Université Jean Monnet

Course: Machine Learning

Practical Session 3

Support Vector Machines

Jorge Chong, Oussama Bouldjedri

March 26, 2017

1 SVM with Scikit-Learn

The next graphs, shows some settings made by hand, comparing the classifiers using different sets of parameters. The source code for this part is in 2.py

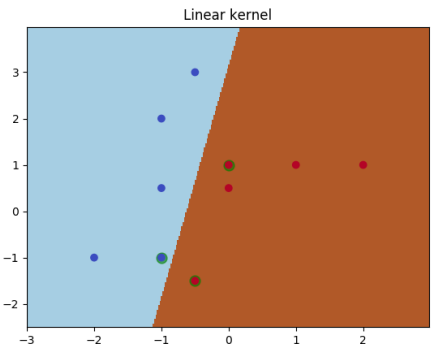

Figure 1: Linear, C = 4, surrounded in green the support vectors

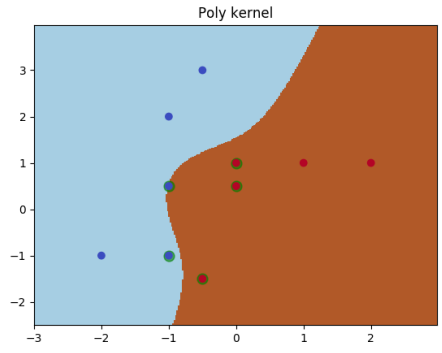

Figure 2: Poly, C = 4, Gamma = 1, degree = 3, surrounded in green the support vectors

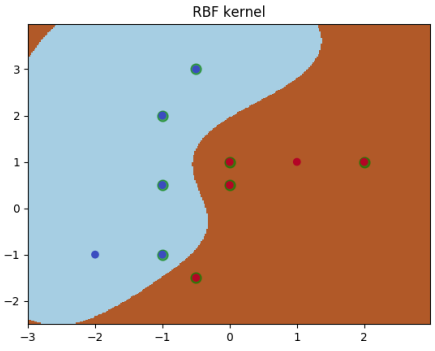

Figure 3: RBF, C = 4, surrounded in green the support vectors

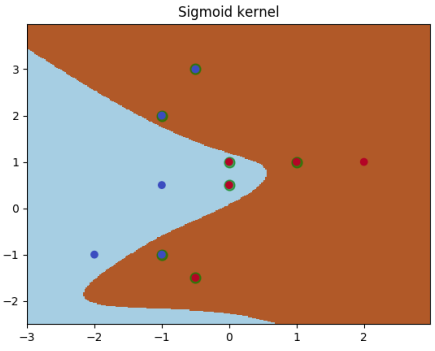

Figure 4: Sigmoid, C = 4, Gamma = 1, surrounded in green the support vectors

2 Data Generation. Random datasets

We generated random data and did a grid search function included in the scikit learn package that do the cross-validation procedure. The results are:

1. For a Linear Kernel. The grid search reduces to search the best C in [1..5]. Results: C = 4, score = 0.88
2. For a RBF Kernel. The grid search for C in [1..5], and Gamma in [0.001, 0.01, 0.1, 1]. Results: Gamma = 1, C = 4, score = 0.88
3. For a Polynomial Kernel. The grid search for C in [1..5], degree in [0.001, 0.01, 0.1, 1] and Gamma in [1, 2, 3, 4, 5, 6, 7]. Results: Degree = 2, Gamma = 1, C = 4, score = 0.90
4. The source code for this part is in 3_1.py

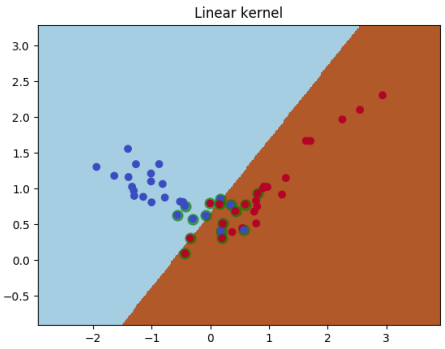

Figure 5: Linear, C = 4, Score = 0.88, surrounded in green the support vectors

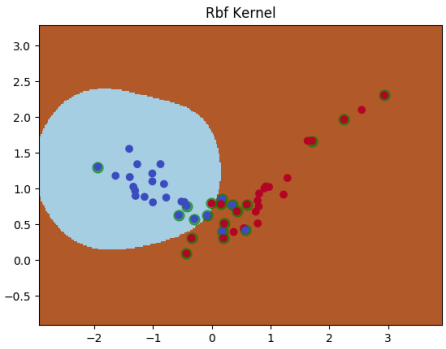

Figure 6: RBF, C = 2, Gamma = 1, Score = 0.88, surrounded in green the support vectors

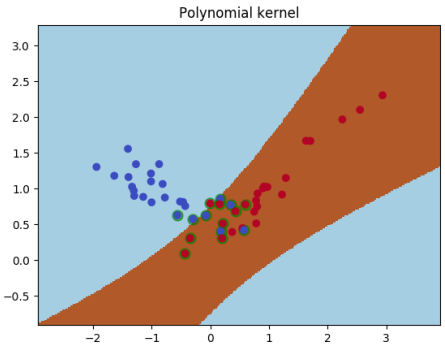

Figure 7: Polynomial, C = 4, degree = 2, Gamma = 1, Score = 0.90, surrounded in green the support vectors

3 Existing Datasets

We used 2-moon dataset and did a grid search function included in the scikit learn package that do the cross-validation procedure. The results are:

1. For a Linear Kernel. The grid search reduces to search the best C in [1..7]. Results: C = 1, score = 0.85

2. For a RBF Kernel. The grid search for C in [1..5], and Gamma in [0.001, 0.01, 0.1, 1]. Results: Gamma = 1, C = 5, score = 1.00

3. For a Polynomial Kernel. The grid search for C in [1..7], Gamma in [0.001, 0.01, 0.1, 1] and degree in [1, 2, 3, 4, 5, 6, 7]. Results: Degree = 5, Gamma = 1, C = 2, score = 0.88

4. The source code for this part is in 3_2.py

For the iris multiclass dataset we got:

1. For a Linear Kernel. The grid search reduces to search the best C in [1..7]. Results: C = 1, score = 0.97

2. For a RBF Kernel. The grid search for C in [1..5], and Gamma in [0.001, 0.01, 0.1, 1]. Results: Gamma = 0.1, C = 4, score = 0.97

3. For a Polynomial Kernel. The grid search for C in [1..7], Gamma in [0.001, 0.01, 0.1, 1] and degree in [1, 2, 3, 4, 5, 6, 7]. Results: Degree = 7, Gamma = 1, C = 1, score = 0.97

4. The source code for this part is in 3_2_eris.py

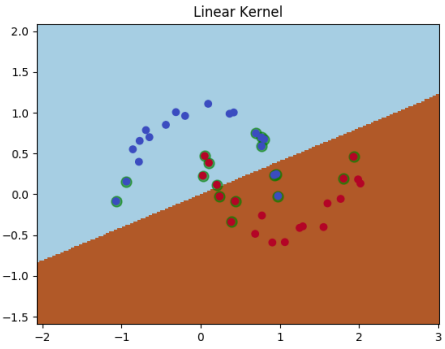

Figure 8: 2-Moon dataset. Linear, C = 1, Score = 0.85, surrounded in green the support vectors

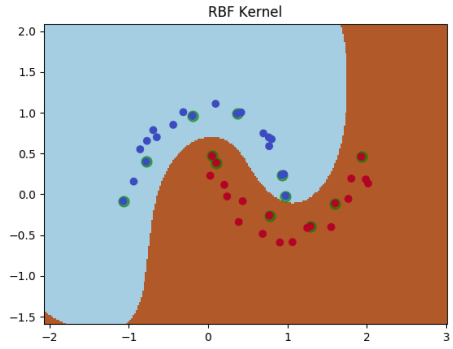

Figure 9: 2-Moon dataset. RBF, C = 5, Gamma = 1, Score = 1.00, surrounded in green the support vectors

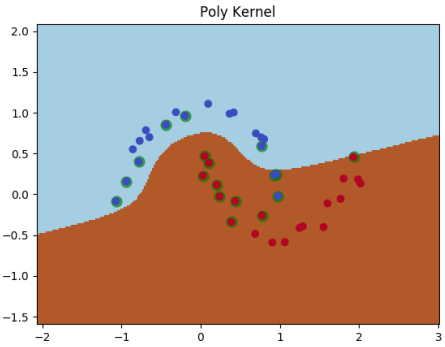

Figure 10: 2-Moon dataset.Polynomial, C = 2, degree = 5, Gamma = 1, Score = 0.88, surrounded in green the support vectors

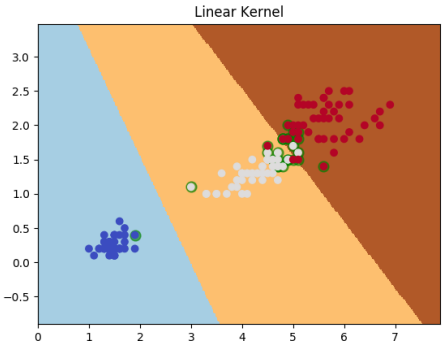

Figure 11: Iris dataset. Linear, C = 1, Score = 0.97, surrounded in green the support vectors

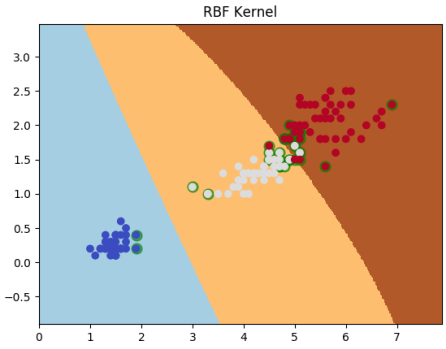

Figure 12: Iris dataset. RBF, C = 4, Gamma = 0.1, Score = 0.97, surrounded in green the support vectors

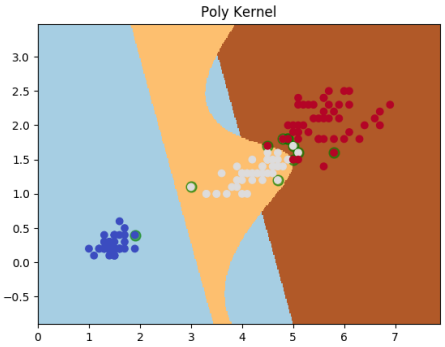

Figure 13: Iris dataset. Polynomial, C = 1, degree = 7, Gamma = 1, Score = 0.97, surrounded in green the support vectors

4 Real Datasets

Analysing the Dataset for Ozone levels, we found that the target class is unbalanced. There are 863 out of 1041 rows which have an Ozone level less than 150 versus 178 rows out of 1041 which have a level of ozon over 150, therefore this could create problems in interpreting the correct way to scoring our model selection an evaluation. To avoid the problems of using accuracy, we used the roc_auc scoring function inside all the procedures that do cross validation and grid search, and also for assessing the final performance of the model.

We compared models for normalized versus non-normalized data. And tried to find the best classifier for Linear, RBB, Polynomial kernel. We did that using a grid search package provided by Scikit Learn.

We got the following results:

1. For Normalized data, we found that the best Classifier uses a RBF kernel with Gamma = 0.01 and C = 3.
   The best parameters are 'C': 1 with a score of 0.89
   The best parameters are 'gamma': 0.01, 'C': 3 with a score of 0.90
   The best parameters are 'gamma': 0.01, 'degree': 1, 'C': 4 with a score of 0.90
   For Normalize data:
   Scores for model Linear: 0.8900975695424159
   Scores for model RBF: 0.9007702504259708
   Scores for model Poly: 0.8976778897295296
   Selecting the best model:
   Best classifier is RBF
   The best parameters are 'C': 1 with a score of 0.90

2. For Non normalized data, we tested only linear and RBF kernels

The source code for this part is in 4.py

5 Bonus Support Vector Reggression

We tried the package proposed for Support Vector Regression, with the target attribute is the value of the Ozone Level. We did the same as before to try to find the best kernel and combination of parameters. The search takes hours when we use a Polynomial kernel due to the interval of the search [1..4], but it is still takes long time to compare every combination of parameters. We found the following results

1. The best kernel for the support vector regression is still linear

   The best parameters are 'C': 4 with a score of 0.43
   The best parameters are 'gamma': 0.01, 'C': 4 with a score of 0.35
   The best parameters are 'degree': 1, 'gamma': 1, 'C': 4 with a score of 0.43
   the best is linear

The source code for this part is in bonus.py