Lab Activity Report
Title: Dataset Generation and Data Exploration
Course: BS in Computer Engineering
Name: Sean Kieran Sain
Date: 04/20/2O25

Objective

The purpose of this lab activity is to demonstrate how to generate a synthetic dataset suitable for machine learning, specifically for neural network classification tasks, and to explore the dataset using Python data science libraries. This includes analyzing feature distributions, detecting patterns, and preparing insights that can be useful for future model development.

Methodology

Using Python and common data science libraries such as pandas, numpy, seaborn, and matplotlib, a dataset was generated using make_classification from scikit-learn. This function was used to simulate a binary classification dataset with the following characteristics:
- Samples: 1000
- Features: 10 (5 informative, 2 redundant)
- Classes: 2
- Random State: 42 (for reproducibility)

The generated dataset was saved as a CSV file (synthetic_dataset.csv) and reloaded for exploration.

Data Exploration

Several steps were performed to understand the structure and nature of the dataset:
- Preview of Data: The first few rows (df.head()) showed numerical features named feature_1 to feature_10, along with a target label (0 or 1).
- Dataset Shape: The dataset consists of 1000 rows and 11 columns.
- Class Distribution: The target column was nearly balanced, with approximately equal counts for both classes.
- Descriptive Statistics: The .describe() method revealed that most features are centered around zero, suggesting standard normalization.
- Correlation Analysis: A heatmap visualized the relationships between features. Some features showed moderate correlation with the target, which may be useful for model input selection.

Visualizations
- Histograms were used to observe the distribution of each feature.
- Boxplots helped detect the presence of outliers.
- Correlation Matrix Heatmap indicated inter-feature relationships and potential redundancy.

Conclusion

This activity successfully demonstrated how to generate and explore a synthetic dataset for binary classification. The dataset is suitable for training neural network models, with well-structured and informative features. The data exploration process provided useful insights that can assist in model development and feature engineering.

In future activities, this dataset may be used for model training, tuning, and evaluation. Furthermore, localization or thematic alignment with real-world Filipino datasets (e.g., agriculture, education) can be explored for contextual learning.

This notebook demonstrates how to generate a synthetic dataset for a Neural Network project and perform basic data exploration.

```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import make_classification
```

```
# Generate synthetic dataset (binary classification)
X, y = make_classification(n_samples=1000, n_features=10,
                           n_informative=5, n_redundant=2,
                           n_classes=2, random_state=42)

# Create a DataFrame
columns = [f'feature_{i+1}' for i in range(X.shape[1])]
df = pd.DataFrame(X, columns=columns)
df['target'] = y

# Save dataset to CSV
df.to_csv("synthetic_dataset.csv", index=False)

print("✅ Dataset Generated and Saved.")
```

```
✅ Dataset Generated and Saved.
```

```
# Load the dataset
df = pd.read_csv("synthetic_dataset.csv")

# Display first 5 rows
df.head()
```

	feature_1	feature_2	feature_3	feature_4	feature_5	feature_6	feature_7	feature_8	feature_9	feature_10	target
0	1.125100	1.178124	0.493516	0.790880	-0.614278	1.347020	1.419515	1.357325	0.966041	-1.981139	1
1	-0.564641	3.638629	-1.522415	-1.541705	1.616697	4.781310	3.190292	-0.890254	1.438826	-3.828748	0
2	0.516313	2.165426	-0.628486	-0.386923	0.492518	1.442381	1.332905	-1.958175	-0.348803	-1.804124	0
3	0.537282	0.966618	-0.115420	0.670755	-0.958516	0.871440	0.508186	-1.034471	-1.654176	-1.910503	1
4	0.278385	1.065828	-1.724917	-2.235667	0.715107	0.731249	-0.674119	0.598330	-0.524283	1.047610	0

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
# Dataset shape
df.shape
```

```
(1000, 11)
```

```
# Check for missing values
df.isnull().sum()
```

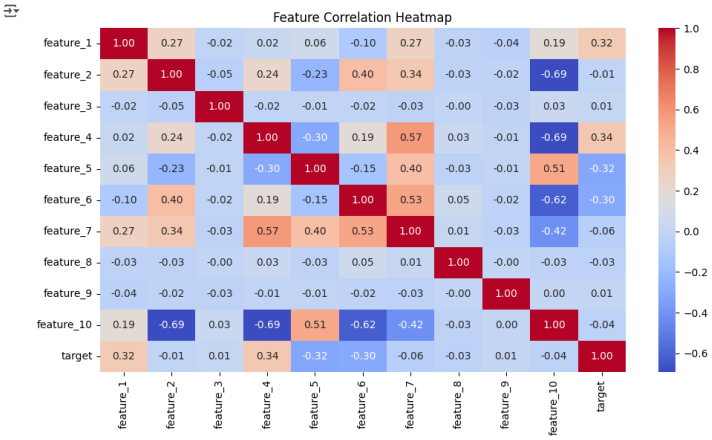
	0
feature_1	0
feature_2	0
feature_3	0
feature_4	0
feature_5	0
feature_6	0
feature_7	0
feature_8	0
feature_9	0
feature_10	0
target	0

dtype: int64

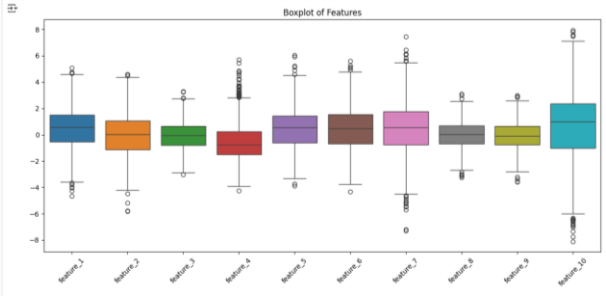
```
# Summary statistics
df.describe()
```

	feature_1	feature_2	feature_3	feature_4	feature_5	feature_6	feature_7	feature_8	feature_9	feature_10	target
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000
mean	0.480472	-0.011035	-0.072376	-0.492447	0.469059	0.499845	0.455850	0.017115	-0.058077	0.631419	0.503000
std	1.583809	1.618548	1.024196	1.518933	1.489777	1.621358	2.000162	1.029048	1.046402	2.806143	0.500241
min	-4.661168	-5.814203	-3.031194	-4.258034	-3.889284	-4.341477	-7.298063	-3.254479	-3.582063	-8.102614	0.000000
25%	-0.542215	-1.132212	-0.779532	-1.504082	-0.609955	-0.674893	-0.776099	-0.676648	-0.744499	-1.034635	0.000000
50%	0.583361	0.029747	-0.041891	-0.760929	0.541561	0.449055	0.553223	0.025772	-0.081367	0.987519	1.000000
75%	1.521656	1.077206	0.644444	0.230285	1.447875	1.531734	1.757939	0.679153	0.659029	2.371964	1.000000
max	5.066061	4.605669	3.276399	5.685693	6.036793	5.608412	7.456970	3.089890	2.986329	7.933944	1.000000

```
# Correlation heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm', fmt='.2f')
plt.title("Feature Correlation Heatmap")
plt.tight_layout()
plt.show()
```




```
# Boxplot of all features
plt.figure(figsize=(12, 6))
sns.boxplot(data=df.iloc[:, :-1])
plt.title("Boxplot of Features")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```




```
# Histogram for one feature
df['feature_1'].hist(bins=30, edgecolor='black')
plt.title("Histogram of Feature 1")
plt.xlabel("Feature 1")
plt.ylabel("Frequency")
plt.grid(False)
plt.show()
```