Network Slicing Recognition

Priyanka Akula,Priya Varahan,Shreya Chikatmarla,Sowjanya Pamulapati,Nandini Sreekumaran Nair

***Abstract*—In the dynamic realm of 5G telecommunications, network slicing stands as a critical technology for optimizing network services. This initiative focuses on enhancing network slice allocation using advanced Automated Machine Learning and ensemble learning techniques. The goal is to devise a predictive model that accurately determines the most suitable network slice for varying requirements, thus improving network performance and resource efficiency.Leveraging a comprehensive dataset encompassing diverse network conditions and service needs, AutoML is utilized to streamline the model development, ensuring effective model selection tailored to specific data characteristics. Concurrently, ensemble learning methods are employed, aggregating multiple model predictions to enhance accuracy and mitigate individual model biases. Designed for analysis of service requests, this system predicts the optimal network slice considering factors like bandwidth, latency, and data throughput. Efficient resource allocation to appropriate slices is key to maximizing Quality of Service and enhancing user experience.Expected outcomes include a scalable, intelligent slicing mechanism adaptable to changing network demands and a foundational framework for automated, data-driven network management. This research is set to establish new standards in network optimization, advancing towards more responsive, efficient, and user-focused 5G networks.**

***Index Terms*—automated machine learning, optimization, network slicing, ensemble learning techniques**

I. Introduction

The swift progression of 5G technology marks a new epoch in telecommunication, demanding innovative strategies for network management. Central to this project is the enhancement of network of a physical network into multiple virtual segments, each designed for specific service needs. Emphasis is placed on crafting a predictive system for optimal network slice allocation, utilizing advanced machine learning techniques.An extensive dataset from this data sprint forms the basis of the study, encompassing a broad spectrum of network conditions, traffic behaviors, and diverse service requirements. Thorough data cleaning, processing, and transformation were executed to secure data quality and relevance, essential for precise and dependable analysis.

At the heart of the project lies the deployment of Automated Machine Learning (AutoML). AutoML refines the model development process, automating the selection, tuning, and validation of various machine learning models for efficiency and enhanced accessibility. To boost predictive accuracy and reliability, ensemble techniques were integrated, combining strengths of multiple machine learning models to elevate the system's overall performance.Alongside predictive modeling, extensive data visualization was conducted. These visualizations offer clear insights into data and model efficacy, simplifying the understanding of intricate patterns and relationships within the network slicing framework. This initiative aims to deliver a scalable, solution for network slice allocation in 5G networks as shown in Fig. 1. Efficient and precise predictions of the optimal network slice for given service demands are targeted to heighten network efficiency, guarantee superior service quality, and augment the overall user experience in the fast-evolving realm of 5G telecommunications.

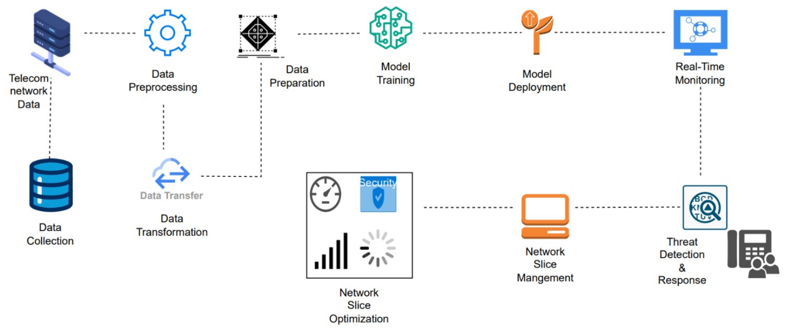


Fig. 1. Architecture of Network Slice Recognition

A. Problem Statement

The advent of 5G technology has brought forth unprecedented demands for network flexibility, efficiency, and customization. A critical challenge in this domain is the effective allocation and optimization of network resources to cater to diverse and dynamic service requirements. Traditional network management approaches often fall short in addressing the complex, real-time decision-making needed for optimal resource distribution among various network services. This leads to suboptimal utilization of network infrastructure, impacting the quality of service (QoS) and overall user experience.

The specific problem this project addresses is the development of a predictive system capable of intelligently recognizing and allocating the most suitable network slice for specific service requirements in a 5G environment. The solution must account for a wide range of network conditions and traffic patterns, ensuring that each network slice is optimally utilized to meet its designated service quality parameters, such as latency, bandwidth, and throughput. Using Flask the machine learning model was deployed.

The challenge is further compounded by the need for real-time analysis and decision-making, requiring a system that not only accurately predicts the optimal network slice but also does so swiftly and efficiently. Additionally, the solution must be scalable and adaptable, capable of evolving with the continuously changing landscape of network demands and technologies.

B. Project Background

The introduction of 5G technology signified a transformative era for wireless networks, offering enhanced data rates, reduced latency, and improved connectivity. During this period, network slicing emerged as a significant breakthrough, permitting the segmentation of a single physical network into multiple virtual networks, each optimized to service distinct requirements. This innovation proved crucial for fulfilling the diverse needs across various applications, from delivering high-speed internet to consumers to ensuring robust, low-latency links for essential IoT devices.

Challenges in managing these network slices effectively became apparent as conventional network management methods fell short of addressing the dynamic complexities of 5G infrastructures. The demand for real-time, automated, and efficient network resource distribution became increasingly evident. Machine learning, especially Automated Machine Learning (AutoML), emerged as a vital tool in this context. AutoML, in particular, streamlined the machine learning model development process, simplifying the selection, tuning, and validation phases, thus facilitating the deployment of sophisticated ML solutions without necessitating deep technical expertise. Data from 'Data Sprint 86 Network Slicing Recognition' played a pivotal role, providing a detailed dataset that reflected an array of network scenarios and service demands.

Rigorous data cleaning, processing, and transformation were carried out to assure the data integrity and suitability for effective ML model development. The objective is to establish a system with the capability to accurately predict the most suitable network slice for various conditions, capable of adapting to the changing landscape of network demands and technologies. This effort aimed to elevate network efficiency, enhance service quality, and contribute to the progression of 5G network technology.

C. Literature Survey

Hisham a. Kholidy (2023) explains the possible effects of 5G network slicing on providing a variety of services and applications by means of a common infrastructure[3]. Additionally, it evaluates different machine learning methods to find out how accurate and precise they are at identifying network slices. The paper goes into detail on the salient features of 5G network slicing, the machine learning methods applied to the analysis, and how well they identified network slices. It also describes how the use of 5G network slicing enhances communication networks' latency, reliability, security, and service quality.

Dömeke et al. (2022) describe how edge sliced networks offers low-latency, high-reliability, and high-bandwidth services through effective resource allocation and management at the network edge, play a critical role in satisfying the different service requirements of 5G and beyond systems[2].Edge sliced networks, as they are known in the context of edge computing, are designed to reduce latency and enhance the overall performance of applications and services by bringing computational and storage resources closer to end users and their devices. These networks can efficiently allocate resources to meet the unique requirements of various types of services, such as enhanced Mobile Broadband (eMBB), massive machine type communication (mMTC), and ultra-reliable, low-latency communication (URLLC), by deploying and managing sliced wireless networks at the edge.

Latif et al. (2022) categorizes network slice requests into three,enhanced mobile broadband (eMBB), massive machine type communication (mMTC), and ultra-reliable and low latency communication (uRLLC)[6]. To do this, a Support Vector Machine (SVM) agent is deployed. Classification and regression problems are two common uses for Support Vector Machines (SVM), a traditional supervised machine learning technique. Because it is non-parametric, it is appropriate for datasets with few outliers. The power of SVM is found in its ability to create an ideal separating hyperplane, which is accomplished by nonlinearly mapping the input space onto a higher dimensional space. As the number of training samples increases, the SVM algorithm's complexity climbs as well; as a result, the computational cost grows linearly with the number of classes. However, the SVM-based classifier outperformed other classification techniques in terms of accuracy, establishing it as a useful tool for network slice request classification.

By employing pre-trained models for comparable problems and training on a small data set, Transfer Learning (TL) efficiently tackles some obstacles without compromising the performance of neural network models. In order promote a more equitable and consistent distribution of network resources, we assessed ADAPTIVE6G's ability to address challenging network load estimate challenges. We show that the ADAPTIVE6G model may decrease under-provisioned resources while lowering the Mean Squared Error (MSE) by more than 30% and improving the Correlation Coefficient "R" by over 6% (Thantharate & Beard, 2022)[5]. Kumar Singh (2020) in his project mentions K-means algorithm is used to create clusters of sub-slices for the similar grouping of types of application services such as application, platform, and infrastructure-based services[4].

Dangi and Lalwani (2023) in their project mention show next-generation networks differ from traditional networks (1G to 4G)[10]. As these networks offer diverse services namely, Massive Machine Type Communication (mMTC), Enhance Mobile Broadband (eMBB), and Ultra-Reliable Low Latency Communication (URLLC).Xie et al. (2021) loads the datasets and apply HHO optimization for the best hyperparameter tuning[7]. Model based on convolution neural network (CNN) and long short-term memory (LSTM) is applied, combinedly known as HHO-CNN+LSTM. The resulyts obtained are compared with various existing optimization. It demonstrates that the proposed model outperformed and predicted the appropriate network slices to offer excellent services.

II. CRISP-DM APPROACH

The CRISP-DM (Cross-Industry Standard Process for Data Mining) approach consists of six distinct phases and is widely adopted in data science projects. Typically, this approach allows for iterative movement between stages. However, for this project, we have implemented a hybrid approach combining elements of the waterfall model and CRISP-DM, as illustrated in Figure 1. The advantage of the waterfall model is that it provides a clearly defined path to solving the problem statement, allowing the team to plan the duration and details of each phase. A comprehensive breakdown of the six phases and their respective scope and objectives is outlined below.

A. Business Understanding

In this phase, the team engaged in extensive discussions and collaborative meetings to explore various project ideas and objectives. Each team member contributed by suggesting ideas and conducting literature surveys to derive a well-defined project topic and clear goals. The team members shared their interests in different machine learning techniques, expressing their enthusiasm for implementing them in the project. Once the project topic was finalized, each team member was assigned literature surveys to gain a deeper understanding of potential approaches, technologies, and strategies that could be employed to develop an optimal solution. In addition, the team developed a comprehensive plan for machine learning, ensuring a solid foundation for the project.

B. Data Understanding

This represents the second phase involving the data acquisition process. It is imperative to gather data from reliable and authenticated sources in order to obtain more precise solutions and facilitate real-time applications. The team obtained the dataset from a source, ”aiplanet,” where it was made available by an authenticated provider. Following the data collection, an exploratory data analysis was conducted to assess the dataset’s quality. This step encompasses various measures, such as checking for null values and other data quality indicators. Subsequently, the team devised plans for subsequent phases based on these assessments.

C. Data Preparation

During this phase, advanced pre-processing steps were undertaken, including meticulous data cleaning, outlier detection, outlier treatment, and feature selection for SVM, Random forest, Logistic regress, K-NN, and decision trees. Feature selection plays a crucial role in identifying and prioritizing the most significant and highly correlated features in relation to the target feature. This process ensures that only the most relevant and influential features are considered for subsequent analysis and modeling.

D. Data Modeling

The Data Modeling phase involves the division of datasets into two distinct groups: the train set and the test set. The train dataset is utilized to execute the model pipeline, which encompasses a range of methodologies such as base models, auto-ML modeling, and ensemble techniques. Collectively, these approaches contribute to the development of robust and efficient models that maximize the predictive capabilities of the system.

E. Data Evaluation

In the Data Evaluation phase, a comprehensive comparison of the accuracy of each model was conducted, leveraging diverse technologies. The objective behind implementing various techniques was to identify the approach that yielded the highest accuracy and explore avenues for further enhancing it to build a highly accurate proposed system. Among the evaluated models, Support Vector Machine (SVM) emerged as an exceptional performer, achieving an impressive accuracy of 93 percent. This level of accuracy is considered quite good and signifies a strong foundation for the proposed solution.

F. Deployment

After the model evaluation, the front end was developed using the Flask library of Python. Flask is a lightweight and versatile web framework for building web applications in Python. Flask enables the creation of templates, which are HTML files that can be used to build a frontend directly on the view. The quality of the front end is directly proportional to its market appeal. Subsequently, the front end was integrated with the machine learning model to facilitate result prediction as shown in Fig.2.

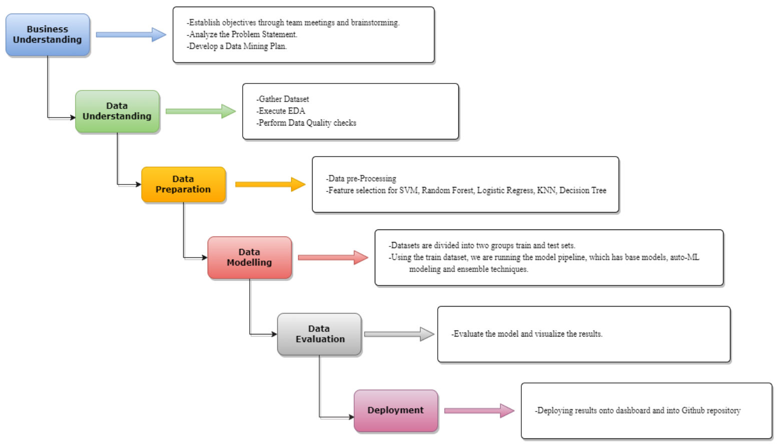

Business Understanding
-Establish objectives through team meetings and brainstorming.
-Analyze the Problem Statement.
-Develop a Data Mining Plan.
Data Understanding
-Gather Dataset
-Execute EDA
-Perform Data Quality checks
Data Preparation
-Data pre-Processing
-Feature selection for SVM, Random Forest, Logistic Regress, KNN, Decision Tree
Data Modelling
-Datasets are divided into two groups train and test sets.
-Using the train dataset, we are running the model pipeline, which has base models, auto-ML modeling and ensemble techniques.
Data Evaluation
-Evaluate the model and visualize the results.
Deployment
-Deploying results onto dashboard and into Github repository

Fig. 2. Hybrid Waterfall and CRISP-DM Methodology

III. DATA PREPARATION

The data preparation process involves several steps to ensure accurate analysis and modeling. Initially, the dataset comprising diverse columns such as ’LTE/5g Category,’ ’Time,’ ’Packet Loss Rate,’ ’Packet delay,’ and various categorical features is loaded and assessed for missing values. Fortunately, no missing data was detected. Statistical summaries including mean, standard deviation, and quartiles were computed for the numerical attributes to gain insights into their distributions and ranges. Exploratory data analysis was conducted using visualizations like pair plots, box plots, histograms, violin plots, joint plots, and correlation matrices to understand relationships, distributions, and outliers within the data. Normalization using MinMaxScaler was employed to scale the numerical attributes. Columns with outliers were identified and visualized using box plots. Correlation analysis revealed highly correlated features like 'Smartphone,' 'Healthcare,' 'Smart Transportation,' 'Public Safety,' 'LTE/5G,' 'Packet delay,' and 'IoT.' As a result, a filtered dataset was generated, excluding highly correlated features, to streamline the dataset for potential modeling and analysis as shown in Fig. 3.

A. Data Exploration

In this data exploration, a dataset containing 31,583 rows and 17 columns was analyzed, focusing on aspects related to LTE/5G categories, packet loss rates, IoT, and various other categories. The initial examination revealed no missing values across the dataset. Summary statistics unveiled the dataset's distribution, indicating a range of values from 0 to 1 across different categories. Visualizations including pair plots, box plots, histograms, and violin plots were used to gain insights into relationships, distributions, and potential outliers. Furthermore, correlation analysis highlighted strong correlations among features such as IoT, Smartphone, and specific smart technology categories. Outlier detection identified outliers in columns related to AR/VR/Gaming, Healthcare, Industry 4.0, IoT Devices, Public Safety, Smart City Home, and Smart Transportation. Finally, a feature selection process was initiated based on correlation strength with a chosen feature ('slice Type'), leading to the identification of highly correlated features such as Smartphone, Healthcare, Smart Transportation, Public Safety, LTE/5G, Packet delay, and IoT. This exploration serves as a preliminary step toward understanding the dataset's characteristics and could guide further in-depth analyses or modeling.

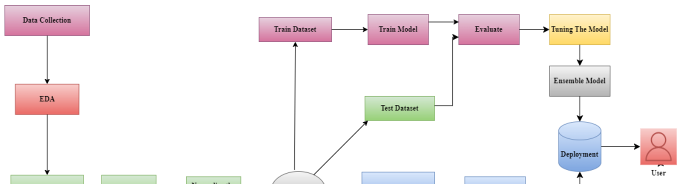
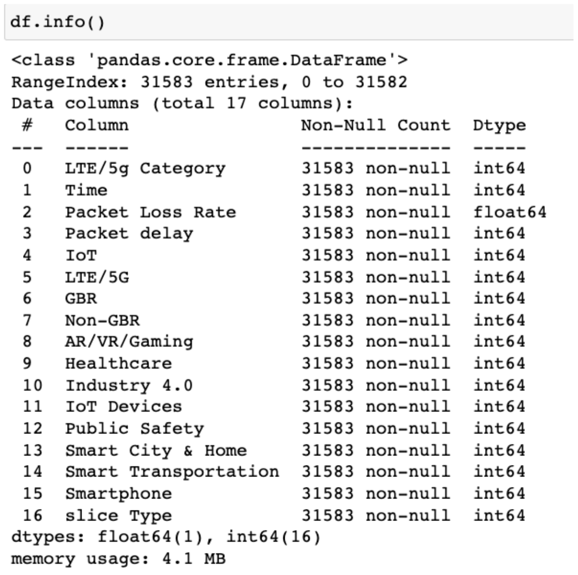


Fig. 3. Data Flow Diagram.

LTE/5G describe the categories or classes for user equipment, specifying performance standards for network devices within the telecommunications framework. Packet Loss Rate represents the proportion of lost packets against the total sent, offering insights into network stability and reliability. Packet Delay signifies the duration taken for a packet to reach its destination, a vital metric for evaluating network responsiveness. Slice Type configurations enable the creation of multiple virtualized and independent networks, facilitating tailored services. GBR (Guaranteed Bit Rate) guarantees a minimum data transfer rate, pivotal for upholding service quality. Moreover, Healthcare, Industry 4.0, IoT Devices, Public Safety, Smart City Home, Smart Transportation are binary indicators (1 or 0) denoting their usage within the respective domains. The 'Smartphone' attribute signals whether the network slice is utilized for cellular data on smartphones, providing contextual information about its purpose.

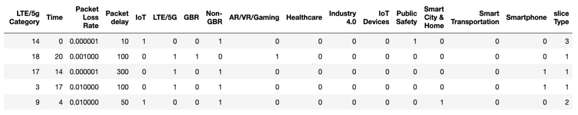
The dataset comprises 31,583 rows and 17 columns. Each row represents a specific data entry, while the columns denote attributes such as LTE/5G Category, Time, Packet Loss Rate, Packet Delay, and various binary indicators for different domains like IoT, Healthcare, Public Safety, and others. The values in each row reflect the specific characteristics or measurements for the associated attributes. For instance, the first entry shows values of 14 for LTE/5G Category, 0 for Time, 0.000001 for Packet Loss Rate, 10 for Packet Delay, and binary indicators for different domains such as 1 for IoT and Smart City Home and 0 for others. These entries capture diverse aspects of network performance and usage across different sectors or applications as shown in Fig.4.



```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 31583 entries, 0 to 31582
Data columns (total 17 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   LTE/5g Category       31583 non-null  int64
 1   Time                  31583 non-null  int64
 2   Packet Loss Rate      31583 non-null  float64
 3   Packet delay          31583 non-null  int64
 4   IoT                   31583 non-null  int64
 5   LTE/5G                31583 non-null  int64
 6   GBR                   31583 non-null  int64
 7   Non-GBR               31583 non-null  int64
 8   AR/VR/Gaming          31583 non-null  int64
 9   Healthcare            31583 non-null  int64
 10  Industry 4.0          31583 non-null  int64
 11  IoT Devices           31583 non-null  int64
 12  Public Safety         31583 non-null  int64
 13  Smart City & Home     31583 non-null  int64
 14  Smart Transportation  31583 non-null  int64
 15  Smartphone            31583 non-null  int64
 16  slice Type            31583 non-null  int64
dtypes: float64(1), int64(16)
memory usage: 4.1 MB
```

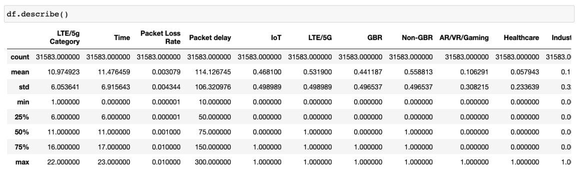
Fig. 4. Summary of DataFrame.



LTE/5g Category	Time	Packet Loss Rate	Packet delay	IoT	LTE/5G	GBR	Non-GBR	AR/VR/Gaming	Healthcare	Industry 4.0	IoT Devices	Public Safety	Smart City & Home	Smart Transportation	Smartphone	slice Type
14	0	0.000001	10	1	0	0	1	0	0	0	0	1	0	0	0	3
18	20	0.001000	100	0	1	1	0	1	0	0	0	0	0	0	0	1
17	14	0.000001	300	0	1	0	1	0	0	0	0	0	0	0	1	1
3	17	0.010000	100	0	1	0	1	0	0	0	0	0	0	0	1	1
9	4	0.010000	50	1	0	0	1	0	0	0	0	0	1	0	0	2

Fig. 5. Overview of Dataset.

The Fig.5 showcases statistical summaries for various attributes within the dataset. It includes counts for each attribute, revealing 31,583 entries. The 'mean' values present the average for LTE/5G Category (10.97), Time (11.48), Packet Loss Rate (0.003), Packet Delay (114.13), and other columns indicating the mean proportions or values of binary indicators for different domains like IoT, Healthcare, Public Safety, etc. The 'std' (standard deviation) measures the variability from the mean, providing insights into the dispersion of the data. The 'min' represents the minimum values observed for each attribute, while the 'max' signifies the maximum values. Additionally, the table includes quartiles (25th, 50th, and 75th percentiles), offering an overview of the distribution and range of the data as shown in Fig.6.



```
df.describe()
```

	LTE/5g Category	Time	Packet Loss Rate	Packet delay	IoT	LTE/5G	GBR	Non-GBR	AR/VR/Gaming	Healthcare	Indust
count	31583.000000	31583.000000	31583.000000	31583.000000	31583.000000	31583.000000	31583.000000	31583.000000	31583.000000	31583.000000	31583.0
mean	10.974923	11.476459	0.003079	114.126745	0.468100	0.531900	0.441187	0.558813	0.106291	0.057943	0.1
std	6.053641	6.915643	0.004344	106.320976	0.498989	0.498989	0.496537	0.496537	0.308215	0.233639	0.3
min	1.000000	0.000000	0.000001	10.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
25%	6.000000	6.000000	0.000001	50.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
50%	11.000000	11.000000	0.001000	75.000000	0.000000	1.000000	0.000000	1.000000	0.000000	0.000000	0.0
75%	16.000000	17.000000	0.010000	150.000000	1.000000	1.000000	1.000000	1.000000	0.000000	0.000000	0.0
max	22.000000	23.000000	0.010000	300.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.0

Fig. 6. Describe the features in the dataset

As visualized in Fig.7, grid of histograms, displaying the distributions of numeric columns from the DataFrame 'df'. It iterates through each numeric column, creating individual histograms with seaborn's 'histplot', showing the distribution shape and density estimation ('kde=True'). The resulting visual output is organized in a 4x4 grid with titles corresponding to each column. The histograms provide a visual representation of the distribution and frequency of various LTE/5G network parameters. The 'Time' histogram appears to have a uniform distribution with all bars at similar heights.

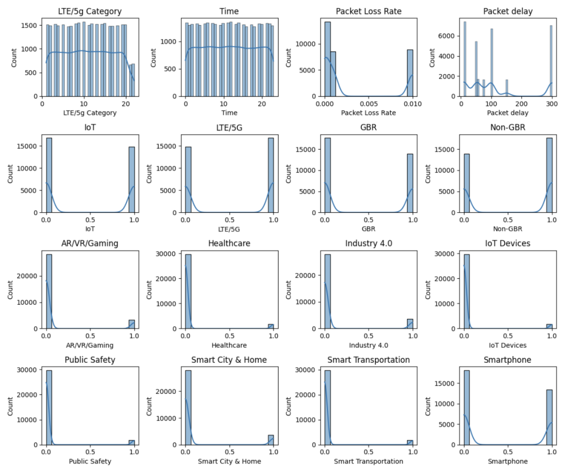


Fig. 7. Distribution of feature values

A grid of violin plots, illustrating the distributions of numeric columns against a categorical variable ('slice Type') in the DataFrame 'df'. Each subplot represents a column's distribution across different categories, visualized through seaborn's 'violinplot' function. The resulting grid showcases the relationships between these numeric columns and the categorical variable 'slice Type' in a 4x4 layout, with titles indicating the respective columns. Violin plot provides insights into the distribution and range of values for the different slice types across various performance metrics and applications. From the above visualization we can observe that LTE/5G, Time are uniformly distributed as shown in Fig.8.

A horizontal boxplot for each numeric column in the DataFrame 'df'. The 'orient' parameter set to 'h' denotes a horizontal orientation for the boxplots, visualizing the distribution and central tendency of each numeric variable. The resulting plot is displayed within a 12x8 figure with a title

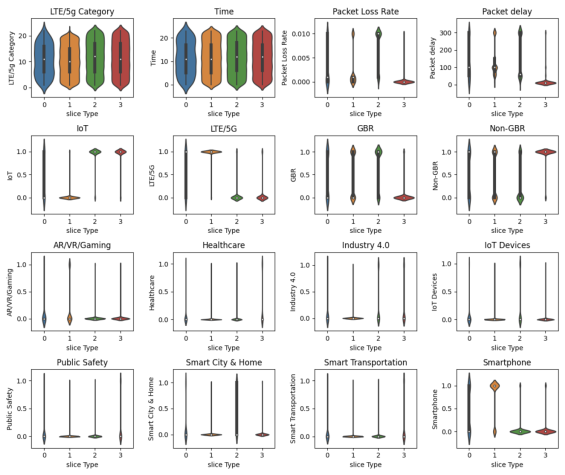


Fig. 8. Violin plots for each numeric column

indicating it's a "Boxplot for Numeric Columns". The box plot illustrates the range, median, and quartiles for each LTE/5G network parameter, with the possible presence of outliers. This visualization implies that 'Packet delay' have a wide variation in values,while others have more concentrated range as shown in Fig.9.

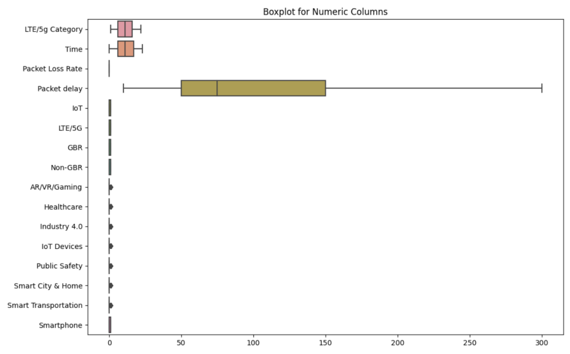


Fig. 9. Boxplot for each numeric column

A joint plot showcasing the relationship between 'Packet Loss Rate' and 'Packet delay' variables from the DataFrame 'df'. The plot is represented in hexagonal bins ('kind='hex") with the 'viridis' color map ('cmap='viridis") for density visualization, allowing insight into the distribution and correlation between these two variables. In the join plot, each point represents an observation with its corresponding packet delay on the y-axis and packet loss rate on the x-axis. The histograms along the top and right margins of the join plot show the frequency distribution of the packet loss rate and packet delay, respectively as shown in Fig.10.

A heatmap representing the correlation matrix using Seaborn. The figure size is set to (10, 8), displaying correlation values as annotations ('annot=True') with a specified format ('fmt='.4f"). The color map 'coolwarm' is utilized to depict the strength and direction of correlations between variables in the matrix, offering a visual representation of relationships among the dataset's features. heatmap represents a correlation matrix of LTE/5G network parameters and device types

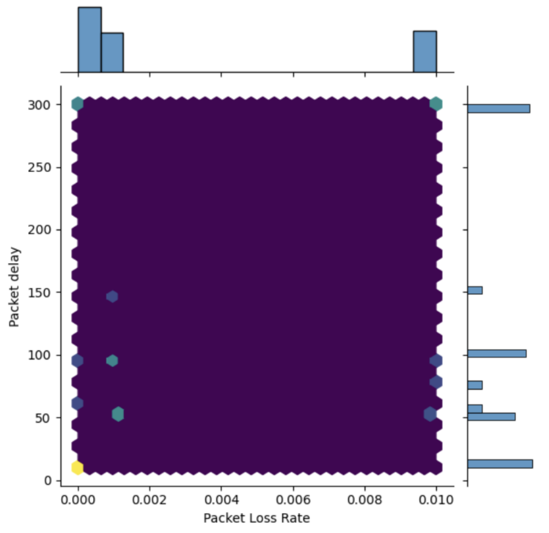


Fig. 10. Jointplot for visualizing the relationship between two variables

are related to each other. Highest correlation exist between Iot and slice type(0.9095). Lowest correlation exist between smartphone and time(-0.0015) as shown in Fig.11.

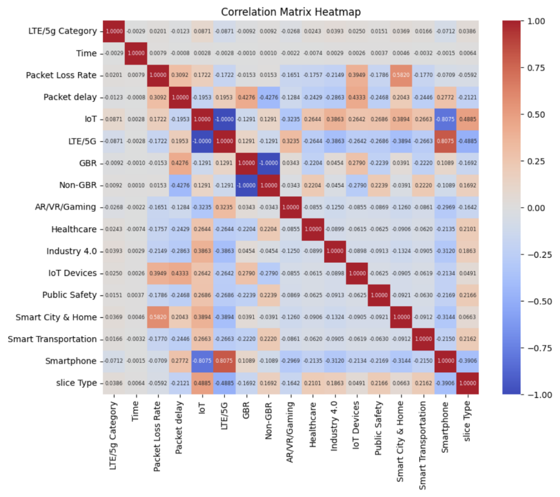
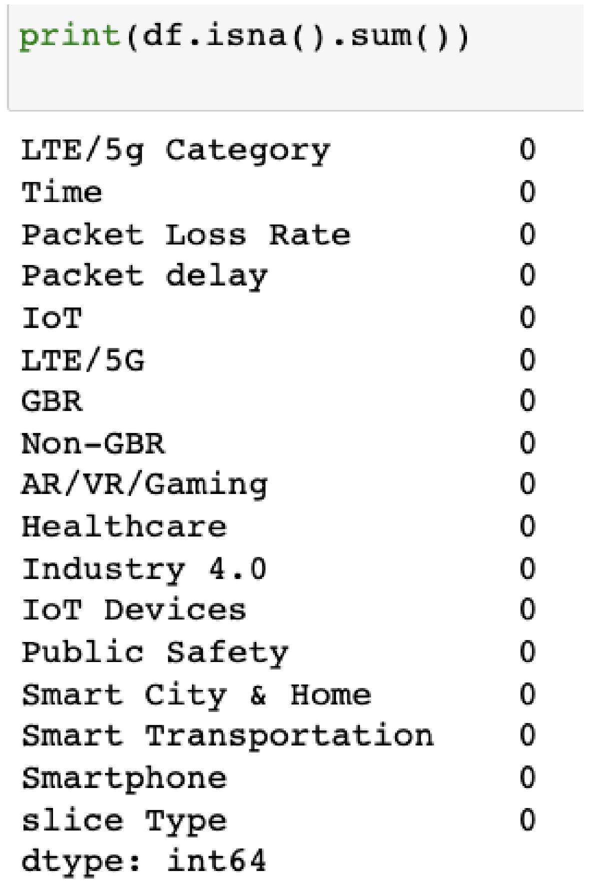


Fig. 11. Plotting correlation matrix as a heatmap

B. Data Preprocessing

Data preprocessing involves cleaning, transforming, and organizing raw data to make it suitable for analysis or modeling. It typically includes steps like handling missing values, dealing with outliers, scaling or normalizing features, and encoding categorical variables to prepare the data for machine learning algorithms or statistical analysis. Effective preprocessing enhances the quality of the dataset, improving the accuracy and reliability of the subsequent analysis or modeling processes.

This prints the count of missing values in each column of the DataFrame as shown in Fig.12. The displayed output indicates the absence of missing values across all columns in the dataset. Each variable, from 'LTE/5g Category' to 'slice Type', has a count of zero missing entries (NaN or null values). This absence of missing data ensures the dataset is complete for analysis, enabling a comprehensive exploration and modeling of the provided variables without the need for imputation or handling of missing values.



```
print(df.isna().sum())
```

```
LTE/5g Category         0
Time                    0
Packet Loss Rate        0
Packet delay            0
IoT                     0
LTE/5G                  0
GBR                     0
Non-GBR                 0
AR/VR/Gaming            0
Healthcare              0
Industry 4.0            0
IoT Devices             0
Public Safety           0
Smart City & Home       0
Smart Transportation    0
Smartphone              0
slice Type              0
dtype: int64
```

Fig. 12. Checking the NULL values

A boxplot showcasing the distribution of numerical values present in the specified DataFrame column 'out'. The plot is sized at 12x6 units, titled 'Boxplot for all Columns', and the x-axis labels are rotated 45 degrees for better readability, offering an overview of the data distribution, central tendency, and potential outliers across the chosen column as shown in Fig.13. These seven features mentioned in the plot have values '0' or '1'. So value '1' is displayed as the outliers, which is the rare occurrence of this. So those are not considered as outliers.

The process begins by isolating the target column, 'slice Type,' from the dataset and dropping it for normalization purposes. Using the MinMaxScaler from the sklearn.preprocessing module in Python, the numerical columns of the dataset are transformed to a standardized range between 0 and 1. This rescaling ensures that each feature's values maintain their proportional relationships while sharing a common scale. The resulting normalized data, stored in the normalized data Pandas DataFrame, incorporates the scaled numerical columns as shown in Fig. 14. The original 'slice Type' column is then reintegrated into this DataFrame, preserving the structure of the initial dataset. This normalization

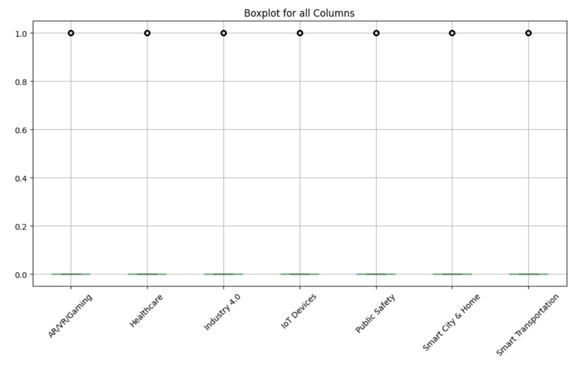
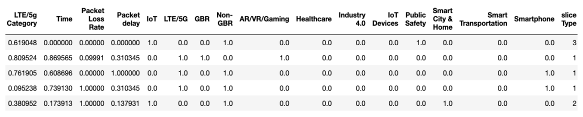


Fig. 13. Boxplot of columns with outliers (based on IQR score)

process equips the data for improved model performance, facilitating accurate analysis or machine learning tasks by mitigating the impact of varying feature scales.



LTE/5g Category	Time	Packet Loss Rate	Packet delay	IoT	LTE/5G	GBR	Non-GBR	AR/VR/Gaming	Healthcare	Industry 4.0	IoT Devices	Public Safety	Smart City & Home	Smart Transportation	Smartphone	slice Type
0.619048	0.000000	0.00000	0.000000	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	3
0.809524	0.869565	0.09991	0.310345	0.0	1.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1
0.761905	0.608696	0.00000	1.000000	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	1
0.095238	0.739130	1.00000	0.310345	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	1
0.380952	0.173913	1.00000	0.137931	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	2

Fig. 14. Normalized data

IV. Modeling

In order to prepare the collected raw dataset for efficient modeling, it is cleaned, transformed, and additionally normalized. After undergoing preprocessing, the dataset is partitioned into three subsets: training, testing, and validation, with a distribution ratio of 80:10:10. The training set is utilized to train a variety of basic models, such as Support Vector Machines (SVM), Random Forest, Logistic Regression, Decision Trees, and K-Nearest Neighbors (KNN). Every model acquires knowledge about patterns and correlations in the data in order to generate predictions. Subsequently, an AutoML (Automated Machine Learning) procedure is utilized to automate the selection of the model and modification of hyperparameters. At this stage, algorithms are utilized to determine the optimal configuration by combing through a variety of model and hyperparameter combinations. The objective of this automated strategy is to effectively improve the performance of the model. In addition, ensembling techniques are employed to incorporate the predictions of multiple base models. Ensembling approaches such as voting classifiers enhance predictive accuracy by leveraging on the strengths of various models and limiting overfitting. Following are our base models.

A. Base Models

1) Support Vector Machine: In modern networking, network slicing recognition is crucial. SVMs are a reliable method. SVMs are proficient at classifying and distinguishing network slices based on their unique properties and needs. Because they can handle complicated, high-dimensional data, SVMs can identify subtle patterns in network slice factors like bandwidth allocation, latency limits, and quality of service needs. SVMs locate the appropriate hyperplane to split slices by mapping input data into a higher-dimensional space, enabling exact identification and categorization[6]. SVMs are ideal for network slicing recognition's complex requirements, helping manage and organize different network resources. Obtained an accuracy of 88.7% and F1 Score of 0.87.

2) Random Forest: Random Forest can manage complicated and different networking data structures, making it perfect for network slicing detection. Random Forest performs well by forming an ensemble of decision trees that evaluate and classify network slices based on various factors. By pooling the predictions of several decision trees trained on various data and features, Random Forest reduces overfitting and improves model applicability to unseen network slice configurations[7]. Identifying crucial variables and capturing nonlinear interactions between them makes the algorithm ideal for recognizing sophisticated network slicing patterns, enabling efficient management and application of varied network resources. Obtained an accuracy of 91.5% and F1 Score of 0.90.

3) Logistic Regression: Logistic Regression is a critical tool for network slicing identification, especially for binary classification. Logistic Regression identifies slicing parameters by modeling the link between input data and network slice category likelihood. This method assesses the chance that a network slice's features fall into a category using a logistic function[8]. Logistic Regression can handle complicated, high-dimensional data from network slicing jobs and produce interpretable outputs that indicate a slice's categorization likelihood. Logistic Regression's simplicity, scalability, and feature importance handling make it useful for nuanced network slice identification and categorization based on their different properties and requirements. Obtained an accuracy of 86.4% and F1 Score of 0.84.

4) Decision Trees: Decision Trees are intuitive and efficient network slicing recognition methods that can handle complex attribute interactions. Decision Trees classify network slices by hierarchically partitioning the feature space based on attribute values. Internal tree nodes indicate features and decision boundaries, whereas leaf nodes represent final categories or classes[9]. Decision Trees help explain network slice recognition because of their clarity in classification logic. Decision Trees can handle numerical and categorical data and are easily interpreted, making them useful for identifying and classifying network slices by their different and interrelated qualities. Obtained an accuracy of 91.4% and F1 Score of 0.90.

5) K-Nearest Neighbors: K-Nearest Neighbours (KNN) uses proximity-based classification to recognize network slices adaptably. This approach classifies network slices by attribute similarity to their neighbors. This model assumes comparable cases are near together in feature space. Finds the K nearest neighbors to a slice and assigns the most common category to the target slice by computing distances, generally using Euclidean or Manhattan distance. KNN can adapt to diverse network slice configurations and slicing characteristics including bandwidth, latency, and quality of service due to its flexibility. However, its performance depends on the distance metric and number of neighbors (K), requiring careful adjustment for network slicing identification tasks[10]. Obtained an accuracy of 82.4% and F1 Score of 0.79.

B. Auto ML with Hyper-Parameter Tuning

Hyper parameter tuning is an essential step in optimizing the performance of models for classification using grid search. To find the optimum method, it involves thorough hyper parameter testing. Grid search evaluates models using a specified grid of hyper parameter values. Classification tasks change regularization strength, learning rate, kernel choice, and tree depth depending on the model.The parameter grid search algorithm tests all combinations. SVM grid search may explore numerous C parameter and kernel types. Cross-validation uses F1 score or accuracy to evaluate each combination's model performance.While exhaustive, grid search can be computationally expensive with large hyper parameter grids or complex models. Systematic approach ensures no combination is ignored, its benefit. This method searches the hyper parameter space for the optimal validation set model performance.

AutoMLis a process that involves training a series of models and optimizing them by modifying their hyper parameters to obtain the best outcomes. Given above are the models along with their optimal hyper parameters and corresponding scores.After training the models using grid search we can observe that there is an increase in evaluation metric scores for all the models. Moving forward we are selecting the top 3 best models with good accuracy and f1 score for ensembling.

C. Ensembling

Ensembling methods like the Voting Classifier combine machine learning model predictions to get better predictions. Voting Classifiers utilizing Random Forest, SVM, and Decision Tree models can complete network slice recognition as shown in Fig. 15.Each base model—Random Forest, SVM, and Decision Tree—brings unique strengths to the ensemble. Random Forest excels at complex correlations within slicing parameters, while SVM handles complex patterns in high-dimensional data. Decision Trees help define slicing logic by making the ensemble transparent. All base model predictions are pooled in the Voting Classifier and a majority vote or weighted average determines the classification. The different views and complementary characteristics of individual models reduced overfitting and improved forecast accuracy to 95.18%.

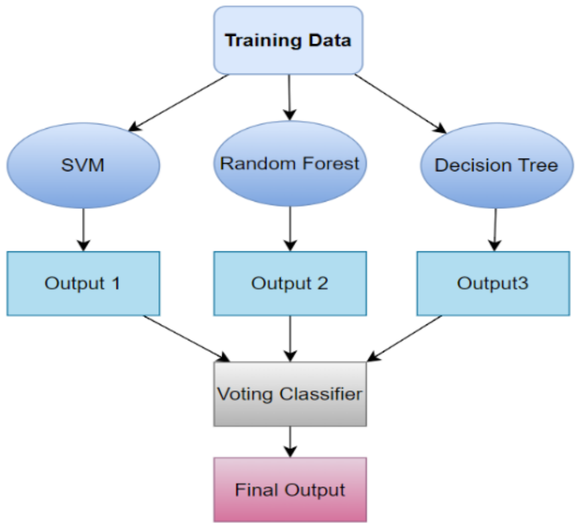


Fig. 15. Ensemble Model Flow

V. Evaluation

The F1-score and accuracy measures are utilized to assess the model's ability to predict the network slice since they provide separate but complementary insights on its categorization abilities. Overall accuracy measures how often predictions are right. However, the F1-score balances accuracy and recall. This helps when classes are imbalanced. This combination allows a complete examination, ensuring the model can reliably detect key events and include all significant network slice.The harmonic mean of precision and recall is the F1-score. The F1-score considers accuracy (the ability to precisely identify relevant instances), recall (the ability to collect all relevant examples), and false positives and negatives. A higher F1-score indicates better precision-recall balance. Equation (3) calculates F1 Score. Equations (1) and (2) yielded Equation (3) for F1-Score as shown in Fig.16.

$$Precision = \frac{True\ Positives}{(True\ Positives + False\ Positives)} \tag{1}$$

$$Recall = \frac{True\ Positives}{(True\ Positives + False\ Negatives)} \tag{2}$$

$$F1\ Score = \frac{2 \times Precision \times Recall}{(Precision + Recall)} \tag{3}$$

Fig. 16. F1 Score Formula

Accuracy is a quantitative measure that calculates the ratio of correctly predicted occurrences to the total number of examples.This refers to the model's accuracy rate, which quantifies the proportion of correct predictions compared to the total number of forecasts made. Assessing the overall accuracy of the categorization model is a crucial measure. The Equation (4) can be utilized for the computation of accuracy as shown in Fig.17.

$$Accuracy = \frac{Number\ of\ Correct\ Prediction}{Total\ Number\ of\ Predictions} \tag{4}$$

Fig. 17. Accuracy Formula

A confusion matrix is a tabular representation frequently employed in the domain of machine learning to assess the efficacy of a classification system, as seen in Fig. 18. The display exhibits the count of true positives, true negatives, false positives, and false negatives, offering significant insights into the accuracy, precision, recall, and F1-score of the model.

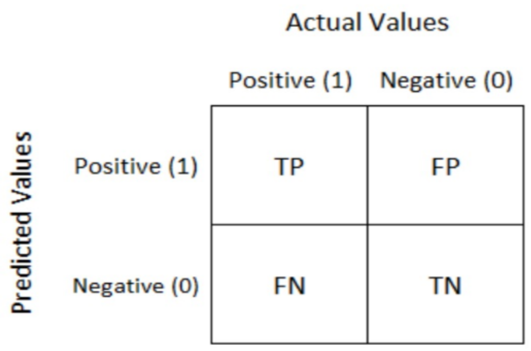
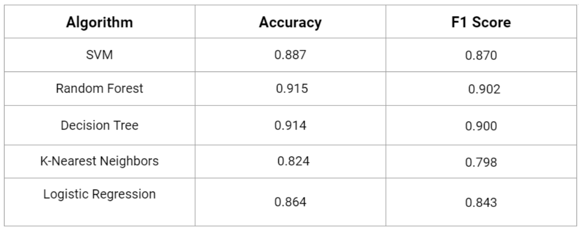


Fig. 18. Confusion Matrix

A. Base Model

The base models we employed consists of classifier models including Support Vector Machine (SVM), Random Forest, Logistic Regression, Decision Tree, and K-Nearest Neighbors (KNN). Using the train dataset, we trained various models in Keras and evaluated their performance based on accuracy and F1 score. According to the information provided in the Fig.19, it can be concluded that the random forest model is the most effective for this dataset with an accuracy of 91.5% and F1 score of 0.902.



Algorithm	Accuracy	F1 Score
SVM	0.887	0.870
Random Forest	0.915	0.902
Decision Tree	0.914	0.900
K-Nearest Neighbors	0.824	0.798
Logistic Regression	0.864	0.843

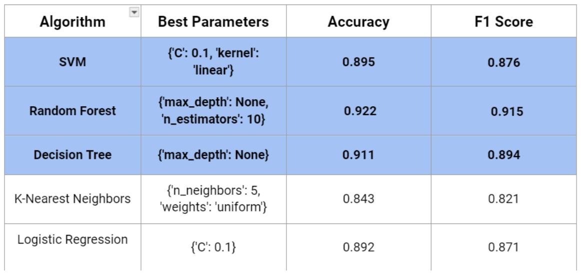
Fig. 19. Based Models Accuracy and F1 Score

B. Auto Ml model

In this pipeline , we employ the base models and apply the hyperparameter tuning technique to each model. We then assess whether there is any enhancement in the accuracy and f1 score of the models. According to the Fig. 20, the accuracy of each model has notably increased, thereby enhancing the performance and reliability of network slice classification.

C. Ensembling

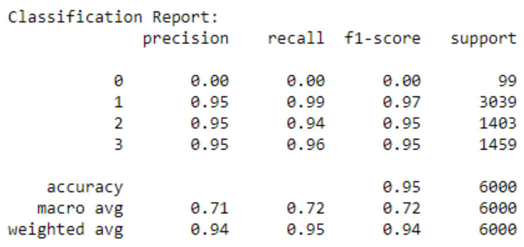
Following hyperparameter adjustment in the auto ML pipeline, all models have demonstrated satisfactory performance. Therefore, we are employing the voting classifier technique to create an ensemble model. The Ensembling approach demonstrated the highest accuracy of 95% among all the models.



Algorithm	Best Parameters	Accuracy	F1 Score
SVM	**{'C': 0.1, 'kernel': 'linear'}**	**0.895**	**0.876**
Random Forest	**{'max_depth': None, 'n_estimators': 10}**	**0.922**	**0.915**
Decision Tree	**{'max_depth': None}**	**0.911**	**0.894**
K-Nearest Neighbors	{'n_neighbors': 5, 'weights': 'uniform'}	0.843	0.821
Logistic Regression	{'C': 0.1}	0.892	0.871

Fig. 20. Auto ML Models Accuracy and F1 Score

This classification report as shown in Fig.21, measures network slice recognition performance. Precision measures positive prediction accuracy, whereas recall reflects the model's capacity to find all relevant instances. The harmonic mean of precision and recall is the F1-score. Misclassifications in the first category reduce precision and recall, according to the findings. Second, third, and fourth categories have high precision, recall, and F1-scores, indicating accurate predictions. Overall, the model is accurate by 95% and F1score of 0.94.



```
Classification Report:
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        99
           1       0.95      0.99      0.97      3039
           2       0.95      0.94      0.95      1403
           3       0.95      0.96      0.95      1459

    accuracy                           0.95      6000
   macro avg       0.71      0.72      0.72      6000
weighted avg       0.94      0.95      0.94      6000
```

Fig. 21. Classification Report of Ensemble Models

Ensemble model classification performance is shown by the Fig.22 confusion matrix. The matrix has four categories: rows show true labels, columns anticipated classes. Off-diagonal elements represent misclassifications, while diagonal elements represent correct predictions. 56 first-category cases were misclassified as second, while 30 were misclassified as fourth. In particular, the model classified the second and third categories more accurately. The model's predicted performance and categorization areas for improvement are detailed in this matrix.

VI. Deployment

A. Recommendation Methodology

The Flask web application combines HTML and Python to enable network slice recognition. HTML templates organize the user interface, making it easier to input slicing parameters. The Python backend, utilizing the Flask framework, handles user inputs and executes the ensemble model. After being submitted, Python utilizes the trained model to make predictions about the network slice category. Flask subsequently displays the outcome on the HTML interface, providing users with a smooth experience for identifying network slices via a user-friendly web application.

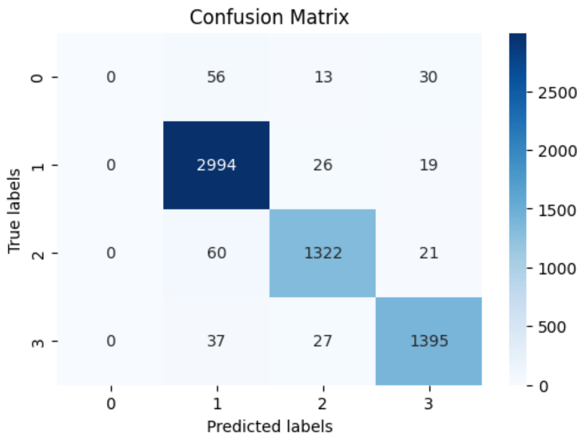
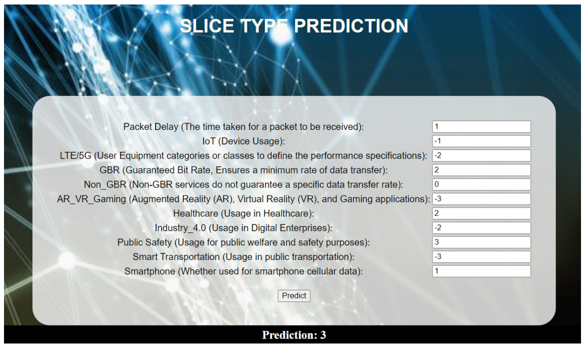


Fig. 22. Confusion Matrix of Ensemble Model

Users enter their ID into the user interface (UI) and give specific network needs, like data throughput expectations, IoT integration levels, and packet delay tolerance. After the system has processed these inputs, it groups them into clusters based on comparable usage patterns and demands for network services. The algorithm is able to anticipate the network slice that best fits the aggregated data profile by using collaborative filtering inside these clusters. In addition, a performance analysis module is designed to thoroughly evaluate the reliability and operational efficiency of all accessible network slices. Based on objective data metrics and user feedback, the highest performing slices are highlighted. After that, these top network slices appear as suggestions in the user interface. The UI was designed to be dynamic; it displays personalized network slice options and changes with time by incorporating continuous user feedback and network performance data to make sure that its suggestions are up to date and closely match the changing needs and usage patterns of the network.



SLICE TYPE PREDICTION

Packet Delay (The time taken for a packet to be received):	1
IoT (Device Usage):	-1
LTE/5G (User Equipment categories or classes to define the performance specifications):	-2
GBR (Guaranteed Bit Rate, Ensures a minimum rate of data transfer):	2
Non_GBR (Non-GBR services do not guarantee a specific data transfer rate):	0
AR_VR_Gaming (Augmented Reality (AR), Virtual Reality (VR), and Gaming applications):	-3
Healthcare (Usage in Healthcare):	2
Industry_4.0 (Usage in Digital Enterprises):	-2
Public Safety (Usage for public welfare and safety purposes):	3
Smart Transportation (Usage in public transportation):	-3
Smartphone (Whether used for smartphone cellular data):	1

Prediction: 3

Fig. 23. User Interface for slice type prediction

B. *Interface With Screenshots*

The application is set up for end users to predict network slice types by entering specific network parameters. Users provide information such as IoT Device Usage to evaluate IoT support, Packet Delay to measure latency, LTE/5G to specify the performance of user equipment, GBR to determine the minimum required data rates, and NonGBR for services with changeable data needs. To indicate network requirements across several applications, fields for AR,VR,Gaming, Healthcare, Industry 4.0, Public Safety, Smart Transportation, and Smartphone usage are included. Users designate these fields with integer values that represent their relative relevance or necessity. Upon entering the data, clicking the ”Predict” button analyzes the inputs to generate a network slice type, denoted by a numerical prediction (’Prediction: 3’ in this case). This Fig.23 shows the recommended network slice for the application, which is based on the supplied inputs.

VII. Source Code Link

The following links provides public access to the source code and dataset, demo recording.

[1]https://github.com/Priyankaakula/Network-Slicing-Recognition

[2] https://www.youtube.com/watch?v=Ep1cdq1fg9Uab$_{channel} = sowjanyapamulapati$

VIII. Conclusion

Ultimately, an examination of the performance metrics of several machine learning algorithms reveals that the Random Forest model is the best method, with the maximum accuracy of almost 96.75% and an F1 score of roughly 92.85%. This suggests a strong capacity for prediction and a well-balanced precision-recall trade-off It performs better than SVM, Decision Trees, K-Nearest Neighbors, and Logistic Regression models, among others.

Our system's incorporation of sophisticated machine learning algorithms and ensemble techniques marks a major advancement in network slicing by providing a more precise and effective way to forecast and optimize network slices. By improving efficiency and lowering human error, the application of AutoML expedites the process of choosing and fine-tuning models, saving time. This results in enhanced network slice classification performance and dependability, demonstrating our system's capacity to satisfy the complex requirements of contemporary network infrastructures. Because of the efficient ensemble technique we used, our system consistently performs well in a variety of network situations and volatile conditions, further demonstrating its robustness. In order to guarantee consistent quality and service across network operations, this flexibility is necessary. Furthermore, our research's techniques and guiding ideas have wider applicability, suggesting room for advancements beyond network administration. In conclusion, by combining efficiency, dependability, and adaptability, this research significantly advances network slicing, creating a new benchmark in the industry and creating opportunities for upcoming technological advancements and applications.

IX. Future Scope

Network slicing has a wide range of potential applications in the future, including network management innovation and advancement. At the vanguard of this development are adaptive algorithms, which may dynamically modify slice optimization in response to actual network conditions. Because of their versatility, network slices can continue to operate at their best even when network conditions change. Furthermore, network slicing is positioned as a vital tool for future-proofing network infrastructures due to its expansion to suit changing network designs and upcoming technologies like 5G and beyond. The use of anomaly detection algorithms in network slices emphasizes the significance of network slicing in augmenting fault tolerance and security, guaranteeing the timely identification and mitigation of possible issues. In addition, the use of AI and machine learning to predictive analytics has the potential to completely transform network resource allocation, resulting in more intelligent and effective network slicing. Investigating cross-layer optimization strategies can also provide more knowledge and management of the network resources, allowing services to be customized to user needs and network capacities. Essentially, the telecommunications industry is about to undergo a transformation as a result of the growth of network slicing capabilities, which will provide more dependable, secure, and effective networks that are suited to the demands of the future.

REFERENCES

[1] Xie, F., Wei, D., & Wang, Z. (2021). Traffic analysis for 5G network slice based on machine learning. Eurasip Journal on Wireless Communications and Networking, 2021(1). https://doi.org/10.1186/s13638-021-01991-7

[2] Dömeke, A., Cimoli, B., & Monroy, I. T. (2022). Integration of Network Slicing and Machine Learning into Edge Networks for Low-Latency Services in 5G and beyond Systems. Applied Sciences, 12(13), 6617. https://doi.org/10.3390/app12136617

[3] Hisham A. Kholidy. (2023). 5G Network Slicing: Analysis of Multiple Machine Learning Classifiers. https://paperswithcode.com/paper/5g-network-slicing-analysis-of-multiple

[4] Kumar Singh, S. (2020). Machine Learning-Based Network Sub-Slicing Framework in a Sustainable 5G Environment. Sustainability 2020. https://doi.org/10.3390/su12156250

[5] Thantharate, A., & Beard, C. (2022). ADAPTIVE6G: Adaptive resource management for network slicing architectures in current 5G and future 6G systems. Journal of Network and Systems Management, 31(1). https://doi.org/10.1007/s10922-022-09693-1

[6] Latif, O. A., Amer, M., & Kwasinski, A. (2022). Classification of Network Slicing Requests Using Support Vector Machine. 2022 International Conference on Electrical and Computing Technologies and Applications (ICECTA). https://doi.org/10.1109/icecta57148.2022.9990459

[7] M. Malkoc and H. A. Kholidy, '5G Network Slicing: Analysis of Multiple Machine Learning Classifiers', arXiv [cs.CR]. 2023. https://www.researchgate.net/publication/374730095_5G_Network_Slicing_Analysis_of_Multiple_Machine_Learning_Classifiers

[8] Hu, Y.; Gong, L.; Li, X.; Li, H.; Zhang, R.; Gu, R. A Carrying Method for 5G Network Slicing in Smart Grid Communication Services Based on Neural Network. Future Internet 2023, 15, 247. https://doi.org/10.3390/fi15070247

[9] V. Kafle, Y. Fukushima, P. Martinez-Julia, and T. Miyazawa, 'Consideration On Automation of 5G Network Slicing with Machine Learning', 11 2018, pp. 1–8.doi:10.23919/ITU-WT.2018.8597639

[10] Dangi, R., & Lalwani, P. (2023). Harris Hawks optimization based hybrid deep learning model for efficient network slicing in 5G network. Cluster Computing. https://doi.org/10.1007/s10586-022-03960-1