Comp1804 Report: **Predicting the topic of customers' banking questions**

Student ID:001278084

Submission Date:12/04/2023

Word count: 3250

Executive summary

In this report, we will discuss on the machine learning techniques we used to predict the topic of customers' banking questions in an online chat. The purpose of this report is to provide insights into the feasibility and effectiveness of automated topic classification to improve the customer service online for a bank or a commercial entity. Specifically, we aim to predict the topic of a customer's question in an online chat using machine learning algorithms. The bank has provided us with a dataset containing sample questions and their associated topics. The dataset has four categories: "card queries or issues", "needs troubleshooting", "top up queries or issues", and "other". And we want to use machine learning to predict the category to which each question belongs to. Also, we need to compare traditional machine learning algorithms with neural networks to determine if the latter offer significantly higher performance.

We started with exploratory data analysis (EDA) to get an understanding of the dataset and to detect and correct data errors like missing values and data imbalance. Also performed various techniques for data preprocessing such as data cleaning to eliminate duplicate, missing or irrelevant data, feature selection, splitting of the data, oversampling to handle data imbalance, feature encoding and text processing. Traditional machine learning modeling algorithms such as MultinomialNB, SGDClassifier and Logistic Regression are performed, along with an Artificial Neural Network developed using Keras.

The model's performance were calculated with the parameters like accuracy, precision, recall, and F1-score. The findings showed that Artificial neural network using Keras has the highest accuracy rate of 87% on the test data and the traditional modeling algorithms such as SGD Classifier and Logistic Regression has the second highest accuracy rate of 86% on the test data, followed by the

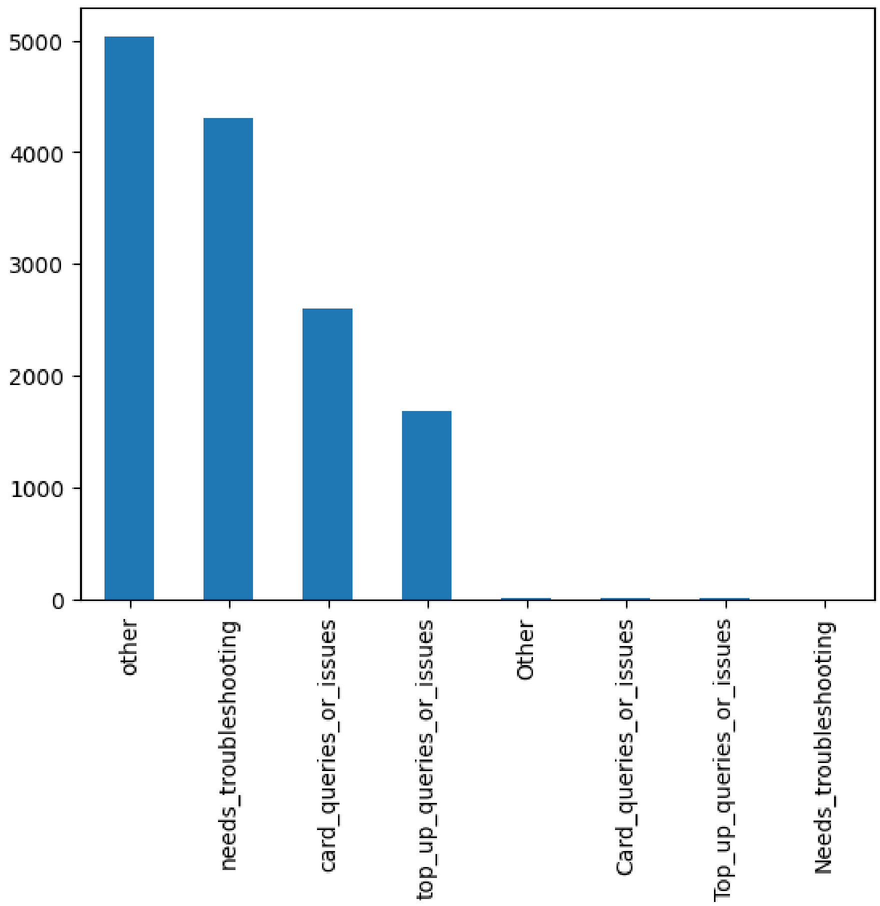
Multinomial NB at 85%. From this we can conclude that Artificial neural network is the best machine learning model.

1. Exploratory data analysis

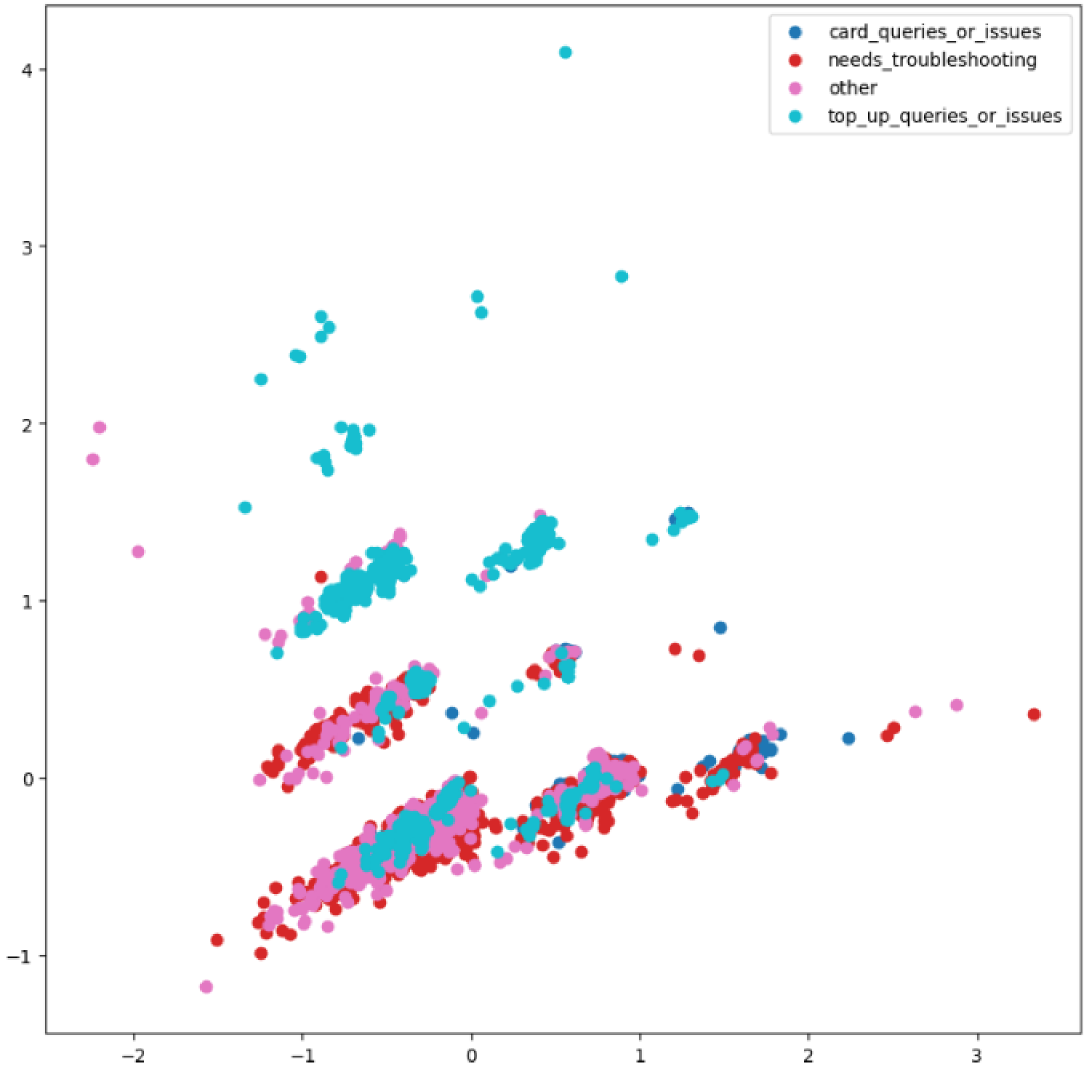
Exploratory Data Analysis(EDA)is a process of examining and analyzing datasets to summarize their main characteristics, identify patterns, and detect anomalies or outliers. EDA aims to reveal the underlying structure, patterns, and features of the data through visual and statistical methods.

To perform EDA, we began by loading the dataset and examining its fundamental characteristics using the info() method to obtain information on the number of rows, columns, and data types of each column. Next, we checked for missing values using the "isna()" method and computed the sum of missing values in each column using the "sum()" method. We also explored the distribution of the target labels using the "value_counts()" method, which enabled us to determine if there is any class imbalance. To visualize the high-dimensional data in a 2D plot, we applied two dimensionality reduction techniques: PCA and t-SNE.

The dataset provided consisted of 14195 rows and three columns: "text", "label", and "query_index". The text column contained the text data, which served as the primary input for the machine learning task. The label column contained the target labels, while the query_index column contained the unique identifier for each query. The dataset had 521 missing values in the label feature, and the target labels were slightly imbalanced, with the "top_up_queries_or_issues" category having significantly fewer instances than the others. The figure below shows the imbalanced dataset.



The dimensionality reduction techniques, PCA and t-SNE, allowed us to visualize the distribution of data points in a 2D space, with each point colored according to its corresponding target label. From the visualizations, we were able to observe that some of the target labels were closer to each other, indicating that they might be more challenging to classify. The figure below shows PCS technique performed as part of dimensionality reduction technique



EDA provided us with a better understanding of the dataset, including its distribution, pre-processing requirements, and dimensionality reduction techniques for visualizing the data. This knowledge is beneficial in guiding the selection of appropriate machine learning algorithms and fine-tuning hyperparameters for better model performance.

2. Data preprocessing

Data pre-processing refers to the preparation and transformation of raw data into a format that is suitable for analysis or modeling. Data pre-processing is an essential step in data-driven projects, as it helps to clean, normalize, and transform the data into a more meaningful representation.

Data Cleaning

Data cleaning is the process of identifying and correcting or removing inaccurate, incomplete, or irrelevant data in a dataset. Data cleaning is an essential step in data pre-processing, as it helps to ensure the accuracy and completeness of the data, which can affect the quality of subsequent analysis or modelling. Steps involved are:

- Removing irrelevant and duplicate data: This involves identifying and removing rows that contain identical data in all columns. Duplicate data can arise due to data entry errors or system issues and can lead to over-representation of certain data points.
- Handling missing data: This involves dealing with rows or columns that contain missing data. Techniques for handling missing data include imputation, where the missing values are replaced with estimated values based on the available data, and deletion, where rows or columns containing missing data are removed from the dataset.
- Outlier detection and removal: This involves identifying and removing data points that are significantly different from the rest of the data. Outliers can arise due to data entry errors, measurement errors, or other factors and can distort the analysis or modeling results.
- Feature selection: Features that were not relevant to the modeling task were removed. This helps to reduce the dimensionality of the data and improve the accuracy of the model.
- Data standardization: This involves transforming data into a standard format, such as converting categorical data into numerical data or normalizing numerical data to a standard scale.
- Handling inconsistent data: This involves identifying and resolving inconsistencies in the data, such as different spellings or formats of the same data.

In this case, we have performed data cleaning to eliminate any irrelevant or duplicate data from the dataset. We excluded the 'query_index' column as it held no value for the machine learning model. We further ensured that there were no duplicate rows in the dataset. We also inspected the data for missing values and deleted any rows without label information. Moreover, we carried out text cleaning by removing stop words, punctuation marks, and converting all text to lowercase for uniformity in text representation.

Data Splitting

Data splitting is the process of dividing a dataset into two or more subsets, typically for the purposes of training and evaluating a machine learning model.

Here, the bank dataset is divided into two subsets - a training set and a test set - using a split ratio of 80:20, which was performed using the "train_test_split" function in Scikit Learn. This ensured that there was enough data available for both training and testing. The training set was utilized to train the machine learning models, carry out hyperparameter tuning and model selection, while the test set was used to evaluate the final performance of the selected model .Also made sure that the distribution of the classes was consistent across both sets to avoid bias in the results.

Pre-processing

The steps performed include normalization, feature encoding, and text processing.

- Normalization: This involves transforming the data to have a standardized scale or distribution, such as min-max normalization or z-score normalization. Normalization helps to ensure that different features have equal importance in the analysis or modeling process.
- Feature Encoding: It involves categorical features were one-hot encoded to convert them into numerical features that can be used by the machine learning algorithms. In this case, we utilized the LabelEncoder technique to convert categorical features, such as the label, into a numerical format that is compatible with machine learning models. By doing so, we were able to represent these features in a way that can be effectively utilized by the machine learning algorithms.
- Text Processing: Text data was preprocessed by removing stop words, stemming or lemmatizing the words, and converting the text into numerical features using techniques such as bag-of-words or TF-IDF. This helps to ensure that the text data can be used by the machine learning algorithms. In this context, we conducted text processing in order to translate textual information into a numerical format that can be interpreted by machine learning models. To achieve this, we employed a method called count vectorization, which generates a numerical feature matrix from text by assigning weights to each word based on its frequency and

significance within the corpus. This process enables us to express each question as a numerical feature vector.

- Class Imbalance: As previously noted during the exploratory data analysis, there was an issue with imbalanced class distribution. To rectify this, we utilized the Synthetic Minority Oversampling Technique (SMOTE) to oversample the minority classes. This technique involved generating synthetic examples of the minority classes to balance the class distribution and address the issue of imbalanced classes.

3. Classification using traditional machine learning

To solve the classification task, the Stochastic Gradient Descent (SGD) classifier with the GridSearchCV algorithm was utilized for hyperparameter tuning. The GridSearchCV algorithm searched a predefined parameter grid to find the optimal combination of hyperparameters that maximized the classification accuracy. The hyperparameters that were fine-tuned include Penalty, Alpha, and Max_iter.

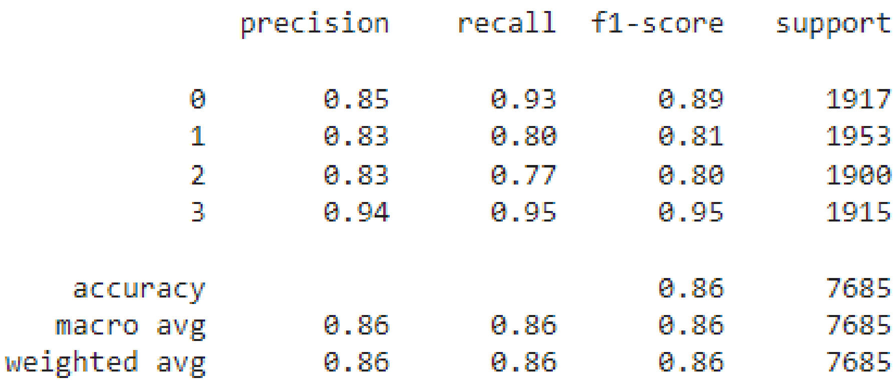
Penalty signifies the norm used in the penalization of the model coefficients and can be set as 'l2' or 'elasticnet'. Alpha denotes the regularization strength of the model, which is the inverse of the regularization parameter, and can have the values of 0.0001 and 0.01. Max_iter is the maximum number of iterations required to converge for the model and can be set to 1000 or 5000 or 10000.

To ensure that the model is generalizable to new data, the GridSearchCV algorithm was trained using a 5-fold cross-validation strategy. The model achieved a best score of 0.857104 using {'alpha': 0.0001, 'max_iter': 10000, 'penalty': 'elasticnet'} . The best combination of hyperparameters are shown in the table below.

Hyperparameter	Value
Penalty	elasticnet
Alpha	0.0001
Max_iter	10000

The SGDClassifier algorithm operates by fitting a linear model to the training dataset and adjusting the coefficients using stochastic gradient descent. This algorithm is effective for large datasets because it randomly selects a single sample during each iteration to update the model. Hyperparameter optimization was carried out using the GridSearchCV algorithm, which searched a predefined parameter grid to determine the best combination of hyperparameters that maximized the classification accuracy. The hyperparameters that were considered were penalty, alpha, and max_iter.

The algorithm was trained utilizing a 5-fold cross-validation strategy to ensure that the model could be applied to new data. To assess the model's performance, the precision, recall, and F1-score were determined for each class. The classification report also includes the overall accuracy, weighted average precision, and weighted average recall for the model, as illustrated in the figure.

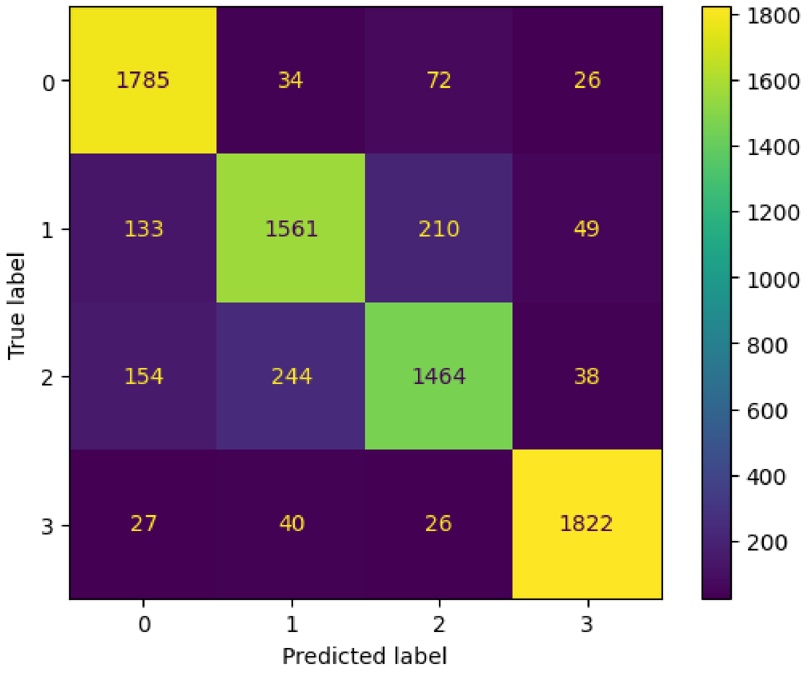


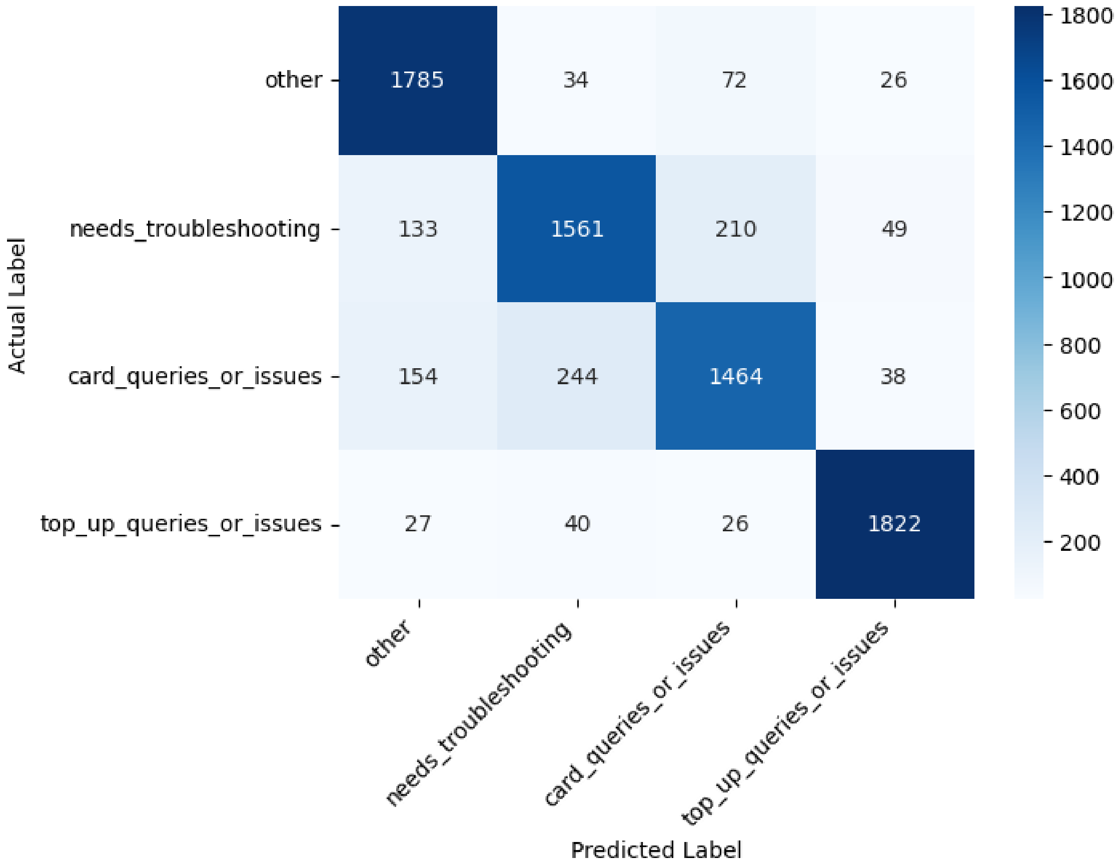
```
              precision    recall  f1-score   support

           0       0.85      0.93      0.89      1917
           1       0.83      0.80      0.81      1953
           2       0.83      0.77      0.80      1900
           3       0.94      0.95      0.95      1915

    accuracy                           0.86      7685
   macro avg       0.86      0.86      0.86      7685
weighted avg       0.86      0.86      0.86      7685
```

To assess the model's performance, two performance metrics, precision, and recall, can be utilized along with a confusion matrix. The confusion matrix is a table that provides a summary of the classification model's performance, indicating the number of true positives, true negatives, false positives, and false negatives for each class. The confusion matrix for the model is presented below:

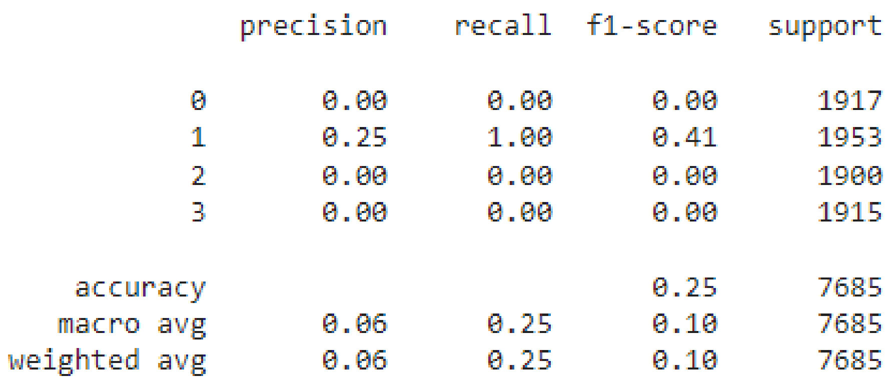




By examining the confusion matrix, we can observe that the model has a high precision and recall score for the "other" and "top_up_queries_or_issues" classes, indicating good performance. However, it has lower precision and recall scores for the "card_queries_or_issues" and "needs_troubleshooting" classes, indicating poorer performance when compared to the former.

Precision and recall are commonly used performance metrics for evaluating classification models. Precision measures the ratio of true positives to all positive predictions, while recall measures the ratio of true positives to all actual positives. In this classification task, precision and recall are appropriate metrics to use because they provide insights into the model's performance for each class.

To compare the performance of the SGD classifier to a "trivial" baseline, we can utilize the majority class classifier. The majority class classifier predicts the most frequently occurring class in the training data for all instances in the test data.



```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00      1917
           1       0.25      1.00      0.41      1953
           2       0.00      0.00      0.00      1900
           3       0.00      0.00      0.00      1915

    accuracy                           0.25      7685
   macro avg       0.06      0.25      0.10      7685
weighted avg       0.06      0.25      0.10      7685
```

The results demonstrate that the majority class classifier achieves a much lower overall accuracy of 0.25 compared to the SGD classifier's accuracy of 0.87. Additionally, the precision, recall, and f1-score for all classes are also very low, indicating that the majority class classifier performs very poorly on this task. These findings lead to the conclusion that the SGD classifier is a meaningful model for this task and performs significantly better than a trivial baseline that always predicts the majority class.

4. Classification using neural networks

Neural networks are commonly used for classification tasks due to their ability to comprehend intricate relationships in data. In this scenario, a neural network can be employed to grasp the link between an input question and its corresponding label. The architecture of the suggested neural network comprises two dense layers with "ReLU" activation functions and a dropout layer in between, followed by a "softmax" activation layer. To compile the model, the sparse categorical cross-entropy loss function and Adam optimizer with a learning rate of 0.001 are utilized.

To optimize the neural network, a grid search approach is employed, where various hyperparameter values are tested to determine the optimal combination resulting in the highest accuracy. The use of GridSearchCV automates the hyperparameter tuning process and lowers the risk of overfitting to the validation set by selecting the best hyperparameters based on cross-validation performance. Additionally, it enables the comparison of different models with varying hyperparameters, assisting in identifying the best model for the given task. The hyperparameters being optimized are determined based on prior knowledge and experimentation, including the batch size, number of epochs, and dropout rate.

The batch size is chosen based on the dataset size and the computational resources available, with smaller batch sizes leading to faster convergence but higher variance in gradients. The number of epochs is chosen based on the desired trade-off between model complexity and training time, with higher numbers leading to overfitting. The dropout rate is chosen based on the desired degree of regularization, with higher rates resulting in stronger regularization but potentially underfitting.

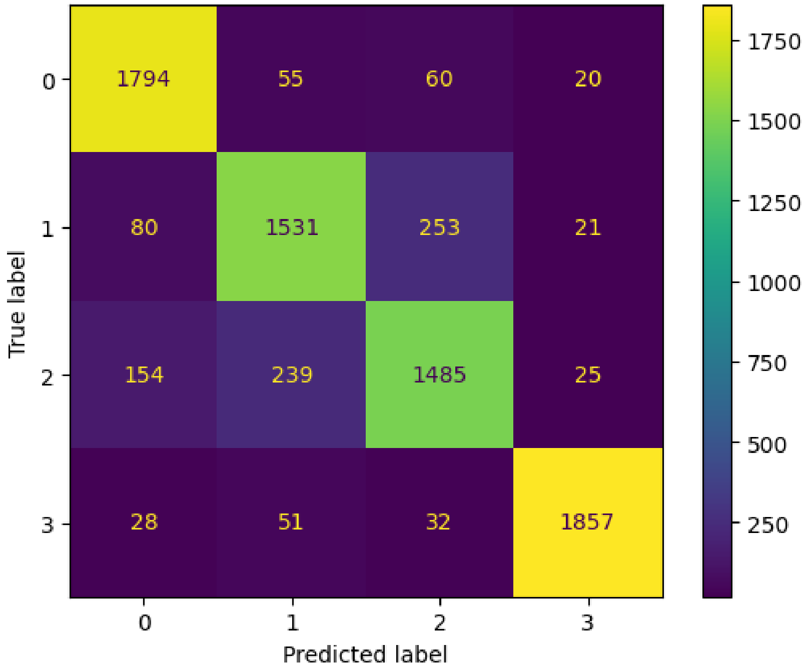
To reduce overfitting and increase generalization performance, the model is evaluated using 3-fold cross-validation on various subsets of the training data. The hyperparameters being considered include batch sizes of 10, 50, and 120, and the number of epochs being considered are 20 and 60, with a dropout rate of 0.2. Various hyperparameters were experimented with to determine the optimal combination, with the results summarized in a table.

Hyper Parameter	Value

Epochs	60
Drop out	0.2
Batch size	10
Learning rate	0.001

First, the KerasClassifier wrapper is used to initialize the GridSearchCV object. Then, the GridSearchCV object is fitted to the training data using 3-fold cross-validation. Once the GridSearchCV object has been fitted to the training data, the best hyperparameters and corresponding accuracy score are printed.

The confusion matrix for the neural network model on the test set is displayed below.



The confusion matrix provides insight into the model's performance, revealing that the majority of questions are correctly classified across all categories. However, there are some misclassifications, particularly between the "needs_troubleshooting" and "other" categories.

To evaluate the model's performance, we use two metrics: accuracy and F1-score. Accuracy represents the proportion of correctly classified samples, while F1-score is the harmonic mean of precision and recall. We select these metrics as they provide a comprehensive evaluation of the model's overall performance.

The accuracy and F1-score for the neural network model and traditional machine learning algorithms are presented in the following table:

Model	Accuracy	F1_score
MultinomialNB	0.85	0.88
SGDClassifier	0.86	0.89
Logistic Regression	0.86	0.90
Artificial Neural Network	0.87	0.90

The table shows that the neural network model surpasses the traditional machine learning algorithms in terms of both accuracy and F1-score. This implies that the neural network can grasp more intricate relationships in the data and offer a better representation of the input features. Additionally, it's evident that the neural network model substantially surpasses the majority class baseline, which suggests that the model can identify significant patterns in the data and generalize effectively to new samples.

5. Ethical discussion

Using the Ethical OS Toolkit, we can identify several social and ethical implications of developing an ML model for predicting the topic of customers’ banking queries in an online chat.

- Bias and Fairness: One of the potential issues is the presence of bias in the data, which can lead to unfair predictions. The dataset provided may not be representative of the entire population, and as a result, the model may be biased towards certain groups. This can result in certain customers being disadvantaged or discriminated against in the chat service.

- Privacy: The dataset provided may contain personally identifiable information (PII) of the customers, such as their names or contact information. The bank must ensure that they are compliant with data privacy regulations and that customer data is handled appropriately.

- Transparency: The ML model's decisions may not be transparent, which can lead to distrust and confusion among customers. The bank must ensure that the ML model's decision-making process is transparent and explainable, so that customers understand how their questions are being classified.

- Accountability: If the ML model makes incorrect or unfair predictions, it is important to establish who is accountable for these decisions. The bank must have processes in place to monitor and audit the model's performance, and to take corrective action when necessary.

- Accessibility: The online chat service must be accessible to all customers, including those with disabilities or who speak languages other than English. The bank must ensure that the ML model and the chat service itself are accessible and inclusive to all customers.

- Informed consent: The bank must obtain informed consent from the customers before using their data for training the ML model. The customers must be informed about how their data will be used and must have the option to opt-out if they so choose.

In conclusion, developing an ML model for topic classification in online customer service has social and ethical implications that must be carefully considered. It is essential to ensure that the model is fair, transparent, and accountable, and that customer privacy and accessibility are protected. The bank must also obtain informed consent from the customers and be aware of any potential biases in the data. By addressing these issues, the bank can develop a model that is ethical, responsible, and beneficial to all customers.

6. Recommendations.

- From our findings, the neural network model seems to be the most suitable approach for the task, achieving the highest accuracy and F1-score. Although the SGDClassifier and logistic regression models also exhibit satisfactory performance, the neural network model outperforms them.

- While the final model could be adequate for practical use, its utility would depend on the specific requirements and limitations of the bank. Although an accuracy of 0.87 is quite impressive, it is essential to consider other factors such as interpretability, computational resources, and training time interpretability when evaluating the feasibility of a model.

- In terms of future enhancements, it may be beneficial to collect more data, especially for the "other" category, which seems to have the lowest accuracy and F1-score. Furthermore, exploring alternative neural network architectures and hyperparameters may lead to even better performance. Finally, employing techniques such as transfer learning or ensemble methods could potentially increase the model's accuracy and resilience.

7. Retrospective

If I had the opportunity to redo the Machine Learning training for this project, I would like to study in depth about how data augmentation techniques impact the model's performance, particularly for the neural network model. This is because using data augmentation techniques can increase the variety and amount of the training data, which may improve the model's ability to generalize and perform better.