

Taxi tip prediction

Course: Big Data - IU S25

Team 23

Gleb Pavlov, Laith Nayal, Elina Pavlova, Aleksandra Kuzmich

May 8, 2025

Introduction

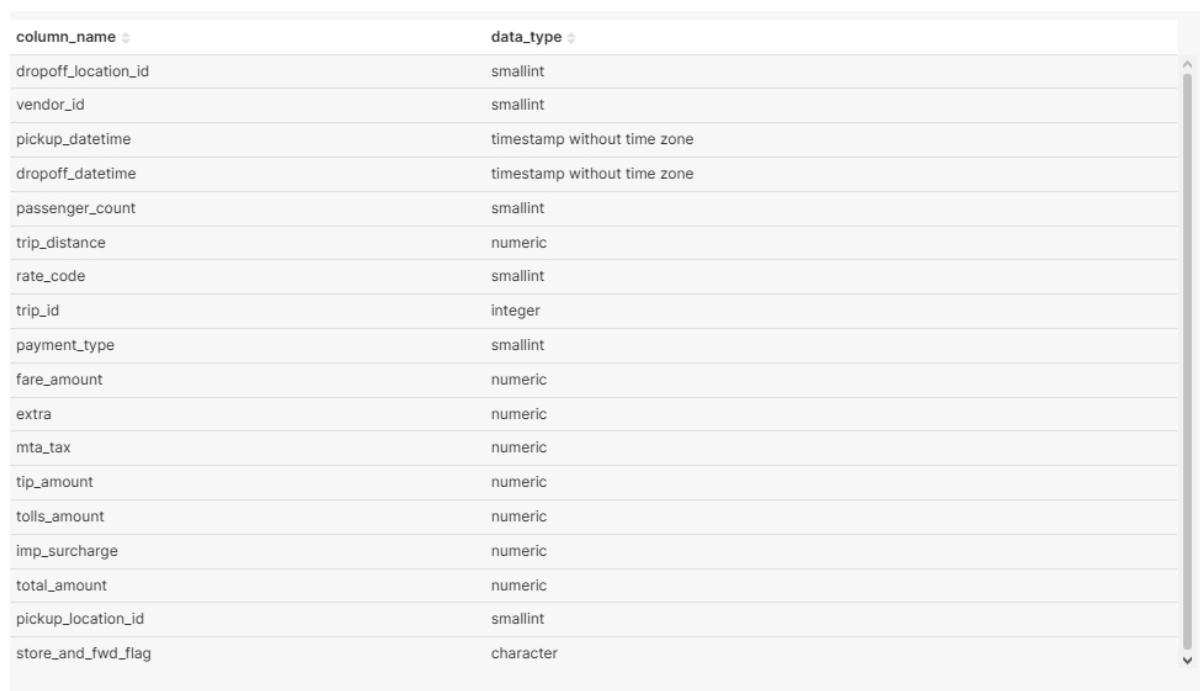
In today's world, organizations generate massive amounts of data daily. This project aimed to develop an end-to-end big data pipeline, designed to efficiently manage data ingestion, storage, analysis, and visualization. Our primary focus was to use the Hadoop system, including PostgreSQL, HDFS, Hive, Spark, and Apache Superset, to ensure robust data processing and analysis.

Business Objectives

The data pipeline begins with the ingestion of raw data from PostgreSQL into HDFS using Sqoop. The data were then organized into Hive tables, where we performed EDA, revealing key insights and patterns. To further extend our analysis, we utilized Spark MLlib to build and optimize machine learning models, transforming the data into predictive insights. Finally, Apache Superset was used to design an interactive dashboard, providing a clear visualization of the insights and model results.

Data Description

The dataset used in this project is a large-scale collection of NYC Yellow Taxi trip records, sampled to 10,000,000 records. It contains the following data described in the Fig. 1.



column_name	data_type
dropoff_location_id	smallint
vendor_id	smallint
pickup_datetime	timestamp without time zone
dropoff_datetime	timestamp without time zone
passenger_count	smallint
trip_distance	numeric
rate_code	smallint
trip_id	integer
payment_type	smallint
fare_amount	numeric
extra	numeric
mta_tax	numeric
tip_amount	numeric
tolls_amount	numeric
imp_surcharge	numeric
total_amount	numeric
pickup_location_id	smallint
store_and_fwd_flag	character

Figure 1: Dataset Schema

Data Characteristics

- **Total Columns:** 18, covering a range of data attributes
 - **Numeric**
 - * Passenger count, trip distance, fare amount, tip amount, total amount.

- **Categorical**
 - * Payment type, store-and-forward flag.
- **Temporal**
 - * Pickup and dropoff timestamps.
- **Geospatial**
 - * Pickup and dropoff location IDs.
- **Missing Values**
 - Certain records have null values in optional fields, such as *tip amount*.
 - Geospatial information is provided as IDs (pickup and dropoff location) rather than direct latitude/longitude coordinates.
- **Storage Format**
 - Data stored in Avro format with Snappy compression in HDFS. Avro was chosen for its efficient row-based storage and schema evolution support

Architecture of Data Pipeline

The project consist of 4 primary stages.

Stage I: Data Ingestion and Storage

- **Input:** Raw data file (taxi_trip_data.csv) from NYC Yellow Taxi Trip dataset.
- **Process:**
 - Data ingested into PostgreSQL for structured storage.
 - SQL scripts were used to create and populate tables.
 - Data exported to HDFS using Sqoop with Avro + Snappy compression.
- **Output:** Avro files stored in HDFS at /user/team23/project/warehouse/trips.

Stage II: Data Preparation (Hive Table Creation and EDA)

- **Input:** Avro files in HDFS from Stage I.
- **Process:**
 - Hive external table created from Avro files for efficient querying.
 - Data partitioned by pickup_date and bucketed by vendor_id.
 - Five exploratory data analysis queries performed using HiveQL.
- **Output:** EDA results stored as CSV files in HDFS and visualized using Apache Superset.

Stage III: Predictive Data Analytics (Spark ML)

- **Input:** Hive table containing preprocessed data from Stage II.
- **Process:**
 - Data loaded into Spark DataFrames.
 - Feature engineering and data preprocessing applied.
 - Two ML models trained (Linear Regression, Gradient Boosting Trees).
 - Model performance evaluated using RMSE and R2 metrics.
- **Output:** Model evaluation results (evaluation.csv) and model predictions (model1_predictions.csv, model2_predictions.csv).

Stage IV: Data Presentation (Apache Superset Dashboard)

- **Input:** EDA results (CSV files), ML model predictions (CSV files).
- **Process:**
 - Data imported into Apache Superset.
 - Interactive dashboard created with multiple charts.
- **Output:** Published dashboard accessible to stakeholders for data exploration.

Data Preparation

ER Diagram

Since our project focuses on a single table, the data model is straightforward. The trips table contains detailed trip-level records for NYC Yellow Taxi services.

Sample Data from the Database

These samples provide an overview of the type of data stored in the table.

trip_id	vendor_id	pickup_datetime	dropoff_datetime	passenger_count	trip_distance	rate_code	store_and_fwd_flag	payment_type	fare_amount
1	2	2018-03-29	2018-03-29	1	18.15	3	N		1
2	2	2018-03-29	2018-03-29	1	4.59	1	N		1
3	2	2018-03-29	2018-03-29	1	0.3	1	N		1
4	2	2018-03-29	2018-03-29	2	16.97	1	N		1
5	2	2018-03-29	2018-03-29	5	14.45	1	N		1
6	2	2018-03-29	2018-03-29	1	11.6	1	N		1
7	1	2018-03-29	2018-03-29	1	5.8	1	N		1
8	2	2018-03-29	2018-03-29	1	3.38	1	N		1
9	2	2018-03-29	2018-03-29	1	16.98	3	N		1
10	2	2018-03-29	2018-03-29	1	4.99	1	N		1

Figure 2: Dataset Sample

payment_type ↕	fare_amount ↕	extra ↕	mta_tax ↕	tip_amount ↕	tolls_amount ↕	imp_surcharge ↕	total_amount ↕	pickup_location_id ↕	dropoff_location_id ↕
1	70	0	0	16.16	10.5	0.3	96.96	161	1
1	25	0	0.5	5.16	0	0.3	30.96	13	230
1	3	0	0.5	0.78	0	0.3	4.56	231	231
1	49.5	0	0.5	5.61	5.76	0.3	61.67	231	138
1	45.5	0	0.5	10.41	5.76	0.3	62.47	87	138
1	42	0	0.5	14.57	5.76	0.3	63.13	68	138
1	24	0	0.5	4.95	0	0.3	29.75	100	87
1	25	0	0.5	5.16	0	0.3	30.96	144	161
1	85	0	0	15	12.5	0.3	112.8	87	1
1	22	1	0.5	4.76	0	0.3	28.56	13	161

Figure 3: Dataset Sample

Creating Hive Tables and Preparing Data for Analysis

Hive Database Creation.

A dedicated Hive database named team23_projectdb was created to manage the project data.

Creating an External Table.

An external Hive table named trips was created to store the NYC Yellow Taxi data in Avro format, leveraging Snappy compression for efficient storage.

Partitioning and Bucketing.

Initially, the table was attempted to be loaded without partitions, but failed due to environmental restrictions. By introducing partitioning on pickup_date and bucketing by vendor_id, we managed to move further. This highlighted the importance of designing table structures aligned with query patterns and importance of resource awareness.

A partitioned and bucketed table named trips_part_buck was created.

Validation and Data Verification.

The table structure and data were verified using simple SQL queries

Ensuring data integrity required careful verification using SQL queries. This was essential, as the raw data contained missing values and inconsistencies, particularly in fields like tip_amount and store_and_fwd_flag. These issues were identified and addressed through EDA in later stages.

Data Analysis

Analysis Results

The following stage targeted key metrics related to tipping behavior, trip characteristics, and fare distributions. Each query was created to show a specific correlation in the data in avg_tip and other fields being described in separate charts.

Charts and Insights

The results of the analysis were visualized in five key charts, each representing a data insight:

- **Chart 1: Average Tip by Pickup Month**

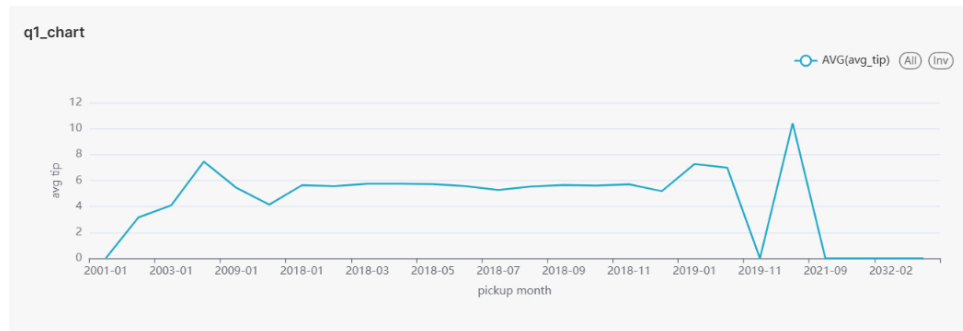


Figure 4: Average Tip by Pickup Month

The average tip amount varies noticeably across months and years, with prominent peaks around late 2018 and mid-2019. These deviations suggest underlying seasonal or external influences on passenger tipping behavior (Figure 4).

- **Chart 2: Average Tip by Hour of Day**

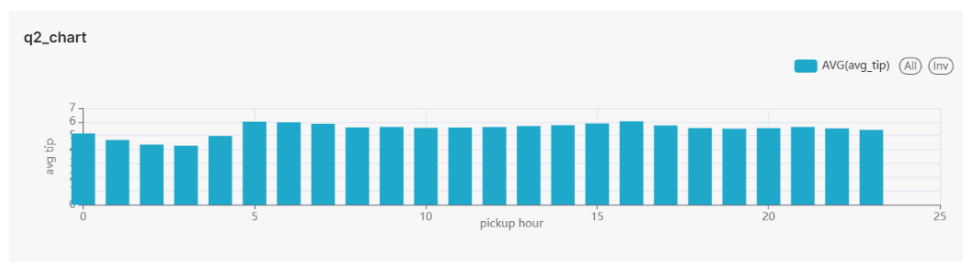


Figure 5: Average Tip by Hour of Day

Tips are generally lower during the night and early morning hours, with a peak around midday. This pattern aligns with the assumption that passengers may tip more generously during business hours, possibly due to business-related trips (Figure 5).

- **Chart 3: Average Tip by Route (Pickup & Dropoff IDs)**

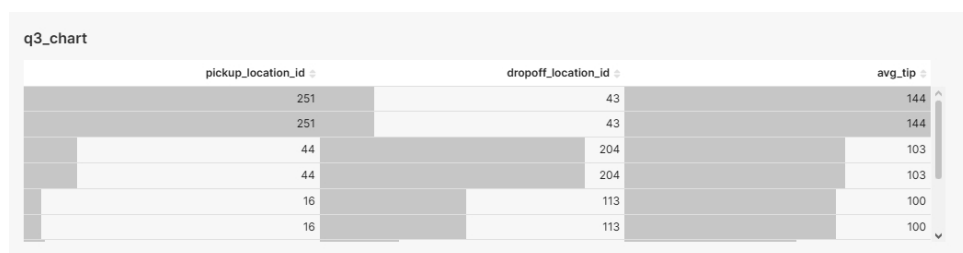


Figure 6: Average Tip by Route (Pickup & Dropoff IDs)

Certain routes receive exceptionally high tips (e.g., from 238 to 243 and from 48 to 142). This behavior may indicate popular tourist destinations where passengers are more likely to tip (Figure 6).

- **Chart 4: Average Tip by Trip Distance**

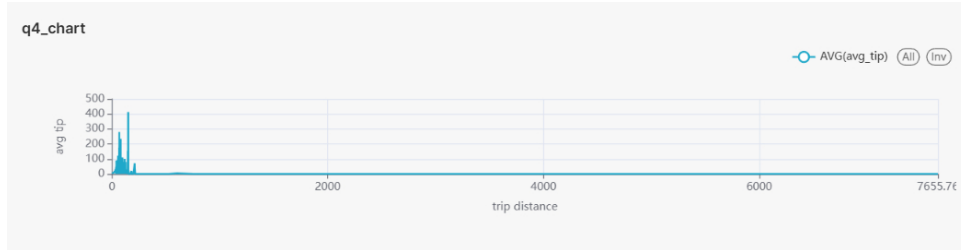


Figure 7: Average Tip by Trip Distance

Very short trips (under 1 mile) dominate the dataset, but the average tip value increases with trip distance, though outliers exist. Longer distances tend to correlate with higher average tips, as expected from fare proportions (Figure 7).

- **Chart 5: Average Tip by Fare Amount**

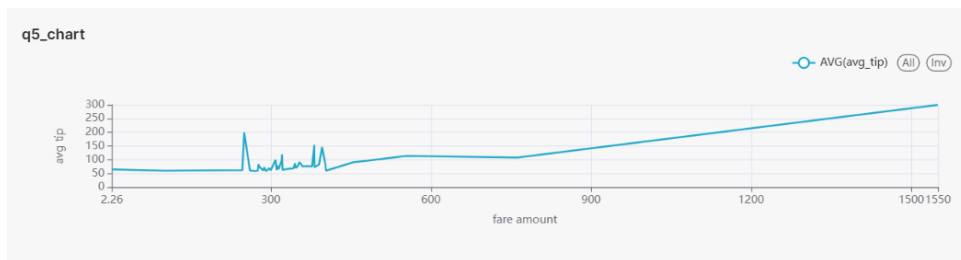


Figure 8: Average Tip by Fare Amount

There are major outliers with extremely high tips for high fare values, suggesting suspicious or anomalous entries. While most tipping behavior scales with fare, the outliers may indicate possible VIP customer behavior (Figure 8).

Interpretation

The analysis of tipping behavior revealed several important patterns that can be leveraged for ML model:

- **Seasonal Trends:** Understanding that certain months show higher average tips can help in feature engineering by incorporating seasonal indicators as predictive variables, improving model accuracy.
- **Time-Based Insights:** The clear difference in tipping behavior during night hours suggests that time-based features can be critical predictors in the model.
- **Location-Based Patterns:** Recognizing that some pickup and dropoff locations consistently receive higher tips can guide the model to use location IDs or zone clusters as categorical features, capturing the geographical influence on tipping.

- **Distance-Dependent Tipping:** The decrease in tip percentage on longer trips can be useful for creating interaction terms between distance and fare amount, allowing the model to account for the diminishing return on tips for longer distances.
- **Fare-Driven Generosity:** The positive correlation between fare amount and tip amount can be directly used as a feature in the model. Additionally, fare-to-tip ratios can be calculated to further capture tipping behavior.

ML modeling

Feature Selection:

Feature selection was guided by the Pearson correlation between each feature and the target variable. Features exhibiting a strong positive or negative correlation were retained due to their predictive relevance. Features with correlations near zero were dropped. For features with moderate correlations, we evaluated models with and without these features to determine their impact on performance.

Pipeline Design:

Categorical features were encoded using a `StringIndexer`, and numerical features were scaled using a standard scaler. A `VectorAssembler` was then used to combine all features into a single vector. This preprocessing was encapsulated into a pipeline to ensure consistency during model training and evaluation.

Data Splitting:

The dataset was divided into 70% for training and 30% for testing.

Models Used:

We trained two models—Linear Regression and Gradient-Boosted Tree (GBT) Regressor. Hyperparameter tuning was performed to identify the best configurations.

Table 1: Best Hyperparameter Values

Model	Hyperparameter	Best Value
Linear Regression	Regularization Param (<code>regParam</code>)	0.1
	Elastic Net Param (<code>elasticNetParam</code>)	1
GBT Regressor	Step Size (<code>stepSize</code>)	0.1
	Max Depth (<code>maxDepth</code>)	5

Table 2: Final Model Evaluation on Test Set

Model	RMSE	R ²
Linear Regression	3.6681455890473145	0.40207776140191664
GBT Regressor	2.552106406030934	0.7310987322461698

Data Presentation

The analysis results of our project are presented in a dashboard in Apache Superset. All data insights derived from the EDA stage have already been described in detail in the Data Analysis section of this report. This includes the explanations, interpretations, and the significance of each EDA chart.

In this section, we focus on the Machine Learning results, which are presented through additional charts. These charts showcase the performance of our ML models, feature importance, and the relationship between input features and predicted outcomes.

Analysis Results

In the machine learning stage of our project, we focused on building and optimizing predictive models using the Spark MLlib. Our objective was to predict the tip amount based on trip characteristics such as distance, fare amount, and other relevant features. This section presents the results of our model training, including feature analysis, model performance, and hyperparameter optimization.

Charts and Insights

The results of the ML analysis are visualized in four key charts, each highlighting a different aspect of model performance:

- **Chart 1: Pearson Correlation**

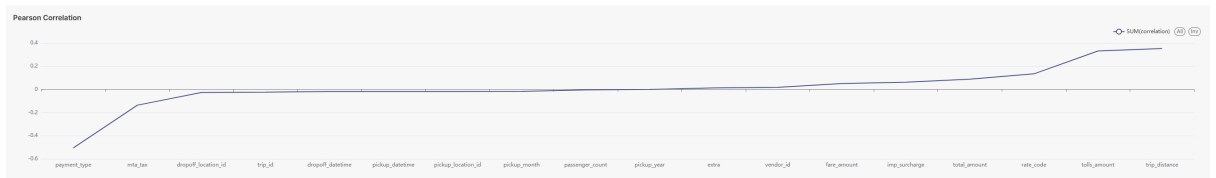


Figure 9: Pearson Correlation between Features and Target

This chart displays the Pearson correlation values between each input feature and the target variable (tip amount). The most positively correlated feature is total_amount, while payment_type shows a negative correlation. This analysis helped us understand which features have the strongest impact on predictions (Figure 9).

- **Chart 2: Feature Statistics**

metric	mean_value	stddev_value	min_value	max_value
trip_id	5000000.5	2886751.490285687	1	10000000
vendor_id	1.6143277	0.5146575584508939	1	4
pickup_datetime	1529866760502.2021	9095116203.217701	978298824000	2635842333000
dropoff_datetime	1529866807528.6384	90951846687.51886	978298850000	2635842333000
passenger_count	1.6029494	1.2457816727809203	0	9
trip_distance	8.848208209999992	5.882028461543538	0	7655.76
rate_code	1.2012385	1.2507334338503844	1	99
payment_type	1.1892995	0.43398757831794127	1	5
fare_amount	31.652551951000152	160.60113941564482	-800	398480.05
extra	0.33837806500000006	0.5512911467900136	-80	84
mta_tax	0.481928941	0.12072823130547782	-0.5	150
tip_amount	5.598526988998746	4.840296207475438	-322.42	496
tolls_amount	2.1378179970021194	3.750309296286839	-52.5	918.25
imp_surcharge	0.29788313800085975	0.03406027438649299	-0.3	1
total_amount	40.51607352905553	161.16336870778715	-800.3	398521.96
pickup_location_id	153.5631899	60.76455590224333	1	265
dropoff_location_id	148.1428256	75.74852785257227	1	265
pickup_year	2017.9998715	0.0389035170530679	2001	2053
pickup_month	6.2701029	3.423858140170408	1	12

Figure 10: Descriptive Statistics of Features

This table provides an overview of the key statistics for each feature, including mean, standard deviation, minimum, and maximum values. Understanding the distribution of feature values is critical for effective model training and performance tuning (Figure 10).

• Chart 3: Hyperparameter Optimization

model_type	model_version	rmse	r_squared	optimization_status
linear_regression	v1_baseline	3.6724482908056566	0.4254532595381808	before optimization
linear_regression	v2_optimized	3.66814558905473145	0.402077761940191664	after optimization
gbt regressor	v1_baseline	2.576230415629213	0.7141985849933504	before optimization
gbt regressor	v2_optimized	2.552106406030924	0.7310987323461698	after optimization

Figure 11: Hyperparameter Optimization Results

This chart illustrates the performance of our models before and after hyperparameter tuning. The results show clear improvements in model performance (reduced RMSE and increased R^2) after optimization, especially for the Gradient Boosting Regressor model (Figure 11).

• Chart 4: Model Evaluation

Evaluation		
model	rmse	r2
GBRegressor(Metric(GBRegressor_5018187765ee))	2.4930483132587285	0.7319154291488008
LinearRegression(Metric(LinearRegression_40320359a0))	2.7058385976986467	0.4188109102072488

Figure 12: Final Model Performance Metrics

This table presents the final evaluation results for the two best-performing models: Gradient Boosting Trees (GBT) and Linear Regression. The GBT model outperformed Linear Regression in both RMSE and R^2 metrics (Figure 12).

Conclusion

In this project, we successfully developed an end-to-end big data pipeline capable of efficiently ingesting, storing, analyzing, and visualizing large-scale data. Starting from raw

NYC Yellow Taxi trip data, we used a series of big data technologies, including PostgreSQL for relational storage, HDFS for distributed file storage, Hive for data warehousing, Spark MLlib for machine learning, and Apache Superset for dashboard visualization. Our analysis provided valuable insights into tipping behavior, while our ML models enabled accurate predictions of trip-related outcomes. This project demonstrates the value of an integrated big data architecture in transforming raw data into actionable insights.

Summary of the Report

This report provides a detailed account of our project, covering the following key aspects:

- **Data Ingestion and Storage:** Raw data was collected and ingested into PostgreSQL, then transferred to HDFS in Avro format with Snappy compression for efficient storage.
- **Data Preparation:** Hive tables were created and optimized through partitioning and bucketing, enabling efficient query performance.
- **Data Analysis:** A comprehensive EDA was performed, resulting in actionable insights into tipping behavior, visualized through Apache Superset.
- **Machine Learning:** Predictive models were developed using Spark MLlib, evaluated for performance, and visualized in the dashboard.
- **Data Presentation:** Insights and model results were presented in an Apache Superset dashboard for intuitive exploration.

Reflections on Own Work

Working on this project provided hands-on experience with big data, highlighting the importance of efficient data storage and processing. Partitioning and bucketing in Hive proved essential for query performance, while building machine learning models with Spark MLlib taught us about feature selection, model complexity, and hyperparameter tuning. Designing the Apache Superset dashboard transformed raw data into clear insights, emphasizing the value of effective data visualization. This project was an interesting path of technical challenges, teamwork, and turning complex data into actionable insights.

Challenges and Difficulties

- **Data Size:** Handling 10 million records required efficient data storage and processing techniques, demanding careful optimization of data formats and compression methods.
- **Hive Table Optimization:** Partitioning and bucketing introduced complexities, requiring a balance between query performance and storage efficiency.
- **Modeling Complexities:** Feature selection and hyperparameter tuning for machine learning models were critical yet challenging, requiring iterative experimentation for optimal performance.

- **Error Handling and Debugging:** Debugging data ingestion, HiveQL, and Spark scripts in a distributed environment was time-consuming, highlighting the need for testing and troubleshooting.

Recommendations

- **Advanced Model Tuning:** Explore advanced hyperparameter optimization methods, such as Bayesian optimization, to enhance model performance.
- **Improved Dashboard Design:** Enrich the dashboard with interactive elements, providing a more intuitive data exploration experience for users.
- **Regular Monitoring:** Establish automated monitoring for data quality and model performance, ensuring continuous reliability.
- **Future Expansion:** Explore real-time data processing capabilities using Apache Kafka and Spark Streaming for dynamic data analysis.

Contributions of Each Team Member

Project Tasks	Task Description	Aleksandra Kuzmich	Gleb Pavlov	Laith Nayal	Elina Pavlova	Deliverables	Average Hours Spent
Stage I	Data collection, database setup, HDFS import	90	5	0	5	Data, DB Schema, SQL	8
Stage II	Hive table creation, data preparation, EDA	5	80	10	5	Hive tables, EDA Results	10
Stage III	ML model building, hyperparameter tuning	0	10	85	5	ML Models, Evaluation	12
Stage IV	Dashboard creation, final report, presentation	5	5	5	85	Dashboard, Report	6

Figure 13: Table of contributions