Multi-Armed Bandit

by Paul Servino

UCLA CS124, Spring ‘09

© TemplatesWise.com

Outline

- A Machine Learning Problem
- A Biological Problem
- Defining a Solution
- Solution
- Results
- Further Work

A Machine Learning Problem

- Optimization of decisions.
- Traditionally:
  - Slot machine with a number of levers.
  - Each lever has a different probability of winning associated with it.
  - Without any prior knowledge, the gambler must find the best lever(s) to pull to maximize his profit.
  - Presents three choices after each pull:
    - Pull the same lever again (Exploit).
    - Pull one of the levers already pulled, again (Exploit).
    - Try a new lever (Explore).

A Biological Problem

- Now, we collect all SNPs from one invididual at a time.
- The multi-armed bandit problem is choosing one SNP from an inidividual at a time.
- A *theoretical* biological problem.
  - The technology and resources to put a solution to the Multi-Armed Bandit problem to use do not exist.

Slot Machine | Genetics

- Slot machine with a number of levers.
- Each lever has a different probability of winning associated with it.
- Without any prior knowledge, the gambler must find the best lever(s) to pull to maximize his profit.
- Presents three choices after each pull:
  - Pull the same lever again (Exploit).
  - Pull one of the levers already pulled, again (Exploit).
  - Try a new lever (Explore).

- Genome with a number of SNPs.
- Each SNP has actual frequencies: p+A, and p-A, associated with it.
- Without any prior knowledge a researcher must find the best SNPs to test to maximize his gain.
- Presents three choices after each test of an individual's SNP:
  - Test the same SNP again on a different individual (Exploit).
  - Test one of the SNPs already tested, again (Exploit).
  - Test a new SNP (Explore).

Defining a Solution

- Fewer(est) number of tests possible to reach the same conclusion as from a normal association study.
  - Identifying the same SNPs as associated.
- Higher(est) power attainable in the same number of individual tests as a normal association study.

Solution - Approach

- Biological
  - Actual tests in this fashion are nonexistant.
- Mathematical
  - Numerous ‘solutions’ exist.
  - Not particularly my thing.
- Computer Science
  - Simulation and testing.

Solution - Key Algorithm

- Weighs Exploration vs. Exploitation
  - Explore
    - When to give up on SNPs that are believed to not be associated.
    - When to be convinced enough that the SNP tested is associated to move on.
  - Exploit
    - Not enough tests to make a decision.
    - Gains from testing SNPs again.
- Attempts to move through (a subset of the) genome as quickly as possible, identifying all associated SNPs.

Solution - Design

- Built a **simulator** as a framework in which to test my bandit algorithm.
  - ~600 lines of C++.
  - Very little memory usage as data is randomly generated within set bounds and tabulated.
  - Generates a number of SNPs which are either:
    - Unassociated ($p_A^+ = p_A^-$).
    - Associated ($p_A^+ \neq p_A^-$).
  - Simulates Association Studies.
    - Calculates Association statistics with every individual SNP test.
      - Allows for easy comparison between normal association studies and individual algorithmic (bandit) runs.
  - Simulates bandit runs with ability to explore set of SNPs or to exploit the currently tested one.

Solution - Algorithm

- Simple.
  - Instead of a complicated multipass algorithm, I chose a simple one and optimized its values.
- This was a good starting point and can lead to a better multipass, tiered design in the future.

Solution - Algorithm

- First, I ensure that there is a firm base to be making an Explore / Exploit decision by having the algorithm test a certain number (baseDivisor) of inidividuals for that SNP.
- Then, I calculate the statistic for the values.
- I then see if it is over or under a certain value (compStat), which if:
  - Under (probably not associated) = Explore!
  - Over (probably associated) = Exploit!

```
if(v_SNPs.at(m_currentSNP)->getNumTests() > N_LIMIT / baseDivisor)
{
    if( v_SNPs.at(m_currentSNP)->getStat() > compStat)
      Exploit();
    else
      Explore();
}
```

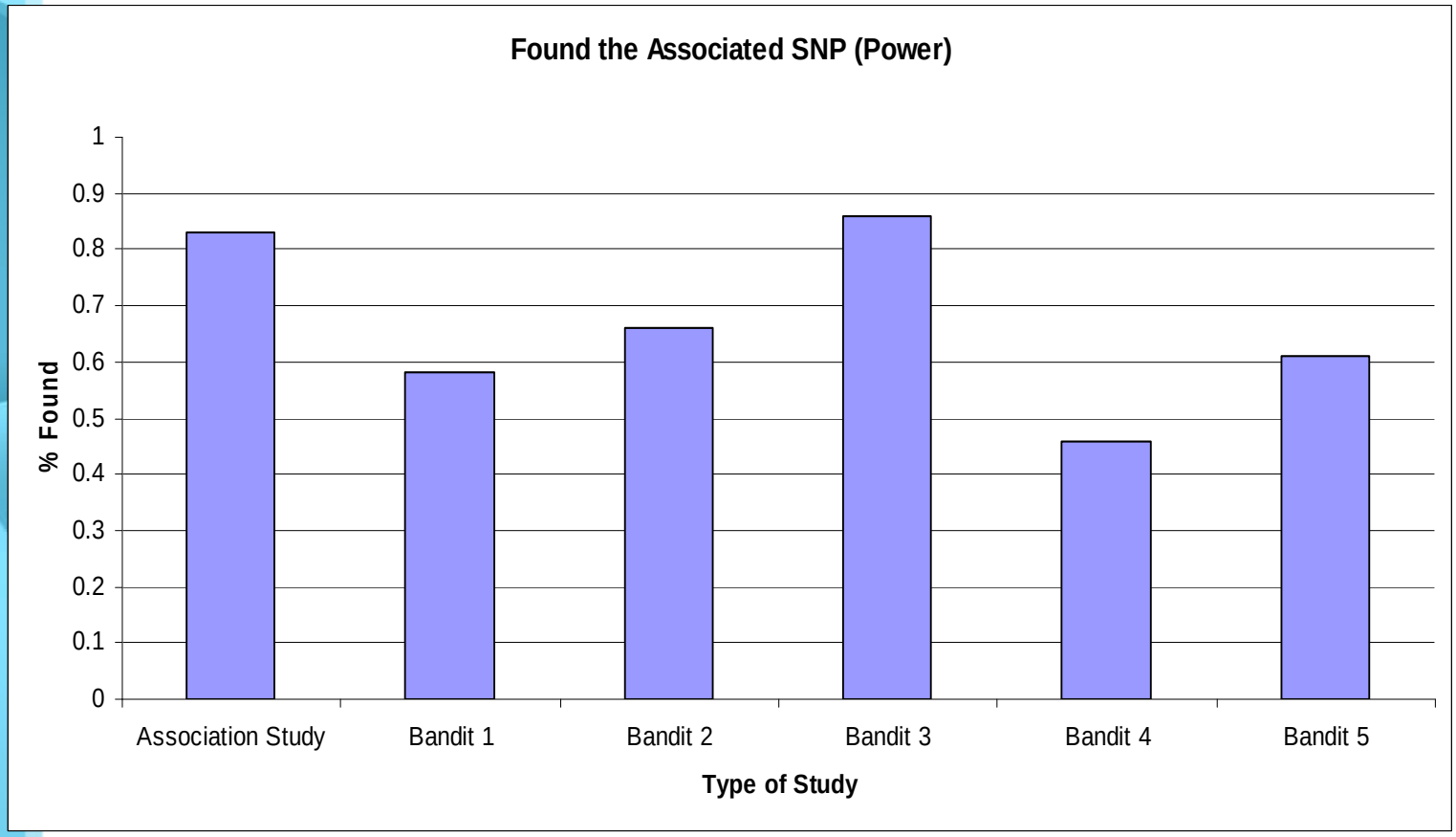
Testing

- 5 Similar Bandit Algorithms.
- 100 Tests, one of which:
  - Generated 100 SNPs.
    - 99 Unassociated ($p_A^+ = p_A^-$)
    - 1 Associated ($p_A^+ \neq p_A^-$)
  - Ran an algorithmically (Bandit) determined number of individual tests on those SNPs (MAX = 300 per SNP).
  - Attempted to find the Associated SNP.

Results - Comparisons

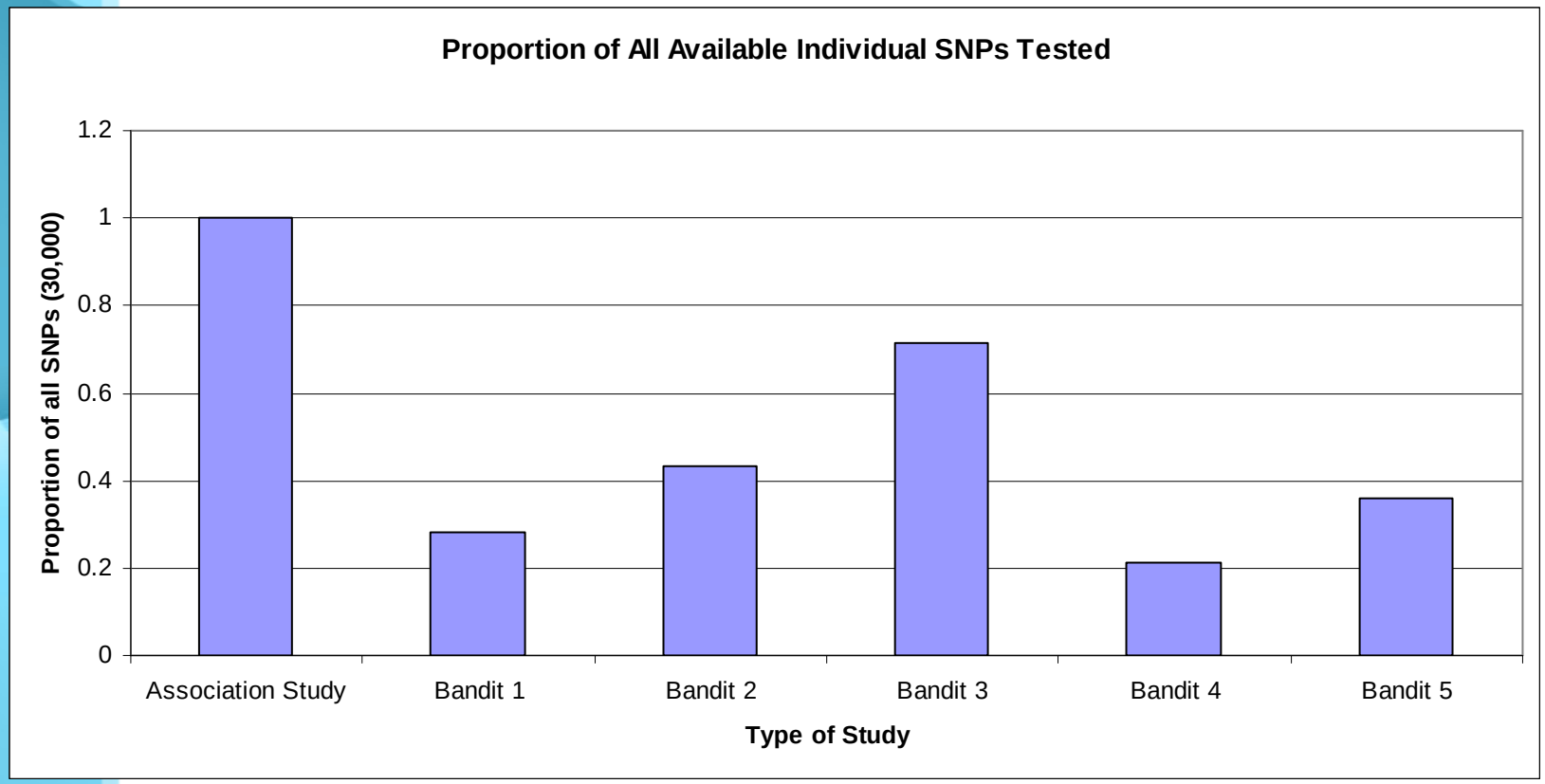
SUMMARY				
Test Type	**% Associated SNP found**	**Number of Individual SNPs tested**	**Average Error**	**Significance Ratio = (%found) / (AverageErr * NumIndivSNPs)**
Association Study	0.83	30000	1.1	25.15
Bandit 1	0.58	8473	0.51	134.22
Bandit 2	0.66	13046	0.59	85.74
Bandit 3	0.86	21480	1.25	32.02
Bandit 4	0.46	6394	0.3	239.80
Bandit 5	0.61	10766	0.71	79.80

Results - Comparisons



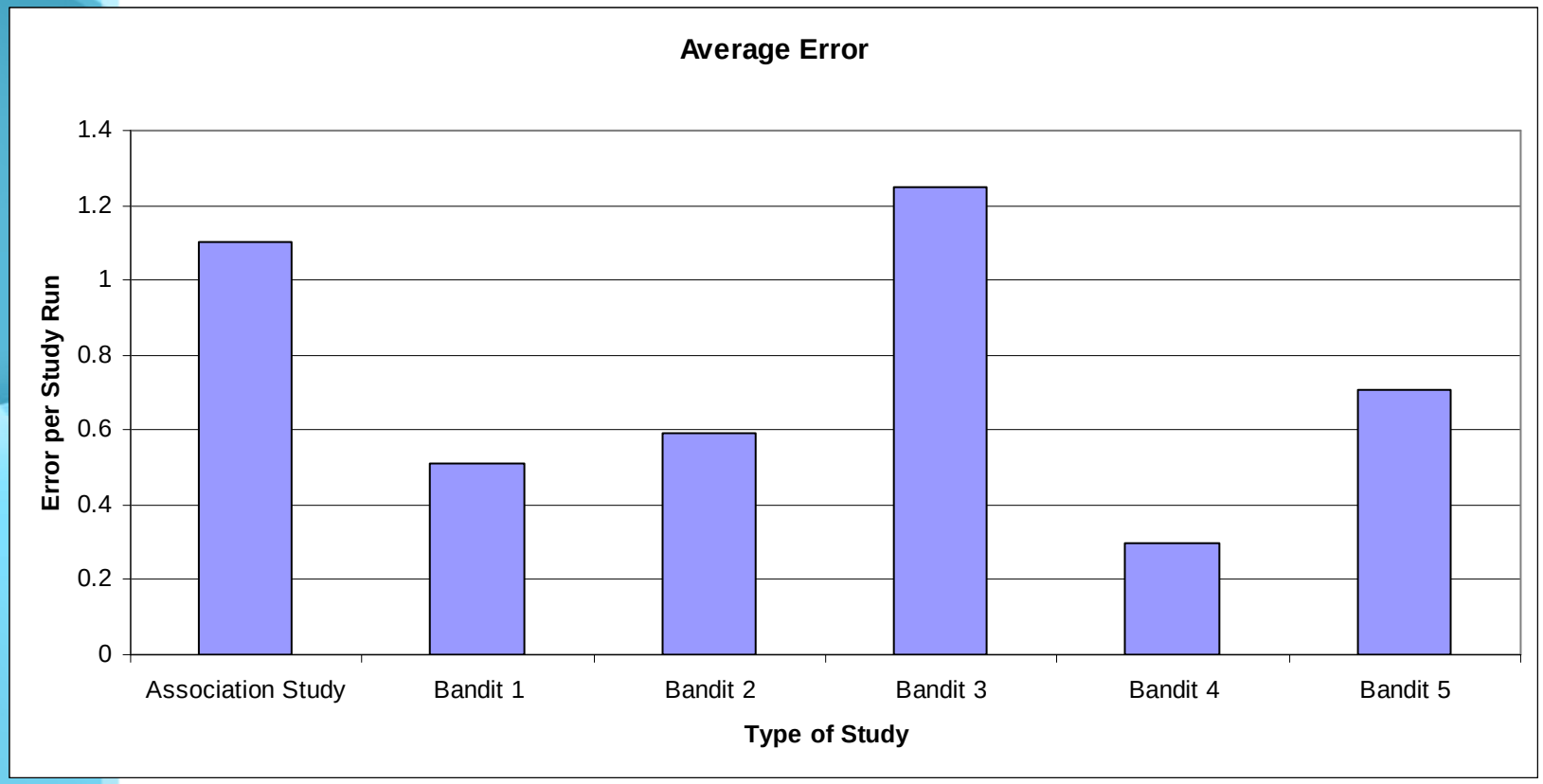
Higher is better.

Results - Comparisons



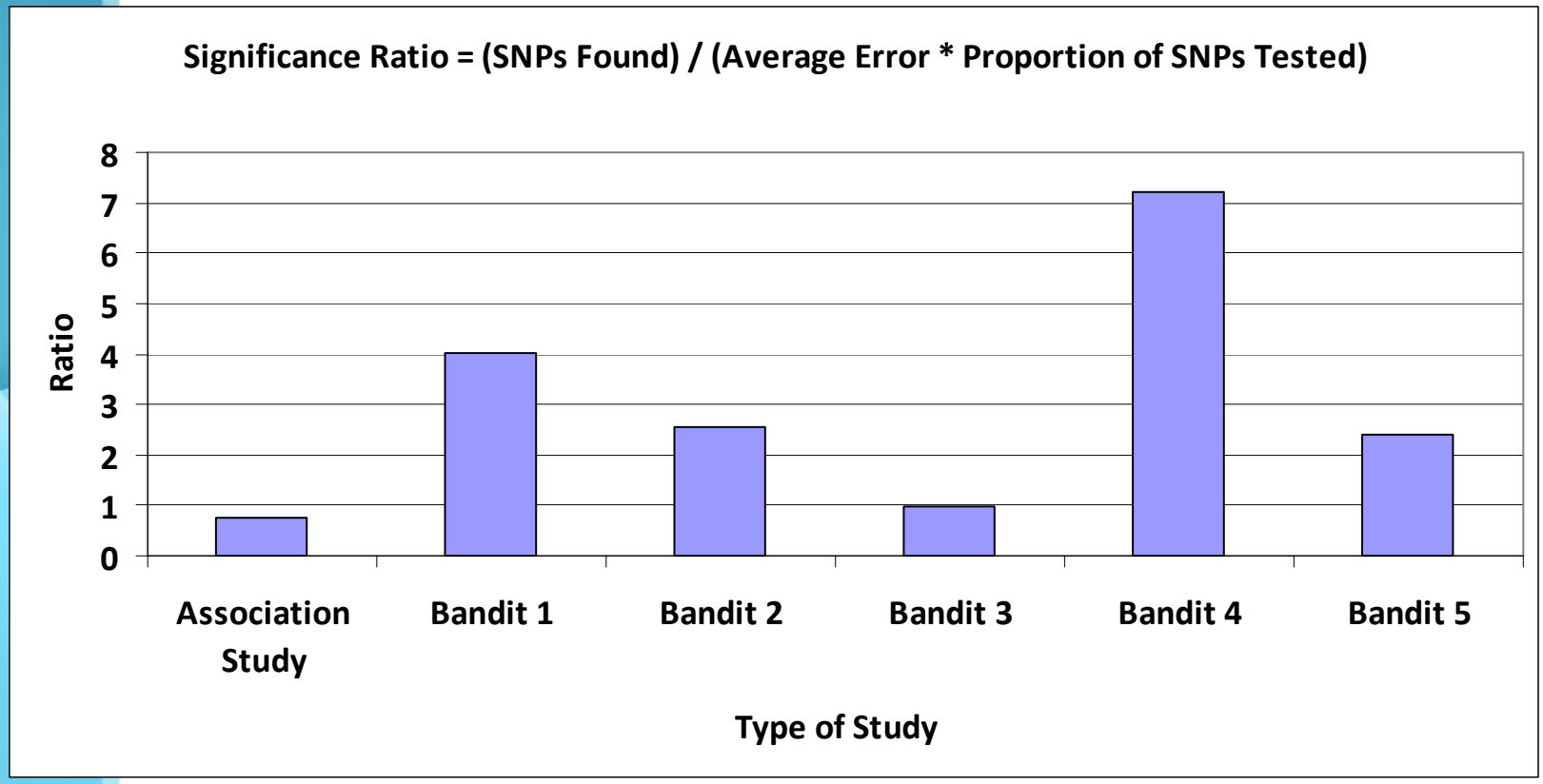
Lower is Better.

Results - Comparisons



Lower is Better.

Results - Comparisons



Higher is Better.

Results - Conclusions

- **Bandit 4** with compStat of 1 and baseDivisor of 10, (considering error, power, and number of total tested SNPs equally), both outperformed the association study and the other Bandits.
- As evidenced by **Bandit 3**, it is possible to achieve similar and even higher power than a normal Association Study with less tests.
- There are tradeoffs one can make if trying to favor a particular aspect of the study.

Further Work

- More Complicated Algorithm.
  - Multi-Pass Approach.
  - Tiered Approach.
- More Extenisve Testing.
  - Maximize power per test.
- Take into account and eploit correlation between SNPs.

Open to questions?

Thank you.

© TemplatesWise.com