Optimizing SVM Performance through Combinatorial Hyperparameter Tuning and Model Selection

Hassan Tariq*, Mehwish Majeed, Mueed Ahmad

Department of Computer Science,
University of Agriculture Faisalabad
Agriculture University Road, 38000 Faisalabad, Pakistan
E-mails: hassan@uaf.edu.pk, mehwishmajeed86@gmail.com, hmedmueed822@gmail.com

**Corresponding author*

Received: February 12, 2024 **Accepted: May 21, 2024**

Published: June 30, 2025

Abstract: *Among the support vector machine (SVM) methods, the support vector classifier (SVC) is widely utilized for binary and multi classification tasks across various datasets. Hyperparameter tuning plays a critical role in optimizing the performance of SVM by helping to prevent overfitting or underfitting, enhancing model stability, adapting the model to different types of datasets, and increasing predictive power. This study aims to maximize SVM performance on datasets related to heart disease, liver disorder, breast cancer, and MINST (a digit dataset), which exhibit diverse sample and feature counts. Our proposed framework leverages Python libraries. It employs a combinatorial approach to tune the kernel, C, and degree hyperparameters for both the train-test-split and cross validation (CV) models with different input values. Model accuracy, the area under the curve (AUC), and the F1 score were used to evaluate the models. The most suitable model, hyperparameters, and validation size or number of folds, are selected to achieve maximum accuracy of SVM across all datasets. Results demonstrate that the train-test-split model generally improves SVM performance, except for the heart disease dataset, on which the CV model performs well. Our contribution lies in the development of a framework that combines combinatorial hyperparameter tuning and model selection, aiming to optimize SVM performance and predictive capabilities. Future research can focus on enhancing SVM performance for large-scale datasets and exploring ensemble techniques or deep learning models to enhance its applications in real-world scenarios.*

Keywords: *Support vector machine, Hyper parameter tuning, Combinatorial optimization, Model selection, Machine learning, Cross validation.*

Introduction

The study of intelligent systems capable of carrying out activities that traditionally require human intelligence is known as artificial intelligence (AI), a subfield of computer science [1, 39, 40, 45, 49]. Machine learning (ML) is a branch of AI that involves the development of models and algorithms capable of automatically improving and making predictions based on given data [16]. Instead of being explicitly programmed for specific tasks, it allows computers to learn from examples, patterns, and experiences [37]. By employing statistical methods and mathematical models, ML algorithms can analyze large datasets, recognize patterns, and make accurate predictions or decisions [56]. AI systems benefit greatly from being able to learn and adapt on their own, as it increases their capacity to perform tasks. In ML, a variety of different algorithms can be used. Which is used depends on the required output. ML algorithms typically belong to one of two learning types, supervised or unsupervised. Supervised learning algorithms are taught to predict outputs from

input features using labelled data, while unsupervised learning algorithms examine unlabelled data in search of patterns or structures but without the need for predefined outputs [7, 8].

Scikit-learn [47] is a popular ML library in Python that provides an implementation of support vector machine (SVM) through its SVM module. The SVM functionality in scikit-learn makes it simple to train SVM models, adjust hyper parameters, and generate predictions. The SVM is frequently employed for problems including classification, regression, and outlier detection [47]. The SVM method has gained wide applications in medicine and biology [36]. It is one of the methods SVC is frequently used for dataset binary and multi classification tasks. It is excellent at locating the best hyperplanes to divide data points into several classes, offering robustness and strong generalization capabilities [2, 66]. The effectiveness of SVMs in dealing with high-dimensional data and their capacity to implicitly map features to higher-dimensional spaces are well recognized [8, 15]. Due to SVMs success, there has been a significant amount of research into enhancing their functionality and examining applications in several fields, including the creation of cutting-edge methods and algorithms [10].

The performance of a SVM can be improved by using methods such as feature scaling to normalize input features, tuning hyper parameters with methods like grid search [3], selecting data-driven kernel functions, utilizing cross validation (CV) for model evaluation and hyper-parameter tuning, performing data pre-processing to improve data quality, utilizing dimensionality reduction techniques for high-dimensional data, utilizing model ensembles to improve generalizability, and taking into account multiple models [14, 28]. Using these methods together, SVM models' precision, efficiency, and generalizability could be increased [29].

Previous studies of optimizing SVM performance and automated hyper parameters tuning has encountered several limitations. Automated parameter tuning studies that target SVM performance optimization are not suitable for low-specification machines. For example, the automated method grid search for hyper parameter tuning has no user involvement and depends on predefined grids; it needs high computational power and specifications to run and also has the issue of scalability [25, 30, 48]. Ensemble methods also call for more computing power [39]. The random search explores the hyper parameter space by randomly sampling different combinations of hyper parameter values. This may require more iterations and does not consider the effect of a combination of multiple parameters on the performance of SVM [6, 38].

Approaches like meta-learning and automated machine learning (autoML) heavily rely on the performance of pre-existing models or the availability of prior knowledge [41]. The gradient-descent approach, when considering multiple parameters, has a dimensionality curse and difficulties in tuning learning rates, hindering efficient optimization [32]. These methods did not model the performance of hyper parameters, how they behave on different datasets with varying sample and feature counts. Furthermore, these approaches do not consider multiple parameters like model selection, the train-test-model, or CV model with different test sizes or folds, along with combinatorial hyper parameter tuning to optimize performance. And user involvement in setting these parameters is often not incorporated.

In order to improve SVM performance, we intend to present a thorough framework that addresses the shortcomings of earlier algorithms. Our framework incorporates user input into setting these parameters and considers multiple parameters at once. Additionally, the user can choose between train-test-split and CV with various inputs when choosing the model evaluation

method [18, 58]. Our research focuses on maximizing SVM performance on diverse datasets with varying sample and feature counts [47, 67]. We propose a novel approach that do combinatorial hyper parameter tuning with model selection to achieve this goal. By systematically exploring all possible combinations of hyper parameters, our method enables the identification of promising settings for each dataset based on their corresponding accuracy scores. This comprehensive evaluation of hyper parameter combinations provides researchers with valuable insights for finding the best hyper parameters for SVM and maximizing its performance. By offering a systematic and comprehensive framework, our research advances the understanding of SVM hyper parameter optimization and empowers researchers with techniques to maximize SVM performance.

Literature review

Several researches examined the effectiveness of hyperparameters tuning, model selection, and other conventional techniques for improving SVM classifier performance.

Hussain et al. [24] investigated the performance of SVM, k-nearest neighbours (KNN), and stochastic gradient descent (SGD) algorithms in classifying syncope cases. They used k-fold CV as well as train-test-split methods to conduct the evaluation. The train-test-split used 20.00% for test data and 10-fold for k-fold CV. A statistical value-based performance analysis was carried out. Their findings show that the SVM-based model can distinguish between syncope and non-syncope events significantly more effectively than the KNN and SGD-based models when using both train-test-split evaluation and k-fold CV.

Kalcheva et al. [27] focused on comparing the accuracy of different kernel functions in SVM for movie review classification. Initially, they tested the SVM method with default parameters. The authors determine the kernel functions and the corresponding parameters that produce high accuracy in order to get the best results. They utilized kernel functions, including the polynomial kernel of degree 2, the linear kernel, and the radial basis kernel and parameters the C and gamma parameter. Their results demonstrate that the accuracy of movie review classification using these kernel functions is higher than 83.00%. Interestingly, their research shows that the sigmoid radial kernel is not appropriate for text classification because it does not produce satisfactory results. This result implies that when performing text classification tasks, alternative kernel functions should be taken into account.

Chidambaram and Srinivasagan [11] reported their work on the significance of knowledge extraction through feature selection and the use of SVM with different kernel methods namely linear, polynomial, radial basis, and sigmoid, for classification. They employ classifier subset evaluation to select the most relevant features, thereby optimizing the feature vectors to enhance accuracy. Additionally, they introduced a novel kernel approach. Their findings demonstrate that the accuracy of the proposed SVM using the novel kernel approach demonstrated experimentally to be significantly higher when compared to conventional methods.

Rojas-Domínguez et al. [49] reported optimal hyperparameter tuning of SVM classifier. In order to accomplish this, they employed a number of techniques, including grid search, random search, estimation of distribution algorithms (EDAs), and bio-inspired metaheuristics. They also mention swarm intelligence algorithms as a well-liked method for optimization, in particular particle swarm optimization (PSO). By evaluating effectiveness, generalization, efficiency, and complexity, they choose the best approach. Their study of 15 medical diagnosis problems demonstrates that EDAs are the best method for hyperparameter tuning to improve the generalization and performance of the SVMs.

Syarif et al. [57] reported their work on SVM parameter optimization for classification tasks by using two popular methods grid search and genetic algorithm (GA). They used 9 different datasets for their experiment. Their findings demonstrated that grid search, when used to optimize SVM parameters, always finds parameter combinations that are close to optimal. It only performs well for datasets with few parameters and low dimensions. The GA algorithm performs better and is more reliable than grid search. They reported that GA is 16 times faster than grid search for parameter tuning of SVM for performance optimization.

Gold et al. [20] described the Bayesian approach to SVM classifiers, which enables the tuning of hyperparameters and the selection of relevant input features using automatic relevance determination (ARD). By maximizing the evidence and approximating evidence gradients using hybrid Monte Carlo sampling, their purposed approach allows for effective feature selection and hyperparameter tuning. They also highlighted the use of a Nystrom approximation of the Gram matrix to speed up sampling times while maintaining classification accuracy. Their experimental results show that the ARD approach, when compared to conventional non-ARD SVM systems, can significantly improve classification performance in problems with a large number of irrelevant features. The tuned hyperparameter values also serve as a criterion for pruning irrelevant features, leading to impressive reductions in the required features. However, in datasets created by human domain experts, non-ARD SVMs tend to be less affected by the presence of less relevant features, making feature elimination through ARD less impactful on classification accuracy but still beneficial in reducing the number of features needed.

Czarnecki et al. [12] focused on the robust optimization of SVM hyperparameters in the classification of bioactive compounds by comparing Bayesian and random search optimization methods with traditional approaches such as grid search and heuristic choice. Bayesian optimization's ability to achieve better classification performance and faster convergence has made it a popular choice for SVM parameter optimization. Their research showed that Bayesian optimization was more effective than other optimization techniques at improving classification accuracy while also requiring fewer iterations. If implementing Bayesian strategy is not possible, a random search optimization strategy should be used instead.

Background

This section discusses the underlying concepts used in the papers.

Support vector machine

SVM has gained significant popularity as powerful supervised learning techniques for various tasks such as classification, regression, and outlier detection [35]. Among the SVM methods, Support vector classifier (SVC) is widely used for both binary and multi-class classification tasks on datasets [44, 47]. SVC implementation is built upon the renowned library, which provides efficient and robust algorithms for SVM-based learning [17]. SVMs are particularly appealing because they can find an optimal hyperplane in a high-dimensional feature space, maximizing the margin between different classes. This property makes SVMs well-suited for handling complex datasets with non-linear decision boundaries [66]. The SVM classifier used in this study based on the following formula [33]:

$$f(x) = sign(w^T x + b), \quad (1)$$

where $f(x)$ represents the predicted class label for the input vector x; w denotes weight vector; x is the input vector or feature vector containing the values of features; and b is the bias term.

We used the One-vs-One (OvO) and One-vs-Rest (OvR) strategies for multi-class classification utilizing SVM in scikit-learn. Following is a representation of the decision functions for these strategies:

OvO: for *N* classes, $N(N-1)/2$ binary classifiers, each distinguishing between two classes, are trained. Among the binary classifiers, the final class prediction is determined by a voting scheme, such as majority voting. The decision function remains the same as Eq. (1).

OvR: *N* binary classifiers are trained, with each classifier distinguishing one class from the others. The class with the highest score from all binary classifiers is selected as the final prediction. The decision function can be expressed as:

$$f(x) = argmax_i(w_i^T x + b_i), \quad (2)$$

where $f(x)$ represents the predicted class label for the input vector x; w_i denotes the weight vector for the *i*-th binary classifier; b_i is the bias term for the *i*-th binary classifier; and $argmax_i$ selects the class with the highest score among all binary classifiers.

Model selection

Our proposed method utilizes two models, namely train-test-split and cross validation, for model selection of SVM.

1) *Train-test-split*

The train-test-split model divides the dataset into two subsets: the training and the test (or validation) sets [58]. The SVM model is trained using the training set, and its effectiveness is assessed using the test set. We can evaluate how well the trained SVM model generalizes to unknown data by using a different test set that was not used during the training process [51, 53, 64]. Our study experimented with different test or validation sizes, specifically 20.00%, 30.00%, and 40.00% of the dataset. We applied various combinations of hyperparameters to the SVM model on each test size to assess its performance.

2) *Cross validation*

We employed CV as another model selection technique in addition to train-test-split model. It involves dividing the available dataset into multiple subsets, or folds [9]. Each fold serves as a validation set, while the remaining folds are used for training, resulting in multiple iterations of the SVM model that are trained and evaluated [31, 65]. Evaluating the model on various subsets of the data aids in determining the model's performance and robustness. A performance estimate is then calculated by averaging the outcomes from each iteration. We conducted experiments using 3-fold, 5-fold, and 10-fold CV [18]. We explored different hyperparameter combinations for each fold to evaluate the SVM model's effectiveness.

Hyper parameters used

Following hyperparameters are combinatorically tuned to enhance the performance of SVM.

1) *Kernel parameter*

The kernel parameter of the SVM model was tuned to sigmoid, linear, and polynomial options. The choice of the kernel is crucial as it determines the transformation applied to the input data, allowing the SVM model to operate in a higher-dimensional feature space. The sigmoid kernel is effective for problems with complex decision boundaries because it captures nonlinear relationships. The linear kernel is appropriate when the decision boundary is straight,

such as in linearly separable datasets [23, 27]. When the features and the target variable are related in a polynomial fashion, the polynomial kernel performs better. The SVM model's performance and predictive accuracy improved by testing different kernels [45, 59].

$$f(x) = \sum\left(a_i * y_i * \mathrm{K}(x_i, \mathrm{x})\right) + b, \quad (3)$$

where a_i is the coefficient associated with the *i*-th support vector; y_i is the class label of the *i*-th support vector; $K(x_i, x)$ is the kernel function that computes the similarity between the training example x_i and the test example *x*; and b is the bias term.

2) *C parameter*

To achieve the best possible balance between training error and margin width, the C parameter in the SVM model was tuned to different values (3, 10, and 50). A lower C value can reduce overfitting by allowing for a wider margin and greater tolerance for misclassification. In contrast, a higher C value favours lower training error, which could lead to a smaller margin and increased sensitivity to data points [13, 55].

The objective function for SVM with parameter C can be represented as follows:

$$Minimize: \frac{1}{2} * \|w\|^2 + C * \sum\left(\max\left(0, 1 - y_i * (w^T * x_i + b)\right)\right), \quad (4)$$

where y_i is the class label of the *i*-th training example; x_i is the feature vector of the *i*-th training example; w^T is the transpose of the weight vector *w*; *b* is the bias term; and C is the C parameter that controls the trade-off between training error and margin violation.

3) *Degree parameter*

In our methodology, the degree parameter, which determines the complexity of the polynomial transformation in SVM models, was incorporated and tuned to values of 1, 2, and 3. The optimal balance between complexity and generalization was identified by assessing the model's accuracy and performance at various levels. This enabled us to choose the best degree parameter for our dataset, resulting in accurate predictions and reliable results [22, 54].

$$f(x) = \Sigma(a_i * y_i * (x_i^T * x + c)^d + b, \quad (5)$$

where a_i is the coefficient associated with the *i*-th support vector; y_i is the class label of *i*-th support vector; x_i^T is the transpose of the feature vector x_i; x is the feature vector of the test example; c is an additional constant term; d is the degree parameter specifying the degree of the polynomial kernel function; and *b* is the bias term.

Accuracy metrics

The model's performance in the train-test-split approach was evaluated using train accuracy, test accuracy, F1 score, and area under curve (AUC) score measurements. And the model accuracy for the CV was assessed by computing the CV accuracy and F1 score. These evaluation metrics were used to perform a thorough evaluation of the model's precision and predictive power.

1) *Train accuracy*

Train accuracy is a metric for evaluating how well a model performs on the training set. It is determined by dividing the proportion of samples that were correctly classified by the total number of samples used for training [42]. Training accuracy formulation can be stated as follows:

$$Train\ accuracy\ = \frac{Number\ of\ correctly\ classified\ samples\ in\ the\ training\ set}{Total\ number\ of\ training\ samples} \tag{6}$$

2) *Test accuracy*

Test accuracy is the measure of model performance on unseen or new data. It's determined by dividing the proportion of the test set that was properly classified by the total number of test.

$$Test\ accuracy\ = \frac{Number\ of\ correctly\ classified\ samples\ in\ the\ test\ set}{Total\ number\ of\ test\ samples} \tag{7}$$

3) *Cross validation accuracy*

The average accuracy of a model over all folds of cross validation is known as CV accuracy. It is determined by training and assessing the model on multiple subsets of the data, and then taking the average of the resulting accuracies [61].

$$Cross\ validation\ accuracy\ = \frac{1}{k}\sum_{i=1}^{k} acc_k, \tag{8}$$

where *k* represents the number of folds or partitions in the CV process; acc_k is the accuracy obtained on the *k*-th fold.

4) *F1 score*

A ML model's efficacy in performing binary classification tasks can be measured using the F1 score. It is a unified score that takes into account both accuracy and recall, providing a balanced measure of a model's performance [21]. Its formulation can be stated as follows:

$$F_1\ = \frac{2}{\frac{1}{recall}+\frac{1}{precision}} = \frac{2\ precision \cdot recall}{precision + recall} = \frac{2TP}{2TP+FP+FN} \tag{9}$$

5) *AUC score*

AUC measures the model's ability to distinguish between positive and negative classes by computing the area under the receiver operating characteristic (ROC) curve [54]. Its formulation is given below:

$$AUC = \int \big(FPR(T) * TPR(T)\big) dT, \tag{10}$$

where $FPR(T)$ represents the false positive rate at a given threshold T; $TPR(T)$ is the true positive rate at the same threshold T; and T represents the range of thresholds used to generate the ROC curve.

Materials and methods

Methodology

A new model was developed utilizing Python programming to enhance the performance of SVM on four different datasets. The model used different Python libraries, namely scikit-learn, Pandas, and NumPy. Two methods from model selection, namely CV and train-test-split,

were employed. Hyperparameters that significantly affected the accuracy of the SVM method from scikit-learn were tuned for each model. Specifically, kernel, C hyperparameter, and degree hyperparameter were combinatorically tuned using the best possible three inputs for each. All combinations resulting from the combinatorial tuning of these hyperparameters and model selection were observed. The most significant hyperparameter and model selection techniques, which led to an improved performance of SVM on all datasets, were identified. The proposed model flow diagram is shown in Fig. 1.

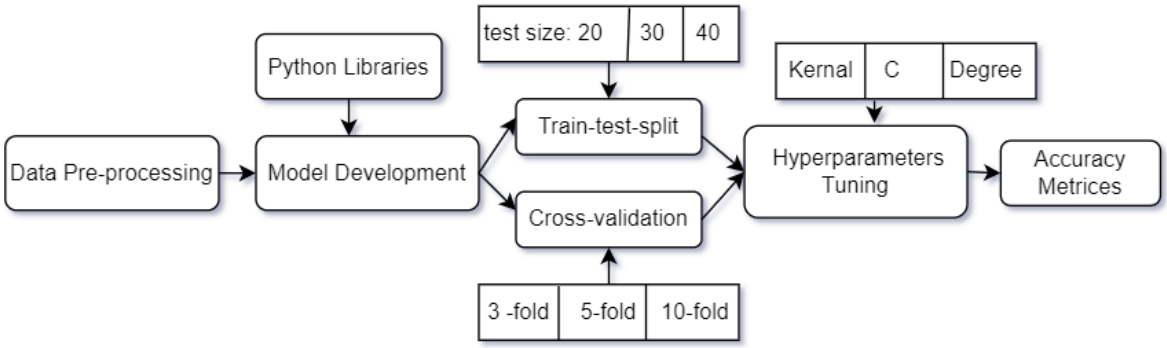

Fig. 1 A comprehensive framework for SVM performance optimization

Dataset description

We used four binary and multi-classification datasets for SVM classification. Each dataset has a varying number of samples and feature counts can be seen in Fig. 2.

1) *Breast cancer dataset* is a binary classification dataset containing 569 samples of breast cancer tumours, each characterized by 30 numerical features representing attributes of the tumour cells. The dataset is widely used in classification algorithms, notably SVM, for distinguishing between malignant and benign tumours. Available at: https://scikit-learn.org/stable/datasets/toy_dataset.html#breast-cancer-dataset.
2) *Heart disease dataset* is a binary classification dataset that contains sample data from 270 patients with suspected heart disease and has 13 features that can be used to predict the presence of heart disease. The complete dataset is available at: https://openml.org/d/43823.
3) *Liver disorder dataset* is a binary classification dataset that contains data from 345 patients with liver disorders, with 6 features that can be used to predict the presence or absence of liver disease. The complete dataset is available at: https://archive.ics.uci.edu/ml/datasets/Liver+Disorders.
4) *Digit dataset* is a multi-classification dataset that contains 1 797 samples of handwritten digits from 0 to 9, with each digit represented as an 8×8 pixel image. Furthermore, it contains 64 features. The complete dataset is available at: https://scikit-learn.org/stable/datasets/toy_dataset.html#digits-dataset.

The MINST and breast cancer dataset is taken from scikit-learn [44], the heart disease dataset from open ML [60], and the liver disorder dataset is taken from the UCI ML repository [67].

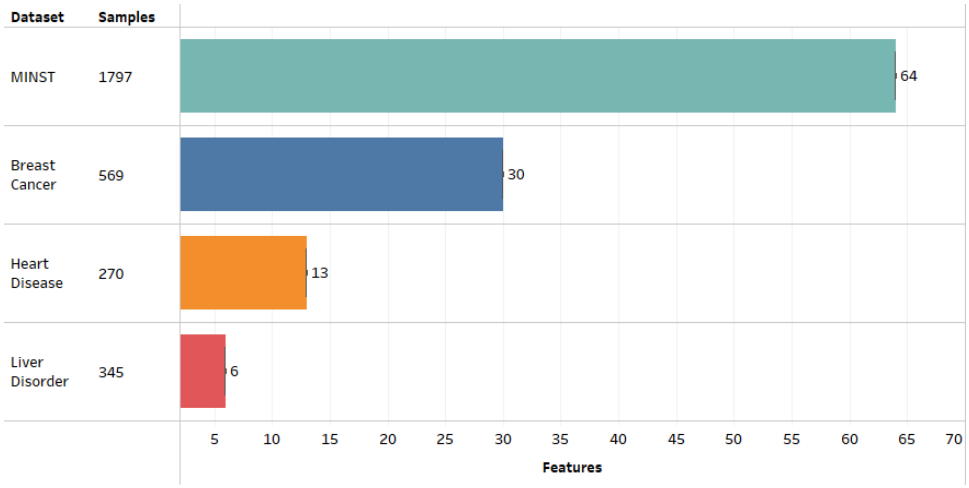

Fig. 2 Dataset's samples and features

Experimental setup

The experimental setup involved developing and evaluating an SVM model using Python libraries, including scikit-learn, NumPy, and Pandas is shown in Fig. 3.

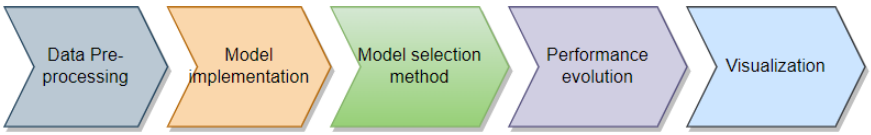

Fig. 3 Experimental setup

Data preprocessing

The required libraries were imported, Pandas and NumPy were used to pre-process the datasets. This includes operations like missing value handling, feature scaling, or feature engineering.

1) *NumPy* is an essential Python library for scientific computing. It offers support for sizable, multidimensional arrays and matrices, as well as a range of mathematical operations for effectively using these arrays. NumPy is widely used in ML for numerical calculations and offers crucial building blocks for data manipulation and transformation [22].

2) *Pandas* is a robust Python library for data manipulation. It offers simple data structures for effective data handling and analysis, like data frames. Pandas is suitable for activities like data cleaning, transformation, and exploration because it has functionality for reading, writing, and pre-processing structured data. It enables data processing pipelines by seamlessly integrating with other libraries, such as NumPy and scikit-learn [40].

Model implementation

Scikit-learn was used to implement the SVM algorithm. To investigate their effects on the model's performance, the model was set up with various hyperparameters, such as the kernel type, C value, and degree. Scikit-learn is a well-known Python ML library that offers many tools and algorithms for data mining, analysis, and modelling. It provides implementations of numerous ML algorithms, including SVM, which are frequently employed for classification and regression tasks. A user-friendly interface, effective data pre-processing, model training, and evaluation are all supported by scikit-learn [23].

Model selection methods
Two model selection methods, namely train-test-split and CV, were employed.

Performance evaluation
The accuracy metric was utilized to assess the performance of the SVM model. This metric measures the proportion of correctly classified instances. It provides a quantitative measure of the model's accuracy in predicting the target variable.

Visualization
The results obtained from the model evaluation were visualized using Tableau [68]. This allowed for further interpretation and comparison of the model's performance across different datasets or experimental conditions.

Results

In this study, the performance of an SVM classifier was assessed on four diverse datasets: breast cancer, MINST, heart disease, and liver disorder. The SVM classifier was tuned by adjusting various hyperparameters, including kernel type (linear, polynomial, sigmoid), C parameter (3, 10, 50), and degree parameter (1, 2, 3). The average test accuracies for the different hyperparameter combinations were plotted in Fig. 4 revealing exciting insights.

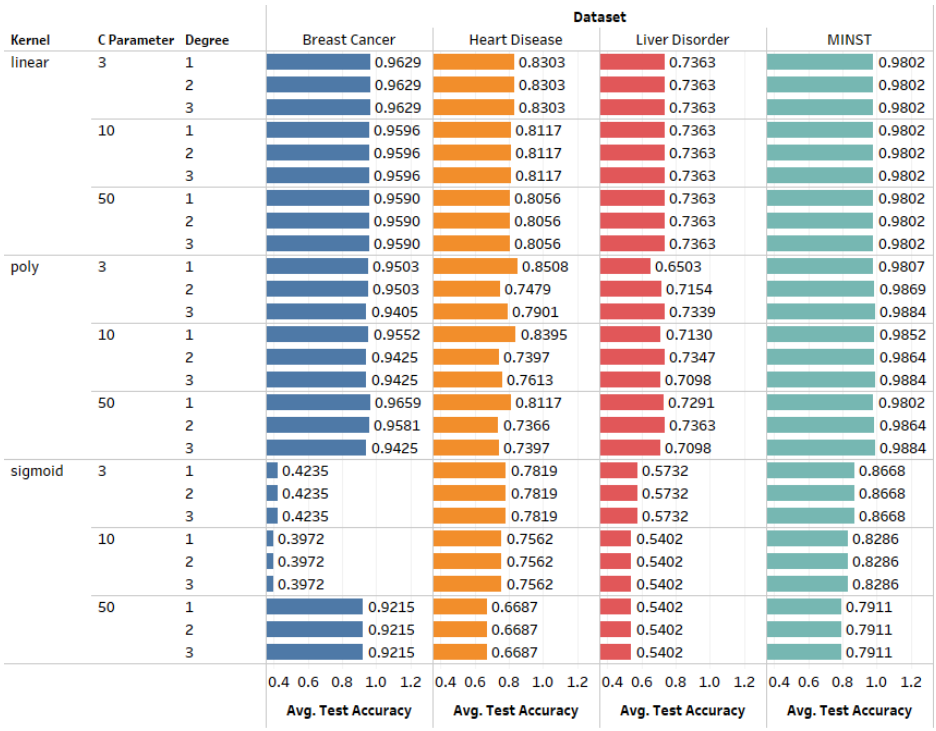

Fig. 4 Test accuracies vary with parameters

Fig. 4 demonstrates that the linear and polynomial kernel with all possible inputs of degree and C parameter, as opposed to the sigmoid kernel, exhibit the highest accuracy up to 99.00%. Contrarily, with the exception of the heart disease dataset, combinations using a sigmoid kernel resulted in lower average test accuracy drops to 40.00%. When comparing the datasets, the MINST dataset performed well than the other achieved the highest average test accuracy of 99.00% because its sample size and feature count were the highest (1 797 and 64, respectively). The liver disorder performed poorly and achieved the lowest accuracies, ranging from 54.00% to 73.63% across all possible parameter combinations due to its only six features.

Furthermore, the heart disease dataset achieved the highest average test accuracy of 83.03% with the linear kernel and also the best accuracies with the polynomial kernel, but it differed from other datasets in that it also performed well with the sigmoid kernel, which is suboptimal for most other datasets. This exception is due to its smaller sample size of 270 and different feature count of 13 than other datasets. Lastly, the breast cancer dataset performed well with linear and polynomial kernel and other hyperparameter values achieving the highest accuracy of up to 96.00%. It shows the lowest accuracy of 39.00% with a sigmoid kernel. These outcomes emphasize the importance of selecting the appropriate hyperparameters when training SVM classifiers to maximize performance. By tuning the hyperparameters of the SVM classifier, the study observed an improvement in the model evaluation results, which resulted in better and maximum performance compared to the original SVM classification with either no parameter tuning or arbitrary hyperparameters.

In ML, one important aspect of performance evaluation is checking the model's train and test accuracy. The training accuracy measures the model's performance on the data it was trained on, while the test accuracy assesses its ability to generalize to new data. A high level of training accuracy may indicate that the model has learned the patterns in the training data well, but it does not necessarily mean that the model will perform well on new, unseen data. In contrast, test accuracy provides a more reliable measure of how well the model will perform in real-world scenarios.

Evaluating the training and test accuracy is essential to ensuring the model is balanced with the training data and can generalize well to new data. By carefully monitoring both train and test metrics during model development, we identify potential issues and make informed decisions to improve the model's performance.

Fig. 5 illustrates our model's average training and test accuracies on each dataset. The breast cancer dataset achieved an impressively consistent classification accuracy of 83.00% average test accuracy and 82.00% average train accuracy, demonstrating the model's ability to generalize to new data on this dataset. Similarly, we got consistent average train and test accuracy on the liver disorder and MINST dataset approximately 66.00% train and test accuracy for liver disorder dataset and 93.00% test and 94.00% train accuracy for the MINST dataset. In addition, the heart disease dataset demonstrates a large discrepancy between the train and test accuracies of approximately 8.00%, indicating that this dataset is over fitted to our model. Fig. 5 also shows the overall median of train and test accuracies across datasets.

Our study emphasizes the importance of carefully evaluating and optimizing SVM model for classification tasks. By analysing their performance on diverse datasets, we can gain valuable insights into the factors contributing to their success or failure. We achieved impressive results on all datasets through hyperparameter optimization.

We used a scatter plot to visualize the relationship between the train and test accuracies obtained from our model. The plot included all the accuracy data points we collected, and we added a power trend line to understand the overall trend of the data better. The trend line *p*-value is less than 0.0001, indicating that the relationship between the variables is statistically significant, which means that it is unlikely to be due to chance. The *R*-squared value of 0.7952 indicates that the power trend line explains approximately 79.52% of the variation in the data, which is a relatively good fit.

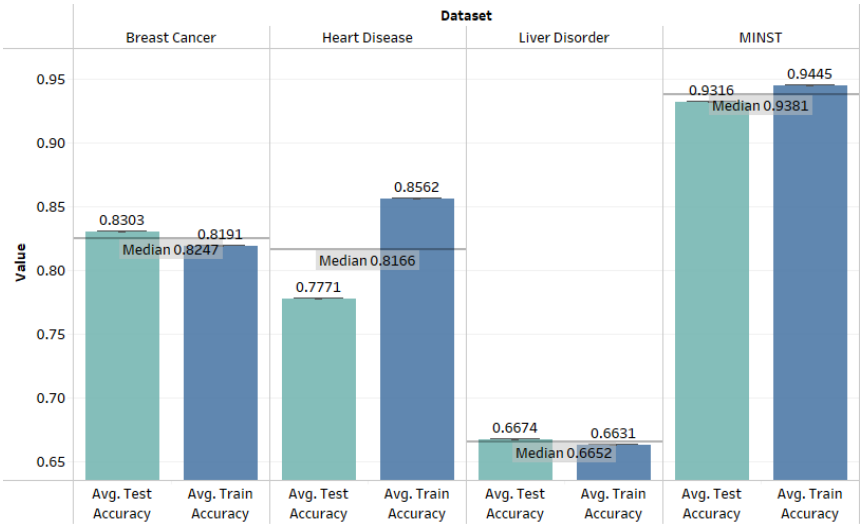

Fig. 5 Average train and test accuracies

Fig. 6 represents the train and test accuracy scatter plot with power trend line.

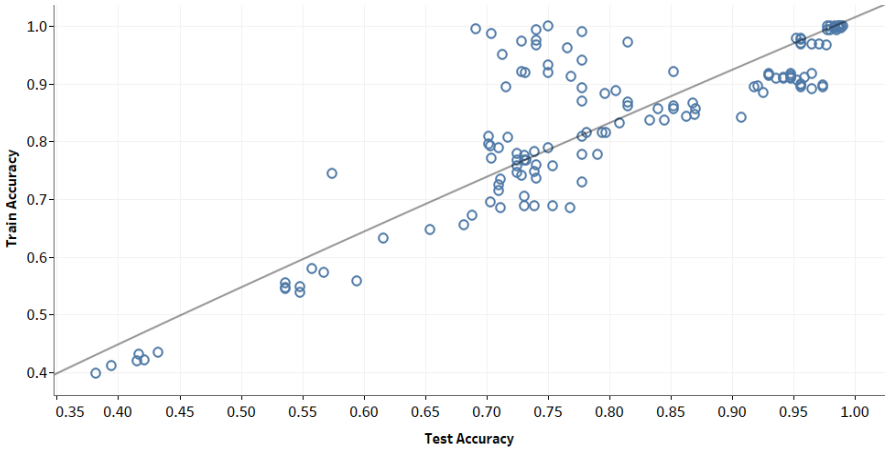

Fig. 6 Train and test accuracy scatter plot with power trend line

The upward-sloping trend line in Fig. 6 indicates that our analysis revealed a positive trend in the data. However, we also observed significant variability in the accuracy data points, with some combinations of hyperparameters yielding lower accuracies in the range of 0.38 to 0.43. Most individual accuracies fell within a central range of approximately 0.70 to 0.86.

We also noted the significant improvement in accuracy that resulted from hyperparameter tuning, with some data points achieving both high train and test accuracies in the range of 0.91 to 1.00. This improvement indicates that our goal of achieving the best possible train and test accuracies was successfully met through our hyperparameter optimization efforts.

While accuracy is a commonly used metric for evaluating classification models, it is not always the best measure of a model's performance. The AUC and the F1 score provide additional information about a model's ability to classify positive and negative instances accurately. AUC measures the model's ability to distinguish between positive and negative instances, regardless of the threshold used to make classification decision. The F1 score considers both precision and recall, balancing the two metrics.

When a class imbalance or misclassification of one class is more important than the other, the AUC and F1 score can be more informative metrics than accuracy. Therefore, it is necessary to calculate the AUC and F1 score in addition to accuracy to gain a more comprehensive understanding of a classification model's performance.

Fig. 7 presents the average AUC, F1 score, and test accuracy with varying train-test-split sizes of 0.2, 0.3, and 0.4 on four datasets. Our objective was to investigate how the average accuracy of the model changes with changes in validation or test size. With 0.3 validation size, the breast cancer dataset demonstrated maximum average accuracies of 84.00%, 81.00%, and 83.00% for all three-accuracy metrics, AUC, F1 score, and test accuracy, respectively. The heart disease dataset, on the other hand, produced the best results with 0.2 validation size. It has nearly the same F1 score and test accuracy of approximately 78.00% but 10.00% difference in AUC score.

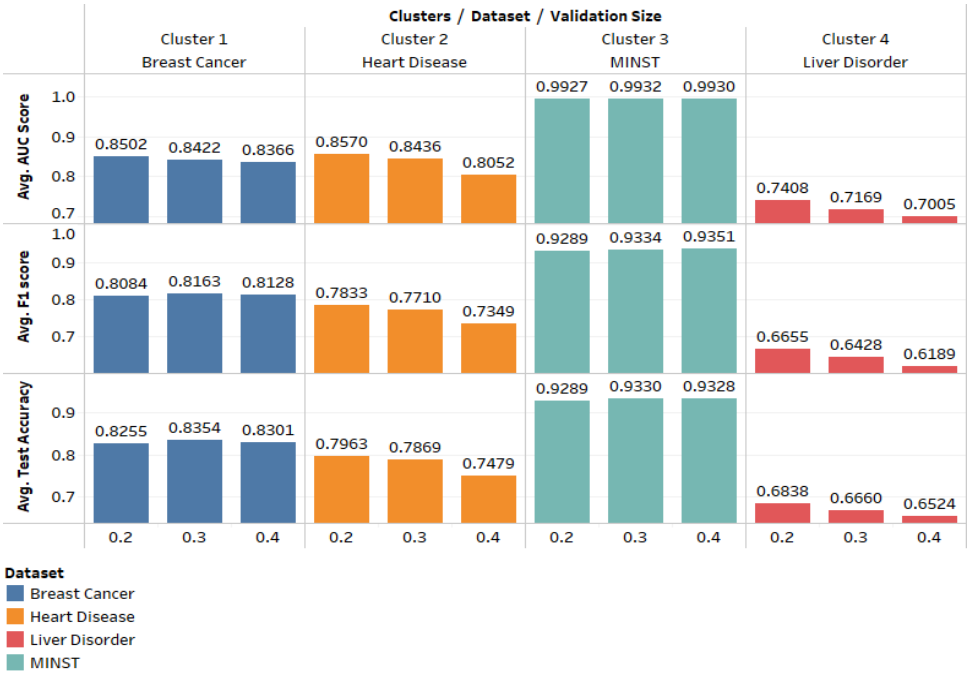

Fig. 7 Average AUC, F1 score, and test accuracy with varying validation sizes

Furthermore, with 0.4 validation size, the MINST dataset performed well, with the highest AUC score of 99.00%, demonstrating the model's ability to classify the true positives and true negatives of the dataset with a larger number of samples and feature count. It got 93.00% average accuracy score for F1 and test accuracy metrics. Lastly, the liver disorder dataset showed optimal results with 0.2 validation size, having the best AUC scores, and 64.00%, and 66.00% for F1 score and test accuracy, respectively.

Our analysis revealed that changing the validation size and hyperparameter tuning with different combinations significantly affected the model's accuracy. We observed some combinations yielding the best accuracy, while others resulted in lower accuracy. Using different metrics to assess the model's accuracy, we better understood its performance on different datasets.

In Fig. 8, the scatter plot demonstrates the correlation of all resultant accuracies between the test accuracy with the AUC and F1 scores. We employed a polynomial trend line to illustrate the relationship between these variables better. The *R*-squared value of polynomial trend line

between the test accuracy and AUC score is 0.7724, and *p*-value less than 0.0001 and in between test accuracy and F1 score, the *R*-squared value of the polynomial trend line between is 0.9901 and *p*-value less than 0.0001 indicating the best fit of resultant data points across this trend line. The trend line enables more precise visualization of the underlying relationship between the studied variables by fitting a polynomial equation to the data points.

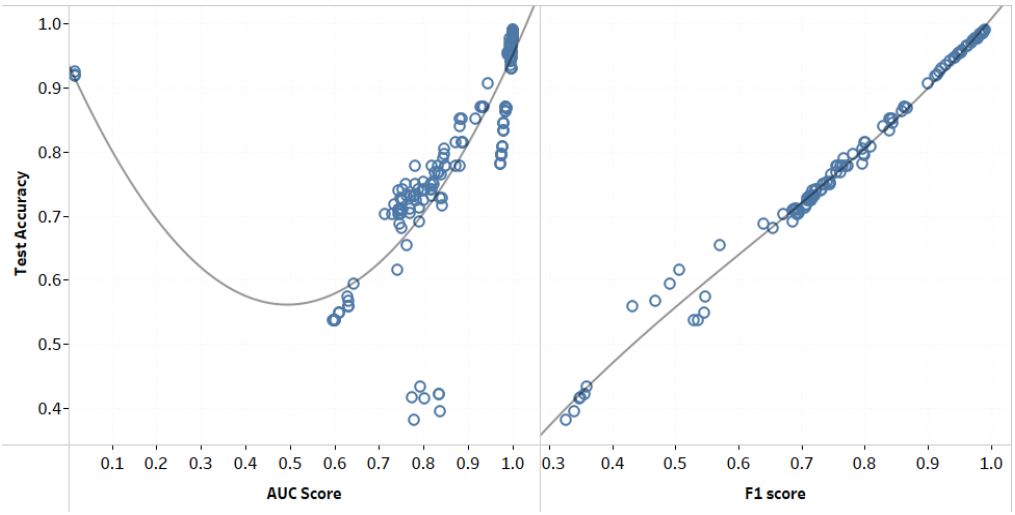

Fig. 8 Test accuracy, AUC, and F1 scores scatter plot with polynomial trend line

The scatter plot reveals a cluster of data points between 0.71 to 0.99 in between the test accuracy and AUC score graph. Additionally, outliers are observed, indicating instances where the results deviated from the expected trend. Furthermore, the scatter plot displays data points spread between 0.68 to 0.99 in the test accuracy and F1 score graph. We also observe that the accuracy between test and F1 score drops to 0.3 to 0.4. Some scatter is also observed between low and higher accuracies. These variations are attributable to the diverse hyperparameter combinations used in our analysis. The scatter plot represents the results obtained and highlights the disparities observed in the various hyperparameter combinations.

Fig. 9 depicts the range of minimum and maximum test accuracies obtained for all hyperparameter combinations. These accuracies were further categorized based on three different validation sizes of 0.2, 0.3, and 0.4. Our model achieved a maximum accuracy of 96.49% for the breast cancer dataset with a validation size of 0.3 and the hyperparameters previously mentioned. In contrast, the maximum accuracy of 90.74% was obtained for the heart disease dataset with a validation size of 0.2. Likewise, the optimal validation size for the liver disorder dataset was determined to be 0.2, yielding a maximum test accuracy of 76.81%. Lastly, the MINST dataset achieved 99.03% test accuracy with a validation size of 0.4, indicating that different datasets may require different validation sizes to achieve optimal performance. When compared to other datasets, MINST has the highest performance and accuracy levels.

Furthermore, Fig. 9 also presents the minimum test accuracies obtained for all the datasets with our model. It is worth noting that the resultant accuracies vary significantly depending on the hyperparameters used. For example, the minimum accuracy obtained for the breast cancer dataset was 38.00%, while the minimum accuracy obtained for the MINST dataset was 78.00%. These results suggest that the slightest hyperparameter change can significantly affect the model's accuracy. Thus, selecting optimal hyperparameters is paramount to achieving the best possible performance from our model.

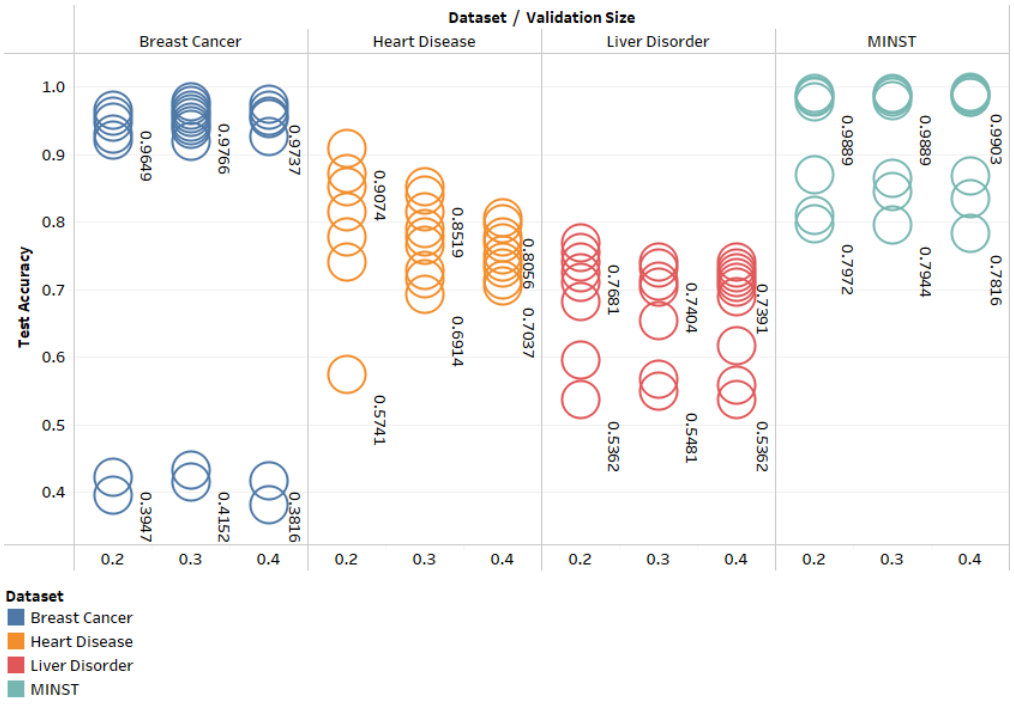

Fig. 9 Minimum and maximum test accuracies with different validation sizes

Fig. 10 highlights the range of F1 scores obtained at varying validation sizes. Our analysis revealed that breast cancer dataset got maximum F1 score of 97.49% with 0.3 validation size, the same as the highest test accuracy we got for this dataset. Moreover, the minimum F1 score for this dataset is 32.69% with 0.4 validation size.

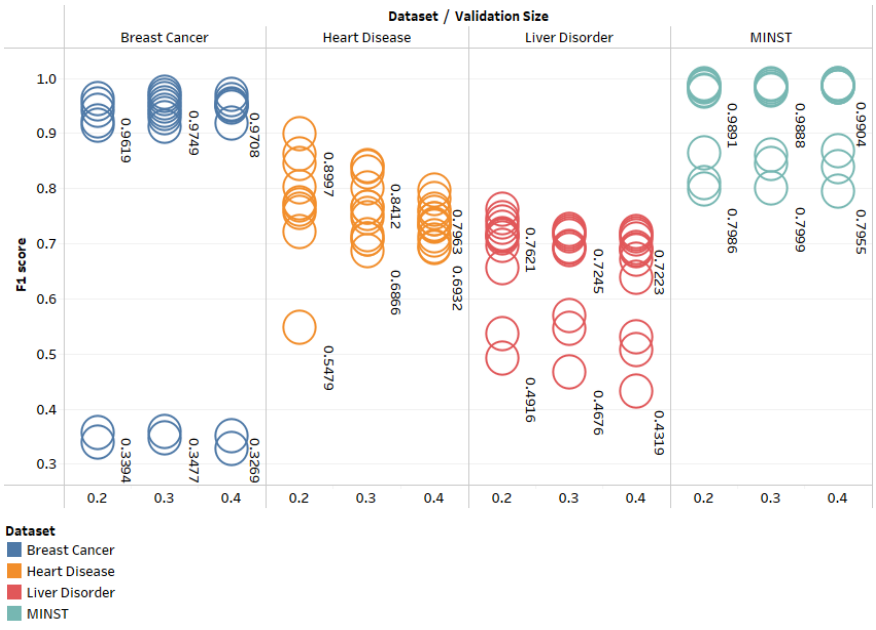

Fig. 10 Minimum and maximum F1 score with different validation sizes

Our findings indicate that the heart disease dataset achieved its maximum F1 score of 84.12% at a validation size of 0.2. Conversely, the minimum F1 score for this dataset was recorded at the same validation size but with different hyperparameter combinations, indicating the importance of hyperparameter tuning for optimal performance.

Furthermore, the liver disorder dataset yielded the maximum F1 score of 76.21% at a validation size of 0.2, while the minimum F1 score was observed to be 43.19%. Finally, our MINST dataset experiments showed that validation size of 0.4 was the best for achieving the highest F1 score of 99.04% and minimum of 79.55% with the same validation size, demonstrating that picking a suitable validation size and hyperparameters are crucial in achieving high test accuracy and F1 scores.

To thoroughly evaluate model performance, we calculated the AUC scores for the same four datasets to obtain the highest possible AUC score in addition to the F1 score and test accuracy. As presented in Fig. 11, our findings reveal that tuning hyperparameters in different combinations can lead to significant variations in AUC scores.

The breast cancer dataset's highest AUC score was 99.87% at 0.2 validation size. We noticed a significant difference in the AUC score in it because the minimum AUC score even dropped to 0.014% with a 0.2 validation size and different parameter combinations. This difference highlights how this dataset differs significantly from others regarding its AUC score. In the case of the heart disease dataset, we obtained a maximum AUC score of 94.37% with a validation size of 0.2; for some results, the AUC score drops to 62.91% with the same validation. Similarly, the liver disorder dataset had the highest AUC score of 82.45% at 0.2 validation size and the lowest drop to 60.00%. Finally, for the MINST dataset, the maximum AUC score achieved is 1.00% with a 0.2 validation size and 97.29 with a 0.4 validation size, with a 3.00% difference.

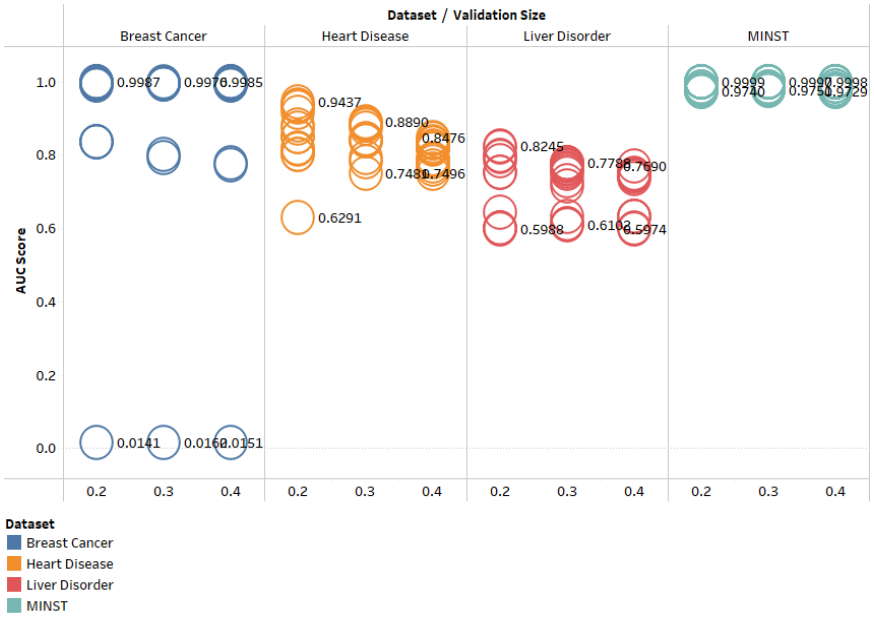

Fig. 11 Minimum and maximum AUC score with different validation sizes

We currently know which validation sizes provide the best test accuracy, F1 score and AUC score for four datasets. It is critical to understand which hyperparameter combinations are producing these results. To gain a deeper understanding of what parameter combinations led to the highest accuracy of metrics, we analyzed our models' performance in more detail. Specifically, we sought to identify the specific hyperparameter combinations that resulted in the best accuracy values for metrics and that could improve upon the default accuracy or other hyperparameter tuning approaches.

Fig. 12 illustrates the best hyperparameter combinations and optimal validation sizes, which give maximum accuracy scores for all three metrics across all datasets. For the breast cancer dataset, we found that a train-test-split size of 0.3 and hyperparameters tuned to a linear kernel, the C parameter value of 3, and any of the three possible inputs for degree resulted in the maximum scores across all evaluation metrics. Similarly, for the heart disease dataset, a validation size of 0.2 and hyperparameters tuned to a poly kernel, the C parameter value of 3, and degree set to 1 was the only combination that yielded the maximum scores.

In the case of the liver disorder dataset, a validation size of 0.2 and hyperparameters tuned to a linear kernel, with all three possible inputs for C parameter and degree, resulted in the maximum scores. Lastly, for the MINST dataset, a validation size of 0.4 and hyperparameters tuned to a poly kernel, with all three possible inputs for the C parameter and degree set to 3, resulted in the maximum scores. We further analyze the performance of the SVM by using CV from model selection with the same hyperparameter combinations and CV values of 3, 5, and 10.

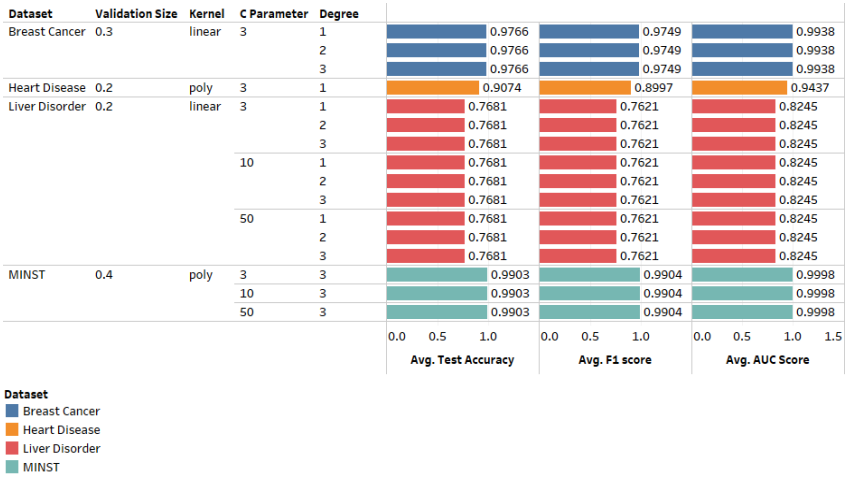

Fig. 12 Optimal hyperparameter combinations and validation size for maximum scores

Fig. 13 shows that for the MINST and breast cancer datasets, combining varying values of the C and the degree hyperparameters with a linear or polynomial kernel resulted in the highest average CV accuracies of up to 95.00%. Except for the heart disease dataset, the sigmoid kernel has the worst performance, and the average CV accuracy of its combinations falls to 40.00%. Results from CV and a train-test-split showed that the sigmoid kernel generally did not yield the best results, suggesting that its use should be avoided in practice.

Across the datasets, the MINST dataset has the highest performance, with an average CV accuracy of 95.07%, followed by the breast cancer dataset, which is only 0.2% less accurate. The greater number of samples and features in both datasets is responsible for the best accuracy. Notably, breast cancer has the lowest CV accuracy of 40.00% when utilizing sigmoid kernel combinations. Furthermore, due to low feature count, the liver disorder dataset performs poorly with CV model like train-test-split model, with a CV accuracy of up to 68.41% and a drop to 54.02% with a sigmoid kernel. Lastly, the heart disease dataset got the highest average CV accuracy of 83.21% with linear kernel and lowest CV accuracy of 73.46% using a sigmoid kernel combination with degree 50 and all possible inputs of C parameter. These outcomes underscore the significance of proper hyperparameter tuning and model selection in achieving robust results in ML applications.

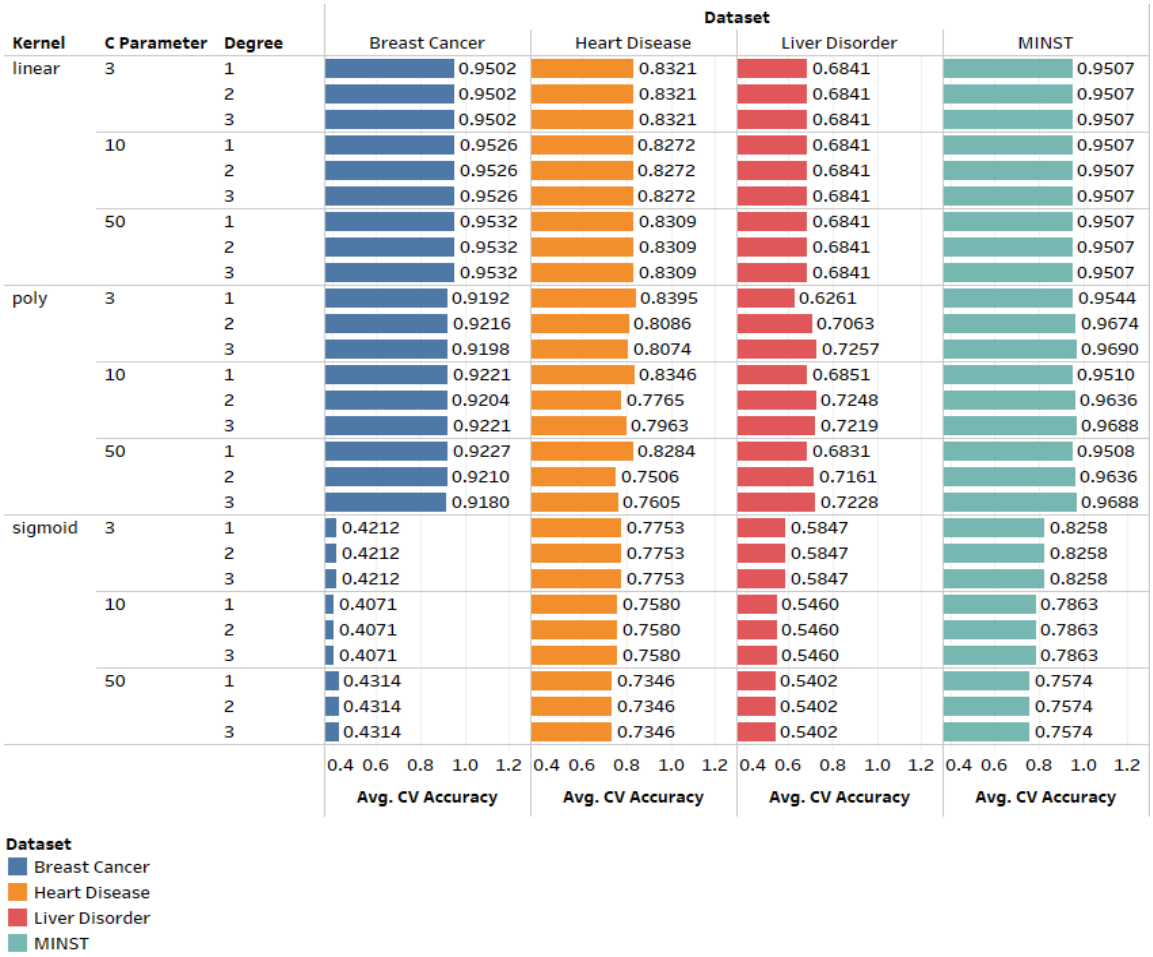

Fig. 13 Cross validation accuracies vary with parameters combination

We further analyzed the average CV accuracies with varied folds to gain a more comprehensive understanding of SVM performance on the datasets. This allowed us to assess the robustness of our model and identify any potential weaknesses or areas for improvement.

Fig. 14 shows the average CV accuracy for the four datasets with varied CV fold values. For the breast cancer dataset, we found that the highest average CV accuracy was achieved with a 5-fold value of 77.19%, while the lowest was achieved with a 10-fold value of 75.79%. Conversely, for the heart disease dataset, 3-fold results in the highest average CV accuracy of 80.07% and 5-fold results in the lowest average of 79.00%.

Overall, the liver disorder dataset has the worst performance, but the highest average CV accuracy we obtained was 64.93% with a 3-fold value, and the lowest was 64.61% with a 10-fold value. Lastly, the MINST dataset with 10-fold got the highest average CV accuracy and the lowest of 89.68 with 3-fold. Fig. 13 also demonstrates that the average CV score across all data sets is 77.70%. These findings highlight the significance of CV folds in achieving maximum CV accuracies. Despite hyperparameter combinations, the selection of CV fold also significantly affects the performance of SVM in CV model.

To delve deeper into evaluating our CV model for SVM, we employed the F1 score metric to assess its performance. By incorporating multiple evaluation metrics, we can obtain a more comprehensive understanding of how our model performs and identify any improvement areas.

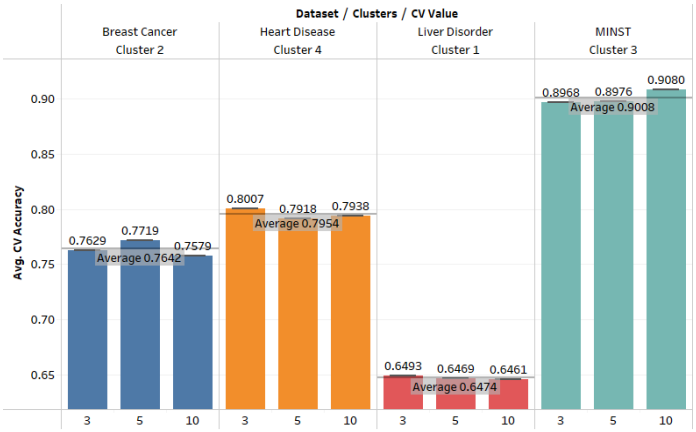

Fig. 14 Average CV accuracies with varied CV fold values

Fig. 15 presents our analysis of the F1 score for our CV model for SVM with 3, 5, and 10-fold values on four datasets. We found that the highest F1 score, 75.22% was achieved with a 5-fold value for the breast cancer dataset, suggesting that a moderate number of folds may be optimal for this dataset.

In contrast, the heart disease dataset exhibited the highest F1 score of 79.75% with a 3-fold value. Finally, a 10-fold value and an F1 score of 90.87% and 61.55%, respectively, were seen to be the best in the MINST and liver disorder datasets. It suggests that higher folds are needed for these datasets to achieve the highest F1 score. Fig. 13 also shows that the average F1 score across the datasets is 76.23%.

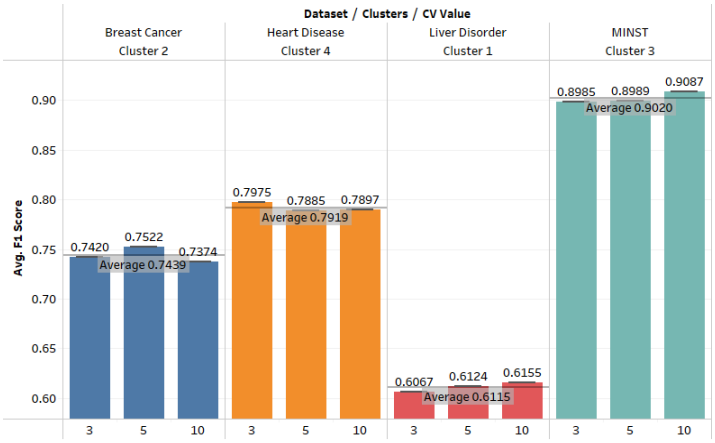

Fig. 15 Average F1 score with varied CV fold values

Interestingly, our results showed that the optimal fold value for the F1 score was similar to that for average CV accuracies for each dataset, except for the liver disorder dataset. When we choose the best combination of hyperparameters and fold value, we consider different metrics scores and select those that perform best for all metrics.

Fig. 16 shows the individual result entities scatter plot between CV accuracy and F1 score that we obtained from our model of combinatorial hyperparameter tuning. The power trend line has been used to better understand the data trend. The *R*-squared value of the trend line is 0.9847, which indicates a strong positive correlation between the hyperparameter combinations and the accuracy of the SVM model. The *p*-value is less than 0.0001, indicating that the observed correlation is statistically significant.

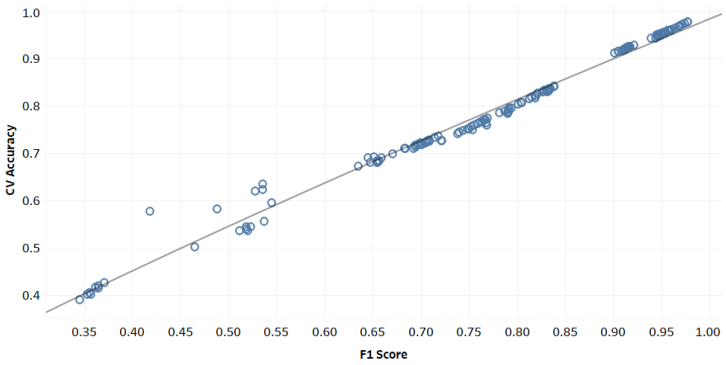

Fig. 16 CV accuracy and F1 score scatter plot with power trend line

The scatter plot reveals a cluster of result data points with accuracies ranging from 0.65 to 0.85, suggesting that most of the tested hyperparameter combinations yield relatively high accuracy for the SVM model. Additionally, some data points with higher accuracy ranging from 0.91 to 0.97 were also obtained, indicating that certain combinations of hyperparameters perform exceptionally well.

On the other hand, some data points have lower accuracy, ranging from 0.41 to 0.59, which suggests that some hyperparameter combinations are less effective for the SVM model. Moreover, some data points with the lowest values of 0.3 to 0.4 indicate that certain combinations of hyperparameters yield poor accuracy.

Using different fold values, our study also explored the maximum and minimum CV accuracies obtained for the four datasets. The results, depicted in Fig. 17, show that for the breast cancer dataset, the maximum CV accuracy of 95.96% was obtained with a CV fold value of 3 and a minimum of 40.24% with 3 and 5 folds but with different hyperparameter combinations. On the other hand, the heart disease dataset saw a maximum CV accuracy of 84.07% with 3 and 5 folds and a minimum of 72.59% with 5 and 10 folds using varying hyperparameters.

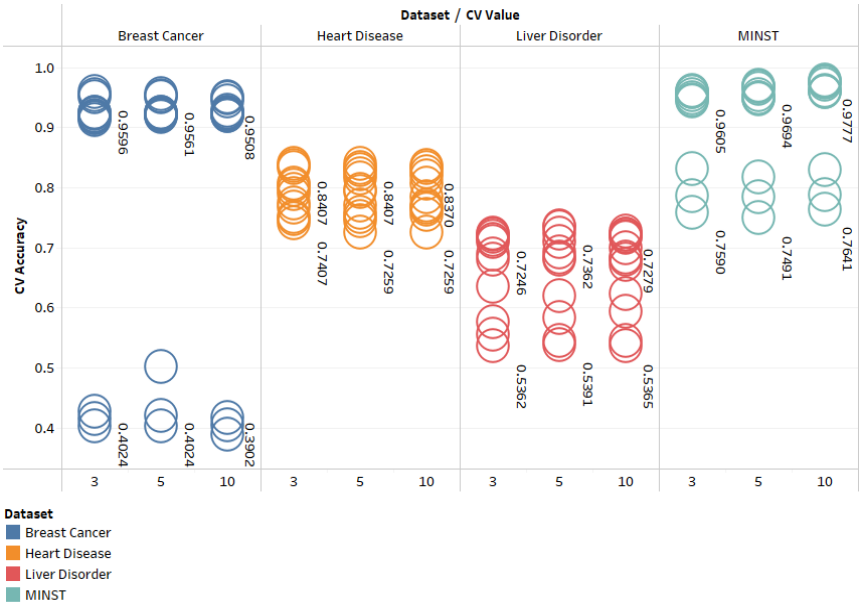

Fig. 17 Minimum and maximum CV accuracy with varied fold values

Conversely, the liver disorder dataset exhibited a maximum CV accuracy of 73.62% with a 5-fold value and a minimum of 53.62% with a 3-fold value. The MINST dataset obtained the maximum accuracy of 97.77% with a 10-fold value, and its minimum accuracy drops to 74.91% with a 5-fold value. These findings indicate the maximum accuracies we can get for different datasets of varying samples and features with different hyperparameter tuning and CV fold values.

Maximum and minimum F1 scores for the four datasets with different fold values are displayed in Fig. 18. The breast cancer dataset has an impressive maximum F1 score of 95.65% at CV value of 3-folds and a minimum F1 score of 34.53% at CV value of 10-folds. The maximum F1 score we obtain for the heart disease dataset is 83.83% at a 3-fold, and the minimum is 72.18% at a 10-fold. Furthermore, for the liver disorder dataset, the maximum F1 score we obtain is 71.81% with 5-folds, while the minimum F1 score we obtain is 41.83% with 3-folds. Finally, for the MINST dataset, we get the maximum F1 score of 97.74% with 10- folds and minimum F1 score of 75.36% with 5-folds.

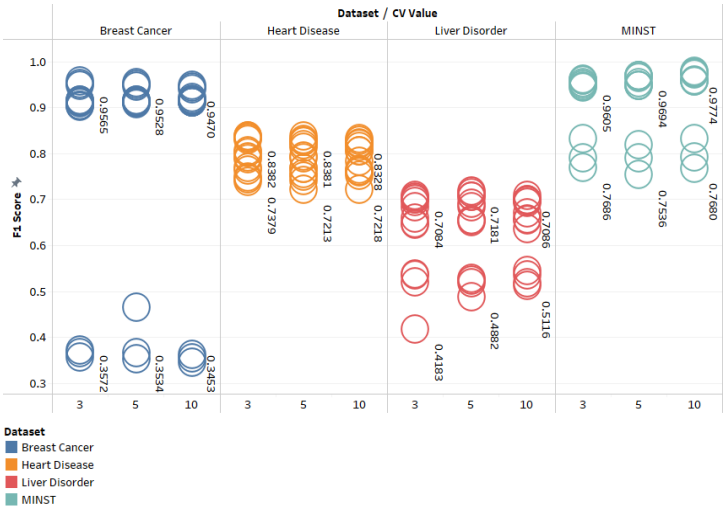

Fig. 18 Minimum and maximum F1 scores with varied fold values

These results demonstrate the effectiveness of hyperparameter tuning in maximizing the F1 score and CV accuracy with the same CV fold value. In light of these findings, we selected the most promising hyperparameter combinations that have produced these outstanding results. The selected optimal combinations are depicted in Fig. 19.

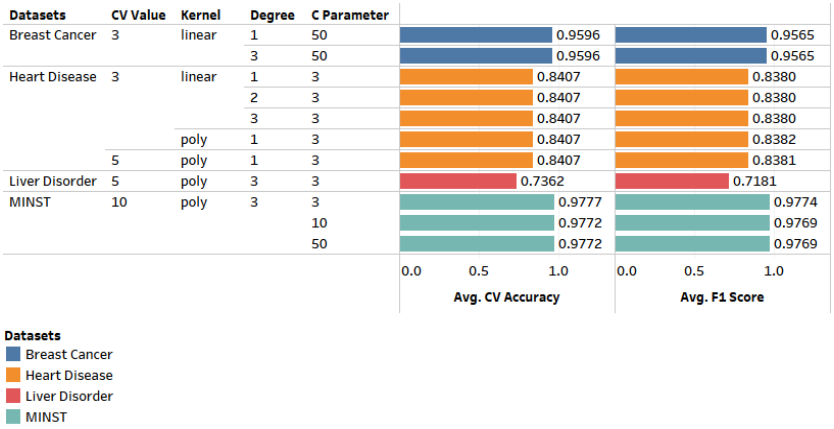

Fig. 19 Optimal hyperparameter combinations and CV fold values for maximum scores

The findings presented in Fig. 19 demonstrate the optimal combinations of hyperparameters and CV fold values for maximizing the performance of our SVM model on each dataset. For the breast cancer dataset, selecting a 3-fold CV value, a linear kernel, a degree parameter of either 1 or 3, and a C parameter of 50 results in the maximum CV and F1 score.

For the heart disease dataset, selecting either the 3 or 5-fold cross, and tuning the kernel parameter to either linear or poly, a degree parameter of 1 for linear kernel and all possible inputs for the poly kernel, and a C parameter of 3 gives the best results. In the case of the liver disorder dataset, a 5-fold CV value with a poly kernel, a degree parameter of 3, and C parameter of 3 gives the maximum score. Lastly, for the MINST dataset, setting a 10-fold CV value, tuning the kernel parameter to poly, a degree parameter of 3, and exploring all possible inputs for the C parameter yield the maximum results. These optimal hyperparameter combinations are critical for maximizing the performance of our SVM model in classifying these datasets. In order to determine which model, train-test-split or CV, performs better with our chosen hyperparameter combinations, we compared their average model evaluation accuracies.

Our final results for choosing the optimal model to optimize SVM performance across four datasets are shown in Fig. 20. It suggests that the accuracy of SVM on the breast cancer, liver disorder, and MINST datasets, where the number of samples ranges from 1 797 to 345 and the number of features ranges from 64 to 6, was found to be significantly improved by using the train-test-split model with varying validation sizes of 0.2, 0.3, and 0.4 and combinatorial hyperparameter tuning of C, degree, and kernel hyperparameters. On the other hand, the heart disease dataset with only 270 samples and a moderate number of features (13) shows the best SVM accuracy when using a CV model with CV fold value 3, 5, and 10 and combinatorial hyperparameters tuning.

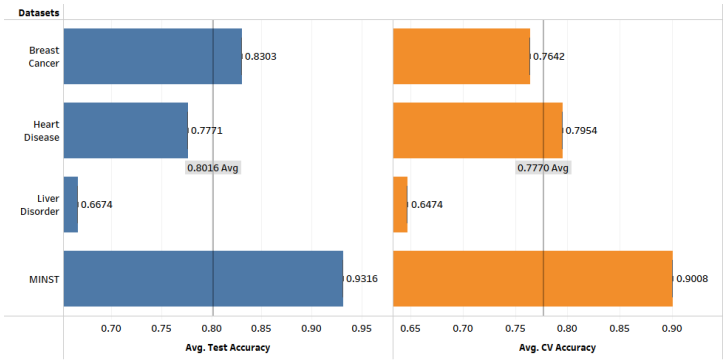

Fig. 20 Contrast of train-test-split and CV model average accuracy

The overall average accuracy of the train-test-split model is 80.00%, which is higher than that of the CV model, with an overall average accuracy of 77.00%. These findings suggest that the train-test-split model is a more effective and efficient approach for SVM optimization than CV.

Discussion

We found that the performance of SVM on different datasets can be significantly maximized by tuning the hyperparameters kernel, C, and degree. Except for a single dataset, where CV performs admirably, train-test-split with varying validation sizes achieves impressive results. It depends on the sample size and dataset features. We successfully determined which hyperparameters and model values, such as the cross-fold value for CV and the validation size for train-test-split, were best for each dataset. SVM's accuracy has been shown to improve significantly using a variety of accuracy metrics.

In terms of novelty, our study builds upon existing research by examining the performance of SVMs on multiple datasets with varying characteristics. While SVMs have been widely studied, our analysis focused on identifying the optimal hyperparameter combinations and validation strategies specifically for each dataset. By doing so, we could highlight the key factors contributing to improved SVM performance. Our findings will aid data scientists in selecting the optimal model and hyperparameters for their datasets, considering the available feature and sample sizes. This highlights the need to explore and optimize hyperparameters to achieve optimal results thoroughly.

Additionally, our results emphasize the importance of considering the dataset's characteristics when selecting the most suitable validation approach. We found that train-test-split with varying validation sizes yielded higher accuracy for datasets with higher samples and features, such as breast cancer and MINST. In contrast, the heart disease dataset demonstrated improved performance with CV, suggesting that the dataset size and complexity play a crucial role in determining the optimal validation strategy.

Comparing our results with previous studies, we observed consistent trends regarding the impact of hyperparameters on SVM performance. Our findings align with existing knowledge that linear and polynomial kernels yield better results than the sigmoid kernel [5, 27]. This reaffirms the established paradigm and supports the notion that the sigmoid kernel may not be the most effective choice for SVM applications in practice. The user involvement and new framework fulfills the gaps of automatic algorithms for SVM performance enhancement with more computational time and no manual tuning [14, 29].

While our study provides valuable insights into SVM performance, it is essential to acknowledge its limitations. The analysis was conducted on specific datasets, and the optimal hyperparameter combinations may vary for different datasets and classification tasks. Therefore, further studies are needed to validate our findings across broader datasets.

Future research should aim to optimize SVM performance on large-scale datasets and high-dimensional feature spaces, addressing big data challenges. Additionally, investigating the incorporation of ensemble techniques or deep learning models can improve SVM performance and increase the scope of its applications in real-world applications.

In conclusion, our study underscores the significance of hyperparameter tuning and model selection in achieving optimal SVM performance. By identifying the optimal hyperparameter combinations and validation approaches for each dataset, we obtained valuable insights into maximizing SVM's model accuracy, F1 score, and AUC scores. These findings contribute to the broader understanding of SVMs in ML applications and provide a foundation for further research and improvements in classification tasks.

Conclusion

Our results show that SVM's performance can be improved on datasets with varying sample sizes and feature counts by employing combinatorial hyperparameter tuning and various model selection techniques, such as CV and train-test-split. The intended framework enables the selection of the hyperparameters kernel, C, and degree with various inputs, and their combination enhances the overall performance of SVM for binary and multi-class classification tasks.

References

1. Abubakar M., O. EriOluwa, M. Teyei, F. Al-Turjman (2022). AI Application in the Aviation Sector, International Conference on Artificial Intelligence of Things and Crowdsensing (AIoTCs), 52-55.
2. Adem H. M., A. W. Tessema, G. L. Simegn (2022). Classification of Parkinson's Disease Using EMG Signals from Different Upper Limb Movements Based on Multiclass Support Vector Machine, International Journal Bioautomation, 26(1), 109.
3. Ahmad I., M. Hussain, A. Alghamdi, A. Alelaiwi (2014). Enhancing SVM Performance in Intrusion Detection Using Optimal Feature Subset Selection Based on Genetic Principal Components, Neural Computing and Applications, 24, 1671-1682.
4. Bäck T., H. P. Schwefel (1993). An Overview of Evolutionary Algorithms for Parameter Optimization, Evolutionary Computation, 1(1), 1-23.
5. Ben-Hur A., J. Weston (2010). A User's Guide to Support Vector Machines, Data Mining Techniques for the Life Sciences, 223-239.
6. Bergstra J., Y. Bengio (2012). Random Search for Hyper-parameter Optimization, The Journal of Machine Learning Research, 13(2), 281-305.
7. Berry M. W., A. Mohamed, B. W. Yap (2020). Supervised and Unsupervised Learning for Data Science, Cham, Switzerland: Springer.
8. Bi Q., K. E. Goodman, J. Kaminsky, J. Lessler (2019). What is Machine Learning? A Primer for the Epidemiologist, American Journal of Epidemiology, 188(12), 2222-2239.
9. Chacón A. M. P., I. S. Ramírez, F. P. G. Márquez (2023). K-nearest Neighbour and K-fold Cross-validation Used in Wind Turbines for False Alarm Detection, Sustainable Futures, 6, 100132.
10. Chapelle O., B. Schölkopf, A. Zien (2006). Introduction to Semi-supervised Learning, MIT Press, 1-12.
11. Chidambaram S., K. Srinivasagan (2019). Performance Evaluation of Support Vector Machine Classification Approaches in Data Mining, Cluster Computing, 22, 189-196.
12. Czarnecki W. M, S. Podlewska, A. J. Bojarski (2015). Robust Optimization of SVM Hyperparameters in the Classification of Bioactive Compounds, Journal of Cheminformatics, 7, 1-15.
13. Da Silva Santos C. E., R. C. Sampaio, L. Dos Santos Coelho, G. A. Bestard, et al. (2021). Multi-objective Adaptive Differential Evolution for SVM/SVR Hyperparameters Selection, Pattern Recognition, 110, 107649.
14. Das M. C., R. Dash, S. C. Swain, V. Subburaj (2021). Performance Enhancement of PI-controller Using SVM for DFIG-grid Interconnected System, 2nd International Conference for Emerging Technology, 1-6.
15. Duan K., S. S. Keerthi, A. N. Poo (2003). Evaluation of Simple Performance Measures for Tuning SVM Hyperparameters, Neurocomputing, 51, 41-59.
16. El Naqa I., M. J. Murphy (2015). What is Machine Learning, Machine Learning in Radiation Oncology: Theory and Applications, 3-11.
17. Faleiros C. M., M. H. Nogueira-Barbosa, V. F. Dalto, J. R. Ferreira, et al. (2020). Machine Learning Techniques for Computer-aided Classification of Active Inflammatory Sacroiliitis in Magnetic Resonance Imaging, Advances in Rheumatology, 60, 25.
18. Fushiki T. (2011). Estimation of Prediction Error by Using K-fold Cross-validation, Statistics and Computing, 21, 137-146.
19. Gaspar P., J. Carbonell, J. L. Oliveira (2012). On the Parameter Optimization of Support Vector Machines for Binary Classification, Journal of Integrative Bioinformatics, 9, 33-43.

20. Gold C., A. Holub, P. Sollich (2005). Bayesian Approach to Feature Selection and Parameter Tuning for Support Vector Machine Classifiers, Neural Networks, 18(5-6), 693-701.
21. Goutte C., E. Gaussier (2005). A Probabilistic Interpretation of Precision, Recall and F-score with Implication for Evaluation, European Conference on Information Retrieval, 345-359.
22. Harris C. R., K. J. Millman, S. J. Van Der Walt, R. Gommers, et al. (2020). Array Programming with NumPy, Nature, 585(7825), 357-362.
23. Holmes J., L. Sacchi, R. Bellazzi (2004). Artificial Intelligence in Medicine, Annals of the Royal College of Surgeons of England, 86, 334-8.
24. Hussain S., Z. Raza, G. Giacomini, N. Goswami (2021). Support Vector Machine-based Classification of Vasovagal Syncope Using Head-up Tilt Test, Biology, 10(10),1029.
25. Ji X. (2017). Application of Improved SVM Image Segmentation Algorithm in Computer Tomography Image Analysis, International Journal Bioautomation, 21(1), 59-68.
26. Jiménez Á. B., J. L. Lázaro, J. R. Dorronsoro (2008). Finding Optimal Model Parameters by Discrete Grid Search, Innovations in Hybrid Intelligent Systems, 120-127.
27. Kalcheva N., M. Karova, I. Penev (2020). Comparison of the Accuracy of SVM Kemel Functions in Text Classification, International Conference on Biomedical Innovations and Applications, 141-145.
28. Kambalimath S. S., P. C. Deka (2021). Performance Enhancement of SVM Model Using Discrete Wavelet Transform for Daily Streamflow Forecasting, Environmental Earth Sciences, 80(3), 101.
29. Kang D. K., M. J. Kim (2010). Performance Enhancement of SVM Ensembles Using Genetic Algorithms in Bankruptcy Prediction, 3rd International Conference on Advanced Computer Theory and Engineering, 2, 154.
30. Kazemi A., R. Boostani, M. Odeh, M. R. Al-Mousa (2022). Two-layer SVM, Towards Deep Statistical Learning, International Engineering Conference on Electrical, Energy and Artificial Intelligence, 1-6.
31. Kohavi R. (1995). A Study of Cross-validation and Bootstrap for Accuracy Estimation and Model Selection, International Joint Conferences on Artificial Intelligence, 14(2), 1137-1145.
32. Le C., T. Bruns, D. Tortorelli (2011). A Gradient-based, Parameter-free Approach to Shape Optimization, Computer Methods in Applied Mechanics and Engineering, 200(9-12), 985-996.
33. Lee Y. J., O. L. Mangasarian (2001). SSVM: A Smooth Support Vector Machine for Classification, Computational Optimization and Applications, 20, 5-22.
34. Leeflang M. M. G., J. J. Deeks, C. Gatsonis, P. M. Bossuyt (2008). Systematic Reviews of Diagnostic Test Accuracy, Annals of Internal Medicine, 149(12), 889-897.
35. Lin C. H., J. C. Liu, C. H. Ho (2008). Anomaly Detection Using LibSVM Training Tools, International Conference on Information Security and Assurance, 166-171.
36. Liu Y., B. Wang, R. Li, S. He, et al. (2020). Relapse Risk Prediction for Children with Henoch-Schönlein Purpura based on GA-SVM, International Journal Bioautomation, 24(2), 117-130.
37. Mahesh B. (2020). Machine Learning Algorithms – A Review, International Journal of Science and Research, 9(1), 381-386.
38. Mantovani R. G., A. L. Rossi, J. Vanschoren, B. Bischl, et al. (2015). Effectiveness of Random Search in SVM Hyper-parameter Tuning, International Joint Conference on Neural Networks, 1-8.
39. McCarthy J. (1987). Generality in Artificial Intelligence, Communications of the ACM, 30(12), 1030-1035.

40. McKinney W. (2010). Data Structures for Statistical Computing in Python. SciPy, 445(1), 51-56.
41. Molina M. M., J. M. Luna, C. Romero, S. Ventura (2012). Meta-learning Approach for Automatic Parameter Tuning: A Case Study with Educational Datasets, 5th International Conference on Educational Data Mining, 180-183.
42. Moroshko E., B. E. Woodworth, S. Gunasekar, J. D. Lee, et al. (2020). Implicit Bias in Deep Linear Classification: Initialization Scale vs Training Accuracy, Advances in Neural Information Processing Systems, 33, 22182-22193.
43. Ng A. (2016). What Artificial Intelligence Can and Can't Do Right Now, Harvard Business Review, 9(11), 1-4.
44. Noble W. S. (2006). What is a Support Vector Machine, Nature Biotechnology, 24(12), 1565-1567.
45. Patle A., D. S. Chouhan (2013). SVM Kernel Functions for Classification, International Conference on Advances in Technology and Engineering, 1-9.
46. Pawluszek-Filipiak K., A. Borkowski (2020). On the Importance of Train-test Split Ratio of Datasets in Automatic Landslide Detection by Supervised Classification, Remote Sensing, 12(18), 3054.
47. Pedregosa F., G. Varoquaux, A. Gramfort, V. Michel, et al. (2011). Scikit-learn: Machine Learning in Python, The Journal of Machine Learning Research, 12, 2825-2830.
48. Rodi W. (2006). Grid-search Event Location with Non-Gaussian Error Models, Physics of the Earth and Planetary Interiors, 158(1), 55-66.
49. Rojas-Domínguez A., L. C. Padiema, J. M. C. Valadez, H. J. Puga-Soberanes, et al. (2017). Optimal Hyperparameter Tuning of SVM Classifiers with Application to Medical Diagnosis, IEEE Access, 6, 7164-7176.
50. Rosset S. (2004). Model Selection via the AUC, 21th International Conference on Machine Learning, 89.
51. Roy K., A. S. Mandal (2009). Predictive QSAR Modeling of CCR5 Antagonist Piperidine Derivatives Using Chemometric Tools, Journal of Enzyme Inhibition and Medicinal Chemistry, 24(1), 205-223.
52. Russell S. J., P. Norvig, P. (2010). Artificial Intelligence a Modern Approach, Pearson.
53. Salazar J. J., L. Garland, J. Ochoa, M. J. Pyrcz (2022). Fair Train-test Split in Machine Learning: Mitigating Spatial Autocorrelation for Improved Prediction Accuracy, Journal of Petroleum Science and Engineering, 209, 109885.
54. Schratz P., J. Muenchow, E. Iturritxa, J. Richter, et al. (2018). Performance Evaluation and Hyperparameter Tuning of Statistical and Machine-learning Models Using Spatial Data, arXiv Preprint arXiv:1803.11266.
55. Sherin B. M., M. H. Supriya (2015). Selection and Parameter Optimization o SVM Kernel Function for Underwater Target Classification, Underwater Technology, 1-5.
56. Singh A., N. Thakur, A. Sharma (2016). A Review of Supervised Machine Learning Algorithms, 3rd International Conference on Computing for Sustainable Global Development, 1310-1315.
57. Syarif I., A. Prugel-Bennett, G. Wills (2016). SVM Parameter Optimization Using Grid Search and Genetic Algorithm to Improve Classification Performance, Telecommunication Computing Electronics and Control, 14(4), 1502-1509.
58. Tan J., J. Yang, S. Wu, G. Chen, et al. (2021). A Critical Look at the Current Train/Test Split in Machine Learning, arXiv Preprint arXiv:2106.04525.
59. Tharwat A. (2019). Parameter Investigation of Support Vector Machine Classifier with Kernel Functions, Knowledge and Information Systems, 61, 1269-1302.

60. Vanschoren J., J. N. Van Rijn, B. Bischl, L. Torgo (2014). OpenML: Networked Science in Machine Learning, ACM SIGKDD Explorations Newsletter, 15, 49-60.
61. Vanwinckelen G., H. Blockeel, B. De Baets, B. Manderick, et al. (2012). On Estimating Model Accuracy with Repeated Cross-validation, 21st Belgian-Dutch Conference on Machine Learning, 39-44.
62. Vishwanathan S. V. M., M. N. Murty (2002). SSVM: A Simple SVM Algorithm, International Joint Conference on Neural Networks, 3, 2393-2398.
63. Wenzel F., J. Snoek, D. Tran, R. Jenatton (2020). Hyperparameter Ensembles for Robustness and Uncertainty Quantification, Advances in Neural Information Processing Systems, 33, 6514-6527.
64. Wu J., X. Y. Chen, H. Zhang, L. D. Xiong, et al. (2019). Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimization, Journal of Electronic Science and Technology, 17(1), 26-40.
65. Yadav S., S. Shukla (2016). Analysis of K-fold Cross-validation over Hold-out Validation on Colossal Datasets for Quality Classification, 6th International Conference on Advanced Computing, 78-83.
66. Yang Y., J. Li, Y. Yang (2015). The Research of the Fast SVM Classifier Method, 12th International Computer Conference on Wavelet Active Media Technology and Information Processing, 121-124.
67. http://archive.ics.uci.edu/ml (Access date 02 June 2025).
68. https://www.tableau.com/resource/business-intelligence (Access date 02 June 2025).

Assist. Prof. Hassan Tariq, Ph.D.
E-mail: hassan@uaf.edu.pk

Dr. Hassan Tariq got his Ph.D. degree in Computer Sciences from the Victoria University of Wellington, New Zealand. Currently, he is serving as an Assistant Professor in the Department of Computer Science, University of Agriculture, Faisalabad, Pakistan. He has more than 14 years of teaching and research experience. He is the head of various departmental committees. He is also a convener of stakeholders and industrial linkages. He served as an examiner of various Master and Ph.D. thesis.

Mehwish Majeed, M.Sc. Student
E-mail: mehwishmajeed86@gmail.com

Mehwish Majeed graduated with B.Sc. degree from the Department of Computer Science, University of Agriculture, Faisalabad, Pakistan in 2023. Her major research interests are in the fields of bioinformatics and machine learning methods applications for various bioinformatics problems. Currently, she is pursuing her Master degree in China.

Mueed Ahmad, M.Sc. Student
E-mail: ahmedmueed822@gmail.com

Ahmad Mueed graduated from the Department of Computer Science, University of Agriculture, Faisalabad, Pakistan in 2023. His major research interests are in the fields of bioinformatics and machine learning methods applications for various bioinformatics problems. Currently, he is pursuing his Master degree in China.