Análisis de datos - Trabajo final integrador

Se propone realizar el ciclo completo del desarrollo de un modelo de aprendizaje automático supervisado.

Trabajaremos con un dataset de Kaggle con datos de por distintas estaciones metereológicas de Australia.

El objetivo es predecir si lloverá o no al día siguiente (variable *RainTomorrow*), en función datos metereológicos del día actual.

1. Análisis exploratorio inicial

El trabajo práctico comienza con la exploración inicial de dataset. Una rápida visualización se logra imprimiendo las primeras filas para conocer cuales son las variables. De ahí podemos observar rápidamente tipos de datos y posible existencia de valores faltantes.

Generamos luego una tabla con:

- Cantidad de instancias nan por variable
- Porcentaje de nan por variable
- Cantidad de valores cero por variable
- Porcentaje de valores cero por variables
- Cardinalidad por variable (valores únicos)
- Tipo de dato

De esta manera confirmamos algunas de las hipótesis que salieron de observar las primeras filas. El dataset es muy variado: existen datos de tipo númericos, categóricos y compuestos, todas las variables menos Date y Location poseen valores faltantes y algunas de ellas también muchos valores nulos.

Clasificamos entonces las variables según dos criterios: Clasificación de variables I:

- Fecha/Hora: Date
- Numéricas: MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine, WindGustSpeed, WindSpeed9am, WindSpeed3pm, Humidity3pm, Pressure9am, Pressure3pm, Temp9am, Temp3pm

- Categóricas nominales: Location, WindGustDir, WindDir9pm, WindDir3pm, RainToday, Cloud9am, Cloud3pmRainTomorrow

Clasificación de variables II:

- Entrada: Date, Location, Cloud9am, Cloud3pm, MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine, MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine, WindGustSpeed, WindSpeed9am, WindSpeed3pm, Humidity3pm, Pressure9am, Pressure3pm, Temp9am, Temp3pm, WindGustDir, WindDir9pm, WindDir3pm, RainToday
- Salida: RainTomorrow

Nos encontramos entocnes ante un problema de clasificación y todas las variables de entrada podrian ser informativas en principio.

Se realizan análisis de distribuciones sobre las variables de entrada y salida mediante histogramas, box plots, qq plots, gráficos de torta y gráficas de barra para cardinalidad (para más detalle referirse al desarrollo del Trabajo Práctico). De los análisis se concluye:

- Las variables MinTemp, MaxTemp, Pressure3pm, Pressure9am, Temp3pm y Temp9am están distribuidas de forma normal. Esta característica se observa en el histograma y se confirma en el QQ Plot.
- De Humidity9am y Humidity3pm observamos como la humedad tiende a ser mayor con el avance del día. El valor máximo de humedad es 100, pudiendo ser el caso de una representación porcentual.
- Del histograma, concluimos que WindSpeed9am y WindSpeed3pm presentan distribuciones sesgadas a la derecha. Del Boxplot podemos observar los outliers para hacia los valores positivos.
- La temperatura máxima y mínima tienen distribiciones normales, siendo claramente la media de la máxima mayor a la mínima.
- Rainfall y Evaporation presentan muchos valores en cero, por lo que cualquier valor positivo se reflejará como un outlier en el Boxplot. Resta entender si existe alguna relación entre los valores cero y las demás variables.
- La correlación arroja como resultado que existen relaciones lineales entre variables de la misma unidad, es decir entre Temp, Pressure y Humidity, para las distintas mediciones diarias. Se analiza entonces si la variación diaria es significativa, de manera de entender si es posible prescindir de una variable del par.
- La variable de mayor cardinalidad es Location. Respecto a la aparición de nuevas etiquetas, sólo podría darse para Location, ya que no deberían aparecer nuevas WindDir (habiendo considerado todas en la codificación de variables) y no se esperaría valores mayores a 10 para Cloud. Esto será tenido en cuenta al momento de codificar las variables categóricas.

Tratamiento de la variable compuesta Date:

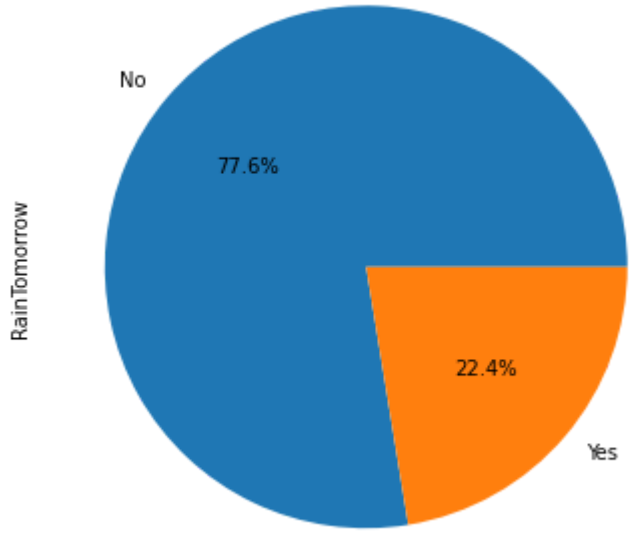
- Podría ser dividida en año, mes y día, convirtiendose en variables numericas discretas.

- Si permanece como un string, entonces seria lo mismo que un id, ya que no habria dos observaciones para el mismo dia, lo cual no agregaría valor al modelo.
- Otra opción mas compleja sería entrenar un embedding de la fecha, de manera de tener una variable en vez de tres.
- También podría usarse una Binary Encoding o Feature Hashing, a costa de aumentar el número de variables.
- OneHotEncoding no está considerado debido a la gran cantidad de valores únicos.

De la distribución de la variable de salida, vemos un claro sesgo. Nos encontramos ante un caso de clasificacion con dos clases de salida, por lo que la solucion mas adecuada es un Binary Enconding, donde Yes=1 y No=0. Si consideramos un modelo constante, y predecimos lluvia para un dia considerando que llovió el dia anterior, entonces obtendremos un 73,78% de acierto aproximado en el set de entrenamiento.

```
df.RainTomorrow.value_counts().plot(figsize=(12,6), kind='pie', autopct='%1.1f%%')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f3e5c304490>
```



2. Esquema de validación de resultados

Partición del dataset en entrenamiento y validación.

3. Limpieza y preparación de datos

- En primer lugar revisamos los valores de salida encontrando valores faltantes. Como el porcentaje el bajo (2.2%), eliminamos las filas del dataset para evitar un overfitting al momento de imputarlos.

- Calculamos la cantidad de observaciones y valores faltantes para cada variable.
- Se identifican todas las variables cuyo porcentaje de Nans es menor al 5% y se refleja en un grafico de barras ordenado.
- Para cada una de las variables del gráfico anterior se filtran las filas con Nan y se calculan los porcentajes de Nan de todas las demás variables. La idea es entender si se da intersección de Nan entre variables.
- Observamos que un gran porcentaje de las demás variables presenta valores faltantes por lo que procedemos a eliminar todas las filas de las variables indicadas en el gráfico de barras anterior. Posteriormente calculamos el porcentaje de reducción del dataset para validar nuestra suposición. Por ello procedemos a eliminar todos las filas con Nan para las variables cuyo porcentaje es menor al 5% y verificamos que porcentaje del dataset original estariamos perdiendo.
- Se genera una función drop_nan la cual permite eliminar las filas con valores nan, considerando las variables cuyo porcentaje de nan sea menor a un valor customizable. La misma luego acomoda los índices con la salida para evitar incongruencias.
- A partir de aquí nos quedamos con una lista de variables que aún tienen valores Nan. Realizaremos la imputación sólo sobre estas variables, corriendo el riesgo que si en el dataset de testeo apareciera alguna otra faltante podría dar un error en la predicción. La práctica óptima sería tener un método de imputación para cada variable, independientemente si presenta valores faltantes en el dataset de train o no.

Imputación de variables:

- Imputación por media sobre Pressure3pm y Pressure9am, debido a su distribución normal.
- Para Evaporation y WindGustSpeed usaremos una imputación por mediana, ya que si usamos algún valor arbitrario podríamos modificar la distribución o hasta generar mayor cantidad de ouliers.
- Para lo valores faltante de Sunshine usaremos 0.
- Los valores faltantes de Cloud3pm y Cloud9am son completadas con el valor 0 para no agregar una nueva categoría.
- Para el caso de WindgustDir y WindDir9am completamos la faltante con una nueva categoría. También podría elegirse la dirección más probable considerando la salida.

Codificación de variables categóricas:

- Una forma de representar la dirección de los vientos es a través de los ángulos asociados de 0° a 360°. (https://www.surfertoday.com/windsurfing/how-to-read-wind-direction). Para el caso Nan reemplazado por la palabra "Faltante" utilizamos el número -1.
- Tanto para la salida, como para la variable RainToday llevamos a cabo una codificación binaria ya que sólo toma uno de dos valores posibles.
- La fecha es dividida en año, mes y dia, todas como variables enteras.

Location puede ser codificada de diversas maneras:

- OneHotEncoding: simple pero agregaría una cantidad de variables igual a la cantidad de ciudades.
- Binary Encoding: permite trabajar con menos variables que OneHotEncoding pero podría no estar representando correctamente la relación en distintas etiquetas.
- Latitude & Longitude: se podría conseguir la Latitud/Longitud de las ciudades para representarlas.
- Embedding: se podría crear un embedding, ya sea a partir de sólo el nombre (lo cual no mapaería de la manera más precisa la relación).

Una vez imputados los datos faltantes y codificadas las variables categóricas procedemosa identificar las variables de entrada de más importancia.

Podemos observar que tanto para el coeficiente de Pearson como Spearman, la salida no presenta una correlación lineal significativa con alguna de las variables de entrada. Las relaciones más prepoderantes se dan con la humedad de tarde y la condición de lluvia/ no lluvia del día actual.

4. Entrenamiento de modelos

Dado que estamos frente a un problema de clasificación, haremos uso de dos modelo: LogisticRegression y RandomForest, un método lineal y uno no lineal. Como nuestro foco está en el preprocesamiento de los datos, usaremos los hiperpámetros por defecto, variando de caso a caso sólo los métodos anterios tales como imputación, codificación o ingeniería de features.

Siendo un dataset desbalanceado usamos la métrica de validación BalancedAccury, definida como el promedio del Recall obtenido en cada clase.

Analizaremos 4 casos diferentes sobre los dos modelos de aprendizaje supervisado:

- Caso 1: Caso principal descripto con anterioridad.
- Caso 2: CCA.
- Caso 3: Caso principal + PCA con 4 componentes.
- Caso 4: Caso principal + MinMax Scaler.

```
resultados
```

	LogReg Train	LogReg Test	RandomForest Train	RandomForest Test
caso_01	72.27	72.05	100.0	72.74

5. Conclusiones

Se podrían realizar muchas más combinaciones, como así también análisis exploratorios más complejos y búsqueda de relaciones. También si se tuvieran más información sobre la forma de recolección de datos sería más fácil poder sacar conclusiones acerca de los datos faltantes. De los distintos casos podemos afirmar lo siguiente:

- RandomForest realiza overfitting sobre el test de entrenamiento. Seguramente se podría mejorar mediante la modificación de los parámetros por defecto del método.
- Tanto de las simulaciones como del análisis inicial podemos asentir sobre la no linealidad en la relación entre los datos. De esta manera RandomForest tendría mejores resultados.
- LogisticRegression no alcanza a aprender las características no lineales del dataset, por ello su performance no es tan alta.
- Para el Caso 02 (eliminación de Nan) nos quedamos con sólo un 40% del dataset original. Al compararlo con el Caso 01 podemos suponer que las imputaciones realizadas afectaron la distribución de las variables y por ende existen oportunidades de mejora.
- Los resultados para el Caso 03 (PCA, n_comp = 4) nos confirman que 4 variables no son suficientes para representar la mayor parte de la varianza del dataset. Se podría ir aumentando la cantidad gradualmente o indicar el valor de corte del método a través del porcentaje de varianza que se quiera representar.
- Observamos como para el Caso 04, al utilizar MixMaxScaler, se modifican las distribuciones sobre el set de test y empeora la performance respecto al procesamiento original.