PROMiDAT Iberoamericano
Machine Learning con Python III
Métodos Supervisados

- Las tareas tienen fecha de entrega una semana después a la clase y deben ser entregadas antes del inicio de la clase siguiente.
- Cada día de atraso en implicará una pérdida de 10 puntos.
- Las tareas son estrictamente de carácter individual, tareas iguales se les asignará cero puntos.
- En nombre del archivo debe tener el siguiente formato: `Tarea1_nombre_apellido.pdf`. Por ejemplo, si el nombre del estudiante es Luis Pérez: `Tarea1_luis_perez.pdf`. Para la tarea número 2 sería: `Tarea2_luis_perez.pdf`, y así sucesivamente.
- Esta tarea tiene un valor de un 25 % respecto a la nota total del curso.

# Tarea Número 3

- **Pregunta 1:** [40 puntos] En este ejercicio usaremos los datos (`voces.csv`). Se trata de un problema de reconocimiento de género mediante el análisis de la voz y el habla. Esta base de datos fue creada para identificar una voz como masculina o femenina, basándose en las propiedades acústicas de la voz y el habla. El conjunto de datos consta de 3.168 muestras de voz grabadas, recogidas de hablantes masculinos y femeninos.

  El conjunto de datos tiene las siguientes propiedades acústicas (variables) de cada voz:

  - `meanfreq`: frecuencia media (en kHz).
  - `sd`: desviación estándar de frecuencia.
  - `median`: frecuencia mediana (en kHz).
  - `Q25`: primer cuantil (en kHz).
  - `Q75`: tercer cuantil (en kHz).
  - `IQR`: rango intercuantile (en kHz).
  - `skew`: sesgo (ver nota en la descripción de specprop).
  - `kurt`: kurtosis (ver nota en la descripción de specprop).
  - `sp.ent`: entropía espectral.
  - `sfm`: planitud espectral.
  - `mode`: modo frecuencia.
  - `centroide`: centroide de frecuencia (ver specprop).
  - `peakf`: frecuencia de pico (frecuencia con mayor energía).
  - `meanfun`: promedio de la frecuencia fundamental medida a través de la señal acústica.
  - `minfun`: frecuencia mínima fundamental medida a través de la señal acústica.
  - `maxfun`: máxima frecuencia fundamental medida a través de la señal acústica.
  - `meandom`: promedio de la frecuencia dominante medida a través de la señal acústica.

- `mindom`: mínimo de la frecuencia dominante medida a través de la señal acústica.
- `maxdom`: máximo de la frecuencia dominante medida a través de la señal acústica.
- `dfrange`: rango de frecuencia dominante medido a través de la señal acústica.
- `modindx`: índice de modulación. Calculado como la diferencia absoluta acumulada entre las mediciones adyacentes de las frecuencias fundamentales dividida por la gama de frecuencias.
- `género`: Masculino o Femenino (variable a predecir).

Realice lo siguiente:

1. Cargue la tabla de datos `voces.csv` en **Python**.
2. Use Bosques Aleatorios, ADABoosting y XGBoosting en **Python** (con los parámetros por defecto) para generar un modelo predictivo para la tabla `voces.csv` usando el 80 % de los datos para la tabla aprendizaje y un 20 % para la tabla testing, luego calcule para los datos de testing la matriz de confusión, la precisión global y la precisión para cada una de las dos categorías. ¿Son buenos los resultados? Explique.
3. Usando la función programada en el ejercicio 1 de la tarea anterior, los datos `voces.csv` y los modelos generados arriba construya un `DataFrame` de manera que en cada una de las filas aparezca un modelo predictivo y en las columnas aparezcan los índices *Precisión Global, Error Global Precisión Positiva (PP), Precisión Negativa (PN), Falsos Positivos (FP), los Falsos Negativos (FN), la Asertividad Positiva (AP) y la Asertividad Negativa (AN)*. ¿Cuál de los modelos es mejor para estos datos?
4. Repita los ejercicios 1-3, pero esta vez use una combinación diferente de los parámetros de los métodos. ¿Mejora la predicción?.
5. Repita los ejercicios 1-4, pero esta vez use 2 combinaciones diferentes de selección de 6 variables predictoras. ¿Mejora la predicción?.

- **Ejercicio 2:** [30 puntos] Esta pregunta utiliza los datos (`tumores.csv`). Se trata de un conjunto de datos de características del tumor cerebral que incluye cinco variables de primer orden y ocho de textura y cuatro parámetros de evaluación de la calidad con el nivel objetivo. La variables son: Media, Varianza, Desviación estándar, Asimetría, Kurtosis, Contraste, Energía, ASM (segundo momento angular), Entropía, Homogeneidad, Disimilitud, Correlación, Grosor, PSNR (Pico de la relación señal-ruido), SSIM (Índice de Similitud Estructurada), MSE (Mean Square Error), DC (Coeficiente de Dados) y la variable a predecir `tipo` (1 = Tumor, 0 = No-Tumor).

  Realice lo siguiente:

  1. Use Bosques Aleatorios, ADABoosting y XGBoosting en **Python** para generar un modelo predictivo para la tabla `tumores.csv` usando el 70 % de los datos para la tabla aprendizaje y un 20 % para la tabla testing.
  2. Usando la función programada en el ejercicio 1 de la tarea anterior, los datos `tumores.csv` y los modelos generados arriba construya un `DataFrame` de manera que en cada una de las filas aparezca un modelo predictivo y en las columnas aparezcan los índices *Precisión Global, Error Global Precisión Positiva (PP), Precisión Negativa (PN), Falsos Positivos (FP), los Falsos Negativos (FN), la Asertividad Positiva (AP) y la Asertividad Negativa (AN)*. ¿Cuál de los modelos es mejor para estos datos?

3. Repita los ejercicios 1-2, pero esta vez use una combinación de los parámetros del método de cada uno de los métodos citados arriba. ¿Mejora la predicción?

- **Pregunta 3:** [30 puntos] La idea de este ejercicio es programar una Clase en **Python** para un nuevo método de **Consenso Propio**, esto basado en los métodos K-vecinos más cercanos, Árboles de Decisión, Método de Potenciación (XGBoosting) y Método de Potenciación (ADABoosting), para esto realice los siguiente:

1. Programe una Clase en **Python** denominada `ConsensoPropio` que tiene, además del constructor, al menos los siguientes métodos `fit(X_train, y_train, ...)` que recibe la tabla de entrenamiento y genera 4 muestras aleatorias con reemplazo (Boostraps) de los datos de aprendizaje y luego aplica en cada una de estas muestras uno de los métodos predictivos mencionados arriba. Este método debe generar un nuevo modelo predictivo que es un atributo de clase, tipo diccionario, que incluya los 4 modelos generados (todos los métodos usarán todas las variables) y las 4 de precisiones globales, respectivamente de cada modelo[1], que denotamos por $(PG_1, PG_2, \ldots, PG_4)$, donde $0 \leq PG_j \leq 1$ para $j = 1, 2, \ldots, 4$.
2. Programe una función `predict(X_test)` que recibe la tabla de testing. Luego, para predecir aplica en cada una de las filas de la tabla de testing los 4 modelos predictivos que están almacenados dentro de la Clase en el atributo incluido para este efecto; y se establece un consenso de todos los resultados. Se debe programar una fórmula en **Python** que le dé mayor importancia a los métodos con mejor precisión global.

   Si denotamos por $M_j(h, i)$ la probabilidad que retorna el $j$-ésimo modelo en el individuo $i$-ésimo para la categoría $h$ de variable a predecir, donde $j$ varía de 1 hasta 4, $h$ varía desde 1 hasta $p$=número de categorías de la variable a predecir e $i$ varía de 1 hasta $s$ = cantidad de individuos en la tabla de testing, esta fórmula de define como sigue:

   $$C(i) = m,$$

   donde $m$ es el valor que toma $h$ cuando se alcanza el valor máximo en la siguiente fórmula:

   $$\max_{h=1,2,\ldots,p} \left\{ \sum_{j=1}^{4} p_j M_j(h, i) \right\};$$

   y $p_k = \dfrac{PG_k}{\sum_{j=1}^{4} PG_k}$ para $k = 1, 2, \ldots, 4$, note que $\sum_{j=1}^{4} p_j = 1$, pues son pesos.

   La función `predict(X_test)` debe retornar un vector, una lista o un diccionario con las predicciones para todas las filas de la tabla de testing usando la función $C(i)$.
3. Usando la tabla de datos `voces.csv` genere al azar una tabla de testing con un 20 % de los datos y con el resto de los datos construya una tabla de aprendizaje.

[1] Estas serán calculados separando la tabla de aprendizaje en dos tablas, una de entrenamiento y otra de testing, en esta tabla de testing se calculará dicha precisión.

4. Genere modelos predictivos usando la Clase `ConsensoPropio` y el método `fit` de la clase `RandomForestClassifier` (con solamente 4 árboles, es decir, 4 boostraps), luego para la tabla de testing calcule, para ambos métodos, calcule la precisión global, el error global y la precisión por clases. ¿Cuál método es mejor?