**Трофимцова Е. Е., ИВТ-1, проектно-технологическая практика, задание 2.2**

**Развертывание среды разработки Julia и решение одной задачи в Jupyter Notebook**

Задача

**Проект 1: Классификация цветов Ирисов**

В этом проекте вы будете использовать данные о цветах ирисов для построения модели, которая сможет классифицировать цветок в один из трёх видов (Setosa, Versicolor, Virginica) на основе его характеристик (длина и ширина чашелистиков и лепестков).

**Основные шаги:**

* Загрузить набор данных о цветах ирисов (Iris dataset).
* Разделить данные на обучающую и тестовую выборки.
* Построить модель классификации с использованием метода ближайших соседей (K-Nearest Neighbors, KNN).
* Оценить точность модели на тестовой выборке.

Решение

В соответствии с указаниями из предоставленной презентации

* с официального сайта был скачан, установлен и добавлен в PATH язык научного программирования Julia;
* в командной строке Julia был добавлен пакет IJulia (add IJulia);
* по инструкции с сайта Visual Studio Code было установлено соответствующее расширение для этой среды разработки;
* в папке проекта создан файл с расширением .ipynb и выбрано ядро “Julia release channel”.

Далее были изучены статьи о том, в чем вообще заключается задача об Ирисах Фишера, выяснено, что задачу можно решить следующими методами: логистическая регрессия, метод k-ближайших соседей (kNN), дерево решений и SVM (метод опорных векторов). Как и было рекомендовано в задании, я приступила к подробному изучению метода k-ближайших соседей. Он заключается в том, что определяет, к каким классам принадлежат ближайшие к тестовому образцу k экземпляров, и присваивает тестовый образец к тому, классу, который встречается чаще [11].

Был скачан стандартный для этой задачи датасет, включающий 150 экземпляров, в формате .csv [8]. Были добавлены пакеты CSV для работы с файлами этого формата и DataFrames для удобной работы с таблицами. Для проверки были выведены первые 5 строк таблицы датасета.

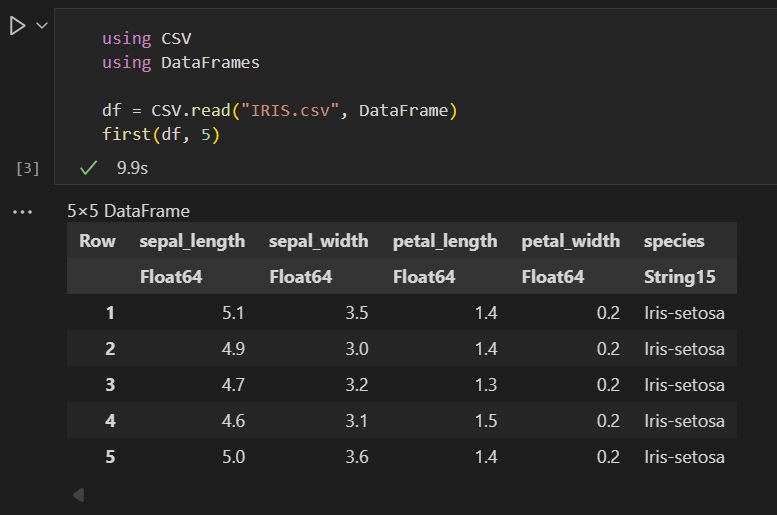


Рисунок – Чтение и вывод таблицы данных

Если построить графики зависимости ширины чашелистика от его длины и ширины лепестка от его длины, можно заметить, что по лепесткам виды ирисов отделяются друг от друга намного четче, чем по чашелистикам. Графики были построены с использованием пакета StatsPlots, который позволяет строить графики на данных из таблиц [4].

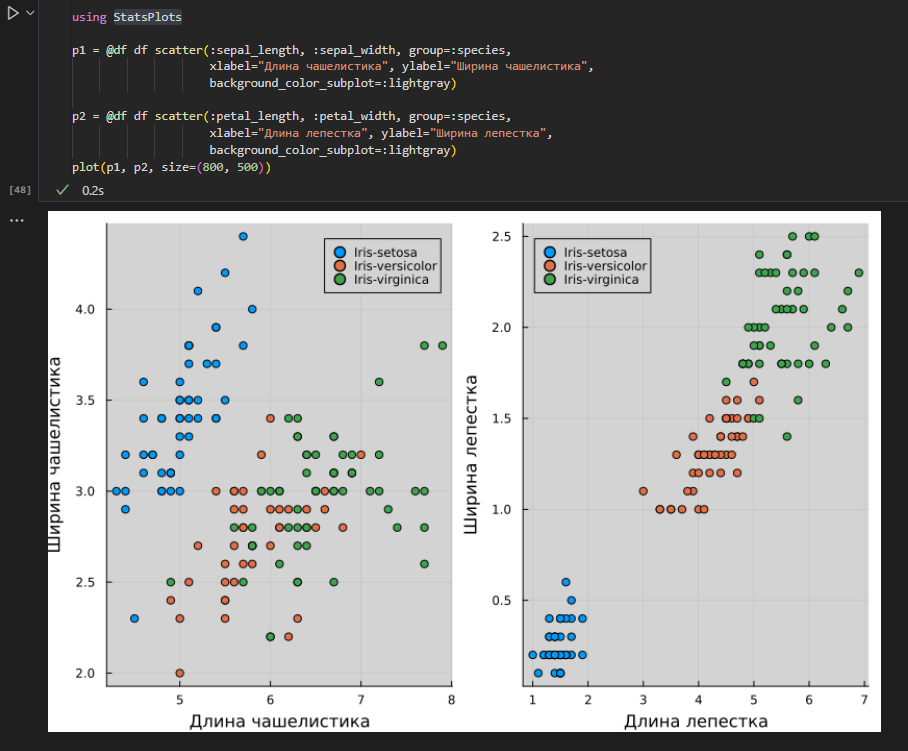


Рисунок – Построение графиков зависимости ширины чашелистика от его длины и ширины лепестка от его длины

Для оценки качества работы метода k-ближайших соседей при различных случайных разбиениях выборки был реализован цикл повторных экспериментов. Сначала был приведён столбец меток классов *species* к категориальному типу при помощи функции coerce!, чтобы модель каждый отличающийся текст распознавала как новый класс и работала корректно [10]. Далее таблица была разделена на матрицу признаков *X*, включающую четыре числовых параметра каждого экземпляра ириса, и вектор меток *y*, содержащий виды ирисов (*setosa*, *versicolor*, *virginica*) [11].

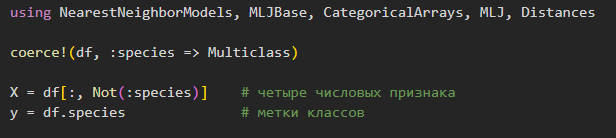


Рисунок – Формирование матрицы признаков и вектора меток классов

Теперь нужно разделить датасет на обучающую и тестовую выборки. После изучения нескольких статей, было замечено, что обычно их разделяют в соотношении 70:30 или 80:20. Разделим наши данные в соотношении 70:30. Так как в таблице виды ирисов идут по порядку (сначала 50 строк *setosa*, затем 50 строк *versicolor* и 50 строк *virginica*), сначала нужно их перемешать. Для разделения на обучающую и тестовую выборки использовалась функция partition() из пакета MLJBase [9]. На каждой итерации данные случайным образом перемешивались и делились в соотношении 70:30. После формирования индексов обучающей и тестовой выборок таблица признаков и вектор меток были разделены на соответствующие части. В переменные Xtrain и ytrain сохранились признаки и метки для обучения модели, а в Xtest и ytest – данные для проверки её качества. Такое разделение позволяет обучить алгоритм на одной части выборки и объективно оценить точность на другой, ранее не использованной.

Изображение выглядит как текст, снимок экрана, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок – Разделение данных на обучающую и тестовую выборки

Для корректной работы метода k-ближайших соседей использовалась последовательность из двух шагов, называемая конвейером (*pipeline*): стандартизация признаков (Standardizer()) и последующая классификация с помощью алгоритма ближайших соседей (KNNClassifier). Стандартизация необходима, чтобы привести все признаки к единому масштабу, так как метод kNN чувствителен к различию в диапазонах значений. Такой подход позволяет автоматически применять одинаковое преобразование данных как на этапе обучения, так и на этапе тестирования, что делает процесс более надёжным и воспроизводимым. В качестве метрики расстояния в параметрах классификатора был выбран алгоритм Евклида (Euclidean()). Оператор |> означает «передать результат работы одного шага следующему», т. е. после обработки данных стандартизатором, передать их классификатору.

Изображение выглядит как текст, Шрифт, снимок экрана

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок – Конвейер: стандартизация и классификация

Для каждого из 100 запусков происходило обучение модели на тренировочных данных и предсказание меток на тестовой выборке. После этого вычислялась точность (accuracy), и значение точности каждой итерации записывалось в список accs. Сначала создаем «машину» – объект machine, в который передаем наш пайплайн (pipe), состоящий из стандартизатора и классификатора, и матрицу свойств с метками классов обучающей выборки (Xtrain, ytrain). Далее запускаем процесс обучения функцией fit!. После обучения можно переходить к «предсказаниям». Для этого используется функция predict\_mode, которая возвращает итоговые метки классов для объектов тестовой выборки (Xtest). Переменная y\_pred обозначает предсказанные моделью значения, в отличие от исходных правильных меток ytest. И наконец, чтобы оценить работу модели, рассчитывается точность (accuracy) – доля верно предсказанных классов по сравнению с реальными ответами в тестовой выборке, и текущая точность добавляется в список для дальнейшего расчета средней точности. И по завершении цикла вычислялось среднее значение точности по всем 100 экспериментам.

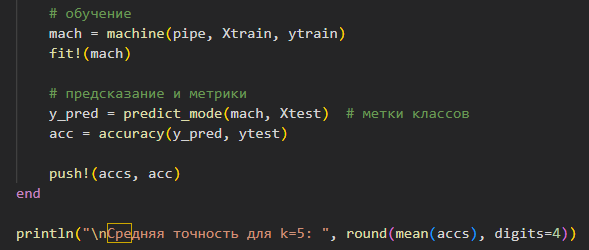


Рисунок – Обучение и оценка модели

Сравним результаты, полученные для значений k, равных 3, 4, 5:

* Средняя точность для k=5: 0.9538
* Средняя точность для k=4: 0.9464
* Средняя точность для k=3: 0.9473

Видно, что результаты отличаются некритично, в пределах сотых долей, однако лучше себя показывает модель с k=5.

Вывод

В процессе выполнения данного задания я познакомилась с основами машинного обучения на примере классификации ирисов Фишера, а также с языком научного программирования Julia. Я на начальном уровне освоила работу с таблицами данных и использование пакетов MLJ и NearestNeighborModels для построения и оценки модели методом k-ближайших соседей. В результате был получен практический опыт применения Julia для решения задач анализа данных.

**Список источников**

* 1. Кодинг и тестирование kNN в Julia. — Текст : электронный // Хабр : [сайт]. — URL: <https://habr.com/ru/articles/417967/> (дата обращения: 07.09.2025).
  2. Нейронные сети для начинающих. Решение задачи классификации Ирисов Фишера. — Текст : электронный // Хабр : [сайт]. — URL: <https://habr.com/ru/companies/ruvds/articles/679988/?ysclid=mfmo2rwi8u118150286> (дата обращения: 07.09.2025).
  3. Шитиков, В. К. Классификация, регрессия и другие алгоритмы Data Mining с использованием R / В. К. Шитиков. — Текст : электронный // GitHub Pages : [сайт]. — URL: <https://ranalytics.github.io/data-mining/071-Multiclass-Classification.html> (дата обращения: 07.09.2025).
  4. Breloff, T. Plots / T. Breloff. — Текст : электронный // JuliaPlots : [сайт]. — URL: <https://docs.juliaplots.org/stable/> (дата обращения: 07.09.2025).
  5. DataFrames.jl. — Текст : электронный // DataFrames : [сайт]. — URL: <https://dataframes.juliadata.org/stable/> (дата обращения: 07.09.2025).
  6. IRIS Flowers Classification Using Machine Learning - Analytics Vidhya. — Текст : электронный // Analytics Vidhya : [сайт]. — URL: <https://www.analyticsvidhya.com/blog/2022/06/iris-flowers-classification-using-machine-learning/> (дата обращения: 07.09.2025).
  7. Julia. Знакомство. — Текст : электронный // PVSM : [сайт]. — URL: <https://www.pvsm.ru/tutorial/293174> (дата обращения: 07.09.2025).
  8. Kharwal, A. Iris Flower Classification with Machine Learning | Aman Kharwal / A. Kharwal. — Текст : электронный // AmanXai : [сайт]. — URL: <https://amanxai.com/2021/10/17/iris-flower-classification-with-machine-learning/> (дата обращения: 07.09.2025).
  9. MLJ. — Текст : электронный // JuliaAI : [сайт]. — URL: <https://juliaai.github.io/MLJ.jl/dev/> (дата обращения: 07.09.2025).
  10. NearestNeighborModels.jl. — Текст : электронный // GitHub Pages : [сайт]. — URL: <https://juliaai.github.io/NearestNeighborModels.jl/dev/> (дата обращения: 07.09.2025).
  11. Phystech@DataScience — Занятие 2. Метод k ближайших соседей. — Текст : электронный // GitLab : [сайт]. — URL: <https://mipt-stats.gitlab.io/courses/ad_mipt/lec2_knn.html> (дата обращения: 07.09.2025).