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:
Студент 3 курса
Группы АС-66
Езепчук А.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.

Вариант 3

Ход работы

Wine Quality

Классифицировать вино на "хорошее" (оценка >= 7) и "обычное" (оценка < 7)

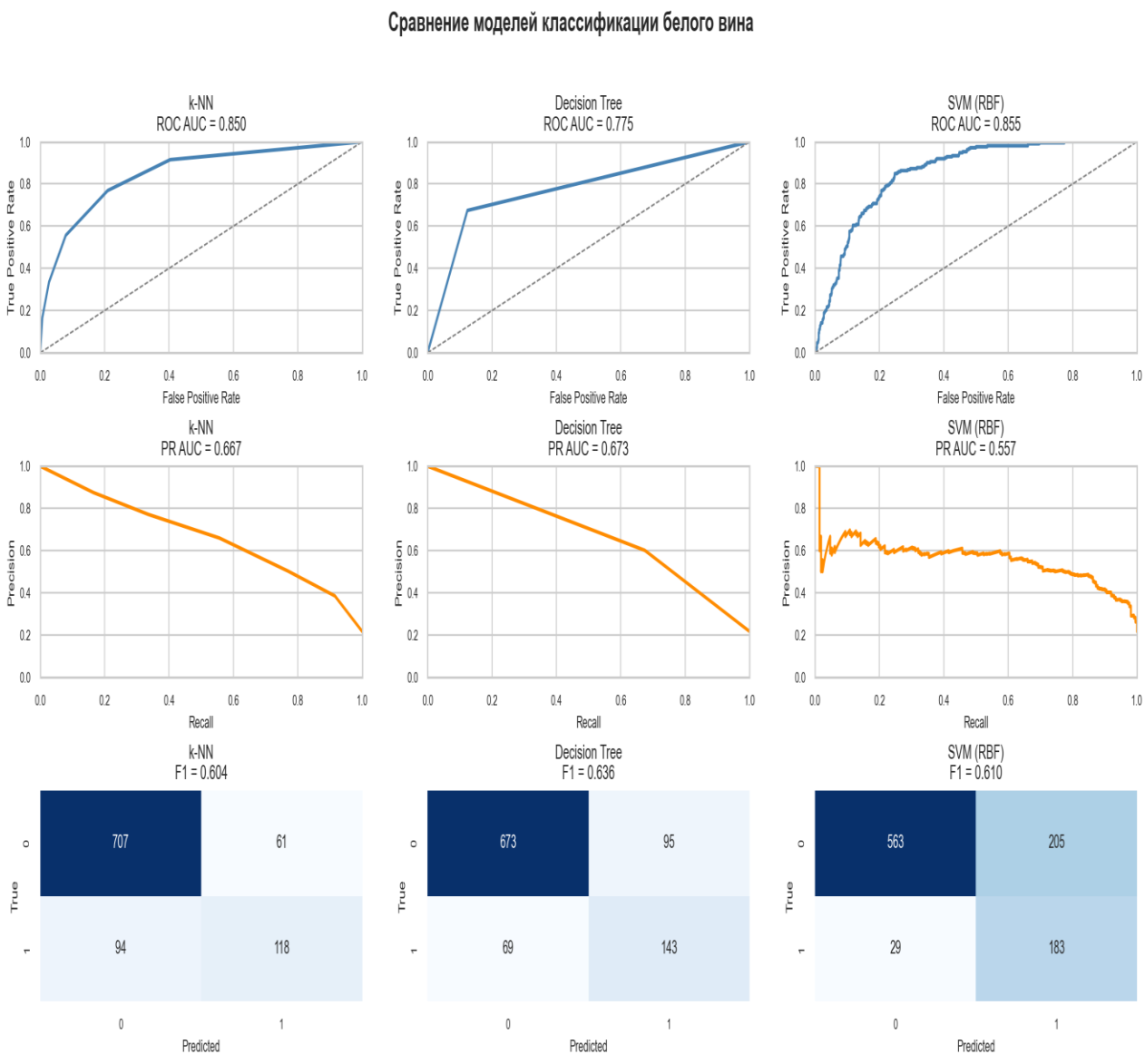
Задания:

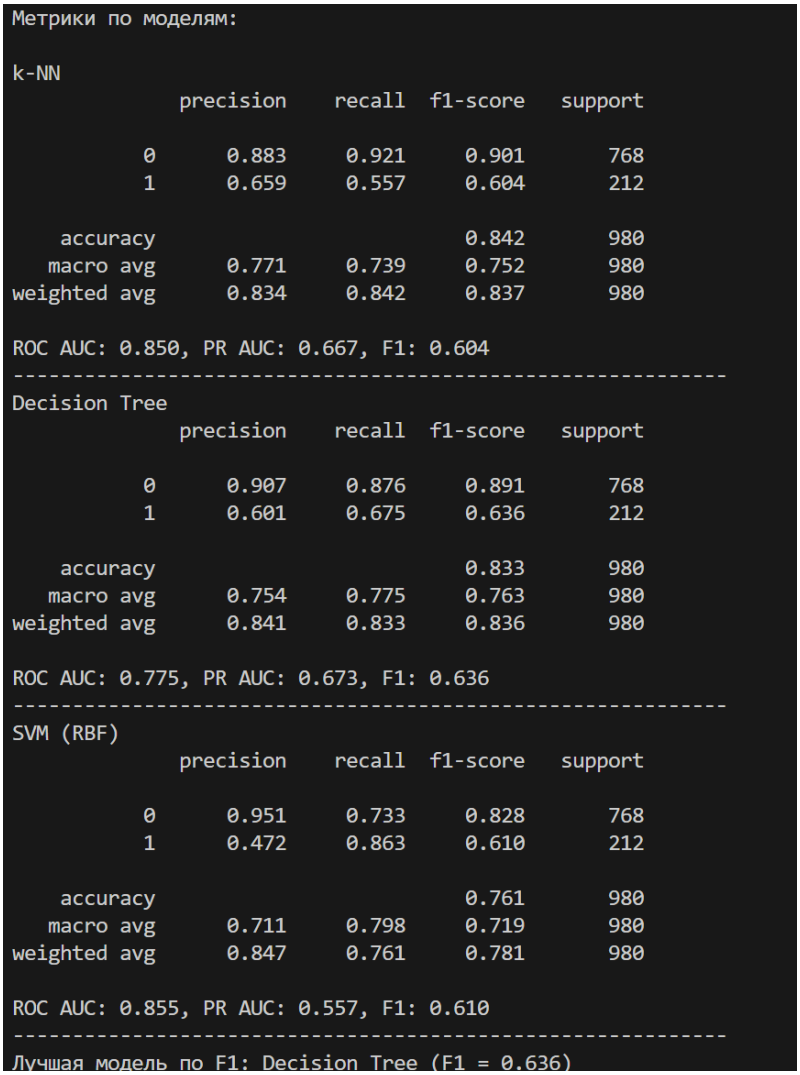
1. Загрузите данные и создайте бинарную целевую переменную;
2. Стандартизируйте признаки и разделите выборку;
3. Обучите модели k-NN, Decision Tree и SVM;
4. Сравните F1-score для каждой модели, так как классы могут быть несбалансированы;
5. Определите, какой алгоритм показал наилучший баланс между точностью и полнотой.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

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 (
    classification_report,
    confusion_matrix,
    roc_curve,
    roc_auc_score,
    precision_recall_curve,
    auc,
    f1_score
)

df = pd.read_csv("winequality-white.csv", sep=';')
df['good'] = (df['quality'] >= 7).astype(int)
X = df.drop(columns=['quality', 'good'])
y = df['good']

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

models = {
    "k-NN": KNeighborsClassifier(n_neighbors=5),
    "Decision Tree": DecisionTreeClassifier(random_state=42),
    "SVM (RBF)": SVC(kernel='rbf', class_weight='balanced', probability=True,
random_state=42)
}

results = {}
```

```
for name, model in models.items():
    if name == "Decision Tree":
        model.fit(X_train, y_train)
        y_pred = model.predict(X_test)
        y_proba = model.predict_proba(X_test)[:, 1]
    else:
        model.fit(X_train_scaled, y_train)
        y_pred = model.predict(X_test_scaled)
        y_proba = model.predict_proba(X_test_scaled)[:, 1]

    f1 = f1_score(y_test, y_pred)
    roc_auc = roc_auc_score(y_test, y_proba)
    prec, rec, _ = precision_recall_curve(y_test, y_proba)
    pr_auc = auc(rec, prec)
    cm = confusion_matrix(y_test, y_pred)

    results[name] = {
        "f1": f1,
        "roc_auc": roc_auc,
        "pr_auc": pr_auc,
        "fpr": roc_curve(y_test, y_proba)[0],
        "tpr": roc_curve(y_test, y_proba)[1],
        "precision": prec,
        "recall": rec,
        "cm": cm,
        "report": classification_report(y_test, y_pred, digits=3)
    }

sns.set(style="whitegrid", font_scale=1.0)

fig, axes = plt.subplots(3, len(models), figsize=(13, 9))
fig.subplots_adjust(hspace=0.8, wspace=0.6)
fig.suptitle("Сравнение моделей классификации белого вина", fontsize=15,
weight="bold", y=0.98)

for idx, (name, res) in enumerate(results.items()):
    # ROC
    ax = axes[0, idx]
    ax.plot(res["fpr"], res["tpr"], color="steelblue", lw=2)
    ax.plot([0, 1], [0, 1], color="gray", linestyle="--", lw=1)
    ax.set_title(f"{name}\nROC AUC = {res['roc_auc']:.3f}", fontsize=11)
    ax.set_xlabel("False Positive Rate", fontsize=9)
    ax.set_ylabel("True Positive Rate", fontsize=9)
    ax.set_xlim(0, 1)
    ax.set_ylim(0, 1)
    ax.tick_params(axis='both', labelsize=8)

    # Precision-Recall
    ax = axes[1, idx]
    ax.plot(res["recall"], res["precision"], color="darkorange", lw=2)
    ax.set_title(f"{name}\nPR AUC = {res['pr_auc']:.3f}", fontsize=11)
    ax.set_xlabel("Recall", fontsize=9)
    ax.set_ylabel("Precision", fontsize=9)
    ax.set_xlim(0, 1)
    ax.set_ylim(0, 1)
    ax.tick_params(axis='both', labelsize=8)

    # Confusion Matrix
    ax = axes[2, idx]
    sns.heatmap(
        res["cm"],
        annot=True,
        fmt="d",
        cmap="Blues",
        cbar=False,
```

```
            ax=ax,
            annot_kws={"size": 10}
        )
        ax.set_title(f"{name}\nF1 = {res['f1']:.3f}", fontsize=11)
        ax.set_xlabel("Predicted", fontsize=9)
        ax.set_ylabel("True", fontsize=9)
        ax.tick_params(axis='both', labelsize=8)

plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()

print("Метрики по моделям:\n")
for name, res in results.items():
    print(f"{name}")
    print(res["report"])
    print(f"ROC AUC: {res['roc_auc']:.3f}, PR AUC: {res['pr_auc']:.3f}, F1:
{res['f1']:.3f}")
    print("-" * 60)

best_model = max(results.items(), key=lambda kv: kv[1]['f1'])
print(f"Лучшая модель по F1: {best_model[0]} (F1 = {best_model[1]['f1']:.3f})")
```





```
Метрики по моделям:

k-NN
              precision    recall  f1-score   support

           0      0.883     0.921     0.901       768
           1      0.659     0.557     0.604       212

    accuracy                          0.842       980
   macro avg      0.771     0.739     0.752       980
weighted avg      0.834     0.842     0.837       980

ROC AUC: 0.850, PR AUC: 0.667, F1: 0.604
------------------------------------------------------------
Decision Tree
              precision    recall  f1-score   support

           0      0.907     0.876     0.891       768
           1      0.601     0.675     0.636       212

    accuracy                          0.833       980
   macro avg      0.754     0.775     0.763       980
weighted avg      0.841     0.833     0.836       980

ROC AUC: 0.775, PR AUC: 0.673, F1: 0.636
------------------------------------------------------------
SVM (RBF)
              precision    recall  f1-score   support

           0      0.951     0.733     0.828       768
           1      0.472     0.863     0.610       212

    accuracy                          0.761       980
   macro avg      0.711     0.798     0.719       980
weighted avg      0.847     0.761     0.781       980

ROC AUC: 0.855, PR AUC: 0.557, F1: 0.610
------------------------------------------------------------
Лучшая модель по F1: Decision Tree (F1 = 0.636)
```

Вывод: 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