# Introduction

This file is a comparative analysis on Cycon’s ability to perform SVM classification. This serves as proof that the Cycon page is able to perform SVM. The following shows SVM results for various datasets.

## Iris.csv

|  |  |
| --- | --- |
| **Dataset:** | |
| Shape: 150 x 5  Samples: 50 samples for 3 classes  Classes: Iris-setosa, Iris-versicolor, Iris-virginica  Purpose: Identify class of iris flowers given petal information. | |
| **Comparative Work:**  <https://www.kaggle.com/code/arshid/support-vector-machine-on-iris-flower-dataset> | **Cycon Work:** |
| **Settings:** | |
|  |  |
| **Results:** | |
|  |  |
| **Any Additional Information:** | |
| The comparative work uses the same model which we can reproduce. However, in the original work, a custom shuffling on the dataset is used to create the testing and training set that isn’t repeatable. Instead, we shuffle and can achieve similar results. Note also that the test\_size is the same, however the amount of the test is different. Unsure if this is an error with the original work. However, in the result provided, we obtain similar results. | |

## activity.csv

|  |  |
| --- | --- |
| **Dataset:** | |
| Shape: 10299 x 561  Purpose: Tell the particular activity based on information such as body position, heart beat etc. | |
| **Comparative Work:**  [**https://www.kaggle.com/code/pranathichunduru/svm-for-multiclass-classification/notebook**](https://www.kaggle.com/code/pranathichunduru/svm-for-multiclass-classification/notebook) | **Cycon Work:** |
| **Settings:** | |
|  |  |
| **Results:** | |
|  |  |
| **Any Additional Information:** | |
| Note that in the comparative work, the training and testing set was already split. Furthermore, that they shuffled the dataset without implementing a random\_state. As such, a perfect reproduction of the results is near impossible. However, we can obtain similar results by combining the training and testing set and validation split at the point where the training and testing would be the same, i.e. 0.2861 as the testing will be the last entries and splitting by 0.2861 would create nearly the same amount of training and testing sets. Also note that this csv is fairly large, as such the run time may take a few minutes to process. | |

## Diabetes.csv

|  |  |
| --- | --- |
| **Dataset:** | |
| A picture containing text, screenshot, number, font  Description automatically generated  Shape: 768 x 9  Samples: 384 for yes and 384 for no  Classes: 0 (no) and 1 (yes)  Purpose: Determine if the person has diabetes. 0 means the person does not have diabetes while 1 means they have diabetes. | |
| **Comparative Work:**  [**https://www.kaggle.com/code/berkayalan/classification-with-support-vector-machines/notebook**](https://www.kaggle.com/code/berkayalan/classification-with-support-vector-machines/notebook) | **Cycon Work:** |
| **Settings:** | |
|  |  |
| **Results:** | |
|  |  |
| **Any Additional Information:** | |
| Note that Cycon currently doesn’t have a set number to split the training and testing set, instead it uses a percentage. However, we can get the same amount through calculation (168/768 = 0.21875). Note that we did not get the exact same and that one 1 (true) label was different. This could be due to the model not having a random\_state input to obtain the exact same result. However, the results are still very similar. | |