

Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №3
По дисциплине: «ОМО»

Выполнил:
Студент 3-го курса
Группы АС-66
Куган Н. Л.
Проверил:
Крощенко А.А.

Цель работы: На практике сравнить работу нескольких алгоритмов классификации, таких как метод k-ближайших соседей (k-NN), деревья решений и метод опорных векторов (SVM). Научиться подбирать гиперпараметры моделей и оценивать их влияние на результат.

Задачи:

1. Загрузить датасет по варианту;
2. Разделить данные на обучающую и тестовую выборки;
3. Обучить на обучающей выборке три модели: k-NN, Decision Tree и SVM;
4. Для модели k-NN исследовать, как меняется качество при разном количестве соседей (k);
5. Оценить точность каждой модели на тестовой выборке;
6. Сравнить результаты, сделать выводы о применимости каждого метода для данного набора данных.

Ход работы

Вариант 6

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.metrics import recall_score
import matplotlib.pyplot as plt

# === 1. Загрузка корректного набора данных Pima Indians Diabetes ===
# Данные идентичны твоему варианту (из UCI Machine Learning Repository)
url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"
columns = [
    "Pregnancies", "Glucose", "BloodPressure", "SkinThickness",
    "Insulin", "BMI", "DiabetesPedigreeFunction", "Age", "Outcome"
]
data = pd.read_csv(url, names=columns)

print("Первые строки данных:")
print(data.head(), "\n")

# === 2. Разделение данных ===
X = data.drop('Outcome', axis=1)
y = data['Outcome']

# === 3. Стандартизация ===
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# === 4. Разделение выборки ===
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3,
random_state=42, stratify=y)

# === 5. Обучение моделей ===
models = {
    "k-NN (k=5)": KNeighborsClassifier(n_neighbors=5),
    "Decision Tree (max_depth=4)": DecisionTreeClassifier(max_depth=4,
random_state=42),
    "SVM (linear)": SVC(kernel='linear', random_state=42)
}
```

```

recalls = {}

for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    recalls[name] = recall_score(y_test, y_pred)

# === 6. Вывод результатов ===
print("\nRecall (наличие диабета = 1):")
for name, score in recalls.items():
    print(f"{name}: {score:.3f}")

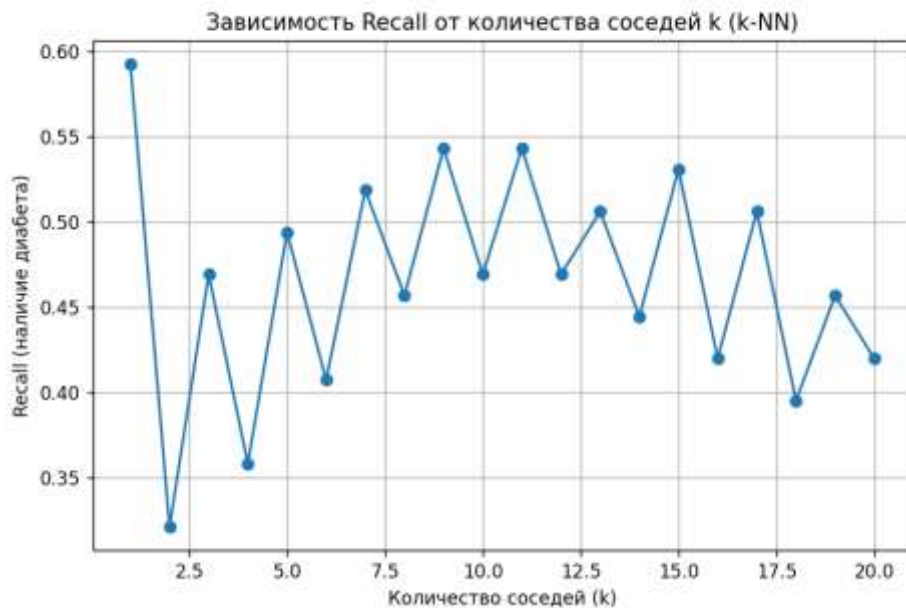
# === 7. Исследование зависимости recall от k для k-NN ===
k_values = range(1, 21)
recall_list = []

for k in k_values:
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)
    y_pred = knn.predict(X_test)
    recall_list.append(recall_score(y_test, y_pred))

plt.figure(figsize=(8, 5))
plt.plot(k_values, recall_list, marker='o')
plt.title("Зависимость Recall от количества соседей k (k-NN)")
plt.xlabel("Количество соседей (k)")
plt.ylabel("Recall (наличие диабета)")
plt.grid(True)
plt.show()

# === 8. Итоговый вывод ===
best_model = max(recalls, key=recalls.get)
print(f"\nНаилучший результат показала модель: {best_model} (Recall = {recalls[best_model]:.3f})")

```



```
Recall (наличие диабета = 1):  
k-NN (k=5): 0.494  
Decision Tree (max_depth=4): 0.605  
SVM (linear): 0.494
```

```
Наилучший результат показала модель: Decision Tree (max_depth=4) (Recall = 0.605)
```

Вывод: Я на практике сравнил работу нескольких алгоритмов классификации, таких как метод k-ближайших соседей (k-NN), деревья решений и метод опорных векторов (SVM). Научился подбирать гиперпараметры моделей и оценивать их влияние на результат.