Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

Кафедра прикладной математики и кибернетики

Отчет

По лабораторной работе №2

Логические классификаторы.

Решающие деревья.

Выполнил:

студент гр. ИП-712

Алексеев С.В.

Работу проверил:
Ассистент кафедры
Морозова К.И.

Новосибирск 2020 г.

Задание

Реализовать классификатор на основе решающих деревьев.

Процесс выполнения работы

1. Данные были разбиты на обучающий и тестовый наборы с соотношением 7:3
2. На обучающем наборе данных было создано решающее дерево.
3. На тестовом наборе данных было протестировано созданное решающее дерево.

- Точность на 10 разбиениях по умолчанию составила:
  0.76, 0.77, 0.71, 0.76, 0.74, 0.76, 0.78, 0.72, 0.72, 0.72.
- Точность на 5 разбиениях с максимальной глубиной дерева равной 5 и максимальным количеством листьев 10:
  0.79, 0.77, 0.76, 0.82, 0.81
- Точность на 5 разбиениях с максимальной глубиной дерева равной 10 и максимальным количеством листьев 5:
  0.77, 0.84, 0.76, 0.8, 0.78

Вывод

В лабораторной работе был реализован классификатор на основе решающих деревьев. При изменении глубины дерева и количества листьев можно наблюдать небольшое улучшение результатов. Классификатор на основе решающих деревьев показывает точность в данной задаче в районе 0.75.

Листинг

```
import numpy as np
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.impute import SimpleImputer

dataset = pd.read_csv('heart_data.csv')
dataset.head()
X = dataset.loc[:, 'age':'thal']
y = dataset['goal']
X = X.replace('?', np.nan)
y = y.replace('?', np.nan)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
imputer = SimpleImputer()
X_train = imputer.fit_transform(X_train)

i = 1
while i < 20:
    clf = DecisionTreeClassifier(max_depth=i, min_samples_leaf=1)
    clf.fit(X_train, y_train)
    X_test = imputer.transform(X_test)
    y_pred = clf.predict(X_test)
```

```
    print(clf.score(X_train, y_train))
    print(clf.score(X_test, y_test))
    print('')
    i = i + 1

clf.fit(X_train, y_train)
X_test = imputer.transform(X_test)
y_pred = clf.predict(X_test)
print(clf.score(X_train, y_train))
print(clf.score(X_test, y_test))
# print(y_pred)

print(clf.max_depth)
print(clf.min_samples_leaf)
```