**Interpret Support Vectors**

In this section, we used Support Vectors approach to train our training data and after applying this training to our validation dataset we got an accuracy of **%90.**

In order to interpret the dataset, we calculate the following parameters.

1. Looking into Support vectors we got the following array and plot the distribution of vectors: (sv\_classifier.support\_vectors\_)

[[1.3860e+04 0.0000e+00 0.0000e+00 ... 1.0230e+03 1.0230e+03 1.0000e+02]

[4.0000e+03 0.0000e+00 0.0000e+00 ... 3.7200e+02 3.7200e+02 1.0000e+02]

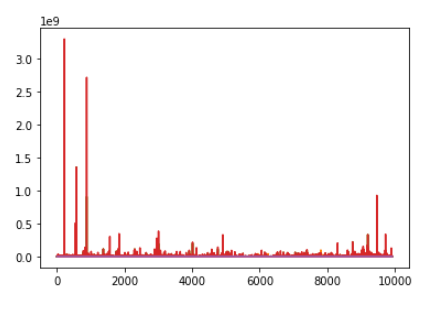
[4.6350e+03 0.0000e+00 0.0000e+00 ... 1.2852e+05 1.2852e+05 1.0000e+02]

...

[7.5690e+03 1.0000e+00 2.2978e+04 ... 1.3050e+05 5.6655e+05 1.0000e+02]

[1.4100e+03 1.0000e+00 3.0000e+03 ... 5.8296e+04 7.1848e+04 1.0000e+02]

[1.5450e+04 1.0000e+00 2.0000e+04 ... 1.5480e+06 4.0284e+06 1.0000e+02]]



1. Looking into fitting status (0 means correctly fitted and 1 otherwise) we got 0 means correctly fitted of data. (sv\_classifier.fit\_status\_)
2. Also, if we look at number of support vectors for each class we have (sv\_classifier.n\_support\_):  
   [8795 699 192 236]

Here is the code:

sv\_classifier.support\_

Supportvectors = sv\_classifier.support\_vectors\_

print(Supportvectors)

plot(Supportvectors)

print(sv\_classifier.fit\_status\_)

print(sv\_classifier.n\_support\_)