Project Description:

For the 2025 Spring DSA Capstone project, we were assigned by the client Dr. Cengiz Gunay with Dr. Anca Doloc-Mihu's supervision to work on a machine learning model that could perform natural language processing. The goal of this is to submit a csv to the Natural Language Processing with Disaster Tweets Kaggle competition and receive a place on the leaderboard. Due to the english language having certain words be slang and not quite literal to the intended meaning of the actual word, some tweets may sound off. Because of this, our machine learning model should decipher and identify whether a tweet is announcing a literal disaster or if the tweet is being metaphorical.

Team Members' Introductions:

Kazuki Susuki, Jonathan Tran, Jeampy Kalambayi

Client Presentation:

Client: Dr. Cengiz Gunay

Dr. Gunay is an Associate Professor of Information Technology at Georgia Gwinnett College. He provided us guidance on how to approach this project and gave us requirements for iterations of the project.

Team Plan:

During the tenure of the project, we first exchanged information to communicate with each other through Zoom, Discord, and email. Jira was used to create tasks for ourselves and plan out sprints with established deadlines. The first sprint was used to grab the data from the competition to clean and prepare it for our models. The second sprint involved creating different kinds of AI models for NLP to see which of the three would perform the best. Kazuki used the BERT model, Jonathan used SVM(Support Vector Machine), and Jeampy used PCA, which each produced accuracy results. In the third and final sprint, each member worked on visualizations for each model, along with a demo for the BERT model that accepted user text input.

Roles

Jonathan Tran - Project Manager and Data Analyst

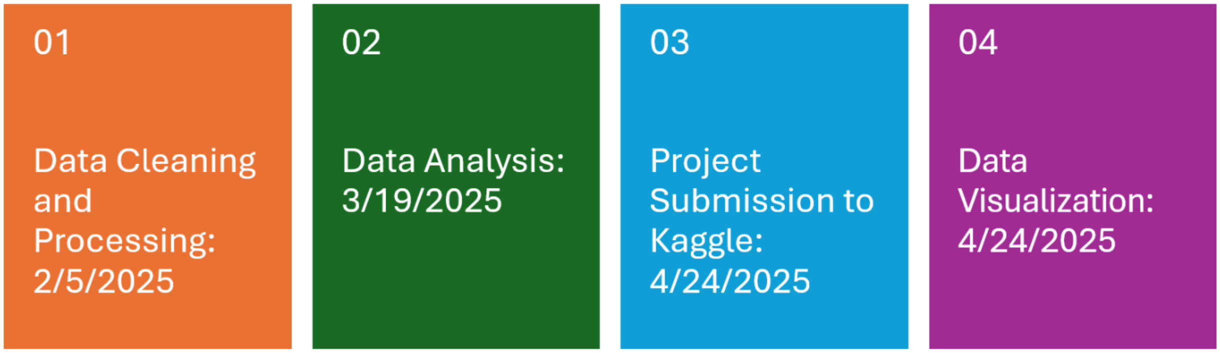
Kazuki Susuki - Client Liaison and Data Visualizer

Jeampy Kalambayi - Data Modeler and Project Documentation

Technologies

- **Notebooks:** Google Colab
- **Project Management:** Jira
- **Website:** Pythonanywhere, Flask
- **Repository:** Github

Project Flowchart



01 Data Cleaning and Processing: 2/5/2025

02 Data Analysis: 3/19/2025

03 Project Submission to Kaggle: 4/24/2025

04 Data Visualization: 4/24/2025

Data Collections

Kaggle Competition Data: **https://www.kaggle.com/competitions/nlp-getting-started/data**

Data Cleaning

- Removal of stopwords, punctuation, special characters, html tags, and URLs.

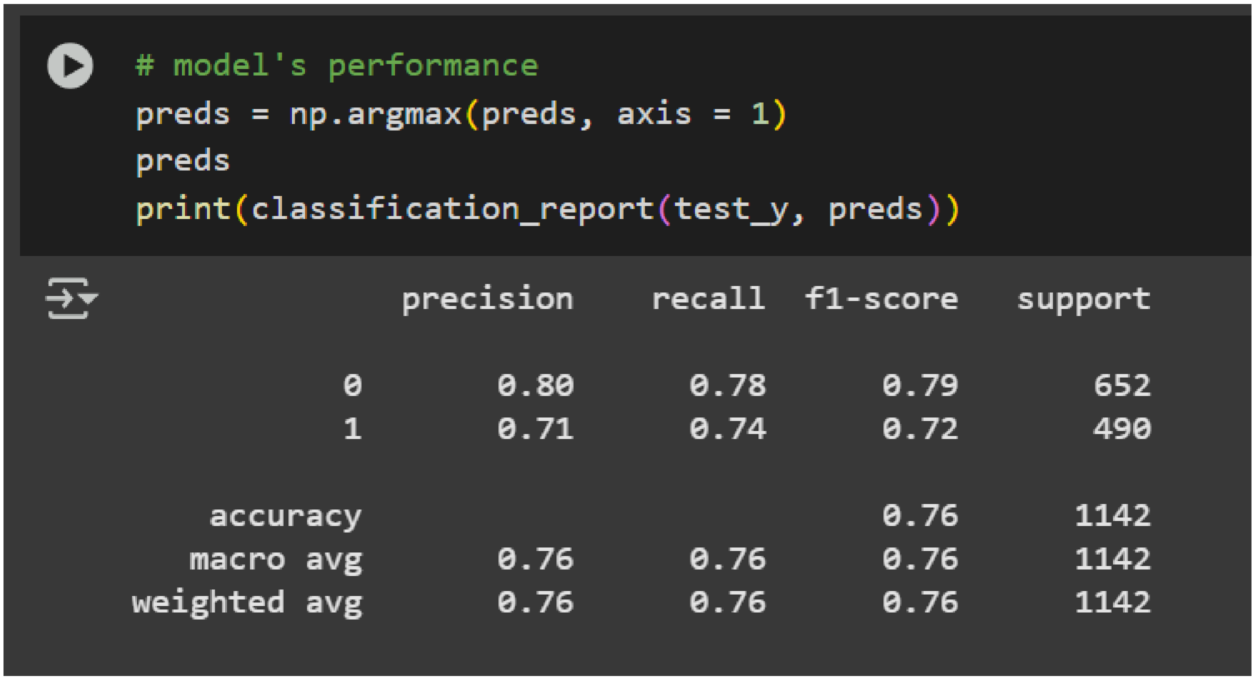
Data Analysis

- Support Vector Machine, a supervised learning model that is used for classification, regression, and outliers detection.
- BERT, a context-sensitive learning model used for classification tasks.
- PCA

Results

Model Accuracy

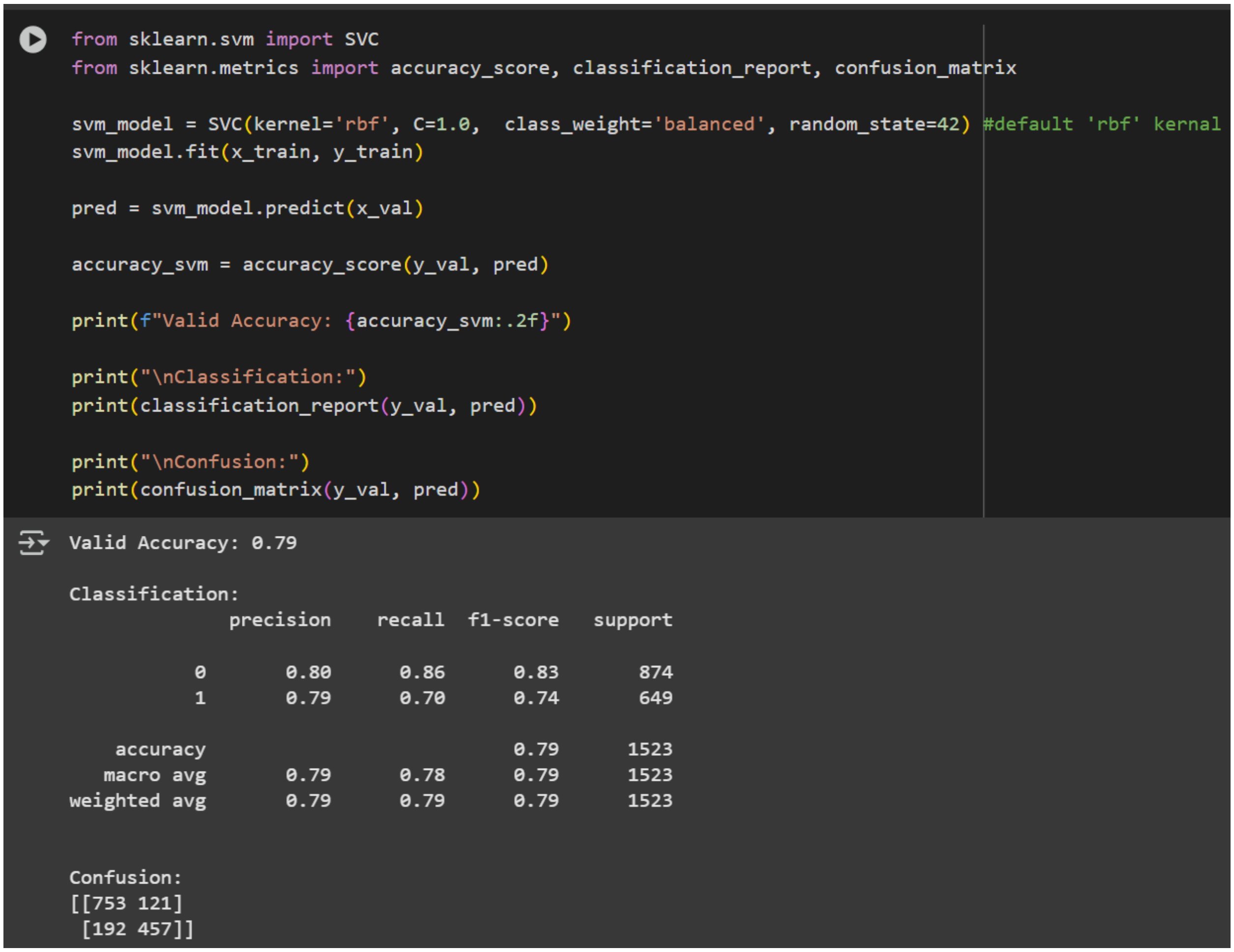
BERT



```python
# model's performance
preds = np.argmax(preds, axis = 1)
preds
print(classification_report(test_y, preds))
```

```
              precision    recall  f1-score   support

           0       0.80      0.78      0.79       652
           1       0.71      0.74      0.72       490

    accuracy                           0.76      1142
   macro avg       0.76      0.76      0.76      1142
weighted avg       0.76      0.76      0.76      1142
```

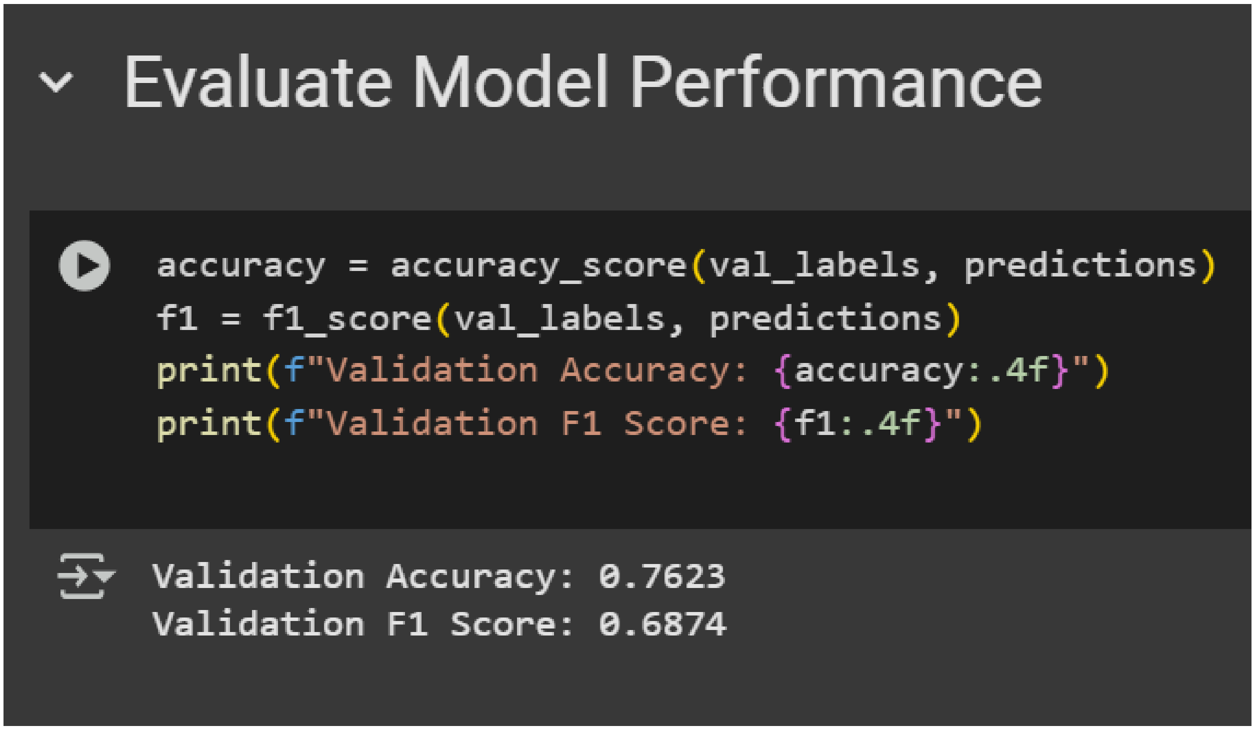
SVM RBF Kernel



```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

svm_model = SVC(kernel='rbf', C=1.0,  class_weight='balanced', random_state=42) #default 'rbf' kernal
svm_model.fit(x_train, y_train)

pred = svm_model.predict(x_val)

accuracy_svm = accuracy_score(y_val, pred)

print(f"Valid Accuracy: {accuracy_svm:.2f}")

print("\nClassification:")
print(classification_report(y_val, pred))

print("\nConfusion:")
print(confusion_matrix(y_val, pred))
```

```
Valid Accuracy: 0.79

Classification:
              precision    recall  f1-score   support

           0       0.80      0.86      0.83       874
           1       0.79      0.70      0.74       649

    accuracy                           0.79      1523
   macro avg       0.79      0.78      0.79      1523
weighted avg       0.79      0.79      0.79      1523


Confusion:
[[753 121]
 [192 457]]
```

PCA

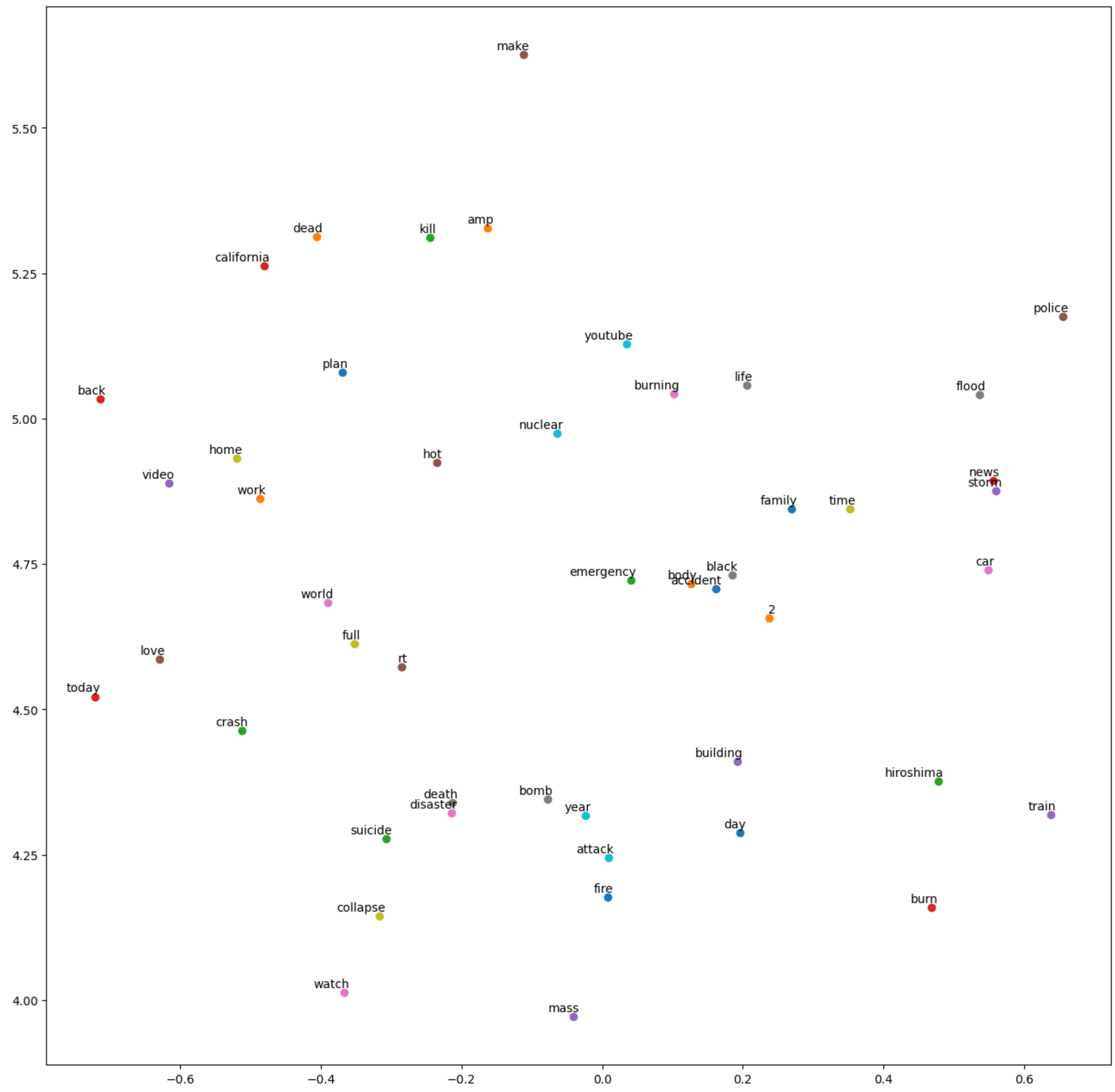


Evaluate Model Performance

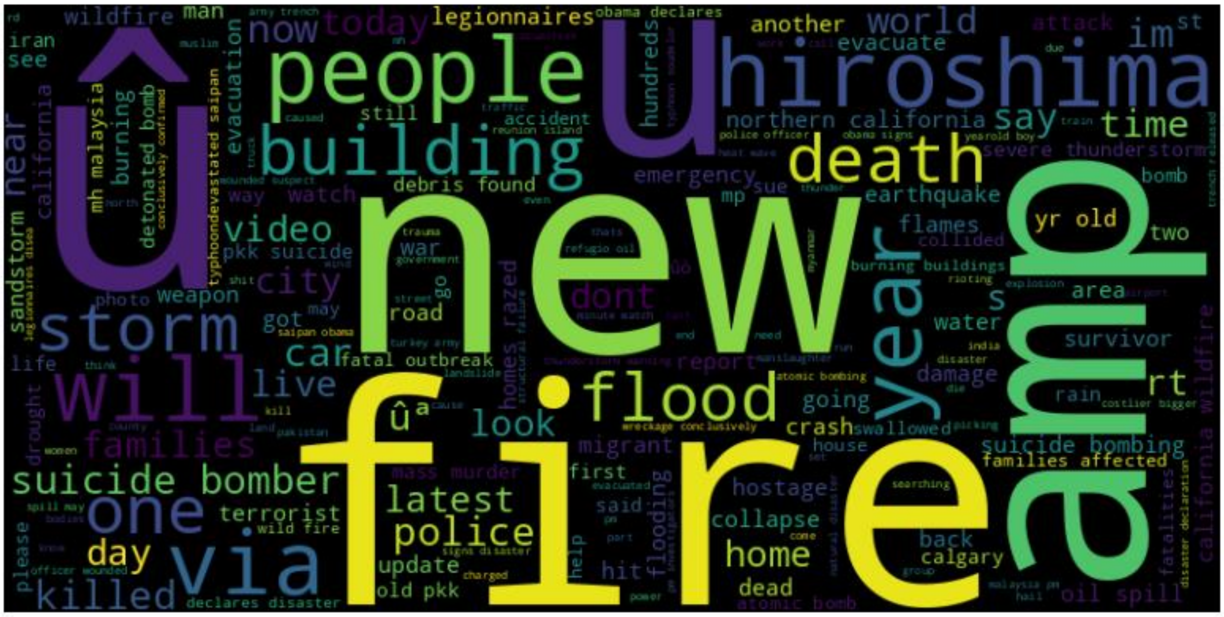
```
accuracy = accuracy_score(val_labels, predictions)
f1 = f1_score(val_labels, predictions)
print(f"Validation Accuracy: {accuracy:.4f}")
print(f"Validation F1 Score: {f1:.4f}")
```

```
Validation Accuracy: 0.7623
Validation F1 Score: 0.6874
```

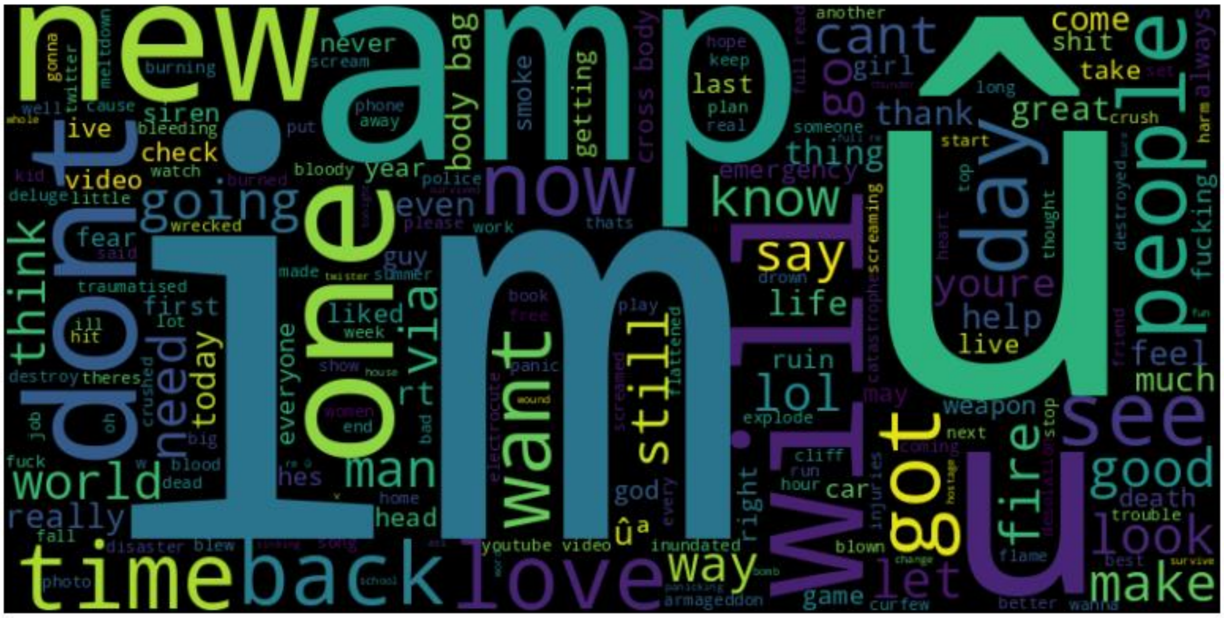
SVM TSNE Graph

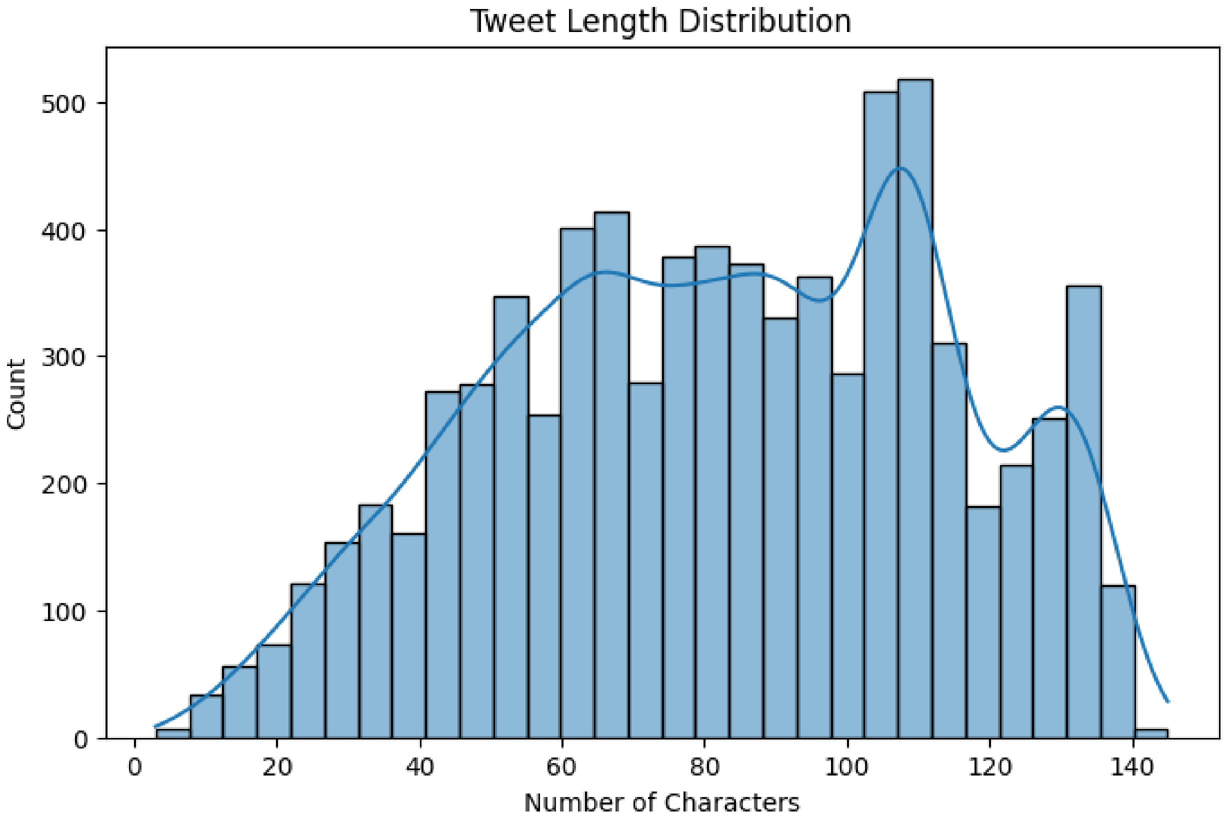

make
5.50
amp
dead
kill
california
5.25
police
youtube
plan
life
back
burning
flood
5.00
nuclear
home
hot
video
news
storm
work
family
time
4.75
emergency
black
body
accident
car
world
2
full
love
rt
today
4.50
crash
building
hiroshima
bomb
death
disaster
year
train
suicide
day
attack
4.25
fire
burn
collapse
watch
4.00
mass
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6

Word Cloud - Disaster Tweets

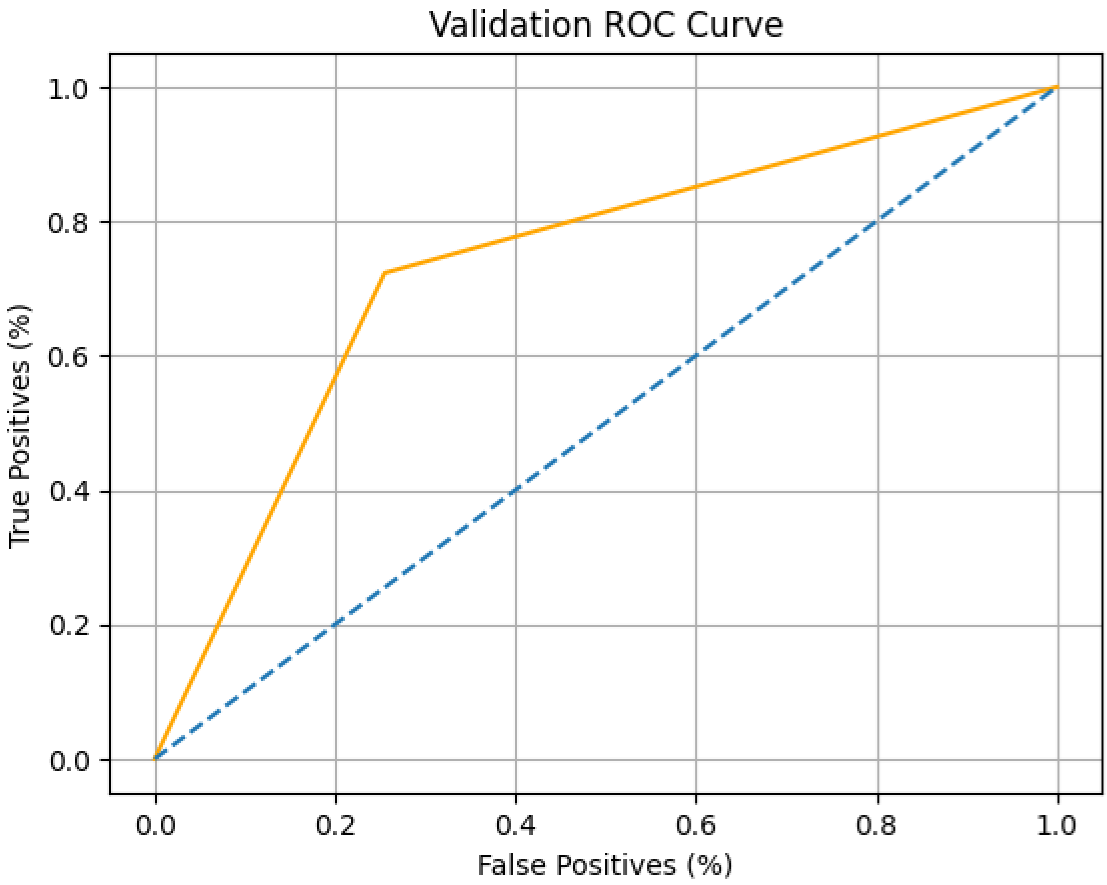


Word Cloud - Non-Disaster Tweets





BERT Model Performance Curve



Iteration Summaries:

Iteration 1:

For iteration 1, we focused on cleaning the data and preparing it for usage in our NLP models. This way, the models can produce more accurate classifications for the tweets as stopwords and HTML tags can change how the model perceives new text. We created some graphs like tweet length distribution and a world cloud to show the most common keyword for some tweets like “fire”.

Iteration 2:

On iteration 2 we assigned ourselves one model each to test out, which was BERT, SVM, and PCA. Using the train.csv from the Kaggle data, each model was trained on the data so that they can classify text in the test.csv as 1(Disaster) or 0(Not Disaster) in the form of targets. Once this classification is done, the targets and the ID of each tweet were exported as a submission file for the competition.

Iteration 3:

In our final iteration, we worked on visuals for each model. For the SVM model, Jonathan made a TSNE graph that grouped certain words together based on how strongly they related. For the BERT model, Kazuki made an ROC curve to evaluate the performance of the model based on true positive and false positives. Jeampy designed and hosted the website using Flask and Pythonanywhere to showcase the entire project.

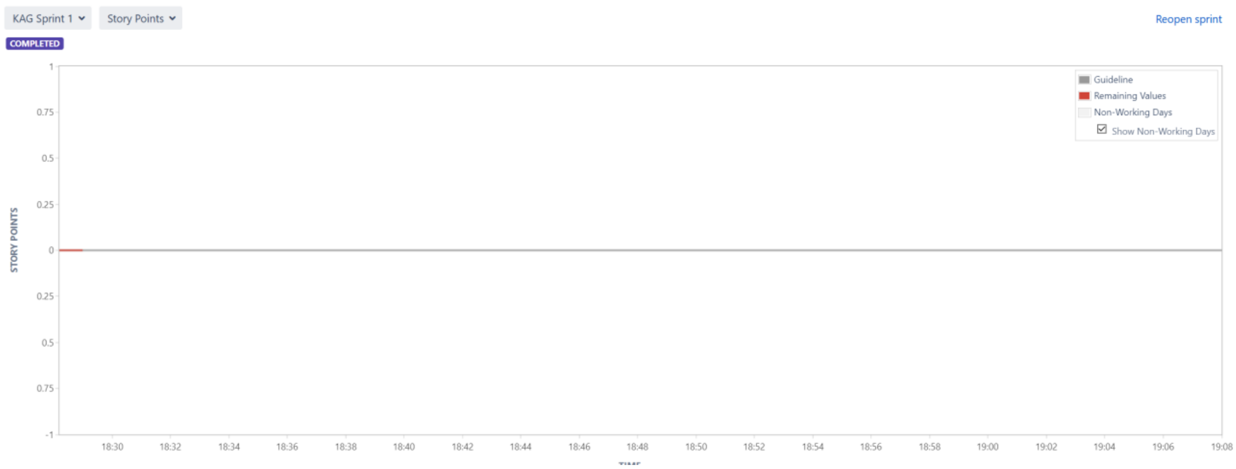
Github Repository

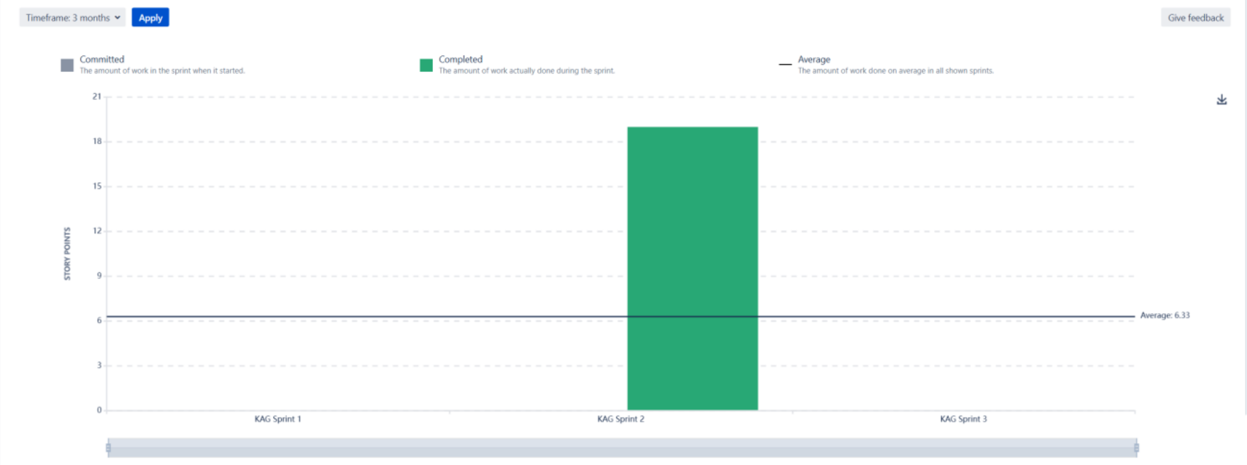
Link: https://github.com/GGC-DSA/Kaggle-Spr2025

Repo Overview

- Includes notebooks for each model under the code folder
- Website code and assets under the doc folder
- Media (CREATE poster, vlogs)

Sprint 1:



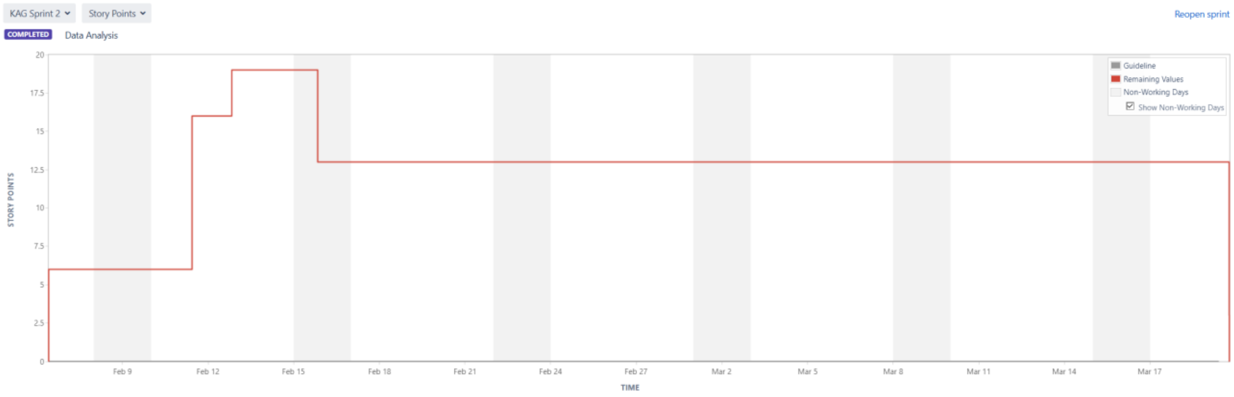


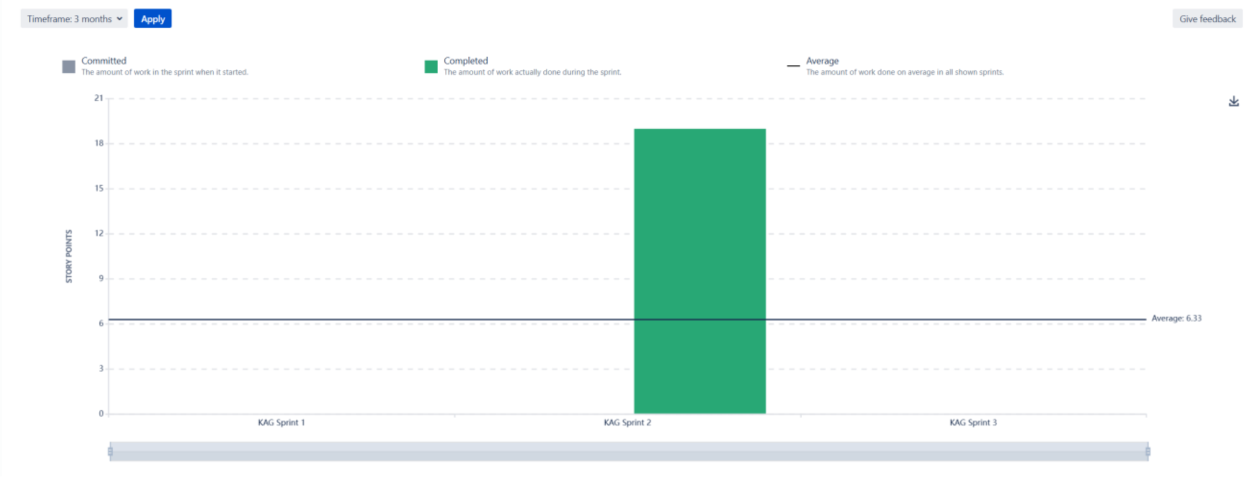
Committed
The amount of work in the sprint when it started.

Completed
The amount of work actually done during the sprint.

Average
The amount of work done on average in all shown sprints.

Sprint 2:



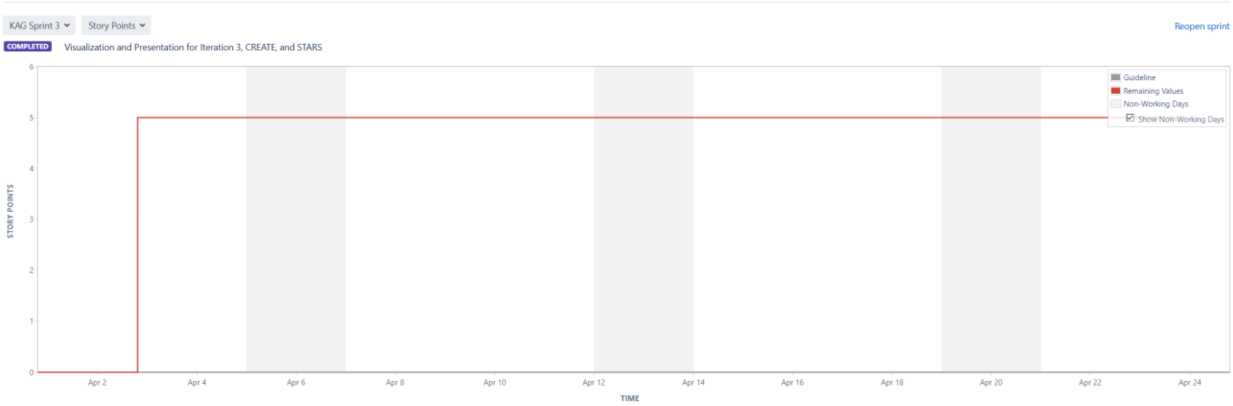


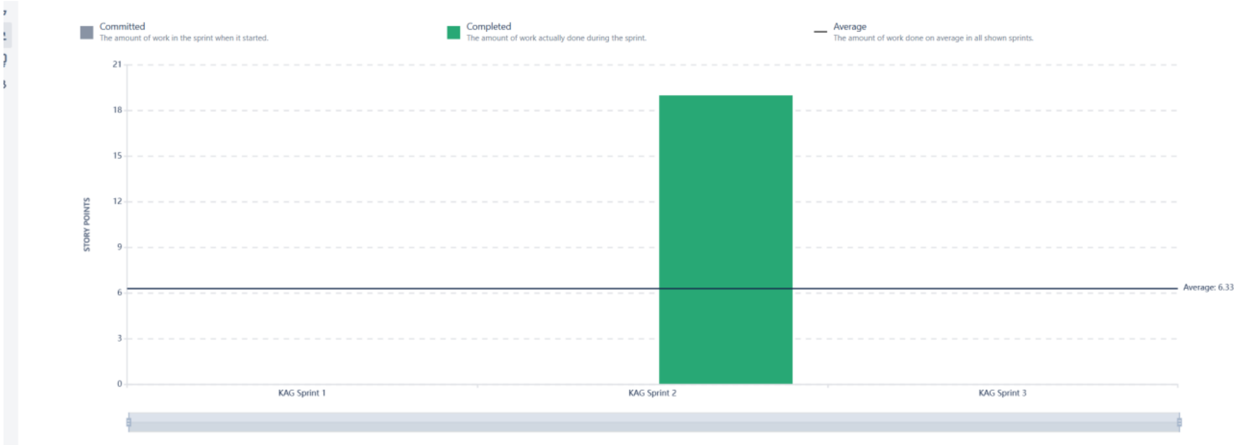
Committed: The amount of work in the sprint when it started.

Completed: The amount of work actually done during the sprint.

Average: The amount of work done on average in all shown sprints.

Sprint 3:





Committed: The amount of work in the sprint when it started.

Completed: The amount of work actually done during the sprint.

Average: The amount of work done on average in all shown sprints.

Demo/Vlog

https://drive.google.com/file/d/1g82qCIiIPsXPwBJ195X1OXq41fKVjcdd/view?usp=sharing

Features:

1. Natural Language Processing: Each model can train on dataset and process text to assign a text as a disaster or not.
2. Demo: Using the BERT Model notebook, a user can input text and the model will assign that text as a disaster or not based on the training data.

Known Issues:

1. The demo is fairly inaccurate in classifying text. Some text examples such as “this house is on fire” may register as non-disaster.

TODO:

Optimization: We want to focus on increasing the accuracy of the BERT model to get a higher place on the scoreboard for the Kaggle Competition.

Computer Vision: Some of the tweets include images of the disaster being described. We wanted to leverage computer vision to not only process text but also images for classification.

Improve ROC Curve: The ROC Curve is not correctly visualized with the true and false positives.

Improve TSNE: The TSNE graph is not in 3D, we have only visualized it in 2D.