Buenas noches chicos, felicitaciones por la entrega. Hay muchas mejoras notables respecto a las anteriores, y si no fuera por alguna que otra corrección importante, se iría con un aprobado directo. ¡Felicitaciones!

A continuación les dejo observaciones que encuentro sobre su trabajo en esta 2da entrega del desafío final:

Muy buen uso del apply para transformar columnas, esta es la mejor manera de aplicar transformaciones eficientes sobre columnas.

**ok**

Desconozco si están queriendo usar programación funcional, pero no es común hacer el procesamiento sobre el dataframe desarrollando funciones. Hacemos funciones cuando queremos escribir código una sola vez y luego utilizarlo varias veces en distintas partes del código. Como aquí solo se hace el procesamiento una sola vez, lo mejor es dejarlo explicito y con la mayor claridad posible.

No entiendo a cual función se refiere: Las transform Se usan una única vez pero son necesarias para después usar apply.

Y la procces\_data pensé que estaba al pedo, pero se usa dos veces, para df\_train y df\_test. El tema que arrancamos con dos dataset de entrada ya spliteados desde kaggle. Y a uno lo llamamos sólo df (al df\_train), y al otro df\_test de ahí debe venir la confusión.

Empiecen a ponerle grid de fondo a los gráficos de barras y parecidos, porque sino es muy difícil saber el valor de cada barra. Adicionalmente, encima de cada barra, pueden mostrar el valor medido (redondeado con 2 decimales a lo mucho).

Los gráficos de torta mostrados necesitan tener letra blanca porque sino se pierde con fondos oscuros.

**Grafico de torta: Se cambio el color de fondo, y sigue letra negra. Se cambio la repetición de código y se hizo por iteración**

**Se agrego el método grid() a los gráficos de barras y algunos otros.**

**Se le agregaron etiquetas encima de cada barra, no tienen sentido los decimales ya que son enteros.**

Me gustó mucho el groupby por target y clase, yo lo trataría de llevar a un mapa de calor con los valores de las medias, quedaría muy bien.

**Hecho**

Los últimos 4 gráficos de mapas de calor, vería la forma de automatizar el código. Es decir, en vez de hacer 4 gráficos explícitamente, uno debajo del otro, vería la manera de ir iterando y generando los gráficos de a uno por iteración. Puede que no le vean sentido para 4 gráficos, pero el código sería reutilizable para graficar de a muchos análisis a la vez, algo que van a querer saber hacer siempre.

**Corregido se hace por iteración al igual que el de torta**

Sería bueno hacer un backup de los datos en algunos puntos claves del código, especialmente antes de empezar a entrenar los modelos. Así, cuando quieran volver a entrenar, no tienen que volver a repetir todo el procesamiento de los datos, que por lo general será lo más costoso.

**Hecho backup antes de empezar a entrenamiento. Por ahora solo ahí.**

La normalización tiene que ir antes que el split. No tiene sentido separar los datos y hacer dos normalizaciones distintas, y escalar las mismas variables con dos conjuntos de estadísticos diferentes (el valor medio y el desvío estándar). Lo que sí está muy bien es excluir la variable target de la normalización.

**Tiene sentido el planteo, pero se uso así porque los data set ya venían separados en train y test desde kaggle.**

**Se corrigió como indicó , de todas formas no vario el accuracy. Sigue igual que antes.**

El mapa de calor de correlación está muy prolijo, pero faltó redondear los decimales para que no suceda lo que sucedió. Traten de desarrollar el criterio ustedes, de no esperar una observación para corregir algo que está a la vista que no queda bien. Ustedes no pueden presentar un gráfico donde no se pueda leer correctamente la información que están queriendo mostrar, porque en una reunión con la gerencia y/o especialistas, les puede costar muy caro.

Ok, este mes no podemos pedir aumento lpm…

**Corregido a dos decimales**

Aunque en principio la idea es buena y parece dejar el trabajo prolijo, crear una función para entrenar y validar los modelos provoca, en principio, que se tengan que duplicar todos los datos. Esto es ineficiente, y no es común encapsular todos los pasos en una función. Aún así, siguiendo la lógica:

No hace falta un if sobre verbose, si pueden pasarle el verbose tal como viene.

La función en principio debería importar dentro de su cuerpo, las funciones de otra librería que vaya a utilizar. De esta manera, consiguen que sea independiente y que funcione por sí misma sin necesidad de ayudarla con código externo.

**Ok las correcciones de la función, import adentro y el if estaba demás.**

**De todas formas se saco la función, para ganar eficiencia, (se hizo copy y paste de los pasos repetidos)**

Cuando arman la tabla resumen, aprovechen que ya tienen un montón de métricas calculadas y agreguenlas.

**Se agregó las otras dos medidas cuando se hacen los modelos, cuales otras habría que agregar??**

Aunque no hay tanto desbalance de los datos (56%/44%), sería bueno que hagan un balanceo de los datos, para demostrar que conocen el problema que conlleva tener datos desbalanceados. Esto iría para el próximo (y último) desafío final.

OK Para mejorar en la tercer entrega

Las conclusiones de los resultados obtenidos siguen siendo bastante vagas, y desde las primeras correcciones que les vengo pidiendo que se pongan las pilas con esto. Se espera que justifiquen sus conclusiones con resultados, no con palabras subjetivas. Ustedes solo deben comunicar lo que los datos hayan demostrado.

Las conclusiones finales sobre el tipo de modelo no son del todo cierta:

¿A qué se refieren cuando hablan de que un modelo demando el doble de tiempo que el otro?

Random Forest tiene una cantidad excesiva de árboles que hace crecer linealmente el costo del entrenamiento. Quizás con la 5ta parte de árboles consiguen los mismos resultados.

No se arriesguen a sacar conclusiones de los algoritmos en general: KNN no entrena ningún modelo y por cada clasificación necesita medir la distancia a todas las muestras de entrenamiento. Por el otro lado, Random Forest tiene un entrenamiento medianamente costoso, pero para clasificar una muestra el proceso es muy rápido (siendo esta una de las mejores ventajas de estos modelos).

* **Se modificaron las conclusiones básicamente lo que cambie fue según el grafico tal esto, segun el grafico tal lo otro.. La única que estaba mal era la que decía “el tipo de clase no afecta la conformidad de las personas.” Las demás estaban todas bien no se que más agregar.**

Si bien no es necesario que apliquen todas las correcciones en esta entrega (salvo la que está en negrita), si será necesario que vengan aplicadas a la primer entrega del tercer desafío final. Queda muy poco y están muy bien encaminados, pero no se confíen porque todavía tienen mucho para mejorar. Tengan en cuenta que es deseable que incorporen la mayor cantidad de herramientas que dimos en todo el curso, así sea para mejorar el accuracy en un 1%. No se confíen porque hayan obtenido métricas altas al momento, porque eso ya vino así en el dataset que eligieron (incluso con un árbol simple se consiguen buenos modelos). El profesor espera en el desafío final que ustedes hayan acabado todos los recursos para llegar al mejor modelo posible.

gfgfgdgdfgdfg