New York City Taxi Fare Prediction

Raymond Chen

January 23th, 2019

Domain Background

Tourists travel in an unfamiliar city may have a taxi ride, but sometimes they don't know reasonable taxi fare in this city. Few wicked taxi drivers may charge unreasonable fare by sneakily taking long route or adding initial charge.

If tourists have a tool to predict reasonable taxi fare based on some simple features like time, pickup location or dropoff location, they can notice unusual charge, take some actions and prevent from fraud. tourists make their budget on travel expense conveniently. For personal reason, when I have a business trip and have to make a budget in advance, I would use this tool to plan my means of transport. If I have sufficient budget, I can take a taxi for a more comfortable trip. Otherwise, maybe I need to take a train or a bus.

This type of problem is so-called regression problem that demands to predict one continuous target value (e.g. taxi fare) using a set of features. There are many academic research addresses on it : For example, long term travel time is predicted from time, wind speed, temperature ,… etc. features using several state of the art regression methods in [1], Internet slangs for sentiment score is predicted in [2], and sentiment score is predicted using Tweets' messages in [3].

Problem Statement

Our target is to predict taxi fare in New York city, and we have several features like pickup GPS location, dropoff GPS location, or number of passengers, etc. to help us build a model to predict. This is a **regression problem** and we can express it as:

$$y = f(x_0, x_1, x_2, \dots)$$

where y is taxi fare for a ride, x_0, x_1, ... are features like time, GPS location, etc. of this ride, and f is a function or model we want to derive.

Given a dataset with many samples having ground truth taxi fare and features, we can apply different machine learning algorithms or even deep neural network to train a model based on them, i.e. finding some set of parameters that can describe the model mathematically. After model is developed, we can predict taxi fare for a given features.

After model is developed, we can evaluate the model performance using certain metric that can describe the value difference between predicted taxi fare y and ground truth taxi fare $\hat{y}$. A simple

metric for the problem is mean square error (MSE) that calculates mean square difference of predicted taxi fare and ground truth taxi fare, sum and average them for the number of samples in given dataset. There're also other metrics, it will be discussed in evaluation metrics section later.

Datasets and Inputs

In this project, **New York City Taxi Fare Prediction** [4] dataset provided in Kaggle is used.

- **File description**
  - train.csv - Input features and target fare_amount values for the training set (about 55M rows).
  - test.csv - Input features for the test set (about 10K rows). Our goal is to predict fare_amount for each row.
  - sample_submission.csv - a sample submission file in the correct format (columns key and fare_amount). This file 'predicts' fare_amount to be $11.35 for all rows, which is the mean fare_amount from the training set.
- **Data fields**
  - **ID**
    - key - Unique string identifying each row in both the training and test sets.
  - **Input (features)**
    - pickup_datetime - timestamp value indicating when the taxi ride started.
    - pickup_longitude - float for longitude coordinate of where the taxi ride started.
    - pickup_latitude - float for latitude coordinate of where the taxi ride started.
    - dropoff_longitude - float for longitude coordinate of where the taxi ride ended.
    - dropoff_latitude - float for latitude coordinate of where the taxi ride ended.
    - passenger_count - integer indicating the number of passengers in the taxi ride.
  - **Output (taxi fare)**
    - fare_amout - float dollar amount of the cost of the taxi ride. This value is only in the training set

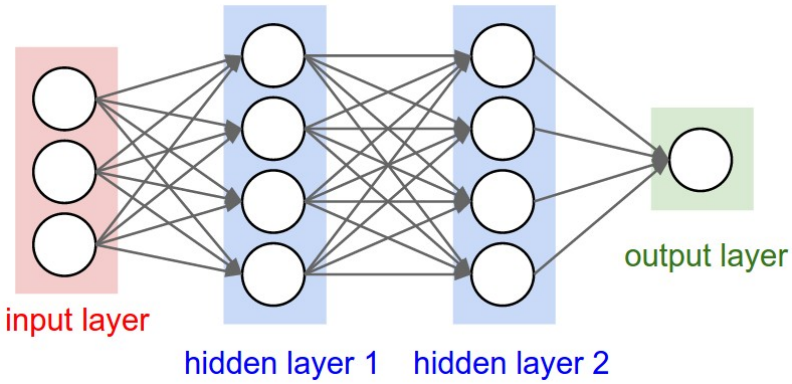
Solution Statement

Several models will be tried in this project, which are well-known to be good candidates for regression problem [5, 6]. Here are some candidates :

- Polynomial regression model
- SVM regression model
- Random forest regression model
- Multiple layer perceptron (MLP) regression model

Some of them have hyper-parameters to be resolved and grid method will be applied to choose the best one. In MLP regression model, different hidden layers and network structure will be tested. I'll try some of them to find a best model and for all cases, I use R^2 as evaluation metric.

Here I take MLP regression model as an example to describe why it works for regression problem. A simple multiple layer perceptron model is illustrated as follows :



A graphical representation of neural network [7]

Features x are fed in input layer (here we have 3 features), two hidden layers with 4 nodes are designed and finally only 1 node for output y (taxi fare). Here nodes are fully connected and form a linear combination relationship. We can insert nonlinear function (a.k.a. activation function) like ReLU function between each layer to introduce some nonlinearity into this model.

$$L1 = ReLU\big(h_1(x)\big)$$
$$L2 = ReLU\big(h_2(L1)\big)$$
$$y = ReLU\big(h_2(L2)\big)$$

We can define loss as mean squared error or mean absolute error and use gradient descent to train the model and get the model weights (parameters). Finally we can predict y (taxi fare) using these model weights and input features x. Different layers, nodes, and activation layers can be designed in this method.

Benchmark Model

A simple benchmark model for this problem is **multiple linear regression model**. Mathematically, it can be expressed as :

$$y = \theta_0 + \theta_1 * x_1 + \theta_2 * x_2 + \cdots + \theta_n * x_n = h_\theta(\boldsymbol{x}) = \theta^T \cdot \boldsymbol{x}$$

where θ_j is the j-th model parameters to be trained, $x_i$is the i-th feature value and n is the number of features, θ^T is the transpose of θ (model's parameter vector), $\boldsymbol{x}$ is the feature vector, and h_θ is the hypothesis function using the model parameter θ.

For a simple case, we can use all features (pickup_datetime, pickup_datetime, …) in dataset as $x_0 x_0, x_1, \ldots$, use fare_amount as y. By using mean square error function $MSE(\theta)$ as cost function and applying gradient descent method, we can derive a set of model parameter θ and thus the model.

$$MSE(\theta) = \frac{1}{m}\sum_{i=1}^{m}\big(\hat{y}^{(i)} - y^{(i)}\big)^2$$

This model will then be applied in the test dataset and R^2 will be calculated as the evaluation metric. Because this is a very simple model that generally performs poor in most complex datasets, we can use it as a baseline benchmark. What we designed and trained model should outperform this one in evaluation metric R^2.

Evaluation Metrics

The metric used in this project is coefficient of determination R^2.

$$R^2 = 1 - \frac{SSE}{SST} = 1 - \frac{MSE}{Var(y)}$$

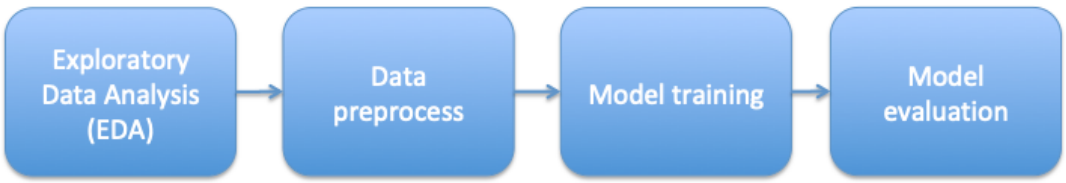
$$SSE = \sum_{i=1}^{m} \left(\hat{y}^{(i)} - y^{(i)}\right)^2, SST = \sum_{i=1}^{m} \left(\mu_y - y^{(i)}\right)^2$$

where SSE is sum squared error and SST is total sum of squared.

The values for R^2 range from 0 to 1, which captures the percentage of squared correlation between predicted and actual value of the target variable. Value between 0 and 1 indicates how well the model can explain the data. Higher R^2 is, better the model is. Note that a model can be given $R^2 < 0$, which means it's arbitrarily worse than always predicts the mean of the target variable.

Project Design

Here's the flow chart of this project :



Step 1 : Explore data and find features' relationship

Step 2 : Preprocess data such as : impute missing field, transform data, split data into train/valid/test data with data shuffling, feature selection/extraction, feature scaling, outlier removal, …

Step 3 : Select and implement a model such as SVM regression model, Random Forest regression model, Multiple Layer Perceptron regression model … to train using train data and valid data

Step 4 : Evaluate model using test data

Reference

[1]
https://www.researchgate.net/publication/230819938_Comparing_state-of-the-art_regression_methods_for_long_term_travel_time_prediction/download

[2]
https://www.researchgate.net/publication/283318703_Detection_and_Scoring_of_Internet_Slangs_for_Sentiment_Analysis_Using_SentiWordNet

[3] https://www.researchgate.net/deref/http%3A%2F%2Faclweb.org%2Fanthology%2F%2FS%2FS13%2FS13-2053.pdf
[4] https://www.kaggle.com/c/new-york-city-taxi-fare-prediction#description
[5] Python Machine Learning 2nd edition, by Sebastian Raschka and Vahid Mirjalili
[6] Hands-On Machine Learning with Scikit-Learn & TensorFlow, by Aurelien Geron
[7] https://beamandrew.github.io/deeplearning/2017/02/23/deep_learning_101_part2.html