МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ

ИНСТИТУТ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Лабораторные работы по дисциплине

«Введение в машинное обучение»

Выполнил:

Студент гр.09-305

Шарафеев Марат

Преподаватель:

Шустова Е.П.

Казань-2024

# **Задание №10. K-means (K-means++, K-means).**

**Выполните следующее задачи:**

1. Подберите свои данные для решения задачи классификации с помощью дерева решений. Для этого надо воспользоваться известными репозиториями данных (см., например, http://archive.ics.uci.edu/ml/index.php). К указанному ниже сроку закрепите эти данные в Teams через ответить в начатой мною беседе. Закрепите это означает, что надо привести краткое описание выбранных Вами данных и интернет-ссылку на эти данные. Перед поиском своих данных посмотрите в указанной выше беседе какие данные уже заняты другими студентами.
2. Установите необходимые внешние библиотеки.
3. Приведите скриншоты путей, в которые Вы установили Python, PyCharm и внешние библиотеки.
4. Импортируйте необходимые пакеты и классы:
5. Осуществите описание Вашего набора данных. Загрузите Ваши данные и распечатайте первые 10 наблюдений.
6. Подготовьте данные для работы. А именно: определите, есть ли в наборе данных пропущенные значения; если в данных присутствуют пропущенные значения, то либо удалите соответствующие наблюдения, либо рассчитайте их значение известным Вам методом; удалите выбросы, если они есть.
7. Рассчитайте и выведите матрицу корреляции для Ваших данных. Укажите какой признак Вы берете за переменную отклика, а какие за факторы. Обоснуйте свое решение.
8. Примените метод k средних для решения задачи клаcтеризации (в качестве исходных данных Вы можете использовать те же данные, для которых ранее решалась задача классификации, удалив из них столбец со значениями переменной отклика).
9. Если число признаков m = 2, визуализуйте данные (постройте облако точек, "раскрасив" точки в цвета, соответствующие номерам кластеров) и сравните полученные кластеры с облаками точек в соответствующей задаче классификации.
10. Исследуйте работу метода, варьируя значения параметра.
11. Проанализируйте полученные результаты, оформите отчёт.

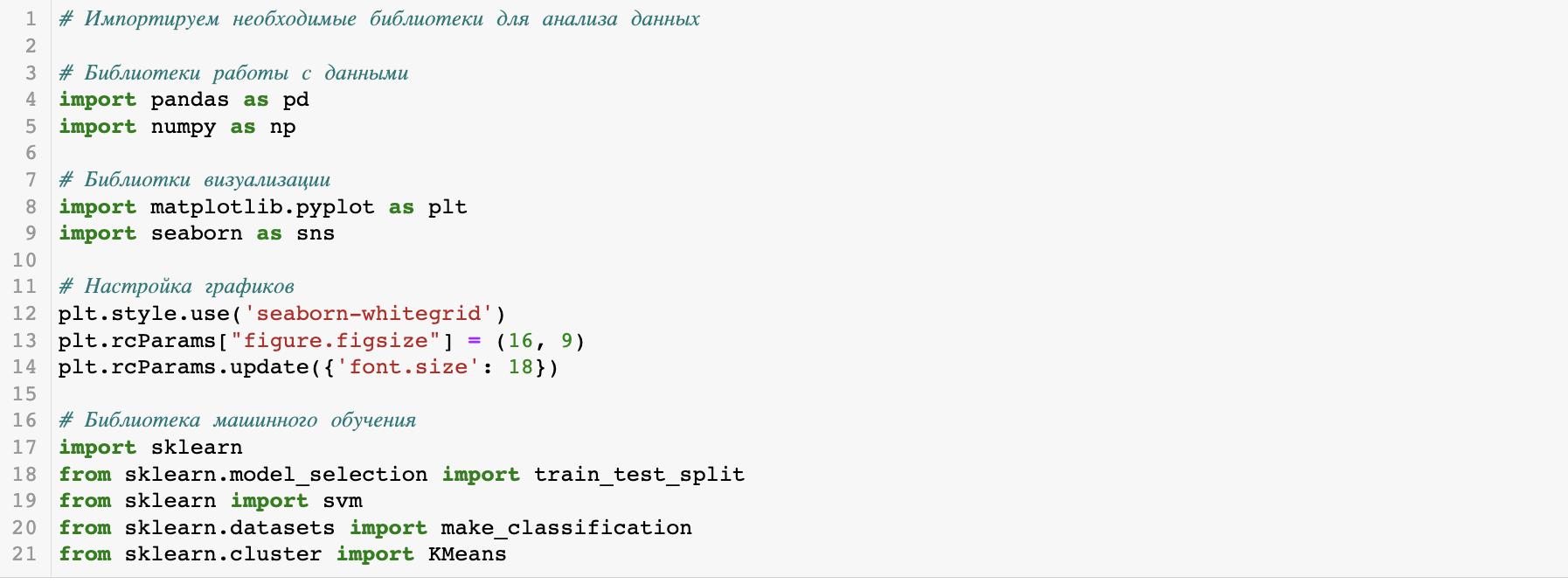
**Ход работы:**

1. Для задачи классификации были выбраны данные о предсказании возраста морского ушка (моллюск) по физическим измерениям. Автор набора данных утверждает, что процесс нахождения возраста моллюска весьма трудоемкий и по этой причине был выпущен датасет, который был призван для разработки модели, которая бы по внешним физическим показателем мог бы с достаточно большой точностью определить возраст ушка. Набор данных содержит 4177 экземпляров. С оригинал датасета можно получить по [ссылке](http://archive.ics.uci.edu/ml/datasets/Abalone).

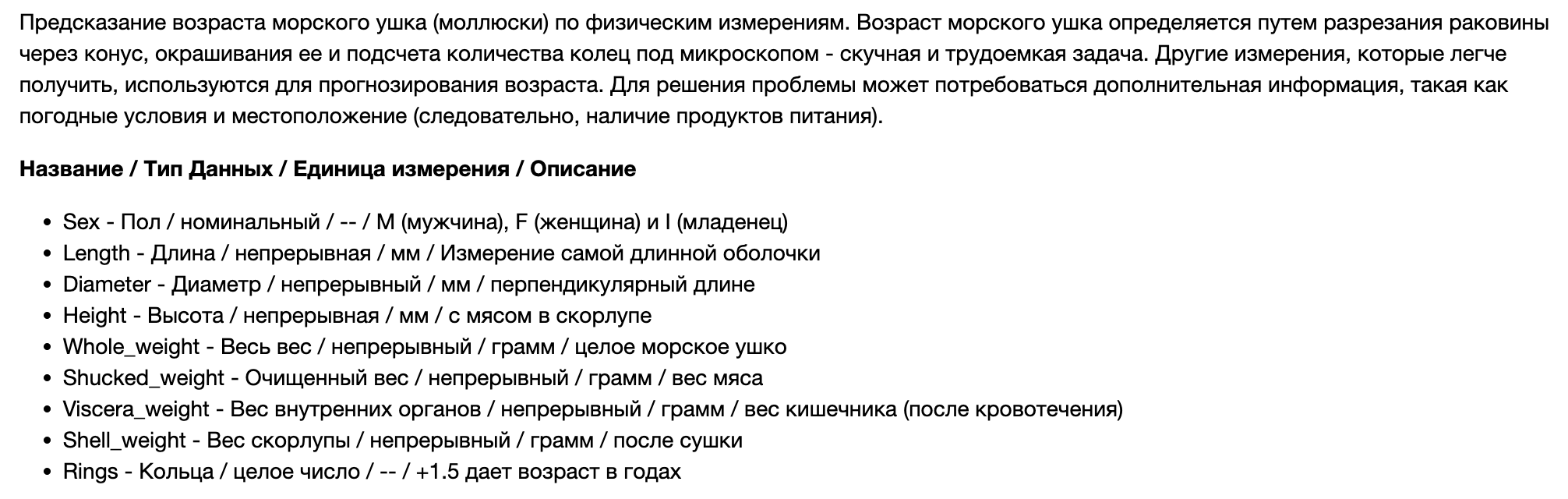
Данные содержат следующие характеристики:

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Тип данных** | **Ед. изм.** | **Описание** |
| Sex - Пол | номинативная | – | M (мужчина), F (женщина) и I (младенец) |
| Length - Длина | непрерывная | мм | Измерение самой длинной оболочки |
| Diameter - Диаметр | непрерывный | мм | перпендикулярный длине |
| Height - Высота | непрерывная | мм | с мясом в скорлупе |
| Whole\_weight - Весь вес | непрерывный | грамм | целое морское ушко |
| Shucked\_weight - Очищенный вес | непрерывный | грамм | вес мяса |
| Viscera\_weight - Вес внутренних органов | непрерывный | грамм | вес кишечника (после кровотечения) |
| Shell\_weight - Вес скорлупы | непрерывный | грамм | после сушки |
| Rings - Кольца | целое число | – | +1.5 дает возраст в годах |

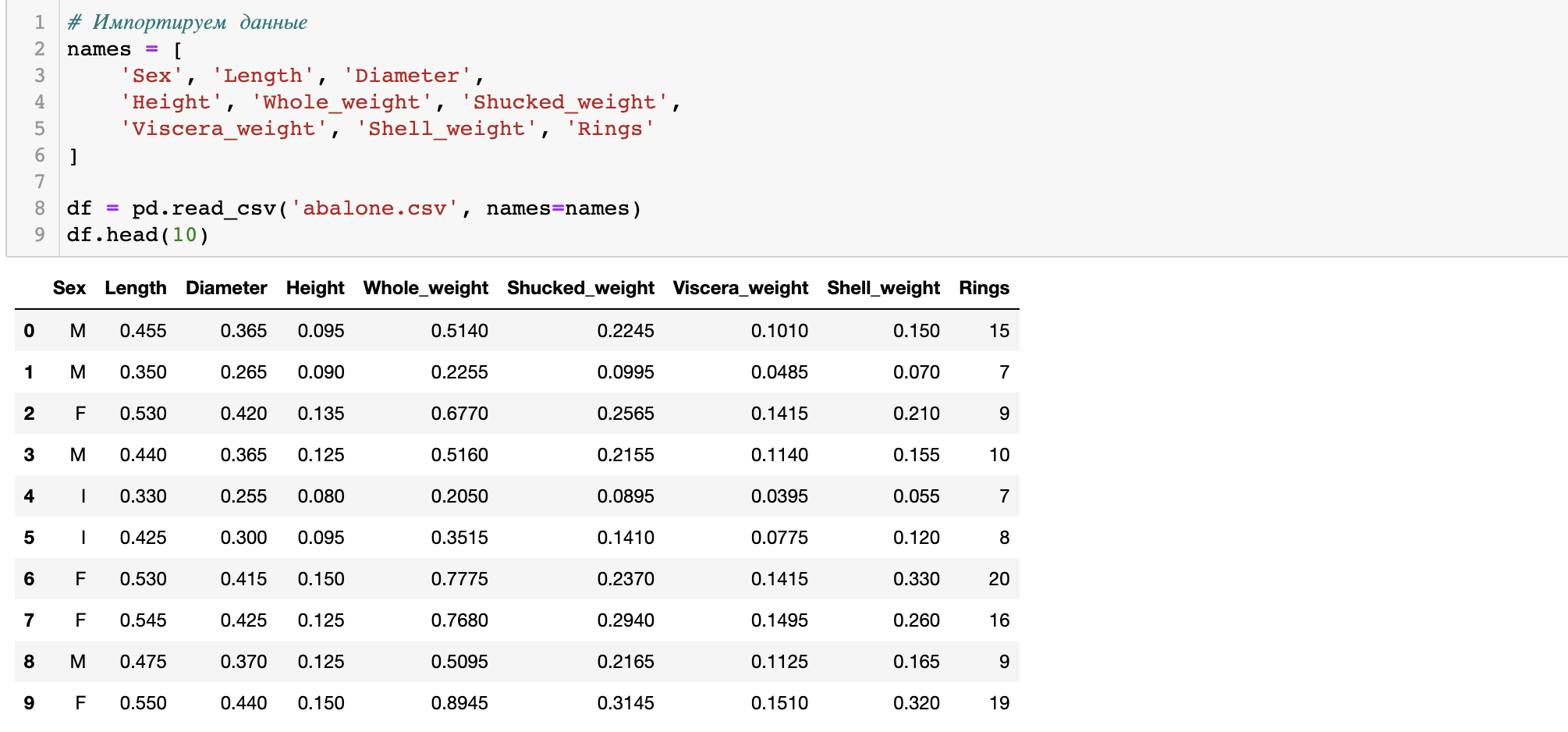
1. Импортируем все необходимые библиотеки:



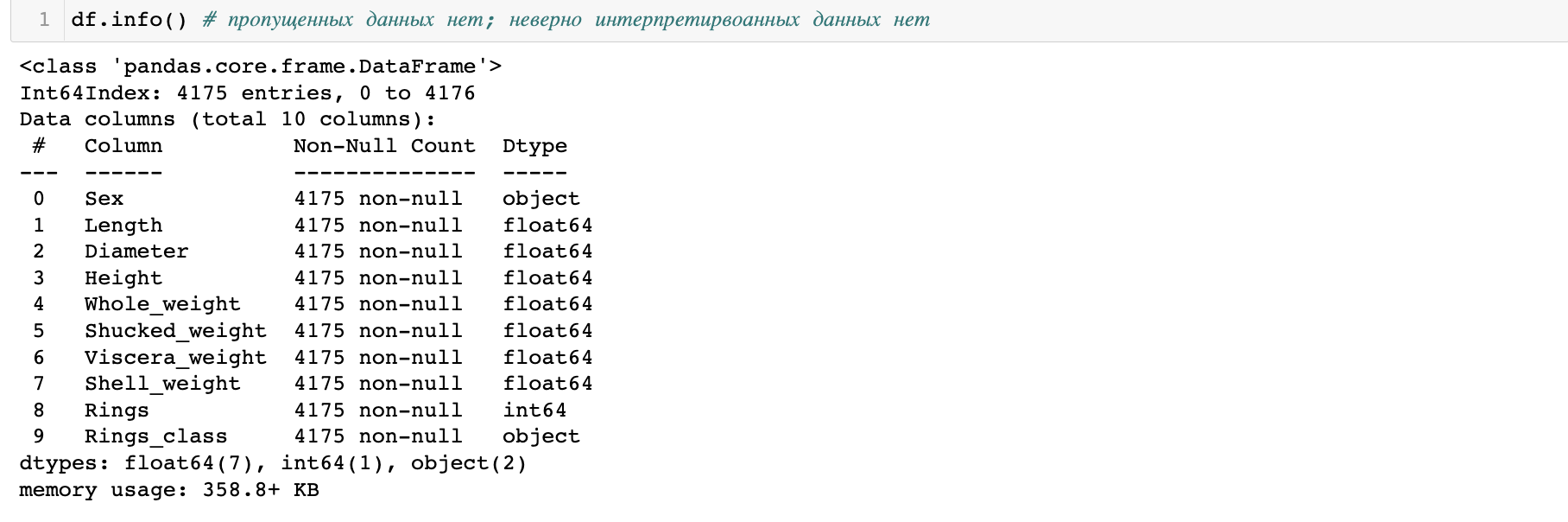
1. Осуществим описание набора данных.



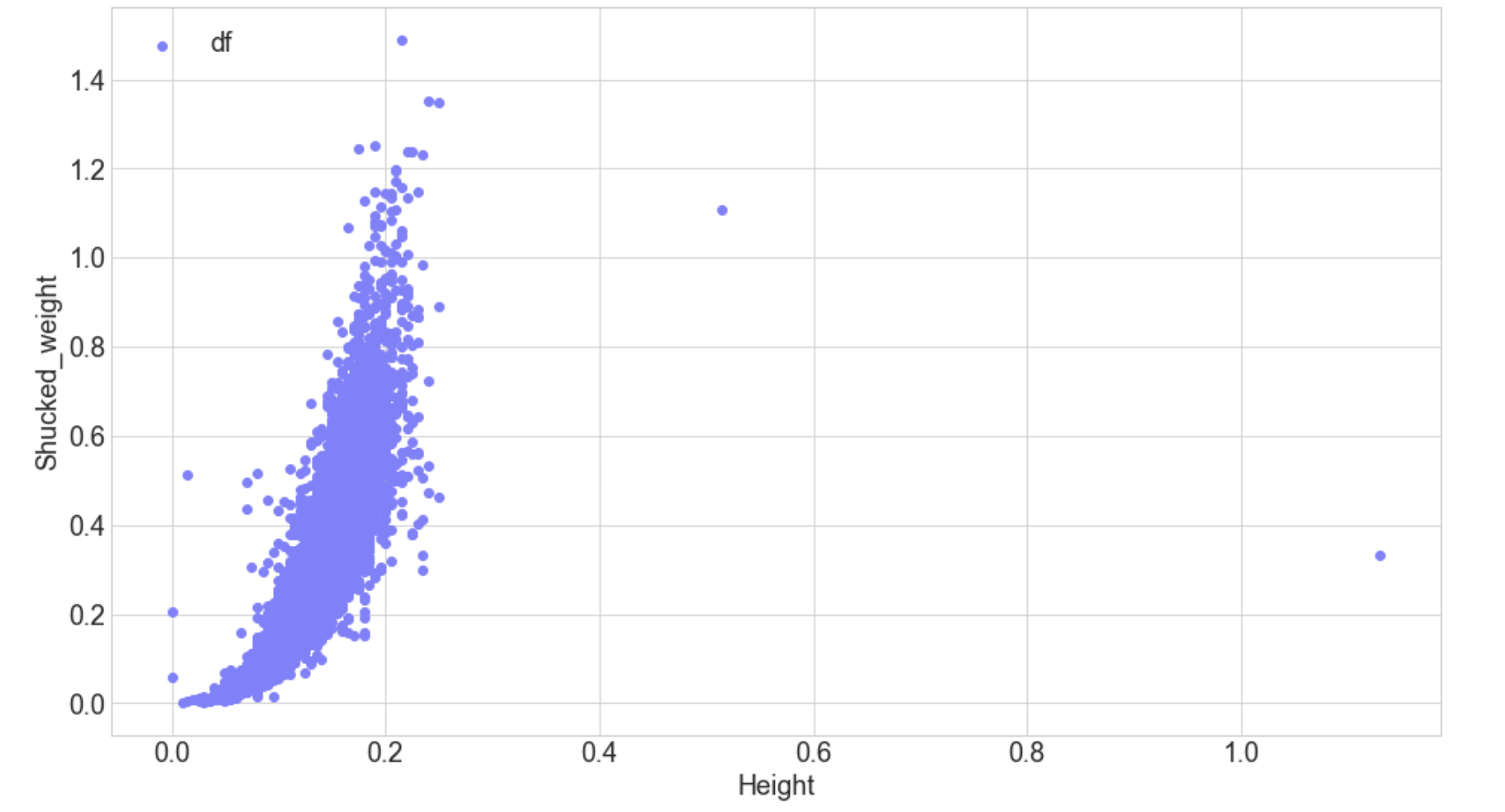
Загрузим наши данные и распечатаем первые 10 наблюдений.



1. Подготовим данные для работы. Определим имеются ли в наших данных пропуски:



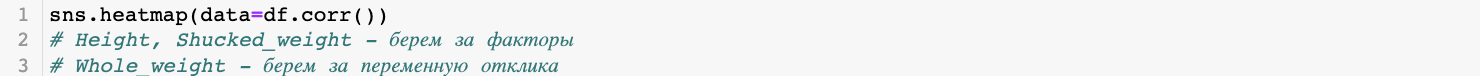
Видно, что пропусков нет и все данные корректно интерпретированы в свои типы. Теперь проверим визуально на выбросы.

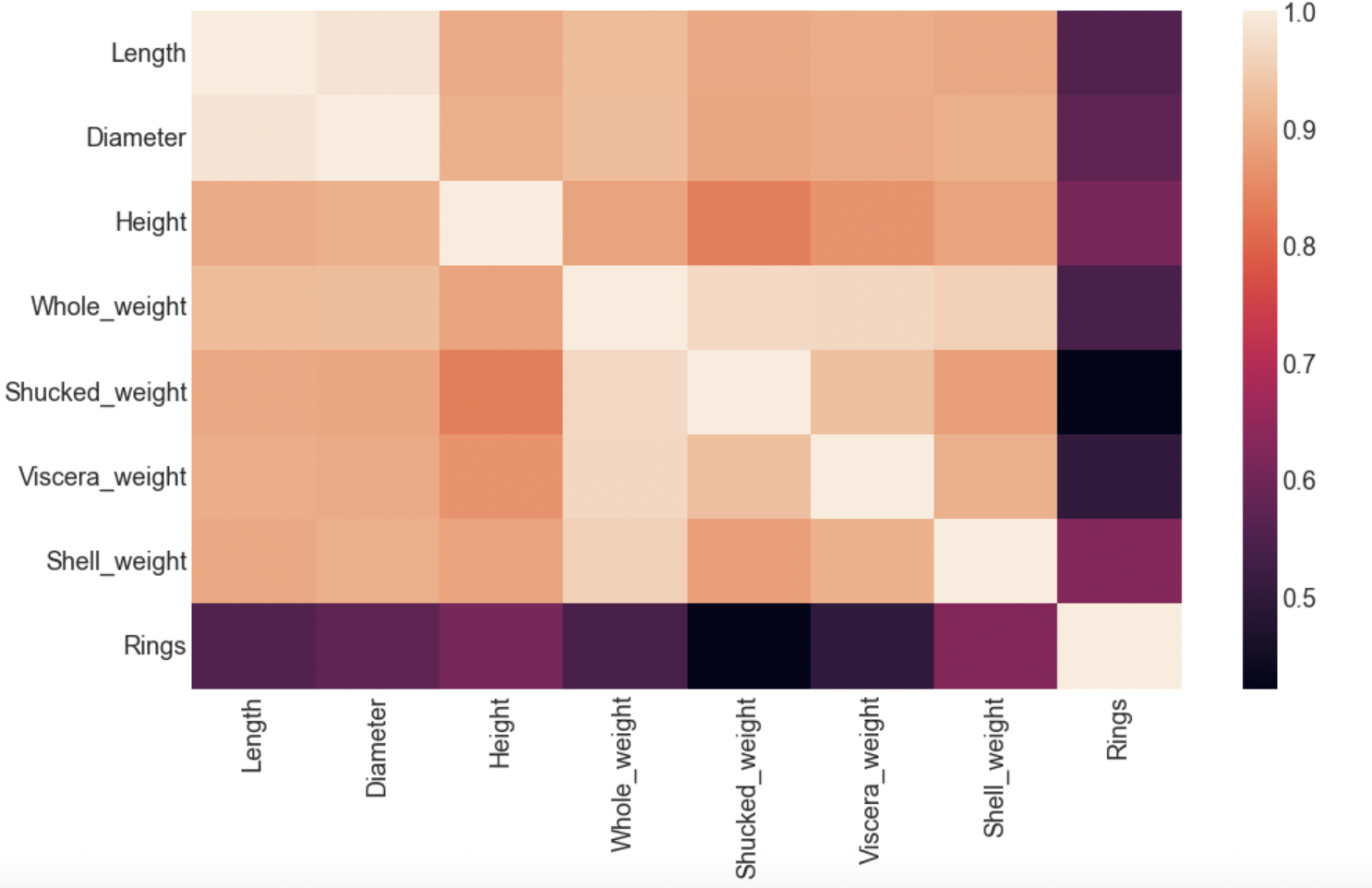
 

Видно, что имеется два выброса. Удалим их.

Рисунок 64

1. Рассчитаем матрицу корреляции.





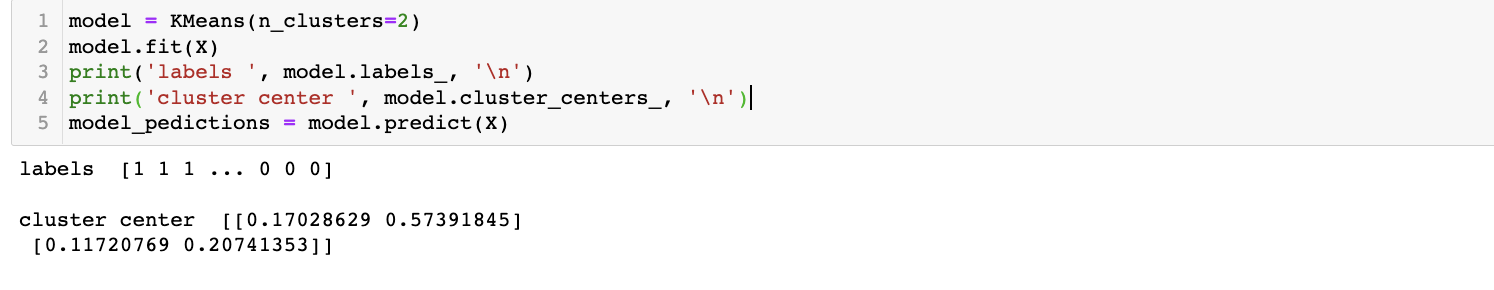
Разделим переменную Whole\_weight на два класса: «Over 1» и «Less 1».

Рисунок 6

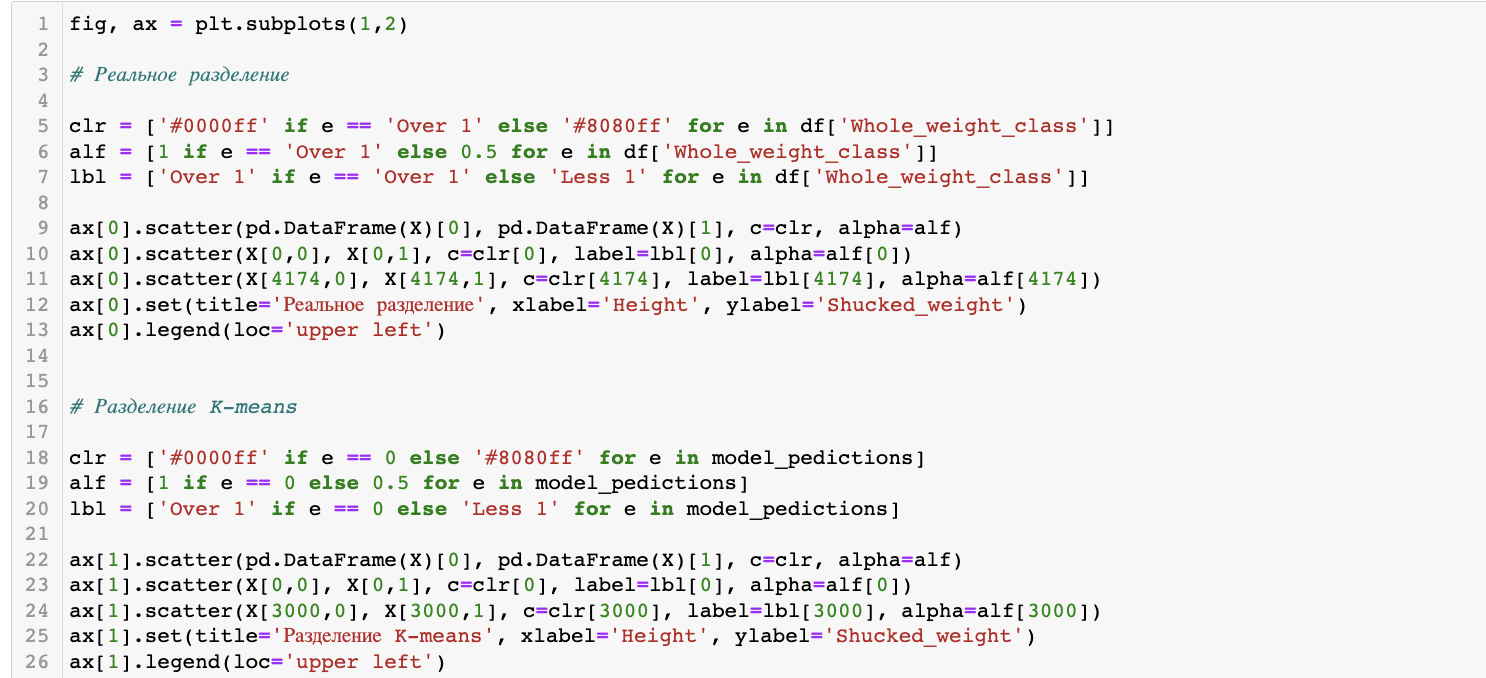
Переменные Height (Высота), Shucked\_weight (Очищенный вес) - менее всего коррелируют, по этой причине мы берем их за факторы, а Whole\_weight\_class - берем за переменную отклика

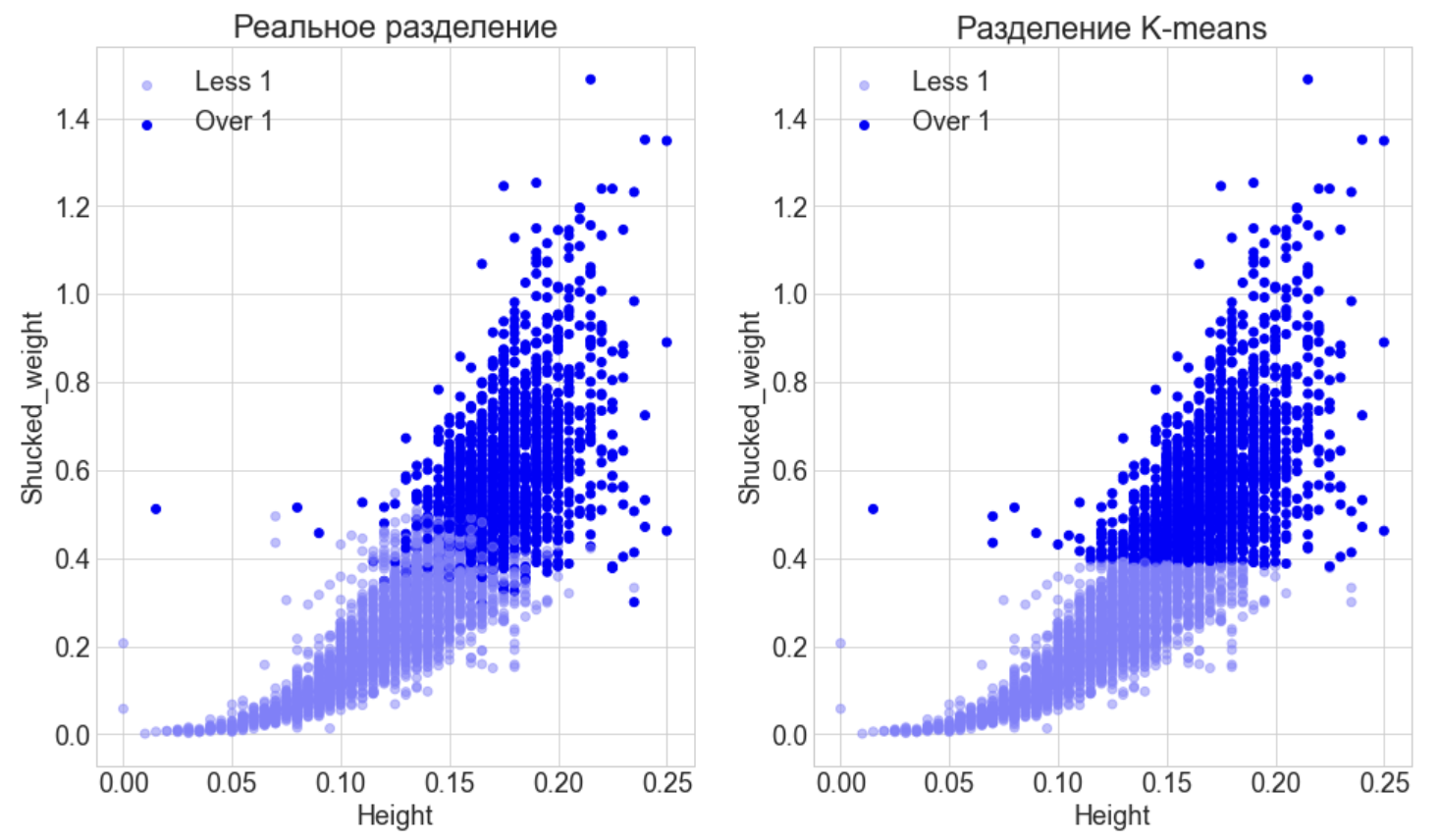
Рисунок 7

1. Нам заранее известно, что в нашем наборе данных имеется два класса, по этой причине установим параметр модели n\_clusters равный двум и обучим нашу модель.



1. Построим два графика – график слева будем отображать реальное разделение (цветовое) двух классов, правый график будет отображать разделение (цветовое), полученное с помощью k-means.

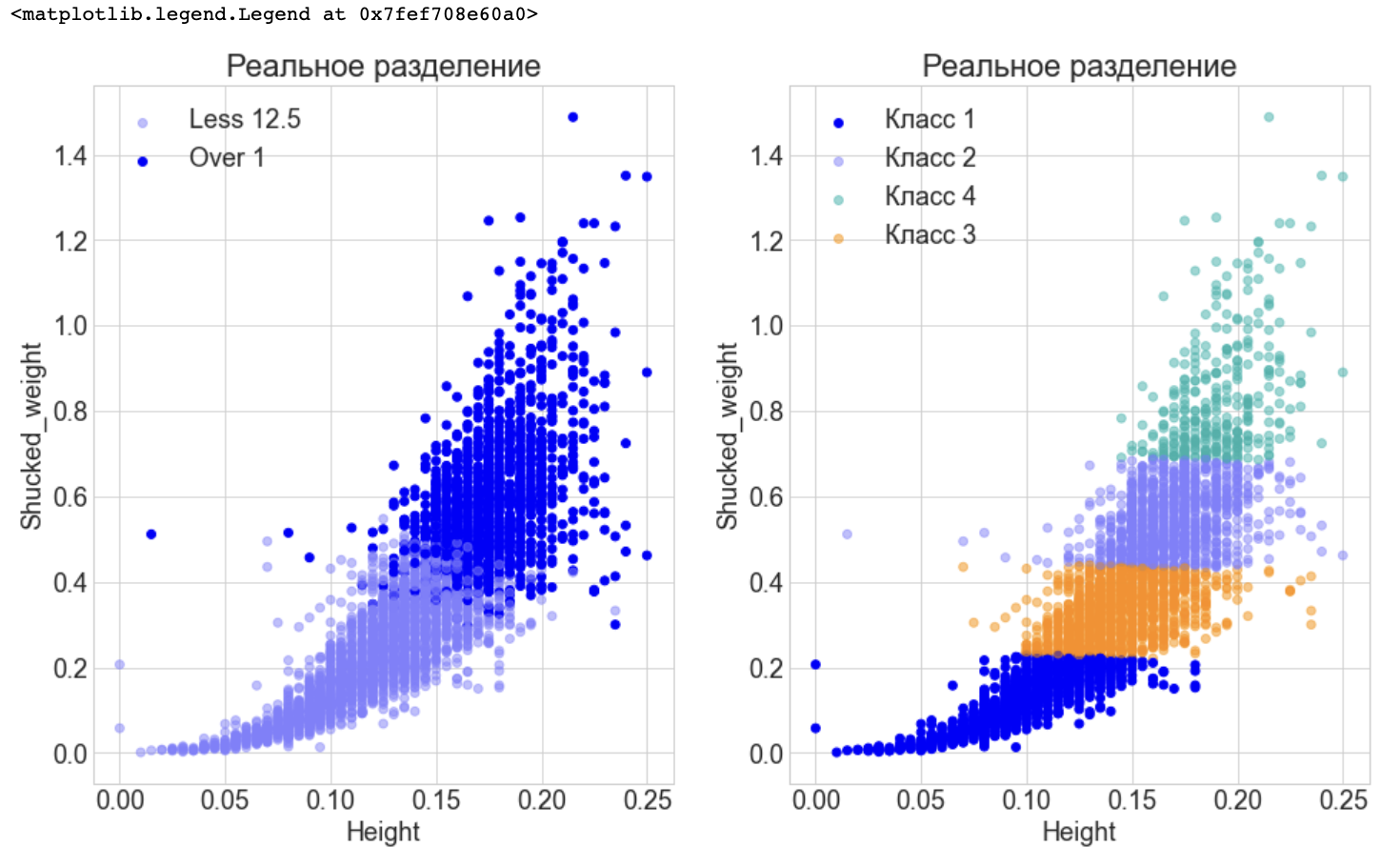




Видно, что в среднем модель работает приемлемо, но слабо справляется с точками сильно удаленные от общей массы – по этой причине впредь нужно выбирать такие данные, которые визуально можно разделить на классы.

1. Попробуем задать большее количество классов.



Как видим, наша выборка разделилась на 4 класса и если бы мы изначально не имели представления о классах, мы бы могли не правильно интерпретировать данные.

1. Вывод: в алгоритме k-means при выборе числа кластеров = 2 данные разделились в соответствии с используемом набором данных, где уже было выделено 2 класса, однако с заметными ошибками.

**Приложение:**

|  |
| --- |
| # Импортируем необходимые библиотеки для анализа данных  # Библиотеки работы с данными  import pandas as pd  import numpy as np  # Библиотки визуализации  import matplotlib.pyplot as plt  import seaborn as sns  # Настройка графиков  plt.style.use('seaborn-whitegrid')  plt.rcParams["figure.figsize"] = (16, 9)  plt.rcParams.update({'font.size': 18})  # Библиотека машинного обучения  import sklearn  from sklearn.model\_selection import train\_test\_split  from sklearn import svm  from sklearn import tree  from sklearn.tree import DecisionTreeClassifier  from sklearn.metrics import classification\_report  from sklearn.metrics import confusion\_matrix  from sklearn.metrics import accuracy\_score |
| # Импортируем данные  names = [  'Sex', 'Length', 'Diameter',  'Height', 'Whole\_weight', 'Shucked\_weight',  'Viscera\_weight', 'Shell\_weight', 'Rings'  ]  df = pd.read\_csv('abalone.csv', names=names)  df.head(10) |
| df['Whole\_weight\_class'] = np.where(df.Whole\_weight > 1, 'Over 1', 'Less 1') |
| df.head(5) |
| df.info() # пропущенных данных нет; неверно интерпретирвоанных данных нет |
| fig, ax = plt.subplots()  ax.scatter(df.Height, df.Shucked\_weight, label='df', c='#8080ff')  ax.set\_xlabel('Height')  ax.set\_ylabel('Shucked\_weight')  plt.legend(loc='upper left') |
| df = df.drop(index=list(df.query("Height>0.4").index)) |
| sns.heatmap(data=df.corr(numeric\_only=True))  # Height, Shucked\_weight - менее всего коррелируют  # Height, Shucked\_weight - берем за факторы |
| X, y = df[['Height', 'Shucked\_weight']].values, df['Whole\_weight\_class'] |
| model = KMeans(n\_clusters=2)  model.fit(X)  print('labels ', model.labels\_, '\n')  print('cluster center ', model.cluster\_centers\_, '\n')  model\_pedictions = model.predict(X) |
| fig, ax = plt.subplots(1,2)  # Реальное разделение  clr = ['#0000ff' if e == 'Over 1' else '#8080ff' for e in df['Whole\_weight\_class']]  alf = [1 if e == 'Over 1' else 0.5 for e in df['Whole\_weight\_class']]  lbl = ['Over 12.5' if e == 'Over 1' else 'Less 1' for e in df['Whole\_weight\_class']]  ax[0].scatter(pd.DataFrame(X)[0], pd.DataFrame(X)[1], c=clr, alpha=alf)  ax[0].scatter(X[0,0], X[0,1], c=clr[0], label=lbl[0], alpha=alf[0])  ax[0].scatter(X[4174,0], X[4174,1], c=clr[4174], label=lbl[4174], alpha=alf[4174])  ax[0].set(title='Реальное разделение', xlabel='Height', ylabel='Shucked\_weight')  ax[0].legend(loc='upper left')  # Разделение K-means  clr = ['#0000ff' if e == 0 else '#8080ff' for e in model\_pedictions]  alf = [1 if e == 0 else 0.5 for e in model\_pedictions]  lbl = ['Over 1' if e == 0 else 'Less 1' for e in model\_pedictions]  ax[1].scatter(pd.DataFrame(X)[0], pd.DataFrame(X)[1], c=clr, alpha=alf)  ax[1].scatter(X[0,0], X[0,1], c=clr[0], label=lbl[0], alpha=alf[0])  ax[1].scatter(X[3000,0], X[3000,1], c=clr[3000], label=lbl[3000], alpha=alf[3000])  ax[1].set(title='Разделение K-means', xlabel='Height', ylabel='Shucked\_weight')  ax[1].legend(loc='upper left') |
| model = KMeans(n\_clusters=4)  model.fit(X)  print('labels ', model.labels\_, '\n')  print('cluster center ', model.cluster\_centers\_, '\n')  model\_pedictions = model.predict(X) |
| fig, ax = plt.subplots(1,2)  # Реальное разделение  clr = ['#0000ff' if e == 'Over 1' else '#8080ff' for e in df['Whole\_weight\_class']]  alf = [1 if e == 'Over 1' else 0.5 for e in df['Whole\_weight\_class']]  lbl = ['Over 1' if e == 'Over 1' else 'Less 12.5' for e in df['Whole\_weight\_class']]  ax[0].scatter(pd.DataFrame(X)[0], pd.DataFrame(X)[1], c=clr, alpha=alf)  ax[0].scatter(X[0,0], X[0,1], c=clr[0], label=lbl[0], alpha=alf[0])  ax[0].scatter(X[4174,0], X[4174,1], c=clr[4174], label=lbl[4174], alpha=alf[4174])  ax[0].set(title='Реальное разделение', xlabel='Height', ylabel='Shucked\_weight')  ax[0].legend(loc='upper left')  # Разделение K-means  clr = ['#0000ff' if e == 0 else '#8080ff' if e==1 else '#FF8C00' if e==2 else '#20B2AA' for e in model\_pedictions]  alf = [1 if e == 0 else 0.5 if e==1 else 0.5 if e==2 else 0.5 for e in model\_pedictions]  lbl = ['Класс 1' if e == 0 else 'Класс 2' if e==1 else 'Класс 3' if e==2 else 'Класс 4' for e in model\_pedictions]  ax[1].scatter(pd.DataFrame(X)[0], pd.DataFrame(X)[1], c=clr, alpha=alf)  ax[1].scatter(X[0,0], X[0,1], c=clr[0], label=lbl[0], alpha=alf[0])  ax[1].scatter(X[1500,0], X[1500,1], c=clr[1500], label=lbl[1500], alpha=alf[1500])  ax[1].scatter(X[4174,0], X[4174,1], c=clr[4174], label=lbl[4174], alpha=alf[4174])  ax[1].scatter(X[2,0], X[2,1], c=clr[2], label=lbl[2], alpha=alf[2])  ax[1].set(title='Реальное разделение', xlabel='Height', ylabel='Shucked\_weight')  ax[1].legend(loc='upper left') |