A purple and black rectangular sign

Description automatically generated

AI (ITRI626) ASSIGNMENT 1

**B Leotlela**

**34854304**

Table of Contents

[**1.** **INTRODUCTION** 1](#_Toc144328756)

[**2.** **MATERIALS** 1](#_Toc144328757)

[**3.** **METHOD** 2](#_Toc144328758)

[3.1. **Naïve Bayes** 2](#_Toc144328759)

[3.1.1. HOW IT WORKS 2](#_Toc144328760)

[3.1.2. STRENGTHS 2](#_Toc144328761)

[3.1.3. LIMITATIONS 3](#_Toc144328762)

[**3.2.** **K-Nearest Neighbour** 3](#_Toc144328763)

[3.2.1. HOW IT WORKS 4](#_Toc144328764)

[3.2.2. STREGNTHS 4](#_Toc144328765)

[3.2.3. LIMITATIONS 5](#_Toc144328766)

[**3.3.** **Support Vector Machine** 5](#_Toc144328767)

[3.3.1. HOW IT WORKS 5](#_Toc144328768)

[3.3.2. STRENGTHS 6](#_Toc144328769)

[3.3.3. LIMITATIONS 6](#_Toc144328770)

[**3.4.** **Training models** 6](#_Toc144328771)

[**4.** **RESULTS AND DISCUSSIONS** 7](#_Toc144328772)

[**4.1.** **Results** 7](#_Toc144328773)

[4.1.1. PERFORMANCE RESULTS FOR NAÏVE BAYES, KNN AND SVM 7](#_Toc144328774)

[**4.2.** **Discussion** 9](#_Toc144328775)

[**4.3.** **Related Work** 9](#_Toc144328776)

[**5.** **CONCLUSION** 10](#_Toc144328777)

[**6.** **REFERENCES** 11](#_Toc144328778)

# Table of Figures

[Figure 1: The Naive Bayes Classifier (Analytics Learn) 2](#_Toc144332597)

[Figure 2: Overview of KNN (Boateng et al., 2020:346) 4](#_Toc144332598)

[Figure 3: illustration of SVM in 2-D (Boateng et al., 2020) 5](#_Toc144332599)

[Figure 4: Overview of training process 7](#_Toc144332600)

[Figure 5: Flowchart of Naive Bayes 7](#_Toc144332601)

[Figure 6: Flowchart of KNN model 8](#_Toc144332602)

[Figure 7: Flowchart of SVM 8](#_Toc144332603)

[Figure 8: ROC Curve for Naive Bayes 9](#_Toc144332604)

[Figure 9: ROC Curve for KNN 10](#_Toc144332605)

[Figure 10: ROC Curve for SVM 10](#_Toc144332606)

# List of Tables

[Table 1: Results of metrics by which models were evaluated 9](#_Toc144332635)

# **INTRODUCTION**

In the dynamic landscape of modern technology, the field of machine learning has emerged as a transformative force, enabling computers to learn from data and make intelligent decision without explicit programming. One of the fundamental applications of machine learning is classification, a process that involves discerning intricate patterns within data to predict categorical outcomes. This report delves into the realm of classification, where the convergence of data scienve and machine learning yields powerful tools to unravel complex relationships within datasets.

At its core, machine learning refers to the art of training algorithms to recognize patterns within data and use those patterns to make predictions or decisions. It encompasses a diverse array of techniques that allow computers to autonomously adapt and improve their performance over time, all by learning from the information they encounter. The fundamental aim is to extract meaningful insights from data, transform them into actionable knowledge, and ultimately facilitate informed decision-making.

The primary goal of this report is to apply and assess the efficacy of three distinct machine learning algorithms: K-Nearest Neighbours (KNN), Naïve Bayes, and Support Vector Machines (SVM), in tackling a categorical prediction problem. Each of these algorithms embodies a unique philosophy of learning from data, ranging from nearest neighbour interactions to probabilistic inference and geometric hyperplane optimization.

By training each algorithm and evaluating their performance using classification metrics including accuracy, precision, recall, F1-score, Receiver Operating Characteristic (ROC) curve, and the Area Under the Curve (AUC), the strengths and limitations of each algorithm in the context of the chosen dataset will be unveiled.

# **MATERIALS**

Here the dataset that has been chosen for this report is described. The dataset chosen for this report is the Iris dataset, which can be found at <https://archive.ics.uci.edu/dataset/53/iris> . It is a well-known and frequently used dataset. This dataset was introduced by British biologist and statistician Ronald A. Fisher in 1936. The dataset consists of measurements of three different species of iris flowers. Each species is represented by 50 samples, making a total of 150 samples in the dataset.

The Irish dataset contains four features (the attributes or variables) that represent different measurements of the iris flowers’ floral parts, and one categorical target variable (the class label) that indicates the species of each iris flower. The four features are as follows:

1. Sepal Length (cm): The length of the iris flower’s sepal, which is the outermost whorl of the flower.
2. Sepal Width(cm): The width of the iris flower’s sepal.
3. Petal Length (cm): The length of the iris flower’s petal, which is the inner whorl of the flower.
4. Petal Width (cm): The width of the iris flower’s petal

The target variable is the species of the iris, the species in the dataset are as follows:

* Iris-setosa: Characterised by relatively small sepal and petal sizes.
* Iris-versicolor: Characterised by medium-sized sepal and petal sizes.
* Iris-versicolor: Characterised by large sepal and petal sizes (larger than the other two species).

# **METHOD**

This report will evaluate three classification models: the Naïve Bayes, K-nearest neighbour (K-NN) and Support Vector Machine (SVM) models. The models are briefly discussed in terms of what they are, how they work, their strengths and limitations. The method of training the models will also be documented.

## **Naïve Bayes**

The Naïve Bayes algorithm is derived from the widely recognized Bayes theorem and stands out as a significant probabilistic technique for classification(Wickramasinghe & Kalutarage, 2021:2277) (See Figure 1).

### HOW IT WORKS

Utilizing the Bayes rule, Naïve Bayes classifiers assess the likelihood that an entity belongs to a specific class based on the available data related to the entity (See figure 1). Notably, Naïve Bayes classifiers possess a distinctive characteristic that makes them naïve. The algorithm assumes that each feature is independently indicative of the class (Wickramasinghe & Kalutarage, 2021:2278). The significance of each feature is determined through training on labelled data (Hajirasouliha & Elemento, 2020:909) (See Figure 5).

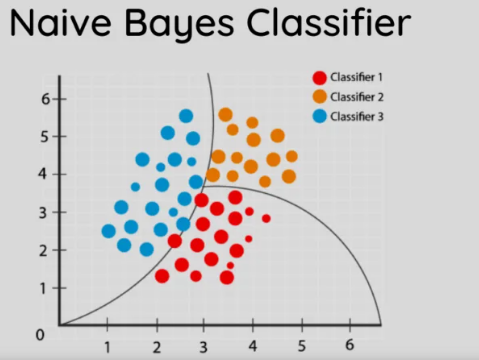


Figure : The Naive Bayes Classifier (Analytics Learn)

Equation :Bayes Theorem

In equation 1;

P(H|E) = Posterior probability of Hypothesis (H) given that the Evidence ( E) is true.

P(E|H) = Likelihood of the evidence given that the hypothesis is true.

P(H) = Prior Probability of the hypothesis

P(E) = Prior Probability that the evidence is true.

### STRENGTHS

Naïve Bayes models are appealing for a wide range of tasks because of their simplicity, comprehensiveness, interpretability, and modular framework, deeply rooted in statistical principles. Not only are they simple, but they are robust and efficient too (Wickramasinghe & Kalutarage, 2021:2278). These classifiers adeptly incorporate Bayes theorem (see equation 1) within their decision-making mechanism, resulting in notable efficiency and scalability. Their prowess becomes particularly pronounced when handling extensive datasets (Hajirasouliha & Elemento, 2020:909).

### LIMITATIONS

Naïve Bayes encounters various constraints that can impact its effectiveness under specific circumstances. A primary constraint lies in its presumption of feature independence, a hypothesis that may not align with real-world scenarios where features often exhibit correlations. Consequently, this can result in erroneous predictions and diminished accuracy. Another constraint comes from its inability to accommodate missing data. Naïve Bayes necessitates complete data for all attributes; any absence of data obstructs its predictive capability. Furthermore, the algorithm assumes uniform significanc across all features, a presumption that may not hold in certain contexts. This tendency can lead to suboptimal outcomes when some features hold greater relevance than others (Wickramasinghe & Kalutarage, 2021:2282).

Variations of Naïve Bayes have been developed to address some of the limitations of the original algorithm and to ensure that it caters to general data, these variations are: Weighted Naïve Bayes, Semi-Naïve Bayes and Feature-Dependent Naïve Bayes. Collectively, these variants of Naïve Bayes accommodate diverse datasets by relaxing the assumption of feature independence and embracing the intricate feature dependencies. They can support classifier accuracy in situations where the original Naïve Bayes algorithm might fall short(Wickramasinghe & Kalutarage, 2021:2282).

## **K-Nearest Neighbour**

K-Nearest Neighbours (KNN) algorithm is a versatile and straightforward supervised machine learning approach that addresses classification, regression, and search tasks. It operates on the assumption that similar items tend to cluster together in proximity. Functioning as an instance-based learning method, KNN utilizes the features of training examples in the feature space to categorize objects. Through a majority vote of its nearest neighbours, an object is assigned to the class that prevails amongst its k-closest counterparts. This algorithm determines the classification of a new test feature vector based on the collective classes of its k-nearest neighbours (Boateng *et al.*, 2020:346), see figure 2. In the realm of the supervised algorithms, KNN leverages the features and labels of training data to predict the classification of unlabelled data (Uddin *et al.*, 2022) (See Figure 2).

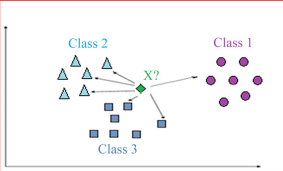


Figure : Overview of KNN (Boateng et al., 2020:346)

### HOW IT WORKS

In essence, KNN undertakes classification using a training model that closely resembles the test query. It accomplishes this by considering the k nearest training data points, also known as neighbours, which exhibit the closest proximity to the query being evaluated. Subsequently, the algorithm employs a majority voting rule to determine the final classification (Uddin *et al.*, 2022).

The classification comprises of two key stages; firstly, determining the nearest neighbours based on a distance metric, which can further be leveraged for determining an object’s class. The selection process involves various techniques, including assigning the majority class of the nearest neighbours or employing distance-weighted voting to give closer neighbours greater influence in the class decision (Cunningham & Delany, 2021). Simultaneously, KNN employs Euclidean distance metrics for locating the nearest neighbour, using the Euclidean distance formula (equation 2). The KNN classifier estimates conditional distributions for classifying observations, identifying the K nearest training data points in proximity to a test observation and assigning it to the class with the highest probability. This approach proves beneficial for cases reliant on recognizing similar objects, contributing to a versatile and effective classification technique (Boateng *et al.*, 2020) (See Figure 6).

Usually the value of K is an odd number to avoid inconclusive outcomes during majority voting among classes (Uddin *et al.*, 2022).

Equation : Euclidean Distance Formula

In Equation 2:

x, y = points in Euclidean n-space

, = Euclidean vectors starting from the initial point

= n-space

### STREGNTHS

KNN exhibits resilience against noisy training data and showcases efficiency when dealing with large training datasets. Nevertheless, the selection of the parameter K (number of nearest neighbours) and the choice of distance metric demand attention for this algorithm (Boateng *et al.*, 2020). Positioned among the simplest machine learning methods, the KNN algorithm holds popularity in classification tasks due to its adaptable and comprehensible nature. However, its versatility paves the way for diverse KNN forms or variations. These variations encompass different algorithmic considerations such as optimizing the k parameter, refining distance calculations, introducing varied weights to data points, and even curtailing training datasets to address earlier-mentioned challenges (Uddin *et al.*, 2022).

### LIMITATIONS

The traditional KNN algorithm faces multiple limitations that diminish its classification effectiveness, including its impartial treatment of all classification-dependent neighbours, the absence of distance calculation capabilities between data points, and the consideration of redundant dataset features. (Uddin *et al.*, 2022). Computation time can extend due to the need to calculate distances between each observation and all training samples, particularly slowing down as the number of examples and independent variables rises. KNN have a tendency to slow down considerably as data volume expands, rendering it unsuitable for scenarios requiring fast predictions (Boateng *et al.*, 2020).

## **Support Vector Machine**

Support Vector Machines (SVM) are supervised learning models with related learning algorithms that analyse data for classification and regression analysis. According to (Boateng *et al.*, 2020), the SVM method, seeks to identify the best hyperplane within an n-dimensional classification space, with the goal of achieving the maximum separation margin between classes. In its most basic form, SVM functions as a linear binary classifier, identifying a single boundary between two groups. This linear technique implies that the input data in several dimensions is linearly separable, as shown in Figure 3. SVMs use training data to establish an optimum hyperplane—often a line in simpler scenarios—to successfully segregate a dataset into discrete predetermined classes.

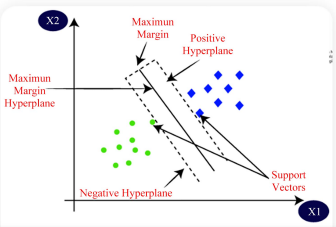


Figure : illustration of SVM in 2-D (Boateng et al., 2020)

### HOW IT WORKS

SVMs build a model that defines decision boundaries for different classes, identifying the hyperplane that separates these classes and distinguishing objects within the analysed groups. SVMs can learn to predict intricate class distinctions by transforming input data into higher-dimensional spaces, making them adept at handling complex object categorizations (Hajirasouliha & Elemento, 2020). Increasing the distance between classes by using a larger hyperplane margin improves classification accuracy. Furthermore, SVMs excel at handling non-linear classifications, broadening their applicability (Boateng *et al.*, 2020).

The SVM classifier, which was designed specifically for binary classification, uses a kernel-based supervised learning algorithm to divide data into two or more classes. However, this method is less suitable for large training datasets. Kernel functions play an important role in this process, mapping the training set to improve its similarity to a linearly separable dataset and thus efficiently increasing its dimensionality. There are several kernel functions available, such as linear, RBF, quadratic, Multilayer Perceptron, and Polynomial kernels. Linear kernels are better suited to linearly separable data, whereas RBF kernels are better suited to non-linear data. Training time with linear kernels is generally faster than with RBF kernels, and the former is also less prone to overfitting (Boateng *et al.*, 2020). The SVM is produced by combining non-linear kernels and the support vector classifier (See Figure 7).

### STRENGTHS

This algorithm stands out as a robust and highly accurate approach within well-known machine learning algorithms. This method has gained traction across research and industry due to its remarkable effectiveness in tracking non-linear problems. Its versatility extends to solving nonlinear regression estimation problems and even forecasting time series. SVM finds broad application in diverse analyses, encompassing regression, classification and nonlinear function approximation(Boateng *et al.*, 2020).

One of SVMs notable strengths lies in its insensitivity to the underlying data distribution, setting it apart from other statistical techniques. SVM excels in efficiently managing small training datasets and proves to be an adept classifier in high-dimensional spaces. This efficacy is apparent in its ability to assign new data points using a subset of training data, providing memory-efficient performance as only this subset needs storage. SVMs adaptability extends further through the application of various kernels offering flexibility in decision boundaries and enhancing classification performance. Nevertheless, challenges persist, including selecting an appropriate kernel, optimizing kernel parameters, and grappling with the relatively intricate mathematical underpinnings of SVM (Sheykhmousa *et al.*, 2020)

### LIMITATIONS

The SVMs have some disadvantages, including a considerable computational cost caused by limited optimization programming, which increases processing time. In response to this worry, an alternate technique known as Least Square Support Vector Machine (LSSVM) arose, which converts SVM's inequality constraint into an equality criterion (Ahmad *et al.*, 2014).

SVMs excel in categorizing high-dimensional large data by identifying a small number of support vectors, allowing for quick subgroup differentiation. Nonetheless, dealing with large datasets remains computationally hard. SVMs may anticipate complex object classifications, prospering especially when the number of dimensions exceeds the number of examples. They are, however, slow and unsuitable for large datasets (Hajirasouliha & Elemento, 2020; Sheykhmousa *et al.*, 2020).

## **Training models**

The dataset was loaded, and labels were added to the dataset. Then after, the data frame was shuffled randomly and split into a 60:20:20 ratio, i.e., training: 60%, validation: 20% and testing: 20%. The data frame is then divided into features (denoted by x) and the target variable, which is the class (denoted by y). The training, validation and testing dataset are preceded by x and y to denote the features and target variable in each data frame.

To implement these algorithms and train our models, we leveraged the ‘**scikit-learn’** (sklearn) library, a widely acclaimed python library for machine learning. For each of the algorithms, the relevant classifier class was imported, these classes are **GaussianNB**, **KNeighborsClassifier**, and **SVC** for Naïve Bayes, KNN and SVM respectively. These classes were imported from ‘**sklearn.naive\_bayes**’, ‘**sklearn.neighbors**’ and ‘**sklearn.svm**’. After the classifier classes were imported the model was fitted using the training data and corresponding labels with the ‘**.fit()**’ method. No hyperparameter tuning was done on any of the algorithms. The standard version of the algorithms was used, and no additional parameters were specified or used.

After training each model, their performance was evaluated using a suite of classification metrics, including accuracy, precision, recall, F1-score, Receiver Operating Characteristic (ROC) curve and the Area Under the Curve (AUC). The ‘**scikit-learn**’ library also provided convenient functions for calculating these metrics.

The train/validate/test split ratio was 60-20-20, ensuring that a significant portion of the data was used for training while preserving an independent set for testing the models’ generalization capabilities.

The diagrams (see Figure4-7) below provide a high-level overview of the training process, and an overview for how each of the models, and are a graphical representation of how the models actually work.

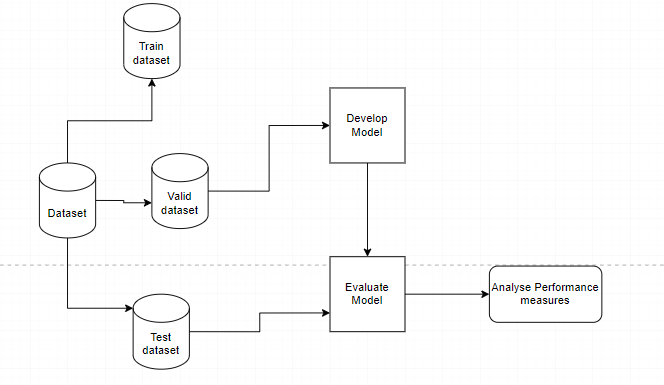


Figure : Overview of training process

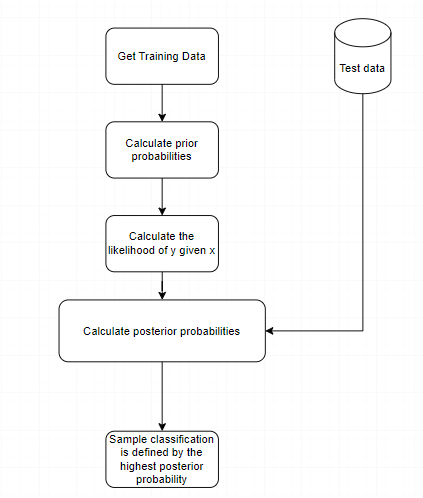


Figure : Flowchart of Naive Bayes

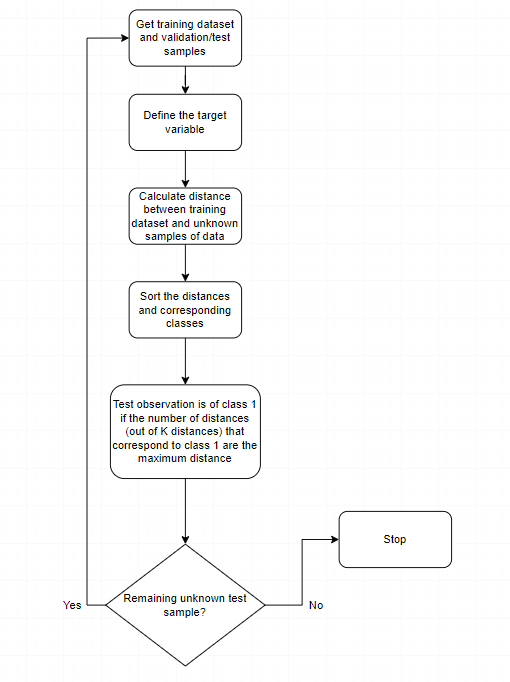


Figure : Flowchart of KNN model

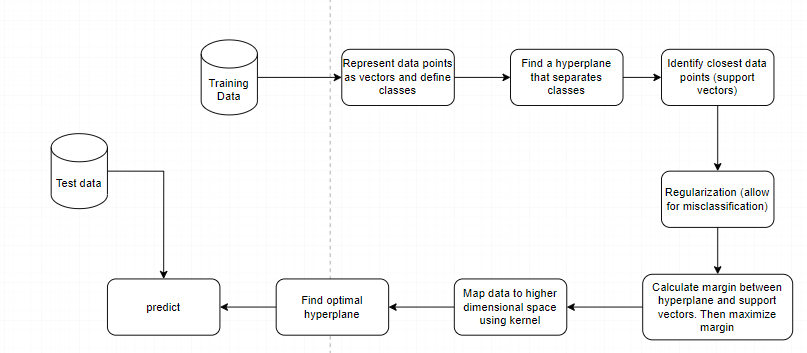


Figure : Flowchart of SVM

# **RESULTS AND DISCUSSIONS**

## **Results**

As previously mentioned, the models will be evaluated using the accuracy, precision, recall, F1-score and Roc and AUC score metrics. These metrics are briefly explained below, and the results of each model for each metric is recorded in Table 1.

1. Accuracy signifies the ratio of correctly classified instances of a specific class to the total number of samples (Bhandari, 2020).
2. Precision pertains to the proportion of correctly identified positive instances among all instances, showcasing the accuracy in discerning the desired categories (Bhandari, 2020).
3. Recall, also known as sensitivity, highlights the ability to accurately identify favourable classes, amidst all classes, presenting a comprehensive perspective on the classification scheme (Bhandari, 2020).
4. The F1 score assesses both precision and recall simultaneously by employing the harmonic mean rather than the regular arithmetic mean (Bhandari, 2020)
5. ROC curve is a graph that shows how well the classification model in question performs (Bhandari, 2020).
6. AUC measures the ability of a classifier to differentiate between classes and is a summary of the ROC curve (Bhandari, 2020).

### PERFORMANCE RESULTS FOR NAÏVE BAYES, KNN AND SVM

|  |  |  |  |
| --- | --- | --- | --- |
|  | Naïve Bayes | KNN | SVM |
| Accuracy | 0.967 | 0.967 | 0.967 |
| Precision | 0.967 | 0.967 | 0.967 |
| Recall | 0.967 | 0.967 | 0.967 |
| F1-Score | 0.967 | 0.967 | 0.967 |
| ROC and AUC | 0.990 | 0.945 | 0.994 |

Table : Results of metrics by which models were evaluated

#### ROC CURVE NAÏVE BAYES

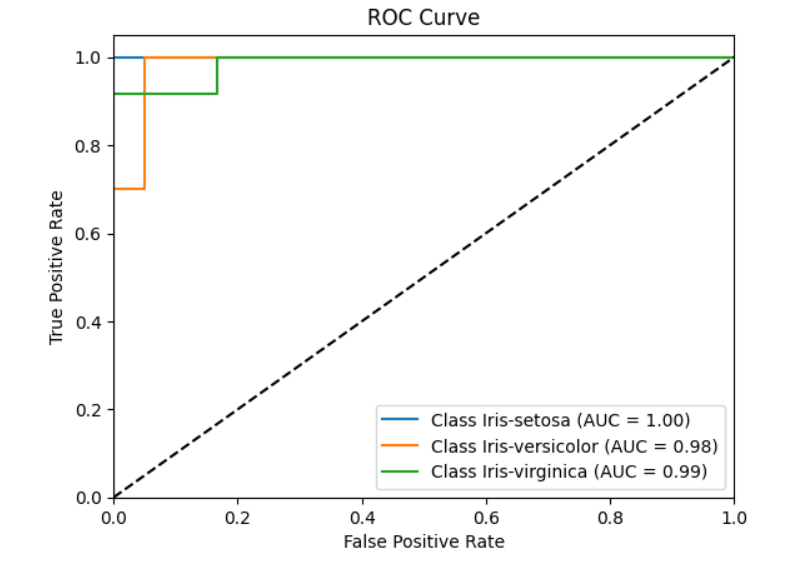


Figure : ROC Curve for Naive Bayes

#### ROC CURVE KNN

A graph of a curve

Description automatically generated with medium confidence

Figure : ROC Curve for KNN

#### ROC CURVE SVM

A graph with a line and a line

Description automatically generated with medium confidence

Figure : ROC Curve for SVM

## **Discussion**

The performance metrics of the classification models – Naïve Bayes, KNN and SVM – that have been applied to the Iris dataset will be discussed.

Interestingly, all three models demonstrated similar results as seen in Table 1. This uniformity indicates a high level of consistency in how these models classify and predict the iris species in the dataset.

However, when delving into the ROC and AUC performance, nuances emerge. The SVM model stands out with a ROC-AUC score of 0.994, showcasing its capacity to effectively distinguish between classes. KNN follows with a ROC-AUC score of 0.945. In contrast, the Naïve Bayes model exhibits a slightly lower ROC-AUC score of 0.990. This implies that SVM and Naïve Bayes are particularly efficient in distinguishing between classes, with SVM taking the lead.

The parity in the metrics used to evaluate the models across all 3 models implies that the models are consistently performing well on the dataset. It is important to note that while these metrics indicate strong performance, other factors such as model complexity, computational efficiency, and interpretability could be additional criteria to consider when choosing the best model for this specific task.

## **Related Work**

Shaw Vinay used the Naïve bayes classifier to perform a classification task for which the model had 100% accuracy. This is more than what was achieved in this report. The difference in accuracy could be attributed to the fact that Shaw used the ‘label encoder’ to encode the target variable – the species of the flower – into the numbers 0, 1 and 2 which correspond to the classes Iris-setosa, Iris-versicolor and Iris-virginica respectively. The author did this because apparently ‘**KNeighborsClassifier’** does not work very well with string labels. The encoded data was used to train the KNN model, the Naïve Bayes model, the SVM model and other models. In addition to label encoding, the train and test split ratio used by the author is 70-30; meaning that 70% of the data was for training and 30% was for testing. In this report, the data was split in a 60-20-20 ratio, where 60% was for training, 20% for validation and 20% for testing.

The train-test split is done to ensure that there is data to fit and train the model – train dataset – and data to make predictions and compare them to labels in the data – test dataset. The goal is to measure the performance of the model on data it is not familiar with – not the training data. This is ideal for large datasets, not so much for smaller datasets, as the data in the training dataset will not be enough to aid the model in learning to map inputs to outputs. There also will not be sufficient data for the test set for the models’ performance to be evaluated (Browniee, 2020). It is also important to not that having a high train or test ratio could result in underfitting and/or overfitting. When a large portion of the data is used for training, the model might not have enough data to learn patterns effectively, and the model instead learns the noise in the training dataset. This refers to overfitting, and it adversely impacts the performance of a model. Underfitting on the other hand is when a model does not have the ability to generalize or model the training dataset, it could be a result of using a high test ratio (Browniee, 2016; Elite Data Science, 2022).

There are various factors that could affect the accuracy of a model, such as bias, certain parameters of the algorithm and features of the dataset to name a few.

# **CONCLUSION**

In this exploration of classification algorithms using the Iris dataset, the strengths, and limitations of three machine learning models – Naïve Bayes, K-Nearest Neighbours (KNN), and Support Vector Machine (SVM). The goal was to evaluate their performance in predicting the species of iris flowers based on their measurements. Through rigorous analysis and evaluation, valuable knowledge about the capabilities of these algorithms has been somewhat obtained.

The report has shown that each algorithm, despite its unique approach, demonstrated consistency in predicting iris species. The metrics employed – accuracy, precision, recall, F1-score, and the ROC-AUC curve, provided a comprehensive understanding of their performance. Surprisingly, all three models exhibited comparable results across these metrics, suggesting that their learning strategies have a strong grasp of the underlying dataset patterns.

However, the differentiating factor emerged when examining ROC-AUC scores. Here, the SVM model stood out, with a ROC-AUC score of 0.994. This implies that SVM excels at distinguishing between classes, making it a potential go-to choice for applications requiring precise classification.

# **REFERENCES**

Ahmad, A.S., Hassan, M.Y., Abdullah, M.P., Rahman, H.A., Hussin, F., Abdullah, H. & Saidur, R. 2014. A review on applications of ANN and SVM for building electrical energy consumption forecasting. *Renewable and Sustainable Energy Reviews*, 33:102-109. <https://www.sciencedirect.com/science/article/pii/S1364032114000914> <https://doi.org/10.1016/j.rser.2014.01.069>

Bhandari, A. 2020, 30 August. *Guide to AUC ROC Curve in Machine Learning : What Is Specificity?* [Blog post]. <https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/> Date of access: 30 August 2023.

Boateng, E.Y., Otoo, J. & Abaye, D.A. 2020. Basic Tenets of Classification Algorithms K-Nearest-Neighbor, Support Vector Machine, Random Forest and Neural Network: A Review. *Journal of Data Analysis and Information Processing*, 8(4):341-357. 10.4236/jdaip.2020.84020

Browniee, J. 2016, 30 August. *Overfitting and Underfitting With Machine Learning Algorithms* [Blog post]. <https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/> Date of access: 30 August.

Browniee, J. 2020, 30 August 2023. *Train-Test Split for Evaluating Machine Learning Algorithms* [Blog post]. <https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/> Date of access: 30 August 2023.

Cunningham, P. & Delany, S.J. 2021. k-Nearest Neighbour Classifiers - A Tutorial. *ACM Comput. Surv.*, 54(6):Article 128. <https://doi.org/10.1145/3459665> 10.1145/3459665

Elite Data Science. 2022, 30 August. *Overfitting in Machine Learning: What It Is and How to Prevent It* [Blog post]. <https://elitedatascience.com/overfitting-in-machine-learning> Date of access: 30 August.

Hajirasouliha, I. & Elemento, O. 2020. Precision medicine and artificial intelligence: overview and relevance to reproductive medicine. *Fertility and Sterility*, 114(5):908-913. <https://www.sciencedirect.com/science/article/pii/S0015028220323980> <https://doi.org/10.1016/j.fertnstert.2020.09.156>

Sheykhmousa, M., Mahdianpari, M., Ghanbari, H., Mohammadimanesh, F., Ghamisi, P. & Homayouni, S. 2020. Support Vector Machine Versus Random Forest for Remote Sensing Image Classification: A Meta-Analysis and Systematic Review. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 13:6308-6325. 10.1109/JSTARS.2020.3026724

Uddin, S., Haque, I., Lu, H., Moni, M.A. & Gide, E. 2022. Comparative performance analysis of K-nearest neighbour (KNN) algorithm and its different variants for disease prediction. *Scientific Reports*, 12(1):6256. <https://doi.org/10.1038/s41598-022-10358-x> 10.1038/s41598-022-10358-x

Wickramasinghe, I. & Kalutarage, H. 2021. Naive Bayes: applications, variations and vulnerabilities: a review of literature with code snippets for implementation. *Soft Computing*, 25(3):2277-2293. <https://doi.org/10.1007/s00500-020-05297-6> 10.1007/s00500-020-05297-6