Predicting the Delivery Time of Wolt's Orders
Wolt Data Science Internship Assignment

Nikolay Pashov
https://github.com/nikifaets

January 29, 2021

Abstract

I am given a structured tabular dataset containing information about Wolt's deliveries. A data sample includes delivery time, estimated delivery, according to an unnamed algorithm, and some independent variables, such as weather condition, time, etc., which the delivery time may depend on. My goal is to create a model that predicts the delivery time based on these variables. I first apply data exploration and preprocessing steps to drop unuseful data and extract more information from the dataset. Then, I try to design a machine learning model that finds a relationship between the independent variables and the delivery time and can thus predict the delivery time. The final model is a neural network that predicts delivery times that are closer to the ground truth than the estimates from the dataset.

1 Data Exploration

1.1 The Dataset

See Table 1 for a sample row from dataset. The column $ACTUAL\_DELIVERY\_MINUTES$ holds the value I want to predict.

1.2 Data Exploration

At first sight, I was skeptical that all the variables contribute to the delivery time. For example, even though Wolt makes deliveries on bicycles, I find it unlikely that the wind speed is a big factor. Especially without specifying the wind direction. Also, all latitudes and longitudes in the dataset barely differ by hundredths. It can be difficult for a model to take advantage of this small difference given the scale of other variables in the dataset. A similar analysis can be made for other variables. This leads to the conclusion that preprocessing should be performed and some data can be dropped to reduce noise.

1.3 Preprocessing

The $TIMESTAMP$ value is a string which means it cannot be used in a mathematical model. I dropped this column and replaced it with two new ones - $TIMESTAMP\_HOURS$ and $TIMESTAMP\_MINUTES$, which represent respectively the hour and the minutes from the timestamp.

Column Name	Sample Value
TIMESTAMP	2020-08-01 06:07:00.000
ACTUAL_DELIVERY_MINUTES - ESTIMATED_DELIVERY_MINUTES	-19
ITEM_COUNT	2
USER_LAT	60.158
USER_LONG	24.946
VENUE_LAT	60.16
VENUE_LONG	24.946
ESTIMATED_DELIVERY_MINUTES	29
ACTUAL_DELIVERY_MINUTES	32
CLOUD_COVERAGE	25
TEMPERATURE	17.8
WIND_SPEED	3.01033
PRECIPITATION	4.73684

Table 1: A sample row from the dataset. Note that the table is inverted for presentation purposes, i.e. a row in this table is a column in the original dataset.

As already mentioned, the locations of the venue and the client could be valuable information. To make it more interpretable, I converted the two coordinates to distance in kilometers. I dropped the four provided columns and inserted *DISTANCES_KM*.

1.4 Variable Relations

After preprocessing the dataset, I investigated how each variable relates to the delivery time (see Fig. 1). I observe that, for most of the variables, it is hard to say they have barely any contribution to the delivery time. It is worth noticing that this analysis holds only for single variables' relations to the delivery time. It may be the case that a combination of multiple variables has more impact on the delivery time than each variable on its own.

After observing the relations in Fig. 1 and playing with the data for some time, I decided I want to only use the following variables for my models:

- *ITEM_COUNT*
- *DISTANCES_KM*
- *WIND_SPEED*
- *ESTIMATED_DELIVERY_MINUTES*
- *TEMPERATURE*

2 Methods

I used two approaches for approximating the relationship between these variables and delivery time. Firstly, I tried to fit linear regression, and then I built a neural network as a more complex approximation function. I also tried to find a more optimal neural network using *Autokeras* but I did not have the time nor the computing power to be efficient to make proper use of this approach. To evaluate the models' accuracies, I used both MSE (Mean Squared Error) and MAE

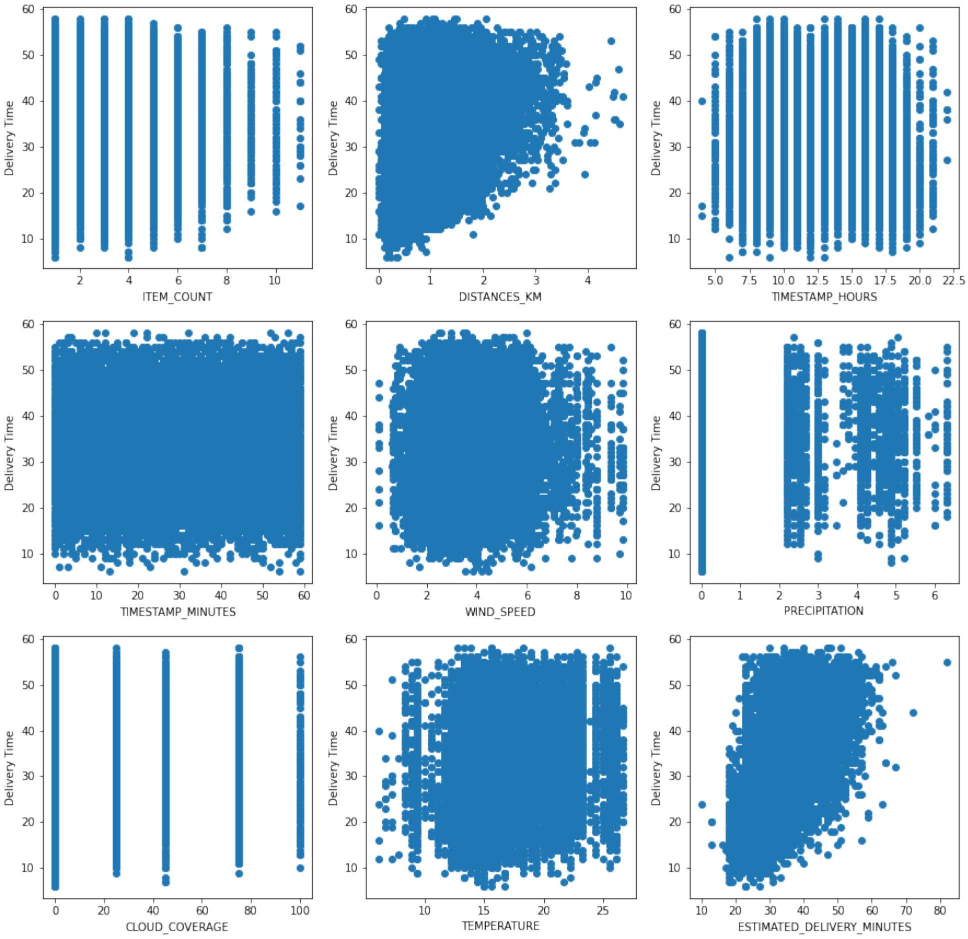

Figure 1: Scatter plot of each variable against the delivery time.

(Mean Absolute Error) loss functions with the actual delivery time as the target value. Thus, I have the following metrics:

$$MSE(Y_{\text{pred}}, Y_{\text{actual}}),$$
$$MAE(Y_{\text{pred}}, Y_{\text{actual}}),$$

$$MSE(Y_{\text{given\_estimate}}, Y_{\text{actual}}),$$
$$MAE(Y_{\text{given\_estimate}}, Y_{\text{actual}}),$$

where

$$MSE(\vec{x}, \vec{y}) := \frac{1}{n} \sum_{i=1}^{n} (x_i - y_i)^2$$

$$MAE(\vec{x}, \vec{y}) := \frac{1}{n} \sum_{i=1}^{n} |x_i - y_i|$$

and $n = |\vec{y}| = |\vec{x}|$. Y_{pred} is my model's predicted delivery time. Depending on the context, Y_{pred} can be the linear regression's prediction as well as the neural network's. $Y_{\text{given\_estimate}}$ is the estimated delivery time provided in the dataset and Y_{actual} is the actual delivery time, also from the dataset. The inputs to MSE and MAE are vectors because I calculate the losses on batches of predictions. My goal is to achieve more accurate prediction than the provided estimated delivery times from the dataset. This means

$$Err(Y_{\text{pred}}, Y_{\text{actual}}) < Err(Y_{\text{pred}}, Y_{\text{given\_estimate}}),$$

where $Err \in \{MSE, MAE\}$.

2.1 Fitting a Linear Regression

In this current task, fitting a linear regression means finding coefficients $a_1, a_2, ..., a_5$, such that

$$\sum_{i=1}^{5} a_i \cdot x_i = Y_{actual}$$

where $x_1, x_2, ..., x_5$ are the variables listed in Section 1.4.

I used *scikit-learn*'s implementation of linear regression.

2.2 Building a Neural Network

Since the data space's dimensionality is fairly small, the neural network is simple. It is a sequence of fully-connected layers with ReLU activations. I also added batch normalization and dropout layers because the network was overfitting quickly. See a Tensorboard graph of the architecture in Fig. 2. The network is implemented using *PyTorch*. The input of the network is a tensor of shape (batch_size, $|\vec{x}|$), where *batch_size* is a hyperparameter with the value of 128 in the latest experiments and $\vec{x}$ is a 5-dimensional vector containing the variables from Section 1.4.

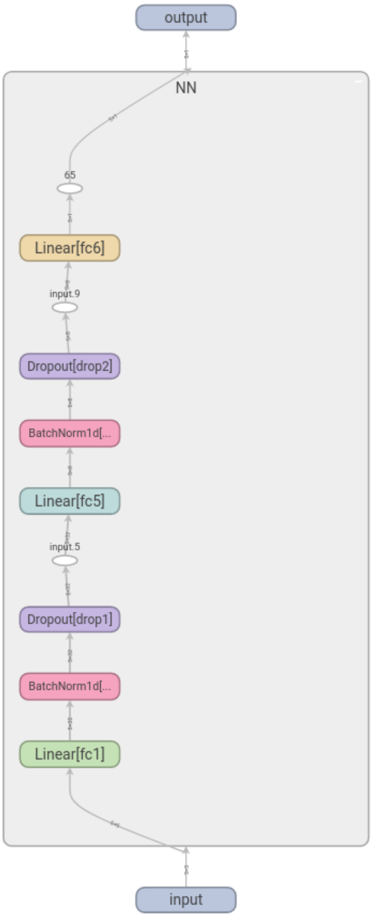

Figure 2: Tensorboard graph of the neural network's architecture.

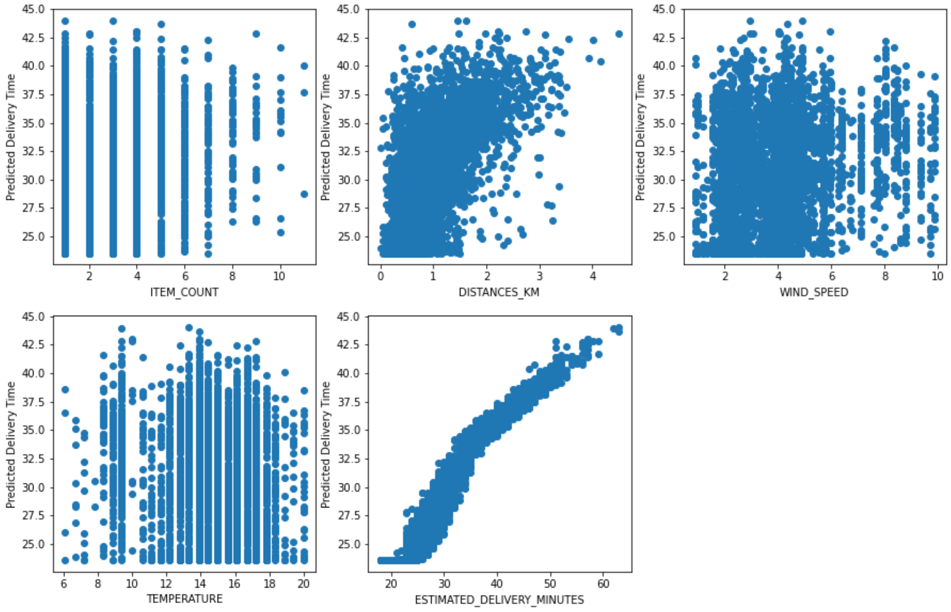

Figure 3: Relationship between the variables listed in Section 1.4 and the output of the neural network.

Results from:	MSE	MAE
Dataset estimate	82.10	7.25
Linear regression estimate	74.70	7.01
Neural network estimate	**70.95**	**6.81**

Table 2: Both the linear regression model and the neural network output more accurate delivey time estimates than the estimates from the dataest.

3 Results

After I trained the network and achieved desirable results, I made a similar plot to Fig. 1 to see how each of the variables contributes to the output. See the plot in Fig. 3. As in Fig. 1 most of the variables do not seem to relate much to the delivery time. All variables except for *DISTANCES_KM* and *ESTIMATED_DELIVERY_MINIUTES* seem almost randomly scattered. An interesting observation is that there is certainly a strong relationship between the estimated delivery time and the actual delivery time.

A hypothesis can be made that, if the neural network has found a mapping between the estimated delivery time and the actual delivery time, it has found a way to express the inaccuracy of the model that produces the estimated delivery time. It then uses this inaccuracy as input to produce better predictions.

Both the linear regression and the neural network achieved results that are closer to the actual delivery time than the estimated delivery times from the dataset. As described in Section 2,

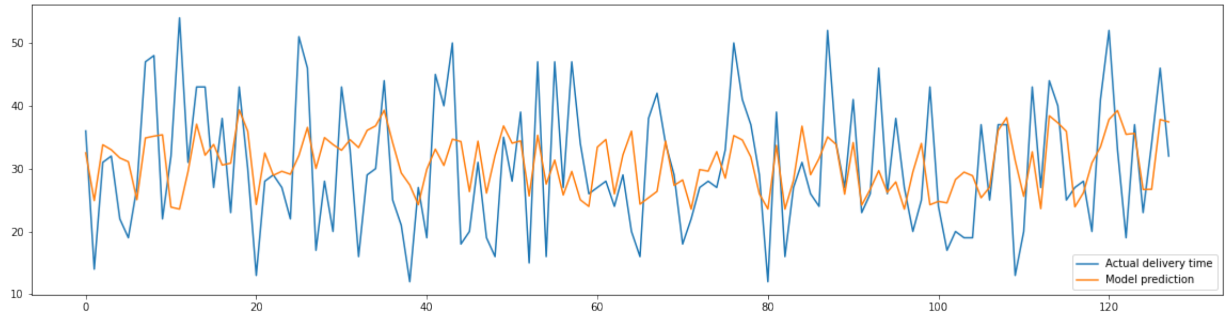

Figure 4: A batch of delivery times and the respective predictions from the neural network.

the metrics I used for evaluating the models' performances are *MSE* and *MAE*. Table 2 shows a detailed comparison of results. See Fig. 4 for a comparison between the neural network's predictions and the actual delivery times. Values are from a random batch.

4 Conclusion

In this assignment, I designed a model to predict the needed time for Wolt's deliveries based on independent variables, shown in Table 1. I executed transformation on the dataset to enhance the relevance of some values and explored which variables have a direct impact on the delivery time (see Fig. 1). Then, I used the seemingly most important variables to approximate the relationship between them and the delivery time. I did that with a linear regression and a neural network. They both estimated more accurate values than the estimated delivery time provided in the dataset (see Table 2). The neural network produced better results than the linear regression.

5 Further Development

I would say I am satisfied with the results and the assignment was fun. However, if provided more time and expertise, I would work on the following to improve my results:

- Make better use of *Autokeras*. Allow it more time to let it come up with a good solution.
- Both Fig. 3 and Fig. 1 show that the variable that contributes most to the delivery time is the estimate provided in the dataset. This looks odd and the importance of the other variables should be studied. It is worth trying to not explore the contribution of single variables, but a combination of multiple. E.g. the wind speed may not contribute on its own, but there may exist a function

  $$F: \text{wind\_speed, precipitation} \longrightarrow \text{new\_variable},$$

  such that new_variable has a significant relationship to the delivery time.

  One way to look for such relationships is to do similar plots to Fig. 3, 1 but instead of plotting raw variables on the X-axis, to plot the output of an intermediate neuron of the neural network. This neuron would serve the purpose of F.

Another possibility is to apply a dimensionality reduction algorithm to the raw data, such as PCA, and study which variables carry the most significance.

6 My Background and Wolt

6.1 Projects I have worked on

My background in data science consists mostly of computer vision and deep learning projects.

3 years ago, when I was in the 11th grade, I participated in Research Science Institute in MIT where I worked on my first machine learning project. I developed an unsupervised machine learning algorithm for phenotyping brain cells. It involved an autoencoder that compressed images of different cells type and clustering algorithms that detected different data classes (cell types) from batches of compressed images.
See more: `https://github.com/nikifaets/Phenotyping`.

Last year I attended a university class in deep learning and my coursework was a method for selectively transferring face features between different face images. I achieved it by using a (novel in my opinion) mixed neural architecture consisting of both autoencoder and GAN.
See more: `https://github.com/nikifaets/semantic-face-editing`.

When I interned in VMware, I developed a support bot that detects bug report messages in Slack channels and automatically opens tickets in an issue tracking system.

6.2 Data Science Ambitions

I enjoy working on data-related problems because data science is a field that is applicable to problems in various areas and industries. Generally, I say I like computer science because it provides solutions everywhere - sports, mathematics, transport, you name it. Data science, in my opinion, is the subfield for which this statement holds particularly true.

Take Wolt for example. One of the task proposals for this assignment was to detect whether a meal is vegetarian, given a picture. (Using an object detection algorithm such as YOLO or Faster R-CNN) One can automatically check if all the meal components are vegetarian. The same could be done, say, to find allergens, or even to determine whether a meal is allowed in a specific diet. With such algorithms, manual labor is reduced and the customer receives more feature-rich experience.

Similar thinking can be applied to all aspects of a business. The UI can make use of good suggesting algorithms, support chat bots and what not. But we can take this a bit further with imagination. Why not develop a method that suggests food based on more exotic inputs like mood, day schedule, world news? Efficient route planning and time estimation (wink wink) algorithms can be greatly improved with data science to make use of more variables.

Data science solutions include more non-triviality and unpredictability than "classic" algorithms. If we put aside the downsides, this makes developing such solutions more artistic, non-repeatable and fun.