Análisis y visualización con pandas, nivel intermedio

Departamento de Física.

Corodinadora: C Loyola
Profesores C Femenías / F Bugini / D Basantes

Primer Semestre 2025

Universidad Andrés Bello
Departamento de Física y Astronomía

Universidad Andrés Bello®

python powered

Índice

Fuentes de datos

Fuentes de datos $\in$ Fuentes para la práctica

- Meteoritos NASA (limitado a 5 000 filas)
  `https://data.nasa.gov/resource/y77d-th95.csv?\protect\TU\textdollar\protect\TU\textdollarlimit=5000`
- Exoplanetas Seaborn
  `https://raw.githubusercontent.com/mwaskom/seaborn-data/master/planets.csv`
- Masas PDG (extracto)
  `https://raw.githubusercontent.com/particle-physics-book/data/master/pdg_mass.csv`

Copia estas direcciones directamente en tu notebook.

Carga y limpieza

Carga y limpieza ∈ Lectura de los tres conjuntos

```
met   = pd.read_csv("https://data.nasa.gov/resource/y77d-th95.csv?$$limit=
pln   = pd.read_csv("https://raw.githubusercontent.com/mwaskom/seaborn-dat
pdg   = pd.read_csv("https://raw.githubusercontent.com/particle-physics-bo
```

‘read_csv‘ descarga y parsea automáticamente; no requiere bibliotecas externas.

Carga y limpieza ∈ Eliminar faltantes y outliers en meteoritos

```
met = met[met.mass.notna()].copy()
met["massKg"] = met.mass.astype(float) / 1000     # gramos a kg
met = met[met.massKg < 1e4]                       # quitar >10 t
```

Tres líneas bastan para dejar el conjunto listo para análisis.

Diagnóstico de faltantes

Diagnóstico de faltantes $\in$ Porcentaje de valores nulos

```
faltantes = met.isna().mean() * 100
faltantes.sort_values(ascending=False).head()
```

Así priorizas las columnas que necesitan limpieza.

Agrupaciones avanzadas

Agrupaciones avanzadas ∈ Resumen por década

```
met["year"] = pd.to_datetime(met.year,
↪  errors="coerce").dt.year
decade = (met.groupby(met.year // 10 * 10)["massKg"]
              .agg(conteo='size', promedio='mean',
              ↪  total='sum'))
decade.head()
```

Agrupar por transformación matemática del año genera la tabla por décadas.

Agrupaciones avanzadas ∈ Pivot table visual

```
pln["decada"] = pln.year // 10 * 10
tabla = pln.pivot_table(index="method", columns="decada",
                        values="distance", aggfunc="count")
plt.imshow(tabla.fillna(0), aspect='auto')
plt.colorbar(label="número de exoplanetas");
↪  plt.yticks(range(len(tabla)), tabla.index)
plt.xlabel("decada"); plt.show()
```

La matriz calor ayuda a descubrir métodos dominantes en cada período.

Reshape melt y pivot

Reshape melt y pivot ∈ Convertir tabla ancha a larga

```
wide = decade.reset_index().rename(columns={"year":"decada"})
long = wide.melt(id_vars="decada",
                 var_name="metrica", value_name="valor")
long.head()
```

La forma “larga” es conveniente para bibliotecas de visualización como Seaborn.

Series temporales

Series temporales ∈ Resample anual de masa caída

```
metTs = (met.set_index(pd.to_datetime(met.year, format='%Y'))
               .massKg.resample("1Y").sum())
metTs.plot(title="Masa total de meteoritos por año");
↪  plt.ylabel("kg"); plt.show()
```

‘resample‘ es la versión temporal de ‘groupby‘.

Merge ilustrativo

Merge ilustrativo ∈ Normalizar masas de planetas

```
proton = pdg.loc[pdg.symbol == "p", "mass_GeV"].iat[0]
pln["massOverMp"] = pln.mass / proton
pln[["name", "mass", "massOverMp"]].head()
```

Combinamos física de partículas con astronomía en un solo DataFrame.

Visualización integrada

Visualización integrada ∈ Scatters y densidad KDE

```
pln.plot.scatter(x="mass", y="orbital_period",
                 alpha=0.4, logx=True, logy=True,
                 title="Masa vs periodo orbital")
pln["distance"].plot.kde(title="Densidad de distancias"); plt.show()
```

La API integrada evita llamadas adicionales a Matplotlib.

Estilos

Estilos ∈ Aplicar estilo SciencePlots

```
plt.style.use("science")
decade.total.plot(marker='o')
plt.title("Toneladas de meteoritos por década");
↪  plt.ylabel("kg"); plt.show()
```

El resultado queda listo para copiar en un informe o paper.

Exportar resultados

Exportar resultados $\in$ Guardar en disco

```
decade.to_csv("meteoritos_por_decada.csv", index=True)
pln.to_parquet("planetsCompact.parquet", index=False)
```

Parquet brinda alta compresión y lectura más veloz que CSV.

Rendimiento

Rendimiento ∈ Consejos de eficiencia

- Columnas con pocos valores → ‘astype(”category”)‘ para reducir memoria.
- ‘read_csv‘ con ‘chunksize‘ permite procesar archivos gigantes en porciones.
- ‘eval‘ y ‘query‘ compilan expresiones y evitan copias intermedias.

Mini proyecto en parejas

Mini proyecto en parejas $\in$ Instrucciones de treinta minutos

1. Clasifica cada meteorito en continente según latitud y longitud.
2. Histograma logarítmico de masa por continente.
3. Pivot table de cantidad de impactos por década y continente.
4. Exporta la pivot table a un archivo Excel para compartir.

Guía rápida de solución

Guía rápida de solución ∈ Fragmento de código orientativo

```
bins = np.logspace(-2, 4, 30)
for cont in met.continent.unique():
    met.loc[met.continent == cont, "massKg"].plot.hist(
        bins=bins, alpha=0.5, label=cont)
plt.xscale("log"); plt.legend(); plt.show()
```

Recursos recomendados

Recursos recomendados ∈ Para profundizar

- Cheat-sheet oficial: `https://pandas.pydata.org/Pandas_Cheat_Sheet.pdf`
- Libro “Python Data Science Handbook”, capítulos ocho a once.
- Lista Awesome Public Datasets en GitHub.

Preguntas frecuentes

Preguntas frecuentes $\in$ FAQ en vivo

- **Warning SettingWithCopy** → usa ‘loc‘ y ‘copy‘.
- CSV muy grande → convierte a Parquet y gana 5-10 x en velocidad.
- Problemas en merge → usa ‘validate=”one_to_one”‘ para asegurar unicidad.

Próximos temas

Próximos temas $\in$ En la próxima semana

Estadística básica, bootstrap y visualización con Seaborn aplicadas a datos físicos.

¡Fin de la sesión!