SENTIMENT ANALYSIS ON GCP

Terrence E. White

SEIS 738 Data Science Project

Jigsaw Unintended Bias in Toxicity Classification
Detect toxicity across a diverse range of conversations

TABLE OF CONTENTS

Abstract

This project explores two services available on the Google Cloud Platform (GCP) for performing sentiment analysis: Natural Language API and AutoML Natural Language. The former provides powerful prebuilt models that can be invoked as a service allowing developers to perform sentiment analysis, along with other features like entity analysis, content classification, and syntax analysis. The latter allows developers to train more domain specific models whenever the pre-trained models offered by the Natural Language API are not sufficient. Experiments have been conducted with both of these offerings and the results are presented herein.

SENTIMENT ANALYSIS ON GCP

This section covers all of the key topics that relate to the data used in the project including how the data was labeled and how the data was cleaned to address issues.

BACKGROUND OF DATA

This project takes inspiration from two related Kaggle competitions on sentiment analysis. The first competition, called "Toxic Comment Classification Challenge", provides data from a large number of Wikipedia comments which were labeled by human raters for toxic behavior. The second competition, called "Jigsaw Unintended Bias in Toxicity Classification", leverages data collected from the now defunct "Civil Comments" platform, which was a full-featured commenting plugin for independent news sites. Labels for the data were generated by the Civil Comments users who were motivated to provide ratings of the comments by the platform's peer-review submission process.

Kaggle performed the original sourcing of the data to support both competitions and in turn makes the data publicly available for research purposes. The data used in this project was retrieved directly from the Kaggle website.

DATA PREPROCESSING

The datasets retrieved from the Kaggle competitions referenced above provide over 1.5 million samples of labeled comments for sentiment analysis. This section outlines the preprocessing steps taken to prepare the data for analysis on the GCP.

1. Data cleansing of the original data:
    a. End-of-line characters were embedded in many of the comment strings, which caused parsing errors with the file. This issue was addressed using a series of UNIX command line tools to temporarily mark the true end-of-line location, then removing all end-of-line and carriage return characters from the entire file, and then finally replacing the temporary marker with the true end-of line character.
    b. All embedded double quotes were removed from comments to avoid parsing errors with the eventual .csv file that would be used for the AutoML model training.
    c. All non-English characters were removed from the comment strings. Neither of the GCP sentiment analysis services support non-English charter sets.
2. Addressing unbalanced data:
    a. The original data contained the following counts of target classes:
    Total count of positive sentiment messages: 1,675,189
    Total count of negative sentiment messages: 104,796
    b. To support the AutoML Natural Language model training, it was required to have a similar number of samples of messages in both classes (positive and negative message sentiments).
    c. Under-sampling was performed on both classes of messages to produce the following breakdown of messages:
    Total positive sample count: 50,000
    Total negative sample count: 50,000

NOTE: GCP has a limit of 100,000 records for performing AutoML Natural Language model training.

3. Addressing duplicate data:
    a. Unfortunately, duplicate comments were found in the data. A Python routine was developed to eliminate all duplicate message strings to protect the integrity of the separate data sets (train/validation/test) which will be discussed in item 5 below.
4. Interpolating toxicity scores:
    a. The original data contains target scores ranging from 0 to 1, and were fractional values representing the percent of human raters who believed toxic sentiment applied to the given comment. A target >= 0.5 is considered to be in the positive class (toxic)
    b. The Natural Language API uses sentiment ranges between -1.0 (toxic) and 1.0 (positive) with 0.0 representing relatively neutral text.
    c. The AutoML Natural Language offering used a flexible scale starting from 0 (toxic) up to a maximum sentiment level (positive or non-toxic sentiment) defined by the user depending on how granular the assessment is required to be.
    d. For this project all target labels in the data were interpolated onto a scale of 0 (toxic) to 1 (non-toxic) in accordance with the most basic AutoML scale. This was accomplished using the numpy.interp() function.
5. Splitting the data:
    a. The GCP AutoML offering provides an option to label each record as “TRAIN”, “VALIDATION”, or “TEST” when the user wants to assert control over how each record would be used. This option was ideal for this project because the intention was to test both GCP sentiment analysis offerings with the same set of records. With this in mind, the 100,000 records described above were further split using an 80/10/10 ratio as shown below:
    Train record count: 80,000 [positive:40,000 and negative 40,000]
    Validation record count: 10,000 [positive:5,000 and negative 5,000]
    Test record count: 10,000 [positive:5,000 and negative 5,000]

SENTIMENT ANALYSIS ON GCP

EXPERIMENT 1: NATURAL LANGUAGE API

The section describes the process taken to execute the Natural Language API and the resulting quality analysis of the results.

EXECUTING NATURAL LANGUAGE API

As previously stated, the Natural Language API provides powerful prebuilt models that can be invoked as a service allowing developers to perform sentiment analysis, therefore training and validation data were not required for this first experiment. Only the records tagged “TEST”, as described above in the “Splitting the data” section, were used in this experiment:
Test record count: 10,000 [positive:5,000 and negative 5,000]

The GCP provides multiple APIs for invoking the natural language service for sentiment analysis, and Python was selected for use in this project.

The following high-level steps were required:

1. Logon to the GCP console to:
    a. Create a project
    b. Enable the Google Natural Language API
    c. Create a service account
    d. Download a private key as a JSON file and set the environment variable GOOGLE_APPLICATION_CREDENTIALS to the file
2. Install the google-cloud-language client library into a virtual Python environment.
3. Execute sentiment analysis using the Python API. The salient portions of the API are highlighted below:

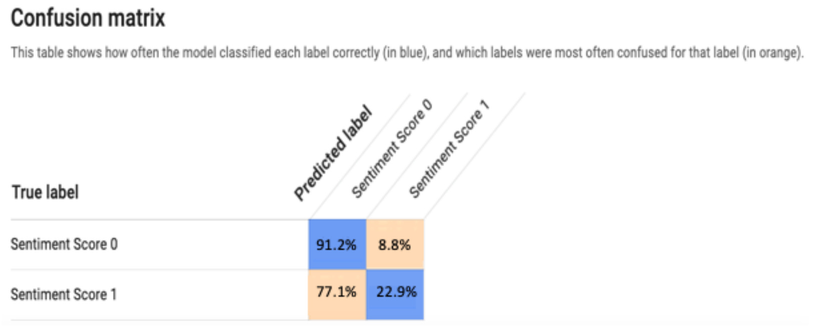
```
# Wrap the comment string in a Document object
document = types.Document(content=comment_text, type=enums.Document.Type.PLAIN_TEXT)


# Detect the sentiment of the Document
sentiment = client.analyze_sentiment(document=document).document_sentiment
```

NATURAL LANGUAGE RESULTS

The results of the Python API invocation described above were compared to the labeled test data to determine the overall quality of the sentiment analysis performed by the GCP Natural Language API. The results are presented in this section.

1. The confusion matrix displayed in Figure 1 below indicates that the Natural Language API has a particularly tough time with false positives, i.e. non-toxic comments (sentiment 1) were incorrectly classified as being toxic (sentiment 0). Almost 40% of the predication fell into this category. Also, the overall accuracy was only 57%.



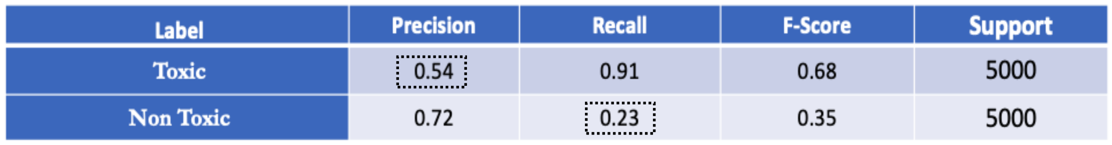
Confusion matrix

This table shows how often the model classified each label correctly (in blue), and which labels were most often confused for that label (in orange).

True label \ Predicted label	Sentiment Score 0	Sentiment Score 1
Sentiment Score 0	91.2%	8.8%
Sentiment Score 1	77.1%	22.9%

Figure 1: Confusion Matrix:

2. The following table outlines other key metrics describing the quality of the sentiment analysis. Looking at the numbers, we can see that the precision score for toxic messages is low, which explains the large number of false positives in Figure 1. Also, the model very rarely predicts a message to be non-toxic, as shown by the low recall score.



Label	Precision	Recall	F-Score	Support
Toxic	0.54	0.91	0.68	5000
Non Toxic	0.72	0.23	0.35	5000

Figure 2: Natural Language Quality Metrics

3. One final graphic describing the quality of the model is shown below. The low area under the curve (AUC), shows the low quality of the predictions of the model.

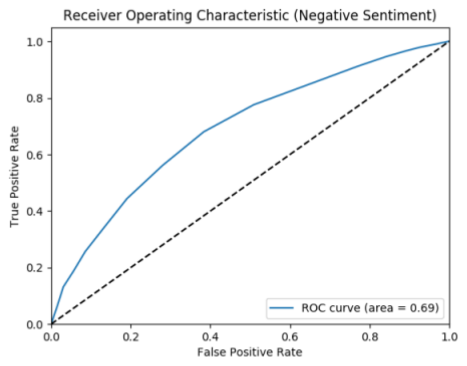


Figure 3: ROC Curve for Negative Sentiment

EXPERIMENT 2: AUTOML NATURAL LANGUAGE

The goal of the AutoML Natural Language offering is to enable users to train custom models in cases where the Natural Language API does not perform well. This can happen with challenging datasets, especially those involving domain specific terms and phrases. This section describes the process taken to train the AutoML Natural Language custom model and the quality of the results.

TRAINING AN AUTOML CUSTOM MODEL

When training a custom AutoML model, labeled data is required. In this experiment, we use all of the prepared data previously tagged as “TRAIN”, “VALIDATION”, and “TEST”. The AutoML service will automatically train a model using the training and validation data. After the training is complete, the service uses the testing data to determine the true quality of the sentiment analysis and makes various quality metrics available on the GCP console.

The following data was used to complete this experiment:
Train record count: 80,000 [positive:40,000 and negative 40,000]
Validation record count: 10,000 [positive:5,000 and negative 5,000]
Test record count: 10,000 [positive:5,000 and negative 5,000]

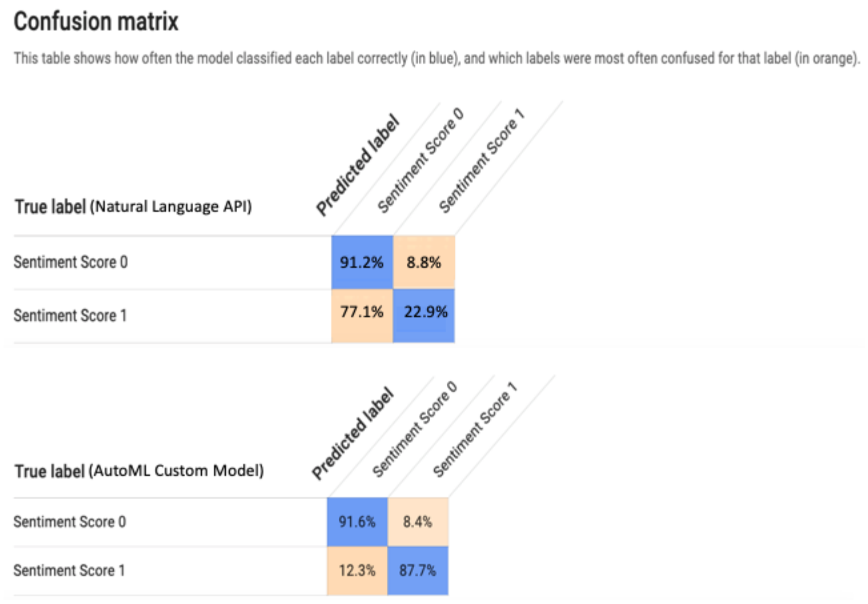
The GCP provides multiple APIs for training a custom AutoML model. The standard web user interface, which is a part of the GCP console was use for this experiment. The following high-level steps were required:

1. Prepare an input file in the following format:
   TRAIN,”… message text …”,0
   VALIDATION," ,”… message text …",0
   TEST,” ,”… message text …”,1
   NOTE: Each record is labeled as “TRAIN”, “VALIDATION”, or “TEST” to indicate how it should be used.
2. Logon to the GCP console to:
   a. Create a dataset
   b. Import the data from a local .csv file. Records are validated in this step during the upload. Common errors include:
      i. Non-English characters found in the comment
      ii. Duplicate comments found
      iii. Unsupported labels in column 1. Only “TRAIN”, “VALIDATION”, or “TEST” are expected.
      iv. This process took about two hours of runtime to complete. An email was sent when the upload was complete.
   c. Initiate the training, which took approximately 5 hours of runtime for this experiment. An email was sent when the training was complete.

AUTOML CUSTOM MODEL RESULTS

The results of the sentiment analysis using the trained custom model were generated by the AutoML process using the records labeled as “TRAIN” in the dataset. The results are presented in this section.

1. The confusion matrix displayed in Figure 4 highlights the improvements that were made by the custom AutoML model. Recall that the Natural Language API model had a particularly tough time with false positives. Almost 40% of the original predications were false positives. Now we can see that roughly 6% of the predictions are false positives, and the overall accuracy has improved from 57% to 90%. The two models appear similar in their ability to correctly identify negative sentiment (the true positive rates for sentiment score 0 are similar), but the AutoML model is much better at correctly identifying positive sentiment (sentiment score 1) since the false positive issue has been resolved.



Confusion matrix

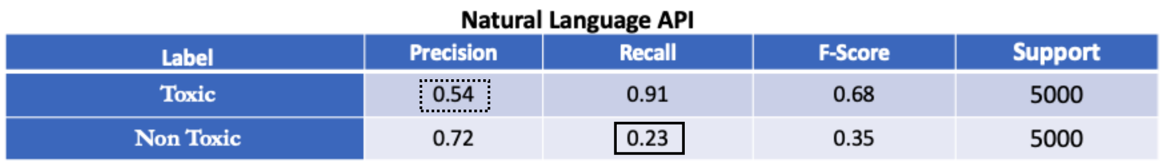
This table shows how often the model classified each label correctly (in blue), and which labels were most often confused for that label (in orange).

True label (Natural Language API) / Predicted label	Sentiment Score 0	Sentiment Score 1
Sentiment Score 0	91.2%	8.8%
Sentiment Score 1	77.1%	22.9%

True label (AutoML Custom Model) / Predicted label	Sentiment Score 0	Sentiment Score 1
Sentiment Score 0	91.6%	8.4%
Sentiment Score 1	12.3%	87.7%

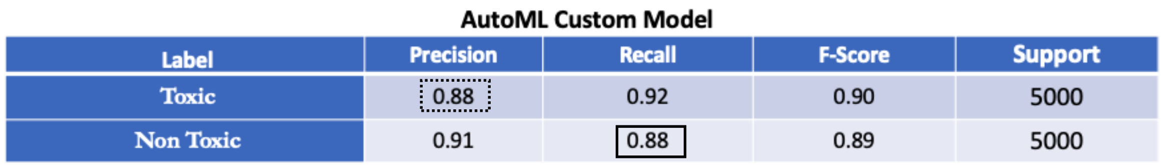
Figure 4: Confusion Matrix Comparison

2. The following table outlines the improvements made with the other key metrics describing the quality of the sentiment analysis. When we looked at the improved confusion matrix, we noticed that the two models were similar in their ability to correctly identify negative sentiment, but here we see that AutoML is more precise in this regard. Also, we see a dramatic improvement in recall for non-toxic message. The new model is much more likely to predict the non-toxic classes when appropriate.



Natural Language API

Label	Precision	Recall	F-Score	Support
Toxic	$\boxed{0.54}$	0.91	0.68	5000
Non Toxic	0.72	$\boxed{0.23}$	0.35	5000



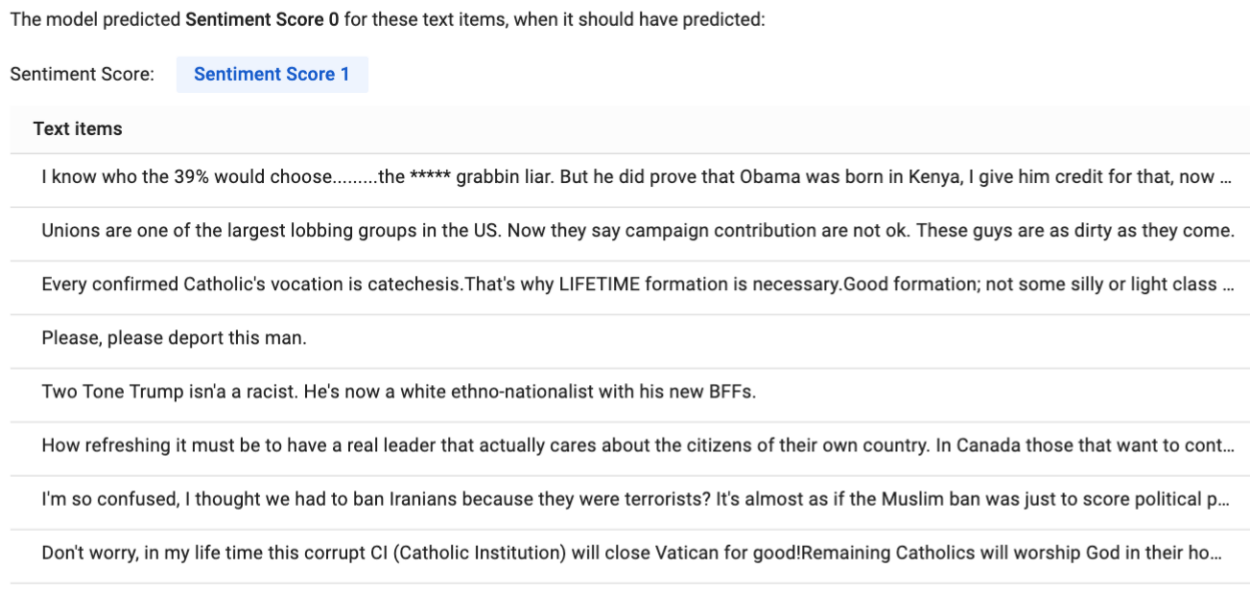
AutoML Custom Model

Label	Precision	Recall	F-Score	Support
Toxic	$\boxed{0.88}$	0.92	0.90	5000
Non Toxic	0.91	$\boxed{0.88}$	0.89	5000

Figure 5: Quality Metrics Comparisons

CONCLUSIONS AND RETROSPECTIVE

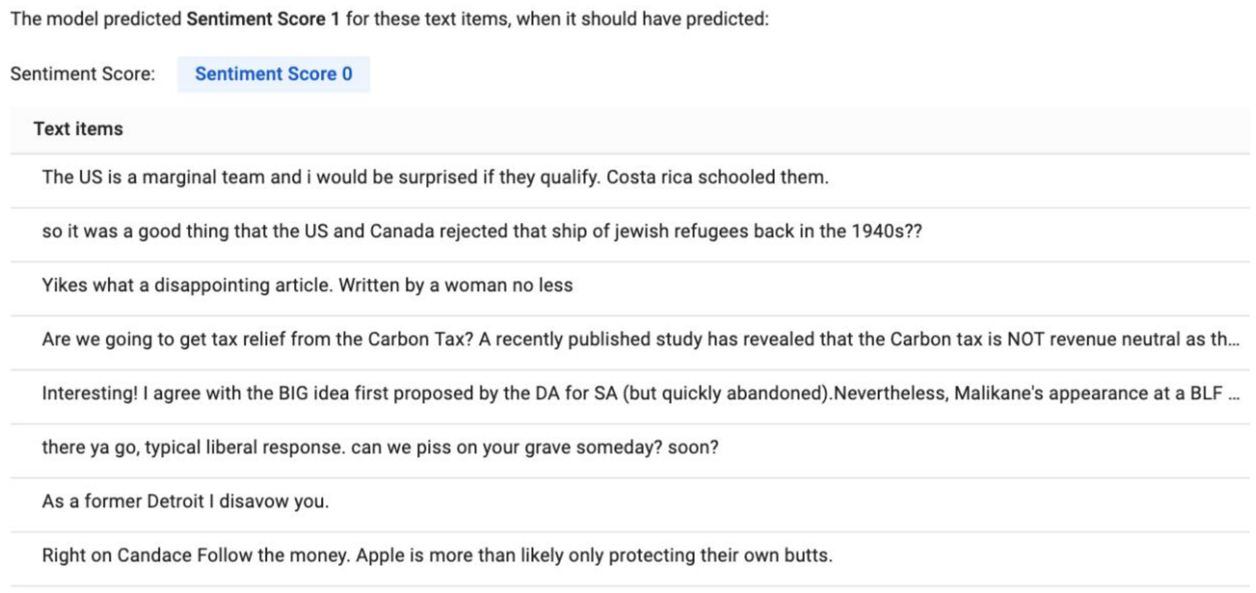
The improvements highlighted in the quality metrics above clearly demonstrate the benefits of building a custom model using the AutoML Natural Language service offered on the GCP platform. In addition to its ease of use and convenient reporting of the model's quality metrics, AutoML also highlights some specific examples that were misclassified to assist with the potential retraining of the model, and to provide insights on the type of messages the model struggled to classify. Here are some samples of misclassified messages from the model that was built for this project:


The model predicted **Sentiment Score 0** for these text items, when it should have predicted:

Sentiment Score: Sentiment Score 1

Text items
I know who the 39% would choose.........the ***** grabbin liar. But he did prove that Obama was born in Kenya, I give him credit for that, now ...
Unions are one of the largest lobbing groups in the US. Now they say campaign contribution are not ok. These guys are as dirty as they come.
Every confirmed Catholic's vocation is catechesis.That's why LIFETIME formation is necessary.Good formation; not some silly or light class ...
Please, please deport this man.
Two Tone Trump isn'a a racist. He's now a white ethno-nationalist with his new BFFs.
How refreshing it must be to have a real leader that actually cares about the citizens of their own country. In Canada those that want to cont...
I'm so confused, I thought we had to ban Iranians because they were terrorists? It's almost as if the Muslim ban was just to score political p...
Don't worry, in my life time this corrupt CI (Catholic Institution) will close Vatican for good!Remaining Catholics will worship God in their ho...

Figure 6: False positives from the perspective of the toxic class


The model predicted **Sentiment Score 1** for these text items, when it should have predicted:

Sentiment Score: Sentiment Score 0

Text items
The US is a marginal team and i would be surprised if they qualify. Costa rica schooled them.
so it was a good thing that the US and Canada rejected that ship of jewish refugees back in the 1940s??
Yikes what a disappointing article. Written by a woman no less
Are we going to get tax relief from the Carbon Tax? A recently published study has revealed that the Carbon tax is NOT revenue neutral as th...
Interesting! I agree with the BIG idea first proposed by the DA for SA (but quickly abandoned).Nevertheless, Malikane's appearance at a BLF ...
there ya go, typical liberal response. can we piss on your grave someday? soon?
As a former Detroit I disavow you.
Right on Candace Follow the money. Apple is more than likely only protecting their own butts.

Figure 7: False positives from the perspective of the non-toxic class

Figure 6 above lists a few examples of messages that were originally labeled by the human raters assigned to assess our original dataset as being non-toxic (score 1), but it is easy to browse this list and find a few that some other people might have labeled toxic (score 0). Figure 7 lists a few messages that were labeled by our human raters as toxic, but even here, there are a few cases that can be made for a different label. This shows the subjective nature of sentiment analysis and drives home the point that not even humans can reach a consensus on every message. This makes the model's job much more difficult, and cautions data scientists to be careful as they interpret the results of any sentiment analysis. Objective truth exists, but it should not be carelessly taken for granted when we receive labeled data to be used for AI model training. Ultimately all labeled sentiment analysis data is subjective, and the model is being trained to predict sentiment based on the subjective views of the group of humans responsible for labeling the data. Thus, the model will be subject to the inherent bias of the group of people chosen to assess the message content.

POSSIBLE EXTENSIONS TO THE WORK

Based on the discussion of subjectivity covered in the previous section, one interesting extension to this work could be to present the same set of messages to different groups of people across the country to get multiple sets of labels describing the toxicity of the same set of messages. For example, we could produce labeled data from more conservative parts of the country, and a different set of labels from liberal parts of the country. The group of raters could also be broken down in other ways, like by race, age, religious background, nationality, etc. We could build multiple models using the same techniques on various sets of labeled data to study the differences in sentiment, and potential bias, amongst various groups. In this approach, the data scientist would keep the model's architecture and hyperparameters consistent, and vary the labels from model to model, then make comparisons with the results.

A more technical extension of this project might include a lower level Tensorflow model on the GCP platform where the data scientist would have more control over the model's architecture. For this extension to the work, the modeler would work with a single set of labels, and experiment with various model configurations and vary the hyperparameters using Tensor flow in an attempt to improve the model's performance above what was achieved with the AutoML service offering.

BIBLIOGRAPHY

- Google Cloud AutoML Natural Language Sentiment Analysis
- Cloud Natural Language API documentation
- Google Cloud Platform Dashboard
- Google Cloud Platform Essential Training
- Google Cloud Platform for Machine Learning Essential Training
- Toxic Comment Classification Challenge
- Jigsaw Unintended Bias in Toxicity Classification
- Civil Comments