Practice session Model Selection

Goals
- Implement a full SVM model tuning using scikit-learn utilities
- Application on real data

Preliminaries Some cross validation steps may may be time consuming. To avoid the issue may use only a subset of the data.

1 Splice classification

Splice junctions are points on a DNA sequence at which 'superfluous' DNA is removed during the process of protein creation in higher organisms. The problem we aim to solve is to recognize, given a sequence of DNA, the boundaries between exons (the parts of the DNA sequence retained after splicing) and introns (the parts of the DNA sequence that are spliced out). For more details see UCI [1] and DELVE site [2].

1. Load the dataset `splice`.

```
from sklearn.datasets import load_svmlight_file
import numpy as np

X, Y = load_svmlight_file("./splice")
```

How many samples and features does the dataset include ? Is the classification problem balanced ?

Hereafter you may use X and Y to learn your best classification function. Once completed, ask for the release of the test set.

2. We aim to estimate a non-linear SVM model using a RBF (gaussian) kernel $k(\mathbf{x}, \mathbf{z}) = \exp^{-\gamma\|\mathbf{x}-\mathbf{z}\|}$. This implies to tune the hyper-parameters C of the SVM and γ. For this, we will rely on the utility functions provided by Scikit-Learn, especially the GridSearchCV function.

3. Build two large logarithmic grids of size 25 for the kernel parameter γ in the range 10^{-3} to 1 and for C in the interval $10^{-1/2}$ to 10^2

```
gamma_grid = ....
C_grid = ...
```

4. To select the best model we will perform a grid search over the previously defined grids using GridSearchCV function of Scikit-Learn. First let set up the classifier and the grid search module. The best model will be selected based on the accuracy performance and K-fold validation (feel free to adapt the value of K).

```
from sklearn.model_selection import GridSearchCV
from sklearn.svm import SVC
# the grid
parameters = [{"gamma": gamma_grid, "C": C_grid}]
```

1. https://archive.ics.uci.edu/ml/datasets/Molecular+Biology+(Splice-junction+Gene+Sequences)
2. http://www.cs.toronto.edu/~delve/data/splice/spliceDetail.html

```
# the classifier
clf_rbf = SVC(kernel="rbf", tol=0.01)
# Perf a K-fold validation using the accuracy as the performance measure
K = 3
# we will dot it on a grid search using n_jobs processors
clf_rbf = GridSearchCV(clf_rbf, param_grid=parameters, cv=K, scoring="
    accuracy", verbose=1, n_jobs = 2)
```

After that the best model is learned

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

5. Let check the best selected hyper-parameters, the corresponding accuracy and the validation performance graphic along the grid. Is the figure in accordance with the chosen hyper-parameters ?

```
# Get the best parameters
print("\n Rbf kernel - optimal hyper-parameters = {}".format(clf_rbf.
    best_params_))
print("\n Rbf kernel - best cross-val accuracy = {}".format(clf_rbf.
    best_score_))

## Plot the validation performance w.r.t. C and gamma
import matplotlib.pyplot as plt

plt.imshow(clf_rbf.cv_results_["mean_test_score"].reshape(gamma_grid.
    shape[0],C_grid.shape[0]), extent=[min(gamma_grid),max(gamma_grid),
    min(C_grid),max(C_grid)],interpolation="bicubic",aspect="auto")
plt.colorbar()
plt.show()
```

6. At this stage, ask for the test set. Load it as in question 1. Compute the prediction accuracy on the test set. Compute and plot the confusion matrix. Comment on the obtained results Hint : refer to the documentation of confusion matrix for more details.

```
from sklearn.metrics import confusion_matrix, accuracy_score

Ytest_pred = ...
confmat = confusion_matrix(..., ...)
fig, ax = plt.subplots(figsize=(9, 9))
ax.matshow(confmat, cmap=plt.cm.Blues, alpha=0.5)
for i in range(confmat.shape[0]):
  for j in range(confmat.shape[1]):
    ax.text(x=j, y=i, s=confmat[i, j], va="center", ha="center")

plt.xlabel("Prediction"); plt.ylabel("Truth");
```

7. Inspiring from the previous questions, learn a linear SVM. It's hyper-parameter must be selected using the grid search tool. Compare the obtained results to the non-linear SVM case.