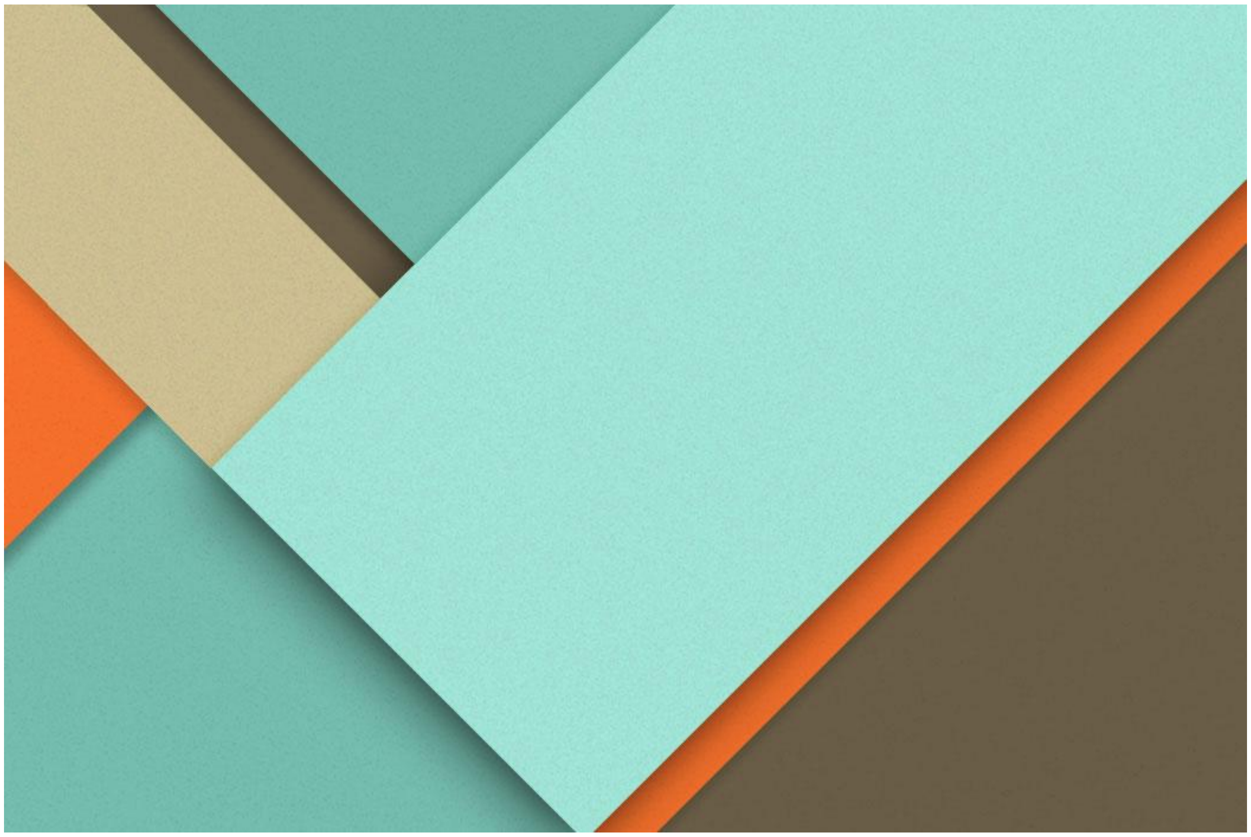


Personalized Financial Advice using NLP

MarkovML

Team Members

Prashant Gupta

A.V.V Meghashyam

Mentors

Ashita Boyina

Piyush Bhopalka

Abstract

This project introduces a novel approach to address the user confusion and information overload experienced on bank websites by proposing a Personalized Financial Advice Chatbot (PFAC). Traditional bank websites often suffer from complex navigation structures with numerous submenus, leading to user frustration in locating relevant information. PFAC aims to mitigate this issue by employing advanced natural language processing and machine learning techniques to provide tailored financial advice and answers to user queries. By understanding user intents and preferences, PFAC streamlines the information-seeking process, enhancing user experience and facilitating better financial decision-making.

Introduction

In the digital age, where convenience and efficiency are paramount, online platforms have become the primary mode of interaction between customers and businesses. This trend holds true for the banking sector as well, where customers increasingly rely on bank websites for various financial transactions and information retrieval. However, the complexity of these websites' navigation systems often leads to user confusion and frustration, inhibiting the seamless access to relevant information. This project introduces a transformative solution to this issue by proposing the development of a Personalized Financial Advice Chatbot (PFAC).

Modern bank websites are replete with a multitude of submenus and intricate navigation paths, intended to cater to a diverse range of financial inquiries and services. While the intention behind these elaborate designs is to offer comprehensive assistance, they often have the opposite effect. Users, overwhelmed by the sheer number of choices and unsure about which submenu could address their specific queries, often find themselves trapped in a lot of links, resulting in a suboptimal user experience.

To sum up, the Personalized Financial Advice Chatbot is a smart solution to the problem of confusing bank websites. By using simple language and advanced technology, it helps users get the financial information they need without the stress of navigating complex menus. This project will explore the technical details, how it's made, and the benefits it brings, showing how this chatbot could change the way we get financial advice online.

Data Collection

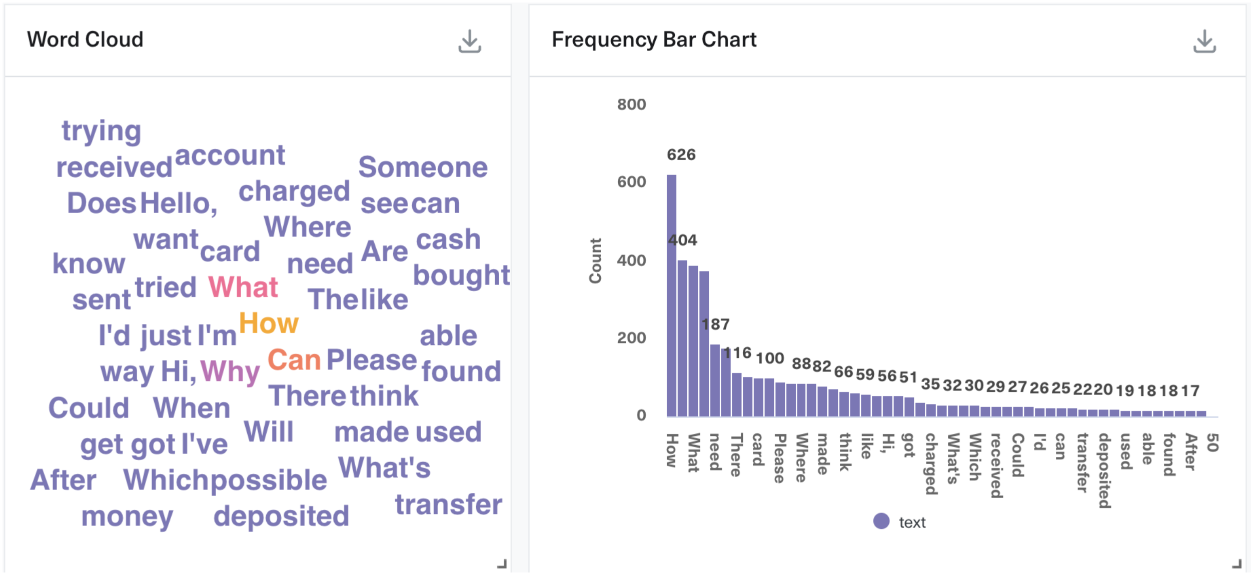
All of the datasets were collected from the HuggingFace dataset collection. The keyword for searching the datasets was Banking. All the datasets were in .parquet format. They were then converted to .csv format using the pandas library in Python for uploading to MarkovML since .parquet format is not supported on the platform.

Data Cleaning

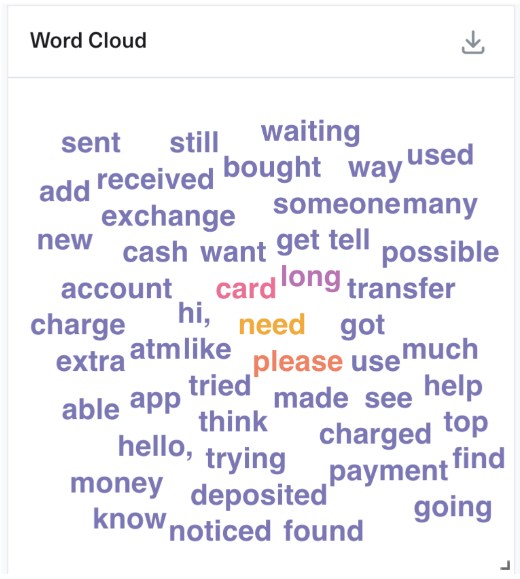
All the datasets were cleaned to remove the stop words in the dataset which were in very high frequency. We used the standard NLTK stop words collection to perform the data cleaning. We removed some words from the collection such as : "not", "wasn't", and many more. This was done so that the meaning of the statement does not change i.e., a positive statement does not become a negative one and vice versa. We converted all the words to lowercase.

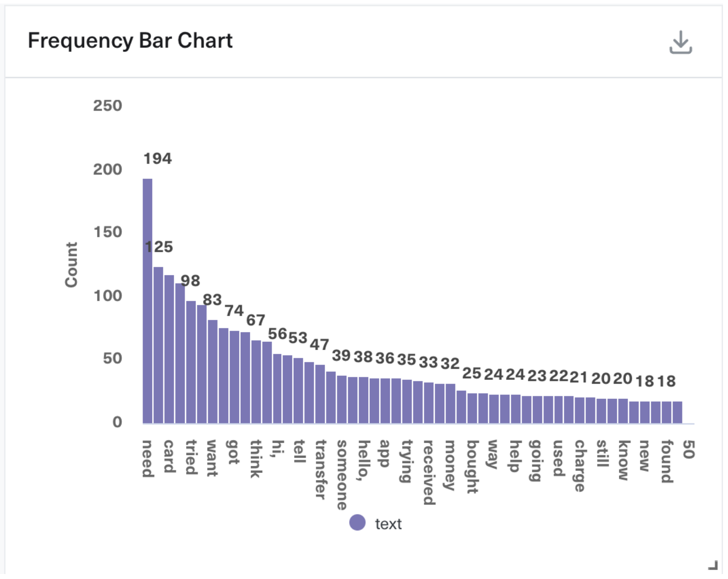
The Word Cloud before and after cleaning looks like this:

Before Cleaning : This dataset has a lot of stop words - "How", "What", "can" , "why"



After Cleaning : The dataset has much more relevant words.

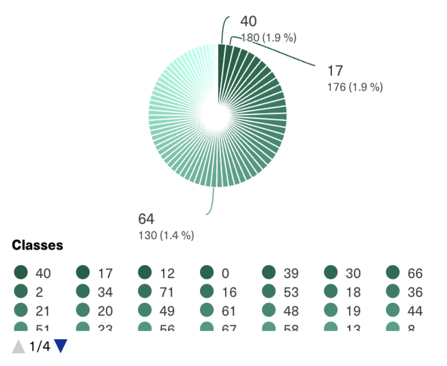


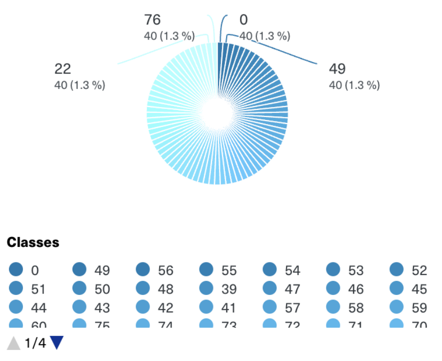


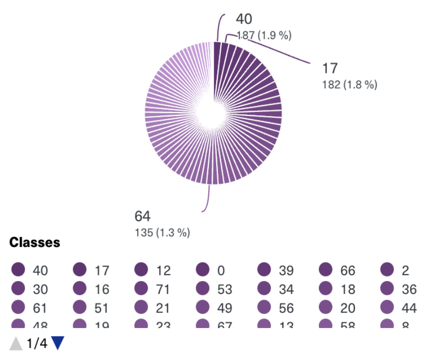
Key insights of the Dataset Using MarkovML platform:

1. The Dataset consisted a total of 22,568 rows where the train dataset consisted of 9503 rows, test of 10,003 rows and validation of 3,080 rows.

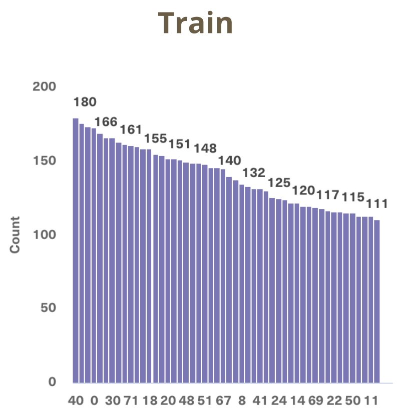
Train(42.07%): **Validation(13.64%) :** **Test(44.29%):**

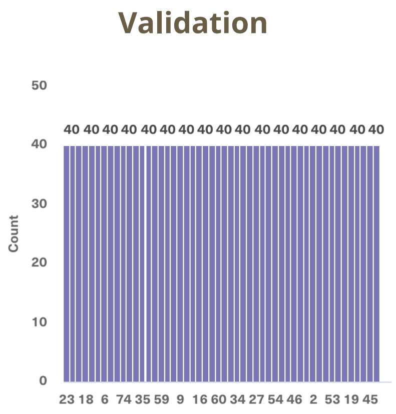


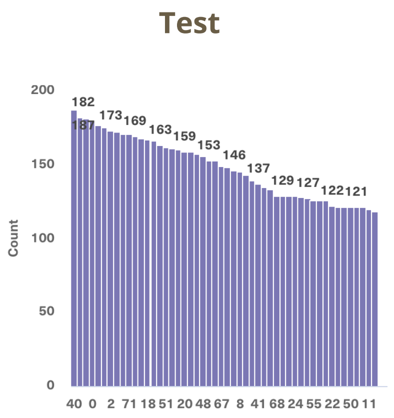




2. Good Distribution within the labels of the Data. All the three parts (train, val and test) of the dataset consisted of 77 labels and they were almost equally distributed among them. This ensures that the model is not biased towards some set of words/sentences. When the classes are imbalanced (i.e., some classes have significantly more samples than others), a model may inadvertently learn to predict the majority class well while neglecting the minority classes. Balancing the dataset helps to avoid this issue and provides a more accurate representation of the model's performance across all classes.







3. No missing data in the dataset. Machine learning models tend to perform better on complete datasets. Missing data can lead to suboptimal model performance due to imputation errors or the loss of valuable information. Models trained on complete data are more likely to generalize well to new, unseen data.

4. High Cardinality and no duplicate columns in train data (Uncleaned dataset). High cardinality in a dataset refers to having a large number of distinct values or categories in a categorical variable. High cardinality often implies that the categorical variable contains a diverse range of information. Each distinct value represents a specific category or subcategory. Categorical variables with high cardinality can be highly discriminative, meaning they can provide significant insights in the models. Here we need to consider the Uncleaned dataset since after data cleaning many words might get removed and so it can have duplicate rows. Hence, for this section we have considered the Unclean version of the dataset.

Types of Models used

Brief Description of Model Architectures Used for Training

1. **Simple Neural Network**

   The basic neural network architecture for text processing includes an embedding layer for numerical conversion, followed by flattening and fully connected layers to capture data patterns. The final layer employs softmax for multi-class classification, optimizing via parameter adjustments and a defined loss function during training.

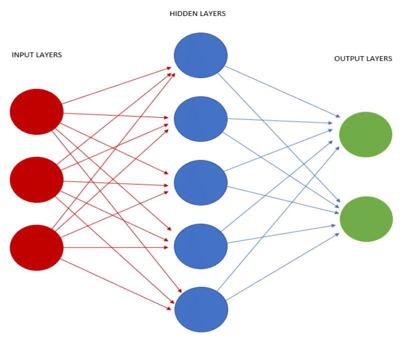


Fig : Simple Neural Network Architecture

2. **Bi-directional LSTM**

   Bidirectional LSTM modifies the traditional approach by processing sequences in both forward and backward directions, enhancing contextual understanding from preceding and subsequent words.

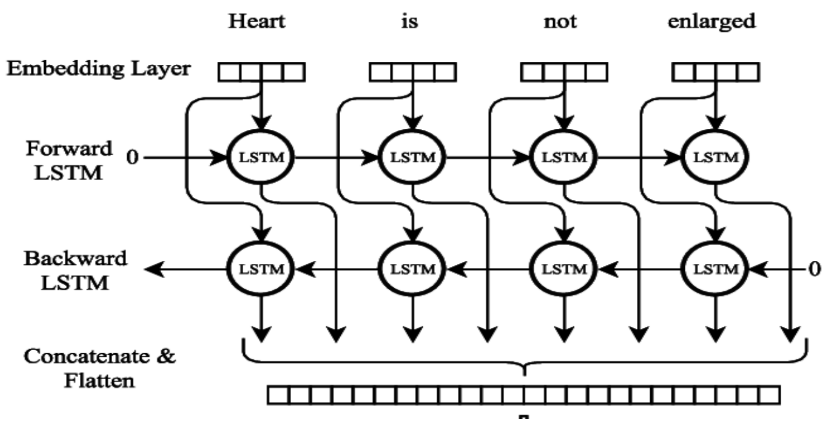


Fig : Bi-Directional LSTM Architecture

3. RNN (LSTM)

RNN with LSTM layers excels in capturing patterns and dependencies. These Long Short-Term Memory layers excel in retaining relevant data over extended sequences, ideal for tasks emphasizing temporal understanding in input data.

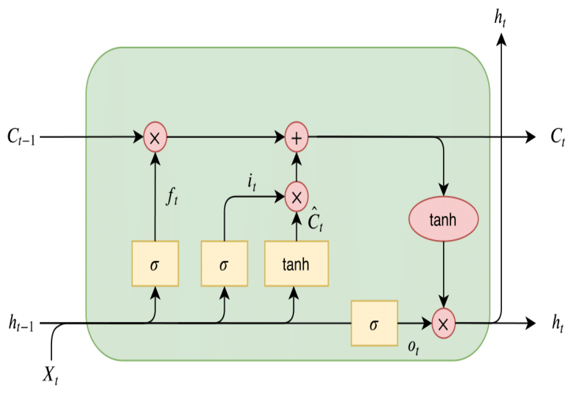


Fig : LSTM Architecture

4. Gated Recurrent Unit (GRU)

Gated Recurrent Units (GRUs) provide an alternative to LSTM layers in recurrent networks, simplifying architecture for computational efficiency while retaining the ability to capture sequential patterns.

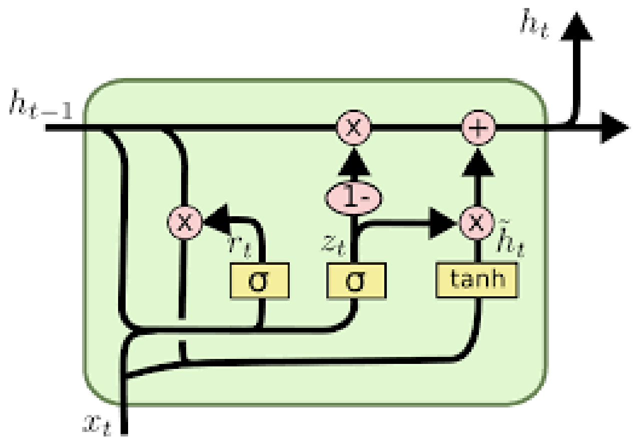


Fig : GRU Architecture

5. **1D Convolution Neural Network**

   Unlike traditional image-focused CNNs, 1D CNNs process data along a single dimension. This is beneficial when dealing with sequences, as they can capture local patterns within the text to understand the context better.

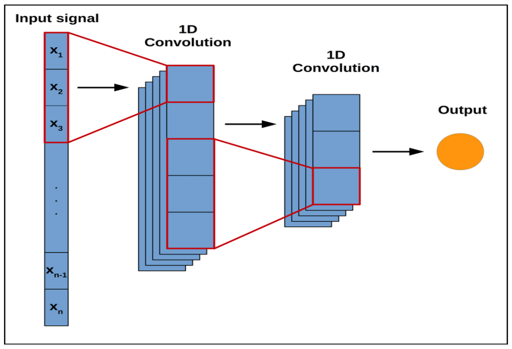


Fig : 1D CNN Architecture

Why RNN (LSTM & GRU) perform poorly compared to Simple NN and 1D CNN (Refer Training graph)

RNNs, especially more complex variants like LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Unit), tend to have a larger number of parameters compared to simple feedforward neural networks. RNNs might suffer from overfitting, whereas a simpler neural network might generalize better with fewer parameters. Additionally, RNNs are prone to the vanishing gradient problem, which can make it difficult for the network to capture long-range dependencies in sequences. The simple neural network might outperform the RNN due to better gradient flow and learning of dependencies.

The Reason for 1D CNN to perform well is that it is capturing the local patterns instead of long range patterns which an RNN does and since from the training data the size of questions is small so the idea of Long Range dependencies could be a reason why RNN performed poorly and it was relevant to capture local patterns for this dataset and so 1D CNN performed better.

Why did Bi-Directional LSTM perform Well??

Bidirectional LSTM addresses vanishing gradient problems in RNNs by using two LSTM networks, one processing the sequence forward and the other backward. This allows gradients to flow from both directions, which can improve training stability.

Reason for not using Other ML Models

1. Linear SVM and Clustering Based Approach

   The below image shows the Embedding of various data points and we can clearly observe that the data cannot be clustered into well defined boundaries and also the data does not look to have a linear boundary. Hence, these approaches were not used.

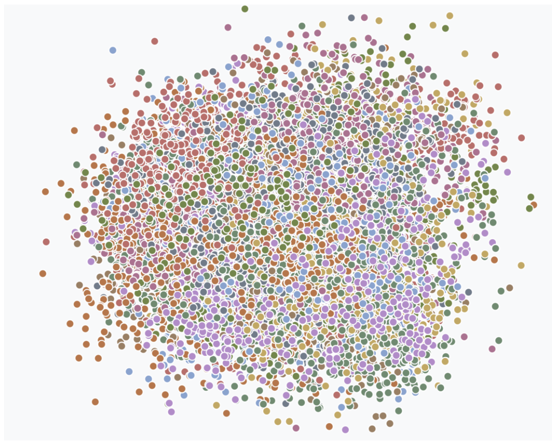


Fig. Embeddings

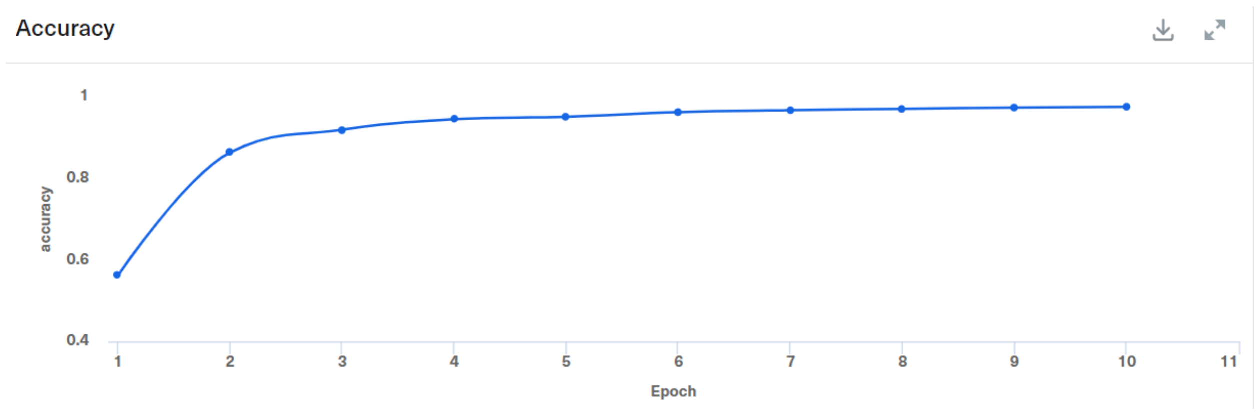
2. Decision Trees

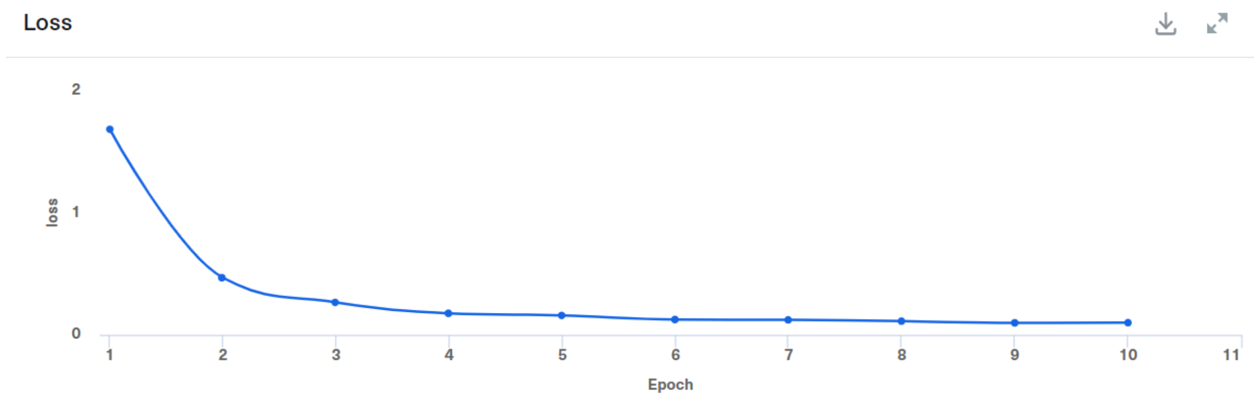
   Neural networks are particularly well-suited for tasks that involve complex non-linear relationships in the data. Decision trees are limited in their ability to capture such complex patterns, especially when the decision boundaries are intricate which can be observed from the above image. Neural networks can approximate highly nonlinear functions, making them better for tasks like natural language processing.

Model Training

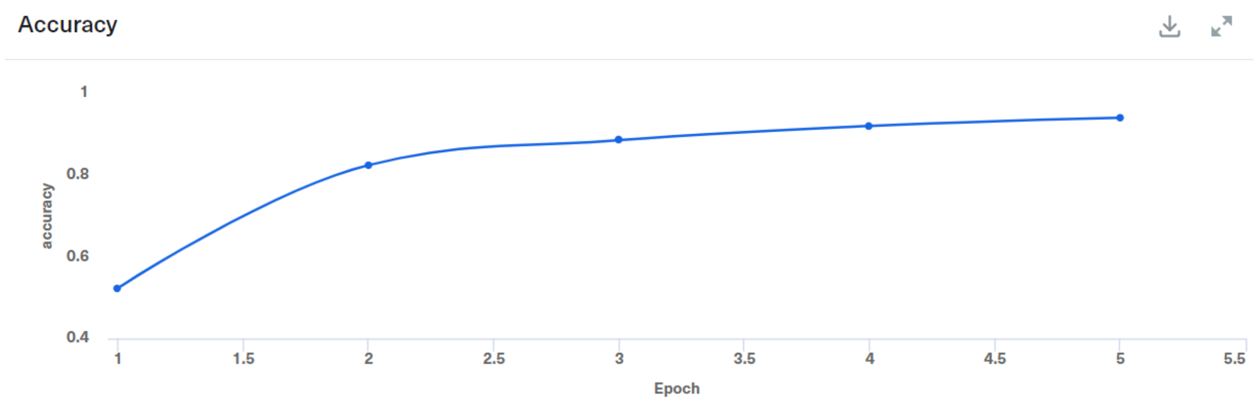
Our Experiments for each of the Models is given below.

1. Simple Neural Network

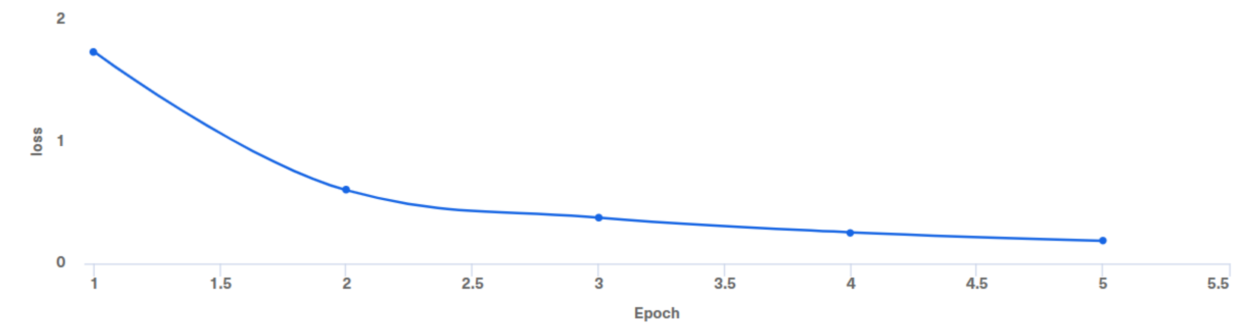




2. Bi-directional LSTM

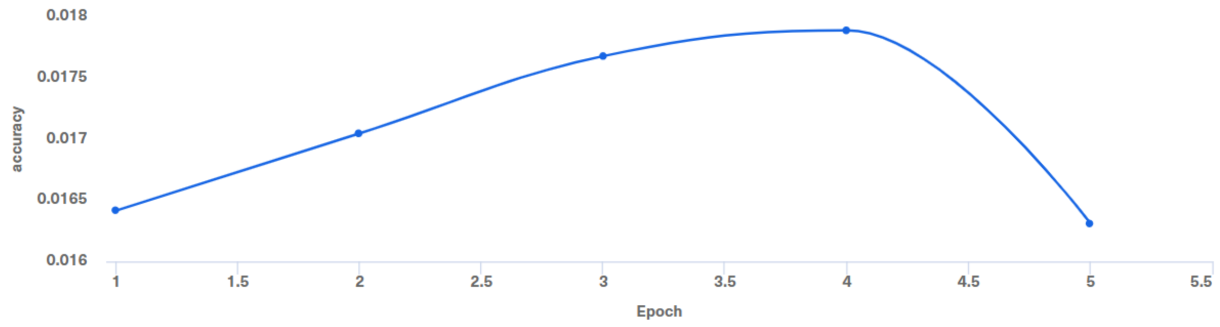


Loss

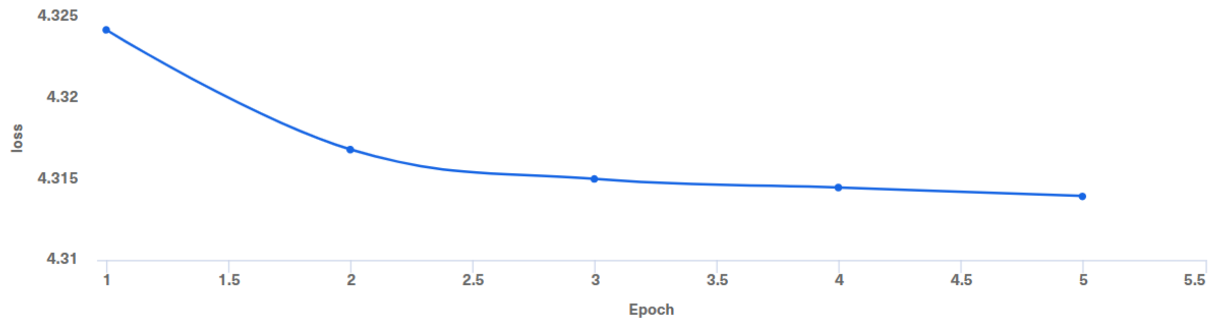


3. RNN (LSTM)

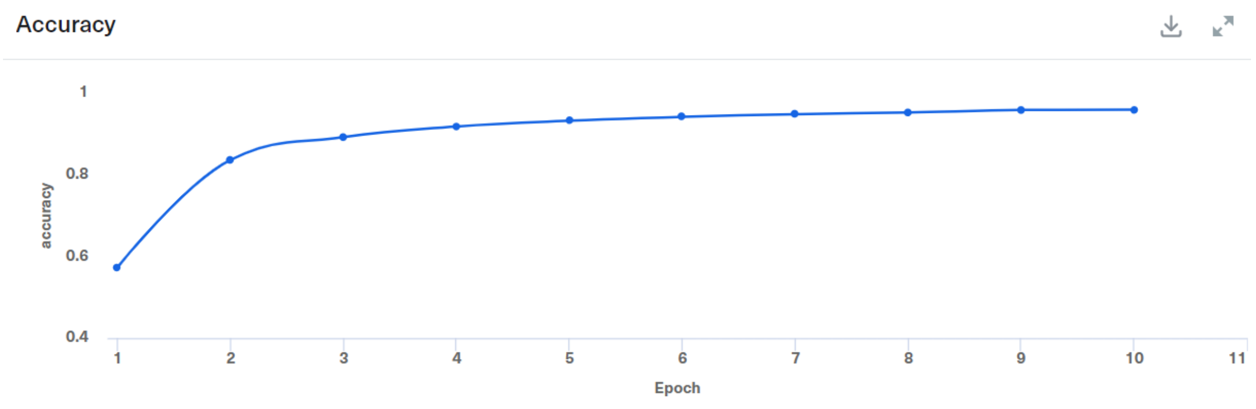
Accuracy

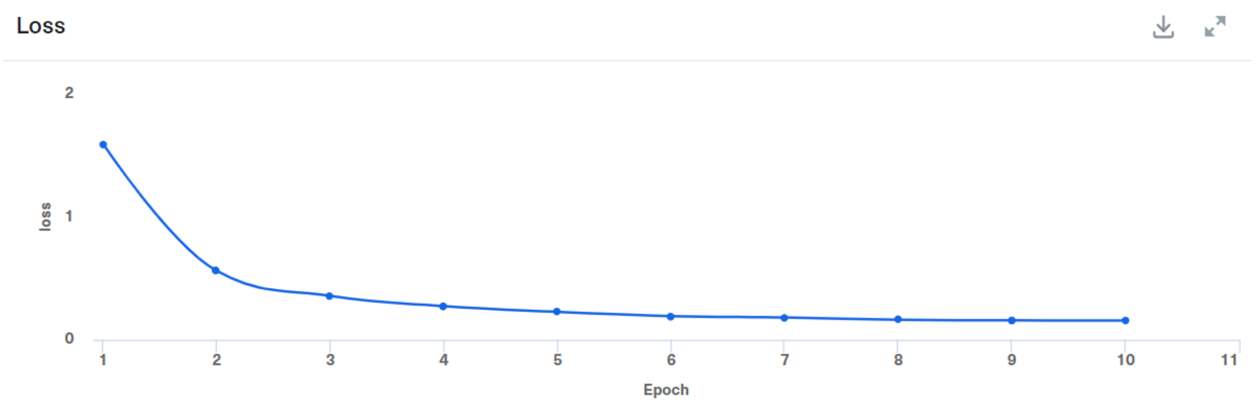


Loss

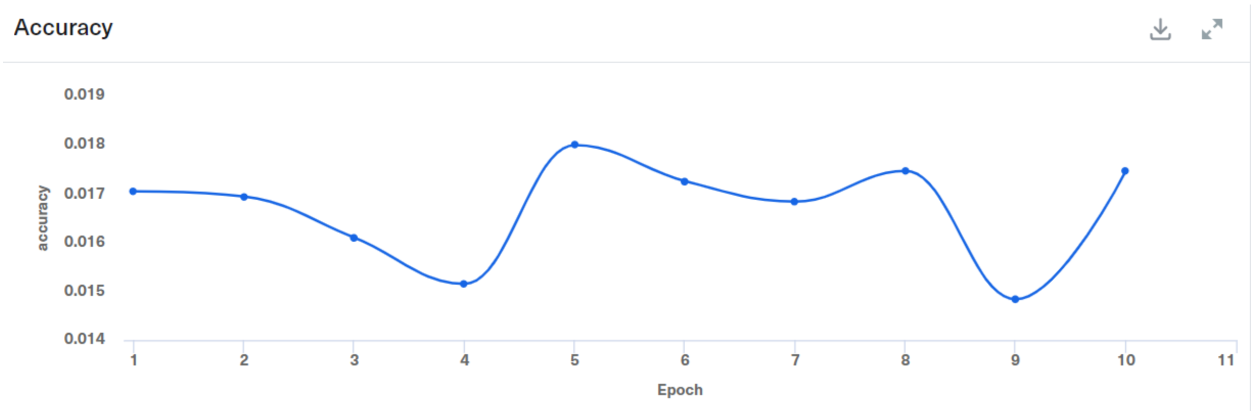


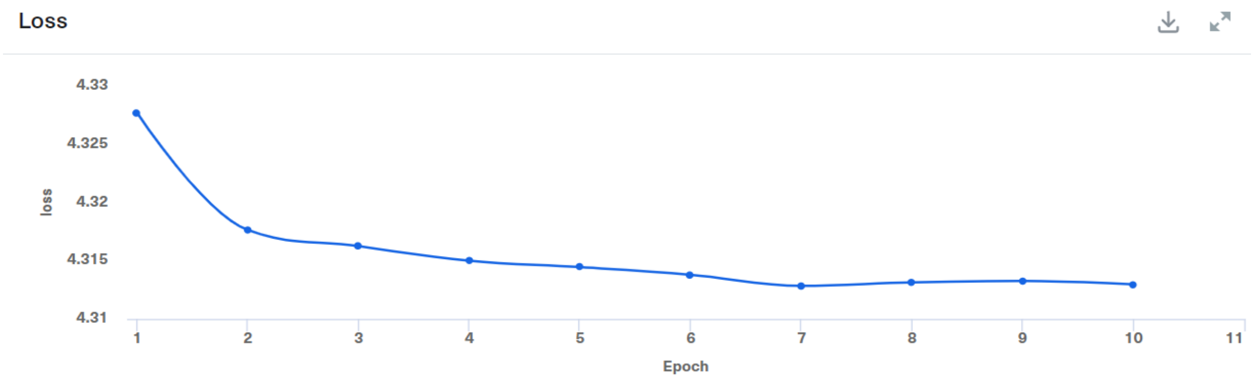
4. **1D Convolution Neural Network**





5. **Gated Recurrent Unit (GRU)**





Activation Function, Loss Function, Optimizer parameter & Metrics

- **Activation Function:** We have chosen ReLU because it solves the problem of Vanishing gradient by allowing gradients to pass unchanged as long as the input is positive and also because it is computationally less expensive compared to sigmoid. We have used Softmax in the final layer since the Softmax function normalizes the logits by exponentiating them and then dividing by their sum. This normalization ensures that the output probabilities are between 0 and 1 and sum up to 1. Hence we just need to pick up the label with highest probability and it also gives a better intuitive understanding.

- **loss='sparse_categorical_crossentropy':** The loss parameter specifies the function that measures how well the model's predictions match the actual labels during training. In this case, 'sparse_categorical_crossentropy' is the chosen loss function. It's used for multi-class classification problems where the labels are integers, not one-hot encoded as in case of categorical_crossentropy.

- **optimizer='adam':** We have chosen Adam's optimizer because it adapts the learning rate for each parameter individually based on the historical gradients. This adaptive learning rate adjustment can lead to faster convergence and better performance, as it allows the algorithm to respond differently to the gradient behavior of each parameter.

- **metrics=['accuracy']:** The metrics parameter specifies the evaluation metrics that the model will track during training and testing. 'accuracy' is a common metric used for classification tasks. It measures the ratio of correctly predicted instances to the total number of instances.

Evaluation and Model Selection

To evaluate model performance, precision (P), recall (R), average precision (AP), and mean average precision (mAP) are common metrics. Accuracy measures correctly identified positive instances among all detected positives. Recall rate gauges the percentage of actual positives correctly identified by the binary classifier, indicating the efficacy of recalling genuine positive instances.

$$Precision = \frac{TP}{TP + FP} \times 100\%$$

$$Recall = \frac{TP}{TP + FN} \times 100\%$$

In this context, TP represents correct target detections, FP indicates incorrect identifications, and FN accounts for missed samples. A higher average precision suggests a better classifier, calculated as the area under the precision-recall curve. This curve shows how precision and recall vary, with epochs marking complete training instances across all datasets. The mean average precision (mAP) is the average AP across categories. Calculations for both mAP and AP are outlined below.

$$AP = \int_0^1 P(R)dR$$

$$mAP = \frac{\sum PA}{NC}$$

Accuracy (mAP) stands as a crucial metric in target detection algorithms, confined to the [0, 1] range. A greater value signifies improved performance. mAP curves are generated using thresholds above 0.5, spanning a spectrum of IoU thresholds (0.5 to 0.95, incremented by 0.05).

Models	Precision	Recall	F1-Score	Accuracy (mAP)	Model Class
Simple Neural network	0.979	0.979	0.979	0.979	Classification
Bi-directional LSTM	0.961	0.959	0.959	0.959	Classification
RNN (LSTM)	0.000	0.019	0.001	0.019	Classification
1D CNN	0.969	0.968	0.968	0.968	Classification
RNN (GRU)	0.011	0.018	0.001	0.018	Classification

Table 1: Evaluation Metrics

Since the training accuracy is quite low for both the variants of RNN we observe that they have a similar result in the evaluation i.e., having poor metrics. We observe quite good evaluation from the rest of the three models which implies they are not overfitting.

Based on these metrics and due to the simplicity of Simple NN , we are selecting the "Simple Neural Network". Another reason for selecting a Simple Neural Network is that they might be less prone to Overfitting compared to the other two better performing models since they can generalize better.

Chatbot Implementation

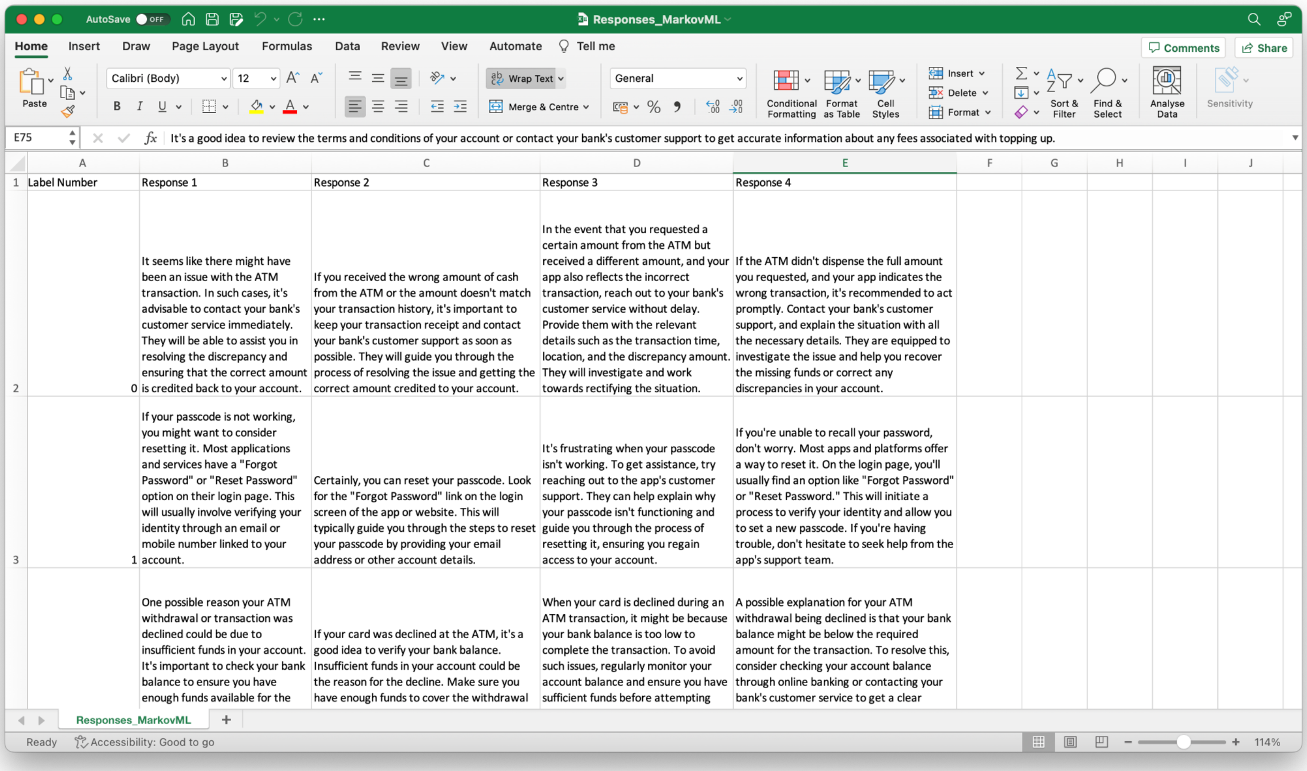
For the Implementation 2 very important things have to be done.

1. Generation of the Responses for each label.
2. Deployment of the model on a Website to make it user friendly.

We will discuss both the things in detail below.

Generation of the Responses for each label

In order to enhance user experience, a pool of four unique responses was generated for each of the 77 labels. This approach was implemented to prevent users from receiving repetitive replies to their queries. By introducing this variety, the system ensured that users would encounter diverse responses even when asking the same question. The selection of a response associated with a predicted label was randomized. This strategy aimed to provide a richer and more engaging conversational context for users while maintaining an element of unpredictability in the responses they received. The entire generation of around 300 responses was done manually. An example of the same can be seen below.



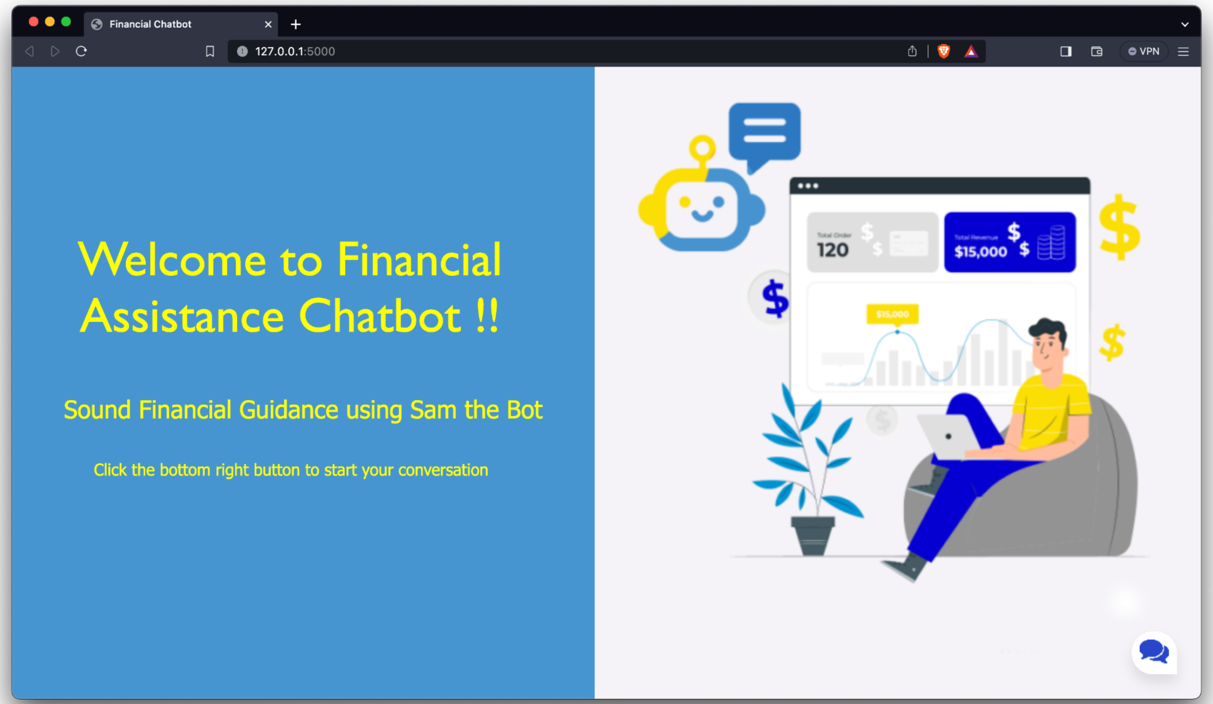
E75 — It's a good idea to review the terms and conditions of your account or contact your bank's customer support to get accurate information about any fees associated with topping up.

Label Number	Response 1	Response 2	Response 3	Response 4
0	It seems like there might have been an issue with the ATM transaction. In such cases, it's advisable to contact your bank's customer service immediately. They will be able to assist you in resolving the discrepancy and ensuring that the correct amount is credited back to your account.	If you received the wrong amount of cash from the ATM or the amount doesn't match your transaction history, it's important to keep your transaction receipt and contact your bank's customer support as soon as possible. They will guide you through the process of resolving the issue and getting the correct amount credited to your account.	In the event that you requested a certain amount from the ATM but received a different amount, and your app also reflects the incorrect transaction, reach out to your bank's customer service without delay. Provide them with the relevant details such as the transaction time, location, and the discrepancy amount. They will investigate and work towards rectifying the situation.	If the ATM didn't dispense the full amount you requested, and your app indicates the wrong transaction, it's recommended to act promptly. Contact your bank's customer support, and explain the situation with all the necessary details. They are equipped to investigate the issue and help you recover the missing funds or correct any discrepancies in your account.
1	If your passcode is not working, you might want to consider resetting it. Most applications and services have a "Forgot Password" or "Reset Password" option on their login page. This will usually involve verifying your identity through an email or mobile number linked to your account.	Certainly, you can reset your passcode. Look for the "Forgot Password" link on the login screen of the app or website. This will typically guide you through the steps to reset your passcode by providing your email address or other account details.	It's frustrating when your passcode isn't working. To get assistance, try reaching out to the app's customer support. They can help explain why your passcode isn't functioning and guide you through the process of resetting it, ensuring you regain access to your account.	If you're unable to recall your password, don't worry. Most apps and platforms offer a way to reset it. On the login page, you'll usually find an option like "Forgot Password" or "Reset Password." This will initiate a process to verify your identity and allow you to set a new passcode. If you're having trouble, don't hesitate to seek help from the app's support team.
	One possible reason your ATM withdrawal or transaction was declined could be due to insufficient funds in your account. It's important to check your bank balance to ensure you have enough funds available for the	If your card was declined at the ATM, it's a good idea to verify your bank balance. Insufficient funds in your account could be the reason for the decline. Make sure you have enough funds to cover the withdrawal	When your card is declined during an ATM transaction, it might be because your bank balance is too low to complete the transaction. To avoid such issues, regularly monitor your account balance and ensure you have sufficient funds before attempting	A possible explanation for your ATM withdrawal being declined is that your bank balance might be below the required amount for the transaction. To resolve this, consider checking your account balance through online banking or contacting your bank's customer service to get a clear

Deployment of the model on a Website to make it user friendly.

The development of our website involved a combination of technologies. Leveraging the Flask framework, we constructed an interactive website, incorporating HTML, CSS, and JavaScript to ensure an engaging user interface. At the core of the system, a Python script operates in the backend, utilizing predictive capabilities to generate appropriate responses for user queries. This integration allows for efficient user interaction. Additionally, we've implemented both a terminal-based interface and a user-friendly website.

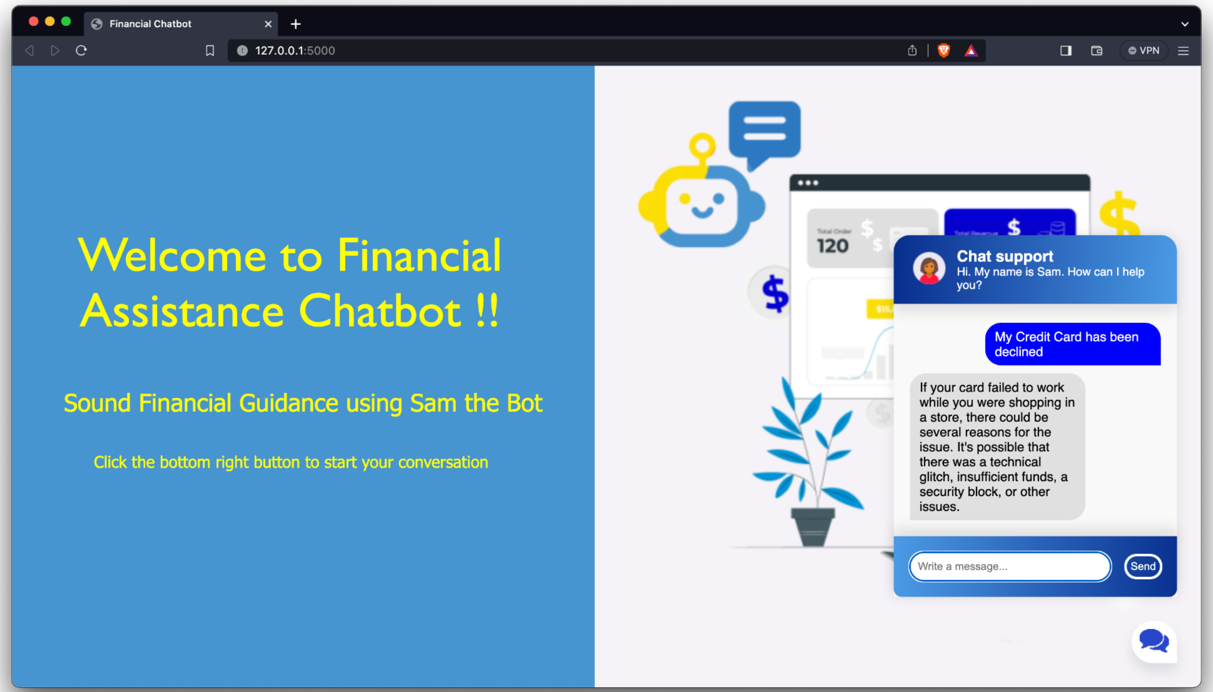
The website before and after clicking the chat button for a query are shown below.



Welcome to Financial Assistance Chatbot !!

Sound Financial Guidance using Sam the Bot

Click the bottom right button to start your conversation



Welcome to Financial Assistance Chatbot !!

Sound Financial Guidance using Sam the Bot

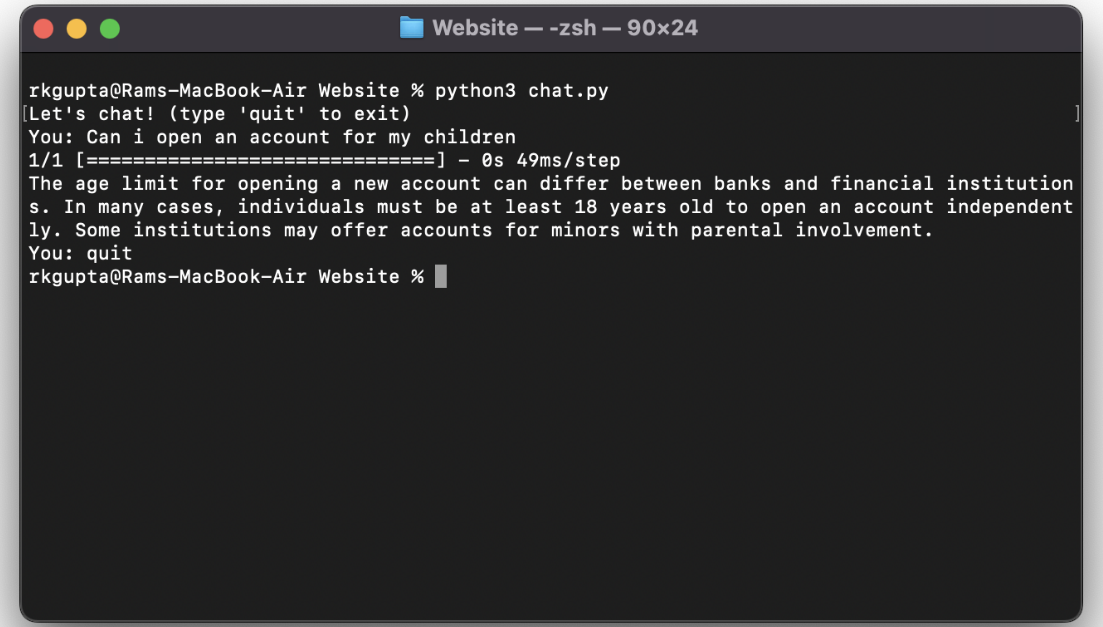
Click the bottom right button to start your conversation

Chat support
Hi. My name is Sam. How can I help you?

My Credit Card has been declined

If your card failed to work while you were shopping in a store, there could be several reasons for the issue. It's possible that there was a technical glitch, insufficient funds, a security block, or other issues.

The terminal chatbot is as shown below.



```
rkgupta@Rams-MacBook-Air Website % python3 chat.py
[Let's chat! (type 'quit' to exit)                                                        ]
You: Can i open an account for my children
1/1 [==============================] - 0s 49ms/step
The age limit for opening a new account can differ between banks and financial institution
s. In many cases, individuals must be at least 18 years old to open an account independent
ly. Some institutions may offer accounts for minors with parental involvement.
You: quit
rkgupta@Rams-MacBook-Air Website % 
```

Results

The successful culmination of this research project highlights the pivotal role of meticulous model selection and training in achieving substantial accuracy improvements. Through rigorous evaluation and experimentation, we identified the simple neural network model as the optimal choice, resulting in an impressive 97% accuracy on our datasets. The model's exceptional performance was further complemented by its straightforward coding, reaffirming its suitability for practical implementation. The acquisition of the weights file further solidified the credibility of our results. As we transition into the implementation phase, we anticipate that the seamless integration of our chosen model into real-world applications will underscore its potential for reshaping industries and inspiring further advancements in the realm of machine learning. This project not only contributes to our understanding of effective model selection and training but also sets the stage for innovative applications in diverse domains.

Future Work

In our project's future plans, we're aiming to connect our system with a banking database and enhance security measures. This will help us provide users with more detailed responses, making their experience much better. By linking our app with a banking database, users can quickly get info about their finances, transactions, and accounts in real-time.

However, dealing with such sensitive data means we need to focus even more on security. We'll use strong encryption, extra verification steps, and things like fingerprints or face recognition to keep users' info safe. This will build trust, and more people will likely use our system.

With these changes, we want to make the user experience even better. By securely accessing and analyzing banking data, we'll offer personalized answers to financial questions. This will help our platform stand out as a friendly and secure virtual assistant in the banking world.

Conclusion

In conclusion, our project is centered around the creation of an interactive website that will feature a chatbot as its central component. This chatbot will be driven by a simple model, harnessing the capabilities of natural language processing. This model will empower the chatbot to engage users in meaningful conversations, effectively understanding their queries and delivering contextually relevant responses. The integration of this model into the website holds the promise of offering users a seamless and interactive experience, catering to diverse inquiries and needs. As we move forward, we anticipate that this innovative approach will contribute to a more engaging and user-centric online platform.

A heartfelt appreciation to Ashita Boyina, Piyush Bhopalka, and the entire MarkovML team for their invaluable contributions throughout our project.

References

- Dataset 1 : https://huggingface.co/datasets/generalization/banking_intent_Full-p_05
- Dataset 2 : https://huggingface.co/datasets/argilla/sentiment-banking
- Dataset 3 : https://huggingface.co/datasets/dvilasuero/banking_with_vectors
- MarkovML Developer docs : https://developer.markovml.com/
- Tensorflow Documentation : https://www.tensorflow.org/api_docs