



МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт информационных технологий (ИИТ)
Кафедра прикладной математики (ПМ)

КУРСОВАЯ РАБОТА

по дисциплине

«Системы управления данными»

Тема курсовой работы: «Сбор, предобработка и анализ данных о продажах автомобилей в ОАЭ»

Студент группы ИНБО-06-21 Хозин Марат Дамирович


(подпись)

Руководитель
курсовой работы

ст. преподаватель, Юрченков И.А.


(подпись)

Работа представлена к защите «__» _____ 2024 г.

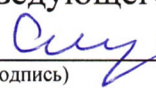
Допущен к защите «__» _____ 2024 г.

Москва 2024 г.



МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт информационных технологий (ИИТ)
Кафедра прикладной математики (ПМ)

Утверждаю
и.о. заведующего кафедрой ПМ

(подпись) Смоленцева Т.Е.
«22» мая 2024 г.

ЗАДАНИЕ
на выполнение курсовой работы
по дисциплине «Системы управления данными»

Студент Хозин Марат Дамирович

Группа ИНБО-06-21

Тема «Сбор, предобработка и анализ данных о продажах автомобилей в ОАЭ»


Исходные данные: набор данных о выставленных на продажу автомобилях, собранный с сайта объявлений dubizzle.com по состоянию на апрель 2024 года.

Перечень вопросов, подлежащих разработке, и обязательного графического материала:

1. Являются ли автомобили с автоматической коробкой передач более популярными, чем автомобили с механической коробкой передач в ОАЭ?
2. Есть ли корреляция между возрастом автомобиля и запрашиваемой ценой?
3. Как распределены объявления о продажах по городам?
4. Как пробег автомобиля влияет на его стоимость?
5. Какова средняя стоимость продаваемых автомобилей?

Срок представления к защите курсовой работы:

Задание на курсовую работу выдал


Подпись руководителя

до «22» мая 2024 г.

Юрченков И.А.

(ФИО руководителя)


«28» февраля 2024 г.

Хозин М.Д.

(ФИО обучающегося)

«28» февраля 2024 г.

Задание на курсовую работу получил


Подпись обучающегося

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	4
1 СОЗДАНИЕ КОНВЕЙЕРА ДЛЯ ПЕРЕДАЧИ ДАННЫХ	5
1.1 Структура и описание данных	5
1.2 Стек используемых технологий.....	6
2 АНАЛИЗ ПОЛУЧЕННЫХ ДАННЫХ	13
ЗАКЛЮЧЕНИЕ	23
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	24

ВВЕДЕНИЕ

В цифровую эпоху анализ данных является основой прогресса и инноваций. Сейчас предприятия имеют возможность получить ценную информацию, скрытую в больших наборах данных. Это способствует лучшему пониманию поведению клиентов, рыночных тенденций и эффективности компании.

Анализ данных позволяет специалистам в этой области принимать обоснованные решения в самых различных секторах экономики. Одним из таких секторов является автомобильный рынок.

Рынок автомобилей динамично меняется. Аналитики данных каждый день обрабатывают данные о продажах автомобилей для формирования бизнес-стратегий и стимулирования инноваций. Результаты, полученные из анализа информации о продажах автомобилей, важны и для производителей автомобилей, и для владельцев, продающих свои автомобили.

Анализ данных в этой сфере актуален всегда, и поэтому данная тема исследования также актуальна.

Цель данной курсовой работы — обработать данные, проанализировать исходную выборку и сделать выводы о текущих тенденциях на автомобильном рынке, распределении объявлений о продажах автомобилей по городам и влиянии характеристик автомобилей на их стоимость.

Предметом исследования являются данные о продажах автомобилей, собранные с сайта объявлений.

Задачи, выполняемые в данной курсовой работе:

- построить конвейер для предобработки и маршрутизации данных;
- определить ключевые вопросы для проведения анализа данных;
- провести корреляционный анализ;
- объяснить результаты, полученные в ходе анализа данных;
- визуализировать результаты.

1 СОЗДАНИЕ КОНВЕЙЕРА ДЛЯ ПЕРЕДАЧИ ДАННЫХ

1.1 Структура и описание данных

В курсовой работе исследуются данные о рынке автомобилей в Объединенных Арабских Эмиратах. В наборе данных 9953 строк наблюдений и 26 столбцов [1.1].

Столбцы содержат следующие переменные:

- price — запрашиваемая цена автомобиля в дирхамах ОАЭ (целочисленный тип);
- brand — производитель автомобиля (строковый тип);
- model — модель автомобиля (строковый тип);
- trim — комплектация автомобиля (строковый тип);
- kilometers — пробег автомобиля в километрах (целочисленный тип);
- year — год изготовления (целочисленный тип);
- vehicle age years — возраст автомобиля с даты его изготовления (целочисленный тип);
- regional_specs — региональные характеристики (строковый тип);
- doors — количество дверей в автомобиле (целочисленный тип);
- body_type — тип кузова (строковый тип);
- fuel_type — тип топлива, потребляемое автомобилем (строковый тип: «Petrol», «Diesel», «Electric», «Hybrid»);
- seating_capacity — количество мест в автомобиле (целочисленный тип);

- `transmission_type` — тип коробки передач (строковый тип: «Manual Transmission» или «Automatic Transmission»);
- `engine_capacity_cc` — объем двигателя в кубических сантиметрах (целочисленный тип);
- `horsepower` — диапазон мощности двигателя в лошадиных силах (строковый тип);
- `no_of_cylinders` — количество цилиндров в двигателе (целочисленный тип);
- `exterior_color` — цвет кузова (строковый тип);
- `interior_color` — цвет салона (строковый тип);
- `warranty` — указывает, распространяется ли гарантия на автомобиль (строковый тип: «Yes», «No», «Does not apply»);
- `country` — страна продажи автомобиля (строковый тип);
- `city` — город продажи автомобиля (строковый тип);
- `area_name` — регион продажи автомобиля (строковый тип);
- `location_name` — информация о месте продажи автомобиля (строковый тип);
- `latitude` — широта (вещественный тип);
- `longitude` — долгота (вещественный тип);
- `seller_type` — тип продавца (строковый тип: «Owner», «Dealership/Certified Pre-Owned», «Dealer»).

1.2 Стек используемых технологий

Платформы в сфере электронной коммерции активно собирают данные о продажах автомобилей, охватывая миллионы различных транзакций. Эти массивные объемы информации требуют обработки с использованием

специализированных конвейеров, чтобы быть готовыми к последующему анализу.

Проведем предобработку исследуемого набора данных о продажах автомобилей. Схематично конвейер представлен на Рисунке 1.1.

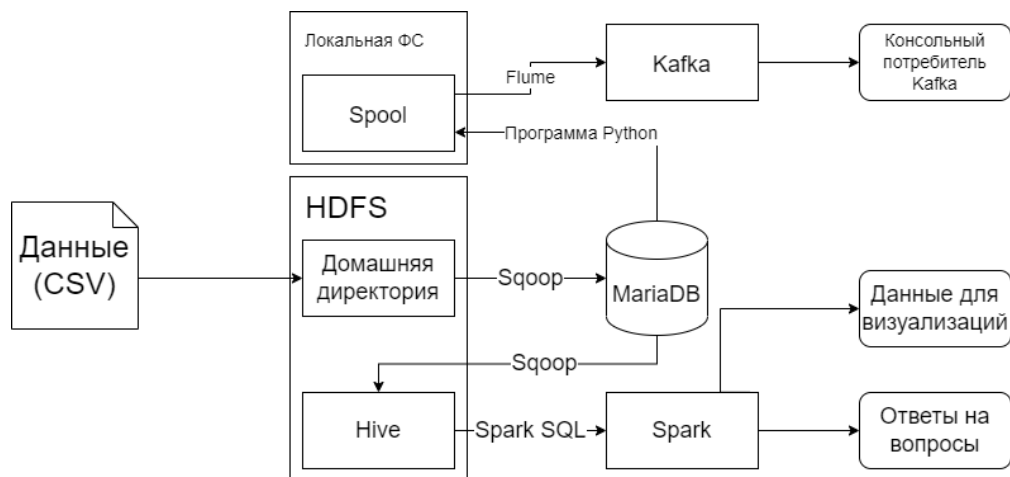


Рисунок 1.1 — Схема конвейера

В данной курсовой работе будем использовать инструменты, предназначенные для обработки и хранения больших данных.

Данные будут храниться и обрабатываться в распределенной файловой системе HDFS. HDFS обладает высокой отказоустойчивостью и масштабируемостью, что позволяет обрабатывать большие объемы данных. Благодаря репликации данных и распределению их по различным узлам кластера, HDFS обеспечивает надежное хранение информации и ускоряет процессы обработки данных [1.2].

Для хранения данных используются инструменты MariaDB и Apache Hive. MariaDB — это система управления базами данных, которая представляет собой ответвление MySQL и поддерживает высокую совместимость с ней. MariaDB — быстрая, масштабируемая и надежная система. Она поддерживает больше механизмов хранения, чем MySQL [1.3].

Apache Hive — это система для запроса и управления структурированными данными, построенная поверх Hadoop. Hive позволяет пользователю запрашивать данные с помощью языка HiveQL, похожего на SQL. Запросы HiveQL транслируются в Java-код заданий MapReduce.

Таблица в Hive — это каталог в HDFS. Все файлы в каталоге являются содержимым таблицы. Информация о схеме и о том, как разделяются строки и столбцы, хранится в хранилище метаданных Hive [1.4].

Для передачи данных из HDFS в MariaDB используется Apache Sqoop — приложение с интерфейсом командной строки для передачи данных между реляционными базами данных и кластерами Hadoop [1.5].

Для организации потоковой передачи данных будет использована Apache Kafka — гибрид распределённой базы данных и брокера сообщений с возможностью горизонтального масштабирования.

Kafka собирает у приложений данные, хранит их в своем распределённом хранилище, группируя по топикам, и отдаёт компонентам приложения по подписке. При этом сообщения хранятся на различных узлах-брокерах, что обеспечивает высокую доступность и отказоустойчивость [1.6].

Для управления потоком данных и их передачи в Kafka будем использовать Flume. С помощью Flume можно собирать, агрегировать и передавать большие объёмы данных с различных источников в хранилища данных [1.7].

Сначала исходные данные были загружены в домашнюю директорию HDFS. Структура данных представлена на Рисунке 1.2.

```
cars = spark.read.option("header", "True").option("delimiter", ",").csv("cars.csv")
cars.printSchema()
```

```
root
|-- price: string (nullable = true)
|-- brand: string (nullable = true)
|-- model: string (nullable = true)
|-- trim: string (nullable = true)
|-- kilometers: string (nullable = true)
|-- year: string (nullable = true)
|-- vehicle_age_years: string (nullable = true)
|-- regional_specs: string (nullable = true)
|-- doors: string (nullable = true)
|-- body_type: string (nullable = true)
|-- fuel_type: string (nullable = true)
|-- seating_capacity: string (nullable = true)
|-- transmission_type: string (nullable = true)
|-- engine_capacity_cc: string (nullable = true)
|-- horsepower: string (nullable = true)
|-- no_of_cylinders: string (nullable = true)
|-- exterior_color: string (nullable = true)
|-- interior_color: string (nullable = true)
|-- warranty: string (nullable = true)
|-- country: string (nullable = true)
|-- city: string (nullable = true)
|-- area_name: string (nullable = true)
|-- location_name: string (nullable = true)
|-- latitude: string (nullable = true)
|-- longitude: string (nullable = true)
|-- seller_type: string (nullable = true)
```

Рисунок 1.2 — Данные после загрузки в HDFS

Затем в MariaDB была создана таблица для хранения данных. Команда для создания таблицы представлена на Рисунке 1.3.

```
MariaDB [labs]> create table cars(price char(100), brand char(100), model char(100), trim char(100), kilometers char(100), year char(100), vehicle_age_years c
char(100), regional_specs char(100), doors char(100), body_type char(100), fuel_type char(100), seating_capacity char(100), transmission_type char(100), engine
_capacity_cc char(100), horsepower char(100), no_of_cylinders char(100), exterior_color char(100), interior_color char(100), warranty char(100), country char(
100), city char(100), area_name char(100), location_name char(100), latitude char(100), longitude char(100), seller_type char(100));
Query OK, 0 rows affected (0.02 sec)
```

Рисунок 1.3 — Создание таблицы в MariaDB

Далее данные были экспортированы из HDFS в MariaDB с помощью Apache Sqoop [1.8].

Команда для экспорта представлена на Рисунке 1.4.

```
[student@localhost ~]$ sqoop export --connect jdbc:mysql://localhost/labs --username student --password student --e
xport-dir cars.csv --table cars --fields-terminated-by ','
```

Рисунок 1.4 — Экспорт данных в MariaDB

Проверим, что данные успешно экспортированы в MariaDB. Результат запроса данных из таблицы представлен на Рисунке 1.5.

```
MariaDB [labs]> select price, brand, model from cars limit 10;
+-----+-----+-----+
| price | brand | model |
+-----+-----+-----+
| 253500 | Bentley | Continental |
| 25500 | Chevrolet | Tahoe |
| 34000 | Kia | Optima |
| 270000 | Toyota | Land Cruiser |
| 67900 | Ford | Explorer |
| 50900 | Infiniti | QX70 |
| 139000 | BMW | X3 |
| 66000 | Smart | Other |
| 30500 | Mitsubishi | Pajero |
| 32000 | Volvo | XC60 |
+-----+-----+-----+
10 rows in set (0.00 sec)
```

Рисунок 1.5 — Запрос к таблице на получение данных

Изначально таблица создавалась с типом данных char для всех столбцов, но в ней присутствуют не только строки, но и числа. Поэтому тип целочисленных данных изменен на int, а тип вещественных данных изменен на double. После чего добавлен столбец id, являющийся первичным ключом. Структура таблицы после проведенных изменений представлена на Рисунке 1.6.

Field	Type	Null	Key	Default	Extra
price	int(11)	YES		NULL	
brand	char(100)	YES		NULL	
model	char(100)	YES		NULL	
trim	char(100)	YES		NULL	
kilometers	int(11)	YES		NULL	
year	int(11)	YES		NULL	
vehicle_age_years	int(11)	YES		NULL	
regional_specs	char(100)	YES		NULL	
doors	int(11)	YES		NULL	
body_type	char(100)	YES		NULL	
fuel_type	char(100)	YES		NULL	
seating_capacity	int(11)	YES		NULL	
transmission_type	char(100)	YES		NULL	
engine_capacity_cc	char(100)	YES		NULL	
horsepower	char(100)	YES		NULL	
no_of_cylinders	int(11)	YES		NULL	
exterior_color	char(100)	YES		NULL	
interior_color	char(100)	YES		NULL	
warranty	char(100)	YES		NULL	
country	char(100)	YES		NULL	
city	char(100)	YES		NULL	
area_name	char(100)	YES		NULL	
location_name	char(100)	YES		NULL	
latitude	double	YES		NULL	
longitude	double	YES		NULL	
seller_type	char(100)	YES		NULL	
id	int(11)	NO	PRI	NULL	auto_increment

27 rows in set (0.01 sec)

Рисунок 1.6 — Структура таблицы в базе данных

Затем создан топик для создания потока данных с помощью Apache Kafka (Рисунок 1.7).

```
[student@localhost ~]$ kafka-topics --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions
1 --topic topic_hozin
WARNING: Due to limitations in metric names, topics with a period ('.') or underscore ('_') could collide. To avoid
issues it is best to use either, but not both.
Created topic topic_hozin.
[student@localhost ~]$ kafka-topics --list --bootstrap-server localhost:9092
topic_hozin
```

Рисунок 1.7 — Создание топика Kafka

Для передачи данных в Kafka написан конфигурационный файл (Рисунок 1.8).

```
agent1.sources = src1
agent1.channels = ch1 ch2
agent1.sinks = sink1 sink2

agent1.sources.src1.type = spooldir
agent1.sources.src1.spoolDir = /home/student/spool

agent1.channels.ch1.type = memory
agent1.channels.ch1.capacity = 10000
agent1.channels.ch1.transactionCapacity = 100

agent1.channels.ch2.type = memory
agent1.channels.ch2.capacity = 10000
agent1.channels.ch2.transactionCapacity = 100

agent1.sinks.sink1.type = org.apache.flume.sink.kafka.KafkaSink
agent1.sinks.sink1.kafka.bootstrap.servers = localhost:9092
agent1.sinks.sink1.kafka.topic = topic_hozin
agent1.sinks.sink1.kafka.flumeBatchSize = 5
agent1.sinks.sink1.channel = ch1

agent1.sinks.sink2.type = logger
agent1.sinks.sink2.channel = ch2

agent1.sources.src1.channels = ch1 ch2
```

Рисунок 1.8 — Конфигурационный файл агента Flume

Для создания потока данных был написан скрипт, который непрерывно извлекает данные из базы данных MariaDB и создает файл из полученных строк в формате CSV в папке Spool. Каждые 10 секунд программа запрашивает 5 % всех строк в таблице с данными. Код программы представлен на Рисунке 1.9.

```
import time
import pymysql
from pyspark.sql import SparkSession
from pyspark.sql import Row

db_config = {
    'host': 'localhost',
    'user': 'student',
    'password': 'student',
    'database': 'labs'
}

spark = SparkSession.builder.appName('MariaDBToSpark').getOrCreate()

def fetch_data():
    try:
        connection = pymysql.connect(**db_config)
        cursor = connection.cursor()
        query = """
        SELECT * FROM cars
        WHERE RAND() < 0.05
        """
        cursor.execute(query)
        rows = cursor.fetchall()
        cursor.close()
        connection.close()
        return rows
    except Exception as e:
        print(f"Error: {e}")
        return []

def save_to_csv(data):
    if data:
        rows = [Row(*row) for row in data]
        df = spark.createDataFrame(rows)
        output_path = '/home/student/spool/cars_data.csv'
        df.write.option('header', 'true').mode('append').csv('file:/home/student/spool')
        print("Data written to CSV successfully.")
    else:
        print("No data to write.")

def main():
    while True:
        data = fetch_data()
        save_to_csv(data)
        time.sleep(10)

if __name__ == '__main__':
    main()
```

Рисунок 1.9 — Скрипт для создания файла в папке Spool

Во время работы скрипта папку Spool просматривает агент Flume [1.8]. После запуска агента Flume и написанного скрипта данные начали поступать в обработчик (Рисунок 1.10).

```
2024-05-27 19:00:36,521 [pool-3-thread-1] INFO org.apache.flume.client.avro.ReliableSpoolingFileEventReader - Preparing to move file /home/student/spool/part-00001-71285565-f69e-45c4-82cc-ca7c34b0ef2a-c000.csv to /home/student/spool/part-00001-71285565-f69e-45c4-82cc-ca7c34b0ef2a-c000.csv.COMPLETED
```

Рисунок 1.8 — Получение файлов

Логирование полученных данных в консоль показано на Рисунке 1.11.

```
2024-05-27 19:00:36,506 [SinkRunner-PollingRunner-DefaultSinkProcessor] INFO org.apache.flume.sink.LoggerSink - Event: { headers:{} body: 37 32 30 30 30 2C 4D 47 2C 52 58 35 2C 4C 75 78 72000,MG,RX5,Lux }
2024-05-27 19:00:36,506 [SinkRunner-PollingRunner-DefaultSinkProcessor] INFO org.apache.flume.sink.LoggerSink - Event: { headers:{} body: 38 34 39 30 30 2C 54 6F 79 6F 74 61 2C 46 6F 72 84900,Toyota,For }
2024-05-27 19:00:36,523 [SinkRunner-PollingRunner-DefaultSinkProcessor] INFO org.apache.flume.sink.LoggerSink - Event: { headers:{} body: 39 35 30 30 30 2C 48 79 75 6E 64 61 69 2C 54 75 95000,Hyundai,Tu }
2024-05-27 19:00:36,523 [SinkRunner-PollingRunner-DefaultSinkProcessor] INFO org.apache.flume.sink.LoggerSink - Event: { headers:{} body: 32 33 39 39 30 30 2C 43 68 65 76 72 6F 6C 65 74 239900,Chevrolet }
2024-05-27 19:00:36,523 [SinkRunner-PollingRunner-DefaultSinkProcessor] INFO org.apache.flume.sink.LoggerSink - Event: { headers:{} body: 37 38 30 30 30 2C 46 6F 72 64 2C 45 78 70 6C 6F 78000,Ford,Explo }
2024-05-27 19:00:36,523 [SinkRunner-PollingRunner-DefaultSinkProcessor] INFO org.apache.flume.sink.LoggerSink - Event: { headers:{} body: 31 39 36 39 30 30 2C 4A 61 67 75 61 72 2C 46 2D 196000,Jaguar,F- }
2024-05-27 19:00:36,523 [SinkRunner-PollingRunner-DefaultSinkProcessor] INFO org.apache.flume.sink.LoggerSink - Event: { headers:{} body: 31 32 30 30 30 2C 42 4D 57 2C 58 35 2C 78 44 120000,BMW,X5,x0 }
```

Рисунок 1.9 — Логирование полученных данных в консоль

Все данные, попавшие в Kafka, выводились в отдельном окне терминала с помощью потребителя Kafka (Рисунок 1.12).

```
[student@localhost ~]$ kafka-console-consumer --bootstrap-server localhost:9092 --topic topic_hozin --from-beginning
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28
66000,Smart,Other,Other,74000,2016,8,American Specs,2,Coupe,Petrol,0,Automatic Transmission,"",160 - 199 HP,3,Green,Green,No,"Abu Dhabi","UAE\
",UAE,Abu Dhabi,"",24.4538352,54.3774814,8
77900,Nissan,Altima,SV,47490,2021,3,GCC Specs,4,Sedan,Petrol,5,Automatic Transmission,2500 - 2999 cc,100 - 199 HP,4,Black,Beige,No,"Nad Al Ham
ar",Dubai,"UAE\","UAE,Dubai,Nad Al Hamar,"",25.2364297,17
590000,Mercedes-Benz,V-Class,V 250,0,2024,0,GCC Specs,5,Van,Petrol,7,Automatic Transmission,2000 - 2499 cc,200 - 299 HP,4,Black,Beige,Yes,"Al
Quoz Industrial Area 3",Al Quoz Industrial Area,Al Quoz,Dubai,"UAE\","UAE,Dubai,Al Quoz,24
47000,Volkswagen,Golf,GTI,141230,2016,8,GCC Specs,5,Hatchback,Petrol,5,Automatic Transmission,2000 - 2499 cc,200 - 299 HP,4,White,Black,No,"Al
Qusais",Dubai,"UAE\","UAE,Dubai,Al Qusais,"",25.269769,38
```

Рисунок 1.10 — Запуск потребителя Kafka

Реализуем второй путь данных из MariaDB. Импортируем таблицу из MariaDB в Apache Hive [1.8].

Команда для импорта представлена на Рисунке 1.13.

```
[student@localhost ~]$ sqoop import \
> --connect jdbc:mysql://localhost/labs \
> --username student --password student \
> --table cars \
> --hive-import --hive-table cars
```

Рисунок 1.11 — Импорт данных в Apache Hive

Результат импорта таблицы в Hive показан на Рисунке 1.14.

```
0: jdbc:hive2://> show tables;
OK
+-----+
| tab_name |
+-----+
| cars      |
+-----+
1 row selected (1.465 seconds)
```

Рисунок 1.12 — Результат импорта

Таким образом, мы построили конвейер для данных и с помощью него обработали их и подготовили к последующему анализу.

В ходе реализации конвейера были использованы такие технологии, как HDFS, MariaDB, Sqoop, Hive, Kafka, Flume. Каждая из этих технологий имеет важное значение в нашем конвейере данных. Далее проведем анализ данных, используя Apache Spark.

2 АНАЛИЗ ПОЛУЧЕННЫХ ДАННЫХ

Полученные данные из Hive загружены Apache Spark. С помощью DataFrame API создан DataFrame для дальнейшего анализа и визуализации. Фрагмент DataFrame представлен на Рисунке 2.1.

```
| price|      brand|      model|      trim|kilometers|year|vehicle_age_years|regional_specs|doors| bo  
dy_type|fuel_type|seating_capacity| transmission_type|engine_capacity_cc| horsepower|no_of_cylinders|exterior_c  
olor|interior_color| warranty|country| city|      area_name| location_name| latitude| lon  
gitude|      seller_type| id|  
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----  
[295000| Land Rover| Range Rover|SV Autobiography| 54971|2018| 6|European Specs| 5| Blu  
SUV| Petrol| 4|Automatic Transmi...| 4000+ cc|500 - 599 HP| 25.0778021| 55.13518094856  
e| White| No| UAE| Dubai|Jumeirah Beach Re...|  
796| Owner| 1|  
[ 85000|Mercedes-Benz|C-Class Coupe| Other| 187000|2013| 11| GCC Specs| 2| R  
Coupe| Petrol| 4|Automatic Transmi...| 4000+ cc|400 - 499 HP|  
ed| Black| No| UAE|Abu Dhabi| Al Reem Island|Shams Abu Dhabi|24.49175605| 54.4084643139  
0834| Owner| 2|  
[129900| Dodge| Ram| 1500 SLT Crew| 47566|2021| 3| GCC Specs| 4|Pick U  
p Truck| Petrol| 5|Automatic Transmi...| 4000+ cc|300 - 399 HP|  
hite| Unknown| Yes| UAE| Dubai| Deira| Al Khabaisi| 25.266173| 55.3  
377718|Dealership/Certif...| 3|  
[ 45000|Mercedes-Benz| A-Class| A 250| 230000|2014| 10| GCC Specs| 4| Ha  
tchback| Petrol| 5|Automatic Transmi...| 2500 - 2999 cc|200 - 299 HP|  
lack| Black|Does not apply| UAE|Abu Dhabi| Al Reef| Al Reef Villas| 24.4589837| 54.671088412  
408736| Owner| 4|  
[ 16000| Volvo| XC60| T6 Comfort| 182000|2009| 15| GCC Specs| 5| Blac  
SUV| Petrol| 5|Automatic Transmi...| 3000 - 3499 cc|300 - 399 HP|  
k| Beige|Does not apply| UAE| Dubai| Arabian Ranches| 25.04875135| 55.26728851815  
184| Owner| 5|  
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----  
only showing top 5 rows
```

Рисунок 2.1 — Фрагмент данных в Spark

В исследуемом наборе данных каждая строка представляет собой подробную информацию об автомобиле.

Приступим к анализу исходных данных. Сначала узнаем, являются ли автомобили с автоматической коробкой передач более популярными в ОАЭ, чем автомобили с механической коробкой передач. Для этого воспользуемся столбцом `transmission_type` (Листинг 2.1).

Листинг 2.1 — Запрос количества автомобилей, сгруппированных по коробке передач

```
count_transmission = spark.sql(  
    """ SELECT transmission_type, COUNT(*) AS count  
    FROM cars  
    GROUP BY transmission_type """  
count_transmission.show()
```

Результат запроса представлен на Рисунке 2.2.

```

+-----+
| transmission_type|count|
+-----+
|Automatic Transmi...| 9697|
| Manual Transmission|  256|
+-----+

```

Рисунок 2.2 — Количество автомобилей по типу коробки передач

Значение Automatic Transmission соответствует автоматической коробке передач, а Manual Transmission — механической коробке передач.

В результате мы получили, что примерно 97 % всех автомобилей в выборке поставляются с автоматической коробкой передач, а около 3 % автомобилей поставляются с механической коробкой передач.

Популярность автомобилей первого типа может быть связано с их большей востребованностью среди покупателей. Автоматическая коробка передач обеспечивает более комфортное управление автомобилем в условиях пробок, так как водителю не нужно постоянно переключать передачи. Так что полученный результат может быть связан с предпочтениями клиентов или особенностями городского движения в ОАЭ [2.1].

Посмотрим есть ли корреляция между переменными в данных (Рисунок 2.3).

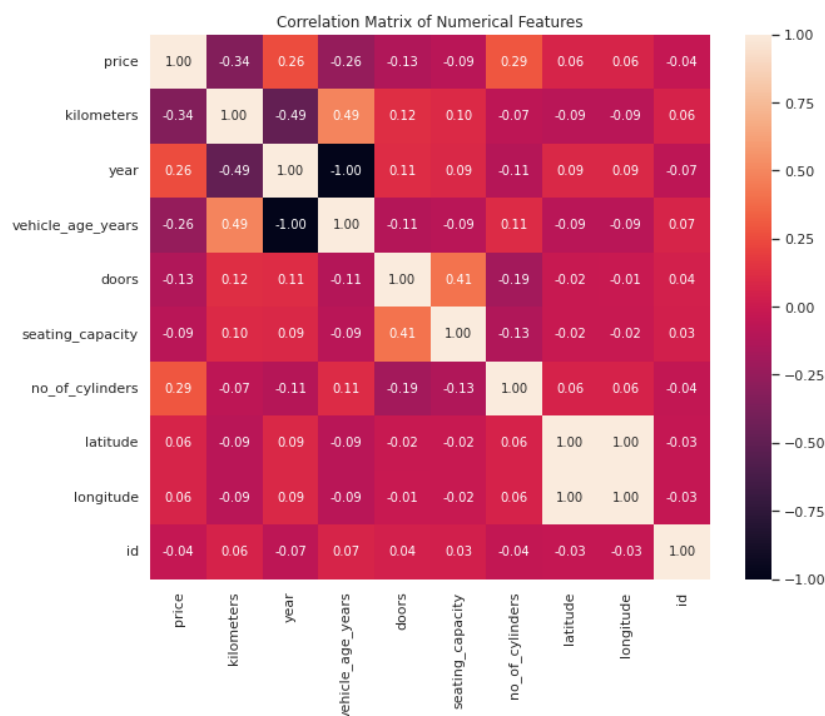


Рисунок 2.3 — Корреляционная матрица

Исходя из значений в корреляционной матрице, можно сделать следующие выводы:

1. Положительная корреляция между пробегом и возрастом автомобиля. Это указывает на то, что старые автомобили, как правило, имеют больший пробег. Это ожидаемо, поскольку чем дольше автомобиль находится в эксплуатации, тем большее расстояние он, скорее всего, преодолеет.
2. Отрицательная корреляция между пробегом и годом изготовления автомобиля. Отрицательная корреляция здесь означает, что более новые автомобили, то есть с более поздним годом выпуска, как правило, имеют меньший пробег.
3. Отрицательная корреляция между пробегом автомобиля и запрашиваемой ценой. Большой пробег может быть признаком износа, который может снизить стоимость автомобиля. Покупатели часто готовы платить больше за подержанный автомобиль с меньшим пробегом, ожидая, что у него будет более длительный срок службы и потенциально меньше механических проблем.
4. Отрицательная корреляция между возрастом автомобиля и запрашиваемой ценой. Однако корреляция не столь высокая, как ожидалось. Это связано с наличием в выборке старых автомобилей, считающихся классикой. Такие автомобили со временем могут дорожать из-за их востребованности. Поэтому в дальнейшем исключим из рассмотрения автомобили, произведенные 20 лет назад и раньше [2.1].

Построим линейный график, отражающий зависимость между возрастом автомобиля и запрашиваемой ценой. (Рисунок 2.4).

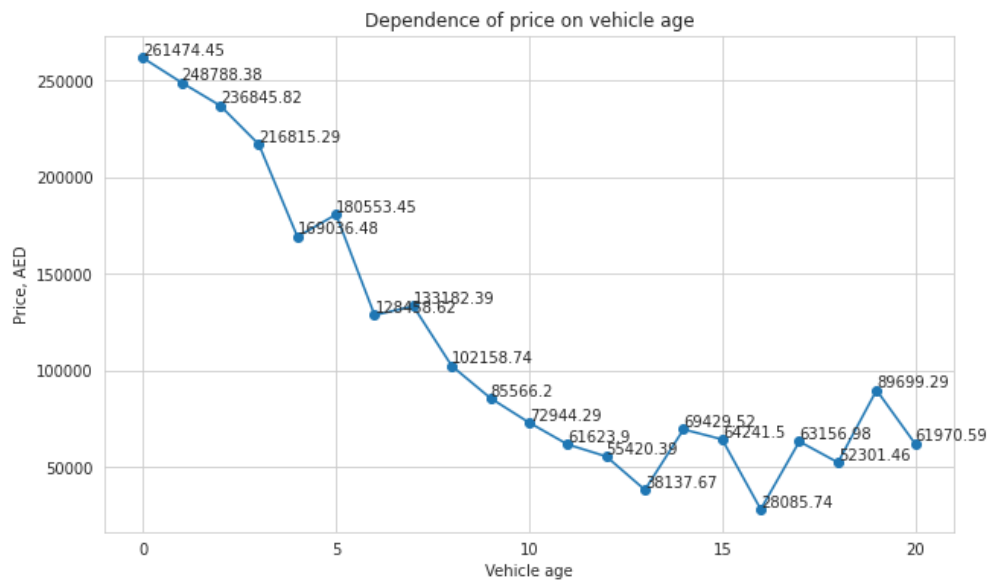


Рисунок 2.4 — Зависимость между возрастом автомобиля и запрашиваемой ценой

В рассматриваемых временных рамках средняя цена постепенно снижается с увеличением возраста. Однако далее средняя цена растет, и это объясняется, во-первых, низким числом старых автомобилей в выборке и, во-вторых, появлением в выборке классических автомобилей.

Теперь найдем среднюю, минимальную и максимальную стоимости продаваемых автомобилей (Листинг 2.2).

Листинг 2.2 — Запрос минимального, среднего и максимального значения цены в выборке

```
SELECT min(price), avg(price), max(price)
FROM cars
```

В результате выполнения запроса мы получили среднюю стоимость — 162009.23 дирхам. Самый дешевый автомобиль из выборки продается по цене 1000 дирхам, а самый дорогой — по цене 999900 дирхам (Рисунок 2.5).

min(price)	avg(price)	max(price)
1000	162009.2363106601	999900

Рисунок 2.5 — Минимальная, средняя и максимальная цена в выборке

Чтобы получить представление о распределении цен на автомобили в выборке, была создана гистограмма, отражающая это распределение [2.2].

Гистограмма представлена на Рисунке 2.6.

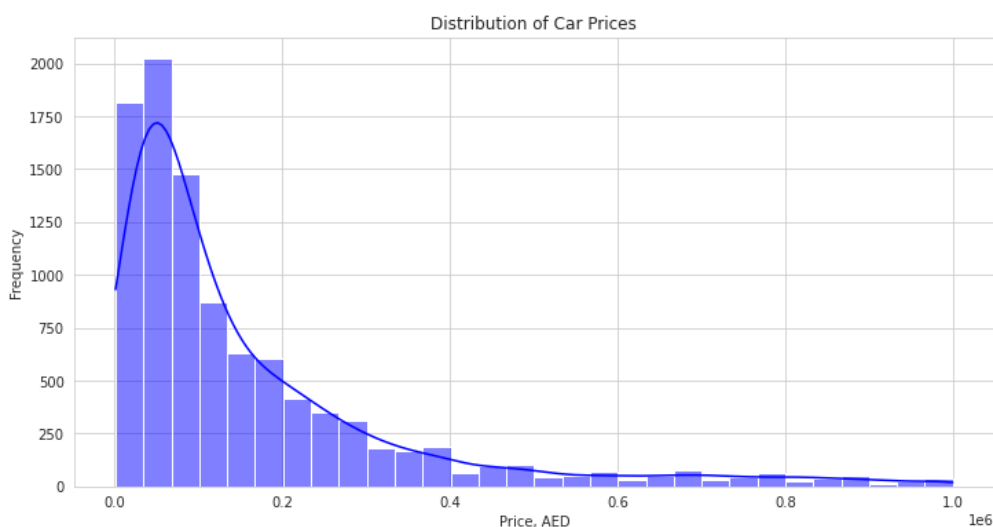


Рисунок 2.6 — Распределение цен на автомобили

Полученная ранее средняя цена автомобиля оказалась низкой по сравнению с представленными в выборке ценами. Около 5300 автомобилей, то есть более 50 % всех представленных в выборке автомобилей, продаются по цене менее ста тысяч дирхам. Примерно 7,5 тысяч автомобилей продаются по цене менее двухсот тысяч дирхам. Автомобили с более высокими ценами встречаются реже.

Найдем средние цены среди автомобилей самых популярных брендов в ОАЭ. Мы не рассматриваем все бренды, чтобы диаграммы были наглядными. Сначала отсортируем бренды по числу их встречаемости в наборе данных (Рисунок 2.7).

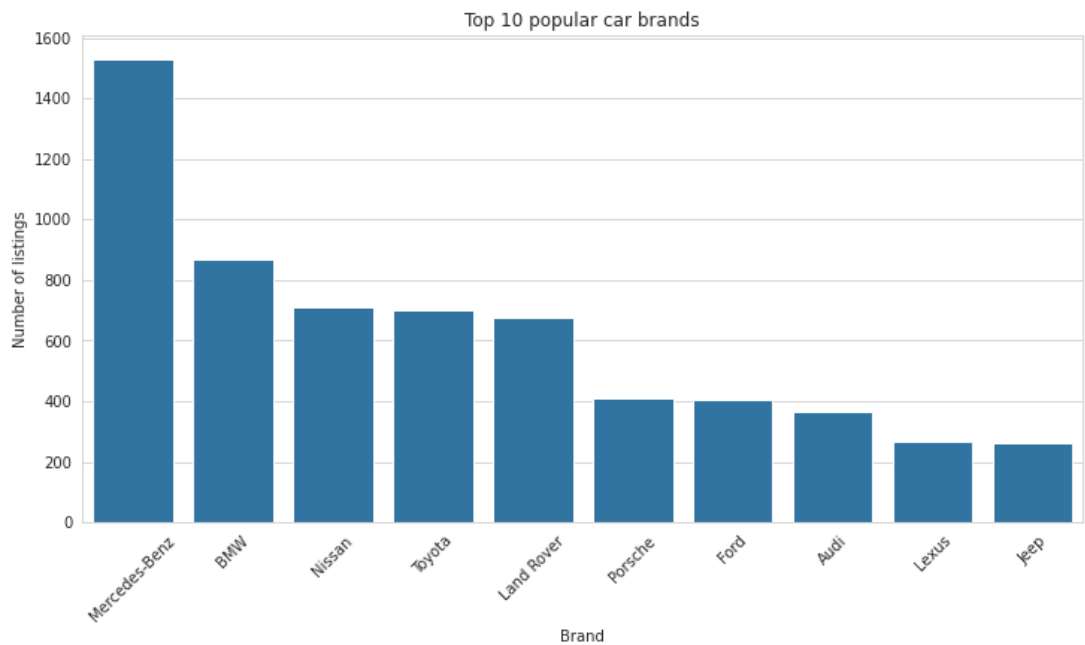


Рисунок 2.7 — Самые популярные бренды в выборке

Далее для всех десяти брендов найдем средние цены автомобилей (Рисунок 2.8)

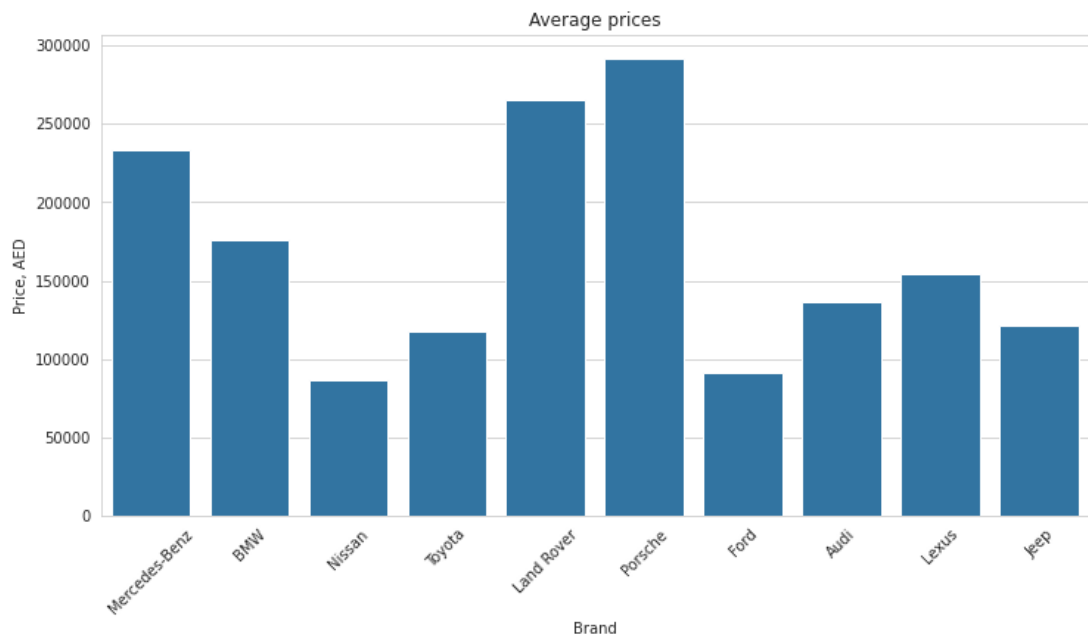


Рисунок 2.8 — Средние цены среди автомобилей самых популярных брендов

Таким образом, среди самых популярных брендов средние цены варьируются в пределах от 50 тысяч до 300 тысяч дирхам. Чаще всего в выборке встречаются автомобили Mercedes-Benz, их средняя цена в пределах бренда находится в диапазоне от 200 тысяч до 250 тысяч дирхам.

Далее узнаем, как влияет тип продавца на стоимость машин. Для этого воспользуемся box plot (Рисунок 2.9).

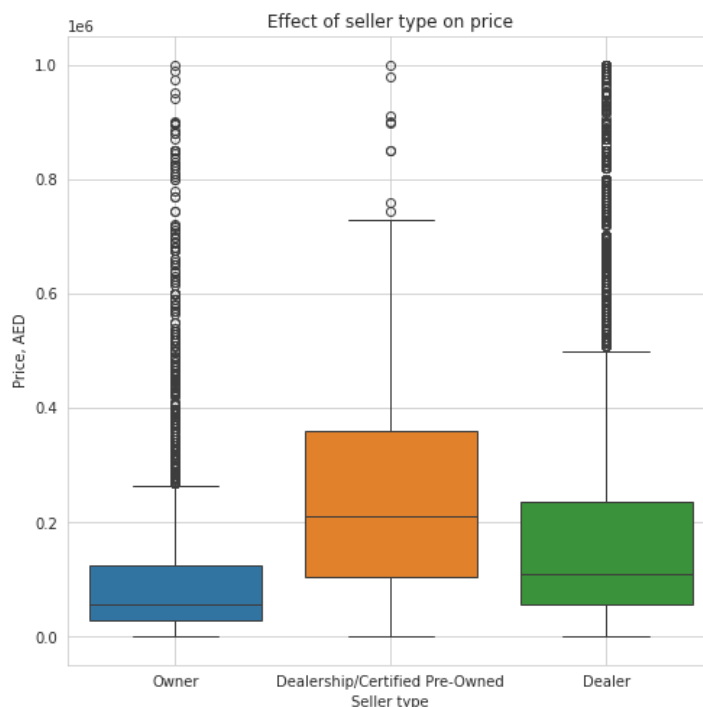


Рисунок 2.9 — Влияние типа продавца на стоимость автомобиля

Существуют три типа продавца в рассматриваемом наборе данных:

1. Владелец. Медиана цены находится на уровне 50 тысяч дирхам. Это самое низкое значение по сравнению с другими типами продавцов.
2. Сертифицированный продавец подержанных автомобилей. Медианное значение их стоимости — около 100 тысяч дирхам.
3. Дилер. Медиана цены находится на уровне 200 тысяч дирхам [2.3].

Таким образом, выгоднее всего купить автомобиль у его текущего владельца. У дилера стоимость выше ввиду надежности продавца.

Теперь посмотрим, как гарантия влияет на стоимость автомобиля. Соответствующий box plot представлен на Рисунке 2.10.

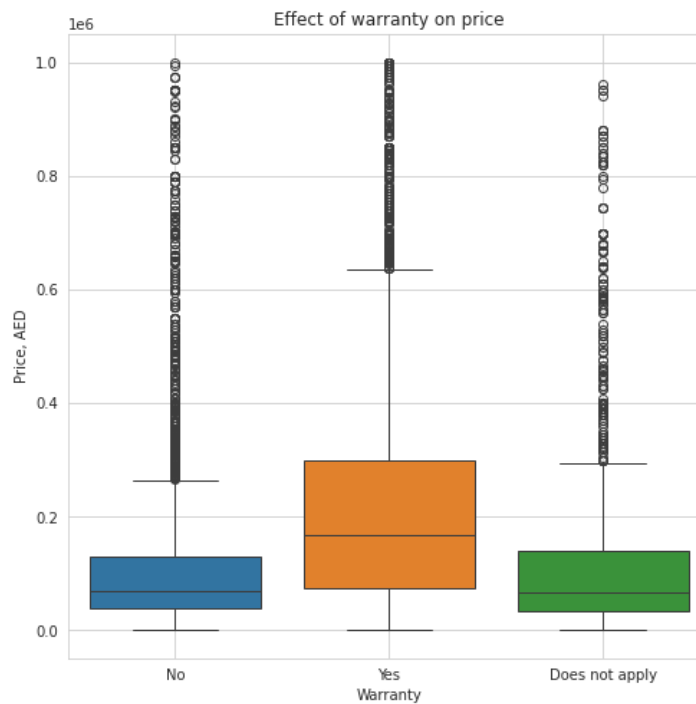


Рисунок 2.10 — Влияние гарантии на стоимость автомобиля

Самые дешевые автомобили — те, на которые не предоставляется гарантия. Они обозначены как «No» и «Does not apply». Их медианная цена равна примерно 70 тысячам дирхам. У автомобилей с гарантией медиана цены равна приблизительно 145 тысячам дирхам [2.3].

Визуализируем местоположения продающихся автомобилей на карте. Для этого воспользуемся Hue, так он обладает удобным интерфейсом и необходимым нам функционалом.

Поскольку в наборе данных есть записи, в которых значения latitude и longitude пустые, необходимо предварительно обработать данные. С помощью HiveQL выберем нужные нам записи (Листинг 2.3)

Листинг 2.3 — Выбор строк, в которых значение координаты не равно (0, 0)

```
select latitude, longitude from cars
where not (latitude == 0 and longitude == 0)
```

Карта с местоположениями объявлений показана на Рисунке 2.11.



Рисунок 2.11 — Местоположения объявлений

Таким образом, набор данные включает себя только объявления из ОАЭ.

Получим более точную информацию о количестве объявлений в каждом городе (Листинг 2.4).

Листинг 2.4 — Запрос количества автомобилей в каждом городе

```
SELECT city, COUNT(*) AS count
FROM cars
GROUP BY city
ORDER by count DESC
```

Результат обработки запроса представлен на Рисунке 2.12.

city	count
Dubai	8335
Abu Dhabi	737
Sharjah	698
Ajman	115
Al Ain	21
Umm Al Qawain	19
Ras Al Khaimah	17
Fujeirah	11

Рисунок 2.12 — Количество автомобилей в каждом городе

Более наглядную информацию мы можем получить из столбчатой диаграммы (Рисунок 2.13).

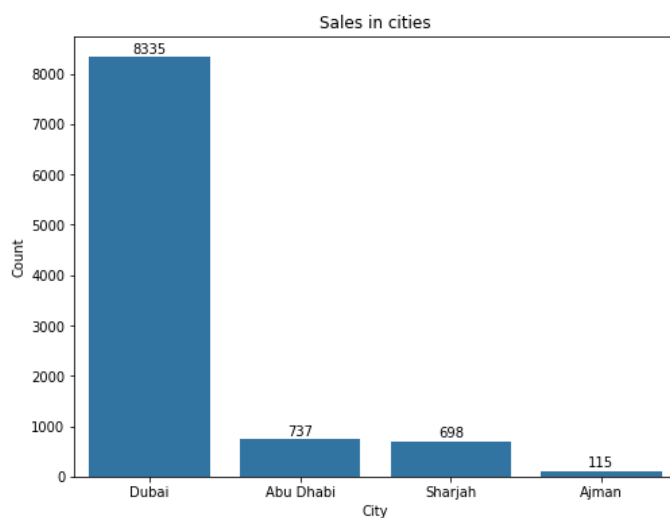


Рисунок 2.13 — Распределение объявлений по городам

Таким образом, наибольшее количество объявлений (около 83,7 %) приходится на Дубай.

В результате, проведен анализ полученных данных. Мы выяснили, что автоматическая коробка передач используется больше водителями в ОАЭ, чем механическая, согласно исходному набору данных. Затем проведен корреляционный анализ и объяснена корреляция различных параметров автомобиля. Наконец, были проанализированы местоположения исследуемых объявлений.

Анализ данных проведен в среде Apache Spark. Spark обеспечивает масштабируемую, быструю и отказоустойчивую потоковую обработку данных в реальном времени. Он обрабатывает данные в оперативной памяти, поэтому работает очень быстро. Этими преимуществами обоснован выбор Apache Spark для анализа данных.

ЗАКЛЮЧЕНИЕ

Роль аналитиков данных заключается в сборе, обработке, анализе и интерпретации больших объемов данных с целью принятия обоснованных стратегических решений.

В целом, понимание и правильное использование больших данных становится неотъемлемой частью конкурентоспособности компаний, в частности, в автомобильном секторе.

Данные о продажах автомобилей, собранные, например, с сайтов объявлений, помогают продавцам предложить актуальную цену за автомобиль, а компаниям эти данные помогают в оптимизации цен. Динамические модели ценообразования могут быть разработаны с использованием анализа данных для корректировки цен на автомобили в режиме реального времени на основе различных факторов, таких как спрос, конкуренция и рыночные условия.

Таким образом можно убедиться в том, что тема данной курсовой работы является актуальной не только в пределах одной страны, но и для всего мира.

Цель данной курсовой работы — обработать данные, проанализировать исходную выборку и сделать выводы о текущих тенденциях на автомобильном рынке, распределении объявлений о продажах автомобилей по городам и влиянии характеристик автомобилей на их стоимость — достигнута.

Задачи, выполненные в ходе данной курсовой работы:

- построен конвейер для предобработки и маршрутизации данных;
- определены ключевые вопросы для проведения анализа данных;
- проведен корреляционный анализ;
- объяснены результаты, полученные в ходе анализа данных;
- результаты визуализированы.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

СОЗДАНИЕ КОНВЕЙЕРА ДЛЯ ПЕРЕДАЧИ ДАННЫХ

- 1.1. Kaggle. UAE Auto Market Sales Data [Электронный ресурс]. — Режим доступа: <https://www.kaggle.com/datasets/azharsaleem/uae-auto-market-sales-data-for-advanced-analytics/data>.
- 1.2. Apache Hadoop — HDFS Architecture [Электронный ресурс]. — Режим доступа: <https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html>.
- 1.3. MariaDB Tutorial [Электронный ресурс]. — Режим доступа: <https://www.mariadbtutorial.com/getting-started/what-is-mariadb/>.
- 1.4. Школа больших данных. Apache Hive [Электронный ресурс]. — Режим доступа: <https://bigdataschool.ru/wiki/hive>.
- 1.5. Sqoop User Guide [Электронный ресурс]. — Режим доступа: <https://docs.cloudera.com/sqoop/1.4.7.7.1.6.0/user-guide/index.html>.
- 1.6. Yandex Cloud. Apache Kafka: что это и где применяется [Электронный ресурс]. — Режим доступа: <https://yandex.cloud/ru/blog/posts/2021/02/managed-kafka-overview>.
- 1.7. Habr. Flume — управляем потоками данных [Электронный ресурс]. — Режим доступа: <https://habr.com/ru/companies/dca/articles/280386/>.
- 1.8. Samsung Innovation Campus. Курс: Большие данные [Электронный ресурс]. — Режим доступа: <https://innovationcampus.ru>.

АНАЛИЗ ПОЛУЧЕННЫХ ДАННЫХ

- 2.1. How much is my car worth? [Электронный ресурс]. — Режим доступа: <https://autotrader.co.nz/advice/how-much-is-my-car-worth>.

- 2.2. Глинский В. В., Ионин В. Г. Статистический анализ. — М.: Инфра-М, 2002. — 241 с. — (Высшее образование). — 5000 экз. — ISBN 5-16-001293-1.
- 2.3. Loginom Wiki. Box-plot [Электронный ресурс]. — Режим доступа: <https://wiki.loginom.ru/articles/box-plot.html>.