

Week 5 – Flask App Webserver Deployment Assignment Report

Online deployment enables remote access of the ML applications developed.

For the assignment, the Heroku webserver platform was utilized to host the Flask application.

Flask Application

The functionality of the Flask application developed was to predict on the probability of clients interacting/clicking on an online advertisement based on the following criteria:

1. Age
2. Daily Internet Use – amount of time spent online in minutes
3. Daily Site Access – amount of time spent on the website
4. Area Income – average income in the customer's area.

Given the provided dataset, is it possible to determine the survivors based on the key features?

Data Acquisition & Preparation

Data obtained from:

<https://www.kaggle.com/imakash3011/customer-personality-analysis/download>

Data Intake Report

Name: Deployment on Flask – Advertisement Engagement Predictor

Report date: 28th October 2021

Internship Batch: <Enter your batch code from Canvas course>

Version: 1.0

Data intake by: Teddy Waweru

Data intake reviewer:

Data storage location:

Tabular data details:

Total number of observations	1000
Total number of files	1
Total number of features	10
Base format of the file	.csv
Size of the data	212KB

Note: Replicate same table with file name if you have more than one file.

Proposed Approach:

- Mention approach of dedup validation (identification)
Verified unique rows in the dataset.
- Mention your assumptions (if you assume any other thing for data quality analysis)
The advertisement content was assumed to be neutral for all engagements. I.e. all candidates in the dataset would react in a similar way to the advertisements that were displayed, & there was no particular bias to or against the advertisement.

=

Model creation

The model utilized was the RandomForestTreeClassifier, which would develop leaf nodes dependent on the features selected to generate the model. Based on analysis, it was clear that the features that were selected affected the predicted value in varying ways. The RandomForestTreeClassifier would **develop conclusions** whilst considering the effects of each of these features.

Model training

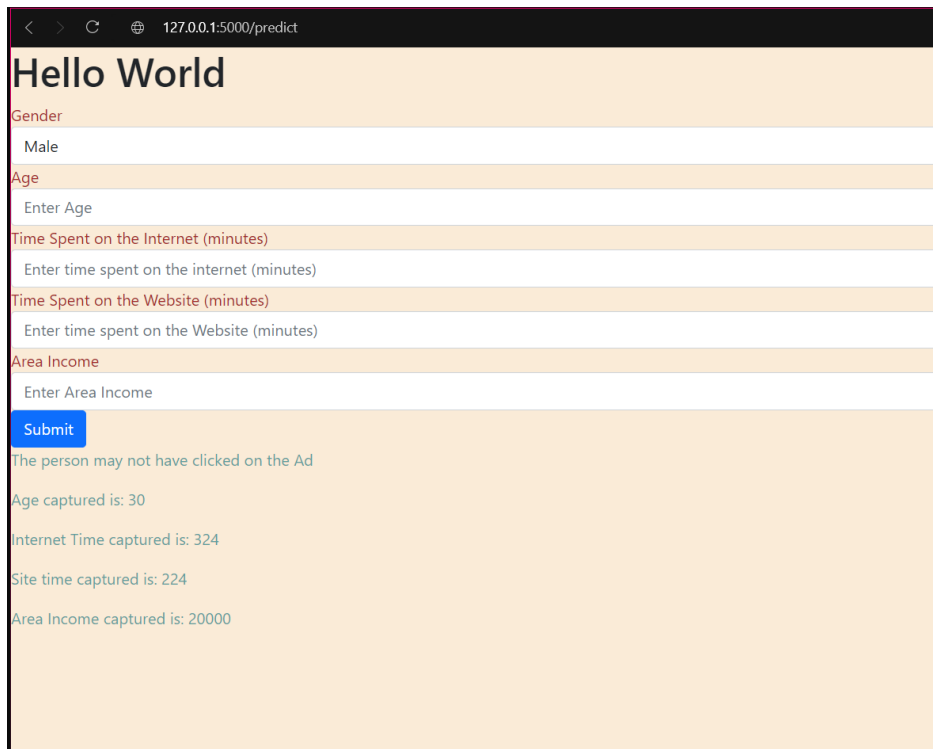
The data was split automatically by utilizing `scikit-learn.model_selection train_test_split()` module.

Model evaluation

By utilizing `scikit-learn.metrics mean_absolute_error`, the error range for the model was calculated as **0.04**, which would suffice for the application to be developed.

Application Development

Application setup was a basic app.py referencing HTML files, & carrying out the predictions based on the features collected from a HTML form.



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:5000/predict'. The page has a light orange background and a dark header with the text 'Hello World'. Below the header, there are five input fields, each with a label in red text: 'Gender', 'Age', 'Time Spent on the Internet (minutes)', 'Time Spent on the Website (minutes)', and 'Area Income'. The 'Gender' field contains the text 'Male'. Below the input fields is a blue 'Submit' button. Underneath the button, there is a message in green text: 'The person may not have clicked on the Ad'. Below this message, there are four lines of green text showing the captured values: 'Age captured is: 30', 'Internet Time captured is: 324', 'Site time captured is: 224', and 'Area Income captured is: 20000'.

Figure 1: HTML index page, with a form that submits a POST call to the FLASK APP (development mode).

Heroku App Setup

The following steps were taken to develop the application:

1. Create the necessary documentation to ensure that the application has the necessary files for Heroku deployment ie. Procfile, Python packages list.
The application files were developed
 - a. Develop the list of packages required to deploy the app:
`pip list --format=freeze > requirements.txt`
 - b. Create the process file/ Procfile for Heroku to PaaS to recognize the application as a HTTP based application:
web: gunicorn app:app
 - c. Some changes in the file naming within the app.py were necessary in order for the application to run successfully, ie. the model.pkl file was moved to the static folder.
2. Initialize the Heroku application
 - a. Create an account on Heroku platform & create the application.
The created application was linked to a Github branch (Week5_Heroku_app_Deployment), where the application files were pushed to.
 - b. Build the application through the Heroku platform. After a successful build, the application was deployed under the following web address:
<https://dg-heroku-app.herokuapp.com>

dg-heroku-app.herokuapp.com/predict

Hello World

Gender

Male

Age

Enter Age

Time Spent on the Internet (minutes)

Enter time spent on the internet (minutes)

Time Spent on the Website (minutes)

Enter time spent on the Website (minutes)

Area Income

Enter Area Income

Submit

The person may not have clicked on the Ad

Age captured is: 30

Internet Time captured is: 324

Site time captured is: 224

Area Income captured is: 20000

Figure 2: Successfully deployed application on Heroku Platform

Results & Conclusion

The application runs successfully on the Heroku Platform & integrates well with the developed prediction model.

Github Link to Application Files:

https://github.com/teddywaweru/DataGlacier/tree/Week5_Heroku_app_Deployment

Link to Application:

<https://dg-heroku-app.herokuapp.com>