SPLEX TME 6

Support Vector Machines
Hyperparameter tuning and K-fold cross validation

The goal of the TME is to learn how to use linear and non-linear Support Vector Machines (SVM), how to tune hyperparameters, and to understand the notions of the margins and support vectors.

Data (three simulated data sets + one real)

- Simulated data
- Diabetic Retinopathy Debrecen Data Set

  `http://archive.ics.uci.edu/ml/datasets/Diabetic+Retinopathy+Debrecen+Data+Set`

Libraries

You will need to load the following packages:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification
from sklearn.datasets import make_blobs
from sklearn.datasets import make_moons
from sklearn import svm
```

Analysis

Here is the description and examples of running SVM in Python:
`http://scikit-learn.org/stable/modules/svm.html`

- To define a model and its parameters

  ```
  model = svm.SVC(... parameters...)
  ```

  and to fit a model

  ```
  model.fit(X, Y)
  ```

- We are in particularly interested in the parameters C (the penalty term) and `kernel`.
- To find and to fix optimal parameters, we will run the 10-fold cross validation.
- We will test “linear”, “poly”, and “rbf” kernels.

$$\text{Linear kernel: } \langle X, X' \rangle \quad (1)$$

$$\text{Polynomial kernel: } (\gamma\langle X, X' \rangle + r)^d \quad (2)$$

$$\text{RBF kernel: } \exp(-\gamma\|X - X'\|^2) \quad (3)$$

- Let vary the hyperparameter C $\in [0 : 0.1 : 10]$
- For all other parameters, we will use their default values.

1. Generate three artificial data sets (see previous TME) with the number of samples = 10 000.

2. We will fix the optimal parameters on a validation set.

3. First, we will split the simulated data into 2 sets, one for training + test, and the validation set.

4. Implement a k-fold cross validation on the validation set. You will need to split the validation set into training and testing data. Test all the chosen kernels with all possible values of C, and fix the optimal C and kernel based on the test accuracy of the validation set.

5. Once the value of C and a kernel are fixed (the ones which lead to the best performance), run the 10-fold cross validation on the data which was not used for validation.

6. Run the analysis on three simulated data sets.

7. Boxplot the test accuracy of the optimal parameters settings which can be different for each data set.

8. For each data set, for each kernel you have an optimal C. Given C, visualize the class boundaries for three tested kernels. Here is an example (similarly to what we already did):

   `http://scikit-learn.org/stable/auto_examples/exercises/plot_iris_exercise.html#sphx-glr-auto-examples-exercises-plot-iris-exercise-py`

9. In SVM, the margin and the support vectors play a very important role. In a 2-dimensional case, you can visualize them. Here is an example how to plot the support vectors,

   `https://jakevdp.github.io/PythonDataScienceHandbook/05.07-support-vector-machines.html`

   What is the role of the support vectors?

10. Run the same analysis on the Diabetic Retinopathy data (do not visualize anything). Tune the hyperparameters, find an optimal kernel and C, and boxplot the error rate.

11. What are the optimal parameters (kernel and C) for the simulated data? And for the Diabetic Retinopathy data? Do you have an intuition, why some kernels are more suitable than others?

References:

1. *A biologist's introduction into support vector machines.*

   `https://noble.gs.washington.edu/papers/noble2006biologists.pdf`

2. *Support Vector Machines and Kernels for Computational Biology*

   `http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1000173`