Report #2

Classifier for Humanitarian Response

Exploration, Data Visualization, and Data Pre-processing

Authors: Ghiath Al Jebawi, Bercin Ersoz, Allaeldene Ilou, Caspar Stordeur

0. Table of Contents

0. List of Tables

0. List of Figures

1. Final Dataset Structure and Sampling

The dataset focuses on humanitarian aid provided to IDP households. For this analysis 32.912 rows were available. The key variables under investigation include:

Target Variables: Humanitarian assistance provided to IDP households in the community over the last 30 days. This group consists of 16 binary columns representing different types of assistance (e.g., Shelter, Food, Health). We approach in this report our target variables in 2 ways:

> Approach 1: A single numeric target aggregating assistance metrics. This simplifies modeling but may obscure granularity.
>
> Approach 2: A matrix of 16 binary targets, where each represents a specific type of aid. This allows for multi-output classification modeling.

Features: Challenges and barriers to accessing aid, community demographics, coping strategies and so on.

Data Preprocessing & Feature Engineering for Report 2:

Feature Selection: Initially, the dataset contained 975 features. After applying a 20% missing value threshold, the count was reduced to 789. Further reduction to 497 features was achieved by removing low-variance features (threshold = 0.01).

Handling Missing Values: Missing values were imputed using KNN Imputation (k=5) to preserve the underlying data distribution.

Feature Scaling: Min-Max normalization was applied to numerical columns with values exceeding 1 to ensure consistency across features.

2. One Target Approach

The primary concept that emerged was the formulation of a single action-no-action target variable, derived from the 16 target categories previously introduced in Report #1. To accomplish this objective, two methodologies were considered. The initial approach entailed the creation of a variable that would assume a value of 1 if any of the 16 targets equaled 1, and 0 otherwise. The secondary approach was derived from the following question: "Were IDP households in the community able to access humanitarian aid in the last 30 days?". The distribution of both options is quite similar. In the created target variable, 41.7% (13,720 obs.) of the 32,912 observations are ones and 58.3% (19,192 obs.) are zeroes. Utilising the question, the distribution would range from 41.0% to 59.0%, as illustrated in Figure 1. We tested only the first approach.

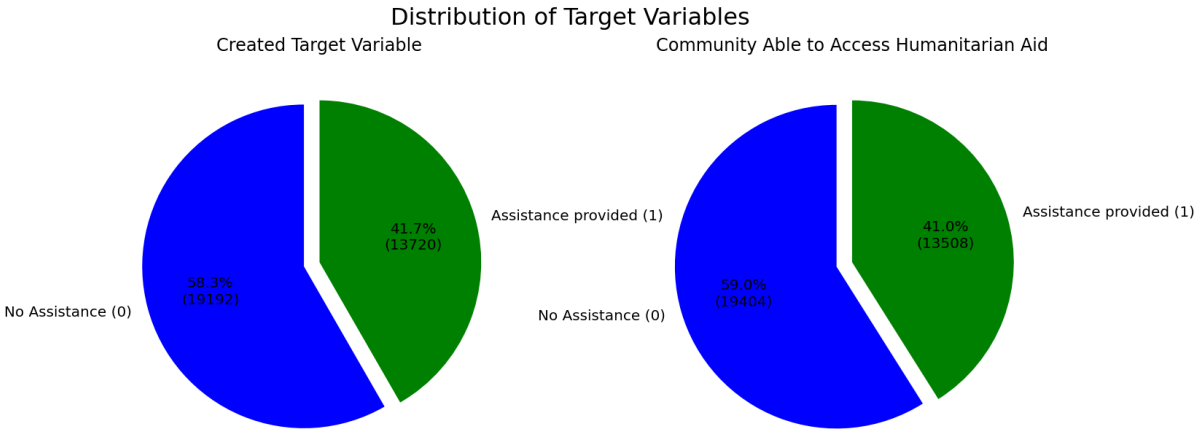

Figure 1 One Target Distribution

2.1 Defining Baseline Models

Following the completion of the train test (with a test size of 0.2 and a random state of 1234), the decision was taken to proceed with the evaluation of four baseline models. Thereafter, the most promising results will be refined to identify the optimal model.

2.1.1 Logistic Classifier

A logistic classifier was selected as the initial model due to its rapid and straightforward interpretation. These classifiers are most effective when the data can be divided linearly. The model was trained initially with 500 iterations, a maximum solver setting, in order to ensure convergence and avoid converging to an undesired solution.

Logistic Classifier				
	precision	recall	f1-score	support
0	0.88	0.95	0.91	2,739
1	0.96	0.91	0.93	3,844
accuracy			0.92	6,583

Table 1 OneTarget - Logistic Classifier - Classification Report

Confusion Matrix		Predicted Values	
		0	1
Real Values	0	2,590	149
	1	347	3,497

Table 2 OneTarget - Logistic Classifier - Confusion Matrix

Intuitively, precision is defined as the ability of the classifier to avoid labeling a negative sample as positive, while recall is defined as the classifier's ability to identify all positive samples. The F1-measure can be interpreted as a weighted harmonic mean of the precision and recall. The accuracy computes the count (or fraction) of correct predictions out of all predictions (scikit-learn.org, 2025a). As illustrated in Table 1, the overall prediction accuracy is satisfactory, with the negative prediction value falling just below the 0.9 threshold. A review of the confusion table (Table 2) reveals that the number of false negative predictions is 347, which exceeds the number of false positive predictions by more than twofold. The model's accuracy is 0.9246, indicating that a mere 8% of the predictions fall outside the categories of true positives or true negatives. To assess the model's overfitting, the cross-validation function was employed with five iterations. The resulting cross-validation accuracy of 0.9200 closely mirrors the initial value, indicating that the model is not overfitting.

2.1.2 Random Forest

Our data is tabular and in binary coding. To catch non-linear relationships, we deployed a Random Forest (RF) model with 100 estimators and a random state of 42.

Random Forest				
	precision	recall	f1-score	support
0	0.92	0.95	0.94	2,739
1	0.96	0.94	0.95	3,844
accuracy			0.95	6,583

Table 3 OneTarget - Random Forest - Classification Report

Confusion Matrix		Predicted Values	
		0	1
Real Values	0	2,598	141
	1	219	3,625

Table 4 OneTarget - Random Forest - Confusion Matrix

The performance overall is better than the logistic classification. The precision, recall, f1-score and accuracy improved. The confusion matrix shows less False Negatives: 219 compared to 347.

2.1.3 XGBoost

The XGBoost classifier is a powerful and fast model. It also captures non-linear relationships and returns the importance of features, which can be used to eliminate unimportant features. This is a great asset to reduce overfitting and improve model performance by eliminating unnecessary features.

XGBoost				
	precision	recall	f1-score	support
0	0.92	0.97	0.94	2,739
1	0.98	0.94	0.96	3,844
accuracy			0.95	6,583

Table 5 OneTarget - XGBoost - Classification Report

Confusion Matrix		Predicted Values	
		0	1
Real Values	0	2,649	90
	1	231	3,613

Table 6 OneTarget - XGBoost - Confusion Matrix

The XGBoost results show great performance. There is a slight improvement in Precision for 1 and Recall for 0. The Confusion Matrix shows more True Negatives and also slightly more False Negatives. On the other hand, the False Positives are significantly reduced.

2.1.4 Support Vector Machine

The Support Vector Machine (SVM) demonstrates a high degree of proficiency in identifying complex classifications between classes. It provides stable differentiation in small to medium-sized datasets. The radial basis function (RBF) kernel enables the application of SVM to non-linear problems. Despite its effectiveness as a model, SVM exhibits suboptimal performance on larger datasets. The calculations required for the model took approximately half an hour to complete on our dataset.

SVM				
	precision	recall	f1-score	support
0	0.90	0.96	0.93	2,739
1	0.97	0.93	0.95	3,844
accuracy			0.95	6,583

Table 7 OneTarget - SVM - Classification Report

Confusion Matrix		Predicted Values	
		0	1
Real Values	0	2,619	120
	1	280	3,564

Table 8 OneTarget - SVM - Confusion Matrix

The SVM demonstrates a performance that is not as proficient as that of the XGBoost model, yet it exhibits a level of proficiency that is analogous to that of the Random Forest model. Among these three models, the SVM exhibits the highest number of False Negatives. However, it demonstrates a superior performance in terms of False Positives when compared with the Random Forest model. The XGBoost and RF models both exhibit higher F1 rates, yet their levels of accuracy are approximately equivalent.

2.1.5 Stochastic Gradient Descent Classifier

The present classifier implements a regularized linear model with stochastic gradient descent (SGD) learning. The SGD Classifier is, by default, a linear support vector machine (SVM) optimized for large datasets (scikit-learn.org, 2025b). The model's chosen loss function was a logistic loss function, with a random state of 42.

SGD				
	precision	recall	f1-score	support
0	0.91	0.89	0.90	2,739
1	0.93	0.94	0.93	3,844
accuracy			0.92	6,583

Table 9 OneTarget - SGD - Classification Report

Confusion Matrix		Predicted Values	
		0	1
Real Values	0	2,449	290
	1	247	3,597

Table 10 OneTarget - SGD - Confusion Matrix

The SGD demonstrates commendable performance; however, it does not reach the level of proficiency exhibited by RF, XGBoost, and SVM. The recall value for 0 is less than 0.90, and the f1 score is also inferior to that of the previous models.

Logistic Classifier				
	precision	recall	f1-score	support
0	0.88	0.95	0.91	2,739
1	0.96	0.91	0.93	3,844
accuracy			0.92	6,583
Random Forest				
0	0.92	0.95	0.94	2,739
1	0.96	0.94	0.95	3,844
accuracy			0.95	6,583
XGBoost				
0	0.92	0.97	0.94	2,739
1	0.98	0.94	0.96	3,844
accuracy			0.95	6,583
SVM				
0	0.90	0.96	0.93	2,739
1	0.97	0.93	0.95	3,844
accuracy			0.95	6,583
SGD				
0	0.91	0.89	0.90	2,739
1	0.93	0.94	0.93	3,844
accuracy			0.92	6,583

Table 11 OneTarget - All Results

2.2 Model Optimization

2.2.1 Gridsearch SGD

The SGD model is expected to yield analogous results to those of the SVM model. To ascertain the viability of optimization, a grid search was conducted to optimize the model. The optimal parameters were determined to be alpha = 0.001, loss function = hinge, maximum iterations = 1000, and the elasticnet penalty. Despite optimization, the model

did not demonstrate superior performance in comparison to the SVM, RF, or XGBoost models. The best results were a accuracy of 0.9232 and F1 scores of 0.9339.

2.2.2 GridSearch & BayesSearch XGBoost

In the context of the baseline test, XGBoost was identified as the most effective model. It is acknowledged that XGBoost is dependent on hyperparameter tuning for its optimization. To enhance the model's performance, the tuning process was executed using GridSearch and the intelligent optimizer, BayesSearch. The GridSearch approach yielded the optimal results, as outlined below: The optimal configuration parameters were determined to be 'colsample_bytree': 1, 'learning_rate': 0.2, 'max_depth': 9, 'n_estimators': 300, and 'subsample': 1. This configuration yielded an accuracy of 0.9530, representing a marginal enhancement over the baseline. The F1-Score reached 0.9603.

BayesSearch identified the feature sample per tree as 0.6, the learning rate as 0.1056, the maximum depth as 8, the number of estimators as 500, and a subsample of 1.0. The accuracy was 0.9526, indicating a negligible enhancement.

2.2.3 XGBoost Feature Importance & Feature Removal

XGBoost shows the most influential features on its classification. For the best estimator Figure 2 shows the Top 30 contributors. Figure 3 shows 27 feature that had no contribution at all.

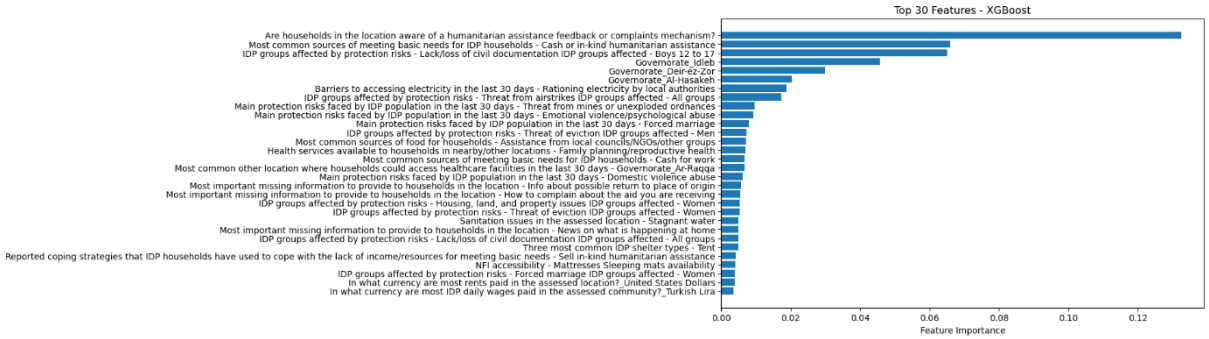

Figure 2 OneTarget - Top 30 Features

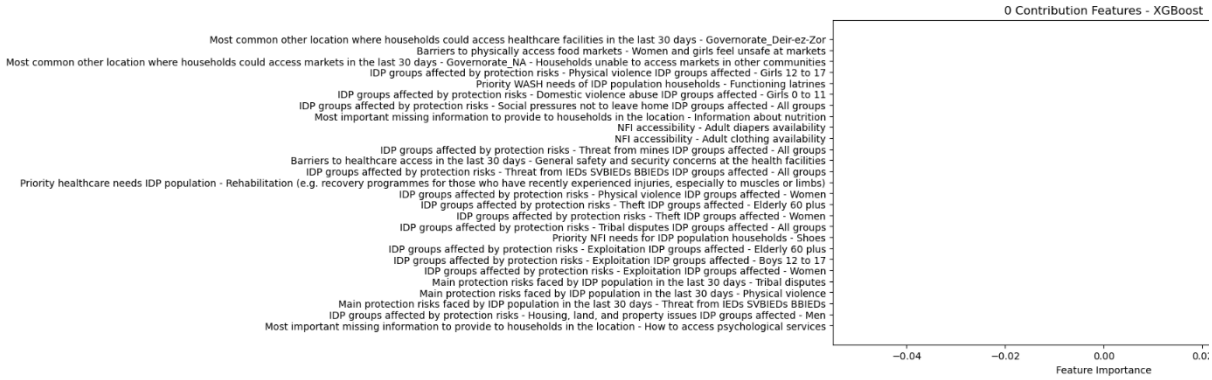

Figure 3 OneTarget - Zero Contribution Features

After excluding these 27 features, the XGBoost model achieved an accuracy of 0.9550 and an F1-Score of 0.9609.An alternative approach involved identifying a threshold to remove the features, thereby achieving the highest possible F1-Score.

Through experimental analysis, it was determined that a feature importance of 0.008 yielded the optimal F1-Score, leading to the removal of features up to 294 (refer to Table 12 and Figure 4).

Num	Features	Threshold	Accuracy	F1 Score	Precision	Recall
7	294	0.0008	0.9533	0.9594	0.9734	0.9458
8	421	0.0005	0.9533	0.9593	0.9757	0.9435
6	259	0.0009	0.9527	0.9589	0.9729	0.9453
5	228	0.001	0.9515	0.9578	0.9721	0.9440
4	118	0.0015	0.9491	0.9556	0.9727	0.9391
3	72	0.002	0.9447	0.9517	0.9697	0.9344
2	50	0.0025	0.9363	0.9442	0.9670	0.9224
1	21	0.005	0.9111	0.9210	0.9573	0.8873
0	8	0.01	0.8787	0.8900	0.9458	0.8405

Table 12 OneTarget - Feature Removal Threshold

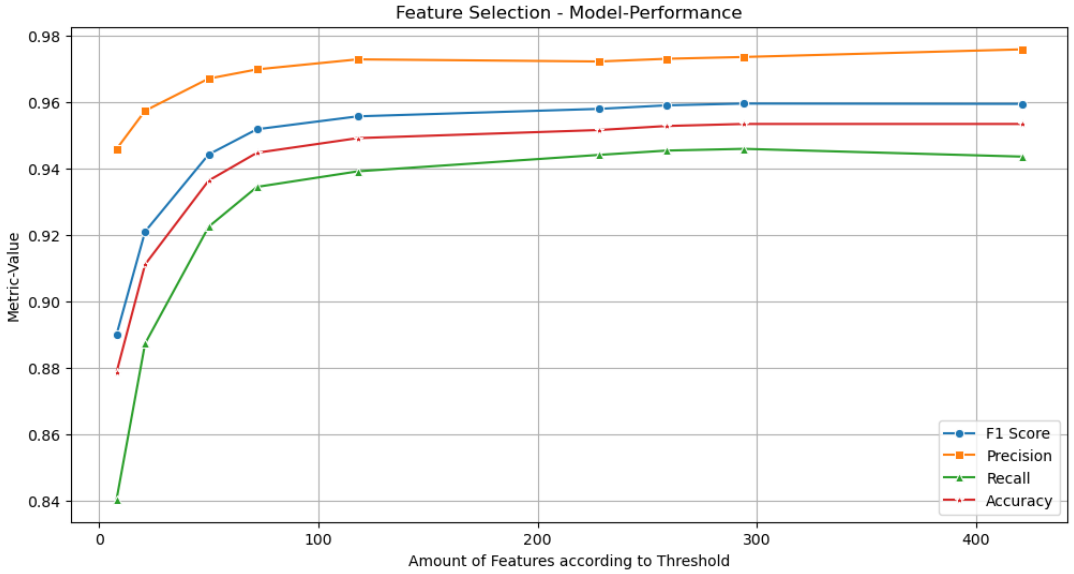

Figure 4 OneTarget - Feature Removal Threshold

Following the execution of all potential thresholds, it was determined that 0.000877577 emerged as the optimal threshold. This resulted in the retention of 267 features within the dataset, yielding an F1-score of 0.9624.

2.2.4 Voting Classifier XGBoost + Random Forest + Logistic Classifier

In the preliminary evaluation, XGBoost and Random Forest exhibited notable outcomes, prompting the exploration of a combined approach through a Voting Classifier to further enhance performance. This investigation involved the implementation of a Logistic Classifier as a baseline model. The initial trial was executed using a soft voting method, while the subsequent trial employed a hard voting method. The first run yielded an accuracy of 0.9535 and an F1-Score of 0.9594, while the second run attained an accuracy of 0.9504 and an F1-Score of 0.9568. The results obtained from the combination of RF and XGBoost through a Voting Classifier were not significantly different from those of XGBoost alone.

2.3 Interpretability

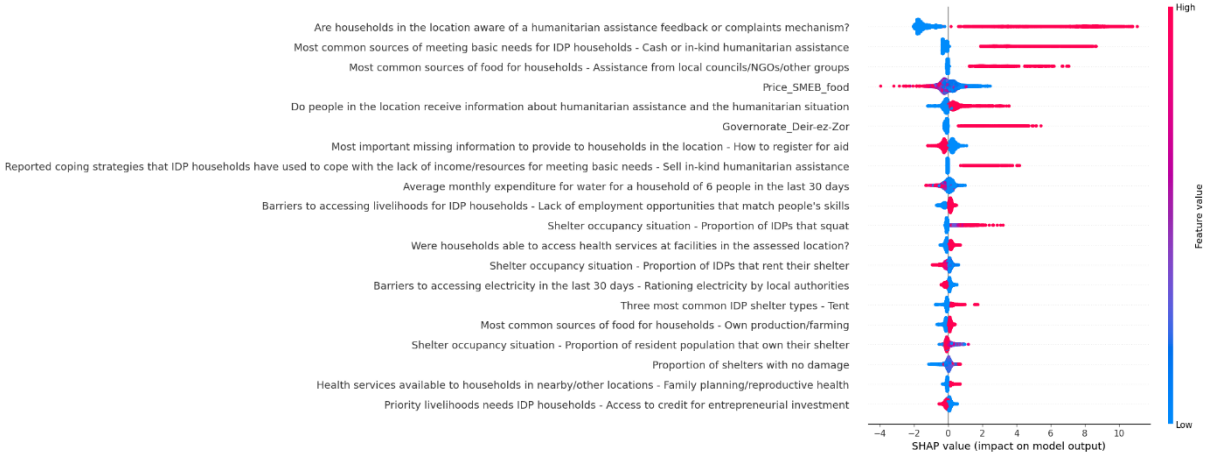

Figure 5 OneTarget - XGBoost Shap

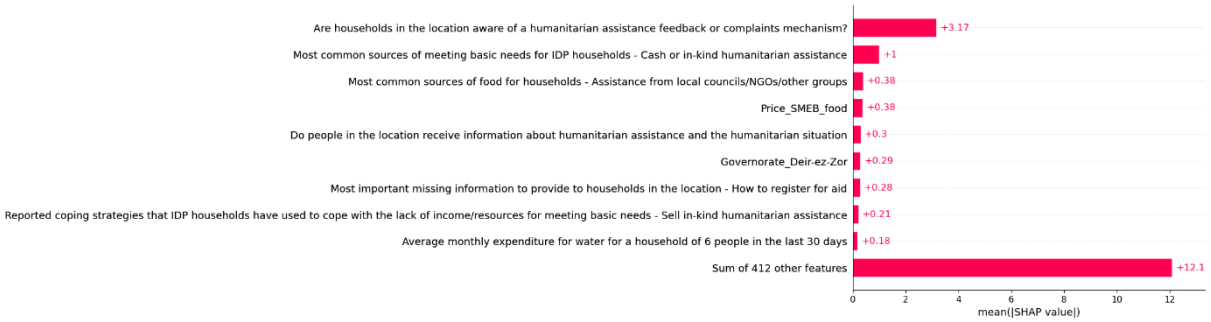

Figure 6 OneTarget - XGBoost Shap Important Features

The SHAP summary plot is a graphical representation of the impact of key features on the XGBoost model's predictions. Features are ranked by importance, with the most influential features at the top. Each data point represents a SHAP value, indicating the extent to which a feature contributes to a higher or lower prediction outcome. Red points indicate high feature values, while blue points represent low values. If high values (red) are predominantly located to the right, the feature enhances the prediction probability. Conversely, features positioned to the left of the plot contribute to a reduction in prediction probability. The analysis identifies awareness of humanitarian assistance, food sources, and economic conditions as the most critical factors. While certain features exhibit a unidirectional impact, others demonstrate a more multifaceted effect.

2.4 Deep Learning

The initial model evaluated was a sequential dense neural network (DNN) with a binary crossentropy loss function and a sigmoid activation function. After 30 epochs, the model demonstrated an accuracy of 0.9966, with a validation accuracy of 0.9368. These results suggest that the model is overfitting, indicating that it has overtrained on the training set.

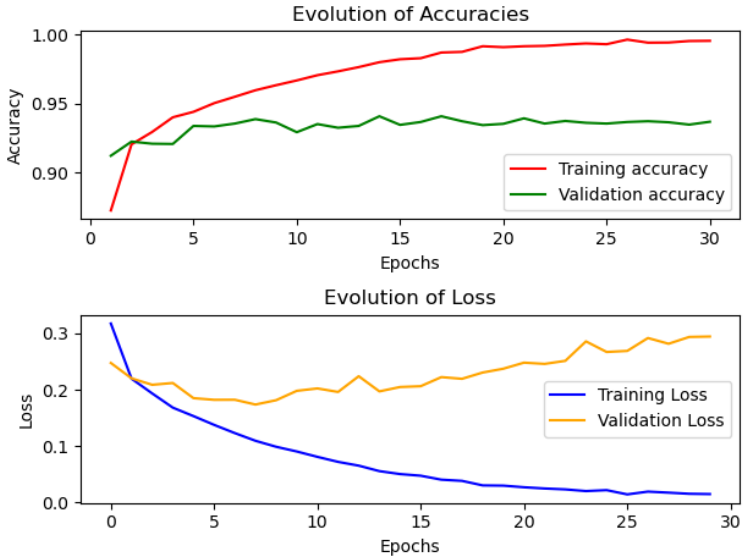

Figure 7 OneTarget - DL Basic Dense Model

In order to address the issue of overfitting, a more complex model was created, incorporating tuning dropout layers and batch normalization layers. Additionally, the Adam optimizer was enabled to select the learning rate from a range of 0.001, 0.0005, and 0.0001. Along with the more complex model, the hyperparameters were tuned using a 10-fold random search from the KerasTuner package. The optimal parameters were then employed in a model comprising 30 epochs with the lr_scheduler callback (monitor='val_loss', factor=0.5, patience=3). The callback was activated eight times in epochs 9, 12, 15, 18, 21, 24, 27, and 30. The final accuracy achieved was 0.9947 and 0.9373 on the validation set.

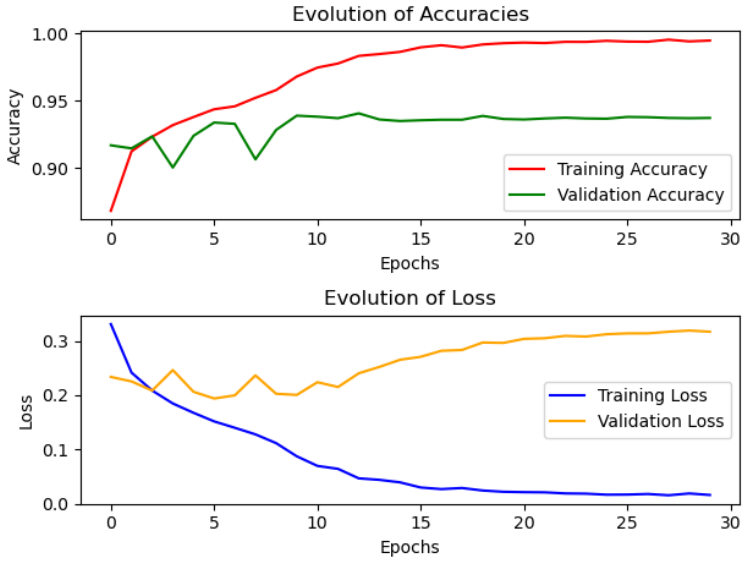

Figure 8 OneTarget - DL Improved Model

Notwithstanding the remarkably high accuracy observed in the training set, the validation accuracy remains distinctly lower, thereby suggesting the occurrence of overfitting. In contrast, machine learning approaches, particularly XGBoost, exhibited notably robust performance without encountering this issue.

3. Multi Label Approach

3.1 Classification of the Problem

Machine Learning Type: This part takes into consideration models into **MultiOutputClassifier to handle with** a multi-label classification problem, where each instance may belong to multiple categories simultaneously.

Application Domain: The model is designed to predict different forms of assistance provided in a humanitarian setting, categorizing each instance into one or more relevant aid types.

Performance Metrics:

Hamming Loss: Directly measures the fraction of misclassified labels in multi-label classification.

F1-Score (Macro & Micro): Evaluates balance between precision and recall across all labels.

The Macro F1-score calculates the F1-score for each individual label and then takes the unweighted average across all labels.

The Micro F1-score calculates the total True Positives (TP), False Positives (FP), and False Negatives (FN) across all labels first and then computes the F1-score from these totals.

$$F1_{micro} = (2 \times \text{Total Precision} \times \text{Total Recall}) / (\text{Total Precision} + \text{Total Recall})$$

Accuracy: Provides an overall correctness of the model but is not the primary focus due to the multi-label nature.

3.2 Model Choice & Optimization

1. Initial Models

We evaluated multiple models to establish baseline performance and identify the most effective approach:

Logistic Regression (MultiOutputClassifier): Baseline linear classifier, interpretable but not optimal for complex interactions.

Random Forest (MultiOutputClassifier): Robust to non-linearity and missing data, performed moderately well.

SVM (MultiOutputClassifier): Strong performance but computationally expensive for large datasets.

Naive Bayes: Discarded due to its assumption of feature independence, which does not hold in this dataset.

Decision Tree (MultiOutputClassifier): Performed poorly according to Hamming Loss due to high variance. As a single tree model, it lacks the generalization power of

ensemble methods like Random Forest and XGBoost, which combine multiple trees to improve robustness and reduce overfitting.

KNN (MultiOutputClassifier): KNN was excluded due to its sensitivity to irrelevant features and computational inefficiency with high-dimensional data. Despite reasonable performance, it struggles with scalability, making it less suitable for our dataset with high number of features. Additionally, KNN tends to perform poorly when features are highly imbalanced.

Results:

Model	Hamming Loss	F1-Score (Macro)	F1-Score (Micro)	Accuracy
Logistic Regression	0.0293	0.5	0.77	0.6922
Random Forest	0.0216	0.44	0.81	0.7597
Naive Bayes	0.1007	0.31	0.5	0.384
SVM	0.0304	0.49	0.77	0.6859
KNN	0.0261	0.54	0.8	0.6913
Decision Tree	0.031	0.49	0.77	0.6868

Table 13 MultiTarget - Base Models

Final Conclusion: Based on the evaluation metrics, SVM and Random Forest have demonstrated solid performance, particularly in terms of accuracy and micro F1-score. However, XGBoost consistently outperforms them in most metrics, making it a strong candidate for inclusion in the final model selection. Moving forward, we will proceed with SVM, Random Forest, and XGBoost, leveraging XGBoost's robustness to further enhance predictive performance in the next phase.

2. Hyperparameter Tuning with GridSearchCV

To optimize model performance, GridSearchCV (3-fold cross-validation) was applied to Random Forest, SVM, and XGBoost. The best hyperparameters obtained were:

Random Forest:

```
max_depth=20, min_samples_split=2, n_estimators=200
```

SVM:

```
C=10, kernel='rbf'
```

XGBoost:

```
learning_rate=0.1, max_depth=10, n_estimators=200
```

3. Model Performance Comparison

	Hamming Loss	F1-Score (Macro)	F1-Score (Micro)	Accuracy
Random Forest	0.021799	0.456789	0.825133	0.758013
SVM	0.020194	0.592332	0.846237	0.771229
XGBoost	0.019463	0.599793	0.850648	0.783381

Table 14 MultiTarget - Hyperparameter Tuning Models

Based on the results, XGBoost and SVM demonstrated superior performance, leading to their selection for ensemble methods.

XGBoost consistently delivers superior results due to its ability to efficiently handle large feature spaces while maintaining computational efficiency. Its robust performance on imbalanced datasets makes it particularly well-suited for complex classification tasks. Additionally, XGBoost leverages parallelization, significantly reducing training time compared to traditional boosting methods. Another key advantage is its built-in regularization mechanisms, which help prevent overfitting, ensuring better generalization on unseen data. These factors collectively contribute to its strong predictive capabilities and make it a powerful choice for high-dimensional and imbalanced classification problems.

3.3 Ensemble Models (Stacking & Voting)

3.3.1 Stacking Classifier

Chosen Base Models: Best XGBoost and SVM models (trained separately for each label).

Meta Model: Logistic Regression (selected for its generalization capability).

Implementation: Separate stacking classifiers were trained for each label using StackingClassifier, ensuring each label had its own specialized ensemble.

3.3.2 Voting Classifier (Hard & Soft Voting)

Hard Voting: Assigns the most common label among models.

Soft Voting: Uses averaged probabilities for final decision.

Results:

Model	Hamming Loss	F1-Score (Macro)	F1-Score (Micro)	Accuracy
Stacking	0.019311	0.587003	0.852073	0.78171
Hard Voting	0.020052	0.562929	0.841465	0.773204
Soft Voting	0.01914	0.596515	0.85289	0.783837

Table 15 MultiTarget - Voting Classifier

Findings:

Soft Voting achieved the best overall performance, outperforming both individual models and Hard Voting.

Stacking had slightly better accuracy than Soft Voting but lower F1-score.

However, Soft voting didn't significantly outperform other models, meaning its contribution to overall predictive performance is marginal. Given this, instead of focusing on ensemble voting, we should analyze feature importance and model interpretability, particularly within XGBoost, which provides built-in feature importance metrics.

3.4 SMOTE-Enhanced Models (XGBoost, SVM, Random Forest)

To address the issue of class imbalance in multi-label classification, MLSMOTE was applied, following the methodology from the thesis "Handling Data Imbalance in Multi-label Classification" by Sukhwani (2020). Traditional SMOTE is not directly compatible with multi-label problems, as it assumes binary or multi-class settings. Instead, MLSMOTE was used to oversample minority labels, ensuring that each underrepresented label reached the median frequency of 744.

This approach was necessary to avoid overfitting to dominant labels while improving the recall of minority classes. The SMOTE-enhanced models were evaluated, and the results are compared below.

Model	Hamming Loss	F1-Score (Macro)	F1-Score (Micro)	Accuracy
XGBoost	0.02107	0.59	0.84	0.7653
SVM	0.02047	0.59	0.84	0.7691
Random Forest	0.02294	0.44	0.81	0.7448

Table 16 MultiTarget - MLSMOTE Results

XGBoost and SVM consistently outperform Random Forest across all major evaluation metrics. Among them, SVM achieves the highest Exact Match Accuracy (76.91%), narrowly surpassing XGBoost. However, XGBoost demonstrates a better balance between precision and recall, effectively mitigating overfitting while maintaining strong predictive performance. In contrast, Random Forest struggles significantly with minority classes, particularly in GBV Services, Explosive Hazard Risk Awareness, and Agricultural Supplies, where both precision and recall remain notably low. Additionally, the macro F1-score is substantially lower than the micro F1-score across all models, highlighting the ongoing challenge of accurately predicting underrepresented labels in this multi-label classification setting.

Final Conclusion: SMOTE slightly improved the micro F1-score, indicating better performance across all labels, especially for minority classes. However, the drop in macro F1-score suggests that predicting minority classes remains a challenge. Additionally, the slight increase in Hamming loss indicates more label-level misclassifications. Despite applying SMOTE, XGBoost and SVM continue to be the best-performing models. Given that overall accuracy decreased slightly and the improvements in minority classes were minimal, it seems more practical to proceed with the original models combined with feature selection rather than relying on SMOTE.

3.5 Feature importance analysis using XGBoost:

This will help identify redundant features that could be removed for better efficiency.

Several criteria can be considered to determine a threshold based on feature importance scores:

Natural Breakpoint (Elbow Method) analysis shows that a decrease of around 0.01 in the graph. The few most important features (e.g. top 10-20) have significantly higher importance scores. Below 0.005, the decline becomes less pronounced, so it may be less effective from here on.

We looked a narrow selection, that take features at ‘0.01 and above’. Then, for a wider but balanced selection, it makes sense to take ‘0.005 and above’ or ‘0.001 and above’.

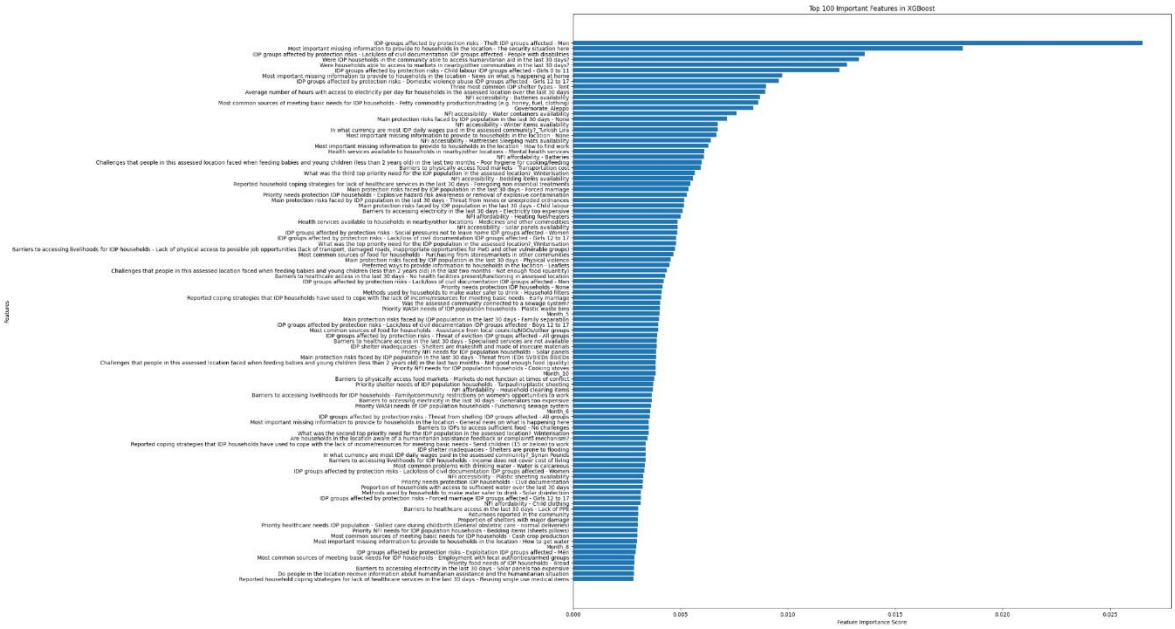

Figure 9 MultiTarget - Feature Importance

Feature_Selection_vs_Performance

	Number of Selected Features	Hamming Loss	F1-Score (Macro)	F1-Score (Micro)	Accuracy
XGBoost (Thr: 0.001)	340	0.019615	0.580884	0.849219	0.780799
XGBoost (Thr: 0.005)	33	0.035024	0.419686	0.725623	0.674009
XGBoost (Thr: 0.01)	6	0.049484	0.088401	0.54055	0.615069

Table 17 MultiTarget - Feature Selection & Performance

The table presents the impact of different feature importance thresholds on model performance. As the threshold increases, the number of selected features decreases significantly—from 340 at 0.001 to just 6 at 0.01.

From the results, we observe that selecting only the most important features (higher thresholds) leads to a sharp decline in performance across all key metrics (Hamming Loss, F1-Score, and Accuracy). The model with 340 selected features (Threshold: 0.001) achieves the best balance, maintaining high accuracy (78.08%) and strong F1 scores (Macro: 0.58, Micro: 0.85) while keeping Hamming Loss low.

Given this analysis, continuing with the 340 selected features is the optimal approach, as it retains strong predictive power while reducing dimensionality compared to using all features.

3.6 Interpretability

3.6.1 Error Analysis

Since this is multi-label classification, we can generate label-wise confusion matrices rather than a standard one. For each label, we generated a confusion matrix to compare true vs. predicted values. Then, we visualized it using a heatmap to easily spot misclassifications.

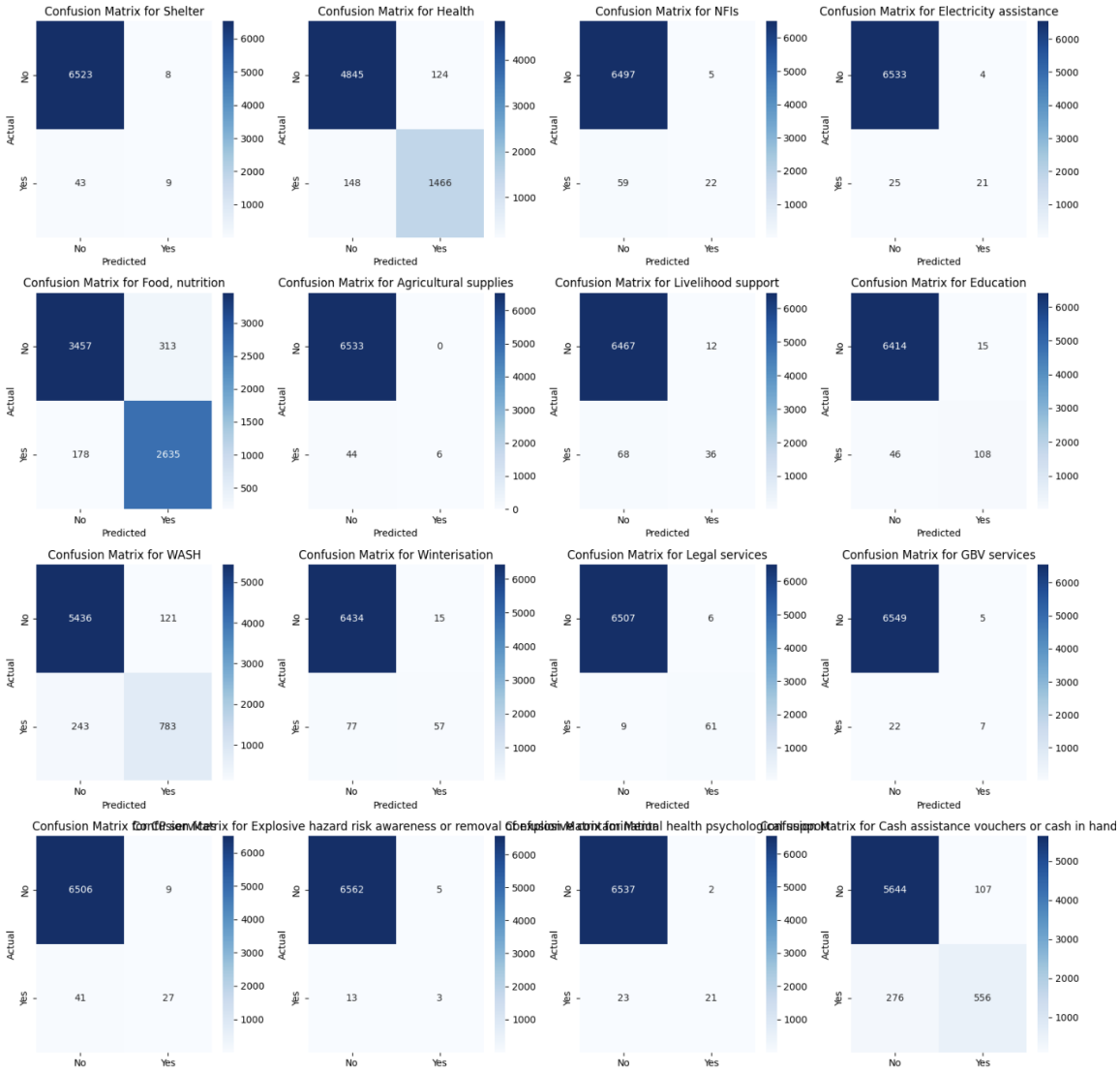

Figure 10 MultiTarget - Confusion Matrix Heatmap

The confusion matrices reveal key insights into the model's performance across different labels. High-imbalance issues are evident in categories like Agricultural Supplies, Explosive Hazard Risk Awareness, and Mental Health Psychological Support, where very few positive cases were correctly predicted, indicating poor recall. In contrast, labels like Food, Nutrition, WASH, and Cash Assistance Vouchers show better performance, with a higher number of true positives and relatively fewer false negatives. However, some labels, such as Education and Livelihood Support, exhibit moderate false negatives, suggesting that the model struggles to correctly classify minority classes in certain categories.

Overall, the model performs well on frequently occurring labels but still struggles with rare ones, reinforcing the need for further optimization, such as threshold tuning, or alternative loss functions tailored for imbalanced multi-label classification.

3.6.2 Per-Class Threshold Optimization for XGBoost (Threshold: 0.001)

```
XGBoost (Per-Class Thresholds) Results:
Hamming Loss: 0.019538963998177124
F1-Score (Macro): 0.635697740089314
F1-Score (Micro): 0.8540425531914894
Accuracy: 0.7792799635424579

Optimized Per-Class Thresholds:
{'Shelter': 0.39999999999999997, 'Health': 0.39999999999999997, 'NFIs': 0.3, 'Electricity assistance': 0.3, 'Food, nutrition': 0.4999999999
9999994, 'Agricultural supplies': 0.3, 'Livelihood support': 0.3, 'Education': 0.39999999999999997, 'WASH': 0.3, 'Winterisation': 0.3, 'Leg
al services': 0.3, 'GBV services': 0.3, 'CP services': 0.35, 'Explosive hazard risk awareness or removal of explosive contamination': 0.3,
'Mental health psychological support': 0.3, 'Cash assistance vouchers or cash in hand': 0.35}
```

The per-class threshold tuning has notably improved performance, especially for macro F1-score and micro F1-score, while maintaining accuracy at a similar level. Some common labels (e.g., Food, nutrition, Cash assistance, CP services) have higher thresholds (0.35 - 0.5). This prevents over-prediction and reduces false positives.

Some minority labels (e.g., GBV services, Electricity assistance, NFIs, WASH) have lower thresholds (0.3 - 0.35). This improves recall, helping to detect rare events better.

Conclusion:

Macro F1-score improved significantly (from 0.5809 to 0.6357). This suggests better performance across all labels, including minority classes.

Adjusting thresholds helped balance precision and recall for underrepresented labels.

Micro F1-score slightly increased (from 0.8492 to 0.8540). This indicates that the overall model improved without sacrificing general performance.

Accuracy remained stable (from 0.7808 to 0.7793). This suggests that the per-class thresholding did not introduce more incorrect predictions at the sample level.

Hamming Loss slightly improved (from 0.0196 to 0.0195). The decrease in Hamming Loss suggests fewer individual label misclassifications.

3.6.3 SHAP analysis using XGBoost

The SHAP summary plot visualizes the impact of features on the model's predictions. Even if this is a multi-label classification problem with 16 targets, the SHAP values shown in the plot belong to a selected target namely Shelter as an example.

Features with a wide spread of SHAP values (large variance) have a stronger effect on predictions. If a feature has a mostly positive impact, it increases the likelihood of predicting a target label as "Yes" (1). If a feature's SHAP values are centered around 0, it has little influence (Dong et al., 2021).

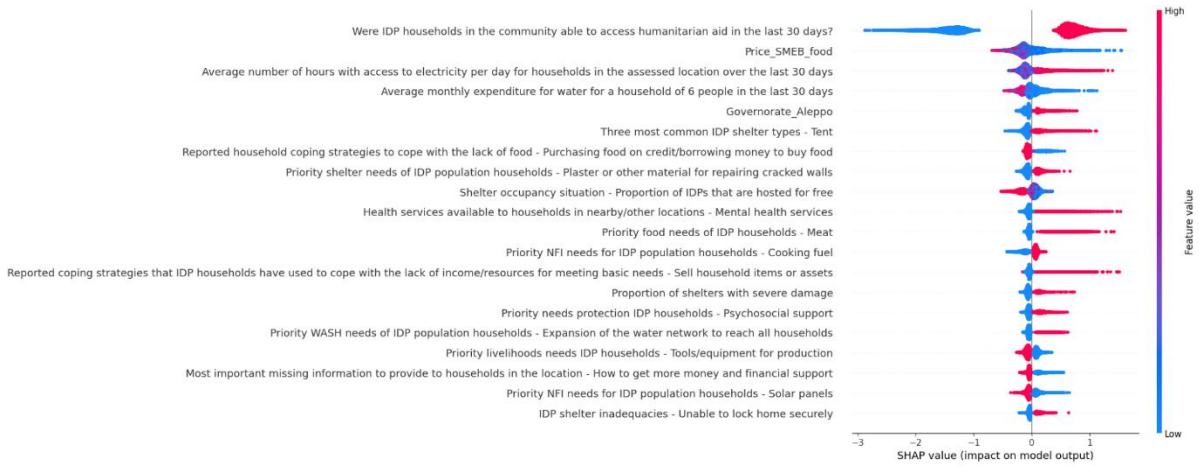

Figure 11 MultiTarget - SHAP Values XGBoost

This SHAP summary plot illustrates the impact of the 340 most important features in the XGBoost model for Target Schelter. The left axis lists the top contributing features, where those at the top have the highest influence on the model's predictions. The SHAP values on the x-axis indicate how much each feature pushes the prediction positively (right) or negatively (left). Red points represent high feature values, while blue points indicate low values.

For instance, "Were IDP households in the community able to access humanitarian aid in the last 30 days?" has the most significant effect on predictions. Other key features, such as food prices, access to electricity, and shelter inadequacies, play a crucial role in determining outcomes. The distribution patterns suggest that some features consistently drive predictions in a certain direction, reinforcing their importance. This analysis helps validate feature selection and guides further model refinement by focusing on the most impactful variables.

3.6.4 SHAP Interpretation Across Labels

The SHAP analysis has provided critical insights into the drivers of humanitarian assistance predictions across 16 labels. These insights can help prioritize resources, target interventions more effectively, and build trust in the model's predictive capabilities through transparent, interpretable outputs.

Assistance Type	Top Contributing Features	SHAP Insights
Shelter Assistance	Access to humanitarian aid, shelter damage, material shortages	Locations with severe shelter damage and resource limitations show increased need for assistance.
Health Assistance	Barriers to healthcare, malnutrition treatment gaps	Lack of healthcare access strongly correlates with higher health assistance needs.
NFIs (Non-Food Items)	Cooking fuel, shelter repair materials, coping strategies	Communities selling household items indicate a higher demand for NFIs.
Electricity Assistance	Fuel costs, generator usage, power reliability	Areas with frequent power outages and high fuel expenses see increased predictions.
Food Assistance	Food price inflation, meal skipping, reliance on credit	Food insecurity and high dependency on credit purchases predict higher food assistance needs.
Agricultural Supplies	Food crop production levels, financial limitations	Low agricultural output correlates with increased need for agricultural assistance.
Livelihood Support	Employment barriers, financial insecurity	Unemployment rates strongly influence livelihood support predictions.
Education Assistance	School availability, child protection risks	Limited educational facilities and high early marriage risks drive education support needs.
WASH Assistance	Water infrastructure issues, drinking water concerns	Poor water network conditions increase WASH assistance demands.
Winterization Assistance	Heating deficiencies, seasonal conditions	Cold seasons and heating shortages predict higher winterization needs.
Legal Services Assistance	Documentation challenges, protection risks	Households without legal documents show increased legal assistance requirements.
GBV Services Assistance	Early marriage risks, protection concerns	Higher GBV risks in a region predict increased allocation of services.
CP Services Assistance	Child labor prevalence, school dropout rates	Protection issues like child labor and dropout rates influence CP assistance needs.

Explosive Hazard Awareness Assistance	Reported contamination risks, historical conflict data	Regions with known contamination show heightened awareness predictions.
Mental Health Assistance	Psychological distress reports, lack of mental health services	Communities experiencing mental health crises predict increased support demand.
Cash Assistance	Employment opportunities, financial barriers	Areas with severe financial insecurity predict higher cash assistance needs.

Table 18 MultiTarget - SHAP Interpretation Table

4. Bagging and Boosting Multi Label RF and SVM

The approach here is to focus on the random forest and the support vector machine models as they have shown the best results while wrapped within the MultiOutputClassifier. The approach is to implement the Bagging and the Boosting techniques within in different ways: separately, wrapped together, and through a Bayes search.

4.1 Imbalance of the data and removal of low frequency targets

Before starting with Bagging and Boosting however, we are looking at 16 target variables many of which are of low frequency

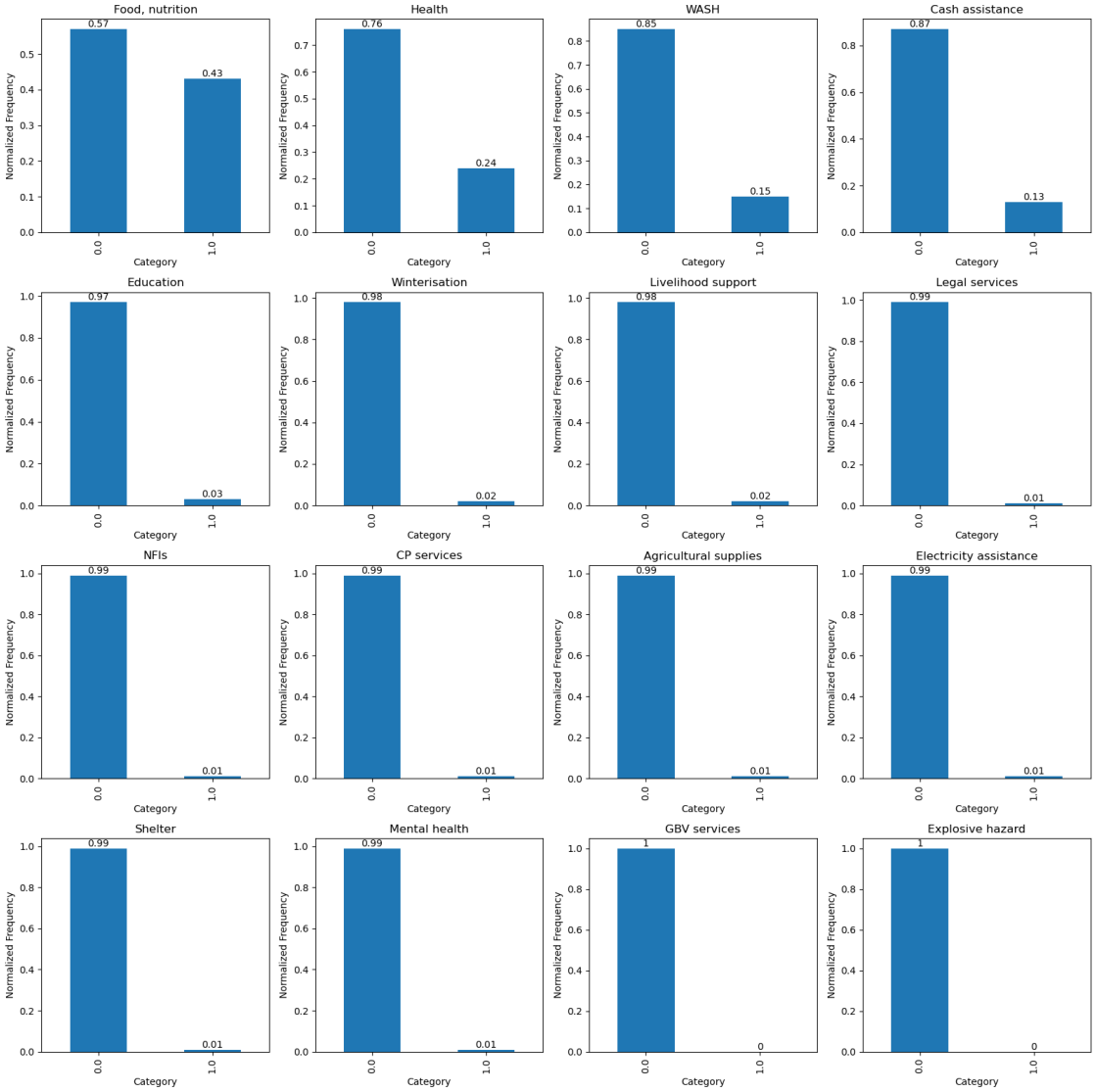

Figure 12 MultiTarget - Imbalance Targets

As seen only 25% of the target variables have a frequency above 1%. On this basis we have decided to look at the MultiOutputClassifier of Random Forest and Support Vector Machine with the entire 16 targets and then with the highest 4 targets.

Our assumption was that with less targets that are balanced the models will perform better than with more imbalanced targets. However the results show that more data is a better performance for the overall metrics of the models.

We will first look at the RF:

RF with 16 targets

```
Evaluation Metrics:
Hamming Loss: 0.02099156919337688
Exact Match Accuracy: 0.7631778824244265
Classification Report:
```

	precision	recall	f1-score	support
Food, nutrition	0.90	0.92	0.91	2862
Health	0.95	0.88	0.91	1638
WASH	0.90	0.67	0.77	1018
Cash assistance vouchers or cash in hand	0.89	0.57	0.69	828

RF with 4 targets

```
Evaluation Metrics:
Hamming Loss: 0.06178793862980404
Exact Match Accuracy: 0.8035849916451466
Classification Report:
```

	precision	recall	f1-score	support
Food, nutrition	0.90	0.92	0.91	2862
Health	0.95	0.88	0.91	1638
WASH	0.90	0.67	0.77	1018
Cash assistance vouchers or cash in hand	0.89	0.57	0.69	828

And then the SVM:

SVM with 16 targets

```
SVM - Evaluation Metrics:
Hamming Loss: 0.05606296521342853
Exact Match Accuracy: 0.5503569801002582
Classification Report:
```

	precision	recall	f1-score	support
Food, nutrition	0.84	0.95	0.89	2862
Health	0.78	0.92	0.84	1638
WASH	0.52	0.88	0.65	1018
Cash assistance vouchers or cash in hand	0.41	0.88	0.56	828

SVM with 4 targets

```
SVM - Evaluation Metrics:
Hamming Loss: 0.12661400577244417
Exact Match Accuracy: 0.6497037824699985
Classification Report:
```

	precision	recall	f1-score	support
Food, nutrition	0.84	0.95	0.89	2862
Health	0.78	0.92	0.84	1638
WASH	0.52	0.88	0.65	1018
Cash assistance vouchers or cash in hand	0.41	0.88	0.56	828

In the above as well as all the following Random Forest our parameters are left not changed for comparability:

RF:

```
random_state=1234, n_estimators=100, n_jobs=-1, class_weight='balanced'
```

SVM:

```
random_state=1234, C=1.0, max_iter=5000, class_weight='balanced'
```

The results between 16 and 4 targets models don't show much difference. Sometimes there is a trade between the hamming loss and the accuracy, but essentially the recall (which is most important metric in our case remains unchanged). A very major reason for this could be the fact that we are balancing the imbalanced data by adding a weighting factor which is proportionally equal to the fraction of the class to the others.

We have also explored the same comparison for the bagged and boosted RF and SVM and the results in terms of the recall were the same, with a difference with the Hamming Loss and Accuracy of the overall model. But it is not comparable as well because it measures the accuracy and loss of two different models (4 and 16 targets models).

4.2 Bagging and Boosting in different methods

For each of our models RF and SVM we have performed bagging and boosting in different ways:

4.2.1. Bagging the model and separately boosting the model

In this experiment the model is bagged and results are compared between the model itself and the bagged model. Then the original model itself is boosted and the results are compared between the original model and the boosted model.

4.2.1.1. Random Forest bagging and boosting separately

In this experiment the RF model is bagged and results are compared between the RF model itself and the bagged RF model. Then the RF model itself is boosted and the results are compared between the RF model and the boosted RF model. Bagging parameters:

n_estimators=10, random_state=1234

And the results of the bagging are same quality of the model itself with recall and hamming loss as follows

```
Bagging Random Forest - Evaluation Metrics:
Hamming Loss: 0.022244797204921767
Exact Match Accuracy: 0.7537596840346347
Classification Report:
```

	precision	recall	f1-score	support
Food, nutrition	0.89	0.92	0.91	2862
Health	0.95	0.87	0.91	1638
WASH	0.91	0.63	0.74	1018
Cash assistance vouchers or cash in hand	0.89	0.52	0.66	828
NFIs	0.93	0.59	0.72	174
Electricity assistance	0.79	0.23	0.36	129
Agricultural supplies	0.72	0.14	0.24	125
Livelihood support	0.96	0.89	0.92	81
Education	1.00	0.12	0.21	67
Shelter	0.82	0.15	0.25	60
Winterisation	0.50	0.02	0.04	47
Legal services	1.00	0.29	0.45	45
GBV services	1.00	0.03	0.06	31
CP services	1.00	0.25	0.40	40
Explosive hazard risk awareness or removal of explosive contamination	0.00	0.00	0.00	31
Mental health psychological support	1.00	0.04	0.08	24

Figure 13 MultiTarget - RF Bagging

Whereas the boosting parameters are:

n_estimators=50, random_state=1234

And the results of the boosted RF model are

```
Boosting Random Forest - Evaluation Metrics:
Hamming Loss: 0.02259608081421844
Exact Match Accuracy: 0.7413033571320067
Classification Report:
```

	precision	recall	f1-score	support
Food, nutrition	0.90	0.87	0.89	2862
Health	0.95	0.89	0.92	1638
WASH	0.91	0.66	0.77	1018
Cash assistance vouchers or cash in hand	0.89	0.47	0.62	828
NFIs	0.95	0.62	0.75	174
Electricity assistance	0.74	0.27	0.40	129
Agricultural supplies	0.79	0.22	0.34	125
Livelihood support	0.96	0.90	0.93	81
Education	1.00	0.18	0.30	67
Shelter	0.65	0.18	0.29	60
Winterisation	0.50	0.02	0.04	47
Legal services	1.00	0.38	0.55	45
GBV services	1.00	0.06	0.12	31
CP services	1.00	0.33	0.49	40
Explosive hazard risk awareness or removal of explosive contamination	1.00	0.13	0.23	31
Mental health psychological support	1.00	0.04	0.08	24

Figure 14 MultiTarget - RF Bagging & Boosting

We see that there is no improvement

4.2.1.2. Support Vector Machine bagging and boosting separately

In this experiment the SVM model is bagged and results are compared between the SVM model itself and the bagged SVM model. Then the SVM model itself is boosted and the results are compared between the SVM model and the boosted SVM model. The results of the SVM itself are better in terms of Hamming loss and recall in comparison to the RF model. Therefore the results of the bagging and boosting are better than what we have seen above, but still the don't exceed the quality of the SVM model itself by far.

The parameters of bagging in SVM are like those in the RF, but the results of bagging the SVM:

```
Bagging SVM - Evaluation Metrics:
Hamming Loss: 0.04062547470758013
Exact Match Accuracy: 0.615828649551876
Classification Report:
```

	precision	recall	f1-score	support
Food, nutrition	0.84	0.95	0.89	2862
Health	0.79	0.92	0.85	1638
WASH	0.54	0.86	0.66	1018
Cash assistance vouchers or cash in hand	0.42	0.85	0.57	828
Education	0.52	0.76	0.62	174
Winterisation	0.34	0.63	0.44	129
Livelihood support	0.31	0.46	0.37	125
Legal services	0.99	0.85	0.91	81
NFIs	0.15	0.36	0.22	67
CP services	0.39	0.47	0.43	60
Agricultural supplies	0.08	0.11	0.09	47
Electricity assistance	0.47	0.44	0.45	45
Shelter	0.25	0.29	0.27	31
Mental health psychological support	0.82	0.57	0.68	40
GBV services	0.57	0.26	0.36	31
Explosive hazard risk awareness or removal of explosive contamination	0.42	0.21	0.28	24

Figure 15 MultiTarget - SVM Bagging

So in this specific case, the bagging has improved the results. While maintaining the recall at 0.95 and precision for Food, nutrition, the hamming loss has decreased from 0.05 to 0.04.

The boosting, however, with the same boosting parameters for RF as of SVM doesn't improve the model performance.

```
Boosting SVM - Evaluation Metrics:
Hamming Loss: 0.07123461947440377
Exact Match Accuracy: 0.49840498253076104
Classification Report:
```

	precision	recall	f1-score	support
Food, nutrition	0.82	0.89	0.85	2862
Health	0.81	0.77	0.79	1638
WASH	0.51	0.82	0.63	1018
Cash assistance vouchers or cash in hand	0.40	0.81	0.53	828
Education	0.30	0.82	0.44	174
Winterisation	0.21	0.88	0.33	129
Livelihood support	0.18	0.81	0.29	125
Legal services	0.66	0.93	0.77	81
NFIs	0.06	0.61	0.11	67
CP services	0.14	0.78	0.23	60
Agricultural supplies	0.05	0.51	0.08	47
Electricity assistance	0.17	0.71	0.27	45
Shelter	0.06	0.61	0.10	31
Mental health psychological support	0.26	0.88	0.40	40
GBV services	0.17	0.68	0.28	31
Explosive hazard risk awareness or removal of explosive contamination	0.09	0.71	0.16	24

Figure 16 MultiTarget - SVM Bagging & Boosting

4.2.2. Boosting the bagged model

In this experiment the model is bagged and results are wrapped within a booster model, and then results are compared between the model itself and the boosted bagged model. In both cases the same parameters of bagging and boosting are introduced here but wrapped inside their functions:

Bagging params: n_estimators=10, random_state=1234
Boosting params: n_estimators=50, random_state=1234

4.2.2.1. Random Forest boosting the bagged models

So here the RF model is bagged and results are wrapped within a booster model, and then results are compared between the RF model itself and the boosted bagged RF model.

4.2.2.2. Support Vector Machine boosting the bagged models

And here the SVM model is bagged and results are wrapped within a booster model, and then results are compared between the SVM model itself and the boosted bagged SVM model.

```
Boosting of Bagging RandomForest - Evaluation Metrics:
Hamming Loss: 0.022254291356524382
Exact Match Accuracy: 0.7514810876500075
Classification Report:
```

	precision	recall	f1-score	support
Shelter	1.00	0.06	0.12	31
Health	0.94	0.88	0.91	1638
NFIs	1.00	0.13	0.24	67
Electricity assistance	1.00	0.33	0.50	45
Food, nutrition	0.90	0.92	0.91	2862
Agricultural supplies	0.00	0.00	0.00	47
Livelihood support	0.79	0.15	0.26	125
Education	0.93	0.59	0.72	174
WASH	0.91	0.61	0.73	1018
Winterisation	0.79	0.23	0.36	129
Legal services	0.96	0.86	0.91	81
GBV services	0.00	0.00	0.00	31
CP services	0.91	0.17	0.28	60
Explosive hazard risk awareness or removal of explosive contamination	0.00	0.00	0.00	24
Mental health psychological support	1.00	0.23	0.37	40
Cash assistance vouchers or cash in hand	0.88	0.52	0.66	828

Figure 17 MultiTarget - SVM Bagging & Boosting

4.2.3. Boosting the bagged model with a bayes search for parameters

This experiment is similar to the one above but a bayes search is performed across 4 different combinations (to reduce computation) to search for the best parameters of the bagging boosting.

```
Boosting of Bagging SVM - Evaluation Metrics:
Hamming Loss: 0.07052255810420781
Exact Match Accuracy: 0.5053926781102841
Classification Report:
```

	precision	recall	f1-score	support
Shelter	0.06	0.68	0.10	31
Health	0.80	0.76	0.78	1638
NFIs	0.05	0.61	0.09	67
Electricity assistance	0.19	0.73	0.30	45
Food, nutrition	0.82	0.90	0.86	2862
Agricultural supplies	0.04	0.49	0.07	47
Livelihood support	0.17	0.74	0.28	125
Education	0.27	0.80	0.40	174
WASH	0.50	0.82	0.62	1018
Winterisation	0.17	0.84	0.29	129
Legal services	0.82	0.89	0.85	81
GBV services	0.16	0.68	0.26	31
CP services	0.15	0.77	0.25	60
Explosive hazard risk awareness or removal of explosive contamination	0.10	0.67	0.17	24
Mental health psychological support	0.26	0.82	0.40	40
Cash assistance vouchers or cash in hand	0.48	0.65	0.55	828

Figure 18 MultiTarget - SVM Bagging & BayesSearch

In SVM and RF boosting the bagged model didn't increase the model performance.

4.2.3.1. Random Forest boosting the bagged models with a bayes search

Here we implement the experiment first with the RF model using the bayes search

```
search_spaces_rf = {
    "estimator__n_estimators": (10, 100),                # AdaBoost n_estimators
    "estimator__learning_rate": (0.01, 2.0, 'log-uniform'),     # AdaBoost learning_rate
    "estimator__estimator__n_estimators": (10, 100),          # Bagging n_estimators
    "estimator__estimator__max_samples": (0.5, 1.0, 'uniform'),   # Bagging max_samples
```

```
    "estimator__estimator__estimator__n_estimators": (50, 200),   # RandomForest n_estimators
    "estimator__estimator__estimator__max_depth": (5, 20)        # RandomForest max_depth}
```

4.2.3.2. Support Vector Machine boosting the bagged models with a bayes search

And then with the SVM model using the bayes search we have searched the following search spaces:

```
search_spaces = {
    "estimator__n_estimators": (10, 100),                # AdaBoost n_estimators
    "estimator__learning_rate": (0.01, 2.0, 'log-uniform'),    # AdaBoost learning_rate
    "estimator__estimator__n_estimators": (10, 100),          # Bagging n_estimators
    "estimator__estimator__max_samples": (0.5, 1.0, 'uniform'), # Bagging max_samples
    "estimator__estimator__estimator__C": (0.1, 10.0, 'log-uniform'),  # LinearSVC C
    "estimator__estimator__estimator__max_iter": [1000, 3000, 5000, 7000]  # LinearSVC max_iter}
```

And the results didn't show any development in comparison to the others as seen below. Same with the RF Model

```
Boosting of Bagging SVM - Evaluation Metrics:
Hamming Loss: 0.06362030988910831
Exact Match Accuracy: 0.537293027495063
Classification Report:
```

	precision	recall	f1-score	support
Shelter	0.08	0.61	0.15	31
Health	0.76	0.88	0.82	1638
NFIs	0.06	0.70	0.12	67
Electricity assistance	0.22	0.76	0.35	45
Food, nutrition	0.84	0.91	0.87	2862
Agricultural supplies	0.06	0.57	0.10	47
Livelihood support	0.19	0.79	0.31	125
Education	0.30	0.85	0.44	174
WASH	0.52	0.84	0.64	1018
Winterisation	0.19	0.84	0.32	129
Legal services	0.85	0.90	0.87	81
GBV services	0.30	0.68	0.42	31
CP services	0.18	0.73	0.30	60
Explosive hazard risk awareness or removal of explosive contamination	0.21	0.62	0.32	24
Mental health psychological support	0.34	0.88	0.49	40
Cash assistance vouchers or cash in hand	0.41	0.81	0.55	828

Figure 19 MultiTarget - SVM Bayes Optimized Model

We conclude that the different methods of bagging and boosting the models didn't increase significantly the models' performance. So we move on to the interpretability of the models and we choose SVM because it has a better performance metrics.

5. Interpretability of SVM classifier

Based on the results above it makes most sense to work with the SVM classifier which is the most reliable of all models. On this basis we will explore the following aspects of the model:

5.1 ROC and AUC

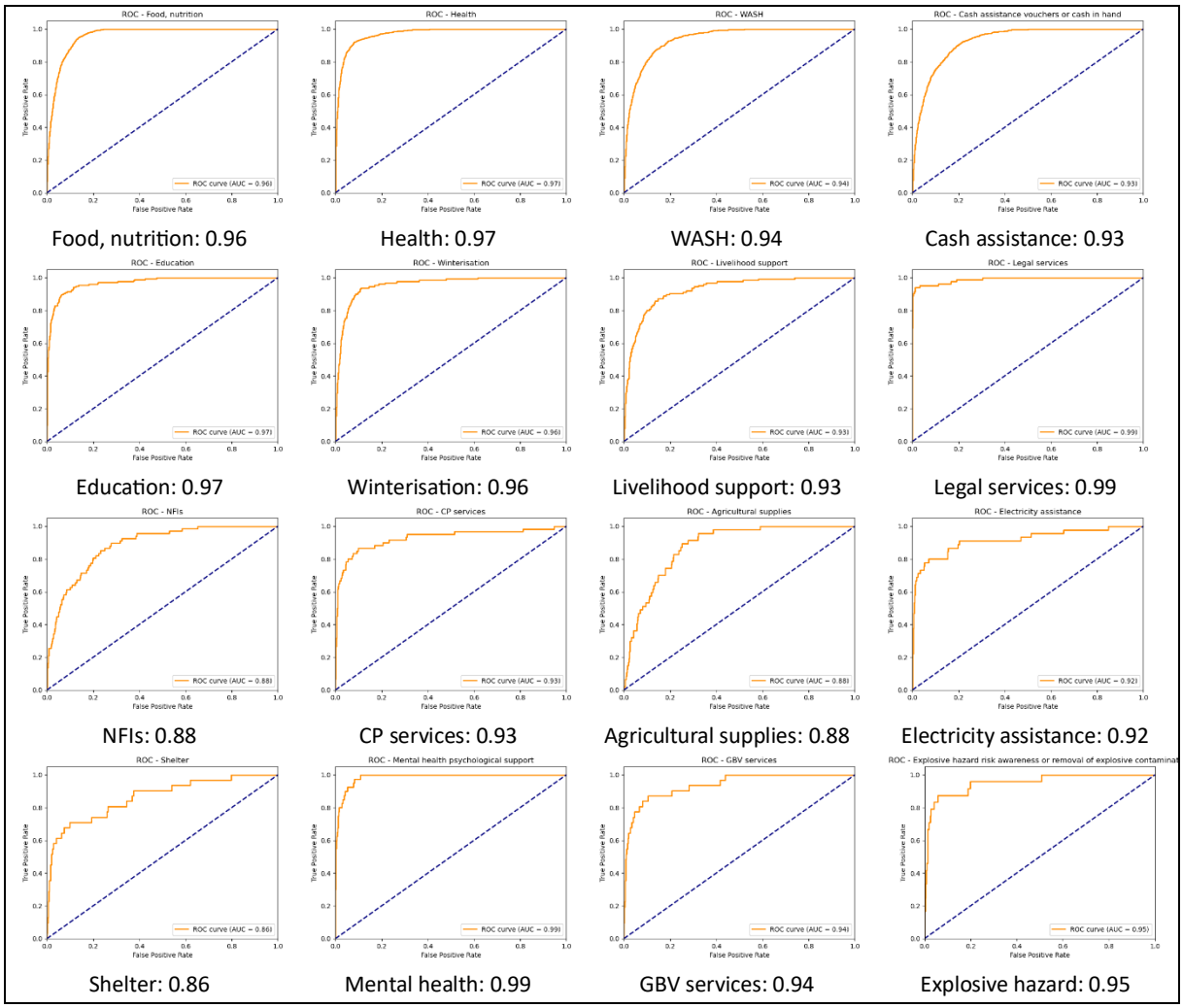

Figure 20 MultiTarget - ROC and AUC

Overall, the ROC curves indicate that all single classifiers are performing quite well. The slight differences in AUC suggest they each do a good job of identifying their respective categories. However still in our case because we are dealing with **critical needs**, missing a true positive (a household that actually needs food assistance) is very costly. So for our problem a high recall is crucial to ensure those who need help do not get overlooked. Therefore we will overlook the ROC values.

5.2 Feature importance and decision function

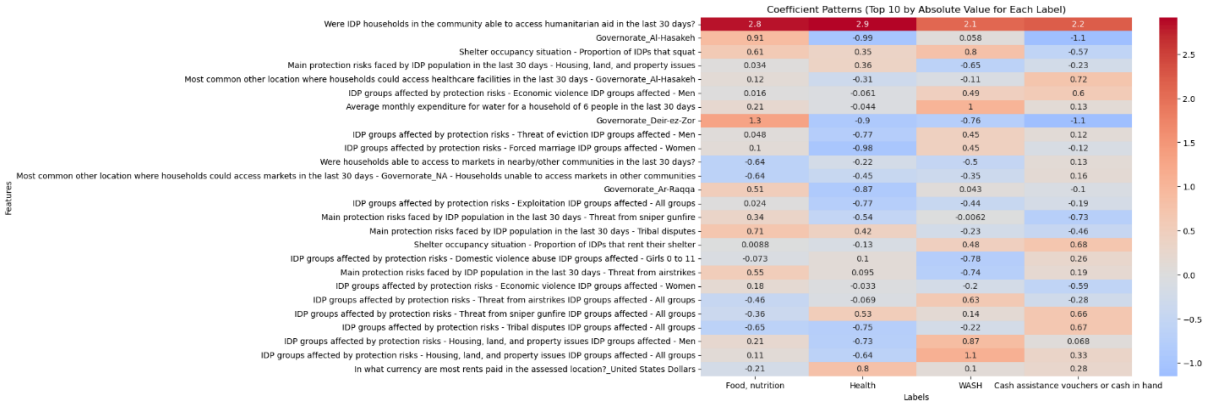

Figure 21 MultiTarget - SVM Feature Importance

The above matrix looks at the most important (recurrent) 4 targets and shows the correlation with the most influential 20 features. Which shows that there is a large amount of common features that influence both.

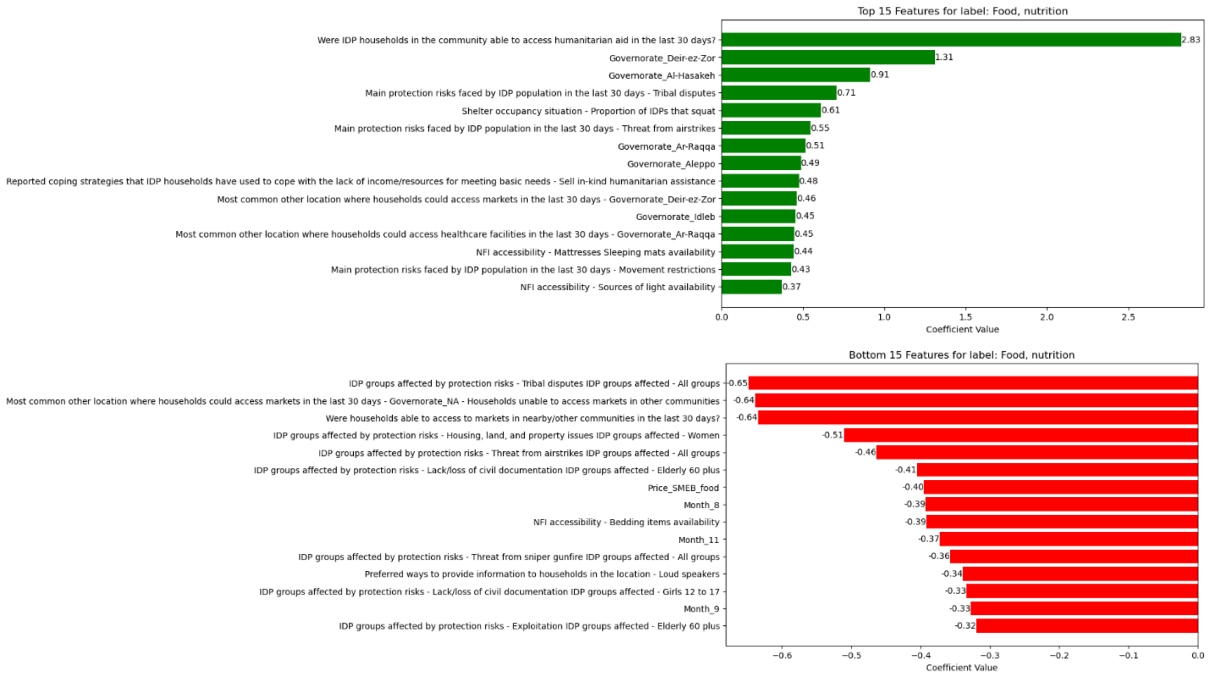

Figure 22 MultiTarget – SVM Model Decision

The barplots above show the magnitude of features importance on the models decision to classify Food, nutrition. By far if the community is able to access aid, the model pushes toward a 1 decision, by a magnitude of 1.31, and 0.91 the community being in Deir ez-Zor or Hasakeh governorates respectively is a key factor.

Similarly, factors that influence the model to classify food and nutrition negatively are protection risks, and households being not able to access markets in other communities

The boxplot calculates the decision function values for each of the 4 most important labels and illustrates how confident the model is (distance from the decision boundary) across each label.

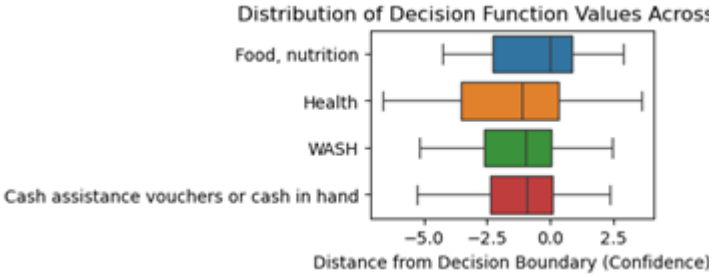

Figure 23 MultiTarget – SVM Boxplot Decision Function

5.3. Correlation matrix of top 20 influencing features on each target

We will explore here the correlation matrix that is relevant for the Food, nutrition. It shows that the community's ability to access aid is correlated 0.73 with receiving Food, nutrition. It is also correlated 0.65 with the community awareness of feedback and complaints mechanisms.

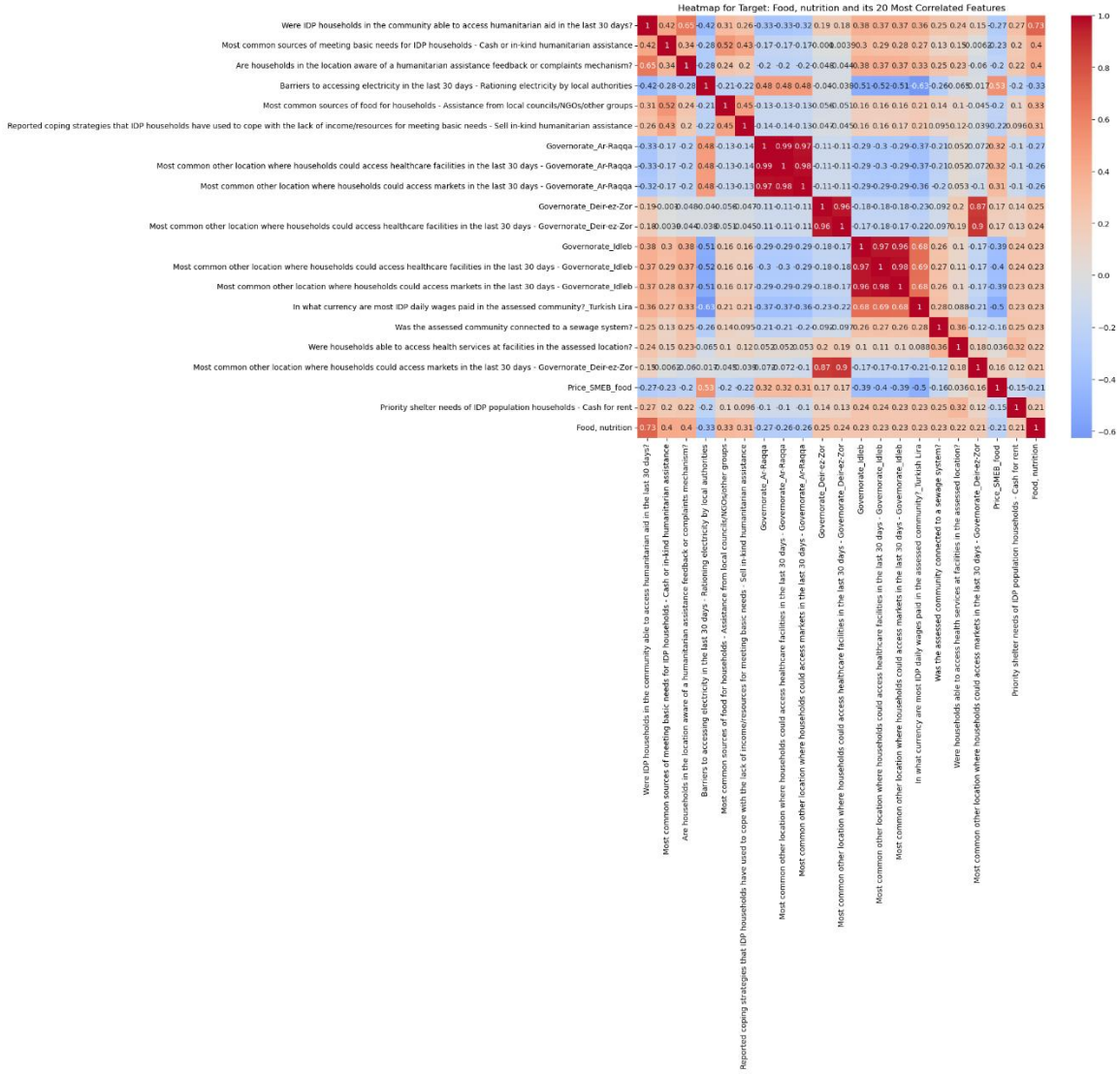

Figure 24 MultiTarget – SVM Heatmap Targets and Features

5.4. SHAP analysis of features importance

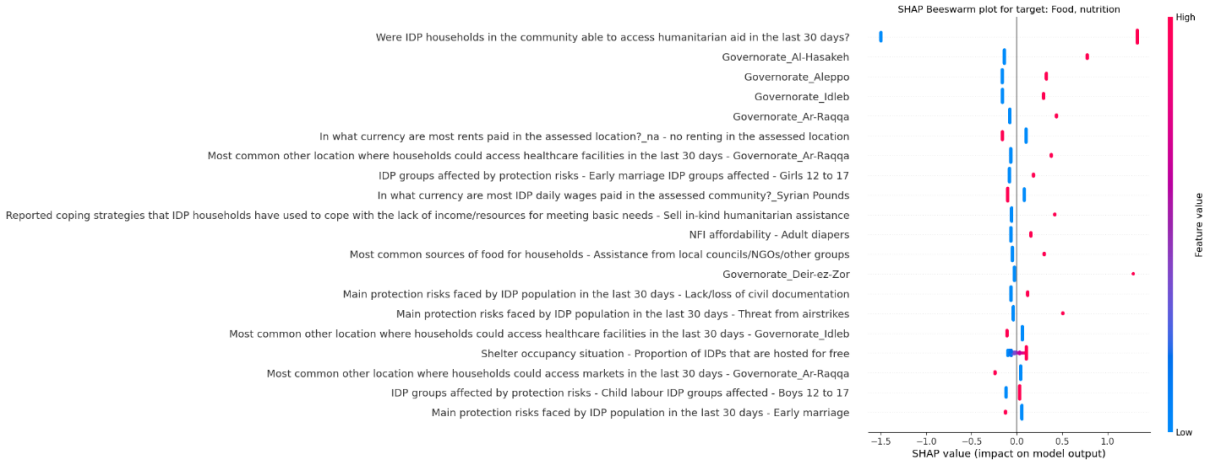

Figure 25 MultiTarget - SVM SHAP Analysis

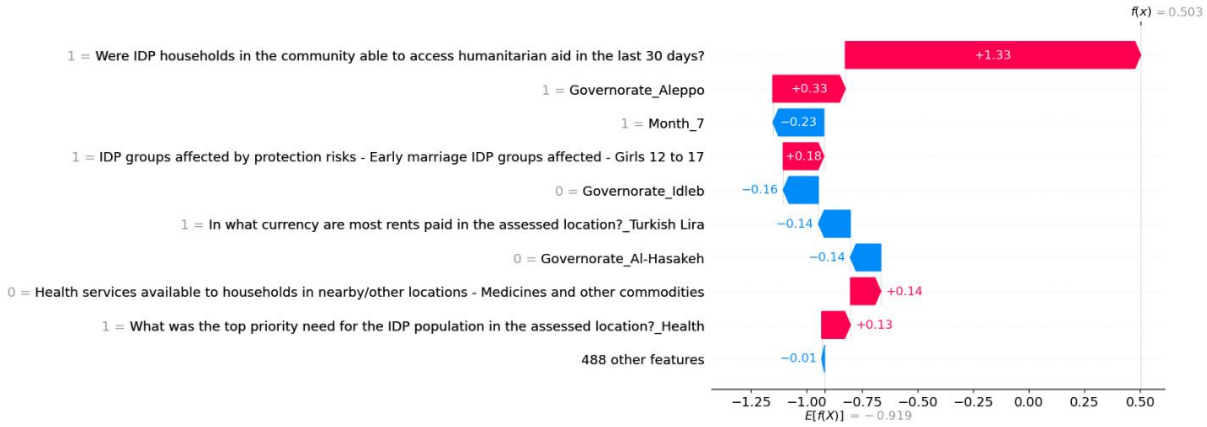

Figure 26 MultiTarget - SVM Beeswarm SHAP

The beeswarm visual confirms the results even though it uses another technique which is game theory. We can see how the community ability to accessing aid is the major factor positively and negatively, and also how the location is a main factor.

5.5. Learning curve

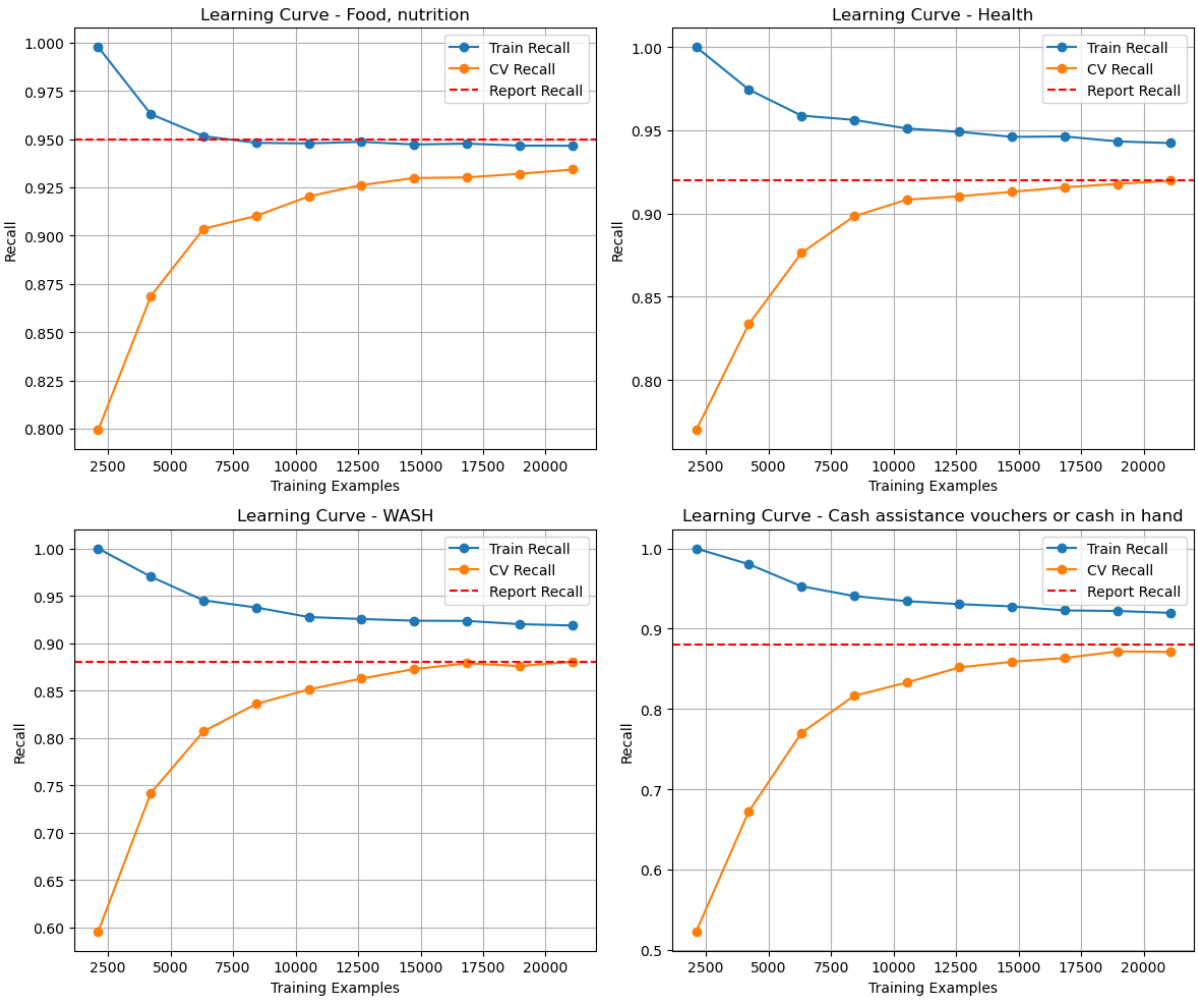

Figure 27 MultiTarget - SVM 4 Target Learning Curve

The learning curves above are plotted for the 4 targets separately. Each of them show that actually more data would not be more useful as the two lines (train and cross validation metrics) converge. We meant to plot the recall rather than the accuracy, because recall indicates the models ability to predict true positives which is most important in our case for the importance of correctly predicting which communities are most in need for what aid assistance without missing communities in need. The recall constant also indicates that the model is not overfitting as it is located in between the converging lines.

6. Multi Label Deep Learning Models

This section presents the deep learning models applied to the multi-label classification problem. The models were selected based on their ability to handle tabular data, capture complex feature interactions, and generalize well.

6.1. Multi-Layer Perceptron (MLP)

MLP is a universal approximator and can model complex, non-linear relationships.

Works well with structured/tabular data with sufficient hidden layers and regularization.

Baseline deep learning model for comparison with other architectures.

Results

Model	Hamming Loss ↓	F1-Score (Macro) ↑	F1-Score (Micro) ↑	Accuracy ↑
MLP	0.02508	0.47954	0.80345	0.73583

Table 19 MultiTarget - Multi-Layer Perceptron (MLP)

6.2. Optimized MLP (Hyperparameter Tuning)

The initial MLP model was improved through hyperparameter tuning using Keras Tuner.

The best architecture was found with 384 and 128 neurons, L2 regularization, and a learning rate of 0.0005.

Results

Model	Hamming Loss ↓	F1-Score (Macro) ↑	F1-Score (Micro) ↑	Accuracy ↑
Best MLP	0.02397	0.48027	0.81587	0.73963

Table 20 MultiTarget - Optimized MLP

Key Observations

Tuning improved all performance metrics, especially Micro F1-Score and Accuracy.

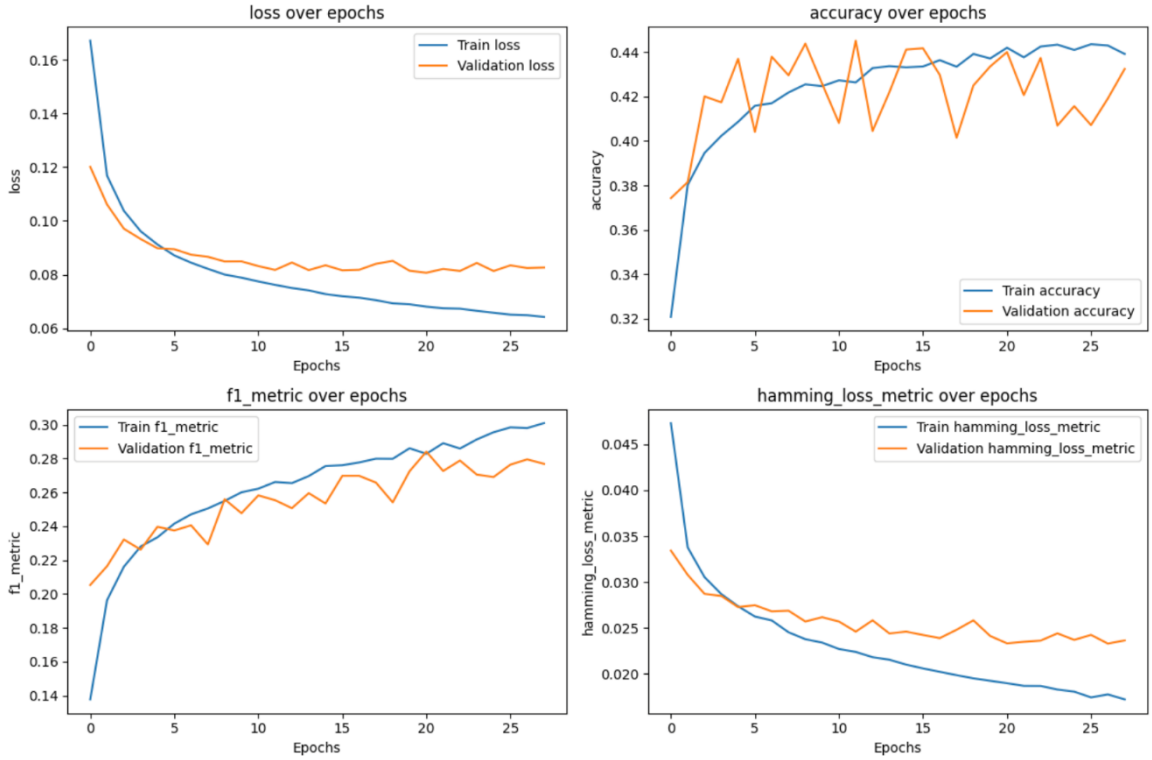

Figure 28 MultiTarget - Optimized MLP Metrics

The Best MLP model demonstrates a well-optimized training process, with steady improvements across loss, accuracy, F1-score, and Hamming loss. The loss curves indicate a smooth decline in both training and validation loss, suggesting that the model is learning effectively without signs of overfitting. The accuracy graph shows rapid improvements in the early epochs, stabilizing around 0.44, with validation accuracy exhibiting some fluctuations, which may indicate sensitivity to batch variations. The F1-score trends suggest consistent performance gains, with validation F1-score closely tracking training performance, further confirming generalization. Finally, the Hamming

loss reduction over epochs shows improved multi-label classification performance. Overall, the model effectively balances learning and regularization, benefiting from the tuned hyperparameters.

6.3 TabNet Classifier

The attention-based feature selection technique has been demonstrated to enhance interpretability. This approach has proven to be effective for tabular data by emulating decision tree-based models through the utilization of deep learning methodologies. Furthermore, the incorporation of sparse representation has been shown to mitigate the occurrence of overfitting. The TabNetMultiTaskClassifier was used.

Model	Hamming Loss ↓	F1-Score (Macro) ↑	F1-Score (Micro) ↑	Accuracy ↑
TabNet	0.02856	0.32815	0.77748	0.71669

Table 21 MultiTarget - TabNet Classifier

The performance of TabNet is lower than that of the MLP model, particularly with regard to the F1-Score (Macro). This finding suggests that TabNet may not fully capture multi-label dependencies. However, TabNet demonstrates comparable accuracy to the Autoencoder + MLP model.

6.4 Autoencoder + MLP

Reduces the high-dimensional input space to a latent representation of 100 features.

Helps denoise and extract hidden patterns before classification.

Results

Model	Hamming Loss ↓	F1-Score (Macro) ↑	F1-Score (Micro) ↑	Accuracy ↑
Autoencoder + MLP	0.02759	0.43861	0.77921	0.71669

Table 22 MultiTarget - Autoencoder + MLP

Key Observations:

Slightly worse than the Best MLP, but performs similarly to TabNet. Dimensionality reduction improved generalization but led to some information loss.

6.5 Convolutional Neural Network (CNN)

CNNs, typically used in image processing, have been adapted for tabular data by treating features as a 1D spatial sequence. This allows the network to capture local feature interactions, potentially improving multi-label classification performance. However, NNs can work for multi-label classification, but they are typically better suited for spatial or sequential data rather than purely tabular data. Since our features are mostly categorical and binary, a fully connected MLP or TabNet is more appropriate.

Key Features:

- Feature extraction through convolutional layers to capture hidden patterns in tabular data.
- Batch normalization to stabilize training and prevent overfitting.
- Dropout regularization to enhance generalization.
- Lower learning rate tuning (0.0005) to improve convergence.

Results:

Model	Test Accuracy	Test Loss
Initial CNN	62.43%	0.1994
Optimized CNN	64.64%	0.1702

Table 23 MultiTarget - CNN Results

Key Observations:

- The optimized CNN model improved test accuracy by 2.2%, showing an ability to learn structured data patterns.
- Loss reduction (from 0.1994 to 0.1702) suggests that deeper convolutional layers and fine-tuned hyperparameters led to better generalization.
- CNN performed competitively with other deep learning models but did not surpass the optimized MLP.
- Validation loss fluctuated slightly, indicating some sensitivity to batch variations.
- Despite performance gains, XGBoost remains the best model for structured tabular data due to its tree-based boosting mechanism.

6.6 Comparison with XGBoost

XGBoost is a tree-based boosting model that typically outperforms deep learning on **structured/tabular** data.

Provides feature importance and efficient training with regularization.

Model	Hamming Loss ↓	F1-Score (Macro) ↑	F1-Score (Micro) ↑	Accuracy ↑
XGBoost (Per-Class Thresholds)	0.01954	0.6357	0.85404	0.77928
Best MLP (Tuned)	0.02397	0.48027	0.81587	0.73963
TabNet	0.02856	0.32815	0.77748	0.71669
Autoencoder + MLP	0.02759	0.43861	0.77921	0.71669

Table 24 MultiTarget - Deep Learning & XGBoost

Conclusion:

XGBoost consistently outperforms all deep learning models, demonstrating its superiority in handling structured tabular data. Among the deep learning approaches, the tuned MLP model achieves the best performance, but it still lags behind XGBoost in key metrics such as F1-score and accuracy. Despite its attention-based feature selection mechanism, TabNet struggles to effectively capture the multi-label dependencies, leading to suboptimal results. Similarly, the Autoencoder + MLP approach, while beneficial for dimensionality reduction, does not provide a significant accuracy boost and slightly underperforms compared to the standard MLP model. Overall, while deep learning models can be competitive, their performance in this case remains inferior to XGBoost, which is better suited for structured data due to its boosting mechanism and ability to handle complex feature interactions efficiently.

7. References

Dong, H., Sun, J., & Sun, X. (2021). A Multi-Objective Multi-Label Feature Selection Algorithm Based on Shapley Value. *Entropy*, *23*(8), Article 8. https://doi.org/10.3390/e23081094

scikit-learn.org. (2025a). *3.4.4.9 Precision, recall and F-measures*. Scikit-Learn. https://scikit-learn/stable/modules/model_evaluation.html

scikit-learn.org. (2025b). *SGDClassifier*. Scikit-Learn. https://scikit-learn/stable/modules/generated/sklearn.linear_model.SGDClassifier.html

Sukhwani, N. (2020, Juni 16). Handling Data Imbalance in Multi-label Classification (MLSMOTE). *TheCyPhy*. https://medium.com/thecyphy/handling-data-imbalance-in-multi-label-classification-mlsmote-531155416b87