Data Glacier

Your Deep Learning Partner

Week4: Deployment on Flask

Name: Rayhanul Islam Rumel
Batch Code: LISUM17
Submission Date: 28/01/2023
Submission To: Data Glacier

Agenda

Project Overview
Explore and Understand the Dataset
Preprocessing the dataset
Build the ML model
Building Webapp using Flask
User guide

Data Glacier

Your Deep Learning Partner

Project Overview

Project Overview

- **Dataset:** Spam text message classification
- **No of columns:** 2 (Message, Category)
- **ML Model:** Random Forest Classifier
- **Target:** We will input a text, the model will predict whether the text is spam or not.
- **Front-end:** HTML, CSS, and JavaScript
- **Back-end:** Flask

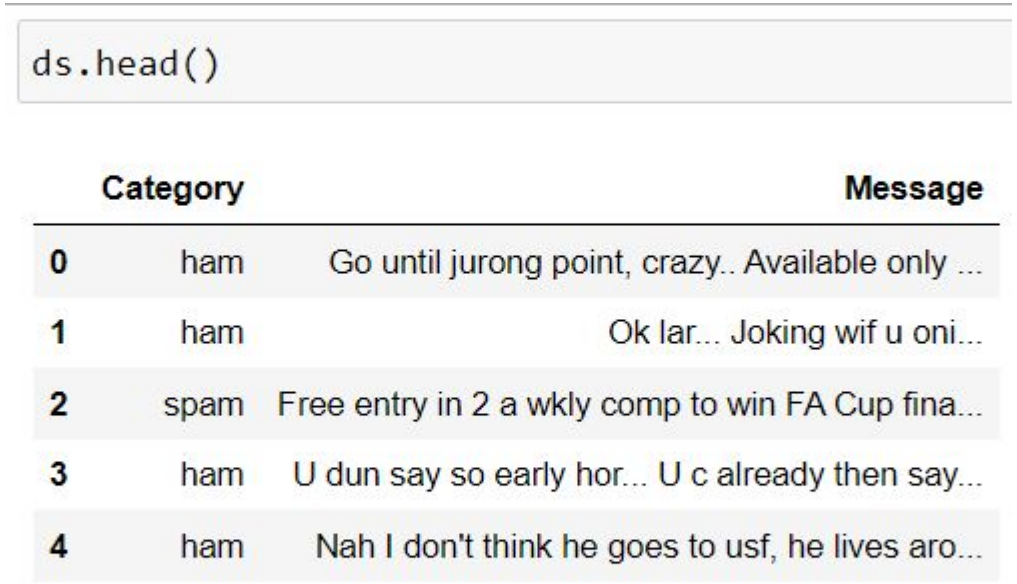
Data Glacier

Your Deep Learning Partner

Explore and Understand the Dataset

Explore and Understand the dataset

- The dataset has 2 columns: Category, Message
- The shape of the dataset is: 5572, 2
- The dataset contains no null or duplicate values

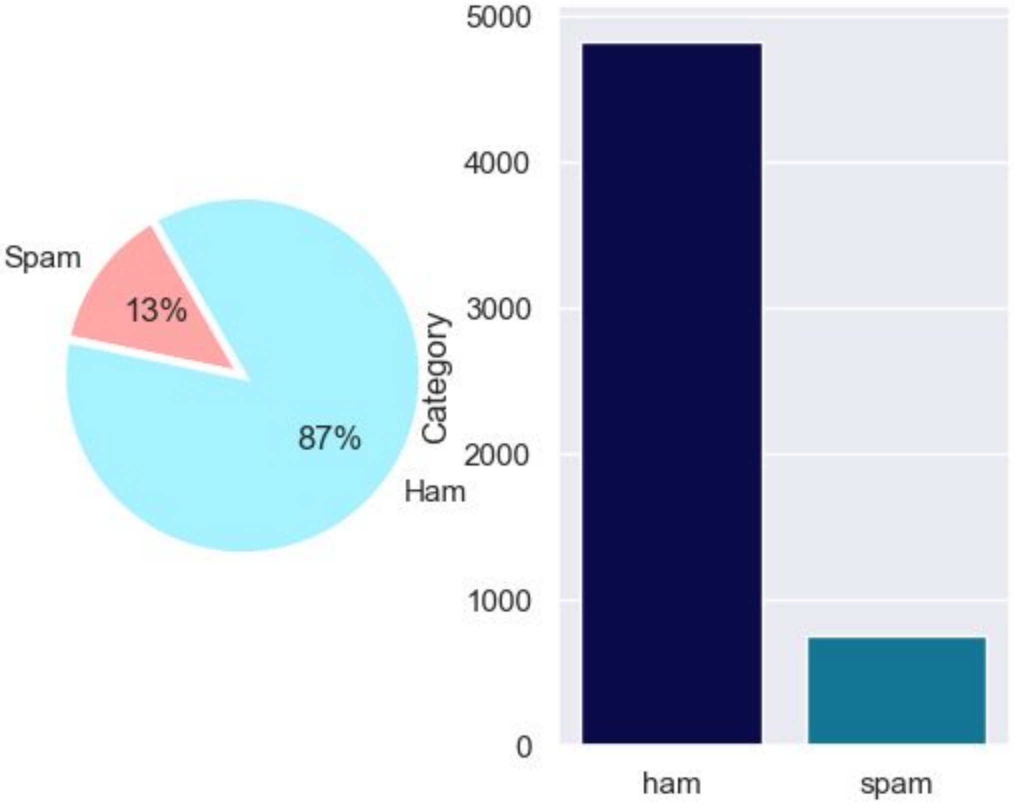


```
ds.head()
```

	Category	Message
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

Explore and Understand the dataset

- In total 13% of the total messages are spam and the rest (87%) is ham messages.



Data Glacier

Your Deep Learning Partner

Preprocessing the dataset

Preprocessing the dataset

- Before preparing the train and test set, we processed the Message column with the following steps:
  - Standardized the messages by converting all of the text to lowercase
  - Filtered all of the message in order to remove any hyperlinks available in the message
  - Removed punctuation from the message
  - Removed stop words from the corpus
  - Stemmed the words of each text

- Then I split the dataset into training set and test set:
  - 80% of the dataset is used as Training set, 20% of the dataset is used as Test set

- With TF-IDF I extracted the features of each message of the both train and test set as vectors

Data Glacier

Your Deep Learning Partner

Build the ML model

Build the ML model

- Then I defined random forest classifier:

```
rfc = RandomForestClassifier(n_estimators= 300)
```

- Once the model is defined, with train and test sets, I trained the model and evaluated the model accuracy
- The model accuracy is approximately 98%

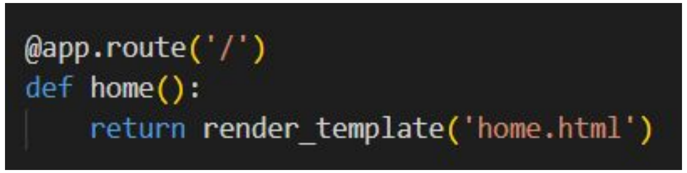
Data Glacier

Your Deep Learning Partner

Building Webapp using Flask

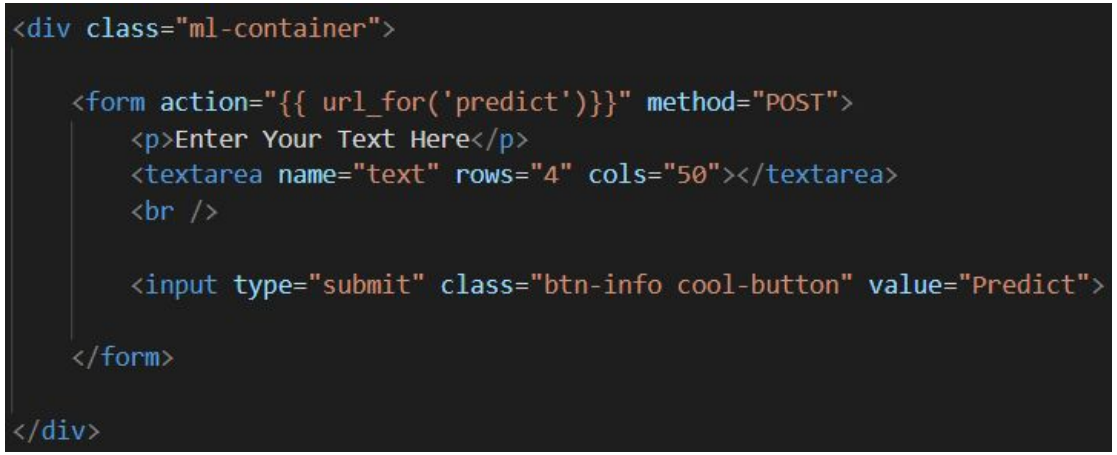
Building Webapp using Flask

- app.py : The controller that controls the app.
- Here I created two app routes:
  - **home**



```
@app.route('/')
def home():
    return render_template('home.html')
```

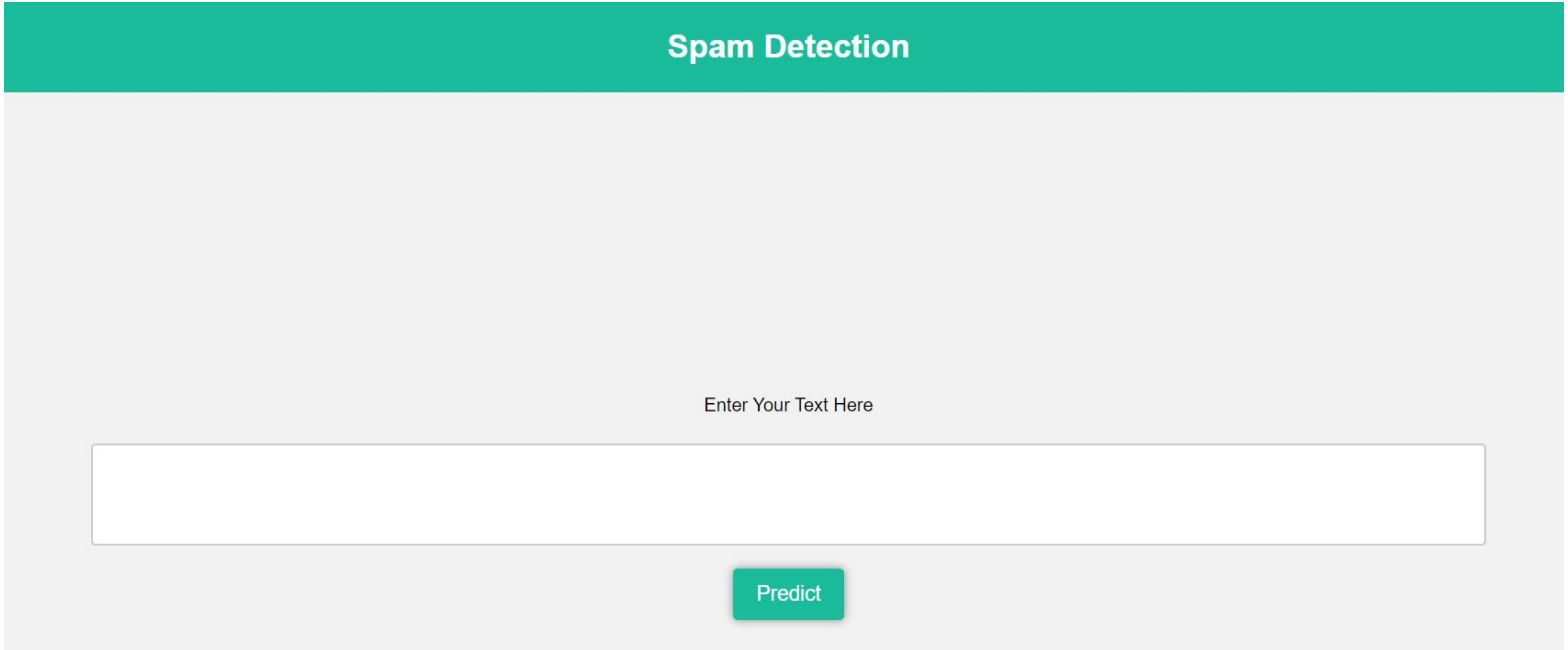
  - home renders an HTML file where I created a simple form with one text field and a submit button.
    - Users will type text message in the text field and would click on the *Predict* button



```
<div class="ml-container">

    <form action="{{ url_for('predict')}}" method="POST">
        <p>Enter Your Text Here</p>
        <textarea name="text" rows="4" cols="50"></textarea>
        <br />

        <input type="submit" class="btn-info cool-button" value="Predict">

    </form>

</div>
```

Building Webapp using Flask

- With CSS, the home.html looks like this:

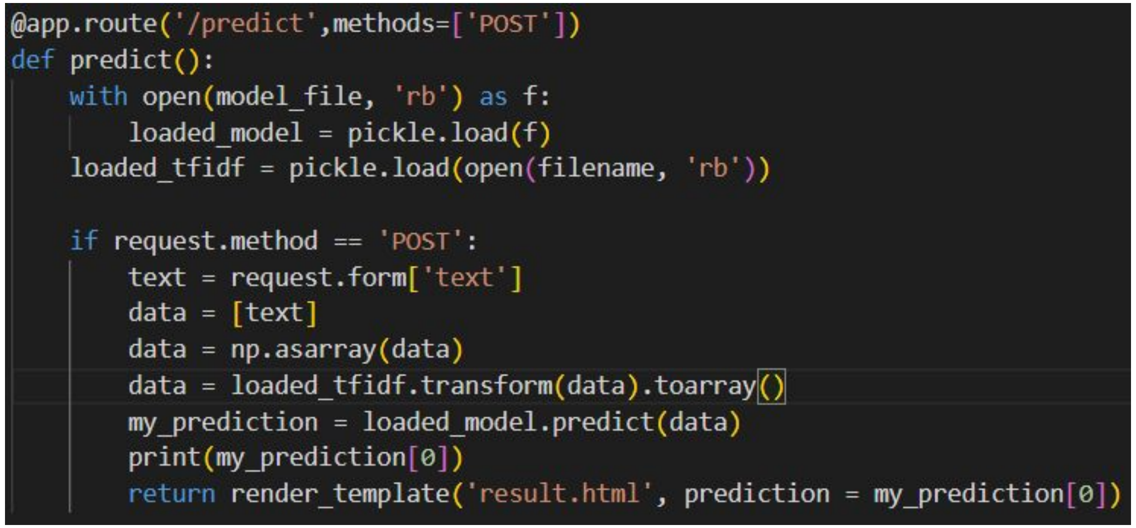


Spam Detection

Enter Your Text Here

Building Webapp using Flask

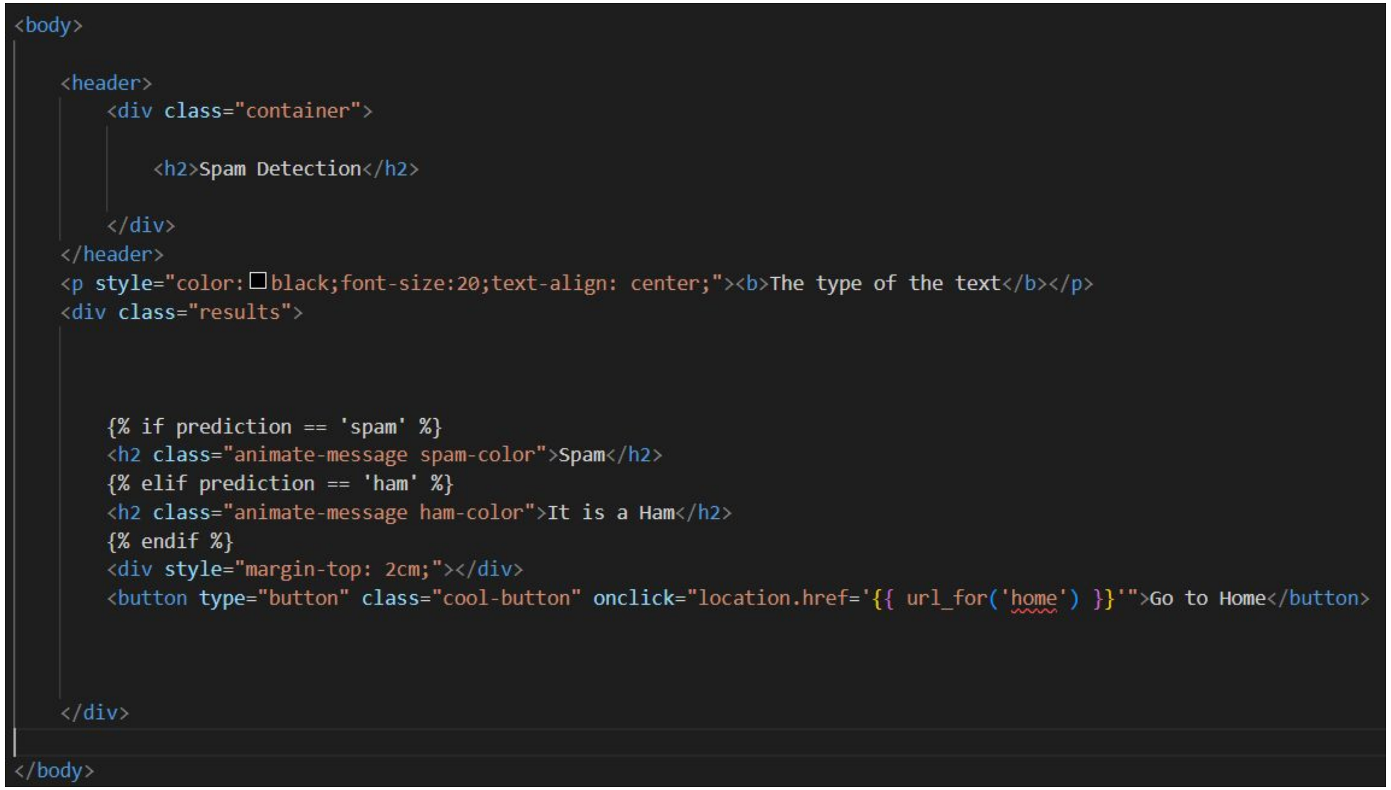
- app.py : The controller that controls the app.
- Here I created two app routes:
    - **predict** route, on the other hand loads the trained random forest classifier model
    - Receives the user submitted text from home.html
    - Extracts the features as vectors from the text using TF-IDF that was used during training phase
    - With the help of the trained model, predict the text
    - Forwards the predicted result to result.html



```python
@app.route('/predict',methods=['POST'])
def predict():
    with open(model_file, 'rb') as f:
        loaded_model = pickle.load(f)
    loaded_tfidf = pickle.load(open(filename, 'rb'))

    if request.method == 'POST':
        text = request.form['text']
        data = [text]
        data = np.asarray(data)
        data = loaded_tfidf.transform(data).toarray()
        my_prediction = loaded_model.predict(data)
        print(my_prediction[0])
        return render_template('result.html', prediction = my_prediction[0])
```

Building Webapp using Flask

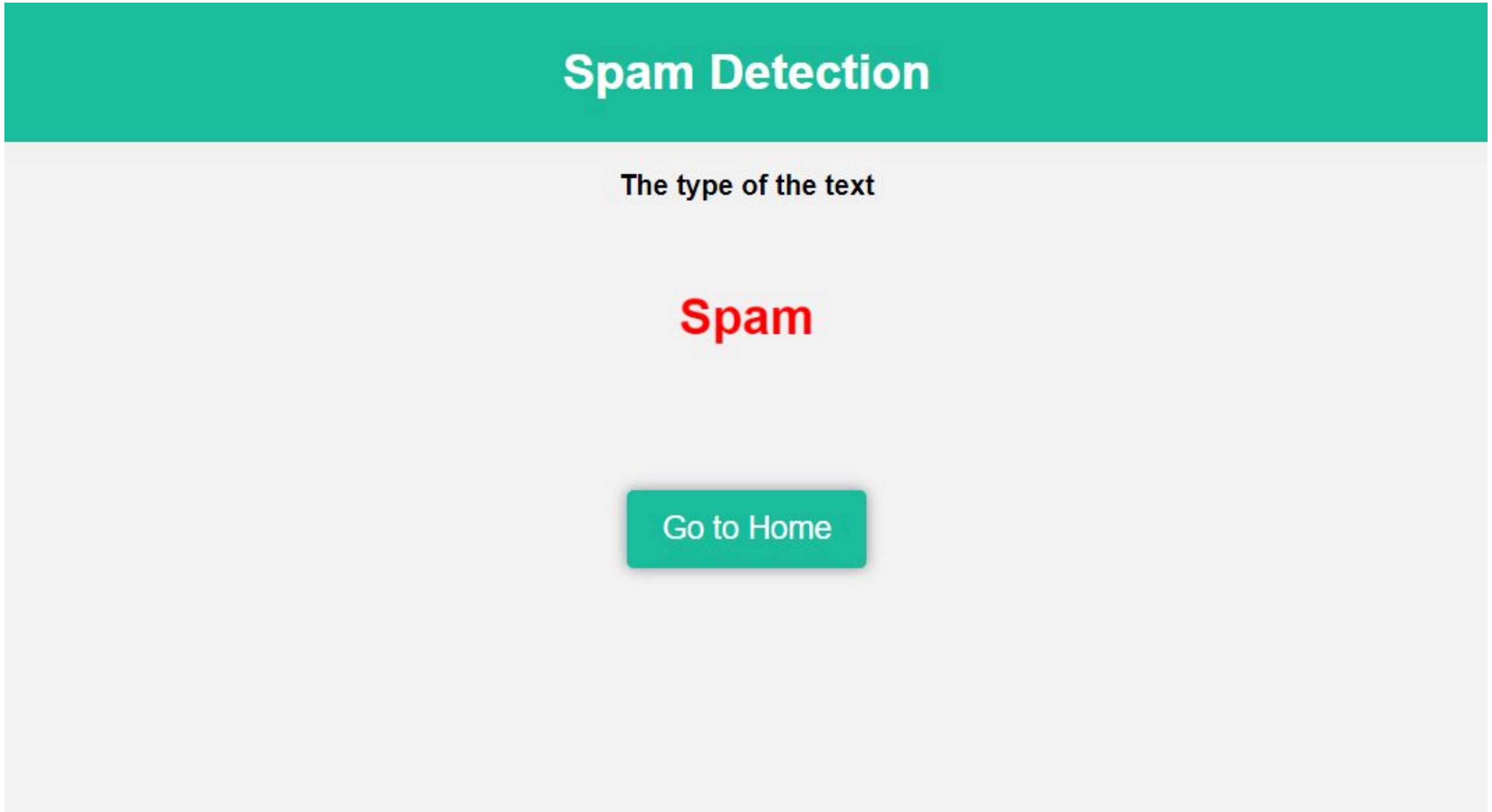
- result.html:
  - It receives the value of prediction from predict and prints whether the text is spam or ham
  - Once a user submits a text, he/she views this page the next



```
<body>

    <header>
        <div class="container">

            <h2>Spam Detection</h2>

        </div>
    </header>
    <p style="color:□black;font-size:20;text-align: center;"><b>The type of the text</b></p>
    <div class="results">



        {% if prediction == 'spam' %}
        <h2 class="animate-message spam-color">Spam</h2>
        {% elif prediction == 'ham' %}
        <h2 class="animate-message ham-color">It is a Ham</h2>
        {% endif %}
        <div style="margin-top: 2cm;"></div>
        <button type="button" class="cool-button" onclick="location.href='{{ url_for('home') }}'">Go to Home</button>



    </div>

</body>
```

Building Webapp using Flask

- With CSS, the result.html looks like this:



Spam Detection

The type of the text

Spam

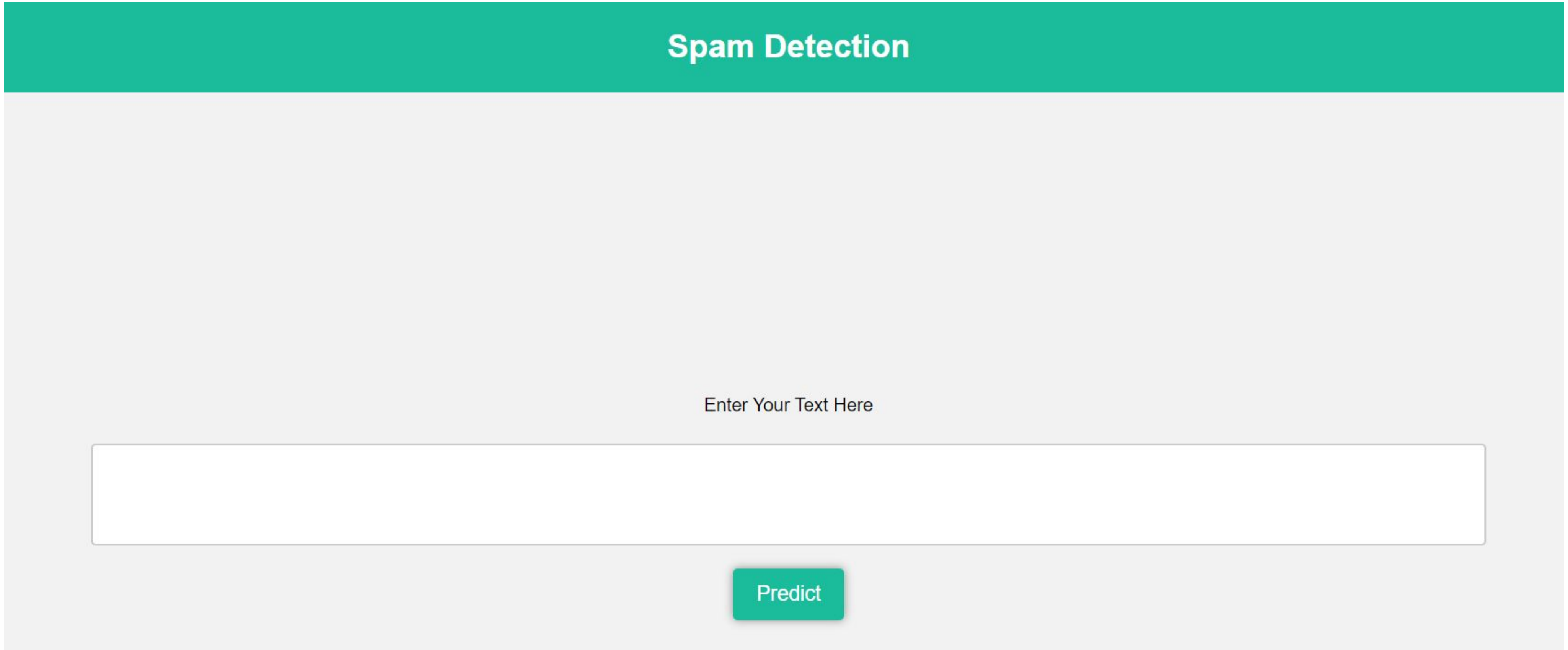
Data Glacier

Your Deep Learning Partner

User Guide

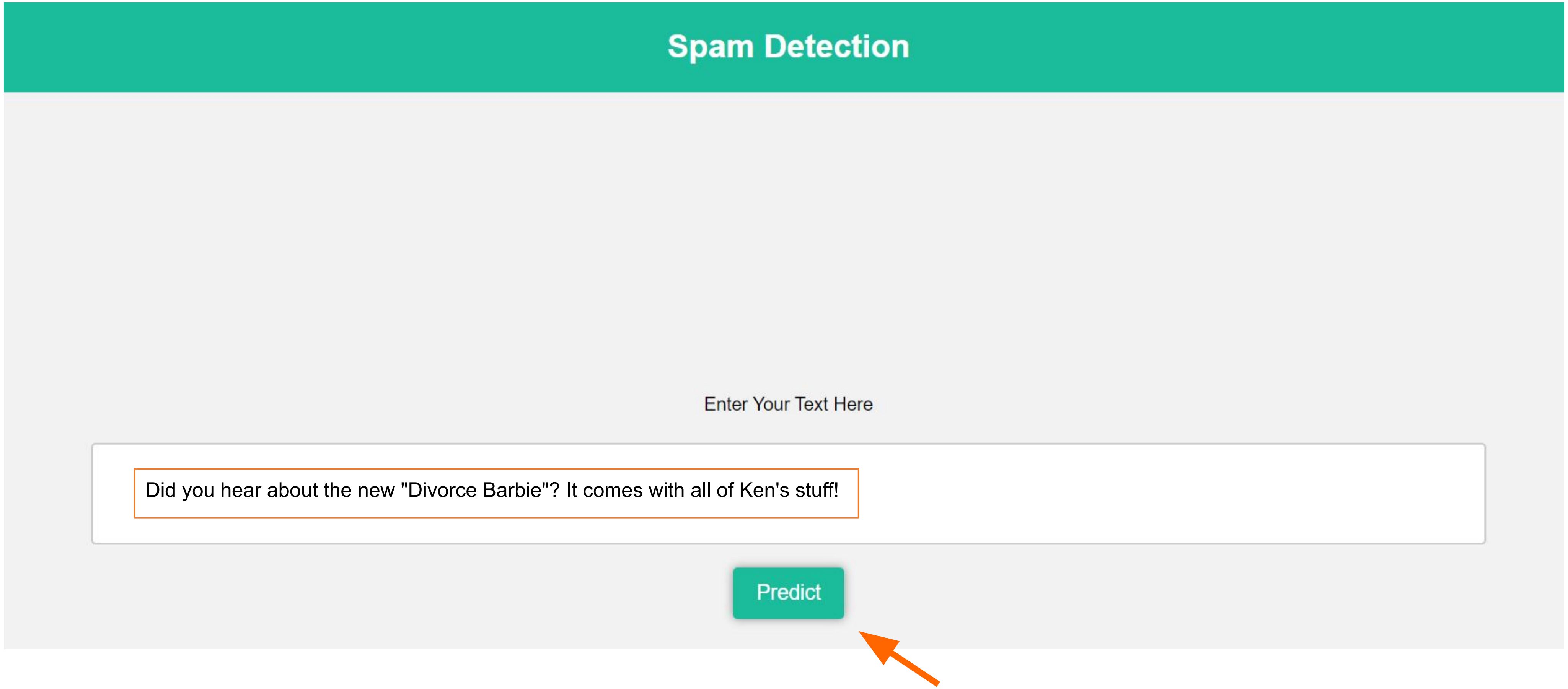
User Guide

- After visiting the homepage of Spam Detection, you would see the following user interface.



User Guide

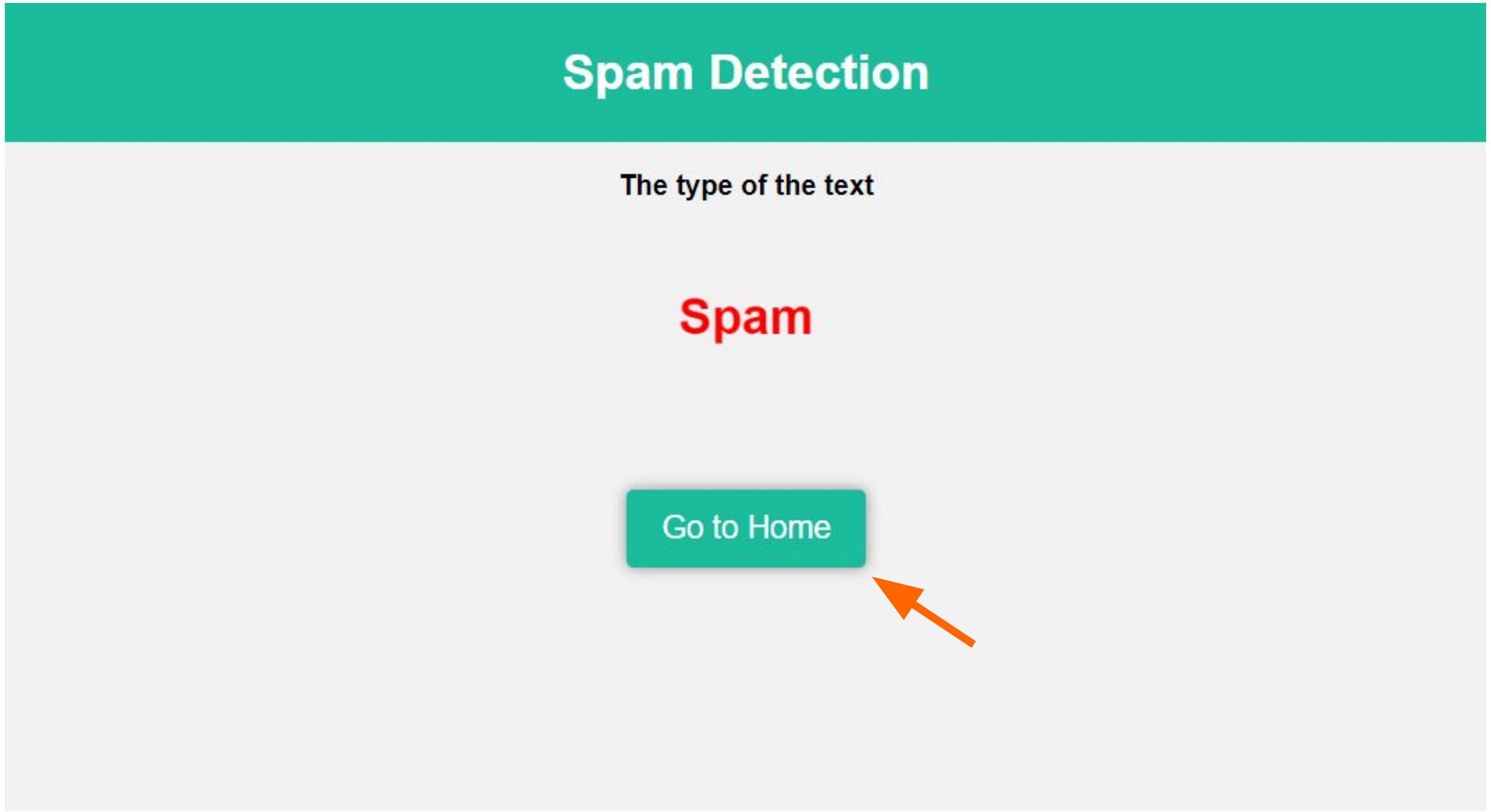
- Type your text in the text field and click on the 'Predict' button.



Did you hear about the new "Divorce Barbie"? It comes with all of Ken's stuff!

Building Webapp using Flask

- The app will redirect you to the next page and will show you the type of text you entered.
- You can click on ‘Go to Home’ button to visit the homepage of the app.



Spam Detection

The type of the text

Spam

Thank You

Data Glacier

Your Deep Learning Partner