Spam Detection Using Support Vector Machine

Group 1 Members:

- Chenlyu Zhao - czhao96@wisc.edu
- Hangpeng Li - hli578@wisc.edu
- Zhengjia Mao - zmao27@wisc.edu

Abstract

In this project, the concept and background of Support Vector Machine(SVM) will be introduced to the students who are interested in machine learning. SVM is a supervised machine learning model and is a powerful tool to solve many real-world classification problems. Students will have a chance to mathematically understand the process of classifying the data by maximizing the margin in both linear problems and non-linear problems. Then a hands-on SVM problem related to spam detection will be given to students, which makes this project unique and interesting. Students are required to identify the bag of words correlated to spam emails. The well-designed warm-ups will help students to gradually prepare themselves and eventually solve this real-world problem using kernel tricks. Also, the students will have a glimpse at dual representation and kernel's relation, and know what else kernel based SVM is capable of. After completing the exercises, students will understand the concepts of SVM in-depth and have a taste of using the techniques in practice.

Background

Support Vector Machine is a supervised machine learning model that can output the best decision boundaries, also known as hyperplane, to separate data with different features. Before the 1980s, the majority of learning methods were linear linear decision surfaces. Non-linear classifiers such as Decision Trees and NearestNeighbors were invented around the 1980s and continued to evolve to SVM based on the development in computational learning theory[1].

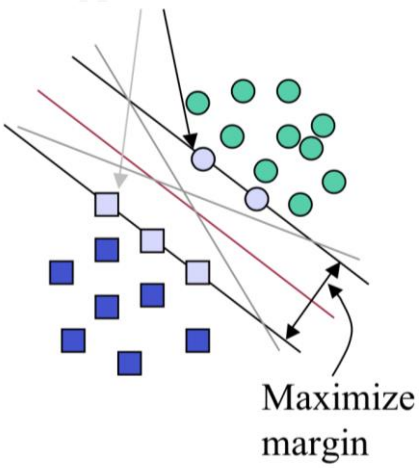
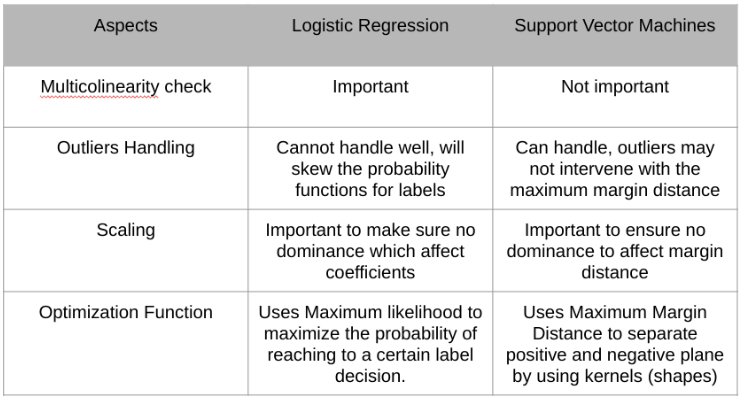


Figure 1: Separating the data by maximizing margin[1]

SVM and logistic regression are both used to solve classification problems. However, unlike logistic regression, which has different decision boundaries according to different weights, SVM is trying to find the best line (maximizes the margin) that separates classes.



Aspects	Logistic Regression	Support Vector Machines
Multicolinearity check	Important	Not important
Outliers Handling	Cannot handle well, will skew the probability functions for labels	Can handle, outliers may not intervene with the maximum margin distance
Scaling	Important to make sure no dominance which affect coefficients	Important to ensure no dominance to affect margin distance
Optimization Function	Uses Maximum likelihood to maximize the probability of reaching to a certain label decision.	Uses Maximum Margin Distance to separate positive and negative plane by using kernels (shapes)

Figure 2: A comparison between Logistic Regression and SVM [2]

When the users have a large number of features, SVMs are particularly helpful as it stands out with its effectiveness among other supervised binary classifiers. SVM can classify nonlinear data by mapping space to a higher dimension while sidestepping the time-consuming calculation by applying kernel trick. But keep in mind, SVM is not a wild card. Some known issues with SVM are that it performs poorly when the model is trained from weakly-labeled, noisy, and poor-quality data and the model selection is computationally expensive[3]. As maximizing margin has been mentioned many times above, margin has a form as shown in Equation 1.

$$m = \frac{1}{||w||_2} = ||w||_2^{-1} \tag{1}$$

So a correct classifier is derived as shown in Equation 2.

$$d_i x_i^T w_i \geq 1 \tag{2}$$

Given Matrix *D* represents the labels of the data, *X* represents the features of the data, and *W* represents the weights of each feature. To eliminate the effect of easy-to-classify data, we use hinge loss as our loss function in SVM. It is shown below as Equation 3.

$$\underline{min_w \sum_{i=1}^{N} \quad (1 - d_i x_i^T w_i)_+} \tag{3}$$

The symbol $(\,)_+$means if the value inside the parenthesis is positive, it takes the value; if the value is negative instead, it takes zero. To maximize margin (equation 1), it's the same to minimize its inverse: $|w||_2^2$. So the SVM using hinge loss will be expressed in Equation 4.

$$\underline{min_w \sum_{i=1}^{N} \quad (1 - d_i x_i^T w_i)_+ + \lambda ||w||_2^2} \tag{4}$$

At the same time, the potential ethical problems raised by SVM should also be aware of. Tons of personal data would be used as training data sets, and their information should be protected. Personal information in this study will only be served for experimental purposes. Hard copies of transcribed notes will be shredded. Additionally, soft-copies of transcriptions will be deleted from all electronic media storage and portable forms of electronic media storage containing this data such as: USBs and memory sticks will be destroyed. Also technicians and researchers might rely on the output given by SVM too much so that bias could occur.

Warm-ups

By applying the materials learned in class, you should be able to understand how to apply SVM models to classify data sets conceptually. To be more specific, SVM finds the best hyperplane by maximizing the margin from different classes. They are supposed to identify the best hyperplane in different scenarios: hyperplanes that segregate the data well; having the wrong catalog as an outlier; data that is not linearly available. You should also be able to find the weight vector w and represent the equation corresponding to the decision boundary(hyperplane) using Regression or SVM. They are supposed to find that the square-error way is easily affected by outliers. While this is not required, try to achieve zero tolerance by applying a perfect partition, you can recognize the trade-off. In the following warm-ups, you will be introduced to a mathematical aspect of SVM and how it handles linear and non-linear problems.

Warm-up #1 Classify 1-D linear classes by maximizing margin by hands

Before diving into the mathematical derivation, let’s start with doing some 1-D classification that separates the data while maximizing the margin. Table 1 contains 6 data points of fruits. The only feature here is the weight, and the label +1 represents it is an apple, and label -1 represents it is not an apple. Draw the following data points on an axis and a boundary that maximizes the margin.

Table 1: sample data points to recognize apples

Weight(lbs)	Label
0.33	+1
0.72	-1
0.45	-1
0.26	+1
0.38	+1
0.52	-1

Answer:

You might have realized while the boundary is not unique, but there is only one max margin boundary in the above problem. The features can be written as a 6x2 matrix X, and the labels are represented in a 6x1 matrix y. The weight is a 2x1 matrix w. So we have the following relationship as $y = sign(Xw)$. Calculate the weight matrix w. And then calculate the hinge loss using Equation(3).

X = [0.33 1; 0.72 1; 0.45 1; 0.26 1; 0.38 1; 0.52 1]
y = [+1 -1 -1 +1 +1 -1]^T
w = [w_1 w_2]^T

Answer:

Warm-up #2 [4]: Classify 2-D linear classes by maximizing margin using Jupyter

Download the given *WarmUp2.ipynb* and open it with jupyter notebook. Complete the missing lines, save your output figures, and attach them to the following question if needed.

a. Is your simple line that separates the classes unique?
b. Which margin could be chosen as the optimal model? WhatJustify your answer.
c. Does changing the training points to 120 affect the classification result? Describe what you observe.

Answer:

There are more than one dividing line possible to discriminate between the two classes. SVM selects the best fitting line via finding the line with maximized margin, since points near the decision surface represent very uncertain classification decisions, A classifier with a large margin makes no low certainty classification decisions. This gives a classification safety margin as a slight error in measurement or a slight variation will not cause a misclassification. Even though the training points doubled, none of the support vectors changed. So the best fitting line hasn't changed. This insensitivity toward the distant points or outlayer makes SVM more accurate.

Warm-up #3: Solving nonlinear separation problem with SVM

When the scenario has more than one feature, separating data becomes complicated. Sometimes data points are not linearly separable, a linear classifier is not sophisticated enough. You will be using SVM with kernels then. Kernel trick is a technique which takes a low dimensional input space and transforms it to a higher dimensional space, so that we don't have to add a new feature manually to have a hyper-plane.

Kernel methods include linear kernel, polynomial kernel, gaussian kernel, etc. They are used differently depending on the scenario. Let's take the gaussian kernel as an example. The gaussian kernel is just like a function that measures the distance between the data points. The function is shown below in Equation 5.

$$K(x^i, x^j) \ = \ exp(-\frac{\sum_{k=1}^{n} \quad (x_k^i - x_k^j)^2}{2\sigma^2} \ ) \tag{5}$$

Solve by hands to find the K(X$_1$, X$_2$). Given X$_1$ = [2, 3, 4], X$_2$ = [4, 5, 6], σ = 2.

Answer:

Warm-up #4 [5]: Download the given *WarmUp4.ipynb* and open it with jupyter notebook. In this problem, the data points are obviously categorized into two classes with circular shape.

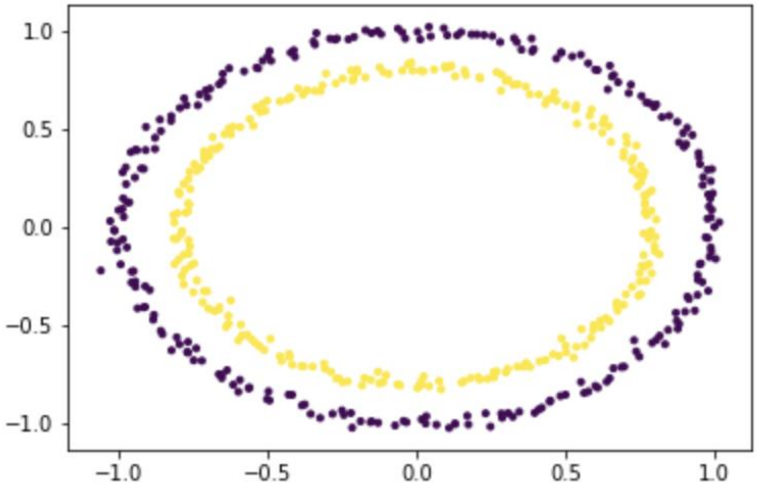


Figure 3: Data points in circular shape

A trick to separate them is to convert the 2-D problem to a 3-D problem. Run through the given code and you will see the data points are now linearly separable in the 3rd axis. A linear kernel will then be enough to separate them while maximizing the margin. The equation(6) is shown.

$$K(x, y) = x^T w + b \tag{6}$$

Complete the missing lines and save your output figures. Does the linear kernel hyper-plane make sense? Justify your answer.

Answer:

Exercises[7]

The appetizers might not be satisfied, so let's switch gears to some hands-on spam ham problems with SVM. First, download the given *BestExercise.ipynb* and *BestExerciseDataset.zip*. Unzip the dataset file and put them into the same folder with the ipynb file, open them with jupyter notebook. Take a brief look at the content and work through the following tasks. While the project content is mostly covered in the class, students will also get a chance to learn a new and useful technique called preprocessing. For example, unifying the formats and removing the useless symbols before extracting the features are important. Since our aim is to classify on Messages or Emails, students are only required to conceptually understand the mechanism behind preprocessing of the dataset.

Task #1: The dataset you will be using has already been preprocessed by our genius staff. Due to the package use limitations, you will not be required to perform a preprocessing by yourself. Compare the preprocessed data(ham_word and spam_word) and the original email data(data). Please describe how you would preprocess the original email data to the given data format?

Answer:

Task #2: Complete the sum_indicator() function and the sub_gradientdescent() function in the given file. Reference to the lecture video 5.7 for formulas. Then complete the my_svm function that calls the previous two functions and use the input training set to select the best w using gradient descent. This my_svm() will serve the next task for cross-validation. You will then use the given function to find the best lambda. What is the best lambda value you get?

Answer:

Task #3: Use the best lambda you found from the previous task, run a cross-validation on the dataset. The folds have been splitted for you. Use the training set with different fold combinations to train a group of w (you will get one best w for each running). Evaluate how each w performans on the validation set. Store the best w with the minimal error count in the validation set. You will be using this best w in the next task. Running the code might take a few minutes, be patient!

Answer:

Task #4: You have come a long way preparing your classifier, now use the classifier you have developed with the best parameter and run a prediction on the test set. This time you will be using the built-in svm function from sklearn package and run three different kinds of kernel methods and count the number of errors. Which performs the best in our case, and which one performs the worst? Make a reasonable speculation to explain the difference using your knowledge about kernel methods.

Answer:

Task #5 You might have noticed that the number of features in the SVM implemented by you is a huge number. In real life problem solving, we cannot rely on the number of features. To do so, there is a new optimization problem called: dual representation. The previous optimization is called: primal representation. The weight in the primal representation, which we have seen before, are weights on the *features*. Dual representation gives weights of *training vectors.*

New classifier: $Y(x) = sign(\beta) + \sum_{i=1}^{N} \quad \alpha i \; yi \; (xi \cdot x)$

In the formula, α is the weights over the training data. The way to get α and β is using the perceptron algorithm. Repeating increasing αi by 1 and set β = β + yi*R**2 if the training vector is mis-clarified.

Since this requires a deep math foundation of Lagrange and KKT condition, we are not letting you implement or derive a formula. Instead, look at the kernel method expression below, and answer the question.

Kernel based SVM decision boundary: $Y(x) = sign(\sum_{i=1}^{N} \alpha i\ K(xi, x))$, $K(xi, x) = \sum_{j=1}^{q} \varphi j(xi) * \varphi j(x)$

Look at the form of expression of dual representation and kernel based SVM, what information do you get?

Answer:

By Authors

Congratulations! You have completed this project and we hope you enjoyed it while learning the new topic. Understanding the mathematics behind the algorithms is extremely helpful so that you are able to apply them in different programming languages even when Python becomes the past tense. A long term expectation is that you are able to break down the big concept of machine learning into specific applications that can handle different real-life scenarios.

References

[1]
Bob Berwick. Massachusetts Institute of Technology. An Idiot's guide to Support vector machines (SVMs). http://web.mit.edu/6.034/wwwbob/. Accessed April 25, 2020.

[2]
Vincent Tatan. Medium. Your Beginner Guide to Basic Classification Models: Logistic Regression and SVM. https://towardsdatascience.com/your-beginner-guide-to-basic-classification-models-logistic-regression-and-svm-b7eef864ec9a. Accessed April 25, 2020.

[3]
Jakub Nalepa and Michal Kawulok. Springer. Selecting training sets for support vector machines: a review. https://link.springer.com/article/10.1007/s10462-017-9611-1. Accessed April 26, 2020.

[4]

Jake VanderPlas. Github. In-Depth: Support Vector Machines. https://jakevdp.github.io/PythonDataScienceHandbook/05.07-support-vector-machines.html. Accessed April 25, 2020.

[5]
Savyakhosla. GeeksforGeeks. Using SVM to perform classification on a non-linear dataset. https://www.geeksforgeeks.org/ml-using-svm-to-perform-classification-on-a-non-linear-dataset/. Accessed April 25, 2020.

[6]
Mohtadi Ben Fraj. Medium. In Depth: Parameter tuning for SVC. https://medium.com/all-things-ai/in-depth-parameter-tuning-for-svc-758215394769. Accessed April 25, 2020.

[7]
Deja vu. Kaggle. SMS: Spam or Ham (Beginner). https://www.kaggle.com/dejavu23/sms-spam-or-ham-beginner/notebook#Part-0:-Imports,-define-functions. Accessed April 25, 2020.

[8]
Cosma Shalizi. Carnegie Mellon University. Support Vector Machines. https://www.stat.cmu.edu/~cshalizi/350/lectures/27/lecture-27.pdf. Accessed August 30, 2020.