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

Лабораторная работа №3
По дисциплине: «Основы машинного обучения»
Тема**: «Сравнение классических методов классификации»**

Выполнил:
3-го курса
Группы АС-65
Кисель М. С.
Проверил:
Крощенко А.А.

Брест 2025

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

Ход работы

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.

Вариант 8

• Seeds

• Классифицировать семена на три сорта пшеницы (Kama, Rosa, Canadian) на основе их геометрических параметров

• **Задания:**

1. Загрузите и стандартизируйте данные;
2. Разделите выборку на обучающую и тестовую;
3. Обучите три классификатора;
4. Сравните общую точность (accuracy) всех трех моделей;
5. Визуализируйте данные в 2D (например, с помощью PCA), раскрасив точки в соответствии с предсказаниями лучшей модели.

1. Загрузите и стандартизируйте данные:

```
import os
import pandas as pd
# Переходим в директорию с файлом
os.chdir("d:/Универ/ОМО/ОМО2025/Лаба3/")

# Загружаем данные
columns = [
    'area',
    'perimeter',
    'compactness',
    'length_of_kernel',
    'width_of_kernel',
    'asymmetry_coefficient',
    'length_of_kernel_groove',
    'class'
]

df = pd.read_csv("seeds_dataset.txt", delim_whitespace=True, names=columns)

print(" ◇  Первые 5 строк:")
print(df.head())
print("\n ◇  Информация о данных:")
print(df.info())
print("\n ◇  Проверка на пропуски:")
print(df.isnull().sum())
print("\n ◇  Статистика по числовым признакам:")
print(df.describe())
```

```
# Стандартизация (обучаем scaler только на обучающей выборке)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

2. Разделите выборку на обучающую и тестовую:

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# Разделяем признаки и целевую переменную
X = df.drop('class', axis=1)
y = df['class']

# Разделяем выборку на обучающую и тестовую (80% / 20%)
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)
```

3. Обучите три классификатора:

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
# Инициализируем модели
knn = KNeighborsClassifier(n_neighbors=5)
tree = DecisionTreeClassifier(random_state=42)
svm = SVC(kernel='rbf', random_state=42)

# Обучаем модели на обучающей выборке
knn.fit(X_train_scaled, y_train)
tree.fit(X_train_scaled, y_train)
svm.fit(X_train_scaled, y_train)

# Делаем предсказания на тестовой выборке
y_pred_knn = knn.predict(X_test_scaled)
y_pred_tree = tree.predict(X_test_scaled)
y_pred_svm = svm.predict(X_test_scaled)
```

4. Сравните общую точность (accuracy) всех трех моделей:

```
from sklearn.metrics import accuracy_score
# Вычисляем accuracy для каждой модели
acc_knn = accuracy_score(y_test, y_pred_knn)
acc_tree = accuracy_score(y_test, y_pred_tree)
acc_svm = accuracy_score(y_test, y_pred_svm)

# Выводим результаты
print("Точность (Accuracy) каждой модели:")
print(f"KNN: {acc_knn:.3f}")
print(f"Decision Tree: {acc_tree:.3f}")
print(f"SVM: {acc_svm:.3f}")

# Определим лучшую модель
best_model = max(
    [('KNN', acc_knn), ('Decision Tree', acc_tree), ('SVM', acc_svm)],
    key=lambda x: x[1]
)
print(f"\nЛучшая модель: {best_model[0]} (accuracy = {best_model[1]:.3f})")
```

```
Точность (Accuracy) каждой модели:
KNN: 0.905
Decision Tree: 0.881
SVM: 0.905

Лучшая модель: KNN (accuracy = 0.905)
Лучшее k: 1, Accuracy = 0.905
```

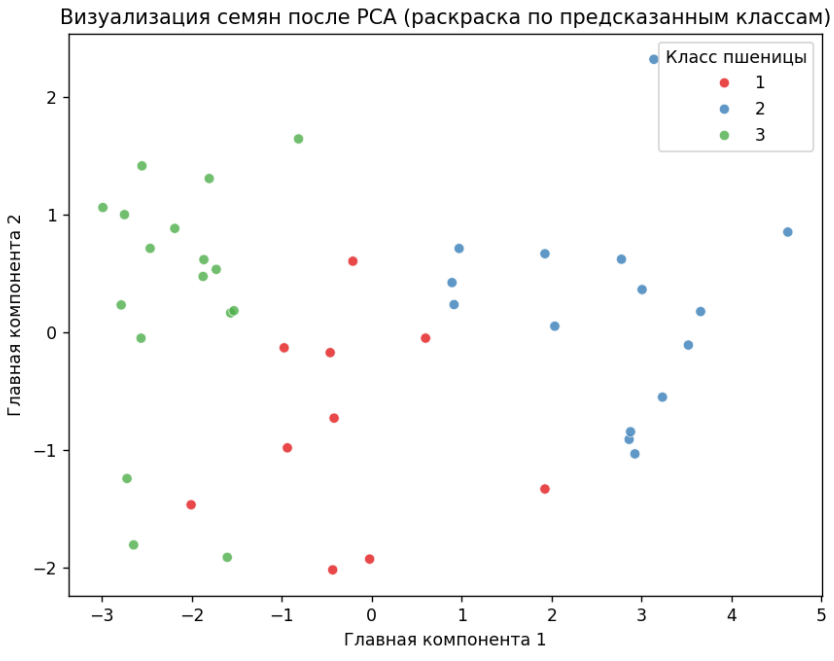
5. Визуализируйте данные в 2D (например, с помощью PCA), раскрасив точки в соответствии с предсказаниями лучшей модели:

```python
from sklearn.decomposition import PCA
import seaborn as sns
# Предсказания лучшей модели (допустим, KNN с оптимальным k)
best_knn = KNeighborsClassifier(n_neighbors=best_k)
best_knn.fit(X_train_scaled, y_train)
y_pred_best = best_knn.predict(X_test_scaled)

# Применяем PCA (уменьшаем размерность до 2 признаков)
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_test_scaled)

# Строим DataFrame для удобства
pca_df = pd.DataFrame(data=X_pca, columns=['PCA1', 'PCA2'])
pca_df['Predicted class'] = y_pred_best

# Визуализация
plt.figure(figsize=(8, 6))
sns.scatterplot(
    x='PCA1',
    y='PCA2',
    hue='Predicted class',
    data=pca_df,
    palette='Set1',
    alpha=0.8
)
plt.title('Визуализация семян после PCA (раскраска по предсказанным классам)')
plt.xlabel('Главная компонента 1')
plt.ylabel('Главная компонента 2')
plt.legend(title='Класс пшеницы')
plt.show()
```

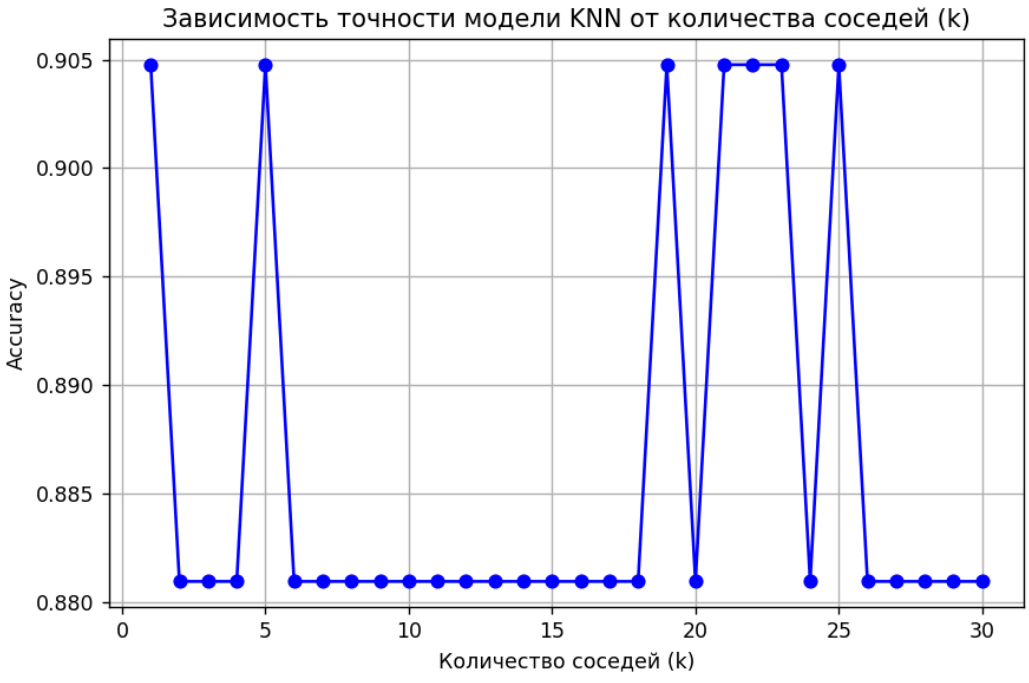


6. Для модели k-NN исследовать, как меняется качество при разном количестве соседей (k):

```
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
from sklearn.neighbors import KNeighborsClassifier
# Проверяем значения k от 1 до 30
k_values = range(1, 31)
accuracies = []

for k in k_values:
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train_scaled, y_train)
    y_pred_k = knn.predict(X_test_scaled)
    acc = accuracy_score(y_test, y_pred_k)
    accuracies.append(acc)

# Строим график
plt.figure(figsize=(8, 5))
plt.plot(k_values, accuracies, marker='o', linestyle='-', color='blue')
plt.title('Зависимость точности модели KNN от количества соседей (k)')
plt.xlabel('Количество соседей (k)')
plt.ylabel('Accuracy')
plt.grid(True)
plt.show()
# Находим лучшее значение k
best_k = k_values[accuracies.index(max(accuracies))]
best_acc = max(accuracies)
print(f"Лучшее k: {best_k}, Accuracy = {best_acc:.3f}")
```



7. Сравнить результаты, сделать выводы о применимости каждого метода для данного набора данных.

Вывод:Модель KNN показала наилучший результат (0.905).KNN и SVM работают примерно одинаково, что указывает на хорошо разделяемые классы.Decision Tree чуть слабее — возможно, из-за переобучения на малой выборке.