Profesor: Dr. Oldemar Rodríguez Rojas
PF-1319 y PF-1320 Análisis de Datos II
Fecha de Entrega: Domingo 25 de septiembre a las 12 media noche
Instrucciones:

- La solución a cada tarea se debe subir en el aula virtual, no pueden ser enviadas por correo.
- Las tareas son estrictamente individuales.
- Tareas idénticas se les asignará cero puntos.
- Todas las tareas tienen el mismo valor en la nota final del curso.
- Las tareas se pueden entregar tarde, pero cada día de atraso tendrá un rebajo de 20 puntos.

Tarea Número 5

- **Ejercicio 1:** [30 puntos] En este ejercicio vamos a usar la tabla de datos `raisin.csv`, que contiene es resultado de un sistema de visión artificial para distinguir entre dos variedades diferentes de pasas (Kecimen y Besni) cultivadas en Turquía. Estas imágenes se sometieron a varios pasos de preprocesamiento y se realizaron 7 operaciones de extracción de características morfológicas utilizando técnicas de procesamiento de imágenes.

  El conjunto de datos tiene 900 filas y 8 columnas las cuales se explican a continuación.

  - `Area`: El número de píxeles dentro de los límites de la pasa..
  - `MajorAxisLength`: La longitud del eje principal, que es la línea más larga que se puede dibujar en la pasa.
  - `MinorAxisLength`: La longitud del eje pequeño, que es la línea más corta que se puede dibujar en la pasa.
  - `Eccentricityl`: Una medida de la excentricidad de la elipse, que tiene los mismos momentos que las pasas.
  - `ConvexArea`: El número de píxeles de la capa convexa más pequeña de la región formada por la pasa.
  - `Extent`: La proporción de la región formada por la pasa al total de píxeles en el cuadro delimitador.
  - `Perimeter`: Mide el entorno calculando la distancia entre los límites de la pasa y los píxeles que la rodean.
  - `Class`: Tipo de pasa Kecimen y Besni (Variable a predecir).

  Realice lo siguiente:

  1. Use Bosques Aleatorios, ADABoosting y XGBoosting en **Python** para generar un modelo predictivo para la tabla `raisin.csv` usando el 75 % de los datos para la tabla aprendizaje y un 25 % para la tabla testing. Use la configuración de parámetros que usted mejor considere o identifique.

2. Según los gráficos de importancia de variables, indique cuáles son las 4 variables más importantes para los modelos.
3. Construya un `DataFrame` que compare los modelos construidos arriba con los mejores modelos generados en tareas anteriores para la tabla `raisin.csv`. Para esto en cada una de las filas debe aparecer un modelo predictivo y en las columnas aparezcan los índices *Precisión Global, Error Global, Precisión Positiva (PP) y Precisión Negativa (PN)*. ¿Cuál de los modelos es mejor para estos datos? Guarde los datos de este DataFrame, ya que se irá modificando en próximas tareas.

- **Ejercicio 2:** [40 puntos] En este ejercicio usaremos la tabla de datos `abandono_clientes.csv`, que contiene los detalles de los clientes de un banco.

  La tabla contiene 11 columnas (variables), las cuales se explican a continuación.

  - `CreditScore`: Indica el puntaje de crédito.
  - `Geography`: País al que pertenece.
  - `Gender`: Género del empleado.
  - `Age`: Edad del empleado.
  - `Tenure`: El tiempo del vínculo con la empresa.
  - `Balance`: La cantidad que les queda.
  - `NumOfProducts`: Los productos que posee.
  - `HasCrCard`: Tienen tarjeta de crédito o no.
  - `IsActiveMember`: Es un miembro activo o no.
  - `EstimatedSalary`: Salario estimado.
  - `Exited`: Indica si el cliente se queda o se va.

  Realice lo siguiente:

  1. Cargue en **Python** la tabla de datos `abandono_clientes.csv`.
  2. Use Bosques Aleatorios, ADABoosting y XGBoosting en **Python** para generar modelos predictivos para la tabla datos `abandono_clientes.csv` usando el 75 % de los datos para la tabla aprendizaje y un 25 % para la tabla testing. Use la configuración de parámetros que usted mejor considere o identifique. 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. Repita el ítem anterior, pero esta vez seleccione 6 variables predictoras. Para esto utilice el gráfico de importancia de variables del método Bosques Aleatorios. ¿Mejora la predicción?.
  4. Construya un `DataFrame` que compare los mejores modelos construidos arriba con los mejores modelos generados en tareas anteriores para la tabla `abandono_clientes.csv`. Para esto en cada una de las filas debe aparecer un modelo predictivo y en las columnas aparezcan los índices *Precisión Global, Error Global, Precisión Positiva (PP) y Precisión Negativa (PN)*. ¿Cuál de los modelos es mejor para estos datos? Guarde los datos de este DataFrame, ya que se irá modificando en próximas tareas.

- **Ejercicio 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 $p_k = \dfrac{PG_k}{\sum_{j=1}^{4} PG_j}$ 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 con las predicciones para todas las filas de la tabla de testing usando la función $C(i)$.
  3. Usando la tabla de datos datos `raisin.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.
  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?

[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.

oldemar rodríguez

CONSULTOR en M1NƐR14 DƐ D4T0S