DATA.ML.200 Pattern Recognition and Machine Learning

Exercise Set 6: November 30–December 4, 2020

- Exercises consist of both `pen&paper` and `python` assignments.
- Prepare a single PDF and return to Moodle on Friday, December 4th at 23:55 at the latest.
- Mark on 1st page which exercises you did.

1. `pen&paper` *Error rate confidence limits.*

   We train a classifier with a set of training examples, and test the accuracy of the resulting model with a set of $N = 100$ test samples. The classifier misclassifies $K = 5$ of those.

   a) Find the 90% confidence interval of the result. Hint: The classification accuracy can be modeled using binomial distribution, whose confidence intervals are discussed here:

      `https://en.wikipedia.org/wiki/Binomial_distribution#Confidence_intervals`

   b) Another classifier misclassifies only 3 test samples. Is it better than the first one with statistical significance at 90% confidence level?

2. `pen&paper` *Design a regularized LDA classifier.*

   Let's revisit the LDA design of Exercise set 4, but add a regularization term. The non-regularized LDA solution is given by as

   $$\mathbf{w} = (\mathbf{\Sigma}_0 + \mathbf{\Sigma}_1)^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$$

   The regularized solution with regularization parameter $\lambda > 0$ is defined as

   $$\mathbf{w} = (\mathbf{\Sigma}_0 + \mathbf{\Sigma}_1 + \lambda\mathbf{I})^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$$

   However, as the scale of $\mathbf{w}$ is not important—only the direction—let us use an alternative definition instead:

   $$\mathbf{w} = \lambda (\mathbf{\Sigma}_0 + \mathbf{\Sigma}_1 + \lambda\mathbf{I})^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0).$$

   This definition avoids the convergence of $\mathbf{w}$ towards zero as $\lambda \to \infty$.

a) Compute the regularized LDA weight vector[1] for $\lambda = 100$ and

$$\boldsymbol{\mu}_0 = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \qquad \boldsymbol{\mu}_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$
$$\boldsymbol{\Sigma}_0 = \begin{pmatrix} 4 & 1 \\ 1 & 2 \end{pmatrix} \qquad \boldsymbol{\Sigma}_1 = \begin{pmatrix} 3 & -1 \\ -1 & 2 \end{pmatrix}$$

b) Where does $\mathbf{w}$ converge as $\lambda \to \infty$?

3. **python** Download a high-dimensional ovarian cancer dataset in `*.mat` format from: `http://www.cs.tut.fi/courses/SGN-41007/exercises/arcene.zip`

   Use `scipy.io.loadmat` to open the file. Note that your have to ravel `y_train` and `y_test` so that `sklearn` will accept them.

   a) Train a random forest classifier with 100 trees.
   b) Plot a histogram of its feature importances.
   c) Compute the accuracy on `X_test` and `y_test`.

4. **python** Apply the recursive feature elimination approach (`sklearn.feature_selection.RFECV`) with Linear discriminant analysis classifier for the arcene dataset.

   a) Instantiate an RFECV selector (call it `rfe` from now on). To speed up computation, set `step = 50` in the constructor. Also set `verbose = 1` to see the progress.
   b) Fit the RFECV to `X_train` and `y_train`.
   c) Count the number of selected features from `rfe.support_`.
   d) Plot the errors for different number of features:
      `plt.plot(range(0,10001,50), rfe.grid_scores_)`
   e) Compute the accuracy on `X_test` and `y_test`. You can use `rfe` as any other classifier.

5. **python** Apply L_1 penalized Logistic Regression for feature selection with the arcene dataset. We wish to find a good value for parameter C by 10-fold cross-validating the accuracy and study the sparseness of the solution: how many features were selected.

[1] Remember the inversion rule for 2×2 matrices:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \frac{1}{ad - bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

Follow these steps:

a) Instantiate a LogisticRegression classifier. Set `penalty = 'l1'` in the constructor.

b) Estimate the accuracy of the classifier with $C = 10^{-4}, 10^{-3}, \ldots, 10^{2}$ using 5-fold CV, and find the best C.

c) Fit the LogisticRegression to `X_train` and `y_train` with the best C.

d) Count the number of selected features from `clf.coef_`, where clf is your logistic regression classifier.

e) Compute the accuracy on `X_test` and `y_test`.