Introduction to Machine Learning
Work 4
Support Vector Machine exercises

Contents

1 Support vector machine exercises

1.1 Introduction

The goals of this exercise are:

1. Learn about support vector machine (SVM).
2. Understand the basis of kernels and use them appropriately in a data set.
3. Understand the importance of statistical comparison and use some methods for hypothesis testing.

For the validation of the different combinations of SVMs, you need to use a T-Test or another statistical method. Remember that you have a **mandatory reading proposal** on this topic:

[1] Janez Demšar. 2006. Statistical Comparisons of Classifiers over Multiple Data Sets. J. Mach. Learn. Res. 7 (December 2006), 1-30.

This article [1] details how to compare two or more learning algorithms with multiple data sets.

1.2 Methodology of the analysis

As in the previous work assignment, you will analyze the behavior of the different algorithms by comparing the results in two well-known data sets (medium and large size) from the UCI repository. In that case, you will only use a SVM algorith. In this assignment, you will also receive the data sets defined in .arff format but divided in ten training and test sets (they are the *10-fold cross-validation* sets you will use for this exercise). As detailed below, this work is divided in two exercises.

1.3 Exercise 1

1. Download from racó a python file named **`exercise1_svm.py`** in which you have to analyze three simple data sets.

2. For each data set you have a corresponding python function. In this function, you will call a SVM algorithm with three kernel functions. You will plot the hyperplane that separate the data set and its support vectors.

3. Make a prediction for the test data set with the SVM classifier and, in the console, write the number of instances correctly predicted and the total number of instances to predict. You can use `sklearn` library and the SVM algorithm included on it inside your python function.

4. In the report, for each one of the data sets, plot the different kernel functions analyzed and justify which is the best option for each one of them, taking into account that each data set has a different linear distribution.

1.4 Exercise 2

1. Use the parser developed in previous assignment for reading and saving the information from a training and their corresponding testing files in arff format.

2. Use the Python function that automatically repeats the process described in previous step for the 10-fold cross-validation files. That is, read automatically each training case and run each one of the test cases in the selected classifier.

3. Write a Python function for classifying, using a SVM algorithm, each instance from the `TestMatrix` using the `TrainMatrix` to a classifier called `SVM_Algorithm(...)`. You decide the parameters for this classifier. You can use `sklearn` library and the SVM algorithm included on it inside your python function. **Justify your implementation and add all the references you have considered for your decisions.**

4. For the kernel function, you must consider different alternatives (at least three, you can use the predefined kernels included in `sklearn` or implement your own as a `precomputed` kernel in `sklearn`) and optimize the parameters of them to obtain the best results with the SVM algorithm in your data sets. That is, each one of the data sets analyzed may have a different set of parameters.

    a. For evaluating the performance of the SVM algorithm, we will use the percentage of correctly classified instances. This information will be used for the evaluation of the algorithm. You can store your results in a memory data structure or in a file. Keep in mind that you need to compute the average accuracy over the 10-fold cross-validation sets.

    At the end, you will have a SVM algorithm with several kernel functions (you will choose them) and with different settings in the hyper-parameters of the classifier. You should analyze the behavior of these kernel functions and parameters in the SVM algorithm and decide which combination results in the **best SVM algorithm for each data set**.

    You can compare your results in terms of classification accuracy and efficiency. Extract conclusions by analyzing **two data sets** (at least one should be large).

1.5 Work to deliver

In this work, you will use a SVM algorithm with different kernel functions in both exercises to extract conclusions from your analysis. At the end, you will find a list of the data sets available for the second exercise.

You will use your code in Python to extract the performance of the different combinations. Performance will be measured in terms of classification accuracy and efficiency. The accuracy

measure is the average of correctly classified cases. That is the number of correctly classified instances divided by the total of instances in the test file. The efficiency is the average problem-solving time. For the evaluation, in the second exercise, you will use a T-Test or another statistical method [1].

From the accuracy and efficiency results, you will extract conclusions showing graphs of such evaluation and reasoning about the results obtained.

In your analysis, you will include several considerations.

1. You will analyze the SVM (with different kernel functions). You will analyze which is the most suitable combination of the different kernel functions analyzed. The one with the highest accuracy. This SVM combination will be named as the best SVM.

2. Once you have decided the best SVM combination. You will analyze it in front of using this combination with different hyper-parameters. The idea is to improve the SVM algorithm at each one of the algorithms analyzed.

   For example, some of questions that it is expected you may answer with your analysis:

   - Which is the best kernel function for a SVM classifier?
   - Did you find differences in performance among the different kernel functions used at the SVM algorithm?
   - According to the data sets chosen, in both exercises, which combination of hyper-parameters let you to improve at the maximum the accuracy of the SVM?

   Apart from explaining your decisions and the results obtained, it is expected that you reason each one of these questions along your evaluation.

   Additionally, **you should explain how to execute your code**. Remember to add any reference that you have used in your decisions.

You should deliver a **ZIP file,** which will include the code of both exercises in Python, the datasets analyzed as well as the report, in Racó by **January, 7th, 2018**.

2 Data sets

Below, you will find a table that shows in detail the data sets that you can use in this work. All these data sets are obtained from the UCI machine learning repository. First column describes the name of the domain or data set. Next columns show #Cases = Number of cases or instances in the data set, #Num. = Number of numeric attributes, #Nom = Number of nominal attributes, #Cla. = Number of classes, Dev.Cla. = Deviation of class distribution, Maj.Cla. = Percentage of instances belonging to the majority class, Min.Cla. = Percentage of instances belonging to the minority class, MV = Percentage of values with missing values (it means the percentage of unknown values in the data set). When the columns contain a '-', it means a 0. For example, the Glass data set contains 0 nominal attributes and it is complete as it does not contain missing values.

	Domain	#Cases	#Num.	#Nom.	#Cla.	Dev.Cla.	Maj.Cla.	Min.Cla.	MV
	Adult	48,842	6	8	2	26.07%	76.07%	23.93%	0.95%
	Audiology	226	-	69	24	6.43%	25.22%	0.44%	2.00%
	Autos	205	15	10	6	10.25%	32.68%	1.46%	1.15%
*	*Balance scale*	625	4	-	3	18.03%	46.08%	7.84%	-
*	*Breast cancer Wisconsin*	699	9	-	2	20.28%	70.28%	29.72%	0.25%
*	*Bupa*	345	6	-	2	7.97%	57.97%	42.03%	-
*	*cmc*	1,473	2	7	3	8.26%	42.70%	22.61%	-
	Horse-Colic	368	7	15	2	13.04%	63.04%	36.96%	23.80%
*	*Connect-4*	67,557	-	42	3	23.79%	65.83%	9.55%	-
	Credit-A	690	6	9	2	5.51%	55.51%	44.49%	0.65%
*	*Glass*	214	9	-	2	12.69%	35.51%	4.21%	-
*	*TAO-Grid*	1,888	2	-	2	0.00%	50.00%	50.00%	-
	Heart-C	303	6	7	5	4.46%	54.46%	45.54%	0.17%
	Heart-H	294	6	7	5	13.95%	63.95%	36.05%	20.46%
*	*Heart-Statlog*	270	13	-	2	5.56%	55.56%	44.44%	-
	Hepatitis	155	6	13	2	29.35%	79.35%	20.65%	6.01%
	Hypothyroid	3,772	7	22	4	38.89%	92.29%	0.05%	5.54%
*	*Ionosphere*	351	34	-	2	14.10%	64.10%	35.90%	-
*	*Iris*	150	4	-	3	-	33.33%	33.33%	-
*	*Kropt*	28,056	-	6	18	5.21%	16.23%	0.10%	-
*	*Kr-vs-kp*	3,196	-	36	2	2.22%	52.22%	47.78%	-
	Labor	57	8	8	2	14.91%	64.91%	35.09%	55.48%
*	*Lymph*	148	3	15	4	23.47%	54.73%	1.35%	-
	Mushroom	8,124	-	22	2	1.80%	51.80%	48.20%	1.38%
*	*Mx*	2,048	-	11	2	0.00%	50.00%	50.00%	-
*	*Nursery*	12,960	-	8	5	15.33%	33.33%	0.02%	-
*	*Pen-based*	10,992	16	-	10	0.40%	10.41%	9.60%	-
*	*Pima-Diabetes*	768	8	-	2	15.10%	65.10%	34.90%	-
*	*SatImage*	6,435	36	-	6	6.19%	23.82%	9.73%	-
*	*Segment*	2,310	19	-	7	0.00%	14.29%	14.29%	-
	Sick	3,772	7	22	2	43.88%	93.88%	6.12%	5.54%
*	*Sonar*	208	60	-	2	3.37%	53.37%	46.63%	-
	Soybean	683	-	35	19	4.31%	13.47%	1.17%	9.78%
*	*Splice*	3,190	-	60	3	13.12%	51.88%	24.04%	-
*	*Vehicle*	946	18	-	4	0.89%	25.77%	23.52%	-
	Vote	435	-	16	2	11.38%	61.38%	38.62%	5.63%
*	*Vowel*	990	10	3	11	0.00%	9.09%	9.09%	-
*	*Waveform*	5,000	40	-	3	0.36%	33.84%	33.06%	-
*	*Wine*	178	13	-	3	5.28%	39.89%	26.97%	-
*	*Zoo*	101	1	16	7	11.82%	40.59%	3.96%	-