Ćwiczenie laboratoryjne

Techniki interpretowalności modeli: implementacja metod wyjaśniania decyzji modeli uczenia maszynowego w medycynie

Katedra Informatyki i Automatyki

1 Cel ćwiczenia

Celem ćwiczenia jest praktyczne opanowanie metod interpretowalności (XAI) dla modeli stosowanych na danych medycznych:

- ważność i wpływ cech (Permutation Importance, SHAP, LIME),
- zależności globalne i lokalne (PDP/ICE, Partial/Individual Dependence),
- wyjaśnianie na poziomie próbki (SHAP values, LIME, Integrated Gradients),
- dobre praktyki i pułapki interpretacji w zastosowaniach klinicznych.

2 Podstawy teoretyczne (definicje i wzory)

Niech $f : \mathbb{R}^d \to \mathbb{R}$ będzie modelem predykcyjnym, $x \in \mathbb{R}^d$ wektorem cech, a $N = \{1, \ldots, d\}$ zbiorem indeksów cech.

Permutation Importance (PI)

Dla wybranej cechy j definiujemy spadek jakości po permutacji tej cechy:

$$\Delta_j \;=\; \mathbb{E}\big[L(y, f(X))\big] \;-\; \mathbb{E}\big[L\big(y, f(X_{\pi(j)})\big)\big],$$

gdzie $X_{\pi(j)}$ to macierz z permutowaną kolumną j, a L jest funkcją straty (np. log-loss, MSE).

Partial Dependence (PDP) i ICE

Globalny wpływ cechy j w punkcie z:

$$\mathrm{PD}_j(z) \;=\; \mathbb{E}_{X_{-j}}\big[f\big(z, X_{-j}\big)\big].$$

Krzywe ICE to ślady $f\big(z, x^{(i)}_{-j}\big)$ dla kolejnych obserwacji i (lokalne profile zależności).

Shapley values (SHAP)

Wkład cechy i do predykcji f w punkcie x:

$$\phi_i(f,x) \;=\; \sum_{S\subseteq N\setminus\{i\}} \frac{|S|!(|N|-|S|-1)!}{|N|!}\Big(f_{S\cup\{i\}}(x) - f_S(x)\Big),$$

gdzie $f_S(x)$ to predykcja modelu przy uwzględnieniu wyłącznie cech z S (pozostałe wypełnione wartością referencyjną). Własności: efektywność, symetria, brakowy wkład, addytywność.

LIME (Local Interpretable Model-agnostic Explanations)

LIME aproksymuje lokalnie f prostym modelem $g \in \mathcal{G}$ (np. regresją liniową) ważoną odległością π_x:

$$\hat{g} = \arg\min_{g\in\mathcal{G}} \; \mathcal{L}\big(f, g, \pi_x\big) + \Omega(g),$$

gdzie Ω karze złożoność g, a $\mathcal{L}$ mierzy niezgodność g z f w sąsiedztwie x.

Integrated Gradients (IG)

Dla modeli różniczkowalnych F i punktu odniesienia x':

$$\mathrm{IG}_i(x) \;=\; (x_i - x'_i)\int_0^1 \frac{\partial F\big(x' + \alpha(x-x')\big)}{\partial x_i}\, d\alpha.$$

Influence Functions (zarys)

Wpływ próbki uczącej z_{train} na stratę dla z_{test}:

$$\mathcal{I}_{\text{up,loss}}\big(z_{\text{test}}, z_{\text{train}}\big) \approx -\,\nabla_\theta L\big(z_{\text{test}}, \hat{\theta}\big)^\top H_{\hat{\theta}}^{-1} \nabla_\theta L\big(z_{\text{train}}, \hat{\theta}\big),$$

gdzie $H_{\hat{\theta}}$ to hesjan straty empirycznej w optimum $\hat{\theta}$.

Uwaga kliniczna. Przy korelacjach między cechami i przy brakach danych interpretacje mogą być mylące (np. SHAP rozdziela wkład między skorelowane cechy).

3 Kody w Pythonie (notebook-ready)

Poniższe fragmenty zakładają `pandas`, `scikit-learn`, `matplotlib`, opcjonalnie `shap`. Jeżeli plik `dane_pacjentow_demo.csv` jest dostępny, zostanie użyty; w przeciwnym razie dane zostaną wygenerowane.

Setup: dane, model, metryki

```
import os, warnings, numpy as np, pandas as pd, matplotlib.pyplot
   as plt
warnings.filterwarnings("ignore")
from sklearn.model_selection import train_test_split
```

```
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.metrics import (accuracy_score, roc_auc_score,
   roc_curve,
                             mean_squared_error, r2_score,
                                confusion_matrix)
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.inspection import permutation_importance,
   PartialDependenceDisplay

# 1) Dane: wczytaj lub wygeneruj
PATH = "dane_pacjentow_demo.csv"
if os.path.exists(PATH):
    df = pd.read_csv(PATH)
else:
    rng = np.random.default_rng(7); N=600
    age = rng.integers(18, 90, size=N)
    sex = rng.choice(["F","M"], size=N)
    height = rng.normal(170, 10, size=N).clip(140, 200)
    weight = rng.normal(75, 15, size=N).clip(40, 160)
    bmi = weight / (height/100)**2
    sbp = (100 + 0.5*age + 1.2*bmi + (sex=="M")*5 + rng.normal
       (0,10,N)).round()
    dbp = (60 + 0.2*age + 0.6*bmi + (sex=="M")*3 + rng.normal(0,6,N
       )).round()
    glucose = rng.normal(105, 20, size=N).round().clip(60, 300)
    hypertension = ((sbp>=140)|(dbp>=90)).astype(int)
    df = pd.DataFrame(dict(age=age, sex=sex, bmi=bmi.round(1),
                           systolic_bp=sbp.astype(int),
                           diastolic_bp=dbp.astype(int),
                           glucose=glucose.astype(int),
                           hypertension=hypertension))

# 2) Klasyfikacja: przewidywanie nadci nienia
features = ["age","sex","bmi","glucose","systolic_bp","diastolic_bp
   "]
X = df[features].copy(); y = df["hypertension"].astype(int)

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.25, random_state=42, stratify=y)

num = X_train.select_dtypes(include=[np.number]).columns.tolist()
cat = X_train.select_dtypes(exclude=[np.number]).columns.tolist()

pre = ColumnTransformer([
    ("num", Pipeline([("imp", SimpleImputer(strategy="median")),
                      ("sc", StandardScaler())]), num),
```

```
    ("cat", Pipeline([("imp", SimpleImputer(strategy="most_frequent
       ")),
                      ("oh", OneHotEncoder(handle_unknown="ignore")
                         )]), cat)
])

clf = Pipeline([("pre", pre),
                ("model", LogisticRegression(max_iter=1000))]).fit(
                   X_train,y_train)

proba = clf.predict_proba(X_test)[:,1]
pred = (proba>=0.5).astype(int)
print("ACC:", accuracy_score(y_test,pred), "AUC:", roc_auc_score(
   y_test,proba))
```

Permutation Importance (globalna ważność cech)

```
pi = permutation_importance(clf, X_test, y_test, n_repeats=10,
   random_state=42)
imp = pd.DataFrame({"feature": X_test.columns, "PI": pi.
   importances_mean})
print(imp.sort_values("PI", ascending=False))

plt.figure(); plt.barh(imp["feature"], imp["PI"])
plt.title("Permutation Importance (test)"); plt.gca().invert_yaxis
   (); plt.show()
```

3.1 Interpretacja metody Permutation Importance

Permutation Importance (PI) jest metodą niezależną od typu modelu, która pozwala ocenić wpływ poszczególnych cech na jakość predykcji. Zasada działania jest intuicyjna: dla każdej cechy losowo permutuje się jej wartości w zbiorze testowym i obserwuje spadek jakości modelu. Im większy spadek, tym ważniejsza była dana cecha.

Kod przykładowy w Pythonie.

```
from sklearn.inspection import permutation_importance
import pandas as pd
import matplotlib.pyplot as plt

result = permutation_importance(
    clf, X_test, y_test,
    n_repeats=10, random_state=42
)

pi_df = pd.DataFrame({
    "feature": X_test.columns,
    "importance": result.importances_mean,
```

```
    "std": result.importances_std
})

print(pi_df.sort_values("importance", ascending=False))

plt.figure(figsize=(6,4))
plt.barh(pi_df["feature"], pi_df["importance"])
plt.xlabel("Permutation Importance (spadek AUC/ACC)")
plt.title("Permutation Importance - ważność cech")
plt.gca().invert_yaxis()
plt.show()
```

Przykładowy wynik.

Cecha	Ważność (PI)
systolic_bp	0.072
diastolic_bp	0.015
age	0.009
bmi	0.003
glucose	0.001
sex	0.000

Interpretacja wyników.

- **Wysoka wartość PI (np. 0.072 dla `systolic_bp`)** oznacza, że permutacja tej cechy silnie pogarsza jakość modelu. Model intensywnie polega na tej informacji przy przewidywaniu nadciśnienia. W kontekście medycznym jest to zgodne z wiedzą kliniczną: wartość ciśnienia skurczowego jest jednym z najważniejszych predyktorów nadciśnienia.
- **Średnie wartości PI (np. 0.015 dla `diastolic_bp`)** sugerują, że cecha wnosi użyteczną, lecz pomocniczą informację. Model korzysta z niej, ale nie w takim stopniu jak z cechy dominującej.
- **Niskie wartości PI (np. `bmi` = 0.003, `glucose` = 0.001)** wskazują, że cecha ma marginesowy wpływ na decyzję modelu. Permutacja jej wartości praktycznie nie pogarsza jakości predykcji.
- **Wartość PI równa zero (np. `sex`)** oznacza, że model ignoruje daną cechę — losowe mieszanie jej wartości nie wywołuje żadnego spadku jakości. W praktyce może to być zarówno efekt braku związku cechy z etykietą, jak i sygnał, że model korzysta z innych bardziej informacyjnych predyktorów.

Uwagi praktyczne.

- PI mierzy wpływ cechy *na model*, a nie jej znaczenie kliniczne. Jeżeli cecha jest ważna klinicznie, ale została źle zakodowana lub jest skorelowana z inną, model może jej nie wykorzystywać.

- Silna korelacja między cechami może prowadzić do zaniżenia PI. Przykładowo, jeśli `bmi` i masa ciała są silnie skorelowane, permutacja jednej z cech nie musi powodować dużego spadku jakości.
- Permutation Importance jest metodą globalną, dlatego powinna być łączona z wyjaśnieniami lokalnymi (np. LIME lub SHAP) w celu pełniejszej analizy predykcji klinicznych.

Podsumowanie. Permutation Importance pozwala określić, które cechy mają największy wpływ na decyzje modelu. W prezentowanym przykładzie najważniejszym predyktorem była wartość ciśnienia skurczowego, podczas gdy pozostałe cechy odgrywały rolę pomocniczą lub były ignorowane. Metoda ta jest szczególnie cenna w analizach medycznych, ponieważ pozwala zrozumieć, na jakich sygnałach opiera się model przed podjęciem decyzji.

PDP i ICE (zależności globalne/lokalne)

```
fig, ax = plt.subplots(figsize=(6,4))
PartialDependenceDisplay.from_estimator(
    clf, X_test, features=["age","bmi","glucose"], kind="average",
       ax=ax)
plt.tight_layout(); plt.show()

fig, ax = plt.subplots(figsize=(6,4))
PartialDependenceDisplay.from_estimator(
    clf, X_test, features=["age"], kind="individual", subsample
       =100, ax=ax)
plt.tight_layout(); plt.show()
```

3.2 Interpretacja wykresów PDP i ICE (zależności globalne i lokalne)

Metody **Partial Dependence Plots (PDP)** oraz **Individual Conditional Expectation (ICE)** służą do analizy wpływu pojedynczych cech na predykcje modelu. Pozwalają zrozumieć, jak model reaguje na zmiany wartości wybranych zmiennych przy jednoczesnym utrzymaniu pozostałych cech na ich wartościach obserwowanych w zbiorze danych.

Partial Dependence (PDP) — interpretacja globalna. PDP przedstawia *średni* wpływ cechy na wynik predykcji. Formalnie, dla cechy j i wartości z definiujemy:

$$PD_j(z) = \mathbb{E}_{X_{-j}}[f(z, X_{-j})],$$

co oznacza, że dla każdej obserwacji zastępujemy wartość cechy j wartością z, przepuszczamy próbkę przez model, a następnie uśredniamy wyniki.

PDP odpowiada zatem na pytanie:

> *Jak przeciętnie zmienia się predykcja modelu, gdy manipulujemy jedną cechą, a pozostałe cechy pozostają bez zmian?*

Individual Conditional Expectation (ICE) — interpretacja lokalna. ICE prezentuje krzywe zależności *dla poszczególnych obserwacji*. Dla pacjenta i:

$$ICE_j^{(i)}(z) = f(z, x_{-j}^{(i)}),$$

czyli obserwujemy, jak zmienia się predykcja modelu dla konkretnego pacjenta, gdy symulujemy różne wartości cechy j.

ICE pozwala wykryć indywidualne różnice w zachowaniu modelu, które mogą być ukryte w uśrednionym wykresie PDP.

Przykład kodu (Python).

```
from sklearn.inspection import PartialDependenceDisplay
import matplotlib.pyplot as plt

# PDP dla dwóch cech
fig, ax = plt.subplots(figsize=(6,4))
PartialDependenceDisplay.from_estimator(
    clf, X_test, features=["age", "bmi"], kind="average", ax=ax
)
plt.show()

# ICE dla cechy age
fig, ax = plt.subplots(figsize=(6,4))
PartialDependenceDisplay.from_estimator(
    clf, X_test, features=["age"], kind="individual", subsample=100, ax=ax
)
plt.show()
```

Interpretacja przykładowych wyników. Załóżmy, że PDP dla cechy `age` pokazuje rosnącą, niemal liniową zależność. Oznacza to, że:

- globalnie starszy wiek zwiększa szacowane prawdopodobieństwo nadciśnienia,
- model traktuje wiek jako predyktor ryzyka w populacji,
- wynik jest spójny z wiedzą kliniczną o wzroście oporu naczyniowego wraz z wiekiem.

W przypadku ICE dla `age` obserwujemy:

- większość krzywych rośnie w sposób monotoniczny,
- niektóre krzywe są płaskie — wiek nie wpływa istotnie na predykcję dla pacjentów z niskim ciśnieniem,
- dla pacjentów z wartościami granicznymi (np. SBP 130–140) krzywe mogą rosnąć gwałtownie po 50. roku życia, co wskazuje na interakcję *wiek* $\times$ *ciśnienie*.

Porównanie PDP i ICE.

- **PDP** przedstawia *trend populacyjny* i pokazuje średni kierunek związku między cechą a predykcją modelu.
- **ICE** ujawnia *różnice między pacjentami* i pozwala obserwować zróżnicowane reakcje modelu w zależności od profilu klinicznego.
- Jeżeli krzywe ICE są zróżnicowane, PDP może „ukrywać" heterogeniczność populacji poprzez uśrednianie.

Wnioski praktyczne. W analizie danych medycznych wykorzystanie PDP i ICE pozwala:

- identyfikować główne trendy populacyjne (np. wzrost ryzyka nadciśnienia z wiekiem),
- badać zachowanie modelu dla konkretnych pacjentów,
- wykrywać interakcje między zmiennymi (np. wiek a ciśnienie tętnicze),
- ocenić, czy model działa zgodnie z wiedzą kliniczną.

PDP i ICE stanowią zatem kluczowe narzędzia do zrozumienia zarówno globalnej, jak i lokalnej interpretacji decyzji modelu w medycynie.

SHAP (lokalne i globalne wyjaśnienia)

Opcjonalnie (jeżeli `shap` jest zainstalowany). Dla modeli liniowych/lasów można użyć odpowiednich wyjaśniaczy.

```
try:
    import shap
    explainer = shap.LinearExplainer(clf.named_steps["model"],
                                     shap.sample(clf.named_steps["
                                        pre"].transform(X_train),
                                        200),
                                     feature_names=list(X_train.
                                        columns))
    Xte = clf.named_steps["pre"].transform(X_test)
    sv = explainer.shap_values(Xte)
    shap.summary_plot(sv, features=X_test, feature_names=list(
       X_test.columns), show=False)
    plt.show()

    # Wyja nienie pojedynczej pr bki
    i = 0
    shap.force_plot(explainer.expected_value, sv[i,:], X_test.iloc[
       i,:], matplotlib=True)
    plt.show()
except Exception as e:
    print("SHAP niedost pny lub b   d:", e)
```

LIME (wersja tekstowa wyjaśnień lokalnych)

Uwaga: wymaga `lime`. Przykład tworzy lokalny model zastępczy.

```
try:
    from lime.lime_tabular import LimeTabularExplainer

    # 1. Bierzemy tylko cechy numeryczne do LIME
    num_cols = X_train.select_dtypes(include=["number"]).columns.
       tolist()
    X_train_num = X_train[num_cols].to_numpy()
    X_test_num  = X_test[num_cols].to_numpy()

    print("Cechy numeryczne u ywane przez LIME:", num_cols)

    explainer = LimeTabularExplainer(
        training_data=X_train_num,
        feature_names=num_cols,
        class_names=["noHT", "HT"],
        discretize_continuous=True,
        mode="classification"
    )

    # 2. Funkcja predykcji: dostaje NUMPY z cechami numerycznymi
    def make_predict_fn_for_patient(sex_value):
        """Zwraca funkcj  predict_fn, kt ra dokleja p  e
           pacjenta."""
        def predict_fn(x_num):
            # x_num: (n_samples, len(num_cols))
            X_df = pd.DataFrame(x_num, columns=num_cols)
            # dok adamy  sta    p  e   (ten sam sex dla wszystkich
                perturbacji)
            X_df["sex"] = sex_value
            # ewentualnie inne cechy kategoryczne podobnie
            return clf.predict_proba(X_df)
        return predict_fn

    # 3. Wybieramy pacjenta do wyja nienia
    i = 0  # np. pierwszy z testowych
    x0_num = X_test_num[i]
    sex0   = X_test.iloc[i]["sex"]   # p  e   tego pacjenta

    predict_fn = make_predict_fn_for_patient(sex0)

    exp = explainer.explain_instance(
        data_row=x0_num,
        predict_fn=predict_fn,
        num_features=6,
        top_labels=1
    )

    print("Wyja nienie LIME dla pacjenta", i, "| sex =", sex0)
```

```
    for feat, val in exp.as_list(label=1):
        print(feat, "=>", val)

except Exception as e:
    print("LIME niedost pny lub b   d :", e)
```

3.3 Interpretacja wyjaśnień lokalnych LIME

Poniżej przedstawiono przykładowy wynik działania LIME dla pojedynczego pacjenta:

```
Cechy numeryczne używane przez LIME: ['age', 'bmi', 'glucose', 'systolic_bp', 'diastolic_bp']
Wyjaśnienie LIME dla pacjenta 0 | sex = M

149.00 < systolic_bp <= 161.00 =>  0.07644
36.00 < age <= 56.00           => -0.01416
88.00 < diastolic_bp <= 94.00  =>  0.00317
21.30 < bmi <= 25.40           => -0.00165
93.00 < glucose <= 105.50      => -0.00099
```

Interpretacja wyjaśnienia LIME opiera się na analizie lokalnych wpływów cech na predykcję modelu. Każda linia ma postać:

$$\text{(przedział wartości cechy)} \quad \Rightarrow \quad \text{wkład do decyzji modelu}$$

Wartości dodatnie zwiększają prawdopodobieństwo predykcji klasy pozytywnej (np. nadciśnienia), natomiast wartości ujemne przesuwają predykcję w kierunku klasy negatywnej.

Dominujący wpływ cechy `systolic_bp`. Największy wkład w decyzję modelu ma przedział:

$$149 < \text{systolic\_bp} \leq 161 \quad \Rightarrow \quad +0.07644,$$

co oznacza, że podwyższone ciśnienie skurczowe silnie zwiększa prawdopodobieństwo zaklasyfikowania pacjenta jako chorego na nadciśnienie. Jest to zgodne z wiedzą kliniczną, ponieważ wartość skurczowego ciśnienia tętniczego jest kluczowym predyktorem nadciśnienia.

Cechy o działaniu ochronnym. Wiek w zakresie 36–56 lat, prawidłowe BMI oraz glikemia w normie mają niewielkie ujemne wkłady, np.:

$$36 < \text{age} \leq 56 \quad \Rightarrow \quad -0.01416,$$

co oznacza, że dla tego konkretnego pacjenta wpływ tych cech delikatnie zmniejsza prawdopodobieństwo nadciśnienia.

Cechy wspierające predykcję, lecz o małym wpływie. Przedział wartości rozkurczowego ciśnienia tętniczego:

$$88 < \text{diastolic\_bp} \leq 94 \quad \Rightarrow \quad +0.00317,$$

wskazuje na niewielkie, ale dodatnie przesunięcie predykcji w stronę nadciśnienia.

Podsumowanie kliniczne. Dla analizowanego pacjenta najważniejszym czynnikiem decydującym o predykcji modelu jest podwyższone ciśnienie skurczowe. Pozostałe parametry, mimo że mają wpływ kierunkowy zgodny z wiedzą medyczną, oddziałują jedynie marginalnie. Lokalna interpretacja LIME potwierdza więc, że decyzja modelu opiera się przede wszystkim na kluczowej z punktu widzenia klinicznego cesze.

Krzywe ROC oraz macierz pomyłek (kontekst kliniczny)

```
fpr, tpr, thr = roc_curve(y_test, proba)
plt.figure(); plt.plot(fpr,tpr); plt.plot([0,1],[0,1],'--')
plt.xlabel("FPR"); plt.ylabel("TPR"); plt.title("ROC"); plt.show()

cm = confusion_matrix(y_test, pred)
plt.figure(); plt.imshow(cm); plt.title("Confusion matrix"); plt.
   colorbar()
for i in range(cm.shape[0]):
  for j in range(cm.shape[1]):
    plt.text(j,i,str(cm[i,j]), ha="center", va="center")
plt.xticks([0,1],["neg","pos"]); plt.yticks([0,1],["neg","pos"])
plt.tight_layout(); plt.show()
```

Wskazówki. (1) Unikaj przecieku informacji przy interpretowaniu; (2) Przy silnie skorelowanych cechach preferuj metody respektujące współzależności (np. SHAP oparty o warunkowe oczekiwanie).

4 Zadanie

Na podstawie zbioru danych poprzednich zajęć oblicz wskaźniki: PI, PDP i ICE, SHAP, LIME. Interpretuj uzyskane rezultaty