Ćwiczenie: Przestrzeń cech – od danych surowych do reprezentacji geometrycznej

Celem ćwiczenia jest zrozumienie, że dane można traktować jako punkty w przestrzeni cech.
W trakcie zadania studenci nauczą się:
- generować dane syntetyczne,
- wizualizować dane w przestrzeni 2D i 3D,
- przeskalowywać cechy,
- dodawać nowe cechy,
- interpretować granicę decyzyjną modelu.

Część 1 — Generowanie przestrzeni cech

Cel: Student nauczy się tworzyć syntetyczny zbiór danych z cechami numerycznymi.

Zadanie: Uzupełnij kod tak, by wygenerować dane 2D z dwiema klasami (`class_0` i `class_1`).

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification

# TODO: uzupełnij parametry make_classification
X, y = make_classification(
  n_samples=____,     # np. 200
  n_features=____,    # np. 2
  n_redundant=0,
  n_informative=2,
  n_clusters_per_class=1,
  random_state=42
)

plt.scatter(X[:, 0], X[:, 1], c=y, cmap='bwr', edgecolor='k')
plt.title("Przestrzeń cech - dane 2D")
plt.xlabel("Cecha 1")
plt.ylabel("Cecha 2")
plt.show()
```

Część 2 — Skalowanie i analiza cech

Cel: Zrozumienie wpływu skali na kształt przestrzeni.

```
from sklearn.preprocessing import StandardScaler

# TODO: dopasuj i przekształć dane
scaler = StandardScaler()
X_scaled = scaler.____(____)

plt.scatter(____, ____, c=y, cmap='bwr', edgecolor='k')
plt.title("Przestrzeń cech - po skalowaniu")
plt.xlabel("Cecha 1 (scaled)")
plt.ylabel("Cecha 2 (scaled)")
plt.show()
```

Część 3 — Dodanie nowej cechy

Cel: Pokazać, że nowe cechy mogą ujawniać nieliniową strukturę danych.

```
from mpl_toolkits.mplot3d import Axes3D

# TODO: dodaj nową cechę: kwadrat cechy 1
X_new = np.column_stack([X_scaled, ____])

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(____, ____,____, c=y, cmap='bwr', edgecolor='k')
ax.set_xlabel("Cecha 1 (scaled)")
ax.set_ylabel("Cecha 2 (scaled)")
ax.set_zlabel("Nowa cecha = (Cecha 1)^2")
ax.set_title("Przestrzeń cech 3D")
plt.show()
```

Część 4 — Model i granica decyzyjna

Cel: zobaczyć, że model „rysuje linię” w przestrzeni cech.

```
from sklearn.linear_model import LogisticRegression
from mlxtend.plotting import plot_decision_regions

# Trenuj model
model = LogisticRegression()
model.fit(X_scaled, y)

# TODO: narysuj granicę decyzyjną
plot_decision_regions(X_scaled, y, clf=____, legend=2)
```

```
plt.xlabel("Cecha 1 (scaled)")
plt.ylabel("Cecha 2 (scaled)")
plt.title("Granica decyzyjna modelu regresji logistycznej")
plt.show()
```

Część 5 — Interpretacja przestrzeni

Cel: Zrozumienie wpływu cech na separację klas.

Zadanie:
Odpowiedz w komentarzu tekstowym:
- Jak zmieni się granica decyzyjna, jeśli dodamy więcej cech nieliniowych (np. x_1^2, x_2^2)?
- Dlaczego skalowanie cech wpływa na kształt granicy w przestrzeni?