Содержание:

[Введение 4](#_Toc153339192)

[1. Разведочный анализ данных с помощью PySpark 5](#_Toc153339193)

[1.1 Постановка задачи 5](#_Toc153339194)

[1.2 Описание датасета 5](#_Toc153339195)

[1.3 Типы признаков в датасете 6](#_Toc153339196)

[1.4 Выявление пропущенных значений и их устранение 7](#_Toc153339197)

[1.5 Определение выбросов и их устранение 8](#_Toc153339198)

[1.6 Расчет статистических показателей признаков 10](#_Toc153339199)

[1.7 Визуализацией распределения наиболее важных признаков 13](#_Toc153339200)

[1.9 Выводы 23](#_Toc153339201)

[2. Машинное обучение на больших данных 25](#_Toc153339202)

[2.1 Постановка задачи 25](#_Toc153339203)

[2.2 Применение алгоритмов подготовки данных и машинного обучения 25](#_Toc153339204)

[2.3 Задача регрессии – GradientBoostingMachine 26](#_Toc153339205)

[2.4 Задача классификации – LogisticRegression 29](#_Toc153339206)

[2.5 Анализ результатов 33](#_Toc153339207)

[2.6 Вывод 34](#_Toc153339208)

**Введение**

Анализ данных – это процесс извлечения, преобразования и моделирования данных с целью обнаружения новых закономерностей, трендов и паттернов. Data Science включает в себя методы и технологии обработки данных с использованием математических и статистических алгоритмов, машинного обучения и искусственного интеллекта.

Важность, как и **актуальность**, анализа данных и Data Science в современном мире огромна. Они помогают организациям принимать обоснованные решения, оптимизировать бизнес-процессы, предсказывать тренды и поведение клиентов. Без анализа данных и Data Science невозможно представить себе успешное функционирование таких областей, как финансы, маркетинг, медицина и многие другие.

Big Data обозначает массивы данных, которые становятся слишком большими, сложными и быстро меняющимися для того, чтобы их можно было управлять ими с помощью традиционных инструментов обработки данных.

Apache Spark - это высокопроизводительный кластерный фреймворк для обработки больших данных. PySpark - это API для работы с Apache Spark, предназначенное для использования с языком программирования Python. PySpark позволяет использовать все преимущества Apache Spark, используя удобный и мощный Python.

PySpark предоставляет набор инструментов для машинного обучения, включая модуль MLlib, который предлагает широкий выбор алгоритмов машинного обучения и инструменты для обработки признаков.

**Целью** работы является изучение инстумента PySpark и применение навыков для осуществление разведочного анализа и машинного обучения.

Для реализации цели были решены следующие **задачи**:

* Провести разведочный анализ данных с помощью PySpark;
* Разобрать машинное обучение на больших данных.

1. **Разведочный анализ данных с помощью PySpark**
   1. **Постановка задачи**

Цель и задачи работы:

1. Познакомиться с понятием «большие данные» и способами их обработки;
2. Познакомиться с инструментом Apache Spark и возможностями, которые он предоставляет для обработки больших данных.
3. Получить навыки выполнения разведочного анализа данных использованием pyspark.

Необходимо провести разведочный анализ датасета с определением:

* типов признаков в датасете;
* пропущенных значений и их устранением;
* выбросов и их устранением;
* расчетом статистических показателей признаков (средних, квартилей и т.д.);
* визуализацией распределения наиболее важных признаков;
* корреляций между признаками.
  1. **Описание датасета**

Архив с датасетами твитов общественного мнения и политики из Твиттера был выбран, т.к. является индивидуальным для моего варианта.

Ссылка – https://www.kaggle.com/datasets/paultimothymooney/twitter-election-data-archives

В качестве датасета, на основе которого проводился разведочный анализ, был выбран файл формата CSV под названием «iran\_201906\_2\_tweets\_csv\_hashed.csv» находящийся в папке «June 2019». Размер файла составляет 1.40 GB.

Данный датасет содержит порядка 7417013 строк, которые описывают каждый выложенный пост в сервисе «Twitter» за июнь 2019 года. В датасете было обнаружено 31 колонка. Из них, для дальнейшего анализа, было решено оставить 10 колонн.

Датасет состоит из таких колонок, как:

* 1. Идентификатор твита. Является уникальным идентификатором для каждого поста.
  2. Идентификатор пользователя. Является уникальным идентификатором для каждого пользователя.
  3. Количество подписчиков у пользователя.
  4. Количество подписок.
  5. Язык поста.
  6. Ретвит или оригинальный пост.
  7. Количество цитирований поста.
  8. Количество ответов поста.
  9. Количество лайков поста.
  10. Количество ретвитов поста.
  11. **Типы признаков в датасете**

В ходе анализа датасета были выявлены следующие типы признаков в датасете, описанных в табл. 1.

*Таблица 1. Типы признаков в датасете*

|  |  |  |
| --- | --- | --- |
| Название колонки | Тип данных | Тип признака |
| tweetid | long | Категориальный |
| userid | long | Категориальный |
| follower\_count | integer | Количественный |
| following\_count | integer | Количественный |
| tweet\_language | string | Категориальный |
| is\_retweet | boolean | Бинарный |
| quote\_count | integer | Количественный |
| reply\_count | integer | Количественный |
| like\_count | integer | Количественный |
| retweet\_count | integer | Количественный |

В дальнейшем для осуществления обучения модели машинного обучения колонки количественного типа признаков будут преобразованы в тип данных «double».

* 1. **Выявление пропущенных значений и их устранение**

Чтобы корректно проанализировать данные, важно удостовериться, что в наборе данных отсутствуют пропущенные значения.

В ходе анализа пропущенных значений данные были разделены на 2 группы: строковые и числовые колонки, где:

* Строковые колонки проверялись на наличие значений None и null.
* Числовые колонки проверялись на наличие значений None, NaN и null.

После анализа пропущенных значений были получены такие результаты:

* tweetid: 5233116
* userid: 5823087
* follower\_count: 6235316
* following\_count: 6359126
* tweet\_language: 4935972
* is\_retweet: 6705323
* quote\_count: 7110660
* reply\_count: 7109996
* like\_count: 7109808
* retweet\_count: 7109766

Строки с пропущенными значениями решено было удалить из датасета. Количество строк после удаления пропущенных и аномальных нулевых значений – 199496.

* 1. **Определение выбросов и их устранение**

Осуществим обнаружение и удаление выбросов методом межквартильного интервала (Нахождение 25% и 75% квартилей, межквартильного диапазона):

Данные колонок "quote\_count", "reply\_count", "like\_count", "retweet\_count" решено было не обрабатывать, т.к. во всех датасетах, которые имеются в архиве, приводит к тому, что все данные колонок удаляются и остаются нулевые значения, а данные колонки будут нужны для обучения модели и формирования графиков. Это связанно с тем, что большая часть постов не имеют никаких метрических реакций других пользователей.

Исходный код метода межквартильного интервала, с помощью которого определяются и удаляются выбросы, описан ниже:

*Таблица 2. Исходный код метода межквартильного интервала*

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | outlier\_columns = ['follower\_count', 'following\_count']  for column in outlier\_columns:  Q1 = df.approxQuantile(column, [0.25], relativeError=0)[0]  Q3 = df.approxQuantile(column, [0.75], relativeError=0)[0]  IQR = Q3 - Q1  lower\_bound = Q1 - 1.5 \* IQR  upper\_bound = Q3 + 1.5 \* IQR  df = df.filter(col(column).between(lower\_bound, upper\_bound))  df.select(['follower\_count', 'following\_count', 'quote\_count', 'reply\_count', 'like\_count', 'retweet\_count']).describe().toPandas() |

После применения метода устранения выбросов были построены коробчатые диаграммы признаков, которые обрабатывались:

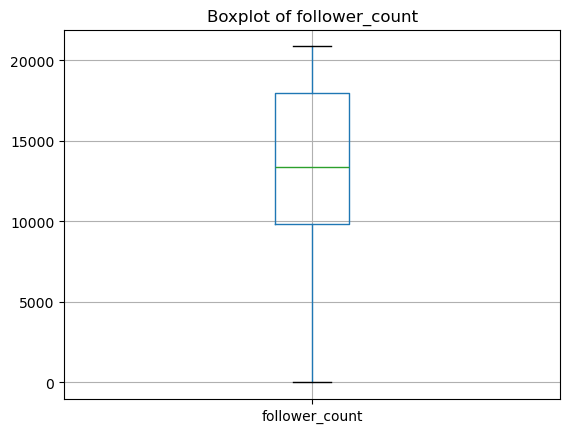


Рисунок 1. Ящик с усами для колонки «follower\_count»

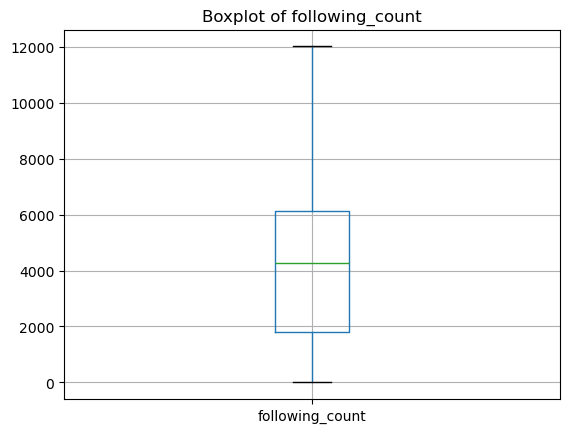


Рисунок 2. Ящик с усами для колонки «following\_count»

Количество строк после выявления и удаления выбросов – 174266.

* 1. **Расчет статистических показателей признаков**

Для реализации этого пункта было решено разобрать категориальные, бинарные и количественные признаки.

Обработаем колонки количественных признаков, а также по одной колонке категориального и одной колонки бинарного признака. Начнём с количественных признаков:

Статистические показатели обширны, мной были выбраны несколько основные из них:

1. Среднее значение (Mean): Обычно является первым показателем, который рассчитывается для числовых данных. Среднее значение предоставляет информацию о центре распределения данных.

2. Медиана (Median): Это значение, которое делит ваш набор данных на две равные части. Она устойчива к выбросам в данных и используется для оценки "среднего" значения, если данные имеют асимметричное распределение.

3. Квартили (Quartiles): Квартили представляют собой три точки данных, которые делят упорядоченный набор данных на четыре равные части. 25-й квартиль - это значение, которое меньше или равно 25% данных, а 75-й квартиль - это значение, которое меньше или равно 75% данных.

4. Мода (Mode): Значение, которое встречается наиболее часто в вашем наборе данных. Мода полезна для определения наиболее распространенных значений в категориальных данных.

5. Стандартное отклонение (Standard Deviation): Это мера разброса данных относительно их среднего значения. Она предоставляет информацию о том, насколько разнообразны значения в вашем наборе данных.

6. Диапазон значений (Range): Простое измерение, которое показывает разницу между самым большим и самым маленьким значениями в наборе данных.

7. Квантили (Quantiles): Квантили обобщают понятие квартиля. В то время как квартили делят данные на четыре равные части, квантили позволяют делить данные на более произвольное количество частей.

Мной были обработаны всё выбранные количественные признаки датасета и представлены в табл. 3.

*Таблица 3. Статистические показатели количественных признаков*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Statistic** | **follower\_count** | **following\_count** | **quote\_count** | **reply\_count** | **like\_count** | **retweet\_count** |
| **Mean** | 14143.411692 | 5001.930870 | 0.042074 | 0.397238 | 4.912054 | 0.526781 |
| **Median** | 13403.000000 | 4278.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| **Mode** | 17956.000000 | 4278.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| **Standard Deviation** | 4691.179063 | 3716.542667 | 2.365007 | 4.550272 | 44.050238 | 5.561052 |
| **Range** | 20874.000000 | 12022.000000 | 668.000000 | 1156.000000 | 3508.000000 | 460.000000 |
| **25th Quantile** | 9847.000000 | 1784.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| **50th Quartiles** | 13403.000000 | 4278.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| **75th Quantile** | 17956.000000 | 6120.000000 | 0.000000 | 0.000000 | 1.000000 | 0.000000 |

Далее рассмотрим статистические показатели категориальных признаков.

Категориальные признаки расчитываются с использованием количественных признаков. В данном случае расчитываются:

* 1. Количество (count) - Указывает какое количество значений имеется в колонке.
  2. Проценты (percentile) - Количественный показатель по определенной части категориального признака в виде процентов (в нашем случае: 25%, 50%, 75% и 95%).
  3. Минимум (min) - Минимальный количественный показатель категориального признака.
  4. Максимум (max) - Максимальный количественный показатель категориального признака.
  5. Среднее (Mean) - Среднее значение количественных показателей категориального признака.
  6. Стандартное отклонение (stddev) - Мера разброса данных относительно среднего значения количественных показателей категориального признака.

Все виды языков рассматривать не будем, и сократим список до 10 уникальных языков.

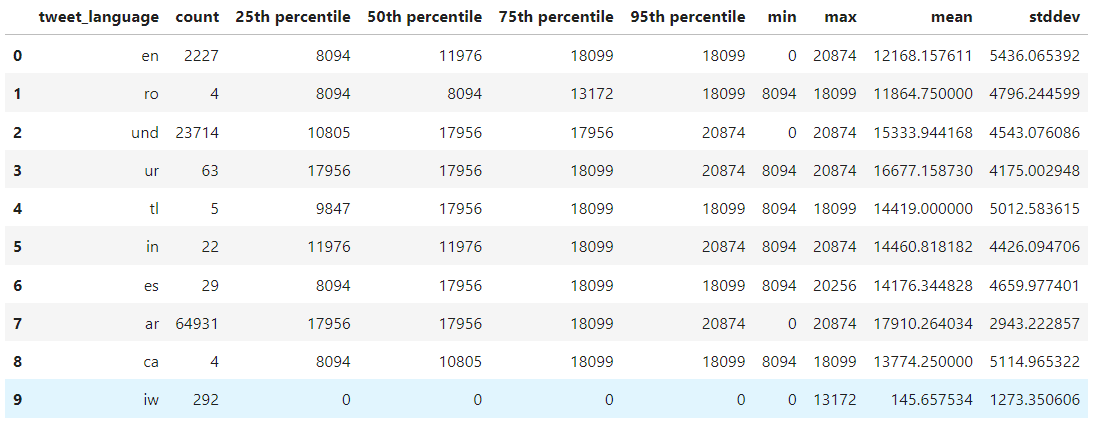


Рисунок 3. Статистические показатели категориального признака колонки «tweet\_language»

Осталось рассмотреть статистические показатели бинарного признака. См. табл. 4.

Бинарный признак:

* 1. Количество (count) - Указывает какое количество значений имеется в колонке
  2. Проценты (percentile) - Отношение количества значений в процентах данного показателя от обшего.

*Таблица 4. Статистические показатели бинарного признака*

|  |  |  |
| --- | --- | --- |
| **is\_retweet** | **count** | **percentage** |
| **true** | 100693 | 57.7812080382863 |
| **false** | 73573 | 42.218791961713706 |

* 1. **Визуализацией распределения наиболее важных признаков**

Для визуализации распределения наиболее важных признаков были использованы следующие графики:

* Гистограммы, в количестве 6 графиков, рис. 4-9;
* Круговая диаграмма, в количестве 1 графика, рис. 10;
* Диаграмма рассеивания, в количестве 5 графиков, рис 11-15.

Это помогает наглядно увидеть распределение данных и выявить возможные закономерности.

Гистограммы представляют количественные параметры важных признаков количественного типа. Было отмечено 6 колонок:

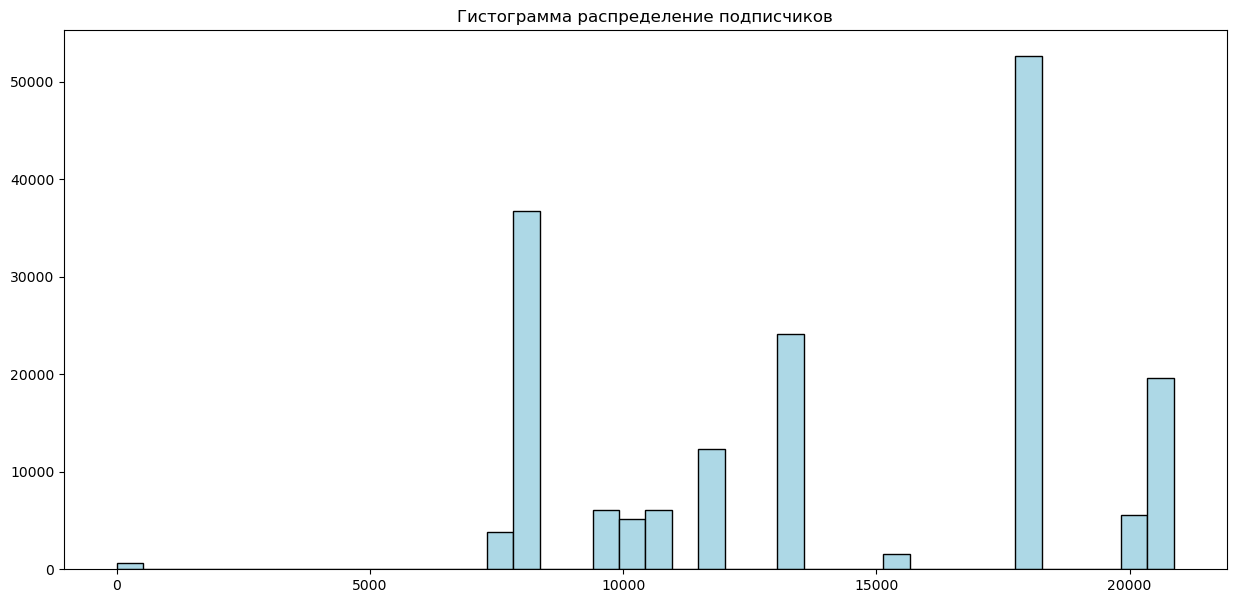


Рисунок 4. Гистограмма распределения количества подписчиков на весь обработанный датасет

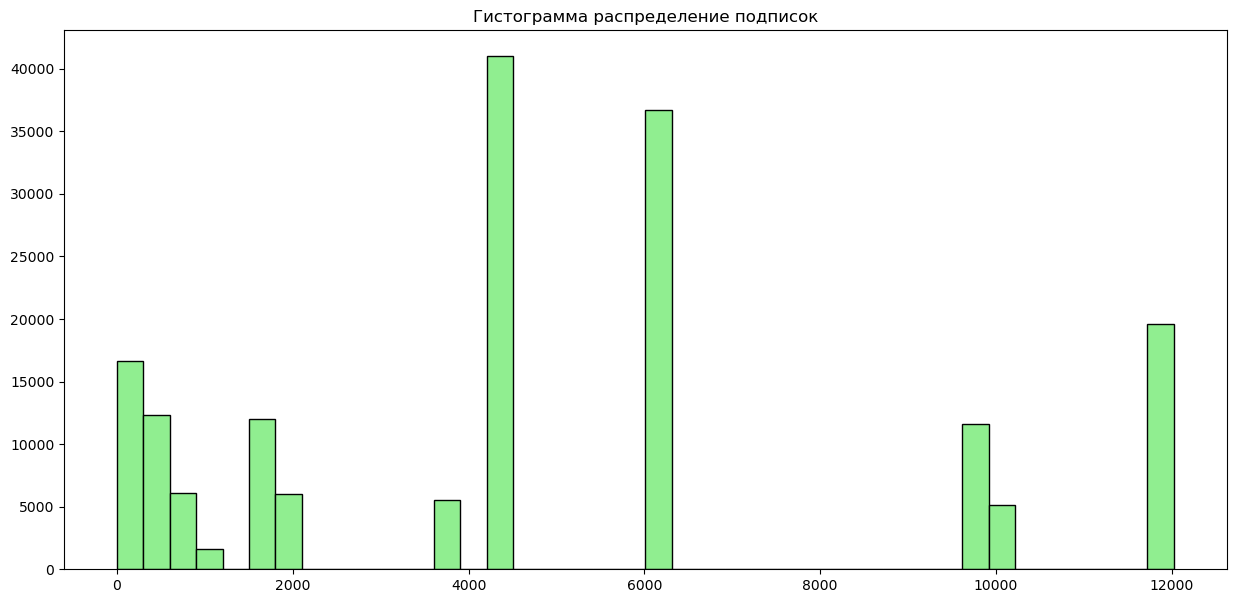


Рисунок 5. Гистограмма распределения количества подписок на весь обработанный датасет

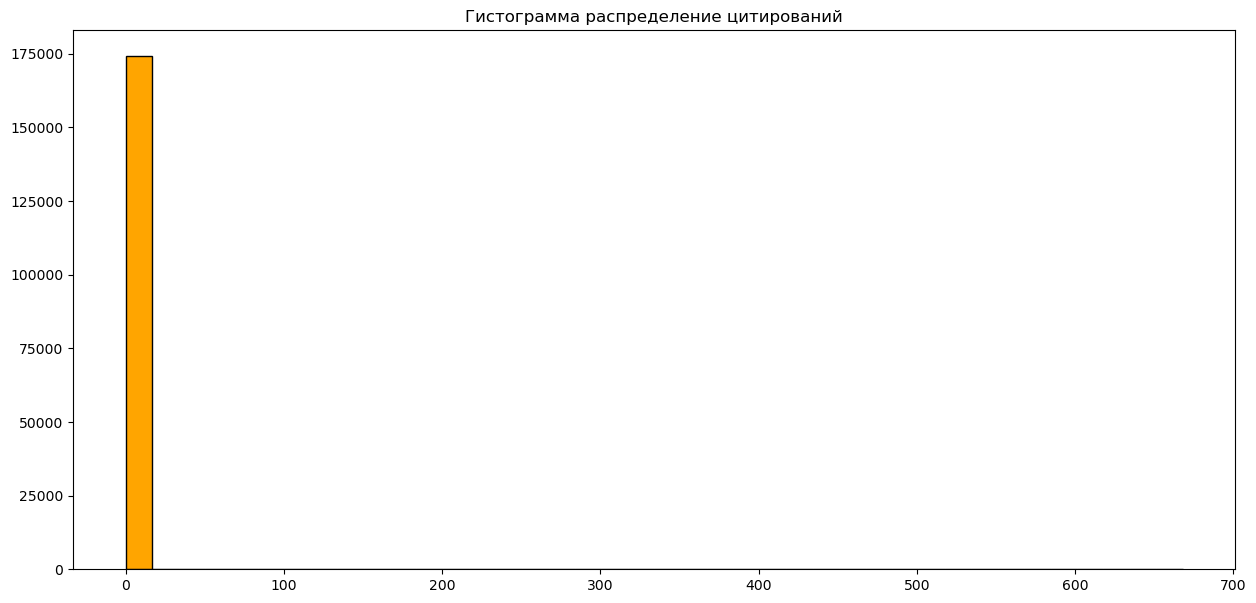


Рисунок 6. Гистограмма распределения количества цитирований поста на весь обработанный датасет

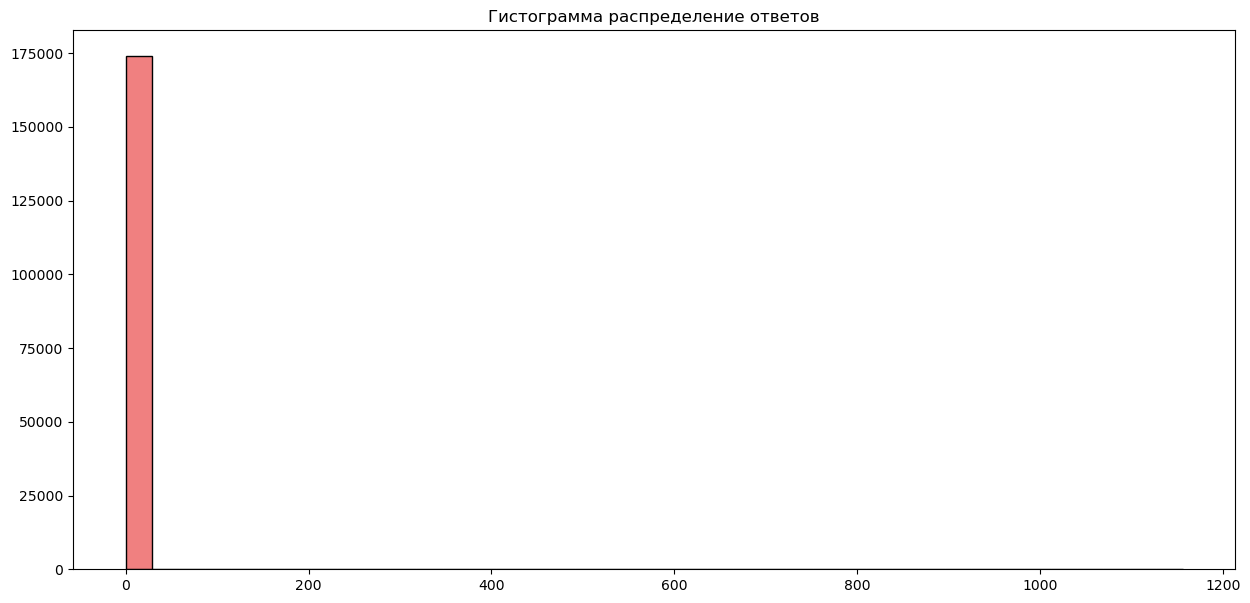


Рисунок 7. Гистограмма распределения количества ответов на пост на весь обработанный датасет

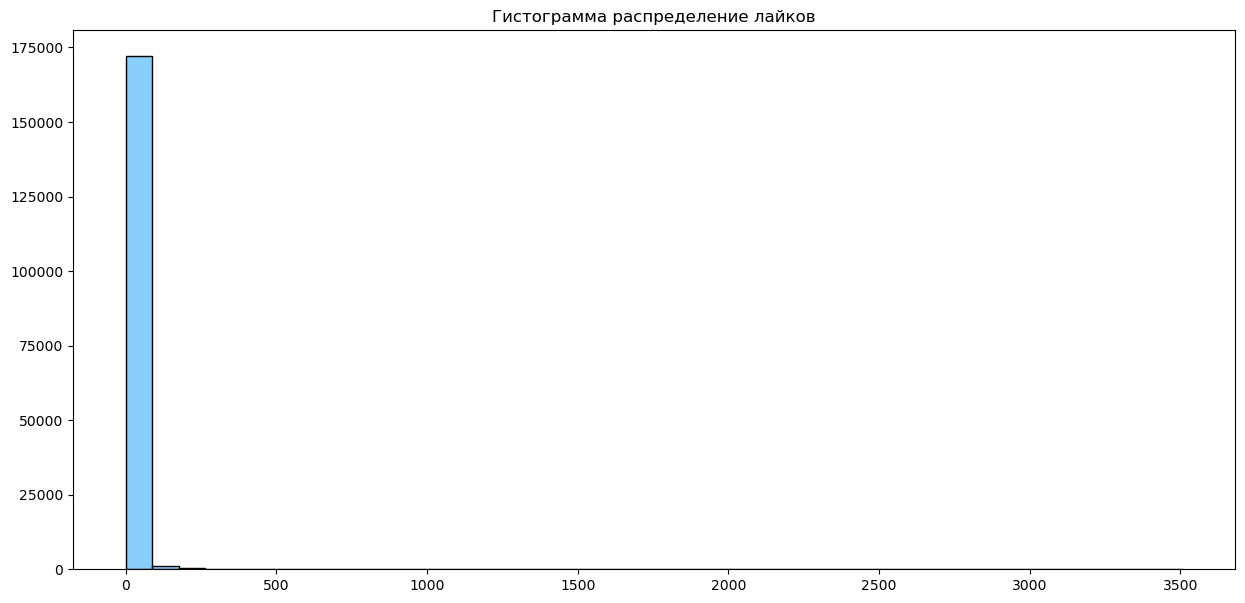


Рисунок 8. Гистограмма распределения количества лайков на весь обработанный датасет

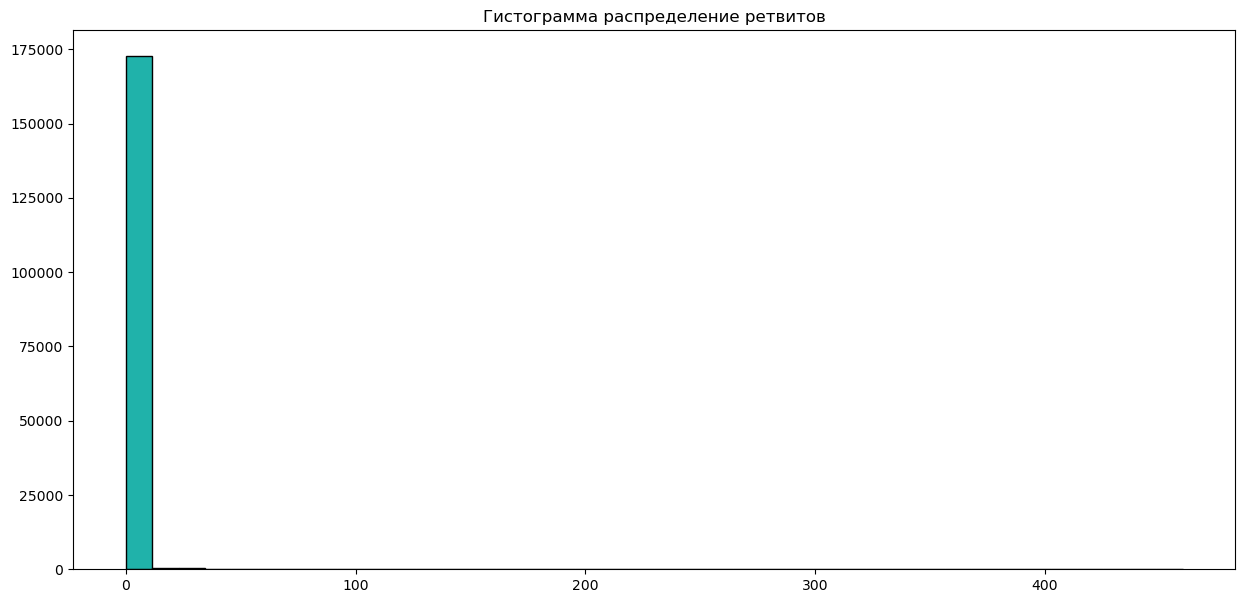


Рисунок 9. Гистограмма распределения количества ретвитов на весь обработанный датасет

Из представленных гистограмм видно то, что колонки цитирований, ответов, лайков и ретвитов очень близки к одинаковому значению и похожи друг на друга. Их количество в большей степени равны нулю и есть немного строк, в которых имеются метрики. Распределение колонок подписок и подписчиков слабо связаны и разнятся друг с другом.

Круговая диаграмма представляют количественные параметры бинарного признака для визуализации данного признака. На рис. ниже представлена диаграмма данного признака.

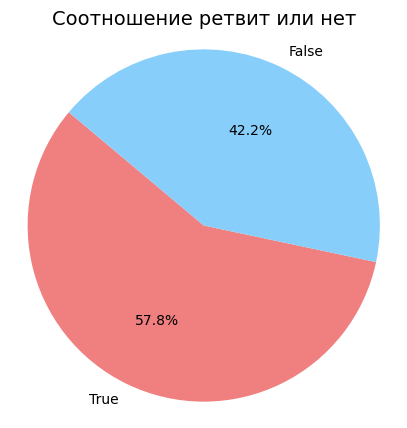
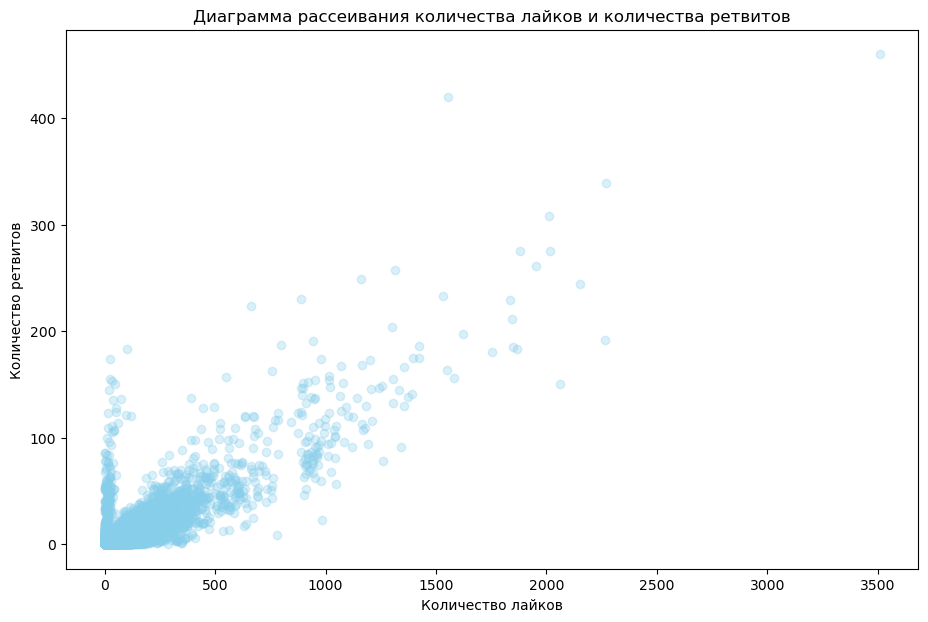


Рисунок 10. Круговая диаграмма бинарного признака

Далее рассмотрим диаграммы рассеивания, которые позволяет сравнить пару колонок друг с другом и наблюдать взаимосвязи между ними.

 Рисунок 11. Диаграмма рассеивания лайков-ретвитов

Из графика то, что данные показатели в большинстве находятся у нулевых значений и в небольших случаях могут пересекаться при других метриках, отличных от нуля.

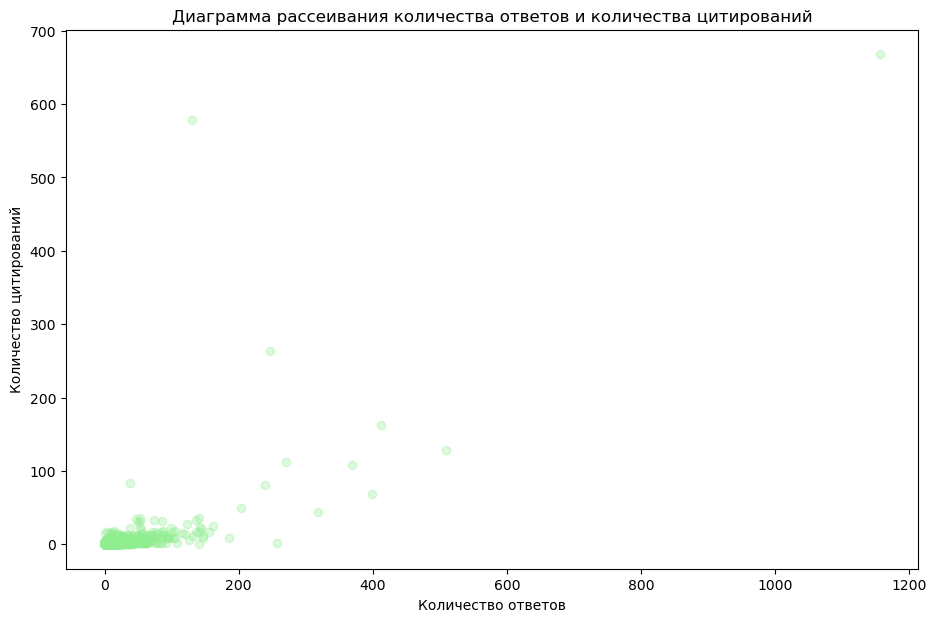


Рисунок 12. Диаграмма рассеивания ответов-цитирований

Здесь уже рассматриваем две других колонки данные, которых на гистограммах были схожи, а именно количество цитирований и ответов. Видно, что также чаще всего они находятся в нулевом диапазоне и гораздо реже пересекаются в других местах, как на рис. 11.

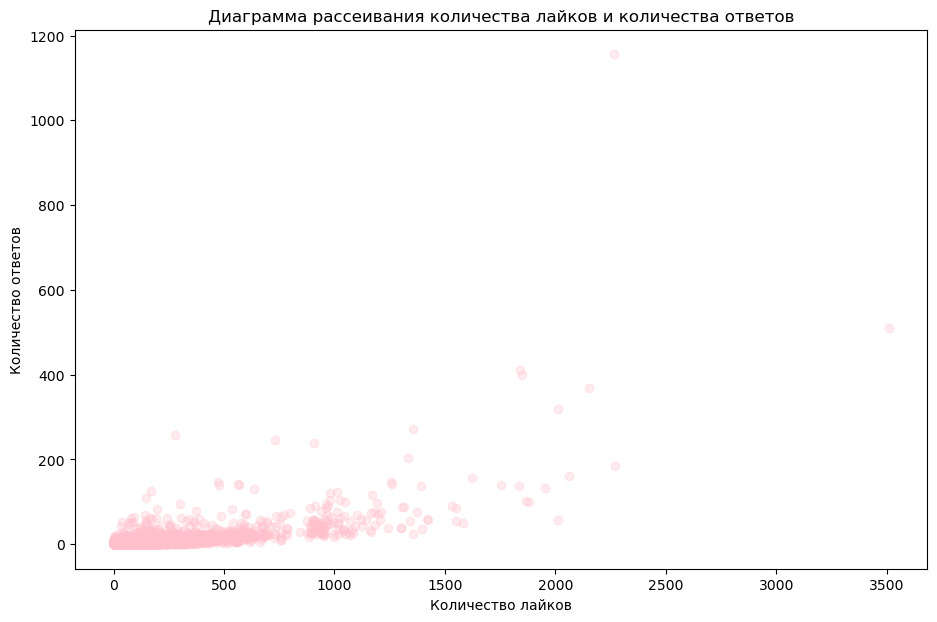


Рисунок 13. Диаграмма рассеивания лайков-ответов

Сравнение количества лайков и количества ответов, с рис. 12, т.к. из предыдущего графика видно, что количество ответов может встречаться чаще в большей метрике, чем цитирований. И из рис. 13 видно что множество точек пересекаются от диапозона 0-1000 с постепенным увеличением ответов.

Такие графики можно сформировать для всех 4 колонок, которые очень близки по значению друг с другом и получится 6 различных графиков. Но у нас еще есть поля подписчиков и подписок.

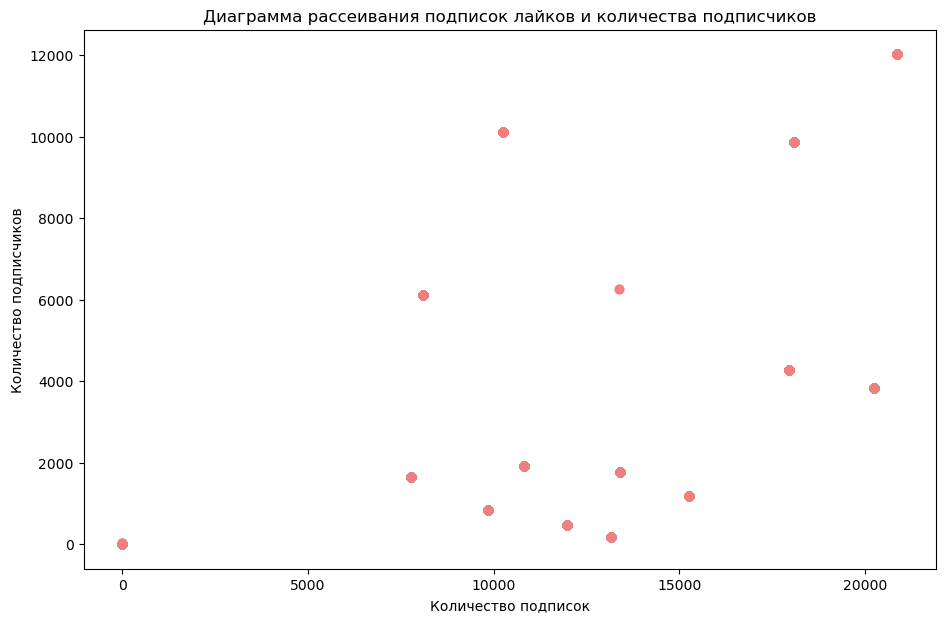


Рисунок 14. Диаграмма рассеивания подписчики-подписок

Из графика видно, что колонки никак не связаны и их значения просто разбросаны по диаграмме. Из диаграммы видно, что из-за того, что точек мало, значения этих показателей часто совпадают друг с другом. Это говорит о том, что возможно это могут быть уникальные пользователи, если не брать в расчёт точку с двумя нулями. Ведь редко когда у уникальных пользователей могут совпасть метрики.

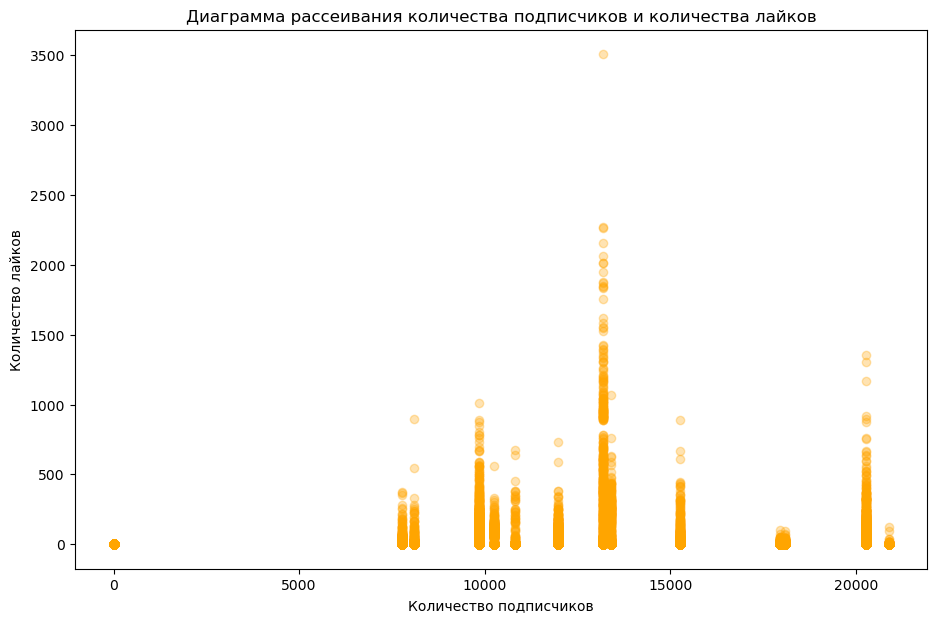


Рисунок 15. Диаграмма рассеивания подписчиков-лайков

Из графика видно, что при любом количестве подписчиков у пользователя и при увеличении метрики лайков точки становятся все прозрачней. Это говорит о том, что множество постов остаются с малым количеством активности вне зависимости от популярности пользователя.

* 1. **Корреляций между признаками**

Чтобы выявить связи между признаками, нужно построить матрицу корреляций.

Поля "tweetid" и "userid" нет необходимости включать в списки для анализа корреляции. Идентификаторы обычно не имеют значения для анализа корреляций и могут даже помешать получению точных результатов.

Корреляцию будем считать только по количественным признакам. В данном датасете их шесть:

1. 'follower\_count' - Количество подписчиков

2. 'following\_count' - Количество подписок

3. 'quote\_count' - Количество цитирований

4. 'reply\_count' - Количество ответов

5. 'like\_count' - Количество лайков

6. 'retweet\_count' - Количество ретвитов

Матрица корреляции выглядит следующим образом:

*Таблица 5. Матрица корреляции*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *follower\_count* | *following\_count* | *quote\_count* | *reply\_count* | *like\_count* | *retweet\_count* |
| *1.0* | *0.38* | *0.001* | *-0.016* | *-0.02* | *0.01* |
| *0.38* | *1.0* | *-0.015* | *-0.063* | *-0.096* | *-0.057* |
| *0.001* | *-0.015* | *1.0* | *0.707* | *0.302* | *0.241* |
| *-0.016* | *-0.063* | *0.707* | *1.0* | *0.654* | *0.544* |
| *-0.02* | *-0.096* | *0.302* | *0.654* | *1.0* | *0.857* |
| *0.01* | *-0.057* | *0.241* | *0.544* | *0.857* | *1.0* |

Обнаружено множество отрицательных корреляций между числом подписчиков/подписок и ответами/цитирований/лайками/ретвитами.

У колонок "reply\_count", "like\_count", "retweet\_count" и "quote\_count" корреляция находятся в диапозон 0.241 – 0.857.

Самая сильная связь имеется у колонок "like\_count" и "retweet\_count".

Корреляции "диагональных" признаков всегда равны 1, т.к. это и есть те же самые признаки и они всегда будут изменяться, если их изменяют.

* 1. **Выводы**

В ходе работы был изучен функционал Apache PySpark. Я научился пользоваться функционалом Apache Spark благодаря Python API - PySpark. Целью работы было провести разведочный анализ датасета. В ходе работы были выполнены данные пункты:

* 1. Типы признаков в датасете: Определил, какие данные включены в датасет (числовые, категориальные и бинарные).
  2. Пропущенные значения и их устранения: Определил наличие пропущенных значений в данных и их устранил. Строки с пропущенными значениями было решено полностью удалить.
  3. Выбросы и их устранения: Провел работу по устранению выбросов используя метод межквартильного интервала, а после проверил их отсутствие с помощью boxspot-ов.
  4. Расчет статистических показателей признаков (средних, квартилей и т.д.): Провел расчёт статистических показателей категориальных, бинарных и числовых признаков, которые включают: средние значения, медианы, квартили, моды, стандартные отклонения, диапазоны значений, квантили, процентные соотношения и другие характеристики, чтобы лучше понять распределение данных.
  5. Визуализация распределения наиболее важных признаков: Выполнил визуализацию распределения наиболее важных признаков. Визуализация помогает визуально оценить данные и выявить особенности.
  6. Корреляция между признаками: Исследовал корреляции между признаками, позволяющие понять, какие признаки взаимосвязаны между собой, а какие – нет.

Разведочный анализ помогает лучше понять характеристики датасета, подготовить данные для построения моделей и выявить интересные закономерности, которые могут стать основой для более глубокого исследования.

1. **Машинное обучение на больших данных**
   1. **Постановка задачи**

Цель и задачи работы:

1. Познакомиться с базовыми алгоритмами машинного обучения;

2. Познакомиться с реализацией машинного обучения в библиотеке Spark ML.

3. Получить навыки разработки программного обеспечения для анализа данных с использованием pyspark.

Необходимо выполнить анализ обработанного датасета с помощью двух алгоритмов машинного обучения:

* Задача регрессии - GradientBoostingMachine
* Задача бинарной классификации - LogisticRegression

При анализе датасета предпочтительно использовать признаки, показавшие наилучшую корреляцию при выполнении разведочного анализа. Для задачи классификации использовать бинарный признак.

Необходимо выполнить обучение и валидацию модели, рассчитайте значения метрик классификации и регрессии. Выполните подбор гиперпараметров моделей по сетке.

* 1. **Применение алгоритмов подготовки данных и машинного обучения**

После осуществления разведочного анализа был сохранен датасет. Подключаем все необходимые библиотеки и отфильтрованный датасет.

*Таблица 6. Исходный код подключения библиотек и датасета*

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  1516  17  18 | from pyspark.sql.types import \*  from pyspark.sql.functions import \*  from pyspark.sql import SparkSession  from pyspark.ml import Pipeline  from pyspark.ml.feature import VectorAssembler, StringIndexer, VectorIndexer, MinMaxScaler  from pyspark.ml.regression import GBTRegressor  from pyspark.ml.classification import LogisticRegression  from pyspark.ml.tuning import ParamGridBuilder, CrossValidator  from pyspark.ml.evaluation import RegressionEvaluator, BinaryClassificationEvaluator  spark = SparkSession.builder.master("local[\*]").getOrCreate()  filename\_data = 'processed\_data/iran\_201906\_2\_tweets\_csv\_hashed.csv'  csv = spark.read.csv(filename\_data, inferSchema=True, header=True)  csv.show(10)  csv.printSchema() |

И выведем первые 10 строк датасета и его типы колонок для проверки подключения.

* 1. **Задача регрессии – GradientBoostingMachine**

Для начала необходимо отсортировать колонки и выбрать только те, что показали наилучшую корреляцию в разведочном анализе. Было решено использовать такие колонки, как:

1. 'quote\_count' - Количество цитирований

2. 'reply\_count' - Количество ответов

3. 'like\_count' - Количество лайков

4. 'retweet\_count' - Количество ретвитов

Далее разделим данные на обучающую и тестовую выборку: Будем использовать 70% данных для обучения и 30% для тестирования.

После применения алгоритмы имеем на выходе: Тренировочных строк – 121833 и тестовых строк – 52433.

Осуществим разделение датасета приступаем к определению конвейера:

Конвейер состоит из серии этапов преобразования и оценки, которые обычно подготавливают фрейм данных для моделирования, а затем обучают прогнозирующую модель. В этом случае вы создадите конвейер с тремя этапами:

1. VectorAssembler: Создает вектор непрерывных числовых признаков.

2. MinMaxScaler: Нормализует непрерывные числовые характеристики.

3. Gradient-boosted tree regression: Обучает модель регрессии.

*Таблица 7. Исходный код настройка конвейера для задачи регрессии*

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | numVect = VectorAssembler(inputCols = ["quote\_count", "reply\_count", "retweet\_count"], outputCol="numFeatures")  minMax = MinMaxScaler(inputCol = numVect.getOutputCol(), outputCol="features")  gbtr = GBTRegressor(featuresCol = 'features', labelCol='like\_count', maxIter = 5, maxDepth=2)  pipeline = Pipeline(stages=[numVect, minMax, gbtr]) |

Как видно из табл. 7, будем предсказывать значение колонки "like\_count" на основе данных из колонок "quote\_count", "reply\_count" и "retweet\_count" с применением в задачи регрессии 5 итераций градиентного бустинга и с глубиной дерева равной 2.

После настройки конвейера, запускаем его на тренировочных данных.

А после этого, на основе тестовых данных, генерируем предсказываемые значения, а для оценки модели используем такие метрики, как:

* 1. RMSE (Root Mean Square Error) - Это классическая метрика, используемая для измерения точности модели в задачах регрессии. RMSE представляет собой квадратный корень из среднеквадратичной ошибки (MSE) и позволяет оценить, насколько сильно отличаются предсказанные значения от фактических. Чем ближе RMSE к нулю, тем лучше модель справляется с прогнозированием. RMSE подвержен влиянию на выбросы в данных, так как он учитывает квадраты отклонений предсказаний от фактических значений.
  2. MSE (Mean Squared Error) - Является средним значением квадратов разностей между предсказанными и фактическими значениями. Чем ниже значение MSE, тем лучше качество модели. Подобно RMSE, MSE также наказывает за большие ошибки из-за возведения их в квадрат.
  3. R2 (R-squared) - Это коэффициент детерминации, который предоставляет информацию о том, как хорошо модель соотносится с данными. R2 принимает значения от 0 до 1, где 1 означает идеальное соответствие модели данным. Он позволяет сравнивать модель с простым средним значением зависимой переменной. Чем ближе R2 к 1, тем лучше модель объясняет изменчивость зависимой переменной.
  4. MAE (Mean Absolute Error) - Это среднее абсолютное значение разностей между предсказанными и фактическими значениями. MAE менее подвержен выбросам, чем MSE и RMSE, так как не возводит отклонения в квадрат и, следовательно, не учитывает их взаимную агрессию.

В результате предсказаний имеем такие результаты метрик:

RMSE = 33.60

MSE = 1129.00

R2 = 0.35

MAE = 3.49

Полученные данные можно оптимизировать и улучшить показатели метрик с помощью применения параметрической сетки и кросс-валидации модели.

Установим такие гиперпараметры модели градиентного бустинга:

* Задаем значения для максимальной глубины деревьев: 2, 4 и 6.
* Устанавливаем различные значения для максимального числа бинов: 20 и 60.
* Определяем различные значения для максимального числа итераций: 10 и 20.

И передаем эти параметры в метод кросс-валидации. Запускаем конвейер на тренировочных данный и генерируем предсказания на тестовых, как и ранее. На этот раз у нас имеется больше параметров и в результате предсказаний имеем такие результаты метрик:

RMSE = 2.39

MSE = 957.87

R2 = 0.45

MAE = 2.39

* 1. **Задача классификации – LogisticRegression**

Для задачи классификации будет использован бинарный признак «is\_retweet», который уже был в датасете изначально, поэтому создавать новый признак не имеет смысла.

Для начала необходимо преобразовать колонку, в которой будут значения: 1 – true / 0 – false. Данную колонку решено было назвать «label». По итогу сформированный датасет для создания модели влючает следующие колонки:

1. 'follower\_count' - Количество подписчиков

2. 'following\_count' - Количество подписок

3. 'quote\_count' - Количество цитирований

4. 'reply\_count' - Количество ответов

5. 'like\_count' - Количество лайков

6. 'retweet\_count' - Количество ретвитов

7. 'label' – Бинарный признак является ли пост ретвитом

Повторно формируем тренировочные и тестовые выборки, с такими же процентами, где тренировочных строк – 121960, а тестовых – 52306.

Осуществим разделение датасета приступаем к определению конвейера:

Конвейер состоит из серии этапов преобразования и оценки, которые обычно подготавливают фрейм данных для моделирования, а затем обучают прогнозирующую модель. В этом случае мы также создаем конвейер с тремя этапами:

1. VectorAssembler: Создает вектор непрерывных числовых признаков.

2. MinMaxScaler: Нормализует непрерывные числовые характеристики.

3. Logistic Regression: Обучает модель классификации.

*Таблица 8. Исходный код настройка конвейера для задачи классификации*

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | numVect = VectorAssembler(inputCols = ["follower\_count", "following\_count", "quote\_count", "reply\_count", "like\_count", "retweet\_count"], outputCol="numFeatures")  minMax = MinMaxScaler(inputCol = numVect.getOutputCol(), outputCol="features")  lr = LogisticRegression(labelCol="label",  featuresCol="features",  maxIter=10,  regParam=0.3)  pipeline = Pipeline(stages=[numVect, minMax, lr]) |

Как видно из табл. 8, будем предсказывать значение колонки "label " на основе данных из колонок "quote\_count", "reply\_count", "retweet\_count", "like\_count", "following\_count" и "follower\_count" с применением в задачи классификаций 10 итераций логистической регрессии и параметр регуляризации равным 0.3, который помогает предотвратить переобучение путем штрафования больших весов в модели.

После настройки конвейера, запускаем его на тренировочных данных.

А после этого, на основе тестовых данных, генерируем предсказываемые значения, а для оценки модели классификации используем такие метрики, как:

1. Первая метрика это True Positive (TP). TP представляет собой количество истинных положительных результатов, т.е. случаев, когда модель правильно предсказала положительный результат.
2. Далее у нас есть False Positive (FP), который отражает количество ложноположительных результатов. Это случаи, когда модель неправильно предсказала положительный результат.
3. True Negative (TN), который показывает количество истинно отрицательных результатов - ситуации, когда модель правильно предсказала отрицательный результат.
4. И False Negative (FN) - количество ложноотрицательных результатов, т.е. случаи, когда модель неправильно предсказала отрицательный результат.
5. Precision (точность) трактуется как отношение TP к сумме TP и FP. Это означает, что точность измеряет, насколько много из предсказанных моделью положительных случаев действительно являются положительными.
6. Recall (полнота) определяется как отношение TP к сумме TP и FN. Полнота оценивает, насколько много истинных положительных случаев было правильно предсказано моделью.
7. Наконец, F1-мера - это гармоническое среднее между точностью и полнотой. Она вычисляется как 2 \* Precision \* Recall / (Precision + Recall) и предоставляет баланс между точностью и полнотой модели.

На выходе, после обучения модели, имеем такие метрики:

*Таблица 9. Метрики модели классификации*

|  |  |
| --- | --- |
| Metric | value |
| TP | 28894.0 |
| FP | 10637.0 |
| TN | 11444.0 |
| FN | 1331.0 |
| Precision | 0.7309200374389719 |
| Recall | 0.9559636062861869 |
| F1 | 0.8284305292734675 |

Также есть возможность осуществить оценку качества модели бинарной классификации с использованием метрики AUR.

AUC (Area Under the ROC Curve) - это метрика, используемая для оценки качества модели бинарной классификации. Она представляет собой площадь под кривой ROC (Receiver Operating Characteristic).

Кривая ROC строится путем варьирования порога классификации и вычисления двух показателей: True Positive Rate (чувствительность) и False Positive Rate (специфичность).

- True Positive Rate (TPR) представляет собой долю верно предсказанных положительных примеров относительно всех истинных положительных примеров.

- False Positive Rate (FPR) представляет собой долю ложно предсказанных положительных примеров относительно всех истинных отрицательных примеров.

AUC-ROC описывает производительность модели на всем спектре возможных порогов классификации, объединяя в себе информацию из всех возможных значений порога. Чем ближе значение AUC к 1, тем лучше модель способна различать между классами. Если AUC = 0.5, это означает, что модель дает случайные предсказания без различия между классами.

Метрика AUC, в обученной модели, равна: 0.8111522719437847

Для метода бинарной классификации также можно применять гиперпараметры модели по сетке и кросс-валидацию. В нашем случае, параметры сетки для модели: варьируемые значения параметра регуляризации [0.6, 0.3], параметры максимального числа итераций [10, 20] и значения порогов бинарной классификации [0.5, 0.3].

И с помощью кросс-валидации осуществим поиск наилучшего значения. Запускаем конвейер, генерируем предсказания на тестовой выборке, а после формируем метрики:

*Таблица 10. Метрики модели классификации после кросс-валидации*

|  |  |
| --- | --- |
| Metric | value |
| TP | 28894.0 |
| FP | 10637.0 |
| TN | 11444.0 |
| FN | 1331.0 |
| Precision | 0.7309200374389719 |
| Recall | 0.9559636062861869 |
| F1 | 0.8284305292734675 |

А также метрика AUC равна: 0.7371186175989604

* 1. **Анализ результатов**

В ходе решений задач регрессии и классификации были получены некоторые результаты. В ходе задачи регрессии метрики описаны в табл. 11.

*Таблица 11. Метрики задачи регрессии*

|  |  |  |
| --- | --- | --- |
|  | *До кросс-валидации* | *После кросс-валидации* |
| *RMSE:* | *33.60* | *2.39* |
| *MSE:* | *1129.00* | *957.87* |
| *R2:* | *0.35* | *0.45* |
| *MAE:* | *3.49* | *2.39* |

В ходе анализа полученных результатов видно то, что метрики RMSE, MSE и MAE уменьшились, а метрика R2 – увеличилась. То есть в результате уменьшения метрики RMSE – модель лучше справляется с прогнозированием, т.к. показатель стремится к 0. MSE – качество модели стала лучше, т.к. эта метрика в идеале также стремится к 0. Из-за MAE – модель также лучше справляется с прогнозированием, т.к. тоже показатель должен уменьшаться. А вот R2 – стал больше, что говорит о изменении коэффициента детерминации и говорит о том, что модель лучше справляется с предсказанием, чем изначально. В итоге имеем что изменение всех показателей после кросс-валидации улучшили модель предсказания.

*Таблица 12. Метрики задачи классификации*

|  |  |  |
| --- | --- | --- |
|  | *До кросс-валидации* | *После кросс-валидации* |
| *TP:* | *28894.0* | *28894.0* |
| *FP:* | *10637.0* | *10637.0* |
| *TN:* | *11444.0* | *11444.0* |
| *FN:* | *1331.0* | *1331.0* |
| *Precision* | *0.7309200374389719* | *0.7309200374389719* |
| *Recall* | *0.9559636062861869* | *0.9559636062861869* |
| *F1* | *0.8284305292734675* | *0.8284305292734675* |
| *AUC* | *0.8284305292734675* | *0.7371186175989604* |

В ходе анализа результатов задачи классификации видно то, что с применением кросс-валидации и без неё значения показателей не изменились и говорит о том, что модель стабильна и отсутствует переобучение модели. А метрика AUC хоть и изменилась, но не равняется приближенной к 0.5, а это значит, что модель предсказывает значения не случайно.

* 1. **Вывод**

В ходе работы был изучен функционал Apache PySpark. Я научился пользоваться функционалом машинного обучения Apache Spark благодаря Python API - PySpark. Целью работы было провести машинное обучение на больших данных. Был выполнен анализ с помощью двух алгоритмов машинного обучения:

Задача регрессии: GradientBoostingMachine

Задача бинарной классификации: LogisticRegression

Было выполнено обучение и валидация модели, рассчитаны значения метрик классификации и регрессии, а также выполнен подбор гиперпараметров моделей по сетке.