**Geekbrains**

**Использование машинного обучения в прикладных задачах на примере пищевой промышленности**

**Программа: Аналитик больших данных**

Студентка: Бражникова Нина Викторовна

**г. Лобня**

**2024 г.**

**Содержание**

[Введение 3](#_Toc156246554)

[Актуальность темы дипломной работы 3](#_Toc156246555)

[Цели и задачи итогового проекта 4](#_Toc156246556)

[Глава 1 6](#_Toc156246557)

[Машиное обучение в нашей жизни 6](#_Toc156246558)

[Обработка входных данных 9](#_Toc156246559)

[Линейная регрессия 15](#_Toc156246560)

[Методы классификации в Python 21](#_Toc156246561)

[Глава 2 27](#_Toc156246562)

[Задача 1 - получение данных 27](#_Toc156246563)

[Решение задачи 1: 29](#_Toc156246564)

[Задача 2 - первичный анализ данных 31](#_Toc156246565)

[Решение задачи 2: 33](#_Toc156246566)

[Задача 3 - обучение регрессионной модели предсказания массы рыбы 34](#_Toc156246567)

[Решение задачи 3: 38](#_Toc156246568)

[Задача 4 - обучение модели классификации для предсказания качества разрабатываемой рецептуры шоколада 41](#_Toc156246569)

[Решение задачи 4: 45](#_Toc156246570)

[Заключение 54](#_Toc156246571)

[Список используемой литературы 55](#_Toc156246572)

[Приложения 57](#_Toc156246573)

Введение

## Актуальность темы дипломной работы

При разработке и создании первых систем управления базами данных возникает вопрос алгоритмов анализа собираемой информации. В каждой сфере деятельности происходит рост объема такой информации, и разработка новых эффективных методов, алгоритмов анализа данных приобретает особую важность для организаций.

Необходимо отметить, что существуют различные типы данных, которые могут быть использованы несколькими способами и в разных целях. Например, каждый оператор сотовой связи сохраняет информацию об объеме и характере потребляемых клиентом услуг, динамике расходования трафика; социальные сети работают с данными об активности пользователей, их связях и характере контента. По мере накопления всё большего объема данных у коммерческих и государственных компаний возникает понятное желание найти им полезное применение, в частности – прогнозировать отток пользователей, выявить зависимости, например, между объемами использования всех услуг из пакета и готовностью перейти к другому оператору.

Аналитика больших данных способствует не только принятию обоснованных решений при разработке новых продуктов, в оперативном управлении уже существующих, но и создают дополнительную ценность для потребителей. Что несомненно двигает нашу цивилизацию к новому витку развития. Безусловно, всё чаще и громче слышны голоса возмущения таким положением дел и опасений как частных лиц, так и бизнеса, что хранение и использование больших данных может быть применено не только в пользу (компании могут больше зарабатывать, пользователям предоставляется всё более удобный, персонализированный сервис), но и стать проблемой экологического, социального или этического плана.

Специалисты аналитического фронта обобщают данные, и разрабатывают модели на основе внутренней логики этой информации. В ход идут методы математического и статистического анализа, чтобы данную логику нащупать и использовать в том или ином приложении к ежедневным вопросам компаний и государственных институций.

## Цели и задачи итогового проекта

Все вышесказанное абсолютно применимо и к пищевой промышленности. В частности, можно без существенных затрат на производство пробных партий продукции предсказать успешность той или иной рецептуры на основе только данных о ее составляющих и сопутствующих факторах (например, качество и происхождение ингредиентов, репутация поставщика и иные параметры). В данной работе целью как раз и будет рассмотрение алгоритмов машинного обучения для оценки качества разрабатываемого рецепта продукта в зависимости от различных параметров. По моим ощущениям недавнего экономиста и бухгалтера это очень похоже на приоритизацию инициатив оптимизации бизнес модели с помощью фреймворка Rice. Там так же осуществляется взвешивание влияния ряда факторов на конечный результат без затратных «проб и ошибок». Безусловно, без накопления данных о реальных проектах в жизни подобные интеллектуальные выкладки не имели бы места, требовалось бы всё так же проверять делом и тратить много материальных и физических ресурсов.

Также, с целью показать широту спектра применения анализа больших данных в отдельно взятой производственной сфере построим регрессионную модель, которая может лечь в основу робототехнического комплекса для бесконтактного определения массы продукции (в нашем случае - рыбы, которую выращивают на специализированных фермах) по её размеру и другим параметрам. Для получения такого рода прогноза используется накопленная информация по измерениям разных видов рыб, возрасту и полу особей. В результате обработки данных модель выдает предсказание массы рыбы в зависимости от указанных параметров.

В рамках данной работы предстоит рассмотреть процесс работы с данными на входе, в процессе аналитики и построения моделей, непосредственно само обучение.

1. Загрузка данных;
2. Предобработка данных;
3. Обучение регрессионной модели предсказания массы рыбы;
4. Обучение модели классификации для предсказания качества рецептуры разрабатываемого шоколада.

В рамках настоящего проекта будут использованы сборные данные с портала https://www.kaggle.com, поскольку на данный момент у меня нет официального доступа к реальным бизнес-процессам. Надеюсь, такой формат все же позволит показать необходимые навыки с инструментарием аналитика больших данных, так как данный проект у меня первый в сфере информационных технологий.

Для выполнения работы были использованы Jupyter Notebook, Paint, Excel, Word, Visual Studio Code и Power BI.

Состав команды – Бражникова Нина Викторовна (аналитик)

Глава 1

## Машинное обучение в нашей жизни

Интеллектуальные алгоритмы машинного обучения уже умеют распознавать голос и изображения, занимаются аналитикой бизнес-процессов и используются в решении массы других задач. Их внедрение облегчает процесс программирования, и совершило настоящую революцию во многих отраслях человеческой деятельности. Машинное обучение напоминает ребёнка, который учится чему-то новому, он получает знания, делает на основе этой информации обобщения, выводит понятия высшего порядка. В следующий раз, когда сталкивается с этим же или подобным явлением, понимает, как реагировать на него и какие выводы из происходящего сделать.

В процессе машинного обучения выделяются три основных элемента:

1. сбор данных – чем большим количеством входных данных оперирует алгоритм, тем точнее получаются результаты его работы.;
2. извлечение признаков – в результате изучения входных данных компьютер формирует определённые признаки входных данных, строит так называемое признаковое описание данных;
3. выбор алгоритма – различные алгоритмы отличаются разной эффективностью для решения прикладных задач.

Есть большое количество подходов, согласно которым можно классифицировать способы машинного обучения, однако на практике для упрощения этого процесса принято разделять машинное обучение на три больших группы.

Обучение с учителем - этот вариант очень похож на обучение в реальной жизни, только в данном случае в роли ребёнка выступает алгоритм, на вход которому подаются данные, которые он должен анализировать. Они уже заранее промаркированы, и главной целью алгоритма в процессе становится необходимость выявить все взаимосвязи. Обычно в процессе такого обучения применяются метод опорных векторов, линейная и логистическая регрессии и другие.

Обучение без учителя - в процессе такого обучения алгоритм обладает достаточно большой степенью самостоятельности и не имеет на входе каких-либо предварительно размеченных данных. Он должен самостоятельно обработать весь этот массив, выявить имеющиеся в нём закономерности, а далее, основываясь на этих закономерностях, распознать типы данных и произвести их систематизацию. Здесь можно применять такие алгоритмы, как алгоритмы k внутригрупповых средних (k means), анализ ассоциативных правил и основных и независимых компонентов.

Обучение с подкреплением - этот тип обучения заключается в поиске оптимального пути. В наличии есть набор разных сценариев. Этот алгоритм предназначен для решения задачи выживания в реальной среде. То есть, например, поиск оптимального пути движения робота-пылесоса, ориентация в пространстве автопилота электромобиля и т. д. Для решения этой задачи могут применяться алгоритмы SARSA и DQN, которые строятся на базе родительского алгоритма Q-learning.

Понятие машинного обучения неразрывно связано и с понятием нейронных сетей. Такое название они получили, так как по своему принципу устройства и архитектуры напоминают устройство живых нейронов и их взаимодействие в живом организме. Объединённые в сеть простые нейроны получают на вход сигнал, которым обмениваются друг с другом, в результате чего на выходе рождается определённое решение.

Сегодня нейронные сети достаточно широко применяются и позволяют решать вопросы, например, из области распознавания изображений, речи (в некоторых случаях даже эффективнее, чем сам человек), то есть нейросеть не программируется в классическом смысле, а обучается в процессе обработки входных данных и при постепенном подборе нужных коэффициентов.

Как применить машинное обучение для решения нашей задачи? В упрощённом виде процесс может выглядеть как первоначальный сбор данных, построение модели и развёртывание. Сначала необходимо понять, возможно ли применение машинного обучения в конкретном случае, в процессе необходимо определить: можно ли категории нашей задачи каким-либо образом систематизировать; выражаются ли они в каких-то конкретных числах; может ли быть на основе анализа сформулирован какой-либо набор правил, методов и подходов к обработке данных.

Так как алгоритм работает с входными данными, необходимо понять: насколько они являются структурированными, являются ли они статистической информацией, содержащей массив исторических данных, который практически неизменен, либо являются потоковыми данными, получаемыми в непрерывном режиме или с некой периодичностью, достаточен ли объем имеющихся данных для проведения исследований или же требуется расширение объёма выборки путём моделирования или аугментаций?

Важным вопросом является определение критериев истинности тех прогнозов, которые будут считаться приемлемыми по результатам работы модели. Любые данные обладают определёнными характеристиками, которые можно подразделить на категории, числовые параметры и производные данные.

Несмотря на развитие достаточно сложных технологий, на которых базируется машинное обучение, в настоящее время актуальной темой является активное внедрение легковесных моделей, которые могут быть запущены даже на слабом аппаратном обеспечении (TinyML).

В настоящее время все подходы в той или иной степени требуют постоянного вмешательства человека в процесс, поэтому хорошим вызовом следующего дня станет развитие таких технологий, которые будут самостоятельно обучаться без присутствия человека.

Умелое применение технологий машинного обучения позволит нам реализовать множество интересных применений, которые могут снизить затраты компании, дать новые возможности.

## Обработка входных данных

Данные - хорошего качества ровно до тех пор, пока они могут удовлетворять требованиям предполагаемого использования. Факторы, влияющие на качество данных, включают точность, полноту, непротиворечивость, своевременность, правдоподобность и интерпретируемость. Прежде чем перейти к шагам по улучшению качества данных, давайте уделим немного времени этому разделу, чтобы понять, что именно мы стремимся изменить.

Точность означает, насколько хорошо записанная информация отражает реальное событие или объект. Неточность данных может быть вызвана неисправными инструментами сбора данных или компьютерными ошибками, преднамеренным представлением пользователями неверных значений данных, ошибками при передаче данных, несоответствиями в соглашениях об именовании и форматах ввода и т.д.

Данные считаются полными, когда присутствуют все обязательные функции. Неполнота данных может возникать из-за отсутствия необходимой информации, неисправностей оборудования во время сбора данных, непреднамеренного удаления или невозможности записи истории или изменений.

Если одна и та же информация, хранящаяся и используемая в нескольких экземплярах, совпадает (с несоответствиями форматирования между различными источниками или хранилищами данных, или без них), то данные непротиворечивы. Количественно она выражается в процентах значений, совпадающих в различных сохраненных экземплярах.

Своевременность также влияет на качество данных, поскольку данные имеют ценность только тогда, когда они доступны по мере необходимости. Если данные устарели или исправления были внесены после оценки или анализа набора данных, это влияет на качество данных.

Достоверность описывает доверие пользователей к данным. Если в какой-либо момент было обнаружено, что данные изобилуют ошибками и несоответствиями, их пользователи, скорее всего, будут иметь оговорки, когда дело дойдет до использования этих данных в будущем. Поскольку это препятствует эффективному использованию данных по назначению, достоверность также является фактором, определяющим качество данных.

Интерпретируемость данных определяет, насколько легко понять информацию, представленную в наборе данных, и извлечь из нее смысл. Доступность для пользователей методик сбора и обработки статистических данных может повлиять на интерпретируемость набора данных.

Эти шесть измерений для оценки качества данных ни в коем случае не являются исчерпывающим списком, они лишь должны обозначить многомерность качества данных. Цель определения параметров качества данных остается прежней - гарантировать, что данные могут удовлетворять требованиям их предполагаемого использования.

Рассмотрим этапы предварительной обработки данных в машинном обучении. Хотя существует несколько различных методов предварительной обработки данных, всю задачу можно разделить на несколько общих, важных шагов: [очистка данных](https://translated.turbopages.org/proxy_u/en-ru.ru.37faa164-65a54546-4f777c27-74722d776562/https/www.projectpro.io/article/data-cleaning-projects/740), интеграция данных, сокращение объема данных и преобразование данных.

1. Очистка данных. Задачи, связанные с очисткой данных, можно дополнительно разделить на:

Сбор данных и метаданных: Данные, хранящиеся в СУБД, будут извлекаться с использованием протоколов ODBC или JDBC. Вы также должны получить метаданные, касающиеся типов полей, ролей и описаний. Если данные содержатся в плоском файле, столбцы необходимо будет разделить с помощью разделителя, чтобы проверить согласованность количества полей.

Обработка пропущенных значений: Существуют различные способы обработки пропущенных значений, а именно:

* Игнорирование кортежа - просто удалите кортежи с недостающей информацией. Это возможно для больших наборов данных, где игнорирование небольшой части кортежей существенно не повлияет на дальнейший анализ.
* Ручное заполнение пропущенных значений может быть чрезвычайно утомительным процессом, но может оказаться необходимым, особенно при небольшом наборе данных.
* Присвоение константы всем пропущенным значениям
* Условное вычисление - выполняется путем замены среднего значения всех выборок, среднего значения выборок аналогичной классификацией или результирующими значениями, или любым другим логическим способом.
* Замена отсутствующего значения наиболее вероятным значением может быть выполнена с использованием байесовской формулы, дерева решений и т.д.

Переформатирование включает в себя внесение изменений в формат данных в стандартный формат, чтобы гарантировать, что такие атрибуты, как дата, повсюду имеют одинаковый формат, выполнение бинирования числовых значений, а также обнаружение и обработку ошибок.

Преобразования атрибутов - поскольку некоторые методы требуют ввода только числовых данных, для обработки двоичных, упорядоченных, многозначных данных необходимо использовать различные стратегии.

Идентификация выбросов и сглаживание зашумленных данных: шум - это случайная ошибка в измеряемой переменной. Некоторые из обычно используемых способов устранения шума или сглаживания данных следующие:

* Группирование - это включает сортировку числовых значений по некоторым ячейкам. Это можно сделать, используя ячейки одинаковой ширины, в результате чего получается однородная сетка, или ячейки равной глубины, которые лучше обрабатывают искаженные данные. Помимо сглаживания, биннинг также используется как метод дискретизации.
* Регрессия - в этом методе сглаживание достигается путем подгонки данных к функциям регрессии.
* Анализ выбросов - одним из методов обнаружения выбросов является кластеризация, при которой значения, которые выходят далеко за рамки набора кластеров, охватывающих точки данных, считаются выбросами и впоследствии удаляются.

2. Интеграция данных - это процесс объединения данных из нескольких источников в единый набор данных. включает в себя интеграцию схемы, то есть интеграцию метаданных из разных источников и разрешение конфликтов значений данных, которые могут возникать из-за различий в единицах измерения, представления и т.д. Кроме того, существует необходимость обрабатывать избыточные данные с помощью таких методов, как корреляционный анализ, чтобы обеспечить хорошее качество данных после интеграции данных.

3. Методы сокращения объема данных направлены на получение уменьшенного представления данных с точки зрения объема при тщательном сохранении целостности исходных данных. Могут быть применены различные стратегии сокращения объема данных:

Уменьшение размерности достигается за счет уменьшения количества атрибутов, которые необходимо учитывать. Некоторые методы уменьшения размерности следующие:

* Вейвлет-преобразования - включает в себя преобразование вектора данных в вектор вейвлет-коэффициентов. Затем эти данные, преобразованные в вейвлет, могут быть усечены для получения сжатого приближения к исходным данным путем сохранения только части самого сильного из вейвлет-коэффициентов.
* Анализ главных компонентов - анализ главных компонентов или PCA работает путем поиска набора ортогональных векторов, который меньше исходных векторов атрибутов, которые могут наилучшим образом представлять данные, что приводит к уменьшению размерности. PCA объединяет исходные атрибуты вальтернативный, меньший набор.
* Выбор подмножества атрибутов - включает в себя выбор набора функций таким образом, чтобы были удалены слабо релевантные или избыточные функции. Вы можете использовать эвристические методы, такие как пошаговый прямой отбор, пошаговое обратное исключение или их комбинацию, а также индукцию дерева решений для получения подмножества атрибутов.

Сокращение количества данных - замена исходных данных более мелкими формами представления данных для достижения уменьшения объема. Для этой цели используются два типа методов:

* Параметрические: регрессионные и логарифмически-линейные модели, параметры которых необходимо сохранять вместо фактических данных.
* Непараметрические методы предполагают хранение данных в виде представлений, таких как гистограммы, кластеры, меньшая выборка исходного набора данных или агрегирование куба данных.

Сжатие данных - применение преобразований для получения сжатого представления исходных данных. В зависимости от того, можно ли выполнить реконструкцию с потерей информации или без нее, этот метод называется сжатием без потерь. Методы уменьшения размерности и количества также считаются формами сжатия данных.

4. Преобразование и дискретизация данных, этот этап включает преобразование данных в форму, подходящую для интеллектуального анализа данных. Сглаживание данных, построение объектов и агрегирование также являются частью этапа преобразования данных. Однако они, вероятно, были выполнены на этапе очистки или сокращения объема данных из-за большого совпадения этапов предварительной обработки данных.

Помимо этого, нормализация, дискретизация и генерация иерархии концепций (последние два из которых также являются частью сокращения данных) считаются частями этого шага. Чтобы минимизировать избыточность, давайте подробно рассмотрим только последние три шага:

* Нормализация - масштабирование атрибутов для обеспечения их попадания в единый диапазон. При отсутствии нормализации атрибуты с большими значениями изначально будут иметь больший вес.
* Дискретизация предполагает уменьшение количества значений непрерывного атрибута путем разделения диапазона атрибутов на интервалы для замены фактических значений данных. Дискретизация может быть выполнена с помощью биннинга, анализа гистограмм, кластеризации, анализа дерева решений и [корреляционного анализа](https://translated.turbopages.org/proxy_u/en-ru.ru.37faa164-65a54546-4f777c27-74722d776562/https/www.projectpro.io/article/correlation-vs-covariance/489).
* Создание иерархии концепций включает в себя сокращение объема данных путем изменения уровня детализации номинальных атрибутов. Например, замена атрибута местоположения, описывающего местоположение в штатах, переменной, описывающей местоположение по странам, уменьшает количество уникальных значений.

После обзора первичного анализа и его методов в практической части работы приступим к непосредственному применению. Для работы рассмотрены данные о марках шоколада, его производителях, местах происхождения какао-бобов и так далее. Тематика выбрана из личных предпочтений и экологической с экономической составляющими вопроса: как известно, из-за климатических изменений и иных факторов некоторые виды сельскохозяйственной продукции ставятся все дороже в производстве, плюс логистика и другие накладные расходы. С целью оптимизации расходов на разработку новых рецептур или изменение существующих попробуем создать модель для определения оптимального компонента из предложенных в нашем датасете, но сначала пройдемся по входной информации.

## Линейная регрессия

Алгоритм существует более 200 лет уже был изучен со всех возможных точек зрения и часто каждый новых угол имеет предназначение и другое название. Линейная регрессия — это линейная модель, которая предполагает линейную связь между входными переменными (Xi) и единственной переменной вывода (Y).

Более конкретно, что у может быть рассчитана через линейную комбинации входных переменных (X) (или У = B1\*X1 + B2\*X2 +... Bn\*Xn).

При наличии одной переменной ввода (x) метод называется простой линейной регрессией. Когда существует несколько переменных входных данных, литература из статистики часто называет метод множественной линейной регрессией. Различные методы могут быть использованы для подготовки или обучения линейной регрессии. Наиболее распространенным из которых называется [Метод наименьших квадратов](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%BD%D0%B0%D0%B8%D0%BC%D0%B5%D0%BD%D1%8C%D1%88%D0%B8%D1%85_%D0%BA%D0%B2%D0%B0%D0%B4%D1%80%D0%B0%D1%82%D0%BE%D0%B2?ref=codecamp.ru) (или сокращенно МНК, по-английски это Ordinary Least Squares или OLS).

Теперь давайте подробнее рассмотрим используемое представление. Линейная регрессия является привлекательной моделью, потому что ее представление очень простое. Представление - это линейное уравнение, объединяющее определенный набор входных значений (X) решений, к которому является прогнозируемый вывод для этого набора входных значений (y). Таким образом, как значения входных данных (x), так и выходное значение являются числовыми.

Линейное уравнение присваивает масштабный коэффициент (по-английски "scale factor") к каждому входному значению X. Масштабный коэффициент представлен греческой буквой Beta (B). Добавлен также один дополнительный коэффициент, дополнительная степень свободы (например, движение вверх и вниз по двумерном участку) и часто называют коэффициентом перехвата или смещения (по-английски "bias coefficient").

Наиболее простая задача регрессии - когда на вход подается одна переменная X и есть одно выходящее значение Y. Форма подобной модели будет: Y = B0 + B1 \* X

В случае многомерных измерений (т.е. когда у нас есть более одной вводной переменной (X)), линия превращается в плоскостью или гиперплоскости. Таким образом, представление представляет собой форму уравнения и конкретные значения, используемые для коэффициентов (например, B0 и B1 в приведенном выше примере).

Когда говорят о сложности регрессионной модели, такой как линейная регрессия - имеют ввиду количество коэффициентов, используемых в модели.

Когда конкретный элемент коэффициент Beta становится нулевым, он эффективно удаляет влияние входной переменной на модель и, следовательно, влияния на прогноз модели (0 \* Xi = 0). Это становится актуальным, если вы применяете методы регуляризации (о них мы расскажем отдельно), которые изменяют алгоритм обучения, чтобы уменьшить сложность моделей регрессии, оказывая давление на абсолютный размер коэффициентов, приводя некоторые из них к нулю.

Теперь, когда мы понимаем, что такое представление, используемое для модели линейной регрессии, давайте рассмотрим некоторые способы, с помощью которых мы можем узнать это представление из данных, методы линейной регрессии. Изучение модели линейной регрессии означает исследование получаемых значений коэффициентов, используемых в представлении, на основе имеющихся входных данных.

Важно обратить внимание что на метод наименьших квадратов, потому что это наиболее распространенный метод, используемый в целом в индустрии для задач оптимизации. Также обратите внимание метод Градиентного спуска (по-английски Gradient descent), как наиболее распространенный метод применяемый в различных классах задач машинного обучения.

При простой линейной регрессии, когда у нас есть один входной параметр, мы можем использовать статистику для оценки коэффициентов. Для этого необходимо вычислить статистические свойства на таких данных, как среднее значение, стандартные отклонения, корреляции и ковариантность. Все данные должны быть доступны для обхода и расчета статистик.

Когда у нас есть более одной входной переменной, мы можем использовать метод наименьших квадратов для оценки значений коэффициентов. Процедура наименьших квадратов направлена на минимизацию суммы квадратов остатков. Это означает, что, учитывая регрессионную линию через данные, мы вычисляем расстояние от каждой входной точки Xi до линии регрессии, берем квадрат этого расстояние и суммируем все квадраты подобных расстояний. Это количество, которое обычные наименее квадратов стремится свести к минимуму.

Такой подход рассматривает данные как матрицу и использует операции линейной алгебры для оценки оптимальных значений коэффициентов. Это означает, что все данные должны быть доступны, и вы должны иметь достаточно памяти, чтобы соответствовать размеру датасета и выполнять матричные операции.

Руками подобные вычисления уже давно никто не делает (кроме как для собственного понимания или в рамках домашнего задания в школе или ВУЗе). Вы просто вызываете процедуру из библиотеки линейной алгебры. Эта процедура умеет быстро вычислять подобные задачи.

При наличии одной или нескольких переменных можно использовать процесс оптимизации значений коэффициентов путем итеративной минимизации ошибки модели на обучающихся данных. Эта операция называется [Градиентный спуск](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D0%B4%D0%B8%D0%B5%D0%BD%D1%82%D0%BD%D1%8B%D0%B9_%D1%81%D0%BF%D1%83%D1%81%D0%BA?ref=codecamp.ru) и работает, начиная со случайных значений для каждого коэффициента.

По аналогии с методов наименьших квадратов - мы ищем сумму ошибок в квадрате рассчитывается для каждой пары входных и выходных значений. В качестве масштабного коэффициента в градиентном спуске используется частота обучения (по-английски "learn rate"), а коэффициенты обновляются в направлении минимизации ошибки. Процесс повторяется до тех пор, пока не будет достигнута ошибка в квадрате минимальной суммы или невозможно дальнейшее улучшение.

При использовании этого метода необходимо выбрать параметр скорости обучения (альфа), который определяет размер шага улучшения, чтобы взять на себя каждую итерацию процедуры. На практике градиентный спуск является полезным методом, когда у вас очень большой датасет.

Есть расширения обучения линейной модели, называемой методами регуляризации. Они направлены как на минимизацию суммы квадратов ошибки модели на обучающих данных (с использованием метода наименьших квадратов), но и на снижения сложности модели (например, количество или абсолютный размер суммы всех коэффициентов в модели).

Учитывая, что представление является линейным уравнением, сделать прогнозы так же просто, как решение уравнения для определенного набора входов.

Рассмотрим конкретный пример. Представьте, что мы прогнозируем вес человека (y) в зависимости от высоты человека (x). Наше представление модели линейной регрессии для этой проблемы будет:

Y = B0 + B1 \* X1 или Вес человека = B0 + B1 \* Высота человека,

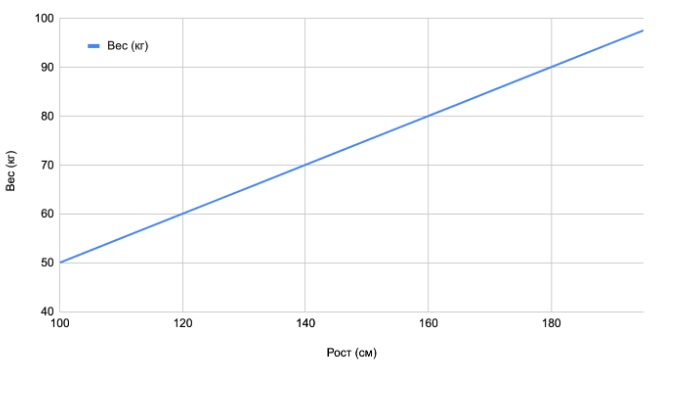
где B0 является коэффициентом смещения, а B1 - коэффициентом для столбца высоты человека. Мы используем технику обучения, чтобы найти хороший набор значений коэффициентов. После того, как найдено, мы можем подключить различные значения высоты, чтобы предсказать вес.

Например, позволяет использовать B0 = 0,1 и B1 = 0,5. Давайте подставим их и рассчитаем вес (в килограммах) для человека с ростом 182 сантиметра.

вес человека = 0,1 + 0,5 \* 182

вес человека = 91.1

Вышеупомянутое уравнение может быть отображена как линия в двух измерениях. Коэффициент B0 является нашей отправной точкой независимо от того, какой рост у человека. Мы можем пробежать через различные высоту человека от 100 до 200 сантиметров подставив в уравнение и получить значения веса, создавая нашу линию.



Теперь рассмотрим некоторые правила подготовки данных, чтобы получить максимум от этого типа модели. Линейная регрессия изучается уже давно, и есть много литературы о том, как ваши данные должны быть структурированы, чтобы наилучшим образом использовать модель МНК или Градиентного спуска.

Линейные предпосылки. Линейная регрессия предполагает, что связь между входными и выходным данными является линейной. Линейная регрессия не поддерживает ничего другого. Это может быть очевидно, но это хорошо, чтобы помнить, когда у вас есть много атрибутов. Возможно, потребуется преобразовать данные, чтобы сделать отношения между ними линейными (например, логарифмическое преобразование для экспоненциальной связи).

Удаление шума - линейная регрессия предполагает, что переменные на выходе и вывода не являются шумными. Рассмотрите возможность использования операций по очистке данных, которые позволяют лучше прояснять данные. Это наиболее важно для переменной вывода.

Удаление коллинеарности - линейная регрессия будет чрезмерно соответствовать вашим данным, когда у вас есть сильно коррелированные входные переменные.

Гауcсово распределение - линейная регрессия сделает более надежные прогнозы, если входные и выходные переменные имеют гауссово распределение. Можно получить некоторую выгоду с помощью преобразований (например, log или BoxCox) на переменных, чтобы сделать их распределение более гауссово.

Нормализация входных данных - линейная регрессия часто делает более надежные прогнозы, если отмасштабировать входные переменные с помощью стандартизации или нормализации.

Во второй главе мы приступим к практической части раздела и попробуем создать модель для прогнозирования веса рыбы. Логично предположить организацию на предприятии системы компьютерного зрения, которая будет с конвеера считывать данные параметров каждой тушки (длина, ширина и др.), и на основе этих данных будет определяться вес.

В рыбной промышленности крайне важно сокращать время от вылова до обработки и применения в пищу, так как от этого очень зависит качество. Стоит лишь немного замешкаться – партию можно утилизировать из-за повышения содержания микроорганизмов.

## Методы классификации в Python

Прежде чем мы продолжим наше исследование, уделим минуту определению терминов. Важно иметь представление о лексике, которая будет использоваться. Система машинного обучения или сеть принимает входы и выходы. Входные данные в структуру машинного обучения часто называют [“функциями”](https://en.wikipedia.org/wiki/Feature_(machine_learning)). Признаки, по существу, те же, что и переменные в научном эксперименте, они являются характеристиками наблюдаемого явления, которые могут быть количественно определены или измерены каким-либо образом.

Когда эти функции вводятся в структуру машинного обучения, сеть пытается различить соответствующие паттерны между этими функциями. Эти шаблоны затем используются для генерации выходных данных фреймворка/сети. Выходы фреймворка часто называют “метками”, поскольку выходные объекты имеют некоторую метку, данную им сетью, некоторое предположение о том, к какой категории относится выход.

В контексте машинного обучения классификация-это тип [контролируемого обучения](https://dataconomy.com/2015/01/whats-the-difference-between-supervised-and-unsupervised-learning/). Контролируемое обучение означает, что данные, подаваемые в сеть, уже помечены, а важные функции/атрибуты заранее разделены на отдельные категории.

Это означает, что сеть знает, какие части входных данных важны, и существует также целевая или основная истина, которую сеть может проверить сама. Примером классификации является сортировка группы различных растений по различным категориям, таким как *папоротники* или *покрытосеменные*.

В отличие от этого, неконтролируемое обучение-это когда данные, подаваемые в сеть, не маркированы, и сеть должна попытаться узнать для себя, какие функции являются наиболее важными. Как уже упоминалось, классификация-это тип контролируемого обучения, и поэтому мы не будем рассматривать в этой работе методы неконтролируемого обучения.

Процесс обучения модели – это процесс подачи данных в нейронную сеть и предоставления ей возможности изучать паттерны данных. Процесс обучения принимает данные и извлекает особенности набора данных. В процессе обучения для контролируемой задачи классификации в сеть передаются как признаки, так и метки обучающих данных. Однако во время тестирования в сеть подаются только функции.

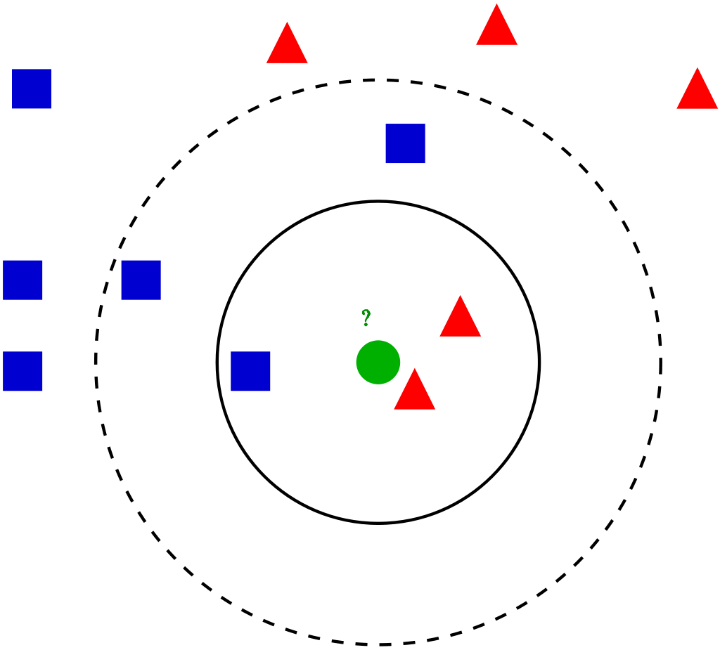
Процесс тестирования-это когда тестируются шаблоны, которые изучила сеть. Характеристики даны сети, и сеть должна предсказать метки. Данные для сети делятся на обучающие и тестовые наборы, два разных набора входных данных. Вы не тестируете классификатор на том же наборе данных, на котором вы его тренируете, так как модель уже изучила паттерны этого набора данных, и это было бы крайним смещением.

Вместо этого набор данных разбивается на обучающие и тестовые наборы, набор, на котором обучается классификатор, и набор, который классификатор никогда раньше не видел.

Scikit-Learn обеспечивает легкий доступ к многочисленным различным алгоритмам классификации. К числу таких классификаторов относятся:

* K-Ближайшие Соседи
* Машины опорных Векторов
* Классификаторы дерева решений / Случайные леса
* Наивный Байес
* Линейный дискриминантный анализ
* Логистическая регрессия

K-Ближайшие соседи

K-Ближайшие соседи оперирует проверкой расстояния от некоторого тестового примера до известных значений некоторого обучающего примера. Выбрана группа точек данных/класс, которая даст наименьшее расстояние между точками обучения и точкой тестирования.

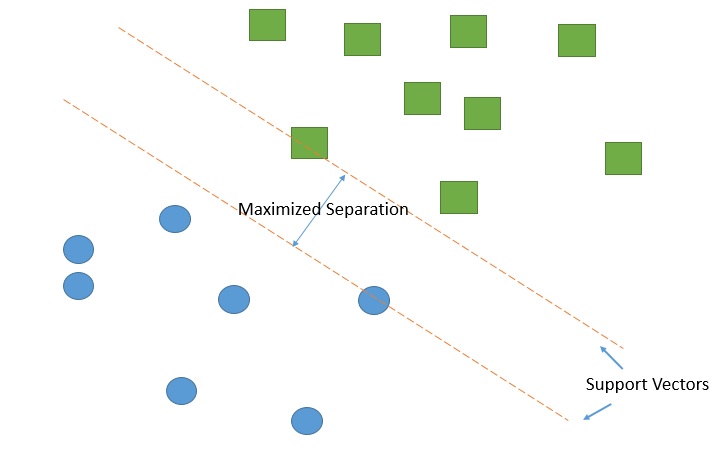
Классификатор дерева решений функционирует путем разбиения набора данных на все меньшие и меньшие подмножества на основе различных критериев. Для разделения набора данных будут использоваться различные критерии сортировки, причем количество примеров будет уменьшаться с каждым делением.

Как только сеть разделит данные до одного примера, пример будет помещен в класс, соответствующий ключу. Когда несколько случайных лесных классификаторов связаны вместе, они называются *Случайными лесными классификаторами*.

Наивный байесовский классификатор определяет вероятность того, что пример принадлежит к некоторому классу, вычисляя вероятность того, что событие произойдет, учитывая, что произошло некоторое входное событие. При выполнении этого расчета предполагается, что все предикторы класса оказывают одинаковое влияние на результат, что предикторы независимы.

Линейный дискриминантный анализ работает путем уменьшения размерности набора данных, проецируя все точки данных на линию. Затем он объединяет эти точки в классы на основе их расстояния от выбранной точки или центроида. Линейный дискриминантный анализ, как можно догадаться, является линейным алгоритмом классификации и лучше всего используется, когда данные имеют линейную зависимость.

Машины опорных Векторов



Машины опорных векторов работают, проводя линию между различными кластерами точек данных, чтобы сгруппировать их в классы. Точки на одной стороне линии будут принадлежать одному классу, а точки на другой стороне-другому. Классификатор попытается максимально увеличить расстояние между линией, которую он рисует, и точками по обе стороны от нее, чтобы повысить свою уверенность в том, какие точки принадлежат к какому классу. Когда тестовые точки строятся, сторона линии, на которую они попадают, – это класс, в который они попадают.

Логистическая регрессия выводит предсказания о точках тестовых данных в двоичной шкале, нулевой или единице. Если значение чего-либо равно 0,5 или выше, оно классифицируется как принадлежащее к классу 1, в то время как ниже 0,5, если оно классифицируется как принадлежащее к классу 0.

Каждая из функций также имеет метку только 0 или 1. Логистическая регрессия является линейным классификатором и поэтому используется, когда существует какая-то линейная связь между данными.

Задачи классификации - это любые задачи, в которые помещены примеры в два или более классов. Определение того, является ли изображение кошкой или собакой, является классификационной задачей, как и определение того, какое качество бутылки вина основано на таких признаках, как кислотность и содержание алкоголя.

В зависимости от поставленной задачи классификации можно использовать различные классификаторы. Например, модель логистической регрессии лучше всего подходит для задач бинарной классификации, даже если существуют модели логистической регрессии с несколькими переменными. Однако распространенной практикой является создание экземпляров нескольких классификаторов и сравнение их производительности друг с другом, а затем выбор классификатора, который работает лучше всего.

Когда дело доходит до оценки классификатора, есть несколько различных способов измерить его производительность.

Классификация точности является самым простым из всех методов оценки точности и наиболее часто используемым. Точность классификации - это просто число правильных предсказаний, деленное на все предсказания, или отношение правильных предсказаний к общему числу предсказаний. Хотя он может дать вам быстрое представление о том, как работает ваш классификатор, его лучше всего использовать, когда количество наблюдений/примеров в каждом классе примерно эквивалентно. Поскольку это происходит не очень часто, вам, вероятно, лучше использовать другую метрику.

[Логарифмическая потеря](https://datawookie.netlify.com/blog/2015/12/making-sense-of-logarithmic-loss/), по существу, оценивает, насколько уверен классификатор в своих прогнозах. LogLoss возвращает вероятности принадлежности примера к данному классу, суммируя их вместе, чтобы дать представление об общей уверенности классификатора. Значение для предсказаний колеблется от 1 до 0, причем 1 является полностью уверенным, а 0-нет уверенности. Потеря или общее отсутствие уверенности возвращается в виде отрицательного числа с 0, представляющего идеальный классификатор, поэтому меньшие значения лучше.

Площадь Под Кривой ROC (AUC) - эта метрика используется только для задач бинарной классификации. Область под кривой представляет способность модели правильно различать отрицательные и положительные примеры, между тем или иным классом. Вся область, попадающая под кривую, представляет собой идеальный классификатор. Это означает, что AUC 0,5 в основном так же хорош, как и случайное угадывание. Кривая ROC рассчитывается с учетом чувствительности (истинная положительная скорость/отзыв) и специфичности (истинная отрицательная скорость).

Матрица путаницы - это таблица или диаграмма, представляющая точность модели по отношению к двум или более классам. Предсказания модели будут располагаться на оси X, а результаты/точность-на оси y. Ячейки заполняются количеством прогнозов, которые делает модель. Правильные предсказания можно найти на диагональной линии, движущейся от верхнего левого угла к нижнему правому.

Отчет о классификации - это встроенная метрика Scikit-Learn, созданная специально для задач классификации. Использование отчета о классификации может дать вам быстрое представление о том, как работает ваша модель. Напомним, что количество примеров вашей модели, помеченной как класс А (некоторый данный класс), сравнивается с общим количеством примеров класса А, и это представлено в отчете. Отчет также возвращает прогноз и f1-балл. Точность-это процент примеров, которые ваша модель обозначила как класс А, который на самом деле принадлежал классу А (истинные положительные результаты против ложных положительных результатов), а f1-оценка-это среднее значение точности и отзыва.

Ну а теперь можно, наконец-таки, приступить к практической части данной работы и применить указанные выше методики и приемы.

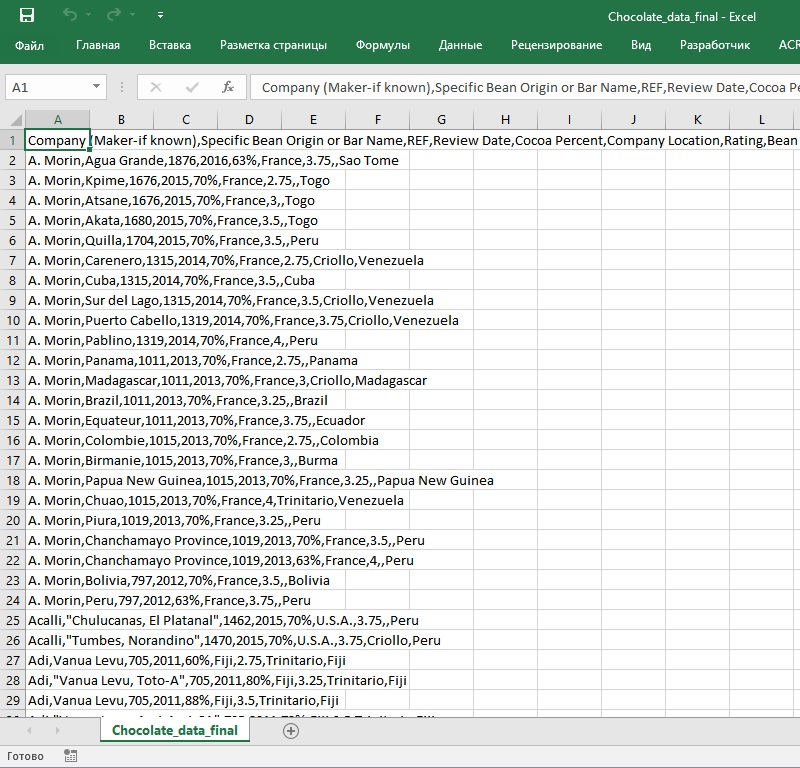
Глава 2

## Задача 1 - получение данных

1. Импорт библиотеки Pandas – это библиотека Python с открытым исходным кодом для специализированного анализа данных. Сегодня все, кто использует Python для изучения статистических целей анализа и принятия решений, должны быть с ней знакомы. Библиотека была спроектирована и разработана преимущественно Уэсом Маккини в 2008 году. В 2012 к нему присоединился коллега Чан Шэ. Вместе они создали одну из самых используемых библиотек в сообществе Python.

import pandas as pd

1. Использование библиотеки Pandas для считывания данных из CSV файла Chocolate\_data\_final.csv



Данный формат (Comma-Separated Values) — текстовый формат, предназначенный для представления табличных данных. Строка таблицы соответствует строке текста, которая содержит одно или несколько полей, разделённых запятыми.

df = pd.read\_csv('Chocolate\_data\_final.csv')

1. Вывод на экран первых 10 строк таблицы с данными. Как правило, таблица очень велика, практично запросить определённое количество, от 10 до 20 её первых строк.

df.head(11)

1. Продолжаем знакомство с данными, осуществим получение списка колонок и сохранение его в отдельную переменную.

df.columns

1. Вывод названия каждой колонки списка, а также уникальных значений, которые содержатся в ней поможет нам понять, с чем именно предстоит работать, а также, будет ли достаточно подобного набора информации для решения поставленной перед нами задачи.

columns = [i for i in df.columns]

columns

for i in df.columns:

print(i, df[i].unique())

1. Применение функции для подсчёта количества нулевых значений NAN в каждом столбце. Или же подсчет числа пропущенных значений с помощью встроенных методов библиотеки Pandas.

for i in df.columns:

print(i, df[i].unique(), df[i].isna().sum())

df.isna().sum()

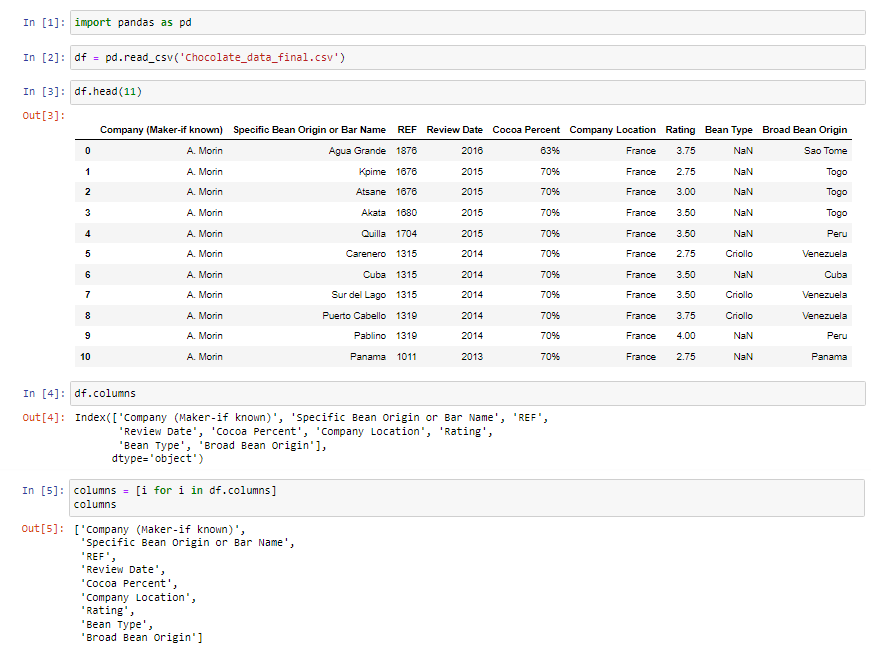
count\_NA = 0

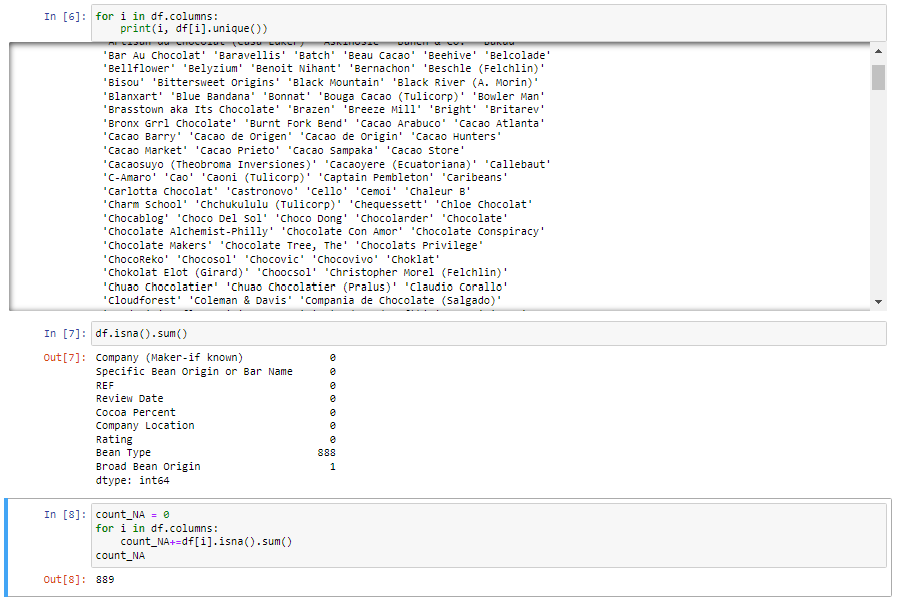
for i in df.columns:

count\_NA+=df[i].isna().sum()

count\_NA

### Решение задачи 1:





В данном разделе проекта нам удалось познакомиться с первоначальными данными файла, содержащего различную информацию о нескольких марках шоколада. Мы уточнили, по каким категориям были представлены данные, насколько полно они отражены и выявили уникальные значения для понимания наших следующих шагов по работе с предоставленной информацией.

Выполненный этап является обязательным и простым элементом при работе с данными, но необходимым. Далее мы приступим к более детальной обработке информации и подготовки её для основных этапов проекта.

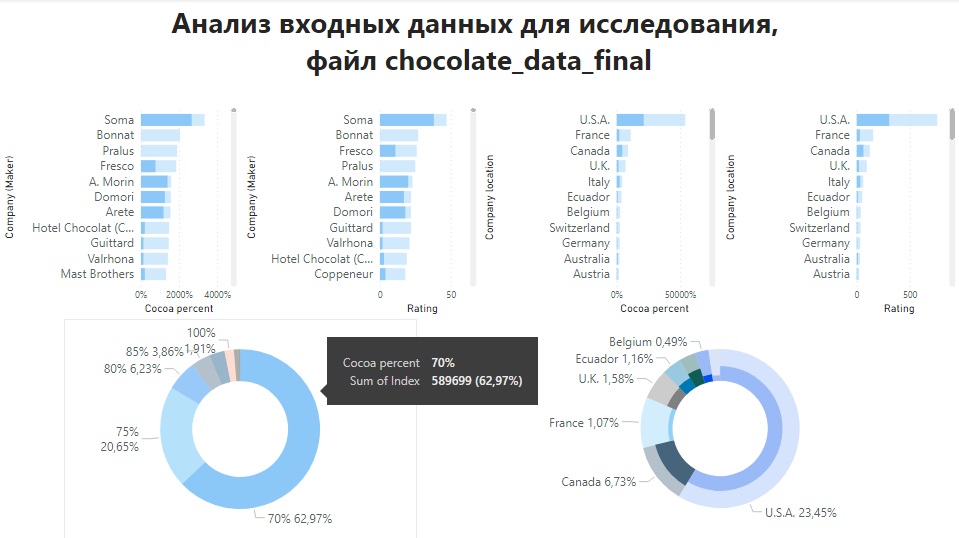
## Задача 2 - первичный анализ данных

1. Импорт библиотеки Pandas.

import pandas as pd

1. Использование библиотеки Pandas для считывания данных из CSV файла (файл - Chocolate\_data\_final.csv).

В данной таблице представлены данные по различным маркам шоколада, используемого в пищевой промышленности. Благодаря найденным зависимостям параметров мы и будем иметь возможность обучить нашу модель. Для общего представления об имеющейся информации можно посмотреть на неё и через аналитические инструменты:



df = pd.read\_csv('Chocolate\_data\_final.csv')

1. Применение функции describe() для описания датасета. Метод выводит сводную статистику всех числовых столбцов, такую как количество, среднее значение, стандартное отклонение, диапазон и квартили. info() -это быстрый способ посмотреть типы данных, пропущенные значения и размер данных в DataFrame.

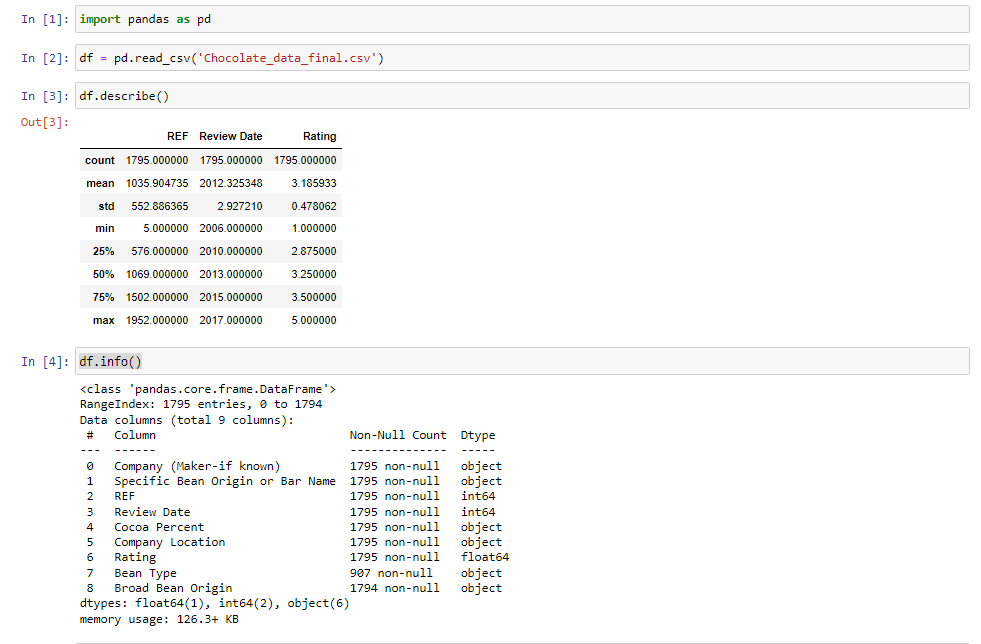
df.describe()

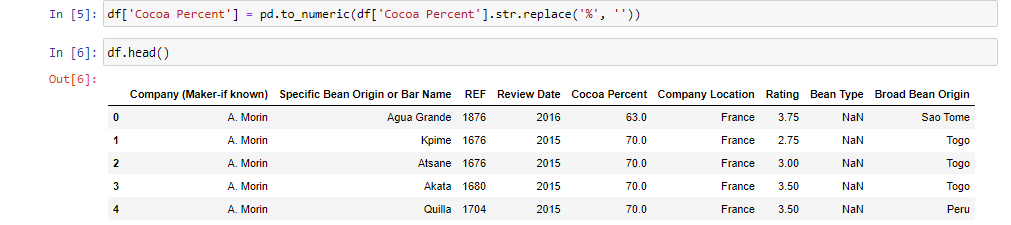
df.info()

1. Поиск в данных колонок с числовыми значениями (float/integer).
2. Приведение данных столбца «Cocoa Percent» в числовой формат, для этого необходимо выполнить два действия:

* удалить знак процентов;
* преобразовать оставшееся строковое значение к числовому.

### Решение задачи 2:





Продемонстрированные методы ни в коем случае не являются полным набором операций предварительной обработки, и в нашем случае являются лишь достаточными для решения нашей задачи. Может показаться, что больше данных – лучше, но цитируя слова американского компьютерщика и директора по исследованиям Google, LLC Питера Норвига:

“Большее количество данных превосходит умные алгоритмы, но более качественные данные превосходят большее количество данных.”

Независимо от того, сколько данных или необычных моделей мы можем использовать, предварительная обработка данных может сделать их лучше.

## Задача 3 - обучение регрессионной модели предсказания массы рыбы

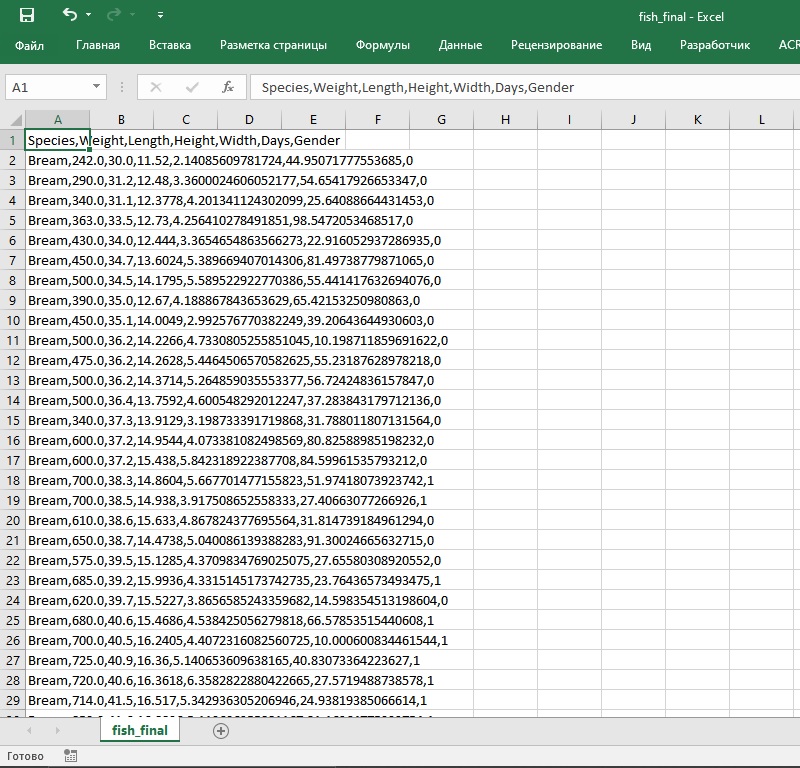
1. Загрузить необходимые библиотеки.

import pandas as pd

import numpy as np

from sklearn import linear\_model

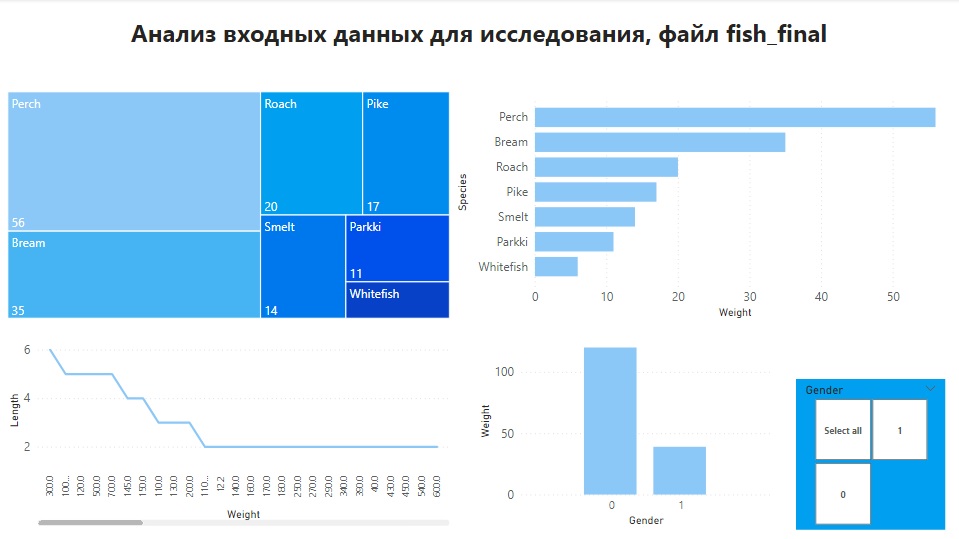
1. Загрузить файл fish\_final.csv.



df = pd.read\_csv('fish\_final.csv')

df.head(3)

При первичном поверхностном анализе можно выявить несколько зависимостей, которые помогут решить нашу задачу.



1. Установить переменную RS (random\_state = 7092022).

RS = 7092022

1. Разделить данные на 2 выборки (тренировочную и тестовую с параметрами: test\_size=0.2, random\_state = RS).

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state = RS)

1. Построить матрицу корреляций: оценить данные и выбрать признаки для использования в модели.

import matplotlib.pyplot as plt

corr = df.corr()

import seaborn as sns

plt.figure(figsize=(10, 3))

sns.heatmap(corr,

xticklabels=corr.columns,

yticklabels=corr.columns, annot=True, linewidths=1, linecolor='black')

plt.show()

1. Использовать масштабирование данных (X) при помощи StandartScaller().

from sklearn.preprocessing import StandardScaler

sc\_x = StandardScaler()

sc\_y = StandardScaler()

X\_train\_scalled = sc\_x.fit\_transform(X\_train)

X\_test\_scalled = sc\_x.transform(X\_test)

y\_train\_scalled = sc\_y.fit\_transform(np.array(y\_train).reshape(-1, 1))

y\_test\_scalled = sc\_y.transform(np.array(y\_test).reshape(-1, 1))

pd.DataFrame(X\_train\_scalled).head()

1. Построить модель множественной линейной регрессии.

from sklearn.metrics import mean\_absolute\_error

reg = linear\_model.LinearRegression()

reg.fit(X\_train, y\_train)

reg.coef\_

1. Для оценки качества модели использовать метрику скорректированного R^2 (Adj\_r2).

def regression\_results(y\_true, y\_pred,n,p):

# Regression metrics

explained\_variance=metrics.explained\_variance\_score(y\_true, y\_pred)

mean\_absolute\_error=metrics.mean\_absolute\_error(y\_true, y\_pred)

mse=metrics.mean\_squared\_error(y\_true, y\_pred)

median\_absolute\_error=metrics.median\_absolute\_error(y\_true, y\_pred)

r2=metrics.r2\_score(y\_true, y\_pred)

print('Adj\_r2: ', round(1-(1-r2)\*(n-1)/(n-p-1),4))

print('r2: ', round(r2,4))

print('MAE: ', round(mean\_absolute\_error,4))

print('MSE: ', round(mse,4))

print('RMSE: ', round(np.sqrt(mse),4))

regression\_results(y\_test, y\_pr, X\_test\_scalled.shape[0], X\_test\_scalled.shape[1])

1. Вывод коэффициентов при модели (только при переменных) - метод.coef\_ (применяется к обученной модели).

import statsmodels.api as sm

import statsmodels.formula.api as smf

X\_train\_scalled = sm.add\_constant(X\_train\_scalled)

X\_test\_scalled = sm.add\_constant(X\_test\_scalled)

model = sm.OLS(y\_train\_scalled, X\_train\_scalled).fit()

y\_pr\_scalled = model.predict(X\_test\_scalled)

y\_pr = sc\_y.inverse\_transform(np.array(y\_pr\_scalled).reshape(-1, 1))

print\_model = model.summary()

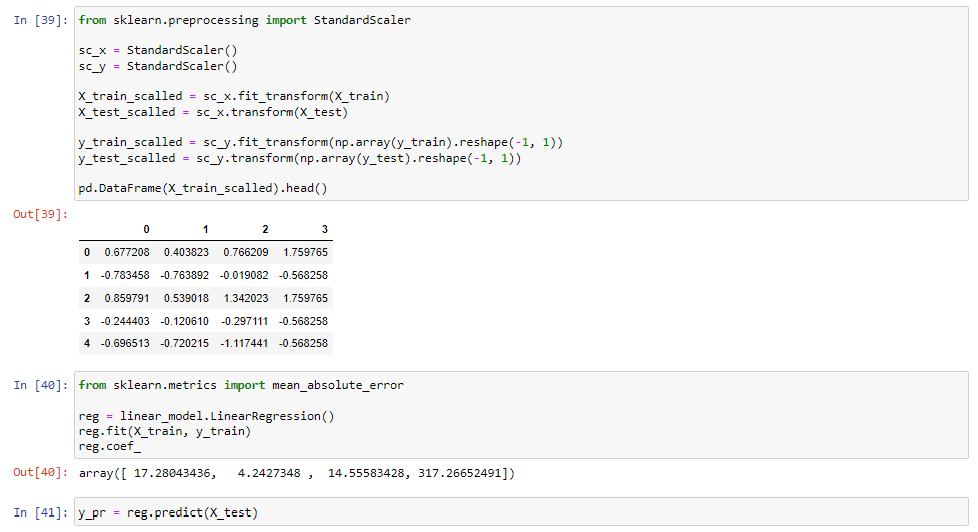
print(df.columns)

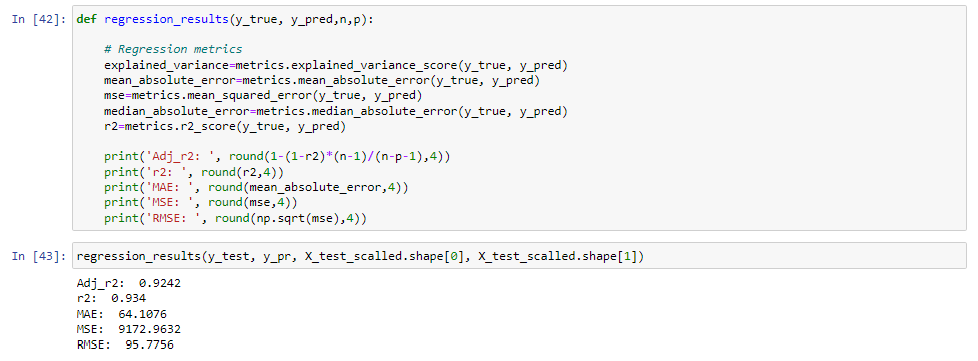
print(print\_model)

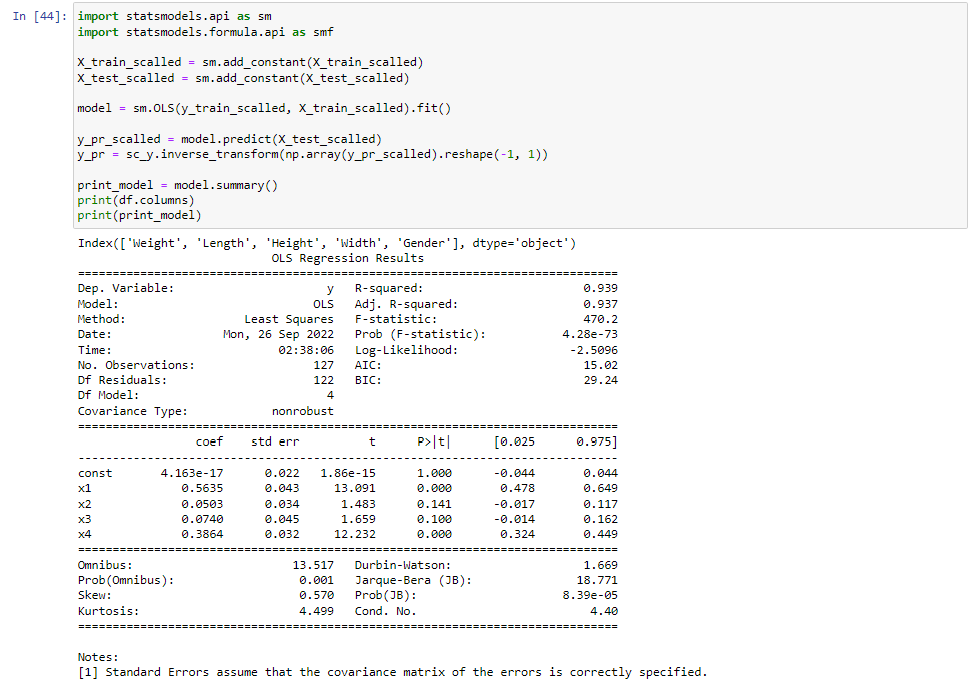
### Решение задачи 3:











В текущем разделе нам удалось провести аналитику содержания наших данных и выявить наиболее важную для нас информацию – параметры для модели. Все данные мы делим на две части, одну будем использовать для обучения модели множественной линейной регрессии, а вторая будет рабочей.

При создании моделей большое внимание стоит уделять их качеству, для чего проводится оценка различных метрик. В нашем случае полученные коэффициенты вполне приемлемы, что позволяет решить, что модель создана удачно и может быть применена в реальных производственных условиях.

## Задача 4 - обучение модели классификации для предсказания качества разрабатываемой рецептуры шоколада

1. Использование набора данных Chocolate\_data\_final\_cl.csv.

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

from matplotlib.pylab import rc, plot

RS = 7092022

df = pd.read\_csv("Chocolate\_data\_final\_cl.csv")

df.head()

df.isnull().sum()

df = df.drop(df.columns[[0]], axis=1)

df = df.drop('Specific Bean Origin or Bar Name', axis=1)

df = df.drop('Company (Maker-if known)', axis=1)

df = df.drop('Bean Type', axis=1)

df = df.drop('Broad Bean Origin', axis=1)

df = df.drop(df.columns[[0]], axis=1)

df = df.drop('Specific Bean Origin or Bar Name', axis=1)

df = df.drop('Company (Maker-if known)', axis=1)

df = df.drop('Bean Type', axis=1)

df = df.drop('Broad Bean Origin', axis=1)

from sklearn.preprocessing import StandardScaler

from sklearn.preprocessing import LabelEncoder, OneHotEncoder

ohe = OneHotEncoder(sparse=False)

encoded\_state = ohe.fit\_transform(df['Company Location'].values.reshape(-1, 1))

tmp = pd.DataFrame(encoded\_state,

columns=['Company Location' + str(i) for i in range(encoded\_state.shape[1])])

df = pd.concat([df, tmp], axis=1)

df = df.drop('Company Location', axis=1)

1. Обучение различных классификаторов.

import itertools

from sklearn.metrics import confusion\_matrix, roc\_curve,roc\_auc\_score

from sklearn.model\_selection import cross\_val\_score

from sklearn.model\_selection import train\_test\_split

from sklearn.model\_selection import cross\_val\_score

from sklearn.linear\_model import LogisticRegression

X = df.drop(['Rating'], axis = 1)

y = df['Rating']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, stratify=y, test\_size=0.3, random\_state=RS)

sc\_x = StandardScaler()

X\_train\_scalled = sc\_x.fit\_transform(X\_train)

X\_test\_scalled = sc\_x.transform(X\_test)

from sklearn.linear\_model import LogisticRegression

from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier

from sklearn.metrics import precision\_recall\_curve, classification\_report

lr = LogisticRegression(random\_state=RS)

lr.fit(X\_train\_scalled, y\_train)

1. Построение матрицы ошибок.

def plot\_confusion\_matrix(cm, classes,

normalize=False,

title='Confusion matrix',

cmap=plt.cm.Blues):

"""

This function prints and plots the confusion matrix.

Normalization can be applied by setting `normalize=True`.

"""

plt.imshow(cm, interpolation='nearest', cmap=cmap)

plt.title(title)

plt.colorbar()

tick\_marks = np.arange(len(classes))

plt.xticks(tick\_marks, classes, rotation=45)

plt.yticks(tick\_marks, classes)

if normalize:

cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]

print("Normalized confusion matrix")

else:

print('Confusion matrix, without normalization')

print(cm)

thresh = cm.max() / 2.

for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):

plt.text(j, i, cm[i, j],

horizontalalignment="center",

color="white" if cm[i, j] > thresh else "black")

plt.tight\_layout()

plt.ylabel('True label')

plt.xlabel('Predicted label')

font = {'size' : 15}

plt.rc('font', \*\*font)

cnf\_matrix = confusion\_matrix(y\_test, lr.predict(X\_test\_scalled))

plt.figure(figsize=(10, 8))

plot\_confusion\_matrix(cnf\_matrix, classes=['Non-churned', 'Churned'],

title='Confusion matrix')

plt.savefig("conf\_matrix.png")

plt.show();

1. Расчет метрик:

print("ROC\_AUC", roc\_auc\_score(y\_test, lr.predict\_proba(X\_test\_scalled)[:,1]))

report = classification\_report(y\_test, lr.predict(X\_test\_scalled),\

target\_names=['Non-churned', 'Churned'])

print(report)

from sklearn import tree

clf = tree.DecisionTreeClassifier(random\_state=RS)

clf.fit(X\_train\_scalled, y\_train)

tree.plot\_tree(clf);

clf2 = GradientBoostingClassifier(random\_state=RS)

clf2.fit(X\_train\_scalled, y\_train)

cnf\_matrix = confusion\_matrix(y\_test, clf2.predict(X\_test\_scalled))

plt.figure(figsize=(10, 8))

plot\_confusion\_matrix(cnf\_matrix, classes=['Non-churned', 'Churned'],

title='Confusion matrix')

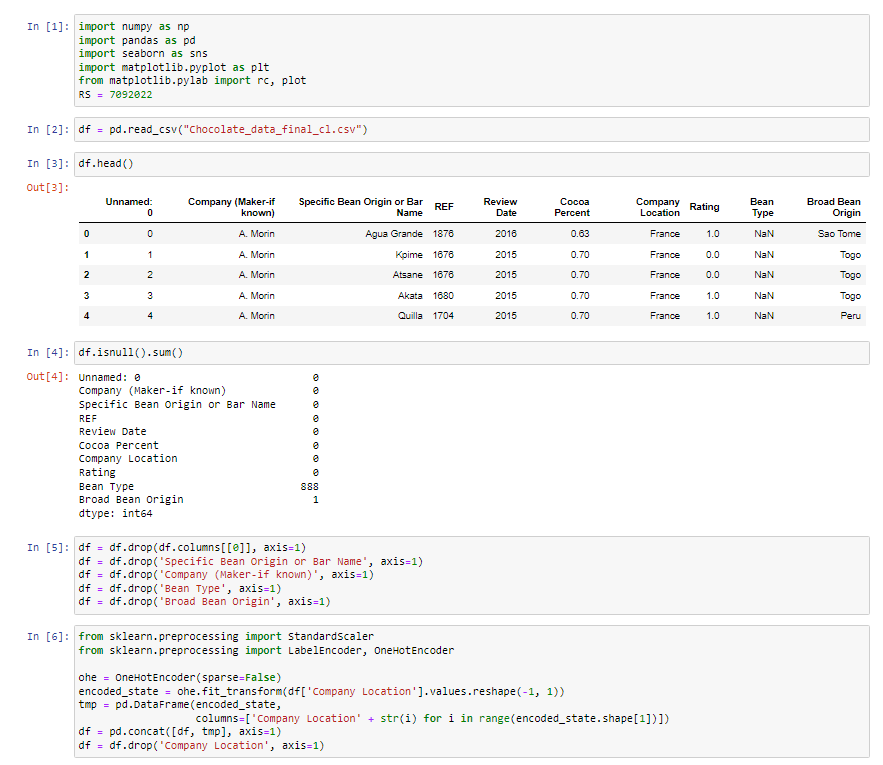
print("ROC\_AUC", roc\_auc\_score(y\_test, lr.predict\_proba(X\_test\_scalled)[:,1]))

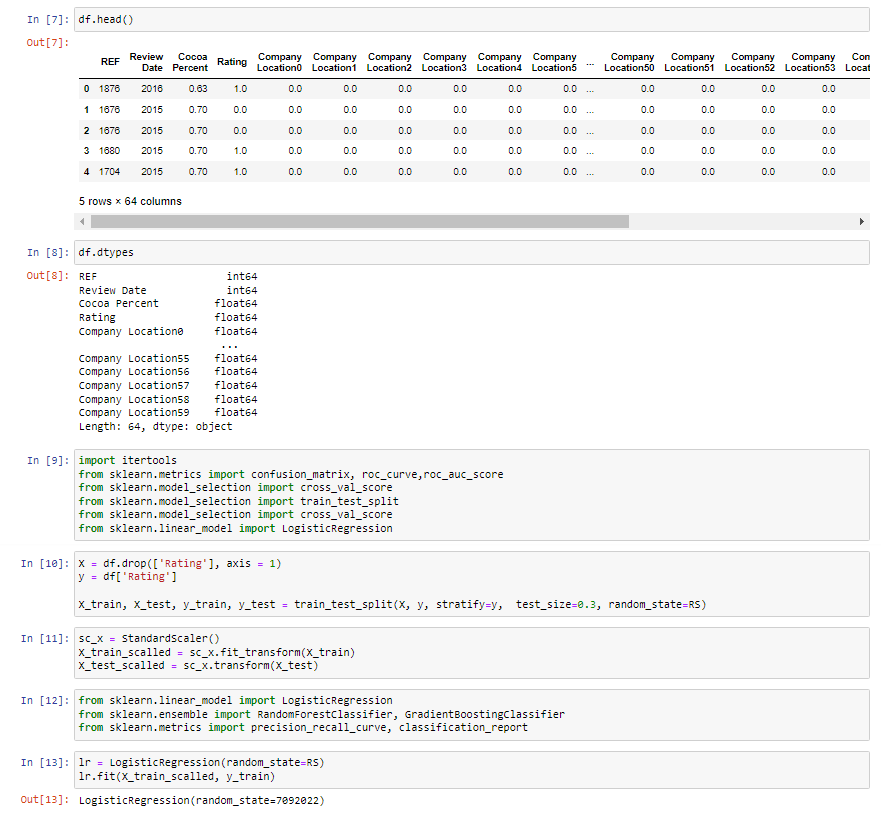
report = classification\_report(y\_test, lr.predict(X\_test\_scalled),\

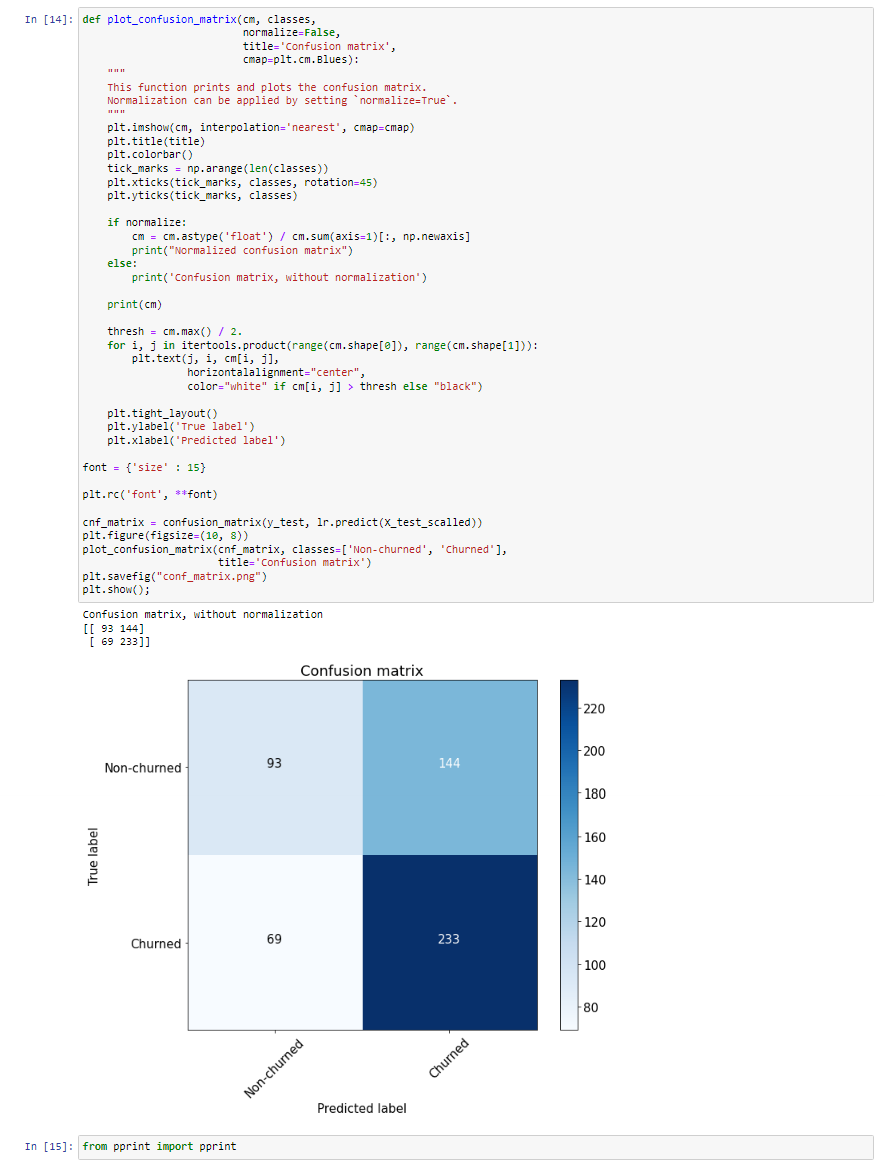
target\_names=['Non-churned', 'Churned'])

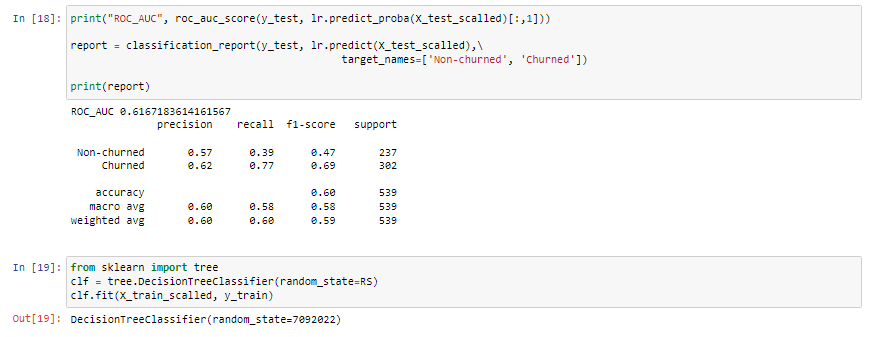
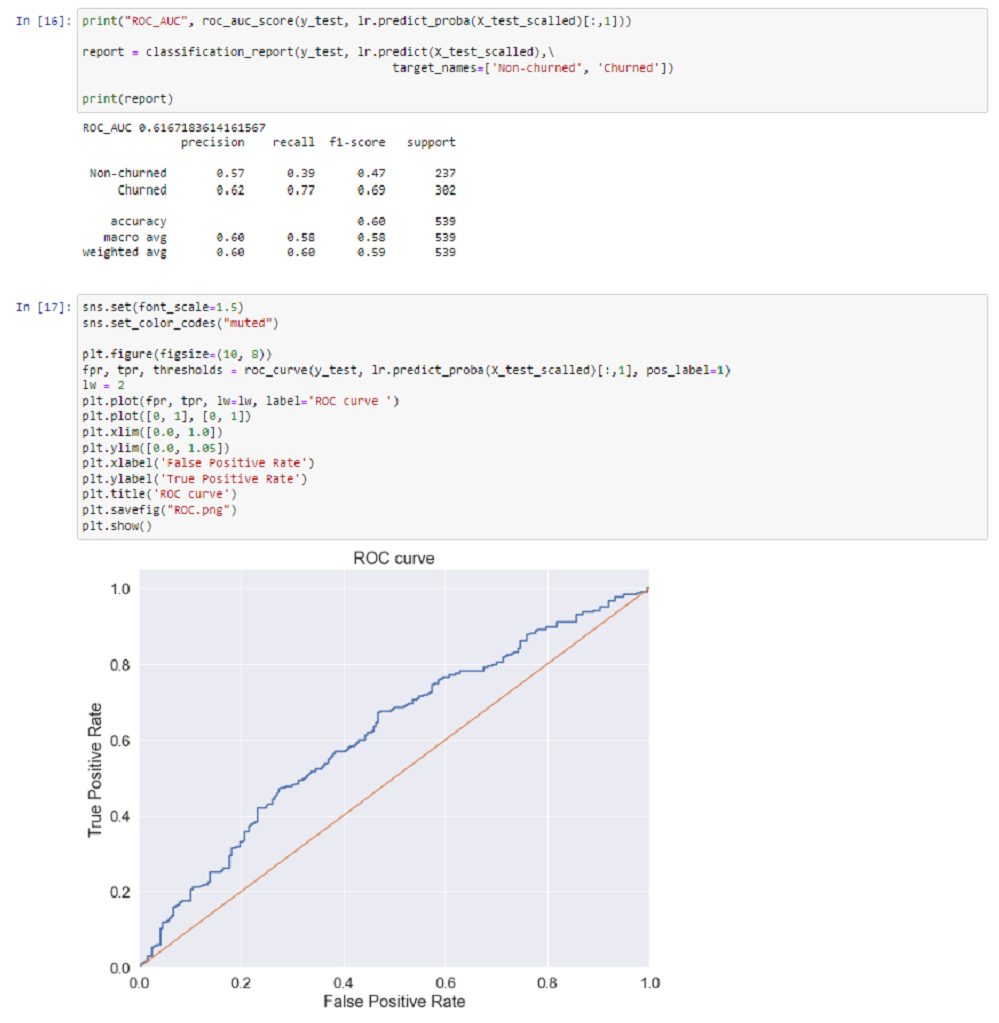
print(report)

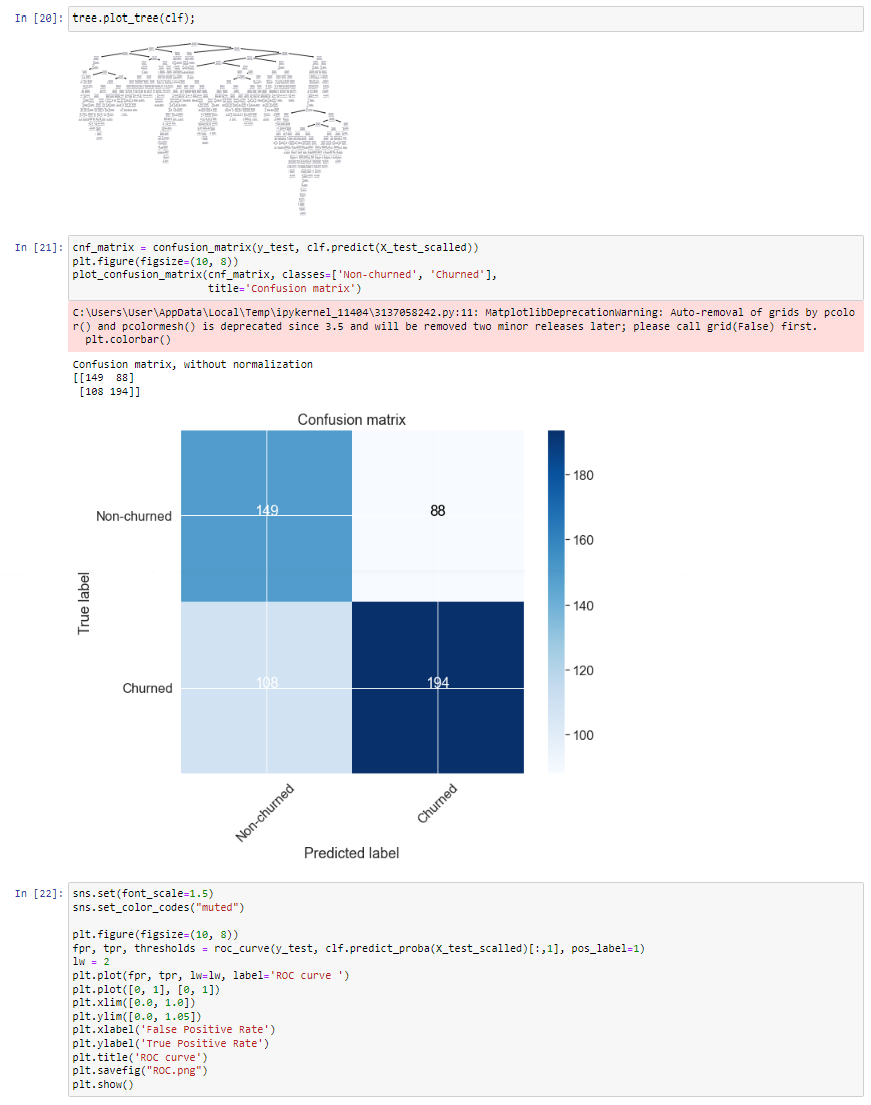
### Решение задачи 4:

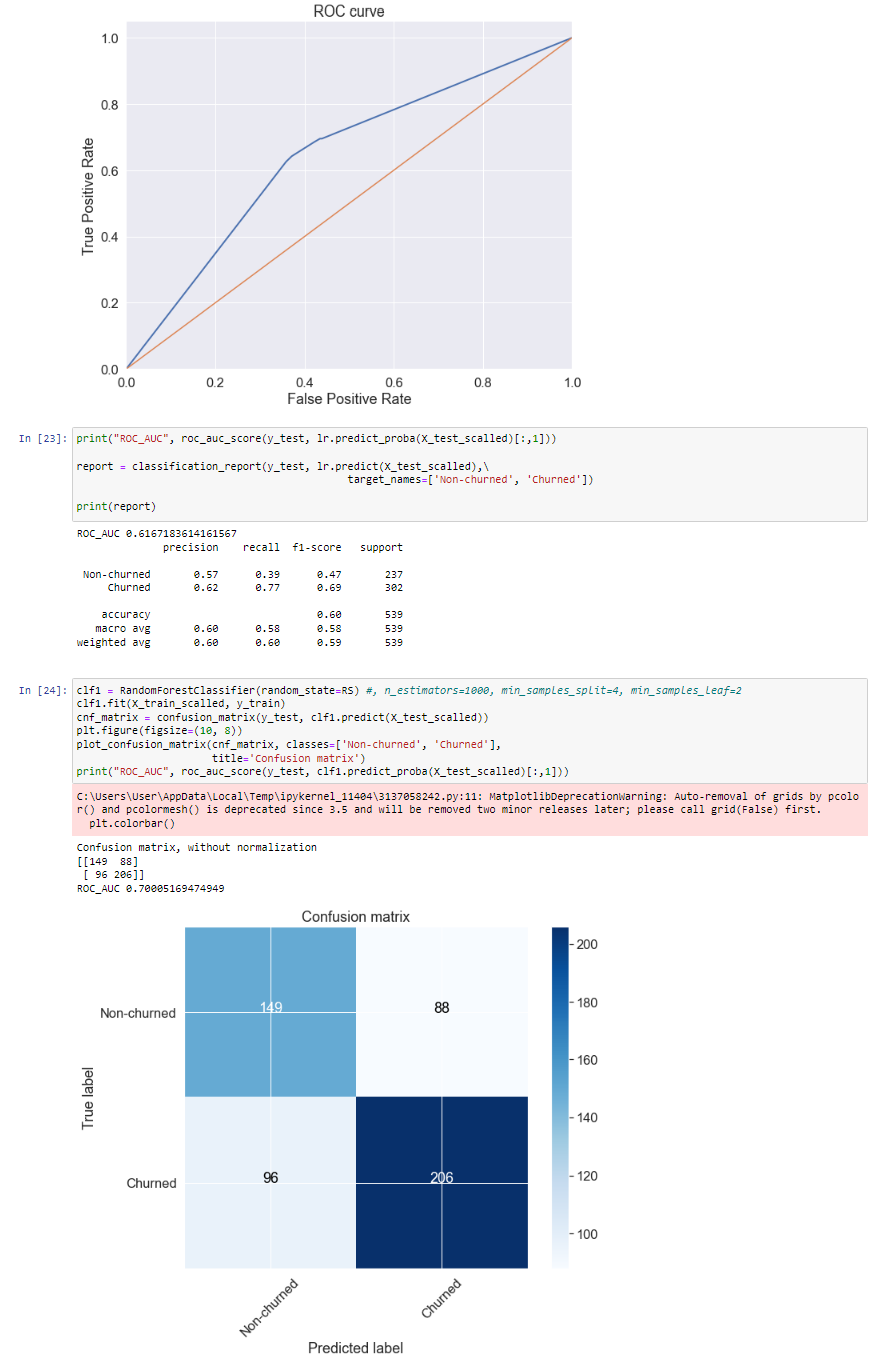


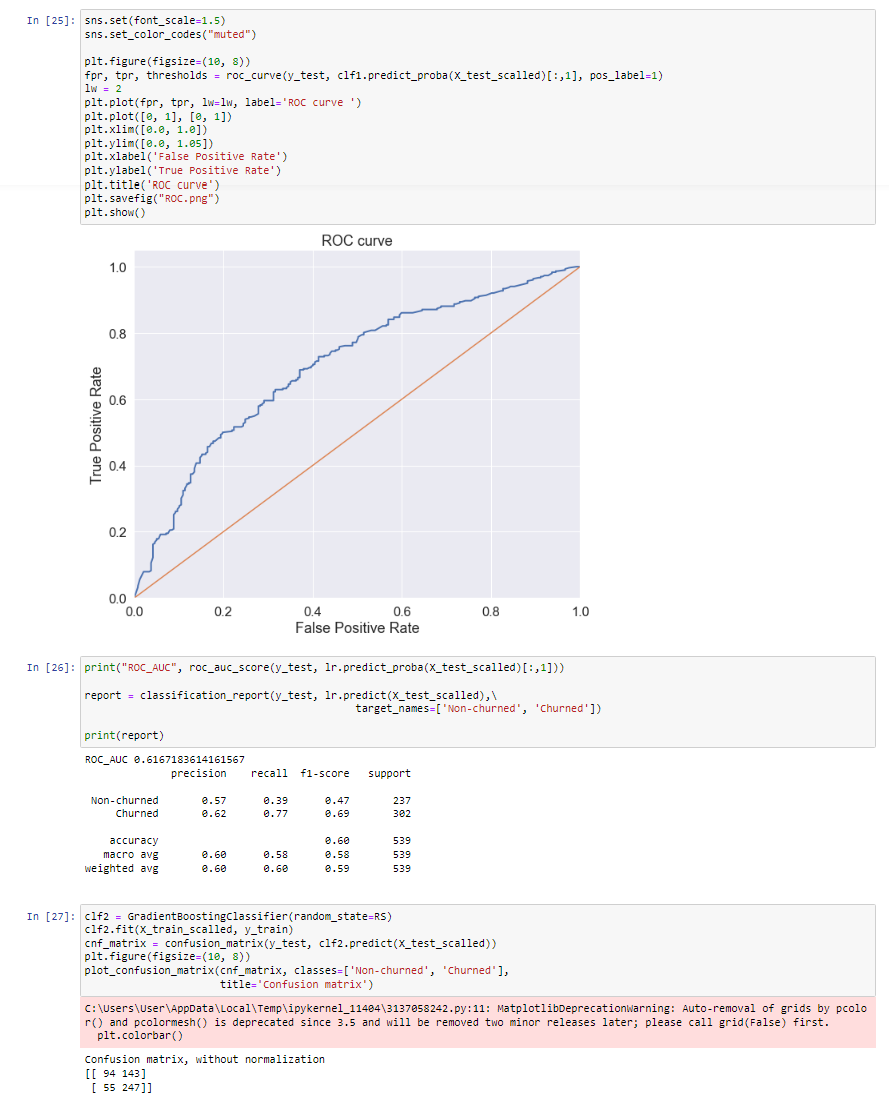


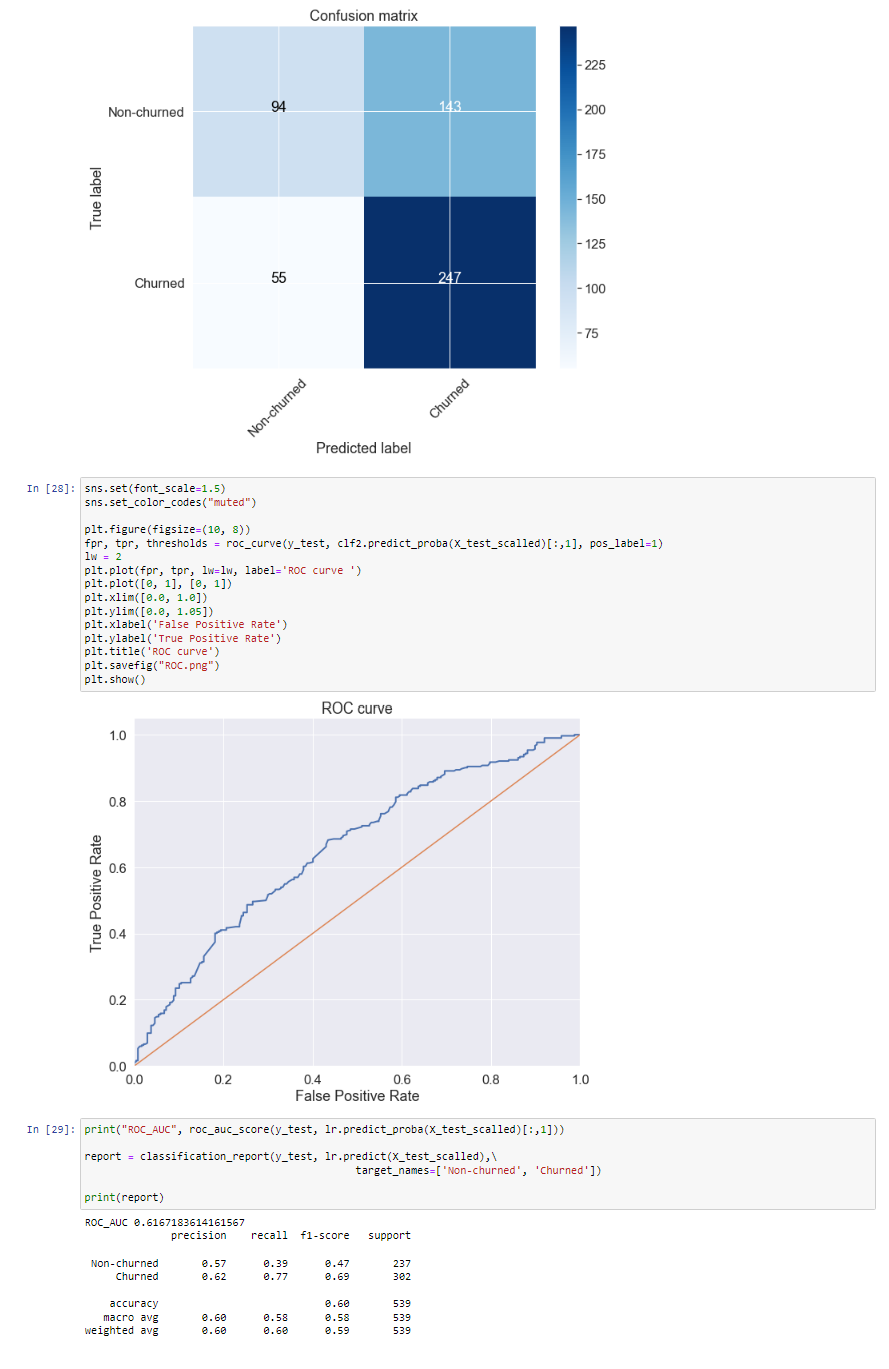












Как видно из финальных данных этого раздела, мы получили весьма неплохой результат и можем с уверенностью использовать обученную модель в целях выявления оптимальных рецептур шоколада на основе указанных марок.

Результаты работы данной модели могут помочь не только оптимизировать расходы на создание массового продукта, но и придумать новые эксклюзивные продукты, так сказать «для избранных» - все зависит от того, под каким углом рассматривать применение модели в жизни.

Заключение

Настоящая работа посвящена применению анализа данных и машинного обучения в пищевой отрасли. Мы с вами рассмотрели теоретически и применили на практике этапы машинного обучения.

Рассмотренными инструменты можно не только редуцировать временные, финансовые и материальные издержки, но и позаботиться об экологии – например, через сокращение утилизации пробных партий продукции с неудачной рецептурой.

Используя технологии искусственного интеллекта и инструментария аналитики данных возможна и роботизация рутинных процессов, которая на выходе дает экономию ресурсов организации и улучшение качества продукции. Например, ускорение процесса измерения тушек рыб при приемке за счет автоматизации измерения их массы без взвешивания каждой по-отдельности, а только на анализе данных визуальных измерений, даёт нам сокращение сроков поставки рыбы, то есть повышение показателя свежести у скоропортящегося продукта, на переработку или прямиком к нашему столу.

В данной проектной работе показан высокий потенциал применения использованных методик к задачам производства более качественной продукции в сочетании с сокращением издержек.

Список используемой литературы

1. Абдрахманов М.И. Python. Визуализация данных. Matplotlib. - Devpractice Team, 2020 – 413 с.
2. Бьерн Андерсен. Бизнес-процессы инструменты совершенствования, 2003 г., 272 с.
3. Джон Джестон, Йохан Нелис. Управление бизнес-процессами. Практическое руководство по успешной реализации проектов, 2015 г.
4. Коллектив авторов Harvard Business Review - Руководство по улучшению бизнес-процессов, 5-е издание, Гарвард, 2022 г., 132 с.
5. Microsoft, Power Query в Excel и Power BI: сбор, объединение и преобразование данных, Москва, 2022 г., 480 с.
6. 50 оттенков matplotlib — The Master Plots (с полным кодом на Python)

https://habr.com/ru/post/468295/

1. Введение в линейную регрессию для машинного обучения, 2020 г.

<https://www.codecamp.ru/blog/introduction-to-linear-regression-python/>

1. Машинное обучение как новая революция, 2022 г.

https://habr.com/ru/companies/sberbank/articles/680436/

1. Многомерные графики в Python — от трёхмерных и до шестимерных

https://habr.com/ru/post/456282/

1. Обзор методов классификации в Python с помощью Scikit-Learn, 2021 г.

<https://pythobyte.com/overview-of-classification-methods-in-python-with-scikit-learn-47463/>

1. Построение графиков в Python при помощи Matplotlib

https://python-scripts.com/matplotlib

1. Предварительная обработка данных - методы, концепции и шаги для освоения, 2023 г.

[https/www.projectpro.io/article/data-preprocessing-techniques-and-steps/512](https://translated.turbopages.org/proxy_u/en-ru.ru.37faa164-65a54546-4f777c27-74722d776562/https/www.projectpro.io/article/data-preprocessing-techniques-and-steps/512)

1. Примеры графиков Matplotlib

https://matplotlib.org/stable/gallery/index.html

1. Руководство пользователя Matplotlib

https://matplotlib.org/stable/contents.html

1. Шпаргалка по визуализации данных в Python с помощью Plotly

https://habr.com/ru/post/502958

1. Шпаргалки по Matplotlib

<https://github.com/matplotlib/cheatsheets#cheatsheets>

1. matplotlib / cheatsheets

https://github.com/matplotlib/cheatsheets#cheatsheets

1. Plotting sine and cosine with Matplotlib and Python https://pythonforundergradengineers.com/plotting-sin-cos-with-matplotlib.html
2. https://www.kaggle.com
3. https://www.python.org

Приложения

1. task\_1 – файл Jupyter с решением первой задачи второй главы

2. task\_2 – файл Jupyter с решением второй задачи второй главы

3. task\_3 – файл Jupyter с решением третей задачи второй главы

4. task\_4 – файл Jupyter с решением четвертой задачи второй главы

5. Исходный файл Chocolate\_data\_final.csv

6. Исходный файл Fish\_final.csv

7. Power BI Chocolate – файл с анализом данных

8. Power BI Fish – файл с анализом данных