When Does Deep Learning Work Better Than SVMs or Random Forests?

Mithun Prasad, PhD
miprasad@Microsoft.com

Microsoft

Cases of Predictive Performance

- Caruana, Rich, and Alexandru Niculescu-Mizil. "*An empirical comparison of supervised learning algorithms.*" Proceedings of the 23rd international conference on Machine learning. ACM, 2006.

Start Simple

- Start with the simplest hypothesis space first:
  i.e., try a linear model such as logistic regression.

- If this doesn't work "well" (in other words, it doesn't meet our performance expectation – accuracy/precision/recall,etc.), move on to the next experiment.

- The next experiment could be a "worry-free" approach if it exists in ML ☺

"Worry-Free" Approach

- Random forests are probably THE "worry-free" approach.
- There are no real hyperparameters to tune (except for the number of trees; typically, the more trees we have the better).
- Lot of knobs to be turned in SVMs: Choosing the "right" kernel, regularization penalties, the slack variable, ...
- Non-parametric models (Random forests / SVMs):
    - The complexity grows as the number of training samples increases.
    - The more trees we have, the more expensive it is to build a random forest. Also, we can end up with a lot of support vectors in SVMs.
    - Although, there are multi-class SVMs, the typical implementation for mult-class classification is One-vs.-All;
    - We have to train an SVM for each class. In contrast, decision trees or random forests can handle multiple classes out of the box.

Microsoft

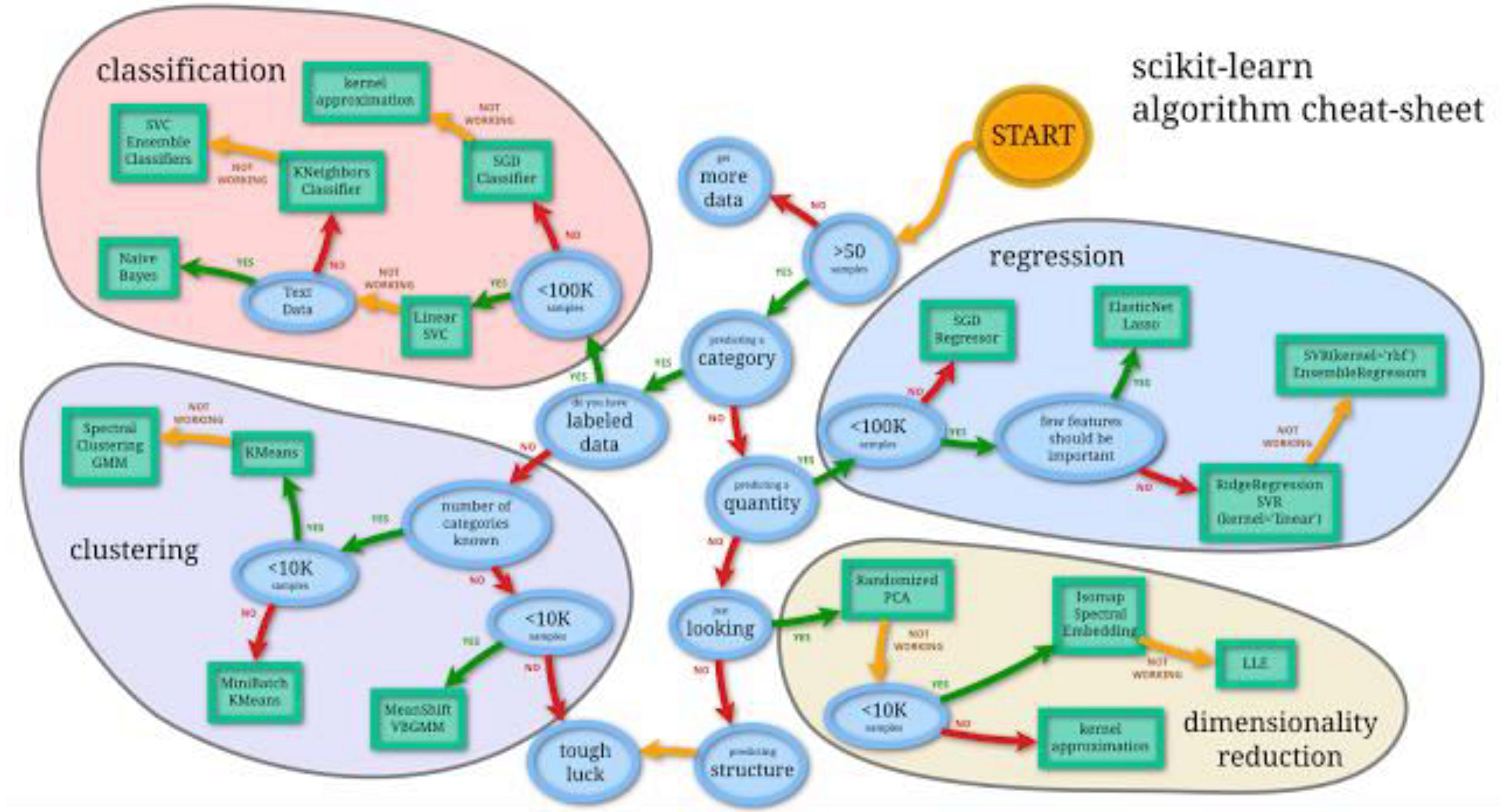
Rule of Thumb

- SVMs are great for relatively small data sets with few outliers.

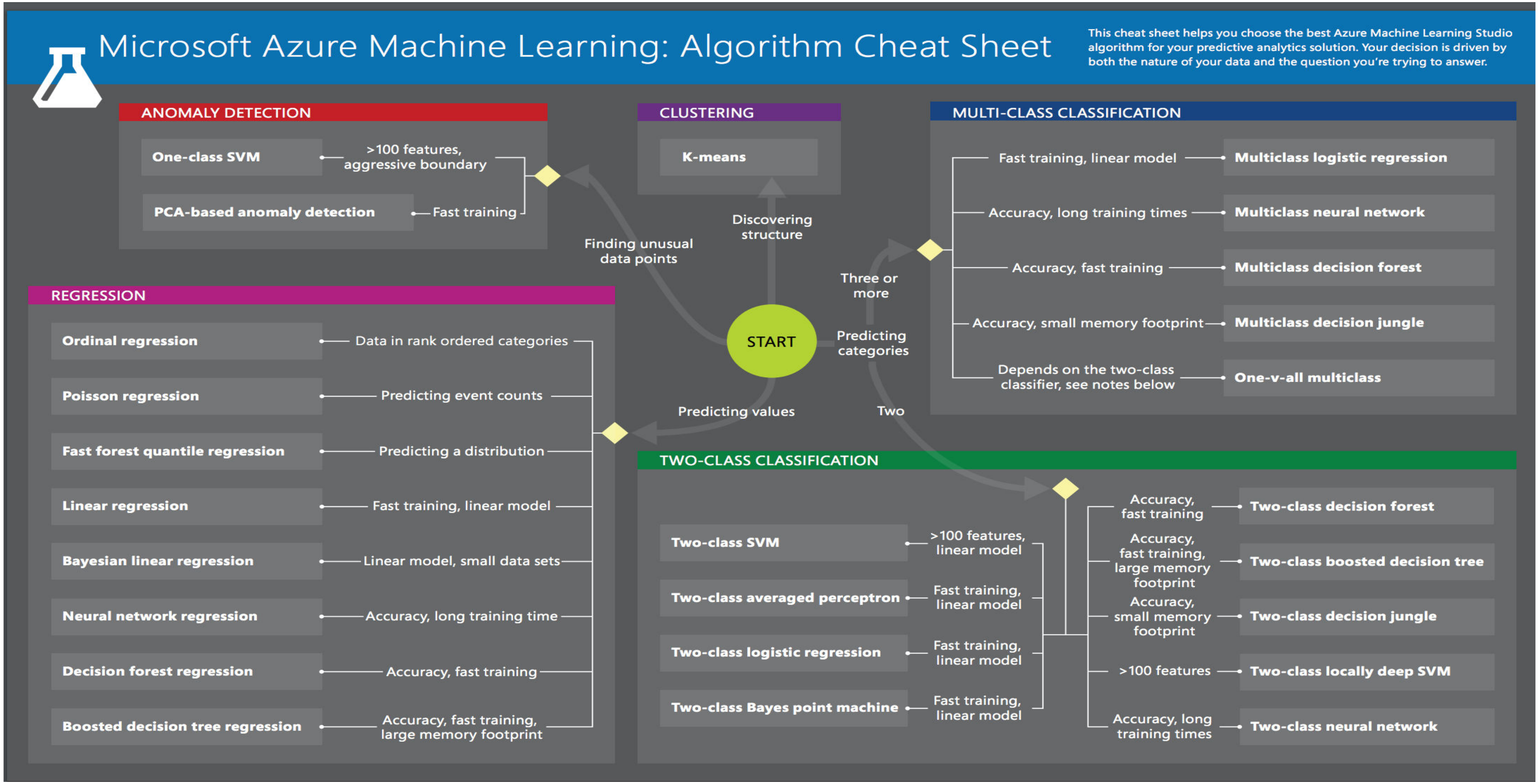
- Random forests may require more data but they almost always come up with a pretty robust model.

- Deep learning algorithms:
  - Cons:
    - Require "relatively" large datasets to work well. In addition, we need the infrastructure to train them in reasonable time. Setting up a neural networks is much more tedious than using an off-the-shelf classifiers such as random forests and SVMs.
  - Pros:
    - Deep learning really shines when it comes to complex problems such as image classification, natural language processing, and speech recognition.
    - Very less effort with Feature Engineering.

Cheat Sheet (Scikit-Learn)



Cheat Sheet (Azure ML)



Conclusion

- In practice, the decision which classifier to choose depends on:
  - your dataset.
  - general complexity of the problem -- that's where your experience as machine learning practitioner kicks in.

- Define a performance metric to evaluate your model.

- Ask yourself: What performance score is desired, what hardware is required, what is the project deadline.

- Start with the simplest model.

- If you don't meet your expected goal, try more complex models (if possible).

Microsoft