Analytics Vidhya
Learn everything about analytics

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

SUNIL RAY, SEPTEMBER 13, 2017 LOGIN TO BOOKMARK THIS ARTICLE

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

Introduction

Mastering machine learning algorithms isn't a myth at all. Most of the 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, sword, blades, bow, dagger 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 them, it can be much more stronger and powerful in building models.

By now, I hope you've now mastered Random Forest, Naive Bayes Algorithm and Ensemble Modeling. If not, I'd suggest you to take out 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.

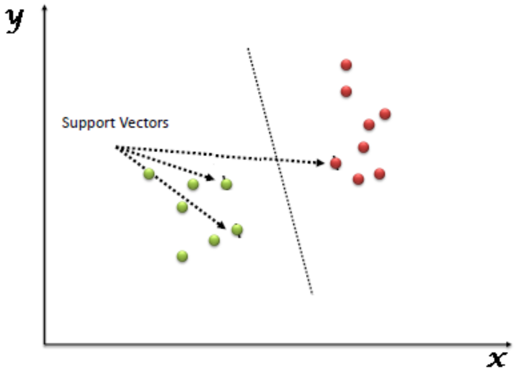
If you're a beginner looking to start your data science journey, you've come to the right place! Check out the below comprehensive courses, curated by industry experts, that we have created just for you:

- Introduction to Data Science
- Certified Program: Data Science for Beginners (with Interviews)

Table of Contents

What is Support Vector Machine?

"Support Vector Machine" (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is 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 differentiate the two classes very well (look at the below snapshot).



Support Vectors are simply the co-ordinates of individual observation. Support Vector Machine is a frontier which best segregates the two classes (hyper-plane/ line).

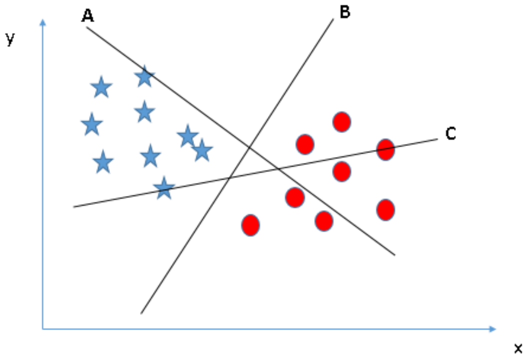
You can look at support vector machines and a few examples of its working here.

How does it work?

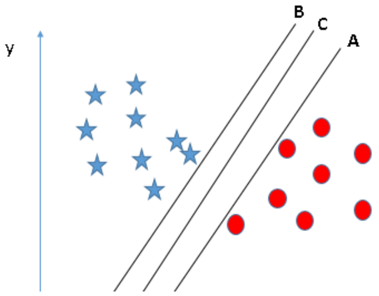
Above, we got accustomed to the process of segregating the two classes with a hyper-plane. Now the burning question is "How can we identify the right hyper-plane?". Don't worry, it's not as hard as you think!

Let's understand:

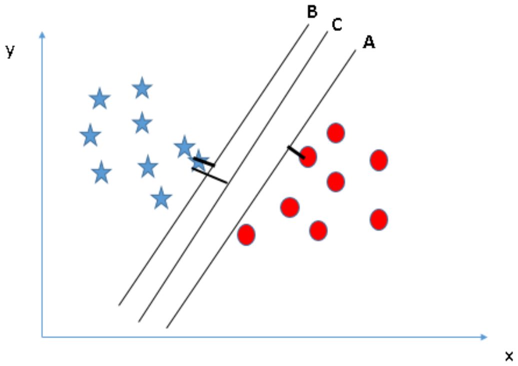
- **Identify the right hyper-plane (Scenario-1):** Here, we have three hyper-planes (A, B and C). Now, identify the right hyper-plane to classify star and circle.



  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?

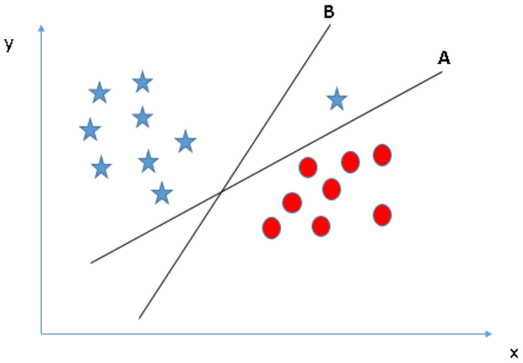


Here, maximizing the distances between nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as **Margin**. Let's look at the below snapshot:



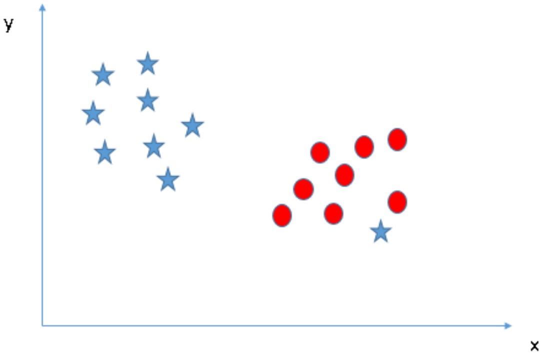
Above, you can see that the margin for hyper-plane C is high as compared to both A and B. Hence, we name the right hyper-plane as C. Another lightning reason for selecting the hyper-plane with higher margin is robustness. If we select a hyper-plane having low margin then there is high chance of miss-classification.

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

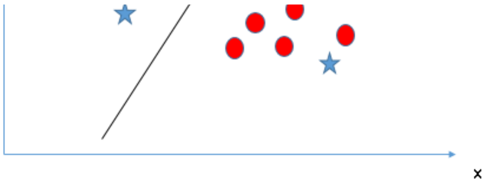


Some of you may have selected the hyper-plane **B** as it has higher margin compared to **A.** But, here is the catch, SVM selects the hyper-plane which classifies the classes accurately prior to maximizing margin. Here, hyper-plane B has a classification error and A has classified all correctly. Therefore,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

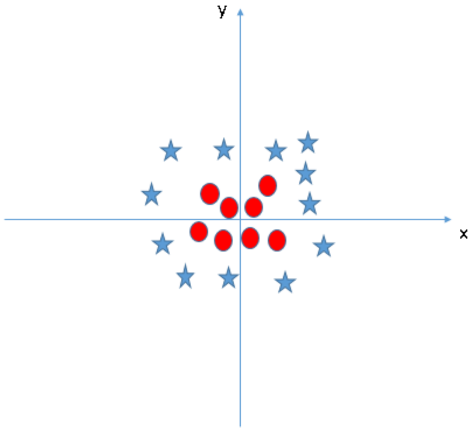
- **Can we classify two classes (Scenario-4)?:** Below, I am unable to segregate the two classes using a straight line, as one of star lies in the territory of other(circle) class as an outlier.



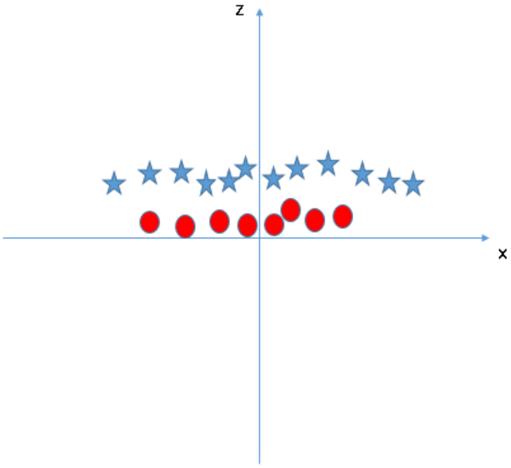
As I have already mentioned, one star at other end is like an outlier for star class. SVM has a feature to ignore outliers and find the hyper-plane that has maximum margin. Hence, we can say, SVM is robust to outliers.



- **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:

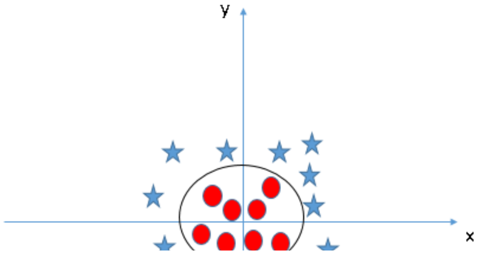


In above plot, points to consider are:

- 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 SVM, 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, SVM has a technique called the **kernel trick**. These are functions which takes low dimensional input space and transform it to a higher dimensional space i.e. it converts not separable problem to separable problem, these functions are called kernels. It is mostly useful in non-linear separation problem. Simply put, it does some extremely complex data transformations, then find 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 algorithm in a data science challenge.

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 scikit-learn library and follow the same structure (Import library, object creation, fitting model and prediction). Let’s look at the below code:

```
#Import Library
from sklearn import svm
#Assumed you have, X (predictor) and Y (target) for training data set and x_test(predictor) of test_dataset
# Create SVM classification object
model = svm.svc(kernel='linear', c=1, gamma=1)
# there is various option associated with it, like changing kernel, gamma and C value. Will discuss more # about it in next section.Train the model using the training sets and check score
model.fit(X, y)
model.score(X, y)
#Predict Output
predicted= model.predict(x_test)
```

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=100)

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

How to tune Parameters of SVM?

Tuning parameters value for machine learning algorithms effectively improves the 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”.

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.

Example: Have linear kernel

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

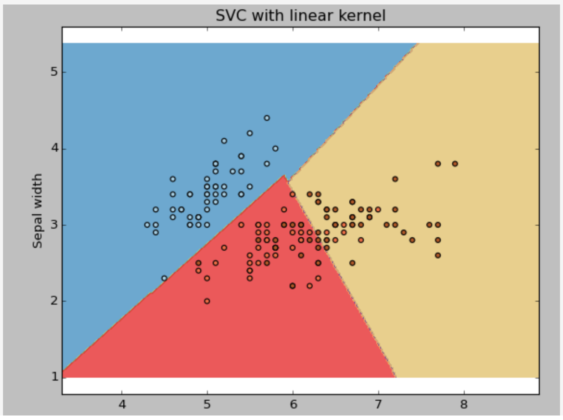
```
# import some data to play with
iris = datasets.load_iris()
X = iris.data[:, :2] # we only take the first two features. We could
 # avoid this ugly slicing by using a two-dim dataset
y = iris.target
```

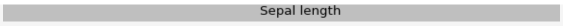
```
# we create an instance of SVM and fit out data. We do not scale our
# data since we want to plot the support vectors
C = 1.0 # SVM regularization parameter
svc = svm.SVC(kernel='linear', C=1,gamma=0).fit(X, y)
```

```
# 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()
```

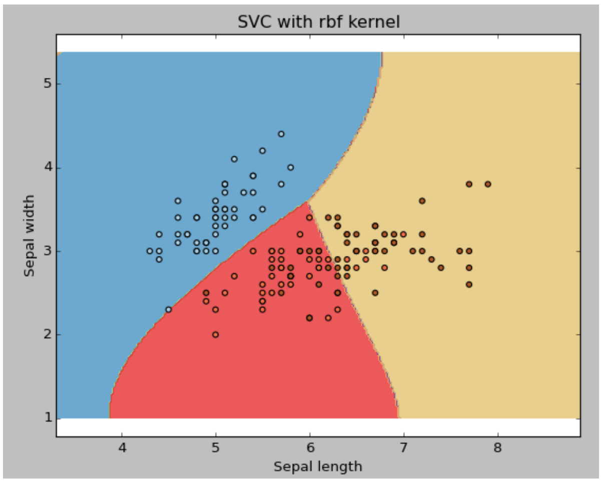




Example: Have rbf kernel

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)
```

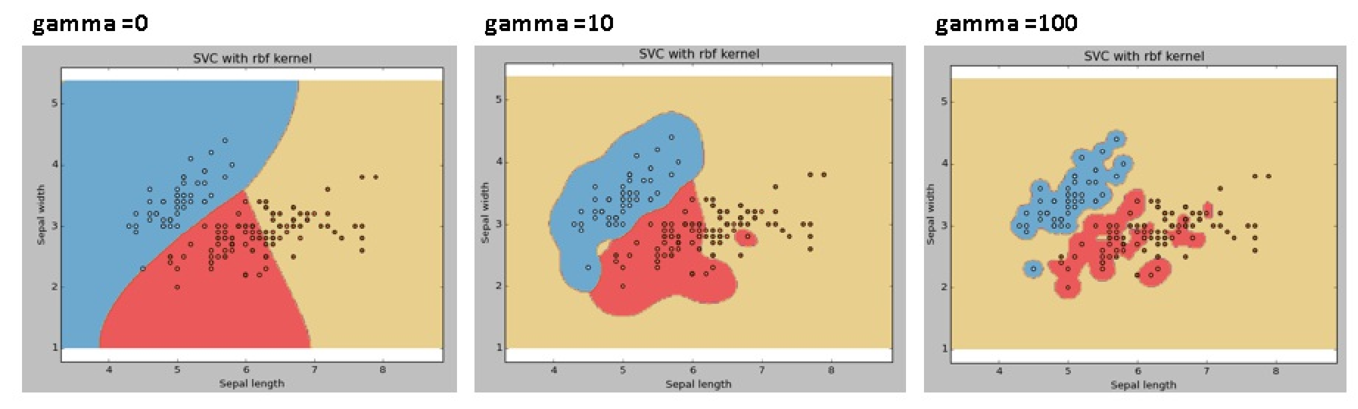


I would suggest you to go for linear kernel if you have large number of features (>1000) because it is more likely that the data is linearly separable in high dimensional space. Also, you can RBF but do not forget to cross validate for its parameters as to avoid over-fitting.

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.

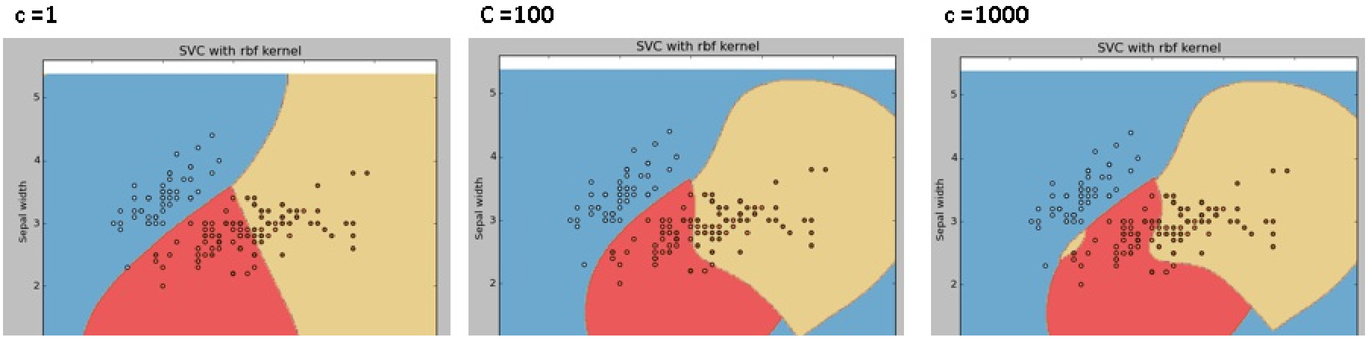
```
svc = svm.SVC(kernel='rbf', C=1,gamma=0).fit(X, y)
```


gamma =0

gamma =10

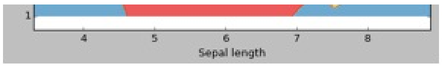
gamma =100

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


c =1

C =100

c =1000



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.

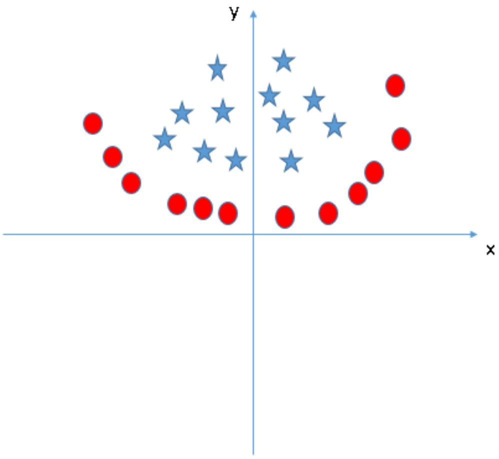
Pros and Cons associated with SVM

- **Pros:**
  - It works really well with clear margin of separation
  - It is effective in high dimensional spaces.
  - It is effective in cases where 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 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, 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 to use SVM and analyse 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.

If you like what you just read & want to continue your analytics learning, subscribe to our emails, follow us on twitter or like our facebook page.

Share this:

TAGS : E1071, KERNEL, MACHINE LEARNING, SCIKIT-LEARN, SVM

NEXT ARTICLE

Comparative Stock Market Analysis in R using Quandl & tidyverse – Part I

•••

PREVIOUS ARTICLE

Python vs. R (vs. SAS) – which tool should I learn?

Sunil Ray

I am a Business Analytics and Intelligence professional with deep experience in the Indian Insurance industry. I have worked for various multi-national Insurance companies in last 7 years.

This article is quite old and you might not get a prompt response from the author. We request you to post this comment on Analytics Vidhya's **Discussion portal** to get your queries resolved

93 COMMENTS

NISHANT

October 7, 2015 at 4:26 am

Reply

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

ASHISH

October 7, 2015 at 7:00 am

Reply

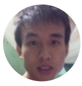
NEW VARIABLE (Z) = SQRT(X) + SQRT (Y)

RISHABH

June 15, 2016 at 11:22 am

Reply

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



DAM VAN TAI

October 6, 2016 at 9:00 am

Reply

i think x coodinates must increase after sqrt

MAHMOOD A. SHEIKH

October 7, 2015 at 9:16 am

Reply

Kernel

MAHMOOD A. SHEIKH

October 7, 2015 at 9:19 am

Reply

I mean kernel will add the new feature automatically.

SANJAY

October 7, 2015 at 3:12 pm

Reply

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

FRANKSAUVAGE

October 12, 2015 at 10:06 am

Reply

Thanks a lot for this great hands-on article!

HARSHA

November 8, 2015 at 4:42 am

Reply

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.

AMAN SRIVASTAVA

November 26, 2015 at 1:59 pm

Reply

kernel

EPHRAIM ADMASSUS

February 14, 2016 at 2:16 pm

Reply

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

JANPREET SINGH

March 4, 2016 at 12:58 pm

Reply

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

KRISHNA KALAPARTI

Reply

KRISHNA KALAPARTI Reply

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?

SHIKHA Reply

May 28, 2016 at 8:33 pm

great explanation 🙂 I think new variable Z should be x^2 + y.

VEERAMANI NATARAJAN Reply

June 3, 2016 at 6:23 am

Nice Articlel

CARLOS Reply

June 14, 2016 at 3:18 pm

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

K.KRITHIGA LAKSHMI Reply

June 15, 2016 at 12:07 pm

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

PFCOHEN Reply

June 19, 2016 at 3:52 pm

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

YC Reply

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))

IRESH Reply

July 14, 2016 at 10:32 pm

z = (x^2 – y)
z > 0, red circles

LI JENG HUANG Reply

August 5, 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.

HARI
August 18, 2016 at 10:02 am

Reply

Great article.. I think the below formula would give a new variable that help to separate the points in hyper plane

z = y – |x|

MADHVI
August 23, 2016 at 5:42 am

Reply

THANKS FOR EASY EXPLANATION

RAGHU
August 31, 2016 at 2:35 pm

Reply

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

YAMANI
September 22, 2016 at 6:25 pm

Reply

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

HARSHEL JAIN
September 23, 2016 at 6:49 am

Reply

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

ASMAE
December 17, 2016 at 6:53 pm

Reply

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

DIANA
October 4, 2016 at 2:57 pm

Reply

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.

MANJUNATH GS
October 27, 2016 at 7:08 pm

Reply

please give us the answer

DIPTESH
October 28, 2016 at 3:48 pm

Reply

This is very useful for understanding easily

This is very useful for understanding easily.

DAN

November 14, 2016 at 6:41 pm

Reply

just substitude x with |x|

MIN

November 23, 2016 at 8:13 am

Reply

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!

ANU

November 29, 2016 at 7:53 pm

Reply

z

RENNY VARGHESE

December 3, 2016 at 7:10 pm

Reply

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.

LUBNA

December 6, 2016 at 8:46 pm

Reply

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

HAFTOM A.

December 7, 2016 at 1:20 pm

Reply

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!!

FRANK

January 6, 2017 at 6:26 pm

Reply

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

BILASHI

January 10, 2017 at 10:26 pm

Reply

great explanation!! Thanks for posting it.

ARUN

January 19, 2017 at 3:22 am

Reply

I think this is |X|

PRIODYUTI PRADHAN

January 21, 2017 at 4:44 pm

Reply

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

WALTER

February 6, 2017 at 9:34 pm

Reply

z=ax^2 + by^2

MADHAVI

February 21, 2017 at 7:48 am

Reply

nice explanations with scenarios and margin values

LISHANTH

March 1, 2017 at 6:19 am

Reply

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

ANWAR

March 1, 2017 at 12:10 pm

Reply

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

KRESLA MATTY

March 20, 2017 at 1:33 pm

Reply

thanks, and well done for the good article

JONATHAN BENITEZ

April 16, 2017 at 7:36 pm

Reply

it’s magnific your explanation

AISHWARYA JANGAM

April 20, 2017 at 1:40 pm

Reply

Great Explanation..Thanks..

HAMS

May 17, 2017 at 8:21 am

Reply

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

SHASHI

May 17, 2017 at 12:13 pm

Reply

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

RAVINDAR

May 20, 2017 at 3:39 pm

Reply

Z= square(x)

NARASIMHA

May 25, 2017 at 8:59 pm

Reply

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

JOHN DOE

May 30, 2017 at 5:54 pm

Reply

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

RADHIKA

June 14, 2017 at 2:38 pm

Reply

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..

KIRANA

June 15, 2017 at 10:30 am

Reply

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

CHRIS

June 20, 2017 at 5:11 pm

Reply

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

- **□□□□-□PYTHON□□□SCIKIT-LEARN□□□□□□□ - □□□□□**

  June 22, 2017 at 5:18 am

[…] □□□□□□□□□□□□□□SVM support vector machines□ […]

CHIRU

June 23, 2017 at 2:03 pm

Reply

wow excellent

ZHEN ZHANG

June 26, 2017 at 9:26 am

Reply

very appreciating for explaining

ANDREY

Reply

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.

BANAVAD

Reply

July 4, 2017 at 2:05 pm

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

NEHA

Reply

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.

NIRAV PINGLE

Reply

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.

MOSTAFA

Reply

August 2, 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

VENKAT

Reply

August 3, 2017 at 7:40 am

higher degree polynomial will separate the points in the problem,

TIRTHANKAR

Reply

August 8, 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

MURTAZA ALI

Reply

August 9, 2017 at 2:01 am

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

KATHERINE

Reply

August 19, 2017 at 9:28 pm

The best explanation ever! Thank you!

RAHUL Reply

August 20, 2017 at 7:53 pm

z = x^2+y^2

- **APPLIED TEXT CLASSIFICATION ON EMAIL SPAM FILTERING [PART 1] – SARAH MESTIRI**

September 1, 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. […]

ROSHAN Reply

September 7, 2017 at 2:40 pm

new variable = ABS(Y)

ROBERT Reply

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!

AMAN GOEL Reply

September 15, 2017 at 6:59 pm

we can use ‘poly’ kernel with degree=2

NETHRA KULKARNI Reply

September 21, 2017 at 6:59 pm

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

S SEN SHARMA Reply

September 23, 2017 at 8:34 pm

z=y-x^2

DALON Reply

October 2, 2017 at 7:13 am

Wonderful, easy to understand explanation.

EKA A Reply

October 11, 2017 at 12:56 pm

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

KEVIN MEKULU Reply

October 19, 2017 at 12:55 pm

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

YADI
October 25, 2017 at 1:12 pm

Reply

Very good explanation, helpful

SHEFALI
November 3, 2017 at 1:31 pm

Reply

valuable explanation!!

VAMI
November 15, 2017 at 8:39 pm

Reply

Very helpfull

SHIVAM MISRA
January 12, 2018 at 4:46 pm

Reply

|X|

PANIMALAR
January 18, 2018 at 12:07 am

Reply

thank u sir ,it is easy to understand

JOHN
February 9, 2018 at 5:47 am

Reply

z = x^2 + y

PAVAN KUMAR
March 7, 2018 at 3:28 pm

Reply

It may be z=x^2+y

JOSE
March 10, 2018 at 12:46 am

Reply

y=x^2

ANOOP
March 21, 2018 at 10:58 am

Reply

z=ax^2 + by^2 + c

QUANDAPRO
March 28, 2018 at 7:56 am

Reply

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

ATHUL

March 31, 2018 at 3:13 pm

Reply

z = |x|

DEYIRE YUSUF UMAR

May 2, 2018 at 5:16 am

Reply

PARABOLA

JASON

May 2, 2018 at 11:47 am

Reply

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)

ILA

May 8, 2018 at 8:48 am

Reply

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

PRACHI

May 26, 2018 at 11:47 am

Reply

z=x-y^2

POPULAR POSTS

- 24 Ultimate Data Science Projects To Boost Your Knowledge and Skills (& can be accessed freely)
- Essentials of Machine Learning Algorithms (with Python and R Codes)
- 7 Types of Regression Techniques you should know!
- Understanding Support Vector Machine algorithm from examples (along with code)
- A Complete Tutorial to Learn Data Science with Python from Scratch
- Introduction to k-Nearest Neighbors: Simplified (with implementation in Python)
- A Simple Introduction to ANOVA (with applications in Excel)
- Stock Prices Prediction Using Machine Learning and Deep Learning Techniques (with Python codes)

RECENT POSTS

Using the Power of Deep Learning for Cyber Security (Part 2) – Must-Read for All Data Scientists

MAY 23, 2019

Data Science Project: Scraping YouTube Data using Python and Selenium to Classify Videos

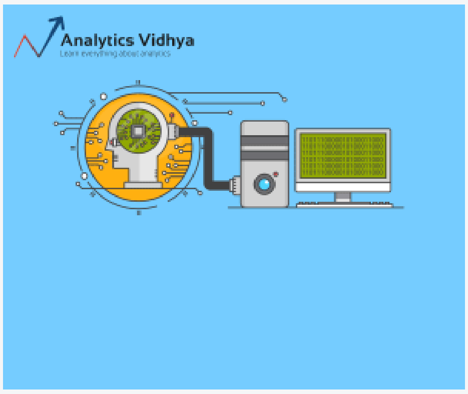
MAY 20, 2019

Statistics for Data Science: Introduction to t-test and its Different Types (with Implementation in R)

MAY 16, 2019

10 Useful Data Analysis Expressions (DAX) Functions for Power BI Beginners

MAY 15, 2019



Analytics Vidhya
Learn everything about analytics

DATA SCIENTISTS