Green Taxi Fare Amount Prediction

Anton Kudryavtsev, Anatoly Soldatov, Sofia Tkachenko, Anastasiia Shvets

May 09, 2025

Contents

1. Introduction

The problem of predicting how much the taxi trip will cost is important, and has to consider several factors, such as availability of drivers nearby, distance covered and how high the demand for taxi is in both pickup and drop-off points.

The project objective is to develop a machine learning model based on the NYC Green taxi dataset to predict taxi fares based on several factors: date, time, number of passengers, etc.

As part of the project, we will have to handle missing values, try to capture hidden trends and fare regulations.

The result of our project can be beneficial to taxi companies by providing an understanding on what factors influence the cost of the trip, so that they can keep the customers happy by providing understandable and explainable prices.

1.1. Business objectives

- Outline the factors which influence the fare amount.
- Understand if certain factors need to be considered to increase company income without making customers angry.
- Understand any pain points company might currently have.

2. Data Description

The New York City Taxi and Limousine Commission (TLC), created in 1971, is the agency responsible for licensing and regulating New York City's Medallion (Green) taxi cabs, for-hire vehicles (community-based liveries, black cars and luxury limousines), commuter vans, and paratransit vehicles.

In particular, the Green Trips dataset includes trips made by boro taxis, a taxi type created to fix the problem with reduced access to legal taxi rides in outer boroughs. This taxis can drop off passengers anywhere, but are allowed to pick them up only in certain locations (the Bronx, Brooklyn, Queens, and Staten Island and Northern part of Manhattan). You can learn more about this type of taxi on the [Wikipedia article about them](#)°.

The dataset includes trips from August of 2013, when the Green Taxi was first introduced, to current date. However, we decided to cover data from 2014 to 2024, since these years have data for all months.

2.1. Data Characteristics

Overall, we collected more than 83,000,000 records of taxi rides from 2014 to 2024. Size of the data is more than 1 GB.

The dataset contains 20 features:
- VendorID (int)
- lpep_pickup_datetime (timestamp)
- lpep_dropoff_datetime (timestamp)
- store_and_fwd_flag (text)
- RatecodeID (int)
- PULocationID (int)
- DOLocationID (int)
- passenger_count (int)
- trip_distance (float)
- fare_amount (float)
- extra (float)
- mta_tax (float)
- tip_amount (float)
- tolls_amount (float)
- improvement_surcharge (float)
- total_amount (float)
- payment_type (int)
- trip_type (int)
- congestion_surcharge (float)
- cbd_congestion_fee (float)

Description of the features is given on the [TLC Trip Record Data page](#)°.

3. Architecture of Data Pipeline

3.1. Stage 1

Input: TLC Official Amazon S3 (slow), parquet files

Step 1: Load to Yandex Cloud S3 (fast), parquet files

Step 2: Download to Local file system, parquet

Step 3: Load to HDFS, parquet files

Step 4: Spark: Preprocess, merge, transform, compress, snappy compressed parquet files

Step 5: Load data to PostgreSQL table, table

Step 6: Sqoop: Export to snappy compressed avro files, avro snappy files
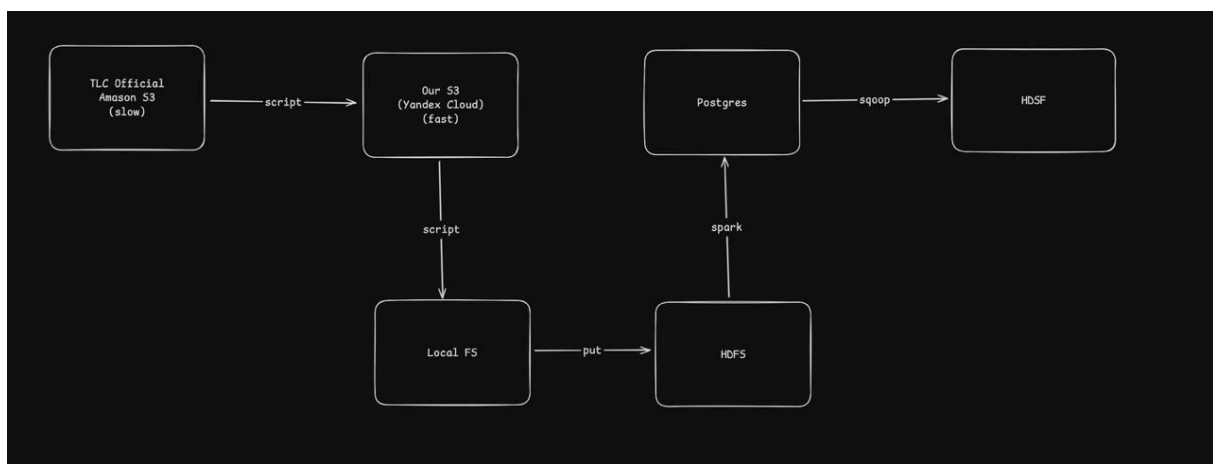
Output: HDFS, avro snappy files

3.2. Stage 2

Input: Database tables from HDFS as Snappy-Avro

Output: Hive tables, EDA in Apache Superset

3.3. Stage 3

Input: Dataframe in Hive

Output: data_split, models, model predictions

3.4. Stage 4

Input: Hive tables, EDA in Apache Superset, model predictions

Output: Dashboard

4. Data Preparation

Learn more about Data Preparation stage in the [README](#)°.

4.1. Data Loading

We load dataset from Yandex Cloud Object Storage. We decided to store dataset this way to ensure dataset is available for longer.

All downloading is performed via script. The script loads data from source based on provided year and month ranges. For example, the following script loads files from January 2014 to December 2024:

```
uv run download-sources.py \
    --base-url https://storage.yandexcloud.net/dartt0n/ibd/ \
    --start-year 2014 \
    --end-year 2024 \
    --start-month 1 \
    --end-month 12 \
    --file-prefix green_tripdata \
    --file-extension parquet \
    --output-dir ./data
```
<div align="center">Listing 1: Example of script for sources loading</div>

Separate scripts are responsible for creating table in PostgreSQL and loading data to it.

Script for table creation is written in Python. It uses the table schema from the sql folder and initializes the table. If table existed previously, it is dropped.

Script for Data Loading is written in Scala. A separate Spark job is responsible for processing each separate file. After all jobs are done, a separate job is created to load data to PostgreSQL. This job also ensures all data is converted to the same schema.

After data is loaded to PostgreSQL, we load it to Hive with sqoop.

4.2. PostgreSQL Dataset Schema

Here is query for creating table in PostgreSQL database.

```
create table green_tripdata (
  id bigserial primary key,
  vendorid bigint,
  lpep_pickup_datetime timestamp,
  lpep_dropoff_datetime timestamp,
  store_and_fwd_flag text,
  ratecodeid bigint,
  pulocationid bigint,
  dolocationid bigint,
  passenger_count bigint,
  trip_distance double precision,
  fare_amount double precision,
  extra double precision,
  mta_tax double precision,
  tip_amount double precision,
  tolls_amount double precision,
  ehail_fee double precision,
  improvement_surcharge double precision,
  total_amount double precision,
  payment_type bigint,
  trip_type bigint,
  congestion_surcharge double precision,
  year int,
  month int
);
```

Listing 2: Query for database creation

4.3. Data Sample

Here is how one row of the dataset looks like:

```
vendorid: 2
lpep_pickup_datetime: 01.01.2014  0:17:26
lpep_dropoff_datetime: 01.01.2014  0:37:11
store_and_fwd_flag: N
ratecodeid: 1
pulocationid: 17
dolocationid: 225
passenger_count: 1
trip_distance: 2.28
fare_amount: 13.5
extra: 0.5
mta_tax: 0.5
tip_amount: 0
tolls_amount: 0
ehail_fee: NULL
improvement_surcharge: NULL
total_amount: 14.5
payment_type: 2
trip_type: NULL
congestion_surcharge: NULL
year: 2014
month: 1
```

Listing 3: Sample row from the dataset

4.4. Hive

Here is the query for creating external Hive table:

```
create external table
  green_tripdata (
      vendorid bigint,
      lpep_pickup_datetime timestamp,
      lpep_dropoff_datetime timestamp,
      store_and_fwd_flag string,
      ratecodeid bigint,
      pulocationid bigint,
      dolocationid bigint,
      passenger_count bigint,
      trip_distance double,
      fare_amount double,
      extra double,
      mta_tax double,
      tip_amount double,
      tolls_amount double,
      ehail_fee double,
      improvement_surcharge double,
      total_amount double,
      payment_type bigint,
      trip_type bigint,
      congestion_surcharge double,
      year int,
      month int
  )
stored as avro location 'project/warehouse/green_tripdata' tblproperties (
  'avro.schema.url' = 'project/warehouse/avsc/schema.avsc',
  'avro.compress' = 'snappy'
);
```

Listing 4: Query for Hive external table

However, this table is really big and has to be partitioned. We partitioned it by month and year.

We did not use bucketing since we don't have a column by which we can create evenly distributed dataset, and we do not need to group the data by any parameters except for the ones already used in partitioning.

```
create external table
  green_tripdata_monthly (
      vendorid bigint,
      lpep_pickup_datetime timestamp,
      lpep_dropoff_datetime timestamp,
      store_and_fwd_flag string,
      ratecodeid bigint,
      pulocationid bigint,
      dolocationid bigint,
      passenger_count bigint,
      trip_distance double,
      fare_amount double,
      extra double,
      mta_tax double,
      tip_amount double,
      tolls_amount double,
      ehail_fee double,
      improvement_surcharge double,
      total_amount double,
      payment_type bigint,
      trip_type bigint,
      congestion_surcharge double
  ) partitioned by (year int, month int)
stored as avro location 'project/hive/warehouse/green_tripdata_monthly'
tblproperties ('avro.compress' = 'snappy');

insert
  overwrite table green_tripdata_monthly partition (year, month)
select
  *
from
  green_tripdata;
```

Listing 5: Partitioned Hive table

5. Data Analysis

We wanted to get valuable insights into the data before proceeding with modeling. For this reason, we created 5 queries that would gather statistics for us, and plotted graphs in the Apache Superset.

5.1. Created Queries

5.1.1. Missing value percentage

The first query simply counts number of missing (NULL) values in every column.

5.1.2. Invalid rows percentage

The second query calculates number of weird rows in our data. The row is considered weird if:
• drop-off happened before pick-up,
• distance of the trip is 0,
• there are 0 passengers,
• pick-up and drop-off locations are the same.

While these rows are technically valid, since they do not contain NULL values, working with them will provide unexpected results, since the row itself does not contain a valid trip.

5.1.3. Correlations with price

We filtered out invalid rows and calculated correlation between price and other features:
• duration,
• distance,
• number of passengers.

These values may give us valuable insights on what influences the price.

5.1.4. Duration

We filtered out invalid rows and calculated a separate table that would store duration for us, so we would not have to recompute it for Apache Superset. Apart for duration, the table contains:
• year,
• month,
• price,
• number of passengers,
• distance.

5.1.5. Pick-up and Drop-off hour

We filtered out invalid rows and created a separate table with the following columns:
• pick-up hour,
• drop-off hour,
• price.

These table was needed to allow us to see if time of the time influences the trip cost.

5.2. Analysis Results

5.2.1. Travel distance

5.2.1.1. Market share loss & trip number collapse

- From 2014 to early 2020, green taxis experienced a significant decline in both total distance travelled and, the total number of trips. This reflects significant market share loss, primarily due to the rise of services like Uber.
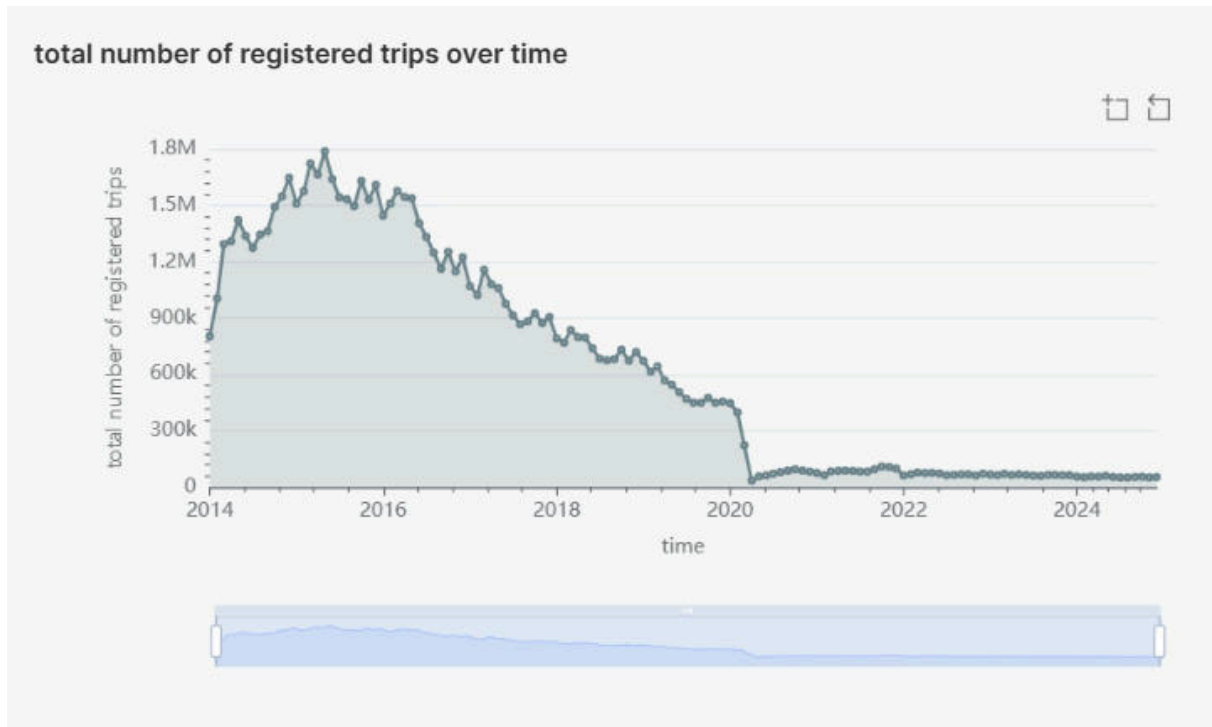- The COVID-19 pandemic then triggered a near-collapse in trip amount, which has remained exceptionally low post-pandemic.

Figure 1: Number of trips over time

5.2.1.2. Long distance travel boom

- Despite low trip numbers post-COVID, the total distance travelled saw a big temporary spike from mid-2020 to mid-2022.
- This was driven by a great increase in the average distance travelled per taxi trip during the same period.
- This indicates a temporary period where green taxis, served a demand for long trips. This might have been due to reduced availability of other long-distance transport during the pandemic's peak, or people moving further from the city due to post-COVID housing market collapse in NYC.
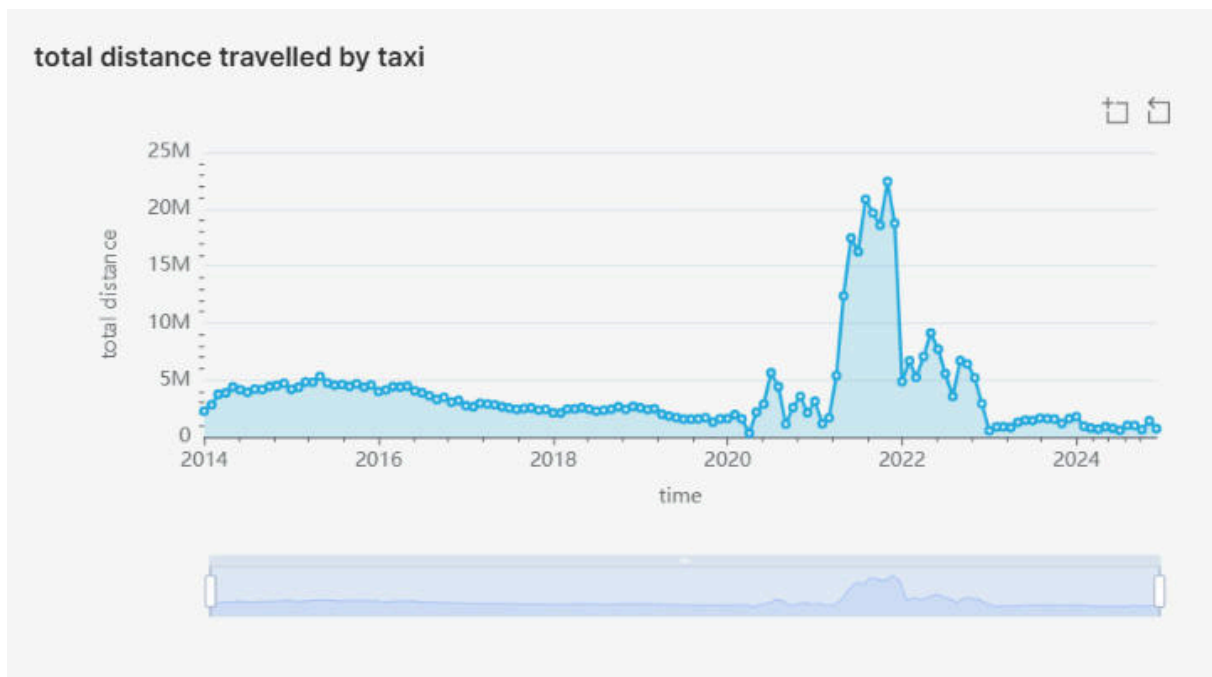
Figure 2: Total trip distance over time

5.2.1.3. Shift to longer trips

- The average trip distance by time and average travel distance per year charts confirm the dramatic increase in trip length, with 2021 standing out as an extreme outlier (average 147.57 miles/trip).
- While the extreme peak was temporary, average trip distances have settled at a new, higher baseline compared to pre-2020 levels.
- Green taxis are now serving a different primary use case – fewer customers taking significantly longer journeys. Business strategy should target this demographic and trip type.
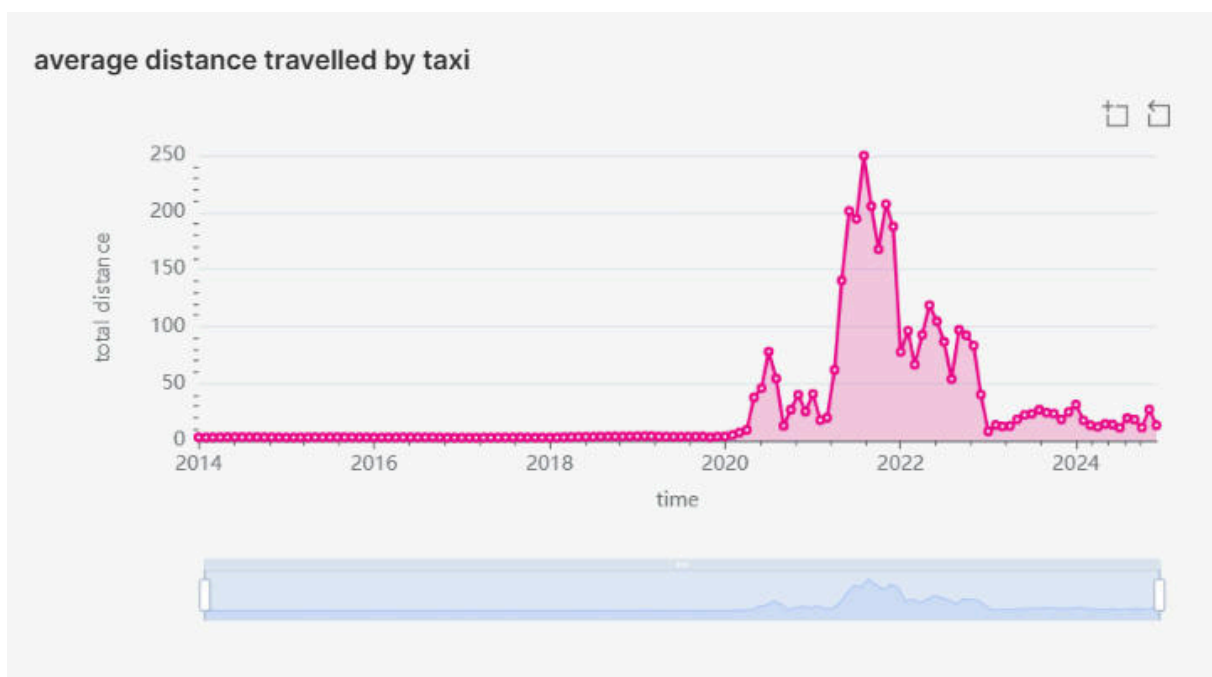
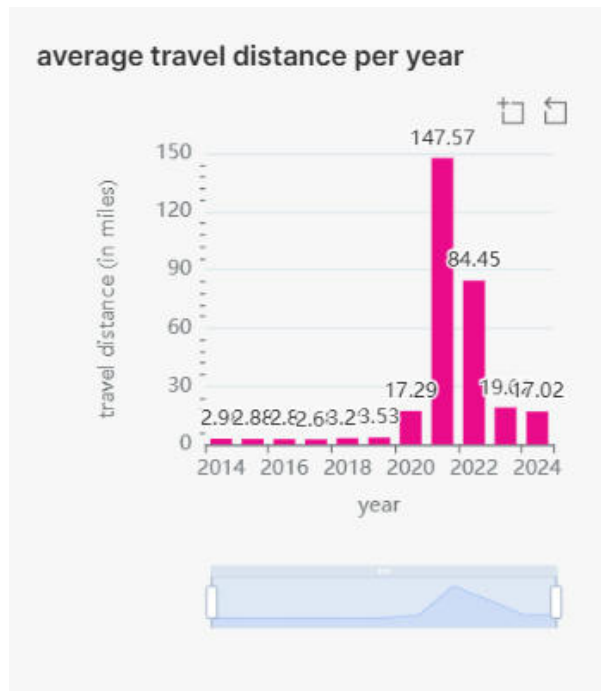Figure 3: Average distance travelled by taxi over time

Figure 4: Average trip distance per year

5.2.1.4. Seasonality and pricing shift

- A seasonal pattern shows slightly longer average trips in the latter half of the year, especially in autumn. This could be related to family holidays such as Thanksgiving, Halloween, Christmas, New Years and etc.

Figure 5: Average trip distance by month

- The correlation of trip distance and price by year shows a fundamental change: a strong correlation (approx. 0.8) between distance and price from 2016-2018 almost completely disappeared from 2020 onwards.

- This removal of distance from the price could be the key factor of the "longer trip" change. With distance becoming less of a factor in fare calculation, green taxis became more economically available for longer distance.

Figure 6: Trip distance and price correlation per year

5.2.2. Hourly demand

5.2.2.1. Demand peaks

- The number of rides per pickup hour shows two primary demand peaks: a morning rush (around 8 AM) and a bigger evening rush (peaking between 5 PM - 7 PM
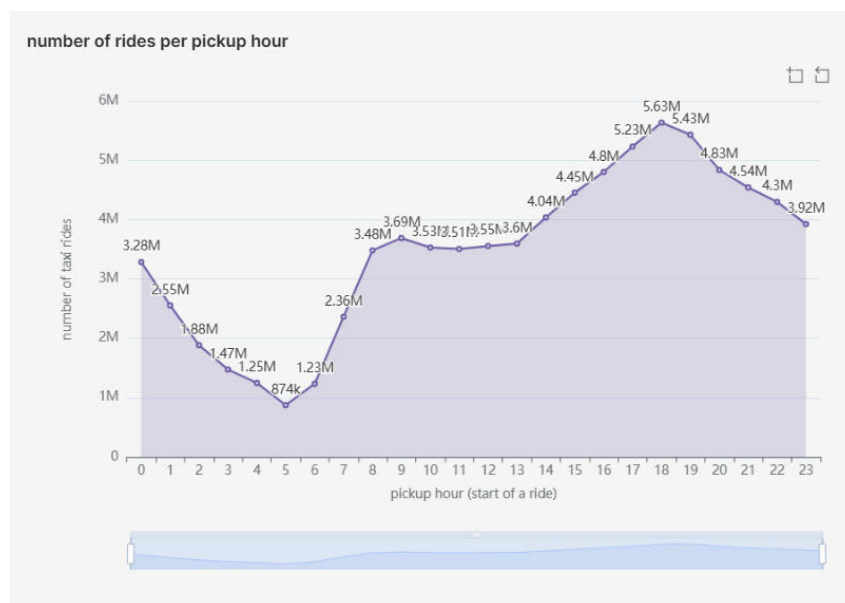

Figure 7: Number of rides per pick-up hour

- Total earnings per pickup hour support this, with the 5 PM - 7 PM window being by far the most earning, generating significantly higher revenue than any other time.
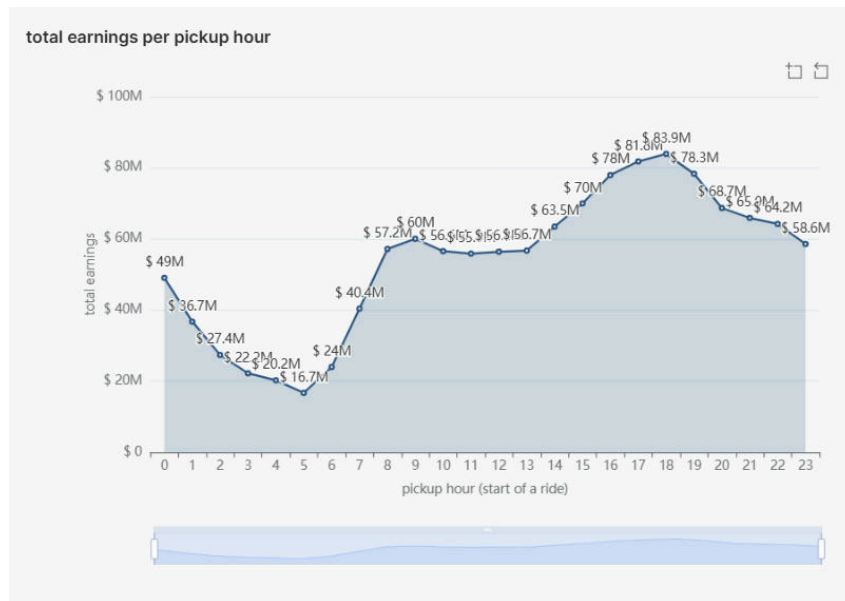
Figure 8: Total earnings per pick-up hour

- Business Implication: Operational focus, including driver availability and deployment, should be maximized during the 5 PM - 7 PM evening peak to capitalize on the highest ride volume and revenue generation. The morning rush is a secondary, but still important, revenue period.

5.2.2.2. Morning peak for average price, evening for total revenue

- Interestingly, the average price per pickup hour peaks earlier, around 6 AM - 7 AM (approx. $16-17$). While ride amount isn't at its highest then, these early morning trips show a higher average fare.
- The lowest average prices, ride volumes, and earnings occur in the very early morning hours (approx. 3 AM - 5 AM).
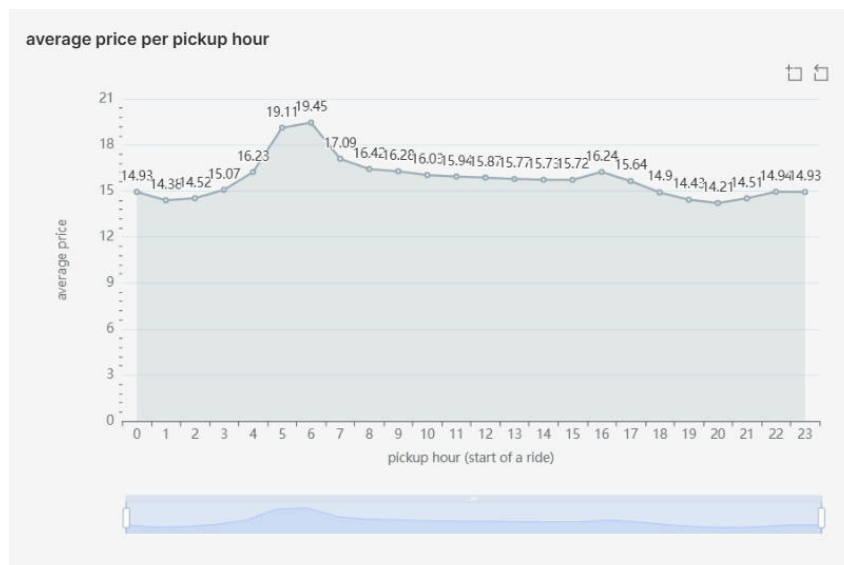

Figure 9: Average price per pick-up hour

5.2.2.3. Surge pricing and Trip duration impact

- The heatmap shows surge pricing well. Trips picked up and dropped off within, or entering, peak demand hours (e.g. pickup 4 PM, dropoff 5-6 PM) are consistently more expensive.
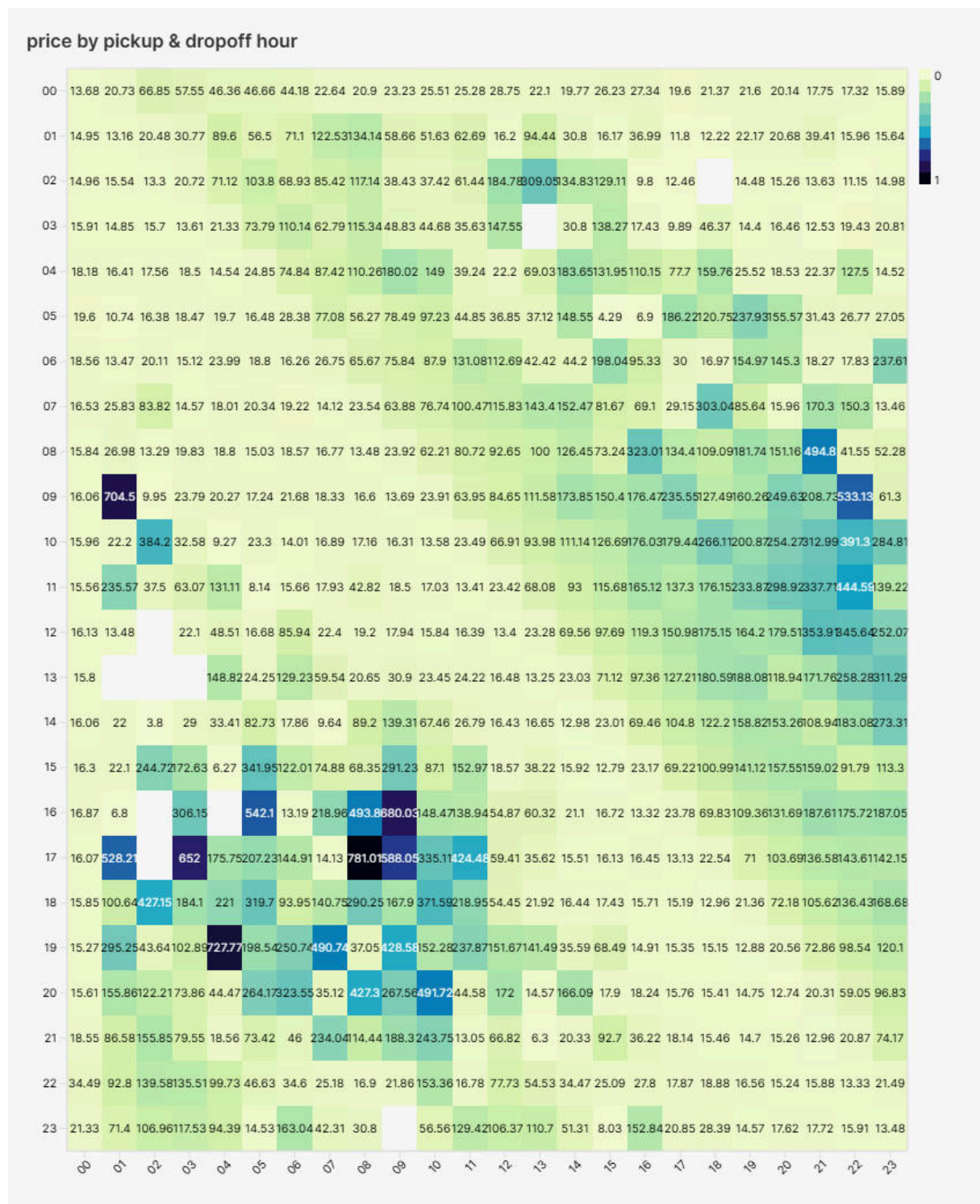

Figure 10: Average price per pick-up and drop-off hour

- The highest prices on the heatmap often correspond to trips that span multiple peak periods or are of very long duration. For example, a trip picked up in the mid-afternoon (e.g., 2-3 PM) and dropped off during the evening peak (5-7 PM) shows significantly higher prices. Similarly, trips picked up before the morning rush and extending into it are also priced higher.

- Business Implication: The fare structure effectively captures premium pricing not just for short trips during peak hours, but also for longer trips. This rewards drivers for longer commitments that coincide with peak demand and compensates for the opportunity cost of potentially missing multiple shorter, high-demand fares.

5.2.2.4. High value trip scenarios

- The darkest cells (highest prices) highlight specific pickup/dropoff hour combinations that are most expensive on a per-trip basis. These often involve trips starting before a peak and ending during or after it, or trips of extended duration.
- While these specific ultra-high-fare scenarios might be less frequent, understanding these patterns helps validate the dynamic pricing algorithm's effectiveness in maximizing revenue for complex trip types. Trip pricing is sensitive not just to the pickup hour, but to the interaction of pickup time, dropoff time, and duration, especially in relation to demand peaks.

5.2.3. Drop-off

5.2.4. Price and Seasonality Insights

5.2.4.1. Average trip price increase

Figure 11: Average trip price over time

average trip price by year [barchart]

Figure 12: Average price by year

- Before 2019, average taxi fares increased slowly, mostly keeping up with US inflation (around 1-2.5% a year). This meant the real cost to customers didn't change much.
- From 2019/2020, average fares jumped up, from about $17 to a peak of $25 in 2021, then settled around $20-$24.
- Even when US inflation was high (like 9% in 2022), taxi fare increases were much bigger. For instance, fares rose about 47% (from $17 to $25) much faster than overall price increases in the country. This might be due to COVID making the job of a driver much more dangerous, making the drivers more scarce. And, at the same time, decreasing the demand for public transport.
- This big jump in average fares means taxis are making more **real** money per trip, not just because of general inflation. Apart from COVID, this is likely due to:
  1. Longer trips naturally costing more.
  2. New pricing. Since fares no longer strictly depend on distance, allowing higher prices for all types of trips.
  3. Fewer Taxis. Less available drivers at times might have allowed higher prices.
  4. Inflation. General price rises made it easier to raise fares, and helped cover higher taxi running costs (like fuel).

5.2.4.2. Demand patterns

Regular weekly patterns in the number of rides and total income, plus steady average monthly fares, show that the price is consistent throughout the year.
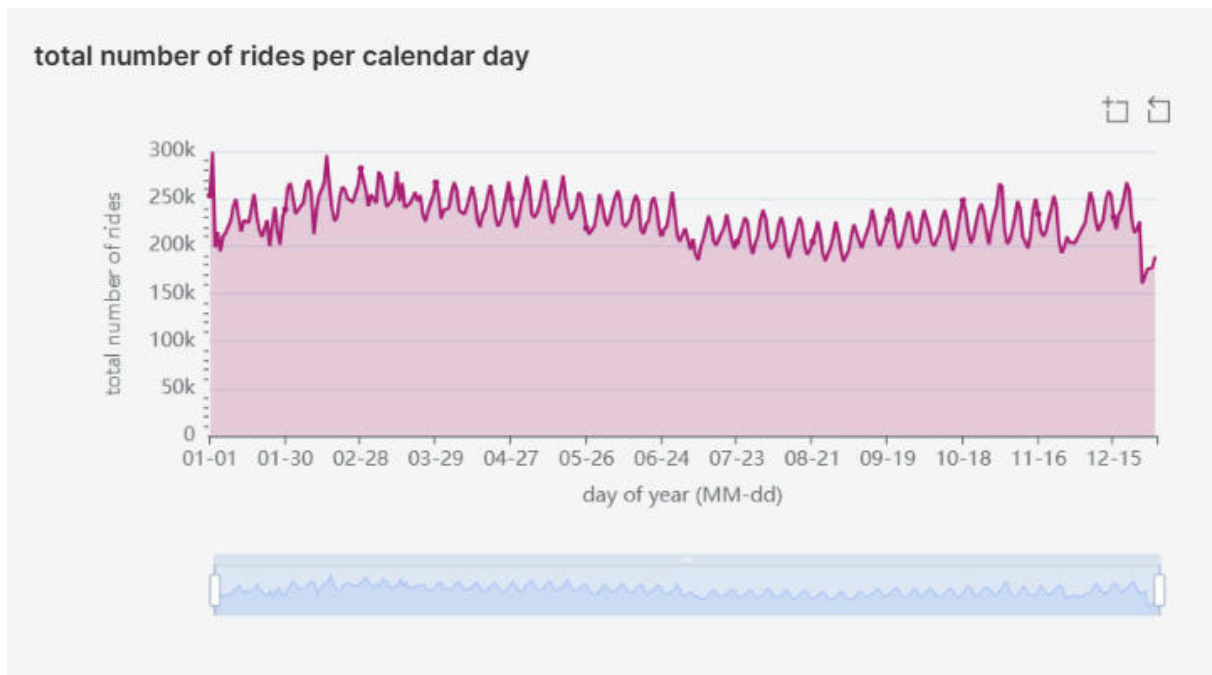

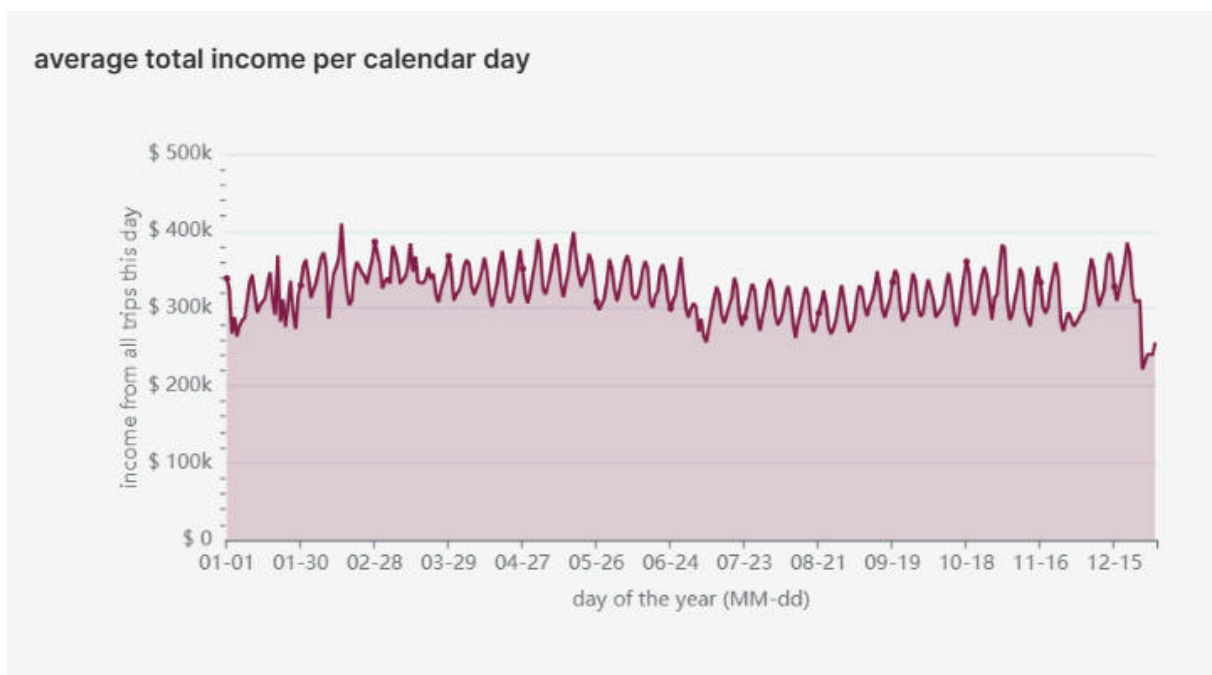Figure 13: Number of rides per date
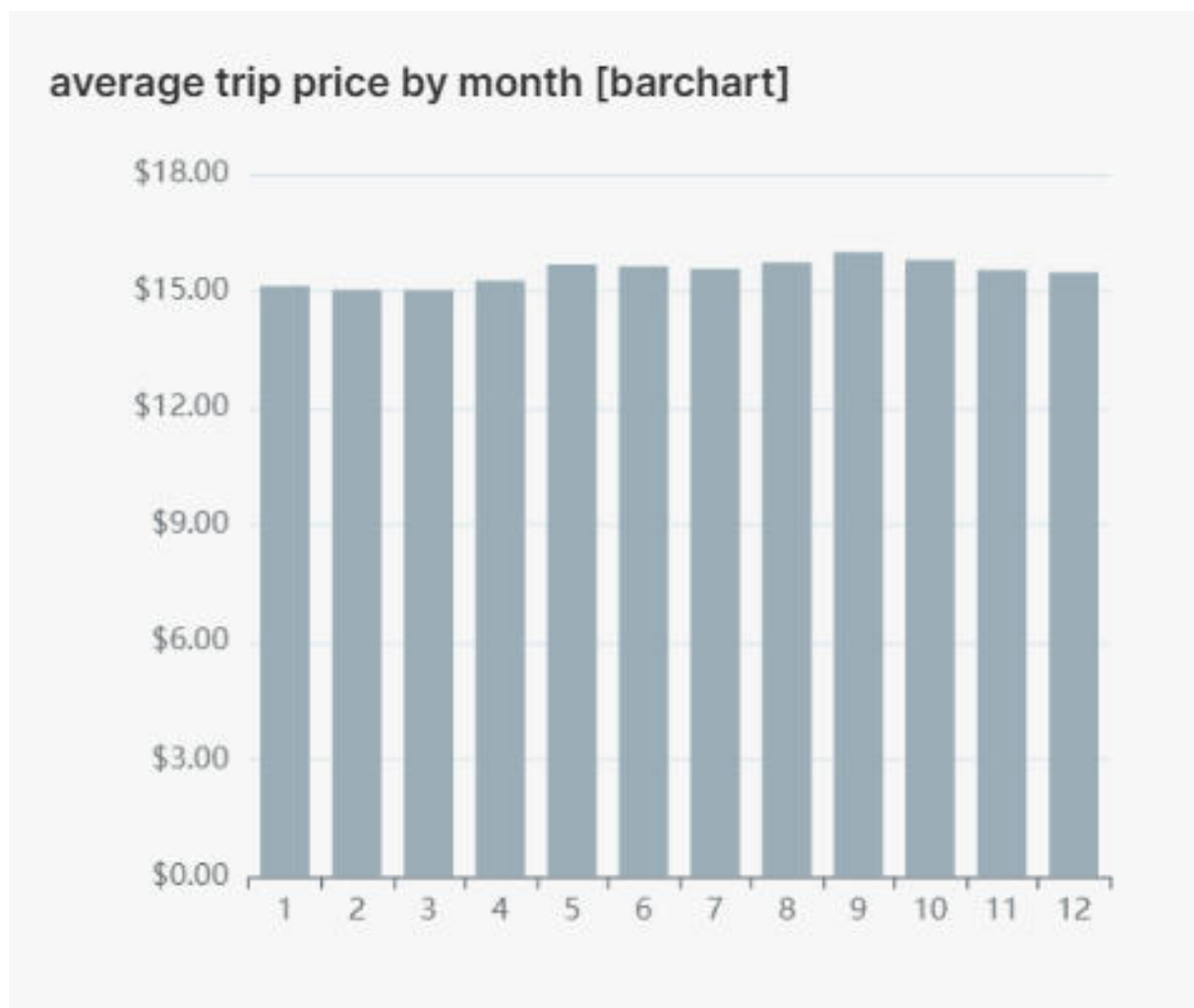

Figure 14: Average total income per date

Figure 15: Average price by month

5.2.5. Conclusion

In short, while US inflation played a part, green taxi fares rose much faster, especially after 2019. This shows big changes in how they operate (longer trips, new pricing rules) and market conditions, letting them earn more real value per ride.

6. ML Modeling

6.1. Feature Extraction and Data Preprocessing

6.1.1. Data dropping

Since some time in 2020 the formula for the price have possibly changed, we decided to train the model only on the data from 2021 and after.

Moreover, we have dropped the features which do not influence the fare_amount, since they are the added costs added on top of it. At the end, we are left with:

- vendorid
- ratecodeid
- payment_type
- trip_type
- lpep_pickup_datetime
- passenger_count
- trip_distance
- extra
- mta_tax
- tip_amount
- tolls_amount

6.1.2. Categorical features

We had four categorical features:

- Vendor ID
- Ratecode ID
- Payment type
- Trip type

They all are already integers, however, their value does not give any useful information. So, we used One Hot Encoder to transform them into categorical vectors that will be more useful for the model.

6.1.3. Date and time features

We encoded `lpep_pickup_datetime` feature into separate values:

- year;
- month, encoded with sin and cos;
- weekday, encoded with sin and cos;
- hour, encoded with sin and cos;
- minute, encoded with sin and cos;
- second, encoded with sin and cos;

Encoding cyclic features with sin and cos is a common practice, which allows to find meaningful correlations, since just encoding as a number will not give too much practical value. For example, January and December are close together, but will be encoded as 1 and 12, and will be the furthest value.

6.1.4. Numerical features

To avoid model preferring some value just because of bigger range, we have to bring all numerical features to same range. We did it using Standard Scaler.

6.1.5. Concatenation

At the end, we concatenate all features into single feature vector using Vector Assembler.

After all the cleaning, we had the following dataset sizes:


```
team18@hadoop-01:~/project$ wc -l data/train.json
2218239 data/train.json
```

Figure 16: Train dataset size


```
team18@hadoop-01:~/project$ wc -l data/test.json
555481 data/test.json
```

Figure 17: Test dataset size

6.2. Training and fine-tuning

We tested different model architectures to see which one would perform the best.

6.2.1. Linear Regression

For linear regression, we have tried out the following hyperparemeters:

- regParam: [0.01, 0.1, 1]
- elasticNetParam: [0.0, 0.5, 1.0]

6.2.2. Random Forest Regression

For random forest, we have tried out the following hyperparemeters:

- numTrees: [25, 50, 75]
- maxDepth: [5, 7, 9]

6.3. Evaluation

For evaluation, we have used R2 and RMSE. These are common metrics for regression, which allow both to see if model performance is good and if model captures data correlations.

We have obtained the following results:

linear regression metrics [table]

model	rmse	r2
LinearRegression	14.752092861088691	0.22984644283559685

Figure 18: Linear Regression results, for regParam = 0.01, elasticNetParam = 0.0

random forest regression metrics

model	rmse	r2	numtree	maxdepth
RandomForestRegressor	11.60975685222611	0.5230022844384274	75	5

Figure 19: Random Forest Regression results, for numtree = 75, maxDepth = 5

As we can observe, Random Forest produces much better results in terms of capturing data correlation.

7. Data Presentation

We have created an [Apache Superset dashboard]° to present our project findings.

The dashboard contains of four tabs. Each tab represents separate view of our project: general, from business point of view, from data analysts point of view and from ML engineer point of view.

7.1. Overview

The overview tab contains description of dataset, dataset sample and general description of steps we performed during the project.

The tab helps understand what was done during the project in general, as well as see an example of data we worked with.
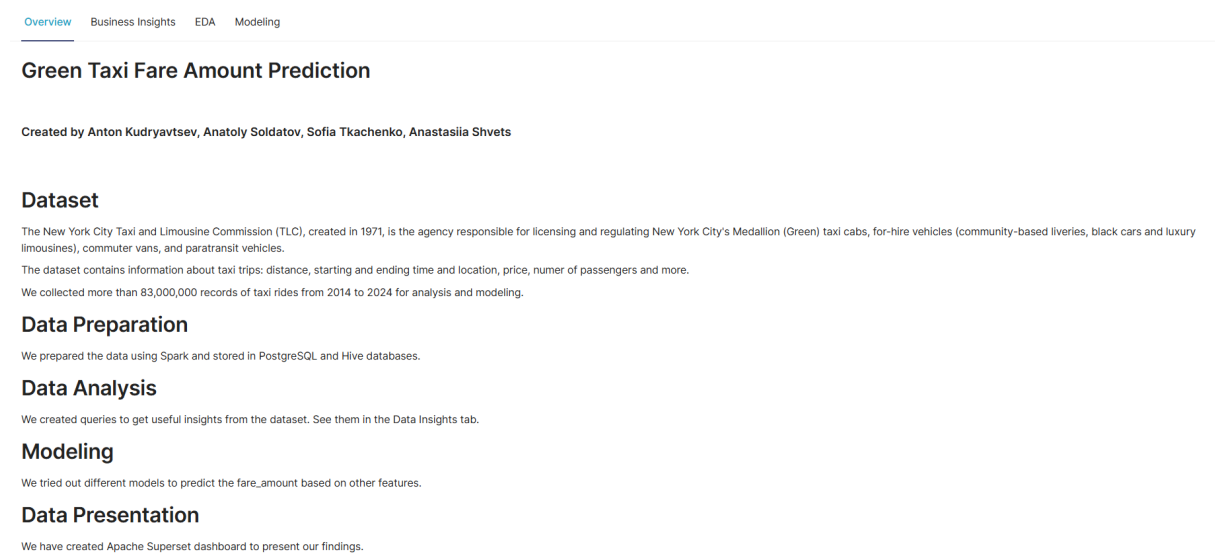
Figure 20: Overview tab, part 1

Figure 21: Overview tab, part 2

7.2. Business Insights

The tab contains insight useful from business point of view. This tab was created to bring value to the company, without needing to explain technical details.

Figure 22: Business Insights tab, part 1

Figure 23: Business Insights tab, part 2

Figure 24: Business Insights tab, part 3

price by pickup & dropoff hour

Hourly demand Insights 2

Surge pricing and Trip duration impact

- The heatmap shows surge pricing well. Trips picked up and dropped off within, or entering, peak demand hours (e.g., pickup 6 AM, dropoff 7-8 AM; or pickup 4 PM, dropoff 5-6 PM) are consistently more expensive.
- The highest prices on the heatmap often correspond to trips that span multiple peak periods or are of very long duration. For example, a trip picked up in the mid-afternoon (e.g., 2-3 PM) and dropped off during the evening peak (5-7 PM) shows significantly higher prices. Similarly, trips picked up before the morning rush and extending into it are also priced higher.
- Business Implication: The fare structure effectively captures premium pricing not just for short trips during peak hours, but also for longer trips. This rewards drivers for longer commitments that coincide with peak demand and compensates for the opportunity cost of potentially missing multiple shorter, high-demand fares.

High value trip scenarios

- The darkest cells (highest prices) highlight specific pickup/dropoff hour combinations that are most expensive on a per-trip basis. These often involve trips starting before a peak and ending during or after it, or trips of extended duration.
- While these specific ultra-high-fare scenarios might be less frequent, understanding these patterns helps validate the dynamic pricing algorithm's effectiveness in maximizing revenue for complex trip types. Trip pricing is sensitive not just to the pickup hour, but to the interaction of pickup time, dropoff time, and duration, especially in relation to demand peaks.

Unfortunately, Superset doesn't allow to add axis names to a heatmap. The y axis is pickup hour and x axis is dropoff hour.

Figure 25: Business Insights tab, part 4

Drop Off Hour Insights

Evening rush is key for trip endings and earnings

- Most trips end, and the most money is made, during the evening rush (around 5 PM - 7 PM), just like with pickups. The average price per dropoff hour is highest for trips ending in the morning rush (6 AM - 8 AM).

number of rides per dropoff hour [barchart]

total earnings per dropoff hour [barchart]

Figure 26: Business Insights tab, part 5
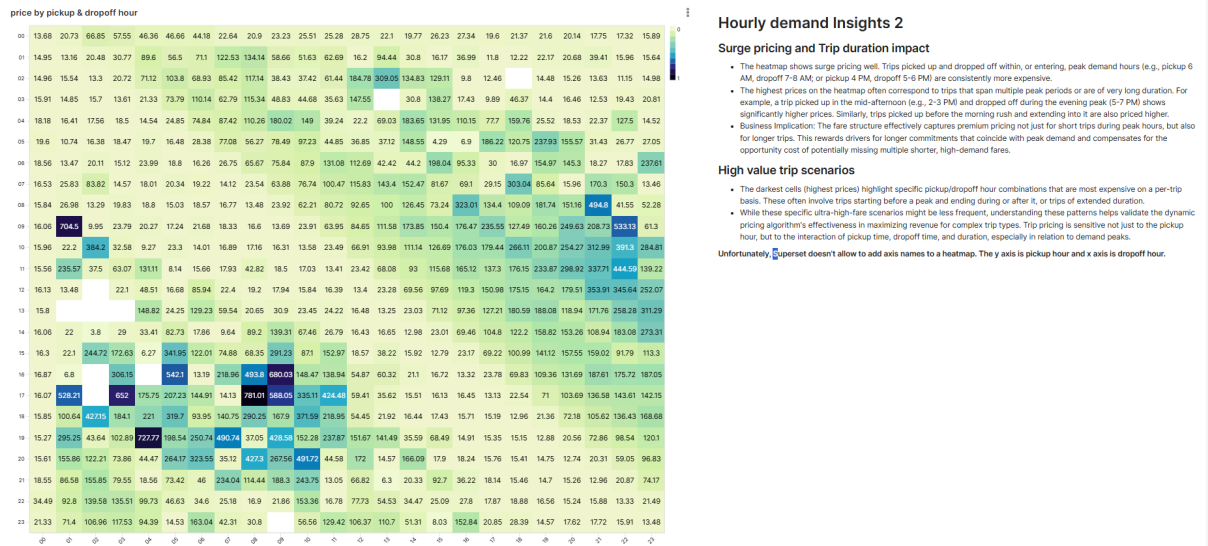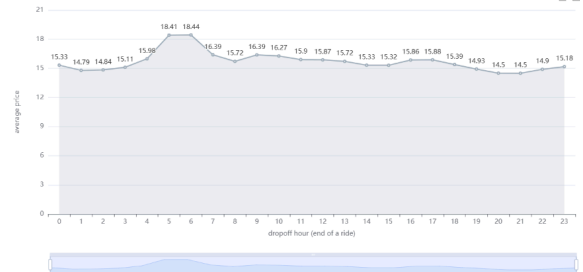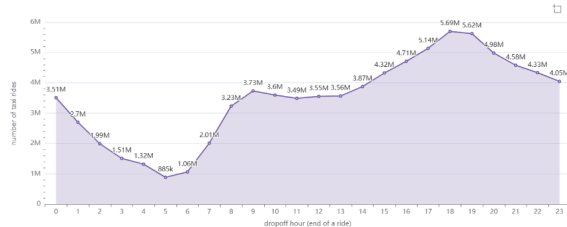
Price & seasonality Insights

Average trip price increase

- Before 2019, average taxi fares increased slowly, mostly keeping up with US inflation (around 1-2.5% a year). This meant the real cost to customers didn't change much.
- From 2019/2020, average fares jumped up, from about $17 to a peak of $25 in 2021, then settled around $20-$24.
- Even when US inflation was high (like 9% in 2022), taxi fare increases were much bigger. For instance, fares rose about 47% (from ~$17 to ~$25) much faster than overall price increases in the country. This might be due to COVID making the job of a driver much more dangerous, making the drivers more scarce. And, at the same time, decreasing the demand for public transportsation.
- This big jump in average fares means taxis are making more *real* money per trip, not just because of general inflation. Apart from COVID, this is likely due to:
    1. Longer trips naturally costing more.
    2. New pricing. Since fares no longer strictly depend on distance, allowing higher prices for all types of trips.
    3. Fewer Taxis. Less available drivers at times might have allowed higher prices.
    4. Inflation. General price rises made it easier to raise fares, and helped cover higher taxi running costs (like fuel).

Demand patterns

- Regular weekly patterns in the number of rides and total income, plus steady average monthly fares, show that the price is consistent throughout the year.

Conclusion

In short, while US inflation played a part, green taxi fares rose much faster, especially after 2019. This shows big changes in how they operate (longer trips, new pricing rules) and market conditions, letting them earn more real value per ride.
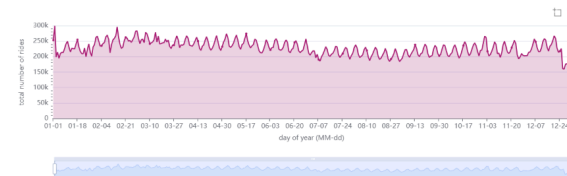
average total income per calendar day

average price per calendar day

total number of rides per calendar day

Figure 27: Business Insights tab, part 6

Figure 28: Business Insights tab, part 7

7.2.1. Travel distance insights

The graph **total number of registered trips over time** shows how total number of trips changed during the years. We can observe that the number of trips decreased dramatically in the Spring of 2020. The possible causes are rise of Uber popularity, as well as Covid-19.

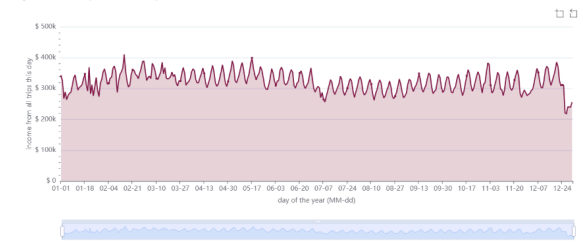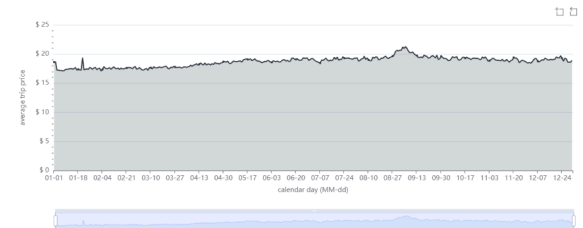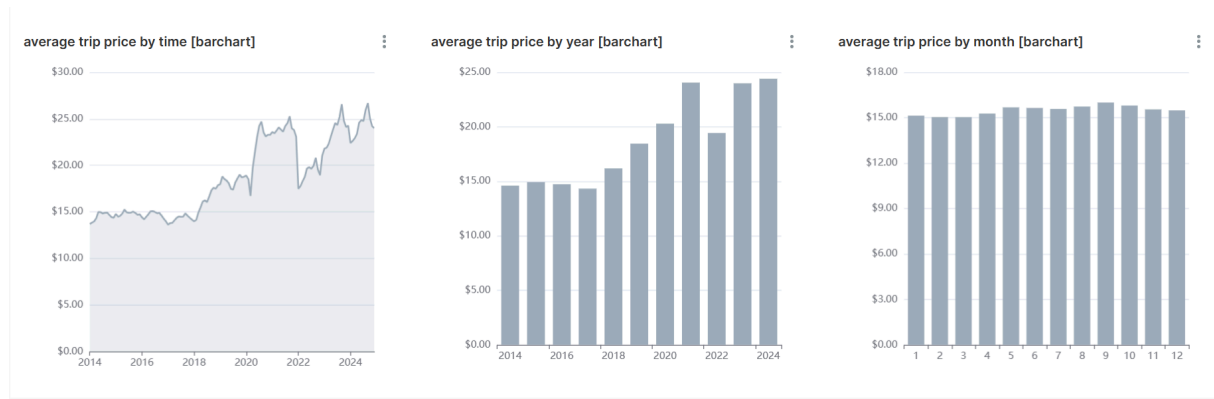The graphs **total distance travelled by taxi**, **average distance travelled by taxi** and **average travel distance per year** show how the total trip distance and average trip distance changed over time. We can observe a plateau around 2-3 km till the beginning of 2020, and numerous spikes till 2023, and again plateau at around 15 km after that. The possible cause of spikes is again Covid-19.

The graph **average trip distance by month** shows how average trip distance changes depending on month. The peaks are present during Summer, Autumn, and December. The possible reasons are holidays during that time of year (Thanksgiving, Christmas), or university enrollment process (Summer).

The graph **correlation of trip distance and price by year** shows that the price correlated significantly with distance till 2019, when the correlation dropped. The reason is change in the formula: the trip distance was no longer a factor.

The business insights from the graphs are:
- Since 2020, NYC taxi company has a much smaller customer base and fewer overall trips.
- The core service has shifted towards longer distance trips.
- Trip fares are no longer strongly tied to distance, a crucial factor supporting the longer trips.

7.2.2. Pick-up and drop-off hours insights

The graph **average price per pickup hour** shows how average price changes during the day. We see that there is little change, expect for slight increase at around 5-6 A.M.

However, the graphs **number of rides per pickup hour** and **total earnings per pickup hour** tell us, that there are actually peak hours at the end of working day (around 18 o'clock), and drop at night (around 5 o'clock) in the morning. This is due to people rarely travelling during the night, especially at dawn hours.

The graph **price by pickup & dropoff hour** shows how average price changes depending on pick-up and drop-off hours. We can see that the rides within 2 hours (diagonal from left top to right bottom corner) are the cheapest ones. Moreover, it is on average more expensive

to take long trips during the night (starting in the evening and ending in the morning, top left corner), than during the day (starting in the morning and ending in the evening, bottom left corner).

The graph **average price per dropoff hour** shows how average price changes depending on the drop-off hour. We, again, can observe peak at around 5-6 o'clock.

Similar to their pick-up counterparts, the graphs **number of rides per dropoff hour** and **total earnings per dropoff hour** show us rush hours at 18-19 o'clock, and slow hours at 5-6 o'clock.

The business insights from the graphs are:
- The company does not increase price during peak hours, while definitely could, since the demand is there. On the contrary, the highest average price is at the least popular hours (5-6).
- The customers may prefer to travel during the day due to cheaper cost.

7.2.3. Price and seasonality insights

The graphs **average total income per calendar day** and **total number of rides per calendar day** show how number of rides and total income change per date. We can observe seasonal patterns (a peak occurs every 7th day, indicating weekly pattern).

The graph **average price per calendar day** shows how average price changes during each date. We can see that the rate does not change that much, with peaks at the beginning of September (rides to schools, colleges and universities) and 1st and 20th of January (probably kids going back to college after Christmas).

7.3. EDA

Data Insights tab contains our findings:

- NULL count and percentage: the graph shows how much NULL values are present in each column, and price column NULL values percentage separately.
  The insight from this graph is that some columns contain high percentage of null values (for example, ehail_fee), but the price column has no null values, so the target value is always present.
- Number of rows and invalid rows: the number shows how much rows we have overall, and how much of them contain weird data (for example, when pick-up happened after drop-off). The insight from this number is that there might be some problem with data transmitters in cars, or with storing the data.
- Rate code by year and Rate code description: the bar chart shows which rate codes are most common for every year, while the text description explains what each number means.
  The insight from this graph is that most of the trips are done using standard rate, and not from the airports.
- Correlation of price with duration and number of passengers: the bar charts present how these values affected price each year.
  The insight from the graphs is that the correlation between the values in small, so this factors alone do not influence the price greatly.

- Average trip duration and passenger count by year, month and overall date: the bar charts show how these values changed during different time periods.

  The insight from the graph is these factors did not change significantly over time. Another insight is that the way of calculating distance could have changed overtime, since distance spiked in 2021, while duration stayed the same.
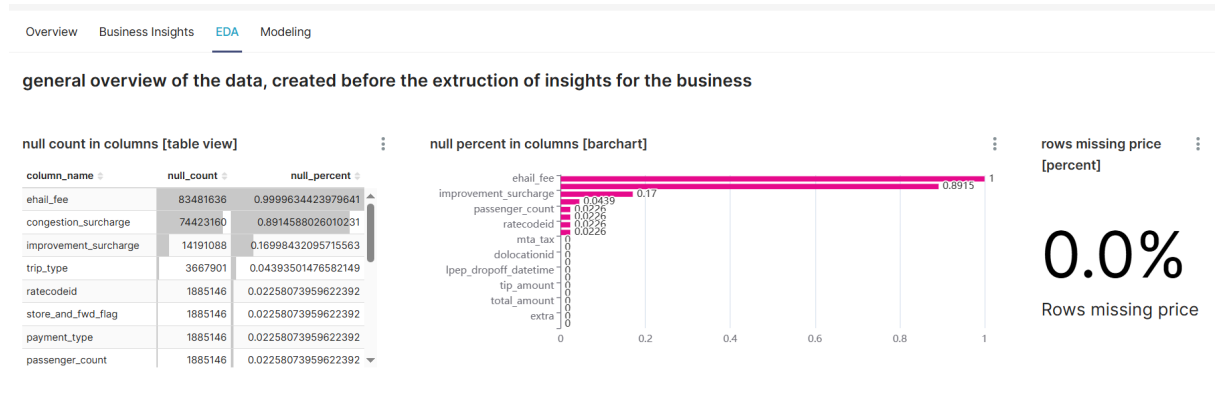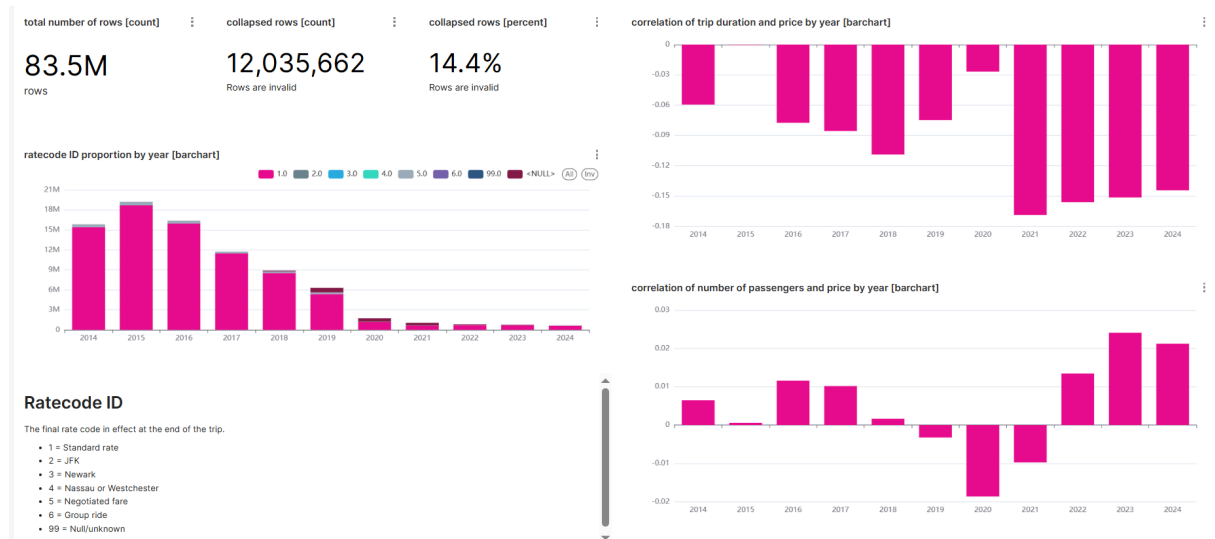

Figure 29: EDA tab, part 1

Figure 30: EDA tab, part 2

Figure 31: EDA tab, part 3

7.4. Modeling

The modeling part contains:

- Training results: best hyperparameters for each model type, and metrics for them
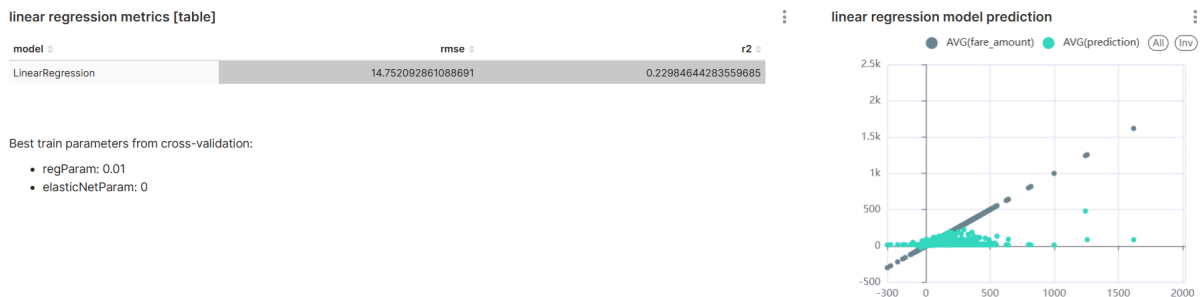- Predictions compared with actual amount.
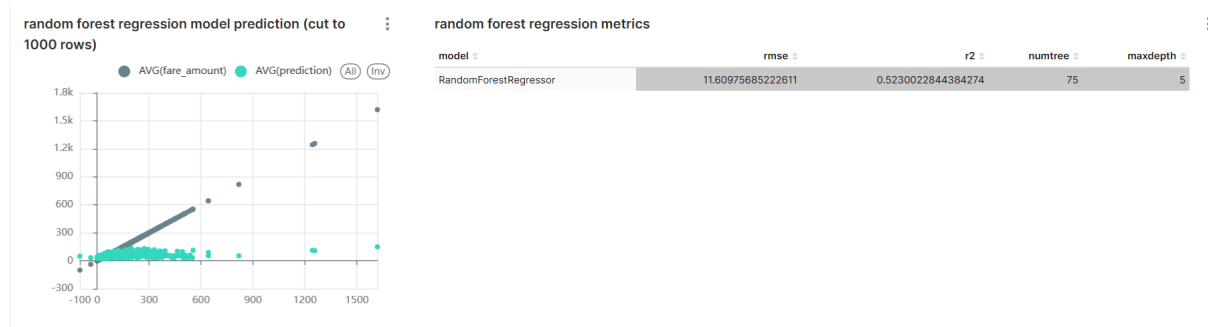

Figure 32: Modeling tab, part 1

Figure 33: Modeling tab, part 2

8. Conclusion

8.1. Summary

We have analyzed a Green Taxi trips dataset and build an ML Model using subset of this dataset.

We have found multiple helpful business insights during EDA stage that can be used by the company to improve their business. However, certain parameters did not have a correlation to fare amount on their own.

The modeling part confirmed our suspicions: the formula for the price must be more complex than we initially thought, so most probably better features need to be created from existing data to get better results.

8.2. Business Insights

While working on the project, we have gathered insights useful for NYC Green taxi:

1. The customer share has dropped significantly since 2020. The way to overcome this could be mimicking the competitors (especially Uber), or switching to different model (for example, doing only pre-booked rides)
2. The core service has shifted towards longer distance trips. This is exactly why pre-booked rides could be useful, since people usually plan longer rides in advance.
3. Trip fares are no longer strongly tied to distance, a crucial factor supporting the longer trips. However, there is a need to keep in mind other factors that could influence the price.
4. Operational focus, including driver availability and deployment, should be maximized during the 5 PM - 7 PM evening peak to capitalize on the highest ride volume and revenue generation. Moreover, the price during these hours should be increased, since the demand is present.
5. Trip pricing is sensitive not just to the pickup hour, but to the interaction of pick-up time, drop-off time, and duration, especially in relation to demand peaks.

9. Reflections of Own Work

9.1. Challenges and Difficulties

9.1.1. Different Data Schemes

One of the first difficulties we encountered was that schema of the dataset during the time period we covered. Some of the fields changed from int32 to int64, some were converted from integers to float point numbers, certain fields started containing null values. We tried out different solutions to overcome this problem, but finally decided to use separate Spark jobs to convert each file to desired schema, and then merge files into one via separate Spark job.

9.1.2. Hive

Certain queries in Hive took too much time compared to Spark queries, but consumed same or even more resources. While we tried to optimize Hive queries, the best solution was to rewrite the query in Spark.

9.1.3. Cluster overconsumption

During some times, cluster had no free memory or CPU. To overcome this problem, Anton had to analyze cluster usage and find zombie or weird processes, that can be stopped to free up cluster memory and CPU.

9.1.4. Debugging

Since HDFS, Spark and Hive are all written on Java, while we operate in Python, there were problems with understanding errors from the Java code. Exceptions and errors from Spark were sometimes omitted. Therefore, we new that, for example, a Spark job has failed but we could not see the internal error. Or, on the other hand, some error messages were too verbose to find the root cause.

9.2. Recommendations

If we were to do the project again, we would have given ourselves the following advice:

- Use Spark for all stages of the project
- Start project earlier to avoid problems with overloaded cluster
- Do not write any commands directly in the terminal, always write in a re-runable script
- If possible, organize a separate cluster for testing purposes, and only deploy the tested version to the communal one

9.3. Table of Contributions

Project Tasks	Task Description	Anton Kudryavtsev	Anatoly Soldatov	Sofia Tkachenko	Anastasiia Shvets	Deliverables	Average hours spent
Load data to S3	Download files from nyc.gov website to Yandex Cloud Object Storage to ensure dataset can be accessed without issues with dynamically loaded links	100%	0%	0%	0%	Data loaded to S3	3
Write scripts for data loading	Write stage1.sh script which should: download the data from S3, create PostgreSQL table, convert all files to uniform	100%	0%	0%	0%	stage1.sh and related scripts	8.5

Project Tasks	Task Description	Anton Kudryavtsev	Anatoly Soldatov	Sofia Tkachenko	Anastasiia Shvets	Deliverables	Average hours spent
	schema, load files from local system to PostgreSQL, load files from PostgreSQL to Hive, and create index for loaded data						
Create database schema	Create schema to be used in PostgreSQL and Hive	50%	0%	50%	0%	create-table-psql.sql db.hql	3
Convert data to one schema	Ensure than all dataset can be loaded using single schema by creating script to convert each separate file to one schema	80%	0%	20%	0%	Script for dataset merging	0.5
Write scripts for EDA	Write stage2.sh script that would perform queries to gain valuable data insights	10%	0%	90%	0%	stage2.sh and related scripts	16
Put EDA graphs to dashboard	Add valuable insights and graphs from EDA stage to the final dashboard	20%	0%	80%	0%	Dashboard (EDA part)	2
Write scripts for ML	Create scripts that would train 3 different models with multiple sets of hyperparameters and find the best model	20%	80%	0%	0%	stage3.sh and related scripts	24
Create PDA scripts	Write scripts to perform queries which would obtain valuable insights from model predictions	30%	70%	0%	0%	stage3.sh and related scripts	1
Put PDA graphs to dashboard	Add valuable insights and graphs from PDA stage to final dashboard	0%	0%	100%	0%	Dashboard (Modeling part)	2
Organize dashboard	Ensure that dashboard tells a consistent story and aligns with project vision	15%	0%	80%	5%	Dashboard	10
Write report	Create a report which will be presented to TA based on provided report plan	10%	5%	10%	75%	Report	15
Update code according to pylint rules	Run pylint on written python scripts to find problems with code and fix those problems ensuring functionality does not change	0%	35%	0%	65%	-	1.5
Write README files	Create README files explaining how different scripts work	25%	5%	0%	75%	README files	5
Create presentation	Create presentation explaining project development, gained insights and obtained results	0%	0%	75%	25%	Presentation	10

Table 1: Table of contributions

Index of Figures

Index of Tables

Index of Listings