Instrucciones del Proyecto del Módulo 1

2024-08-24

Objetivo

El objetivo de este challenge es evaluarte en:

- Desempeñar análisis exploratorio, manipulación y preprocesamiento de datos básicos:
  - Vistazos generales
  - Limpieza
  - Decisión respecto a nulos/missings/`NaN`
  - `Join`/`Merge`s de tablas
  - Métricas por grupos (`groupby`)
  - En general, utilizando lo visto en las clases de **Python**
- Demostrar conocimiento de las operaciones realizadas.
- Realizar ***feature engineering*** (AGUAS con el **leakage**!)
- Entrenar un Modelo de Clasificación Binaria sencillo... el desempeño del modelo no es importante para la calificación pero sí la creatividad y habilidad para obtener una base de desarrollo (base de entrenamiento) con features *"buenas"* de cara a la variable objetivo
- Obtener algunos hallazgos, comentarios, hipótesis, conclusiones acerca de los datos.
- **Subir tus archivos a GitHub**

Datos

Los datos utilizados provienen de la base de datos pública de Kaggle de Partidos del Fútbol Europeo que ya deberíamos tener guardada en la ubicación `../data/raw/database.sqlite`

En la clase ya hemos utilizado las tablas de:

- `Match`
- `Team`
- `Team_Attributes`

Para este trabajo, necesitas hacer lo equivalente utilizando, al menos, las tablas de:

- `Player` &
- `Player_Attributes`

En estas tablas también **hay datos anclados al tiempo**, por lo que el cuidado para **evitar el leakage** al momento de crear features e incluirlas al entrenamiento **es fundamental**.

Recomendación para una base más robusta: Pega la tabla que ya tenemos de la clase (la que hicimos con partidos y atributos de equipos) con la tabla nueva que generes (partidos + atributos de jugadores) para una base de entrenamiento más capaz de identificar patrones.

Tecnología

Para la parte del entrenamiento y guardado del modelo es imprescindible utilizar Python. Para todo lo previo, recomendamos ampliamente utilizar el mismo lenguaje, porque es donde más herramientas se proporcionaron en clase, aunque si quisieras hacerlo desde R (cubriendo los puntos que se evaluarán), también es posible.

- **Toma todo lo que sientas que te pueda servir de los Notebooks y archivos de las clases e incorpóralo a tu análisis!**

Formato de presentación

Siéntete libre de presentar tu trabajo desde un Jupyter Notebook. Si quisieras utilizar otro formato, es bien recibido siempre y cuando tu código esté disponible para revisarlo y comentarlo.

Respecto a esto, te(les) pedimos que **suban todos sus archivos al repositorio** `students_diplomado_data_science_fmat` de la siguiente forma:

- **Crear una carpeta** con su número/nombre de equipo dentro de la ubicación `students_diplomado_data_science_fmat/`
- Copiar allí todos los archivos generados.
- `git add`, `git commit` & `git push` desde una rama nueva (del equipo).
- Crear un `Pull Request`.
- Quedar atentos de los comentarios/revisiones de los profesores.
- Después del `approved`, hacer merge.

De la parte técnica, lo que más nos interesa es conocer tu lógica y la de tus compañeros al momento de la manipulación de datos y, **más especialmente, del *feature engineering*** y **las razones que tuvieron para las variables que crearon**.

Implementación del modelo

Elige un modelo de clasificación binaria. Puedes realizar uno o tantos experimentos como desees y comentar esto en la presentación. Finalmente, a partir de las métricas o criterio que hayas elegido desde un principio, decidirás un modelo ganador. No hace falta que utilices muchas. Basta con una, pero se valorará que compartas todas las que decidiste utilizar/considerar y tus razones para ello.

Recuerda que, si decides trabajar con features ancladas al tiempo, antes de generar tus submuestras *train/test*, una **buena práctica** es separar una muestra futura como período ***Out Of Time*** y utilizarla para verificar cuál modelo se desenvuelve mejor con esos datos.

Como ***target* a predecir** puedes utilizar la que construimos en clase:

- `suma_puntos`: toma el valor de 1 si el equipo sumó puntos (gana o empata) en ese partido y 0 en caso contrario (pierde)... Esta variable ya está construida desde la clase, por lo que, en caso de utilizarla, sería cuestión de pegarla con tus nuevas features.

Así también si se te(les) ocurre construir otra variable *target* interesante (que no sea la que ya se hizo en la clase), se valorará esa iniciativa adicional. Importante que sean variable binaria, es decir que sólo tome el valor de 1 ó 0. Algunas ideas podrían ser:

- `gana`: toma el valor de 1 si el equipo gana y 0 en caso contrario (empata o pierde)
- `arco_en_ceros`: toma el valor de 1 si no recibió ningún gol en ese partido y 0 si al menos recibió 1 gol

Si se te ocurre alguna métrica de negocio no dudes en compartir el desempeño de tu modelo en ella y más, si forma parte de tu criterio para decantarte por el modelo ganador.

Aquí te dejamos un poco de documentación de las métrica estándares para evaluar modelos:

- Performance Metrics in Machine Learning
- F1 Score vs ROC AUC vs Accuracy vs PR AUC

Conclusiones

Agrega algunos comentarios resumiendo tu trabajo. Sientete libre de agregar qué es lo que harías diferente, qué mejorarías y qué te gustaría intentar en una siguiente iteración.

¡MUCHO ÉXITO!