ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ  
«ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра АСУ

Отчет

о лабораторной работе №1

по дисциплине «Интеллектуальный анализ данных»

на тему: «Применение библиотеки Scikit-Learn для решения задач классификации и регрессии»

Выполнил:

студент группы ИСТ-19а

Деркач К. И.

Проверили:

Васяева Т. А.

Шуватова Е. А.

Донецк – 2022

**Цель работы:** изучение алгоритмов построения деревьев и правил решений; создание и исследование классификационных и регрессионных моделей с помощью деревьев и правил решений..

Порядок выполнения работы

1. Выбрать одну из баз данных (согласовать с преподавателем), список которых представлен на сайте http://archive.ics.uci.edu/ml/. Скачать данные с сайта.

2. Исходные данные должны храниться в файле. Программным способом считать эти данные и сформировать обучающую, проверочную и тестовую выборки. Обучающая выборка должна включать достаточное количество примеров для обучения. Тестовая выборка может быть вполовину меньше обучающей. Примеры из тестовой выборки не должны быть включены в обучающую. Для оценки обучения модели можно использовать проверочную (валидационную) выборку (при достаточном количестве исходных данных) или же использовать кросс-валидацию (если данных мало).

3. Разработать программу на языке Python с использованием библиотек Pandas, Scikit-Learn. Программа должна обеспечивать: построение дерева (правил) алгоритмом согласно варианту на обучающей выборке; для деревьев решений реализовать механизм упрощения деревьев; тестирование дерева (правил); вычисление точности и ошибки классификации; сохранение / считывание модели (деревья или правила); вывод на экран структуры дерева, можно в виде правил.

4. Согласно варианту заданий (таблица 1.4) построить классификатор таким образом, чтобы точность на обучающей выборке была не 100%, но более чем 90% (желательно около 97-98%). В процессе построения классификатора, записывайте в таблицу, какие у вас получались точности для различных моделей на обучающей и тестовой выборках в процессе подбора гиперпараметров.

**Индивидуальное задание:** Вариант 4, Random Forests

**Листинг программы**

from sklearn.model\_selection import train\_test\_split

from sklearn import preprocessing

from sklearn.ensemble import RandomForestClassifier

import pandas as pd

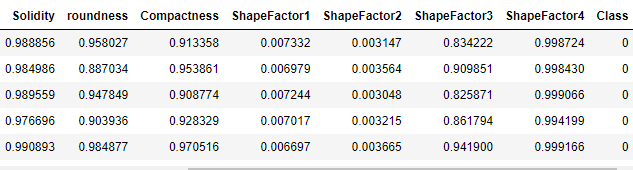
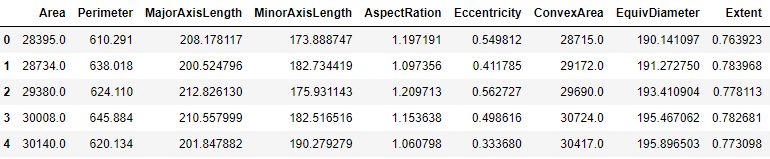
from scipy.io.arff import loadarff

raw\_data = loadarff('d:\Dry\_Bean\_Dataset.arff')

df = pd.DataFrame(raw\_data[0])

df.Class = pd.factorize(df.Class)[0]

df.head()



Первые 5 строк датасета

X = df.drop('Class', axis=1)

y = df['Class']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.33, random\_state = 1)

from itertools import product

n\_estimators = [10, 100]

max\_features = ['sqrt', 'log2']

max\_depth = [1,15]

min\_samples\_split = [2,20,50]

min\_samples\_leaf = [2,20,50]

bootstrap = [True, False]

t = pd.DataFrame(columns=[

"n\_estimators", "max\_features","max\_depth","min\_samples\_split",

"min\_samples\_leaf","bootstrap","score"

])

for n,f,d,s,l,b in product(n\_estimators, max\_features,max\_depth, min\_samples\_split, min\_samples\_leaf,bootstrap):

rf = RandomForestClassifier(

n\_estimators=n,

max\_features=f,

min\_samples\_leaf=l,

max\_depth=d,

bootstrap=b

)

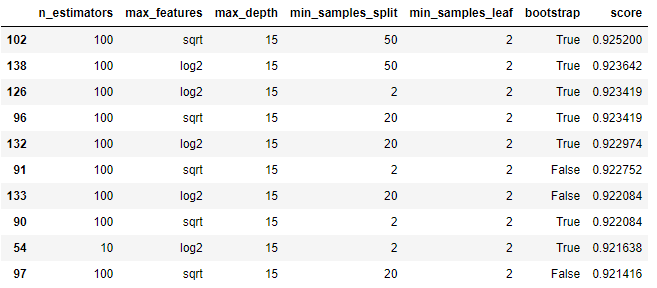
rf.fit(X\_train,y\_train)

sc = rf.score(X\_test,y\_test)

t.loc[t.shape[0]] = [n,f,d,s,l,b,sc]

t = t.sort\_values('score', ascending=False)

t.head(10)



Лучшие результаты подбора гиперпараметров

rf = RandomForestClassifier(

n\_estimators=t.iloc[0]['n\_estimators'],

max\_features=t.iloc[0]['max\_features'],

min\_samples\_leaf=t.iloc[0]['min\_samples\_leaf'],

max\_depth=t.iloc[0]['max\_depth'],

bootstrap=t.iloc[0]['bootstrap']

)

rf.fit(X\_train,y\_train)

print(f'Точность\nна обучающей выборке:{rf.score(X\_train,y\_train):.4f}\nна тестовой:{rf.score(X\_test,y\_test):.4f}')

**Точность**

**на обучающей выборке:0.9830**

**на тестовой:0.9239**

import pickle

with open('rf\_model', 'wb') as file:

pickle.dump(rf, file)

with open('rf\_model', 'rb') as file:

rf = pickle.load(file)

print(f'Точность\nна обучающей выборке:{rf.score(X\_train,y\_train):.4f}\nна тестовой:{rf.score(X\_test,y\_test):.4f}')

**Точность**

**на обучающей выборке:0.9830**

**на тестовой:0.9239**

feat\_importances = pd.Series(rf.feature\_importances\_, index=X.columns).sort\_values(ascending=False)

print(feat\_importances)

**Perimeter 0.100869**

**ShapeFactor3 0.098195**

**Compactness 0.096219**

**ShapeFactor1 0.095139**

**MajorAxisLength 0.079063**

**MinorAxisLength 0.076308**

**ConvexArea 0.075907**

**AspectRation 0.070952**

**Area 0.062326**

**Eccentricity 0.053664**

**roundness 0.052667**

**ShapeFactor2 0.044637**

**EquivDiameter 0.044169**

**ShapeFactor4 0.025771**

**Solidity 0.015511**

**Extent 0.008603**

**dtype: float64**