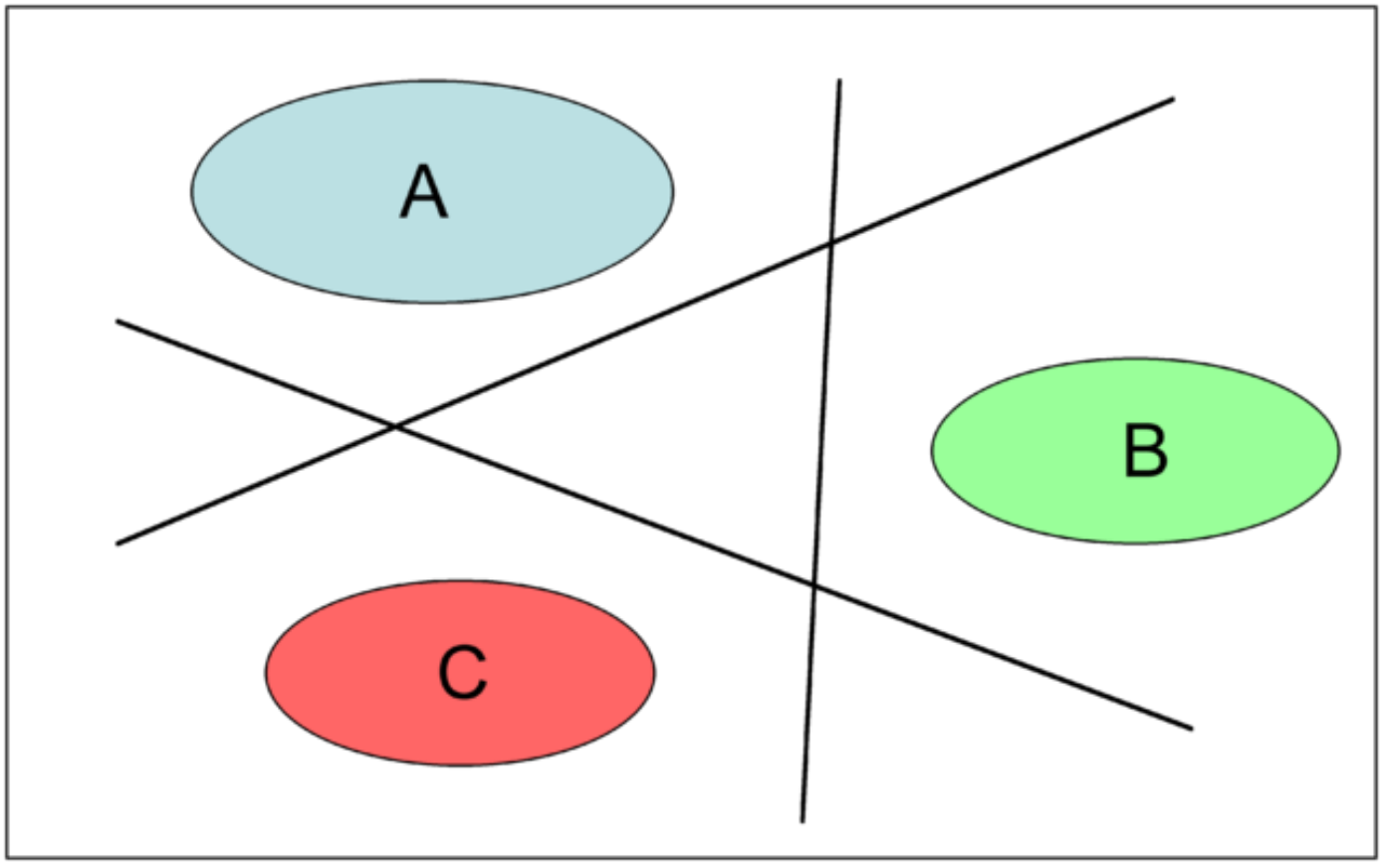
عنوان: **Accord.NET #3**
نویسنده: محسن نجف زاده
تاریخ: ۱۳۹۴/۰۶/۱۰ ۲۲:۵۰
آدرس: www.dotnettips.info
گروه‌ها: Accord.NET, Machine Learning, Artificial Intelligence

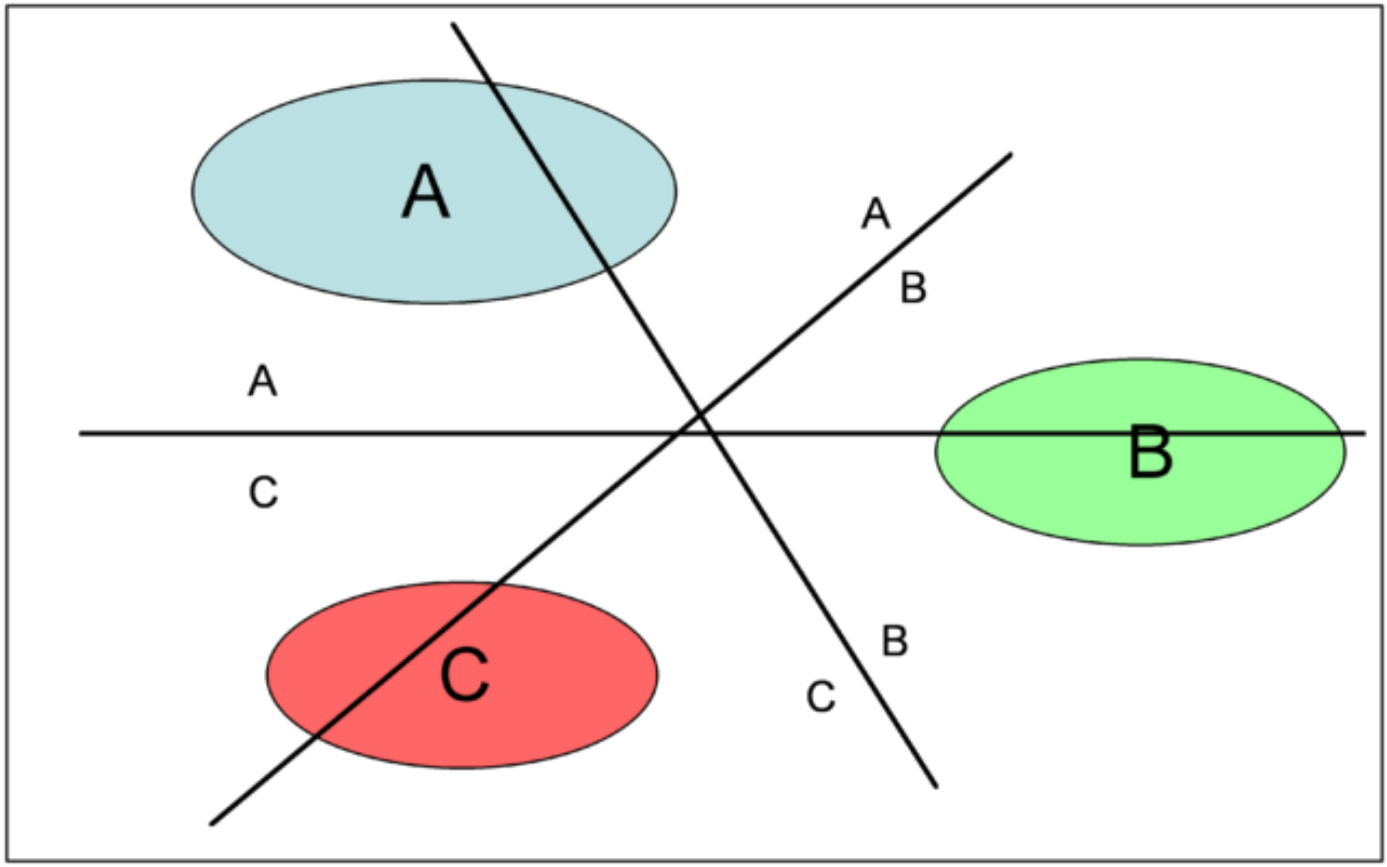
در مطلب قبل یک مثال مفهومی درباره کاربرد SVM بیان شد و دیدیم که این الگوریتم، یک روش دودویی است و عموما برای زمانی به کار می‌رود که مجموعه داده ما شامل دو کلاس باشد.
اگر بخواهیم نوع چهار میوه (سیب، گلابی، موز، پرتقال) را که از خط سورتینگ عبور می‌کنند، تشخیص دهیم و یا اینکه بخواهیم تشخیص اعداد دست نویس را داشته باشیم و یا اینکه حتی مطالب این وب سایت را که شامل چندین برچسب هستند، طبقه بندی کنیم، آیا در این تشخیص‌ها SVM به ما کمک می‌کند؟ پاسخ مثبت است.
در فضای نام یادگیری ماشین Accord.NET دو تابع خوب SupportVectorLearning **Multi class** و Multi **label** برای این گونه از مسائل تعبیه شده است. زمانیکه مسئله‌ی ما شامل مجموعه داده‌هایی بود که در چندین کلاس دسته بندی می‌شوند (مانند دسته بندی میوه، اعداد و ...) روش Multiclass و زمانیکه عناصر مجموعه داده ما به طور جداگانه شامل چندین برچسب باشند (مانند دسته بندی مطالب با داشتن چندین تگ، ...) روش Multilabel ابزار مفیدی خواهند بود. (+)

با توجه به **دودویی بودن ماشین بردار پشتیبان** ، دو استراتژی برای به کارگیری این الگوریتم برای دسته بندی‌های **چند کلاسه** وجود دارد:
روش یک در مقابل همه - One-against-all : در این روش عملا همان روش دودویی SVM را برای هر یک از کلاس‌ها به صورت جداگانه بررسی می‌کنیم. مثلا برای تشخیص میوه، یک بار دو کلاس سیب و غیر سیب (مابقی) بررسی می‌شوند و به همین ترتیب برای سایر کلاس‌ها و در مجموع صفحات ابرصفحه جدا کننده بین هر کلاس در مقابل سایر کلاس‌ها ایجاد می‌شود.



روش یک در مقابل یک - One-against-one (*) : در این روش هر کلاس، با هر یک از کلاس‌های دیگر به صورت تک تک بررسی

می‌شود و صفحات ابرصفحه جدا کننده مابین هر جفت کلاس متفاوت ایجاد می‌شود. (بیشتر در +)



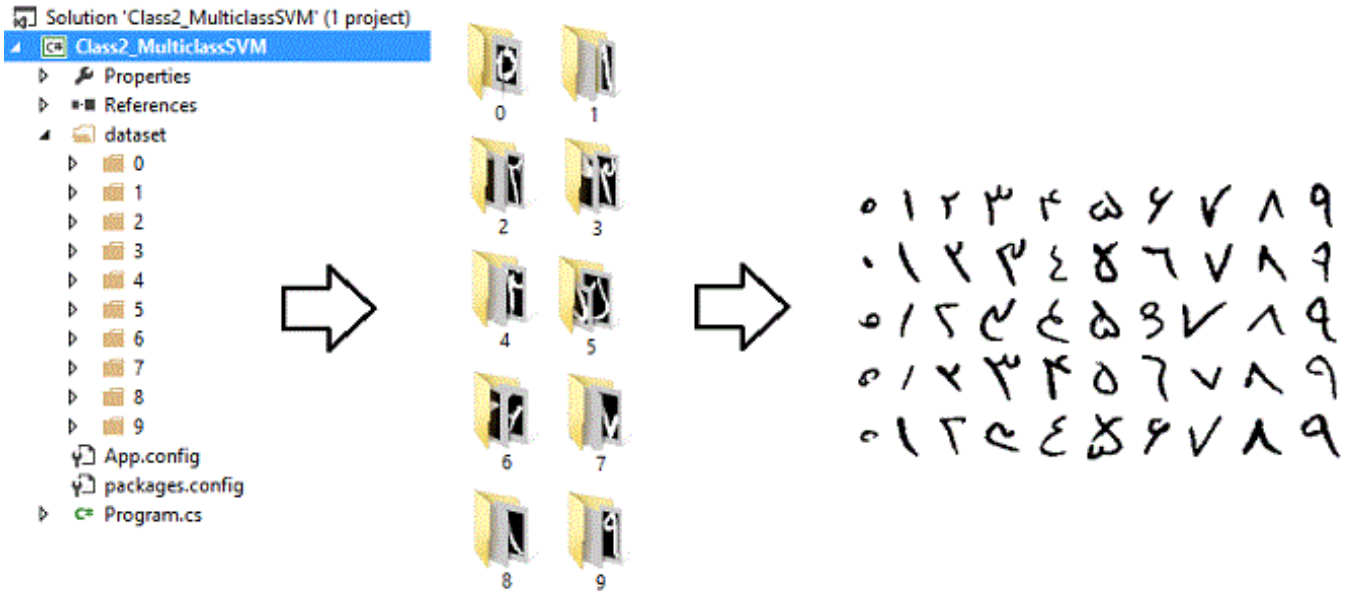
*روش "یک در مقابل یک" یا One-against-one اساس کار دسته بندی MulticlassSupportVectorMachine در فضای نام Accord.MachineLearning است.

یک مثال کاربردی : هدف در این مثال دسته بندی اعداد فارسی به کمک MulticlassSupportVectorMachine است.

به معرفی ابزار کار مورد نیاز می‌پردازیم.

1.مجموعه ارقام دستنویس هدی: مجموعه ارقام دستنویس هدی که اولین مجموعه‌ی بزرگ ارقام دستنویس فارسی است، مشتمل بر ۱۰۲۳۵۳ نمونه دستنوشته سیاه سفید است. این مجموعه طی انجام یک پروژه‌ی کارشناسی ارشد درباره بازشناسی فرم‌های دستنویس تهیه شده است. داده‌های این مجموعه از حدود ۱۲۰۰۰ فرم ثبت نام آزمون سراسری کارشناسی ارشد سال ۱۳۸۴ و آزمون کاردانی پیوسته‌ی دانشگاه جامع علمی کاربردی سال ۱۳۸۳ استخراج شده است. (اطلاعات بیشتر درباره مجموعه ارقام دستنویس هدی) .

تعداد 1000 نمونه (از هر عدد 100 نمونه) از این مجموعه داده، با فرمت bmp در این پروژه مورد استفاده قرار گرفته که به همراه پروژه در انتهای این مطلب قابل دریافت است.



2.استخراج ویژگی (Feature extraction) : در بازشناسی الگو و مفاهیم کلاس بندی، یکی از مهمترین گام‌ها، استخراج ویژگی است. ما موظف هستیم تا اطلاعات مناسبی را به عنوان ورودی برای دسته بندی‌مان معرفی نماییم. روش‌های مختلفی برای استخراج ویژگی وجود دارند. ویژگی‌ها به دو دسته‌ی کلی ویژگی‌های ظاهری (Appearance) و ویژگی‌های توصیف کننده (Descriptive) قابل تقسیم هستند. در تشخیص حروف و اعداد، ویژگی‌هایی مانند شدت نور نقاط (Intensity)، تعداد حلقه بسته، تعداد خطوط راست، تعداد دندانه، تعداد نقطه (برای حروف) و ... در دسته‌ی اول و ویژگی‌هایی مانند شیب خطوط، گرادیان، میزان افت یا شدت نور یک ناحیه، HOG و ... در دسته دوم قرار می‌گیرند. در این مطلب ما تنها از روش شدت نور نقاط برای استخراج ویژگی‌هایمان استفاده کرده‌ایم.

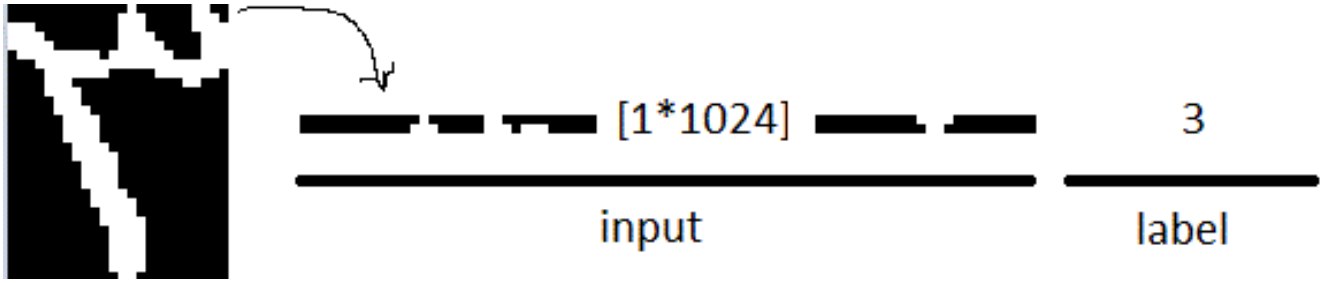
کد زیر با دریافت یک فایل Bitmap، ابتدا ابعاد را به اندازه 32*32 تغییر می‌دهد و سپس آن‌را به صورت یک بردار 1*1024 را بر می‌گرداند:

```
//تابع استخراج ویژگی
        private static double[] FeatureExtractor(Bitmap bitmap)
        {
            bitmap = BitmapResizer(bitmap, 32, 32);

            double[] features = new double[32 * 32];
            for (int i = 0; i < 32; i++)
                for (int j = 0; j < 32; j++)
                    features[i * 32 + j] = (bitmap.GetPixel(j, i).R == 255) ? 0 : 1;

            return features;
        }

        //تابع تغییر دهنده ابعاد عکس
        private static Bitmap BitmapResizer(Bitmap bitmap, int width, int height)
        {
            var newbitmap = new Bitmap(width, height);
            using (Graphics g = Graphics.FromImage((Image)newbitmap))
            {
                g.InterpolationMode = System.Drawing.Drawing2D.InterpolationMode.HighQualityBicubic;
                g.DrawImage(bitmap, 0, 0, width, height);
            }
            return newbitmap;
        }
```

3.ایجاد ورودی‌ها و برچسب : در این مرحله ما باید ورودی‌های دسته بندی SVM را که عملا آرایه‌ای براساس تعداد نمونه‌های مجموعه آموزش (train) است، ایجاد نماییم.

ورودی‌ها (inputs) = با توجه به اینکه تعداد نمونه‌ها 50 مورد از هر عدد (مجموعا 500 نمونه) تعیین شده است و تعداد

ویژگی‌های هر نمونه یک بردار با طول 1024 است، ابعاد ماتریس ورودی مان [1024][500] می‌شود.
برچسب‌ها (labels) = تعداد برچسب مسلما به تعداد نمونه هایمان یعنی 500 مورد می‌باشد و مقادیر آن قاعدتا عدد متناظر آن تصویر است.



برای این کار از قطعه کد زیر استفاده می‌کنیم :

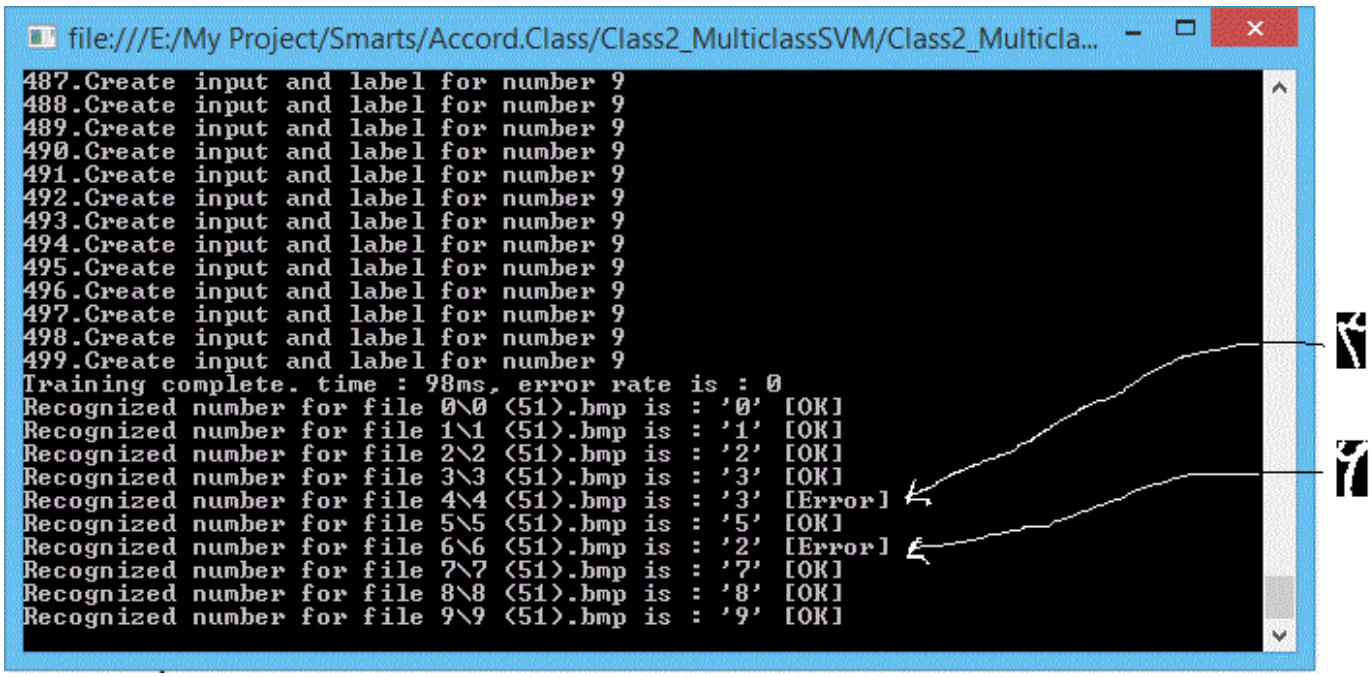
```
var path = new DirectoryInfo(Directory.GetCurrentDirectory()).Parent.Parent.FullName + @"\dataset\";

            // ایجاد ورودی و برچسب
            int trainingCount = 50;
            double[][] inputs = new double[trainingCount * 10][];
            int[] labels = new int[trainingCount * 10];

            var index = 0;
            var filename = "";
            Bitmap bitmap;
            double[] feature;

            for (int number = 0; number < 10; number++)
            {
                for (int j = 0; j < trainingCount; j++)
                {
                    index = (number * trainingCount) + j;
                    filename = string.Format(@"{0}\{0} ({1}).bmp", number, j + 1);
                    bitmap = new Bitmap(path + filename);

                    feature = FeatureExtractor(bitmap);

                    inputs[index] = feature;
                    labels[index] = number;

                    Console.WriteLine(string.Format("{0}.Create input and label for number {1}", index,
number));
                }
            }
```

4.در نهایت به دسته بندمان که همان MulticlassSupportVectorLearning است، خواهیم رسید. همانطور که در مطلب قبل مطرح شد، پس از تعریف پارامترهای Classifier مان، باید آن را به یک الگوریتم یادگیری که در اینجا هم همان روش SMO است، نسبت دهیم.

```
private static double MachineLearning(IKernel kernel, double[][] inputs, int[] labels)
        {
            machine_svm = new MulticlassSupportVectorMachine(1024, kernel, 10);

            // معرفی دسته بندمان به الگوریتم یادگیری SMO
            MulticlassSupportVectorLearning ml = new MulticlassSupportVectorLearning(machine_svm,
inputs, labels)
            {
                Algorithm = (svm, classInputs, classOutputs, i, j) =>
                    new SequentialMinimalOptimization(svm, classInputs, classOutputs)
            };

            var error = ml.Run();
            return error;
        }
```

می‌توانیم پس از اینکه ماشین دسته بندمان آماده شد، برای آزمایش تعدادی از نمونه‌های جدید و دیده نشده (UnSeen) را که در نمونه‌های آموزشی وجود نداشتند، مورد ارزیابی قرار دهیم. برای این کار اعداد 0 تا 9 از مجموعه داده مان را در نظر می‌گیریم و به وسیله کد زیر نتایج را مشاهده می‌کنیم :

```
// بررسی یک دسته از ورودی‌ها
            index = 51;
            for (int number = 0; number < 10; number++)
            {
                filename = string.Format(@"{0}\{0} ({1}).bmp", number, index);
                bitmap = new Bitmap(path + filename);

                feature = FeatureExtractor(bitmap);

                double[] responses;
                int recognizednumber = machine_svm.Compute(feature, out responses);

                Console.WriteLine
                (
                    String.Format
                    (
                        "Recognized number for file {0} is : '{1}' [{2}]",
                        filename,
                        recognizednumber,
                        (recognizednumber == number ? "OK" : "Error")
                    )
                );
                if (!machine_svm.IsProbabilistic)
                {
                    // Normalize responses
                    double max = responses.Max();
                    double min = responses.Min();

                    responses = Accord.Math.Tools.Scale(min, max, 0, 1, responses);
                    //int minIndex = Array.IndexOf(responses, 0);
                }
            }
```



مشاهده می‌شود که تنها بازشناسی تصاویر اعداد 4 و 6، به اشتباه انجام شده است که جای نگرانی نیست و می‌توان با افزایش تعداد نمونه‌های آموزشی و یا تغییرات پارامترها از جمله نوع کرنل و یا الگوریتم آموزنده این خطاها را نیز بر طرف کرد.

همانطور که دیدیم SVM گزینه‌ی بسیار مناسبی برای طبقه بندی خیلی از مسائل دو کلاسه و یا حتی چند کلاسه است. اما آکورد

دات نت Classifierهای خوب دیگری (مانند Naive Bayes و Decision Trees یا درخت تصمیم و ...) را نیز در چارچوب خود جای داده که در مطالب آینده معرفی خواهند شد.

دریافت پروژه

نظرات خوانندگان

نویسنده: زواری
تاریخ: ۱۳۹۴/۰۶/۱۱ ۲۱:۲۲

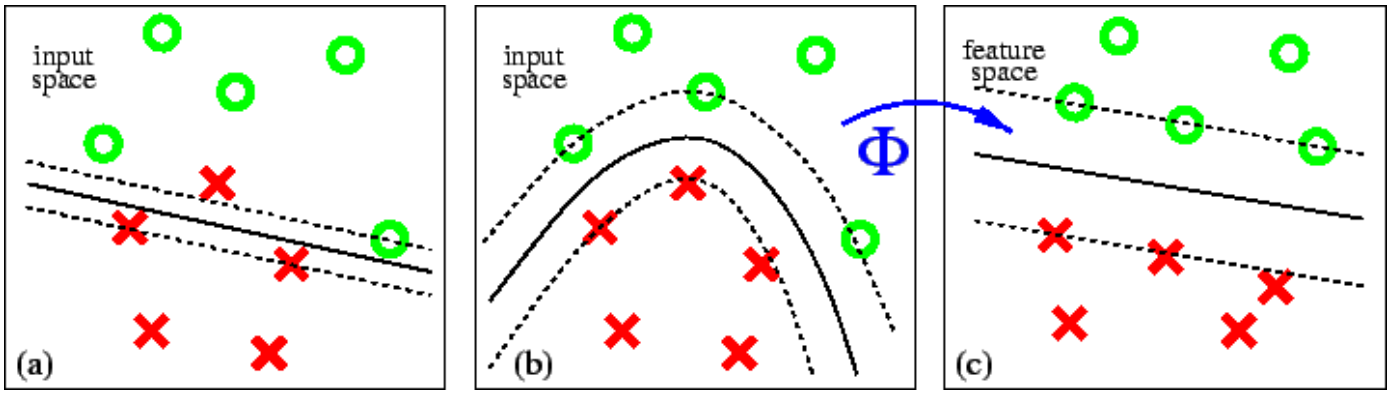
ممنون از مطالب مفیدتون.
اگه ممکنه در مورد کرنل و الگوریتم یادگیری کمی بیشتر توضیح بدید.
برای تغییر اندازه و استخراج ویژگی هم فکر میکنم بشه از توابع زیر استفاده کرد که شاید کاراتر باشن:

```
AForge.Imaging.Filters.ResizeBicubic
AForge.Imaging.Filters.ResizeBilinear
AForge.Imaging.Filters.ResizeNearestNeighbor

Accord.Imaging.Converters.ArrayToImage
Accord.Imaging.Converters.ImageToArray
Accord.Imaging.Converters.ImageToMatrix
Accord.Imaging.Converters.MatrixToImage
```

بازم تشکر و سپاس، درپناه حق موفق باشید.

نویسنده: محسن نجف زاده
تاریخ: ۱۳۹۴/۰۶/۱۷ ۲۱:۲۹

- در مورد کرنل همانطور که در مطلب قبل هم صحبت شد، می‌توان گفت که Kernel عملا نگاشتی را بین خط تفکیک کننده نمونه‌های کلاس‌ها با ابرصفحه جداکننده برقرار می‌کند و با این شرایط می‌توان SVM را به نوعی غیر خطی در نظر گرفت. مثلا در تصویر زیر با پارامتر ، فضای اولیه داده‌های ما را به فضای ویژگی هایی نگاشت می‌شود که می‌توان با همان SVM خطی دسته بندی کرد (+)



- اگر منظورتان از الگوریتم یادگیری روش Sequential Minimal Optimization است می‌توان گفت SMO یکی از روش‌های متداول و سریع برای آموزش SVM به حساب می‌آید که عملا یک مسئله بهینه سازی است که به دنبال بهترین ضرایب همان کرنل است (+)

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j K(x_i, x_j) \alpha_i \alpha_j,$$

- درباره پیشنهادتون در خصوص استفاده از فضای نام Accord.Imaging هم ضمن تشکر، می‌توان گفت که فعلا قصد ورود به فضای نام ی از آکورد دات نت به جز Accord.MachineLearning نداریم ولی در آینده حتما معرفی و استفاده خواهند شد.