UDACITY

‹ Return to "Deep Learning" in the classroom

Deploying a Sentiment Analysis Model

REVIEW

CODE REVIEW 1

HISTORY

Meets Specifications

Congratulations

Great submission!

All functions were implemented correctly and the final algorithm seems to work quite well.

Suggesting some further actions:

- The provided web app in this project is very simple and there is plenty of room for improvement if you wish to stretch your web developer skills.
- The model chosen here is a straightforward RNN with a single hidden layer.
- There are many different model architectures that you could try to see if they improve the results.

Files Submitted

The submission includes all required files, including notebook, python scripts and html files.

Good work!
The submission included all required files: notebook, python scripts and html files.

Preparing and Processing Data

Answer describes what the pre-processing method does to a review.

Awesome!
Nice work describing what the pre-processing method does to a review.

You really understood what is being done to the input.
The **review_to_words()** function removes words which contain very little content, like 'the' (it removes stopwords).

The `build_dict` method is implemented and constructs a valid word dictionary.

Notebook displays the five most frequently appearing words.

Good job!
The five most frequently appearing words: **['movi', 'film', 'one', 'like', 'time']**

Answer describes how the processing methods are applied to the training and test data sets and what, if any, issues there may be.

Nice work describing how the processing methods are applied to the training and test data sets.

The **preprocess_data()** is applied per record and so there is no issue.
The reason that **convert_and_pad_data()** doesn't cause an issue is that **word_dict()** is constructed using only the training data.

It is important to understand that care needs to be taken when processing data so as not to accidentally introduce leakage.

Build and Train the PyTorch Model

The train method is implemented and can be used to train the PyTorch model.

Good work!
Your training method worked very well for 5 epochs on a subset of the training data.

The RNN is trained using SageMaker's supported PyTorch functionality.

The **train.py** file was completed correctly using the your implementation of the train method.
the **estimator.fit()** cell was executed and displayed information about the training progress.

Deploy the Model for Testing

The trained PyTorch model is successfully deployed.

Use the Model for Testing

Answer describes the differences between the RNN model and the XGBoost model and how they perform on the IMDB data.

Good job
Nice work describing the differences between the RNN model and the XGBoost model.

The test review has been processed correctly and stored in the `test_data` variable.

You have correctly applied both **review_to_words()** and **convert_and_pad()** to the test review.
It was executed successfully (calling **predict()** on the processed review).

The `predict_fn()` method in `serve/predict.py` has been implemented.

Deploying the Web App

The model is deployed and the Lambda / API Gateway integration is complete so that the web app works (make sure to include your modified `index.html`).

Awesome!
The index.html file has been edited correctly.

action="https://feqguye608.execute-api.us-east-2.amazonaws.com/prod"

Answer gives a sample review and the resulting predicted sentiment.

Good job
Nice work describing the sample review and the resulting predicted sentiment.

DOWNLOAD PROJECT

1 CODE REVIEW COMMENTS

RETURN TO PATH

Rate this review