uczenie-nadzorowane-klasyfikacja

May 22, 2025

0.1 Uczenie nadzorowane - klasyfikacja

Roksana Jandura nr. 416314

0.1.1 Wczytanie zbioru danych

[1]:
```
import pandas as pd

columns = [
    'Class',
    'Alcohol', 'Malic_acid', 'Ash', 'Alcalinity_of_ash',
    'Magnesium', 'Total_phenols', 'Flavanoids', 'Nonflavanoid_phenols',
    'Proanthocyanins', 'Color_intensity', 'Hue',
    'OD280/OD315_of_diluted_wines', 'Proline'
]

wine_df = pd.read_csv('wine.data', header=None, names=columns)

print(wine_df.head())
```

```
   Class  Alcohol  Malic_acid   Ash  Alcalinity_of_ash  Magnesium  \
0      1    14.23        1.71  2.43               15.6        127
1      1    13.20        1.78  2.14               11.2        100
2      1    13.16        2.36  2.67               18.6        101
3      1    14.37        1.95  2.50               16.8        113
4      1    13.24        2.59  2.87               21.0        118

   Total_phenols  Flavanoids  Nonflavanoid_phenols  Proanthocyanins  \
0           2.80        3.06                  0.28             2.29
1           2.65        2.76                  0.26             1.28
2           2.80        3.24                  0.30             2.81
3           3.85        3.49                  0.24             2.18
4           2.80        2.69                  0.39             1.82

   Color_intensity   Hue  OD280/OD315_of_diluted_wines  Proline
0             5.64  1.04                          3.92     1065
1             4.38  1.05                          3.40     1050
2             5.68  1.03                          3.17     1185
```

```
3              7.80  0.86                          3.45     1480
4              4.32  1.04                          2.93      735
```

Zbiór danych zawiera cechy próbek wina pochodzących z trzech różnych odmian winogron (klasy 1, 2 i 3).

Każdy wiersz odpowiada jednej próbce wina, a kolumny opisują jej właściwości, m.in.:

Alcohol – zawartość alkoholu (%)

Malic_acid – zawartość kwasu jabłkowego

Ash – zawartość popiołu mineralnego

Alcalinity_of_ash – zasadowość popiołu

Magnesium – zawartość magnezu (mg/l)

Total_phenols – suma związków fenolowych

Flavanoids – zawartość flawonoidów

Nonflavanoid_phenols – zawartość fenoli niefalwonoidowych

Proanthocyanins – zawartość proantocyjanin

Color_intensity – intensywność barwy

Hue – odcień

OD280/OD315_of_diluted_wines – absorpcja światła UV (jakość białek)

Proline – zawartość prolina (aminokwas)

Kolumna Class określa przynależność próbki do jednej z trzech klas wina: 1, 2 lub 3.

0.1.2 Podział zbioru danych na zbiór treningowy i testowy

[2]:
```
from sklearn.model_selection import train_test_split

X = wine_df.drop('Class', axis=1)
y = wine_df['Class']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42, stratify=y)

print("X_train shape:", X_train.shape)
print("X_test shape:", X_test.shape)
print("y_train shape:", y_train.shape)
print("y_test shape:", y_test.shape)
```

```
X_train shape: (142, 13)
X_test shape: (36, 13)
y_train shape: (142,)
y_test shape: (36,)
```

0.1.3 Normalizacja danych.

Skomentuj po co jest ten krok i jak może on wpływać na działania algorytmów z kolejnego punktu.

[3]:
```
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier

scaler = StandardScaler() #obiekt skalera, który oblicza średnią i std na␣
 ↪podstawie X train
X_train_scaled = scaler.fit_transform(X_train) #ovblicza średnią i std  na␣
 ↪danych treningowych i odrazu je przekształca
X_test_scaled = scaler.transform(X_test) #przekształcenie danych testowych tą␣
 ↪samą skalą obliczoną na danych treningowych (bez ponownego uczenia)
```

Powyższy kod wykonuje **normalizację danych wejściowych** przy użyciu `StandardScaler`, co oznacza przekształcenie wszystkich cech tak, aby miały:

- średnią = 0
- odchylenie standardowe = 1

KNeighborsClassifier Ten algorytm klasyfikuje punkty na podstawie odległości (np. euklidesowej) między obserwacjami.
Jeśli cechy są w różnych skalach (np. Proline ~1000, a Hue ~1.0), to cechy o większych wartościach zdominują obliczenia, prowadząc do błędnych klasyfikacji. **Dlatego normalizacja danych została zastosowana tylko dla tego algorytmu.**

RandomForestClassifier Random Forest opiera się na drzewach decyzyjnych, które działają na zasadzie warunków (np. x > próg).
Nie korzystają z odległości, więc nie są wrażliwe na skalę danych.

0.1.4 Trening dla algorytmów KNeighborsClassifier oraz RandomForestClassifier (biblioteka scikit-learn)

[4]:
```
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train_scaled, y_train) # trenuje model KNN na danych treningowych

rf = RandomForestClassifier(random_state=42)
rf.fit(X_train, y_train) # trenuje model Random Forest na danych treningowych
```

[4]:
```
RandomForestClassifier(random_state=42)
```

0.1.5 Klasyfikacja

[6]:
```
y_pred_knn = knn.predict(X_test_scaled)

y_pred_rf = rf.predict(X_test)
```

0.1.6 Zapoznaj się z metrykami dostępnymi w: https://scikit-learn.org/stable/modules/model_evaluation.html#classification-metrics. Opisz o czym mówią i w jakim kontekście używamy: accuracy, precision, recall and F-measures, confusion matrix oraz napisz czym jest classification report.

Accuracy (dokładność) - miara ogólnej skuteczności klasyfikacji. Określa, jaki procent wszystkich przykładów został poprawnie sklasyfikowany.

Precision (precyzja) - ile z przewidzianych jako pozytywne przypadków rzeczywiście jest pozytywnych.

Recall (czułość) - ile z rzeczywiście pozytywnych przypadków zostało wykrytych przez model.

F-measure (F1-score) - średnia harmoniczna precision i recall.

Confusion matrix (macierz pomyłek) - tabela pokazująca liczbę trafnych i błędnych klasyfikacji dla każdej klasy.

Classification report - podsumowanie metryk (precision, recall, F1-score i support) dla każdej klasy.

0.1.7 W nawiązaniu do metryk omawianych na wykładzie i tych analizowanych w punkcie 6 - analiza predykcji poszczególnych modeli.

[8]:
```
from sklearn.metrics import classification_report

print("=== KNN - Classification Report ===")
print(classification_report(y_test, y_pred_knn))

print("=== Random Forest - Classification Report ===")
print(classification_report(y_test, y_pred_rf))
```

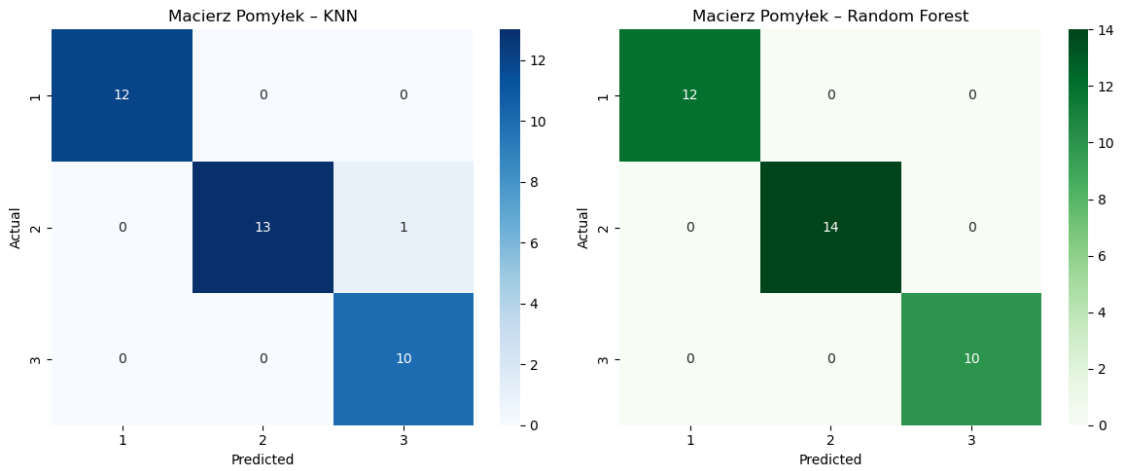
```
=== KNN - Classification Report ===
              precision    recall  f1-score   support

           1       1.00      1.00      1.00        12
           2       1.00      0.93      0.96        14
           3       0.91      1.00      0.95        10

    accuracy                           0.97        36
   macro avg       0.97      0.98      0.97        36
weighted avg       0.97      0.97      0.97        36

=== Random Forest - Classification Report ===
              precision    recall  f1-score   support

           1       1.00      1.00      1.00        12
           2       1.00      1.00      1.00        14
           3       1.00      1.00      1.00        10

    accuracy                           1.00        36
```

```
   macro avg       1.00      1.00      1.00        36
weighted avg       1.00      1.00      1.00        36
```

- Model **KNN** osiągnął bardzo dobrą skuteczność (accuracy = 0.97).
- Nieco niższe recall dla klasy 2 i precision dla klasy 3 — czyli jedna próbka klasy 2 została błędnie przypisana do klasy 3.
- Średnie (macro avg i weighted avg) również oscylują wokół 0.97–0.98.

- Model **Random Forest** sklasyfikował wszystkie próbki bezbłędnie (accuracy = 1.00).
- Każda klasa uzyskała perfekcyjne metryki (1.00 dla precision, recall, f1-score).
- Wszystkie próbki testowe zostały sklasyfikowane poprawnie

[9]:
```python
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix

labels = [1, 2, 3]

fig, ax = plt.subplots(1, 2, figsize=(12, 5))

cm_knn = confusion_matrix(y_test, y_pred_knn, labels=labels)
sns.heatmap(cm_knn, annot=True, fmt='d', cmap='Blues',
            xticklabels=labels, yticklabels=labels, ax=ax[0])
ax[0].set_title("Macierz Pomyłek - KNN")
ax[0].set_xlabel("Predicted")
ax[0].set_ylabel("Actual")

cm_rf = confusion_matrix(y_test, y_pred_rf, labels=labels)
sns.heatmap(cm_rf, annot=True, fmt='d', cmap='Greens',
            xticklabels=labels, yticklabels=labels, ax=ax[1])
ax[1].set_title("Macierz Pomyłek - Random Forest")
ax[1].set_xlabel("Predicted")
ax[1].set_ylabel("Actual")

plt.tight_layout()
plt.show()
```



Macierz pomyłek – KNN

- Model KNN popełnił **jedną pomyłkę**: próbka należąca do klasy 2 została błędnie zaklasyfikowana jako 3.
- Pozostałe klasy (1 i 3) zostały sklasyfikowane całkowicie poprawnie.
- Błąd może wynikać z faktu, że klasy 2 i 3 są do siebie podobne w przestrzeni cech, co wpłynęło na decyzję modelu KNN, który bazuje na odległościach.

Macierz pomyłek – Random Forest

- Model Random Forest zaklasyfikował wszystkie próbki poprawnie – macierz pomyłek zawiera tylko wartości na przekątnej.
- Oznacza to, że model ten dokładnie nauczył się reguł decyzyjnych dla wszystkich klas i żadna z obserwacji testowych nie została błędnie przypisana.

Macierze pomyłek potwierdzają wyniki z raportu klasyfikacji:
model **Random Forest** osiągnął idealną klasyfikację, podczas gdy **KNN popełnił tylko jeden błąd**, co również jest wynikiem bardzo dobrym.

0.1.8 Interpretacja wynikająca z analizy metryk - w szczególności z czego może wynikać różnica w działaniu tych dwóch modeli.

Model KNN osiągnął bardzo dobrą skuteczność klasyfikacji (accuracy 0.97), jednak popełnił jedną pomyłkę pomiędzy klasami 2 i 3, co wpłynęło na obniżenie recall i precision dla tych klas. Model Random Forest sklasyfikował wszystkie próbki bezbłędnie (accuracy 1.00), uzyskując perfekcyjne wyniki we wszystkich metrykach. Wynika to z wysokiej odporności tego algorytmu na skalę i nieliniowość danych. W kontekście rozpatrywanych metryk klasyfikacyjnych (accuracy, precision, recall, F1-score, confusion matrix), Random Forest wykazał się lepszą skutecznością i stabilnością klasyfikacji niż KNN.

[ ]: