Ori Ben-Zaken
311492110

Liz Aharonian
316584960

בס"ד

Final Report

Demand per time interval and area for Uber Taxi Ride

Project Design:

Goal: demand per time interval and area for Uber Taxi Ride.

In cases of very high demand, fares may increase to help ensure those who need a ride can get one. Some riders will choose to pay, while some will choose to wait.

Our goal is to predict demand per time interval for Uber Taxi Ride, using ML and big data processing tools.

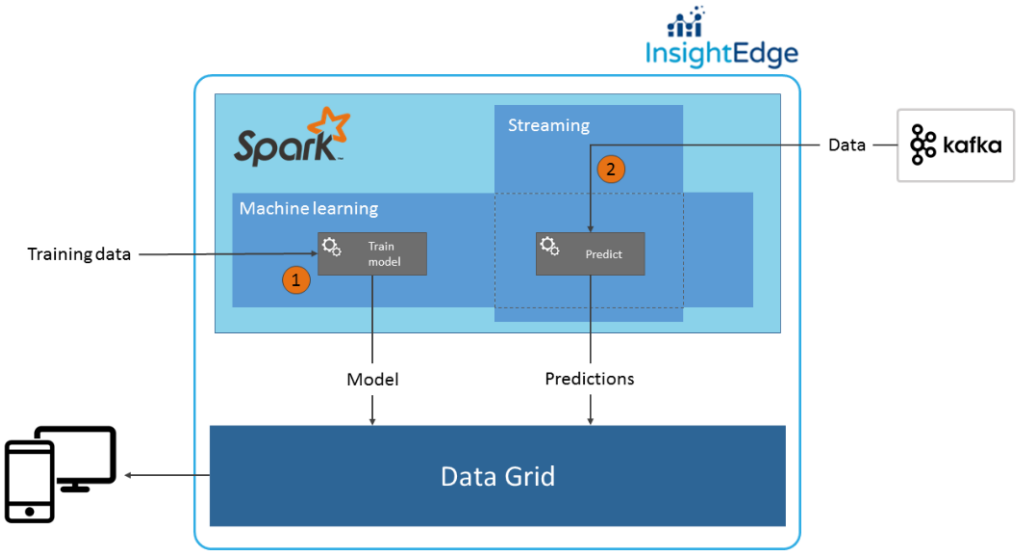
The demand parameter is calculated according to time intervals and area, this gives two major advantages to our predicting model:

1. Help the model's users to know which area they should give service.
2. Using a simple function based on the demand to predict and calculate the fare amount.

Technologies:

- Machine Learning (Apache Spark and ML model – using sklearn library).
- Interactive Development (Apache Zeppelin).
- Prediction of real time records (Kafka streaming).
- BI (Tableau).
- Packaging the model components using Docker.

Architecture:



Project's components and milestones:

Milestone #1 : Find relevant data

- After examining different taxi trips datasets we decided to take our data from 2014_Uber_Data, can be found here.
- Our dataset contains records of ~360K uber taxis trips in NYC. Each record contains the following details: pickup date and time, pickup latitude, pickup longitude and region code.

Milestone #2: Data cleansing (Semester A)

- We performed full data cleansing that includes: removing records with Null values and removing columns that more than half to their values are Null or empty. In our case, no column was removed.
- Our model target is to predict the demand for taxi per time interval and area. In order to do so we added the "demand" column. We counted all of the taxi trips that occurred in a certain region and certain 10-minutes time interval. The result was added to each of relevant trip records.
- Added more features in order to help the prediction. For each record we added the features: is_Holiday, is_Weekend according to the trip date.
- In addition, in order to prepare the data for machine learning model we converted all the string values to numerical values.

Milestone #3: Training ML model (Semester A + B)

- We chose randomly 50K trip records as our dataset for the model. Splitting of course the dataset to training set and validation set.
- We tried many regression models, reached to the best results with Random Forest Regression model.
- The model is a spark model which can perform it's work and computation in distributed system.

Milestone #4: Data streaming (Semester B)

- Using Apache Kafka in order to stream the data in real-time.
- Kafka aims to provide a unified, high-throughput, low-latency platform for handling real-time data feeds, making it highly valuable for enterprise infrastructures to process streaming data.

Milestone #5: Running the model (Semester B)

- Using trained serialized model in order to predict demand per time interval and area on new examples which hasn't been seen by the model.

Milestone #6: Visualization of the data - BI (Semester B)

- Using Tableau in order to present the model results and more analytics on the data.
- Tableau is a BI tool uses to ask new questions on our data, spot trends, identify opportunities, and make data-driven decisions with confidence.

Milestone #7: packaging the model (Semester B)

- Package our model and all its dependencies using Docker, a computer program that performs operating-system-level virtualization.
- We are going to use it, so that our project would not be depend on specific operating system.

Technologies:

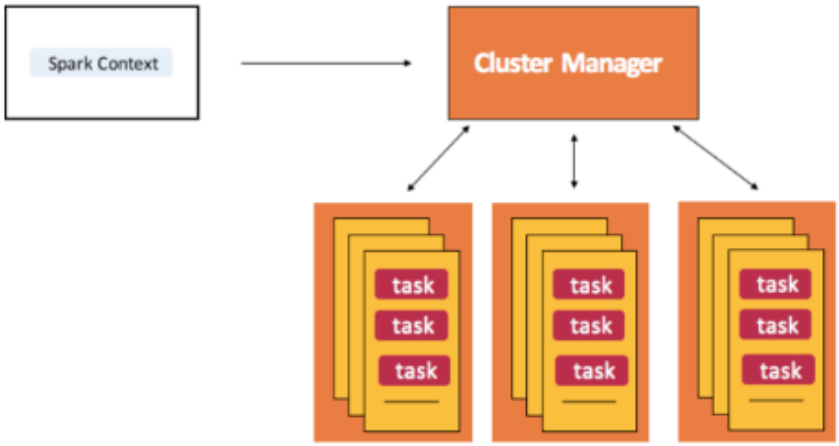
For performing real time predictions we used **Spark** Streaming combined with **Apache Kafka** which simulates endless and continuous data flow. For the hardest part, prediction, we used Spark Machine Learning and decision tree algorithm. Streamed data is processed by a decision tree model and results are saved into InsightEdge data grid for future usage.

Apache Spark:

Our machine learning model is spark based, Apache Spark is known as a fast, easy-to-use and general engine for big data processing that has built-in modules for streaming, SQL, Machine Learning (ML) and graph processing. This technology enables In-Memory computation and Parallel-Processing using APIs for Java, Python, and Scala.

How spark works?

We can submit jobs to run on Spark. On a high level, when we submit a job, Spark creates an operator graph from the code, submits it to the scheduler. There, operators are divided into stages of tasks, that correspond to some partition of the input data.



Spark workers have executors of tasks

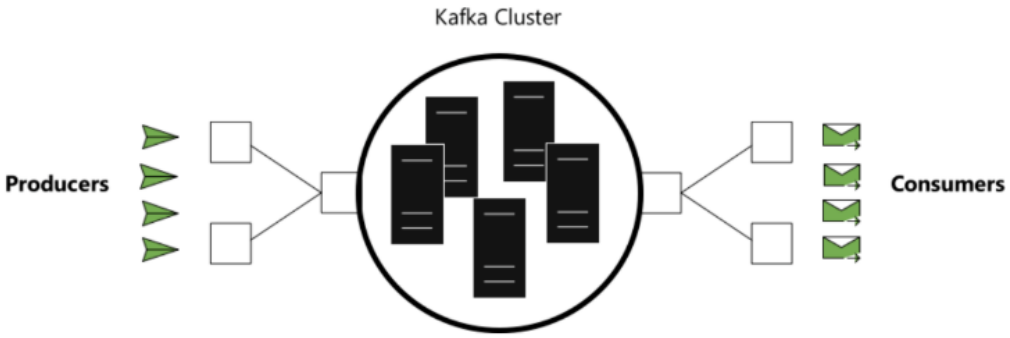
Apache Kafka:

we use apache kafka in order to stream new records and perform prediction on them.
Apache Kafka is an open-source streaming system.
Kafka is used for building real-time streaming data pipelines that reliably get data between many independent systems or applications. Kafka runs as a cluster on one or more servers that can span multiple datacenters. Those servers are usually called brokers.

The core abstraction Kafka provides for a stream of records — is the topic.

The topic is composed of partition, each partition contains sequence of records.

The kafka producer is in charge of publish data to the topic. This data is then consumed by the kafka consumer. Consumers can subscribe to topics and receive messages. Consumers can act as independent consumers or be a part of some consumer group.



InsightEdge:

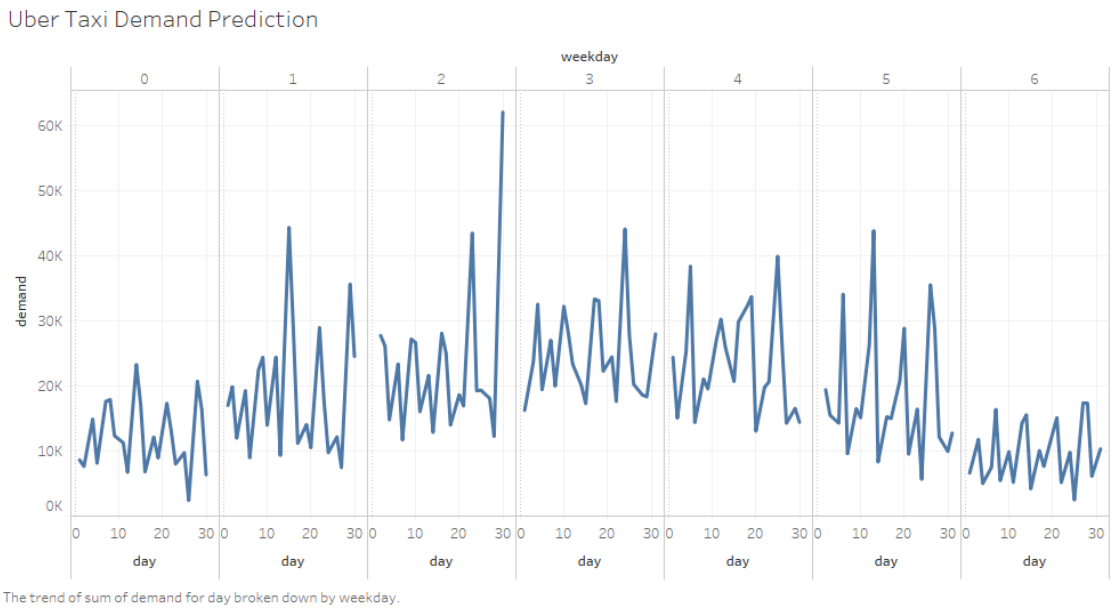
InsightEdge powers real-time analytics on streaming data enriched with historical context to help address time-sensitive decisions.
The software platform contains all the necessary frameworks for scalable data-driven solutions including SQL, Spark, streaming, machine learning and deep learning.
InsightEdge enables applications leverage faster and smarter insights from machine learning models running on any data source whether structured, unstructured or semi-structured while seamlessly accessing historical data from data lakes or Amazon S3.
In-memory performance offers ultra-low latency, high-throughput transaction and stream processing, and co-location of applications and analytics to act on time-critical data in real-time.
We use insight edge in order to save the train data and new records prediction on grid.
In addition, we trained our spark model on Zeppelin notebook supported by I9E. The trained data was loaded from grid and loaded to RAM – that was helpful to improve the running time of the train.

Tableau:

Tableau is a BI tool uses to ask new questions on our data, spot trends, identify opportunities, and make data-driven decisions with confidence.

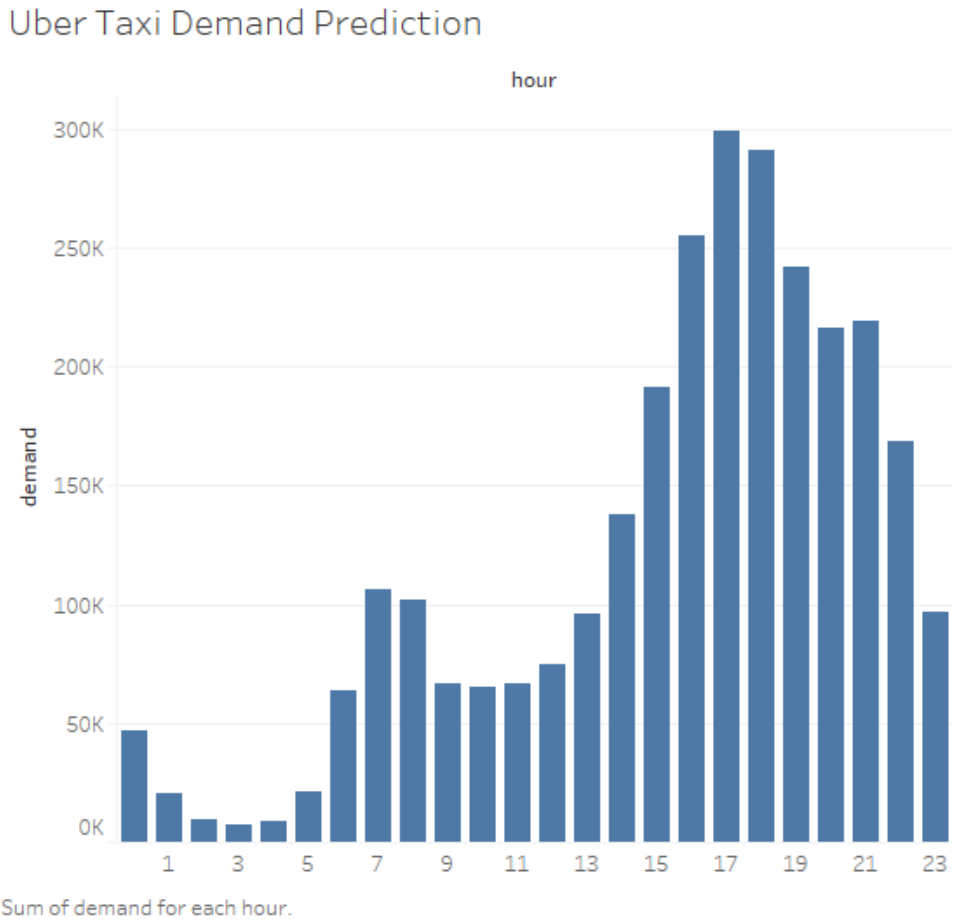
Some analytics:

1. The connection between weekday and day to demand rate:



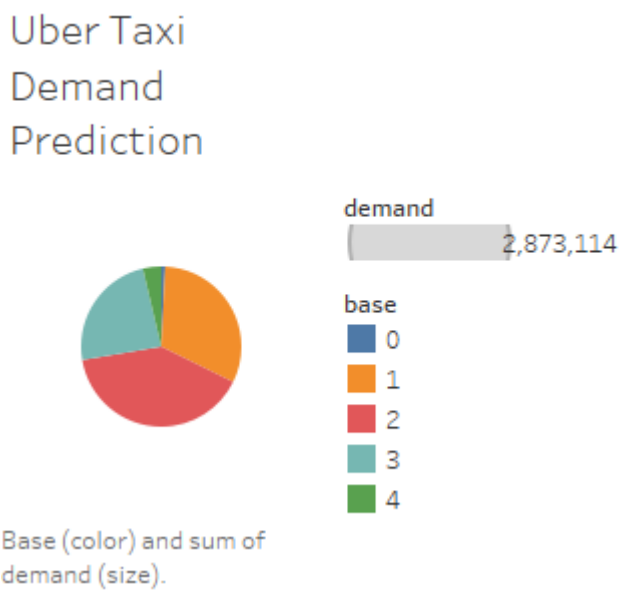
As we can see, the highest demand rate is on the middle of the week.

2. The connection between hour and demand rate:



As we can see, the highest demand rate is at the evening hours.

3. The connection between area and demand rate:



In this graph we can see the demand per area.
As we can see the most wanted area is the area with code 2.

All of these analytics can help Uber drivers - to know which area they should give service. In addition, we are calculating a simple function based on the demand to predict and calculate the fare amount, this could help Uber users to know when they should order a taxi.

Docker:

Docker is a software platform designed to make it easier to create, deploy, and run applications by using containers. It allows developers to package up an application with all the parts it needs in a container, and then ship it out as one package.

Packaging our model and all its dependencies using Docker, a computer program that performs operating-system-level virtualization, helps us make our project not be depend on specific operating system.

Challenges:

The main challenge was finding relevant data in order to train the model.

At first, we searched for uber rides data on Google dataset search, we found there only yellow taxies rides data. We also searched for relevant data in Kaggle. Eventually, we found the data in Uber website.

In addition, another challenge we dealt with doing this project was learning the new technologies we used. We spent lot of hours learning their abilities and ways to use them.

Furthermore, we couldn't run the model on our computer because it's needed lots of in memory computations, so Gigaspaces bought Amazon EC2 remote machine so that we could load and run the model on extended amount of data.