**Project Proposal**

* High Level requirements (ca. 500 wörter, motivation)
* Implementation requirements: 1) analytics 2) architecture
* Total: Up to 2000 words

**Preisvergleich zwischen grün und gelb wird nicht gehen -> sie haben die gleichen Tarife, unterscheiden sich lediglich dadurch, dass sie verschiedene Gebiete anfahren..**

<https://www.quora.com/What-is-the-difference-between-Green-Cabs-and-Yellow-Cabs>

<https://www.quora.com/Are-green-cabs-cheaper-than-yellow-cabs>

**Hier hat einer sehr viel mit dem Datensatz gemacht und wir könnten vllt was abschauen:**

<http://toddwschneider.com/posts/analyzing-1-1-billion-nyc-taxi-and-uber-trips-with-a-vengeance/>

Alternativ: Mit Uber fares vergleichen anhand dieser Hochrechnungen?

<http://uberestimate.com/prices/New-York-City/>

Auch einfach offen lassen und auf diese Seite verweisen, damit der Kunde selbst vergleicht wäre möglich: <https://www.uber.com/de/fare-estimate/>

Können ja sagen „das müsste man noch im nächsten Schritt implementieren, um die App zu verbessern“

**Motivation and goal:**

In big cities such as New York one is overwhelmed by the traffic and just happy to get safely and fast from A to B. Especially when it comes to hailing a cab there are other worries than comparing prices. Still, there are big differences in fares which the consumer might want to take into account when deciding about if he is going to travel by a NYC Cab, Uber or just may take the metro.

Calculating the cab fare in advance might become difficult. It is calculated per mile, but also depends on the time travelled which can vary a lot in NYC traffic. Additionally, extra fees are charged for rides at night and during rush hours. …

The price can easily be calculated retrospectively (the composition of the price will be presented in the analytics part and can be looked up here: <https://de.wikipedia.org/wiki/New_York_City_Taxi_Cabs#Boro-Taxis> ), but a prediction turns out to be difficult due to uncertainty of travelling time and distance. Therefore we are using past data to create a predictive price model.

We use the NYC taxi dataset in order to create an application for the consumer, which will enable him to foresee the NYC Cab price for a future ride. The application takes the time of the day and the distance the consumer wants to travel and forecasts the prices of the New York City company. …

High Level requirements:

A dataset comprising travel data

Implementation requirements:

1. Analytics

Use scikit learn, Python’s machine learning library for clustering and regression analysis

Predicting the taxi price is a regression problem (supervised learning problem). The independent variables to take into account: pick up time, pick up neighborhood, drop off neighborhood + dummy variables for the optional premiums. The regression is most likely going to require previous clustering of drop-off and pick-up coordinates into neighborhoods (clustering algorithm, unsupervised learning problem).

The taxi price is calculated on the basis of several variables. Basis price 2,60 $ + 0,55$ NY state tax per ride + 0,80 $ for a night shift premium (20:00h-6:00h) + 1,00 $ rush hour premium (16:00h-20:00h on weekdays) which will be covered by dummy variables. Of course the price also depends on the miles driven (0,60$ per 1/5 mile) and in case of driving slow or standing in a traffic jamb 0,50$ per 60 seconds. To account for this we will need to regress on the pick-up time and the pick-up and drop-off neighborhoods as this will give us a clue about the typical traffic conditions between the neighborhoods at different times of the day.

Y = 3,15 + 0,8d1 + 1d2 + 0,6\*5x1 + 0,5\*x2

X1 : Average distance between two clusters in miles

X2: Average waiting time due to traffic between to clusters

**bX1X2 – x1 pick up cluster x2 drop off cluster – interaction effect based on distance and time**

Calculate parameters for the model which will account well for the distance and time travelled. For creating the model we will use the dataset 2010 – 2012, this will be our training data. Data from 2013 will serve as test data, so we see if our predictions by the first model were accurate. Include 2013 and 2014 data into the model.

Stream 2015-2017 data and adjust the parameters at the end of day as new data is created.

1. Architecture

As the given data source is not a data stream we will have to simulate just this by ingesting it part by part within a prespecified time frame (maybe even simulate a real data stream with timestamps from the previous year) into our data pipeline. The ingestion will be done with flume.

* Use the datetime package to access the current time and create the timestamps for streaming simulation
* 1) The ingestion will be done with Apache Flume, the distributed Hadoop service for ingesting streaming data. As we only have data from one source and in one single format this should be relatively easy, so we refrain from using more complicated services like Kafka.
* -
* -
* - save files on a local HDFS directory according to the timestamp. This local directory will be used by flume as the source (spooldir).
* - Ingest data with Flume and write it on to HDFS (sink another HDFS directory or HBase).
* - Connect to Spark (pick data from HDFS) and process with an API for Python – use pyspark

- Create RDDs in Spark from data files on HDFS. From those extract the values we need for the analytics part. Perform transformations on the RDDs so only the needed values are left (filter, then action “collect” which will return an array – store these arrays in variables which will be used to improve the model)