**3 Exploit SVMs to verify the decisive factors**

**3.1 Necessity to verify individually**

We are all the youth who have played or have heard some video games. As to the issue of finding out a good enough prediction of Dato2, our experiences and intuitions naturally tell us: There are many factors that may affect the result differently, such as the hero familiarity of its controller, the skill of this match, the mistake or accident during the match.

From the data set, actually we found there are 11 items for each player in a match, which are identified as *denies*, *xp\_per\_min*, *player\_slot*, *kills*, *level*, *deaths*, *hero\_damage*, *last\_hits*, *hero\_id*, *tower\_damage* and *gold\_per\_min* respectively. They are all the information we can make use of.We are asked by the proposer to use the items of “hero-id” and “level” only to predict the winner originally. But our group doubt whether or not the hero composition and the level of players truly are the most decisive factors. If the answer is negative, the accuracy of some machine learning method can be poor and meaningless. So one of our key points is that we need to figure out how much the single item associate with the accuracy of prediction result individually.

**3.2 Apply SVMs efficiently**

Our data set, with the size of 100,000 matches, can be regarded as a 10\*11\*10,000 three-dimensional matrix. It’s a mid-size data! Fortunately, through the course of Machine Learning, we know: 1) SVMs, which are among the best (and many believe is indeed the best) “off-the-shelf” supervised learning algorithm; 2) Kernel trick, which give a way to apply SVMs efficiently in very high dimensional feature spaces. So we choose to use the algorithm of kernel trick which should be suitable to solve this problem.

Since the story of SVMs has been told in detail in our class, here we just simply summarize our idea as follows:

Firstly, we divide 100,000 pieces of match data into two parts, 90,000 pieces for training, 10,000 pieces for testing.

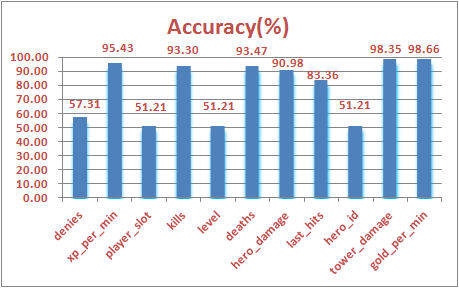
Secondly, execute like following pseudo-code.

1. iterator i = 1 : 11
2. SvmTrainData(i, 1:90000)
3. SvmTestData(i,90001:100000)

**3.3 Reveal the truly most decisive factors**

We make a bar graph to conclude the accuracy of prediction results using 11 kinds

of items individually as follows.



As we can see, two most decisive factors are “gold\_per\_min” and “tower\_damage” rather than “hero\_damage” and “level”. We further infer the accuracy of prediction results may be just so so if we only use the items of “hero-id” and “level”. What we discover is coincident with the baseline, given by the proposer, which gets 61% of accuracy. It seems to draw a conclusion the more money you earn and the more damage you bring to your enemy’s tower, the higher probability you will win.