# Class BinarySVM

## Methods:

1. **def \_\_init\_\_(self, method="boser", C = 1, tolerance = 0.001, tau = 0.000001, maxIterations = 1000000, cacheSize = 8000000, doShrinking = True, kernel = 'linear', sigma = 0,k = 1, b = 0,dtype = float64):**

Constructor to set binary SVM training parameters

*parameters:*

**method:** 'boser', default: 'boser'

computation method

**C: deafult:** 1

Upper bound in conditions of the quadratic optimization problem.

**tolerance**: default: '0.001'

Training accuracy/ stopping criteria

**tau:** default: 0.000001

Tau parameter of the WSS scheme.

**maxiterations:** default: 1000000

Maximal number of iterations for the algorithm.

**cacheSize:** default: 8000000

cachesize for storing values of kernel matrix.

**doShringing:** True/false, default: True

flag to set shrinking optimization technique

**kernel:** 'linear'/'rbf', default: 'linear’

**k:** default: 1

coefficient value of k when kernel function is 'linear'

**b:** default: 0

coefficient value of b of linear function

**dtype:** intc, float32, float63, intc

1. **def training(self, trainData, trainDependentVariables):**

*parameters:*

train data feature values(type nT), train data target values(type nT)

*returns*:

training results object

1. **def predict(self, trainingResult, testData):**

*parameters:*

training result object, test data feature values(type nT)

*returns*:

predicted values of type nT

1. **def compress(self, arrayData):**

*parameters:*

serialized numpy array

*returns*:

Compressed numpy array

1. **def decompress(self, arrayData):**

*parameters:*

deserialized numpy array

*returns*:

decompressed numpy array

1. **def serialize(self, data, fileName=None, useCompression=False):**

*parameters:*

Method 1: data(type nT/model)

-Returns serialized numpy array

Method 2: data(type nT/model), fileName(.npy file to save serialized array to disk)

- Saves serialized numpy array as "fileName" argument

Method 3: data(type nT/model), useCompression = True

-Returns compressed numpy array

Method 4: data(type nT/model), fileName(.npy file to save serialized array to disk), useCompression = True

-Saves compressed numpy array as "fileName" argument

1. **def deserialize(self, serialObjectDict=None, fileName=None, useCompression=False):**

*parameters:*

serialized/ compressed numpy array or serialized/ compressed .npy file

*returns*:

deserialized/ decompressed numeric table/model

1. **def predictWithQualityMetrics(self, trainingResult, testData,testGroundTruth):**

*parameters*:

training result object, test data feature values of type nT, test data actual target values(type nT).

*returns*:

predicted values(type nT), quality metrics object for binary classifier.

1. **def printAllQualityMetrics(self, qualityMetricSetResult):**

*parameters*:

quality metrics object for binary classifier

Prints Accuracy, Precision, Recall, F1-score, Specificity, AUC

# Class MultiSVM:

1. **def \_\_init\_\_(self, nClasses, method="boser", C = 1, tolerance = 0.001, tau = 0.000001, maxIterations = 1000000, cacheSize = 8000000, doShrinking = True, kernel = 'linear', sigma = 0,k = 1, b = 0,dtype = float64):**

Constructor to set Multi class SVM training parameters

*parameters:*

**nClasses**: number of classes

**method:** 'boser', default: 'boser'

Computation method

**C:** default**:** 1

Upper bound in conditions of the quadratic optimization problem.

**tolerance**: default: '0.001'

Training accuracy/ stopping criteria

**tau:** default: 0.000001

Tau parameter of the WSS scheme.

**maxiterations:** default: 1000000

Maximal number of iterations for the algorithm.

**cacheSize:** default: 8000000

cachesize for storing values of kernel matrix.

**doShringing:** True/false, default: True

flag to set shrinking optimization technique

**kernel:** 'linear'/'rbf', default: 'linear’

**k:** default: 1

coefficient value of k when kernel function is 'linear'

**b:** default: 0

coefficient value of b of linear function

**dtype:** intc, float32, float63, intc