# ECE444-Lab8: Deploying ML model to Cloud

**Scenario:**

Fake news has recently become a topic of concern as more of the information we receive about the world is delivered through the web. You, as a developer on the Fake News detection team at Not Fake News Co. have been tasked with building a barebones REST API that takes in a snippet of text (from a news article or source) and determines if it is considered fake news or not (by returning a 1 if it is Fake News, 0 otherwise).

**Objective:**

The objective of this lab assignment is for students to get familiar with deploying machine learning models to the cloud using a cloud provider. We will focus on deploying a model using Heroku, making the model accessible as a service using a REST API call.

**Instructions:**

**Setting Up REST API**

First, set up a Flask App that you can host on AWS using the following guide: <https://medium.com/swlh/deploy-a-machine-learning-model-with-aws-elasticbeanstalk-dfcc47b6043e>

For this task, make sure you have registered an account with AWS, which will give you 750 Hours of compute for free per month, which should be more than enough for this lab. You can register for AWS here: <https://aws.amazon.com/free/>

Once you have set up your AWS elastic beanstalk application, modify your script and add a GET request to the flask app that calls a custom defined function. With this function you will take some text as input, run it through a model, and return the output label (ie. 1 if it is Fake News, 0 otherwise).

For this project, you will not have to train your own model. You can load a trained Fake News detector model by adding the following code snippet in your flask function after adding the 2 included pickle files in your project directory:

Note: you will need to install scikit-learn for this code snippet to work.

A screenshot of a computer

Description automatically generated with medium confidence

Note that you can choose another model of your choice for this task, but you will need to use a model that is lightweight enough to be run on the free tier t3.micro environments used in the medium article.

**Testing Your API**

Once you have implemented the REST API with the model, you will need to test it. There are multiple ways of testing this app, we ask that you implement the following:

1. Functional Tests: Test your API by creating 4 test inputs (two fake news and two real news) and run these test cases with your API to ensure that it works.

2. Since this is a service that will be run live, it is important to know the latency of your API. Take the previous test cases and do 100 API calls to the REST server with your examples. You should record the average time it takes for you to call the API and get a response. Report this result.

**Evaluation (total 10 Marks):**

1. **(Marked out of 6)** On Quercus, submit a link to your GitHub Repository containing the code for your API. You will be evaluated on code quality and functionality. In essence, your code should be readable and function correctly.

2. **(Marked out of 4)** In your submission to Quercus, also include a PDF file detailing each of your 4 test cases, the expected response from the API, and the actual response you received from running your test. Include the average time it takes for you to call the API in this file. Naming format: ECE444-Lab7-[YourFullName].pdf). An example of what one test case would look like is provided below:

|  |  |
| --- | --- |
| Test Input | “Emperor of Mars has declared war on Earth” |
| Expected Response | “1” |
| Actual Response | “1” |
| Average Latency Over 100 Calls | 12.3 ms |