datos_sinteticos

July 8, 2025

1 CREACIÓN DE DATOS SINTÉTICOS

[1]:
```
# INSTALACIÓN DE LIBRERÍAS
%pip install pandas
%pip install numpy
```

```
Requirement already satisfied: pandas in
c:\users\carolina\documents\proyectos_programacion\predestu\.venv\lib\site-
packages (2.3.1)
Requirement already satisfied: numpy>=1.26.0 in
c:\users\carolina\documents\proyectos_programacion\predestu\.venv\lib\site-
packages (from pandas) (2.3.1)
Requirement already satisfied: python-dateutil>=2.8.2 in
c:\users\carolina\documents\proyectos_programacion\predestu\.venv\lib\site-
packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in
c:\users\carolina\documents\proyectos_programacion\predestu\.venv\lib\site-
packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in
c:\users\carolina\documents\proyectos_programacion\predestu\.venv\lib\site-
packages (from pandas) (2025.2)
Requirement already satisfied: six>=1.5 in
c:\users\carolina\documents\proyectos_programacion\predestu\.venv\lib\site-
packages (from python-dateutil>=2.8.2->pandas) (1.17.0)
Note: you may need to restart the kernel to use updated packages.
Requirement already satisfied: numpy in
c:\users\carolina\documents\proyectos_programacion\predestu\.venv\lib\site-
packages (2.3.1)
Note: you may need to restart the kernel to use updated packages.
```

[2]:
```
# IMPORTACIÓN DE LIBRERÍAS
import pandas as pd
import numpy as np
import random
```

[3]:
```
# CONFIGURACIÓN INICIAL
np.random.seed(42)
random.seed(42)
```

[41]:
```
# CARGAR DATASETS
df_escuelas = pd.read_csv('../fuentes_datos/datasets_finales/CSV/escuela.csv',␣
 ↪delimiter=';')
df_semestres = pd.read_csv('../fuentes_datos/datasets_finales/CSV/semestre.
 ↪csv', delimiter=';')
df_planes = pd.read_csv('../fuentes_datos/datasets_finales/CSV/plan.csv',␣
 ↪delimiter=';')
df_estudiantes = pd.read_csv('../fuentes_datos/datasets_finales/CSV/estudiante.
 ↪csv', delimiter=';')
df_cursos = pd.read_csv('../fuentes_datos/datasets_finales/CSV/curso.csv',␣
 ↪delimiter=';')
df_reprobaciones = pd.read_csv('../fuentes_datos/datasets_finales/CSV/
 ↪reprobacion.csv', delimiter=';')
df_tutorias = pd.read_csv('../fuentes_datos/datasets_finales/CSV/tutoria.csv',␣
 ↪delimiter=';')
```

[17]:
```
# PARAMETROS BASE
escuelas = df_escuelas.set_index('id_escuela')['nombre'].to_dict()
planes = df_planes.set_index('id_escuela')['anio_inicio'].to_dict()
semestres = df_semestres['nombre'].astype(str).tolist()
```

[18]:
```
print("DICCIONARIO ESCUELAS:")
print(escuelas)
print("\nDICCIONARIO PLANES:")
print(planes)
print("\nLISTA SEMESTRES:")
print(semestres)
```

```
DICCIONARIO ESCUELAS:
{1: 'Ingeniería de Sistemas', 2: 'Ingeniería de Software', 3: 'Ciencias de la
Computación'}

DICCIONARIO PLANES:
{1: 2024, 2: 2024, 3: 2024}

LISTA SEMESTRES:
['20190', '20191', '20192', '20200', '20201', '20202', '20210', '20211',
'20212', '20220', '20221', '20222', '20230', '20231', '20232', '20240', '20241',
'20242', '20250', '20251', '20190', '20191', '20192', '20200', '20201', '20202',
'20210', '20211', '20212', '20220', '20221', '20222', '20230', '20231', '20232',
'20240', '20241', '20242', '20250', '20251', '20241', '20242', '20250', '20251']
```

[19]:
```
# CREACIÓN DE CALIFICACIONES SINTÉTICAS
# Simular cursos tomados por estudiante (2021-2025)
cursos_tomados = []

for _, est in df_estudiantes.iterrows():
```

```
    for sem in semestres:
        if int(sem[:4]) >= est.anio_ingreso:
            for _ in range(random.randint(3, 7)):
                curso = df_cursos.sample(1).iloc[0]  # elige curso al azar
                id_curso = curso["id_curso"]
                id_ciclo = curso["id_ciclo"]
                t_aprob = curso["t_aprobacion"]  # entre 0 y 1

                # Simular si aprueba o no según tasa
                aprueba = random.random() < t_aprob

                # Nota entre 11-20 si aprueba, entre 0-10 si desaprueba
                if aprueba:
                    nota = np.clip(np.random.normal(14.5, 2), 11, 20)
                else:
                    nota = np.clip(np.random.normal(9, 2.5), 0, 10)

                id_semestre = int(sem)
                cursos_tomados.append({
                    "id_estudiante": est.id_estudiante,
                    "id_curso": id_curso,
                    "id_semestre": id_semestre,
                    "nota": int(round(nota)),
                    "id_ciclo": id_ciclo
                })
```

[23]:
```
# GUARDAR EN UN DATAFRAME
df_cursos_tomados2 = pd.DataFrame(cursos_tomados)
```

[26]:
```
print("DATASET CURSOS_TOMADOS 2021-2025:")
print(df_cursos_tomados2.tail(5))  # Muestra los últimos 5 registros
```

```
DATASET CURSOS_TOMADOS 2021-2025:
        id_estudiante id_curso  id_semestre  nota  id_ciclo
357518       18200108  2010103        20250    14        13
357519       18200108  2020501        20251     8        48
357520       18200108  2010205        20251     9        14
357521       18200108   201003        20251    18         2
357522       18200108  2010501        20251    12        17
```

[27]:
```
# IDENTIFICAR CURSOS_TOMADOS CON ID_CICLO 9
df_cursos_tomados_escuela3 = df_cursos_tomados2[df_cursos_tomados2['id_ciclo']␣
 ↪== 9]
```

[29]:
```
print("N REGISTROS DEL DATASET CURSOS_TOMADOS CON CICLO 9:")
print(len(df_cursos_tomados_escuela3))
print("\nDATASET CURSOS_TOMADOS CON CICLO 9:")
print(df_cursos_tomados_escuela3.head())  # Muestra los primeros 5 registros
```

```
N REGISTROS DEL DATASET CURSOS_TOMADOS CON CICLO 9:
3601

DATASET CURSOS_TOMADOS CON CICLO 9:
     id_estudiante id_curso  id_semestre  nota  id_ciclo
66        20200111   208007        20232    14         9
108       20200111   208003        20192    18         9
208       20200111   208003        20251    13         9
469       22200067   208008        20221    14         9
516       22200067   208001        20250    17         9
```

[30]:
```
# GUARDAR EL DATASET NUEVO DE CURSOS_TOMADOS (2019-2025))
df_cursos_tomados2.to_csv('../fuentes_datos/datasets_finales/CSV/
 ↪curso_tomado_completo.csv', index=False, sep=';')
```

[31]:
```
# CREACIÓN DE REPETICIONES SINTÉTICAS
# Cada estudiante puede tener repitencias si su nota fue < 11
repitencias = df_cursos_tomados2[df_cursos_tomados2["nota"] < 11]
repitencias = repitencias.groupby(["id_estudiante", "id_curso", "id_semestre"]).
 ↪size().reset_index(name="n_reprobaciones")
```

[32]:
```
# GUARDAR EN UN DATAFRAME
df_reprobaciones2 = pd.DataFrame(repitencias)
```

[33]:
```
print("DATASET REPROBACIONES 2021-2025:")
print(df_reprobaciones2.tail(5))  # Muestra los últimos 5 registros
```

```
DATASET REPROBACIONES 2021-2025:
       id_estudiante id_curso  id_semestre  n_reprobaciones
92509       25200216  20W0702        20202                1
92510       25200216  20W0905        20202                1
92511       25200216  20W0905        20222                1
92512       25200216  20W1003        20241                1
92513       25200216   INO106        20190                1
```

[36]:
```
# UNIR DATASETS DE REPROBACIONES (HISTÓRICO Y NUEVO)
df_reprobaciones3 = pd.concat([df_reprobaciones, df_reprobaciones2],␣
 ↪ignore_index=True)
```

[38]:
```
# Sumar reprobaciones por estudiante y curso, conservando el último semestre␣
 ↪donde reprobó
df_reprobaciones4 = df_reprobaciones3.groupby(["id_estudiante", "id_curso"],␣
 ↪as_index=False).agg({
    "n_reprobaciones": "sum",
    "id_semestre": "max"  # Último semestre con reprobación
})
```

[40]:
```
# GUARDAR EL DATASET NUEVO DE REPROBACIONES (2019-2025))
df_reprobaciones4.to_csv('../fuentes_datos/datasets_finales/CSV/
 ↪reprobacion_completo.csv', index=False, sep=';')
```

[42]:
```
# IDENTIFICACIÓN DE NUM_RES_AUTORIZA DEL DATASET DE TUTORIAS
df_tutorias['num_res_autoriza'].unique().tolist()  # Muestra los valores únicos␣
 ↪de la columna num_res_autoriza
```

[42]:
```
['RD N°138-D-FISI-19                    ',
 'RD 125-D-FISI-19                      ',
 'RD125-D-FISI-19                       ',
 'RD 125-D-FISI19                       ',
 'RD 138-D-FISI-19                      ',
 'RD 405-D-FISI-2019                    ',
 'RD 164-D-FISI-19                      ',
 'RD 174-D-FISI-19                      ',
 'RD N°126-D-FISI-19                    ',
 'RD 125-D-FISI-2019                    ',
 'RD 138-D-FISI-2019                    ',
 'RD N° 00126-D-FISI                    ',
 'RD N°174-D-FISI-19                    ',
 'RD N 126-D-FISI-19                    ',
 'RD 125-D-19                           ',
 'RD 126-D-FISI-19                      ',
 'RD 126-D-FISI-2019                    ',
 'RD 126-D-FISI19                       ',
 'RD126-D-FISI-19                       ',
 'RD 139-D-FISI-19                      ',
 'RD 139-D-FISI-2019                    ',
 'RD139-D-FISI-19                       ',
 'RD 165-D-FISI-19                      ',
 'RD 175-D-FISI-19                      ',
 'RD 406-D-FISI-2019                    ',
 'RD 405-D-FISI-19                      ',
 'RD 405-FISI-2019                      ',
 'RD  716-D-FISI-19                     ',
 'RD 00744-D-FISI-19                    ',
 'RD 405 D-FISI-2019                    ',
 'RD 716-D-FISI-19                      ',
 'RD 406 D-FISI-2019                    ',
 'RD 00406-D-FISI                       ',
 'RD 406-D-FISI-19                      ',
 'RD 406-FISI-2019                      ',
 'RD 406-D-FISI-2016                    ',
 '716-D-FISI-2019                       ',
 'RD 107-D-FISI-2020                    ',
 'RD 107-D-FISI2020                     ',
```

```
 'RD 000439-2020-D                          ',
 'RD 00439-2020-D                           ',
 'RD 000451-2020-D                          ',
 'RD 00439                                  ',
 'RD 000445-2020-D                          ',
 'RD 000539-2020-D-                         ',
 nan]
```

[43]:
```
# QUITAR ESPACIOS EN BLANCO DE NUM_RES_AUTORIZA
df_tutorias['num_res_autoriza'] = df_tutorias['num_res_autoriza'].str.strip()
```

[44]:
```
# IDENTIFICACIÓN DE NUM_RES_AUTORIZA DEL DATASET DE TUTORIAS
df_tutorias['num_res_autoriza'].unique().tolist()  # Muestra los valores únicos␣
 ↪de la columna num_res_autoriza
```

[44]:
```
['RD N°138-D-FISI-19',
 'RD 125-D-FISI-19',
 'RD125-D-FISI-19',
 'RD 125-D-FISI19',
 'RD 138-D-FISI-19',
 'RD 405-D-FISI-2019',
 'RD 164-D-FISI-19',
 'RD 174-D-FISI-19',
 'RD N°126-D-FISI-19',
 'RD 125-D-FISI-2019',
 'RD 138-D-FISI-2019',
 'RD N° 00126-D-FISI',
 'RD N°174-D-FISI-19',
 'RD N 126-D-FISI-19',
 'RD 125-D-19',
 'RD 126-D-FISI-19',
 'RD 126-D-FISI-2019',
 'RD 126-D-FISI19',
 'RD126-D-FISI-19',
 'RD 139-D-FISI-19',
 'RD 139-D-FISI-2019',
 'RD139-D-FISI-19',
 'RD 165-D-FISI-19',
 'RD 175-D-FISI-19',
 'RD 406-D-FISI-2019',
 'RD 405-D-FISI-19',
 'RD 405-FISI-2019',
 'RD  716-D-FISI-19',
 'RD 00744-D-FISI-19',
 'RD 405 D-FISI-2019',
 'RD 716-D-FISI-19',
 'RD 406 D-FISI-2019',
```

```
 'RD 00406-D-FISI',
 'RD 406-D-FISI-19',
 'RD 406-FISI-2019',
 'RD 406-D-FISI-2016',
 '716-D-FISI-2019',
 'RD 107-D-FISI-2020',
 'RD 107-D-FISI2020',
 'RD 000439-2020-D',
 'RD 00439-2020-D',
 'RD 000451-2020-D',
 'RD 00439',
 'RD 000445-2020-D',
 'RD 000539-2020-D-',
 nan]
```

[45]:
```
# REEMPLAZAR VALORES DE NUM_RES_AUTORIZA
df_tutorias['num_res_autoriza'] = df_tutorias['num_res_autoriza'].replace({
    'RD N°138-D-FISI-19': 'RD 138-D-FISI-19',
    'RD125-D-FISI-19': 'RD 125-D-FISI-19',
    'RD 125-D-FISI19': 'RD 125-D-FISI-19',
    'RD 405-D-FISI-2019': 'RD 405-D-FISI-19',
    'RD N°126-D-FISI-19': 'RD 126-D-FISI-19',
    'RD 125-D-FISI-2019': 'RD 125-D-FISI-19',
    'RD 138-D-FISI-2019': 'RD 138-D-FISI-19',
    'RD N° 00126-D-FISI': 'RD 126-D-FISI-19',
    'RD N°174-D-FISI-19': 'RD 174-D-FISI-19',
    'RD N 126-D-FISI-19': 'RD 126-D-FISI-19',
    'RD 125-D-19': 'RD 125-D-FISI-19',
    'RD 126-D-FISI-2019': 'RD 126-D-FISI-19',
    'RD 126-D-FISI19': 'RD 126-D-FISI-19',
    'RD126-D-FISI-19': 'RD 126-D-FISI-19',
    'RD 139-D-FISI-2019': 'RD 139-D-FISI-19',
    'RD139-D-FISI-19': 'RD 139-D-FISI-19',
    'RD 406-D-FISI-2019': 'RD 406-D-FISI-19',
    'RD 000451-2020-D': 'RD 451-D-FISI-20',
    'RD 00439': 'RD 439-D-FISI-20',
    'RD 000445-2020-D': 'RD 445-D-FISI-20',
    'RD 000445-2020-D': 'RD 445-D-FISI-20'
})
```

[46]:
```
# IDENTIFICACIÓN DE NUM_RES_AUTORIZA DEL DATASET DE TUTORIAS
df_tutorias['num_res_autoriza'].unique().tolist()  # Muestra los valores únicos␣
 ↪de la columna num_res_autoriza
```

[46]:
```
['RD 138-D-FISI-19',
 'RD 125-D-FISI-19',
 'RD 405-D-FISI-19',
```

```
 'RD 164-D-FISI-19',
 'RD 174-D-FISI-19',
 'RD 126-D-FISI-19',
 'RD 139-D-FISI-19',
 'RD 165-D-FISI-19',
 'RD 175-D-FISI-19',
 'RD 406-D-FISI-19',
 'RD 405-FISI-2019',
 'RD  716-D-FISI-19',
 'RD 00744-D-FISI-19',
 'RD 405 D-FISI-2019',
 'RD 716-D-FISI-19',
 'RD 406 D-FISI-2019',
 'RD 00406-D-FISI',
 'RD 406-FISI-2019',
 'RD 406-D-FISI-2016',
 '716-D-FISI-2019',
 'RD 107-D-FISI-2020',
 'RD 107-D-FISI2020',
 'RD 000439-2020-D',
 'RD 00439-2020-D',
 'RD 451-D-FISI-20',
 'RD 439-D-FISI-20',
 'RD 445-D-FISI-20',
 'RD 000539-2020-D-',
 nan]
```

[47]:
```python
# REEMPLAZAR VALORES DE NUM_RES_AUTORIZA
df_tutorias['num_res_autoriza'] = df_tutorias['num_res_autoriza'].replace({
    'RD 405-FISI-2019': 'RD 405-D-FISI-19',
    'RD  716-D-FISI-19': 'RD 716-D-FISI-19',
    'RD 00744-D-FISI-19': 'RD 744-D-FISI-19',
    'RD 405 D-FISI-2019': 'RD 405-D-FISI-19',
    'RD 406 D-FISI-2019': 'RD 406-D-FISI-19',
    'RD 00406-D-FISI': 'RD 406-D-FISI-19',
    'RD 406-FISI-2019': 'RD 406-D-FISI-19',
    'RD 406-D-FISI-2016': 'RD 406-D-FISI-19',
    '716-D-FISI-2019': 'RD 716-D-FISI-19',
    'RD 107-D-FISI-2020': 'RD 107-D-FISI-20',
    'RD 107-D-FISI2020': 'RD 107-D-FISI-20',
    'RD 000439-2020-D': 'RD 439-D-FISI-20',
    'RD 00439-2020-D': 'RD 439-D-FISI-20',
    'RD 000539-2020-D-': 'RD 539-D-FISI-20'
})
```

[48]:
```python
# IDENTIFICACIÓN DE NUM_RES_AUTORIZA DEL DATASET DE TUTORIAS
```

```
df_tutorias['num_res_autoriza'].unique().tolist()  # Muestra los valores únicos␣
 ↪de la columna num_res_autoriza
```

[48]:
```
['RD 138-D-FISI-19',
 'RD 125-D-FISI-19',
 'RD 405-D-FISI-19',
 'RD 164-D-FISI-19',
 'RD 174-D-FISI-19',
 'RD 126-D-FISI-19',
 'RD 139-D-FISI-19',
 'RD 165-D-FISI-19',
 'RD 175-D-FISI-19',
 'RD 406-D-FISI-19',
 'RD 716-D-FISI-19',
 'RD 744-D-FISI-19',
 'RD 107-D-FISI-20',
 'RD 439-D-FISI-20',
 'RD 451-D-FISI-20',
 'RD 445-D-FISI-20',
 'RD 539-D-FISI-20',
 nan]
```

[49]:
```
# Función auxiliar para generar un num_res_autoriza sintético pero realista
def generar_resolucion(id_semestre):
    anio = str(id_semestre)[:4][-2:]  # Extrae los dos últimos dígitos del año
    numero = random.randint(100, 800)
    return f"RD {numero}-D-FISI-{anio}"
```

[50]:
```
# CREACIÓN DE TUTORÍAS SINTÉTICAS
tutorias = []

for _, row in df_reprobaciones4.iterrows():
    if random.random() < 0.7:  # 70% probabilidad de tener tutoría
        tutorias.append({
            "id_estudiante": row.id_estudiante,
            "id_semestre": row.id_semestre,
            "tipo_autorizacion": random.choice(['AM', 'TO']),
            "num_res_autoriza": generar_resolucion(row.id_semestre)
        })
```

[51]:
```
# GUARDAR EN UN DATAFRAME
df_tutorias2 = pd.DataFrame(tutorias)
```

[52]:
```
print("DATASET TUTORIAS 2021-2025:")
print(df_tutorias2.tail(5))  # Muestra los últimos 5 registros
```

```
DATASET TUTORIAS 2021-2025:
       id_estudiante  id_semestre tipo_autorizacion  num_res_autoriza
```

```
63910       25200216        20250               TO  RD 184-D-FISI-25
63911       25200216        20231               TO  RD 642-D-FISI-23
63912       25200216        20202               TO  RD 381-D-FISI-20
63913       25200216        20241               AM  RD 730-D-FISI-24
63914       25200216        20190               TO  RD 409-D-FISI-19
```

[53]:
```
# UNIR DATASETS DE TUTORIAS (HISTÓRICO Y NUEVO)
df_tutorias3 = pd.concat([df_tutorias, df_tutorias2], ignore_index=True)
```

[54]:
```
# GUARDAR EL DATASET NUEVO DE TUTORIAS (2019-2025))
df_tutorias3.to_csv('../fuentes_datos/datasets_finales/CSV/tutorias_completo.
  ↪csv', index=False, sep=';')
```