

MACHINE LEARNING FROM DATA

Lab Session 6 – Support Vector Machines

1.	Goal.....	2
2.	Instructions	2
3.	Introduction and previous study.....	2
4.	SPAM dataset.....	2
4.1.	Characteristics of the dataset	2
4.2.	Classification using SVM	4
4.3.	Analysis of the classifier performance.....	5

1. Goal

The goal of this session is to

- Apply linear and non-linear Support Vector Machine classifiers
- Analyze the classifier performance with different metrics
- Use grid search for optimizing SVM hyperparameters

2. Instructions

Getting the material:

- Download and uncompress the file Mlearn_Lab6.zip

Handling your work:

- Answer the questions in the document Mlearn_Lab6_SVM_report_surname.doc

Write the new code in the same notebook **Mlearn_lab6_SVM.ipynb**

- Zip and upload to Atenea the pdf report and the notebook in a single file

3. Introduction and previous study

Read the slides corresponding to lecture 4.2: support vector machines.

Open Colab Notebook **Mlearn_lab6_SVM.ipynb**. Read the notebook identifying the different sections in the code.

In the first part, you will play with a toy example using synthetic datasets with 2 classes and 2 features per class to analyze and compare decision boundaries, margins and support vectors obtained with linear and non-linear versions of SVM, using different values of the regularization parameter.

In the second part, you will solve a binary classification problem using the SPAM dataset.

4. SPAM dataset

4.1. Characteristics of the dataset

The SPAM dataset can be found in this repository: <https://archive.ics.uci.edu/ml/datasets/Spambase>.

This dataset contains 4601 vectors obtained from 4601 emails. Each vector of dimension 57 is formed by counting the frequency of a particular word in the email. The last features correspond to run-length attributes that measure the length of sequences of consecutive capital letters. This is the information provided by the dataset authors:

1. Title: SPAM E-mail Database
 2. Sources:
 - (a) Creators: Mark Hopkins, Erik Reeber, George Forman, Jaap Suermondt
Hewlett-Packard Labs, 1501 Page Mill Rd., Palo Alto, CA 94304
 - (b) Donor: George Forman (gforman at nospam hpl.hp.com) 650-857-7835
 - (c) Generated: June-July 1999
 3. Past Usage:
 - (a) Hewlett-Packard Internal-only Technical Report. External forthcoming.

- (b) Determine whether a given email is spam or not.
- (c) ~7% misclassification error.
False positives (marking good mail as spam) are very undesirable.
If we insist on zero false positives in the training/testing set, 20-25% of the spam passed through the filter.

4. Relevant Information:

The "spam" concept is diverse: advertisements for products/web sites, make money fast schemes, chain letters, pornography... Our collection of spam e-mails came from our postmaster and individuals who had filed spam. Our collection of non-spam e-mails came from filed work and personal e-mails, and hence the word 'george' and the area code '650' are indicators of non-spam. These are useful when constructing a personalized spam filter. One would either have to blind such non-spam indicators or get a very wide collection of non-spam to generate a general purpose spam filter.

For background on spam:

Cranor, Lorrie F., LaMacchia, Brian A. Spam!
Communications of the ACM, 41(8):74-83, 1998.

5. Number of Instances: 4601 (1813 Spam = 39.4%)

6. Number of Attributes: 58 (57 continuous, 1 nominal class label)

7. Attribute Information: The last column of 'spambase.data' denotes whether the e-mail was considered spam (1) or not (0), i.e. unsolicited commercial e-mail. Most of the attributes indicate whether a particular word or character was frequently occurring in the e-mail. The run-length attributes (55-57) measure the length of sequences of consecutive capital letters. For the statistical measures of each attribute, see the end of this file. Here are the definitions of the attributes:

48 continuous real [0,100] attributes of type word_freq_WORD = percentage of words in the e-mail that match WORD, i.e. $100 * (\text{number of times the WORD appears in the e-mail}) / \text{total number of words in e-mail}$. A "word" in this case is any string of alphanumeric characters bounded by non-alphanumeric characters or end-of-string.

6 continuous real [0,100] attributes of type char_freq_CHAR = percentage of characters in the e-mail that match CHAR, i.e. $100 * (\text{number of CHAR occurrences}) / \text{total characters in e-mail}$

1 continuous real [1,...] attribute of type capital_run_length_average == average length of uninterrupted sequences of capital letters

1 continuous integer [1,...] attribute of type capital_run_length_longest = length of longest uninterrupted sequence of capital letters

1 continuous integer [1,...] attribute of type capital_run_length_total = sum of length of uninterrupted sequences of capital letters = total number of capital letters in the e-mail

1 nominal {0,1} class attribute of type spam = denotes whether the e-mail was considered spam (1) or not (0), i.e. unsolicited commercial e-mail.

8. Missing Attribute Values: None

9. Class Distribution:

Spam	1813	(39.4%)
Non-Spam	2788	(60.6%)

This file: 'spambase.DOCUMENTATION' at the UCI Machine Learning Repository
<https://archive.ics.uci.edu/ml/datasets/Spambase>

This table shows the content of feature vector (dimension = 57)

Number	Feature	Number	Feature
1	word_freq_make: continuous.	30	word_freq_labs: continuous.
2	word_freq_address: continuous.	31	word_freq_telnet: continuous.
3	word_freq_all: continuous.	32	word_freq_857: continuous.
4	word_freq_3d: continuous.	33	word_freq_data: continuous.
5	word_freq_our: continuous.	34	word_freq_415: continuous.
6	word_freq_over: continuous.	35	word_freq_85: continuous.
7	word_freq_remove: continuous.	36	word_freq_technology: continuous.
8	word_freq_internet: continuous.	37	word_freq_1999: continuous.
9	word_freq_order: continuous.	38	word_freq_parts: continuous.
10	word_freq_mail: continuous.	39	word_freq_pm: continuous.
11	word_freq_receive: continuous.	40	word_freq_direct: continuous.
12	word_freq_will: continuous.	41	word_freq_cs: continuous.
13	word_freq_people: continuous.	42	word_freq_meeting: continuous.
14	word_freq_report: continuous.	43	word_freq_original: continuous.
15	word_freq_addresses: continuous.	44	word_freq_project: continuous.
16	word_freq_free: continuous.	45	word_freq_re: continuous.
17	word_freq_business: continuous.	46	word_freq_edu: continuous.
18	word_freq_email: continuous.	47	word_freq_table: continuous.
19	word_freq_you: continuous.	48	word_freq_conference: continuous.
20	word_freq_credit: continuous.	49	char_freq_::: continuous.
21	word_freq_your: continuous.	50	char_freq_(: continuous.
22	word_freq_font: continuous.	51	char_freq_[: continuous.
23	word_freq_000: continuous.	52	char_freq_!/: continuous.
24	word_freq_money: continuous.	53	char_freq_\$: continuous.
25	word_freq_hp: continuous.	54	char_freq_#: continuous.
26	word_freq_hpl: continuous.	55	capital_run_length_average: continuous.
27	word_freq_george: continuous.	56	capital_run_length_longest: continuous.
28	word_freq_650: continuous.	57	capital_run_length_total: continuous.
29	word_freq_lab: continuous.		

4.2. Classification using SVM

We will study the influence of the regularization parameter P in an SVM classifier. P is a real, positive number that controls the importance given to misclassified training samples. A large value of P tends to produce an overfitted model and increases the computational time. A very low value of P tends to produce an underfitted model with poor performance (for more information check lecture slides).

Note that the P parameter is denoted as ' C ' in the notebook, following the scikit-learn naming convention, and the parameter h for the gaussian kernel is denoted ' γ '.

Read the notebook **Mlearn_lab6_SVM.ipynb** and identify the following sections:

- Dataset splitting into train (80%) and test (20%) subsets. The train set will be later split into train (75%) and validation (25%) subsets.
- Use of a linear SVM classifier with a fixed value for C
- Use of a non-linear SVM classifier with a Gaussian kernel with fixed values for C and h .

Answer the following questions:

Q1: Complete a table with the training and test errors for the linear and Gaussian SVMs. Which are the values of C in each case and the value of h for the Gaussian SVM?

Edit the notebook: Using the training and validation splits, find the optimal values for the hyperparameter C for the linear SVM. Compute the training and validation errors for the following values: C= 0.1, 1, 10, 100, 500. Make a 2D plot with the training and validation scores as a function of C.

Q2: Plot the training and validation scores, find the optimal value of C

Q3: For the best classifier found in the previous step, compute classification error on the test set, compare with the error obtained for the non-optimized linear classifier (Q1).

Edit the notebook: Using the training and validation splits, find the optimal values for the hyperparameters C and h for the Gaussian SVM. Compute the training and validation errors for the following values: C = 0.1, 1, 10, 100, 500 ; h = 0.0001, 0.001, 0.01, 0.1, 1.

Make a 3D plot with the training scores as a function of C and h. Repeat for the validation error.

Q4: Plot the training and validation scores (two 3D plots). Find the optimal values of C and h

Q5: For the best classifier found in the previous step, compute classification error on the test set, compare with the error obtained for the non-optimized Gaussian classifier (Q1) and for the results of the optimized linear SVM (Q3).

4.3. Analysis of the classifier performance

To evaluate the performance of the binary classifier (here SPAM vs NON-SPAM o MAIL) we consider the following metrics. We assume that the positive class is the SPAM class, while MAIL is the negative class.

TP (True Positives): number of SPAM samples classified as SPAM

FP (False Positives): number of MAIL samples classified as SPAM

TN (True Negatives): number of MAIL samples classified as MAIL

FN (False Negatives): number of SPAM samples classified as MAIL

The classification **Error** can be expressed as
$$E = \frac{FP + FN}{TP + FP + TN + FN}$$

The **Accuracy** is defined as
$$Acc = 1 - E = \frac{TP + TN}{TP + FP + TN + FN}$$
, it is the proportion of samples correctly classified among all classified samples).

Error and *Accuracy* are simple and intuitive metrics, yet they may be poor measures for imbalanced data (if we have much more samples from one of the two classes).

Additional measures are **Precision**, **Recall** or **Sensitivity**, **Specificity** and **F-Score**:

$$P = \frac{TP}{TP + FP} = \frac{\text{\#correctly classified as SPAM}}{\text{\#classified as SPAM}}$$

$$R = \frac{TP}{TP + FN} = \frac{\text{\#correctly classified as SPAM}}{\text{\#total of SPAM}}$$

$$Sp = \frac{TN}{TN + FP} = \frac{\text{\#correctly classified as MAIL}}{\text{\#total of MAIL}}$$

$$F_{score} = 2 \frac{P \cdot R}{P + R}$$

In the following we will use the non-linear SVM with a Gaussian kernel with the optimal C and h parameters found in the previous section.

Q6: Compute the confusion matrix for the test set. Compute the six metrics (*error*, *accuracy*, *precision*, *recall*, *specificity* and *f-score*).

Q7: Explain why *precision*, *recall*, *specificity* and *f-score* are more appropriate than *accuracy* and *error* for evaluating the classifier performance.