МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГАОУ ВО «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ ИНФОРМАЦИОННЫХ

ТЕХНОЛОГИЙ И ТЕЛЕКОММУНИКАЦИЙ

КАФЕДРА ПРИКЛАДНОЙ ИНФОРМАТИКИ

**Лабораторная работа №4**

по дисциплине

«Искусственный интеллект и машинное обучение»

**Выполнил:**

Сердюков Никита Анатольевич

Студент 2 курса группы \_ПИН-б-о-22-1

Направления подготовки

09.03.03 Прикладная информатика

очной формы обучения

Ставрополь, 2023 г.

Тема: ЛОГИЧЕСКИЕ МЕТОДЫ КЛАССИФИКАЦИИ.

Цель работы: изучение принципов построения информационных систем с использованием логических методов классификации.

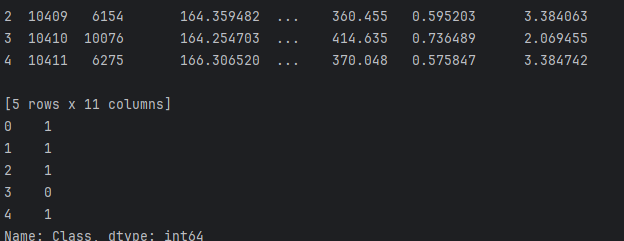
Выполнение работы:

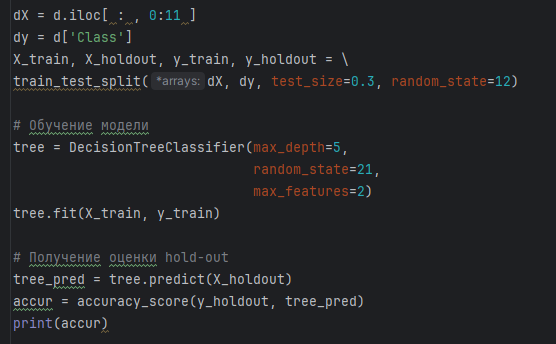
Датасет: Rice type classification

Описание датасета: Это набор данных, созданный для классификации риса.. Это модифицированный набор данных: Jasmine - 1, Gonen - 0.

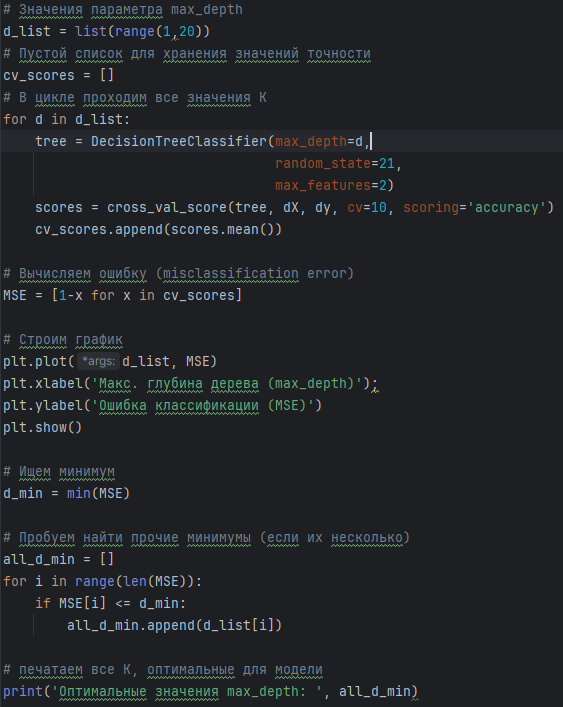
1. Откроем нашу базу данных

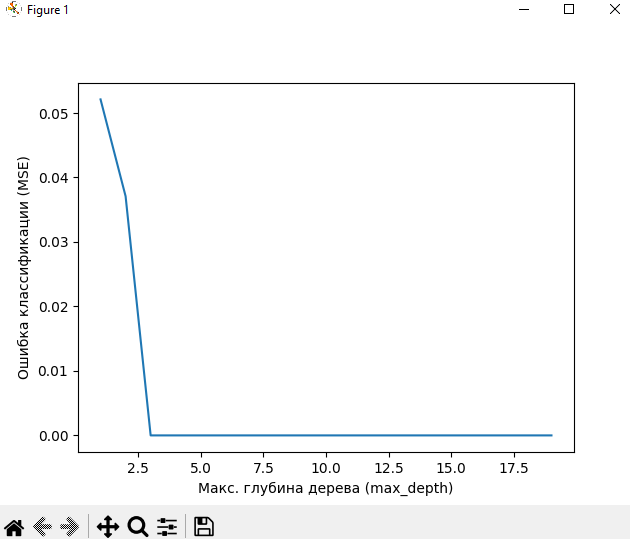




1. Построение логического классификатора с заданием max\_depth (максимальной глубины) и max\_features (максимального количества признаков) пользователем (установить любые)



1. Вычисление оценки cross validation (MSE) для различнх значений max\_depth (построить график зависимости);



Оптимальные значения max\_depth: [3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]

1. Вычисление оценки cross validation (MSE) для различнх значений max\_features (построить график зависимости);

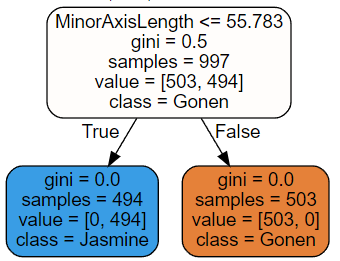


1. Вычислите оптимальные значения max\_depth и max\_features

Лучшее сочетание параметров: {'max\_depth': 1, 'max\_features': 3}

Лучшие баллы cross validation: 1.0

1. Выведите дерево в формате .png



Листинг:

4.1:

import numpy as np  
import pandas as pd  
  
from sklearn.tree import DecisionTreeClassifier  
from sklearn.model\_selection import train\_test\_split  
from sklearn.metrics import accuracy\_score  
  
import seaborn as sns  
from matplotlib import pyplot as plt  
  
data\_source = 'Dataset.txt'  
d = pd.read\_table(data\_source, delimiter=',',  
 header=None,  
 names=['id','Area','MajorAxisLength','MinorAxisLength','Eccentricity','ConvexArea','EquivDiameter','Extent','Perimeter','Roundness','AspectRation','Class'])  
d.head()  
dX = d.iloc[ : , 0:4 ]  
dy = d['Class']  
X\_train, X\_holdout, y\_train, y\_holdout = \  
train\_test\_split(dX, dy, test\_size=0.3, random\_state=12)  
  
# Обучение модели  
tree = DecisionTreeClassifier(max\_depth=5,  
 random\_state=21,  
 max\_features=2)  
tree.fit(X\_train, y\_train)  
  
# Получение оценки hold-out  
tree\_pred = tree.predict(X\_holdout)  
accur = accuracy\_score(y\_holdout, tree\_pred)  
print(accur)

4.2:

import numpy as np  
import pandas as pd  
from sklearn.tree import DecisionTreeClassifier  
import seaborn as sns  
from matplotlib import pyplot as plt  
from sklearn.model\_selection import cross\_val\_score  
  
data\_source = 'Dataset.txt'  
d = pd.read\_table(data\_source, delimiter=',',  
 header=None,  
 names=['id','Area','MajorAxisLength','MinorAxisLength','Eccentricity','ConvexArea','EquivDiameter','Extent','Perimeter','Roundness','AspectRation','Class'])  
d.head()  
dX = d.iloc[ : , 0:10 ]  
dy = d['Class']  
  
# Значения параметра max\_depth  
d\_list = list(range(1,20))  
# Пустой список для хранения значений точности  
cv\_scores = []  
# В цикле проходим все значения K  
for d in d\_list:  
 tree = DecisionTreeClassifier(max\_depth=d,  
 random\_state=21,  
 max\_features=2)  
 scores = cross\_val\_score(tree, dX, dy, cv=10, scoring='accuracy')  
 cv\_scores.append(scores.mean())  
  
# Вычисляем ошибку (misclassification error)  
MSE = [1-x for x in cv\_scores]  
  
# Строим график  
plt.plot(d\_list, MSE)  
plt.xlabel('Макс. глубина дерева (max\_depth)');  
plt.ylabel('Ошибка классификации (MSE)')  
plt.show()  
  
# Ищем минимум  
d\_min = min(MSE)  
  
# Пробуем найти прочие минимумы (если их несколько)  
all\_d\_min = []  
for i in range(len(MSE)):  
 if MSE[i] <= d\_min:  
 all\_d\_min.append(d\_list[i])  
  
# печатаем все K, оптимальные для модели  
print('Оптимальные значения max\_depth: ', all\_d\_min)

4.3:

import numpy as np  
import pandas as pd  
from sklearn.tree import DecisionTreeClassifier  
from sklearn.model\_selection import GridSearchCV, cross\_val\_score  
from sklearn import tree  
  
data\_source = 'Dataset.txt'  
d = pd.read\_table(data\_source, delimiter=',',  
 header=None,  
 names=['id','Area','MajorAxisLength','MinorAxisLength','Eccentricity','ConvexArea','EquivDiameter','Extent','Perimeter','Roundness','AspectRation','Class'])  
d.head()  
dX = d.iloc[ : , 0:10 ]  
dy = d['Class']  
  
dtc = DecisionTreeClassifier(max\_depth=10, random\_state=21, max\_features=2)  
  
tree\_params = { 'max\_depth': range(1,20), 'max\_features': range(1,10) }  
tree\_grid = GridSearchCV(dtc, tree\_params, cv=10, verbose=True, n\_jobs=-1)  
tree\_grid.fit(dX, dy)  
  
print('\n')  
print('Лучшее сочетание параметров: ', tree\_grid.best\_params\_)  
print('Лучшие баллы cross validation: ', tree\_grid.best\_score\_)  
  
# Генерируем графическое представление лучшего дерева (сохранится в файле)  
tree.export\_graphviz(tree\_grid.best\_estimator\_,  
 feature\_names=dX.columns,  
 class\_names=("Gonen","Jasmine"),  
 out\_file='iris\_tree.dot',  
 filled=True, rounded=True)

Контрольные вопросы:

1. Дерево решений строится путем разбиения данных на подгруппы на основе признаков, чтобы последовательно принять решение о классификации или регрессии.

2. Информативность - мера того, насколько хорошо признак разделяет данные на классы. Статистически определяется через критерий информативности, например, критерий Джини или энтропийный критерий.

3. Энтропийная информативность оценивает степень неопределенности в данных. Чем меньше энтропия, тем более информативен признак.

4. Многоклассовая информативность используется для оценки информативности признака при множественных классах. Помогает определить, насколько хорошо признак различает несколько классов.

5. Бинаризация количественных признаков преобразует их в бинарные путем установления порогового значения. Это помогает использовать количественные данные в алгоритмах, которые работают только с бинарными данными.

6. Поиск закономерностей в форме конъюнкций включает алгоритмы, которые ищут правила, объединяющие несколько условий для классификации объектов. Например, "если условие 1 и условие 2 выполняются, то объект принадлежит классу А".

Вывод: изучил принципы построения информационных систем с использованием логических методов классификации.