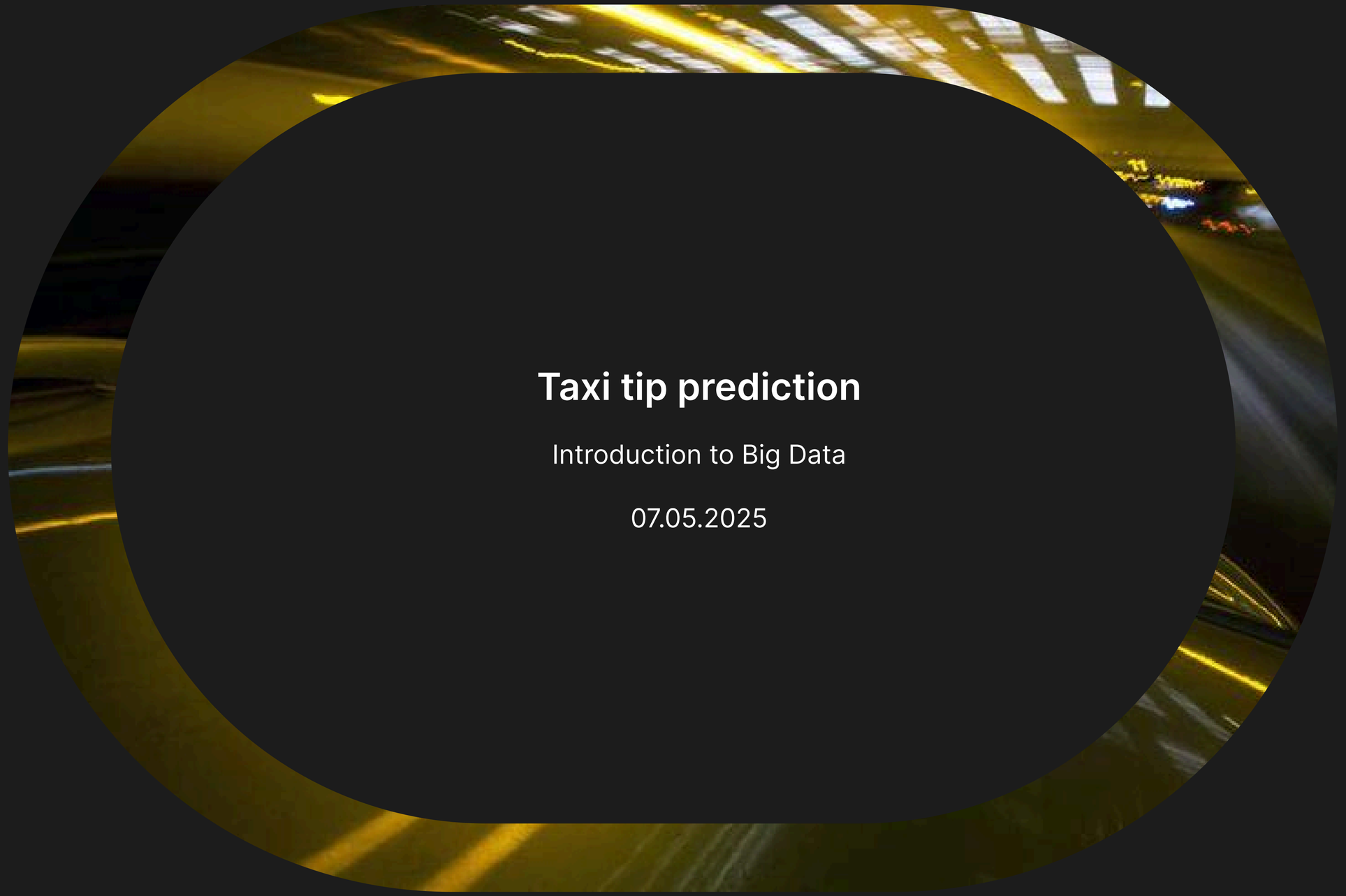


Taxi tip prediction

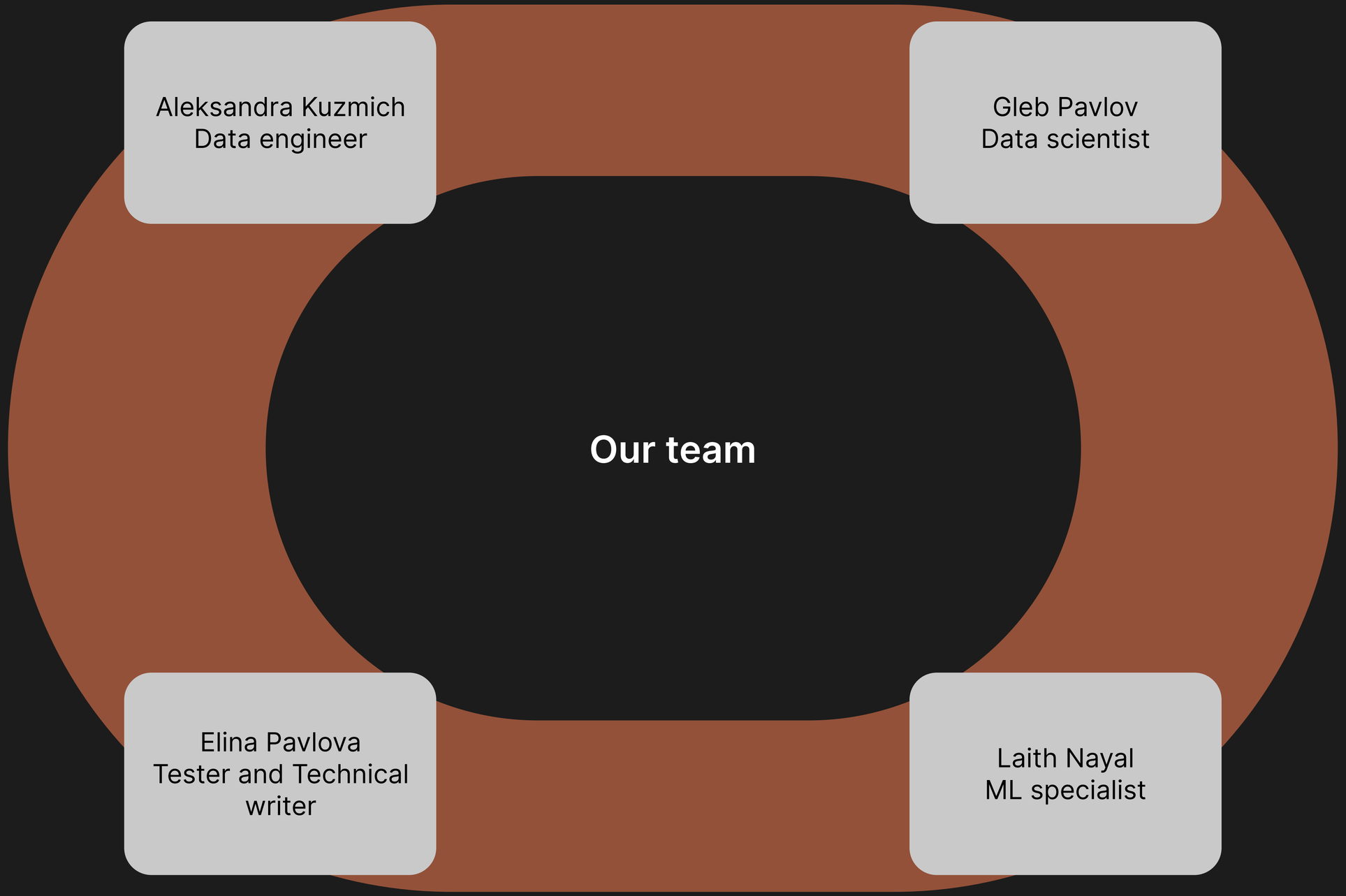
Introduction to Big Data

07.05.2025

Introduction

Goal:
Taxis are highly popular nowadays, and their service should be convenient not only for passengers but also for drivers. The goal of our project is to predict the tips a taxi driver will receive, helping them make decisions about which rides to accept.

Project Aim:
To create a scalable big data pipeline for predicting the tip amount on NYC taxi trips.


Aleksandra Kuzmich
Data engineer
Gleb Pavlov
Data scientist
Our team
Elina Pavlova
Tester and Technical writer
Laith Nayal
ML specialist

Stage I – Pipeline Overview & Dataset

Goal:

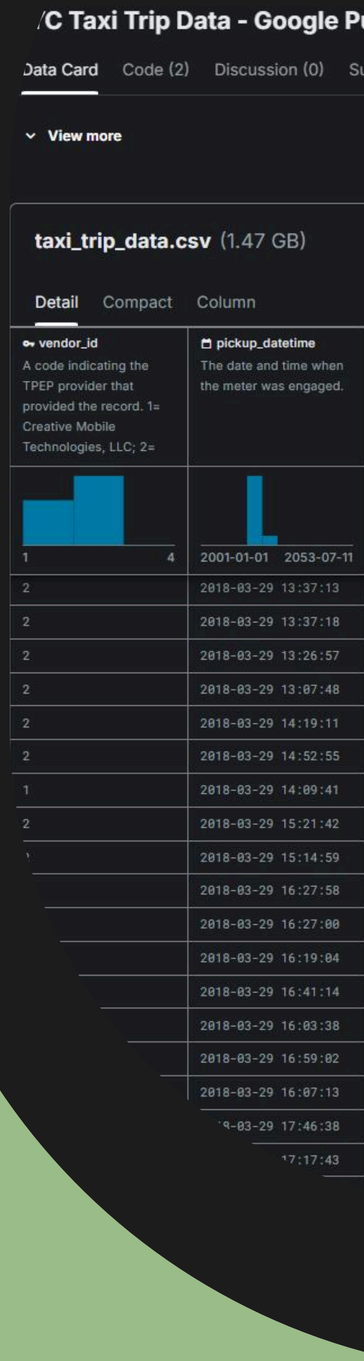
Build a batch ingestion pipeline from CSV to HDFS using PostgreSQL and Sqoop.

Pipeline:

CSV → PostgreSQL → HDFS (Avro + Snappy)

Dataset:

- Source: taxi_trip_data.csv – NYC Yellow Taxi trip records
- Fields:
  - Pickup & dropoff timestamps
  - Passenger count, trip distance
  - Fare, tip, taxes, total amount
  - Payment type, location IDs



PostgreSQL Storage & Export to HDFS

PostgreSQL Setup:

- Created trips table in team23_projectdb with proper data types
- Imported data using Python with COPY FROM STDIN for efficient loading
- Scripts are reusable — existing tables are dropped and recreated
- Verified import with SQL row counts and sample checks

Sqoop Export to HDFS:

- Exported PostgreSQL data to HDFS using Sqoop CLI
- Format: Avro with Snappy compression
- HDFS Target: /user/team23/project/warehouse/trips
- Automated via to_hdfs.sh

Schema Handling, Format Choice & Automation

Avro Schema Handling:

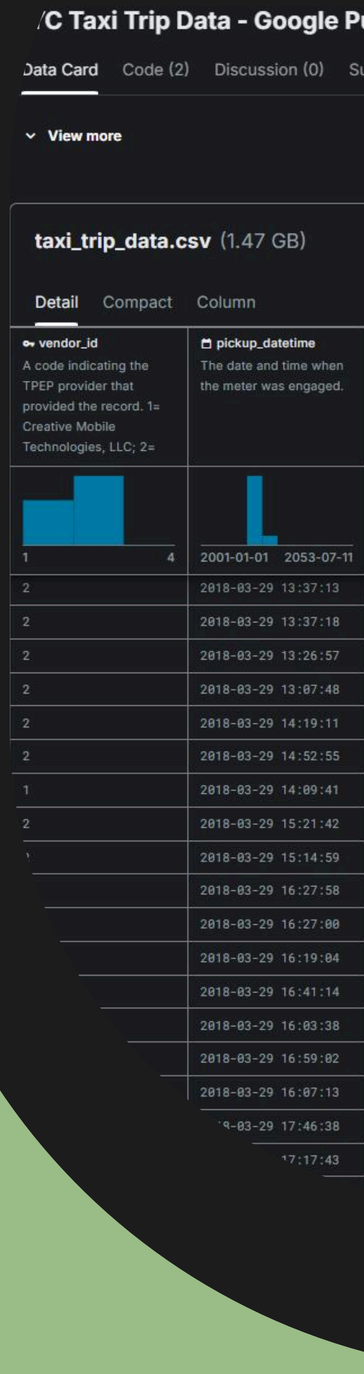
- Sqoop generates .avsc schema file during export
- Schema moved to output/ directory for future use
- Ensures consistent, reusable schema for downstream processing

Format & Compression:

- We used Avro because it stores full records efficiently and includes the schema, which helps with future data processing.
- Snappy helps to work with large datasets.

Scripts:

- stage1.sh: orchestrates entire ingestion pipeline
- data_storage.sh: sets up PostgreSQL and loads data
- to_hdfs.sh: handles Sqoop export and schema file organization



Stage II – Hive Integration Overview

Objective:

Enhance the data pipeline by integrating Hive for scalable, SQL-like analytics on HDFS data.

Data Preparation:

- Created Hive database: team23_projectdb
- Established separation:
  - Sqoop import path: /warehouse
  - Hive processing path: /hive/warehouse

Tables Created:

- trips: External Avro-based table linked to imported data
- trips_part_buck: Partitioned by month and bucketed by vendor_id - for loading in manageable chunks

Partitioning, Bucketing & Exploratory Analysis

Optimization Techniques:

- Partitioning: By pickup_date (from timestamp) for faster query filtering
- Bucketing: By vendor_id for even data distribution
- Dynamic Partitioning:
  - Enabled nonstrict mode so Hive could automatically create partitions during insert without manual definition.

Exploratory Data Analysis (EDA):

- Q1: Average tip by month
- Q2: Average tip by hour
- Q3: Average tip by pickup/dropoff zones
- Q4: Average tip by trip distance
- Q5: Average tip by fare amount

Results Handling:

- Saved to Hive tables: q1_results - q5_results
- Exported to CSV: output/q1.csv - output/q5.csv
- Visualized in Apache Superset

Automation, Outputs & Conclusion

Automation via stage2.sh:

- Runs Hive scripts for table creation and data insertion
- Executes all EDA queries
- Cleans HDFS paths and exports result CSVs

Output Summary:

- Hive Tables: trips, trips_part_buck, q1_results - q5_results
- CSV Exports: q1.csv - q5.csv
- Superset Charts: q1.jpg - q5.jpg

Stage III - Spark ML Setup & Pipeline

Goal:

Predict tip_amount using distributed ML with Spark on YARN

Data Source:

Hive table trips_part_buck (partitioned by month, bucketed by vendor_id)

Train/Test Split:

- 60% training / 40% testing
- Saved as JSON to HDFS and exported locally

Preprocessing Steps:

- Extracted time features (month, hour) from timestamps
- One-hot encoded categorical variables
- Assembled all features into vector format

Models, Tuning & Evaluation

Models Trained:

- Linear Regression
- Gradient Boosted Trees (GBT)

Hyperparameter Tuning:

- Grid Search + 3-Fold Cross-Validation
- Linear Regression: regParam, aggregationDepth
- GBT: maxDepth, maxIter, stepSize

Evaluation Metrics:

- RMSE
- R^2

model	RMSE	R2
LinearRegressionModel: uid=LinearRegression_eb028268bea9, numFeatures=15	3.6681455890473145	0.40207776140191664
GBTRegressionModel: uid=GBTRegressor_9b18197766ee, numTrees=20, numFeatures=15	2.552106406030934	0.7310987322461698

GBT significantly outperformed Linear Regression

Outputs & Automation

Artifacts Produced:

- Trained models: models/model1/, model2/
- Predictions: output/model1_predictions.csv, model2_predictions.csv
- Evaluation summary: output/evaluation.csv

Automation:

- stage3.sh runs the full ML pipeline via spark-submit model.py
- Handles data split, training, evaluation, and export
- Ensures reproducibility and clean execution

Conclusion:

- Distributed ML pipeline completed
- GBT selected as best model

Stage IV - Dashboard Creation

Goal:

Provide a clear summary of the dataset and modeling results — combining raw data, EDA, and ML evaluation in a format accessible to both technical and non-technical viewers.

Structure:

- Dataset Overview: Intro, source, and data context
- Schema Section: Full table schema with types and definitions
- Sample Data: Snapshot of raw records for validation and exploration
- Insights & Modeling: Visual analysis and model comparison for tip prediction

Data Description & Insights

Dataset Section:

- Includes dataset origin (NYC Yellow Taxi Trip Data), schema documentation, and example records
- Data types (e.g., timestamps, numerics) and quality notes on missing or anomalous values (e.g., zero fares or NULL trip distances)

Exploratory Insights:

- Insight 1: Avg tip by pickup month — shows seasonal trends, economic shifts
- Insight 2: Avg tip by hour — peak around mid-morning, lower at night
- Insight 3: Avg tip by route (pickup/dropoff IDs) — certain zones yield higher tips
- Insight 4: Avg tip by trip distance — tips increase with distance, short trips dominate volume
- Insight 5: Avg tip by fare amount — tips rise with fare, but outliers exist

Data Description & Insights

Modeling Section:

- Models Evaluated: GBT Regressor and Linear Regression
- Feature set includes engineered time fields, one-hot encodings
- Predictions generated and saved for both models on test data
- Feature selection guided by Pearson correlation

Hive & Visualization:

- Superset charts built on top of Hive data include:
  - Feature extraction characteristics
  - Hyperparameter tuning results
  - Predictions for each model
  - Evaluation scores per model
  - Visual model comparison

Thank you!