

STEPS OF DEPLOYMENT OF A SIMPLE MACHINE LEARNING MODEL ON FLASK

Name: Daniela Alvarez Zegarra

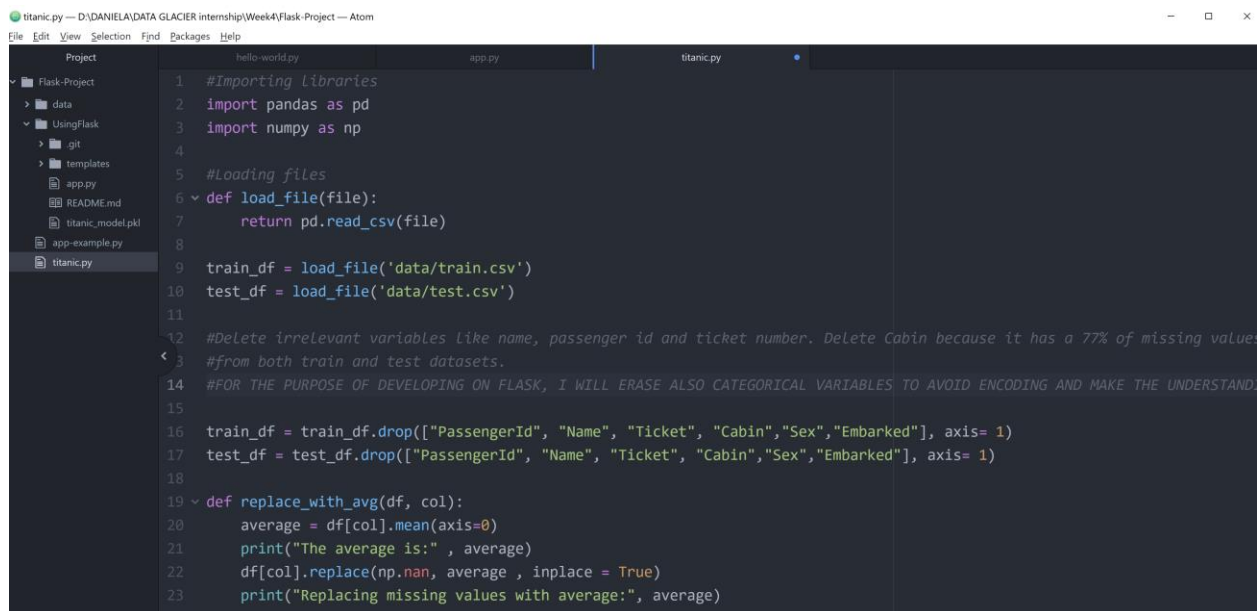
Batch code : LISUM03

Submission date: 20-09-2021

Submitted to: Github

Step 1

Using Titanic Dataset with only numerical variables.



The screenshot shows a code editor with a project explorer on the left and a code editor on the right. The project explorer shows a project named 'Flask-Project' with subfolders 'data', 'UsingFlask', '.git', and 'templates'. The 'data' folder contains 'train.csv' and 'test.csv'. The 'UsingFlask' folder contains 'app.py', 'titanic_model.pkl', and 'titanic.py'. The 'titanic.py' file is selected and its content is displayed in the code editor. The code imports pandas and numpy, loads the training and testing datasets, and drops irrelevant variables like 'PassengerId', 'Name', 'Ticket', 'Cabin', 'Sex', and 'Embarked'. It also defines a function to replace missing values with the average of the remaining numerical variables.

```
1 #Importing libraries
2 import pandas as pd
3 import numpy as np
4
5 #Loading files
6 def load_file(file):
7     return pd.read_csv(file)
8
9 train_df = load_file('data/train.csv')
10 test_df = load_file('data/test.csv')
11
12 #Delete irrelevant variables like name, passenger id and ticket number. Delete Cabin because it has a 77% of missing values
13 #from both train and test datasets.
14 #FOR THE PURPOSE OF DEVELOPING ON FLASK, I WILL ERASE ALSO CATEGORICAL VARIABLES TO AVOID ENCODING AND MAKE THE UNDERSTAND.
15
16 train_df = train_df.drop(["PassengerId", "Name", "Ticket", "Cabin", "Sex", "Embarked"], axis= 1)
17 test_df = test_df.drop(["PassengerId", "Name", "Ticket", "Cabin", "Sex", "Embarked"], axis= 1)
18
19 def replace_with_avg(df, col):
20     average = df[col].mean(axis=0)
21     print("The average is:", average)
22     df[col].replace(np.nan, average, inplace = True)
23     print("Replacing missing values with average:", average)
```

titanic.py — D:\DANIELA\DATA GLACIER internship\Week4\Flask-Project — Atom

File Edit View Selection Find Packages Help

Project

- Flask-Project
 - data
 - UsingFlask
 - .git
 - templates
 - app.py
 - README.md
 - titanic_model.pkl
 - app-example.py
 - titanic.py

```
25 replace_with_avg(train_df, "Age")
26
27 train_target = train_df["Survived"]
28 train_features = train_df.drop(["Survived"], axis = 1)
29
30 #Select algorithm
31 from sklearn.ensemble import RandomForestClassifier
32 clf_rf = RandomForestClassifier()
33 #Training
34 clf_rf.fit(train_features, train_target)
35
36 #Transformations to my test set
37 age_average = train_df['Age'].mean(axis=0)
38 print("The average is:" , age_average)
39
40 test_df['Age'].replace(np.nan, age_average , inplace = True)
41 print("Replacing missing values in test dataset with average:", age_average)
42
43 fare_average = train_df['Fare'].mean(axis=0)
44 print("The average is:" , fare_average)
45
46 test_df['Fare'].replace(np.nan, fare_average , inplace = True)
47 print("Replacing missing values in test dataset with average:", fare_average)
48
49 predictions = clf_rf.predict(test_df)
```

Step 2

Saving the model on disk

```
50
51 #Save the model to disk
52
53 import pickle
54
55 filename = "titanic_model.pkl"
56
57 pickle.dump(clf_rf, open(filename, 'wb'))
58
```

Step 3

Using Flask to make a web API for our ML model

```
app.py — D:\DANIELA\DATA GLACIER internship\Week4\Flask-Project — Atom
File Edit View Selection Find Packages Help

Project
  Flask-Project
    data
    UsingFlask
      git
      templates
      app.py
      README.md
      titanic_model.pkl
      app-example.py
      titanic.py

hello-world.py
app.py
1 import numpy as np
2 from flask import Flask, request, render_template
3 import pickle
4
5 app = Flask(__name__)
6 model = pickle.load(open('titanic_model.pkl', 'rb'))
7
8 @app.route('/') #http://www.google.com/
9 def home():
10     return render_template('index.html')
11 @app.route('/predict', methods=['POST'])
12 def predict():
13     '''
14     For rendering results on HTML GUI
15     '''
16     int_features = [int(x) for x in request.form.values()]
17     final_features = [np.array(int_features)]
18     prediction = model.predict(final_features)
19
20     output = round(prediction[0], 2)
21
22     return render_template('index.html', prediction_text='Would you survive? {}'.format(output))
23
24 if __name__ == '__main__':
25     app.run(port=5000, debug=True)
```

Step 4

Using a simple form with HTML and CSS to gather values and make a prediction.

```
index.html — D:\DANIELA\DATA GLACIER internship\Week4\Flask-Project — Atom
File Edit View Selection Find Packages Help

Project
  Flask-Project
    data
    UsingFlask
      git
      templates
      index.html
      app.py
      README.md
      titanic_model.pkl
      app-example.py
      titanic.py

hello-world.py
app.py
titanic.py
index.html
159 </style>
160 </head>
161 <body>
162     <div id="registration-form">
163         <div class='fieldset'>
164             <legend>Titanic Survival Prediction</legend>
165             <form action="{{ url_for('predict')}}" method="post">
166                 <input type="text" placeholder="Class (1 = 1st; 2 = 2nd; 3 = 3rd)" name='class' id='class' data-required="true" data-error-message="Your class is required">
167                 <input type="text" placeholder="Age" name='age' data-required="true" data-error-message="Your age is required">
168                 <input type="text" placeholder="Number of siblings/spouses aboard" name='sibsp' data-required="true" data-error-message="Your siblings/spouses aboard is required">
169                 <input type="text" placeholder="Number of parents/children aboard" name='parch' data-required="true" data-error-message="Your parents/children aboard is required">
170                 <input type="text" placeholder="Passenger Fare" name='fare' data-required="true" data-error-message="Your Passenger Fare is required">
171                 <input type="submit" value="predict">
172             </form>
173
174             <br>
175             <br>
176             {{ prediction_text }}
177
178         </div>
179     </div>
180 </body>
181 </html>
```

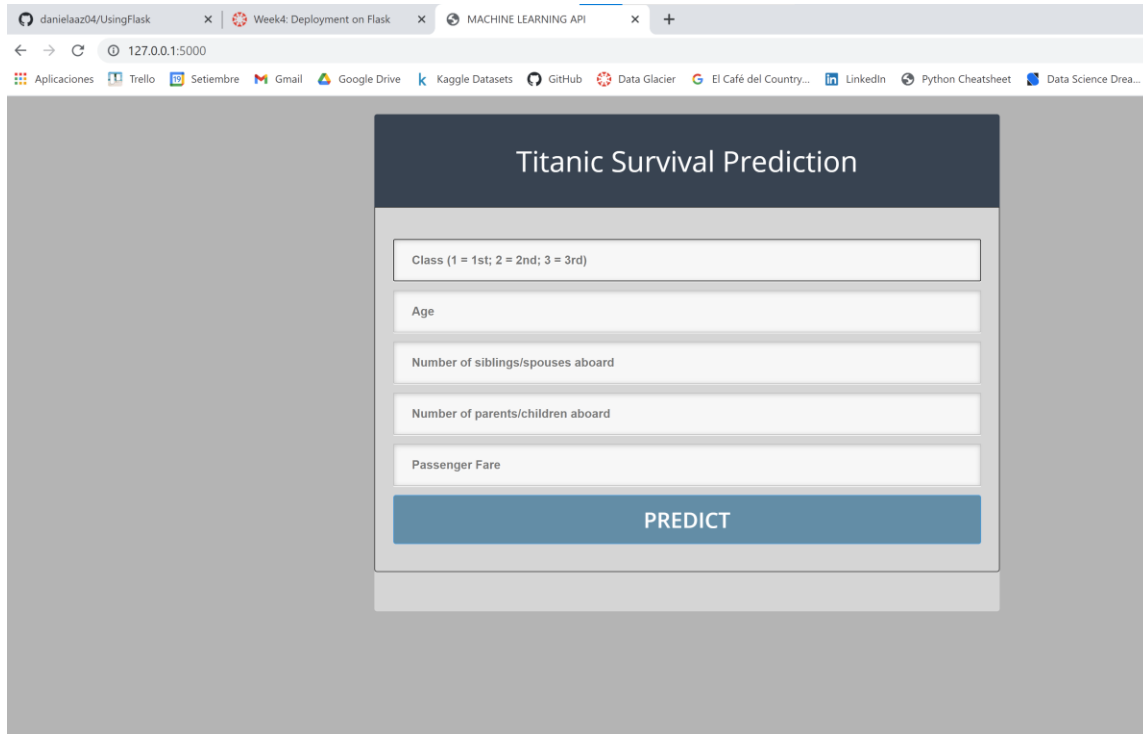
Step 5

After creating a repo on Github and cloning it to my computer, I move the necessary files to the repository and run my app.py file

```
danielaa04@DESKTOP-DANIELA: /mnt/d/DANIELA/DATA GLACIER internship/Week4/Flask-Project/UsingFlask
(base) danielaa04@DESKTOP-DANIELA:/mnt/c/Windows/System32$ cd
(base) danielaa04@DESKTOP-DANIELA:~$ cd ..
(base) danielaa04@DESKTOP-DANIELA:/home$ cd ..
(base) danielaa04@DESKTOP-DANIELA:/ $ cd mnt
(base) danielaa04@DESKTOP-DANIELA:/mnt$ cd d
(base) danielaa04@DESKTOP-DANIELA:/mnt/d$ cd DANIELA/
(base) danielaa04@DESKTOP-DANIELA:/mnt/d/DANIELA$ cd DATA\ GLACIER\ internship/
(base) danielaa04@DESKTOP-DANIELA:/mnt/d/DANIELA/DATA GLACIER internship$ cd Week4/
(base) danielaa04@DESKTOP-DANIELA:/mnt/d/DANIELA/DATA GLACIER internship/Week4$ ls
look-helloWorldApp  Flask-Project
(base) danielaa04@DESKTOP-DANIELA:/mnt/d/DANIELA/DATA GLACIER internship/Week4$ cd Flask-Project/
(base) danielaa04@DESKTOP-DANIELA:/mnt/d/DANIELA/DATA GLACIER internship/Week4/Flask-Project$ ls
UsingFlask  app-example.py  data  titanic.py
(base) danielaa04@DESKTOP-DANIELA:/mnt/d/DANIELA/DATA GLACIER internship/Week4/Flask-Project$ cd UsingFlask/
(base) danielaa04@DESKTOP-DANIELA:/mnt/d/DANIELA/DATA GLACIER internship/Week4/Flask-Project/UsingFlask$ ls
README.md  app.py  templates  titanic_model.pkl
(base) danielaa04@DESKTOP-DANIELA:/mnt/d/DANIELA/DATA GLACIER internship/Week4/Flask-Project/UsingFlask$ python app.py
* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
* Restarting with inotify reloaders
* Debugger is active!
* Debugger PIN: 974-093-224
```

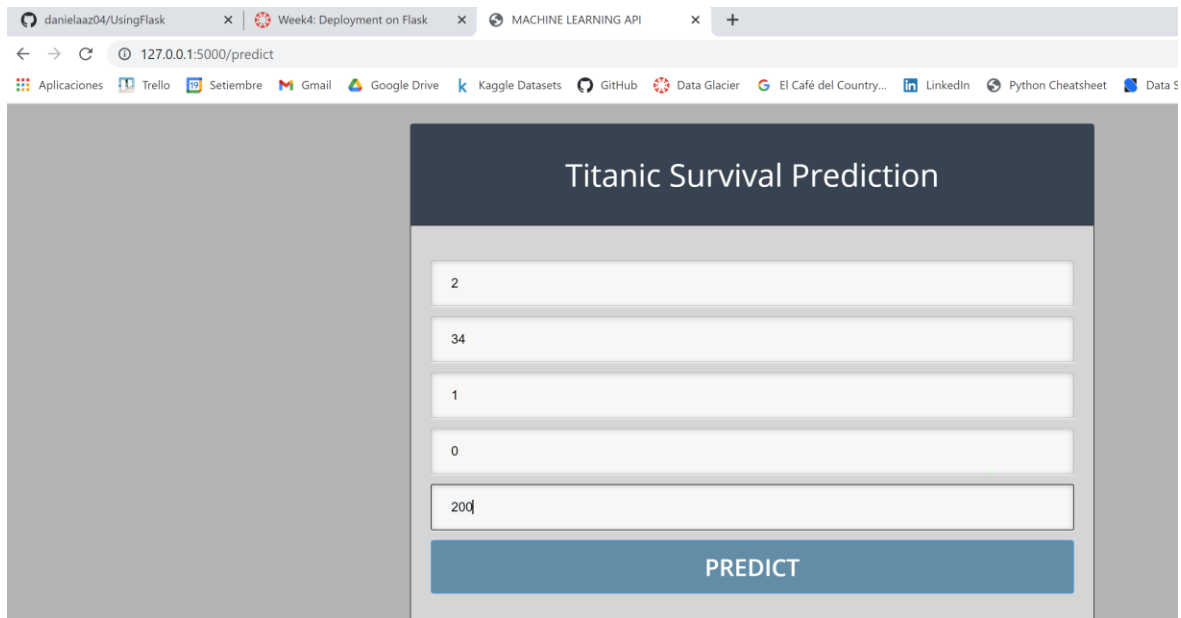
Step 6

In my browser I put <http://127.0.0.1:5000/> and get the form I created



The screenshot shows a web browser with the URL <http://127.0.0.1:5000/>. The browser tabs include "danielaa04/UsingFlask", "Week4: Deployment on Flask", and "MACHINE LEARNING API". The browser's address bar shows "127.0.0.1:5000". The browser's toolbar includes icons for "Aplicaciones", "Trello", "Setiembre", "Gmail", "Google Drive", "Kaggle Datasets", "GitHub", "Data Glacier", "El Café del Country...", "LinkedIn", "Python Cheatsheet", and "Data Science Drea...". The main content of the browser is a form titled "Titanic Survival Prediction". The form has a dark blue header with the title. Below the header, there are five input fields: "Class (1 = 1st; 2 = 2nd; 3 = 3rd)", "Age", "Number of siblings/spouses aboard", "Number of parents/children aboard", and "Passenger Fare". Below these fields is a blue button labeled "PREDICT".

Now I enter the values according to the question and press Predict button.



The screenshot shows the same web browser as the previous image, but the URL is now <http://127.0.0.1:5000/predict>. The browser tabs and toolbar are the same. The main content of the browser is the same "Titanic Survival Prediction" form, but the input fields now contain values: "2" for Class, "34" for Age, "1" for Number of siblings/spouses aboard, "0" for Number of parents/children aboard, and "200" for Passenger Fare. The "PREDICT" button is still visible below the input fields.

And get my prediction in the bottom.

Titanic Survival Prediction

Class (1 = 1st; 2 = 2nd; 3 = 3rd)

Age

Number of siblings/spouses aboard

Number of parents/children aboard

Passenger Fare

PREDICT

Would you survive? 0 (1=survived, 0=deceased)