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:

Студентка гр.09-305

Шарафеев Марат

Преподаватель:

Шустова Е.П.

Казань-2024

**Задание №9. Машина опорных векторов (Support vector machine, SVM)**

**Выполните следующее задачи:**

1. Подберите свои данные для решения задачи классификации с помощью SVM. 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.

Приведите здесь в отчете краткое описание выбранных Вами и закрепленных в Teams данных и интернет-ссылку на эти данные.

Осуществите описание Вашего набора данных. При описании данных приведите названия полей этих данных и на русском языке что означают эти поля.

2. Установите необходимые внешние библиотеки. Описать способ установки, используемый Вами.

3. Приведите скриншоты путей, в которые Вы установили Python, среду разработки (например, PyCharm) и внешние библиотеки.

4. Импортируйте необходимые пакеты и классы:

5. 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. Сформируйте обучающую и тестовую выборки с помощью метода train\_test\_split из scikitlearn (например так: train\_test\_split(X, Y, test\_size = 0.25, random\_state=9).

9. Если число факторов m <= 2, то визуализируйте данные (постройте облако точек).

10. Отделите от Ваших данных прогнозное значение (переменную отклика), то есть удалите из Ваших данных столбец с прогнозными значениями и запишите его в отдельную переменную.

11. Постройте натренированную модель методом SVM. Сделайте выводы из полученного результата.

12. Найдите средние абсолютную и квадратичную оценку ошибки модели и качество модели.

13. Сравните теоретические и эмпирические значения переменной отклика на каких-нибудь двух значениях наблюдений из тестовой выборки. Сделайте вывод.

14. Приведите полный код для решения задачи. Код сопровождайте комментариями.

15. Оформите отчёт о результатах исследования и прикрепите его в ЭОР «Введение в машинное обучение на Python» по ссылке https://repository.kpfu.ru/?p\_id=236426 к указанному преподавателем сроку до или во время своей пары.

16. К этому же заданию прикрепите файл с данными и py-файл с программой.

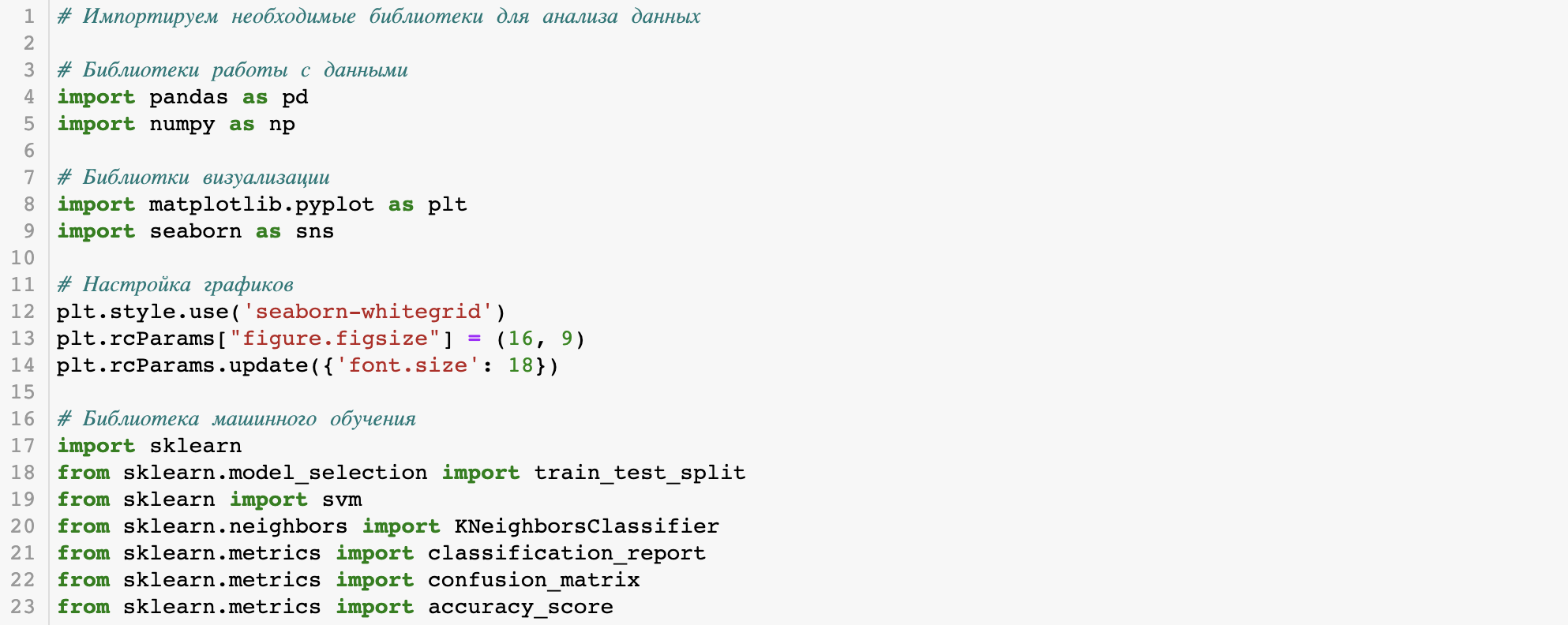
# **Ход работы:**

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).

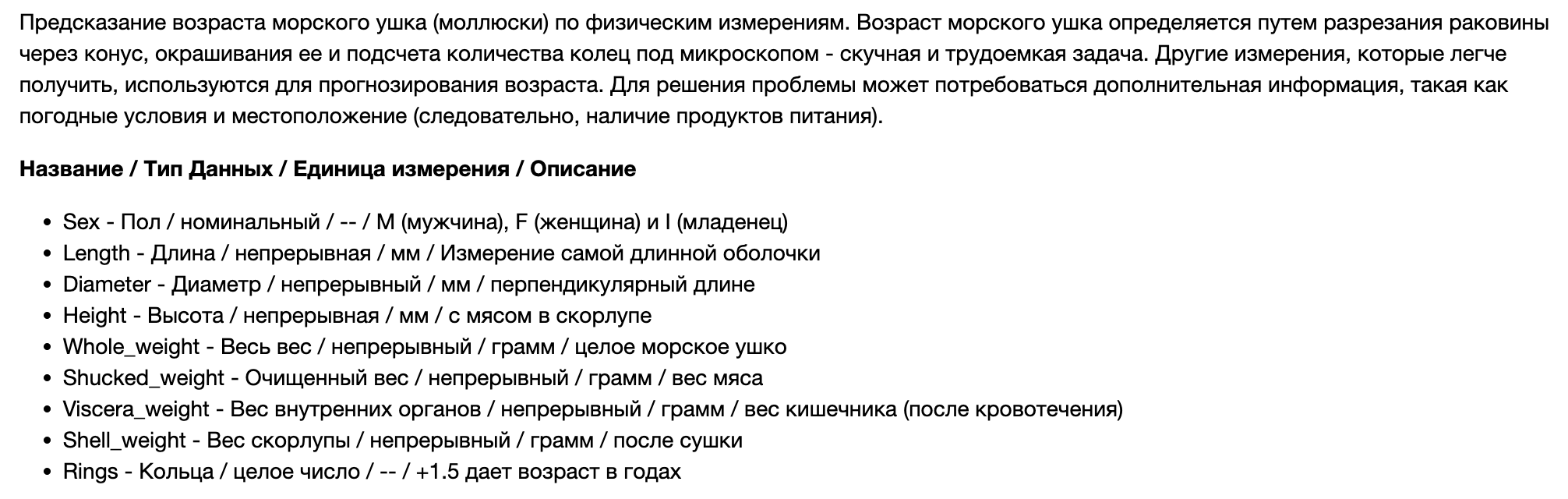
Данные содержат следующие характеристики:

|  |  |  |  |
| --- | --- | --- | --- |
| **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 |

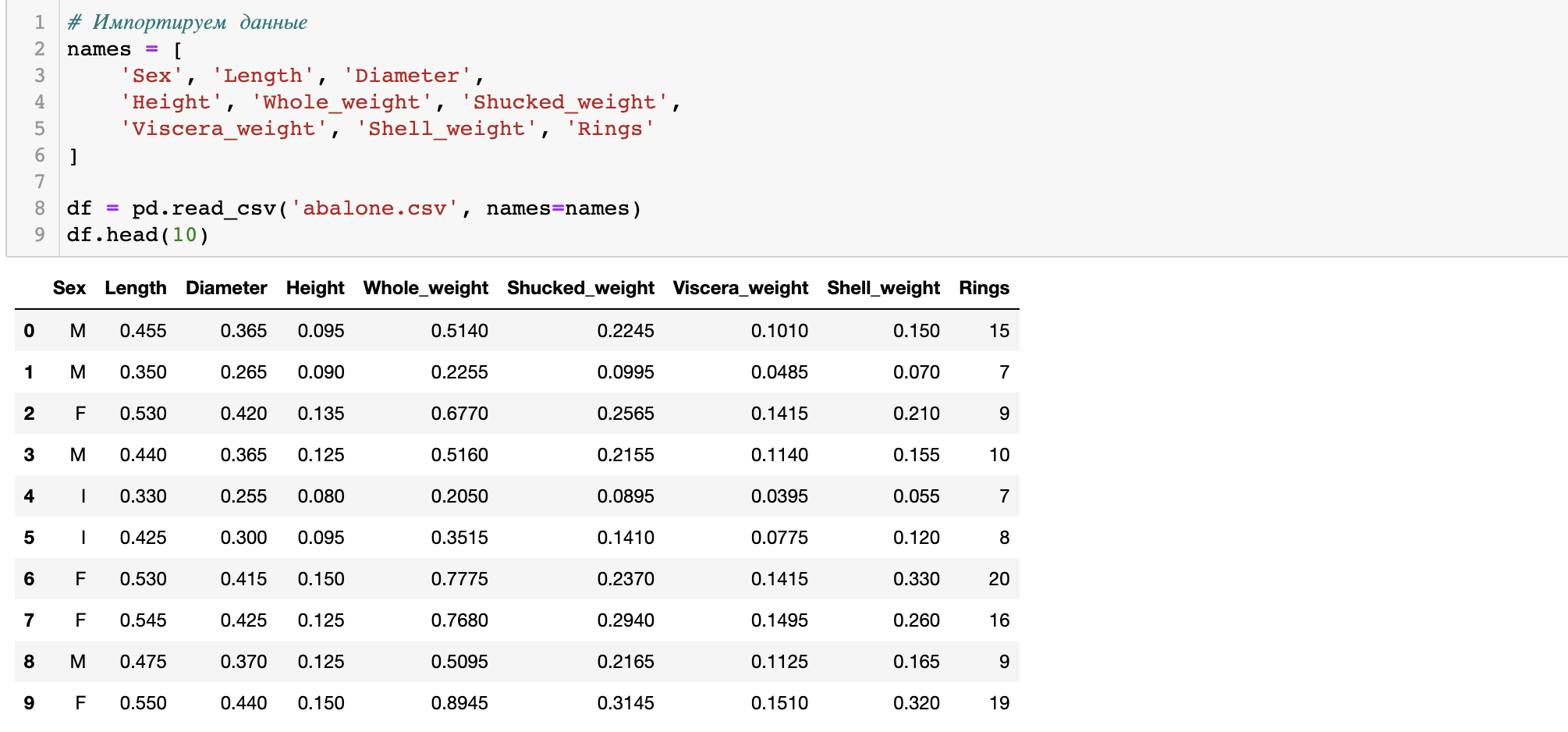
1. Импортируем все необходимые библиотеки:



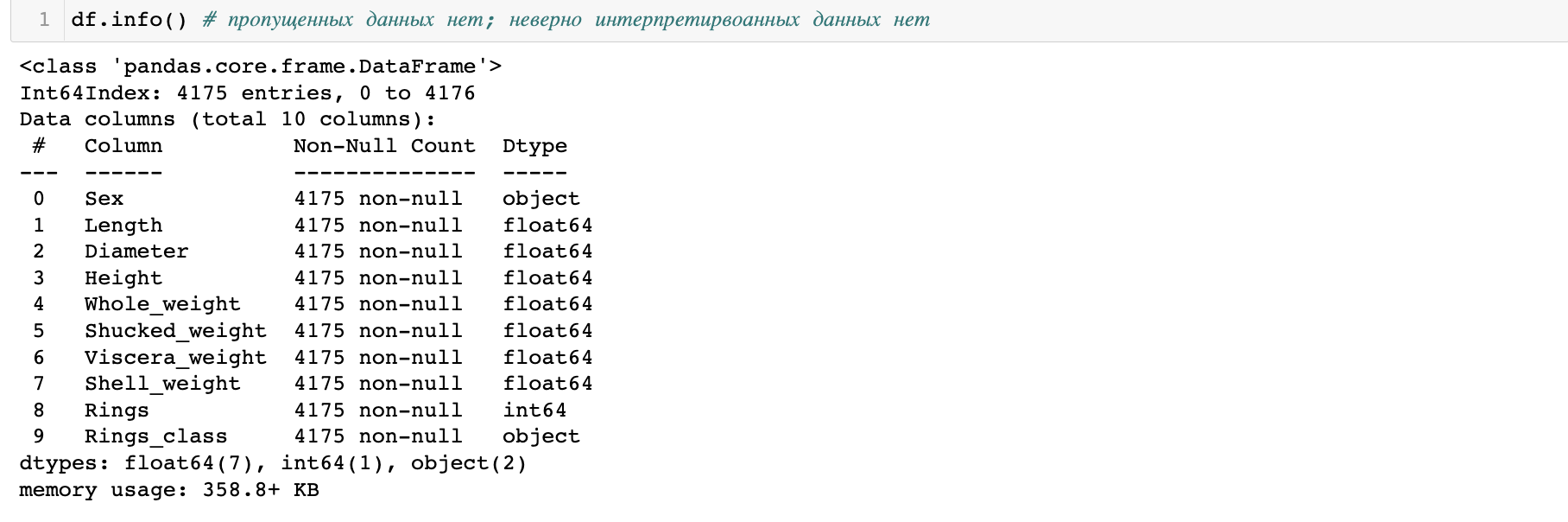
1. Осуществим описание набора данных.



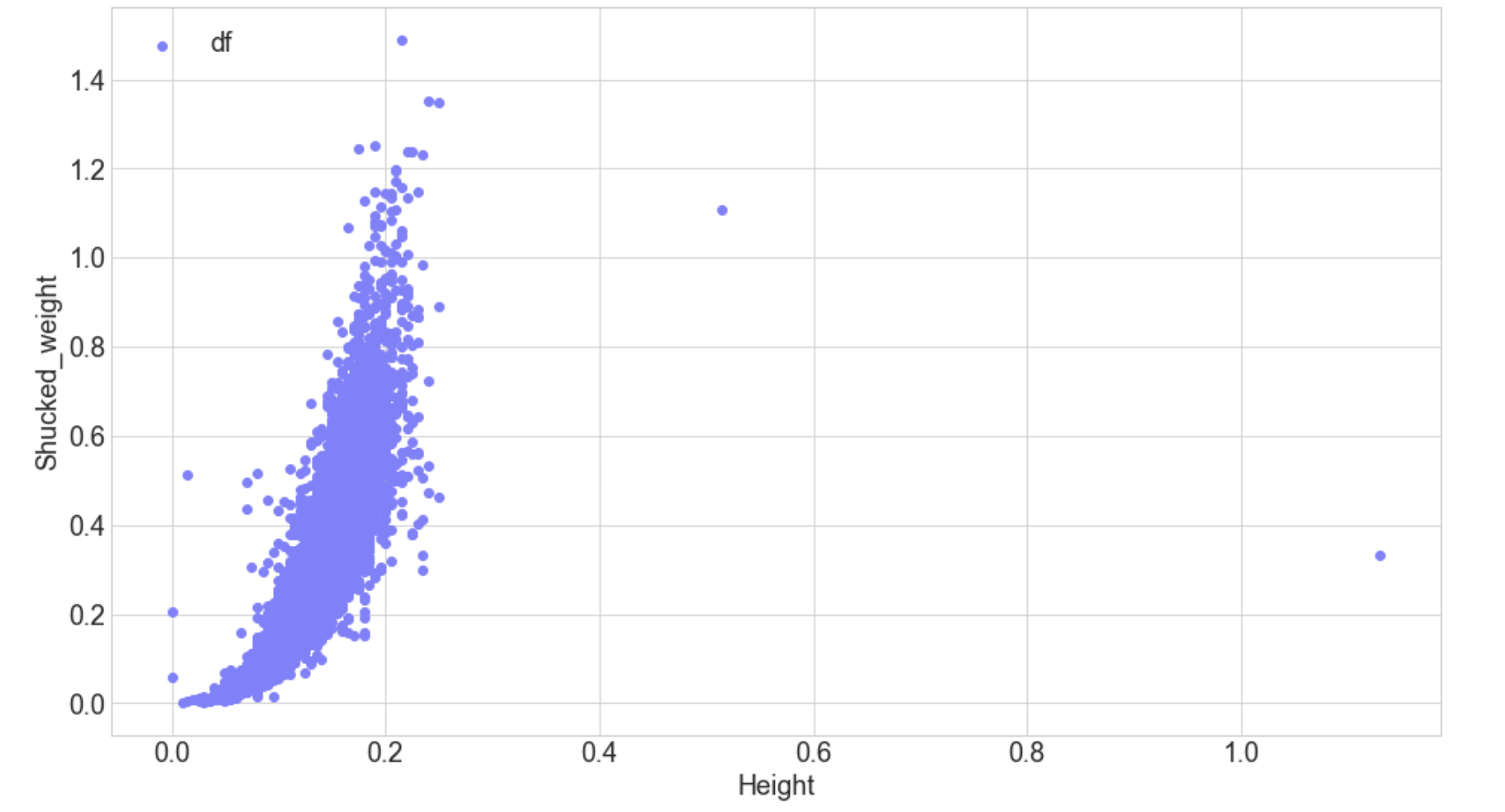
Загрузим наши данные и распечатаем первые 10 наблюдений.



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



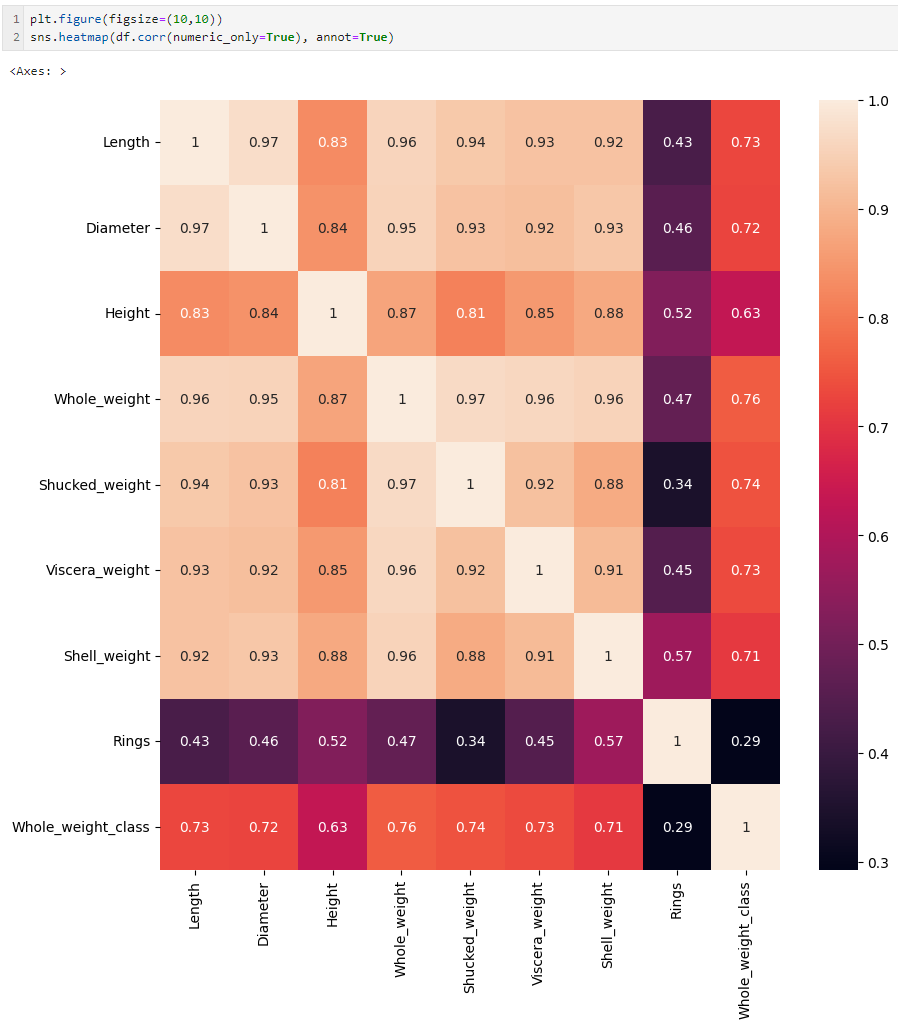
Видно, что пропусков нет и все данные корректно интерпретированы в свои типы.   
Построим гистограмм для числовых признаков.

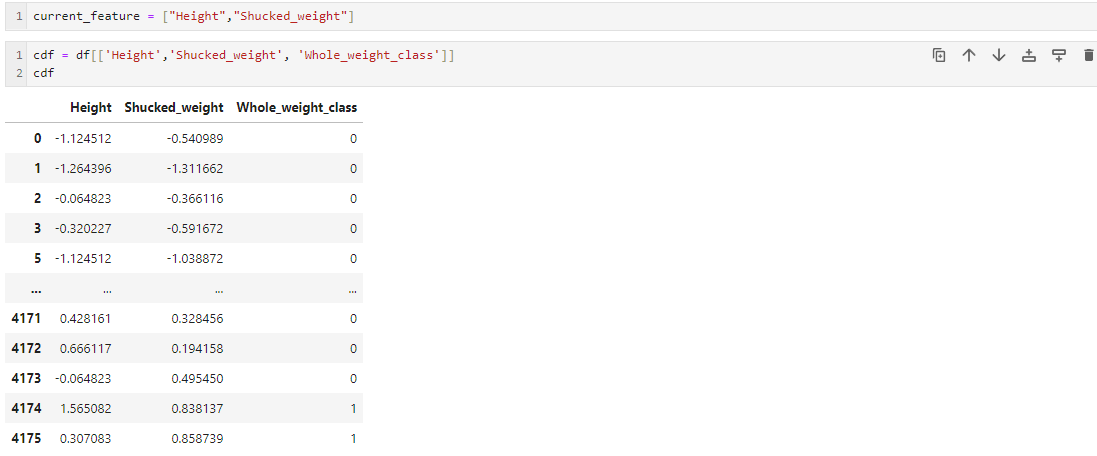
Видно, что имеется два выброса. Удалим их.

Рисунок 64

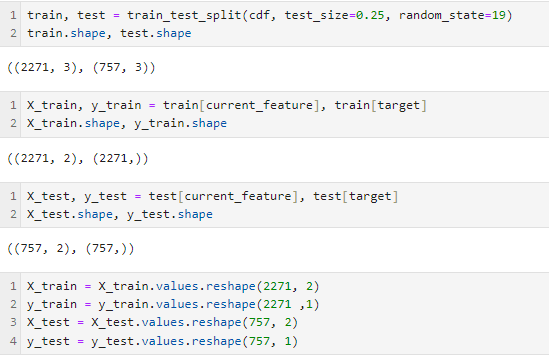
1. Рассчитаем матрицу корреляции.



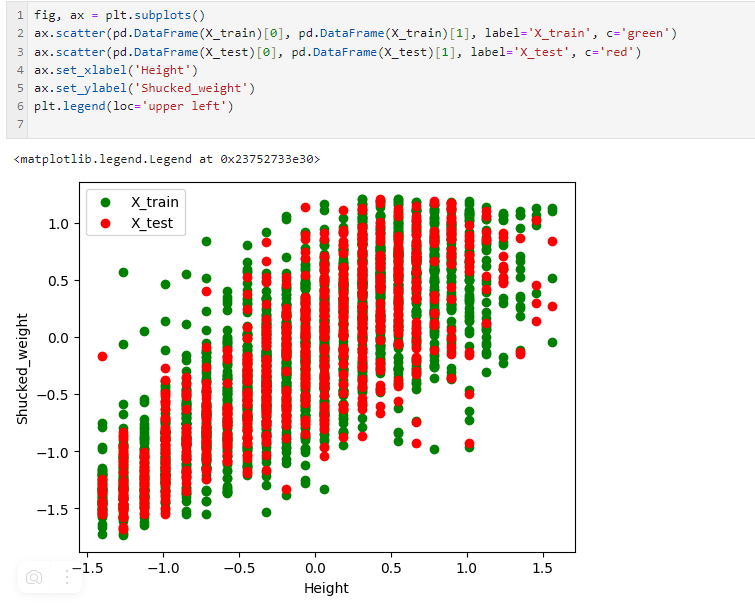
Переменные Height (Высота), Shucked\_weight (Очищенный вес) - менее всего коррелируют, по этой причине мы берем их за факторы, а Whole\_weight\_class - берем за переменную отклика



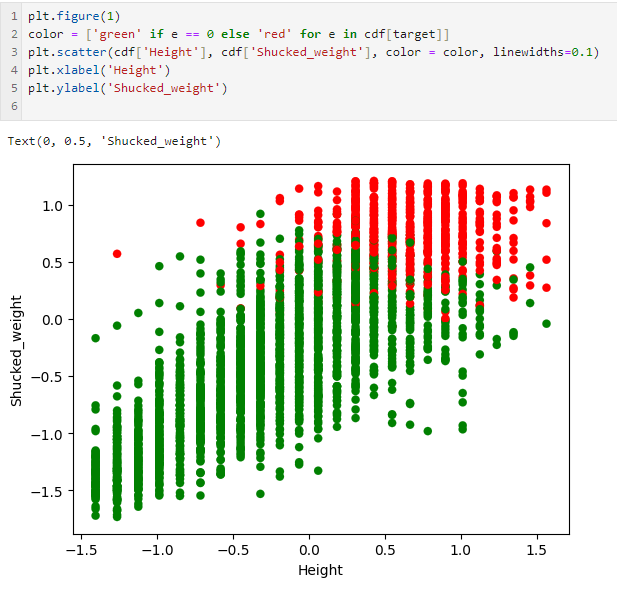
1. Сформируйте обучающую и тестовую выборки.



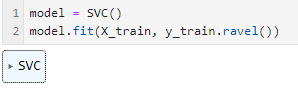
1. Визуализируем наши данные:



На графике разным цветом отмечены тестовые и тренировочные данные. Теперь посмотрим как распределены разные классы.



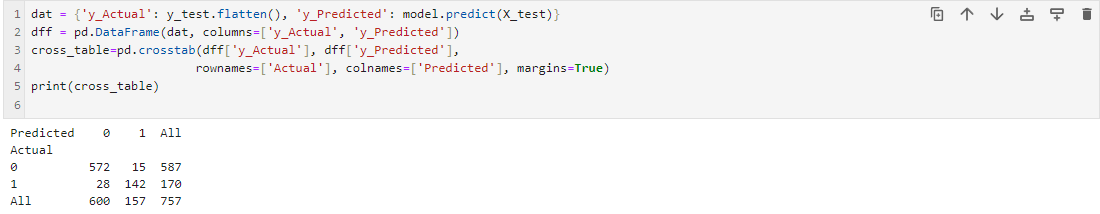
1. Построим натренированную модель методом SVM



Теперь получим предсказание

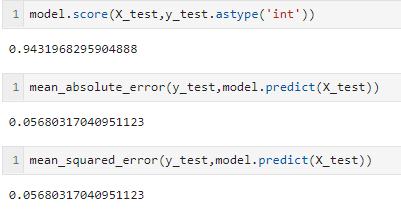
Рисунок 1

1. Для оценки точности используем кросс-валидационную таблицу:



Наша выборка состояла из 757 объектов – 600 объекта первого класса, 157 объектов второго класса. Модель допустила ошибки в 43 случаях из 757, что соответствует точности прогноза в 94%.

1. Воспользуемся оценкой модели для того, чтобы сравнить полученные значения



Результаты оказались идентичными

1. Вывод: Таким образом, мы допускаем, что наличие информации о очищенном весе и высоте морских ушков дает достаточно хорошую оценку о том, молодой моллюск или старый.  
     
   **Приложение:**

|  |
| --- |
| # Импортируем необходимые библиотеки для анализа данных  import numpy as np  import pandas as pd  import matplotlib.pyplot as plt  import seaborn as sns  from sklearn.model\_selection import train\_test\_split  from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score  from sklearn.preprocessing import PowerTransformer  from sklearn.svm import SVC  # 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, 1, 0) |
| df |
| df.info() |
| df.describe() |
| target = "Whole\_weight\_class"  features = [  'Length', 'Diameter',  'Height', 'Whole\_weight', 'Shucked\_weight',  'Viscera\_weight', 'Shell\_weight', 'Rings'  ] |
| df.hist(figsize=(15, 10), bins=20) |
| pt = PowerTransformer()  select\_feats = features  df[select\_feats] = pt.fit\_transform(df[select\_feats])  df.hist(figsize=(15, 10), bins=20) |
| fig = plt.figure(figsize=(15, 15))  for i in range(1, 8):  ax = fig.add\_subplot(3, 3, i)  ax.boxplot(x = df[features[i-1]])  ax.set\_title(features[i-1]) |
| lower\_bound = df['Length'].quantile(q=0.025)  print(lower\_bound)  df=df[df['Length'] > lower\_bound]  lower\_bound = df['Diameter'].quantile(q=0.025)  print(lower\_bound)  df=df[df['Diameter'] > lower\_bound]  upper\_bound = df['Height'].quantile(q=0.975)  lower\_bound = df['Height'].quantile(q=0.025)  print(upper\_bound)  print(lower\_bound)  df=df[df['Height'] < upper\_bound]  df=df[df['Height'] > lower\_bound]  upper\_bound = df['Whole\_weight'].quantile(q=0.975)  print(upper\_bound)  df=df[df['Whole\_weight'] < upper\_bound]  upper\_bound = df['Shucked\_weight'].quantile(q=0.975)  print(upper\_bound)  df=df[df['Shucked\_weight'] < upper\_bound]  upper\_bound = df['Viscera\_weight'].quantile(q=0.975)  print(upper\_bound)  df=df[df['Viscera\_weight'] < upper\_bound]  upper\_bound = df['Shell\_weight'].quantile(q=0.975)  print(upper\_bound)  df=df[df['Shell\_weight'] < upper\_bound]  df |
| plt.figure(figsize=(10,10))  sns.heatmap(df.corr(numeric\_only=True), annot=True) |
| current\_feature = ["Height","Shucked\_weight"] |
| cdf = df[['Whole\_weight\_class', 'Height','Shucked\_weight']]  cdf |
| train, test = train\_test\_split(cdf, test\_size=0.25, random\_state=19)  train.shape, test.shape |
| X\_train, y\_train = train[current\_feature], train[target]  X\_train.shape, y\_train.shape |
| X\_test, y\_test = test[current\_feature], test[target]  X\_test.shape, y\_test.shape |
| X\_train = X\_train.values.reshape(2271, 2)  y\_train = y\_train.values.reshape(2271 ,1)  X\_test = X\_test.values.reshape(757, 2)  y\_test = y\_test.values.reshape(757, 1) |
| fig = plt.figure()  ax = fig.add\_subplot(111, projection = '3d')  #Построим график рассеивания  x = cdf['Shucked\_weight']  y = cdf['Height']  z = cdf['Whole\_weight\_class']  ax.scatter(x, y, z)  ax.set\_xlabel("Shucked\_weight")  ax.set\_ylabel("Height")  ax.set\_zlabel("Whole\_weight\_class")  plt.show() |
| fig, ax = plt.subplots()  ax.scatter(pd.DataFrame(X\_train)[0], pd.DataFrame(X\_train)[1], label='X\_train', c='green')  ax.scatter(pd.DataFrame(X\_test)[0], pd.DataFrame(X\_test)[1], label='X\_test', c='red')  ax.set\_xlabel('Height')  ax.set\_ylabel('Shucked\_weight')  plt.legend(loc='upper left') |
| plt.figure(1)  color = ['green' if e == 0 else 'red' for e in cdf[target]]  plt.scatter(cdf['Height'], cdf['Shucked\_weight'], color = color, linewidths=0.1)  plt.xlabel('Height')  plt.ylabel('Shucked\_weight') |
| model = SVC()  model.fit(X\_train, y\_train.ravel()) |
| y\_test[:10] |
| model.predict(X\_test) |
| dat = {'y\_Actual': y\_test.flatten(), 'y\_Predicted': model.predict(X\_test)}  dff = pd.DataFrame(dat, columns=['y\_Actual', 'y\_Predicted'])  cross\_table=pd.crosstab(dff['y\_Actual'], dff['y\_Predicted'],  rownames=['Actual'], colnames=['Predicted'], margins=True)  print(cross\_table) |
| model.score(X\_test,y\_test.astype('int')) |
| mean\_absolute\_error(y\_test,model.predict(X\_test)) |
| mean\_squared\_error(y\_test,model.predict(X\_test)) |