**Support Vector Machines** are a collection of different machine learning techniques, which includes support vector regressor, support vector classifier. These methods can be used for different projects, depending on the goal of the machine learning model. In our case, the support vector classifier is used since we aimed to classify different diseases within 8 different classes. The regressor is used in cases where the model must predict a value instead of a class.

The support vector classifier is a powerful machine learning model, which can accurately learn and predict new data. The algorithm is also one of the faster ones when it comes to prediction times. The standard parameters provided with the **SVM** usually gives good results, however it is possible to tune the model to give even better results, for this we decided to use the GridSearchCV functions from SkLearn.

The best way to understand how the support vector classifier works is to see 2d data classified by the classifier. This also allows for easier understanding on how the different kernels works. Kernels are mathematical equations that decides on how the model will classify, the SVC has four different kernels, which are linear, polynomial, rbf and sigmoid. Which one used depends on what is required from the project. For simplicity there is an image illustration below (*image 3*) which displays three of the different kernels.

Et bilde som inneholder tekst, hest, skjermbilde

Automatisk generert beskrivelse

*Image 3 (*[*https://scikit-learn.org/stable/modules/svm.html*](https://scikit-learn.org/stable/modules/svm.html)*)*

these images display how the model would separate the iris dataset, comparing sepal width and sepal length depending on which kernel is being used. This illustration shows how the kernels would classify a 2d dataset and would create a line separating the different classes. If the data has more than two dimensions, the line would be called a hyperplane, which would be like a piece of paper shaped to separate the different classes in multidimensional space.

The way that the support vector classifier generates these hyperplanes is by calculating where there is the most separation between the datapoints, this is quite efficient since the algorithm only must consider the datapoints that are the closest to this separation as shown in the image below (*image 4*).

Et bilde som inneholder skjermbilde, line, diagram, Plottdiagram

Automatisk generert beskrivelse

This separation in the data will be the decision boundaries for the classification, as shown in this image the two orange datapoints and the blue datapoint form what is called a **support vector**, which allows the algorithm to calculate the shape of the hyperplane.

*image 4* (<https://scikit-learn.org/stable/modules/svm.html>)

During our testing and evaluation we found the support vector classifier to do a great job on the skin disease dataset, we gathered a few different results depending on if the training was done on the original dataset, augmented dataset and graph dataset from both of these which we generated within our DataProcessor class using the generate\_xy\_by\_axis() function. To summarize the results, we gathered are shown in the table below (*table 3*)

|  |  |  |
| --- | --- | --- |
| **Dataset** | **Graph Data / Image Data** | **Accuracy** |
| Original | Image Data | 56% |
| Augmented | Image Data | 77% |
| Original | Graph Data | 34% |
| Augmented | Graph Data | 49% |

*table 3*

In this table there is one model that did particularly well, which is the graph data on the augmented dataset. The result was not achieved using the standard svc hyperparameters, these hyperparameters were tuned using GridSearchCV in the SciKit-learn library. This allows for the machine learning engineer to create a dictionary with hyperparameters. This dictionary is then used to generate different folds of parameters that will be trained and evaluated. The hyperparameters that were tuned in our model was C and gamma, by tuning these we increased these results by approximately 20%. The reason we decided to only use GridSearchCV on the graph data was considering the computing time it would have taken if we used the entire image data consisting of around 150 000 parameters, compared to the graph images with 672 parameters.

We found the support vector classifier to work better than expected on image data considering the similarities in the different classes. The simplicity of using the support vector classifier also allows for quick prototyping and great results.