**HOME LEARNING TASK - 3rd November 2020**

**Algorithm**

**Support vector machines (SVMs)** are a set of supervised learning methods used for [classification](https://scikit-learn.org/stable/modules/svm.html#svm-classification), [regression](https://scikit-learn.org/stable/modules/svm.html#svm-regression) and [outliers detection](https://scikit-learn.org/stable/modules/svm.html#svm-outlier-detection). It uses a technique called the ‘kernal-trick’ to transform the data and based on the transformations, it finds an optimal boundary between the 2 possible outputs.

**The advantages of support vector machines are:**

* Effective in high dimensional spaces.
* Still effective in cases where number of dimensions is greater than the number of samples.
* Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
* Versatile: different [Kernel functions](https://scikit-learn.org/stable/modules/svm.html#svm-kernels) can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

**The disadvantages of support vector machines include:**

* If the number of features is much greater than the number of samples, avoid over-fitting in choosing [Kernel functions](https://scikit-learn.org/stable/modules/svm.html#svm-kernels) and regularisation term is crucial.
* SVMs do not directly provide probability estimates.

**The most common use of the SVM algorithms is on biological and other sciences.** In the last few years has been extensively applied for protein remote homology detection. These algorithms have been widely used for identifying among biological sequences. For example, classification of genes, patients on the basis of their genes, and many other biological problems.

**Useful Note:**

The support vector machines in scikit-learn support both dense (numpy.ndarray and convertible to that by numpy.asarray) and sparse (any scipy.sparse) sample vectors as input. However, to use an SVM to make predictions for sparse data, it must have been fit on such data. For optimal performance, use C-ordered numpy.ndarray (dense) or scipy.sparse.csr\_matrix (sparse) with dtype=float64.