SPRAWOZDANIE

Zajęcia: Uczenie maszynowe

Prowadzący: prof. dr hab. Vasyl Martsenyuk

Laboratorium Nr 2 Data 9.11.2024 Temat: Praktyczne zastosowanie drzew decyzyjnych i metod ensemble w analizie danych. Wariant 8	Imię Nazwisko Hubert Mentel Informatyka II stopień, niestacjonarne, 1 semestr, gr.1a

1. Zadanie:

Opracować przepływ pracy uczenia maszynowego zagadnienia klasyfikacji (pojedyncze drzewo decyzyjne) oraz klasyfikacji ensemble (używając wszystkie modele wymienione w tutorialu) na podstawie zbioru danych według wariantu zadania:

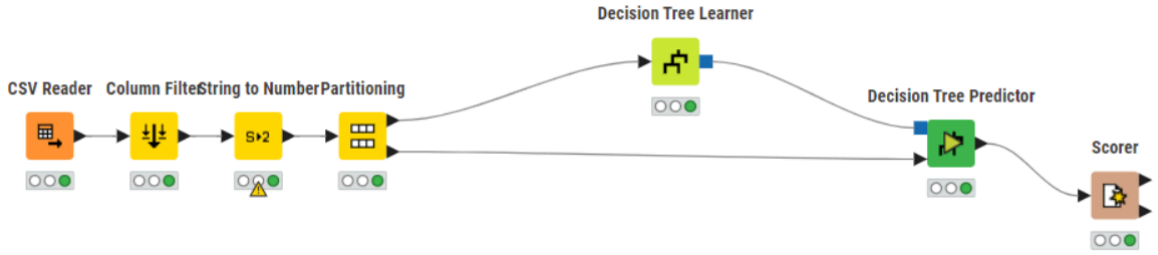
Prostate cancer
https://www.kaggle.com/ashrafalsinglawi/prostate-cancer-survival-data

Pliki dostępne są na GitHubie pod linkiem:

https://github.com/HubiPX/NOD/tree/master/UM/Zadanie%202

2. Opis programu opracowanego (kody źródłowe, zrzuty ekranu)

KNIME:



▶ 1: Confusion matrix ▶ 2: Accuracy statistics Flow Variables

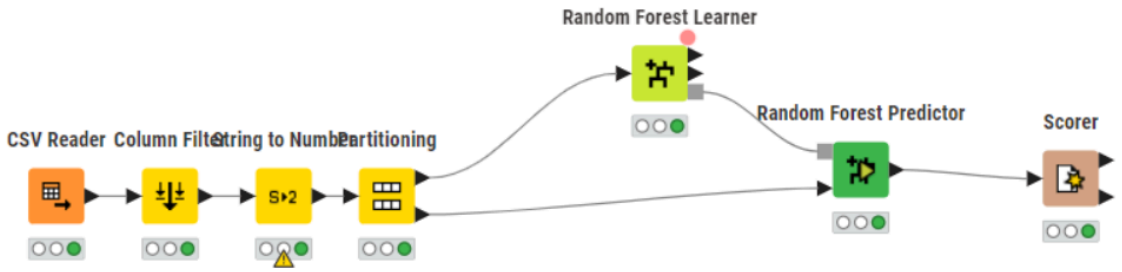
Rows: 7 | Columns: 7 Table Statistics

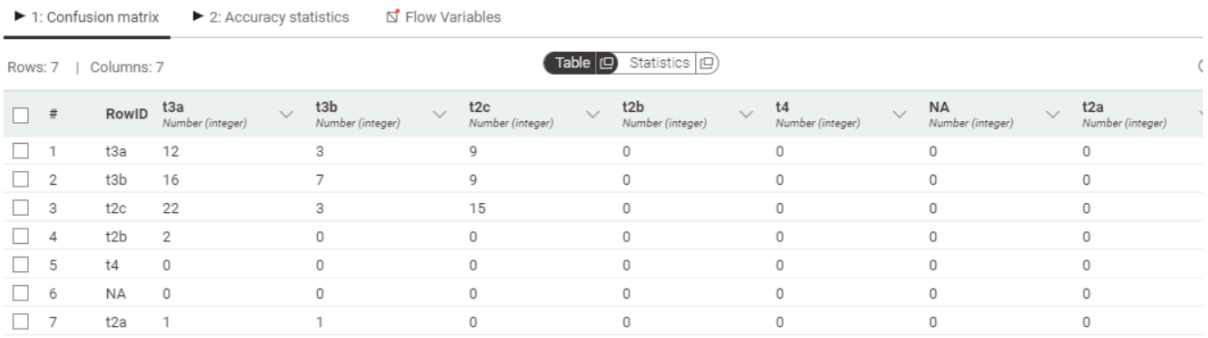
#	RowID	t3a Number (integer)	t3b Number (integer)	t2c Number (integer)	t2b Number (integer)	t4 Number (integer)	NA Number (integer)	t2a Number (integer)
1	t3a	35	52	46	0	0	0	0
2	t3b	21	56	29	0	0	0	0
3	t2c	40	33	54	0	0	0	0
4	t2b	0	8	2	0	0	0	0
5	t4	2	3	3	0	0	0	0
6	NA	2	1	2	0	0	0	0
7	t2a	3	3	5	0	0	0	0

▶ 1: Confusion matrix ▶ 2: Accuracy statistics Flow Variables

Rows: 8 | Columns: 11 Table Statistics

#	RowID	TruePosit... Number (inte...	FalsePosi... Number (inte...	TrueNega... Number (inte...	FalseNeg... Number (inte...	Recall Number (dou...	Precision Number (dou...	Sensitivity Number (dou...	Specificity Number (dou...	F-measure Number (dou...	Accur Number
1	t3a	35	68	199	98	0.263	0.34	0.263	0.745	0.297	?
2	t3b	56	100	194	50	0.528	0.359	0.528	0.66	0.427	?
3	t2c	54	87	186	73	0.425	0.383	0.425	0.681	0.403	?
4	t2b	0	0	390	10	0	?	0	1	?	?
5	t4	0	0	392	8	0	?	0	1	?	?
6	NA	0	0	395	5	0	?	0	1	?	?
7	t2a	0	0	389	11	0	?	0	1	?	?
8	Overall	?	?	?	?	?	?	?	?	?	0.362

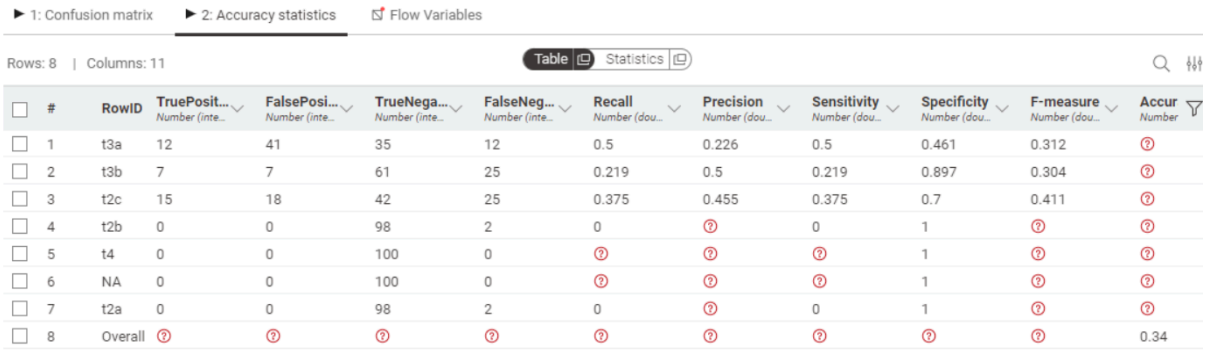




▶ 1: Confusion matrix ▶ 2: Accuracy statistics Flow Variables

Rows: 7 | Columns: 7 Table Statistics

#	RowID	t3a Number (integer)	t3b Number (integer)	t2c Number (integer)	t2b Number (integer)	t4 Number (integer)	NA Number (integer)	t2a Number (integer)
1	t3a	12	3	9	0	0	0	0
2	t3b	16	7	9	0	0	0	0
3	t2c	22	3	15	0	0	0	0
4	t2b	2	0	0	0	0	0	0
5	t4	0	0	0	0	0	0	0
6	NA	0	0	0	0	0	0	0
7	t2a	1	1	0	0	0	0	0



▶ 1: Confusion matrix ▶ 2: Accuracy statistics Flow Variables

Rows: 8 | Columns: 11 Table Statistics

#	RowID	TruePosit... Number (inte...	FalsePosi... Number (inte...	TrueNega... Number (inte...	FalseNeg... Number (inte...	Recall Number (dou...	Precision Number (dou...	Sensitivity Number (dou...	Specificity Number (dou...	F-measure Number (dou...	Accur Number
1	t3a	12	41	35	12	0.5	0.226	0.5	0.461	0.312	?
2	t3b	7	7	61	25	0.219	0.5	0.219	0.897	0.304	?
3	t2c	15	18	42	25	0.375	0.455	0.375	0.7	0.411	?
4	t2b	0	0	98	2	0	?	0	1	?	?
5	t4	0	0	100	0	?	?	?	1	?	?
6	NA	0	0	100	0	?	?	?	1	?	?
7	t2a	0	0	98	2	0	?	0	1	?	?
8	Overall	?	?	?	?	?	?	?	?	?	0.34

PYTHON:

```
import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy_score


# Wczytanie danych

df = pd.read_csv("CancerProstateSurvival.csv")


# Wyświetlenie pierwszych kilku wierszy danych

print(df.head())


df =
df[['times','patient.days_to_birth','patient.stage_event.tnm_categories.pathologic_categories.pathologic_t']]
```

```
df = df.dropna()


# Zakodowanie zmiennych kategorycznych w X

X =
df.drop(columns=["patient.stage_event.tnm_categories.pathologic_categories.
pathologic_t"])

X = pd.get_dummies(X, drop_first=True)  # One-hot encoding dla zmiennych
kategorycznych


# Target (y)

y =
df["patient.stage_event.tnm_categories.pathologic_categories.pathologic_t"]


# Podział na dane treningowe i testowe

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


# Trenowanie modelu

tree = DecisionTreeClassifier(max_depth=2, random_state=42)

tree.fit(X_train, y_train)


# Predykcja i ocena modelu

y_pred = tree.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
```

```
   times  patient.vital_status patient.gender patient.race patient.ethnicity  \
0    621                     0          male          NaN               NaN
1   1332                     0          male          NaN               NaN
2    995                     0          male          NaN               NaN
3    671                     0          male          NaN               NaN
4   1033                     0          male          NaN               NaN

   patient.days_to_birth patient.drugs.drug.therapy_types.therapy_type  \
0               -18658.0                                           NaN
1               -20958.0                                           NaN
2               -17365.0                                hormone therapy
3               -19065.0                                           NaN
4               -25904.0                                           NaN

   patient.stage_event.pathologic_stage  \
0                                   NaN
1                                   NaN
2                                   NaN
3                                   NaN
4                                   NaN

  patient.stage_event.tnm_categories.pathologic_categories.pathologic_t  \
0                                                 t2b
1                                                 t3a
2                                                  t4
3                                                 t2b
4                                                 t3b

   patient.stage_event.tnm_categories.pathologic_categories.pathologic_m
0                                                 NaN
1                                                 NaN
2                                                 NaN
3                                                 NaN
4                                                 NaN
Accuracy: 0.3917525773195876
```

```
from sklearn.ensemble import RandomForestClassifier


# Inicjalizacja modelu Random Forest
rf_model = RandomForestClassifier(n_estimators=100, max_depth=5, random_state=42)
rf_model.fit(X_train, y_train)


# Predykcja i ewaluacja modelu
y_pred_rf = rf_model.predict(X_test)
print("Random Forest Accuracy:", accuracy_score(y_test, y_pred_rf))
```

OUTPUT:

```
Random Forest Accuracy: 0.4329896907216495
```

```
from xgboost import XGBClassifier
y = pd.Categorical(y).codes
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.2, random_state=42)
# Inicjalizacja modelu
xgb = XGBClassifier(n_estimators=100 ,max_depth=1, learning_rate=0.1)
xgb.fit(X_train, y_train)
# Predykcja i ocena modelu
y_pred_xgb = xgb.predict(X_test)
print("XGBoost Accuracy:", accuracy_score(y_test, y_pred_xgb))
```

OUTPUT:

```
XGBoost Accuracy: 0.3917525773195876
```

3. Wnioski

KNIME pozwala na efektywne i intuicyjne tworzenie modeli klasyfikacyjnych za pomocą graficznego przepływu pracy. Modele takie jak drzewa decyzyjne, Random Forest i boosting można łatwo zaimplementować za pomocą odpowiednich węzłów KNIME, co umożliwia analizę danych bez konieczności pisania kodu. Jupyter Notebook daje większą kontrolę nad kodem i parametrami modeli, co pozwala na bardziej zaawansowane analizy.

Drzewa decyzyjne są prostymi, ale skutecznymi modelami uczenia maszynowego, szczególnie w analizie danych. Metody ensemble, takie jak bagging, Random Forest i boosting, poprawiają dokładność i stabilność modeli poprzez łączenie wielu słabszych klasyfikatorów. Random Forest wprowadza losowość w wyborze cech, co zwiększa odporność na przeuczenie, natomiast boosting (XGBoost) iteracyjnie wzmacnia błędnie sklasyfikowane przypadki.

Porównując metody, drzewo decyzyjne jest dobrą bazą dla klasyfikacji, ale jego skuteczność jest ograniczona. Random Forest zapewnia większą odporność na przeuczenie, jednak jego wyniki były porównywalne z pojedynczym drzewem. XGBoost, choć często przewyższa inne metody, w tym przypadku nie osiągnął wyższej dokładności, co może wynikać z charakterystyki zbioru danych lub parametrów modelu.