Big Data and Intelligent Analytics - Spring Semester 2021

INSTRUCTOR: Sri Krishnamurthy
analyticsneu@gmail.com

Assignment 2

In this assignment, you will build a sentiment analysis micro service that could take a new Edgar file in json format and generate sentiments for each statement in the referenced EDGAR file.

To build this service, you need a Sentiment analysis model that has been trained on “labeled”, “Edgar” datasets. Note that you need to have labeled data which means someone has to label the statements and you need to use EDGAR datasets since you want the ML service to be optimized for domain-specific datasets.In order to accomplish this, we need to design 3 pipelines. **You will have to use one of the (Dask, Luigi, Airflow, Sklearn) pipelining tools to define your pipelines.**

Part 1: Annotation pipeline(Choose Dask/Airflow/Luigi/Sklearn)

The annotation pipeline will ingest a minimum of 50 earning call files from various companies and label them.

Example:

Company	Year	Filing
1018724	**2013**	**425**

1. Ingestion: You can use the data provided to you to ingest files and stage them in a Google/S3 bucket ; You are free to use Seeking Alpha to ingest the earnings call transcripts if you like for new data!
2. Pre-process each file to remove white spaces and special characters. Each earnings call file is now a list of sentences.

3. You will now have to label these lines with sentiment scores. Use text analytics APIs from these services. Use the api assigned to your team number:
    1. Amazon Comprehend (https://docs.aws.amazon.com/comprehend/latest/dg/get-started-api-sentiment.html#get-started-api-sentiment-python)
    2. IBM Watson (https://cloud.ibm.com/apidocs/natural-language-understanding/natural-language-understanding)
    3. Google (https://cloud.google.com/natural-language/docs/sentiment-tutorial)
    4. Microsoft (https://docs.microsoft.com/en-us/azure/cognitive-services/text-analytics/how-tos/text-analytics-how-to-sentiment-analysis?tabs=version-3)
4. You will then normalize the scores to a scale of -1-1, -1 negative , 1 positive
5. The output is a labeled dataset in csv format which would look like this. (Note: You will have one column not four)

Statements	Amazon	Google	IBM	Microsoft
The quarter was terrible. We lost a lot of money	**-0.8**	**-0.8**	**-0.7**	**-0.9**
We had a wonderful quarter!	**0.9**	**0.8**	**0.8**	**0.7**

Part 2: Training Pipeline(Choose Dask/Airflow/Luigi/Sklearn)

Now that we have labeled datasets, we will have to build models to train a model that will generate sentiment scores.

5. Model : Use the Tensorflow approach discussed in class and in [2] to use transfer learning and on the labeled dataset you created instead of the IMDB dataset to fine tune the model.
6. Save the model on to a Bucket.

7. Your pipeline should be configurable so that it will generate the Model based on inputs and save the trained model to a Bucket.

Part 3: Model Serving microservice:

We are now ready to deploy the model.

1. You will design one microservice using Flask and the associated docker containers both exposing an API to take each sentence as the json Input and return the sentiment for each sentence.
2. Develop a Docker container to serve an API that will take a JSON of a sentence and return back the predictions as detailed in [2]
3. Write unit tests to test your api (See https://github.com/Harvard-IACS/2020-ComputeFest/blob/master/notebook_to_cloud/ml_deploy_demo/Makefile)

Input:

```
{"data": ["this is the best!", "this is the worst!"]}
```

Output:

```
{
    "input": {
        "data": [
            "this workshop is fun",
            "this workshop is boring"
        ]
    },
    "pred": [
        [
            0.9856576323509216      # closer to 1 => positive
        ],
        [
            -0.19903425872325897     # closer to -1 => negative
        ]
    ]
}
```

Part 4: Inference Pipeline (Choose Dask/Airflow/Luigi/Sklearn):

1. Build an inference pipeline that takes a csv file with the input in the format below

   Input

Company	**Year**	**Filing**
1018724	**2013**	**425**

2. The pipeline should dynamically get the file from EDGAR, pre-process the file,create a list of sentences for inference. Use the code you developed for the training pipeline for this.

3. Jsonify the sentences to look something like this and invoke the ~~2~~ service~~s~~ you created to get back the sentiments

   ```
   Example inputs: {"data": ["this is the best!", "this is the worst!"]}
   ```

4. Format the output to a csv file and store it to a bucket.

Statements	**Sentiment (Model 1)**	**~~Sentiment (Model 2)~~**
The quarter was terrible. We lost a lot of money	**Negative**	**~~Negative~~**
We had a wonderful quarter!	**Positive**	**~~Positive~~**

References:

1. https://github.com/microsoft/nlp-recipes/blob/master/examples/sentiment_analysis/absa/absa.ipynb
2. https://github.com/Harvard-IACS/2020-ComputeFest/tree/master/notebook_to_cloud
3. https://www.tensorflow.org/tutorials/keras/text_classification_with_hub

Deliverables (Due March 26th 11.59am):

1. A 2-5 page report in https://github.com/googlecodelabs/tools format to illustrate your understanding of how various steps are integrated in the pipeline.
2. A fully functional pipeline that when run generates the output in a google bucket.

3. You will be given 10 minutes to present your company analysis in class on March 26th.
4. A github repo link