**PROYECTO ESTRUCTURA DATOS Y ALGORITMOS I**

**DANIEL GARCIA SALCEDO**

**SALOMON VELEZ PEREZ**

**SIMON BETANCUR BEDOYA**

**MEDELLIN**

**UNIVERSIDAD EAFIT**

**2020**

**INTRODUCCION**

Hay muchas formas de ordenar los datos de forma óptima para su análisis, algunas mejor que otras, pero lo importante es saber cual es mas conveniente para cada situación, y por supuesto, utilizar bien el método que se desee emplear. Esto se complica aun mas cuando se ve para que se usaran los datos, y es simplemente registro, si se usaran para ver el promedio de alguna medida u opinión popular, la probabilidad que tiene alguien de conseguir un puesto en una empresa, entre mas usos. En nuestro caso, nuestros datos son de algunos estudiantes que presentaron el ICFES, y nuestro objetivo con estos, es saber la probabilidad que tiene cada estudiante de sacar un buen puntaje en el ICFES, e identificar cuales son los factores en común que tuvieron los estudiantes que obtuvieron un alto puntaje. En este documento explicaremos la razón de porque escogimos el método de ordenamiento “Arboles de decisión”, mostraremos la construcción del árbol, su Matriz de confusión, y plantearemos algunas conclusiones.

**Árboles de clasificación (Pandas-numpy) [VAMOS A ESCOGER ESTE]**

Ganancia de información:

Cuando usamos un nodo en un árbol de decisión para particionar las instancias de formación en subconjuntos más pequeños, la entropía cambia. La ganancia de información medida de este cambio en la entropía.

Comenzar con todos las instancias de formación asociadas al nodo raíz.

Utilizar la ganancia de información para elegir que atributo etiquetar cada nodo con cual.

Construir cada subárbol en el subconjunto de instancias de capacitación que se clasificaran.

Índice de Gini:

Es una métrica para medir la frecuencia con la que un elemento elegido al azar Seria identificado incorrectamente. Esto significa que se debe preferir un atributo con un índice de Gini más bajo.

Ventajas:

Los árboles de decisión son fáciles de interpretar y visualizar y pueden capturar fácilmente patrones no lineales.

Requiere menos preprocesamiento de datos por parte del usuario, por ejemplo, no es necesario normalizar las columnas.

Se puede utilizar para ingeniería de características, como la predicción de valores perdidos, adecuada para la selección de variables.

Se puede utilizar para ingeniería de características, como la predicción valores perdidos, adecuados para la selección de variables.

Desventajas:

Datos sensibles al ruido, puede sobredimensionar los datos ruidosos.

La pequeña variación en los datos puede dar lugar a un árbol de decisión diferente

Están sesgados con un conjunto de datos de desequilibrio, por lo que se recomienda equilibrar el conjunto de datos antes de crear el árbol de decisión.

**Arboles de regresión (Pandas-Numpy)**

Divide el espacio de características p-dimensionales, en M regiones mutuamente distintas que cubren completamente el subconjunto del espacio de características y no se superpone.

Cualquier observación nueva que caiga en una partición particular tiene la respuesta estimada dada por la media de todas las observaciones de entrenamiento.

El objetivo para este algoritmo es minimizar algún tipo de criterio de error.

Deseamos minimizar la suma de cuadrados residual (RSS), una medida de error también utilizadas en la configuración de regresión lineal.

Es demasiada costoso computacionalmente considerar todas las particiones posibles por lo tanto debemos utilizar otro enfoque aquí es donde entra la división binaria recursiva.

La división binaria recursiva aborda el problema comenzando en la parte superior del árbol y dividiendo el árbol en dos ramas, lo que crea una partición de dos espacios.

Limitar el crecimiento de un árbol de decisión se especificará en términos del número máximo de capas, o la profundidad, que puede tener.

La poda del árbol implica probar el árbol original contra versiones podadas de él.

**Decisión de árbol condicional (Random Forest)**

Un bosque está compuesto de árboles. Se dice que cuantos más arboles tenga, más robusto será el bosque.

Crea arboles de decisión a partir de muestra de datos seleccionados al azar, obtiene predicciones de cada árbol y selecciona la mejor solución mediante votación.

Los árboles de decisión individuales se generan utilizando un indicador de selección de atributos:

Ganancia de información

Relación de ganancia

Índice Gini

Es un problema de clasificación, cada árbol vota y se elige la clase mas popular como resultado final. Es más simple y más potente en comparación con otros algoritmos de clasificación no lineal.

Construir un árbol de decisión para cada muestra y obtener un resultado de predicción de cada árbol de decisión

Realizar una votación por cada resultado previsto

Seleccionar el resulto de la predicción con más votos como predicción final

Importancia de las características:

La librería de Python Sklearn proporciona una gran herramienta para esto, que mide la importancia de las características observando en qué medida los nodos de los árboles, que utilizan esas características, reducen la impureza en todos los árboles del bosque.

Ventajas:

Los bosques aleatorios se consideran un método muy preciso y robusto debido al número de árboles de decisión que participan en el proceso.

No sufre el problema del sobreajuste. La razón principal es que toma el promedio de todas las predicciones, lo que anula los sesgos.

Los bosques aleatorios también pueden manejar los valores que faltan.

Puede obtener la importancia relativa de las características, lo que ayuda a seleccionar las características más importantes para el clasificador.

Desventajas:

Los bosques aleatorios son lentos en generar predicciones porque tienen múltiples arboles de decisión.

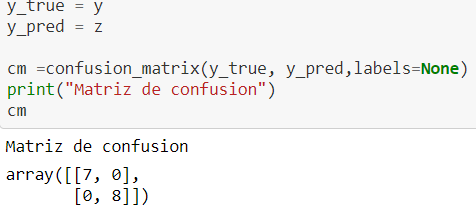
El modelo es difícil de interpretar en comparación con un árbol de decisión, donde se puede tomar una decisión fácilmente siguiendo la ruta del árbol.

Se decidió Utilizar el árbol de clasificación para el proyecto, debido a que tenía las características mas convenientes para lo que se buscaba

**Desarrollo del árbol**

Se implementaron las librerías de pandas, sklearn, pydotplus y matplotlib, y luego estas mismas se instalan, se importa el archivo, y este se mostrará en pantalla. Después de eso se definen las características y si en base a estas el estudiante tuvo éxito o no. Se define el árbol y se muestra en pantalla, y por ultimo se crea un método que logra que el algoritmo prediga si un estudiante tendrá éxito o no en base a las características que presente.

**Matriz de confusión**



**Conclusiones**

El método utilizado resultó ser óptimo para los resultados esperados. Aunque su proceso de implementación fue algo complicado, se logró el objetivo al final, e incluso se logró implementar un método que logra predecir el éxito de un estudiante en la prueba en base a sus características