Machine Learning Assignment: SVC API

Shehzeen Shuaeb Khan

J028, BTech. Data Science, Sem 5

SVC API: sklearn.svm.SVC

class **sklearn.svm.SVC** (*, C=1.0, kernel='rbf', degree=3, gamma='scale', coef0=0.0, shrinking=True, probability=False, tol=0.001, cache_size=200, class_weight=None, verbose=False, max_iter=- 1, decision_function_shape='ovr', break_ties=False, random_state=None)

Parameters:

- C - Regularization parameter;
- Kernel - Specifies the kernel type to be used in the algorithm;
- Degree - Degree of the polynomial kernel function ('poly'). Ignored by all other kernels;
- Gamma - Kernel coefficient for 'rbf', 'poly' and 'sigmoid';
- coef0 - Independent term in kernel function. It is only significant in 'poly' and 'sigmoid';
- Shrinking - Whether to use the shrinking heuristic;
- Probability - Whether to enable probability estimates;
- class_weight - Whether we want to assign weights to our classes;
- max_iter - Limit on the number of iterations of the solver;
- decision_function_shape - One versus rest or one versus one method to solve in case of multi class classification;
- random_state - Numpy seed to be used while generating random numbers.

Attributes:

- class_weight_ - Multipliers of parameter C for each class. Computed based on the class_weight parameter;
- classes_ - Class labels;
- coef_ - Weights assigned to the features;
- dual_coef_ - Dual coefficients of the support vector in the decision function multiplied by their targets. For multiclass, coefficient for all 1-vs-1 classifiers;
- fit_status_ 0 if correctly fitted, 1 otherwise (will raise warning);
- intercept_ - Constants in decision function;
- support_ - Indices of support vectors;
- support_vectors_ - Support vectors;
- n_support_ - Number of support vectors for each class.

Methods:

- decision_function(X)- Evaluates the decision function for the samples in X.
- fit (X, y[, sample_weight])-Fit the SVM model according to the given training data.

- get_params([deep])-Get parameters for this estimator.
- predict(X)- Perform classification on samples in X.
- score(X, y[, sample_weight])-Return the mean accuracy on the given test data and labels.
- set_params(**params)-Set the parameters of this estimator.

How does sklearn handle SVMs?

- Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outliers' detection.

- The advantages of support vector machines are:

i. Effective in high dimensional spaces.
ii. Still effective in cases where number of dimensions is greater than the number of samples.
iii. Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
iv. Versatile: different Kernel functions 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:

i. If the number of features is much greater than the number of samples, avoid over-fitting in choosing Kernel functions and regularization term is crucial.
ii. SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation.

- 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.