[Murphy, 2012] Murphy, K. P. (2012). *Machine Learning, A Probabilistic Perspective.* MIT Press.

[Provost and Fawcett, 2013] Provost, F. and Fawcett, T. (2013). *Data Science for Business.* O'Reilly.

[Sauer, 1972] Sauer, N. (1972). On the density of families of sets. *Journal of Combinatorial Theory, Series A*, (13):145–147.

[Shawe-Taylor et al., 1998] Shawe-Taylor, J., Bartlett, P. L., Williamson, R. C., and Anthony, M. (1998). Structural risk minimization over data-dependent hierarchies. *IEEE Transactions on Information Theory*, 44(5):1926–1940.

[Shelah, 1972] Shelah, S. (1972). A combinatorial problem; stability and order for models and theories in infinitary languages. *Pacific Journal of Mathematics*, (41):247–261.

[Valiant, 1984] Valiant, L. (1984). A theory of the learnable. *Communications of the ACM*, (27).

[Vapnik and Chervonenkis, 1989] Vapnik, N. and Chervonenkis, A. Y. (1989). The necessary and sufficient conditions for consistency of the method of empirical risk minimization. In *Yearbook of the Academy of Sciences of the USSR, on Recognition, Classification, and Forecasting*, volume 2, pages 207–249, Moscow. Naukia.

[Vapnik and Chervonenkis, 1971] Vapnik, V. and Chervonenkis, A. (1971). On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280.

[Vapnik, 1982] Vapnik, V. N. (1982). *Estimation of Dependences Based on Empirical Data*, volume Addendum 1. Springer-Verlag, New York.

[Vapnik, 1998] Vapnik, V. N. (1998). *Statistical Learning Theory.* Wiley.

[Vapnik, 2010] Vapnik, V. N. (2010). *The Nature of Statistical Learning Theory.* Springer-Verlag, 2nd edition.

[Zumel and Mount, 2014] Zumel, N. and Mount, J. (2014). *Practical Data Science with R.* Manning.

A Soft margins are not as good as hard-margins

Another issue in reading through the published results is how to relate hard-margin results (most of what is proven) to soft-margin classifiers (that which are implemented)

Soft margin results are derived from hard margin results in corollary on page 408 of section 10.2.1 of [Vapnik, 1998]. The content is essentially that a support vector machine with a given margin can be expected to have an error rate on new data (identically distributed as the training data) that is no more than the error Δ-margin error rate seen on training data plus a factor that is roughly $VCdimension/n$ where n is the number of training examples. From the definitions of section 10.2.1 it appears the Δ-margin error rate can be taken to be all errors made on training data including all data that falls too close to the decision surface as also being in error (this appears to be implicit in the definition of a Δ-margin separating hyperplane).

Let's explore a bit how to use such a bound. For our problem set up the very simple artificial one dimensional problem with x_i being the effective variable and y_i being the outcome. Our training data is $y_i = +1$ for x_i uniformly distributed in $(0, 1]$ and $y_i = -1$ for x_i uniformly distributed in $[-1, 0)$. We imagine we draw n samples independently from these distributions (picking from the $+$ and - classes with equal probability).

We are going to make the problem simple and assume we are only estimating a single scalar w and our model is $sign(w \cdot x)$. In this case we see for any $w > 0$ we have a perfect model (i.e. see no errors).

Let's look how the standard soft-margin SVM procedure would pick a w. The standard soft-margin SVM optimization problem for this set of concepts (left/right division of the number line at the origin) is (see [Vapnik, 1998] section 10.2.2):

$$\begin{aligned}
&\text{minimize :} \\
&\qquad w \cdot w/2 + C \sum_{i=1}^{n} \xi_i \\
&\text{subject to:} \\
&\qquad \xi_i >= 0 \\
&\qquad y_i(x_i \cdot w) >= 1 - \xi_i
\end{aligned}$$

Where C is a parameter usually picked before looking at the data, and often picked as $C = 1$.

The stated conditions insist $|x_i|w >= 1 - \xi_i$ and therefore $\xi_i >= 1 - |x_i|w$. We then expect the fraction of our data that needs a $\xi_i > 0$ corrections to be $min(1, 1/w)$ and the total mass of corrections to be $Cn \int_{x=0}^{min(1,1/w)} (1 - xw)dx$.

From the theory of SVMs we can bound the error seen on new data to be no more than Δ-margin error rate seen during training plus a $(1 + 1/margin^2)/n$ bound on excess generalization error. Our margin is of radius $1/w$ (measured from the center). So we expect our error bound to be no more than the empirical error rate of an appropriate hard-margin classifier that is considered wrong in the margin/moat area (or about a $1/w$ fraction of the time) plus the excess error bound $((1 + 1/margin^2)/n = (1 + w^2)/n)$. So our total error-bound is $1/w + (1 + w^2)/n$.

Let's see what value of C will minimize this error bound estimate.

```
# Python ipython
import sympy
x,w,C,n = sympy.symbols(['x','w','C','n'])

NormTerm = w*w/2
print('NormTerm',NormTerm)
EMarginTerm = C*n*sympy.integrate(1-x*w,(x,0,1/w))
print('EMarginTerm',EMarginTerm)
Objective = NormTerm + EMarginTerm
print('Objective',Objective)
Derivative = sympy.diff(Objective,w)
print('Derivative',Derivative)
OptimalWsGivenC = sympy.solve(Derivative,w)
print('OptimalWsGivenC',OptimalWsGivenC)
BestWGivenC = [ si for si in OptimalWsGivenC if sympy.N(si.subs([(C,1),(n,1)]))][0]
print('BestWGivenC',BestWGivenC)
ErrorBound = 1/w + (1 + w**2)/n
FnOfC = sympy.simplify(ErrorBound.subs(w,BestWGivenC))
print('FnOfC',FnOfC)
D2 = sympy.simplify(sympy.diff(FnOfC,C))
print('D2',D2)
CPicks = sympy.solve(D2,C)
print('CPicks',CPicks)
```

```
('NormTerm', w**2/2)
('EMarginTerm', C*n/(2*w))
('Objective', C*n/(2*w) + w**2/2)
('Derivative', -C*n/(2*w**2) + w)
('OptimalWsGivenC', [2**(2/3)*(C*n)**(1/3)/2,
   -2**(2/3)*(C*n)**(1/3)/4 - 2**(2/3)*sqrt(3)*I*(C*n)**(1/3)/4,
   -2**(2/3)*(C*n)**(1/3)/4 + 2**(2/3)*sqrt(3)*I*(C*n)**(1/3)/4])
('BestWGivenC', 2**(2/3)*(C*n)**(1/3)/2)
('FnOfC', 2**(1/3)*C/(2*(C*n)**(1/3)) + 2**(1/3)/(C*n)**(1/3) + 1/n)
('D2', 2**(1/3)*(C - 1)/(3*C*(C*n)**(1/3)))
('CPicks', [1])
```

What we did is look for minima of the optimization objectives (the SVM optimization problem and then the error-bound) by inspecting extreme of these functions by looking at zeros of derivatives. This let us confirm the common choice $C = 1$ is in fact an optimal pick for this toy problem.

Let's substitute $C = 1$ into our expressions and see what the SVM optimization procedure chooses for w, and what generalization bound this gives us.

```
CPick = CPicks[0]
wPick = BestWGivenC.subs(C,CPick)
print('wPick',wPick)
print('ErrorBoundw',sympy.simplify(ErrorBound.subs(w,wPick)))
```

```
('wPick', 2**(2/3)*n**(1/3)/2)
('ErrorBoundw', 1/n + 3*2**(1/3)/(2*n**(1/3)))
```

We see with $C = 1$ we get $w = 2^{2/3}n^{1/3}/2$, which is growing as the number of training examples n increases (but not shrinking very fast). Our error bound is then $1/n + 3 \times 2^{1/3}/(2n^{1/3})$, which is slowly shrinking as n grows.

In fact the error bound is shrinking much slower than one might initially expect. The issue is our data doesn't have any intrinsic large margin (a common problem) for real data, but we are having to build our own margin using the soft-margin method.

For a constant margin VC dimension theory tells us excess generalization error shrinks linearly in n, but picking the optimal margin (i.e. shrinking the margin as we get more data) only gives us an overall error-rate bound that is shrinking as $1/n^{1/3}$. Note this is only the bound, as the actual error-rate is zero for any $w > 0$ for this toy problem.

For the fixed $C = 1$ the optimization algorithm is (correctly) picking smaller margins as more data becomes available. This shrinking margin unfortunately obscures some of the sample-size driven improvement in generalization error (the excess generalization error or VC dimension part of the bound). But the trade-off also allows the training algorithm to consider a larger fraction of the training data as being out of the moat (or not in the margin around the decision surface). This is correct behavior on the part of the optimization procedure, and is shrinking the error bound as fast as possible. But "fast" is much slower than any (incorrect) intuition that a fixed C implies a fixed margin as the amount of training data increases. Also note, it is not common to actually explicitly calculate the implied error bound when using SVMs (we just used the obvious structure of our toy problem to allow such a calculation). And further note, we these bounds are approximate as we used simplified forms and not the full equations from the reference (though we are confident we get the overall behavior correct).

This means we expect a halving of the error-bound only each time we multiply our available data by a factor of eight. This is slower than any rule of thumb that states the error-bound shrinks linearly with the amount of training data. Just keep this in mind when deciding how much data you may need for a good SVM result.