Assignment 3: Support Vector Machine and cross-validation.

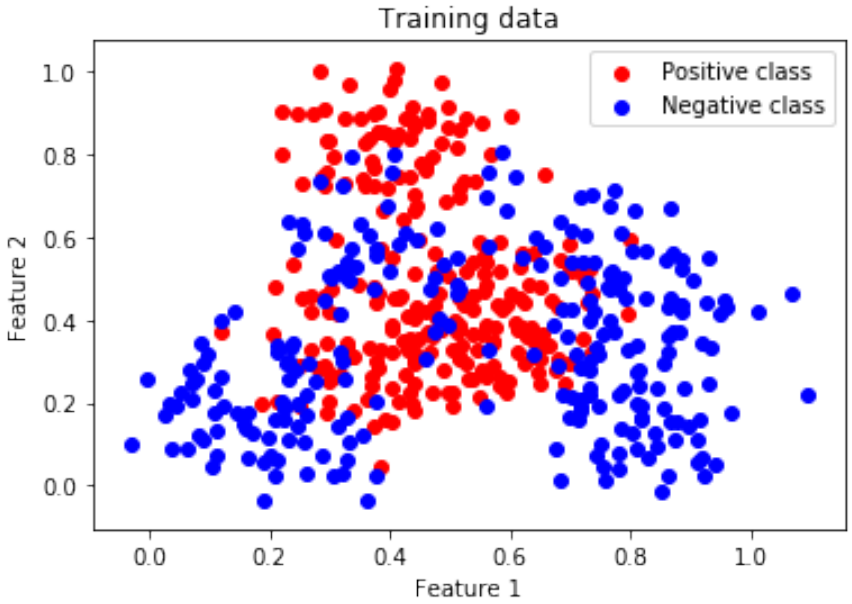
October 23, 2017

Support Vector Machine classifiers (1.5 points)

In this assignment, we use the scikit-learn package to train an SVM classifier. You will use cross-validation to determine the best combination of values for the two hyperparameters used in SVM: the cost, C, and precision, gamma.

- Question 1 (.9) Choosing a Gaussian kernel for the SVM, use 10-fold cross-validation to evaluate the cost and precision parameters (respectively named *C* and *gamma* in the documentation of the *sklearn.svm.SVC* class). The range of values for each parameters are given to you. Visualize the cross-validation error (as a function of the 2 SVM parameters) using *imshow*.
- Question 2 (.5) Plot the decision boundaries, by appropriately modifying the code from the previous exercises. Display the support vectors on the same figure.
- Question 3 (.1) Using the svm model you trained with the best combination of cost and precision, evaluate the classifier's generalization error on the test-set by showing the number of misclassified points.

Load and display the training data



Training with K-fold cross-validation

The function KFold from the scikit-learn library is initialising a "cross-validation" object with as argument the number of fold F (here $F = 10$). You can then use the object created to obtain partition your input data randomly into train sets and validation sets. For example, if you initialise your partitioner as `kf= KFold(n_splits=F)`, then calling `kf.split(input-data)` on your input data will generate a list of F partitions. Each partition has two elements, the first one gives you the indices of the training data, the second the indices of the validation data (the indices refer back to your original input data).

To compute SVM, you need to initialise a SVM object with the `SVC` function with inputs: the hyperparameters, C and gamma, and the type of kernel you want to use ('rbf' for gaussian). For instance: `svm=SVC(C=C,gamma=gamma,kernel='rbf')`. To train the SVM model on the training set, call the function `svm.fit()` with inputs: the features and labels of the training set. Once trained, use the SVM model to classify the validation data based on their features by using the `svm.predict()` function. The error coresponds to the number of mislabeled data between the prediction and the ground truth given by the labels of the validation data,the lower the error the better the model.

Display cross-validation results and decision function