**Data Glacier Internship**

**Week 5: Cloud and API deployment**

**Submitted by: Nahari Terena**

**Batch Code: LISUM15**

**Date: December, 2022**

# Introduction

Using the Flask framework, this project deploys machine learning model (SVM). The model predicts the spam or the ham comment on youtube’s videos.

In this project, we focus on two main points: building the model in Python, then create an API for the model, using Flask.

**Figure 1.Application Workflow**

# Dataset Information

The dataset was donated to UCI Machine Learning repository in March 2017. It is a public set of comments collected for spam research. Although it has five datasets composed by 1,956 real messages extracted from five videos that were among the 10 most viewed on the collection period, we only use three of them: Katy Perry, Eminem, and Shakira because they are favorite ones of this author.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Dataset** | **ID** | **Spam** | **Ham** | **Total Comments** |
| Psy | 9bZkp7q19f0 | 175 | 175 | 350 |
| Katy Perry | CevxZvSJLk8 | 175 | 175 | 350 |
| LMFAO | KQ6zr6kCPj8 | 236 | 202 | 438 |
| Eminem | uelHwf8o7\_U | 245 | 203 | 448 |
| Shakira | pRpeEdMmmQ0 | 174 | 196 | 370 |

Each dataset has five attributes which includes: ID, author, date, content and class.

# Building the Model

In the first part, we import some libraries necessary to aim the objective. Also, we checked the dataset and all the information read.

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

The dataset was split into two other groups: 30% for the test set and 70% for the training one. We fed our dataset into a Term Frequency-Inverse document frequency vectorizer (TF-IDF) which transforms words into numerical features for the two new datasets.

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Now, we may properly implement the machine learning model to classify each comment. For this purpose, we implement Support Vector Machine (SVM) using sckit-learn. Then, fit it into training dataset.

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamenteSubsequently, we analyzed the metrics of it. The, we save the model to use it in the html application.

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

# Web Application

This page consists of a simple web application with a form field so the user can type a message. Then, it will render a classification spam or ham comment. We have two HTML files, *home.html* and *results.html*.

**Figure 2. Home.HTML code**

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**Figure 3. Result.HTML code**

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

The *app.py* file contains the main code that will be executed by the Python interpreter to run the flask application, it includes the ML for classifying.

Texto

Descrição gerada automaticamente

Afterwards, we can run the API on the terminal.

Texto

Descrição gerada automaticamente com confiança média

# Scenarios on Web Application

The first scenario is the one there is a ham comment.

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Other possibility is the spam comment.

Interface gráfica do usuário, Aplicativo, Site

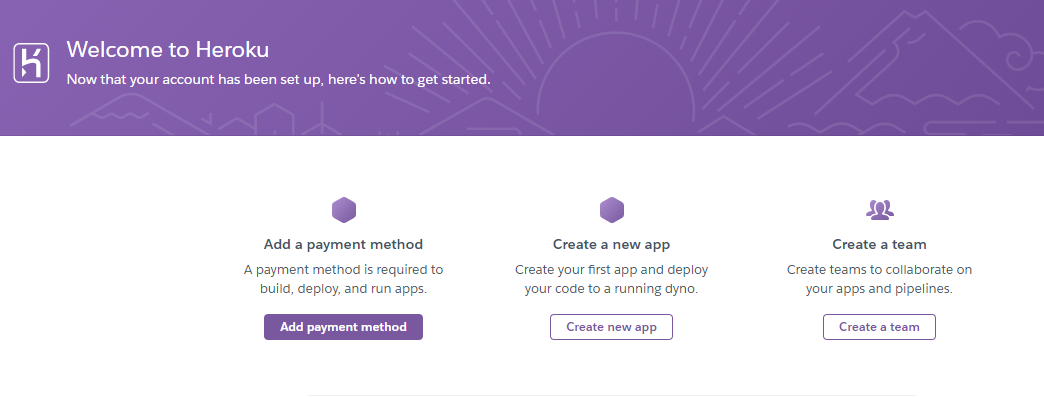
Descrição gerada automaticamente

So, it works just fine.

# Model deployment using Heroku

Thereafter that our model has been trained, the machine learning pipeline has been set up, and the application has been tested locally, we may start the Heroku deployment. We linked our GitHub repository to the Heroku account.

1. Heroku – creating a new app



1. Enter App name and region

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

1. Connect to GitHub repository.
2. Then, I choose the repository where the code is.
3. Deploy branch.
4. We must wait some minutes to get our application ready.