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-65
Ярмак К. А.
Проверил:
Крощенко А. А.

Брест 2025

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

Вариант 12

KDD Cup 1999
Классифицировать сетевые подключения на "нормальные" и "атаки"
Задания:

1. Загрузите данные, преобразуйте категориальные признаки;

```
import pandas as pd
import time
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.metrics import recall_score

from columns import file_path, columns



# 1 encode cat features
df = pd.read_csv(file_path, names=columns)

cat_cols = ['protocol_type', 'service', 'flag']

encoders = {col: LabelEncoder() for col in cat_cols}
for col in cat_cols:
    df[col] = encoders[col].fit_transform(df[col])

print(df.iloc[:, [1, 2, 3]].head())
print("\n")
```

	protocol_type	service	flag
0	1	22	9
1	1	22	9
2	1	22	9
3	1	22	9
4	1	22	9

2. Создайте бинарную целевую переменную (*normal* vs *attack*);

```
# 2 target
df['target'] = df['label'].apply(lambda x: 0 if x == 'normal.' else 1)
df = df.drop('label', axis=1)

print(df.iloc[:, [-1]].head())
print("\n")
```

	target
0	0
1	0
2	0
3	0
4	0

3. Обучите k-NN, Decision Tree и SVM на небольшой подвыборке данных (например, 10 000 строк);
4. Сравните *recall* для класса "атака" и время обучения каждой модели;

```
# 3 k-nn decisionTree svm
df_sample = df.sample(10000, random_state=42)
X = df_sample.drop('target', axis=1)
y = df_sample['target']

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 = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)


results = []

# k-nn
for k in [1, 3, 5, 7, 9]:
    knn = KNeighborsClassifier(n_neighbors=k)
    start = time.time()
    knn.fit(X_train, y_train)
    end = time.time()
    y_pred = knn.predict(X_test)
    recall = recall_score(y_test, y_pred)
    results.append(("k-NN", f"k={k}", recall, end - start))

# decisionTree
tree = DecisionTreeClassifier(random_state=42)
start = time.time()
tree.fit(X_train, y_train)
end = time.time()
y_pred_tree = tree.predict(X_test)
recall_tree = recall_score(y_test, y_pred_tree)
results.append(("Decision Tree", "-", recall_tree, end - start))

# svm
svm = SVC(kernel='rbf', C=1.0, gamma='scale')
start = time.time()
svm.fit(X_train, y_train)
end = time.time()
y_pred_svm = svm.predict(X_test)
recall_svm = recall_score(y_test, y_pred_svm)
results.append(("SVM", "-", recall_svm, end - start))



# 4
res_df = pd.DataFrame(results, columns=["Model", "Params", "Recall (Attack)",
"Train Time (s)"])
```

```
print(res_df)
```

```
           Model Params  Recall (Attack)  Train Time (s)
0          k-NN    k=1         0.998748        0.000761
1          k-NN    k=3         0.995617        0.000922
2          k-NN    k=5         0.995617        0.000972
3          k-NN    k=7         0.995617        0.001001
4          k-NN    k=9         0.995617        0.000926
5  Decision Tree      -         0.998748        0.009533
6           SVM       -         0.998121        0.063806
```

5. Сделайте вывод о том, какая модель эффективнее для обнаружения вторжений.

```
# 5 best model
best_model = res_df.loc[res_df['Recall (Attack)'].idxmax()]
print("\nBest model:")
print(best_model)
```

```
Best model:
Model                 k-NN
Params                 k=1
Recall (Attack)    0.998748
Train Time (s)     0.000761
Name: 0, dtype: object
```

Вывод: на основании проведённого эксперимента можно сделать вывод, что наиболее эффективной моделью для обнаружения сетевых вторжений является k-NN при k=1. Эта модель показала наивысшее значение Recall (0.998748), при минимальном времени ее обучения (0.000761 сек).