentes variables sobre pacientes que tienen problemas de sueño (insomnio/apnea) y sujetos controles que no tienen ningún problema[2]. El objetivo es tener un modelo que pueda clasificar nuevas personas y poder obtener unos criterios que puedan permitir entender el diagnóstico.

Puedes obtener estos datos mediante la función `load_dormir` de la librería `apafib`. Resuelve los siguientes apartados ilustrando los resultados de la manera que te parezca adecuada.

a) Primero elimina las variables identificadoras y transforma la variable *Blood Pressure* de manera que sean dos variables enteras (alta y baja presión). Transforma las variables categóricas con *one hot encoding*. Divide el conjunto de datos en entrenamiento y test (70 %/30 %). Haz una exploración mínima del conjunto de datos de entrenamiento observando las relaciones entre las variables, especialmente con la variable objetivo. Describe las cosas que hayas visto que te parezcan interesantes. Aplica reducción de dimensionalidad con PCA y t-SNE y representa la variable respuesta sobre la transformación, ¿parece haber alguna relación entre las variables con la variable respuesta?

b) Un modelo lineal es directamente interpretable a partir de sus pesos. Ajusta una regresión logística y una SVM lineal explorando adecuadamente sus hiperparámetros. Evalúa la calidad de los modelos y compara la importancia que asignan los modelos a los atributos para cada clase.

c) Otro modelo directamente interpretable es el árbol de decisión. Ajusta un árbol de decisión a los datos explorando adecuadamente sus hiperparámetros. Representa el árbol de decisión resultante y mira cuáles son los atributos que corresponden a los primeros niveles del árbol, compáralos con los atributos más importantes de los modelos lineales del apartado anterior.

d) Los conjuntos de clasificadores permiten adaptarse mejor a la complejidad de los datos. Ajusta un random forest a los datos explorando adecuadamente sus hiperparámetros. Compara la calidad del modelo con los anteriores. Calcula la *permutation importance* sobre el test para determinar qué atributos son más importantes en el modelo para predecir. Compara los atributos más importantes con los de los apartados anteriores. ¿Hay cierto acuerdo entre los modelos respecto a cuáles son los atributos más importantes?

e) Como hemos visto en teoría, cuando tenemos varios modelos podemos combinarlos usando diferentes estrategias. Entrena un `StackedClassifer` y un `VotingClassifier` usando los tres mejores modelos ente los dos modelos lineales y los dos de combinación de

[2]Puedes encontrar la descripción de las variables aquí https://www.kaggle.com/datasets/uom190346a/sleep-health-and-lifestyle-dataset.

árboles de decisión usando sus mejores hiperparámetros ¿Es mejor alguno de estos modelos combinados? Calcula la *permutation importance* sobre el test para determinar qué atributos son más importantes en la mejor combinación. Compara los atributos más importantes con los de los apartados anteriores.

6. **Tráfico aéreo en Barcelona**

   Uno de los datos que recolecta la web de *la ciutat al día* del ayuntamiento de Barcelona es el número de vuelos que llegan y parten del aeropuerto del Prat. Una cosa que nos podemos preguntar es si ese número de vuelos se puede predecir a partir del comportamiento de variables de la ciudad. Es lógico pensar que hay una relación por ejemplo con el tráfico de la ciudad, el número de visitantes de ciudades cercanas o incluso con la temperatura. Igual que en otros problemas que usan este conjunto de datos, correlación no significa causalidad, la relación suele corresponden a otras variables no observadas, pero a falta de conocerlas podemos usar las relaciones que aparecen para formular hipótesis. En este caso nos podemos preguntar si podemos predecir cuanto tráfico aéreo tiene el aeropuerto en un día específico.

   Puedes obtener estos datos mediante la función `load_BCN_vuelos` de la librería `apafib`. Resuelve los siguientes apartados ilustrando los resultados de la manera que te parezca adecuada.

   a) Se nos ocurre que variables relacionada con la fecha podrían tener bastante importancia en esta predicción. El índice del conjunto de datos es la fecha, extrae el día, el día de la semana, la semana del año y el mes, y añádelos al conjunto de datos.

   Divide el conjunto de datos en entrenamiento y test (80 %/20 %). Haz una exploración mínima del conjunto de datos de entrenamiento observando las relaciones entre las variables, especialmente con la variable objetivo. Describe las cosas que hayas visto que te parezcan interesantes. Aplica reducción de dimensionalidad con PCA y t-SNE y representa la variable respuesta sobre la transformación, ¿parece haber alguna relación entre las variables con la variable respuesta? Transforma las variables adecuadamente para poder ajustar modelos de regresión, tanto el conjunto de entrenamiento como el de test. Aplicaremos el logaritmo a la variable a predecir de manera que nos fijaremos en la magnitud del número de vuelos en lugar del número específico de vuelos.

   b) Nos podríamos preguntar si la relación entre las variables es simplemente lineal. Ajusta una regresión LASSO a los datos y usa el MSE como medida base para comparar con el resto de modelos. Fíjate también en el valor de los pesos que asigna la regresión a cada variable.

   c) Ajusta una SVM de regresión con kernel lineal y con kernel RBF explorando los hiperparámetros adecuadamente. Usa el método de *permutation importance* sobre el test para determinar qué atributos son más importantes para la SVM con kernel RBF, mira los pesos que asigna a los atributos la SVM lineal. Compara los resultados de estos modelos con la regresión lineal, acierto e importancia de los atributos.

   d) Ajusta los modelos random forest y gradient boosting para regresión explorando los hiperparámetros adecuadamente. Elige adecuadamente el modelo que parezca mejor y usa el método de *permutation importance* sobre el test para determinar qué atributos son más importantes en el modelo para predecir.

   e) Como hemos visto en teoría, cuando tenemos varios modelos podemos combinarlos usando diferentes estrategias. Entrena un `StackedRegressor` y un `VotingRegressor` con los tres mejores modelos que has encontrado en los apartados anteriores con sus mejores hiperparámetros. ¿Es mejor alguno de estos modelos combinados? Calcula la *permutation importance* de los atributos sobre el test con estos modelos combinados ¿ha cambiado qué utilizan los modelos combinados para obtener las predicciones?