Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Отчёт по лабораторной работе №5 по дисциплине "Методы машинного обучения"

Тема Байесовский классификатор «Ирисов Фишера»

Студент Варламова Е. А.

Группа ИУ7-23М

Оценка (баллы) _____

Преподаватели Солодовников Владимир Игоревич

Москва — 2024 г.

СОДЕРЖАНИЕ

1 | Теоретическая часть

В 1936 году Рональд Фишер опубликовал статью, в которой он представил набор данных об ирисах, состоящий из трех различных видов: Iris setosa, Iris versicolor и Iris virginica. Для каждого вида ириса были измерены четыре характеристики: длина чашелистика, ширина чашелистика, длина лепестка и ширина лепестка.

Фишер использовал этот набор данных для демонстрации метода дискриминантного анализа, который позволяет разделить объекты на классы на основе их признаков. Он показал, что можно эффективно классифицировать ирисы по видам, используя комбинацию этих четырех признаков.

Ирисы Фишера стали одним из самых популярных наборов данных в машинном обучении и статистике. Они часто используются для тестирования алгоритмов классификации и кластеризации, а также для демонстрации методов визуализации данных.

Целью данной лабораторной работы является применение байесовского классификатора решения задачи классификации ирисов Фишера. Для этого необходимо решить следующие задачи:

— формализовать задачу;

— описать алгоритм байесовского классификатора;

— привести особенности реализации ПО, решающего поставленную задачу;

— оценить точность, полноту, F-меру. Построить матрицу ошибоки.

1.1 Постановка задачи

Построить классификатор «Ирисов Фишера» с использованием байесовского подхода. В ходе выполнения работы:

1. Осуществить исследование и подготовку исходных данных.
2. Построить гистограммы распределения значений для каждого признака и для каждого класса.

3. Произвести визуализацию проекций классов на плоскости, где по осям отложены различные комбинации пар признаков.

4. Построить матрицы корреляций между различными признаками, как для всей выборки в целом, так и для отдельных классов.

5. Построить классификатор с использованием байесовского подхода.

6. Оценить точность, полноту, F-меру. Построить матрицу ошибок.

1.2 Алгоритм наивного байесовского классификатора

Классификация с использованием наивного байесовского классификатора основана на применении теоремы Байеса. Пусть дан набор объектов $X = (x_1, x_2, ..., x_n)$ и метки классов $Y = (y_1, y_2, ..., y_m)$. Наивный байесовский классификатор предполагает, что все признаки объектов независимы друг от друга при условии класса. Таким образом, вероятность принадлежности объекта x к классу y может быть вычислена по формуле:

$$P(y|x) = P(x|y)P(y)/P(x) \tag{1.1}$$

где

1. $P(y|x)$ - вероятность принадлежности объекта x к классу y;

2. $P(x|y)$ - вероятность объекта x при условии класса y;

3. $P(y)$ - априорная вероятность класса y;

4. $P(x)$- вероятность объекта x.

Для наивного байесовского классификатора используются различные модели для оценки вероятностей $P(x|y)$ и $P(y)$. Например, для категориальных данных можно использовать модель мультиномиального распределения, а для непрерывных данных - модель нормального распределения.

На практике для классификации объекта выбирается класс с наибольшей вероятностью $P(y|x)$:

$$y' = max_y \in Y P(y|x) \tag{1.2}$$

где y' - предсказанная метка класса для объекта x.

Таким образом, наивный байесовский классификатор является простым и эффективным методом классификации, основанным на принципах вероятностного вывода.

2 | Практическая часть

2.1 Выбор средств разработки

В качестве языка программирования был использован язык Python, поскольку этот язык кроссплатформенный и для него разработано огромное количество библиотек и модулей, решающих разнообразные задачи.

В частности, имеются библиотеки, включающие в себя алгоритм наивного байесовского классификатора в библиотеке [1].

2.2 Исследование ПО

В листинге 2.1 представлен код, решающий задачу классификации ирисов.

Листинг 2.1: Код классификации ирисов

```
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.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, precision_score, recall_score,
    f1_score, confusion_matrix

data = pd.read_csv('data.csv')

print(data.describe())
sns.pairplot(data, hue='iris_type', palette='husl')
plt.savefig("pairs.png")
plt.clf()

correlation_matrix = data.corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('
    ')
plt.savefig("whole_matrix.png")
plt.clf()

for iris_class in data['iris_type'].unique():
```

```
    subset = data[data['iris_type'] == iris_class]
    correlation_matrix = subset.corr()
    sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
    plt.title(f'                                          {iris_class}')
    plt.savefig(f"matrix_{iris_class}.png")
    plt.clf()


X = data.drop('iris_type', axis=1)
y = data['iris_type']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4,
    random_state=42)

model = GaussianNB()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='weighted')
recall = recall_score(y_test, y_pred, average='weighted')
f1 = f1_score(y_test, y_pred, average='weighted')
conf_matrix = confusion_matrix(y_test, y_pred)



sns.heatmap(conf_matrix, annot=True)
plt.title(f'acc={accuracy:.2f}, prec={precision:.2f}, Recall={recall:.2f},
    f1={f1:.2f}')
plt.savefig(f"matrix_errors.png")
plt.clf()
```

С помощью разработанного ПО были построены гистограммы распределения значений для каждого признака и для каждого класса (по диагонали), а также произведена визуализация проекций классов на плоскости, где по осям отложены различные комбинации пар признаков на рисунке 2.1.

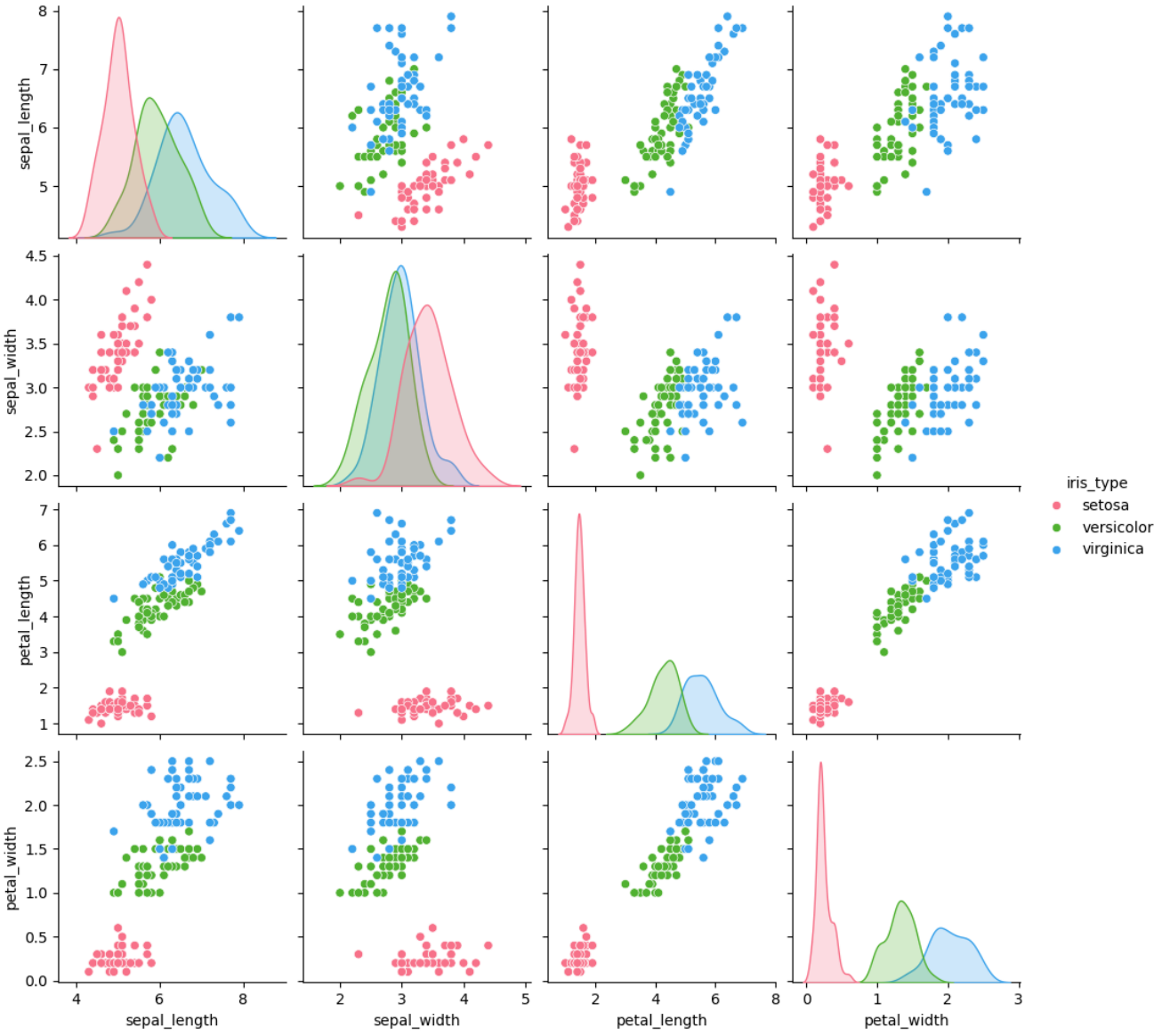

Рис. 2.1: Гистограммы распределения и проекции классов для пар признаков

По рисунку видно, что распределения всех признаков классов близки к нормальным, при этом с разными параметрами закона, что говорит о возможности разделения классов (например, если у одного класса среднее значение признака выше, чем у другого класса, это может быть важным признаком для разделения классов). Аналогично, по проекциям классов на плоскости для пар признаков можно сделать вывод, что данные хорошо разделяются на классы.

Также были построены матрицы корреляций между различными признаками на рисунке 2.2, как для всей выборки в целом, так и для отдельных классов 2.3-2.5.

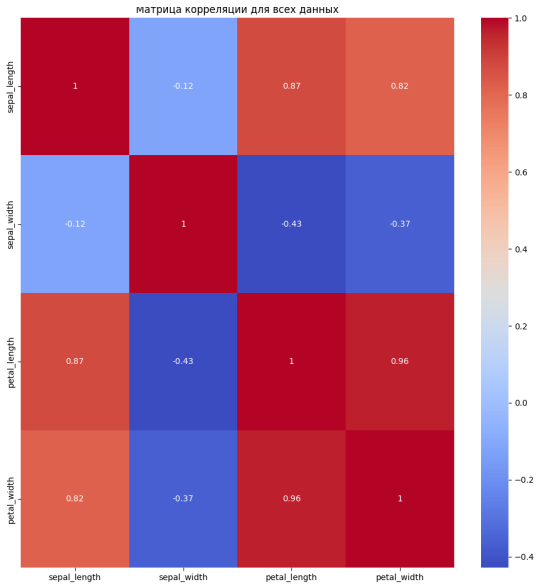

Рис. 2.2: матрица корреляций для всей выборки

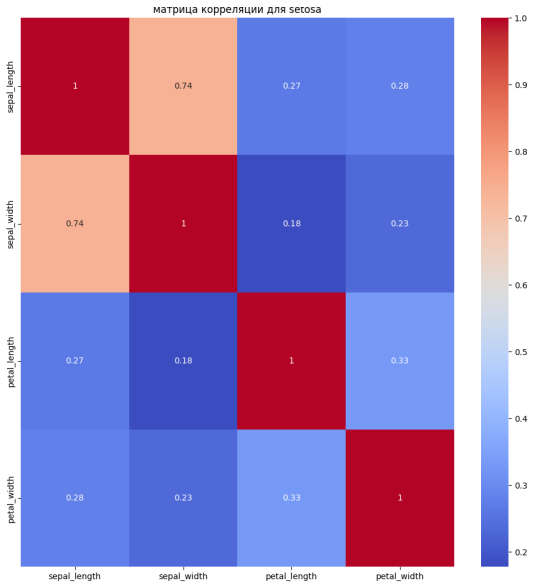

Рис. 2.3: матрица корреляций для setoza

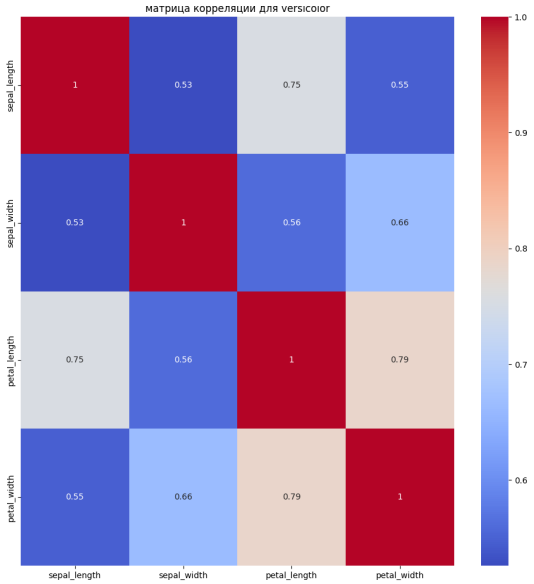

Рис. 2.4: матрица корреляций для versicolor

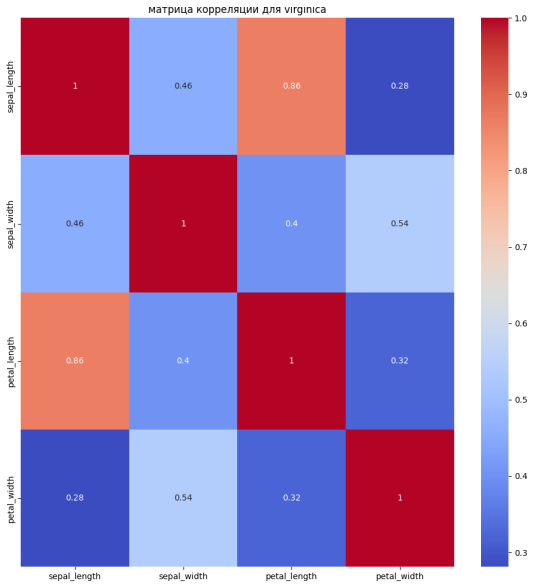

Рис. 2.5: матрица корреляций для virginica

Видно, что в большинстве классов наблюдается слабая корреляция между признаками.

Также была оценена точность, полнота, F-мера и построена матрица ошибки после классификации.

При разделении данных в пропорции 6:4 (обучающая:тестовая) были получены следующие результаты (рисунок 2.6).

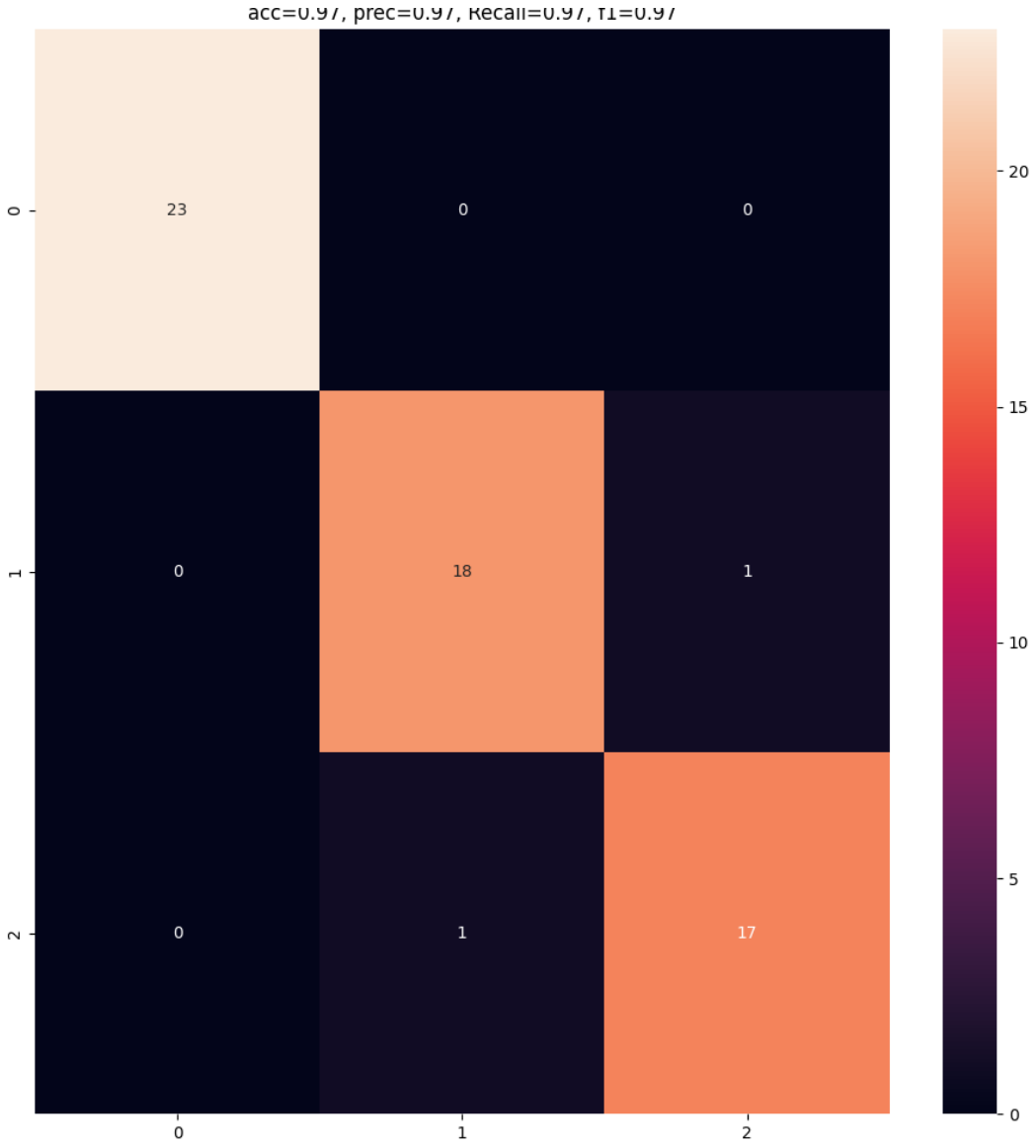

Рис. 2.6: матрица ошибок

— Accuracy: 0.97;

— Precision: 0.97;

— Recall: 0.97;

— F-мера: 0.97 .

При разделении данных в пропорции 8:2 (обучающая:тестовая) были получены следующие результаты (рисунок 2.7).

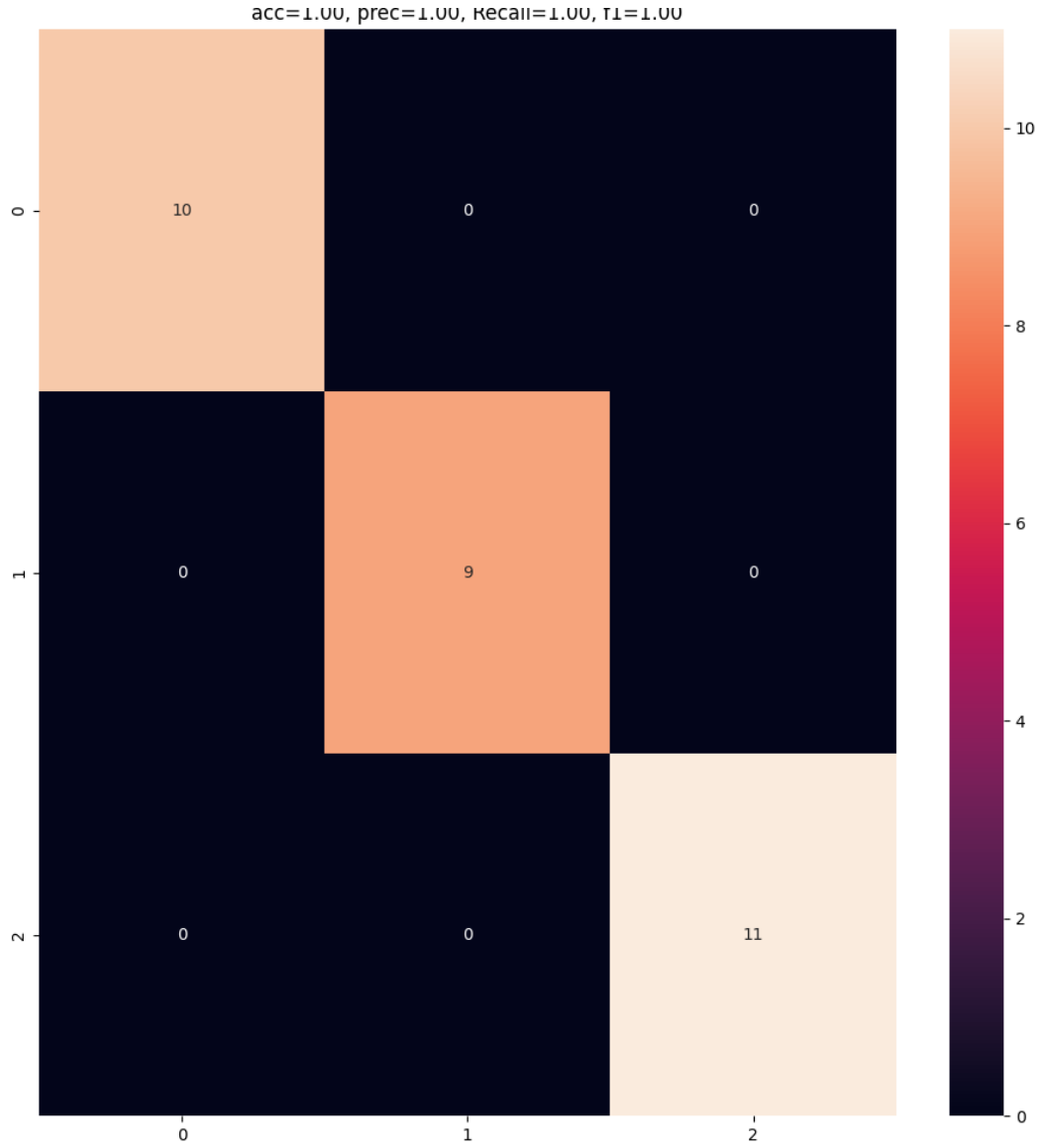

Рис. 2.7: матрица ошибок

— Accuracy: 1;

— Precision: 1;

— Recall: 1;

— F-мера: 1 .

Таким образом, классификация была проведена абсолютно точно при разделении 8:2 и с ошибкой в 3% при разделении 6:4.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Virtanen P.*, *Gommers R.*, *Oliphant T. E.* SciPy: Fundamental Algorithms for Scientific Computing in Python. — 2020. — DOI: 10.1038/s41592-019-0686-2.