Pattern Recognition

Introduction to ScikitLearn

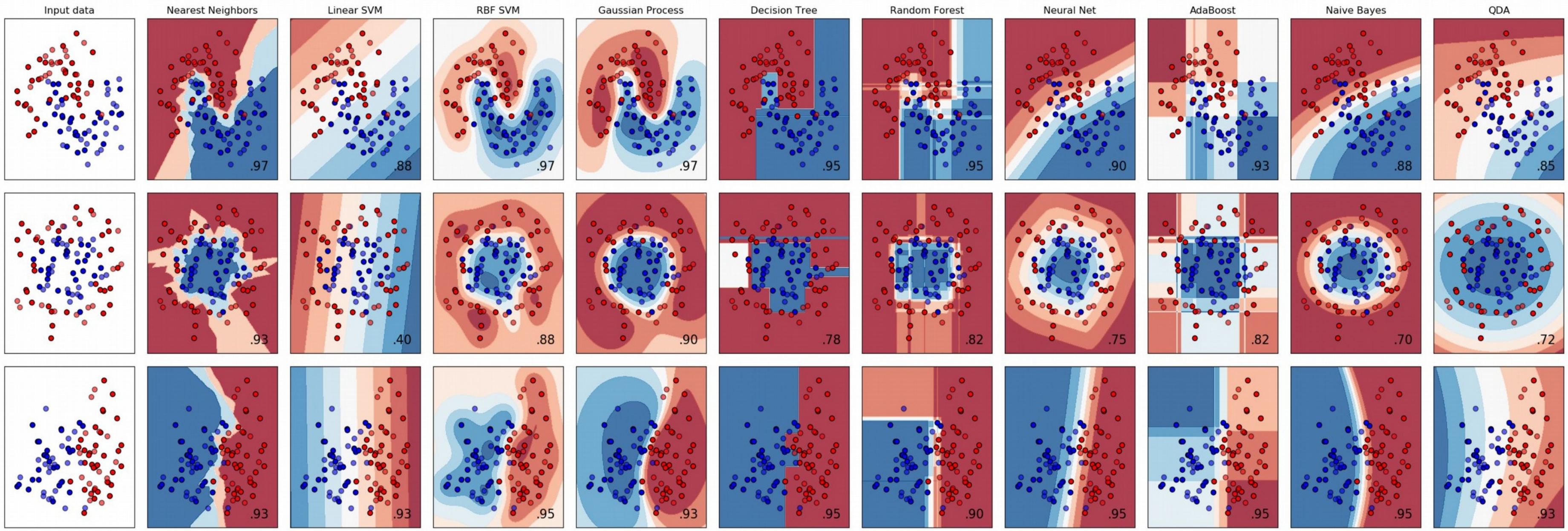
Francesco Tortorella

University of Cassino and Southern Latium

Cassino, Italy

scikit-learn

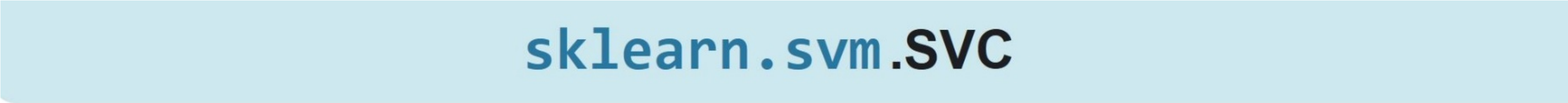
Machine Learning in Python



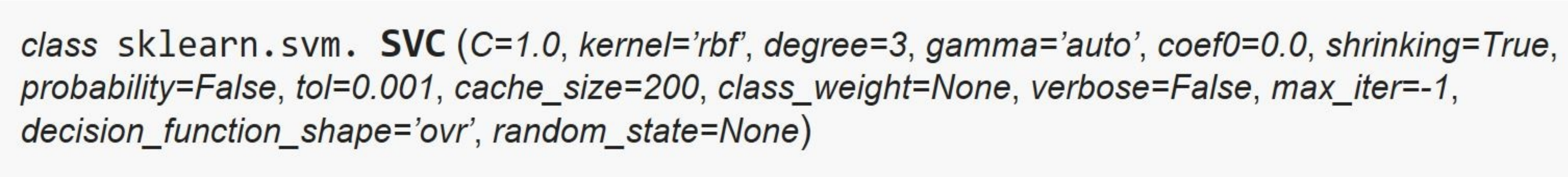
- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

SVM in Scikitlearn

- C-support vector classification
- Implementation based on Svmlib

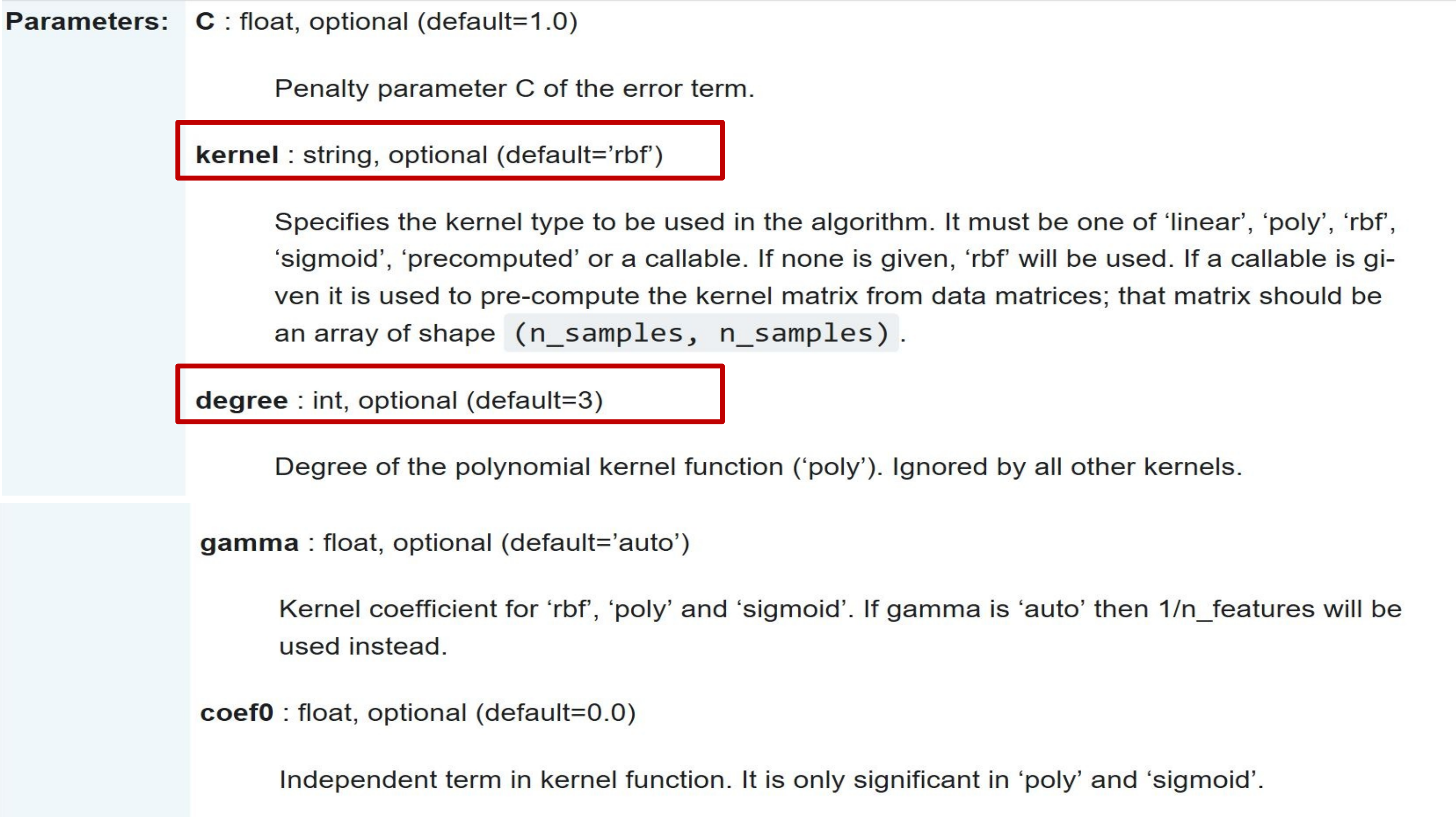


sklearn.svm.SVC



```
class sklearn.svm. SVC (C=1.0, kernel='rbf', degree=3, gamma='auto', 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', random_state=None)
```

SVM in Scikitlearn



Parameters:

C : float, optional (default=1.0)

Penalty parameter C of the error term.

kernel : string, optional (default='rbf')

Specifies the kernel type to be used in the algorithm. It must be one of 'linear', 'poly', 'rbf', 'sigmoid', 'precomputed' or a callable. If none is given, 'rbf' will be used. If a callable is given it is used to pre-compute the kernel matrix from data matrices; that matrix should be an array of shape `(n_samples, n_samples)`.

degree : int, optional (default=3)

Degree of the polynomial kernel function ('poly'). Ignored by all other kernels.

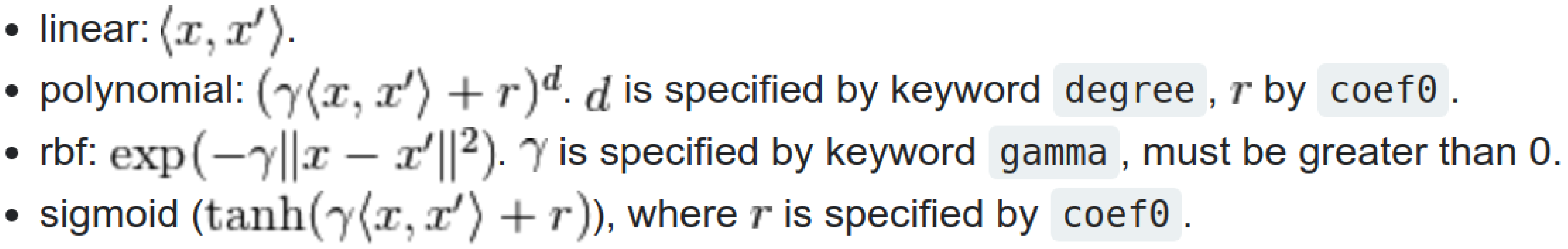
gamma : float, optional (default='auto')

Kernel coefficient for 'rbf', 'poly' and 'sigmoid'. If gamma is 'auto' then 1/n_features will be used instead.

coef0 : float, optional (default=0.0)

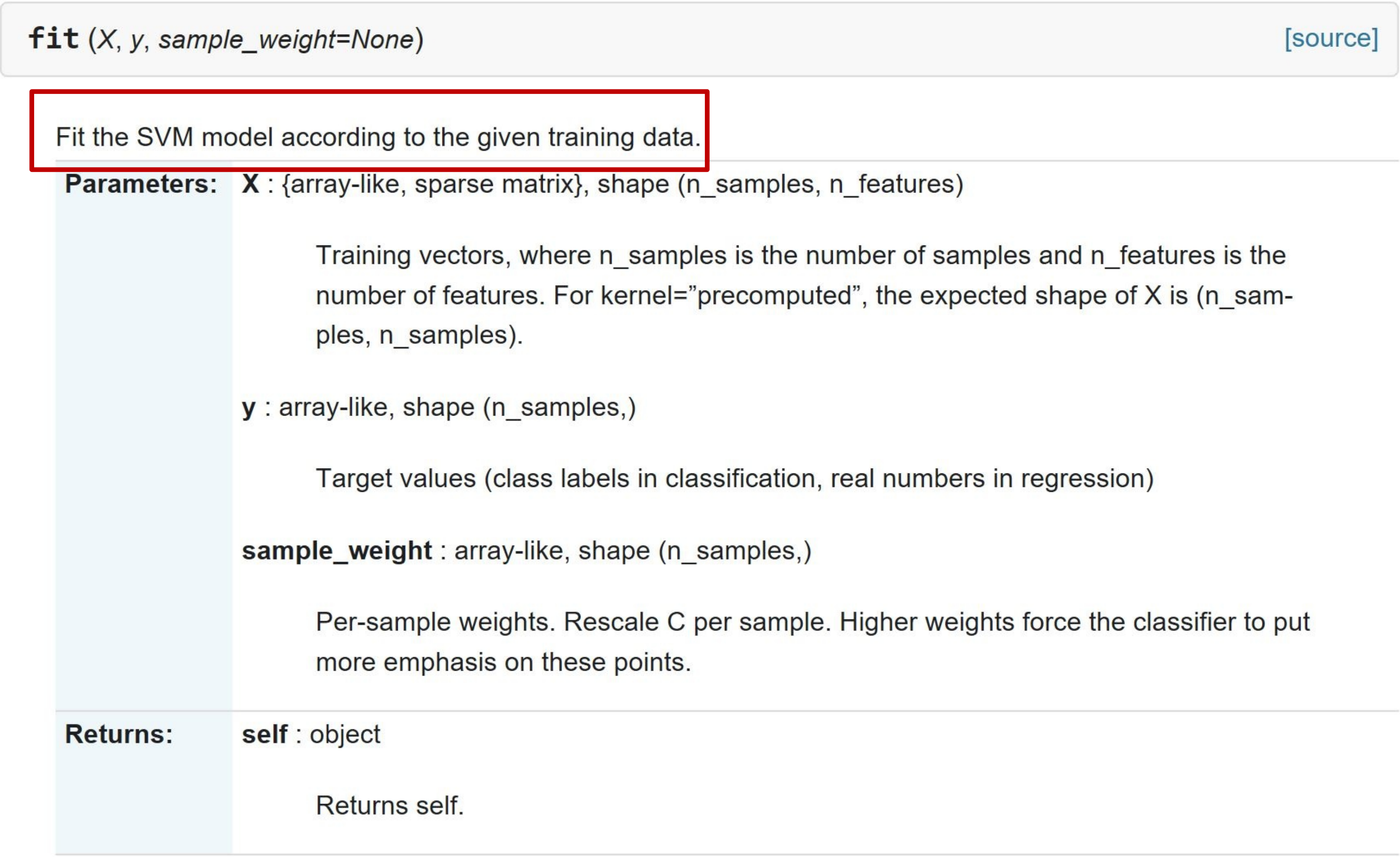
Independent term in kernel function. It is only significant in 'poly' and 'sigmoid'.

SVM in Scikitlearn



- linear: $\langle x, x'\rangle$.
- polynomial: $(\gamma\langle x, x'\rangle + r)^d$. d is specified by keyword `degree`, r by `coef0`.
- rbf: $\exp(-\gamma\|x - x'\|^2)$. γ is specified by keyword `gamma`, must be greater than 0.
- sigmoid $(\tanh(\gamma\langle x, x'\rangle + r))$, where r is specified by `coef0`.

SVM in Scikitlearn

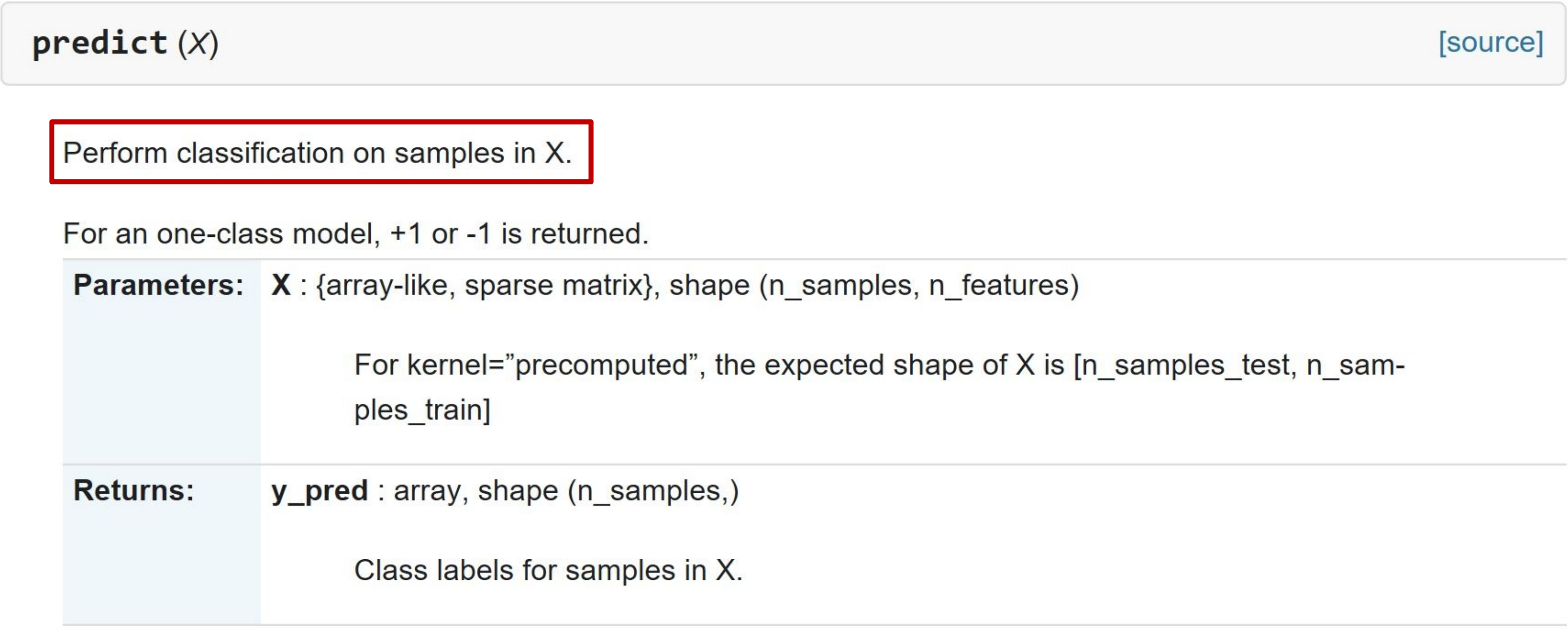


fit (*X*, *y*, *sample_weight=None*) [source]

Fit the SVM model according to the given training data.

Parameters:	**X** : {array-like, sparse matrix}, shape (n_samples, n_features) Training vectors, where n_samples is the number of samples and n_features is the number of features. For kernel="precomputed", the expected shape of X is (n_samples, n_samples). **y** : array-like, shape (n_samples,) Target values (class labels in classification, real numbers in regression) **sample_weight** : array-like, shape (n_samples,) Per-sample weights. Rescale C per sample. Higher weights force the classifier to put more emphasis on these points.
Returns:	**self** : object Returns self.

SVM in Scikitlearn



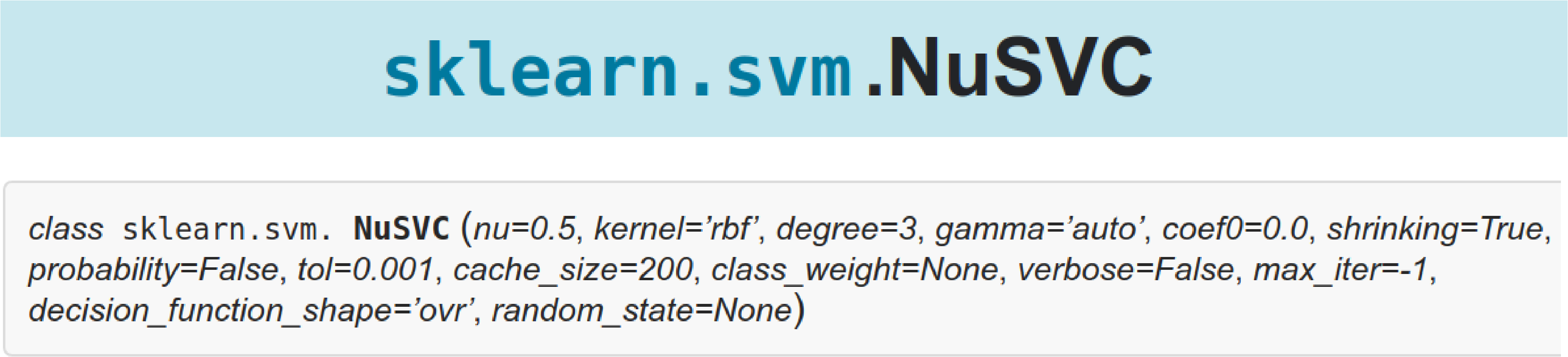
predict (*X*) [source]

Perform classification on samples in X.

For an one-class model, +1 or -1 is returned.

Parameters:	**X** : {array-like, sparse matrix}, shape (n_samples, n_features) For kernel="precomputed", the expected shape of X is [n_samples_test, n_samples_train]
Returns:	**y_pred** : array, shape (n_samples,) Class labels for samples in X.

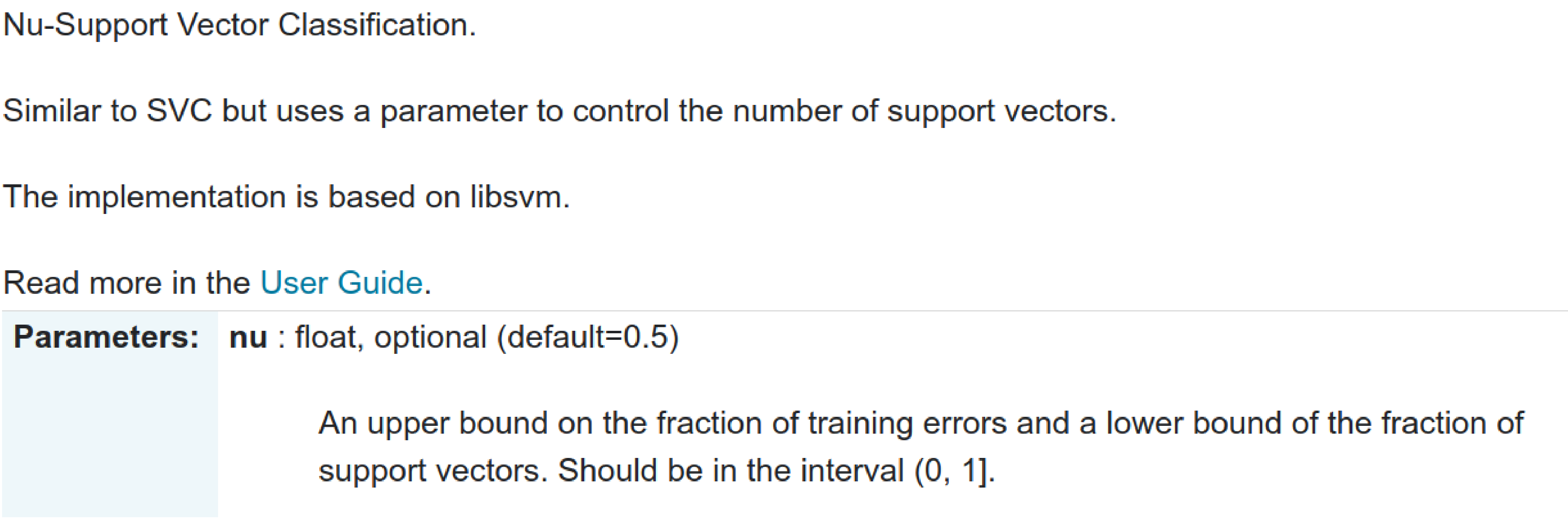
SVM in Scikitlearn



sklearn.svm.NuSVC

```
class sklearn.svm. NuSVC (nu=0.5, kernel=’rbf’, degree=3, gamma=’auto’, 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’, random_state=None)
```

SVM in Scikitlearn



Nu-Support Vector Classification.

Similar to SVC but uses a parameter to control the number of support vectors.

The implementation is based on libsvm.

Read more in the User Guide.

Parameters: **nu** : float, optional (default=0.5)

An upper bound on the fraction of training errors and a lower bound of the fraction of support vectors. Should be in the interval (0, 1].