Министерство образования и науки Республики Башкортостан

Государственное бюджетное профессиональное образовательное учреждение Уфимский колледж статистики, информатики и вычислительной техники

**Документация по проделанной работе**

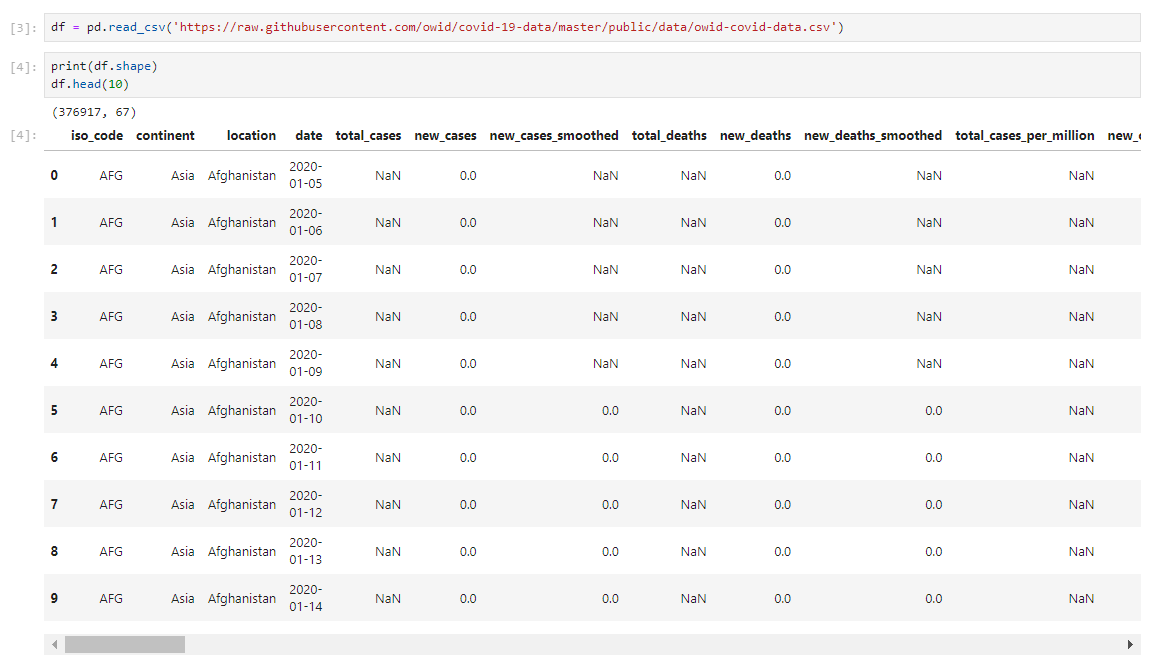
**Автор**: Никифорова Карина,

студентка группы 21П-1

Уфа — 2024

1. Сессия 1
   1. Парсинг данных

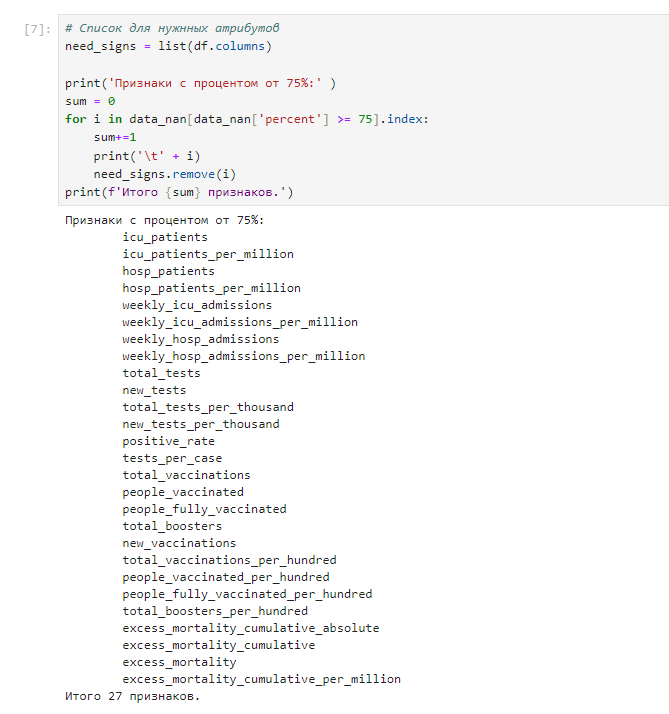
Для начала спарсили данные с owid-covid-data.csv репозитория https://github.com/owid, и построили по ним датасет. Просмотрели все атрибуты.



Так же был сделан вывод, что присутствует слишком много пустых значений, которые в будущем нужно будет заменить.

* 1. Предобработка данных и выделение значимых атрибутов

Был определён выбор нужных атрибутами. Использовали такие параметры как пропущенные значения и корреляцию.



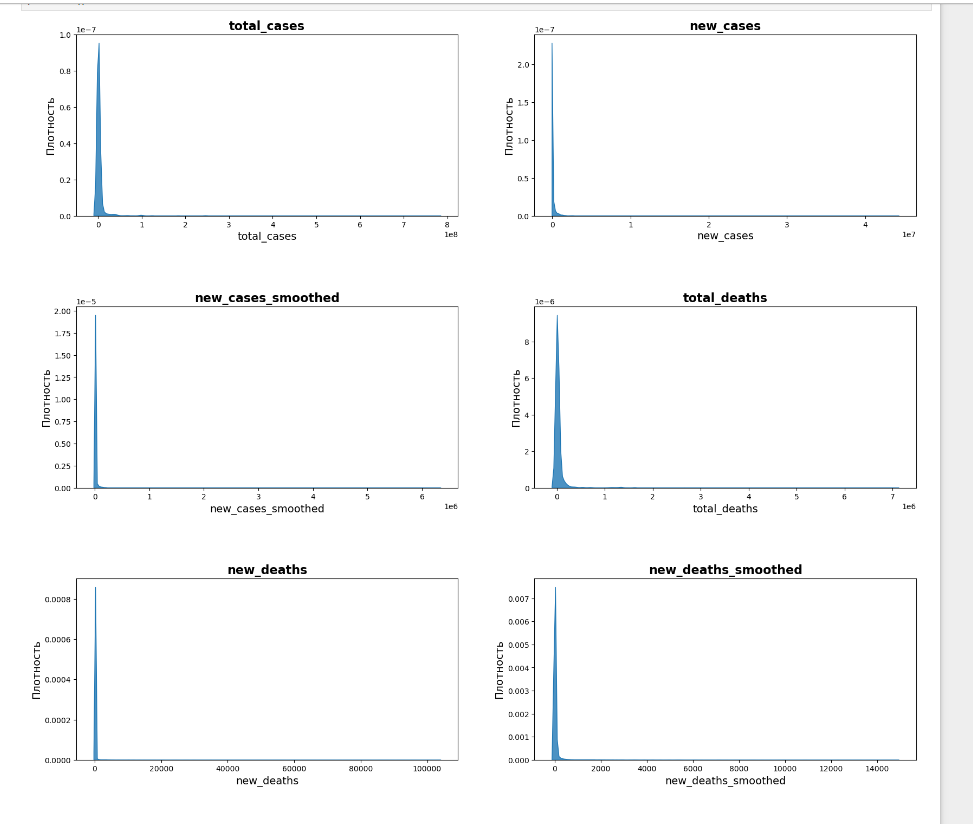
Данные признаки с большим количеством пропущенных значений были удалены, так как их будет сложно корректно заполнить, и если просмотреть какие данные в них находятся, то можно понять, что они не повлияют на кластеризацию.



В итоге было оставлено 33 признака для последующей обработке.

* 1. Описание структуры набора данных

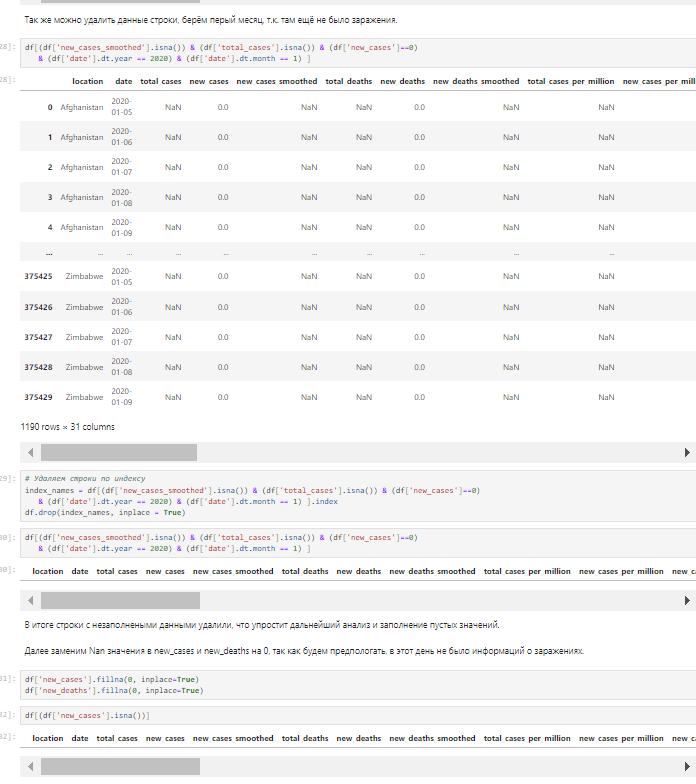
Прописали описание всех атрибутов. Проанализировали сам датасет на нормальность, и увидели, что распределение не являться нормальным, поэтому далее нормализовали выборку.



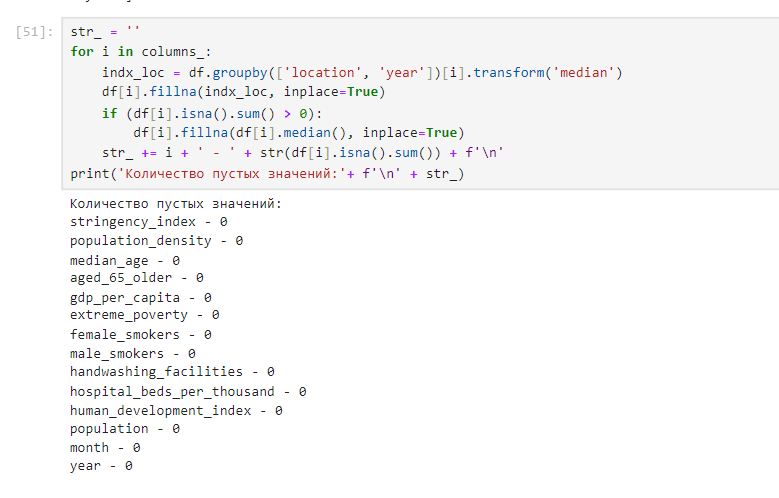
Было замечено, что даже ключевые атрибуты имеют не нормальное распределение.

* 1. Формирование дополнительных атрибутов

Убрали не нужные атрибуты по п. 1.2. Добавили атрибут для оценки распространения болезни, с помощью алгоритма, вычисляя значения по формуле.

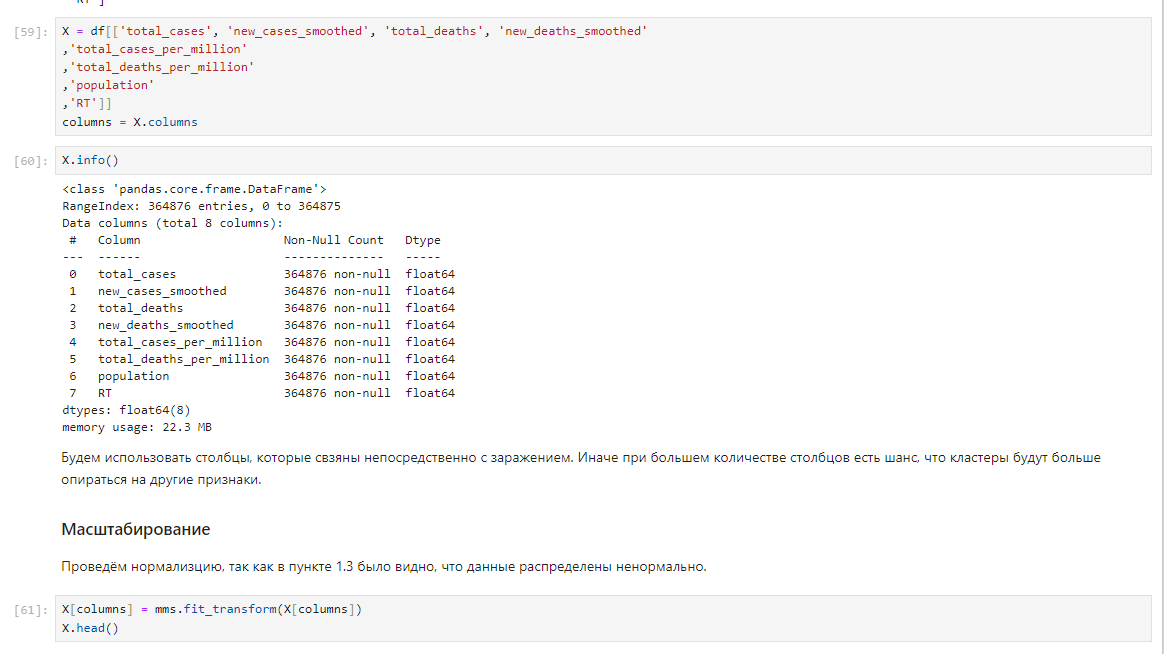


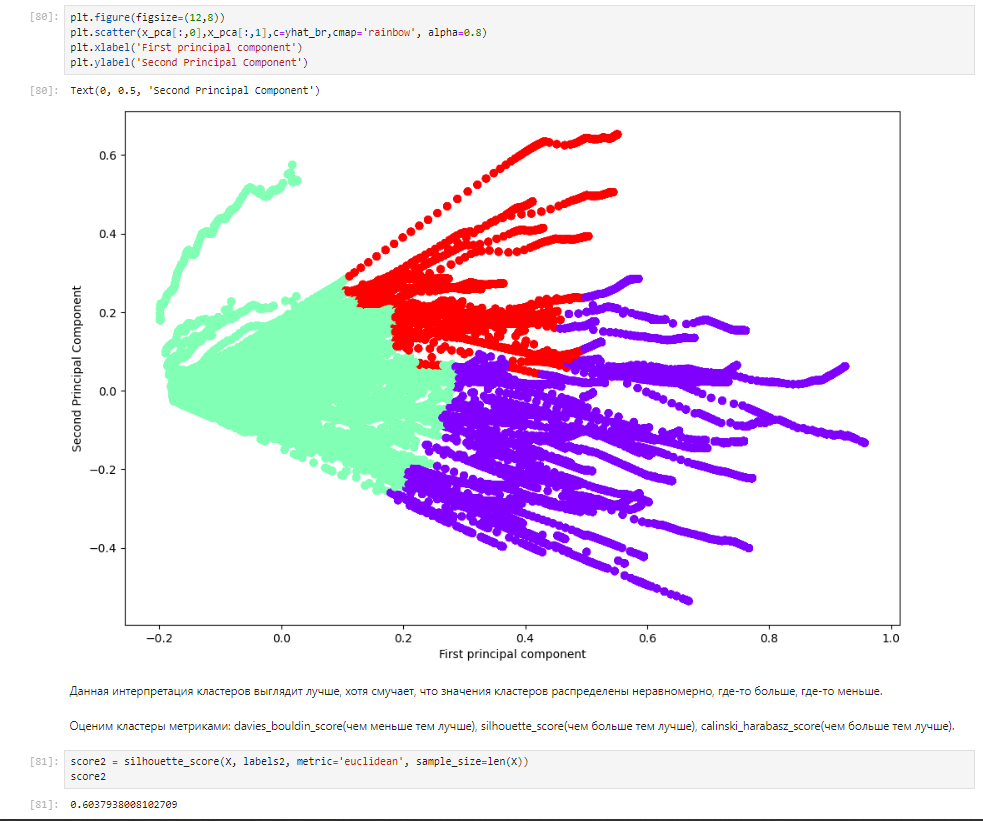
Некоторое количество строк было удалено, был приоритет на заполнение данных. В некоторых строках Nan означал 0, поэтому так эти значения и заменили. Другие значения были заменены с помощью средних значений по группам:



* 1. Кластеризация набора данных

Была проведена нормализация выборки, в итоге оставили нужные атрибуты для кластеризации. Определили кластеры - группы опасности, нахождения в то или иное время в стране(городе), лучшей моделью оказалась модель Birch.



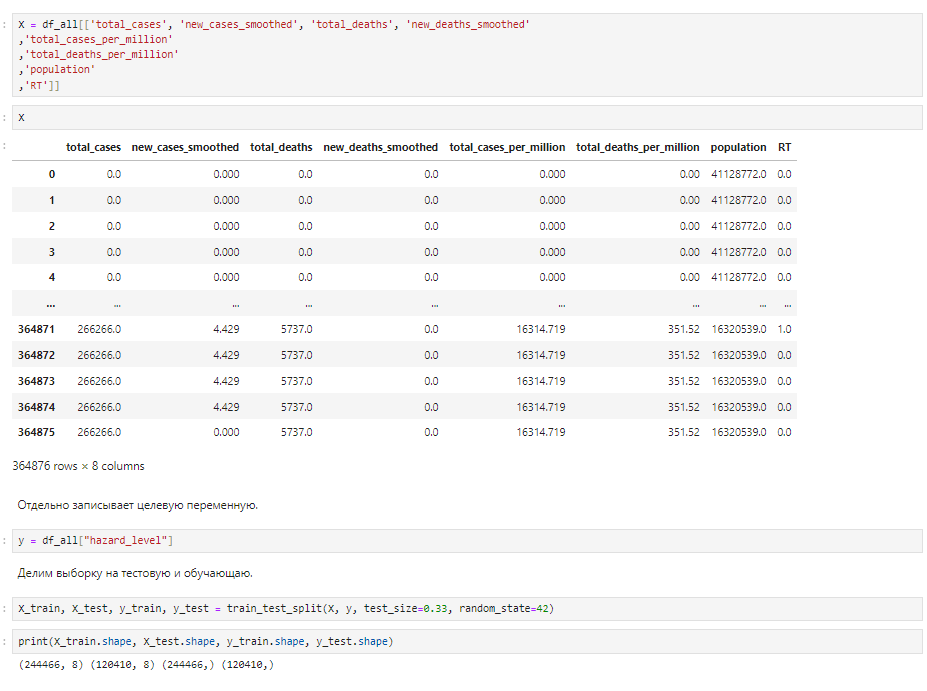


* 1. Подготовка отчета

В итоге был подготовлен набор данных с обработанными атрибутами, так же в этом наборе присутствуют кластеры, которые определяют уровень опасности. Этот фрейм данных был сохранён для последующей работ с ним result\_1.csv.

Был создан отчёт Report\_1.html + Report\_1.ipynb и файл Readme\_1.txt, записали всё в Data\_1.zip.

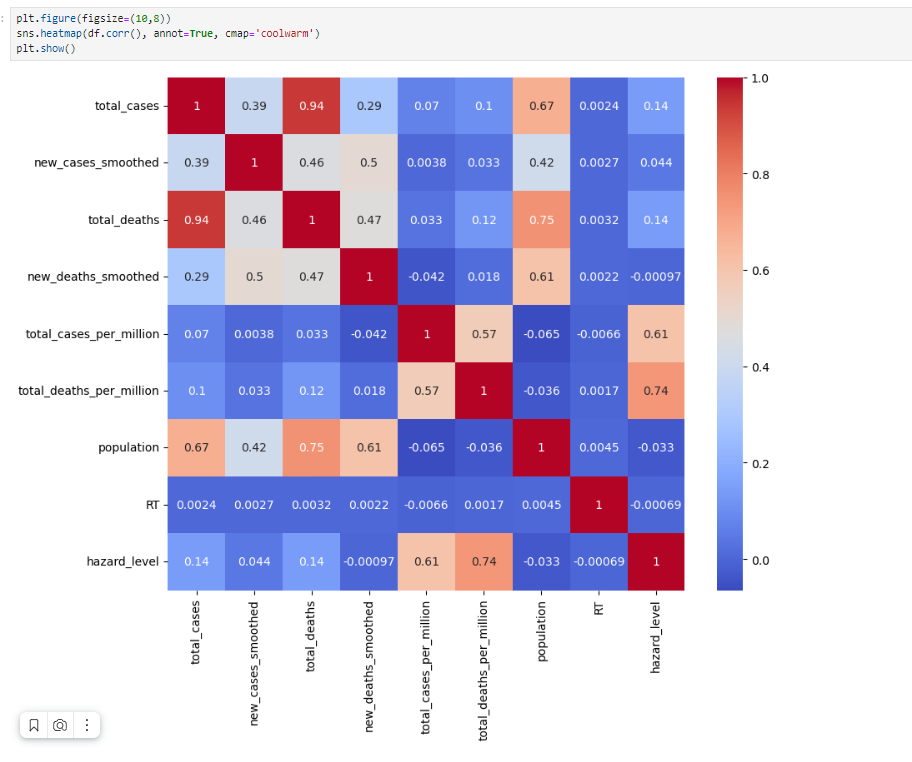
1. Сессия 2
   1. Разбиение набора данных

Для начала записали данные с result\_1.csv в dataframe. Поменяли значения hazard\_level с object на int для дальнейшей работы. 

Было проведено разбиение выборки на обучающую и тестовую с соотношением 0.33 на 0.67. Взяли для данного разбиения значения, которые больше всего подходят для обучения модели.

* 1. Визуализация зависимостей данных

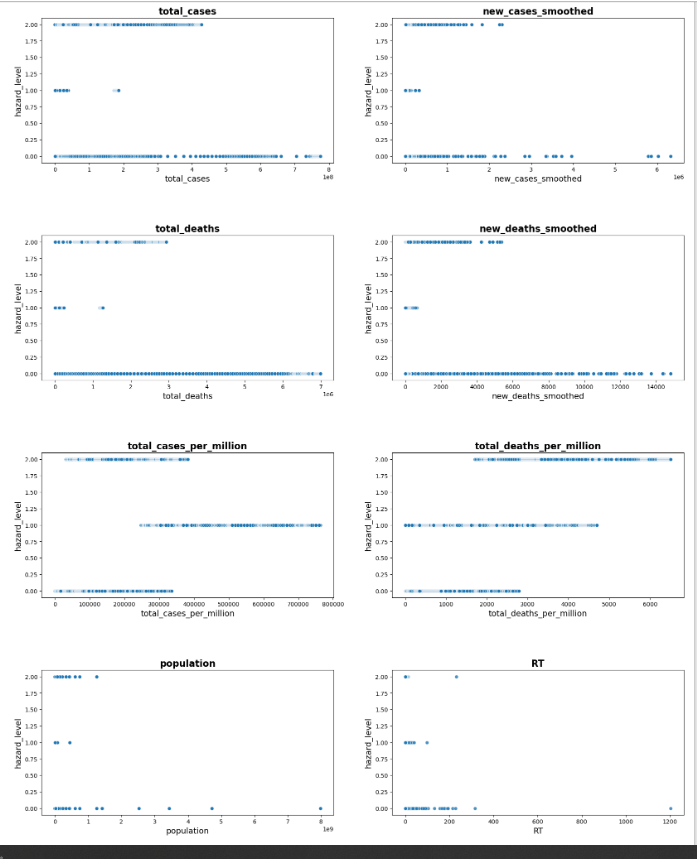
Для того, чтобы определить насколько признаки совместимы с целевой переменной, визуализировали корреляцию признаков, по которой выяснили некоторые зависимости, создали pairplot для зависимости атрибутов между друг другом, и просмотрели линейную зависимость между целевой переменной и другими признаками. Сделали вывод, что данные признаки подходят для классификаций.



Было замечено, что целевая переменная коррелирует с total\_cases\_per\_million и total\_deaths\_per\_million. Так же есть хорошая зависимость у population с total\_cases и total\_deaths.



Увидели, что полностью отсутвует линейная зависимость между признакми. При зависимости total\_cases total\_deaths можно увидеть увеличение смертей при увелечений общих случаев заражения.



Лучше всего была видна зависимость у total\_cases\_per\_million, при безопасном уровне он имеет значения от 0 до 300000, при среднем уровне имеется распределение от 3000000 до 800000, и опасный уровень от 100000(когда только начиналось сильное заражение) до 400000.

* 1. Классификация исходных компетенций

Проанализировав разные модели классификаций, были выбраны три модели опираясь на выводы по прошлому пункту: на нормализацию данных, на количество признаков и их корреляцию. В итоге взяли модели GaussianNB, RandomForestClassifier, LinearDiscriminantAnalysis.

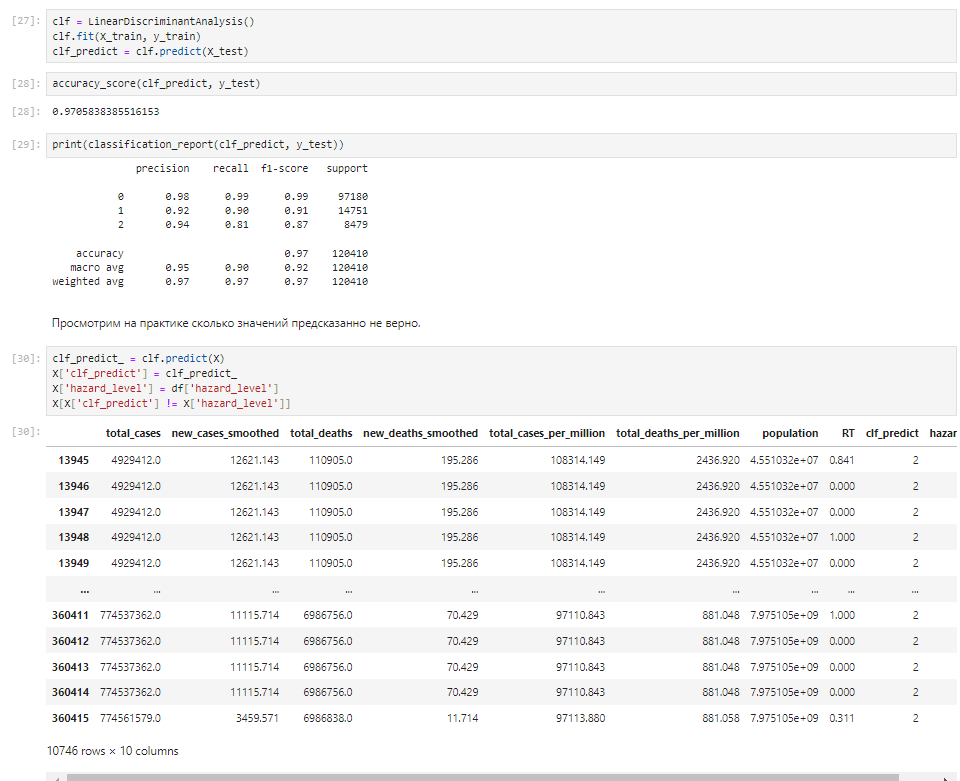
Модель RandomForestClassifier подходит, так как у нас имеются числовые признаки (все), у многих признаков низкая корреляция друг с другом, и признаки не имеют линейной зависимости.

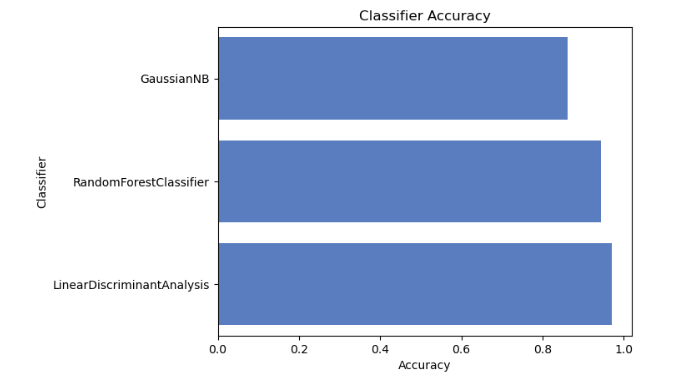
Гауссов наивный Байесовский классификатор — один из нескольких алгоритмов, доступных в машинном обучении, которые можно использовать для решения широкого круга задач. Поэтому возьмём его как предполагаемую модель, которая сработает благодаря своей простоте. Подходят для данных которые непрерывно распределены. Так же GaussianNB плохо работает с категориальными переменными, у нас такие отсевают, так что это будет хороший вариант.

Linear Discriminant Analysis модель подходит для выборки, не используем много характеристик, и в основном она имеет немимические данные.

* 1. Обучение

Провели обучение, выбранных в пункте 2.3, моделей. Проанализировали метрики данных моделей и их значения в обучений, и по итогам выбрали лучшую модель LinearDiscriminantAnalysis, которая так же больше всех подходит данной выборке.

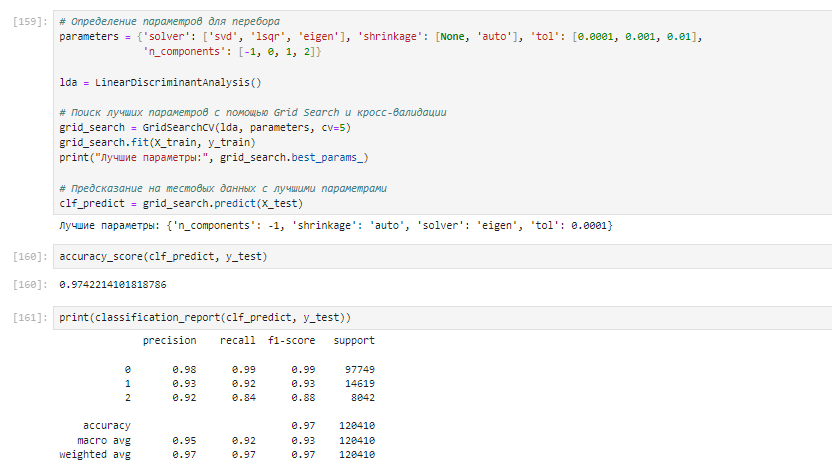




В итоге, если смотреть по метрикам, лучшей моделью является LinearDiscriminantAnalysis. При просмотре df с предсказаниями данной модели, можно увидеть, что она предсказывает почти идеально все уровни опасности, имея неправильный процент предсказаний всего в 2.9%. Так же если проанализовать для каких данных лучше подходит эта модель(п. 2.3), то можно понять, что для данной выборки она лучше других моделей.

* 1. Feature Engineering

Улучшили модель LinearDiscriminantAnalysis, с помощью добавления признаков, и нахождения лучших параметров с помощью GridSearch. В итоге модель была улучшена.



Для улучшения модели, использовали все признаки, не включая признаки даты и локаций, и использовали лучшие параметры {'n\_components': -1, 'shrinkage': 'auto', 'solver': 'eigen', 'tol': 0.0001}.

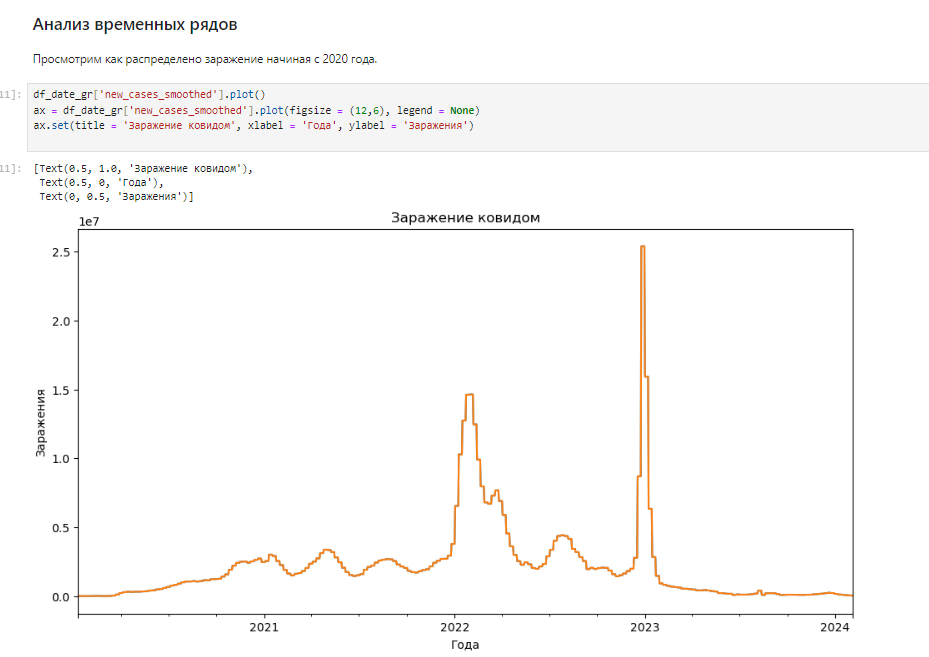
* 1. Подготовка отчета

В итоге был проведён анализ фрейма данных, и по этим анализам, были определенны лучшие модели классификаций. Для дальнейшей работы была сохранена модель классификаций.

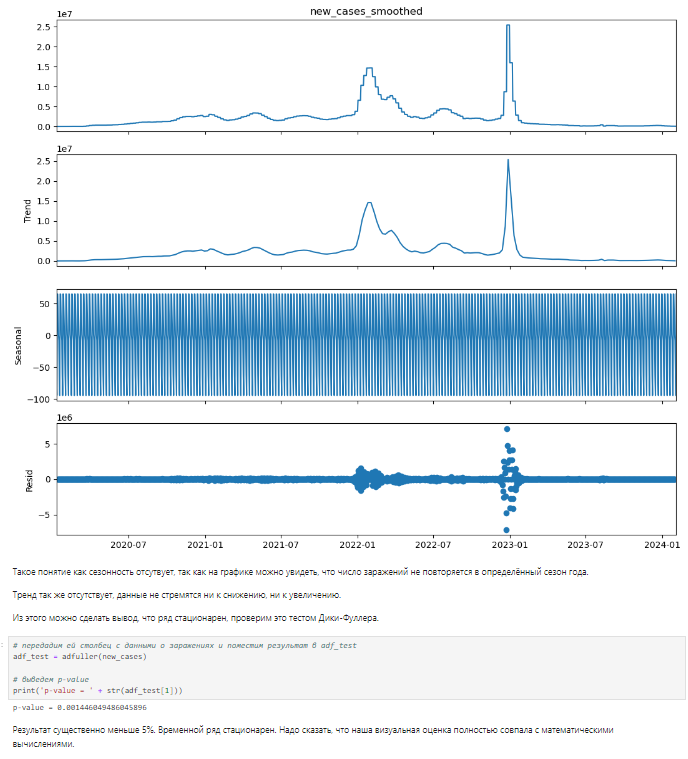
Создали отчёт Report\_2.html + Report\_2.ipynb и файл Readme\_2.txt содержащий информацию о данных, записали всё в Data\_2.zip.

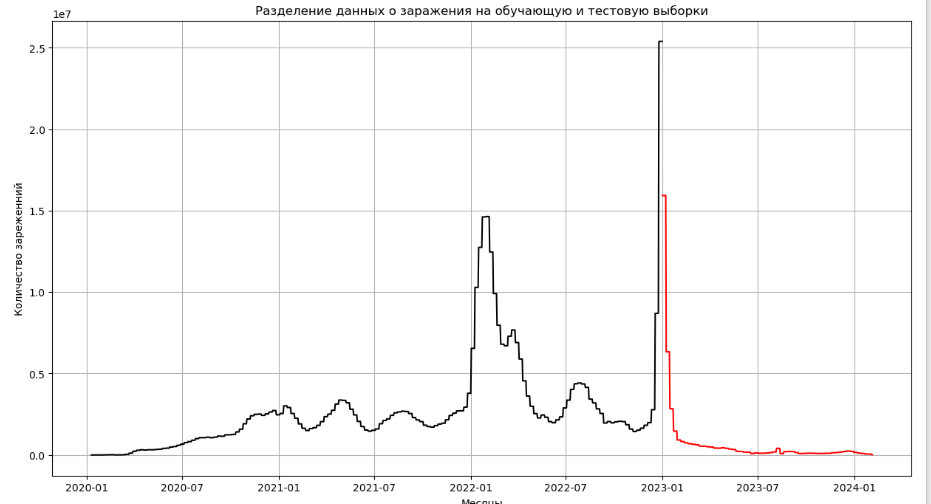
1. Сессия 3
   1. Построение регрессионной модели

Для прогнозирования временного ряда был проведён анализ данных. Для обучения использовали признак new\_cases\_smoothed.



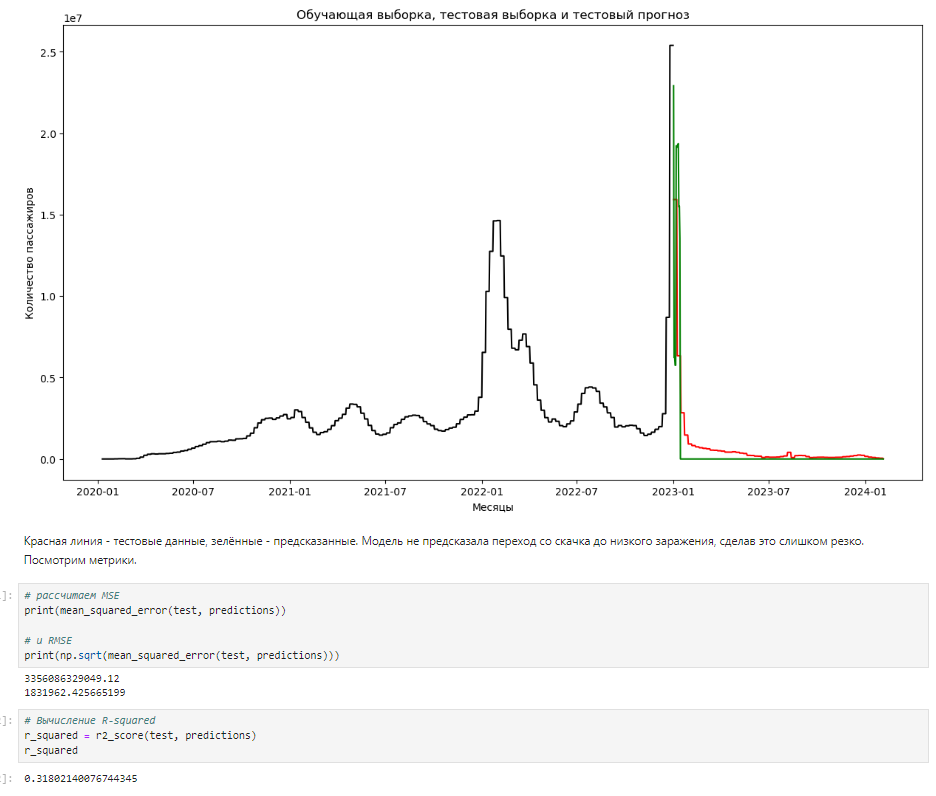
Был сделан вывод, что данные заражения стационарны.



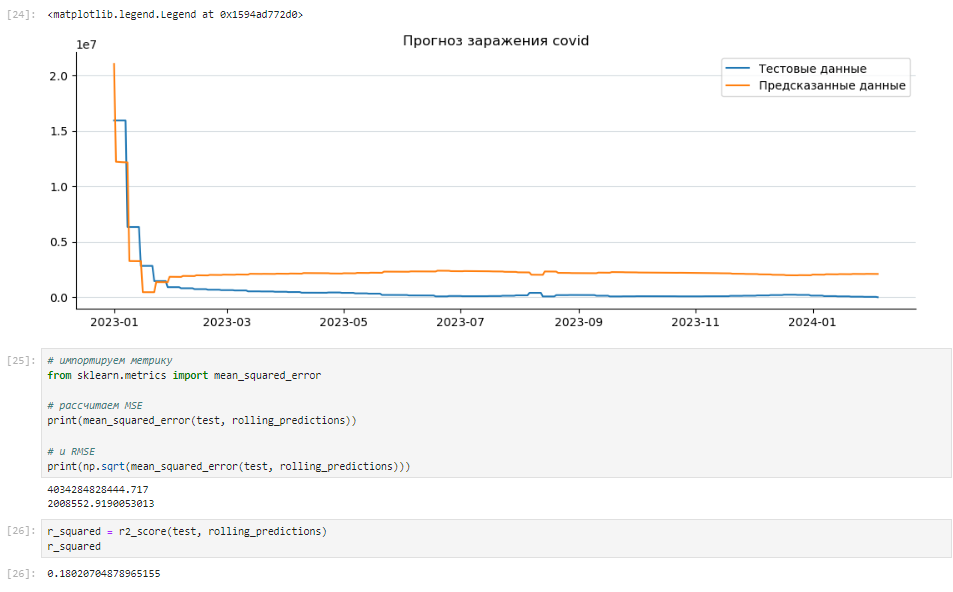
Выборка была разбита на тестовую и обучающая, так как данных мало, процент тестовой выборке установили 26%. 

Были выбраны три регрессионные модели:

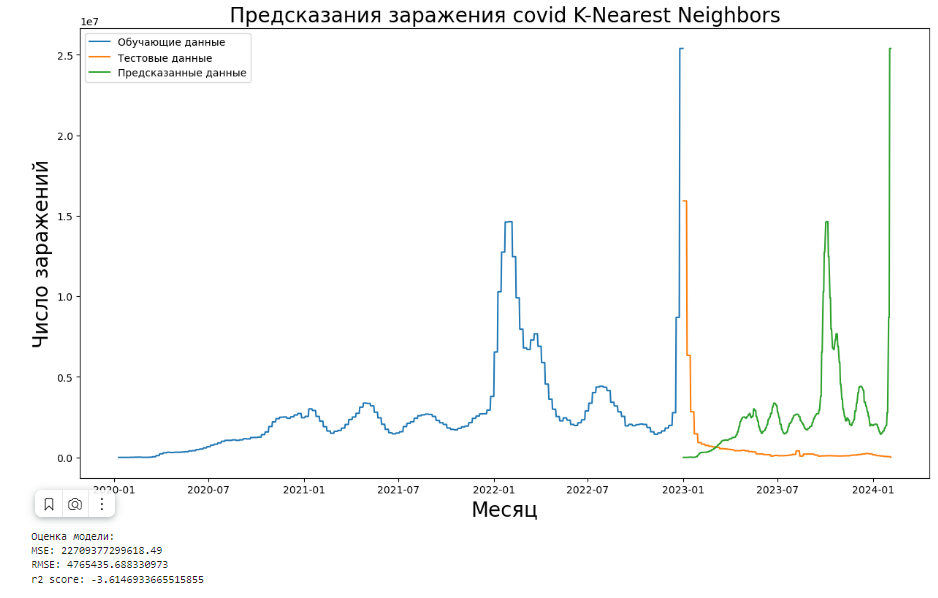
Sarimax - модель показала не плохие результаты, на тестовой выборке, однако были выявлены скачки данных, которые помешали бы дальнейшему прогнозированию;



Garch - модель лучше всех подошла для предсказания заражения covid-19, она правильно определила направление данных, хоть метрики имела хуже прошлой модели;



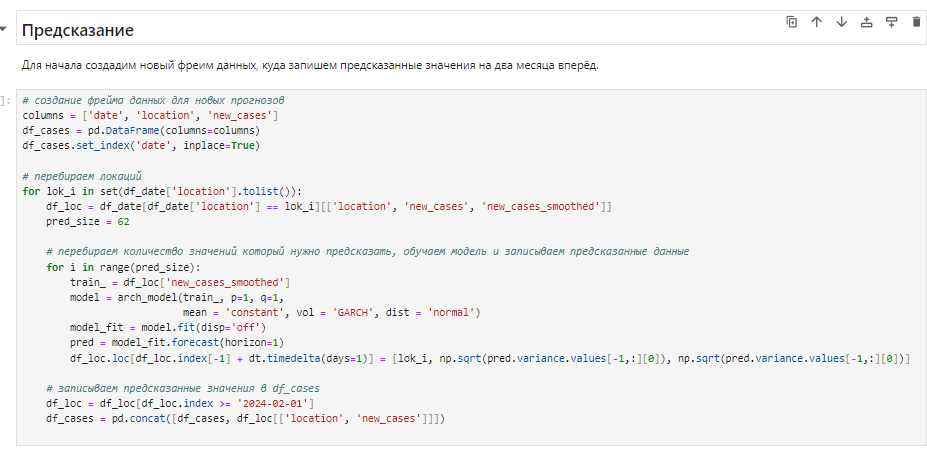
KNeighborsRegressor - модель не подошла под данные заражения, показав самые худшие результаты.



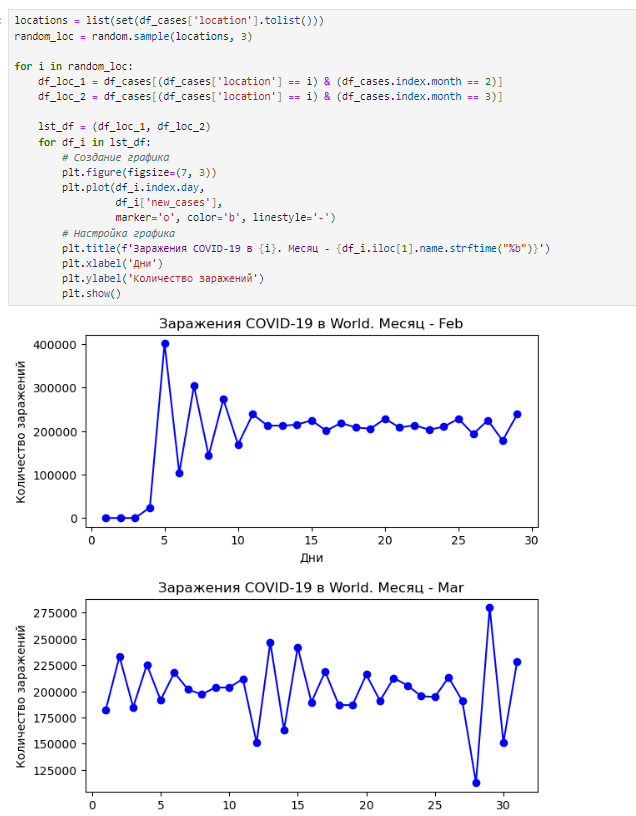
В итоге была выбрана модель Garch.

* 1. Визуализация работы регрессионной модели

Было предсказано число заражений в каждой локаций из исходного набора данных, с помощью модели, выбранной в пункте 3.1.



Так же была проведена визуализация заражений на трёх разных локациях в феврале и марте.



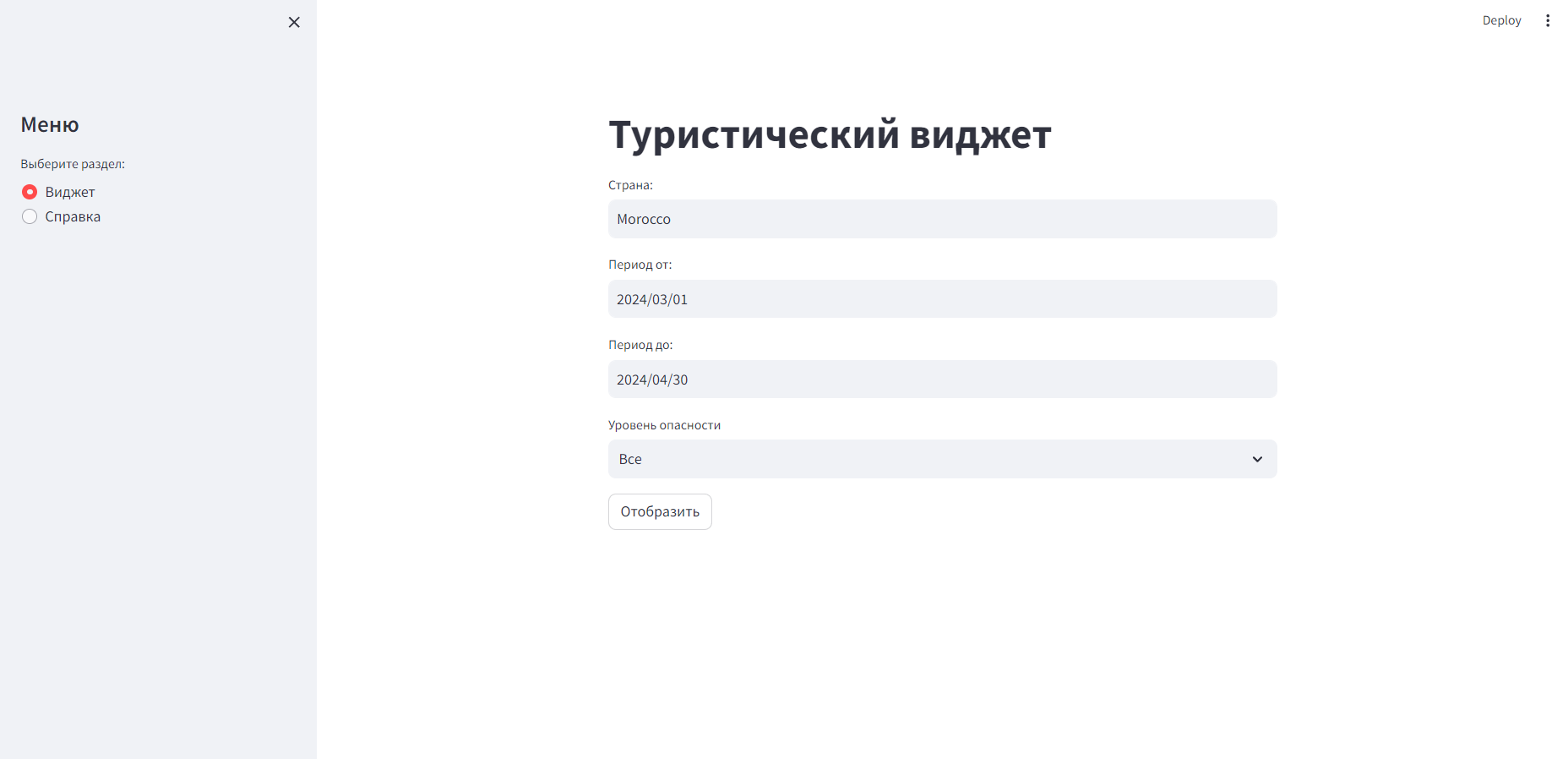
* 1. Подготовка отчета

Была проделана работа по анализу временного ряда, сделан вывод, что ряд стационарен. Выбрали модели обучения на основе анализа. В итоге была выбрана модель Garch. С помощью данной модели предсказали данные на три месяца вперёд. Создали алгоритм для визуализаций прогнозируемых данных в стране.

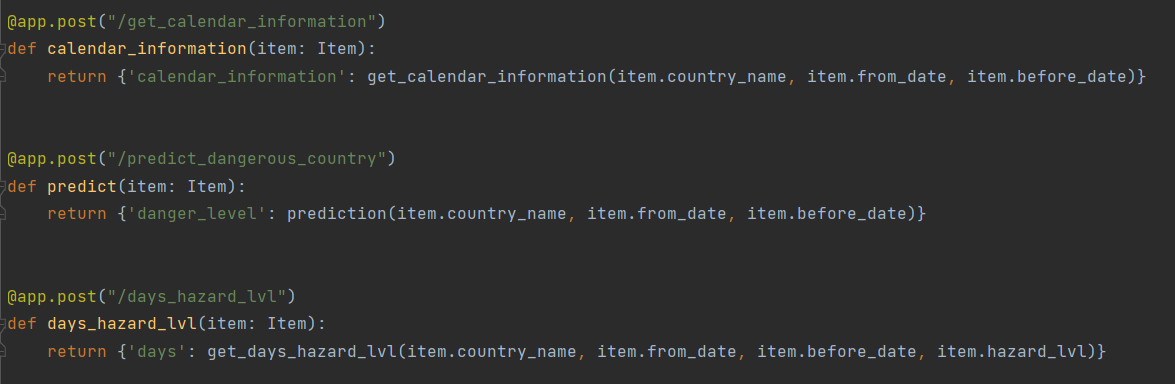
Был создан отчёт Report\_3.html + Report\_3.ipynb и файл Readme\_3.txt содержащий информацию о данных, записали всё в Data\_3.zip.

1. Сессия 4
   1. Разработка API

Был создан программный интерфейс в Python с помощью библиотеки streamlit. Пользователь может вводить страну, выбирать период и уровень опасности.

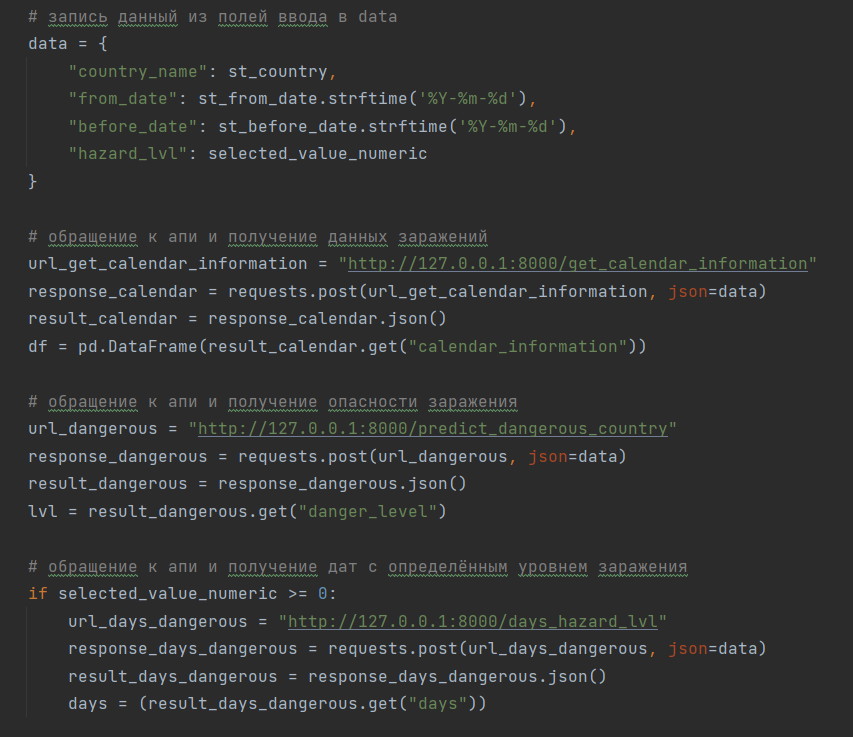


Так же был разработан API с помощью fastapi. Были разработаны такие функций для передачи данных как: передача информаций о заражений в определённой стране за период, передача предсказанного уровня опасности за период в определённой стране, передача дней определённого уровня заражения.



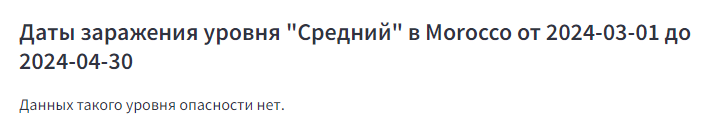
* 1. Разработка приложения

Было произведено «соединение» серверной части и части интерфейса.



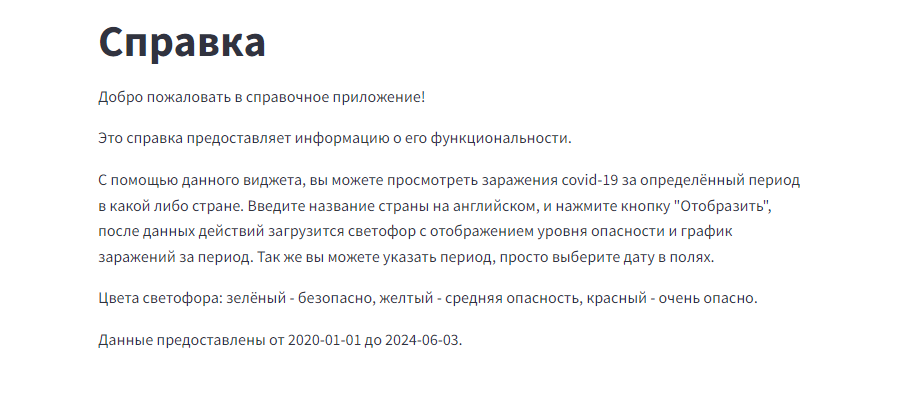
В итоге было получен программный продукт, с которым может взаимодействовать пользователь.





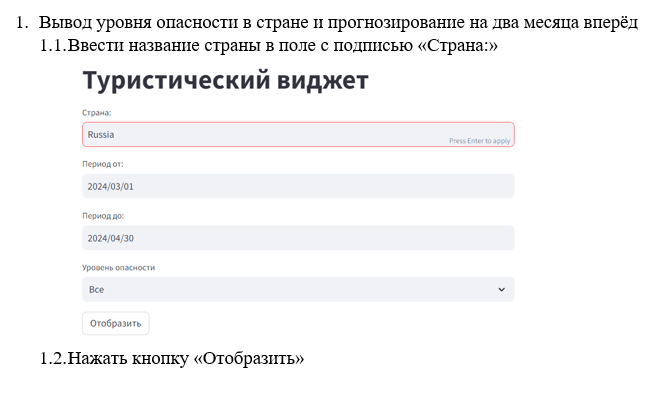
Приложение выводит светофор с уровнем опасности, заражения за определённый период и даты в которых заражения равны уровню опасности выбранных пользователем.

Также была создана справка по приложению.

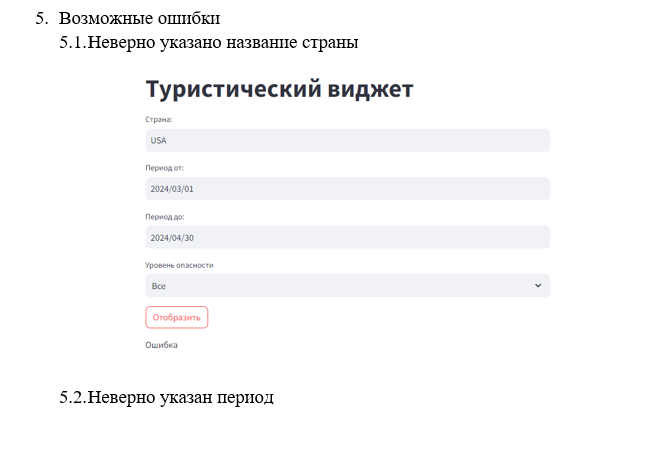


* 1. Программная документация

Для данного приложения было разработано руководство пользователя, оно включало в себя все действия которые может проделать пользователь для вывода какого-либо результата.



Так же руководство пользователя включало в себя возможные ошибки, которые мог допустить пользователь.



Было разработано программное описание, которое включало в себя описание функций серверной и интерфейсной части.

