Planned Parenthood Team 1A

AI Studio Final Presentation

12/05/2024

BREAK THROUGH TECH

Meet Our Team!

Veronica Aragon
University of Texas at El Paso

Dureti Shemsi
North Seattle College

Natalie Kao
San Jose State University

Lily Eap
University of Illinois at Chicago

Introduction

Chatbots like Roo are designed to streamline communication and save time, but when they make mistakes, they can harm user satisfaction and trust.

This is where our project comes in.

We’re working on a model to detect errors in Roo’s responses, aiming to reduce mistakes and make chatbot interactions more reliable and effective.

AI Studio Project Overview

Project Challenge	Creating an error detection model for Roo, Planned Parenthood's AI chatbot, to identify and classify response accuracy
Data	Contains user conversations with Roo, response accuracy labels (TP/TN/FP/FN) & user demographics and feedback
ML Approach	Supervised classification using: Natural language processing, text vectorization techniques & multi-class classification models

Our Goal

- Develop an error detection model to improve Roo's response accuracy by categorizing outputs as True/False Positives/Negatives.

- Focus on reducing False Positives and False Negatives, leading to more accurate and reliable chatbot interactions.

Business Impact:

- Reach an outcome of enhanced user experience with high-quality, error-free interactions, building trust on sensitive topics like sexual and reproductive health.

Our Approach

1

Data Preprocessing

Cleaned data by removing unnecessary columns, Spanish prompts, and generic responses. Handled null values and prepared text data for analysis.

2

Feature Engineering

Applied text preprocessing techniques including lemmatization, stop word removal, and vectorization using TF-IDF and BERT embeddings.

3

Model Development

Implemented and evaluated multiple classification models including SVM, Logistic Regression, and LSTM to identify misfires.

4

Performance Analysis

Evaluated models using metrics like accuracy, precision and recall. Fine-tuned parameters for optimal performance.

Resources We Leveraged

- Scikit
- Python
- Numpy
- Pandas
- VSCode
- Jupyter Notebook
- Transformers (BERT)

Text Processing & Vectorization

Importance:

- Preparation for model analysis
  - numerical data
- Captures underlying structure or meaning of the data
- Different forms of processed texts have different advantages
- Overall improved efficiency

Findings:

- Most common topic: Questions about abortion costs and accessibility

Steps:

- Combined prompt and response text for comprehensive analysis
- Removed stopwords to focus on meaningful content
- Applied lemmatization to standardize word forms
- Feature selection

Data Understanding & Cleaning

Initial Dataset:

- 748 chat interactions between users and Roo
- 22 columns
- Primary focus
  - First_prompt
  - First_response
  - First_label columns

Preprocessing Steps:

- Filtered out
  - non-English prompts
  - automated messages
- Final clean dataset: 478 conversations
- Preserved unlabeled data for testing

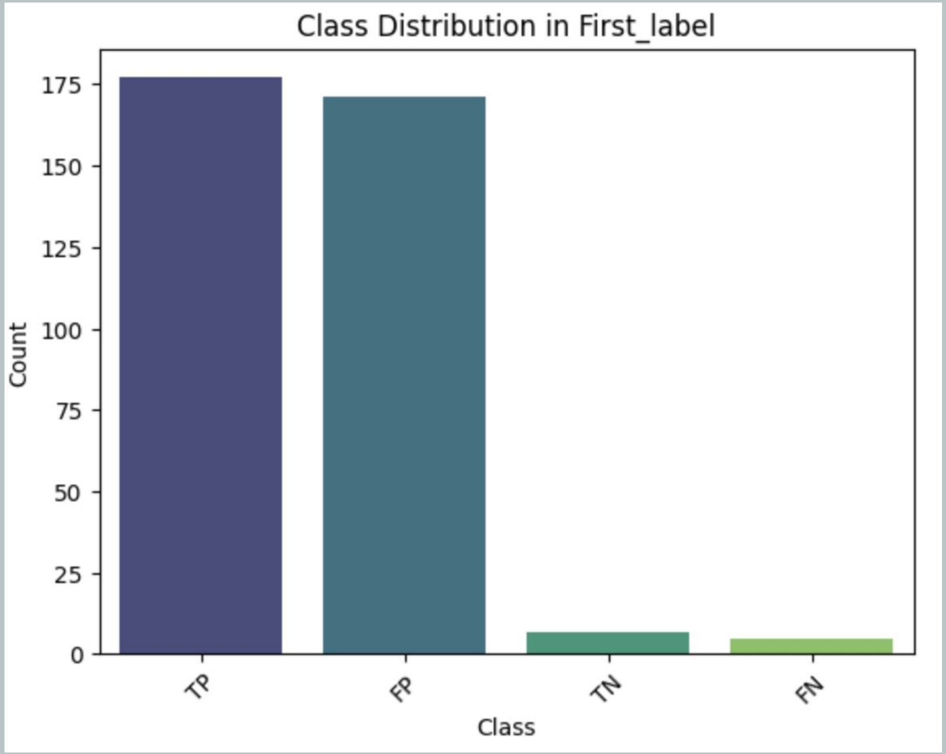
Genesys_interaction_id	Gender	Race	Topic	Age	Full_conversation	Comfort	Quality
56663169-94d3-4a0c-a427-2f4752f20f32	None	None	None	None	[{'text': 'Hey there, I'm Roo! You can ask me ...	None	None
9d8858a7-0821-4546-b69b-51f03e381830	None	None	None	None	[{'text': 'Hey there, I'm Roo! You can ask me ...	None	None
d56a9d48-a45a-4482-9dde-a826ede0c69b	None	None	None	None	[{'text': 'Hey there, I'm Roo! You can ask me ...	None	None
bc3f2707-70f2-4f1c-8080-11885cd0b60a	None	None	None	None	[{'text': 'Hola quiero sacarme una duda', 'use...	None	None
74c9a6bb-fc6c-49e9-8a34-8f90af8250ef	None	None	None	None	[{'text': 'All health educators are currently ...	None	None
b894636a-1bc4-4d80-9476-a60a3a8a79c0	None	None	None	None	[{'text': 'prompt:livechatinstant', 'user': 'c...	None	None

(snippet of the data)

Data Understanding & Cleaning

Key Challenges:

- Class imbalance
- Missing values in crucial fields
- Mix of English and Spanish interactions



Technical Dive

Vectorization Methods

	TF-IDF	BERT Embeddings
Desc.	$tf \times idf$ Term Frequency * Inverse Document Frequency	dynamic word representations by BERT based on context using deep learning
Output (Vector Representation)	Importance of words • Lemmatization (ex: running -> run) • Stop-words (ex: 'the', 'as', 'of')	dense, context-aware vectors for words
Complexity	Simple & faster to compute	Computationally intensive

Post-Processed Data

Vectorization Methods

TF-IDF Lemmatization:

```
TF-IDF DataFrame for First Prompt:
      15  abortion  ago  appointment  birth  bleeding  boyfriend  came  chat  \
0    0.0       0.0  0.0          0.0    0.0  0.000000    0.00000   0.0   0.0
1    0.0       0.0  0.0          0.0    0.0  0.000000    0.00000   0.0   0.0
2    0.0       0.0  0.0          0.0    0.0  0.000000    0.00000   0.0   0.0
3    0.0       0.0  0.0          0.0    0.0  0.494423    0.50936   0.0   0.0
4    0.0       0.0  0.0          0.0    0.0  0.000000    0.00000   0.0   0.0
..   ...       ...  ...          ...    ...       ...        ...   ...   ...
483  0.0       0.0  0.0          0.0    0.0  0.000000    0.00000   0.0   0.0
484  0.0       0.0  0.0          0.0    0.0  0.000000    0.00000   0.0   0.0
485  0.0       0.0  0.0          0.0    0.0  0.000000    0.00000   0.0   0.0
486  0.0       0.0  0.0          0.0    0.0  0.000000    0.00000   0.0   0.0
487  0.0       0.0  0.0          0.0    0.0  0.000000    0.00000   0.0   0.0

     come  ...  unprotected  use    vagina        ve  wait      want     week  \
0     0.0  ...          0.0  0.0  0.000000  0.000000   0.0  0.000000  0.00000
1     0.0  ...          0.0  0.0  0.000000  0.000000   0.0  0.000000  0.00000
2     0.0  ...
3     0.0  ...
4     0.0  ...
..    ...  ...
483   0.0  ...
484   0.0  ...
485   0.0  ...
486   0.0  ...
487   0.0  ...

     weeks  work  wor
0      0.0   0.0
1      0.0   0.0
```

```
[488 rows x 90 columns]
Columns in Prompt DataFrame: Index(['15', 'abortion', 'ago', 'appointment', 'birth', 'bleeding',
       'boyfriend', 'came', 'chat', 'come', 'condom', 'control', 'cost', 'cum',
       'cycle', 'day', 'days', 'depo', 'did', 'didn', 'doctor', 'does', 'don',
       'early', 'getting', 'got', 'happen', 'having', 'health', 'hello',
       'help', 'hi', 'hours', 'human', 'im', 'implant', 'inside', 'insurance',
       'iud', 'june', 'just', 'know', 'late', 'like', 'long', 'mean',
       'medication', 'missed', 'month', 'months', 'morning', 'need',
       'negative', 'new', 'normal', 'okay', 'parenthood', 'period', 'pill',
       'pills', 'plan', 'planned', 'pregnancy', 'pregnant', 'protected',
       'question', 'right', 'ring', 'says', 'sex', 'shot', 'speak', 'start',
       'started', 'std', 'taking', 'talk', 'test', 'time', 'took',
       'unprotected', 'use', 'vagina', 've', 'wait', 'want', 'week', 'weeks',
       'work', 'worried'],
      dtype='object')
```

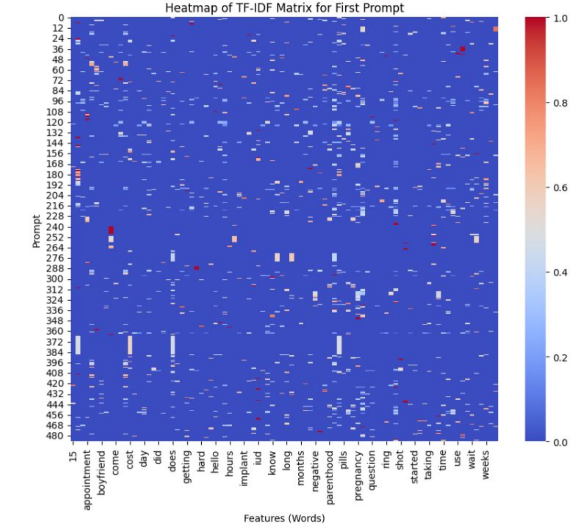
BERT Embeddings:

	numeric_label	embeddings
0	NaN	[0.21650272607803345, -0.15849028527736664, 0....
1	0.0	[0.10410784929990768, -0.12331955134868622, 0....
2	0.0	[0.17700445652008057, 0.020377160981297493, 0....
3	1.0	[0.2062266319990158, -0.09583471715450287, 0.2...
4	0.0	[0.11256039887666702, -0.1905854344367981, 0.3...
...	...	...
485	1.0	[0.07400229573249817, -0.0828939899802208, 0.0...
486	1.0	[0.17363932728767395, -0.1340048909187317, 0.3...
487	1.0	[0.32503053545951843, -0.015318689867854118, 0....
488	0.0	[0.4333673417568207, -0.11571981012821198, 0.2...
489	1.0	[0.45228031277656555, -0.13108395040035248, 0....

Post-Processed Data

TF-IDF Stopwords:

```
[482 rows x 90 columns]
Columns in Prompt DataFrame: Index(['15', 'abortion', 'ago', 'appointment', 'birth'
ing',
       'boyfriend', 'came', 'chat', 'come', 'condom', 'control', 'cost', 'cum',
       'cycle', 'day', 'days', 'did', 'didn', 'doctor', 'does', 'don', 'early',
       'getting', 'got', 'happen', 'hard', 'having', 'health', 'hello', 'help',
       'hi', 'hours', 'human', 'im', 'inside', 'insurance', 'iud', 'june',
       'just', 'know', 'late', 'like', 'long', 'mean', 'month', 'months',
       'morning', 'need', 'negative', 'new', 'normal', 'okay', 'old',
       'parenthood', 'period', 'pill', 'pills', 'plan', 'planned', 'pregnancy',
       'pregnant', 'protected', 'question', 'results', 'right', 'ring', 'says',
       'sex', 'shot', 'speak', 'start', 'started', 'std', 'sure', 'taking',
       'talk', 'test', 'time', 'took', 'unprotected', 'use', 'vagina', 've',
       'wait', 'want', 'week', 'weeks', 'work', 'worried'],
      dtype='object')
```




```
TF-IDF DataFrame for First Prompt:
      15  abortion  ago  appointment  birth  bleeding  boyfriend  came  chat  \
0    0.0       0.0  0.0          0.0    0.0  0.000000   0.000000   0.0   0.0
1    0.0       0.0  0.0          0.0    0.0  0.000000   0.000000   0.0   0.0
2    0.0       0.0  0.0          0.0    0.0  0.000000   0.000000   0.0   0.0
3    0.0       0.0  0.0          0.0    0.0  0.489243   0.511095   0.0   0.0
4    0.0       0.0  0.0          0.0    0.0  0.000000   0.000000   0.0   0.0
..   ...       ...  ...          ...    ...       ...        ...   ...   ...
477  0.0       0.0  0.0          0.0    0.0  0.000000   0.000000   0.0   0.0
478  0.0       0.0  0.0          0.0    0.0  0.000000   0.000000   0.0   0.0
479  0.0       0.0  0.0          0.0    0.0  0.000000   0.000000   0.0   0.0
480  0.0       0.0  0.0          0.0    0.0  0.000000   0.000000   0.0   0.0
481  0.0       0.0  0.0          0.0    0.0  0.000000   0.000000   0.0   0.0

     come  ...  unprotected  use    vagina       ve  wait      want      week  \
0     0.0  ...          0.0  0.0  0.000000  0.00000   0.0  0.000000  0.000000
1     0.0  ...          0.0  0.0  0.000000  0.00000   0.0  0.000000  0.000000
2     0.0  ...          0.0  0.0  0.000000  0.00000   0.0  0.000000  0.000000
3     0.0  ...          0.0  0.0  0.000000  0.50333   0.0  0.000000  0.000000
4     0.0  ...          0.0  0.0  0.000000  0.00000   0.0  0.000000  0.667237
..    ...  ...          ...  ...       ...      ...   ...       ...       ...
477   0.0  ...          0.0  0.0  0.695208  0.00000   0.0  0.000000  0.000000
478   0.0  ...          0.0  0.0  0.000000  0.00000   0.0  0.000000  0.000000
479   0.0  ...          0.0  0.0  0.000000  0.00000   0.0  0.000000  0.000000
480   0.0  ...          0.0  0.0  0.000000  0.00000   0.0  0.541008  0.000000
481   0.0  ...          0.0  0.0  0.000000  0.00000   0.0  0.000000  0.000000

     weeks  work  worried
0      0.0   0.0      0.0
1      0.0   0.0      0.0
2      0.0   0.0      0.0
3      0.0   0.0      0.0
4      0.0   0.0      0.0
..     ...   ...      ...
477    0.0   0.0      0.0
478    0.0   0.0      0.0
479    0.0   0.0      0.0
480    0.0   0.0      0.0
481    0.0   0.0      0.0
```

Modeling & Evaluation

Models Covered in This Project

Support Vector Machine (SVM)

Logistic Regression (LR)

Long Short-Term Memory (LSTM)

Support Vector Machine (SVM)

What is SVM?

- A machine learning algorithm that finds the optimal boundary to separate data into classes by maximizing the margin between them, using mathematical tricks like kernels to handle complex patterns.

Why We Chose It

- Effective for Binary and Multi-class Classification
- High-dimensional Feature Spaces
    - Effective in handling high-dimensional spaces and finding optimal hyperplane that separates the classes
- Robust to Overfitting in High-dimensional Data
- Flexibility with Kernels
    - SVMs allow the use of different kernel functions (linear, polynomial, radial basis function (RBF), etc.) to capture complex relationships between input features and output classes

How Did We Implement It?

- Test with different kernels (linear, poly, RBF, etc), and different parameters and degrees

SVM Results

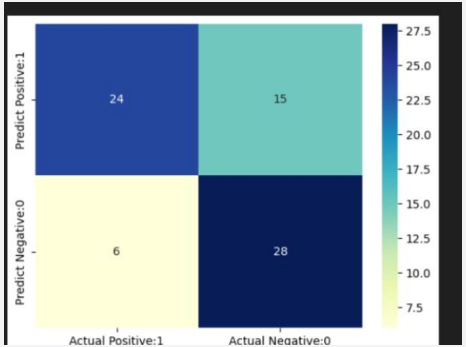
BERT Embeddings

```
 [10 21  0  0]
 [ 0  0  0  1]
 [ 0  0  0  0]]
              precision    recall  f1-score   support

         0.0       0.76      0.74      0.75        42
         1.0       0.66      0.68      0.67        31
         2.0       0.00      0.00      0.00         1
         3.0       0.00      0.00      0.00         0

    accuracy                           0.70        74
   macro avg       0.35      0.35      0.35        74
weighted avg       0.70      0.70      0.70        74

Classification accuracy : 0.7123
Classification error : 0.2877
```



Best performing model:

- (C=1, degree=2, gamma=0.01, kernel='poly')
  Best performing model by accuracy:

```
Model accuracy score: 0.7027
[[31 11  0  0]
 [10 21  0  0]
 [ 0  0  0  1]
 [ 0  0  0  0]]
              precision    recall  f1-score   support

         0.0       0.76      0.74      0.75        42
         1.0       0.66      0.68      0.67        31
         2.0       0.00      0.00      0.00         1
         3.0       0.00      0.00      0.00         0

    accuracy                           0.70        74
   macro avg       0.35      0.35      0.35        74
weighted avg       0.70      0.70      0.70        74

Classification accuracy : 0.7123
Classification error : 0.2877
```

TF-IDF Lemmatized

```
Classification Report:
              precision    recall  f1-score   support

          FN       0.00      0.00      0.00         1
          FP       0.69      0.74      0.71        34
          TN       0.50      1.00      0.67         1
          TP       0.74      0.69      0.71        36

    accuracy                           0.71        72
   macro avg       0.48      0.61      0.52        72
weighted avg       0.70      0.71      0.70        72

Confusion Matrix:
 [[ 0  0  1  0]
 [ 0 25  0  9]
 [ 0  0  1  0]
 [ 0 11  0 25]]
```

TF-IDF StopWords (Prompt + Response)

```
Model accuracy score: 0.5342
[[19 17  1  0]
 [13 20  1  0]
 [ 0  1  0  0]
 [ 1  0  0  0]]
              precision    recall  f1-score   support

         0.0       0.58      0.51      0.54        37
         1.0       0.53      0.59      0.56        34
         2.0       0.00      0.00      0.00         1
         3.0       0.00      0.00      0.00         1

    accuracy                           0.53        73
   macro avg       0.28      0.28      0.27        73
weighted avg       0.54      0.53      0.53        73

Classification accuracy : 0.5652
Classification error : 0.4348
```

Logistic Regression (LR)

What is Logistic Regression?

- A model that predicts categories (e.g., TP, FP, TN, FN) by calculating probabilities and classifying based on thresholds.

Why Did We Use It?

1. Simple and Effective: Ideal for classifying chatbot responses.
2. Great for Text Data: Works with TF-IDF and BERT for converting text into numbers.
3. Handles Imbalances: Uses "class weights" to focus on rare cases (e.g., FN).
4. Fast and Reliable: Quick to train and efficient for small datasets.

How Did We Implement It?

- Merged Text Features: Combined "prompt" and "response" into a single text field to create richer context, improving the model's ability to identify patterns in chatbot responses.
- Class Weights for Imbalance: Addressed the class imbalance (e.g., limited FN examples) by applying class weights, ensuring the model gives more importance to underrepresented categories.

LR Results

TF-IDF Lemmatized (Prompt + Response):

- Accuracy: 77% (Best performance overall)
- Strong with TP and FP but weak for FN due to limited examples.

Classification Report:

	precision	recall	f1-score	support
FN	0.00	0.00	0.00	1
FP	0.76	0.85	0.81	34
TN	0.40	1.00	0.57	2
TP	0.83	0.69	0.76	36
accuracy			0.77	73
macro avg	0.50	0.64	0.53	73
weighted avg	0.78	0.77	0.76	73

TF-IDF StopWords (Prompt Only):

- Accuracy: 56%
- Lower accuracy due to less detailed inputs.

Model accuracy score: 0.5616

```
[[25 12  0  0]
 [18 16  0  0]
 [ 1  0  0  0]
 [ 1  0  0  0]]
```

	precision	recall	f1-score	support
0.0	0.56	0.68	0.61	37
1.0	0.57	0.47	0.52	34
2.0	0.00	0.00	0.00	1
3.0	0.00	0.00	0.00	1
accuracy			0.56	73
macro avg	0.28	0.29	0.28	73
weighted avg	0.55	0.56	0.55	73

BERT Embeddings:

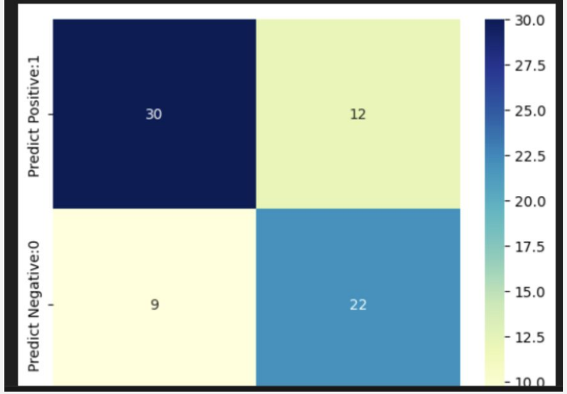
- Accuracy: 71%
- Better for understanding context but slightly behind TF-IDF (Prompt + Response).

Model accuracy score: 0.7162

```
[[30 12  0]
 [ 9 22  0]
 [ 0  0  1]]
```

	precision	recall	f1-score	support
0	0.77	0.71	0.74	42
1	0.65	0.71	0.68	31
2	1.00	1.00	1.00	1
accuracy			0.72	74
macro avg	0.81	0.81	0.81	74
weighted avg	0.72	0.72	0.72	74

Classification accuracy : 0.7123
Classification error : 0.2877



Long Short-Term Memory (LSTM)

What is LSTM?

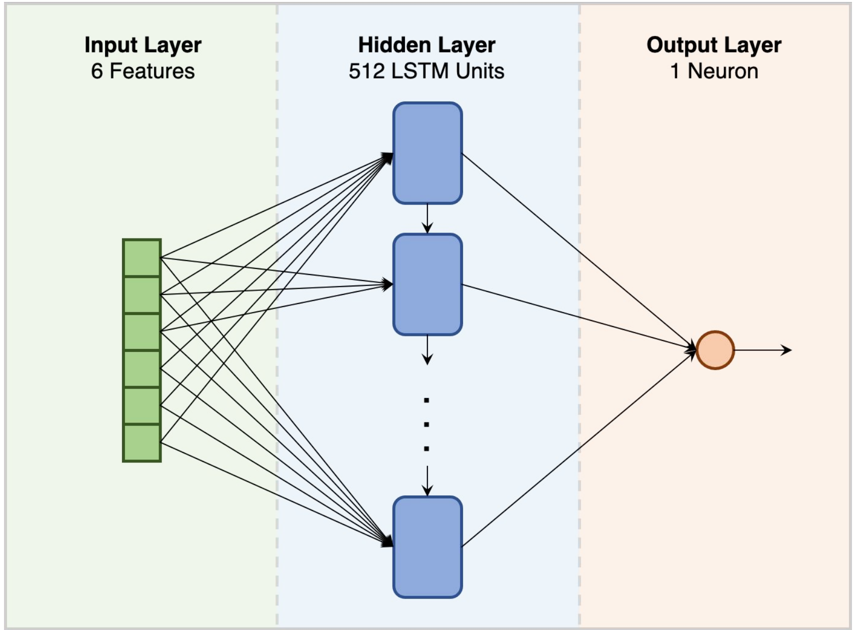
- Type of RNN that can remember important patterns in sequential data over long period
- uses “gates” like decision-makers

Why did we use it?

- particularly good for sequential data
- handles important context while ignoring irrelevant details

How did we implement it?

- deep learning framework: Keras (with TensorFlow) / PyTorch



(example architecture of LSTM)

LSTM Results

TF-IDF Stopwords(Prompt Only):

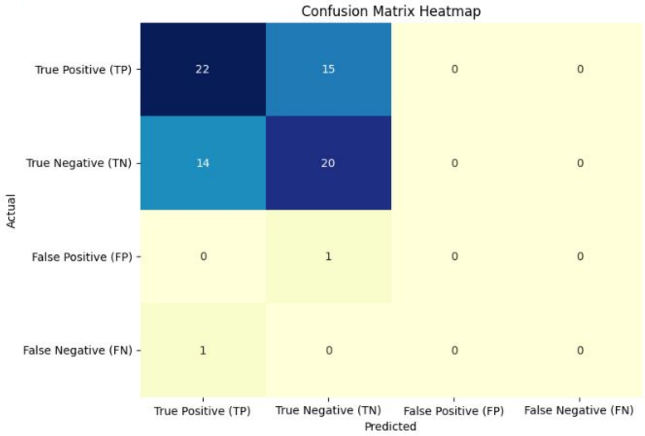
- Accuracy: 58%

```
Non-trainable params: 0 (0.00 B)

Epoch 1/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 8s 201ms/step - accuracy: 0.1252 - loss: 1.3898 - val_accuracy: 0.3621 - val_loss: 1.3682
Epoch 2/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 41ms/step - accuracy: 0.5711 - loss: 1.3618 - val_accuracy: 0.4655 - val_loss: 1.3452
Epoch 3/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 42ms/step - accuracy: 0.6001 - loss: 1.3343 - val_accuracy: 0.4310 - val_loss: 1.3193
Epoch 4/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 43ms/step - accuracy: 0.5740 - loss: 1.3014 - val_accuracy: 0.4483 - val_loss: 1.2867
Epoch 5/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 46ms/step - accuracy: 0.5634 - loss: 1.2602 - val_accuracy: 0.4310 - val_loss: 1.2466
Epoch 6/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 51ms/step - accuracy: 0.4908 - loss: 1.2187 - val_accuracy: 0.4310 - val_loss: 1.1969
Epoch 7/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 49ms/step - accuracy: 0.5298 - loss: 1.1584 - val_accuracy: 0.4138 - val_loss: 1.1399
Epoch 8/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 53ms/step - accuracy: 0.5526 - loss: 1.0776 - val_accuracy: 0.4310 - val_loss: 1.0783
Epoch 9/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 44ms/step - accuracy: 0.4981 - loss: 1.0354 - val_accuracy: 0.4483 - val_loss: 1.0123
Epoch 10/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 43ms/step - accuracy: 0.5578 - loss: 0.9461 - val_accuracy: 0.4483 - val_loss: 0.9571
Epoch 11/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 47ms/step - accuracy: 0.5720 - loss: 0.8785 - val_accuracy: 0.4310 - val_loss: 0.9173
Epoch 12/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 45ms/step - accuracy: 0.6147 - loss: 0.8486 - val_accuracy: 0.3966 - val_loss: 0.8735
Epoch 13/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 43ms/step - accuracy: 0.6734 - loss: 0.7997 - val_accuracy: 0.3793 - val_loss: 0.8472
Epoch 14/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 42ms/step - accuracy: 0.6814 - loss: 0.7883 - val_accuracy: 0.5000 - val_loss: 0.8196
Epoch 15/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 42ms/step - accuracy: 0.7639 - loss: 0.7148 - val_accuracy: 0.4828 - val_loss: 0.8146
Epoch 16/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 1s 42ms/step - accuracy: 0.7282 - loss: 0.7513 - val_accuracy: 0.5000 - val_loss: 0.8056
Epoch 17/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 43ms/step - accuracy: 0.7277 - loss: 0.7173 - val_accuracy: 0.4655 - val_loss: 0.8136
Epoch 18/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 41ms/step - accuracy: 0.7056 - loss: 0.7199 - val_accuracy: 0.5345 - val_loss: 0.8028
Epoch 19/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 44ms/step - accuracy: 0.7179 - loss: 0.6579 - val_accuracy: 0.5345 - val_loss: 0.8054
Epoch 20/20
8/8 ━━━━━━━━━━━━━━━━━━━━ 0s 50ms/step - accuracy: 0.7233 - loss: 0.6671 - val_accuracy: 0.5345 - val_loss: 0.8135
```

```
3/3 ━━━━━━━━━━━━━━━━━━━━ 1s 310ms/step
Model accuracy score: 0.5753

Confusion Matrix:
[[22 15  0  0]
 [14 20  0  0]
 [ 0  1  0  0]
 [ 1  0  0  0]]
```




```
Classification Report:
                    precision    recall  f1-score   support

 True Positive (TP)      0.59      0.59      0.59        37
 True Negative (TN)      0.56      0.59      0.57        34
False Positive (FP)      0.00      0.00      0.00         1
False Negative (FN)      0.00      0.00      0.00         1

           accuracy                          0.58        73
          macro avg      0.29      0.30      0.29        73
       weighted avg      0.56      0.58      0.57        73
```

LSTM Results

TF-IDF Lemmatized (Prompt + Response):

- Accuracy: 75%

BERT Embeddings:

- Accuracy: 75%
- Perfect for capturing semantics per word and context

Classification Report:

	precision	recall	f1-score	support
FN	0.00	0.00	0.00	1
FP	0.76	0.82	0.79	34
TN	0.00	0.00	0.00	2
TP	0.75	0.75	0.75	36
accuracy			$\boxed{0.75}$	73
macro avg	0.38	0.39	0.38	73
weighted avg	0.72	0.75	0.74	73

```
Training model with 256 LSTM units, 0.5 dropout, and 256 dense units...
Epoch 1/10
10/10 ━━━━━━━━━━━━━━━━━━━━ 107s 772ms/step - accuracy: 0.4405 - loss: 0.9
Epoch 2/10
10/10 ━━━━━━━━━━━━━━━━━━━━ 7s 638ms/step - accuracy: 0.6326 - loss: 0.660
Epoch 3/10
10/10 ━━━━━━━━━━━━━━━━━━━━ 6s 659ms/step - accuracy: 0.6415 - loss: 0.606
Epoch 4/10
10/10 ━━━━━━━━━━━━━━━━━━━━ 6s 603ms/step - accuracy: 0.7273 - loss: 0.581
Epoch 5/10
10/10 ━━━━━━━━━━━━━━━━━━━━ 6s 592ms/step - accuracy: 0.7304 - loss: 0.558
Epoch 6/10
10/10 ━━━━━━━━━━━━━━━━━━━━ 6s 582ms/step - accuracy: 0.6896 - loss: 0.593
Epoch 7/10
10/10 ━━━━━━━━━━━━━━━━━━━━ 6s 602ms/step - accuracy: 0.7920 - loss: 0.521
Epoch 8/10
10/10 ━━━━━━━━━━━━━━━━━━━━ 6s 622ms/step - accuracy: 0.7782 - loss: 0.489
Epoch 9/10
10/10 ━━━━━━━━━━━━━━━━━━━━ 6s 601ms/step - accuracy: 0.7695 - loss: 0.524
Epoch 10/10
10/10 ━━━━━━━━━━━━━━━━━━━━ 6s 617ms/step - accuracy: 0.8088 - loss: 0.459
3/3 ━━━━━━━━━━━━━━━━━━━━ 1s 169ms/step - accuracy: 0.7400 - loss: 0.8016
Validation accuracy: 0.7534246444702148
```

Model Comparison

Model Name	Description	Results	Pros	Cons
SVM	Uses a machine learning algorithm to classify Roo's responses into four categories.	71% accuracy; best at identifying True Positives (TP) and False Positives (FP).	Good at handling high-dimensional data and creating clear class separation.	Struggles with False Negatives (FN) due to few examples; some performance drop on minority classes.
Logistic Regression	Applies a statistical model to classify Roo's responses and identify patterns for improvement.	77% accuracy; performs best on TP and FP, but fails to predict FN and struggles with True Negatives (TN).	Easy to interpret; handles imbalanced data better than SVM.	Limited by fewer training examples for FN and TN; lower performance on minority classes.
LSTM	Uses a neural network to detect errors in Roo's responses by learning long-term patterns.	75% accuracy; More stable than simpler models, Struggles with minority classes (FN and TN)	Captures relationships in sequential data, which helps in understanding context in chatbot interactions.	Computationally expensive; requires significant data and tuning to avoid overfitting or poor predictions.

Insights & Key Findings

Performance Visualization:

- Confusion matrices revealed that misclassifications occurred more often between False Positives (FP) and True Positives (TP).

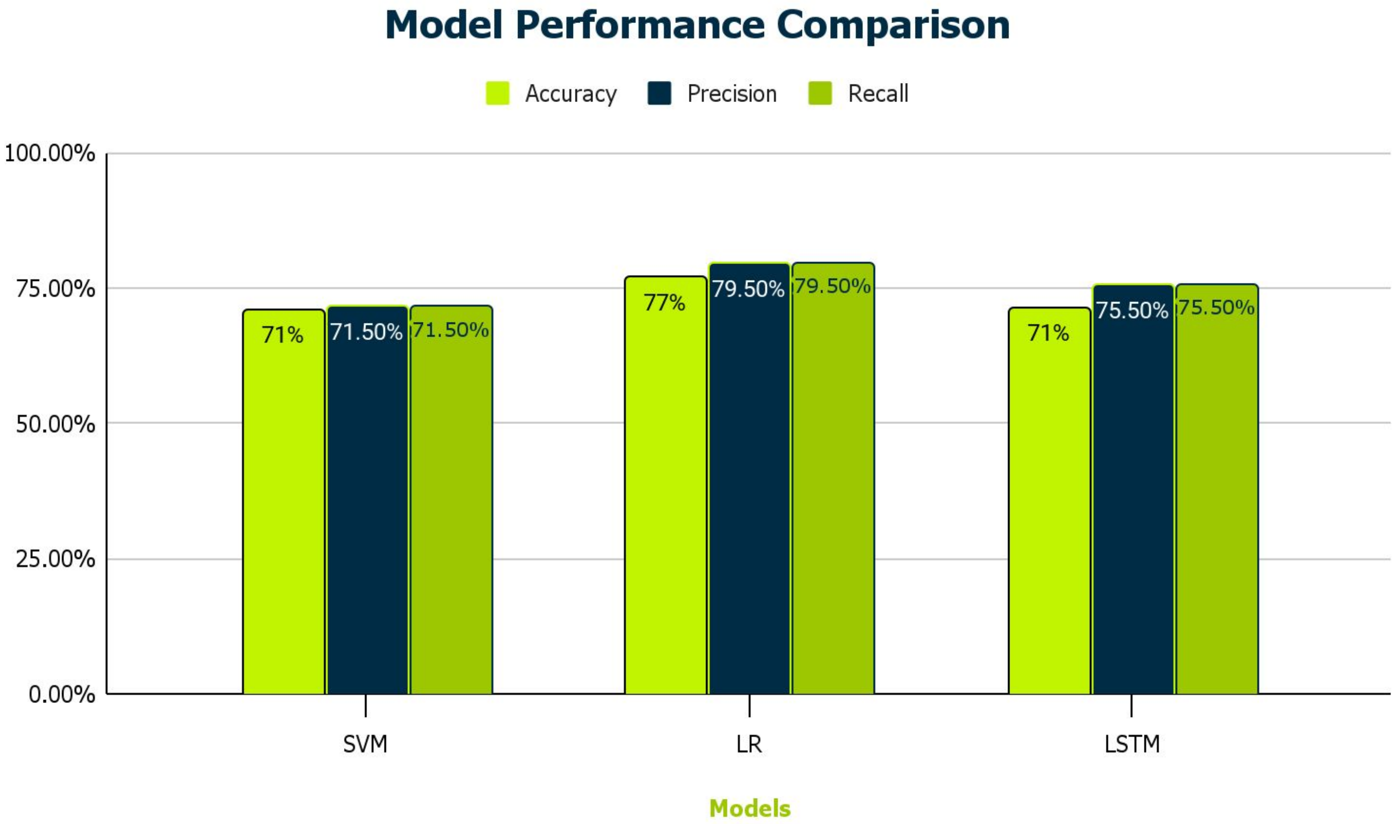
Data Balance Matters:

- All models struggled with classes that had fewer examples (e.g., FN and TN). This highlights the importance of balancing data for training.

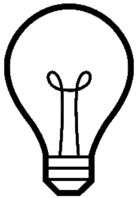
Accuracy Trends:

- Logistic Regression had the highest accuracy (77%), showing strong performance with straightforward classification tasks.

Insights & Key Findings



What We Learned



Model Selection Matters:

- Different models excel at different tasks. Logistic Regression worked well for simpler relationships, while SVM captured more complex ones.

Importance of Context:

- LSTM's ability to understand sequential data offers a significant advantage in capturing context, which may improve predictions for conversational models.

Iterative Testing is Key:

- Trying different configurations (e.g., kernel types for SVM, embeddings for features) helps identify the best-performing model.

Potential Next Steps

- Focus on collecting additional data, especially for underrepresented classes (FN and TN), to improve model accuracy across all categories.
- Fine-tune hyperparameters for SVM and Logistic Regression and explore advanced embedding techniques for LSTM to enhance performance.
- Consider ensemble methods to combine the strengths of different models for better overall classification.

Final Thoughts

- Highest performing model: LR model trained on TF-IDF
    - 77% accuracy
- Consider underrepresented FN and TN classes.
- Throughout this project, we sharpened our skills in NLP, model evaluation, and cross-functional collaboration.

Thank you to our AI Studio TA & Challenge Advisors!

Jumana Rahman
AI Studio TA

Michael O’Keefe
Challenge Advisor

Michelle Bao
Challenge Advisor

Questions?