Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

|  |
| --- |
| Институт информационных технологий и анализа данных |
| наименование института |

**ОТЧЕТ ПО ЛАБОРАТОРНЫМ РАБОТАМ**

по дисциплине «Визуализация данных»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент |  | КСм-21-1 |  |  |  | Мускатин И.А. |
|  |  | шифр группы |  | подпись |  | И.О. Фамилия |
| Проверил |  |  |  |  |  | Григорьев С.В. |
|  |  |  |  | подпись |  | И.О. Фамилия |

Отчет защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Иркутск 2022 г

**Содержание**

[Введение 3](#_Toc122035990)

[1 Технологии и методы визуализации данных 4](#_Toc122035991)

[1.1 Визуализация данных средствами Processing 4](#_Toc122035992)

[1.1.1 Описание данных 5](#_Toc122035993)

[1.1.2 Описание методов визуализации данных 5](#_Toc122035994)

[1.1.3 Результат визуализации данных 6](#_Toc122035995)

[1.2 Визуализация данных средствами D3.js 7](#_Toc122035996)

[1.2.1 Описание данных 8](#_Toc122035997)

[1.2.2 Описание методов визуализации данных 8](#_Toc122035998)

[1.2.3 Результат визуализации данных 10](#_Toc122035999)

[1.3 Визуализация данных средствами Unity 11](#_Toc122036000)

[1.3.1 Описание данных 11](#_Toc122036001)

[1.3.2 Описание методов визуализации данных 11](#_Toc122036002)

[1.3.3 Результат визуализации данных 12](#_Toc122036003)

[1.4 Визуализация данных средствами Python 14](#_Toc122036004)

[1.4.1 Описание данных 14](#_Toc122036005)

[1.4.2 Описание методов визуализации данных 14](#_Toc122036006)

[1.4.3 Результат визуализации данных 15](#_Toc122036007)

[2 Интерпретация полученных результатов 16](#_Toc122036008)

[Заключение 21](#_Toc122036009)

[Список используемых источников 22](#_Toc122036010)

# Введение

Основной целью выполнения лабораторных работ является ознакомление студентов с методами и инструментами визуализации данных, как с теоретической стороны, так и с практической стороны.

Основные задачи лабораторных работ: выполнение заданий нацеленных на получение знаний и навыков работы по визуализации данных с помощью таких инструментов визуализации как Processing, D3.js, Unity и Python.

Визуальная информация лучше воспринимается и позволяет быстро и эффективно донести до зрителя собственные мысли и идеи. Физиологически, восприятие визуальной информации является основной для человека. Есть многочисленные исследования, подтверждающие, что:

* 90% информации человек воспринимает через зрение;
* 70% сенсорных рецепторов находятся в глазах;
* около половины нейронов головного мозга человека задействованы в обработке визуальной информации;
* на 19% меньше при работе с визуальными данными используется когнитивная функция мозга, отвечающая за обработку и анализ информации;
* на 17% выше производительность человека, работающего с визуальной информацией;
* на 4,5% лучше воспоминаются подробные детали визуальной информации.

Таким образом обозначим основную цель данной работы как, изучение профильной литературы для получения новых знаний в данной области, а также выполнение практической части лабораторных для получения практических навыков в данной области.

Для достижения поставленной цели необходимо выполнить следующие задачи:

* Ознакомится с профильной литературой указанной на учебной ресурсе в курсе «Визуализация данных», также ознакомится с иной профильной литературой, найденной в сети «интернет», библиотеке и т.д.;
* Структурировать полученные знания;
* Выполнить практическую часть лабораторных, основываясь на знаниях, полученных из литературы;
* Создать отчёт на основе полученных данных.

# Технологии и методы визуализации данных

Визуализация данных — это наглядное представление массивов различной информации.

Можно выделить несколько типов визуализации:

* Обычное визуальное представление количественной информации в схематической форме. К этой группе можно отнести всем известные круговые и линейные диаграммы, гистограммы и спектрограммы, таблицы и различные точечные графики.
* Данные при визуализации могут быть преобразованы в форму, усиливающую восприятие и анализ этой информации. Например, карта и полярный график, временная линия и график с параллельными осями, диаграмма Эйлера.
* Концептуальная визуализация позволяет разрабатывать сложные концепции, идеи и планы с помощью концептуальных карт, диаграмм Ганта, графов с минимальным путем и других подобных видов диаграмм.
* Стратегическая визуализация переводит в визуальную форму различные данные об аспектах работы организаций. Это всевозможные диаграммы производительности, жизненного цикла и графики структур организаций.
* Графически организовать структурную информацию с помощью пирамид, деревьев и карт данных поможет метафорическая визуализация, ярким примером которой является карта метро.
* Комбинированная визуализация позволяет объединить несколько сложных графиков в одну схему, как в карте с прогнозом погоды.

## **Визуализация данных средствами Processing**

Processing связывает концепции программного обеспечения с принципами визуальной формы, движения и взаимодействия. Он объединяет язык программирования, среду разработки и методику преподавания в единую систему. Processing был создан для обучения основам компьютерного программирования в визуальном контексте, для использования в качестве программного «скетчбука» и в качестве производственного инструмента в определенных контекстах. Студенты, художники, профессионалы в области дизайна и исследователи используют его для обучения, создания прототипов и производства [1].

Он уходит своими корнями в язык под названием Design by Numbers, разработанный профессором Джоном Маэдой из Массачусетского технологического института, и фактически был создан двумя учениками Маэды, Беном Фраем и Кейси Реасом. В то время как большинство языков созданы для создания серьезных приложений, Processing, похоже, был создан для того, чтобы просто развлекаться. Этот язык использовался для создания различных произведений визуализации данных и инсталляции, но чаще всего вы просто видите, как люди «играют» с ним, создавая сложные и красивые картинки и анимацию. На самом деле, вы даже не создаете приложения для обработки; вы делаете наброски, которые помещаются в ваш альбом для рисования. Этот аспект языка привлек многих творческих программистов, которые стирают границы между программированием и искусством [2].

За эти годы Processing значительно вырос. Это развивающийся язык, дополняемый различными плагинами и вкладами преданного сообщества.

* + 1. Описание данных

В качестве данных для работы были взяты статистические данные о здоровье зубов в различных странах мира. Данные представляют из себя комбинацию из 2 файлов формата tsv: badTeeth и countryLocations.

|  |
| --- |
|  |
| Рисунок 1 – Данные из файлов badTeeth и countryLocations |

Файл badTeeth содержит данные о стране и числовой характеристике состояния здоровья зубов у граждан страны, в свою очередь countryLocations содержит данные о физическом расположении стран мира, т.е. их долготу и широту.

* + 1. Описание методов визуализации данных

В рамках лабораторной было принято решение о реализации так называемой картой пропорциональных символов, т.к. символами, используемыми в данной работе, будут круги/пузырьки, то можно сказать, что наша визуализация будет являться некоторой формой пузырьковой диаграммы/карты.

Для создания визуализации также потребуется использовать фоновое изображение контурной карты мира, для этого воспользуемся методом image(mapImage,0,0,width,height), где mapImage – это наше изображение карты, [0,0] – это координаты левого верхнего угла, а width и height ширина и высота созданного окна, что позволит полностью «залить» наше изображение на полный размер окна.

Для загрузки данных в наше приложение используем метод loadTable:

// Загружаем данные из таблицы

locationTable = loadTable("countryLocations.tsv","header,tsv");

dataTable = loadTable("badTeeth.tsv","header,tsv");

Для хранения данных воспользуемся методом dataTable:

// Получаем данные о плохих зубах по стране.

String countryName = dataTable.getString(row,"CountryName");

float badTeeth = dataTable.getFloat(row,"NumBadTeeth");

Далее для отображения данных на карте воспользуемся функцией map() для прямого преобразования из одного диапазона чисел в другой.

// Берем широту и долготу из данных о стране.

float latitude = countryRow.getFloat("latitude");

float longitude = countryRow.getFloat("longitude");

// Масштабируем широту и долготу, чтобы они соответствовали метрике окна.

float x = map(longitude,-180,180,0,width);

float y = map(latitude,-60,85,height,0);

ellipse(x,y,circleSize/scale\*1.45,circleSize/scale\*1.45);

Функция map() имеет пять параметров.

*map(value, low1, high1, low2, high2)*

Параметр value — это число для повторного сопоставления. Аналогично функции norm, параметры low1 и low2 являются минимальными и максимальными значениями текущего диапазона, здесь мы меняем стандартный диапазон широты и долготы. Параметры low2 и high2 являются минимальными и максимальными значениями для нового диапазона [1].

Также воспользуемся функцией ellipse() для создания наших «пузырьков» на карте в соответствии с показателями здоровья зубов.

Для создания некоторой интерактивности нашей визуализации воспользуемся методом MouseEvent, конкретно методом mouseWheel, который позволит нам приближать и отдалять наше изображение при прокрутке колесика мыши.

void mouseWheel(MouseEvent event) {

float zoomFactor = -event.getAmount()\*.1 + 1;

float newScale = constrain(scale \* zoomFactor, minScale, maxScale);

tx -= (mouseX/newScale - mouseX/scale);

ty -= (mouseY/newScale - mouseY/scale);

scale = newScale;

}

* + 1. Результат визуализации данных

В результате выполнения работы мы создали небольшой проект, который представляет из себя пузырьковую карту с возможностью изменения размеров изображения для более конкретного рассмотрения отображаемых данных.

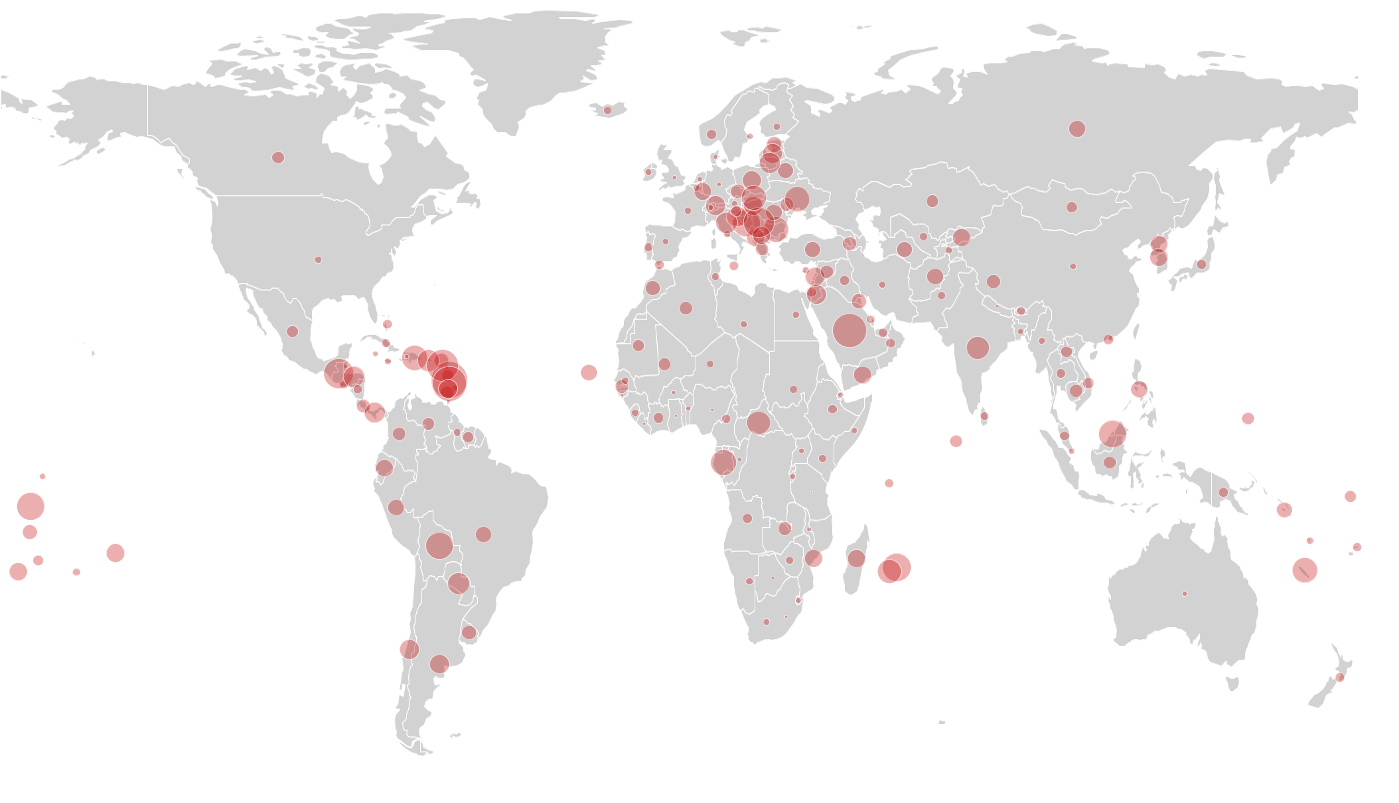


Рисунок 2 – Результат выполнения лабораторной работы №1



Рисунок 3 – Результат выполнения лабораторной работы №1

## **Визуализация данных средствами D3.js**

d3.js — также называемый D3 или d3.js — это библиотека JavaScript для создания визуализаций данных. Аббревиатура D3 ссылается на полное название инструмента — Документы, управляемые данными (Data-Driven Documents). Данные предоставлены вами, а документы — это веб-документы, то есть все, что может быть отображено веб-браузером, например HTML и SVG. D3 выполняет управление в том смысле, что он соединяет данные с документами.

Название также служит намеком на сеть технологий, лежащих в основе самого инструмента: W3, или Всемирная паутина, или, сегодня, просто «Web». Основным автором D3 является Майк Босток, хотя также есть несколько других преданных своему делу авторов. Проект полностью с открытым исходным кодом и свободно доступен на GitHub. D3 выпущен под лицензией BSD, поэтому вы можете использовать, изменять и адаптировать код для некоммерческого или коммерческого использования бесплатно.

* + 1. Описание данных

В качестве данных для работы были взяты данные о популярных туристических местах России у иностранцев посещающих РФ. Данные представляют из себя файл формата json.



Рисунок 4 – Часть данных для лабораторной работы №2

По аналогии с прошлой работой нас будут интересовать данные о широте и долготе наших данных. Так же нам понадобится данные для создания SVG изображения Земли, для этого воспользуемся данными из интернет ресурса – <https://konivatsara.com/resource/tpojson/land-110m.json>.

* + 1. Описание методов визуализации данных

В рамках лабораторной было принято решение о реализации 3D-картограммы, дополненной точеными символами.

Для выполнения поставленной задачи создадим методы: drawGlobe() – для создания SVG изображения 3D-глобуса; drawGraticule() – для создания градусной сетки и drawMarkers() – для создания маркеров точек притяжения иностранных туристов в России.

Так как в работе много работы с геоданными для этого используем методы d3.geo\* для дальнейшей реализации рабочего кода. В частности используем d3.geoOrthographic() что дает нам азимутальную ортографическую проекцию. Это полезно, когда мы хотим увидеть землю как 3D-объект.

Используем geoPath().projection() для указания нашей проекции Земли. Также используем d3.geoGraticule() – генератор геометрии для создания координатных сеток: регулярная координатная сетка меридианов и параллелей для показа искажения проекции. Будем использовать d3.geoDistance, который возвращает расстояние большой дуги в радианах между двумя точками a и b, для расположения наших точечных данных на карте. Каждая точка должна быть указана как двухэлементная антенная решетка [долгота, широта] в градусах. Это - сферический эквивалент path.measure, учитывая LineString двух точек.

Также добавим 2 типа интерактивности нашей карты:

1. Зум и перемещение карты:

var svg = d3.select('.globe')

.attr('width', width)

.attr('height', height)

.call(d3.zoom().on("zoom", function () {

svg.attr("transform", d3.event.transform)

}))

.append("g");

1. Переход на карты OSM при клике на карту:

var drag = d3.drag()

.on("end", geolinkGlobe);

svg.call(drag);

function geolinkGlobe() {

var gps = projection.invert(d3.mouse(this));

if (gps[0] >= 180 || gps[0] <= -180)

return;

if (gps[1] >= 90 || gps[1] <= -90)

return;

window.open("https://www.openstreetmap.org/#map=6/"+gps[1]+"/"+gps[0], "\_opentab");

Ниже представлена часть кода реализующая отрисовку глобуса – drawGlobe().

function drawGlobe() {

d3.queue()

.defer(d3.json, worldDataSource)

.defer(d3.json, locationDataSource)

.await((error, worldData, locationData) => {

svg.selectAll(".segment")

.data(topojson.feature(worldData, worldData.objects.land).features)

.enter().append("path")

.attr("class", "segment")

.attr("d", path)

.style("stroke", "#888")

.style("stroke-width", "1px")

.style("fill", (d, i) => '#e5e5e5')

.style("opacity", ".8");

locations = locationData;

drawMarkers();

});

}

* + 1. Результат визуализации данных

В результате выполнения работы была создана 3D-проекция земного шара с нанесенными на него данными в виде точечных элементов, которые символизируют собой места притяжения иностранных туристов в России, следуя данным полученным из визуализации можно утверждать что большинство туристов в нашей стране приезжает в места западной части страны, в частности много «точек» расположено на территории Калининградской области и Крыма, восточная часть нашей страны привлекает иностранных туристов гораздо меньше, либо доступная информация о туристических местах не достаточна, т.к. «точки интереса» расположены достаточно редко.



Рисунок 5 – Результат выполнения лабораторной работы №2

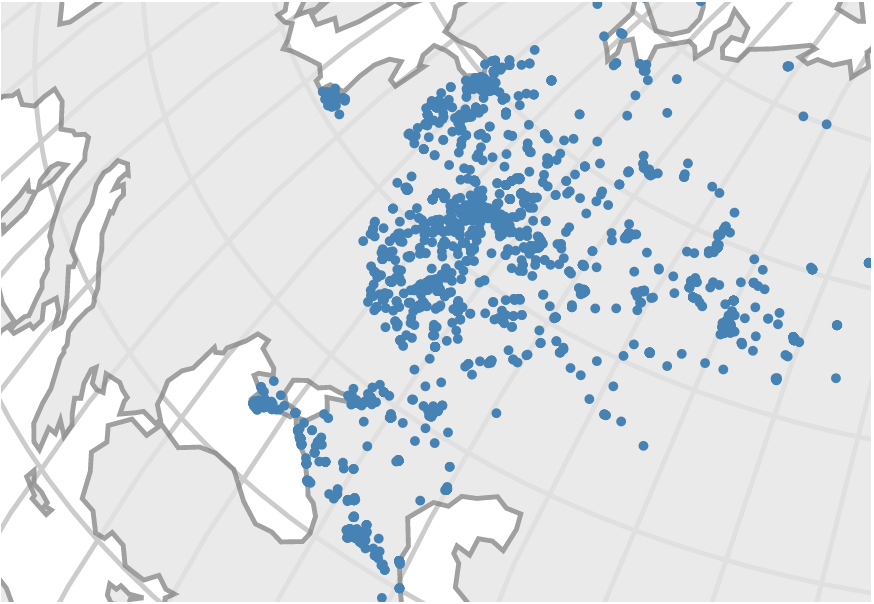


Рисунок 6 – Результат выполнения лабораторной работы №2

## **Визуализация данных средствами Unity**

Unity (или Unity 3D) — это игровой движок, созданный для создания игр. Он может создавать игры и другие возможности для Интернета, ПК, Android, iOS и т. д. Также он предназначен для проектов, связанных с технологиями виртуальной реальности, такими как Oculus Rift или HTC Vive. Сами проекты Unity, по сути, состоят из контента (активов), скриптов (программирования) и самого движка, который объединяет все вместе и заставляет его работать на вашей конкретной платформе.

* + 1. Описание данных

В качестве данных для работы был взят популярный набор данных Iris, используемый во многих обучающих материалах, как по визуализации данных, так и по обработке и анализу данных.

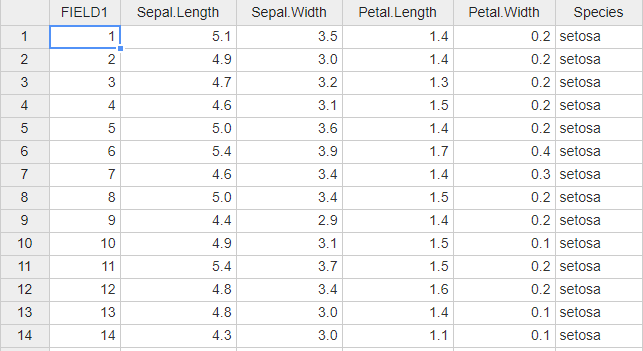


Рисунок 7 – Данные для лабораторной работы №3

Как можем видеть на рисунке данные представляют из себя набор характеристик чашечки ириса (Sepal) и характеристики лепестков ириса (Petal). В ходе работы попробуем добиться визуального представления этих данных с помощью Unity.

* + 1. Описание методов визуализации данных

Для работы с данными в Unity необходимо создать C# скрипт, который будет отвечать за всю логику нашего процесса визуализации. Для чтения нашего CSV файла создадим отдельный скрипт C# CSVReader, в котором стандартным для C# способом прочтем и распарсим наш документ.

Далее создадим скрипт DataPlotter, в котором и будет вся логика нашей работы.

// Цикл по списку точек  
        for (var i = 0; i < pointList.Count; i++)  
        {  
            // Получим значения в списке точек в i-й "строке", в названии "столбца" и нормализуем их  
            float x =  
                (System.Convert.ToSingle(pointList[i][xName]) - xMin)  
                / (xMax - xMin);  
            float y =  
                (System.Convert.ToSingle(pointList[i][yName]) - yMin)  
                / (yMax - yMin);  
            float z =  
                (System.Convert.ToSingle(pointList[i][zName]) - zMin)  
                / (zMax - zMin);  
            // Создать экземпляр как переменную gameobject, чтобы можно было манипулировать ей в цикле  
            GameObject dataPoint = Instantiate(  
                PointPrefab,  
                new Vector3(x, y, z)\* plotScale,  
                Quaternion.identity);  
  
            // Сделать дочерним объектом PointHolder, чтобы сохранить точки внутри контейнера в иерархии

            dataPoint.transform.parent = PointHolder.transform;  
            // Присваиваем исходные значения dataPointName  
            string dataPointName =  
                pointList[i][xName] + " "  
                + pointList[i][yName] + " "  
                + pointList[i][zName];  
            // Присваиваем имя prefab  
            dataPoint.transform.name = dataPointName;  
            // Получает цвет материала и устанавливает для него новый цвет  
            dataPoint.GetComponent<Renderer>().material.color =  
                new Color(x,y,z, 1.0f);  
        }

Выше изложен код формирования нашего 3D представления данных в пространстве на основании числовых характеристик ирисов из набора данных.

* + 1. Результат визуализации данных

В результате выполнения работы было создано 3D-приложение на движке Unity, позволяющая в реальном времени с любого удобного для вас ракурса провести визуальный анализ данных.

Например, можно провести визуальный анализ по выделению кластеров данных, так, на рисунке 8 можно решить, что исходные данные представляют из себя набор из 2 групп Ирисов, для наглядности деления на изображении проведена белая разграничивающая линия. На рисунке 9 уже видно, что различных групп (кластеров) у нашего набора данных можно выделить 3.

Таким образом можно утверждать, что визуализация данных с помощью Unity может предоставить возможность взглянуть на исследуемые данные под разными углами, не прибегая к написанию дополнительных строчек кода.

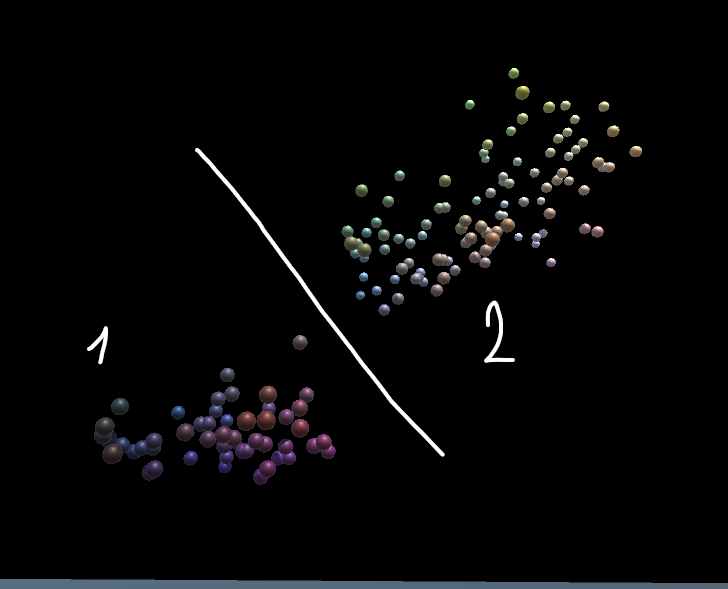


Рисунок 8 – Результат выполнения лабораторной работы №3

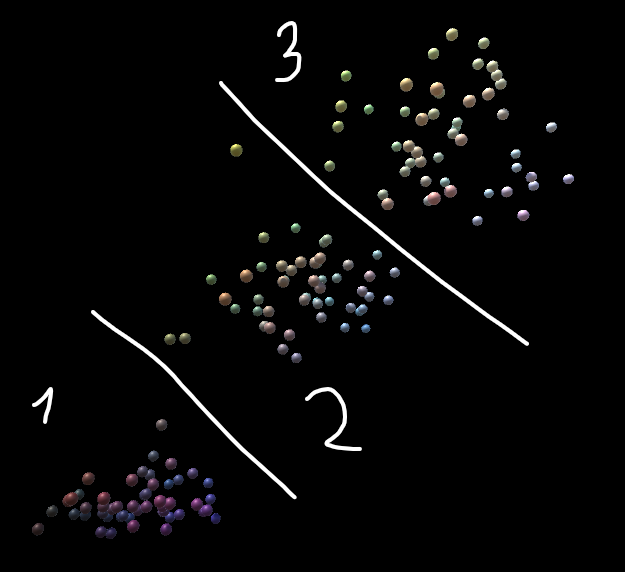


Рисунок 9 – Результат выполнения лабораторной работы №3

## **Визуализация данных средствами Python**

Python – высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ.

Jupyter Notebook — невероятно мощный инструмент для интерактивной разработки и представления проектов в области наук о данных. Проект Jupyter является преемником более раннего проекта IPython Notebook, который впервые был опубликован в качестве прототипа в 2010 году.

* + 1. Описание данных

В качестве данных для работы также возьмем набор данных Iris, используем аналитические возможности, предоставляемые разнообразными библиотеками Python, для анализа нашего набора данных и определения количества кластеров для деления данных.

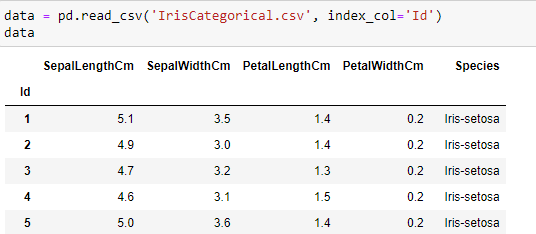


Рисунок 10 – Данные для лабораторной работы №4

* + 1. Описание методов визуализации данных

Для работы с данными в jupyter необходимо импортировать ряд библиотек Python, предназначенных для работы с данными, начиная от библиотек, позволяющих загружать данные в удобном для нас формате, заканчивая библиотеками, отвечающими за стиль оформления наших визуализаций.

Для этого воспользуемся следующими библиотеками:

* pandas — программная библиотека на языке Python для обработки и анализа данных (подключаем для работы с данными и их обработки);
* seaborn — это библиотека для создания статистических графиков на Python она основывается на matplotlib и тесно взаимодействует со структурами данных pandas;
* matplotlib – библиотека на языке программирования Python для визуализации данных двумерной и трёхмерной графикой;
* и другими.

Далее представлена часть кода, которая создает график на рисунке 11.

plt.scatter(data['SepalLengthCm'],data['SepalWidthCm'])

plt.xlim(3,10)

plt.ylim(1,5)

plt.show

* + 1. Результат визуализации данных

В результате выполнения работы были созданы разнообразные графические представления исходных данных, начиная от диаграмм рассеяния, представленной на рисунке 11, заканчивая кластерными картами и дендрограммами.

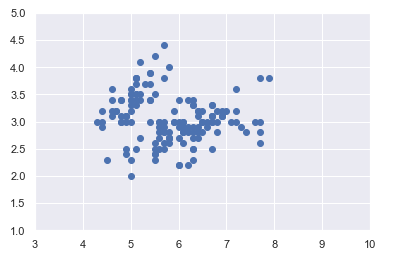


Рисунок 11 – Результат выполнения лабораторной работы №4

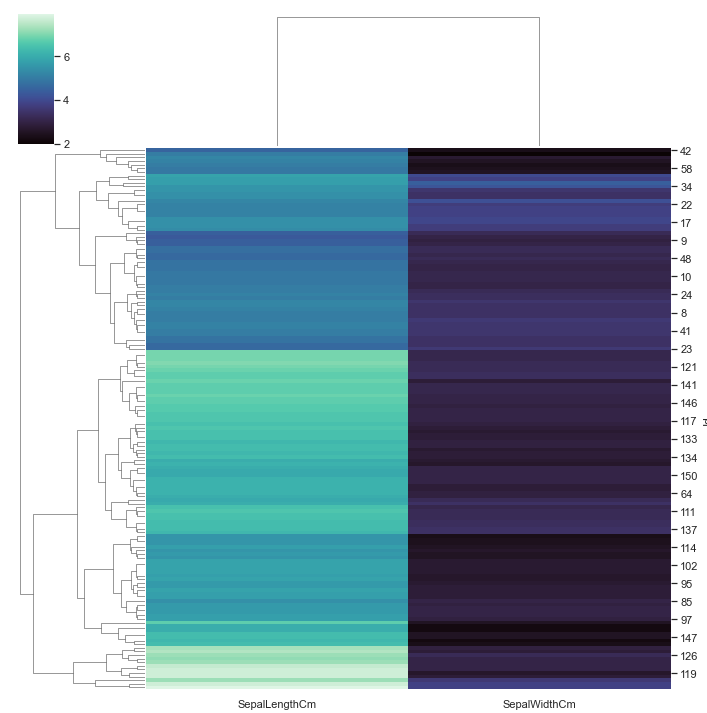


Рисунок 12 – Результат выполнения лабораторной работы №4

# Интерпретация полученных результатов

Интерпретацию полученных результатов будем рассматривать на примере лабораторной работы №4, реализованной на языке Python в jupyter. Для корректного описания полученных данных также укажем что в ходе лабораторной работы была поставлена цель нахождения корректного количества кластеров, на которое можно разделить исходные данные.

Также первым делом данные загружаются и сталкиваются с первым способов их визуализации, простым выводом таблицы на экран. Данные с рисунка 13 мы можем интерпретировать как то, что общий набор данных содержит в себе 150 элементов, для каждого элемента верно утверждение о том, что он имеет 4 качественные характеристики в виде длины и ширины чашечки, и длины и ширины лепестка, помимо этого можно утверждать что в исходном наборе присутствует как минимум 2 разных вида ириса – setosa и virginica.

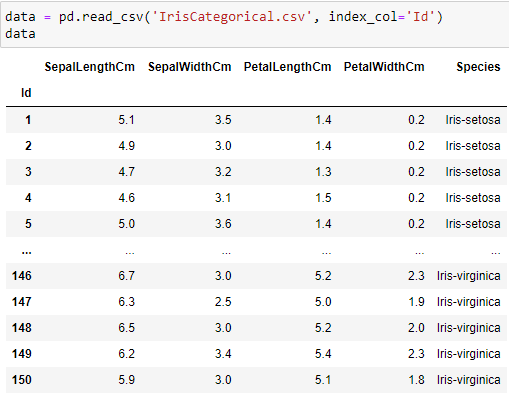


Рисунок 13 – Вывод на экран таблицы данных в jupyter

Этих данных не хватит для ответа на поставленный вопрос о количестве кластеров (групп) наших ирисов, таким образом мы обращаемся к следующей визуализации, а именно к диаграмме рассеяния, представленной в виде точечной диаграммы на рисунке 14. Исходя из данной диаграммы рассеяния можно уже делать предположения что как было выявлено выше, у нас присутствует только 2 группы (кластера) ирисов, т.к. ярко выражены 2 отдаленные друг от друга группы точек.

Для подтверждения нашей теории о том, что в нашем наборе данных имеется только 2 кластера данных построим кластерную карту с помощью библиотеки seaborn и её метода clustermap, то что у нас получилось можно наблюдать на рисунке 15.

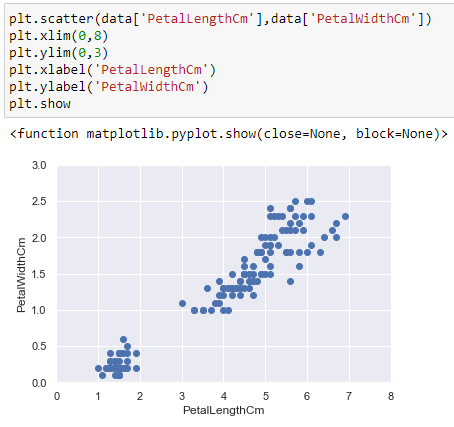


Рисунок 14 – Диаграмма рассеяния в jupyter

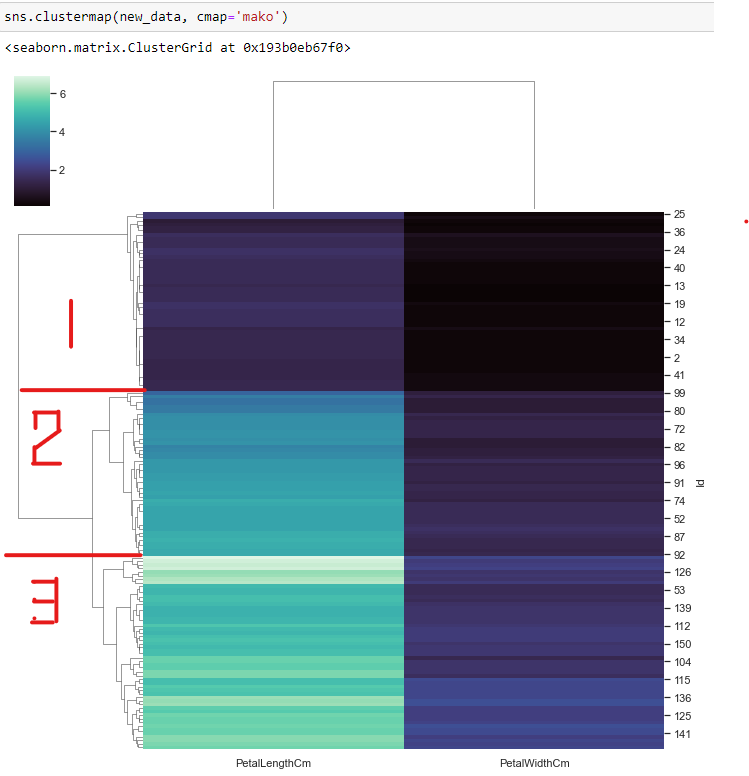


Рисунок 15 – Кластерная карта в jupyter

Как видно из рисунка 15, представляющего из себя своеобразную цветовую интерпретацию исходных данных, данные можно разделить на 3 кластера, т.к. данные образуют подобие цветовых отрезков, так, например верхние отрезки представлены чёрными и темно синими полосами с резким разграничением в районе 2 отрезка и переходом от тёмных цветов на более светлые синие оттенки с вкраплениями зеленых полос, который также резко обрывается в начале 3 отрезка, который начинается со светло-зеленой, почти белой полосы и состоит в большинстве своем из светло-зеленых полос. Такое же деление, но менее заметное можно проследить и во втором столбце, где изображены данные по ширине лепестков (PetalWidthCm). Либо можно расценивать ярко выраженный 1 отрезок за 1 кластер и за второй кластер объединение 2 и 3, т.к. сами отрезки имеют схожую цветовую палитру, в отличии от 1 отрезка.

Построим дендрограмму иерархической кластеризации.

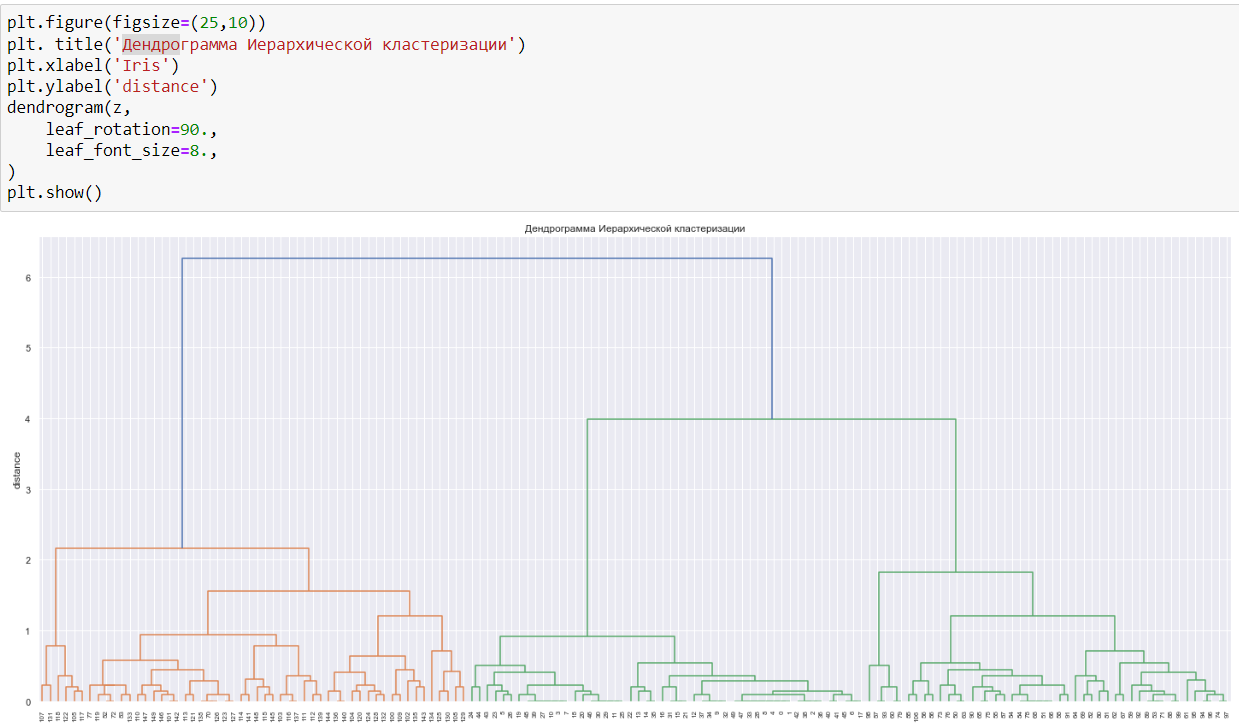


Рисунок 16 – Дендрограмма иерархической кластеризации полный размер

Т.к. дендрограмма на рисунке 16 не совсем информативна, то уменьшим её размер путем объединения данных до 5 возможных кластеров. Таким образом получаем дендрограмму

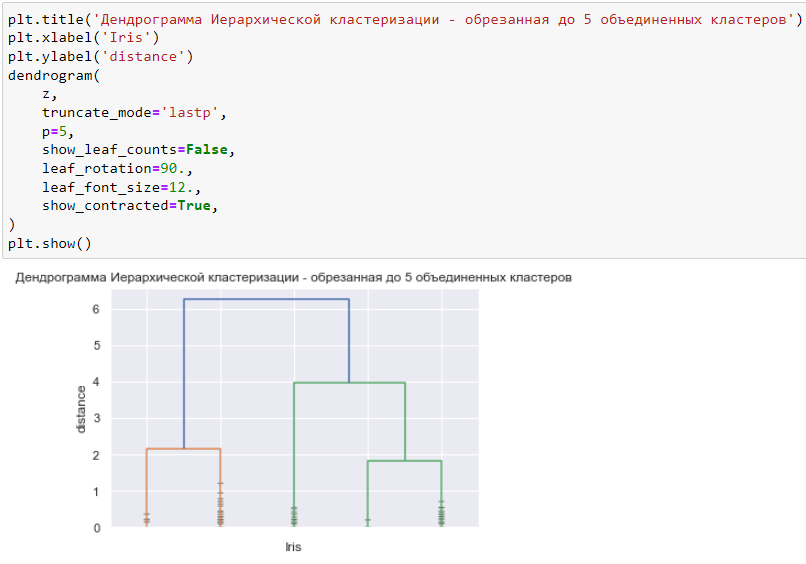


Рисунок 16 – Дендрограмма иерархической кластеризации обрезанная до 5 возможных кластеров

Также в анализе данных в вопросе кластеризации часто обращаются к Knee/Eblow анализу, который быстрее и удобнее всего представлять в формате точечной диаграммы с кривыми и маркерами, т.к. весь смысл данного анализа в нахождении переломов кривых, так называемых локтей и колен.

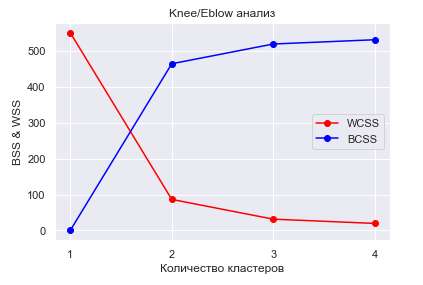


Рисунок 17 – Графическое представление Knee/Eblow анализа

Так как на графике Knee/Eblow анализа четко видно 2 сгиба в точках 2 и 3, то можно предположить что данные в нашем наборе могут быть поделены на 2 или 3 кластера.

Как можем видеть из рисунка 18, наши данные могут быть разбиты на 2 и 3 кластера представленным на нем образом.

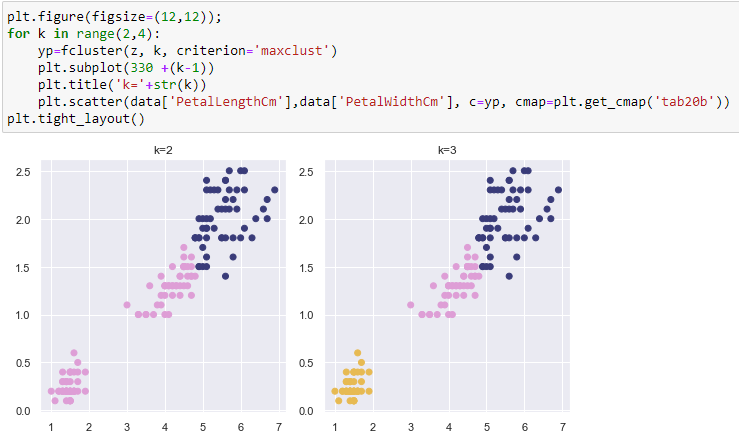


Рисунок 18 – Графическое представление возможного деления на кластеры исходных данных

Заключение

В ходе выполнения данных лабораторных работ был проведен обзор профильной литературы по теме «Визуализация данных».

В первой разделе был проведен анализ технологий и методов визуализации данных в Processing, D3.js, Unity и Python. Во втором разделе был точечно рассмотрен результат выполнения одной лабораторной работы и проведена интерпретация полученных визуализаций.

За время выполнения лабораторных работ были выполнены все поставленные задачи.

Данный курс помог получить много полезной и необходимой информации и навыков связанных с визуализацией данных.

Список используемых источников

1. Casey Reas, Ben Fry Processing: a programming handbook for visual designers and artists. Cambridge: The MIT Press, 2007.
2. Ira Greenberg Processing: Creative Coding and Computational Art. New York: SPRINGER, 2007.
3. Ben Fry Visualizing Data. Sebastopol: O’Reilly Media, 2008.
4. Scott Murray Interactive Data Visualization for the Web. Sebastopol: O’Reilly Media, 2013.
5. Data-Driven Documents URL: https://d3js.org (дата обращения: 15.12.2022).
6. D3 // SPEC-ZONE URL: SPEC-ZONE.RU (дата обращения: 15.12.2022).
7. Data Visualization with Python // geeksforgeeks URL: https://www.geeksforgeeks.org/data-visualization-with-python/ (дата обращения: 15.12.2022).
8. Визуализация матричных моделей // PyProg URL: https://pyprog.pro/sns/sns\_9\_corr\_and\_clust.html (дата обращения: 15.12.2022).
9. СТО ИРНИТУ 005-2015. Система менеджмента качества. Учебно-методическая деятельность. Оформление курсовых проектов (работ) и выпускных квалификационных работ технических специальностей.