

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

**Лабораторная работа №3**  
**По дисциплине: «ОМО»**  
**Тема:** Сравнение классических методов классификации

**Выполнил:**  
Студент 3 курса  
Группы АС-66  
Лысюк Р. А.  
**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 4

- Digits
- Распознать, какая цифра (от 0 до 9) изображена на картинке 8x8 пикселей
- **Задания:**
  1. Загрузите встроенный в `scikit-learn` набор данных `digits`;
  2. Разделите данные на обучающую и тестовую выборки;
  3. Обучите три модели (k-NN, Decision Tree, SVM) для многоклассовой классификации;
  4. Для каждой модели выведите `classification_report` (`sklearn.metrics`), содержащий основные метрики для каждого класса;
  5. Сравните общую точность моделей и определите, какая из них лучше всего подходит для этой задачи.

Код:

```
import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.datasets import load_digits

from sklearn.model_selection import train_test_split

from sklearn.neighbors import KNeighborsClassifier

from sklearn.tree import DecisionTreeClassifier

from sklearn.svm import SVC

from sklearn.metrics import classification_report, accuracy_score, confusion_matrix
```

```
digits = load_digits()
X, y = digits.data, digits.target
```

```
fig, axes = plt.subplots(1, 5, figsize=(10, 3))
for i, ax in enumerate(axes):
    ax.imshow(digits.images[i], cmap='gray')
    ax.set_title(f'Цифра: {digits.target[i]}')
    ax.axis('off')
plt.tight_layout()
plt.show()
```

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=42)
```

```
models = {
    "k-NN": KNeighborsClassifier(n_neighbors=3),
    "Decision Tree": DecisionTreeClassifier(random_state=42),
    "SVM": SVC(kernel='linear', random_state=42)
}
```

```
for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    print(f"\n=== {name} ===")
    print(classification_report(y_test, y_pred))
    print(f"Accuracy: {accuracy_score(y_test, y_pred):.4f}")
```

```
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(6, 5))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title(f"Матрица ошибок: {name}")
plt.xlabel("Предсказано")
plt.ylabel("Истинное")
plt.show()
```

```
k_values = range(1, 11)
accuracies = []
```

```

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

plt.figure(figsize=(8, 4))
sns.lineplot(x=list(k_values), y=accuracies, marker='o')
plt.title("Зависимость точности от числа соседей в k-NN")
plt.xlabel("Число соседей (k)")
plt.ylabel("Точность")
plt.grid(True)
plt.show()

```

Вывод программы:

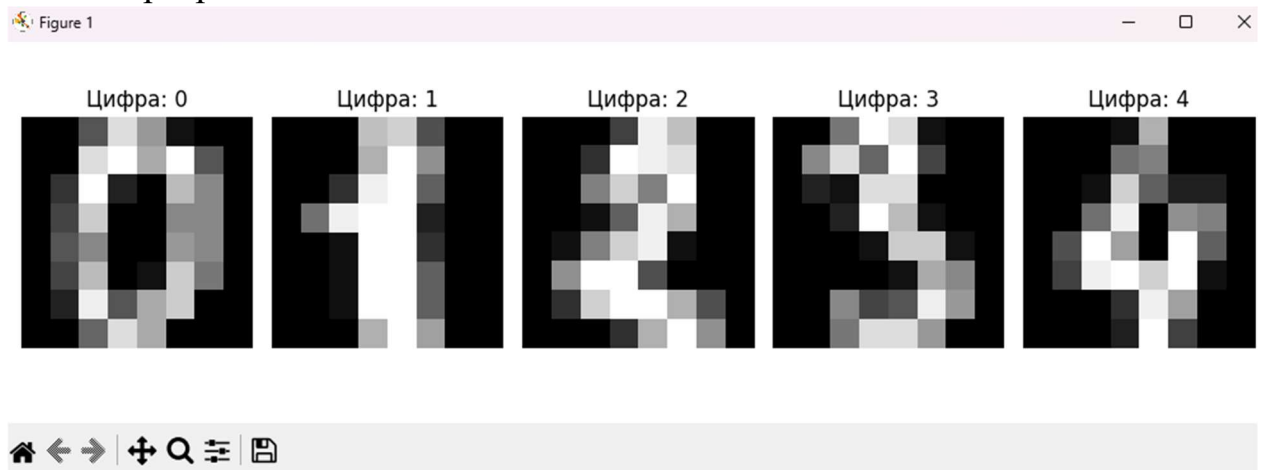


Figure 1

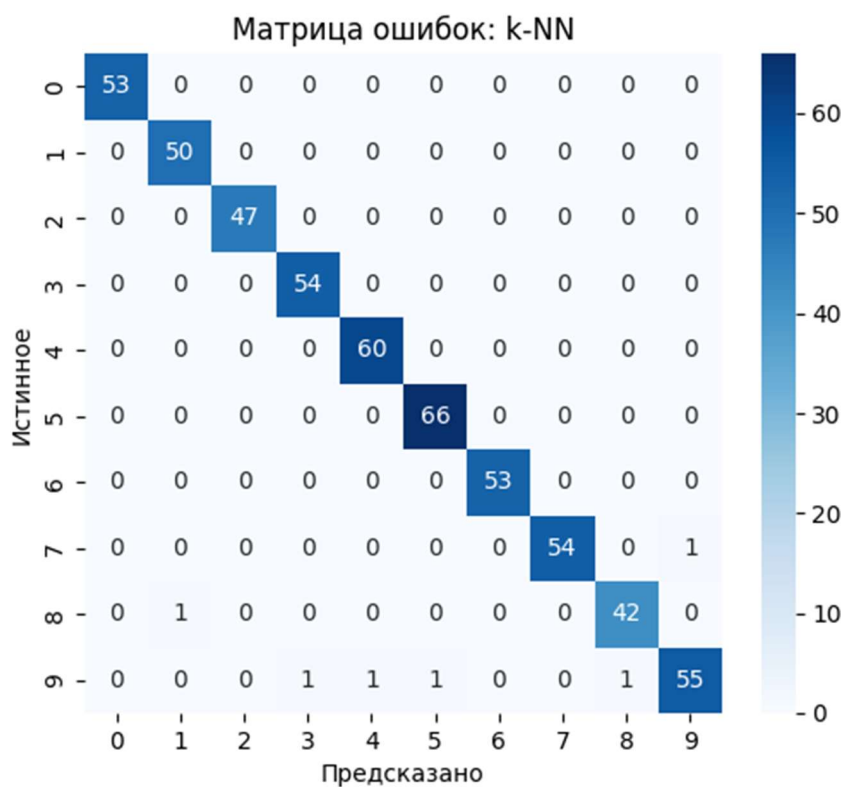
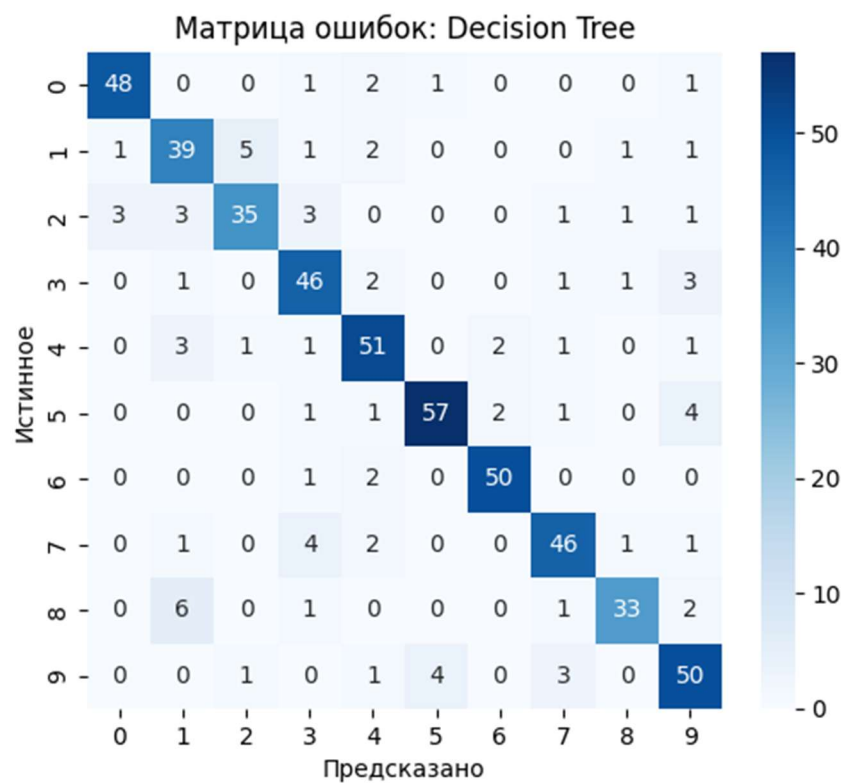
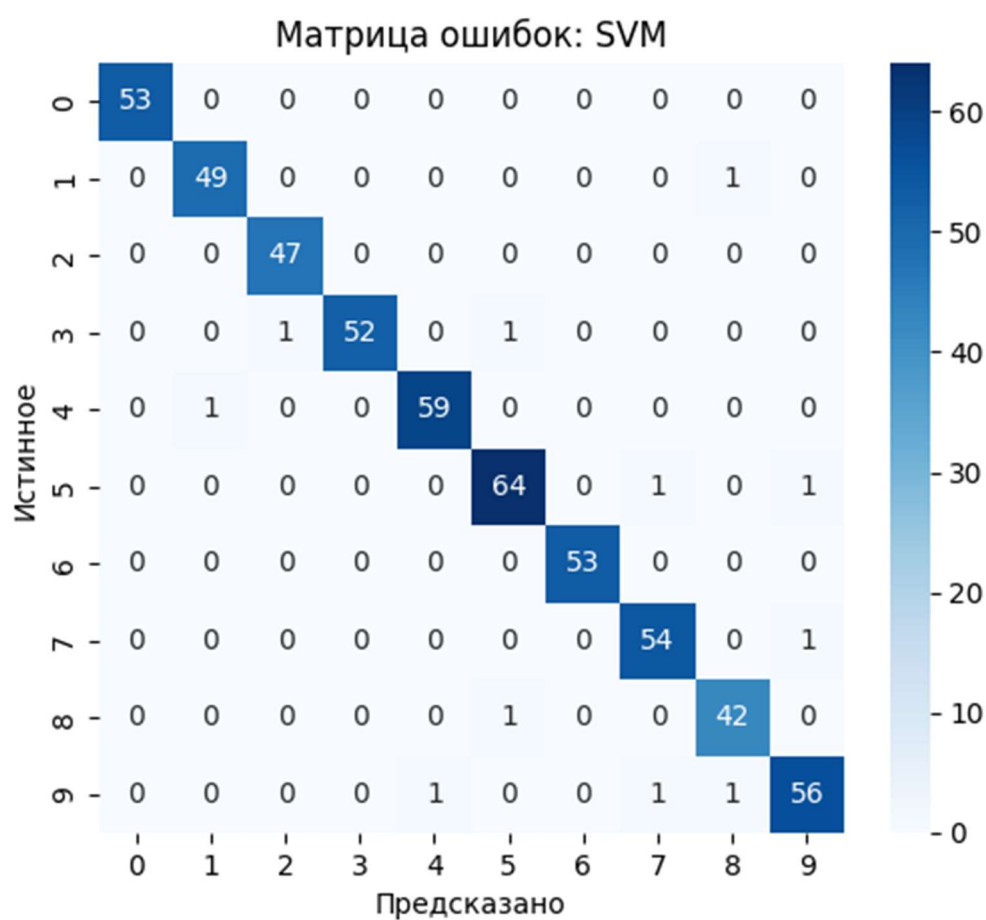


Figure 1



(x, y) = (, )

Figure 1



(x, y) = (, )

Figure 1

