FDA_Assignment1

May 27, 2025

[150]:
```
%pip install --upgrade typing_extensions
```

```
Note: you may need to restart the kernel to use updated packages.Requirement
already satisfied: typing_extensions in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (4.13.2)



[notice] A new release of pip is available: 23.3 -> 25.1.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

[151]:
```
%pip install torch
%pip install numpy
%pip install pandas
%pip install seaborn
%pip install matplotlib
%pip install scikit-learn
%pip install torch-geometric
```

```
Requirement already satisfied: torch in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (2.7.0)Note: you may need
to restart the kernel to use updated packages.

Requirement already satisfied: filelock in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from torch) (3.18.0)
Requirement already satisfied: typing-extensions>=4.10.0 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from torch) (4.13.2)
Requirement already satisfied: sympy>=1.13.3 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from torch) (1.14.0)
Requirement already satisfied: networkx in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from torch) (3.4.2)
Requirement already satisfied: jinja2 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from torch) (3.1.6)
Requirement already satisfied: fsspec in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from torch) (2025.5.0)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from sympy>=1.13.3->torch)
(1.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in
```

```
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from jinja2->torch)
(3.0.2)


[notice] A new release of pip is available: 23.3 -> 25.1.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

```
Requirement already satisfied: numpy in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (1.26.4)
Note: you may need to restart the kernel to use updated packages.


[notice] A new release of pip is available: 23.3 -> 25.1.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

```
Requirement already satisfied: pandas in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (2.2.3)
Requirement already satisfied: numpy>=1.23.2 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from pandas) (1.26.4)
Requirement already satisfied: python-dateutil>=2.8.2 in
c:\users\gulnar\appdata\roaming\python\python311\site-packages (from pandas)
(2.8.2)
Requirement already satisfied: pytz>=2020.1 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from pandas) (2025.2)
Requirement already satisfied: six>=1.5 in
c:\users\gulnar\appdata\roaming\python\python311\site-packages (from python-
dateutil>=2.8.2->pandas) (1.16.0)
Note: you may need to restart the kernel to use updated packages.


[notice] A new release of pip is available: 23.3 -> 25.1.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

```
Requirement already satisfied: seaborn in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (0.13.2)Note: you may need
to restart the kernel to use updated packages.


[notice] A new release of pip is available: 23.3 -> 25.1.1
[notice] To update, run: python.exe -m pip install --upgrade pip


Requirement already satisfied: numpy!=1.24.0,>=1.20 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from seaborn) (1.26.4)
Requirement already satisfied: pandas>=1.2 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from seaborn) (2.2.3)
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from seaborn) (3.10.0)
Requirement already satisfied: contourpy>=1.0.1 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (1.3.1)
Requirement already satisfied: cycler>=0.10 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (4.55.3)
Requirement already satisfied: kiwisolver>=1.3.1 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (1.4.8)
Requirement already satisfied: packaging>=20.0 in
c:\users\gulnar\appdata\roaming\python\python311\site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (23.0)
Requirement already satisfied: pillow>=8 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (3.2.0)
Requirement already satisfied: python-dateutil>=2.7 in
c:\users\gulnar\appdata\roaming\python\python311\site-packages (from
matplotlib!=3.6.1,>=3.4->seaborn) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from pandas>=1.2->seaborn)
(2025.2)
Requirement already satisfied: tzdata>=2022.7 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from pandas>=1.2->seaborn)
(2025.2)
Requirement already satisfied: six>=1.5 in
c:\users\gulnar\appdata\roaming\python\python311\site-packages (from python-
dateutil>=2.7->matplotlib!=3.6.1,>=3.4->seaborn) (1.16.0)
Requirement already satisfied: matplotlib in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (3.10.0)Note: you may need
to restart the kernel to use updated packages.


[notice] A new release of pip is available: 23.3 -> 25.1.1
[notice] To update, run: python.exe -m pip install --upgrade pip


Requirement already satisfied: contourpy>=1.0.1 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from matplotlib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from matplotlib) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from matplotlib) (4.55.3)
Requirement already satisfied: kiwisolver>=1.3.1 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from matplotlib) (1.4.8)
Requirement already satisfied: numpy>=1.23 in
```

```
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from matplotlib) (1.26.4)
Requirement already satisfied: packaging>=20.0 in
c:\users\gulnar\appdata\roaming\python\python311\site-packages (from matplotlib)
(23.0)
Requirement already satisfied: pillow>=8 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from matplotlib) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from matplotlib) (3.2.0)
Requirement already satisfied: python-dateutil>=2.7 in
c:\users\gulnar\appdata\roaming\python\python311\site-packages (from matplotlib)
(2.8.2)
Requirement already satisfied: six>=1.5 in
c:\users\gulnar\appdata\roaming\python\python311\site-packages (from python-
dateutil>=2.7->matplotlib) (1.16.0)
Requirement already satisfied: scikit-learn in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (1.6.1)
Requirement already satisfied: numpy>=1.19.5 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from scikit-learn)
(1.26.4)
Requirement already satisfied: scipy>=1.6.0 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from scikit-learn)
(1.15.3)
Requirement already satisfied: joblib>=1.2.0 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from scikit-learn) (1.5.0)
Requirement already satisfied: threadpoolctl>=3.1.0 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from scikit-learn) (3.6.0)
Note: you may need to restart the kernel to use updated packages.


[notice] A new release of pip is available: 23.3 -> 25.1.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```

```
Requirement already satisfied: torch-geometric in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (2.6.1)Note: you may need
to restart the kernel to use updated packages.


[notice] A new release of pip is available: 23.3 -> 25.1.1
[notice] To update, run: python.exe -m pip install --upgrade pip


Requirement already satisfied: aiohttp in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from torch-geometric)
(3.11.18)
Requirement already satisfied: fsspec in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from torch-geometric)
(2025.5.0)
Requirement already satisfied: jinja2 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from torch-geometric)
(3.1.6)
```

```
Requirement already satisfied: numpy in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from torch-geometric)
(1.26.4)
Requirement already satisfied: psutil>=5.8.0 in
c:\users\gulnar\appdata\roaming\python\python311\site-packages (from torch-
geometric) (5.9.4)
Requirement already satisfied: pyparsing in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from torch-geometric)
(3.2.0)
Requirement already satisfied: requests in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from torch-geometric)
(2.32.3)
Requirement already satisfied: tqdm in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from torch-geometric)
(4.67.1)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from aiohttp->torch-
geometric) (2.6.1)
Requirement already satisfied: aiosignal>=1.1.2 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from aiohttp->torch-
geometric) (1.3.2)
Requirement already satisfied: attrs>=17.3.0 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from aiohttp->torch-
geometric) (24.3.0)
Requirement already satisfied: frozenlist>=1.1.1 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from aiohttp->torch-
geometric) (1.6.0)
Requirement already satisfied: multidict<7.0,>=4.5 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from aiohttp->torch-
geometric) (6.4.4)
Requirement already satisfied: propcache>=0.2.0 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from aiohttp->torch-
geometric) (0.3.1)
Requirement already satisfied: yarl<2.0,>=1.17.0 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from aiohttp->torch-
geometric) (1.20.0)
Requirement already satisfied: MarkupSafe>=2.0 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from jinja2->torch-
geometric) (3.0.2)
Requirement already satisfied: charset-normalizer<4,>=2 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from requests->torch-
geometric) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from requests->torch-
geometric) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from requests->torch-
geometric) (2.3.0)
```

```
Requirement already satisfied: certifi>=2017.4.17 in
c:\users\gulnar\anaconda3\envs\fda\lib\site-packages (from requests->torch-
geometric) (2025.1.31)
Requirement already satisfied: colorama in
c:\users\gulnar\appdata\roaming\python\python311\site-packages (from
tqdm->torch-geometric) (0.4.6)
```

[152]:
```
import torch
import numpy as np
import pandas as pd
import seaborn as sns
import torch.nn as nn
import torch_geometric
from numpy import random
import torch.optim as optim
import matplotlib.pyplot as plt
from sklearn.dummy import DummyClassifier
from sklearn.linear_model import LogisticRegression
from sklearn import datasets, linear_model, metrics
from sklearn.model_selection import train_test_split
```

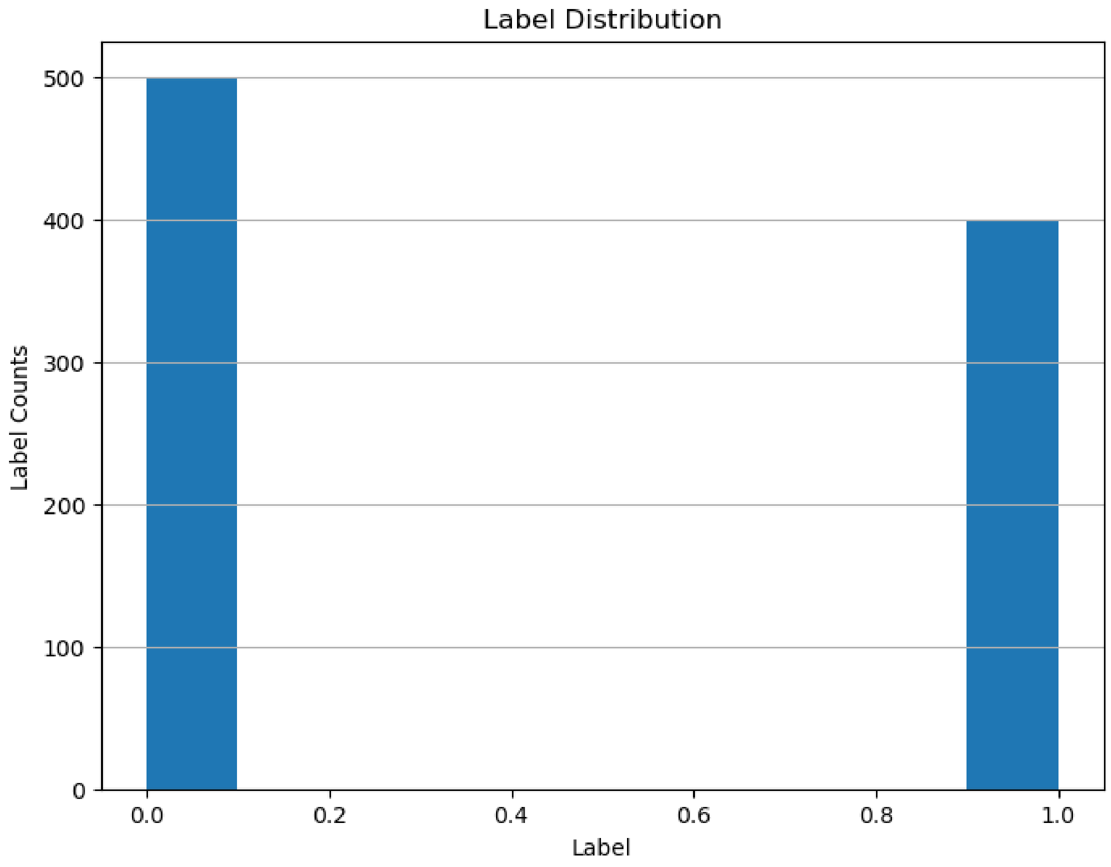
1 Part I

1.0.1 1. Understanding the Data

[153]:
```
# Load the files features.npy and labels.npy
features = np.load("data/features.npy")
labels = np.load("data/labels.npy")

num_samples, num_features = features.shape
labels_shape = labels.shape
print("(a) How many features and samples are present in the dataset?")
print(f"    Number of samples: {num_samples}, number of features:␣
  ↪{num_features}")
print(f"    Shape of labels: {labels_shape}")

unique_labels, label_counts = np.unique(labels, return_counts=True)
print(f"\nUnique labels: {unique_labels}, label counts: {label_counts}")

print("\n(b) Is the task a regression, binary classification, or multiclass␣
  ↪classification?")
print("    binary classification, because there are two unique labels (0 and␣
  ↪1)")

print("\n(c) Plot a histogram of the label distribution. What does this␣
  ↪distribution tell you?")
```

```python
plt.figure(figsize=(8, 6))
plt.hist(labels)
plt.title("Label Distribution")
plt.xlabel("Label")
plt.ylabel("Label Counts")
plt.grid(axis='y')
plt.show()
```

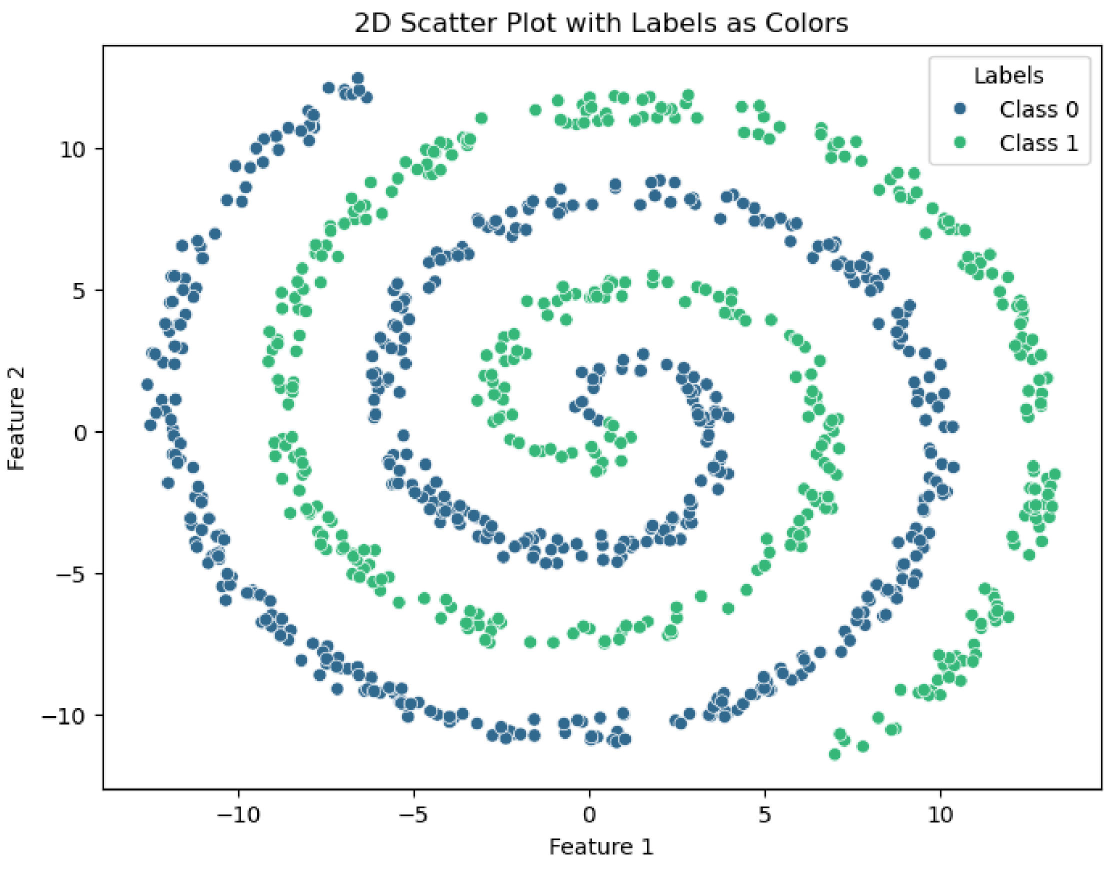
```
(a) How many features and samples are present in the dataset?
    Number of samples: 900, number of features: 2
    Shape of labels: (900,)

Unique labels: [0. 1.], label counts: [500 400]

(b) Is the task a regression, binary classification, or multiclass
classification?
    binary classification, because there are two unique labels (0 and 1)

(c) Plot a histogram of the label distribution. What does this distribution tell
you?
```



It looks like the label consist of zeroes and ones. From the histogram above, we see that there is 500 zeroes and 400 ones, which means zeroes come in the labels more frequent than ones.

1.0.2 2. Data visualisation

[154]:
```python
# Create a DataFrame for the features and labels
df = pd.DataFrame(features, columns=['x', 'y'])

# Map labels to class names
df['label'] = pd.Series(labels).map({0: 'Class 0', 1: 'Class 1'})

# Create a 2D scatter plot
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='x', y='y', hue='label', palette='viridis',␣
 ↪legend='full')
plt.title("2D Scatter Plot with Labels as Colors")
plt.xlabel("Feature 1")
plt.ylabel("Feature 2")
plt.legend(title='Labels')
plt.show()
```

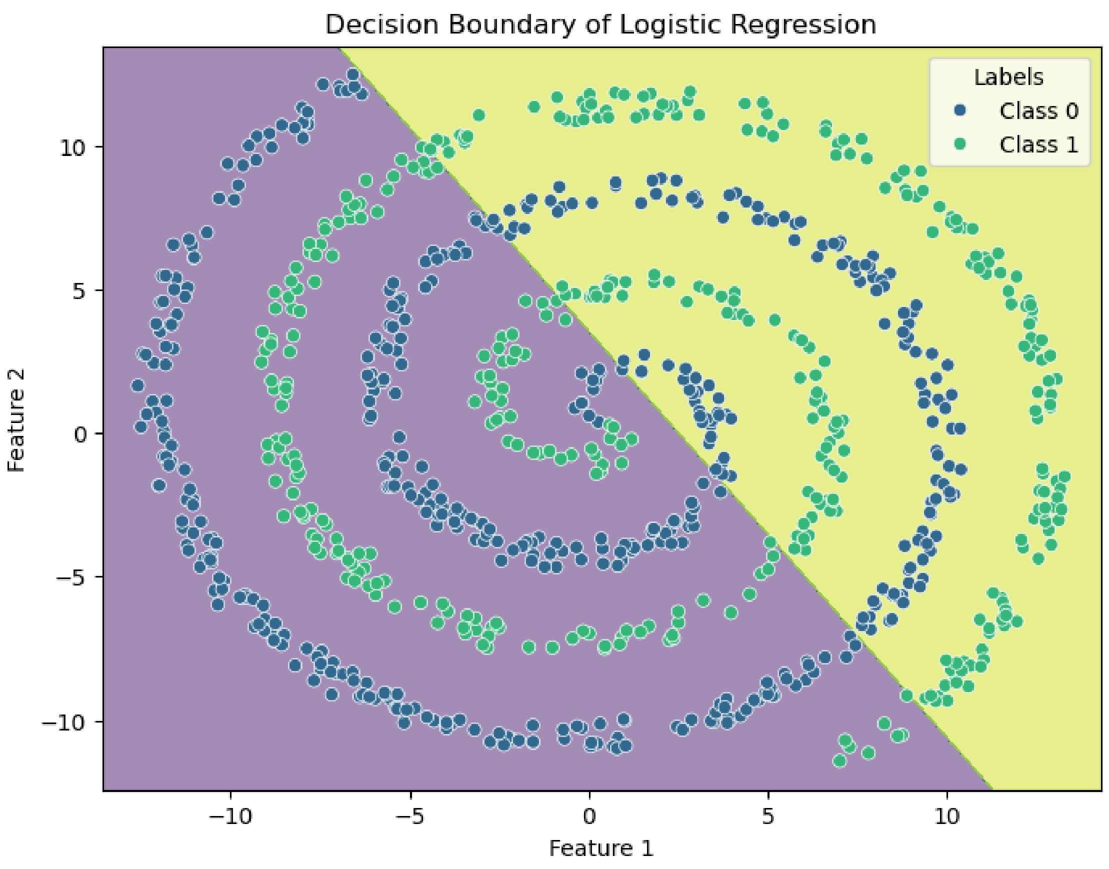


1.0.3 3. Fitting the first classifier

[155]:
```python
# Split the data
X_train, X_test, y_train, y_test = train_test_split(features, labels,␣
 ↪test_size=0.3, random_state=42)

# Train logistic regression model
model = LogisticRegression()
model.fit(X_train, y_train)

# Predict on test set
y_pred = model.predict(X_test)

acc = metrics.accuracy_score(y_test, y_pred)
print(f"Logistic Regression model accuracy: {acc * 100:.2f}%")
```

```
Logistic Regression model accuracy: 64.44%
```

[156]:
```python
# Plot decision boundary
x_min, x_max = features[:, 0].min() - 1, features[:, 0].max() + 1
y_min, y_max = features[:, 1].min() - 1, features[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.01),
                     np.arange(y_min, y_max, 0.01))

# Predict on the mesh grid
Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

# Plotting
plt.figure(figsize=(8, 6))
plt.contourf(xx, yy, Z, alpha=0.5, cmap='viridis')
sns.scatterplot(data=df, x='x', y='y', hue='label', palette='viridis',␣
 ↪legend='full')
plt.title("Decision Boundary of Logistic Regression")
plt.xlabel("Feature 1")
plt.ylabel("Feature 2")
plt.legend(title='Labels')
plt.show()
```



1.0.4 4. Does the model perform better than chance?

(a) Shuffle the data one time Due to using `np.random.permutation` instead of `np.random.shuffle`, we make sure, that the original data stay unchanged. So, with `np.random.permutation` we can shuffle the data and store them as a new shuffled data.

Additionally we use `np.random.seed(42)` (42 is the most common random state) to make use the accuracy will reproduce always the same output every time we run the notebook.

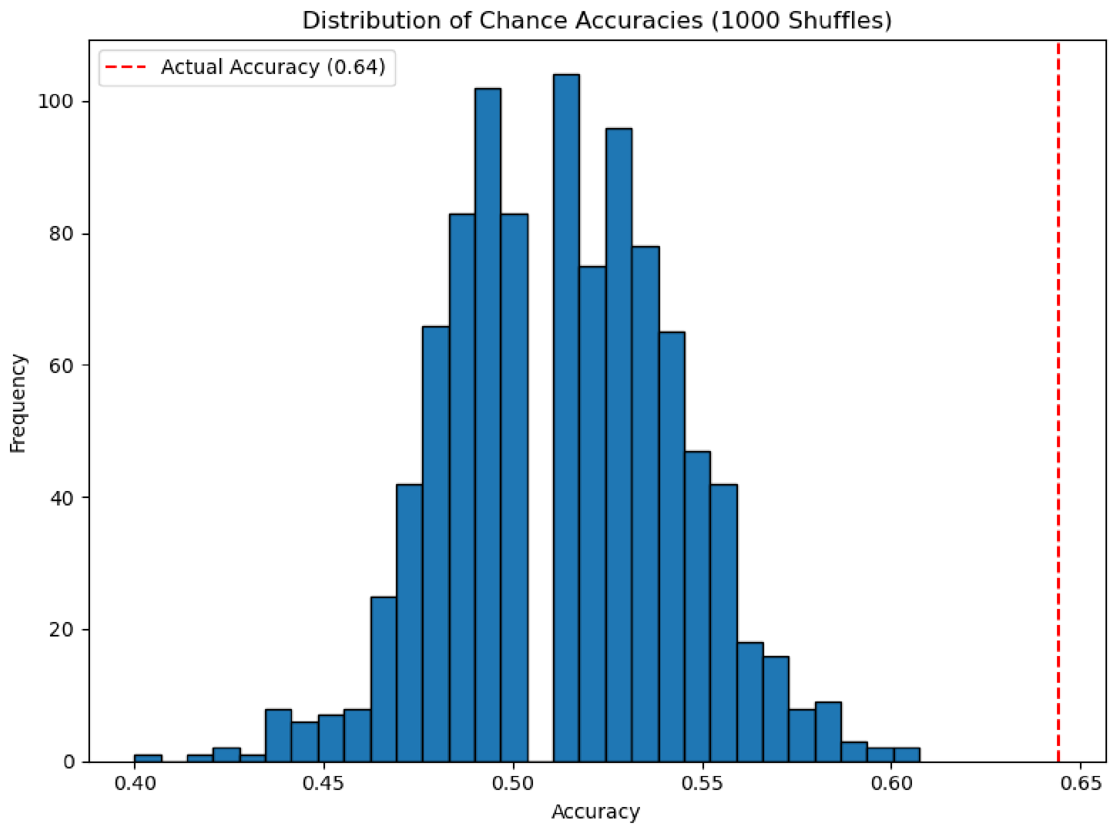
[157]:
```
random.seed(42)
shuffled_labels = random.permutation(y_test)
print(f"Accuracy after shuffling the data one time: {metrics.
  ↪accuracy_score(y_test, shuffled_labels) * 100:.2f}%")
```

```
Accuracy after shuffling the data one time: 51.85%
```

(b) Shuffle the data 1000 times

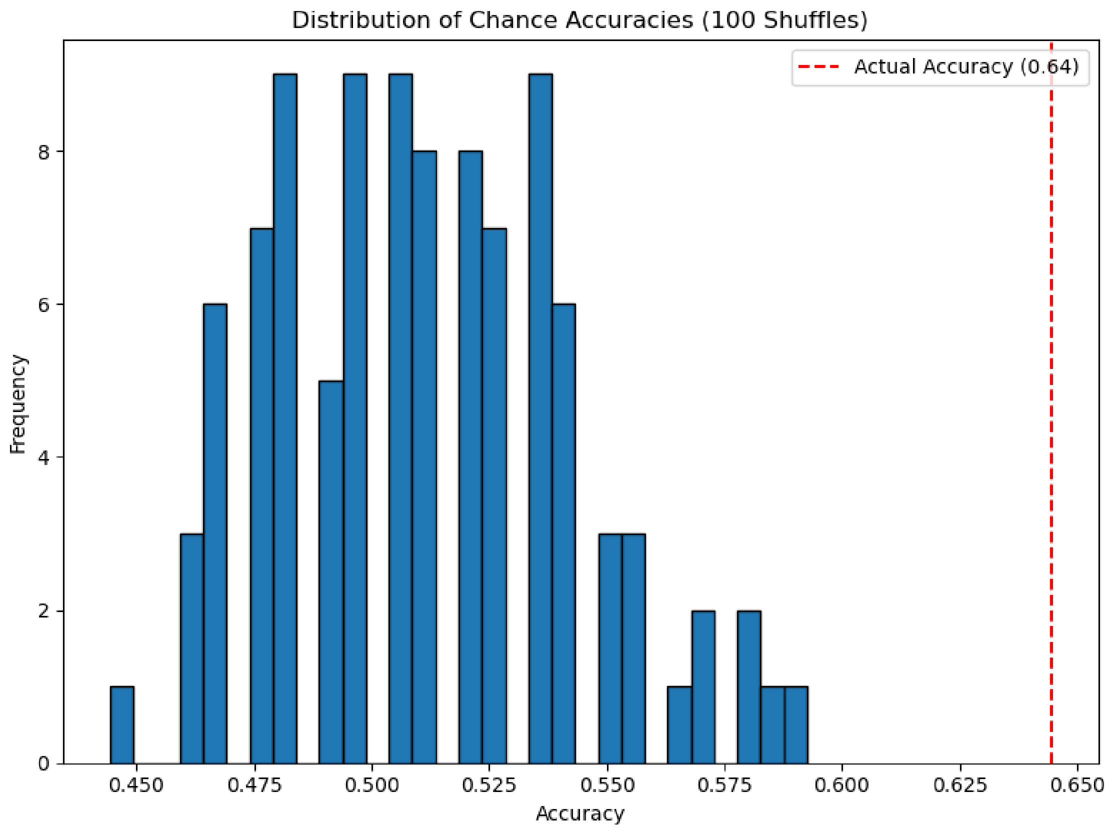
[158]:
```
shuffled_accuracies = [metrics.accuracy_score(y_test, random.
  ↪permutation(y_test)) for _ in range(1000)]
```

```
# Plot
plt.figure(figsize=(8, 6))
plt.hist(shuffled_accuracies, bins=30, edgecolor='black')
plt.axvline(acc, color='red', linestyle='--', label=f'Actual Accuracy ({acc:.
  ↪2f})')
plt.title("Distribution of Chance Accuracies (1000 Shuffles)")
plt.xlabel("Accuracy")
plt.ylabel("Frequency")
plt.legend()
plt.tight_layout()
plt.show()
```



Let’ compare the shuffled data with much less permutations (e.g. 100)

[159]:
```
shuffled_accuracies_fewer = [metrics.accuracy_score(y_test, random.
  ↪permutation(y_test)) for _ in range(100)]

# Plot
plt.figure(figsize=(8, 6))
plt.hist(shuffled_accuracies_fewer, bins=30, edgecolor='black')
plt.axvline(acc, color='red', linestyle='--', label=f'Actual Accuracy ({acc:.
  ↪2f})')
plt.title("Distribution of Chance Accuracies (100 Shuffles)")
plt.xlabel("Accuracy")
plt.ylabel("Frequency")
plt.legend()
plt.tight_layout()
plt.show()
```



(e) Estimate the accuracy of a simple baseline classifier that always predicts the most frequent class

[160]:
```
# Use dummy classifier to identify the most frequent class
most_frequent = DummyClassifier(strategy='most_frequent')
most_frequent.fit(X_train, y_train)
most_frequent_pred = most_frequent.predict(X_test)
print(f"The most frequent class accuracy: {metrics.accuracy_score(y_test,␣
  ↪most_frequent_pred) * 100:.2f}%")
```

```
The most frequent class accuracy: 58.15%
```

1.0.5 5. Neural nets

(a) create a neural network architecture

[ ]:
```
# Device agnostic code
device = "cuda" if torch.cuda.is_available() else "cpu"
print(device)
```

[ ]:
```
'cpu'
```

[252]:
```
#prepare the data
X_train_tensor = torch.from_numpy(X_train).float()
y_train_tensor = torch.from_numpy(y_train).float()
X_test_tensor = torch.from_numpy(X_test).float()
y_test_tensor = torch.from_numpy(y_test).float()

# Send tensors to the appropriate device
X_train_tensor = X_train_tensor.to(device)
y_train_tensor = y_train_tensor.to(device)
X_test_tensor = X_test_tensor.to(device)
y_test_tensor = y_test_tensor.to(device)
```

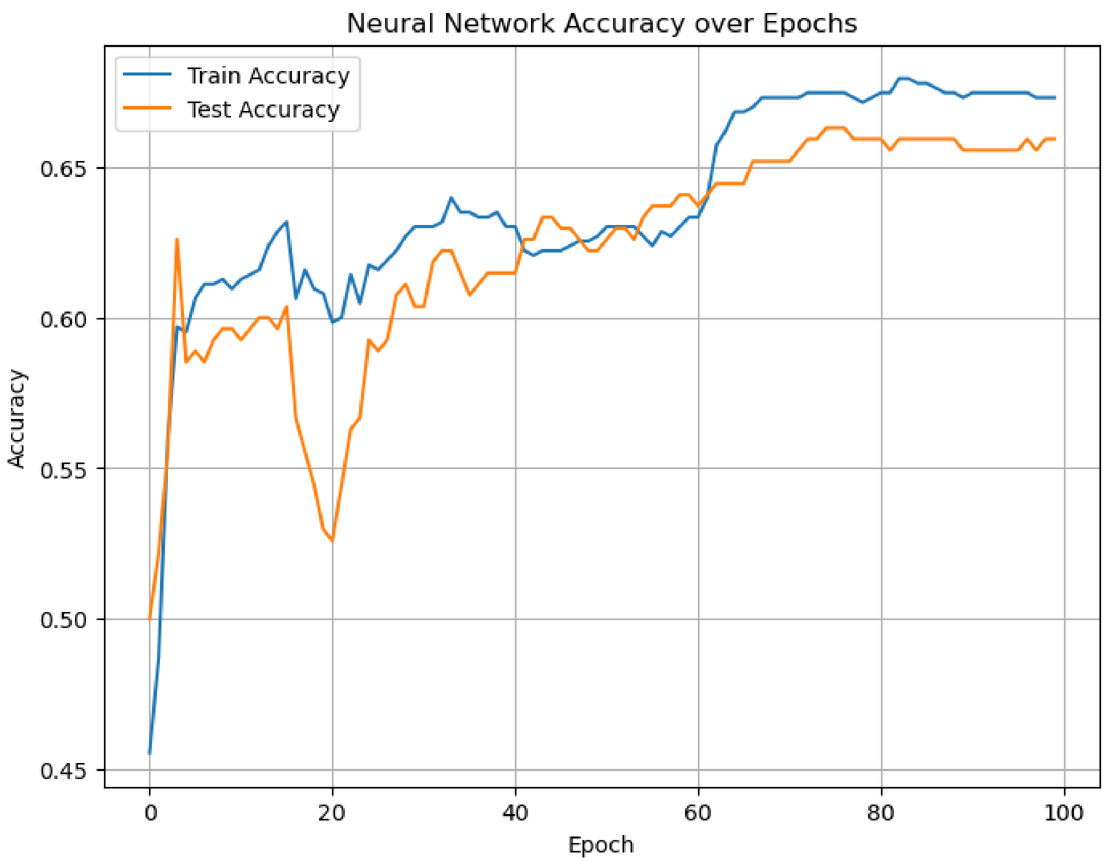
[302]:
```
# Define a simple neural network
class NeuralNetwork(nn.Module):
    def __init__(self):
        super(NeuralNetwork, self).__init__()
        self.fc1 = nn.Linear(in_features=2, out_features=20)
        self.fc2 = nn.Linear(in_features=20, out_features=1)
        self.relu = nn.ReLU()

    def forward(self, x):
        output = self.fc1(x)
        output = self.relu(output)
        output = self.fc2(output)
        return output

# Instantiate model
model = NeuralNetwork().to(device)
print(model)
```

```
NeuralNetwork(
  (fc1): Linear(in_features=2, out_features=20, bias=True)
  (fc2): Linear(in_features=20, out_features=1, bias=True)
  (relu): ReLU()
)
```

[303]:
```
# define loss function and optimizer
loss_fn = nn.BCEWithLogitsLoss()
optimizer = torch.optim.Adam(model.parameters(), lr=0.01)
```

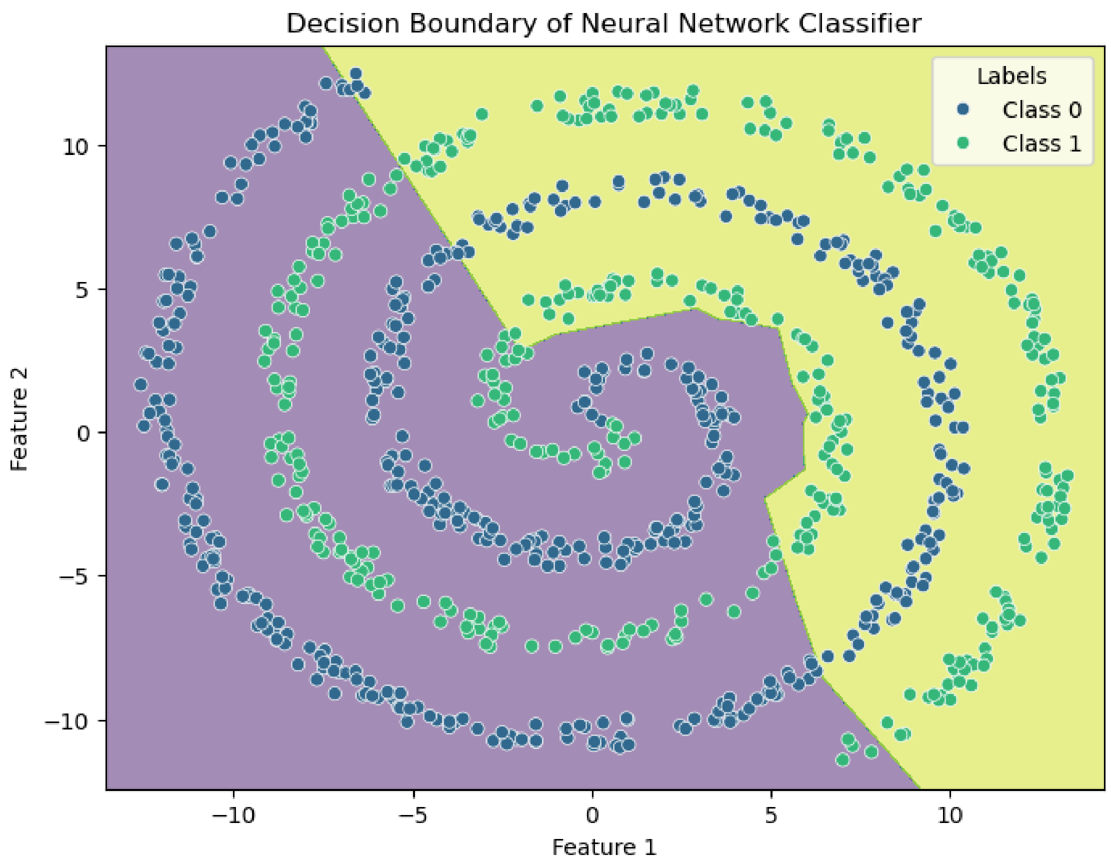
```
# Training loop
train_acc = []
test_acc = []

epochs = 100 # number of epochs to run
for epoch in range(epochs):
    # Training
    model.train()

    # Forward pass
    y_logits = model(X_train_tensor).squeeze()

    # Calculate loss
    loss = loss_fn(y_logits, y_train_tensor)

    # Optimizer zero grad
    optimizer.zero_grad()

    # Loss backwards
    loss.backward()

    # Optimizer step
    optimizer.step()

    # Evaluation
    model.eval()
    with torch.inference_mode():
        # Forward pass train
        train_logits = model(X_train_tensor).squeeze()
        train_pred = torch.round(torch.sigmoid(train_logits))

        # Forward pass tests
        test_logits = model(X_test_tensor).squeeze()
        test_pred = torch.round(torch.sigmoid(test_logits))


        # Calculate accuracies and append to lists
        train_acc.append(metrics.accuracy_score(y_train, train_pred))
        test_acc.append(metrics.accuracy_score(y_test, test_pred))
```

[304]:
```
# Plot accuracies
plt.figure(figsize=(8, 6))
plt.plot(range(epochs), train_acc, label='Train Accuracy')
plt.plot(range(epochs), test_acc, label='Test Accuracy')
plt.title("Neural Network Accuracy over Epochs")
plt.xlabel("Epoch")
plt.ylabel("Accuracy")
plt.legend()
plt.grid(True)
plt.show()
```



[316]:
```
# Predict on the mesh grid
grid_tensor = torch.from_numpy(np.column_stack((xx.ravel(), yy.ravel()))).
 ↪float()

# Predict with trained neural network model
model.eval()
with torch.inference_mode():
   logits = model(grid_tensor).squeeze()
   pred = torch.round(torch.sigmoid(logits))

   # Reshape the predictions to match the grid shape
   Z_pred = pred.reshape(xx.shape).detach().numpy()

# Plotting
```

```
plt.figure(figsize=(8, 6))
plt.contourf(xx, yy, Z_pred, alpha=0.5, cmap='viridis')
sns.scatterplot(data=df, x='x', y='y', hue='label', palette='viridis',␣
 ↪legend='full')
plt.title("Decision Boundary of Neural Network Classifier")
plt.xlabel("Feature 1")
plt.ylabel("Feature 2")
plt.legend(title='Labels')
plt.show()
```



2 Part II

2.0.1 1. Download the MUTAG dataset