Informe de Actividad Práctica

Análisis Exploratorio de Datos Breast Cancer Wisconsin (Diagnostic)

Ignacio Ramírez
Antonia Montecinos
Cristian Vergara

Docente: Michael Miranda
Octubre de 2025

Semana 3

Índice

1. Introducción

El presente informe documenta la actividad práctica correspondiente a la Semana 4 del curso INFB6052 - Herramientas para la Ciencia de Datos. El propósito de esta actividad es aplicar los fundamentos de las librerías de Python para analítica de datos, como Pandas, NumPy, Matplotlib y Seaborn, en un flujo de trabajo de Análisis Exploratorio de Datos (EDA).

Para llevar a cabo este análisis, se ha seleccionado el dataset *Breast Cancer Wisconsin (Diagnostic)*, proveniente del repositorio UCI Machine Learning Repository. El objetivo es realizar una exploración básica que incluye la carga, limpieza, visualización y análisis inicial del dataset. A través de este proceso, se busca comprender la estructura de los datos, evaluar su calidad, visualizar las distribuciones de variables clave e identificar patrones y correlaciones preliminares que podrían ser útiles para un futuro modelo de clasificación (benigno vs. maligno).

Fuente del dataset: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)

El dataset contiene 569 instancias y 30 variables numéricas derivadas de imágenes digitalizadas de células mamarias. La variable objetivo (`diagnosis`) clasifica las muestras en `M` (Maligno) y `B` (Benigno).

2. Desarrollo y Evidencias

2.1. Descripción del Dataset

- Instancias: 569.
- Clases: 2 (Maligno / Benigno).
- Atributos: 30 numéricos (medidas de textura, forma, concavidad y dimensión fractal).
- Identificador: `id` (no usado para modelado).

2.2. Carga de Librerías y Configuración Inicial

Se utilizaron `pandas`, `numpy`, `matplotlib` y `seaborn`. Se configuró un estilo uniforme para facilitar la lectura visual de distribuciones y relaciones.

Evidencia

```
# Celda 1: Importa librerías base para el EDA y configura estilo de gráficos
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
sns.set_theme(style='whitegrid')
plt.rcParams['figure.figsize']=(10,6)
print('¡Hola UTEM - Ingenieria Civil en Ciencia de Datos!')
```

```
¡Hola UTEM - Ingenieria Civil en Ciencia de Datos!
```

2.3. Descarga y Carga de Datos

El archivo comprimido se descargó automáticamente si no existía localmente. Se asignaron nombres de columnas según la documentación original del repositorio UCI para una manipulación legible de los datos.

Evidencia

```
# Celda 2: Descarga (una sola vez) y carga el dataset con nombres de columnas
import os, zipfile, urllib.request
URL = 'https://.../breast+cancer+wisconsin+diagnostic.zip'
ZIP = 'breast_cancer.zip'
DIR = 'breast_cancer_data'
# ... (código de descarga) ...
cols = ['id','diagnosis', 'radius_mean', ... , 'fractal_dimension_worst']
path = os.path.join(DIR,'wdbc.data')
df = pd.read_csv(path, header=None, names=cols)
print(f'Filas: {df.shape[0]}  Columnas: {df.shape[1]}')
df.head()
```

```
Usando datos descargados previamente.
Filas: 569  Columnas: 32
```

	id	diagnosis	radius_mean	texture_mean	...
0	842302	M	17.99	10.38	...
1	842517	M	20.57	17.77	...
2	84300903	M	19.69	21.25	...
3	84348301	M	11.42	20.38	...
4	84358402	M	20.29	14.34	...

```
[5 rows x 32 columns]
```

2.4. Exploración Estructural

Se verificó la forma del DataFrame, los tipos de datos de cada columna y la cantidad de valores no nulos, confirmando una estructura limpia y sin datos faltantes aparentes.

- Forma: (569, 32) considerando `id`, `diagnosis` y 30 atributos numéricos.
- Tipos: 30 `float64`, 1 `int64`, 1 `object`.

Evidencia

```
# Celda 3: Vista de forma general y tipos de datos
print('Shape:', df.shape)
print('Primeras columnas:', df.columns[:6].tolist())
df.info()
```

```
Shape: (569, 32)
Primeras columnas: ['id', 'diagnosis', 'radius_mean',
'texture_mean', 'perimeter_mean', 'area_mean']
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 32 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   id                       569 non-null    int64
 1   diagnosis                569 non-null    object
 2   radius_mean              569 non-null    float64
...
 31  fractal_dimension_worst  569 non-null    float64
dtypes: float64(30), int64(1), object(1)
memory usage: 142.4+ KB
```

2.5. Calidad de Datos

Se confirmó que el dataset no contiene valores nulos ni filas duplicadas exactas. Esto simplifica el preprocesamiento, ya que no se requieren estrategias de imputación o eliminación de registros.

Evidencia

```
# Celda 4: Chequeo de nulos y duplicados
print('Valores nulos totales:', df.isna().sum().sum())
print('Duplicados exactos:', df.duplicated().sum())
```

```
Valores nulos totales: 0
Duplicados exactos: 0
```

2.6. Análisis Descriptivo y de Distribución

Se obtuvieron estadísticas descriptivas para las variables numéricas y se analizó la distribución de la variable objetivo `diagnosis`.

Evidencia: Estadísticas Descriptivas

```
# Celda 5A: Estadísticas descriptivas de variables numéricas
df.describe()
```

```
                 id  radius_mean  texture_mean  ...
count  5.690000e+02   569.000000    569.000000  ...
mean   3.037183e+07    14.127292     19.289649  ...
std    1.250206e+08     3.524049      4.301036  ...
min    8.670000e+03     6.981000      9.710000  ...
25%    8.692180e+05    11.700000     16.170000  ...
50%    9.060240e+05    13.370000     18.840000  ...
75%    8.813129e+06    15.780000     21.800000  ...
max    9.113205e+08    28.110000     39.280000  ...
```

Evidencia: Distribución de Clases

```
# Celda 5B: Distribución absoluta y porcentual de la variable objetivo
print('Distribución diagnosis:')
print(df['diagnosis'].value_counts())
print('Porcentajes:')
print((df['diagnosis'].value_counts(normalize=True)*100).round(2))
```

```
Distribución diagnosis:
diagnosis
B    357
M    212
Name: count, dtype: int64
Porcentajes:
diagnosis
B    62.74
M    37.26
Name: proportion, dtype: float64
```

2.7. Visualizaciones Univariadas

Distribución de la Variable Objetivo

El gráfico circular (Figura 1) muestra la proporción de casos Benignos (B) vs Malignos (M). Existe un ligero desbalance a favor de la clase Benigna (aprox. 63 % B / 37 % M).

- No hay un desbalance extremo, pero conviene monitorear métricas más allá de *accuracy* (como *precision/recall*) en el modelado.
- La presencia suficiente de la clase minoritaria (Maligna) permite el entrenamiento supervisado sin necesidad inmediata de técnicas de re-muestreo.

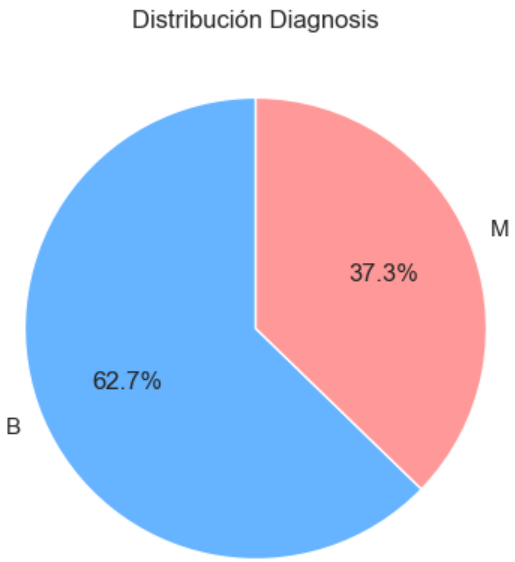

Figura 1: Distribución de la variable objetivo (diagnosis).

Histogramas de Variables Seleccionadas

Los histogramas (Figura 2) revelan las distribuciones de cuatro variables representativas:

- `radius_mean`: Distribución sesgada a la derecha. La cola alta sugiere la presencia de casos con radios mayores, potencialmente malignos.
- `texture_mean`: Distribución más simétrica y concentrada, con variabilidad moderada.
- `area_mean`: Fuerte asimetría positiva, indicando que valores altos corresponden a masas más grandes y posiblemente malignas.
- `concavity_mean`: Muchos valores cercanos a cero y una minoría de valores altos. La concavidad elevada, una característica de bordes irregulares, podría ser muy discriminante.

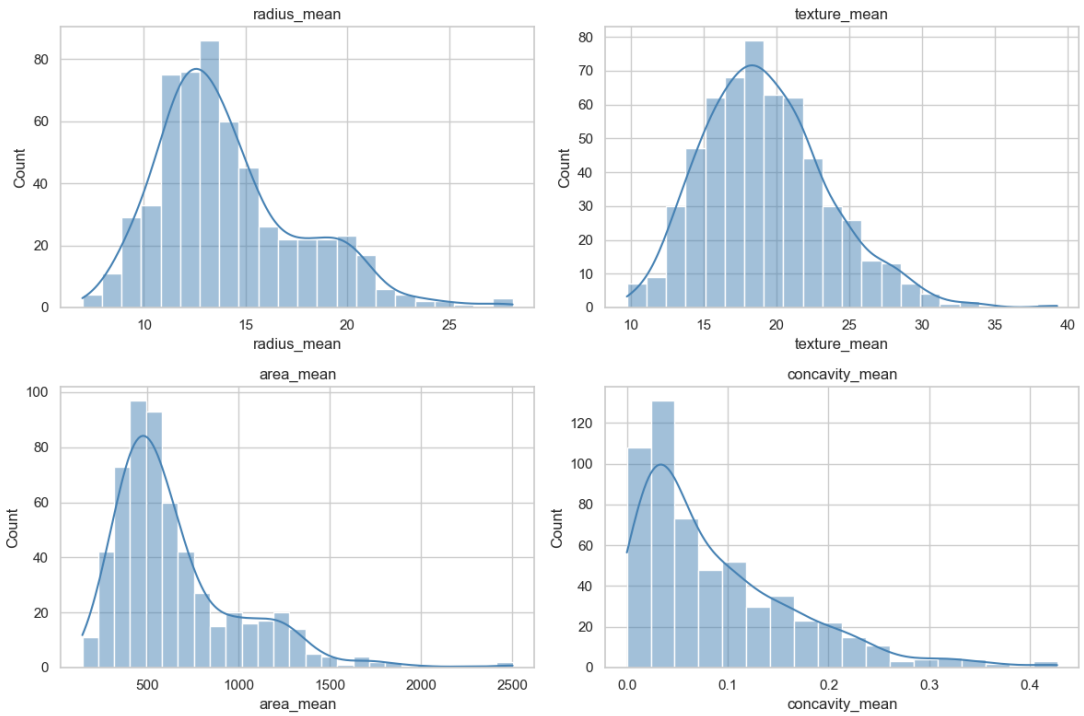

Figura 2: Distribución de variables seleccionadas con estimación de densidad (KDE).

2.8. Visualizaciones Bivariadas

Se compararon variables entre clases para detectar patrones de separación.

Análisis de `radius_mean` por Clase

El boxplot (Figura 3) muestra que la mediana y el rango intercuartílico de los casos Malignos (M) están desplazados hacia valores mayores respecto a los Benignos (B). Esto sugiere que el tamaño medio del núcleo celular es un buen indicador preliminar de malignidad.

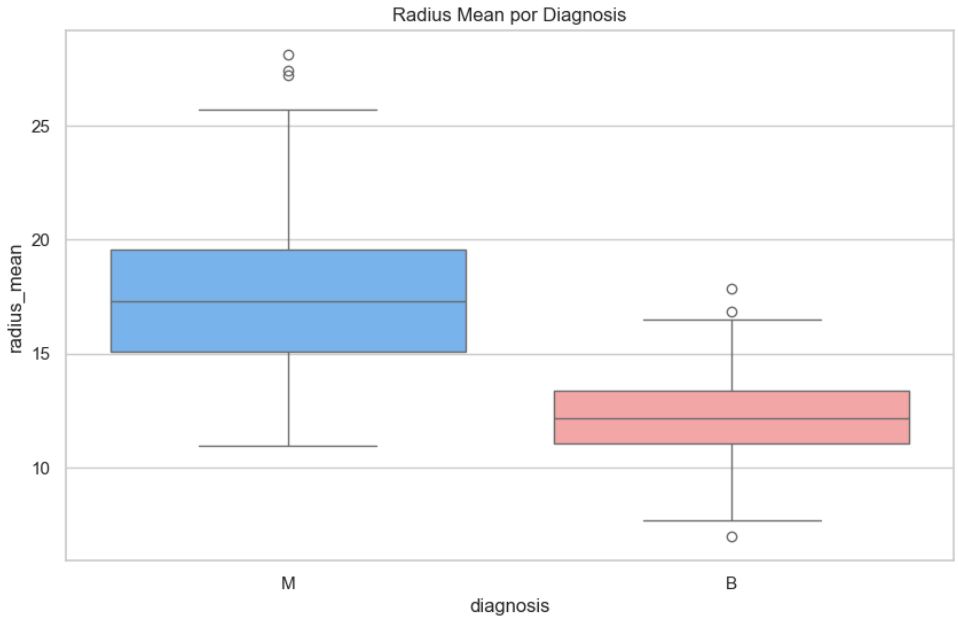

Figura 3: Distribución de 'radius_mean' por clase de diagnóstico.

Relación entre Radio y Área

El gráfico de dispersión (Figura 4) evidencia una relación casi monótona creciente entre `radius_mean` y `area_mean`. Los puntos Malignos (rojo) tienden a concentrarse en la región superior derecha (mayor tamaño), mostrando un patrón separable. La alta relación visual indica que ambas variables aportan información redundante (posible multicolinealidad).

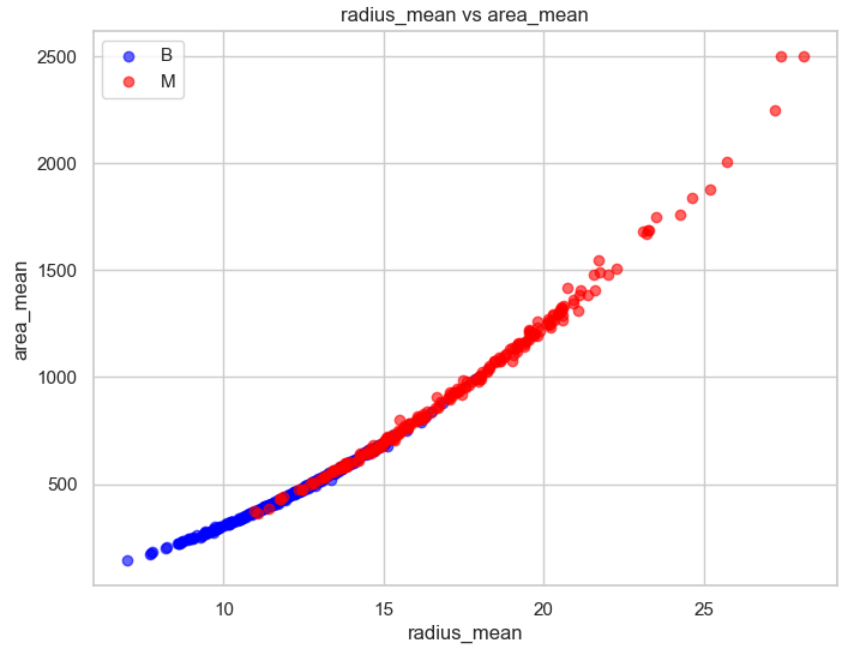

Figura 4: Relación entre 'radius_mean' y 'area_mean', coloreado por diagnóstico.

2.9. Análisis de Correlaciones

Se analizó la matriz de correlaciones numéricas entre los atributos con sufijo `_mean`. El análisis revela que:

- **Alta colinealidad:** Existe una correlación muy alta (cercana a 1.0) entre `radius_mean`, `perimeter_mean`, y `area_mean`, lo que indica una fuerte redundancia. Estas variables miden aspectos del tamaño de forma muy relacionada.
- **Variables discriminantes:** El cálculo de la correlación con la variable objetivo (codificada como 1 para Maligno y 0 para Benigno) muestra que `concave_points_mean`, `perimeter_mean`, `radius_mean`, y `concavity_mean` tienen los coeficientes más altos, sugiriendo su alta importancia para la clasificación.
- **Señales secundarias:** Atributos como `smoothness_mean` y `fractal_dimension_mean` presentan correlaciones más bajas, lo que podría indicar que aportan información distinta y complementaria.

Evidencia

```
# Correlación de atributos 'mean' con 'diagnosis_num' (M=1, B=0)
corr_target.head(8)
```

```
concave_points_mean       0.776614
perimeter_mean            0.742636
radius_mean               0.730029
area_mean                 0.708984
concavity_mean            0.696360
compactness_mean          0.596534
smoothness_mean           0.358560
symmetry_mean             0.330499
Name: diagnosis_num, dtype: float64
```

2.10. Hallazgos Principales

1. El dataset está completo y no presenta valores faltantes ni duplicados, simplificando el proceso de preparación.
2. Existe un desbalance moderado de clases (63 % vs 37 %), lo que justifica el uso futuro de métricas como *precision*, *recall*, y *F1-score* para una evaluación robusta del modelo.
3. Las variables relacionadas con el tamaño (`radius`, `perimeter`, `area`) son altamente redundantes, lo que sugiere la necesidad de seleccionar solo una de ellas para evitar problemas de multicolinealidad en modelos lineales.
4. Las características de concavidad y puntos cóncavos son fuertes candidatas para discriminar entre tumores benignos y malignos.
5. Una posible selección inicial de atributos para un modelo simple podría incluir una variable de tamaño (`radius_mean`), una de forma (`concavity_mean`), y una de textura (`texture_mean`).

3. Conclusiones

El Análisis Exploratorio de Datos (EDA), realizado como parte de la actividad práctica de la Semana 3, ha permitido cumplir con el objetivo de aplicar las librerías fundamentales de Python para la analítica de datos. A través de este proceso, se lograron identificar estructuras, patrones y redundancias clave en el dataset *Breast Cancer Wisconsin (Diagnostic)*.

El análisis confirma que las variables geométricas y de forma, en particular las medidas de tamaño y concavidad, son predictores potentes para la diferenciación entre masas benignas y malignas. Adicionalmente, se verificó la alta calidad de los datos (ausencia de valores nulos y duplicados), lo cual simplifica significativamente las etapas posteriores de modelado.

Este EDA sienta las bases para la siguiente fase del ciclo de vida de la ciencia de datos: el modelado predictivo. Como próximos pasos, y en línea con los futuros temas del curso, se recomienda:

- Realizar una codificación binaria de la variable `diagnosis`.
- Aplicar un escalado o normalización a las variables numéricas (p.ej., `StandardScaler`) para asegurar que los modelos no sean sesgados por la magnitud de las distintas características.
- Implementar una estrategia de selección de características para reducir la redundancia y mitigar la multicolinealidad identificada.
- Evaluar un modelo de clasificación simple, como Regresión Logística, utilizando el conjunto reducido de variables.

Referencias

- UCI Machine Learning Repository. Breast Cancer Wisconsin (Diagnostic). https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)
- Documentación Pandas: https://pandas.pydata.org/docs/
- Documentación Seaborn: https://seaborn.pydata.org/
- Documentación Matplotlib: https://matplotlib.org/stable/