All necessary code is in `shopping_time_model.py`
You should be able to run it out of the box with `requirements.txt` installed and `datafile` changed the location of your '`datafile.tsv`' The only other things that you may want to change are `number_of_runs` or `test_size, sample, scale` in my preprocessing function. I've included all of my scratch code in case you want to look through how I got to my solution, but I haven't been back through any of them to clean them up.

1 – Predicting the Shopping Time for an Order

1. After exploring the dataset, what potential issues do you see with the underlying data provided? How would you go about doing some basic cleaning on this dataset?

The data has some null values (for simplicity, and since it's not much compared to the size of our data I'm going to just drop these)

The data is a mix of continuous and discrete variables. For some of these, like month and shopper gender, I can code them as a category instead (this is done in my preprocessing function, lines 34-53 and should be pretty straight forward.) A large portion of the data is the category of the order. From some investigation, it seems that the number in the category represents the number of items in the total order that are in that category. To try and give meaning to this I instead change the number to its percent of the total order (lines 60-72). For shopper gender, there seems to be a 0, 1, 2. For simplicity I replace 2 with either a 1 or a 0 at a similar ratio of the existing data. I'm also going to drop some features that seem unimportant or definitely don't carry any information like order_id, member_substitution_preference, and shopper_yob (since this information already exists in shopper_age.) I'm also going to remove any data point that is more than 3 standard deviations outside of its mean.

2. Our goal is to predict the actual shopping time required for a given order (`actual_shopping_duration_min`), as having more accurate predictions here would potentially have huge benefits around operational efficiency.

- Build a predictive model for the shopping duration using whatever model class and methods you'd like. (*)

I played around with a whole bunch of different models, but in the end simple linear regression on almost all of the data, and with no feature selection, seemed to do the best for me. You'll also see decision tree regression and support vector machine regression, which were close followers.

- **How would you assess your model's out-of-sample performance? (*)**

In this case I've put my model and preprocessing inside of a loop so each time it runs it's running on a new training/testing set essentially mimicking k-fold cross validation. I split the data into a 66/33 training/testing set within my preprocessing function (with random state set to false). By running prediction only on my testing data, data the model has never seen before, I'm able to get an accurate out of sample performance. My best score was simple linear regression which was off from actual shopping time by about 14 minutes.

run	Average Error	Average Absolute Error	Median Error	R^2
1	-0.201353	14.013188	3.919853	0.194424
2	0.421738	13.614014	4.166046	0.248881
2	-0.102242	14.934239	4.683610	0.143490

Assume that the current system generates a naïve estimate for the estimated shopping duration that multiplies the number of order lines by 2.5 minutes. Given your model from (a), how would assess that model's performance against this baseline estimate. (*)

Simply multiply my `X_test.order_num_order_lines` by 2.5 and then assess that against `y_test`. It's the last part of my `shopping_time_mode.py` It performs with an average absolute error of around 22 minutes.