Гузенко А.М. Группа 7.2. Вариант 4

Лабораторная работа № 6

Алгоритм k ближайших соседей

“Принятие решения сотрудниками банка о выдаче кредита”

Цель

Применить алгоритм kNN (k ближайших соседей) для предсказания решения сотрудниками банка от неизвестных переменных.

Описание данных

	v1_a	v1_b	v2	v3	v4_l	v4_y	v4_u	v5_gg	v5_p	v5_g	v6_j	v6_ff	v6_aa	v6_e	v6_i
0	0	1	30,83	0	0	0	1	0	0	1	0	0	0	0	0
1	1	0	58,67	4,46	0	0	1	0	0	1	0	0	0	0	0
2	1	0	24,5	0,5	0	0	1	0	0	1	0	0	0	0	0
3	0	1	27,83	1,54	0	0	1	0	0	1	0	0	0	0	0
4	0	1	20,17	5,625	0	0	1	0	0	1	0	0	0	0	0
5	0	1	32,08	4	0	0	1	0	0	1	0	0	0	0	0
6	0	1	33,17	1,04	0	0	1	0	0	1	0	0	0	0	0
7	1	0	22,92	11,585	0	0	1	0	0	1	0	0	0	0	0
8	0	1	54,42	0,5	0	1	0	0	1	0	0	0	0	0	0
9	0	1	42,5	4,915	0	1	0	0	1	0	0	0	0	0	0
10	0	1	22,08	0,83	0	0	1	0	0	1	0	0	0	0	0
11	0	1	29,92	1,835	0	0	1	0	0	1	0	0	0	0	0
12	1	0	38,25	6	0	0	1	0	0	1	0	0	0	0	0
13	0	1	48,08	6,04	0	0	1	0	0	1	0	0	0	0	0

	v6_x	v6_d	v6_c	v6_k	v6_cc	v6_r	v6_m	v6_q	v6_w	v7_n	v7_dd	v7_o	v7_z	v7_j	v7_ff	v7_bb
2	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
7	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
12	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
17	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
20	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	AH	AI	AJ	AK	AL	AM	AN	AO	AP	AQ	AR	AS	AT	AU	AV	AW
1	v7_v	v8	v9_f	v9_t	v10_f	v10_t	v11	v12_t	v12_f	v13_p	v13_s	v13_g	v14	v15	desired1	desired2
2	1	1,25	0	1	0	1	1	0	1	0	0	1	202	0	0	1
3	0	3,04	0	1	0	1	6	0	1	0	0	1	43	560	0	1
4	0	1,5	0	1	1	0	0	0	1	0	0	1	280	824	0	1
5	1	3,75	0	1	0	1	5	1	0	0	0	1	100	3	0	1
6	1	1,71	0	1	1	0	0	0	1	0	1	0	120	0	0	1
7	1	2,5	0	1	1	0	0	1	0	0	0	1	360	0	0	1
8	0	6,5	0	1	1	0	0	1	0	0	0	1	164	31285	0	1
9	1	0,04	0	1	1	0	0	0	1	0	0	1	80	1349	0	1
10	0	3,96	0	1	1	0	0	0	1	0	0	1	180	314	0	1
11	1	3,165	0	1	1	0	0	1	0	0	0	1	52	1442	0	1
12	0	2,165	1	0	1	0	0	1	0	0	0	1	128	0	0	1
13	0	4,335	0	1	1	0	0	0	1	0	0	1	260	200	0	1
14	1	1	0	1	1	0	0	1	0	0	0	1	0	0	0	1
15	1	0,04	1	0	1	0	0	0	1	0	0	1	0	2690	0	1
16	1	5	0	1	0	1	7	1	0	0	0	1	0	0	0	1
17	1	0,25	0	1	0	1	10	1	0	0	0	1	320	0	0	1
18	1	0,96	0	1	0	1	3	1	0	0	0	1	396	0	0	1
19	1	3,17	0	1	0	1	10	0	1	0	0	1	120	245	0	1
20	0	0,665	0	1	1	0	0	1	0	0	0	1	0	0	0	1

Число наблюдений – 655.

Число переменных – 15, из них 6 измерены в количественной (непрерывной) шкале, 9 – в шкале наименований (номинальной шкале).

В файле содержится также два столбца, показывающие, была удовлетворена заявка, или она была отвергнута.

Выполнение работы

1. Импортируем нужные библиотеки.

```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
```

2. Объявим константу для пути к программе.

```
""" CONSTS """
PATH = os.path.dirname(os.path.abspath(__file__)) + '\\'
```

3. Прочтем данные из файла

```
""" CONSTS """
PATH = os.path.dirname(os.path.abspath(__file__)) + '\\'
```

4. Вычленим из них нужные для нас переменные.

```
X = input_data.iloc[:, 0:46].values
y = np.array(list(map(lambda x: x[0], input_data.iloc[:, 47:48].values)))
```

5. Разделим данные на данные для обучения модели и тестовые.

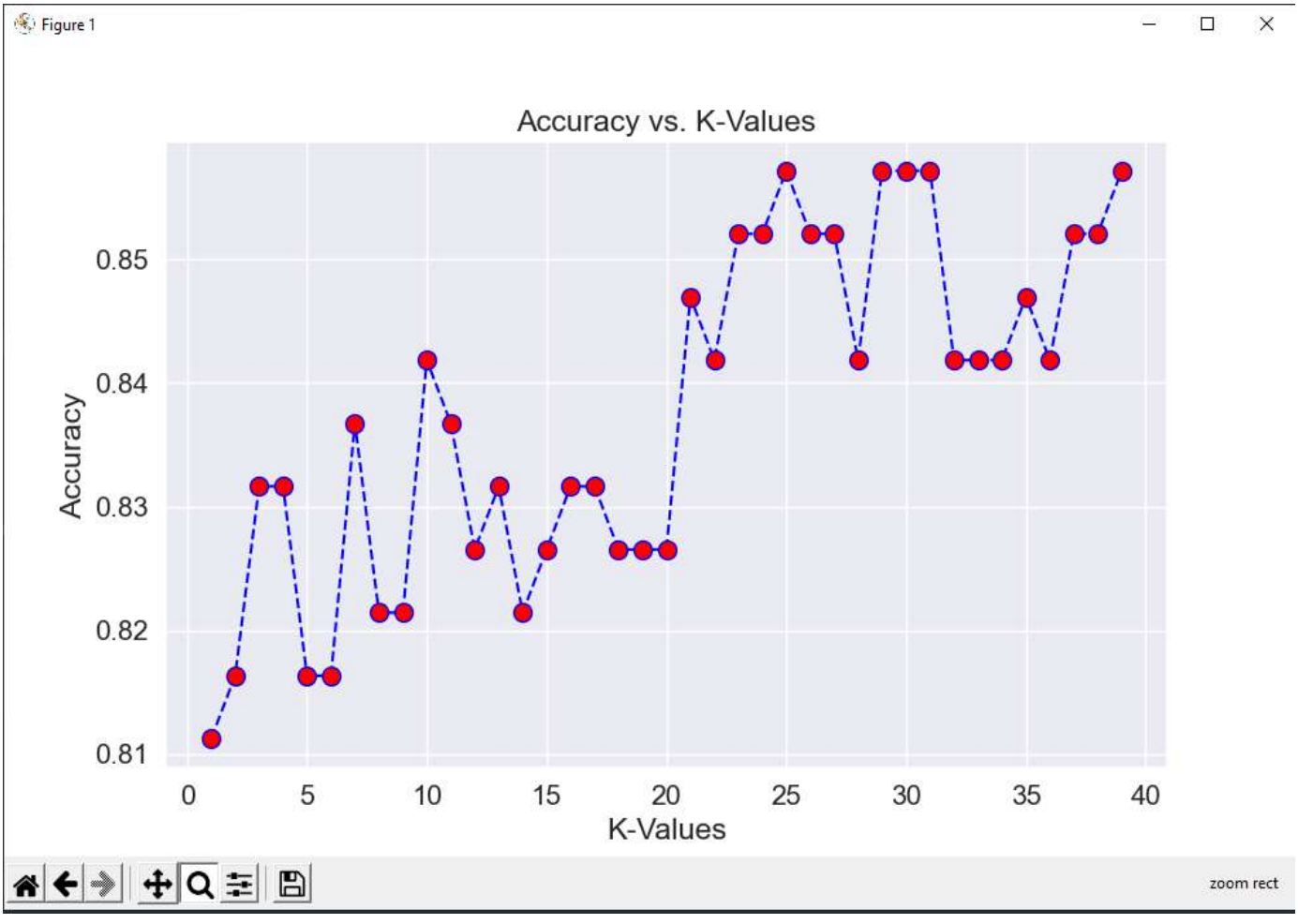
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30)
```

6. Стандартизируем данные путем удаления среднего и масштабирования до единичной дисперсии.

```
scaler = StandardScaler()
scaler.fit(X_train)
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

7. Найдем оптимальное кол-во k.

```
acc = []
for i in range(1, 40):
    knn = KNeighborsClassifier(n_neighbors=i)
    knn.fit(X_train, y_train)
    acc.append(knn.score(X_test, y_test))
plt.figure(figsize=(10, 4))
plt.plot(range(1, 40), acc, color='blue', linestyle='dashed', marker='o',
markerfacecolor='red', markersize=10)
plt.title('Accuracy vs. K-Values')
plt.xlabel('K-Values')
plt.ylabel('Accuracy')
plt.show()
```



8. Возьмем оптимальное кол-во k из предыдущего пункта, создадим модель и обучим ее на данных для обучения.

```
knn = KNeighborsClassifier(n_neighbors=39, p=2, metric='minkowski')
knn.fit(X_train, y_train)
```

9. Сделаем предсказание на тестовых данных.

```
y_pred = knn.predict(X_test)
```

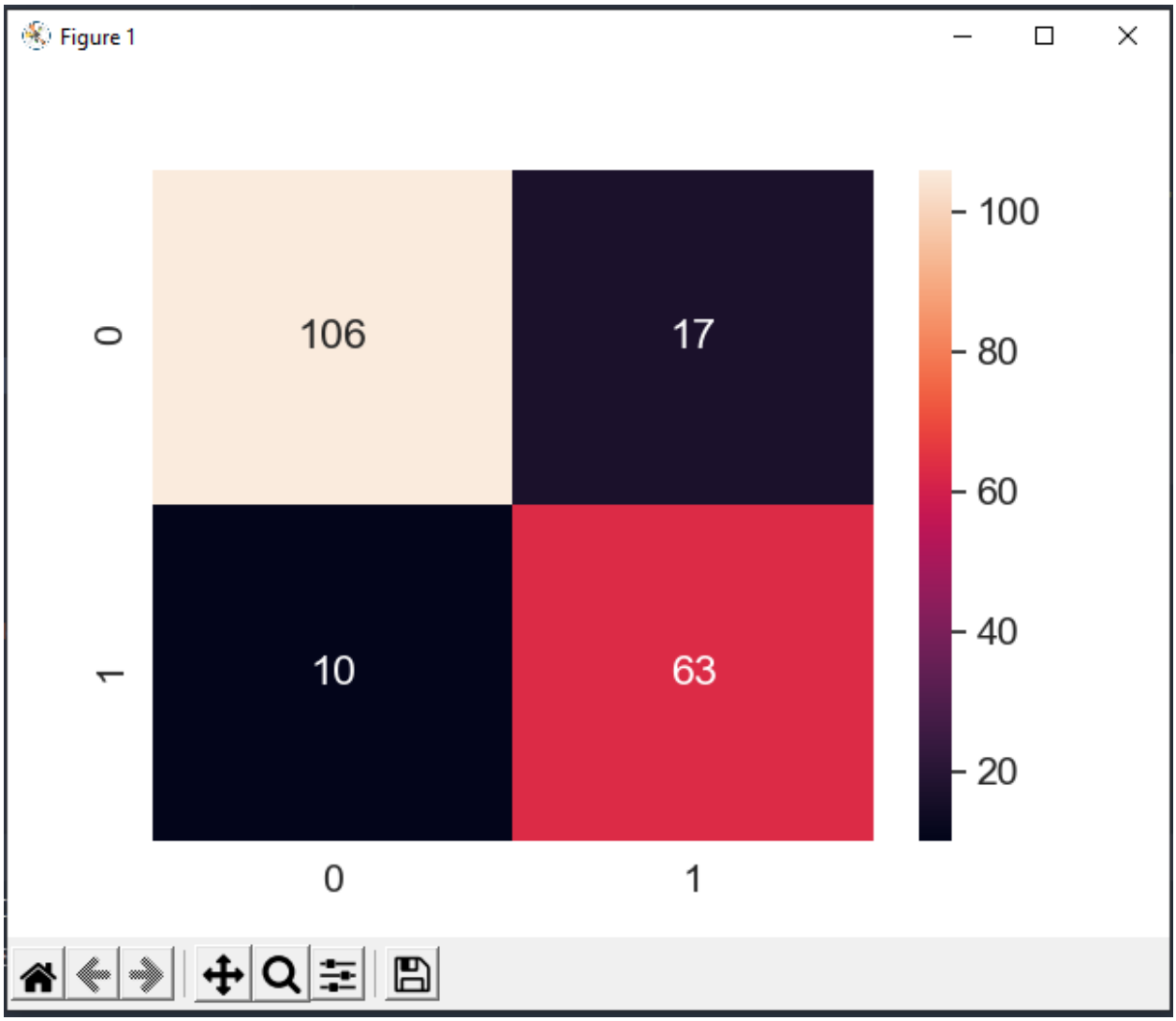
10. Выведем процент ошибки, сравнив предсказанные данные с тестовыми.

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

```
Accuracy: 0.8622448979591837
```

11. Выведем матрицу ошибок (таблицу сопряженности).

```
cmat = confusion_matrix(y_test, y_pred)
sns.set(font_scale=1.4)
sns.heatmap(cmat, annot=True, fmt="d")
plt.show()
```



Вывод

Выполнив данную лабораторную работу, мы создали модель по данным решения сотрудников банка по выдаче кредитов. Мы определили оптимальное кол-во k, равное 39. Модель дает точность в 86%. По матрице ошибок (таблице сопряженности) мы видим, что модель ошибалась в обоих случаях (одобрен/неодобрен кредит) примерно одинаково.