Taxi Fare Prediction System Using Key Feature Extraction in Artificial Intelligence

Dr. Balika J Chelliah
Jai Singh
Devansh Chaturvedi
Avinash Kumar Singh
Chennai

Article History: Received: 10 November 2020; Revised 12 January 2021 Accepted: 27 January 2021; Published online: 5 April 2021

Abstract: Ridesharing is a service that uses travel information to match passengers, thus reducing the total demand of cars on road. However, the problems with the system are that it is expensive, and not suitable for high capacity and long distances. In this paper, we define the problems mentioned above methodologically. The proposed system uses information such as ride requests to achieve efficient taxi-request indexing and thus, improving the matching of taxi and passenger. Particularly, it marks indexes on taxis such as the geographical location and the source and destination, while selecting the most optimum route for the travel and satisfying both online, and offline ride bookings. Considerable amount of evaluation shows the accuracy of the system proposed, which is quick enough to process the requests in milliseconds. Unlike other services, it makes it easier to be deployed of shared and distributed infrastructure.

Keywords: Correlation, Infrastructures, Key Frame Extraction, Ridesharing.

1. Introduction

One of the major problems of travel, and transportation is solved by taxi ridesharing. According to a recent report, most passengers approved of taxi ridesharing. Particularly, the most common ways passengers use ridesharing are by booking a ride on an App, or by taking ridesharing services offline through a taxi. The proposed system looks for routes while satisfying certain travel conditions. Stability of route here refers to the frequency of rerouting or transferring needs come up for a taxi driver. It could also be viewed as the probability of an unexpected road condition coming up and affecting the travel and time requirements of the driver and the passenger.

The system proposed takes origin, as well as destination into consideration. Also, the current systems immediately return a valid taxi as soon as it is found, instead of searching for the most optimum one. Systems like this which work on partial information fail to filter out the irrelevant taxis in the beginning, while also missing the potential best matches with the minimum cost.

The current system solves the problems of the HDFS (Hadoop Distributed File System) by implementing RSP (Random Sample Partitioning). The results of HDFS are more prone to error. This further generates another drawback in this system because, HDFS does not store the properties of the data. The proposed system solves the method of the current ridesharing services by enabling longitude and latitude-based ridesharing, considering passengers on same route, updating requests dynamically, indexing a taxi that supports the most optimum route.

2. Literature Review and Related Work

An approach known as slugging [5] can be called as a modified version of ride-share commuting and hitchhiking. Slugging presumes abandonment of trip from a passenger's side and that passenger then joins

another passenger's trip while keeping the vehicle's capacity in mind. Although this may seem very similar to our proposed model, but only if just one pickup or delivery point is considered. In our proposed system, an enhanced pickup and delivery point for passengers is designed. Here, it seems more efficient if we pick a location that rests near to both the passengers.

Another approach is displayed in [4]. Instead of just submitting the constraints pickup and destination, the driver can submit some other additional constraints like his/her own start and departure time, number of seats available for ride-sharing. So, when the proposed system will find a driver, his/her information (driver's schedule) will also be shared instead of just the usual constraints. This way if the requested trip exceeds the driver's predefined schedule, the system will not choose the given driver for the trip.

An outlook can be seen in [3] where decentralization is the goal. Every node in the system performs only local operations instead of depending on a central system which connects nodes. Each node is matched with other nodes that have a similar route for a ride. This will cut down the travel time of the passengers. Even though this system seems effortlessly efficient, chances of infinite grouping and ungrouping of nodes is very high. Therefore, our system considers such a situation where if once a node is rejected, then that particular node is not grouped again.

Selecting a good route is another problem that is noted. It considers a road network that has already been tested for the computation of the route. Here, the priority is shorter ride while considering travel time, risk and deadline [2]. But in the real world, road conditions are not monitored dynamically. In our proposed system, dynamic road conditions are also examined (traffic, sudden road damage). The model will recompute everytime it gets an update of the road condition.

RSP distributed data model is proposed to represent a big data set as a set of disjoint data blocks [1]. This allows the system to analyze the on the basis of that particular RSP block. Though this method is quick and requires only a small storage value, the results observed are not very promising. The major drawback here is that since the RSP block consists of only a small part of the whole dataset, chances of neglecting important data points is very high. Our model inhabits a similar approach. We use Key Feature Extraction instead of RSP. So, the newly formed features can be shown as a combination of the original set of features.

3. Architecture Diagram

An engineering chart is a graphical portrayal of a bunch of ideas that are essential for a design, including their standards, components and segments. Interaction will begin from readiness model. From willingness model it will go to coarse matching. In coarse matching it will find several details which are related to booking cab like what will be the distance between current location of cab and current location of person. It will also find details like what the pickup location is and what will be the drop location of the booking person.

From coarse matching the process will go to vehicle identification. In this process it will make sure that if the booking person is looking for micro cab then application will check whether the cab showing near the pickup location is micro or not. After this process it will go to the key features extraction. Here we are using some backend engines for the fare prediction like partial feature extraction, temporal feature extraction. The backend engines like partial feature extraction are very helpful in the whole process of fair prediction.

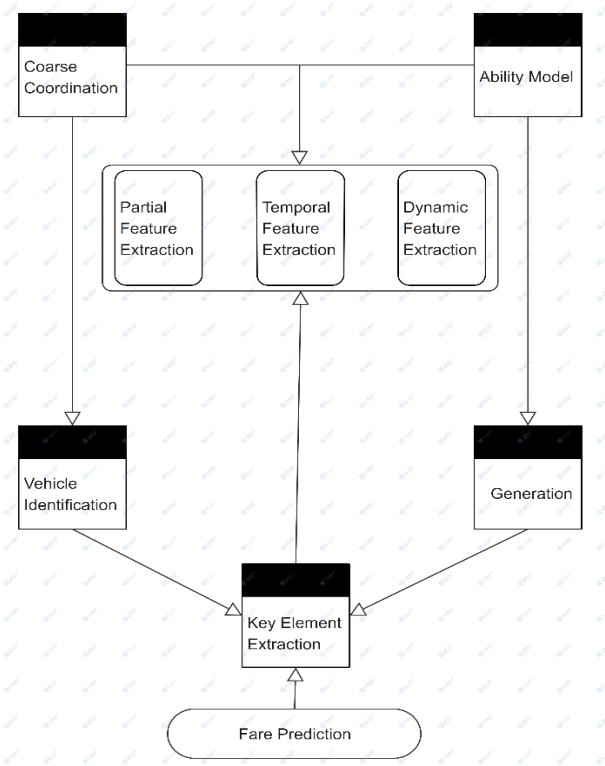

Figure 1. Architecture Diagram

3.1 Key Features Extraction

A raw dataset has abundant features, which makes entire dataset processing chaotic. Hence, Feature Extraction is applied to reduce the initial set of raw data into a more manageable form. Feature extraction also reduces redundant data for a given analysis. The merging of features forms a dataset that is able to contain most of the original information in summarized version.

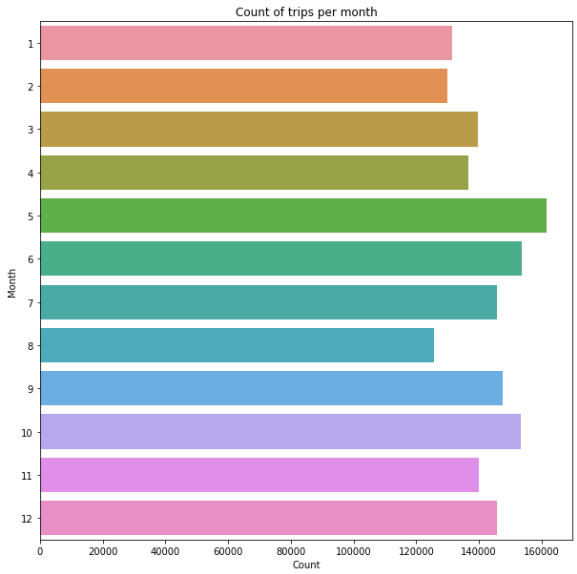

Figure 2. Number of trips on Mondays of every month

Our model uses two ways to find the distance between two points: 1) Euclidean Formula (distance between two vertices on a flat graph) 2) Haversine Formula (Distance between two vertices on a curved surface).

$$d(\mathbf{p},\mathbf{q}) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$

Equation 3.1. Euclidean Formula (Distance Formula)

As we know, that the Earth is a sphere and the surface is curved, which naturally makes our path curved. Here, the curved path is the Earth's surface. Below the equation to Haversine Formula:

$$d = 2rsin^{-1}\left(\sqrt{sin^2\left(\frac{\Phi_2-\Phi_1}{2}\right) + cos(\Phi_1)cos(\Phi_2)sin^2\left(\frac{\lambda_2-\lambda_1}{2}\right)}\right)$$

Equation 3.2. Haversine Formula (Distance Formula)

4. Module Description

4.1.1 Module 1: Data Evaluation

Exploratory Information Investigation is on a very basic level a fundamental plan of performing starting assessments on data to discover plans to check eccentricities to find hypothesis doubts from framework estimations.

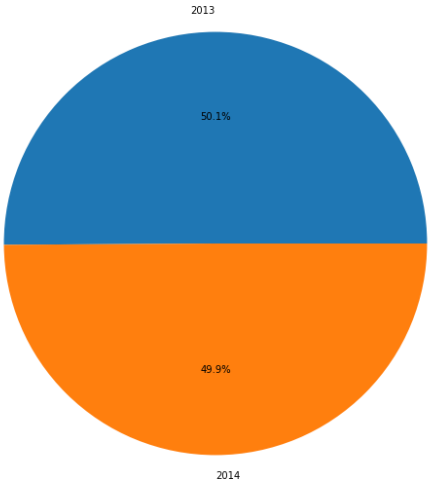

Figure 3. Division of dataset according to year

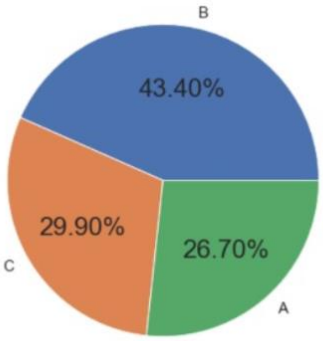

Figure 4. Call Type Pie Chart

EDA depends vigorously on perceptions and graphical understandings of information. While measurable displaying gives a "straightforward" low-dimensional portrayal of connections between factors, they by and large require progressed information on factual procedures and numerical standards. Representations and

diagrams are commonly significantly more interpretable and simple to create, so you can quickly investigate various parts of a dataset. A definitive objective is to produce basic outlines of the information that educate your question(s). It isn't the last stop in the information science pipeline, yet a significant one.

4.1.2 Module 2: Preprocessing

Information pre-handling is a significant advance to set up the information to shape a Riding Fare Prediction model. There are numerous significant strides in information pre-preparing, for example, information cleaning, information change, and highlight determination. Riding Fare Prediction Data cleaning and change are strategies used to eliminate exceptions and normalize the information so they take a structure that can be effectively used to make a model

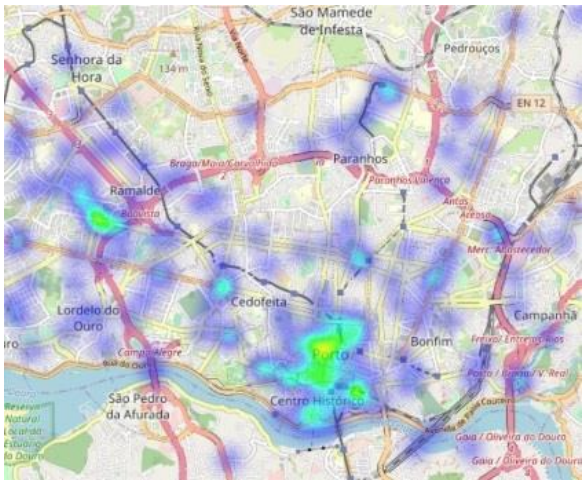

Figure 5. Heatmap of ride call locations

A Riding Fare Prediction informational collection may contain many factors (descriptors); be that as it may, a significant number of these factors will contain excess information. To improve on the dimensionality of the model, it is critical to choose just factors that contain novel and significant data.

4.1.3 Module 3: Prediction

Here we are going to use some backend engines to find fair prediction. We are using prediction conditions which are very crucial when we will make the model.

5. Limitations and Future Work

Our system has a limitation where if a ride is booked in the peak hours of traffic, many complicated routes are also available for the passenger. But these routes are difficult to be computed dynamically. So, future works may include enhancing the algorithm such that it is capable of computing dynamic ride-sharing during peak hours of traffic. Deep neural networks can be employed to overcome this complexity.

6. Conclusion

Contrary to the current ride-sharing services provided by international companies a representation of a trip sharing facility as a novel service is presented by us. Though some of these features have been tried before, the results of the previous work have not been satisfactory. This model is suitable for daily passengers who require sharing their rides to reduce the cost of their trip. Our system is capable of reducing the number of private vehicles on road which will lead to less traffic, shorter rides for the passengers, driver's custom schedule manager, reducing the fares for rides. Basically, it outperforms the existing system of ride-sharing services.

References

[1]. Salman Salloum, Joshua Zhexue Huang and YulinHe. (2019). Random Sample Partition: A Distributed Data Model for Big Data. *IEEE Transactions on Industrial Informatics.*

[2]. San Yeung (2017) PhD Forum: Toward a Human-in-the-Loop Smart Ridesharing System with Self-Driving Technologies. *IEEE International Conference on Smart Computing.*

[3]. Nicolae VlamidirBozdog, Marc X. Makkes, Aart van Halteren, Henri Bal (2018). RideMatcher: Peer-to-peer Matching of Passengers for Efficient Ridesharing. *IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing.*

[4]. Jing Fan, Jinting Xu, Chenyu Hou, Bin Cao, Tianyang Dong, Shiwei Cheng (2018) URoad: An Efficient Algorithm for Large-scale Dynamic Ridesharing Service. *IEEE International Conference on Web Services.*

[5]. Kaijun Liu, Jinwei Zhang and Qing Yang (2019) Bus Pooling: A Large-Scale Bus Ridesharing service. *IEEE Access.*

[6]. P. Goel, L. Kulik, and K. Ramamohanarao. (2016). Privacy-aware dynamic ride sharing. *ACM Transactions on Spatial Algorithms and Systems*, 2(1):1– 41.

[7]. W. He, K. Hwang, and D. Li. (2014). Intelligent carpool routing for urban ridesharing by mining GPS trajectories. *IEEE Transactions on Intelligent Transportation Systems*, 15(5):2286–2296.

[8]. Y. Hou, W. Zhong, L. Su, K. Hulme, A. W. Sadek, and C. Qiao. TASeT (2016). Improving the efficiency of electric taxis with transfer-allowed rideshare. *IEEE Transactions on Vehicular Technology*, 65(12):9518–9528.

[9]. Y. Huang, F. Bastani, R. Jin, and X. S. Wang (2014) Large scale real-time ridesharing with service guarantee on road networks. *Proceedings of the VLDB Endowment.* 7(14):2017–2028