Минобрнауки России

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Волгоградский государственный технический университет»

Факультет Электроники и вычислительной техники   
Направление (специальность) Информатика и вычислительная техника . .  
Кафедра Электронно-вычислительные машины и системы"

Дисциплина Системы обработки больших данных

|  |  |
| --- | --- |
|  | Утверждаю  Зав. кафедрой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | «\_\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20 \_\_\_ г. |

ЗАДАНИЕ

на курсовую работу (проект)

Студент Меркушов Михаил Сергеевич .

(фамилия, имя, отчество)

Группа САПР-1.1

1. Тема: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Утверждена приказом от «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_ г. № \_\_\_\_\_\_\_\_\_

2. Срок представления работы (проекта) к защите «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_ г.

3. Содержание расчетно-пояснительной записки: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. Перечень графического материала: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. Дата выдачи задания «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20 \_\_\_ г.

Руководитель работы (проекта)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата инициалы и фамилия

Задание принял к исполнению\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_ .

подпись, дата инициалы и фамилия

Минобрнауки России

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Волгоградский государственный технический университет»

Факультет \_Электроники и вычислительной техники\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра Системы автоматизированного проектирования и поискового конструирования

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовой работе (проекту)**

по дисциплине Системы обработки больших данных

на тему .

Студент Меркушов Михаил Сергеевич.

(фамилия, имя, отчество)

Группа САПР-1.1

Руководитель работы (проекта) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись и дата подписания) (инициалы и фамилия)

Члены комиссии:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись и дата подписания) (инициалы и фамилия)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись и дата подписания) (инициалы и фамилия)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись и дата подписания) (инициалы и фамилия)

Нормоконтролер \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата подписания) (инициалы и фамилия)

Волгоград 2023 г.

Содержание

[Введение 4](#_Toc152286698)

[1. Разведочный анализ данных с помощью PySpark 5](#_Toc152286699)

[1.1 Постановка задачи 5](#_Toc152286700)

[1.2 Определение типов признаков в датасете 5](#_Toc152286701)

[1.3 Определение пропущенных значений и их устранение 9](#_Toc152286702)

[1.4 Определение и удаление выбросов 11](#_Toc152286703)

[1.5 Расчет статистических показателей признаков 12](#_Toc152286704)

[1.6 Визуализацией распределения наиболее важных признаков 15](#_Toc152286705)

[1.7 Корреляций между признаками 16](#_Toc152286706)

# Введение

В современном мире, где объемы данных растут с небывалой скоростью, специалисты во всех отраслях сталкиваются с необходимостью их анализа и обработки. Эти данные, известные как "большие данные" (Big Data), характеризуются своим объемом, скоростью поступления и разнообразием, что ставит перед исследователями и инженерами новые задачи и требует особого подхода к их обработке.

Ключевым инструментом был выбран PySpark - интерфейс Apache Spark для языка программирования Python. Используя PySpark, на практике были освоены основы работы с RDD (Resilient Distributed Datasets) и DataFrame API, которые являются ключевыми абстракциями в Apache Spark, позволяя обрабатывать данные эффективно и интуитивно понятно.

# 1. Разведочный анализ данных с помощью PySpark

## 1.1 Постановка задачи

Выполнить разведочный анализ датасета авиабилетов из Expedia согласно варианту 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.

<https://www.kaggle.com/datasets/dilwong/flightprices>

## 1.2 Определение типов признаков в датасете

Датасет содержит информацию о ценах на полеты в одну сторону самолетами согласно Expedia на период с 16.04.2022 до 05.10.2022.

Датасет представляет собой CSV-файл, где каждая строка - купленный билет в/из аэропортов(ATL, DFW, DEN, ORD, LAX, CLT, MIA, JFK, EWR, SFO, DTW, BOS, PHL, LGA, IAD, OAK).

Определения столбцов:

1. legId: Идентификатор рейса.
2. searchDate: дата (ГГГГ-ММ-ДД), когда эта запись была взята из Expedia.
3. flightDate: дата (ГГГГ-ММ-ДД) полета.
4. startingAirport: код аэропорта IATA для начального местоположения.
5. destinationAirport: код аэропорта IATA для места прибытия.
6. fareBasisCode: код основного тарифа.
7. travelDuration: длитеельность поездки в часах и минутах.
8. elapsedDays: количество дней в пути
9. isBasicEconomy: флаг, указывающий предназначен ли билет для базовой экономики.
10. isRefundable: флаг, возвратный ли билет.
11. isNonStop: флаг, беспосадочный билет или нет.
12. baseFare: цена билета (в USD).
13. totalFare: цена билета (в USD с налогами.
14. seatsRemaining: целое число для количества оставшихся мест.
15. totalTravelDistance: расстояние поездки в милях.
16. segmentsDepartureTimeEpochSeconds: строка, содержащая время отправления (время Unix) для каждого этапа поездки, записи для каждого этапа разделяются символом '||'.
17. segmentsDepartureTimeRaw: строка, содержащая время отправления (формат ISO 8601: YYYY-MM-DDThh:mm:ss.000±[hh]:00) для каждого этапа поездки. Записи для каждого из этапов разделяются символом '||'.
18. segmentsArrivalTimeEpochSeconds: строка, содержащая время прибытия (время Unix) для каждого этапа поездки. Записи для каждого этапа разделяются символом '||'.
19. segmentsArrivalTimeRaw: строка, содержащая время прибытия (формат ISO 8601: YYYY-MM-DDThh:mm:ss.000±[hh]:00) для каждого этапа поездки. Записи для каждого из этапов разделяются символом '||'.
20. segmentsArrivalAirportCode: строка, содержащая код аэропорта IATA для места прибытия для каждого этапа поездки. Записи для каждого этапа разделяются символом '||'.
21. segmentsDepartureAirportCode: строка, содержащая код аэропорта IATA для места отправления на каждом этапе путешествия. Записи для каждого этапа разделяются символом '||'.
22. segmentsAirlineName: строка, содержащая название авиакомпании, обслуживающей каждый этап путешествия. Записи для каждого этапа разделяются символом '||'.
23. segmentsAirlineCode: строка, содержащая двухбуквенный код авиакомпании, обслуживающей каждый этап путешествия. Записи для каждого этапа разделяются символом '||'.
24. segmentsEquipmentDescription: строка, содержащая тип самолета, используемого на каждом этапе путешествия (например, "Airbus A321" или "Boeing 737-800"). Записи для каждого из этапов разделяются символом '||'.
25. segmentsDurationInSeconds: строка, содержащая продолжительность полета (в секундах) для каждого этапа путешествия. Записи для каждого этапа разделяются символом '||'.
26. segmentsDistance: строка, содержащая пройденное расстояние (в милях) для каждого этапа поездки. Записи для каждого этапа разделяются символом '||'.
27. segmentsCabinCode: строка, содержащая код салона для каждого этапа поездки (например, "coach"). Записи для каждого этапа разделяются символом '||'.

Данные представляют собой значения следующих типов данных: integer, double, string, date, boolean. Типы колонок представлены в таблице 1

| № | Название колонки | Тип данных |
| --- | --- | --- |
| 0 | legId | string |
| 1 | searchDate | date |
| 2 | flightDate | date |
| 3 | startingAirport | string |
| 4 | destinationAirport | string |
| 5 | fareBasisCode | string |
| 6 | travelDuration | string |
| 7 | elapsedDays | int |
| 8 | isBasicEconomy | boolean |
| 9 | isRefundable | boolean |
| 10 | isNonStop | boolean |
| 11 | baseFare | double |
| 12 | totalFare | double |
| 13 | seatsRemaining | int |
| 14 | totalTravelDistance | int |
| 15 | segmentsDepartureTimeEpochSeconds | string |
| 16 | segmentsDepartureTimeRaw | string |
| 17 | segmentsArrivalTimeEpochSeconds | string |
| 18 | segmentsArrivalTimeRaw | string |
| 19 | segmentsArrivalAirportCode | string |
| 20 | segmentsDepartureAirportCode | string |
| 21 | segmentsAirlineName | string |
| 22 | segmentsAirlineCode | string |
| 23 | segmentsEquipmentDescription | string |
| 24 | segmentsDurationInSeconds | string |
| 25 | segmentsDistance | string |
| 26 | segmentsCabinCode | string |

Всего в датасете 1047402 строк.

## 1.3 Определение пропущенных значений и их устранение

Для корректного анализа данных необходимо убедиться, что датасет не имеет какие-либо пропущенные и аномальные нулевые значения.

* Для строковых колонок проверим на None и null;
* Для числовых колонок, допускающих значения ноль, проверим на None и NaN;
* Для числовых колонок, недопускающих значения ноль, проверим на нули, None и NaN;
* Для булевых колонок проверим на None и null;
* Для колонок с датами проверим на None и null.

С помощью кода найдено количество пустых и нулевых значений, а также в каких колонках были найдены эти значения.

string\_columns = [

    'legId', 'startingAirport', 'destinationAirport', 'fareBasisCode', 'travelDuration',

    'segmentsDepartureTimeEpochSeconds', 'segmentsDepartureTimeRaw', 'segmentsArrivalTimeEpochSeconds',

    'segmentsArrivalTimeRaw', 'segmentsArrivalAirportCode', 'segmentsDepartureAirportCode',

    'segmentsAirlineName', 'segmentsAirlineCode', 'segmentsEquipmentDescription', 'segmentsDurationInSeconds',

    'segmentsDistance', 'segmentsCabinCode'

]

numeric\_with\_zeroes\_columns = ['elapsedDays', 'seatsRemaining']

numeric\_without\_zeroes\_columns = ['baseFare', 'totalFare', 'totalTravelDistance']

boolean\_columns = ['isBasicEconomy', 'isRefundable', 'isNonStop']

date\_columns = ['searchDate', 'flightDate']

missing\_values = {}

for index, column in enumerate(df.columns):

    if column in string\_columns: # check None and Null

        missing\_count = df.filter(col(column).eqNullSafe(None) | col(column).isNull()).count()

        missing\_values.update({column:missing\_count})

    if column in numeric\_with\_zeroes\_columns: # check None, NaN and Null

        missing\_count = df.filter(col(column) == None | isnan(col(column)) | col(column).isNull()).count()

        missing\_values.update({column:missing\_count})

    if column in numeric\_without\_zeroes\_columns:  # check zeroes, None, NaN and Null

        missing\_count = df.filter(col(column).isin([0,None]) | isnan(col(column)) | col(column).isNull()).count()

        missing\_values.update({column:missing\_count})

    if column in boolean\_columns:  # check None and Null

        missing\_count = df.filter(col(column).eqNullSafe(None) | col(column).isNull()).count()

        missing\_values.update({column:missing\_count})

    if column in date\_columns:  # check None and Null

        missing\_count = df.filter(col(column).eqNullSafe(None) | col(column).isNull()).count()

        missing\_values.update({column:missing\_count})

missing\_df = pd.DataFrame.from\_dict([missing\_values])

missing\_df

columns\_with\_missing\_values = []

for column in missing\_df:

    if missing\_df[column].values[0] != 0:

        columns\_with\_missing\_values.append(column)

missing\_df[columns\_with\_missing\_values]

print(missing\_values)

Количество удаленных записей и название колонок, содержащих пустые значения, представлен в таблице 2.

Таблица 2 – Удаленное количество строк за каждую колонку

|  |  |  |
| --- | --- | --- |
| № | totalTravelDistance | segmentsEquipmentDescription |
| 0 | 64719 | 19670 |

Количество строк после удаления пропущенных и аномальных нулевых значений – 965739.

## 1.4 Определение и удаление выбросов

df.count()

cleaned\_df = df.dropna()

df.count()

na\_rows\_df = df.filter(reduce(lambda x, y: x | y, (col(c).isNull() for c in df.columns)))

na\_rows\_df.count()

quartiles = cleaned\_df.stat.approxQuantile("totalTravelDistance", [0.25, 0.75], 0.0)

IQR = quartiles[1] - quartiles[0]

lower\_bound = quartiles[0] - 1.5 \* IQR

upper\_bound = quartiles[1] + 1.5 \* IQR

cleaned\_df = cleaned\_df.filter((col("totalTravelDistance") >= lower\_bound) & (col("totalTravelDistance") <= upper\_bound))

cleaned\_df.count()

Количество оставшихся после удаления данных составило 944708 строк/

## 1.5 Расчет статистических показателей признаков

1. Минимальное, среднее и максимальное значения:
   1. Минимальное значение — это наименьшее число в наборе данных.
   2. Среднее значение (или среднее арифметическое) — это сумма всех чисел в наборе, деленная на их количество.
   3. Максимальное значение — это наибольшее число в наборе данных.
2. Среднеквадратичное отклонение (или стандартное отклонение). Стандартное отклонение показывает, насколько в среднем значения в наборе отличаются от среднего значения.
3. Квартили — это значения, которые делят упорядоченный набор данных на три равные части. Они показывают распределение данных и включают в себя:

* Первый квартиль (Q1), который отделяет первые 25% данных;
* Второй квартиль (Q2), который также известен как медиана, отделяет первые 50% данных;
* Третий квартиль (Q3), который отделяет первые 75% данных;

1. Коэффициент асимметрии — это статистический показатель, который описывает степень асимметрии распределения данных относительно его среднего значения. Нормальное распределение имеет коэффициент асимметрии, равный 0, что указывает на симметричность данных. Положительное значение означает правостороннюю асимметрию, а отрицательное значение указывает на левостороннюю асимметрию.

statistics\_summary = cleaned\_df.describe()

statistics\_summary.toPandas()

1. elapsedDays варьируется от 0 до 1, что предполагает, что данные охватывают как минимум один день.
2. Результаты для колонки seatsRemaining показывают, что количество оставшихся мест колеблется от 1 до 10.
3. Результаты для колонок baseFare и totalFare показывают, что стоимость проезда варьируется на широкий диапазон. Значение асимметрии указывает на то, что распределение смещено вправо.
4. Результаты для колонки totalTravelDistance показывают, что общее расстояние путешествия колеблется от 97 до 3958, что указывает на значительное разнообразие в расстояниях между различными маршрутами. Значение асимметрии предельно мало, что указывает на небольшое правостороннее смещение в распределении общего расстояния путешествия.

## 1.6 Визуализацией распределения наиболее важных признаков

Для визуализации распределения наиболее важных признаков были использованы следующие графики:

* Гистограммы, пример которых иллюстрирует рисунок 1;
* Box plot, пример изображён на рисунке 2;
* Круговая диаграмма пример изображён на рисунке 3.

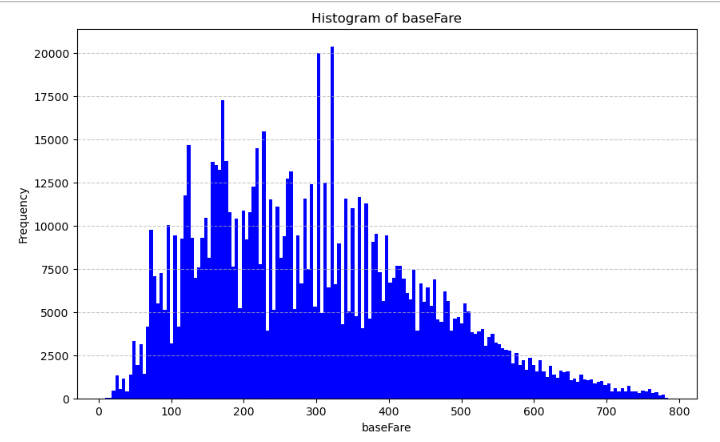


Рисунок 1 – Визуализация гистограммы распределения базовых цен

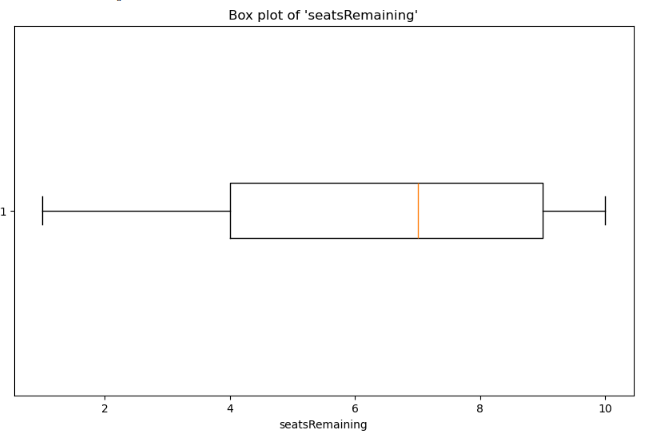


Рисунок 2 – Визуализация Box plot распределения количества оставшихся мест.

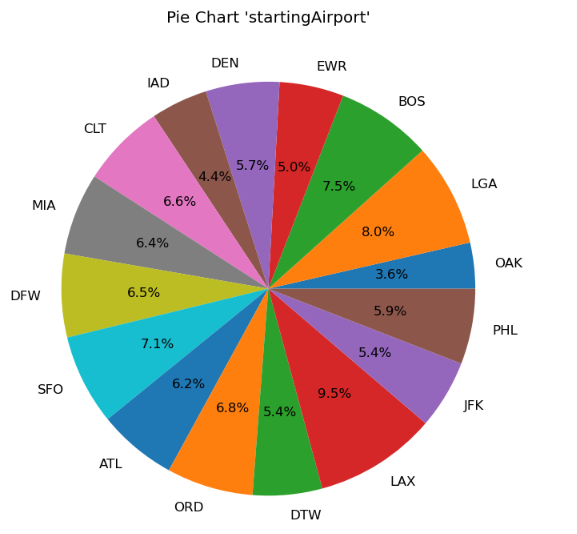


Рисунок 3 – Визуализация круговой диаграммы распределения индексов начальных аэропортов.

## 1.7 Корреляций между признаками

Чтобы выявить связи между признаками, можно построить матрицу корреляций. Матрица корреляций показана на рисунке 4.

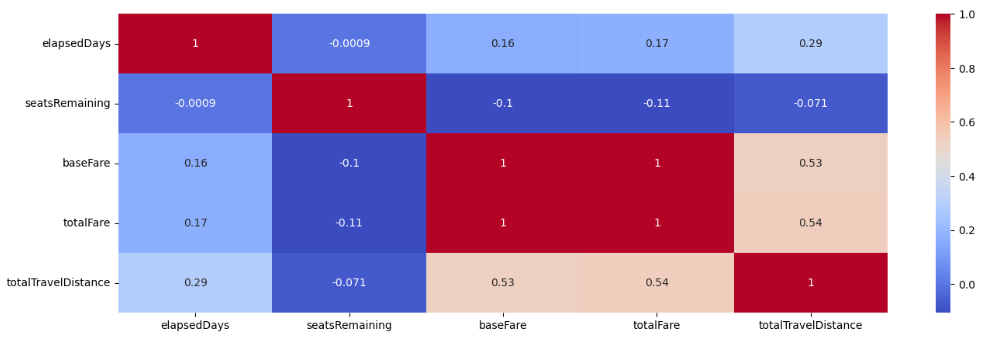


Рисунок 4 – Матрица корреляций