SSH Shell Attacks - Appendix

ANDREA BOTTICELLA*, Politecnico di Torino, Italy
ELIA INNOCENTI*, Politecnico di Torino, Italy
RENATO MIGNONE*, Politecnico di Torino, Italy
SIMONE ROMANO*, Politecnico di Torino, Italy

Contents

A DATA EXPLORATION AND PRE-PROCESSING

B SUPERVISED LEARNING - CLASSIFICATION

B.1 Logistic Regression

In this section, we detail the steps and results obtained from using the Logistic Regression model during the supervised learning phase of the project. Logistic Regression served as a baseline model to provide an initial understanding of the classification problem. Despite its simplicity, it offered valuable insights into the multi-label classification task.

B.1.1 Model Training

The Logistic Regression model was trained using its default configuration. Specifically, the `lbfgs` solver was utilized with a regularization parameter $C = 1$. The training process aimed to identify potential overfitting or underfitting issues and establish baseline performance metrics.

```
# Initialize and train Logistic Regression model
model = LogisticRegression(max_iter=1000, random_state=42)
model.fit(X_train_tfidf, y_train_binary)
```

Listing 1. Train Logistic Regression model

The dataset was preprocessed using the TF-IDF representation of the session texts, which assigned weights to words based on their frequency and relevance within the dataset. Multi-label binary encoding was applied to the ‘Set_Fingerprint‘ column to ensure compatibility with the model.

*The authors collaborated closely in developing this project.

Authors’ Contact Information: Andrea Botticella, andrea.botticella@studenti.polito.it, Politecnico di Torino, Turin, Italy; Elia Innocenti, elia.innocenti@studenti.polito.it, Politecnico di Torino, Turin, Italy; Renato Mignone, renato.mignone@studenti.polito.it, Politecnico di Torino, Turin, Italy; Simone Romano, simone.romano2@studenti.polito.it, Politecnico di Torino, Turin, Italy.

Publication date: January 2025.

B.1.2 Evaluation Metrics

The Logistic Regression model was evaluated using standard classification metrics, including weighted F1-scores, precision, and recall. The evaluation metrics highlighted the strengths and weaknesses of the model in handling imbalanced classes.

```
# Generate classification report
from sklearn.metrics import classification_report
report = classification_report(y_test_binary, y_pred, zero_division=0)
print(report)
```

Listing 2. Generate classification report

The confusion matrix provided a breakdown of true positives, false positives, false negatives, and true negatives for each intent. Figure 1 shows the confusion matrix for the Logistic Regression model.

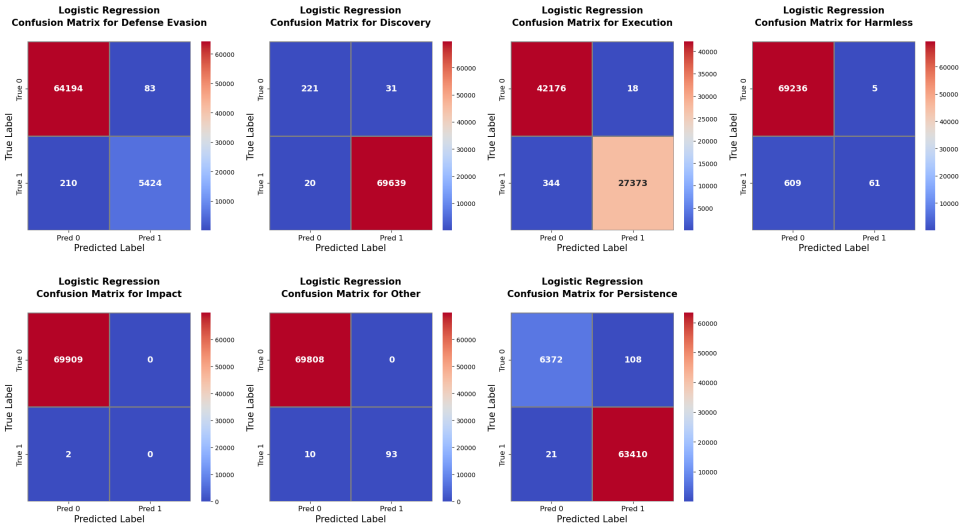

Fig. 1. Confusion Matrix for Logistic Regression Model.

B.1.3 Hyperparameter Tuning

Grid search was performed to optimize the Logistic Regression model's hyperparameters. The search focused on varying the regularization parameter C over a range of values $[0.1, 1, 10, 100]$ to identify the configuration that maximized weighted F1-scores.

```
# Define parameter grid for Logistic Regression
param_grid = {'C': [0.1, 1, 10, 100]}
grid_search = GridSearchCV(LogisticRegression(max_iter=1000,
    random_state=42), param_grid, cv=5)
grid_search.fit(X_train_tfidf, y_train_binary)
```

Listing 3. Parameter grid for Logistic Regression

The optimized model exhibited improved performance compared to the baseline, particularly for intents with smaller sample sizes. Figure 2 illustrates the weighted F1-scores for different values of C.

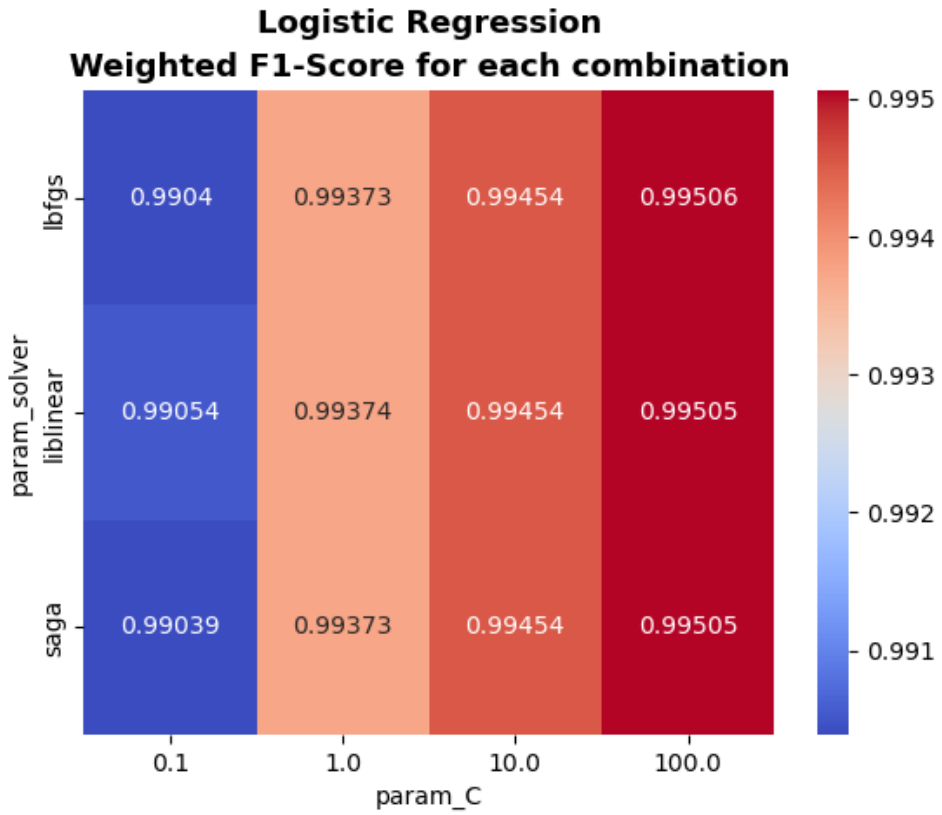

Fig. 2. Weighted F1-Scores for Logistic Regression Hyperparameter Tuning.

B.1.4 Comparative Analysis of Baseline and Optimized Models

The optimized Logistic Regression model demonstrated a moderate improvement in precision and recall compared to the baseline. However, its overall performance remained slightly inferior to more complex models like Random Forest and SVM. The comparative analysis underscores the importance of selecting models suited to the dataset's characteristics and problem requirements.

C UNSUPERVISED LEARNING - CLUSTERING

D LANGUAGE MODEL EXPLORATION