Understanding Support Vector Machine(SVM) algorithm from examples (along with code)

analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code

September 12, 2017

1. Home

Sunil Ray — Published On September 13, 2017 and Last Modified On August 26th, 2021

Note: This article was originally published on Oct 6th, 2015 and updated on Sept 13th, 2017

Overview

- Explanation of support vector machine (SVM), a popular machine learning algorithm or classification
- Implementation of SVM in R and Python
- Learn about the pros and cons of Support Vector Machines(SVM) and its different applications

Introduction

Mastering machine learning algorithms isn't a myth at all. Most beginners start by learning regression. It is simple to learn and use, but does that solve our purpose? Of course not! Because you can do so much more than just Regression!

Think of machine learning algorithms as an armory packed with axes, swords, blades, bows, daggers, etc. You have various tools, but you ought to learn to use them at the right time. As an analogy, think of 'Regression' as a sword capable of slicing and dicing data efficiently, but incapable of dealing with highly complex data. On the contrary, 'Support Vector Machines' is like a sharp knife – it works on smaller datasets, but on complex ones, it can be much stronger and powerful in building machine learning models.

By now, I hope you've now mastered Random Forest, Naive Bayes Algorithm, and Ensemble Modeling. If not, I'd suggest you take out a few minutes and read about them as well. In this article, I shall guide you through the basics to advanced knowledge of a crucial machine learning algorithm, support vector machines.

You can learn about Support Vector Machines in course format here (it's free!):

> Support Vector Machines (SVM) in Python and R

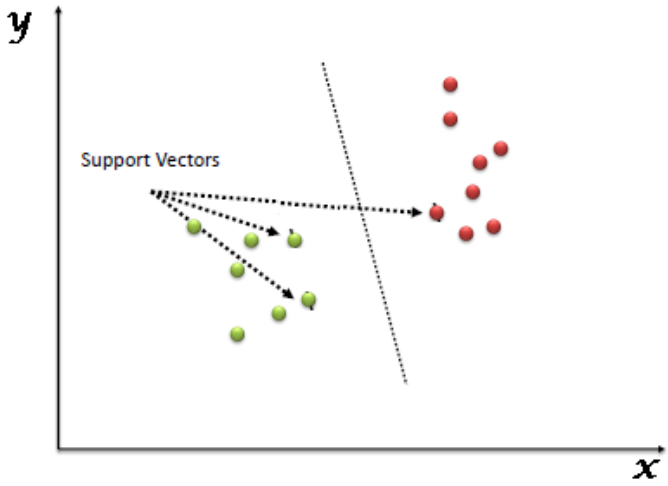
Table of Contents

What is the Support Vector Machine?

"Support Vector Machine" (SVM) is a supervised machine learning algorithm that can be used for both classification or regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is a number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well (look at the below snapshot).

Support Vectors are simply the coordinates of individual observation. The SVM classifier is a frontier that best segregates the two classes (hyper-plane/ line).

You can look at support vector machines and a few examples of their working here.

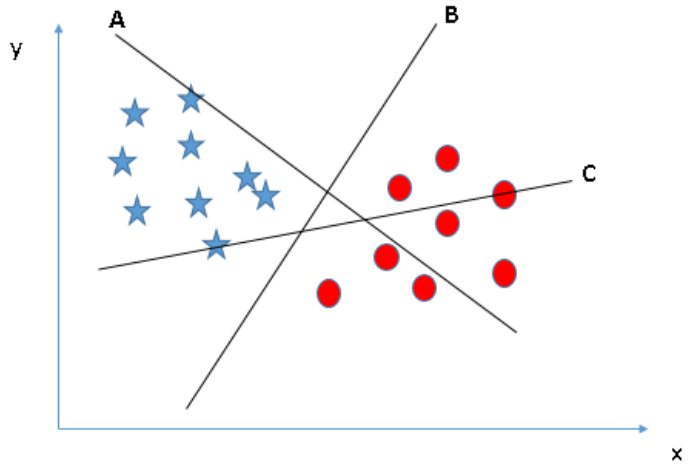


How does it work?

Above, we got accustomed to the process of segregating the two classes with a hyper-plane. Now the burning question is "How can we identify the right hyper-plane?". Don't worry, it's not as hard as you think!

Let's understand:

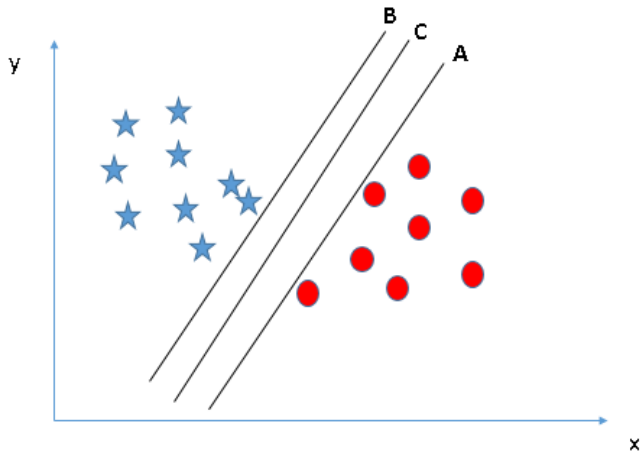
- **Identify the right hyper-plane (Scenario-1):** Here, we have three hyper-planes (A, B, and C). Now, identify the right hyper-plane to classify stars and circles.



You need to remember a thumb rule to identify the right hyper-plane: "Select the hyper-plane which segregates the two classes better". In this scenario, hyper-plane "B" has excellently performed this job.

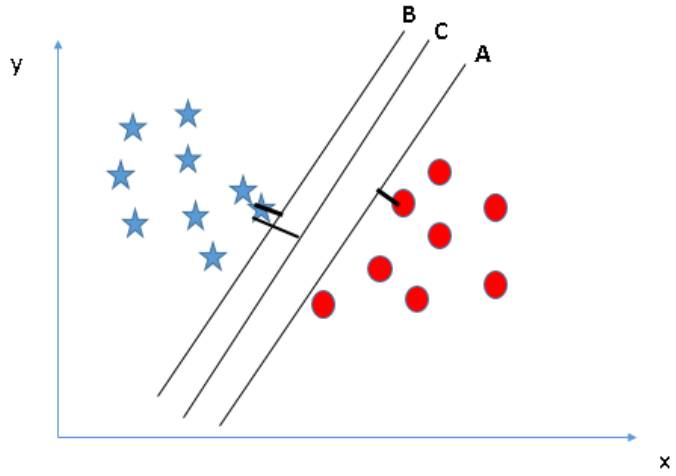
- **Identify the right hyper-plane (Scenario-2):** Here, we have three hyper-planes (A, B, and C) and all are segregating the classes well. Now, How can we identify the right hyper-plane?

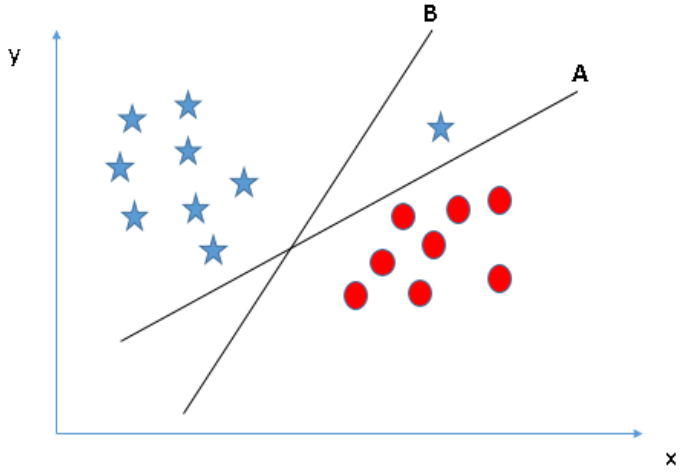
Here, maximizing the distances between nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as **Margin**. Let's look at the below snapshot:



Above, you can see that the margin for hyper-plane C is high as compared to both A and B. Hence, we name the right hyper-plane as C. Another lightning reason for selecting the hyper-plane with higher margin is robustness. If we select a hyper-plane having low margin then there is high chance of miss-classification.

- **Identify the right hyper-plane (Scenario-3):**Hint:Use the rules as discussed in previous section to identify the right hyper-plane

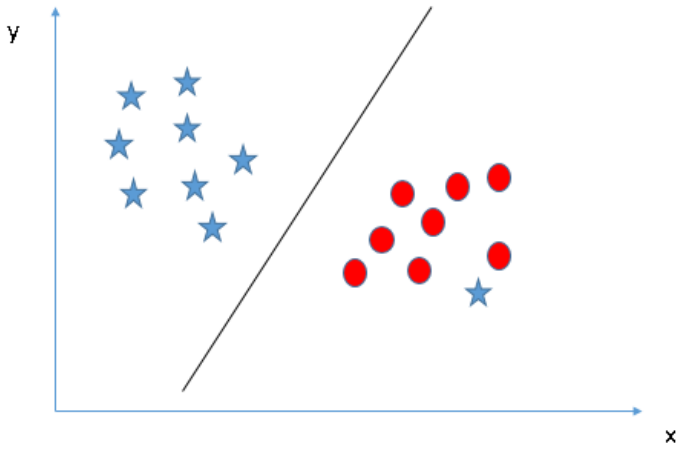




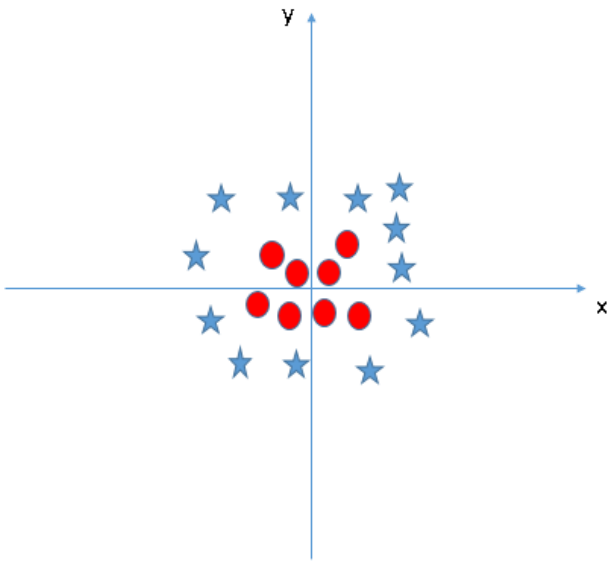
Some of you may have selected the hyper-plane **B** as it has higher margin compared to **A.** But, here is the catch, SVM selects the hyper-plane which classifies the classes accurately prior to maximizing margin. Here, hyper-plane B has a classification error and A has classified all correctly. Therefore, the right hyper-plane is **A.**

- **Can we classify two classes (Scenario-4)?:** Below, I am unable to segregate the two classes using a straight line, as one of the stars lies in the territory of other(circle) class as an outlier.

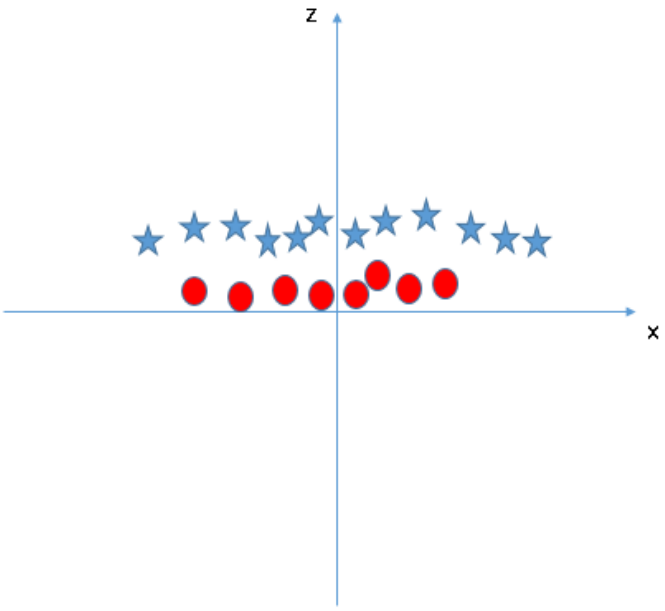
As I have already mentioned, one star at other end is like an outlier for star class. The SVM algorithm has a feature to ignore outliers and find the hyper-plane that has the maximum margin. Hence, we can say, SVM classification is robust to outliers.



- **Find the hyper-plane to segregate to classes (Scenario-5):** In the scenario below, we can't have linear hyper-plane between the two classes, so how does SVM classify these two classes? Till now, we have only looked at the linear hyper-plane.SVM can solve this problem. Easily! It solves this problem by introducing additional feature. Here, we will add a new feature z=x^2+y^2. Now, let's plot the data points on axis x and z:

  In above plot, points to consider are:
  - All values for z would be positive always because z is the squared sum of both x and y
  - In the original plot, red circles appear close to the origin of x and y axes, leading to lower value of z and star relatively away from the origin result to higher value of z.

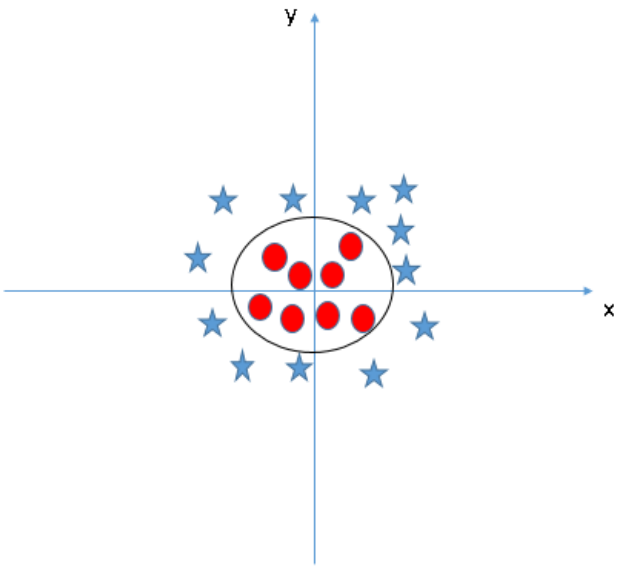
  

  In the SVM classifier, it is easy to have a linear hyper-plane between these two classes. But, another burning question which arises is, should we need to add this feature manually to have a hyper-plane. No, the SVM algorithm has a technique called the **kernel trick**. The SVM kernel is a function that takes low dimensional input space and transforms it to a higher dimensional space i.e. it converts not separable problem to separable problem. It is mostly useful in non-linear separation problem. Simply put, it does some extremely complex data transformations, then finds out the process to separate the data based on the labels or outputs you've defined.

  When we look at the hyper-plane in original input space it looks like a circle:

Now, let's look at the methods to apply SVM classifier algorithm in a data science challenge.

You can also learn about the working of Support Vector Machine in video format from this Machine Learning Certification.



How to implement SVM in Python and R?

In Python, scikit-learn is a widely used library for implementing machine learning algorithms. SVM is also available in the scikit-learn library and we follow the same structure for using it(Import library, object creation, fitting model and prediction).

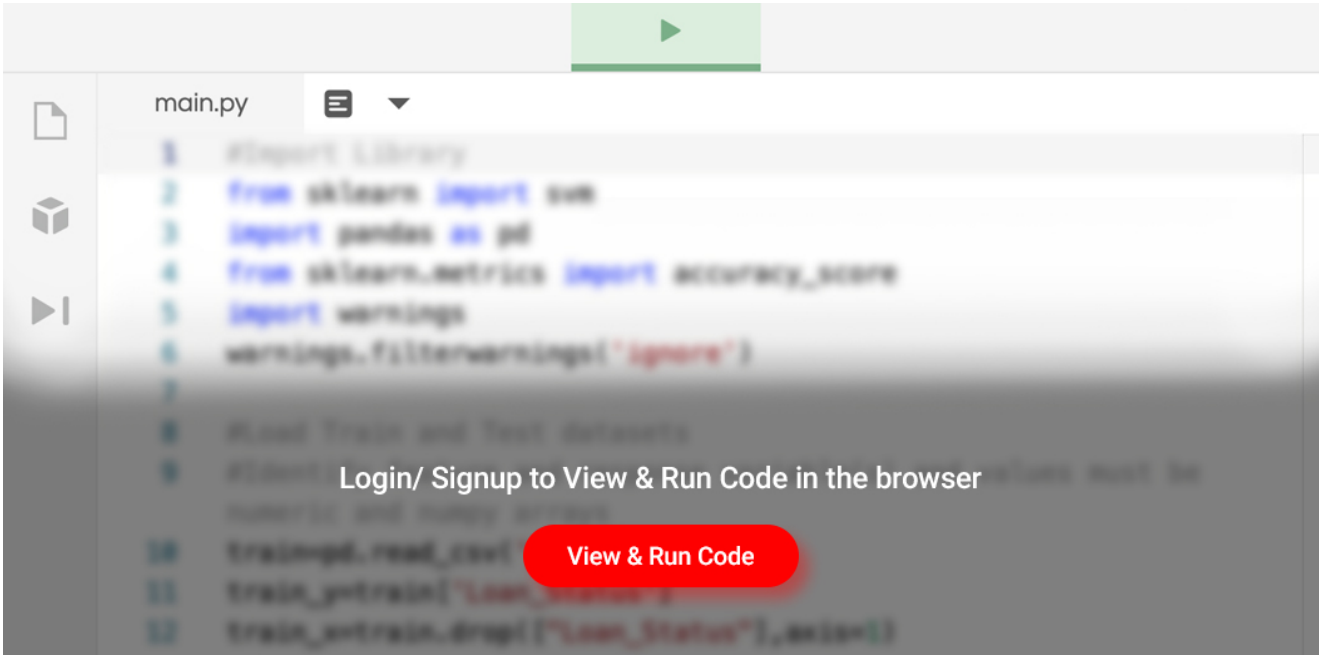
Now, let us have a look at a real-life problem statement and dataset to understand how to apply SVM for classification

Problem Statement

Dream Housing Finance company deals in all home loans. They have a presence across all urban, semi-urban and rural areas. A customer first applies for a home loan, after that the company validates the customer's eligibility for a loan.

Company wants to automate the loan eligibility process (real-time) based on customer details provided while filling an online application form. These details are Gender, Marital Status, Education, Number of Dependents, Income, Loan Amount, Credit History and others. To automate this process, they have given a problem to identify the customers' segments, those are eligible for loan amount so that they can specifically target these customers. Here they have provided a partial data set.

Use the coding window below to predict the loan eligibility on the test set. **Try changing the hyperparameters for the linear SVM to improve the accuracy.**



Support Vector Machine(SVM) code in R

The e1071 package in R is used to create Support Vector Machines with ease. It has helper functions as well as code for the Naive Bayes Classifier. The creation of a support vector machine in R and Python follow similar approaches, let's take a look now at the following code:

```
#Import Library
require(e1071) #Contains the SVM
Train <- read.csv(file.choose())
Test <- read.csv(file.choose())
# there are various options associated with SVM training; like changing kernel, gamma
and C value.

# create model
model <-
svm(Target~Predictor1+Predictor2+Predictor3,data=Train,kernel='linear',gamma=0.2,cost=


#Predict Output
preds <- predict(model,Test)
table(preds)
```

How to tune Parameters of SVM?

Tuning the parameters' values for machine learning algorithms effectively improves model performance. Let's look at the list of parameters available with SVM.

`sklearn.svm.SVC(C=1.0, kernel='rbf', degree=3, gamma=0.0, coef0=0.0, shrinking=True, probability=False,tol=0.001, cache_size=200, class_weight=None, verbose=False, max_iter=-1, random_state=None)`

I am going to discuss about some important parameters having higher impact on model performance, “kernel”, “gamma” and “C”.

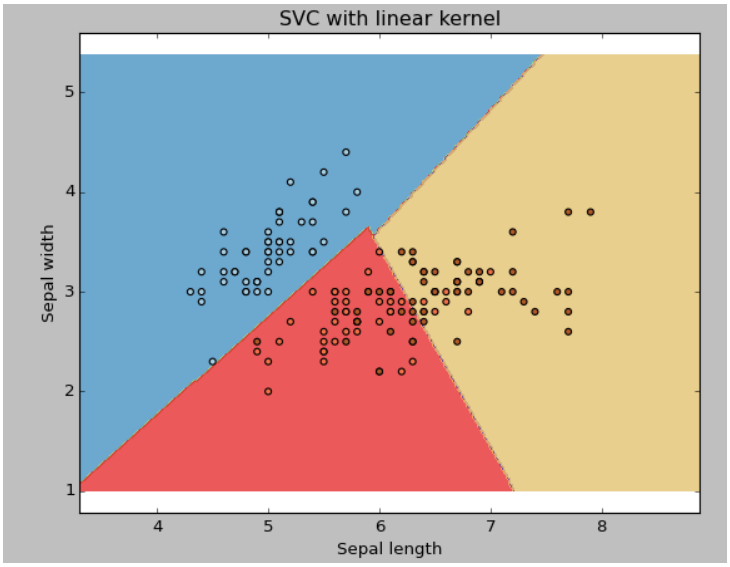
kernel: We have already discussed about it. Here, we have various options available with kernel like, “linear”, “rbf”,”poly” and others (default value is “rbf”). Here “rbf” and “poly” are useful for non-linear hyper-plane. Let’s look at the example, where we’ve used linear kernel on two feature of iris data set to classify their class.

Support Vector Machine(SVM) code in Python

Example: Have a linear SVM kernel

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn import svm, datasets

# import some data to play with
iris = datasets.load_iris()
X = iris.data[:, :2] # we only take the first two features. We could
 # avoid this ugly slicing by using a two-dim dataset
y = iris.target

# we create an instance of SVM and fit out data. We do not scale our
# data since we want to plot the support vectors
C = 1.0 # SVM regularization parameter
svc = svm.SVC(kernel='linear', C=1,gamma=0).fit(X, y)

# create a mesh to plot in
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
h = (x_max / x_min)/100
xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
 np.arange(y_min, y_max, h))

plt.subplot(1, 1, 1)
Z = svc.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)
plt.contourf(xx, yy, Z, cmap=plt.cm.Paired, alpha=0.8)
```

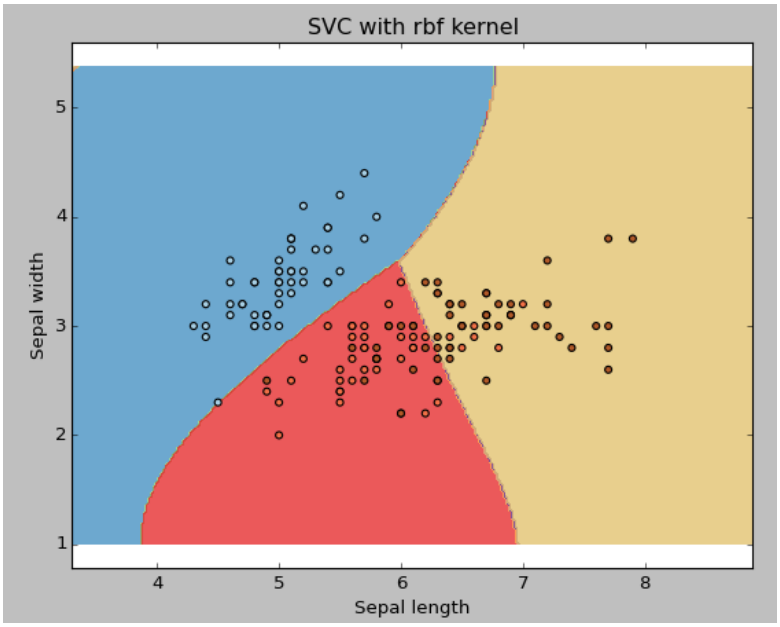
```
plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.Paired)
plt.xlabel('Sepal length')
plt.ylabel('Sepal width')
plt.xlim(xx.min(), xx.max())
plt.title('SVC with linear kernel')
plt.show()
```



Example: Use SVM rbf kernel

Change the kernel type to rbf in below line and look at the impact.

```
svc = svm.SVC(kernel='rbf', C=1,gamma=0).fit(X, y)
```

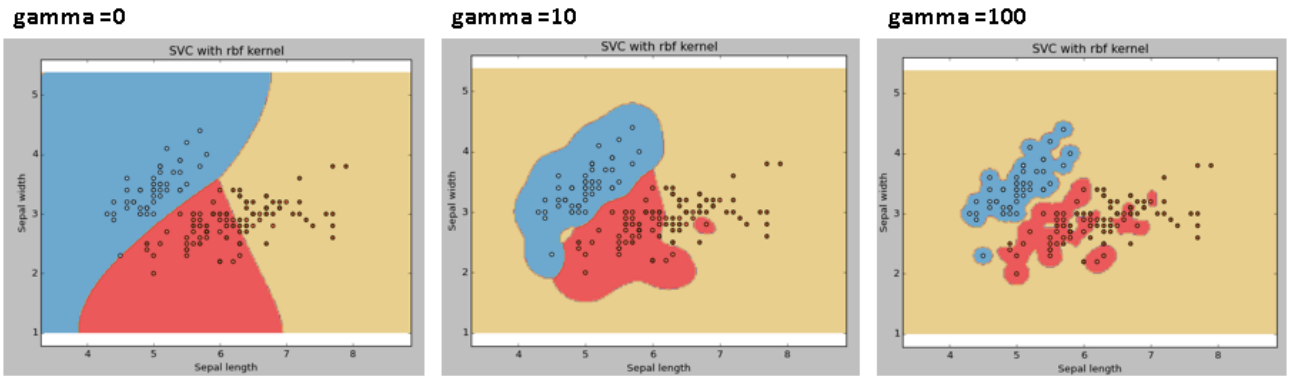


I would suggest you go for linear SVM kernel if you have a large number of features (>1000) because it is more likely that the data is linearly separable in high dimensional space. Also, you can use RBF but do not forget to cross-validate for its parameters to avoid over-fitting.

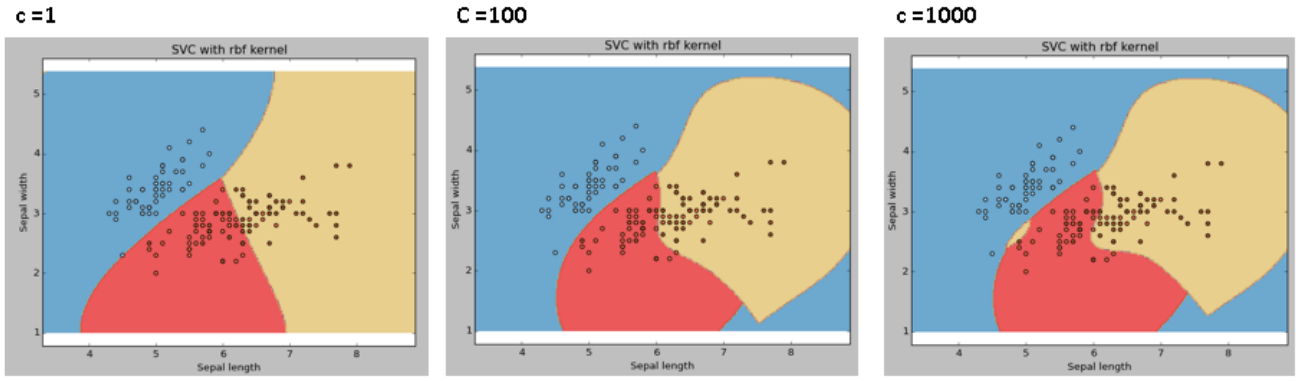
gamma: Kernel coefficient for 'rbf', 'poly' and 'sigmoid'. Higher the value of gamma, will try to exact fit the as per training data set i.e. generalization error and cause over-fitting problem.

Example: Let's difference if we have gamma different gamma values like 0, 10 or 100.

```
svc = svm.SVC(kernel='rbf', C=1,gamma=0).fit(X, y)
```



C: Penalty parameter C of the error term. It also controls the trade-off between smooth decision boundaries and classifying the training points correctly.



We should always look at the cross-validation score to have effective combination of these parameters and avoid over-fitting.

In R, SVMs can be tuned in a similar fashion as they are in Python. Mentioned below are the respective parameters for e1071 package:

- The kernel parameter can be tuned to take "Linear","Poly","rbf" etc.
- The gamma value can be tuned by setting the "Gamma" parameter.
- The C value in Python is tuned by the "Cost" parameter in R.

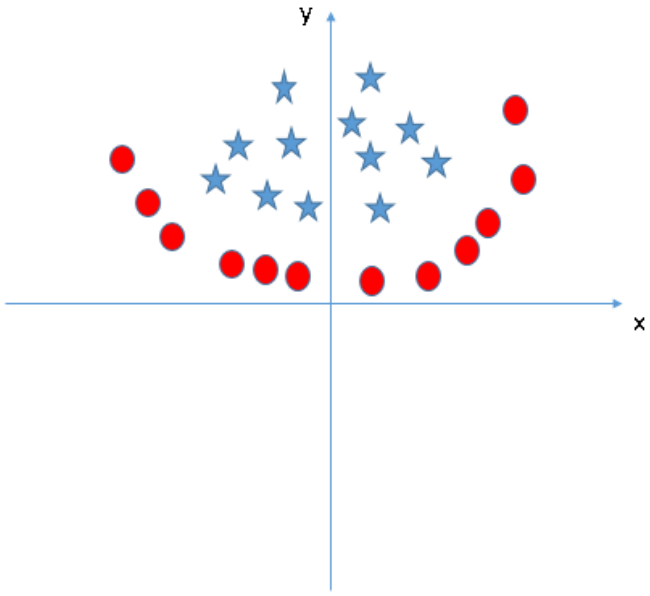
Pros and Cons associated with SVM

- **Pros:**
  - It works really well with a clear margin of separation
  - It is effective in high dimensional spaces.
  - It is effective in cases where the number of dimensions is greater than the number of samples.
  - It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
- **Cons:**
  - It doesn't perform well when we have large data set because the required training time is higher
  - It also doesn't perform very well, when the data set has more noise i.e. target classes are overlapping
  - SVM doesn't directly provide probability estimates, these are calculated using an expensive five-fold cross-validation. It is included in the related SVC method of Python scikit-learn library.

Practice Problem

Find right additional feature to have a hyper-plane for segregating the classes in below snapshot:

Answer the variable name in the comments section below. I'll shall then reveal the answer.



End Notes

In this article, we looked at the machine learning algorithm, Support Vector Machine in detail. I discussed its concept of working, the process of implementation in python, the tricks to make the model efficient by tuning its parameters, Pros and Cons, and finally a problem to

solve. I would suggest you use SVM and analyze the power of this model by tuning the parameters. I also want to hear your experience with SVM, how have you tuned parameters to avoid over-fitting and reduce the training time?

Did you find this article helpful? Please share your opinions/thoughts in the comments section below.

93 thoughts on "Understanding Support Vector Machine(SVM) algorithm from examples (along with code)"

nishant says: October 07, 2015 at 4:26 am

hi, gr8 articles..explaining the nuances of SVM...hope u can reproduce the same with R.....it would be gr8 help to all R junkies like me Reply

ASHISH says: October 07, 2015 at 7:00 am

NEW VARIABLE (Z) = SQRT(X) + SQRT (Y) Reply

Mahmood A. Sheikh says: October 07, 2015 at 9:16 am

Kernel Reply

Mahmood A. Sheikh says: October 07, 2015 at 9:19 am

I mean kernel will add the new feature automatically. Reply

Sanjay says: October 07, 2015 at 3:12 pm

Nicely Explained . The hyperplane to separate the classes for the above problem can be imagined as 3-D Parabola. z=ax^2 + by^2 + c Reply

FrankSauvage says: October 12, 2015 at 10:06 am

Thanks a lot for this great hands-on article! Reply

Harsha says: November 08, 2015 at 4:42 am

Really impressive content. Simple and effective. It could be more efficient if you can describe each of the parameters and practical application where you faced non-trivial problem examples. Reply

Aman Srivastava says: November 26, 2015 at 1:59 pm

kernel Reply

Ephraim Admassus says: February 14, 2016 at 2:16 pm

How does the python code look like if we are using LSSVM instead of SVM? Reply

Janpreet Singh says: March 04, 2016 at 12:58 pm

Polynomial kernel function?! for exzmple : Z= A(x^2) + B(y^2) + Cx + Dy + E Reply

Krishna Kalaparti says: April 18, 2016 at 11:26 am

Hi Sunil. Great Article. However, there's an issue in the code you've provided. When i compiled the code, i got the following error: Name error: name 'h' is not defined. I've faced this error at line 16, which is: "xx, yy = np.meshgrid(np.arange(x_min, x_min, h), ...). Could you look into it and let me know how to fix it? Reply

Shikha says: May 28, 2016 at 8:33 pm

great explanation :) I think new variable Z should be x^2 + y. Reply

VEERAMANI NATARAJAN says: June 03, 2016 at 6:23 am

Nice Articlel Reply

Carlos says: June 14, 2016 at 3:18 pm

The solution is analogue to scenario-5 if you replace y by y-k Reply

Rishabh says: June 15, 2016 at 11:22 am

Given problem Data points looks like y=x^2+c. So i guess z=x^2-y OR z=y-x^2. Reply

K.Krithiga Lakshmi says: June 15, 2016 at 12:07 pm

Your SVM explanation and kernel definition is very simple, and easy to understand. Kudos for that effort. Reply

pfcohen says: June 19, 2016 at 3:52 pm

Most intuitive explanation of multidimensional svm I have seen. Thank you! Reply

yc says: June 27, 2016 at 6:36 pm

what is 'h' in the code of SVM . xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h)) Reply

Iresh says: July 14, 2016 at 10:32 pm

z = (x^2 - y) z > 0, red circles Reply

LI JENG HUANG says: August 05, 2016 at 7:33 pm

very neat explaination to SVC. For the proposed problem, my answers are: (1) z = a* x^2 + b y + c, a parabola. (2) z = a (x-0)^2 + b (y- y0)^2 - R^2, a circle or an ellipse enclosing red stars. Reply

Hari says: August 18, 2016 at 10:02 am

Great article.. I think the below formula would give a new variable that help to separate the points in hyper plane z = y - |x| Reply

MADHVI says: August 23, 2016 at 5:42 am

THANKS FOR EASY EXPLANATION Reply

Raghu says: August 31, 2016 at 2:35 pm

Useful article for Machine learners.. Why can't you discuss about effect of kernel functions. Reply

Yamani says: September 22, 2016 at 6:25 pm

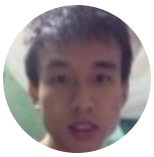
The explanation is really impressive. Can you also provide some information about how to determine the theoretical limits for the parameter's optimal accuracy. Reply

harshel jain says: September 23, 2016 at 6:49 am

how can we use SVM for regression? can someone please explain.. Reply

Diana says: October 04, 2016 at 2:57 pm

That was a really good explanation! thanks a lot. I read many explanations about SVM but this one help me to understand the basics which I really needed it. Reply



dam van tai says: October 06, 2016 at 9:00 am

i think x coodinates must increase after sqrt Reply

Manjunath GS says: October 27, 2016 at 7:08 pm

please give us the answer Reply

Diptesh says: October 28, 2016 at 3:48 pm

This is very useful for understanding easily. Reply

Dan says: November 14, 2016 at 6:41 pm

just substitude x with |x| Reply

Min says: November 23, 2016 at 8:13 am

Same goes with Diana. This really help me a lot to figure out things from basic. I hope you would also share any computation example using R provided with simple dataset, so that anyone can practice with their own after referring to your article. I have a question, if i have time-series dataset containing mixed linear and nonlinear data, (for example oxygen saturation data ; SaO2), by using svm to do classification for diseased vs health subjects, do i have to separate those data into linear and non-linear fisrt, or can svm just performed the analysis without considering the differences between the linearity of those data? Thanks a lot! Reply

anu says: November 29, 2016 at 7:53 pm

z Reply

Renny Varghese says: December 03, 2016 at 7:10 pm

Could you please explain how SVM works for multiple classes? How would it work for 9 classes? I used a function called multisvm here: http://www.mathworks.com/matlabcentral/fileexchange/39352-multi-class-svm but I'm not sure how it's working behind the scenes. Everything I've read online is rather confusing. Reply

lubna says: December 06, 2016 at 8:46 pm

NEW VARIABLE (Z) = SQRT(X) + SQRT (Y) Reply

Haftom A. says: December 07, 2016 at 1:20 pm

Thank you so much!! That is really good explanation! I read many explanations about SVM but this one help me to understand the basics which I really needed it. keep it up!! Reply

asmae says: December 17, 2016 at 6:53 pm

hi please if you have an idiea about how it work for regression can you help me ? Reply

Frank says: January 06, 2017 at 6:26 pm

Thanks for the great article. There are even cool shirts for anyone who became SVM fan ;) http://www.redbubble.com/de/people/perceptron/works/24728522-support-vector-machines?grid_pos=2&p=t-shirt&style=mens Reply

bilashi says: January 10, 2017 at 10:26 pm

great explanation!! Thanks for posting it. Reply

arun says: January 19, 2017 at 3:22 am

I think this is |X| Reply

Priodyuti Pradhan says: January 21, 2017 at 4:44 pm

It is very nicely written and understandable. Thanks a lot... Reply

Walter says: February 06, 2017 at 9:34 pm

z=ax^2 + by^2 Reply

madhavi says: February 21, 2017 at 7:48 am

nice explanations with scenarios and margin values Reply

lishanth says: March 01, 2017 at 6:19 am

wow!!! excellent explanation.. only now i understood the concepts clearly thanks a lot.. Reply

anwar says: March 01, 2017 at 12:10 pm

(Z) = SQRT(X) + SQRT (Y) Reply

Kresla Matty says: March 20, 2017 at 1:33 pm

thanks, and well done for the good article Reply

Jonathan benitez says: April 16, 2017 at 7:36 pm

it's magnific your explanation Reply

Aishwarya Jangam says: April 20, 2017 at 1:40 pm

Great Explanation..Thanks.. Reply

Hams says: May 17, 2017 at 8:21 am

simple and refreshed the core concepts in just 5 mins! kudos Mr.Sunil Reply

Shashi says: May 17, 2017 at 12:13 pm

Best starters material for SVM, really appreciate the simple and comprehensive writing style. Expecting more such articles from you Reply

Ravindar says: May 20, 2017 at 3:39 pm

Z= square(x) Reply

Narasimha says: May 25, 2017 at 8:59 pm

Hey Sunil, Nice job of explaining it concisely and intuitively! Easy to follow and covers many aspects in a short space. Thanks! Reply

John Doe says: May 30, 2017 at 5:54 pm

Very well written - concise, clear, well-organized. Thank you. Reply

Camille says: June 11, 2017 at 6:08 pm

Oh sorry should have asked my question in english... The code I sent in my first comment is the code I took from this website and I cannot manage to make it work, I always got this message when I call the function "ValueError: zero-size array to reduction operation maximum which has no identity" What should I do? Thank you in advance Reply

Radhika says: June 14, 2017 at 2:38 pm

Excellent explanation..Can you please also tell what are the parameter values one should start with - like C, gamma ..Also, again a very basic question.. Can we say that lesser the % of support vectors (count of SVs/total records) better my model/richer my data is- assuming the datasize to be the same.. Waiting for more on parameter tuning..Really appreciate the knowledge shared.. Reply

Kirana says: June 15, 2017 at 10:30 am

Hi could you please explain why SVM perform well on small dataset? Reply

Chris says: June 20, 2017 at 5:11 pm

Another nice kernel for the problem stated in the article is the radial basis kernel. Reply

实用指南-在python中使用Scikit-learn进行数据预处理 - 数据分析网 says: June 22, 2017 at 5:18 am

[...] 资源：阅读这篇文章来理解SVM support vector machines。 [...] Reply

chiru says: June 23, 2017 at 2:03 pm

wow excellent Reply

Zhen Zhang says: June 26, 2017 at 9:26 am

very appreciating for explaining Reply

Andrey says: June 27, 2017 at 5:24 am

Nice tutorial. The new feature to separate data would be something like z = y - x^2 as most dots following the parabola will have lower z than stars. Reply

BanavaD says: July 04, 2017 at 2:05 pm

Very intuitive explanation. Thank you! Good to add SVM for Regression of Continuous variables. Reply

neha says: July 11, 2017 at 3:03 am

this is so simple method that anyone can get easily thnx for that but also explain the 4 senario of svm. Reply

Nirav Pingle says: July 20, 2017 at 12:14 pm

Great article for understanding of SVM: But, When and Why do we use the SVM algorithm can anyone make that help me understand because until this thing is clear there may not be use of this article. Thanks in advance. Reply

Mostafa says: August 02, 2017 at 2:01 pm

It is one of best explanation of machine learning technique that i have seen! and new variable: i think Z=|x| and new Axis are Z and Y Reply

venkat says: August 03, 2017 at 7:40 am

higher degree polynomial will separate the points in the problem, Reply

Tirthankar says: August 08, 2017 at 9:04 am

I guess the required feature is z = x^2 / y^2 For the red points, z will be close to 1 but for the blue points z values will be significantly more than 1 Reply

murtaza ali says: August 09, 2017 at 2:01 am

amazing article no doubt! It makes me clear all the concept and deep points regarding SVM. many thanks. Reply

katherine says: August 19, 2017 at 9:28 pm

The best explanation ever! Thank you! Reply

Rahul says: August 20, 2017 at 7:53 pm

z = x^2+y^2 Reply

Applied text classification on Email Spam Filtering [part 1] – Sarah Mestiri says: September 01, 2017 at 9:37 pm

[…] [1] Naive Bayes and Text Classification. [2]Naive Bayes by Example. [3] Andrew Ng explanation of Naive Bayes video 1 and video 2 [4] Please explain SVM like I am 5 years old. [5] Understanding Support Vector Machines from examples. […] Reply

roshan says: September 07, 2017 at 2:40 pm

new variable = ABS(Y) Reply

Robert says: September 13, 2017 at 7:38 am

Man, I was looking for definition of SVM for my diploma, but I got interested in explanation part of this article. Keep up good work! Reply

Aman Goel says: September 15, 2017 at 6:59 pm

we can use 'poly' kernel with degree=2 Reply

Nethra Kulkarni says: September 21, 2017 at 6:59 pm

Hi.. Very well written, great article !:). Thanks so much share knowledge on SVM. Reply

S Sen Sharma says: September 23, 2017 at 8:34 pm

z=y-x^2 Reply

Dalon says: October 02, 2017 at 7:13 am

Wonderful, easy to understand explanation. Reply

Eka A says: October 11, 2017 at 12:56 pm

Thanks a lot for your explanations, they were really helpful and easy to understand Reply

Kevin Mekulu says: October 19, 2017 at 12:55 pm

It would be a parabola z = a*x^2 + b*y^2 + c*x + d*y + e Reply

Yadi says: October 25, 2017 at 1:12 pm

Very good explanation, helpful Reply

shefali says: November 03, 2017 at 1:31 pm

valuable explanation!! Reply

vami says: November 15, 2017 at 8:39 pm

Very helpfull Reply

Shivam Misra says: January 12, 2018 at 4:46 pm

|X| Reply

panimalar says: January 18, 2018 at 12:07 am

thank u sir ,it is easy to understand Reply

John says: February 09, 2018 at 5:47 am

z = x^2 + y Reply

Pavan Kumar says: March 07, 2018 at 3:28 pm

It may be z=x^2+y Reply

Jose says: March 10, 2018 at 12:46 am

y=x^2 Reply

anoop says: March 21, 2018 at 10:58 am

z=ax^2 + by^2 + c Reply

quandapro says: March 28, 2018 at 7:56 am

Nice. new variable is z = abs(x). Then replace x coordinates with z coordinates Reply

Athul says: March 31, 2018 at 3:13 pm

z = |x| Reply

Deyire Yusuf Umar says: May 02, 2018 at 5:16 am

PARABOLA Reply

Jason says: May 02, 2018 at 11:47 am

I think the boundaryf between two type of snapshot could be a curve (of a part of circle). So I prefer kernel Z=sqrt(X^2+(Y-c)^2) Reply

ILA says: May 08, 2018 at 8:48 am

Thanks a lot. I like how you define a problem and then solve it. It makes things clear. Reply

Prachi says: May 26, 2018 at 11:47 am

z=x-y^2 Reply

Leave a Reply Your email address will not be published. Required fields are marked *