Introduction to Machine Learning Assignment 1

Divij Singh

05/09/18

1 Q1

So, in order to start designing this classifier, we'll start from the wrong end, so to speak, and look at the output we desire from the system. In this case, we want one of two cases, that the applicant is likely to default, or that the applicant is unlikely to default.
We can define this as $P(def\|x)$, where $P(def)$ is the probability of a person x defaulting on their loan. Thus, the probability of the person *not* defaulting on their loan is $P(!def\|x)$.
In order to train this classifier, we look at the history of a person's loans, where we look at whether or not they have defaulted on loans. We can thus get $P(!def\|x)$ by an equation of (non-defaulted loans)/(total loans), which we then use with Bayes's theorem ($P(def\|x) = \frac{P(!def\|x)P(def)}{P(x)}$) to get a value for $P(def\|x)$.
Now that we have a probability of defaulting for each case, we look at other factors. In order to feed the data to the system, we can divide each data point into specific divisions.
We can divide gender into male/female/unspecified, marital status into married/unmarried/divorced, income into numerical brackets, and education into various levels such as state board, bachelors, and so on. In the case of existing loans, we could simplify it into a single variable y, via something such as $y =$ (total amount of money owed in existing loans)/(number of loans) in order to make a single datapoint (please don't use this in an actual case, that is a horrible simplification).
All of this we feed into the classifier, alongside the probability of defaulting which we defined.

2 Q2

This can be optimal in some cases, however there are several real world scenarios where this would fail.
An example is, say, the skill distribution of people playing a video games.
Say we rank people from 1 to 100 based on their skill, and then make uniform divisions of every 10 skill ratings (that is to say, 1-10, 11-20, etc). Now the

average skill ratings would be clustered around the 40-60 range, that is to say if you were to graph skill rating on the x-axis, and number of players on the y-axis, the graph would start from 0, rise till around 50, peak, and then drop again.
As a result, if you wanted to train a system using this data, you would experience a large loss of sensitivity around the 41-50 and 51-60 brackets, due to the large number of players clustered there.
An alternative could be to make divisions by, say, every 1000 players or so, depending on the size of the population.