

ML Take-Home Exercise by Fetch

- Submission by Shreekant Gokhale (gokhale6@illinois.edu)

Repository link: https://github.com/ShreekantGokhale/ML_TakeHome_Fetch

Given Data:

No. of receipts scanned daily in the year 2021.

Objective:

To predict the no. of receipts that will be scanned in the months of next year i.e., 2022

Data processing and feature engineering:

First, the given data was checked for missing data. There was no missing data in the given file. Then through simple visualization of data, linear regression was decided as a model to be trained.

According to the given data, the date is the independent variable while, No. of receipts scanned is the dependent variable. In addition to that, the number of receipts scanned can be dependent on the day of the week and month as well considering the real-world examples. (e.g. higher rush on weekends or low usage during winters etc.) These two features can be extracted from the date itself. Thus, the final data consists of date, day of week, and month.

After the feature engineering step, the data is divided into train-val-test data using sklearn's train test splitter. These dependent and independent variables are scaled using min-max scaling for better training results.

Model:

The model chosen here is a 2 linear layer-deep model with ReLu activation layers. in which hyperparameters are adjusted to give the best results. The batch size is taken to be 1 as the available data is limited. Reducing the batch size to the minimum allows us to train the data more accurately. No. of epochs and learning rate are set to be 150 and 0.0001 which gave the best training results in this case.

How the variables are upgraded:

The MSE loss criterion is used to calculate the loss between the predicted and actual values while training. According to this loss, the model parameters are upgraded using the Adam optimizer. After each epoch MSE loss between the training set and validation set is compared using the loss plot shown below.

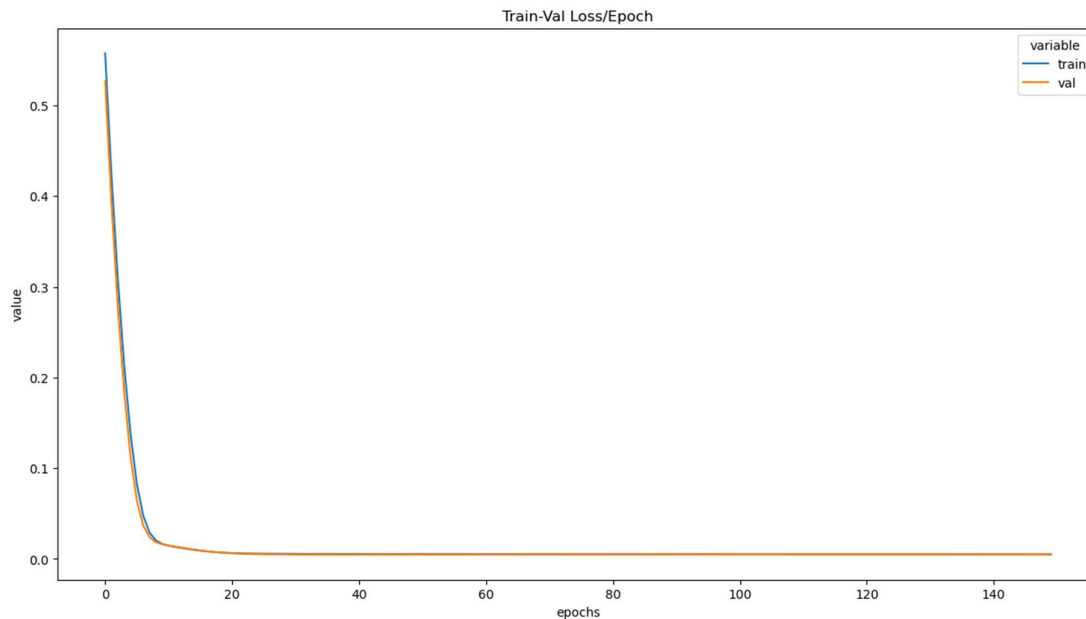


Fig 1: Train-validation data MSE loss plot for each epoch while training

From the above plot, the model is sufficiently trained. This plot is used to set the hyperparameters like Epochs, Batch size, and learning rate according to the best model training results.

Test accuracy:

After the model training, the accuracy of the model is tested on the test data using the MSE loss and R^2 measures. For the model trained here, the R^2 value was 0.912.

```

: mse = mean_squared_error(y_test, y_pred_list)
  r_square = r2_score(y_test, y_pred_list)
  print("Mean Squared Error :",mse)
  print("R^2 :",r_square)

```

```

Mean Squared Error : 0.00625522336832056
R^2 : 0.9122778461947291

```

Fig 2: MSE loss and R^2 measures for the test data

Saving the models:

The Multi-regression model built in Pytorch is saved in 'model_scripted.pt' file using the script functionality in Pytorch. The min-max scalers used for dependent and independent variables are saved in the pickle files named 'scalerx.pkl' and 'scalery.pkl'.

App using HTML and Flask:

A simple user interface is made in HTML where you can insert the name of the Month to get the prediction of the number of receipts that will be scanned in that month of year 2022. The Flask app for the trained ML model is made where all the saved models were loaded.

Running the code for predicting the value of a number of receipts for each day in a loop, the total for the requested month is calculated in the app.py file.

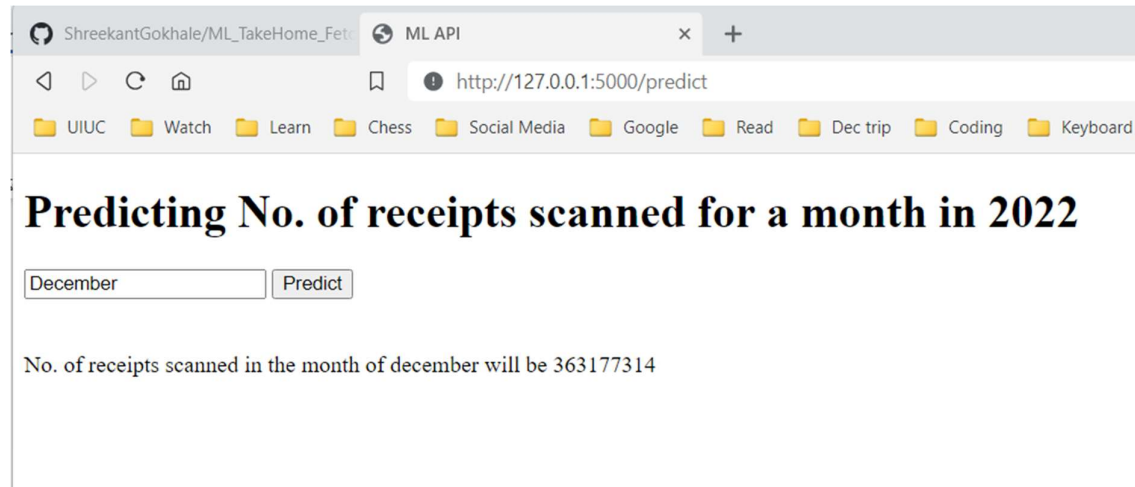


Fig 3: Final User Interface of a Flask application

Docker file:

This file is not included in the submission. I tried to package the app in docker using the deployment in Heroku using Github workflows. I was trying to figure out the docker applications for the first time. I was able to run the workflow, but I was getting an error while building the application in Heroku. That is the reason unfortunately I could not package this app in the docker hub. The screenshots of the error I was getting and the workflow file are shown below. The code along with docker files is saved to a different repository named Fetch ML in the master branch.

Link: https://github.com/ShreekantGokhale/Fetch_ML/settings

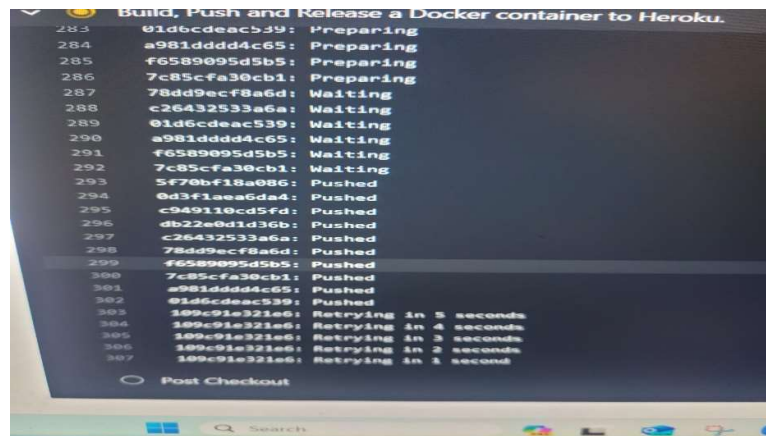
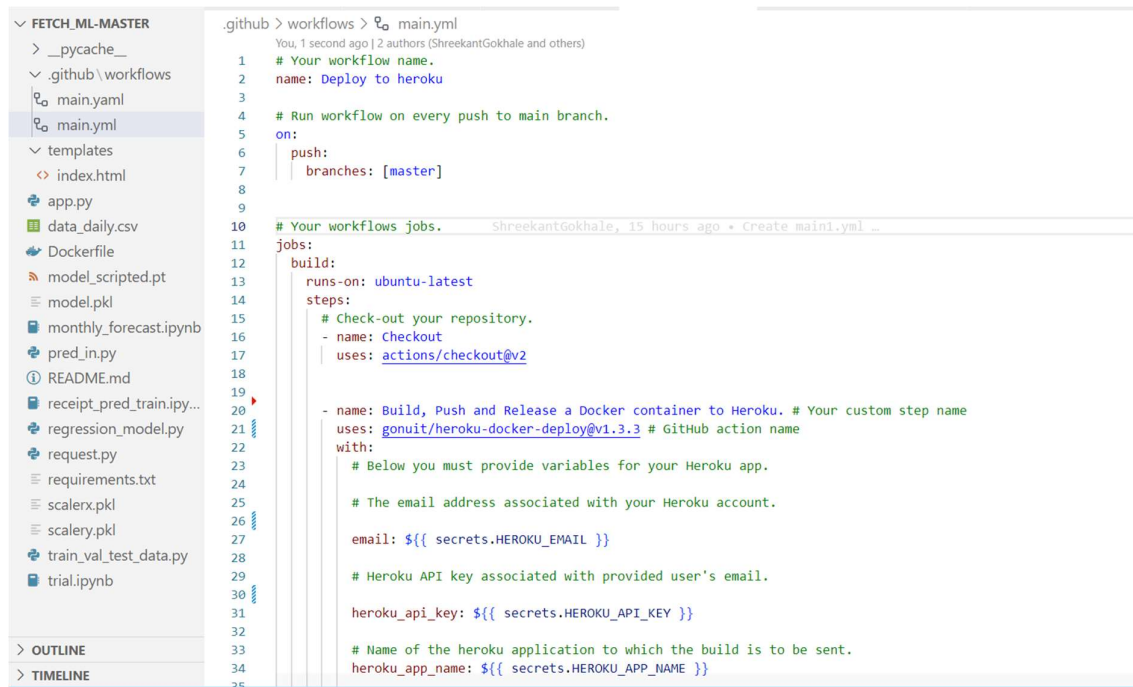


Fig 4: Error during the deployment in Heroku using Github workflows.



```
.github > workflows > main.yml
You, 1 second ago | 2 authors (ShreekantGokhale and others)
1 # Your workflow name.
2 name: Deploy to heroku
3
4 # Run workflow on every push to main branch.
5 on:
6   push:
7     branches: [master]
8
9
10 # Your workflows jobs.
11 jobs:
12   build:
13     runs-on: ubuntu-latest
14     steps:
15       # Check-out your repository.
16       - name: Checkout
17         uses: actions/checkout@v2
18
19       - name: Build, Push and Release a Docker container to Heroku. # Your custom step name
20         uses: gonuit/heroku-docker-deploy@v1.3.3 # Github action name
21         with:
22           # Below you must provide variables for your Heroku app.
23
24           # The email address associated with your Heroku account.
25
26           email: ${ secrets.HEROKU_EMAIL }
27
28           # Heroku API key associated with provided user's email.
29
30           heroku_api_key: ${ secrets.HEROKU_API_KEY }
31
32           # Name of the heroku application to which the build is to be sent.
33           heroku_app_name: ${ secrets.HEROKU_APP_NAME }
```

Fig 5: Github workflows file.

References:

- 1) <https://towardsdatascience.com/machine-learning-for-store-demand-forecasting-and-inventory-optimization-part-1-xgboost-vs-9952d8303b48>
- 2) <https://towardsdatascience.com/pytorch-tabular-regression-428e9c9ac93>
- 3) <https://github.com/krishnaik06/Deployment-flask/blob/master/model.py>
- 4) <https://github.com/krishnaik06/bostonhousepricing/blob/main/Dockerfile>