School of Computing and Information Systems
The University of Melbourne
COMP90049 Knowledge Technologies (Semester 2, 2017)
Workshop exercises: Week 9

1. Finish questions from last week, where necessary.
2. Revise Support Vector Machines, paying particular attention to the terms "linear separability" and "maximum margin".

   - Support vector machines attempt to partition the training data based on the best line (hyperplane) that divide the positive instances of the class that we're looking for from the negative instances.
   - We say that the data is *linearly separable*, when the points in our k-dimensional vector space corresponding to positive instances can indeed be separated from the negative instances by a line (or, more accurately, a $(k-1)$-dimensional hyperplane). This means that all of the positive instances are on one side of the line, and all of the negative instances are on the other side.
   - What is the definition of "best" line? Well, in fact, we choose a pair of parallel lines, one for the positive instances, and one for the negative instances. The pair that we choose is the one that has the greatest perpendicular distance between these parallel lines (calculated using the normal; because the lines are parallel, they have the same normal) — this is called the "margin".
   - These two lines are called the "support vectors" — we can classify test instances by calculating (again, using the normal) which of the support vectors is closer to the point defined by the test instance. Alternatively, we can just use a single line, halfway between the two support vectors, and use the normal to calculate which "side" of the line the test instance is on.

   (a) What are "soft margins", and when are they desirable?

   - Soft margins are when we relax the notion of linear separability. A small number of points are allowed to be on the "wrong" side of the line, if we get a (much) better set of support vectors that way (i.e. with a (much) larger margin).
   - Sometimes the data is *almost* linearly separable — if we accept that we probably aren't going to classify every single test instance correctly anyway, we can produce a classifier that hopefully generalises better to unseen data.

   (b) Why are we interested in "kernel functions" here?

   - The so-called "kernel trick" is often used to transform data.
   - For Support Vector Machines, sometimes the data isn't linearly separable, but after applying some kind of function — where some useful properties of the data remain (for example, monotonicity of the inner product) — the data becomes linearly separable. If we do this, we can apply the algorithm as usual.

   (c) Why are SVMs "binary classifiers", and how can we extend them to "multi-class classifiers"?

   - We're trying to find a line that partitions the data into true and false, suitably interpretted for our data.
   - The point is that for two classes, we only need a single line. (Or more, accurately, a pair of parallel support vectors.)
   - We need (at least) two lines to partition data into three classes. For example, $x < 0$, $0 < x < 1$, $x > 1$ might be a useful partition. But what happens when these lines aren't parallel?
   - In general, we probably want to find three "half–lines", radiating out from some central point between the (points corresponding to) the three different classes. The problem? This is numerically much more difficult (and usually intractable) to solve.

- For problems with more than three classes, the entire notion of separating the data becomes poorly-defined — unless we wish to fall back to some clustering approach, in which case, we might as well use Nearest Prototype.

3. Demo Project 2. If you wish to use Weka, it can be found at:
   `http://www.cs.waikato.ac.nz/ml/weka/`