C964: Computer Science Capstone

Benjamin Prendergast, 00[redacted]
1-28-2024

January 28, 2024

C964: Computer Science Capstone

Task 2 parts A, B, C and D

Part A: Letter of Transmittal

January 23rd, 2024

Roberta Frierson
Vice President, Information Technology
Springfield Psychological
123 Main St
Philadelphia, PA 19134

Subject: Proposal for Implementing Machine Learning Application to Enhance Patient Care

Dear Ms. Frierson,

As your technical account manager here at Valant, it is my respnosibility to ensure that Springfield Psychological is getting the most utility out of the Electronic Health Records (EHR) platform with which we provide you. I am writing to present a proposal that addresses a critical challenge currently faced by your organization and offers a solution that can provide significant benefits.

I have been made aware that since Springfield has added Valant's telehealth therapy functionality to its service offerings, it has experienced a surge in new patients, signaling growth and increased accessibility. However, this development has placed significant strain on clinicians who must now allocate more of their limited time to maintaining health records for the increasing number of patients. Consequently, overall support quality has decreased, and Springfield's business ratings are suffering. As competition among mental health service providers increases nationwide, the need to maintain the highest level of care for your clients has never been more crucial.

To better support your clinicians and get Springfield's reputation back on track, I propose the development of a machine learning application that will leverage the data now generated by telehealth therapy sessions, transforming a source of increased strain into an asset. The application will be designed to work with your system by applying Natural Language Processing (NLP) techniques to transcripts generated from virtual sessions to identify patients who exhibit signs of suicidal ideation. The findings will be passed back to the EHR system, facilitating rapid updates to patient records, and enabling the initiation of follow-up actions and resource engagement for those most in need.

Implementing this solution will offer practical benefits to Springfield Psychological and its patients. By accelerating the categorization of high-risk patients, time spent by clinicians on health record maintenance can be reduced, affording them more time to focus on patient care. Further, Springfield's patients can be assured of ongoing access to exceptional care, with immediate support prioritized for those in critical need. Ensuring that patients at risk of self-harm receive the necessary support and attention will also mitigate potential malpractice suits citing negligence. I am confident that implementing this solution will result in reduced stress for clinicians, enhanced support for patients, and an overall improvement in Springfield's business ratings.

The development and implementation of this solution come with a reasonable cost and can be completed in a relatively short time. Springfield's existing service contract with Valent already includes cloud storage, hosting, maintenance, training, and support. The only additional cost is for billable programming, which is projected to be under $13,000. Implementation time should not exceed nine weeks, including development, testing, deployment, and training for clinicians. The application will operate securely alongside Springfield's existing EHR environment and utilize telehealth session transcripts stored within patient records, ensuring compliance with data protection regulations. The informed consent commitments made between Springfield and its telehealth patients regarding the use of session data will be upheld throughout the process.

I bring to this project eight years of experience supporting and developing the Valent platform and assistive tools. As the technical account manager for Springfield, I am aware of Springfield's business objectives as well as its challenges. I am confident that I have the knowledge and skills necessary to lead this initiative successfully.

In conclusion, I believe that the implementation of this machine learning application will not only address the current challenges faced by Springfield but also make Springfield a leader in leveraging technology to enhance patient care. Together, with your approval, we can make significant strides in supporting Springfield's clinicians and, more importantly, the well-being of its patients.

Thank you for considering this proposal.

Sincerely,

Benjamin Prendergast
Technical Account Manager
Valant EHR

Part B: Project Proposal Plan

Project Summary

As a behavioral health organization, Springfield Psychological aims to provide high-quality mental health support to its patients. Among Valent's largest accounts, Springfield is widely recognized as one of the largest mental health care providers in the mid-Atlantic region, with a historical reputation for providing an elevated level of care. In response to the COVID-19 pandemic, Springfield added Valent EHR telehealth therapy sessions to its service offerings, resulting in a notable surge in new patients. While proving to be beneficial in terms of attracting new patients, the increased caseloads have placed considerable pressure on clinicians, necessitating additional time devoted to administrative tasks such as medical record maintenance and leaving less time to concentrate on patient care. As a result, the quality of care has decreased, leading to a dip in Springfield's business ratings.

Maintaining a positive reputation is crucial for Springfield's continued success. To that end, finding solutions for the internal inefficiencies contributing to Springfield's suffering business ratings is paramount. Springfield requires a solution that will improve patient satisfaction by enabling clinicians to provide the highest level of care. As clinicians are required to dedicate more time to managing patient health records, assisting them in these tasks emerges as a key focus for improvement.

The core deliverable of this project will be a machine learning application that aims to aid clinicians in their daily responsibilities by assisting in the classification of patients in need of heightened care and attention, specifically those who may be at risk of self-harm. The classification will be accomplished by utilizing Natural Language Processing (NLP) to analyze transcripts generated from telehealth sessions. With a simple interface, the Transcript Analysis Tool (TAT) will collect transcript text files, analyze them, and output results in a format suitable for ingestion by the EHR system. This will enable clinicians to efficiently analyze session transcripts and promptly update patient health records.

TAT will be capable of analyzing plain text input, an individual session transcript, or a group of transcripts, satisfying multiple use cases. Alongside exporting analysis results for integration with the EHR system, the tool will maintain an internal record of every analysis, offering potential opportunities for enhancing analysis accuracy in the future. Designated power users will have access to additional protected capabilities which include access to visualizations related to the machine learning model and its training dataset, as well as functionality to incorporate additional entries to the dataset and retrain the model.

In addition to the analysis tool, a comprehensive user guide will be delivered. The guide will include instructions for clinicians to access and use the tool as well as details for super users to access and utilize the model and data-related tools.

To summarize, this machine learning tool will offer several benefits to Springfield Psychological. Aiding clinicians in their day-to-day tasks by assisting with the classification of at-risk patients will allow them to operate more efficiently and focus more on patient care. Timely identification of patients at risk of suicidal behavior will enable clinicians to intervene promptly and provide the necessary support. By enhancing care quality and efficiency, positive patient outcomes are more likely, leading to improved patient satisfaction and business ratings.

Data Summary

The machine learning model will be initially trained on a dataset acquired from Kaggle.com. The dataset, downloaded directly from Kaggle, consists of a repository of more than 230,000 Reddit posts, each classified as suicidal or non-suicidal. The posts were originally submitted to Reddit between 2008 and 2021 and were collected using the Pushshift API. Posts from subreddits focused on topics such as suicidal ideation and depression were selected to represent the suicidal segment of the dataset, while posts from subreddits centered around casual conversations among teenagers were chosen to represent the non-suicidal segment.

The dataset is formatted in CSV (Comma-Separated Values) style, featuring three key features: post ID, post text, and classification. In preparation for use by the machine learning model, the dataset will undergo manual inspection to identify and remove any apparent anomalies, such as outliers or missing data. The post ID attribute will be removed, leaving only the pertinent post text alongside its corresponding classification. Additional data cleaning will occur during the model training process, reducing the risk of training errors, in addition to feature extraction and vectorization.

Once the tool is operational, all classification results will be internally stored. Users with designated access will have the ability to append these results to the existing dataset, enabling the possibility of retraining the model on an expanded, more contextual dataset. This should result in a model that improves over time.

The Kaggle dataset offers a solid foundation for developing the machine learning model in this project. The conversational statements contained within the dataset, particularly those related to suicidal ideation, mirror the nature of interactions and disclosures typically encountered within therapy sessions or patient communications. Further, the size and wide range of statements present in the dataset ensure the model is exposed to an array of language patterns, enabling it to adapt to varying patient communication styles. Using this dataset, the Transcript Analysis Tool can effectively hone its ability to identify cues indicating suicidal risk.

Since the initial dataset is publicly available, privacy concerns are effectively mitigated. However, the context of mental health and suicidal ideation demands a heightened level of sensitivity, which will be upheld by all stakeholders involved in the development process. Also, while the developers will not have direct access to any patient medical records during the development process, HIPAA requirements will need to be considered once the tool is integrated with Springfield’s EHR system.

Implementation

This project will follow the guidelines of the CRISP-DM methodology, which lays out a structured approach to development. CRISP-DM aligns well with this project because it emphasizes data preparation, modeling, and evaluation, all of which are essential to the development of a machine learning model. The phases of CRISP-DM are broken down into these components:

Business Understanding: This phase involves understanding the project objectives and requirements from a business perspective. It includes identifying the goals of the machine learning tool, understanding the needs of end-users, and defining success criteria. In this project, it will entail defining the objectives of the tool, namely assisting in the classification of patients at risk for self-harm and understanding the needs of clinicians who will use the tool.

Data Understanding: In this phase, the focus is on acquiring, exploring, and understanding the data that will be used for model development. It involves gathering relevant datasets, exploring their features, identifying potential issues, and gaining insights into the data that could inform the modeling process. In this project, Data Understanding will involve obtaining the dataset of Reddit posts, exploring its content and structure, and assessing its suitability for training the machine learning model.

Data Preparation: This phase involves preparing the data for analysis and modeling. It includes tasks such as cleaning the data, transforming the data into a format suitable for modeling, and selecting relevant features. In this project, Data Preparation will involve cleaning the Reddit posts dataset and preprocessing the text data for training the machine learning model.

Modeling: In this phase, machine learning models are selected, built, and evaluated using the prepared dataset. It involves experimenting with different algorithms, tuning model parameters, and assessing model performance. In this project, Modeling will involve selecting an appropriate machine learning algorithm for text classification, training the model using the dataset, and evaluating its performance using metrics such as accuracy, precision, and recall.

Evaluation: This phase involves assessing the performance of the developed model and determining whether it is ready for deployment. It includes validating the model against the defined success criteria and identifying any shortcomings or areas for improvement. In this project, Evaluation will involve assessing the performance of the trained models in accurately classifying therapy session transcripts and ensuring that they meet the requirements of clinicians and stakeholders.

Deployment: In this phase, the developed model is deployed to the production environment. It involves integrating the model within the target system, providing necessary documentation and support for end-users, and monitoring the performance of the deployed models over time. In this project, Deployment will involve integrating the trained model and Transcript Analysis Tool with Springfield's EHR system, performing unit testing, and providing training and support for clinicians on using the tool effectively in their daily workflows.

Timeline

Milestone or deliverable	Duration	Projected start date	Anticipated end date
Project Kickoff	1 day	2/5/2024	2/5/2024
Business Understanding	4 days	2/6/2024	2/9/2024
Data Acquisition	2 days	2/12/2024	2/13/2024
Data Exploration	3 days	2/14/2024	2/16/2024
Data Preparation	3 days	2/19/2024	2/21/2024
Model Selection and Training	7 days	2/22/2024	3/1/2024
Model Evaluation	4 days	3/4/2024	3/7/2024
Tool Development	7 days	3/5/2024	3/13/2024
Tool Testing	3 days	3/14/2024	3/18/2024
Tool Integration	4 days	3/19/2024	3/22/2024
User Training	3 days	3/25/2024	3/27/2024
Final Review	3 days	3/28/2024	4/1/2024

Evaluation Plan

Verification will take place throughout the development timeline. Initial verification will begin during the business understanding phase. During this stage, meetings will be conducted between stakeholders and project leaders to ensure all requirements, objectives, and success criteria are clearly defined and understood by all parties. Any ambiguities should be identified and resolved during this verification phase, resulting in a clear path forward.

Verification of the data will occur as it is acquired, explored, and prepared. The machine learning model developer will verify that the dataset meets the project's requirements and will be sufficient for model training. At this stage, clinicians will be consulted to verify that the data set is representative of patient interactions. As data preparation takes place, the model developer will verify that preprocessing steps related to data cleaning and feature engineering are performed correctly.

Verification during the modeling stage will involve validating that the selected machine learning algorithm is appropriate for the project and is implemented correctly. The developers will conduct code reviews and testing to ensure model reliability. Once the model is developed, evaluation will begin, during which the accuracy of the model will be assessed. The targeted accuracy rate will be 80%.

During deployment, verification will involve validating that the deployed tool functions correctly, integrates seamlessly with the EHR system, and operates as expected in the production environment. Springfield’s IT team will be tasked with conducting rigorous system tests to ensure the deployment meets acceptance criteria and that each aspect of the integration is working as intended.

Upon completion of the project, the validation method to be used is User Acceptance Testing (UAT). This process will involve allowing end-users, namely clinicians, to interact with the tool in a controlled environment. Testers will evaluate the tool for usability, functionality, and effectiveness using live data. Any bugs or integration issues uncovered during this phase will be resolved before the project is considered complete.

Resources and Costs

All expenses associated with hardware, hosting, and maintenance are included within Springfield Psychological's existing contract with Valant. All supplementary costs arise solely from billable programming hours.

Item	**Description**	**Cost**
Development Software	PyCharm IDE and associated libraries	$0.00
Modeling Data	Publicly available dataset from Kaggle	$0.00
Billable Hours – Development	Machine learning specialist – 133 Hours	$6,650.00
Billable Hours – Development	Python Developer – 50 Hours	$3,500.00
Billable Hours – Integration	Integration Specialist – 28 Hours	$1,250.00
Billable Hours – Testing	QA Tester – 28 Hours	$1,250.00
Environment	Cloud storage, hosting, maintenance, training, and support	Included in ongoing contract
	Total Cost	$12,650.00

Part C: Application

The application, developed using python, has been deployed as a standalone executable. Application submission is contained in ‘bprend2 C964 Task 2.zip’ and includes the following files and directories:

```
bprend2 C964 Task 2/
│
├── data/
│   ├── data_exploration_plots.png
│   └── dataset.csv
│
├── models/
│   ├── logistic_regression_model.pkl
│   └── tfidf_vectorizer.pkl
│
├── output/
│  (Empty directory)
│
├── transcripts/
│   ├── 2024-01-18_281432_1191_transcript.txt
│   ├── 2024-01-18_282616_1191_transcript.txt
│   └── 2024-01-18_282736_1191_transcript.txt
│
└── TAT.exe
```

TAT.exe was designed to work on devices running Windows 10.

Alongside the application submission and this document is the python source code. The source code is contained in ‘bprend2 C964 Task 2 Source.zip’ and includes the following files:

main.py – the entirety of the Python source code
requirements.txt – a listing of all dependencies and respective versions used in development.

All coding was done in PyCharm. To function correctly, main.py should be placed in the same directory as the data, models, output, and transcripts directories included with the application submission.

Part D: Post-implementation Report

Solution Summary

Springfield Psychological is a behavioral health organization that recently added telehealth therapy sessions to its service offerings. After the service was implemented, Springfield experienced a significant influx of new patients. This resulted in increased caseloads for clinicians who were required to allocate more of their limited time to administrative tasks such as health record maintenance, leaving less time for patient care. Consequently, care quality was adversely affected, causing business ratings to suffer.

Springfield needed a solution to improve the operational efficiency of its clinicians, enabling them to spend less time managing health records so they could refocus their attention on patient care and restoration of the reputation of the business. The Transcript Analysis Tool was designed to do just that. Integrated with Springfield's EHR system, the tool provided clinicians with the ability to quickly run analysis on therapy session transcripts generated from telehealth sessions to classify patients at risk of self-harm. The tool output analysis results in a format that could be ingested by the EHR system to facilitate automatic updates to health records and initiate appropriate follow-up and resource engagement for patients most in need.

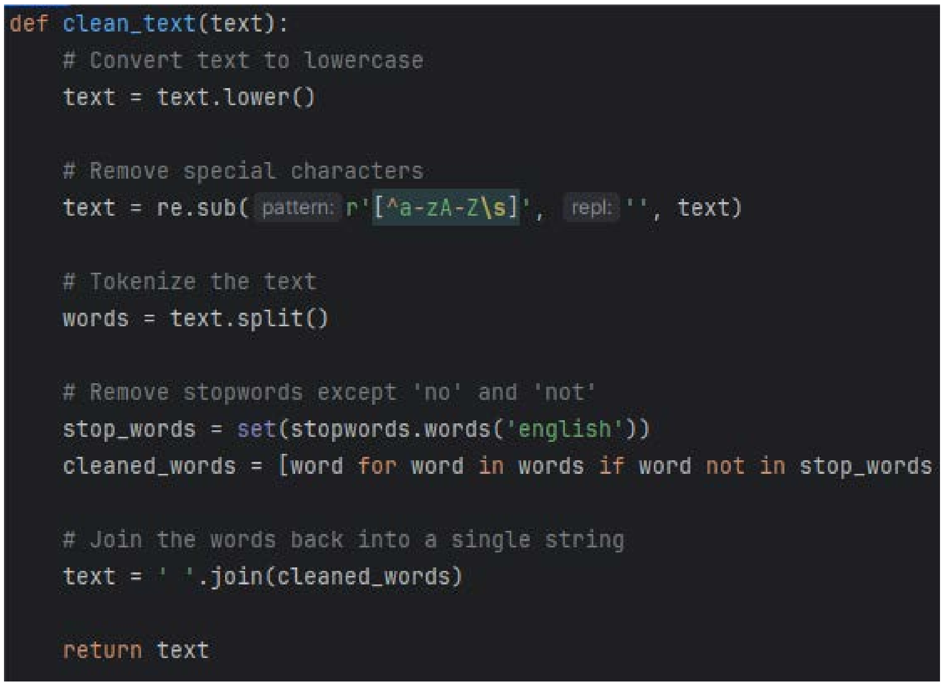
The Transcript Analysis Tool proved to be a valuable asset to Springfield. Clinicians reported several benefits, including less time spent updating health records, more time for patient care, and improved handling of at-risk patients. By supering clinicians in their daily responsibilities to alleviate caseload pressures, there was a noticeable improvement in care quality and a subsequent upswing in business ratings, effectively realizing the overall business objective.

Data Summary

The dataset used for this project was a publicly available dataset acquired from Kaggle.com. The dataset consisted of over 230,000 Reddit posts, each classified as suicidal or non-suicidal. Mental health and depression-related subreddits were targeted to construct the suicidal portion of the dataset, while casual chat subreddits were targeted to construct the non-suicidal portion of the dataset. The data was formatted into a single CSV file with three columns, post ID, text, and classification.

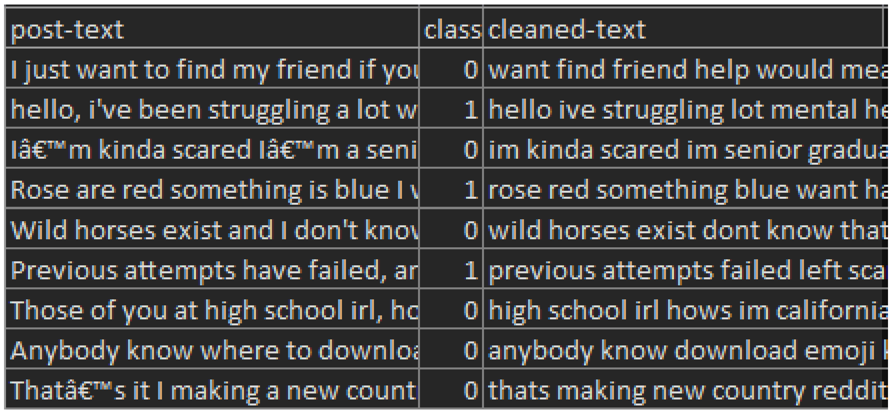
Throughout the design and development stages of the project, the dataset underwent modifications to enhance its compatibility with the machine learning model. Initially, the data was manually inspected for obvious issues and anomalies. Entries with empty or nonsensical text were identified and removed. Additionally, the post ID column was eliminated as it did not contribute to analysis.

Later in development, the CSV dataset was fed into the application, where it was further preprocessed and cleaned. This involved converting text to lowercase, removing special characters, and eliminating stopwords. Each of these steps was implemented to standardize the text and maximize its compatibility with the machine learning model.



```
def clean_text(text):
    # Convert text to lowercase
    text = text.lower()

    # Remove special characters
    text = re.sub( pattern: r'[^a-zA-Z\s]',  repl: '', text)

    # Tokenize the text
    words = text.split()

    # Remove stopwords except 'no' and 'not'
    stop_words = set(stopwords.words('english'))
    cleaned_words = [word for word in words if word not in stop_words

    # Join the words back into a single string
    text = ' '.join(cleaned_words)

    return text
```

Code snippet showing the clean_text method which was applied to text within the dataset.

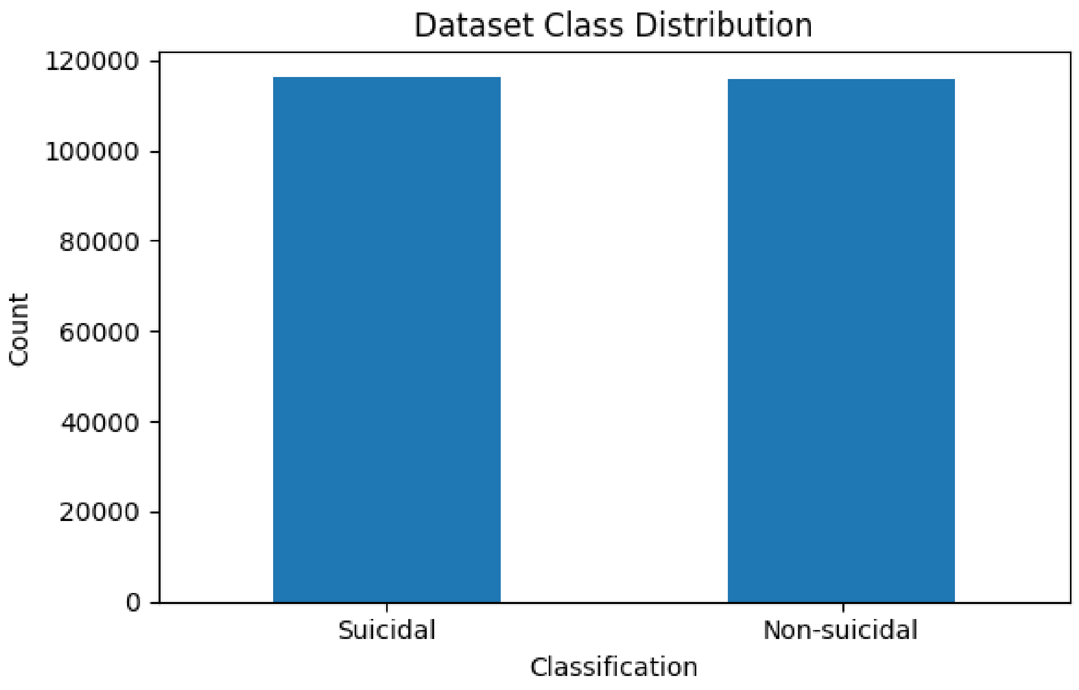


post-text	class	cleaned-text
I just want to find my friend if you	0	want find friend help would mea
hello, i've been struggling a lot w	1	hello ive struggling lot mental he
Iâ€™m kinda scared Iâ€™m a seni	0	im kinda scared im senior gradua
Rose are red something is blue I v	1	rose red something blue want ha
Wild horses exist and I don't knov	0	wild horses exist dont know that
Previous attempts have failed, ar	1	previous attempts failed left sca
Those of you at high school irl, ho	0	high school irl hows im californi
Anybody know where to downloa	0	anybody know download emoji
Thatâ€™s it I making a new count	0	thats making new country reddit

Screenshot showing the results of text cleaning.

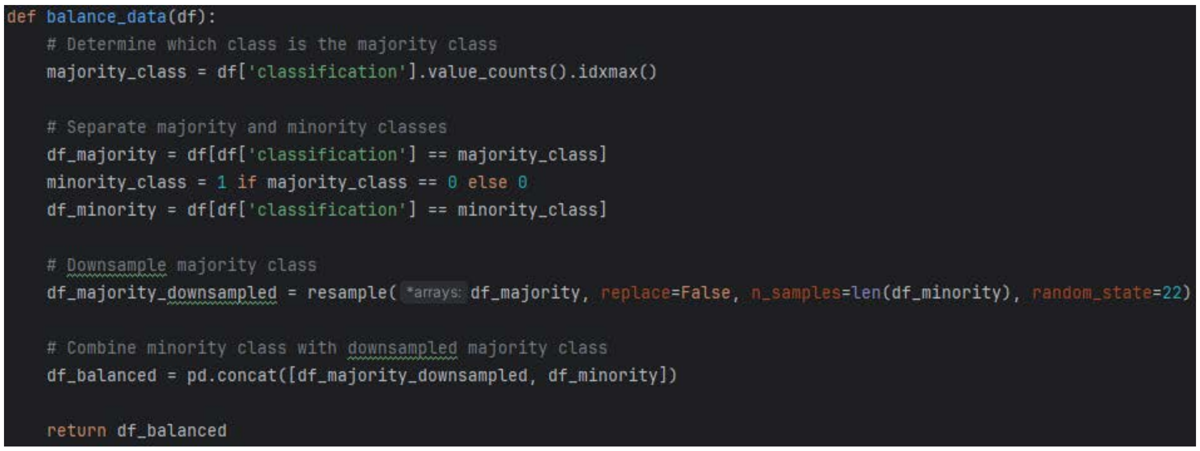
To prepare for model training and evaluation, the preprocessed dataset was split into two subsets using the train_test_split function from the scikit-learn library: 80% of the data was allocated for training, with the remaining 20% designated for testing purposes. Finally, Term Frequency - Inverse Document Frequency (TF-IDF) vectorization was applied to transform the text data into numerical features suitable for use by the machine learning model.

Following deployment to the production environment, ongoing data refinement continued. Anticipating the possibility of applying active learning techniques, the tool was designed to retain the results of every transcript analysis. Accessing the tool through password-protected privileges granted users the capability to append these results to the original dataset and subsequently retrain the model. This process provided the model with more contextual insights derived from the analyses of actual therapy data.



Screenshot of the class distribution bar graph generated from the original dataset.

Though initially evenly balanced, augmenting the dataset exposes the potential for an imbalance to emerge. To prevent this from arising, the application incorporated data balancing routines within its preprocessing steps to ensure the dataset remained within acceptable balance boundaries.



```python
def balance_data(df):
    # Determine which class is the majority class
    majority_class = df['classification'].value_counts().idxmax()

    # Separate majority and minority classes
    df_majority = df[df['classification'] == majority_class]
    minority_class = 1 if majority_class == 0 else 0
    df_minority = df[df['classification'] == minority_class]

    # Downsample majority class
    df_majority_downsampled = resample( *arrays: df_majority, replace=False, n_samples=len(df_minority), random_state=22)

    # Combine minority class with downsampled majority class
    df_balanced = pd.concat([df_majority_downsampled, df_minority])

    return df_balanced
```

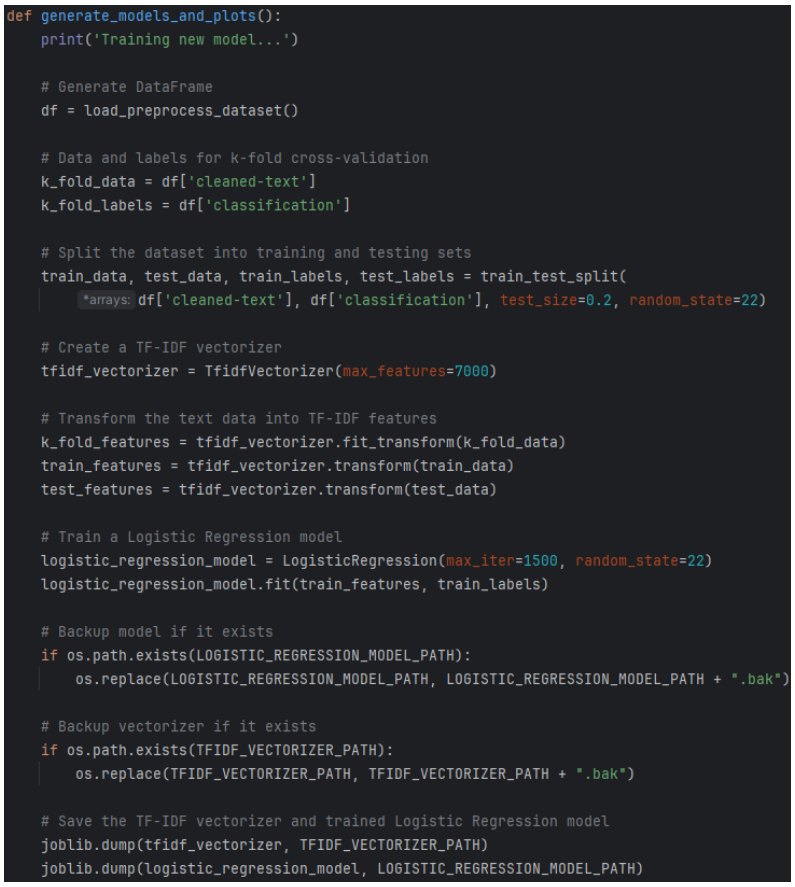
Code snippet showing the balance_data method which was applied in cases of imbalanced data.

Machine Learning

To handle the binary classification predictions necessary for this project, a logistic regression model was implemented. Logistic regression is a supervised learning algorithm that estimates the probability that a given input belongs to a particular class by fitting a logistic function to the data. In this case, the model was applied to textual data that underwent TF-IDF vectorization, a technique common in Natural Language Processing (NLP) that quantifies the importance of a term within a document relative to its occurrence in the entire corpus (Karabiber, 2021).

The machine learning model was implemented using various Python libraries, including scikit-learn for model training and assessment, matplotlib for data visualization, and pandas for data manipulation. To prevent overfitting, the model was trained on one portion of the dataset and tested on another portion. After training, the model was backed up using the joblib library, enabling its reuse without the need for retraining. Additionally, the backup ensured that if a replacement logistic regression model, trained on an updated dataset, failed to perform well, the original model could be restored for continued use.

The logistic regression model was chosen because its strengths closely align with the requirements of this project. It is proven to be highly effective in binary classification tasks and is capable of making predictions quickly, attributes that were essential for this specific use case. Further, the model is straightforward to implement, making it useful to developers operating on a tight schedule and limited budget. Finally, logistic regression is less prone to over-fitting, necessitating minimal fine-tuning and maintenance efforts (“Advantages and Disadvantages of Logistic Regression,” 2023).



```python
def generate_models_and_plots():
    print('Training new model...')

    # Generate DataFrame
    df = load_preprocess_dataset()

    # Data and labels for k-fold cross-validation
    k_fold_data = df['cleaned-text']
    k_fold_labels = df['classification']

    # Split the dataset into training and testing sets
    train_data, test_data, train_labels, test_labels = train_test_split(
        *arrays: df['cleaned-text'], df['classification'], test_size=0.2, random_state=22)

    # Create a TF-IDF vectorizer
    tfidf_vectorizer = TfidfVectorizer(max_features=7000)

    # Transform the text data into TF-IDF features
    k_fold_features = tfidf_vectorizer.fit_transform(k_fold_data)
    train_features = tfidf_vectorizer.transform(train_data)
    test_features = tfidf_vectorizer.transform(test_data)

    # Train a Logistic Regression model
    logistic_regression_model = LogisticRegression(max_iter=1500, random_state=22)
    logistic_regression_model.fit(train_features, train_labels)

    # Backup model if it exists
    if os.path.exists(LOGISTIC_REGRESSION_MODEL_PATH):
        os.replace(LOGISTIC_REGRESSION_MODEL_PATH, LOGISTIC_REGRESSION_MODEL_PATH + ".bak")

    # Backup vectorizer if it exists
    if os.path.exists(TFIDF_VECTORIZER_PATH):
        os.replace(TFIDF_VECTORIZER_PATH, TFIDF_VECTORIZER_PATH + ".bak")

    # Save the TF-IDF vectorizer and trained Logistic Regression model
    joblib.dump(tfidf_vectorizer, TFIDF_VECTORIZER_PATH)
    joblib.dump(logistic_regression_model, LOGISTIC_REGRESSION_MODEL_PATH)
```

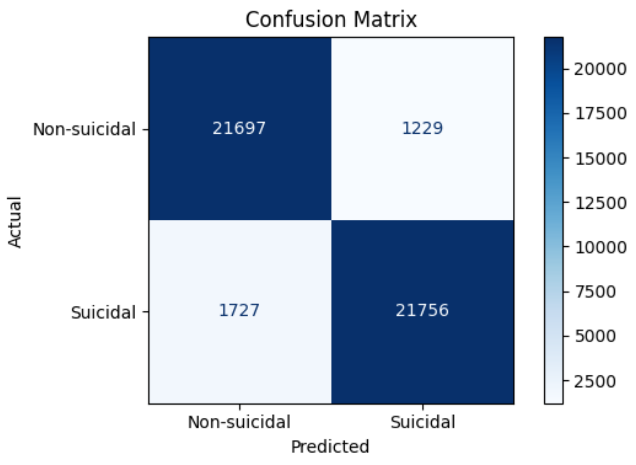
Code snippet showing the implementation of the logistic regression model.

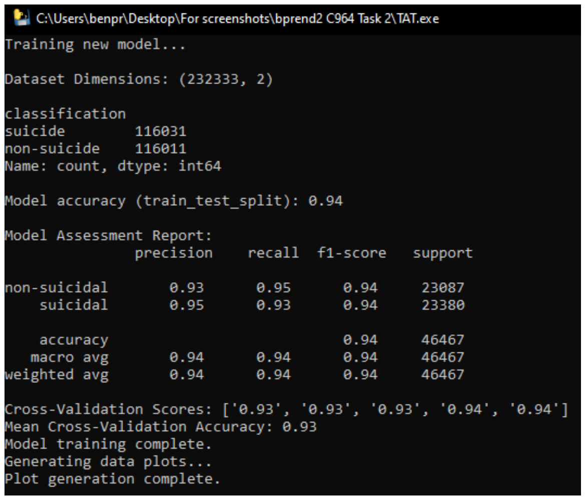
Validation

Two validation methods were implemented for assessing the performance and generalization ability of the logistic regression model: train-test split and k-fold cross-validation. The primary metric utilized for evaluating success was accuracy, representing the ratio of correct predictions to the total number of predictions generated by the model. For the implementation of this tool to be considered effective, a high accuracy rate was necessary. As such, a predefined benchmark of 90% accuracy was established as the threshold for success.

The first accuracy measurement occurred after the train-test split procedure. The model underwent training and then ran predictions on the testing subset. The prediction results were compared to the actual classifications using the scikit-learn accuracy_score function to obtain model accuracy. The scikit-learn classification_report function was run on the same result set to generate a report that included model precision, recall, and f1-score. The initial assessment revealed an accuracy rate of 94%, signaling proficient performance of the model. The results of both reports were printed to the console after the model training process was completed.

The results of the train-test split procedure were also used to generate a confusion matrix to help visualize the model performance. The confusion matrix corroborates the accuracy rate, illustrating a high rate of true positive and true negative predictions and significantly fewer false predictions.



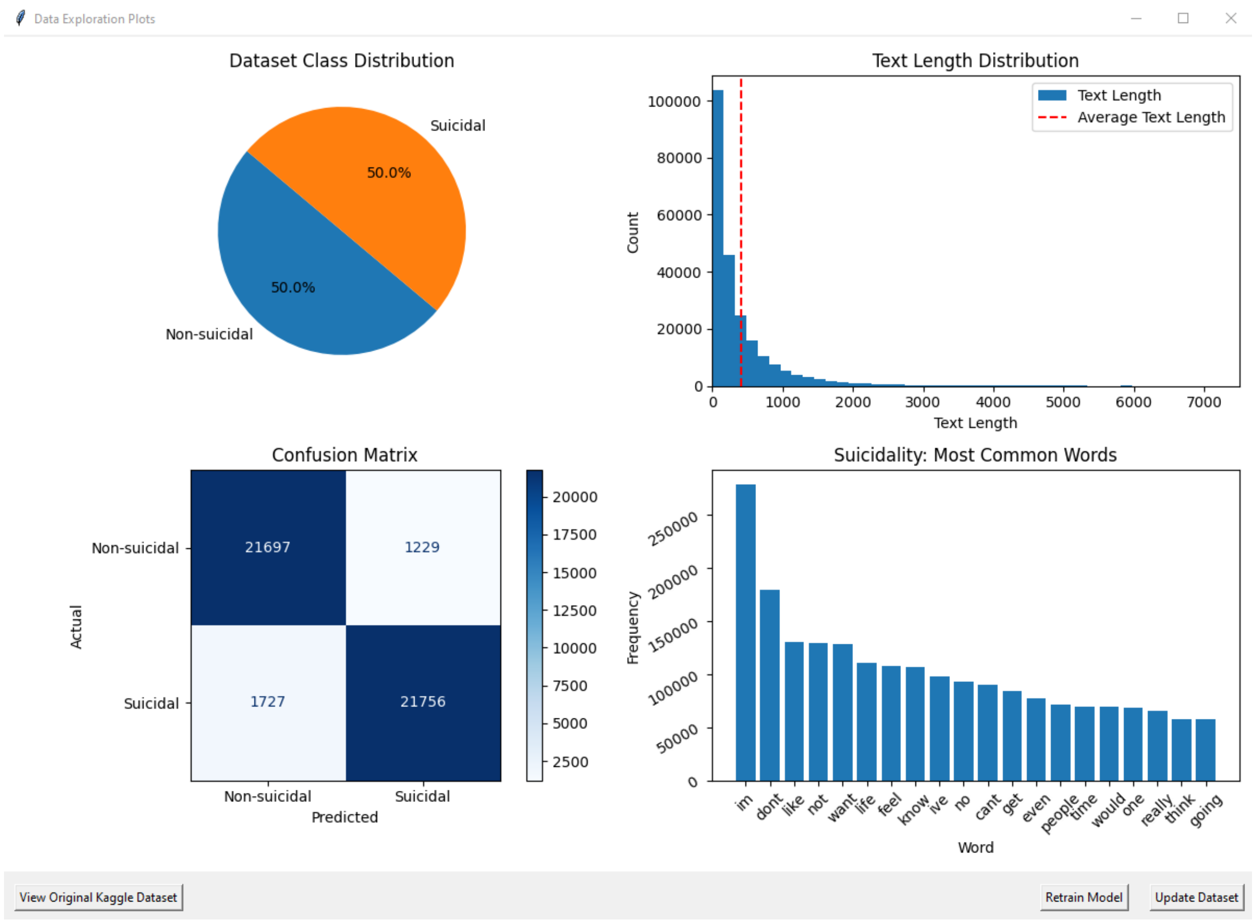


```
C:\Users\benpr\Desktop\For screenshots\bprend2 C964 Task 2\TAT.exe
Training new model...

Dataset Dimensions: (232333, 2)

classification
suicide        116031
non-suicide    116011
Name: count, dtype: int64

Model accuracy (train_test_split): 0.94

Model Assessment Report:
               precision    recall  f1-score   support

non-suicidal       0.93      0.95      0.94     23087
    suicidal       0.95      0.93      0.94     23380

    accuracy                           0.94     46467
   macro avg       0.94      0.94      0.94     46467
weighted avg       0.94      0.94      0.94     46467

Cross-Validation Scores: ['0.93', '0.93', '0.93', '0.94', '0.94']
Mean Cross-Validation Accuracy: 0.93
Model training complete.
Generating data plots...
Plot generation complete.
```

To obtain a more comprehensive understanding of how well the model generalized across different partitions of the data, k-fold cross-validation was also applied. The scikit-learn cross_val_score function was used to assess model performance on the entire dataset, testing the model on five portions of the dataset and printing the results to the console. The cross_val_score results corroborated the initial accuracy assessment with a mean score of 93%, affirming the model's effective functionality. By integrating both validation techniques, a comprehensive understanding of the model's performance and generalization ability was achieved.

Visualizations

Users with administrative privileges gained access to supplemental data and model-related features found within the Model and Data Tools section of the application. Among the additional data were four data plots generated during model training. These plots included a class distribution pie chart, a distribution of text lengths, a confusion matrix, and a bar graph illustrating word frequency. These visualizations were meant to offer insight into the performance of the model and characteristics of the dataset. Refreshed visualizations were generated upon updating the dataset and retraining the model, ensuring the current data was available.



Screenshot showing data visualizations contained within the application.

User Guide

The Transcript Analysis Tool is a standalone application designed to assist in the classification of telehealth therapy session transcripts. The application applies natural language processing to assess whether the statements made by patients indicate suicidality. Formatted transcript text files are placed in the ‘transcripts’ directory by the EHR system (three sample transcripts are included for testing purposes). Transcript analysis results are written to formatted text files and saved in the ‘output’ directory where they can be retrieved by the EHR system and processed into patient health records.

The following are instructions intended for evaluators and administrative users of the application and should not be shared with clinicians.

Downloading and Launching

1. Download the ‘bprend2 C964 Task 2.zip’ file from the WGU assessment submission platform.

2. Extract the contents of the zip file.
    2a. The extracted file should contain the main executable, TAT.exe, as well as ‘data’, ‘models’, ‘output’, and ‘transcripts’ directories. Within the ‘data’ directory should be the dataset, ‘dataset.csv’, and the most recently generated data plots, ‘data_exploration_plots.png’. Within the ‘models’ directory should be the trained logistic regression model and vectorizer, ‘logistic_regression_model.pkl’ and ‘tfidf_vectorizer.pkl’, respectively. The ‘output’ directory should be empty. Finally, the ‘transcripts’ directory should contain three sample transcript text files.

3. To launch the application, double-click TAT.exe

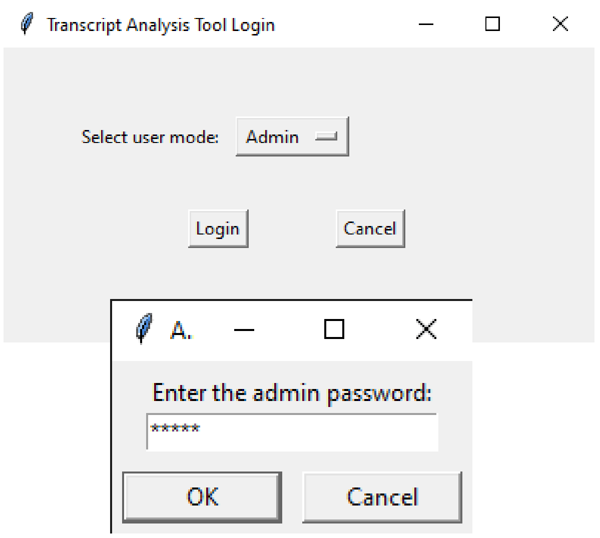
Mode Selection and Login

1. Select ‘Admin’ from the user mode menu.

2. Click Login.

3. Enter the admin password: **1234$**

4. Click OK.



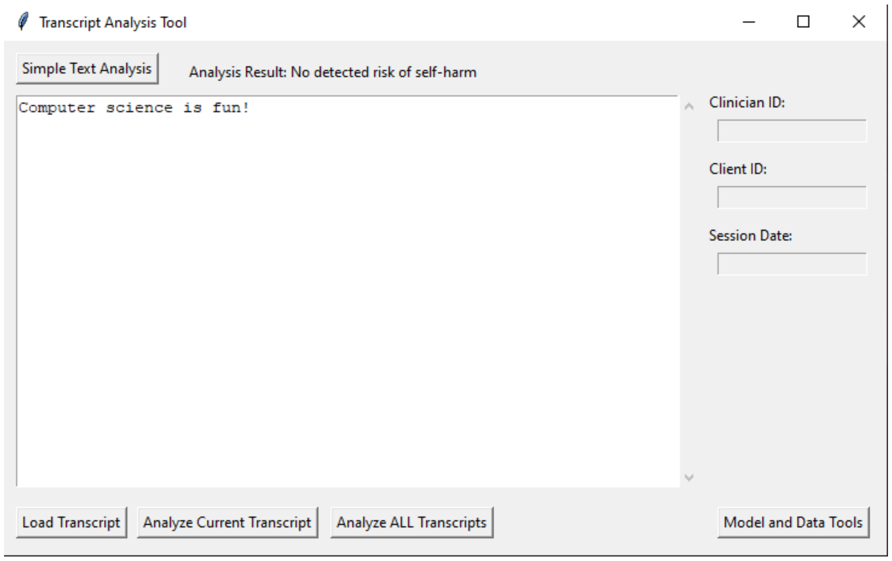
Analysis Tools

Simple Text Analysis: This feature enables the quick analysis of text typed or pasted into the main text field.

1. Type or paste text into the main text field.

2. Click Simple Text Analysis.

3. Review the result displayed above the main text field.



Analysis Result: No detected risk of self-harm

```
Computer science is fun!
```

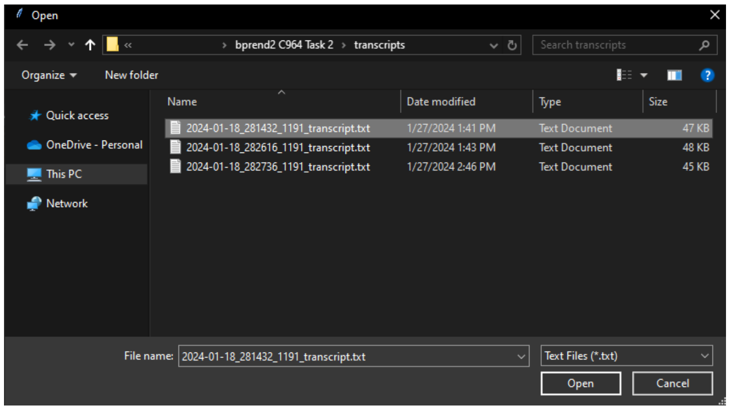
Individual Transcript Analysis: This feature allows the user to select a desired transcript for analysis. *Note: This functionality only works with properly formatted transcript files. Three sample files have been provided for testing purposes.*

1. Click Load Transcript.

2. Select the desired transcript from the file selection window and click Open.

3. Verify that the main text area, Clinician ID, Client ID, and Session Date fields have populated.

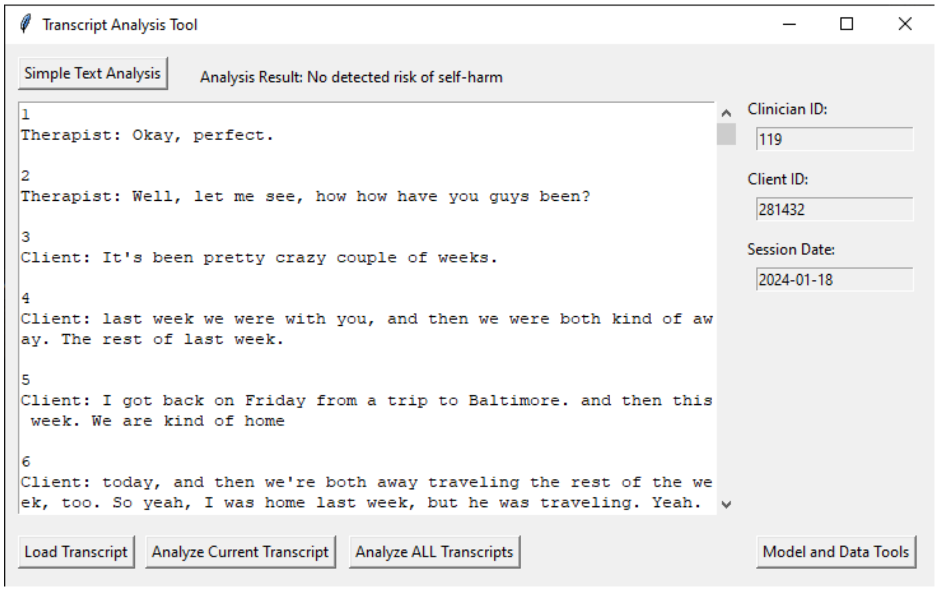
4. Click Analyze Current Transcript.



Name	Date modified	Type	Size
2024-01-18_281432_1191_transcript.txt	1/27/2024 1:41 PM	Text Document	47 KB
2024-01-18_282616_1191_transcript.txt	1/27/2024 1:43 PM	Text Document	48 KB
2024-01-18_282736_1191_transcript.txt	1/27/2024 2:46 PM	Text Document	45 KB

File name: 2024-01-18_281432_1191_transcript.txt

5. Review the results displayed above the main text field. *Note: This function also generates a result output file, saved to the 'output' directory, and formatted to be ingested by the EHR system.*



Analysis Result: No detected risk of self-harm

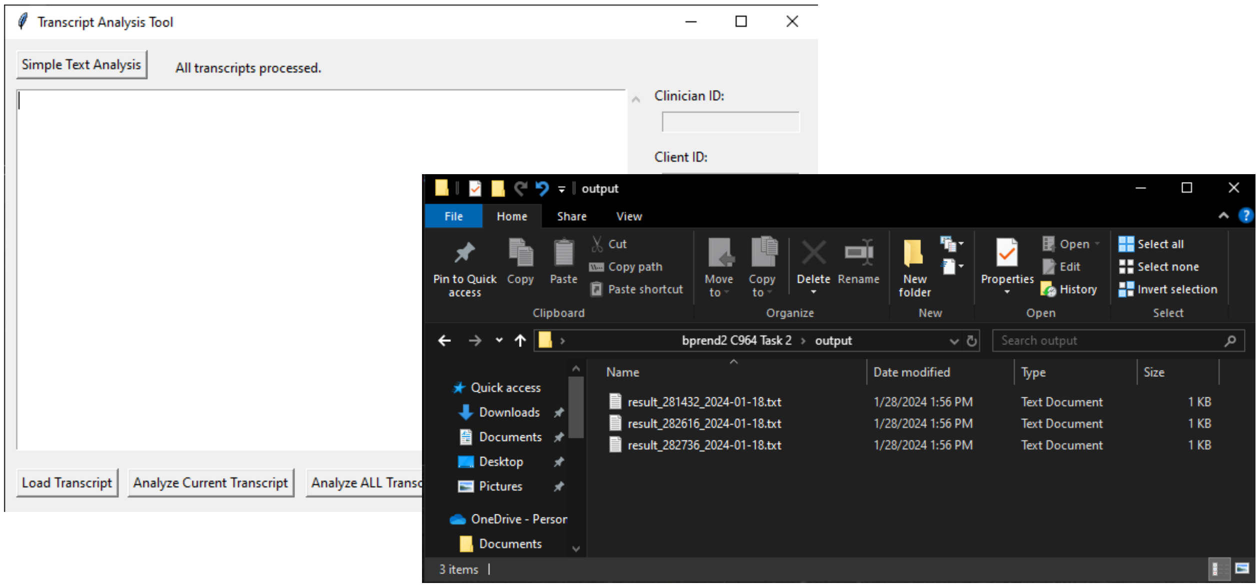
```
1
Therapist: Okay, perfect.

2
Therapist: Well, let me see, how how have you guys been?

3
Client: It's been pretty crazy couple of weeks.

4
Client: last week we were with you, and then we were both kind of aw
ay. The rest of last week.

5
Client: I got back on Friday from a trip to Baltimore. and then this
 week. We are kind of home

6
Client: today, and then we're both away traveling the rest of the we
ek, too. So yeah, I was home last week, but he was traveling. Yeah.
```

Clinician ID: 119
Client ID: 281432
Session Date: 2024-01-18

Batch Transcript Analysis: This function allows the user to analyze all transcripts currently in the 'transcripts' directory with a single click. *Note: This functionality only works with properly formatted transcript files. Three sample files have been provided for testing purposes.*

1. Click Analyze All Transcripts

2. Note the status message indicating that all transcripts were processed. *Note: This function also generates result output files for each analyzed transcript, saved to the 'output' directory, and formatted to be ingested by the EHR system.*



All transcripts processed.

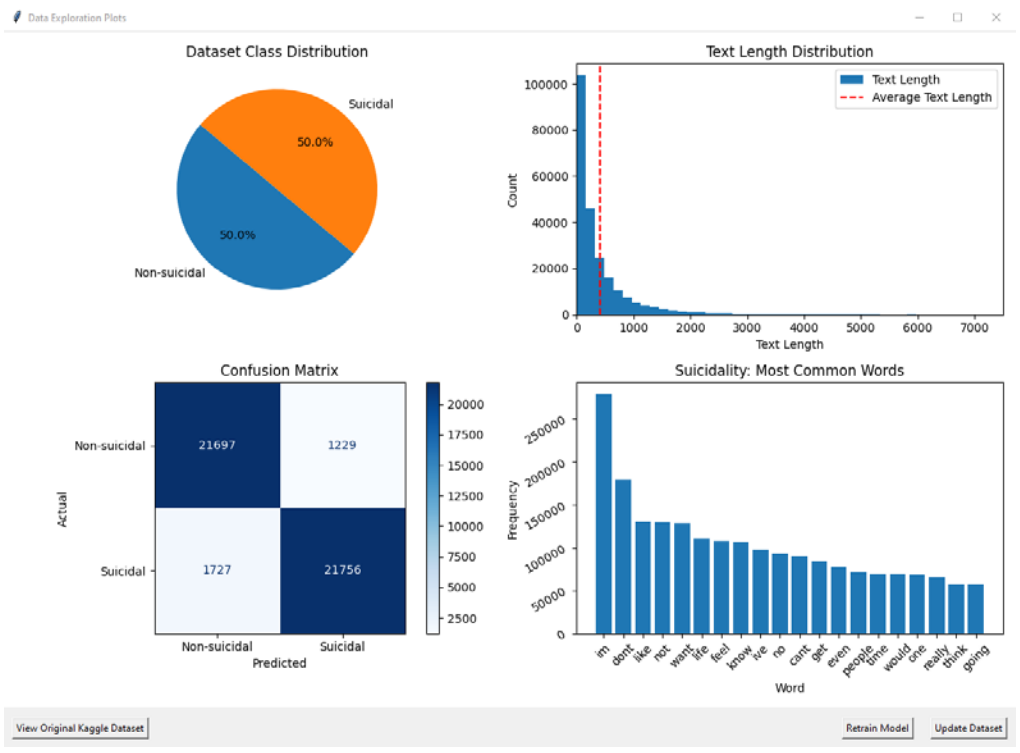
bprend2 C964 Task 2 › output

Name	Date modified	Type	Size
result_281432_2024-01-18.txt	1/28/2024 1:56 PM	Text Document	1 KB
result_282616_2024-01-18.txt	1/28/2024 1:56 PM	Text Document	1 KB
result_282736_2024-01-18.txt	1/28/2024 1:56 PM	Text Document	1 KB

3 items

Administrative Tools

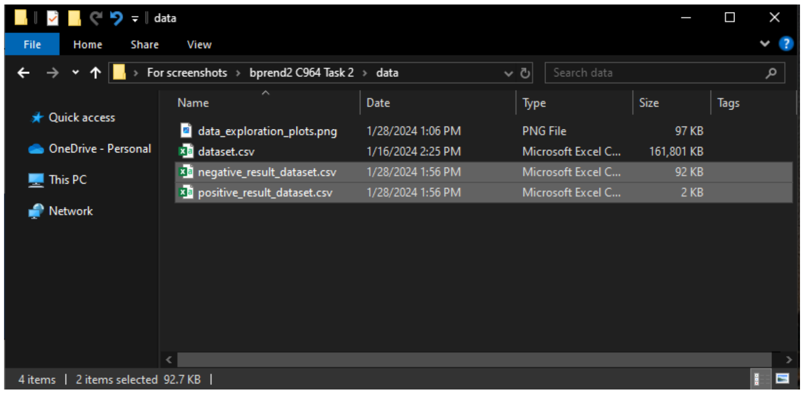
This area of the application contains information and tools related to the machine learning model and dataset and should only be accessed by knowledgeable users. To access, click Models and Data Tools from the main application window.



Updating the Dataset: This feature allows the user to append the existing dataset.csv with the results generated from prior transcript analysis tasks. The result sets generated from transcript analysis can be found in the 'data' directory, labeled negative_result_dataset.csv and positive_result_dataset.csv. It is best practice to manually examine these files for accuracy before triggering the dataset update.

1. To trigger the update, click Update Dataset.

2. Review dataset.csv to validate that the update succeeded. *Note: The existing dataset.csv will be backed up, and the result datasets will be consumed by this process.*



Name	Date	Type	Size	Tags
data_exploration_plots.png	1/28/2024 1:06 PM	PNG File	97 KB	
dataset.csv	1/16/2024 2:25 PM	Microsoft Excel C...	161,801 KB	
negative_result_dataset.csv	1/28/2024 1:56 PM	Microsoft Excel C...	92 KB	
positive_result_dataset.csv	1/28/2024 1:56 PM	Microsoft Excel C...	2 KB	

4 items | 2 items selected 92.7 KB |

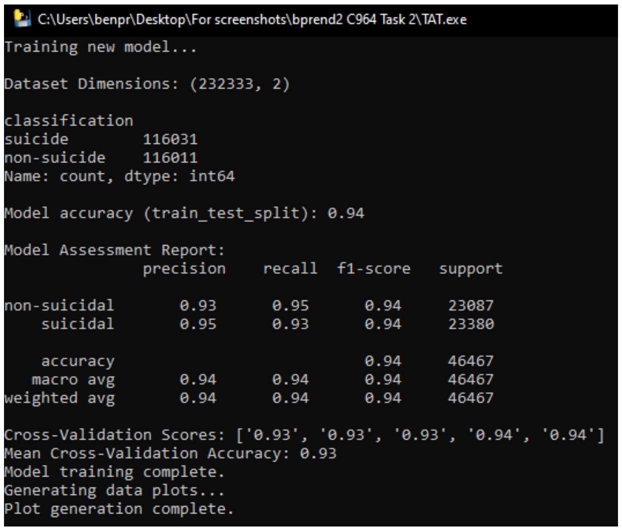
Retraining the Model: This feature allows the user to retrain the logistic regression model on the current dataset.csv. It is meant to improve the model as more data is collected and incorporated into the dataset.

1. To trigger model retraining, click Retrain Model.

2. Wait for the model training to complete.

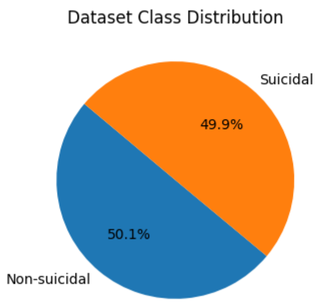
3. Review the model validation output in the console to verify model performance. *Note: the previous model and vectorizer will be backed up during this process.*

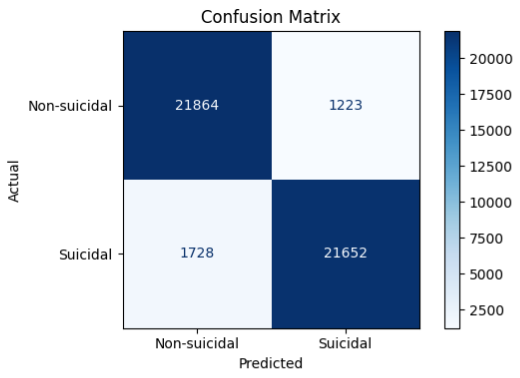
4. Review the updated data plots in the Model and Data Tools window to verify model performance and dataset characteristics.



C:\Users\benpr\Desktop\For screenshots\bprend2 C964 Task 2\TAT.exe

```
Training new model...

Dataset Dimensions: (232333, 2)

classification
suicide        116031
non-suicide    116011
Name: count, dtype: int64

Model accuracy (train_test_split): 0.94

Model Assessment Report:
              precision    recall  f1-score   support

non-suicidal       0.93      0.95      0.94     23087
    suicidal       0.95      0.93      0.94     23380

    accuracy                           0.94     46467
   macro avg       0.94      0.94      0.94     46467
weighted avg       0.94      0.94      0.94     46467

Cross-Validation Scores: ['0.93', '0.93', '0.93', '0.94', '0.94']
Mean Cross-Validation Accuracy: 0.93
Model training complete.
Generating data plots...
Plot generation complete.
```





Accessing the Original Dataset: To download the original dataset used for this project. Click View Original Kaggle Dataset. This will open the data source website in the default web browser. To obtain the data, click Download in the upper right of the screen.

Reference Page

Advantages and Disadvantages of Logistic Regression. (2023, January 10). *GeeksforGeeks*. *https://www.geeksforgeeks.org/advantages-and-disadvantages-of-logistic-regression/*

Karabiber, F. (2021, April 6). TF-IDF—Term Frequency-Inverse Document Frequency. *LearnDataSci*. https://www.learndatasci.com/glossary/tf-idf-term-frequency-inverse-document-frequency/