Learning-based entity extraction model

Task Description:

The task was to build a learning-based entity extraction model to recognize and extract a store number from the dataset. There was the "transaction_descriptor" column from where I needed to extract the store number. The dataset was divided into 3 parts (train, validation and test).

First step:

In the exploratory analysis step, I noticed that none of the store numbers started with 0s or symbols. So, in the data preprocessing part I decided to remove the noise, i.e. all of the leading 0s and symbols from the "transaction_descriptor" field. For that I created the functions: *remove_punctuation()* and *removezeros()*. The following preprocessing functions are then applied to Train, Validation and Test datasets.

Second step:

As there were too many different labels (store numbers), I decided to separate them into 2 main classes: "This", meaning the needed label, and "O", meaning other. Later I will unmask them from my predictions and the word corresponding to "This" label will be the predicted store number. Then, in order to be able to train a model which can understand and extract the store number, I decided to map each of the words in "transaction_descriptor" with their corresponding labels. For this I created the *sentence_to_words()* function and separated the sentences of Train and Validation datasets only. The function also maps the corresponding labels ("O", "This") to words.

Step three:

For this NER(named entity recognition) task I decided to choose the pretrained Bert model from "simpletransformers". After loading the model, I set some of the training hyperparameters and trained the model for 5 epochs. After training with several different hyperparameters I stopped on the ones, which are present in the code. To evaluate my model, I calculated the evaluation loss with the help of my validation dataset and saved the output weights in a folder. You can see the accuracy results printed in the code above.

Time to make predictions on Test dataset:

After applying the preprocessing on the test set, my DataFrame looked like this:

	transaction_descriptor	store_number
200	IN N OUT BURGER 242	242
201	BP 9442088LIBERTYVILLE B	9442088

So that I can feed the sentence in each row to my model (Obviously, the store number was not given to the model) and get a dictionary of words and their predicted labels with the help of *map_outputs()* function. An example of a prediction is shown below.

```
[{'IN': 'O'}, {'N': 'O'}, {'OUT': 'O'}, {'BURGER': 'O'}, {'242': 'This'}]
```

The function also picks the key of the label “This” and returns that key as the store number. This process is done for each row of the test dataset’s “transaction_descriptor” column. Then the predicted store numbers are added to the test dataset in a new column called “pred_store_number”. Then I simply compare the actual store numbers of my test dataset with my predicted values and count the number of right predictions. The model did a pretty good job by predicting with an accuracy of 88%.