

Model Deployment Report

Name: G2M insight for Cab investment

Report date: 09/04/2022

Batch: LISUM05

Data intake by: Arnold Amusengeri

Data intake reviewer: N/A

Data storage location: https://github.com/ahnoamu/DG_work/Week_4

STEPS FOLLOWED FOR DEPLOYING MACHINE LEARNING MODEL THAT PREDICTS VIDEO GAME SALES

(i) Conduct EDA on vgsales data: Obtained video game sales data (vgsales.csv) from Kaggle (<https://www.kaggle.com/datasets/gregorut/videogamesales?resource=download>) and conducted exploratory data analysis.

(ii) Construct model and save as pickle file: Generated a regression model based on 4 features ('NA_Sales', 'EU_Sales', 'JP_Sales', 'Other_Sales'), and 1 label ('Global_Sales'). The model was evaluated and saved using pickle.

(iii) Generate prediction_app.py: This is a regular python file containing flask APIs which receives video game inputs through the user interface, then calculates the predicted Global sale value based on the regression model (reg_model.py). The file returns the predicted value.

(iv) Generate index.html: This is a HTML file containing the structure of the web page. The file points to a web styling file (static/style.css) and company logo (images/Original.svg). The file creates a web interface used to prompt and obtain user input, then displays the predicted sale to the user when the submit button is clicked.

(v) Create templates directory: Move the index.html to this directory. Flask will identify the html file in this directory.

(vi) Create static directory: Contains subdirectories /css, with style.css file for web styling, and /images with company logo file Original.svg

(vii) Deploy the model: Run prediction_app.py to prompt the start of Flask API. Copy and paste the URL provided to a web browser to view the home address. Key in values to obtain Global Video game sale prediction.

Snapshots

1. Server

```
* Serving Flask app "prediction_app"
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [10/May/2022 00:56:15] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [10/May/2022 00:56:15] "GET /static/css/style.css HTTP/1.1" 200 -
127.0.0.1 - - [10/May/2022 00:56:15] "GET /static/images/Original.svg HTTP/1.1" 200 -
127.0.0.1 - - [10/May/2022 00:56:15] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [10/May/2022 00:57:00] "POST /predict HTTP/1.1" 200 -
```


2. Web page

← → ↻ 127.0.0.1:5000/predict ☆

🔍 ⬇️ 🔍 📄 📁 📌 📌 📌 📌 📌

Predict Global VideoGame Sale

Predict

**Data Glacier**
Your Deep Learning Partner

Global videogame sale should be \$
287.99