TP Integrador – Análisis de Datos

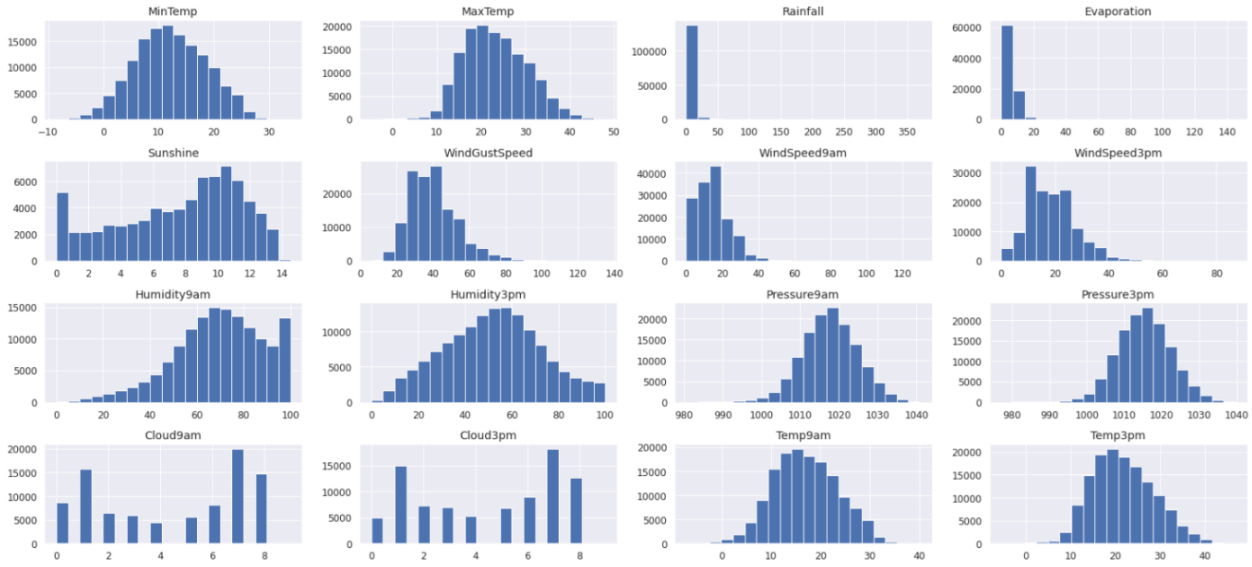
Alumno: Tinelli, Francisco

Link Google Colab – Link Dataset Kaggle – Link del Repositorio Github

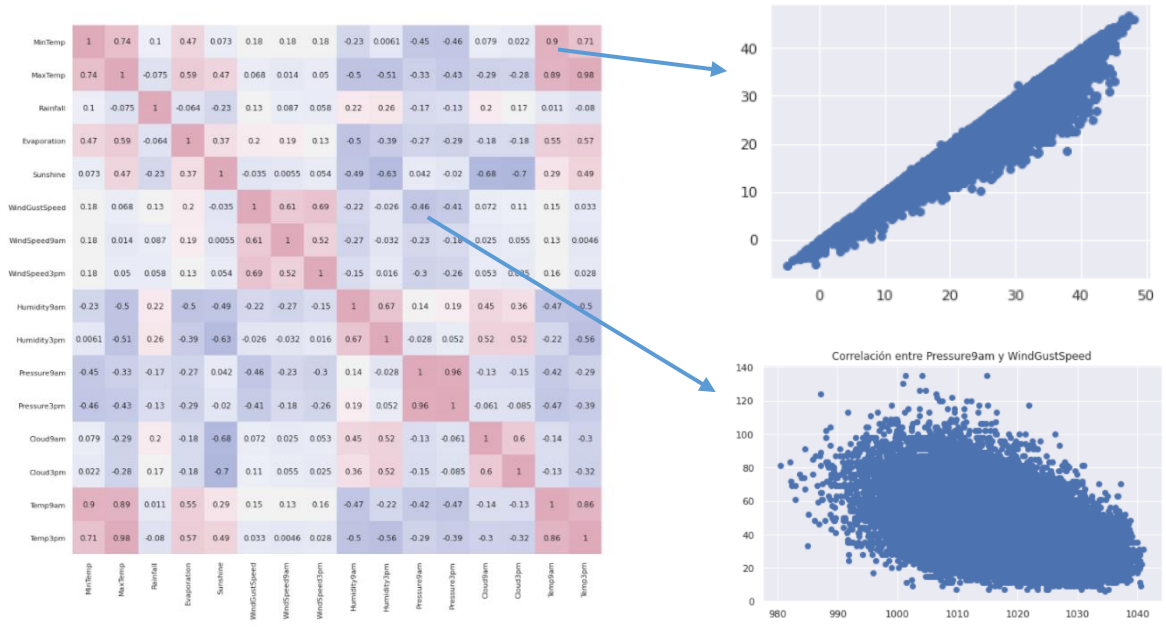
1. Análisis exploratorio inicial

En esta primera etapa, se analizó el dataset, la cantidad de variables presentes y sus tipos (numéricas, categóricas y compuestas) y la variable de salida.

Variables numéricas: En el caso de las variables numéricas, se analizaron mediante histogramas sus distribuciones, algunas de las cuales se asemejan a una normal o a una exponencial.



Se analizó también la correlación entre algunas variables mediante la matriz de correlación y gráficas de correlación.



Variables categóricas: Se analizaron los diferentes valores que pueden tomar las variables categóricas: nombres de ciudades, direcciones de viento (N, NE, etc.) y variables True/False.

Variables compuestas: Es el caso de la fecha para la cual se estudió el rango.

Variable de salida: Es la última columna del dataset *'RainTomorrow'* la cual es booleana y está desbalanceada presentando aproximadamente un 76% de casos *False*, 22% *True* y 2% *NaN*.

2. Esquema de validación de resultados

Se eliminaron las muestras con salida NaN y se particionó el dataset en un 70% para entrenamiento y 30% para test.

3. Limpieza y preparación de datos / ingeniería de features

Datos faltantes: Se analizaron los datos faltantes por variable. La mayor parte de datos faltantes se da en las variables '*Sunshine'*, '*Evaporation'*, '*Cloud3pm'* y '*Cloud9am'*. Mediante agrupamiento de datos por ciudades, se observa que varias presentan datos faltantes en todas sus muestras. Esto podría deberse probablemente a la ausencia del sensor correspondiente en la central de adquisición de datos de esas ciudades.

Imputación de datos faltantes: Se testearon 3 opciones diferentes: supresión de muestras con al menos 1 NaN, imputación por media o moda, supresión de las columnas con mayor cantidad de NaN. Si bien el primer caso descartaría aproximadamente el 40% del dataset, es el que mejor resultado dio. Esto puede deberse a que el número de muestras es muy elevado, las columnas con mayor cantidad de NaN son de vital importancia para el problema y la imputación por media/moda no es adecuada para la cantidad de datos faltantes en este caso.

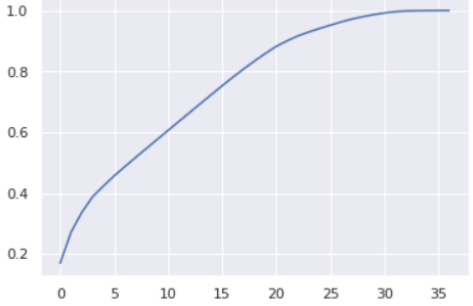
Codificación de la ubicación: Para el caso de la ubicación, se utilizó la API Nominatim de OpenStreetMaps para transformar los nombres de las ciudades en valores de latitud y longitud. Esto no solo reduce drásticamente el número de *features* comparado con la aplicación de OneHotEncoding, sino que también mejora el desempeño ya que la ubicación geográfica permite asociar datos de ciudades cercanas.

Codificación de la dirección del viento: Se transformaron las siglas de dirección en valores de ángulos sexagesimales tomando como referencia 0º el este. A partir de estos valores, se guardan en 2 *features* diferentes los valores de seno y coseno del ángulo. Esto permite al modelo relacionar direcciones de viento similares.

Codificación de la fecha: Se conservó únicamente el mes ya que la estación del año tiene una incidencia fuerte en el clima. Para su codificación se utilizó OneHotEncoding.

Codificación de variables booleanas: Se transformaron en valores numéricos 1/0.

Análisis de features: Se utilizó la matriz de correlación para estudiar la relación entre las variables de entrada y PCA para analizar si existe una jerarquía de features y es posible una reducción de dimensionalidad del dataset. Sin embargo, observando el gráfico acumulado de explicabilidad de la varianza según el número de componentes, vemos que no hay una jerarquía acentuada.



Normalización: Se normalizaron las variables de entrada a media cero y desvío estándar unitario.

4. Entrenamiento de modelos

Se realizaron pruebas con diferentes modelos de clasificación y finalmente se seleccionaron los siguientes 3: Regresión Logística, Random Forest y Gradient Boosting (LGBM), siendo este último el que mejor resultados obtuvo.

Modelo	Accuracy
Regresión Logística	85.43%
Random Forest	86.37%
Gradient Boosting (LGBM)	86.56%

5. Resultados de modelos

Se detallan a continuación los resultados del modelo con mejor desempeño (LGBM Classifier).

	Valor real positivo *(Llovió)*	Valor real negativo *(No llovió)*
Predicción positiva *(Lloverá)*	True Positive **2135**	False Positive **692**
Predicción negativa *(No lloverá)*	False Negative **1586**	True Negative **12538**

Accuracy: 86,56%
Precision: 75,52%
Recall: 57,38%

Utilizando el método *classification_report* de la *librería scikit-learn* se obtiene la siguiente tabla:

```
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
False	0.89	0.95	0.92	13230
True	0.76	0.57	0.65	3721
accuracy			0.87	16951
macro avg	0.82	0.76	0.78	16951
weighted avg	0.86	0.87	0.86	16951

6. Conclusiones

Se lograron aplicar los conceptos aprendidos durante el cursado de la asignatura a un caso práctico real y se obtuvieron resultados muy buenos.

A medida que se desarrolló el trabajo, se intentaron diferentes métodos de codificación de variables categóricas y de imputación de datos faltantes y se analizó su impacto en el número de features, velocidad de entrenamiento y desempeño de los modelos seleccionados.

Actualmente, el código se encuentra con la configuración de mejor desempeño, pero están comentadas a lo largo del código algunas alternativas que fueron testeadas, pero dieron peores resultados.