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

Шевченко М. В.

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 12, Random Forests

**Описание данных:**

Buying (цена) – vhigh, high, med, low (очень высокая, высокая, средняя, низкая);

Maint (цена обслуживания) – те же что и в buying.

Doors (количество дверей) – 2, 3, 4, 5-more (5 и больше)

Persons (Количество мест) – 2, 4, more (более)

Lug\_boot (размер багажника) – small, med, big (маленький, средний, большой)

Safety (безопасность) – low, med, high (слабая, средняя, сильная).

**Листинг программы**

1. **Чтение из файла в таблицу**

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

from sklearn import preprocessing

from sklearn.ensemble import RandomForestClassifier

import pandas as pd

columns=['buying','maint','doors','persons','lug\_boot','safety','class']

df = pd.read\_csv('D:\car.data',names=columns)

df.head(10)

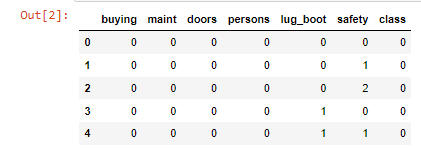


1. **Кодирование строковых данных**

for c in columns:

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

df.head()



1. **Разделение набора данных на входные и результирующие; Разделение данных на тренировочные и тестовые.**

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 = 0)

1. **Перебор параметров для Random Forest и определение лучших.**

n\_estimators = [3,5,10,50,100]

max\_depth = [1,7,10]

min\_samples\_leaf = [2,5,10]

results = pd.DataFrame(columns=[

"n\_estimators", "max\_depth","min\_samples\_leaf",

"score (train)","score (test)"

])

for ne in n\_estimators:

for md in max\_depth:

for ml in min\_samples\_leaf:

rf = RandomForestClassifier(

n\_estimators=ne,

min\_samples\_leaf=ml,

max\_depth=md

)

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

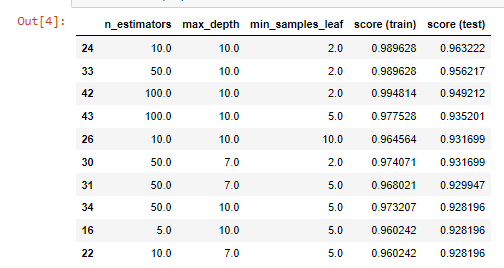
strain = rf.score(x\_train,y\_train)

stest = rf.score(x\_test,y\_test)

results.loc[results.shape[0]] = [ne,md,ml,strain,stest]

results = results.sort\_values('score (test)', ascending=False)

results.head(10)

****

1. **Создание с лучшими параметрами и тренировка, получение точности, вывод данных, запись в файл.**

rf = RandomForestClassifier(

n\_estimators=int(results.iloc[0]['n\_estimators']),

min\_samples\_leaf=int(results.iloc[0]['min\_samples\_leaf']),

max\_depth=int(results.iloc[0]['max\_depth'])

)

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

print(f'Точность = {rf.score(x\_test,y\_test):.4f}')

import pickle

with open('rfmodel', 'wb') as file:

pickle.dump(rf, file)

print('Сохранено в файл')

with open('rfmodel', 'rb') as file:

rf = pickle.load(file)

print('Загружено из файла')

print(f'Точность = {rf.score(x\_test,y\_test):.4f}\n\n')

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

print(feat\_importances)

