Project Name: Malicious URL Detection by Exploring Machine Learning Algorithms

Submitted By: Shutonu Mitra, **Email**: mshutonu@vt.edu

Introduction:

The proliferation of malicious URLs on the internet poses a significant threat to online security. These URLs are used to distribute malware, conduct phishing attacks, and compromise user data. The Malicious URL Detection by Exploring Machine Learning Algorithms is a project designed to develop and deploy an efficient and robust system for detecting and preventing malicious URLs. A brief introduction of the attacks, this project is going to address is given below:

Malware, phishing, and defacement attacks involving URLs are common cyber threats that exploit various vulnerabilities and use deceptive tactics to compromise individuals, organizations, or websites. Malware can be distributed through URLs in various ways. Malicious URLs can be embedded in legitimate websites or advertisements. When a user visits a compromised webpage, the malware is silently downloaded and executed on their device without their knowledge or consent. Phishing is an online social engineering assault that targets digital identity theft by posing as a trustworthy organization. An attack vector is typically sent by the attacker in the form of an email, chat message, blog post, etc. that contains a link (URL) to a malicious website that is used to collect personal data from the victims. Defacement in URL data typically refers to a specific type of cyberattack or compromise where a website's web pages or content are maliciously altered, replaced, or defaced by unauthorized individuals or groups. In the context of URL data analysis or cybersecurity, "defacement" implies that a website's URL or web page has been tampered with in a way that makes it appear different from its original, legitimate state. The Malicious URL Detection project aims to address these critical issues by developing an advanced URL classifier model for identifying and blocking malicious URLs in real-time. The URL classifier can be used to develop and deploy an efficient and robust system for detecting and preventing malicious URLs. Thus, project aims to enhance web security by identifying and blocking URLs that lead to malicious websites causing cyber threats.

Problem Statement:

The core problem tackled in this project is the detection and prevention of malware, phishing, and defacement attacks. Automatic detection of a URL's malicious or benign reputation acts as a defence mechanism, safeguarding users from malicious websites. This project primarily focuses on building a system for URL analysis and classification to identify malware, phishing, and defacement attacks and differentiate them from benign URLs. Stakeholders, including individuals, businesses, governments, and the cybersecurity industry, stand to benefit from effective anti-phishing, anti-malware, and anti-defacement solutions.

This problem lends itself well to a data analytics approach, as it involves analyzing and classifying URLs based on various features and patterns. Machine learning and data analysis techniques are essential for addressing this challenge, as malicious and benign URLs exhibit different patterns and characteristics. These characteristics, such as domain obfuscation, excessive length, or the presence of specific keywords, can be exploited for classification. It's worth noting that the landscape of malicious URLs is continually evolving, as cybercriminals devise new techniques. Therefore, feature extraction, feature engineering, and machine learning are valuable tools for constructing an efficient malicious URL classifier

Dataset Description

The project utilizes the subset of URL Reputation dataset from the UCI Machine Learning Repository. This dataset represents a 120-day anonymized subset of the ICML-09 URL data [1], containing 2.4 million examples and 3.2 million features. It serves as a comprehensive collection of URLs, encompassing both benign and malicious cases, suitable for training and testing the detection system. The dataset consists of 428,103 benign URLs, 96,457 defacement URLs, 94,111 phishing URLs, and 32,520 malware URLs. The distribution of these classes is illustrated in Figure 1.

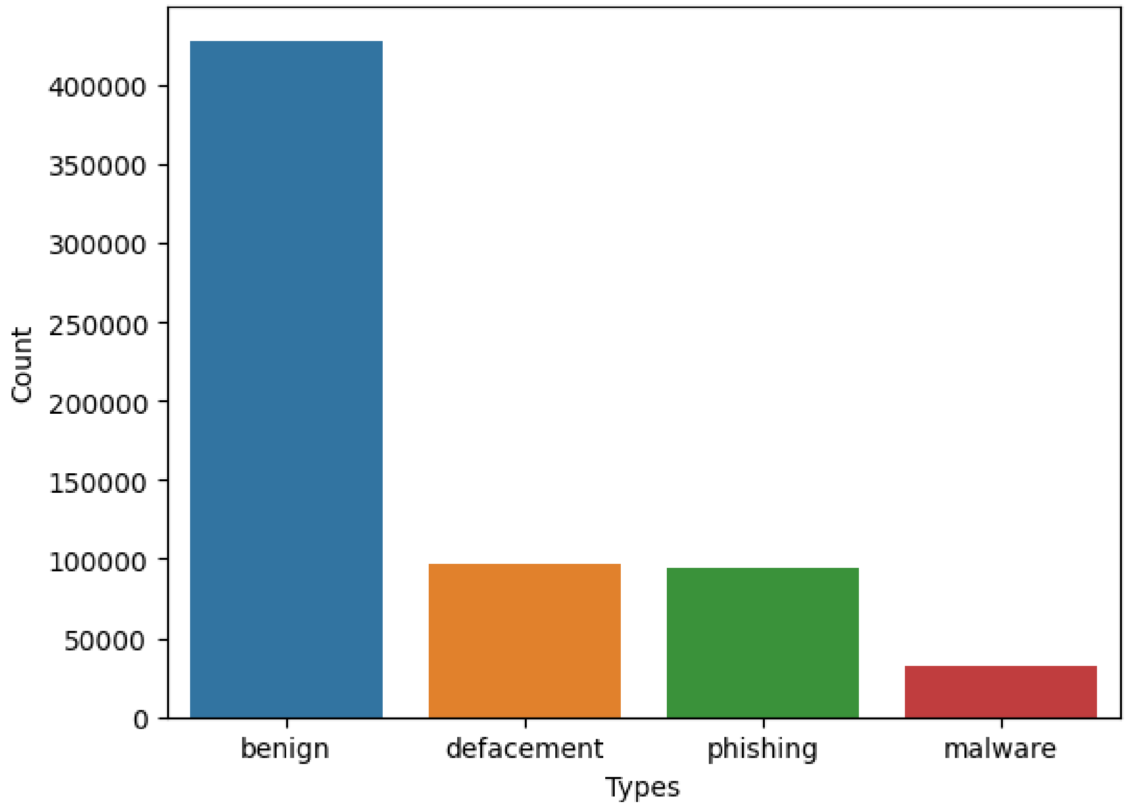


Figure 1: Distribution of the target of the dataset

Methodology

1. **Feature Extraction**: Various features are extracted from the "URL" attribute of the dataset, including URL length, domain, subdomain, top-level domain (TLD), full-length domain (FLD), the entropy of characters, special character counts, and more.

Table 1: Extracted Features of the Dataset

No.	Feature Name	Feature Type	Description
1	URL_len	Integer	Length of the URL
2	domain	Text	The domain of the URL
3	subdomain	Text	The subdomain of the URL
4	tld	Text	The top-level-domain of the URL
5	fld	Text	The full-length domain name of the URL
6	Subdomain_len	Integer	Length of subdomain
7	Domain_len	Integer	Length of domain
8	Tld_len	Integer	Length of TLD
9	fld_len	Integer	Length of FLD
10	url_entropy	Float	The entropy of the characters of the URL
11	domain_entropy	Float	The entropy of the characters of the Domain
12	subdomain_entropy	Float	The entropy of the characters of the Subdomain
13	tld_entropy	Float	The entropy of the characters of the TLD
14	Fld_entropy	Float	The entropy of the characters of the FLD
15	@	Integer	Count of ‘@’ character in the URL
16	?	Integer	Count of ‘?’ character in the URL
17	-	Integer	Count of ‘-’ character in the URL
18	=	Integer	Count of ‘=’ character in the URL
19	.	Integer	Count of ‘.’ character in the URL
20	#	Integer	Count of ‘#’ character in the URL
21	%	Integer	Count of ‘%’ character in the URL
22	+	Integer	Count of ‘+’ character in the URL
23	$	Integer	Count of ‘$’ character in the URL
24	!	Integer	Count of ‘!’ character in the URL
25	*	Integer	Count of ‘*’ character in the URL
26	,	Integer	Count of ‘,’ character in the URL
27	//	Integer	Count of ‘//' characters in the URL
28	Abnormal_url	Boolean	A basic check to identify suspicious URLs based on their structure to see if the URL contains the hostname or not.
29	https	Boolean	If the URL HTTP-Secured or not
30	digits	Integer	Count of digits in the URL
31	letters	Integer	Count of the letters in the URL
32	Shortening_service	Boolean	If the URL contains a popular shortening service or not.

33	Is_ip	Boolean	If the URL contains an IP address or not
34	url_path	Text	Contains path of the URL
35	url_path_len	Integer	The length of the path of the URL
36	count_dir_in_url_path	Integer	The number of sub-directories in the path
37	First_dir_len	Integer	Only the length of the first directory of the URL
38	URL_puncs	Integer	Count of punctuation marks in URL
39	Hostname_len	Integer	Length of Hostname of the URL
40	Has_port_no	Boolean	If the URL contains a port number or not
41	Number_of_params	Integer	Number of parameters in URL specified by ‘&’ character
42	Number_of_fragments	Integer	Number of fragments in URL specified by ‘#’ character
43	Has_client	Boolean	If the URL has a 'client' word in it.
44	Has_admin	Boolean	If the URL has an 'admin' word in it.
45	Has_server	Boolean	If the URL has a 'server' word in it.
46	Has_login	Boolean	If the URL has a 'login' word in it.
47	pc_alphas	Float	Percentage of the letters in the URL
48	pc_digits	Float	Percentage of the digits in the URL
49	pc_puncs	Float	Percentage of the punctuation in the URL.

2. **Data Preprocessing:**

   Excluding Textual Features: The textual features like URL, domain, subdomain, tld, fld, and url_path have been removed as all necessary information has been extracted from them.

   Handling Missing Values: Only 14 instances have missing values in the first_dir_len column as some URLs did not have '/' separating the directories. Since the number of instances is very small in comparison to the dataset size, the missing positions are imputed with 0.

   Encoding the target attribute: Since the target attribute contains the class label names in String, it has been encoded to integer values as: "benign": 0, "defacement": 1, "phishing":2, "malware":3.

   Discretization of numerical features: Since the ‘url_len’ and 'fld_len' have a higher correlation with the target, they have been Discretized into groups such as Short', 'Medium', 'Long', and 'Very Long'. Later, a one-hot encoding technique was applied to the columns.

   Encoding of other categorical columns: All other ‘Boolean’ variables have been encoded by 0 and 1.

   Tokenization for applying CNN: For applying the CNN the original URLs have been tokenized to character level.

3. **Feature Selection:** Two feature selection techniques have been tested on the dataset:

   1. Selecting Features by Correlation with Target: This technique involves evaluating the correlation between each feature and the target variable (output variable). Features with a high correlation to the target are considered important and are selected for the model. By the threshold=0.1, the final selected features by this approach are: ['-', '.','//', 'https', 'Abnormal_URL', 'is_ip', 'count_dir_in_url_path', 'pc_alphas',

'pc_digits', 'pc_puncs', 'has_port_no', 'entropy_of_domain', 'entropy_of_tld', 'entropy_of_fld']

2. Random Forest Feature Importance: Random Forest is an ensemble learning method that builds multiple decision trees and merges them together to get a more accurate and stable prediction. Feature importance is computed based on the contribution of each feature to the reduction in impurity (e.g., Gini impurity) across all decision trees. Figure 2. illustrates the feature importance. Final Selected Features with cut off of 0.01

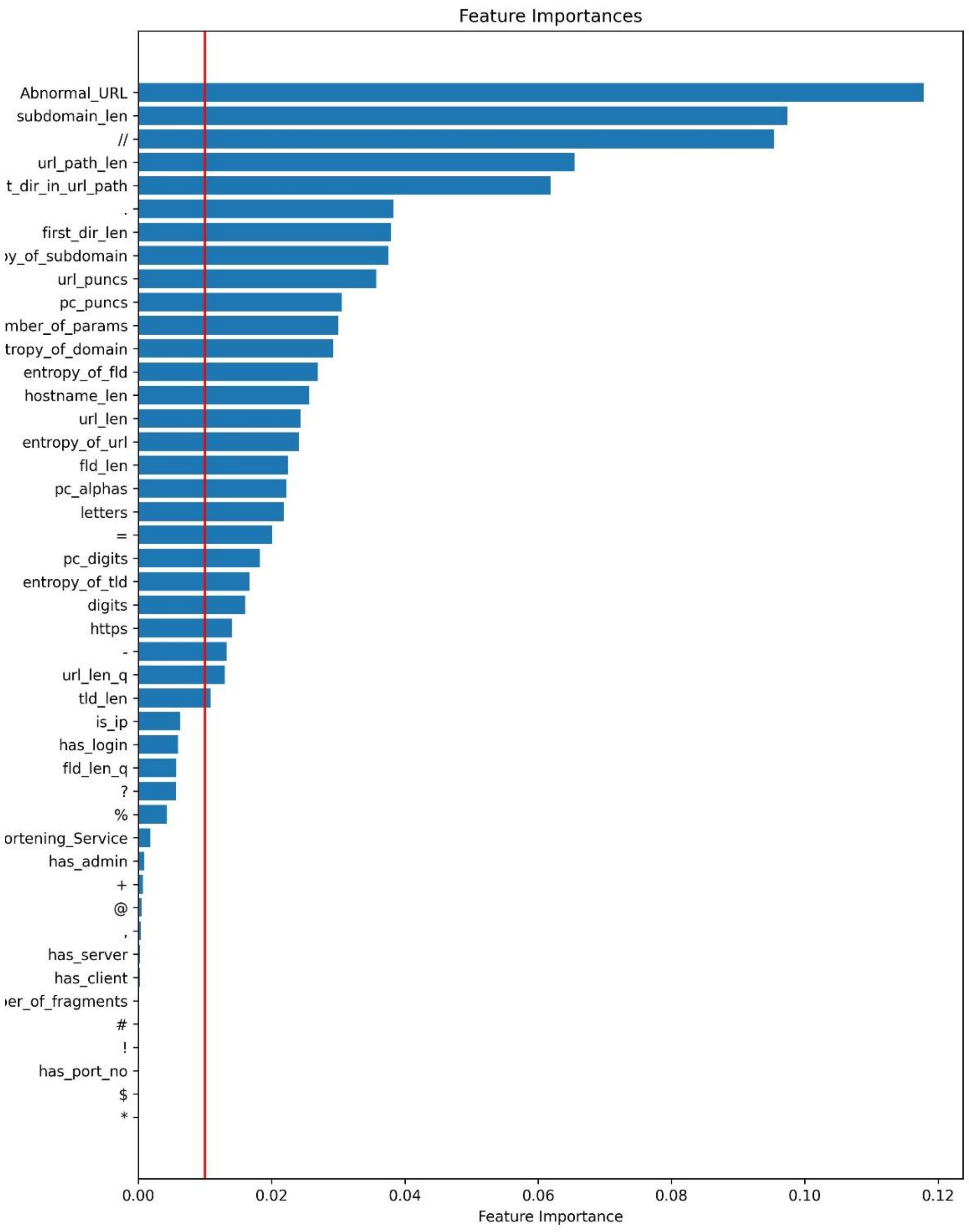


Figure 2: Random Forest Feature Importance

are: 'url_len', '-', '=', '.', '//', 'https', 'digits', 'letters', 'Abnormal_URL', 'url_path_len', 'count_dir_in_url_path', 'first_dir_len', 'subdomain_len', 'tld_len', 'fld_len', 'url_puncs', 'url_len_q', 'pc_alphas', 'pc_digits', 'pc_puncs', 'hostname_len', 'number_of_params', 'entropy_of_url', 'entropy_of_subdomain', 'entropy_of_domain', 'entropy_of_tld' and 'entropy_of_fld'.

4. **Exploring Machine Learning Algorithms:** By splitting the data into 80/20 train/test splits the Machine Learning Algorithms that have been applied are DecisionTree Classifier, RandomForest Classifier, AdaBoost Classifier, KNeighbors Classifier, SGD Classifier, ExtraTrees Classifier, GaussianNB , Support Vector Classifier and 1D Convolutional Neural Network. Here's a brief description of the classifiers used in the context of malicious URL detection:

- **Decision Tree Classifier:** Decision trees make decisions by splitting the data based on features and creating a tree-like structure of decisions. Decision trees can capture complex decision boundaries, making them suitable for datasets with intricate patterns. They can handle both numerical and categorical features, making them versatile for URL analysis.
- **Random Forest Classifier:** Random Forest is an ensemble learning method that constructs a multitude of decision trees and merges their outputs to improve generalization and robustness. Random Forests are effective in handling high-dimensional data and are less prone to overfitting. They can provide feature importance, helping to identify which features contribute the most to classification.
- **AdaBoost Classifier:** AdaBoost is an ensemble learning technique that combines multiple weak classifiers to create a strong classifier. It is useful when there are weak learners in the dataset. It can adapt well to complex datasets and is less likely to overfit. It may be employed to improve the performance of base classifiers.
- **K-Nearest Neighbors (KNN) Classifier:** KNN classifies a data point based on the majority class of its k-nearest neighbors in the feature space. KNN is simple and intuitive. It can capture local patterns in the data. However, it might struggle with high-dimensional data and can be sensitive to irrelevant features.
- **Stochastic Gradient Descent (SGD) Classifier:** SGD is an optimization algorithm that is used in conjunction with linear classifiers, such as Support Vector Machines and logistic regression. SGD classifiers are efficient for large-scale datasets and high-dimensional feature spaces. They can handle sparse data and are suitable for online learning.
- **Extra Trees Classifier:** Extra Trees, or Extremely Randomized Trees, is an ensemble learning method similar to Random Forests but introduces additional randomness in the feature selection process. Extra Trees can be robust to noisy data and are computationally efficient. They are suitable for handling high-dimensional datasets.
- **Gaussian Naive Bayes Classifier:** Naive Bayes is based on Bayes' theorem and assumes independence between features. Naive Bayes classifiers are simple and computationally efficient. They can work well with high-dimensional data but might struggle if the independence assumption is violated.
- **Support Vector Classifier:** Support Vector Classification (SVC) is a type of support vector machine (SVM) that is used for classification tasks. SVMs, including SVC, are popular

machine learning models that excel in both linear and non-linear classification. They work

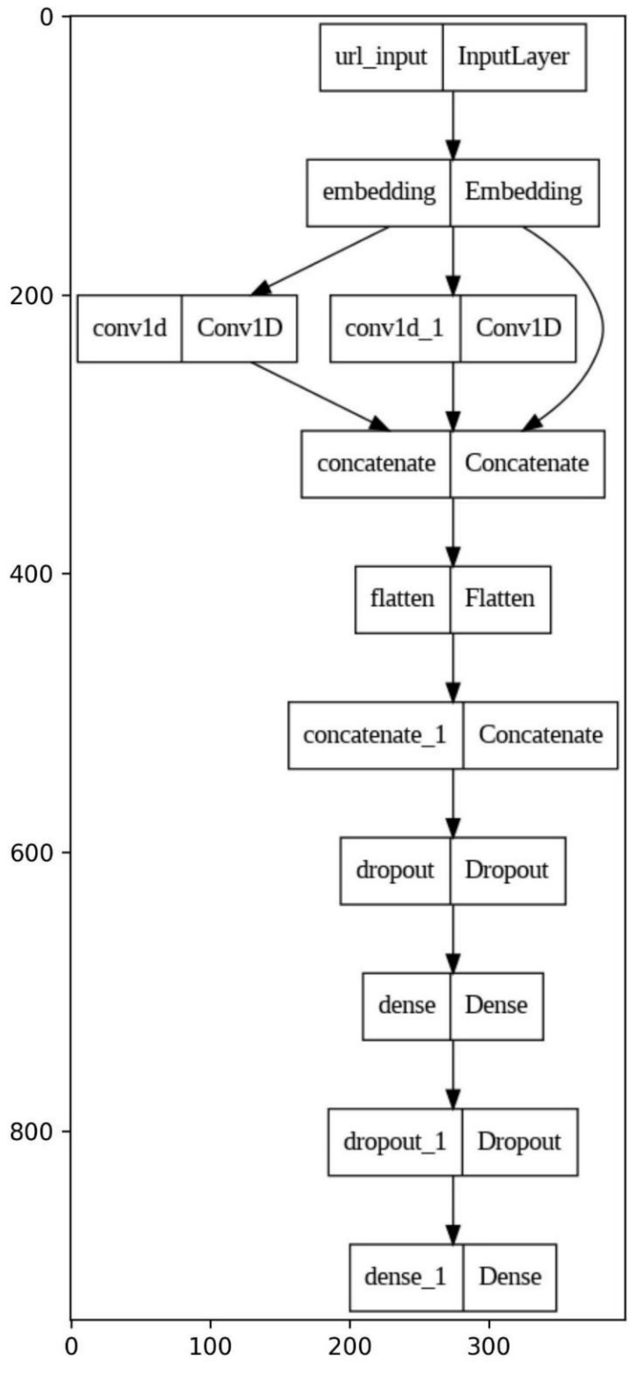


Figure 3: Architecture of 1D-CNN model

by finding the optimal hyperplane that separates data points of different classes in feature space. In URL detection, the feature space can be high-dimensional, considering various

URL attributes. SVM can handle high-dimensional data effectively and is less prone to overfitting.

- **Convolutional Neural Network:** CNN is a type of deep learning model designed for processing structured grid data, such as images or sequences. For URL analysis, a character-level CNN can learn hierarchical representations of the characters in URLs, capturing complex patterns. It's effective when there are intricate relationships between characters. The architecture of the 1D-Character level CNN is demonstrated in Figure 3.

5. **Evaluating the models:** For the performance evaluation of the machine learning models metrics such as precision, recall, F1, accuracy scores, ROC, AUC and Confusion matrix have been used.

Results:

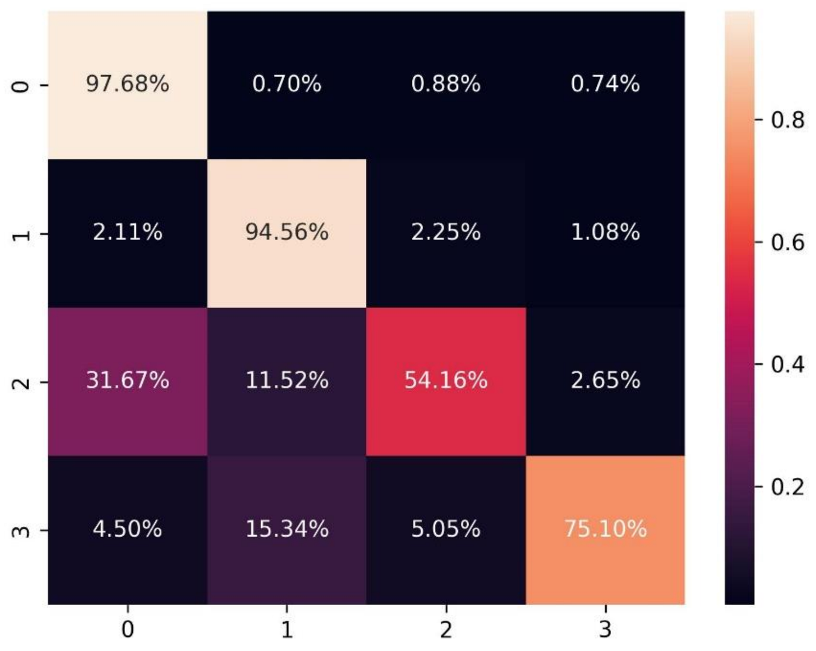
The result analysis in terms of precision, recall, accuracy and F1 score is given below:

Table 2: Result Analyses of the Developed Models

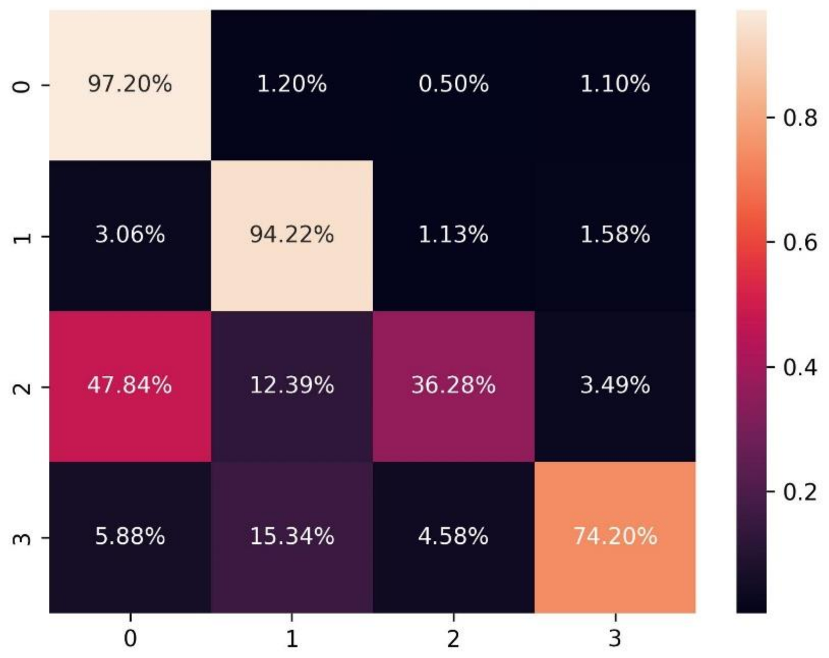
Feature Selection Method	Classifier	Precision	Recall	F1 score	Accuracy (%)
Selecting All Features	Decision Tree	0.96	0.96	0.96	96.03
	Random Forest	0.97	0.97	0.97	97.45
	Adaboost	0.78	0.83	0.79	82.67
	KNN	0.95	0.95	0.95	94.9
	SGD	0.79	0.82	0.79	81.84
	ExtraTree	0.97	0.97	0.97	97.40
	SVC	0.75	0.73	0.66	73.83
	Gaussian NB	0.76	0.79	0.75	79.19
Selecting Features by correlation with the target	Decision Tree	0.93	0.93	0.93	93.23
	Random Forest	0.95	0.95	0.95	94.84
	Adaboost	0.81	0.83	0.82	83.24
	KNN	0.93	0.94	0.94	93.64
	SGD	0.80	0.81	0.75	81.44
	ExtraTree	0.95	0.95	0.95	94.75
	SVC	0.79	0.82	0.76	81.72
	Gaussian NB	0.76	0.79	0.74	78.95
	Decision Tree	0.96	0.96	0.96	95.95

Selecting Features by Random Forest Feature Importance	Random Forest	0.97	0.97	0.97	97.44
	Adaboost	0.84	0.75	0.78	75.25
	KNN	0.95	0.95	0.95	94.87
	SGD	0.76	0.81	0.77	80.90
	ExtraTree	0.97	0.97	0.97	97.30
	Gaussian NB	0.79	0.79	0.78	78.60
	SVC	0.75	0.73	0.66	72.94
	1D CNN	0.97	0.97	0.97	97.67

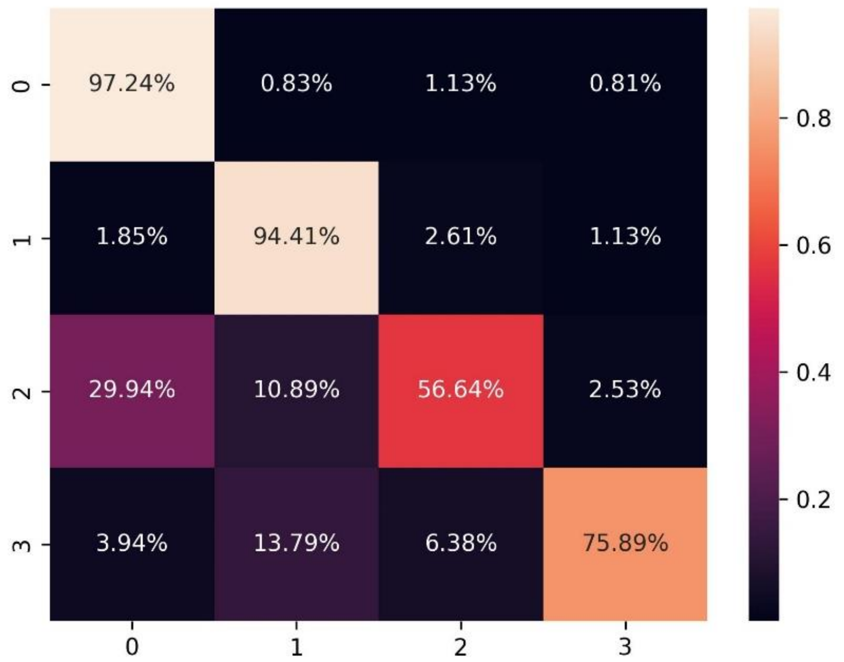
The analysis highlights the consistently strong performance of the Random Forest classifier, particularly when coupled with suitable feature selection methods. Feature selection methods play a significant role in classifier performance, with feature importance-based approaches demonstrating superiority. The confusion matrices and the ROC-AUC of the Random Forest model are provided for various feature selection scenarios, revealing its effectiveness in classifying malicious URLs.



a) Selecting all Features

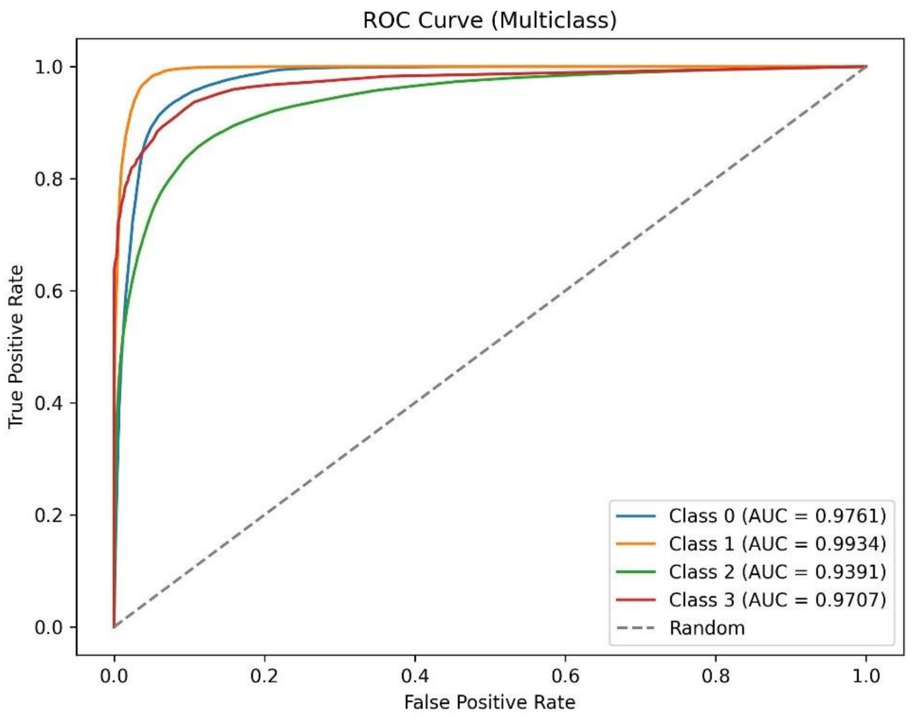


b) Selecting Features by correlation

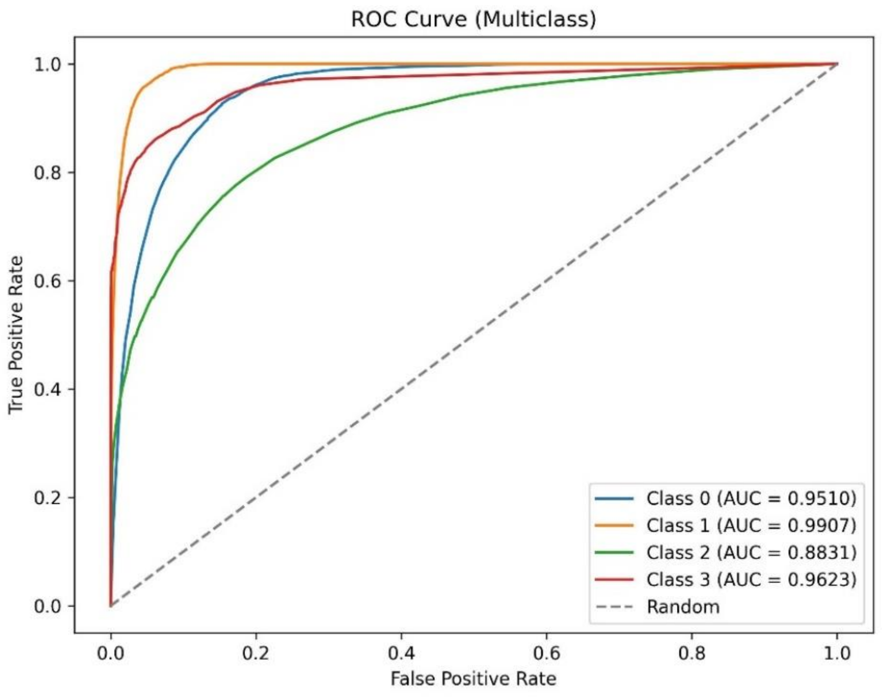


c) Selecting Features by Random Forest Feature Importance

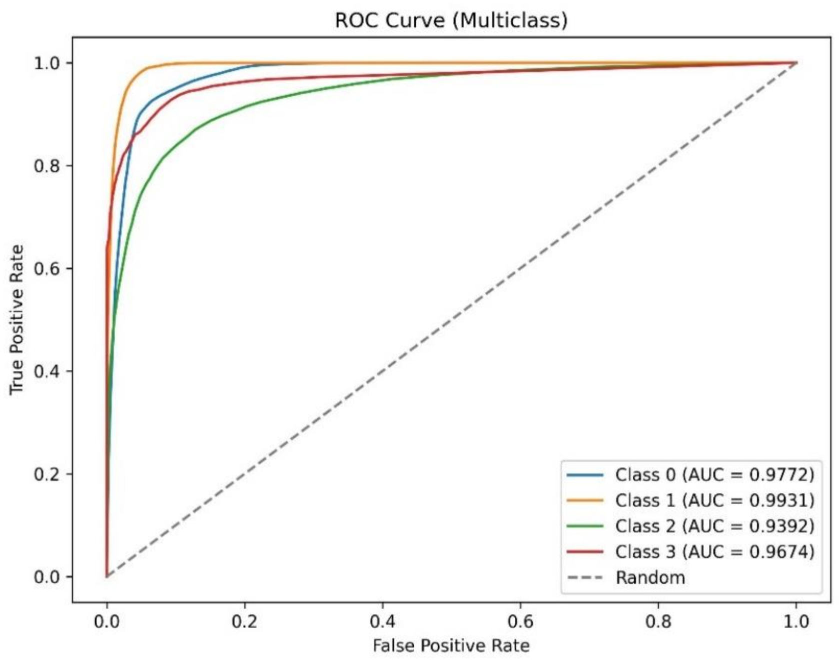
Figure 4: Confusion matrices of the Random Forest Classifier



a) Selecting all Features



b) Selecting Features by correlation



c) Selecting Features by Random Forest Feature Importance

Figure 5: ROC-AUC of the Random Forest Classifier

It is also evident from the table that, the character level 1D CNN classifier exhibits exceptional results, suggesting that deep learning approaches could be effective for the detection task. The confusion matrix of the character level CNN is given below:

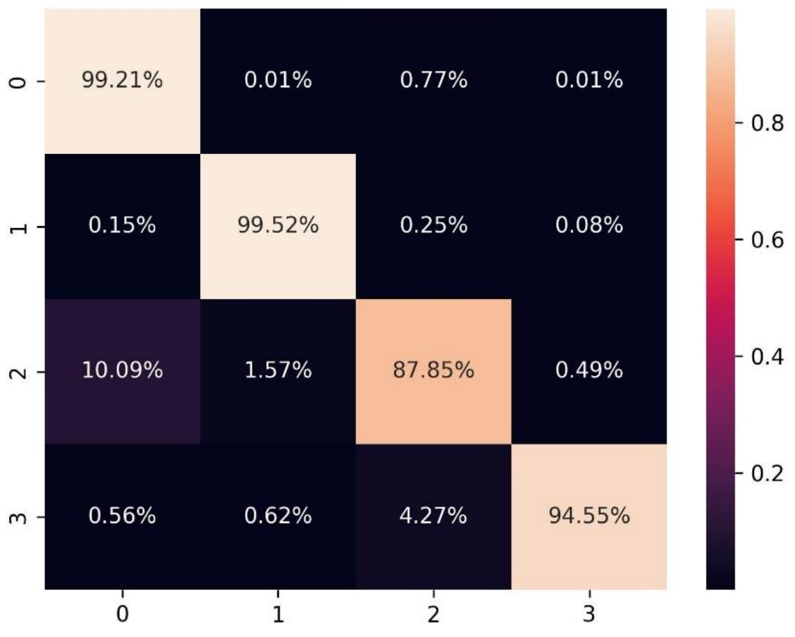


Figure 6: Confusion matrices of the character level CNN

The ROC-AUC of the character level CNN is given below:

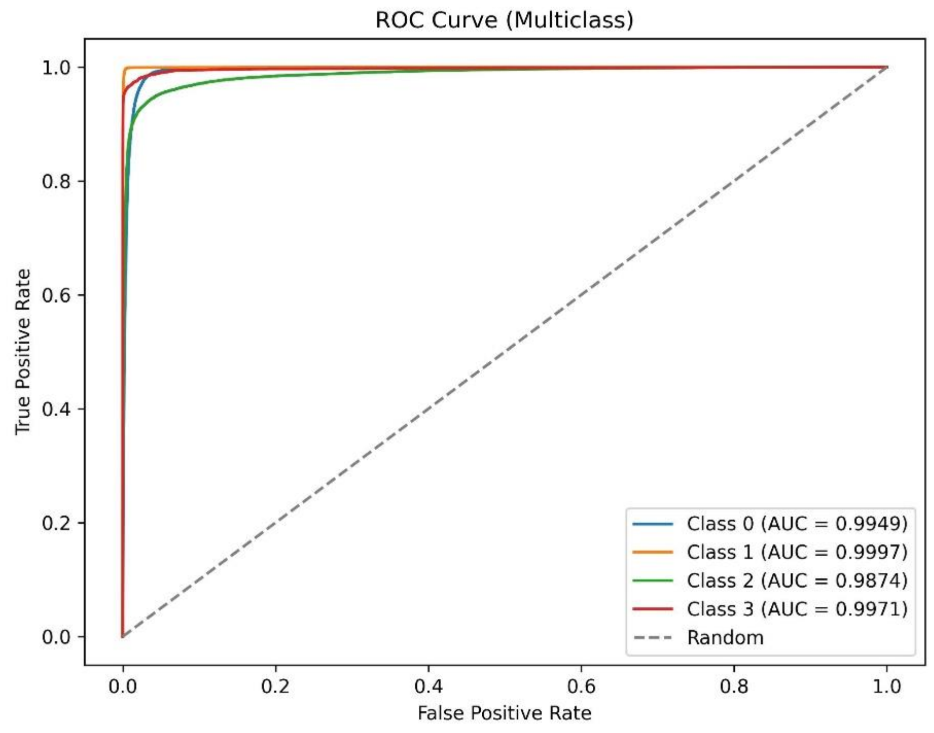


Figure 7: ROC-AUC of the character level CNN

The confusion matrices and the ROC-AUC curves of the rest of the models can be found in the Appendix section.

However, studying the confusion matrices we can see, that even if the models perform better for label 0 and 1 which are classes such as ‘benign’ and ‘defacement’, they all collectively falter in correctly classifying label 2 or ‘phishing’ attacks. The ROC curves also show comparatively less AUC score in class label 2 than other classes.

Overall, the results analysis showcases the performance of various classifiers with different feature selection methods in the task of Malicious URL Detection. It provides valuable insights into the strengths and weaknesses of different combinations. The key findings are summarized as follows:

1. The analysis reveals that Random Forest and ExtraTree classifiers consistently deliver strong performance across multiple feature selection methods. These classifiers, in conjunction with appropriate feature selection, demonstrate high Precision, Recall, F1 scores, and Accuracy.
2. The choice of feature selection method plays a crucial role in classifier performance. Feature importance-based methods, such as those using Random Forest Feature Importance, consistently outperform other feature selection techniques.

3. In contrast, Adaboost and Gaussian NB classifiers exhibit comparatively weaker performance, suggesting the need for further optimization or alternative feature engineering strategies.
4. The character-level 1D CNN classifier showcases exceptional results, hinting at the effectiveness of deep-learning approaches for malicious URL detection.
5. A closer examination of the confusion matrices and ROC curves indicates a collective challenge in correctly classifying label 2, which represents 'phishing' attacks. Despite the strong performance in classifying labels 0 ('benign') and 1 ('defacement'), there is room for improvement in handling phishing attacks.

Conclusion

The Malicious URL Detection project can be successfully developed and implemented as an effective system for identifying and blocking malicious URLs. By leveraging a combination of traditional machine learning models and deep learning techniques, robust results can be achieved in classifying URLs as benign or malicious. This solution has a significant impact, enhancing online security, protecting users from cyber threats, and contributing to a safer online environment. The insights/inferences from this project can be summarized as below:

- Feature extraction, including various characteristics of URLs such as length, entropy, and counts of specific characters, contributes to building a robust model for detecting malicious URLs. Domain-related information and URL structure are crucial in capturing patterns indicative of phishing, malware, and defacement attacks. The choice of features aligns with the understanding that the landscape of malicious URLs is continually evolving. By leveraging features related to URL structure and content, the model can adapt to new techniques employed by cybercriminals.
- The project emphasizes the importance of feature selection for model performance. Features like URL length, domain, subdomain, and various counts (characters, special characters) play a crucial role in distinguishing between benign and malicious URLs. The Random Forest feature importance technique highlights specific features that contribute significantly to the reduction in impurity across decision trees. This emphasizes the need for focusing on relevant features to improve model interpretability and avoid overfitting.
- The exceptional performance of the 1D CNN classifier suggests that deep learning, especially at the character level, is effective in capturing intricate patterns within URLs. This insight implies that the hierarchical representations learned by a CNN can discern complex relationships between characters.
- We can observe a collective challenge in correctly classifying phishing attacks (label 2). This insight indicates a potential area for improvement in the model's ability to identify and differentiate phishing URLs effectively.

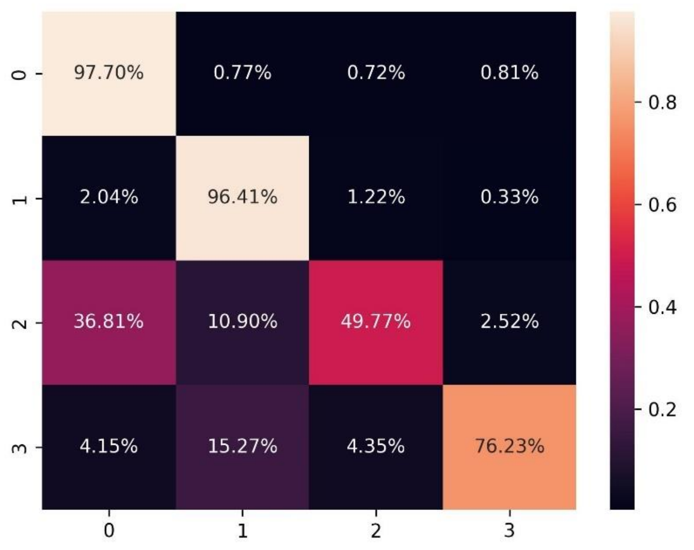
In summary, the impact of features extracted from the data underscore the importance of thoughtful feature selection and the effectiveness of deep learning in addressing the challenges of malicious URL detection. The insights provide guidance for refining the model, improving its capability to handle evolving threats, and enhancing its practical application in real-world scenarios

References:

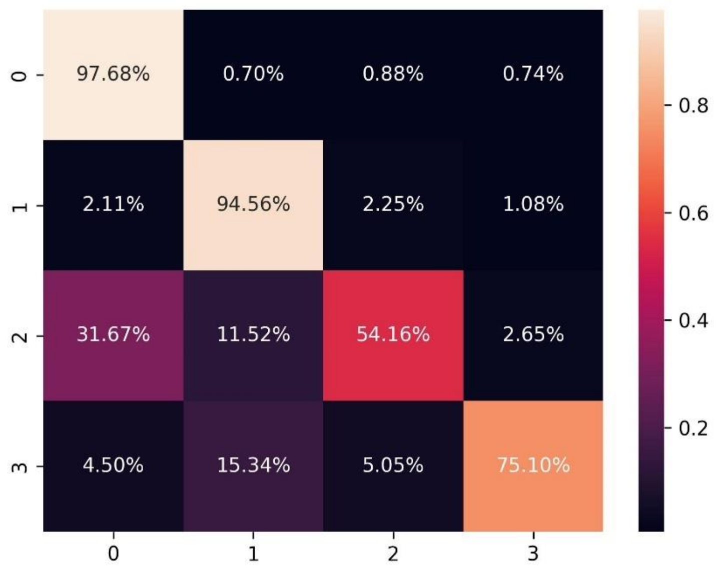
[1] Ma,Justin, Saul,Lawrence, Savage,Stefan, and Voelker,Geoffrey. (2009). URL Reputation. UCI Machine Learning Repository. https://doi.org/10.24432/C5H89Q.

Appendix

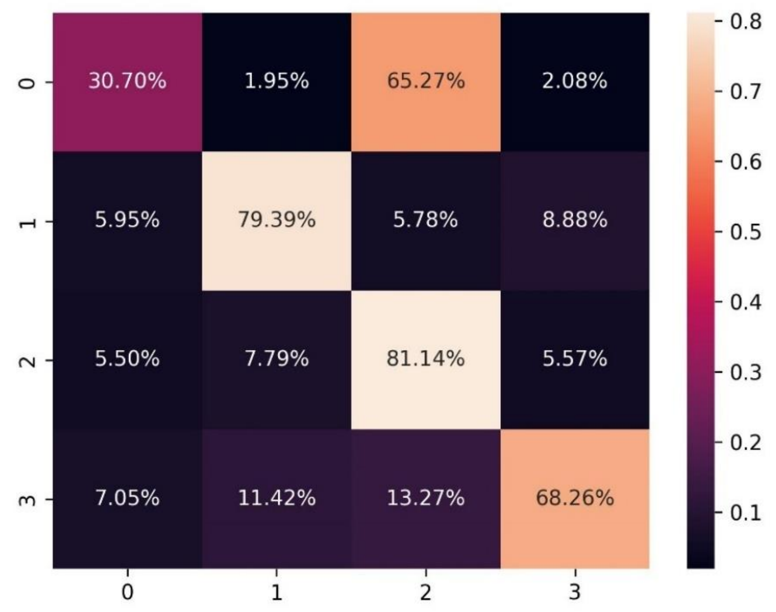
The confusion matrices of the models built by selecting all features are given below:



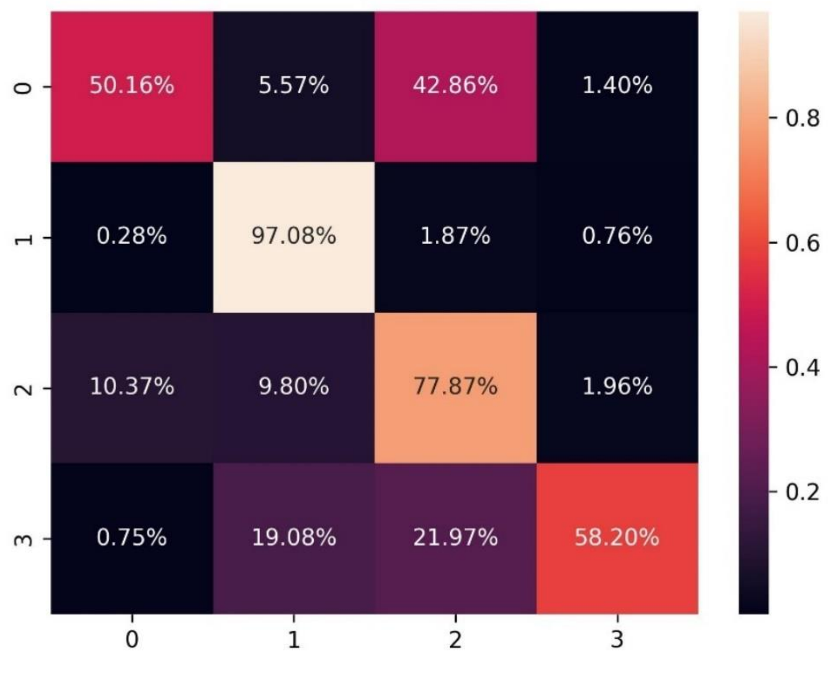
Confusion Matrix of Extra Tree Classifier



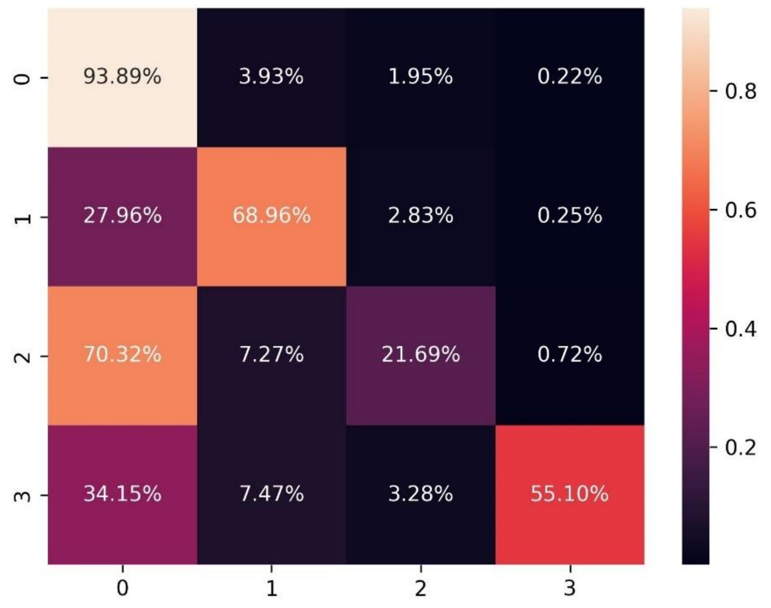
Confusion Matrix of Random Forest



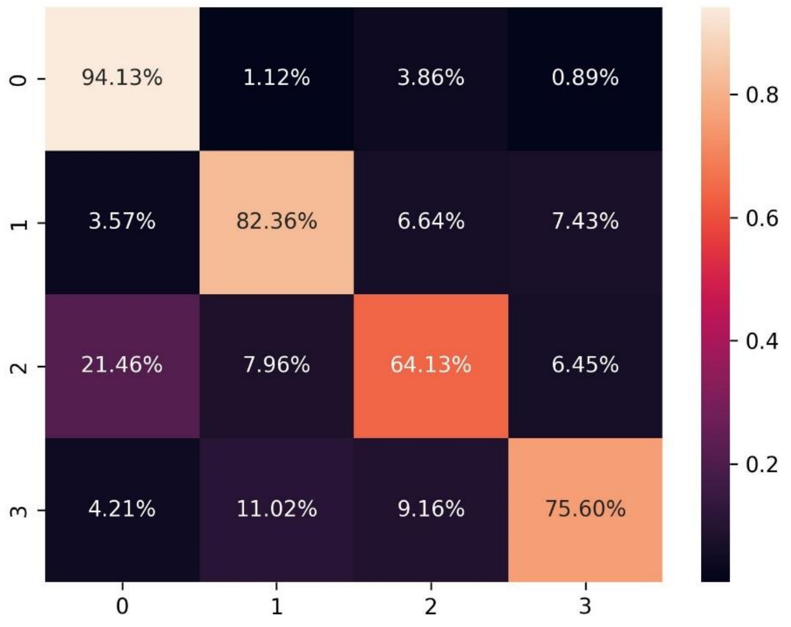
Confusion Matrix of AdaBoost Classifier



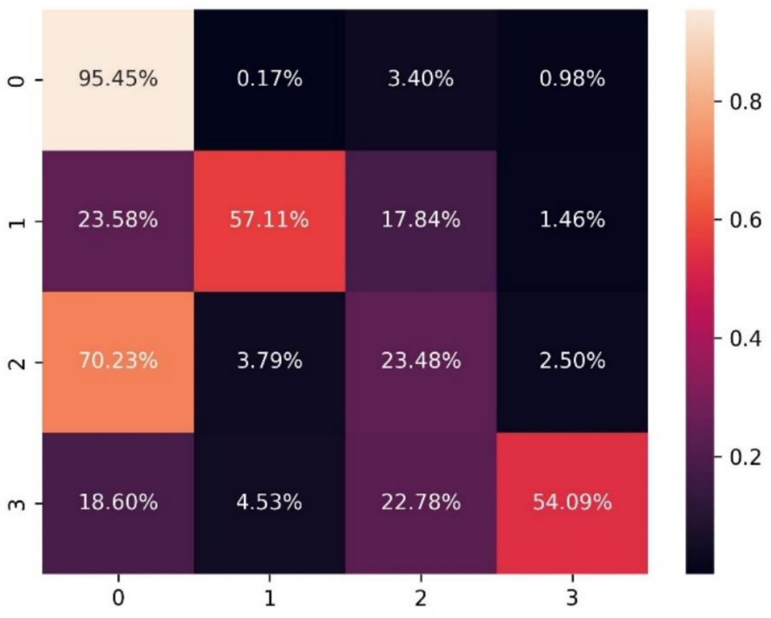
Confusion Matrix of Naïve Bayes



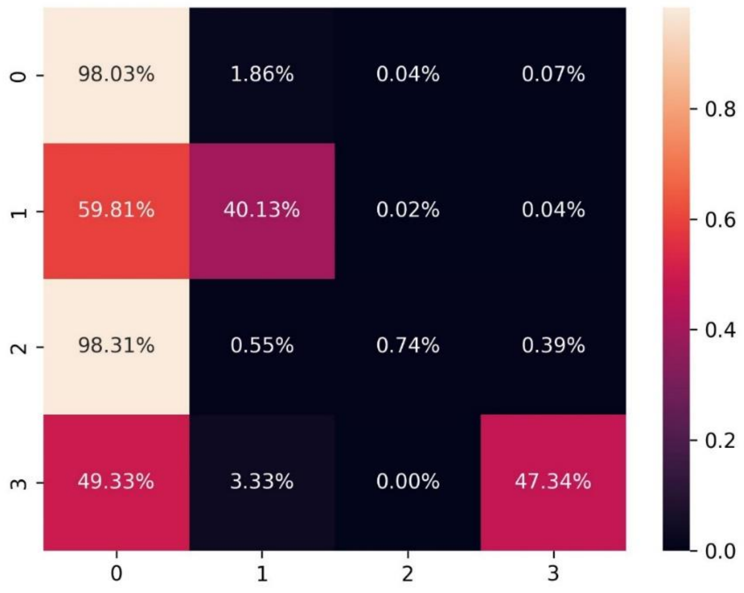
Confusion Matrix of KNN



Confusion Matrix of Decision Trees

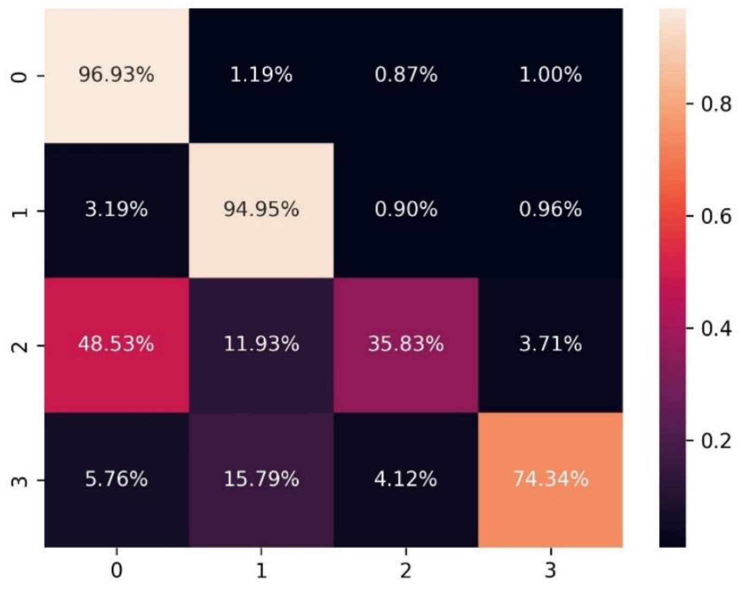


Confusion Matrix of Stochastic Gradient Classifier

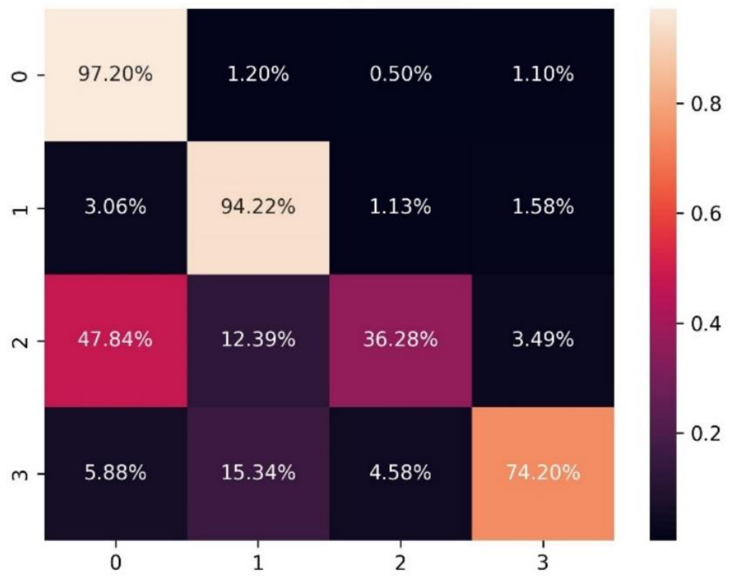


Confusion Matrix of Support Vector Classifier

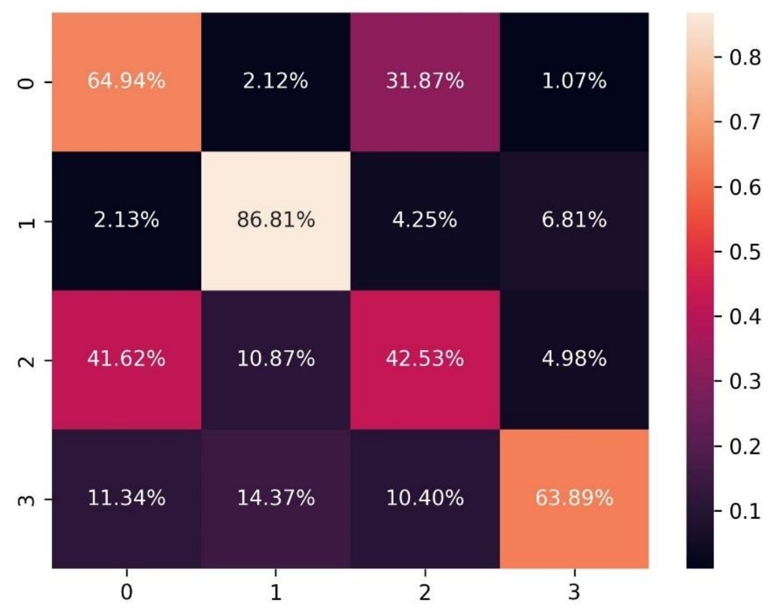
The confusion matrices of the models built by selecting features by correlation are given below:



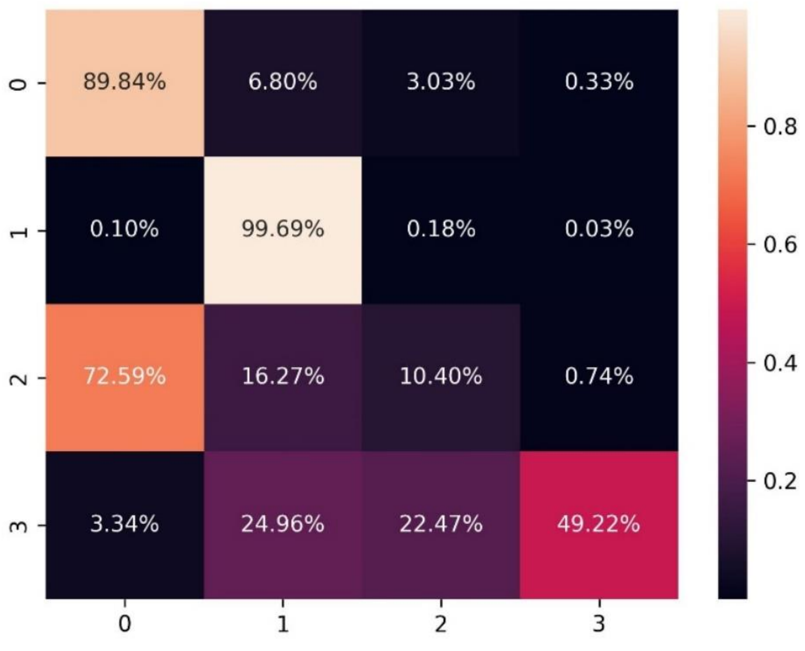
Confusion Matrix of Extra Tree Classifier



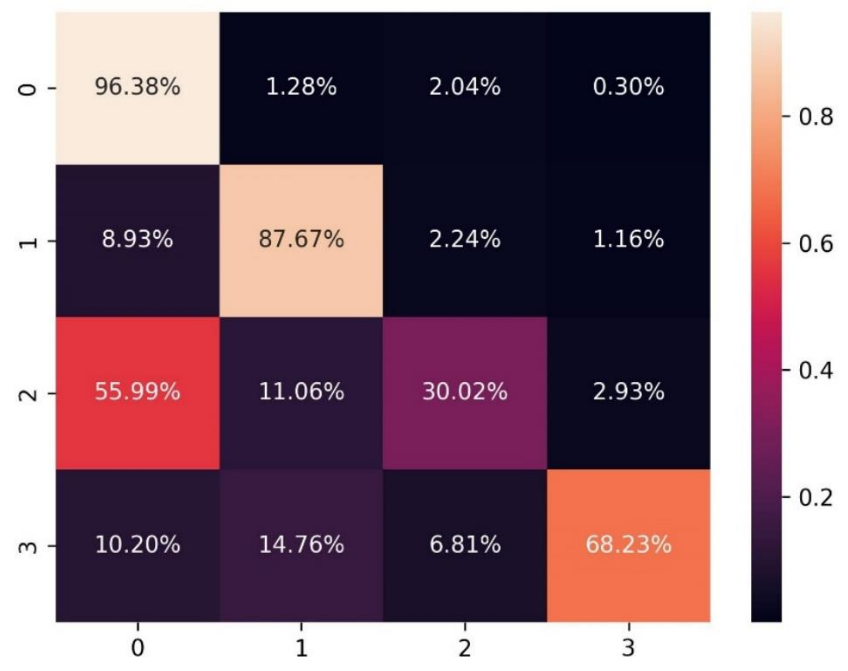
Confusion Matrix of Random Forest



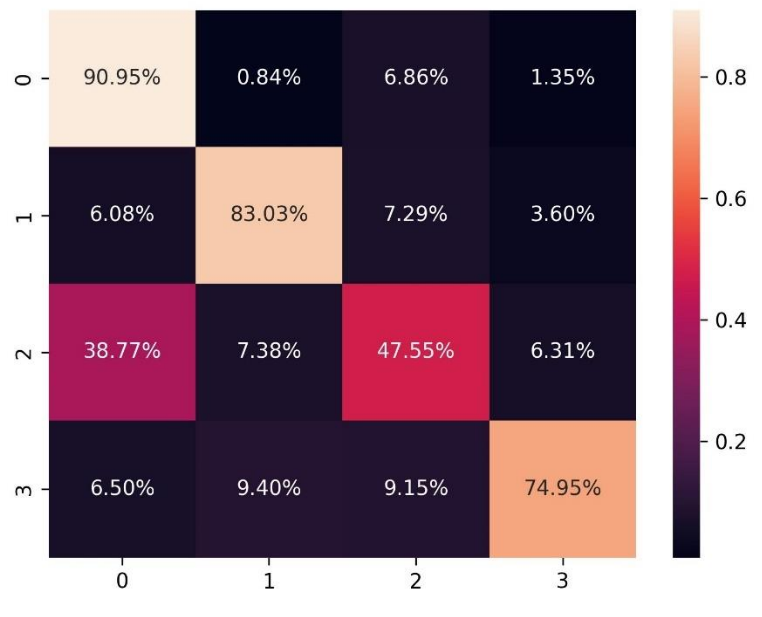
Confusion Matrix of AdaBoost Classifier



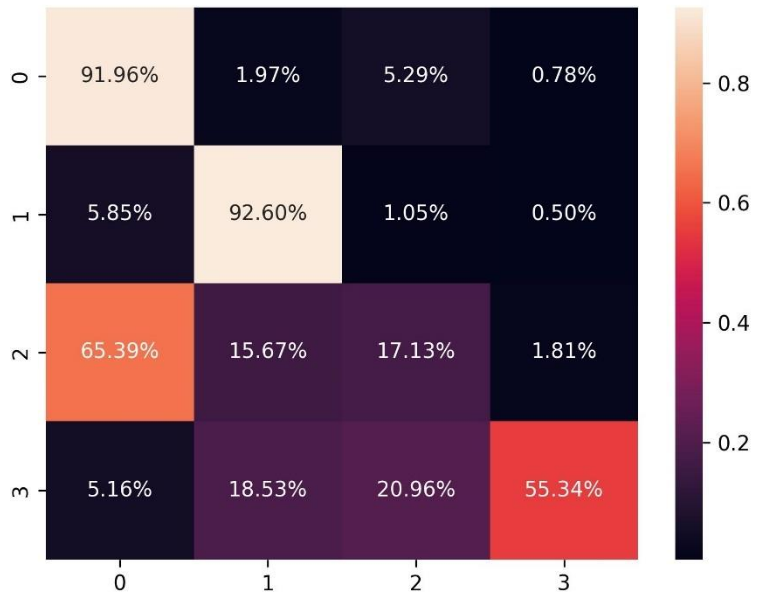
Confusion Matrix of Naïve Bayes



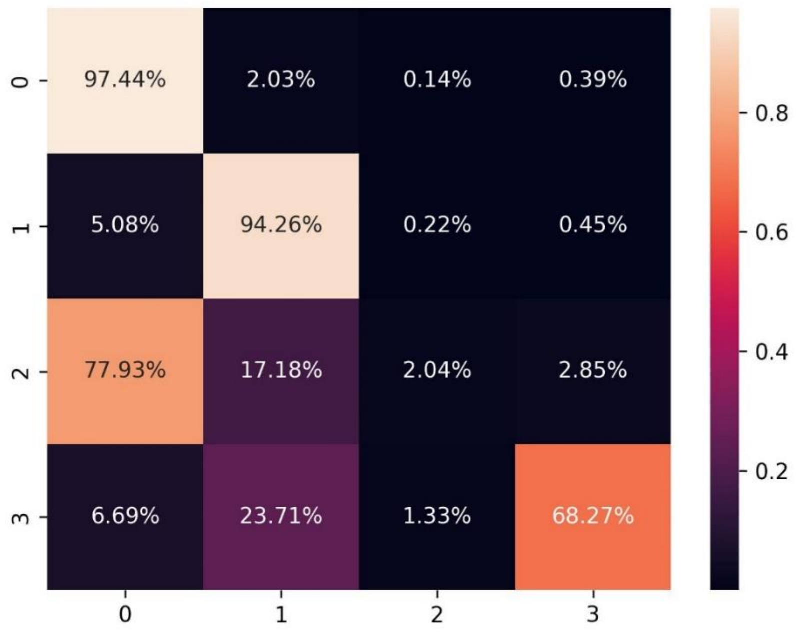
Confusion Matrix of KNN



Confusion Matrix of Decision Trees

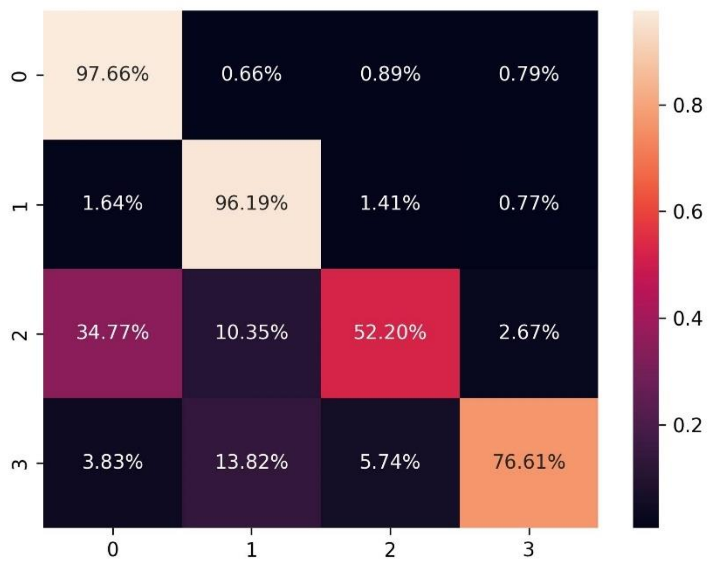


Confusion Matrix of Stochastic Gradient Classifier

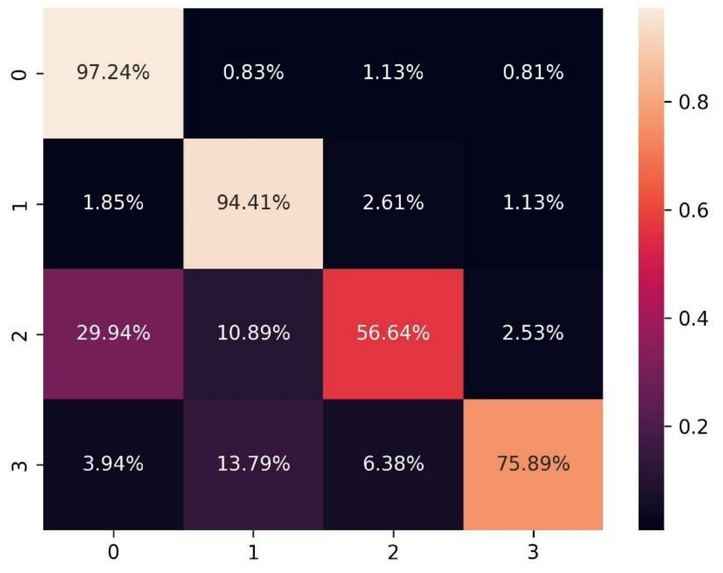


Confusion Matrix of Support Vector Classifier

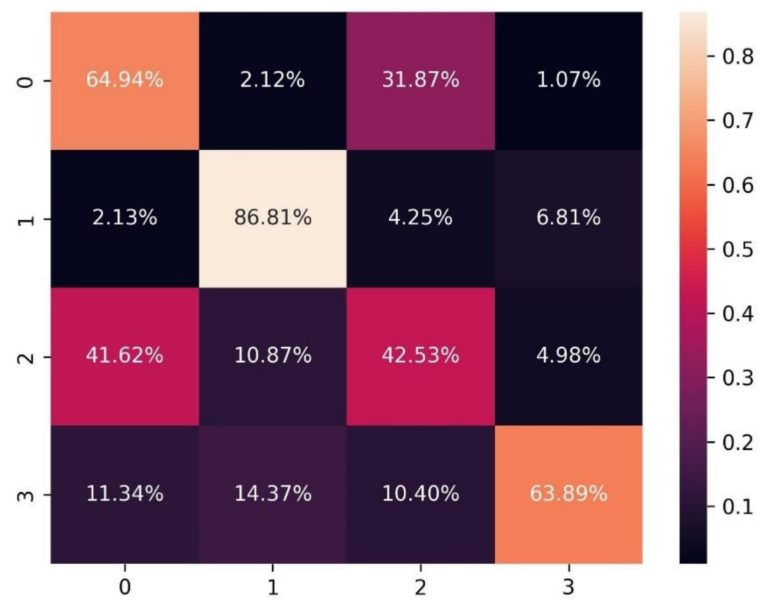
The confusion matrices of the models built by selecting features by random forest feature importance are given below:



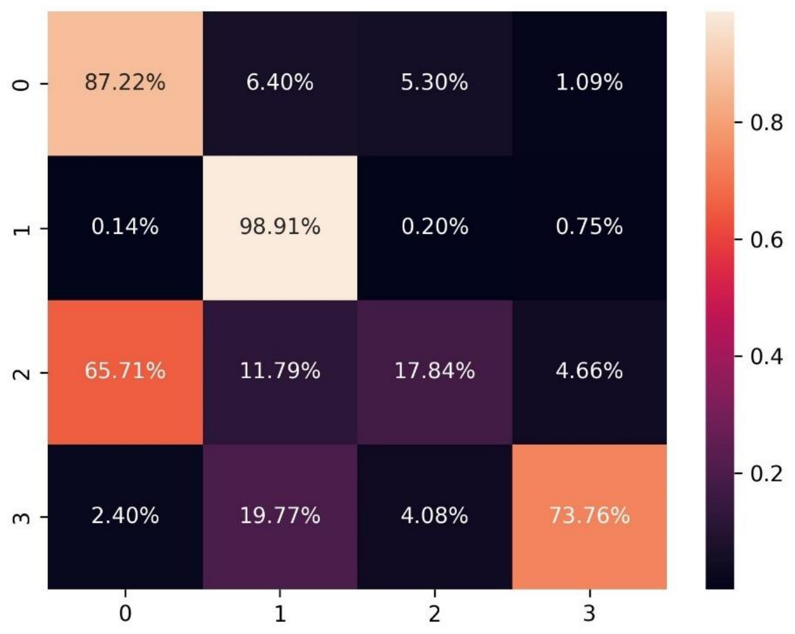
Confusion Matrix of Extra Tree Classifier



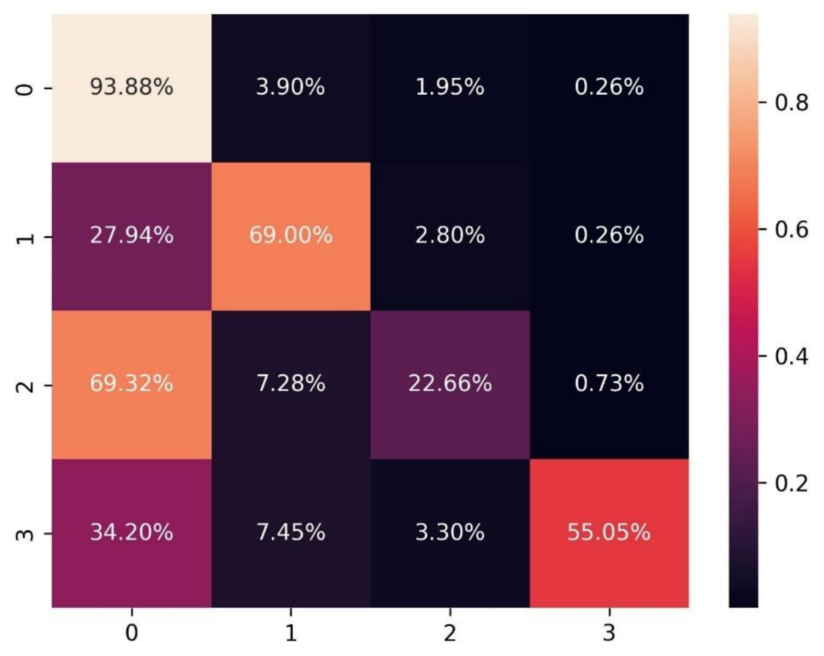
Confusion Matrix of Random Forest Model



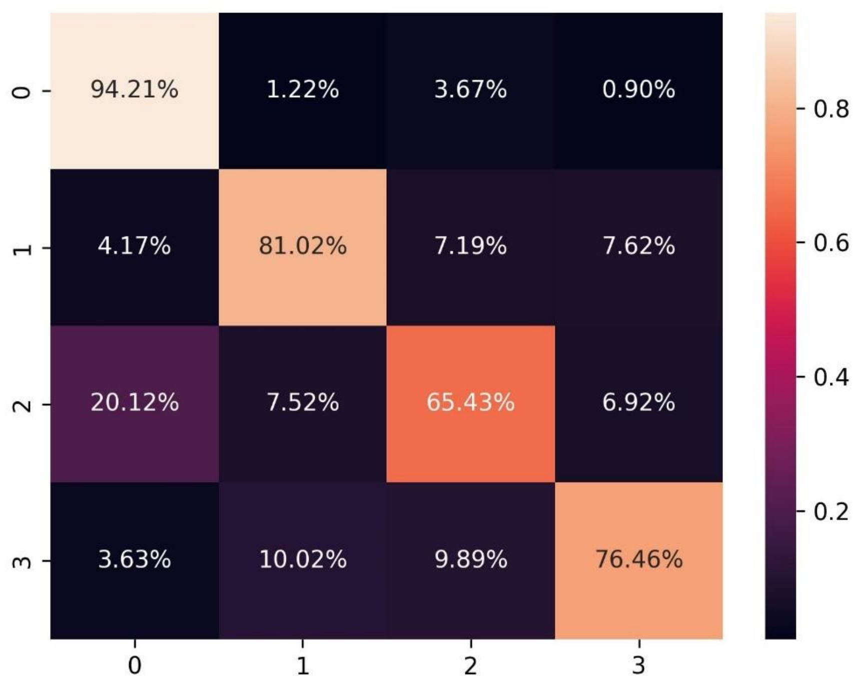
Confusion Matrix of AdaBoost Classifier



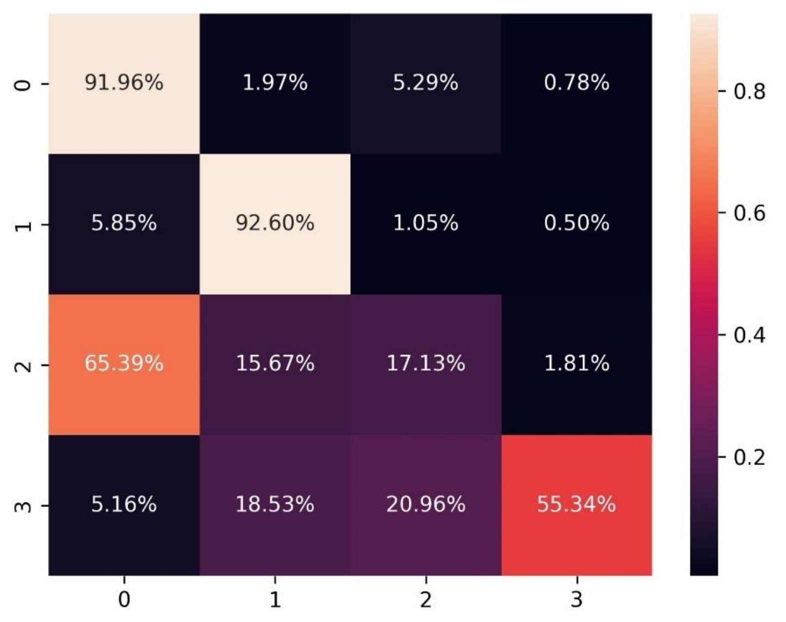
Confusion Matrix of Naïve Bayes



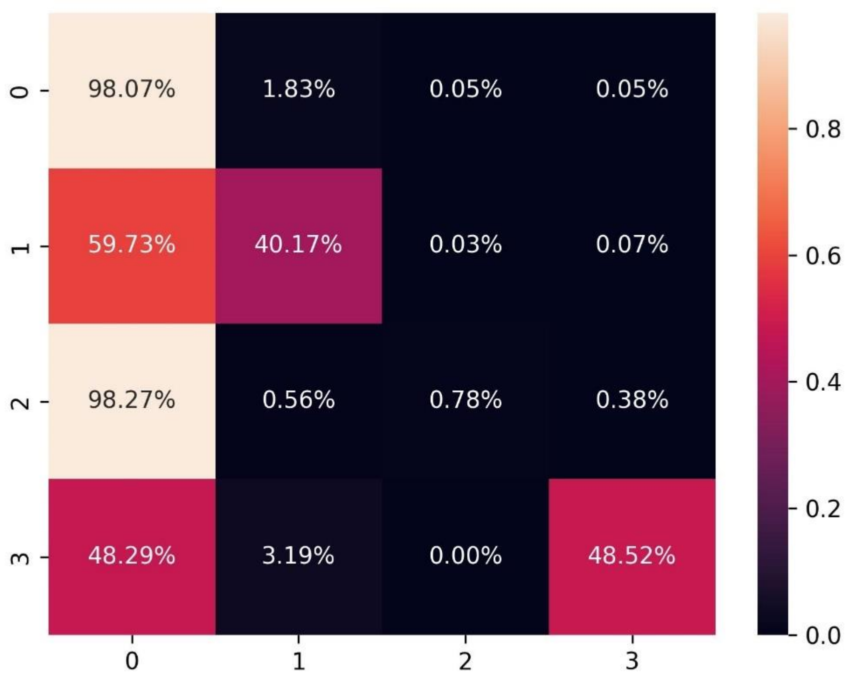
Confusion Matrix of KNN



Confusion Matrix of Decision Trees

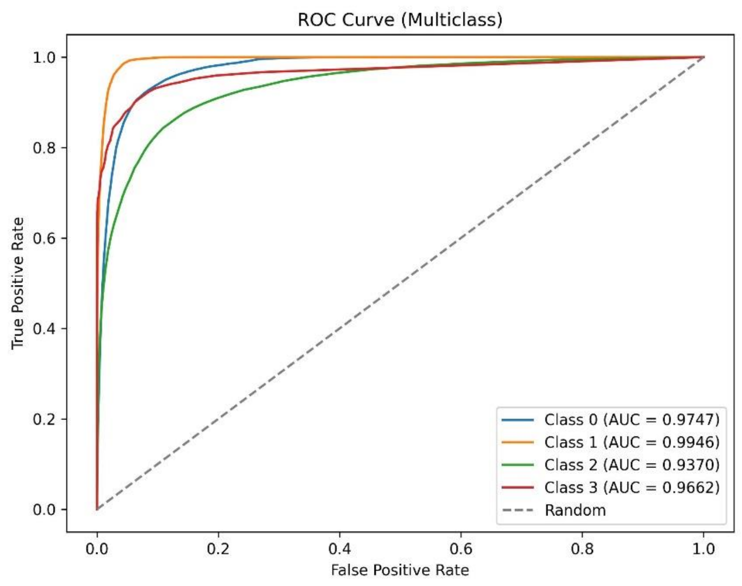


Confusion Matrix of Stochastic Gradient Classifier

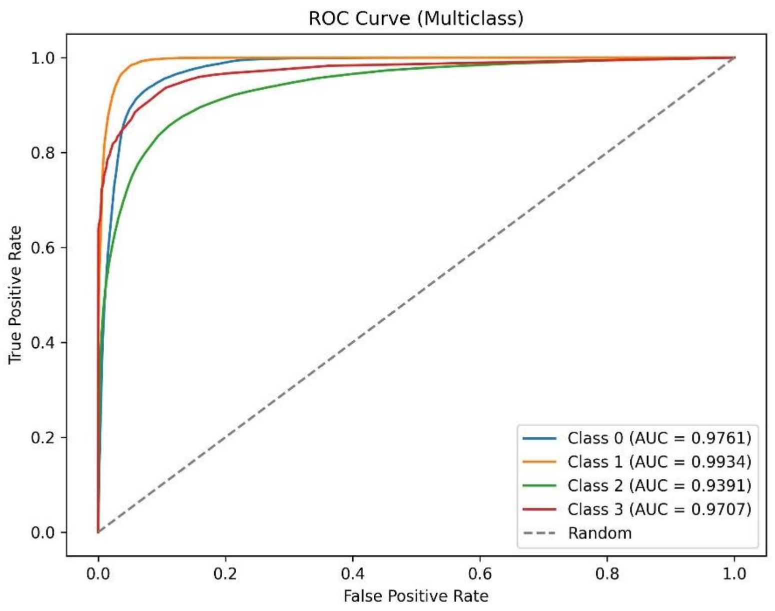


Confusion Matrix of Support Vector Classifier

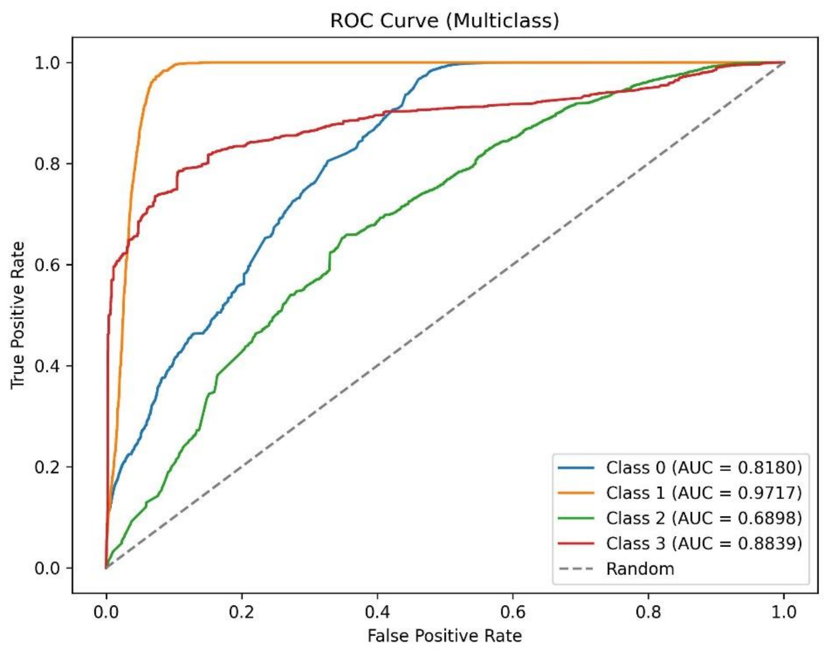
The ROC-AUC of the models built by selecting all features are given below:



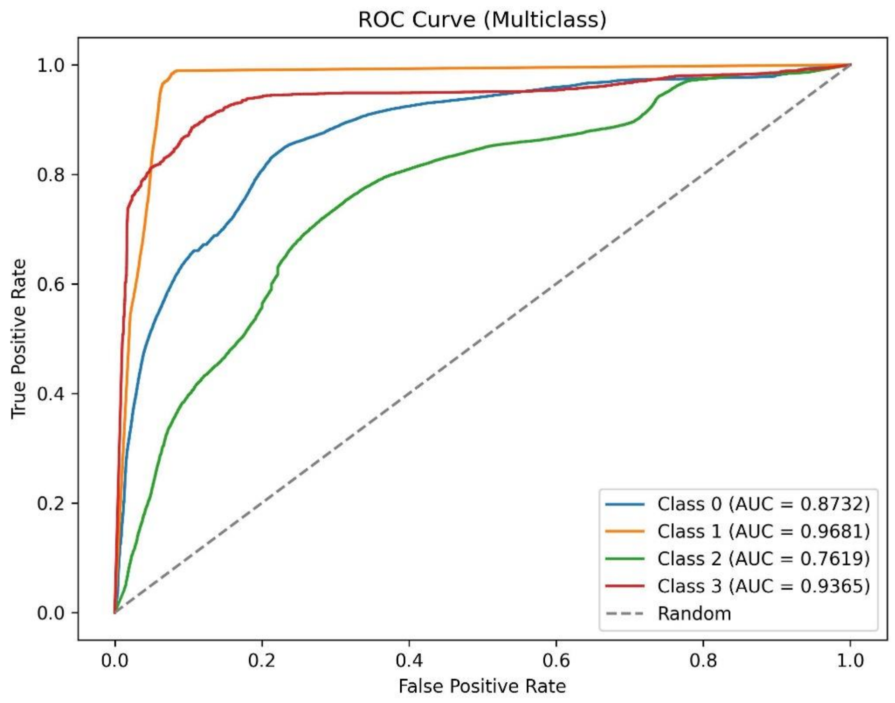
ROC of Extra Tree Classifier



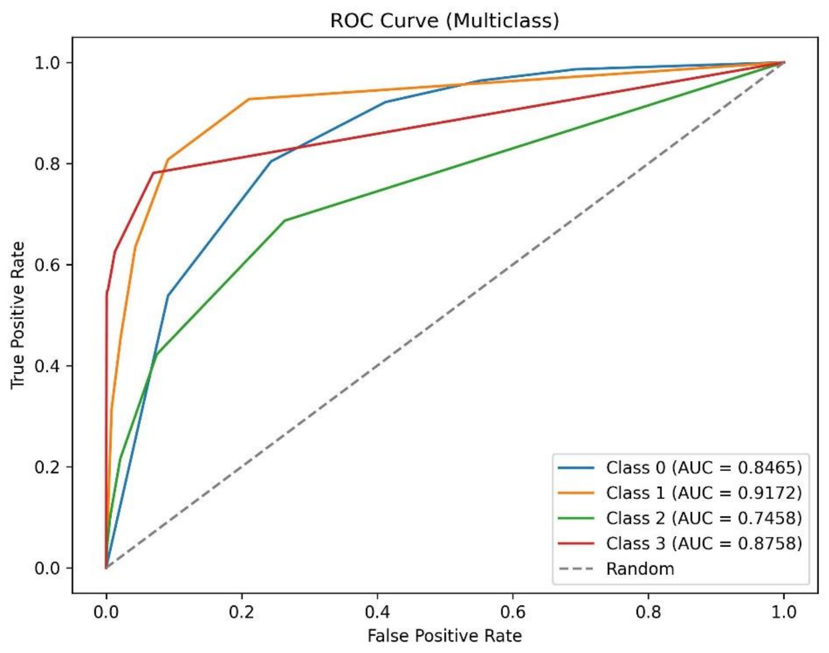
ROC of Random Forest



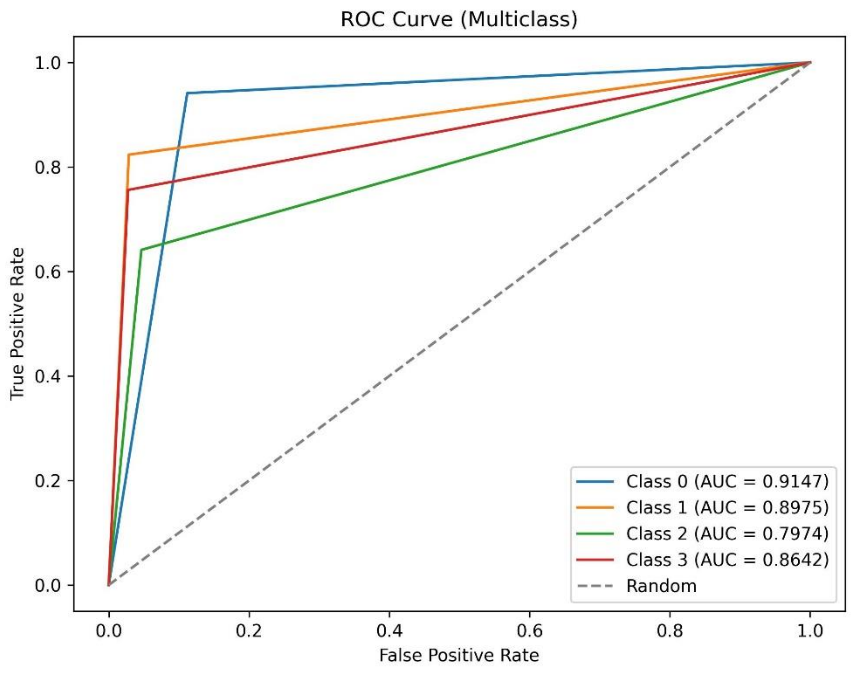
ROC of AdaBoost Classifier



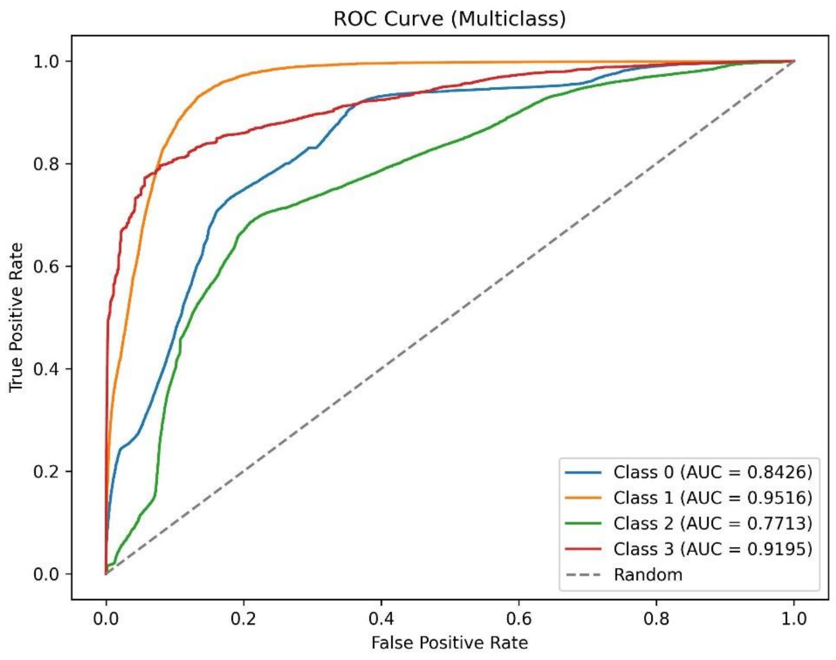
ROC of Naïve Bayes



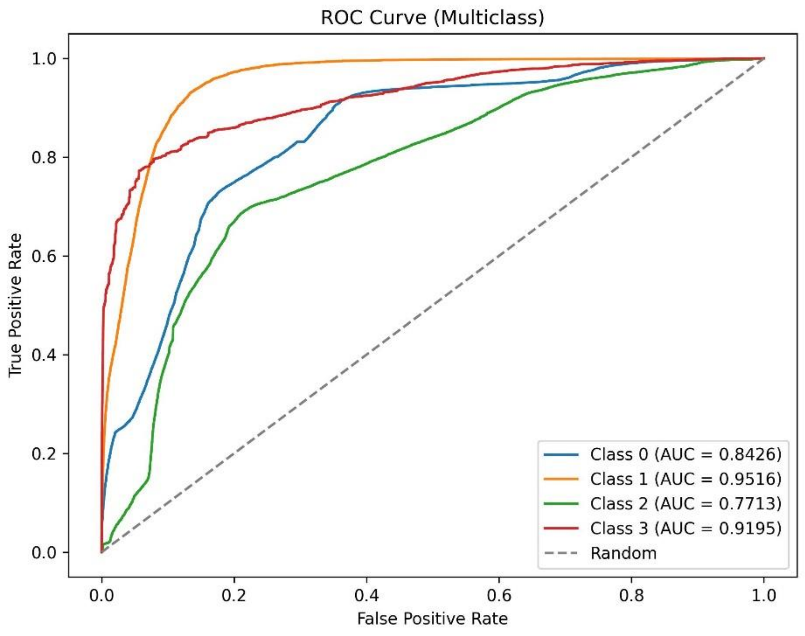
ROC of KNN



ROC of Decision Trees

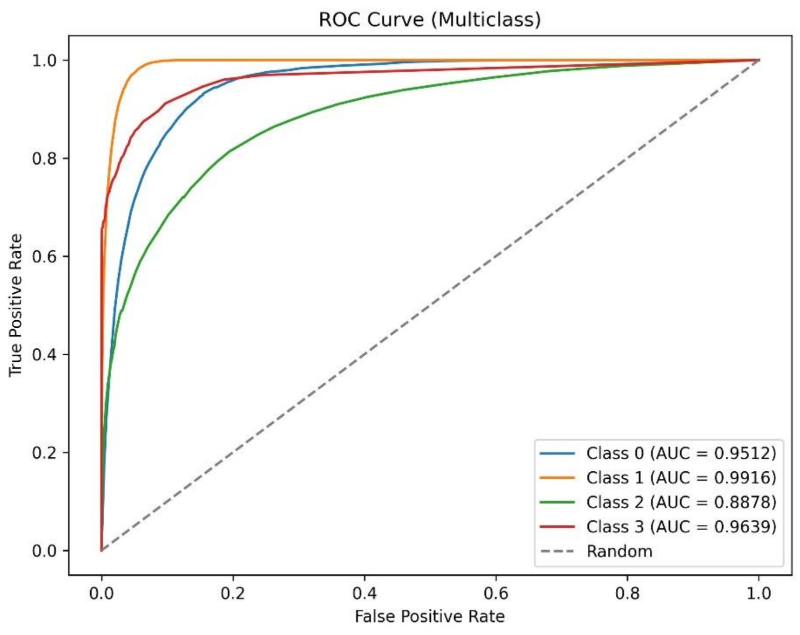


ROC of Stochastic Gradient Classifier

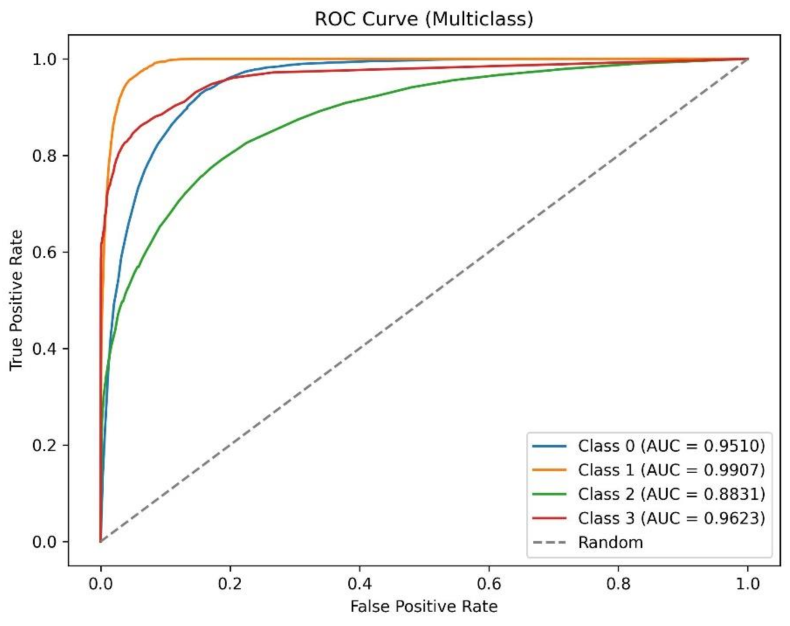


ROC of Support Vector Classifier

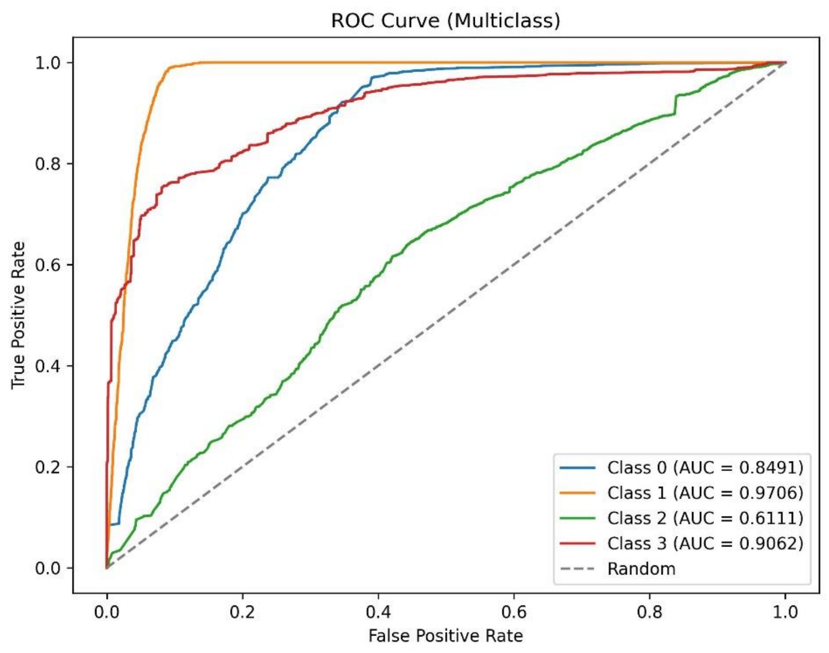
The ROC of the models built by selecting features by correlation are given below:



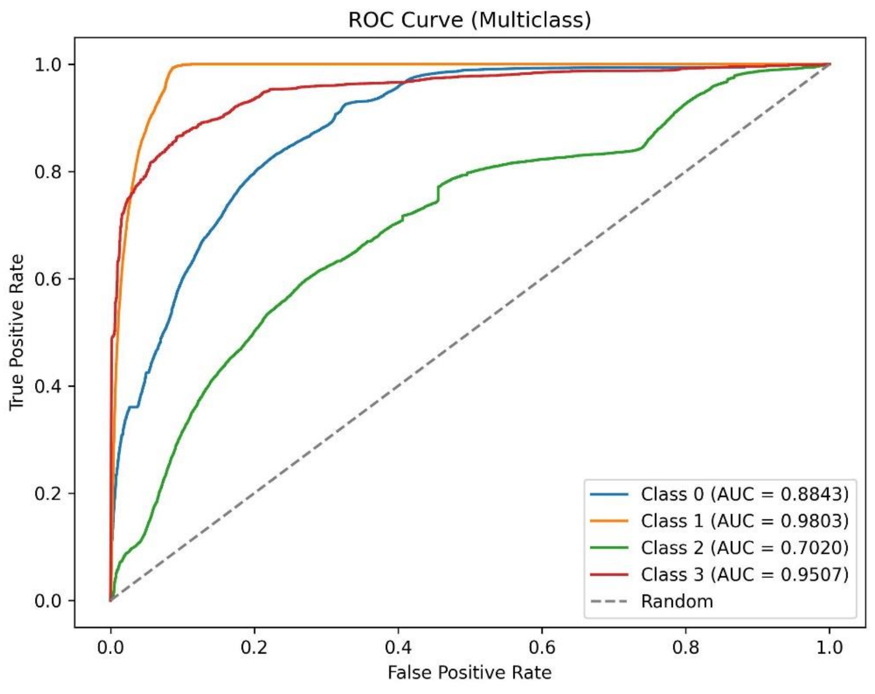
ROC of Extra Tree Classifier



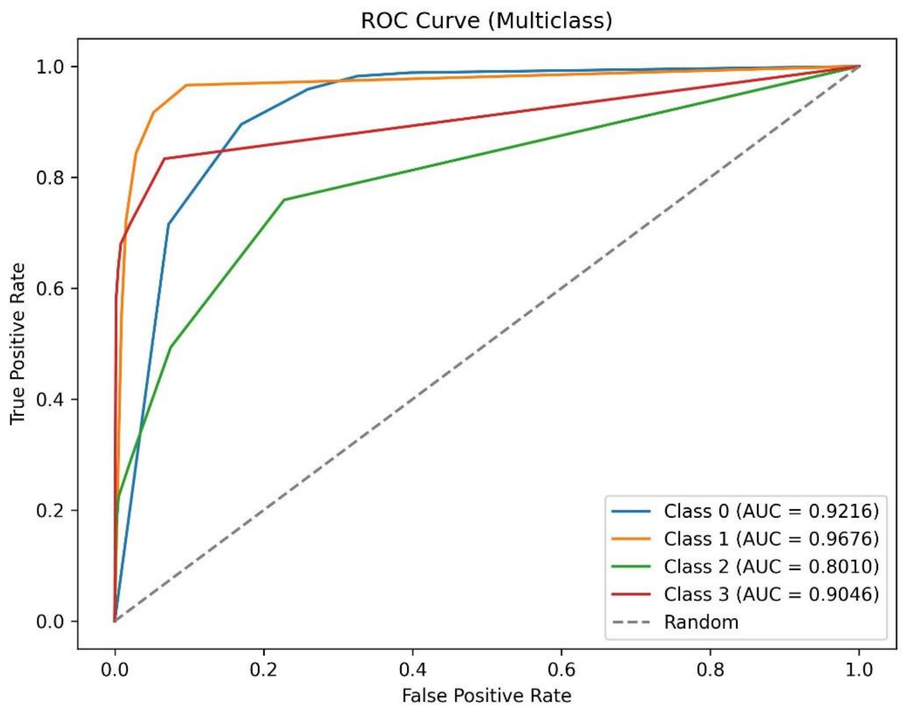
ROC of Random Forest



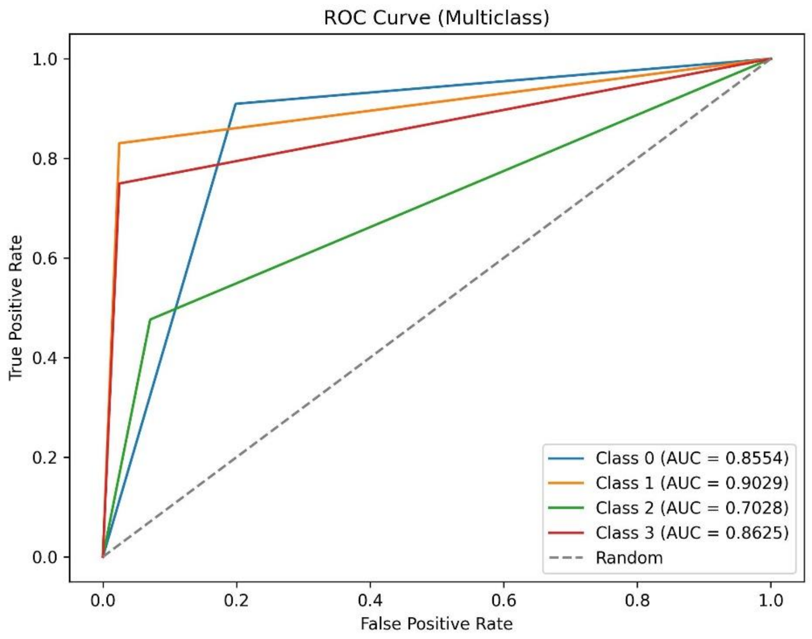
ROC of AdaBoost Classifier



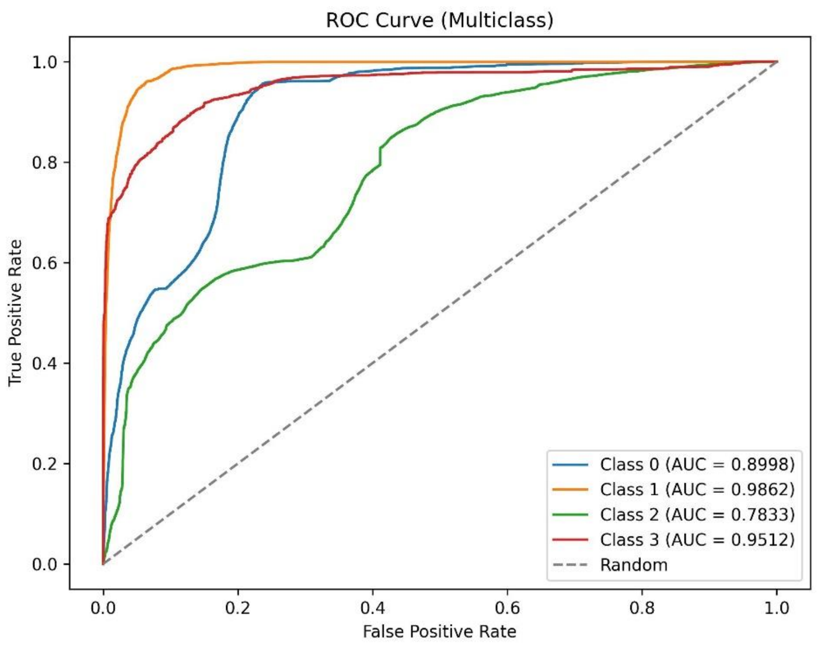
ROC of Naïve Bayes



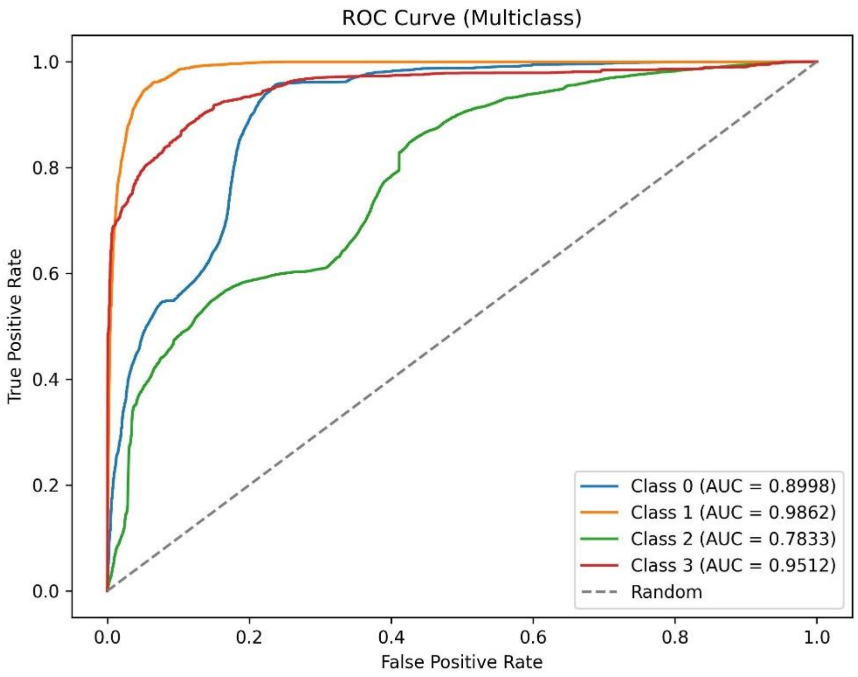
ROC of KNN



ROC of Decision Trees

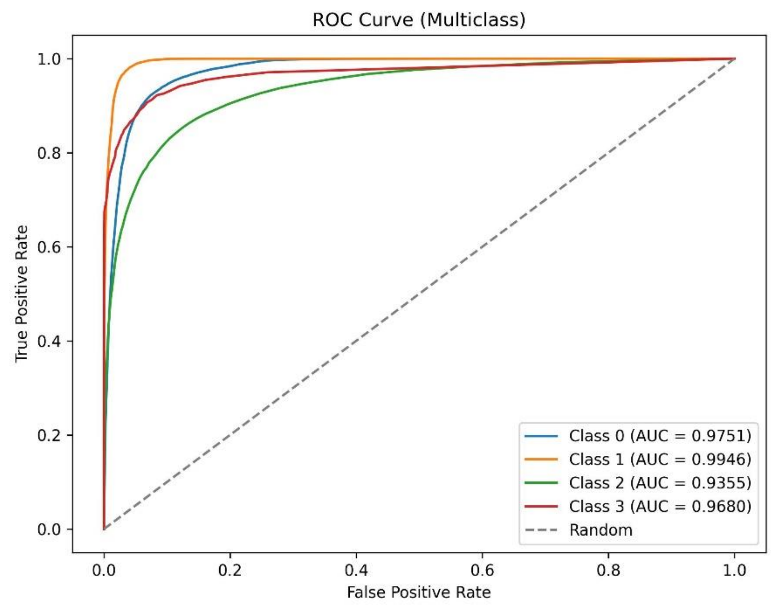


ROC of Stochastic Gradient Classifier

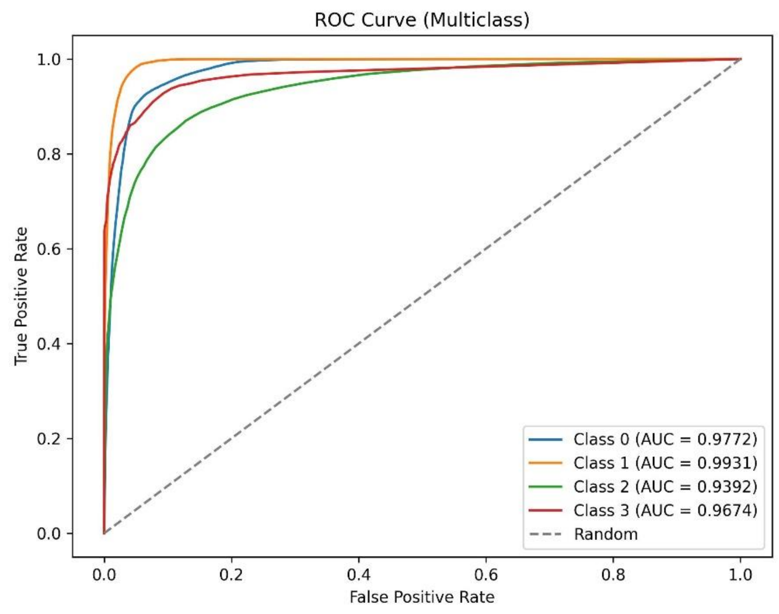


ROC of Support Vector Classifier

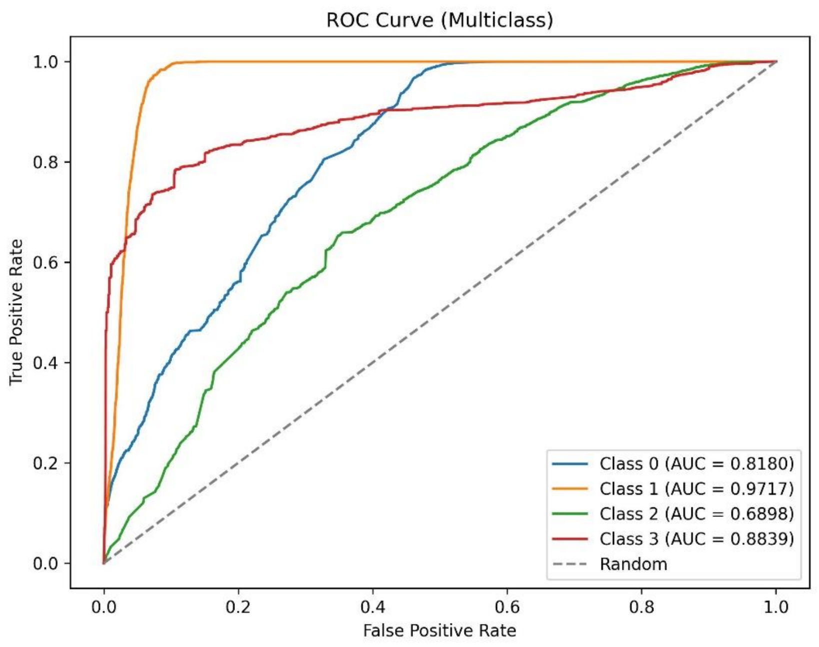
The ROC of the models built by selecting features by random forest feature importance are given below:



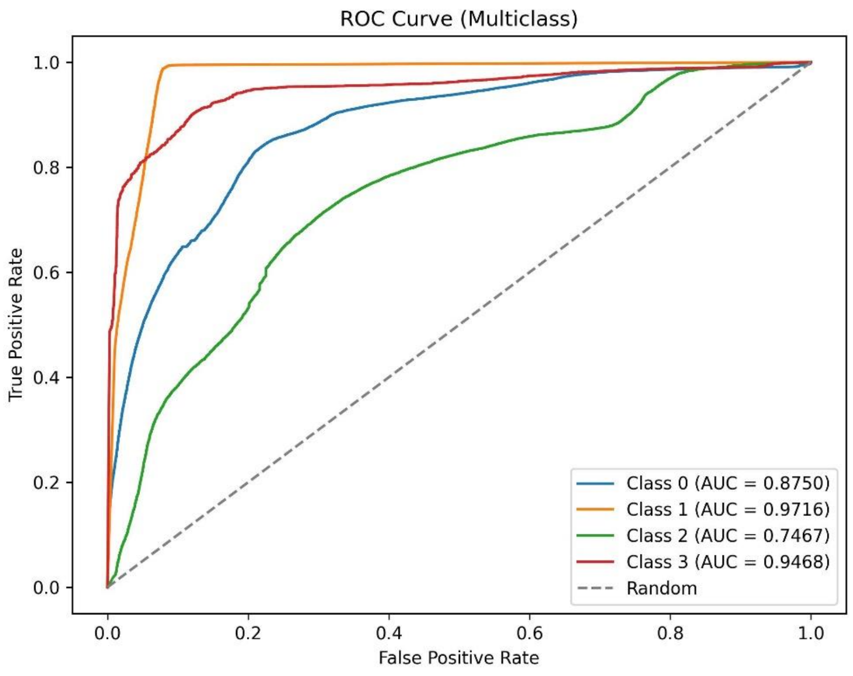
ROC of Extra Tree Classifier



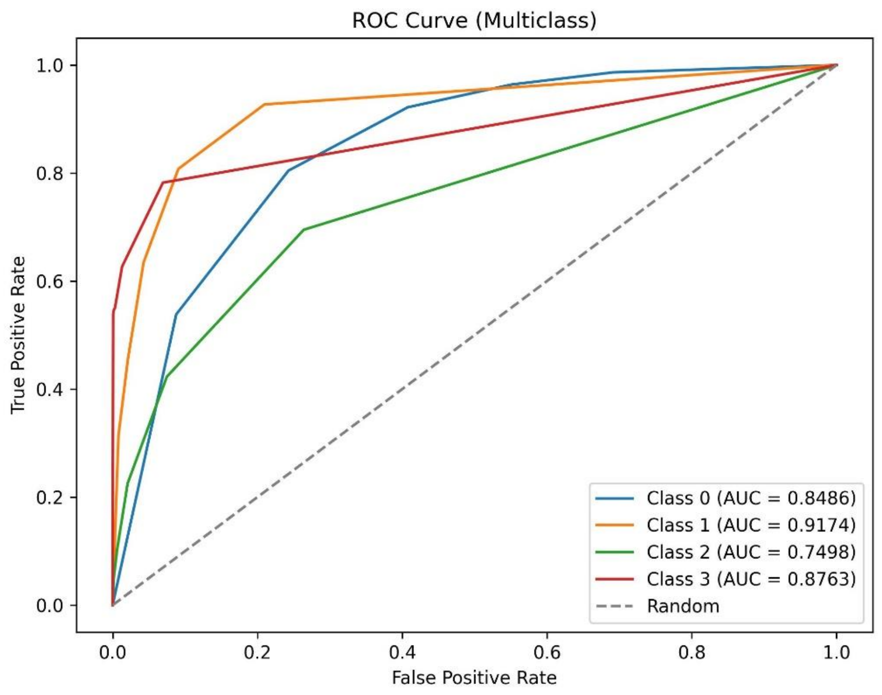
ROC of Random Forest Model



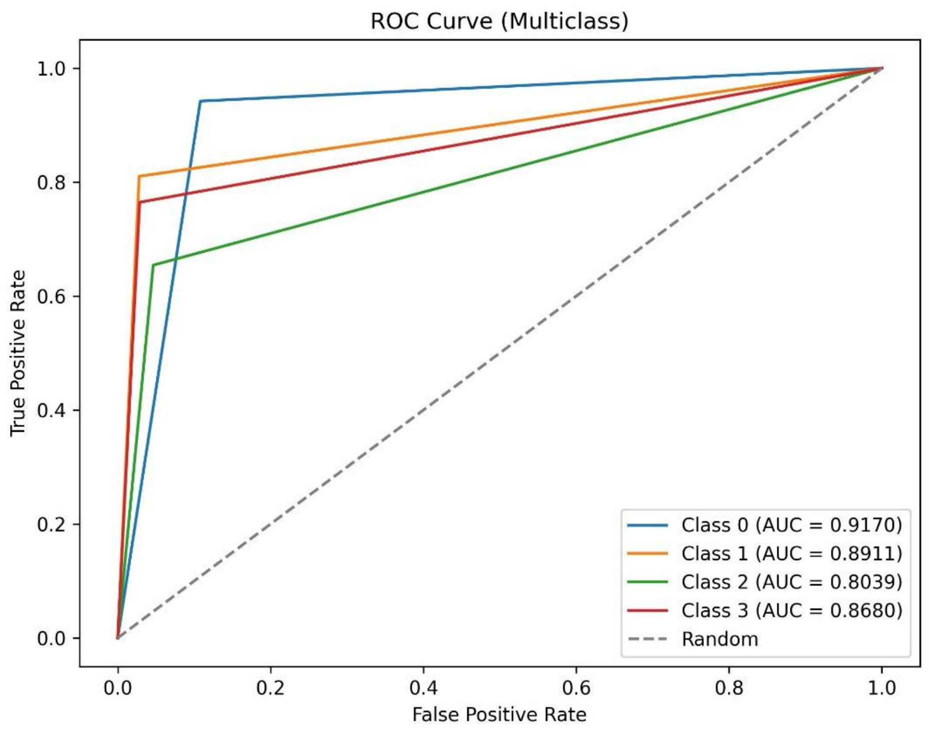
ROC of AdaBoost Classifier



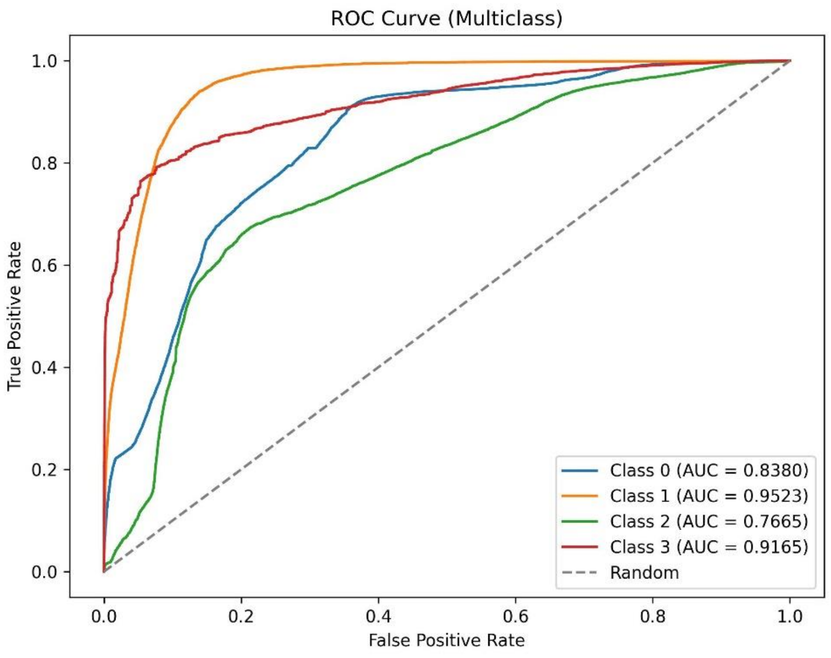
ROC of Naïve Bayes



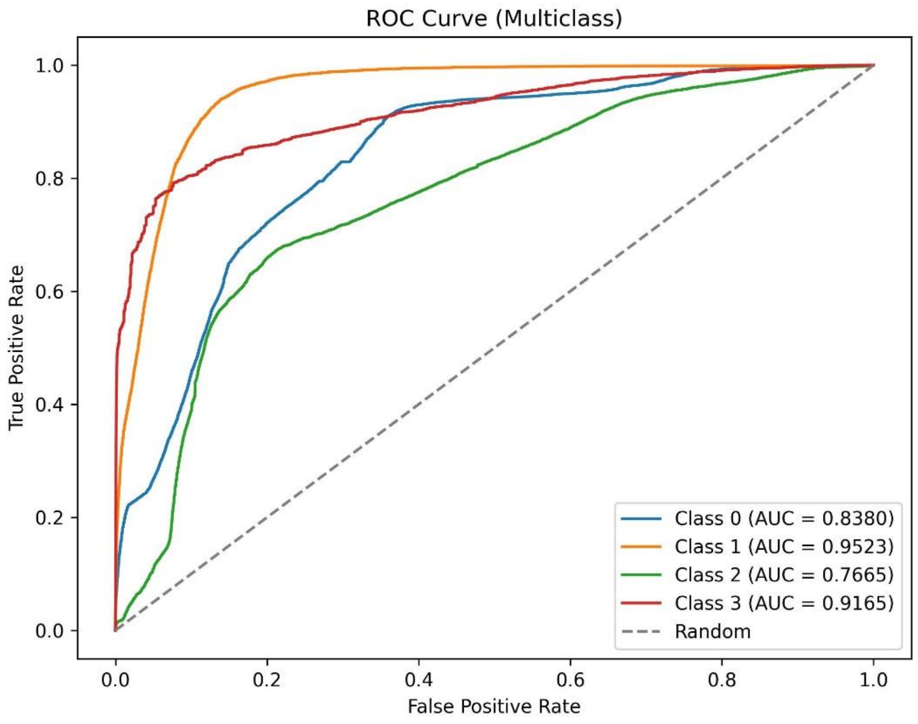
ROC of KNN



ROC of Decision Trees



ROC of Stochastic Gradient Classifier



ROC of Support Vector Classifier