Assignment 3

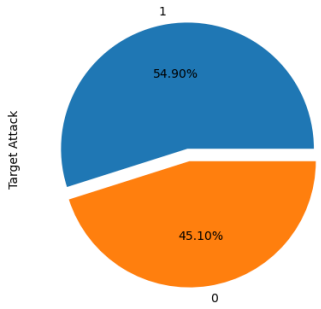
Part 1: Static model

Data Analysis

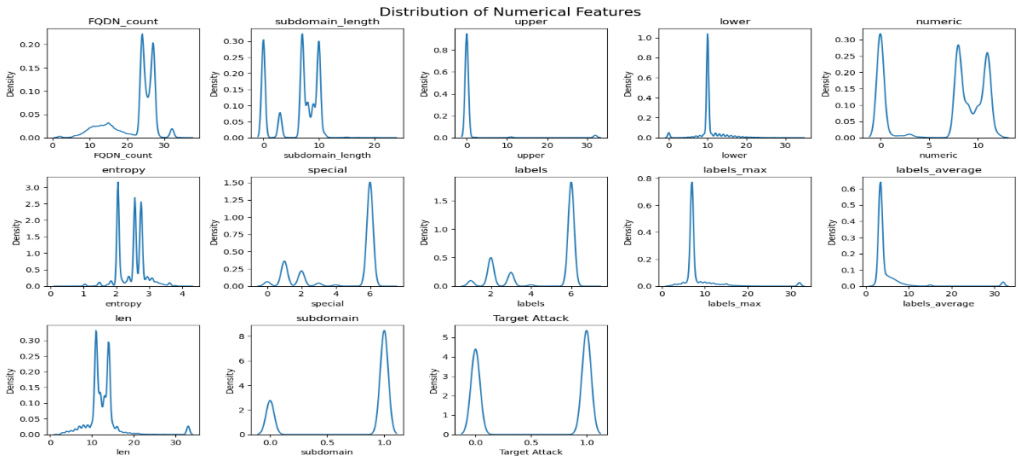
To initiate the data analysis process, the first step involves presenting the initial insights into the dataset by displaying the first five rows. This initial snapshot provides a glimpse into the structure and content of the dataset, allowing for a preliminary understanding of the variables and their respective values.

	timestamp	FQDN_count	subdomain_length	upper	lower	numeric	entropy	special	labels	labels_max	labels_average	longest_word	sld	len	subdomain	Target Attack
0	56:19.8	27	10	0	10	11	2.570417	6	6	7	3.666667	2	192	14	1	1
1	07:23.9	27	10	0	10	11	2.767195	6	6	7	3.666667	2	192	14	1	1
2	23:15.1	26	9	0	10	10	2.742338	6	6	7	3.500000	2	192	13	1	0
3	04:51.9	27	10	0	10	11	2.570417	6	6	7	3.666667	2	192	14	1	1
4	12:44.0	15	9	0	11	0	2.929439	4	3	5	4.333333	local	local	15	1	1

Following this, the focus shifts towards exploring the distribution of the target variable, 'Target Attack.' Utilizing a pie plot for visualization, this step aims to assess the balance or potential bias within the classes of the target variable.



Subsequently, the analysis extends to the examination of the distribution of numerical features across all columns in the dataset. This step is crucial for uncovering any imbalances or variations within the numerical attributes. Visualizing the distribution of these features provides insights into the spread and central tendencies of the data, aiding in the identification of potential outliers or skewed distributions.



Feature engineering and data cleaning

The initial exploration of the dataset involves a meticulous examination of its quality through the display of data information. The Data Frame, comprising 268,074 entries and 16 columns, is presented with details on non-null counts and data types for each attribute. This step is crucial for identifying potential data anomalies, ensuring that each column contains the expected data types and that there are no missing values. Subsequently, null values are dropped, resulting in a refined dataset of 268,066 entries, which is essential for maintaining data integrity and reliability in subsequent analyses.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 268074 entries, 0 to 268073
Data columns (total 16 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   timestamp         268074 non-null  object
 1   FQDN_count        268074 non-null  int64
 2   subdomain_length  268074 non-null  int64
 3   upper             268074 non-null  int64
 4   lower             268074 non-null  int64
 5   numeric           268074 non-null  int64
 6   entropy           268074 non-null  float64
 7   special           268074 non-null  int64
 8   labels            268074 non-null  int64
 9   labels_max        268074 non-null  int64
 10  labels_average    268074 non-null  float64
 11  longest_word      268066 non-null  object
 12  sld               268074 non-null  object
 13  len               268074 non-null  int64
 14  subdomain         268074 non-null  int64
 15  Target Attack     268074 non-null  int64
dtypes: float64(2), int64(11), object(3)
memory usage: 32.7+ MB
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 268066 entries, 0 to 268073
Data columns (total 16 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   timestamp         268066 non-null  object
 1   FQDN_count        268066 non-null  int64
 2   subdomain_length  268066 non-null  int64
 3   upper             268066 non-null  int64
 4   lower             268066 non-null  int64
 5   numeric           268066 non-null  int64
 6   entropy           268066 non-null  float64
 7   special           268066 non-null  int64
 8   labels            268066 non-null  int64
 9   labels_max        268066 non-null  int64
 10  labels_average    268066 non-null  float64
 11  longest_word      268066 non-null  object
 12  sld               268066 non-null  object
 13  len               268066 non-null  int64
 14  subdomain         268066 non-null  int64
 15  Target Attack     268066 non-null  int64
dtypes: float64(2), int64(11), object(3)
memory usage: 34.8+ MB
```

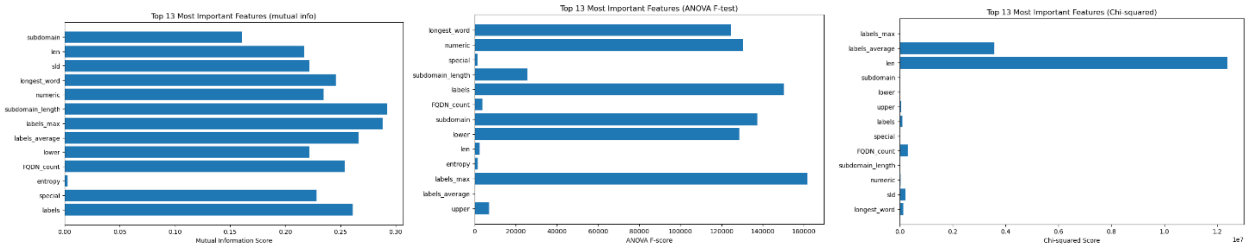
The refinement process extends to the transformation of specific columns. The mode of the "sld" and "longest_word" columns is calculated, and the result is stored as an integer. This calculated mode serves as a representative value, providing a numerical summary for these columns. The subsequent step involves replacing all words within these columns with the calculated mode value.

With the dataset now preprocessed and refined, the focus shifts to feature selection. This involves leveraging various techniques, including information gain, ANOVA, and chi-square tests, to identify the most influential features in predicting the target variable, "Target Attack." Each feature's influence is then visually represented through plotting, offering a comprehensive view of their impact.

In the context of information gain analysis, a comprehensive examination reveals that all features demonstrate substantial influence on the target variable, with "subdomain_length" exhibiting the highest impact and "entropy" displaying the lowest. This suggests that the length of the subdomain is a particularly informative factor in predicting the target variable, while the entropy of the data contributes less significantly.

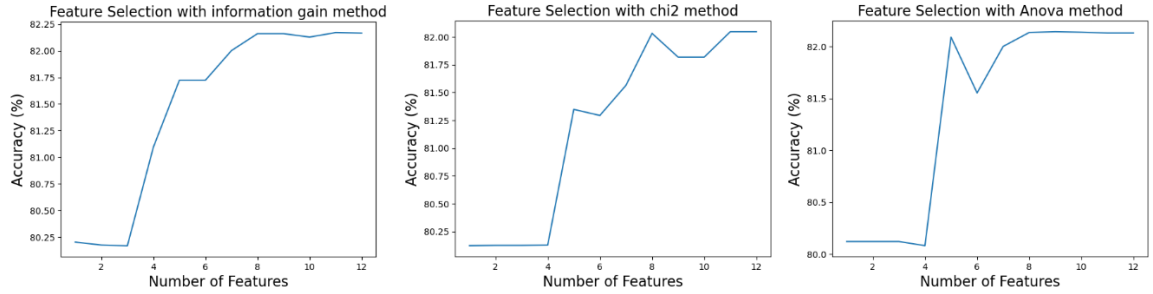
Moving on to ANOVA (Analysis of Variance), the analysis highlights that only six features wield considerable influence in predicting the target variable.

In the case of the chi-square test, the analysis identifies only two features that exert a substantial influence on the target variable. This selective revelation underscores the importance of these specific features in the context of categorical target variables, providing a targeted perspective for feature selection.



Logistic Regression model

To delve deeper into the impact of feature selection methods on model performance, a systematic exploration is conducted by plotting accuracy against the number of selected features for each method.

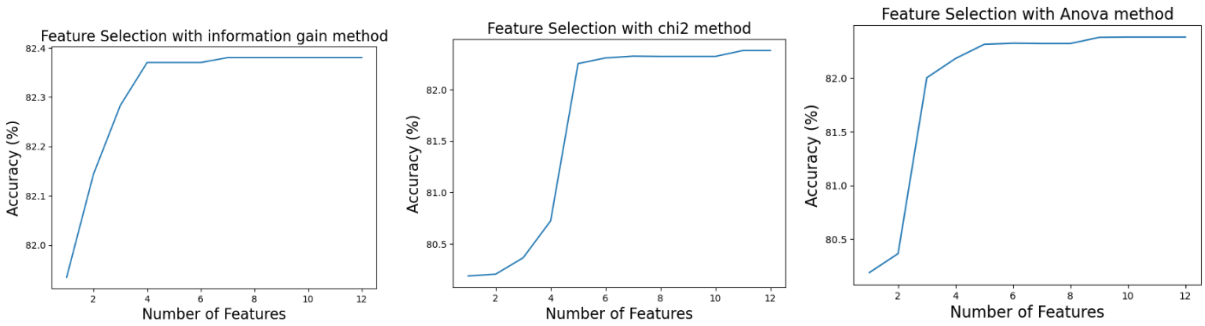


The logistic regression model is then trained using the dataset containing only the selected features, ensuring that the model is tailored to the specific subset identified by each feature selection method.

```
Mutual Info Accuracy: 0.827555334493907
Chi-squared Accuracy: 0.8258890823178314
ANOVA Accuracy: 0.8270828152200945
```

Random Forest Classifier model

To delve deeper into the impact of feature selection methods on model performance, a systematic exploration is conducted by plotting accuracy against the number of selected features for each method.

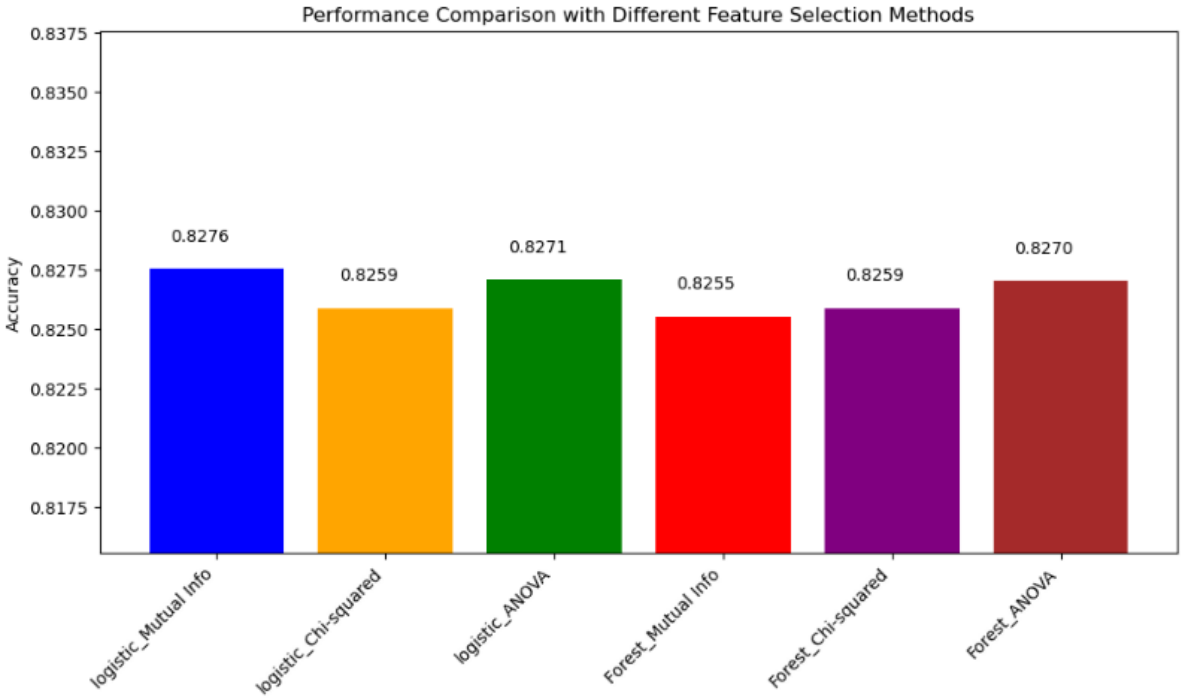


The random forest model is then trained using the dataset containing only the selected features, ensuring that the model is tailored to the specific subset identified by each feature selection method.

```
Mutual Info Accuracy: 0.825540910221338
Chi-squared Accuracy: 0.8258890823178314
ANOVA Accuracy: 0.8270330763491669
```

Model evaluation

The analysis begins with a detailed comparison of two classification models, Logistic Regression (LR) and Random Forest (RF), across different feature selection methods, including Mutual Information, chi-squared (chi2), and ANOVA. The accuracy of each model is plotted for varying numbers of selected features, it is determined that Logistic Regression with Mutual Information feature selection yields the best results, demonstrating superior accuracy compared to other configurations.



Grid Search is conducted to find the best hyperparameters. This systematic search over a predefined parameter grid helps identify the combination of hyperparameters that maximizes the model's performance. Once the best hyperparameters are determined, a pipeline is constructed, encapsulating both the feature selection and the Logistic Regression model with the optimized parameters.

```
Best Parameters: {'classifier__C': 1, 'classifier__penalty': 'l2'}
Best Accuracy Score: 0.8235080962027432
```

the trained and optimized model is saved using the pickle module. Loading the model from the pickle file allows for its seamless integration into dynamic models, enabling efficient and consistent application in real-world scenarios.

Part 2: Dynamic model

Static model

the process begins by loading the model saved using the pickle module in the initial phase. This ensures that the static model is consistent with the one trained and optimized in the previous steps, preserving its state, including the selected features and hyperparameters.

Dynamic model

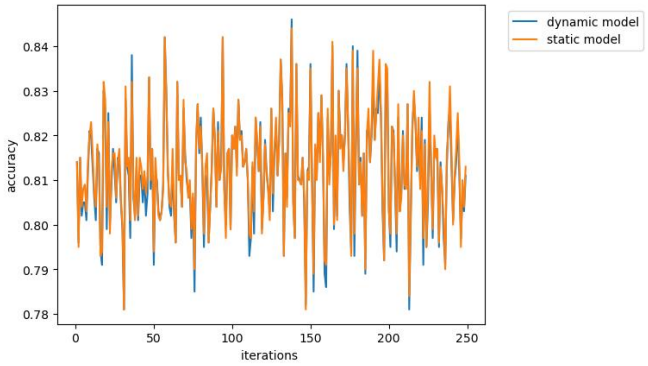
Moving to the dynamic model, the initial configuration mirrors that of the static model, utilizing the saved model from part 1. However, the dynamic model introduces an adaptive element. It assesses the need for retraining by analyzing windows of 1,000 observations. If the accuracy of the dynamic model falls below a predefined threshold, specifically 81%, it triggers a retraining process. This adaptive strategy allows the model to continuously evolve and adapt to potential changes in the underlying data distribution.

```
Window 1
The accuracy of Dynamic Model without retrain = 81.39999999999999%
The accuracy of Static Model = 81.39999999999999%
**********
Window 2
The accuracy of Dynamic Model without retrain = 79.5%
 trained model on the new data
The accuracy of Dynamic Model after retrain = 79.60000000000001%
The accuracy of Static Model = 79.5%
**********
Window 3
The accuracy of Dynamic Model without retrain = 81.5%
The accuracy of Static Model = 81.5%
**********
Window 4
The accuracy of Dynamic Model without retrain = 80.4%
 trained model on the new data
The accuracy of Dynamic Model after retrain = 80.2%
The accuracy of Static Model = 80.4%
```

```
**********
Window 247
The accuracy of Dynamic Model without retrain = 80.60000000000001%
 trained model on the new data
The accuracy of Dynamic Model after retrain = 80.60000000000001%
The accuracy of Static Model = 81.0%
**********
Window 248
The accuracy of Dynamic Model without retrain = 80.30000000000001%
 trained model on the new data
The accuracy of Dynamic Model after retrain = 80.30000000000001%
The accuracy of Static Model = 80.4%
**********
Window 249
The accuracy of Dynamic Model without retrain = 81.10000000000001%
The accuracy of Static Model = 81.3%
**********
```

The analysis involves plotting the accuracies of both the static and dynamic models over 250 iterations. This visual representation provides insights into how the models' performances fluctuate over time and whether the adaptive retraining strategy implemented in the dynamic model yields improvements compared to the static model.

The observation that the highest accuracy is attained before iteration number 150 is noteworthy. This insight suggests that, within the analyzed dataset and under the implemented conditions, the models, both static and dynamic, reach their peak performance within the initial iterations.



Advantages: The assignment demonstrates the practical implementation of a dynamic model in real-life scenarios with continuous data streams, showcasing its adaptability to evolving data patterns, a crucial feature for effective cybersecurity.

Limitations: While the dynamic model proves effective, it has limitations, including potential challenges in defining the decision boundary for retraining and opportunities for improvement in performance metrics and the number of iterations.

Knowledge Gained: The assignment provides valuable hands-on experience with real-time data streaming, model deployment, and dynamic machine learning models tailored for cybersecurity. It emphasizes the importance of continuous learning and adaptation, offering insights into both challenges and opportunities in real-world machine learning applications.