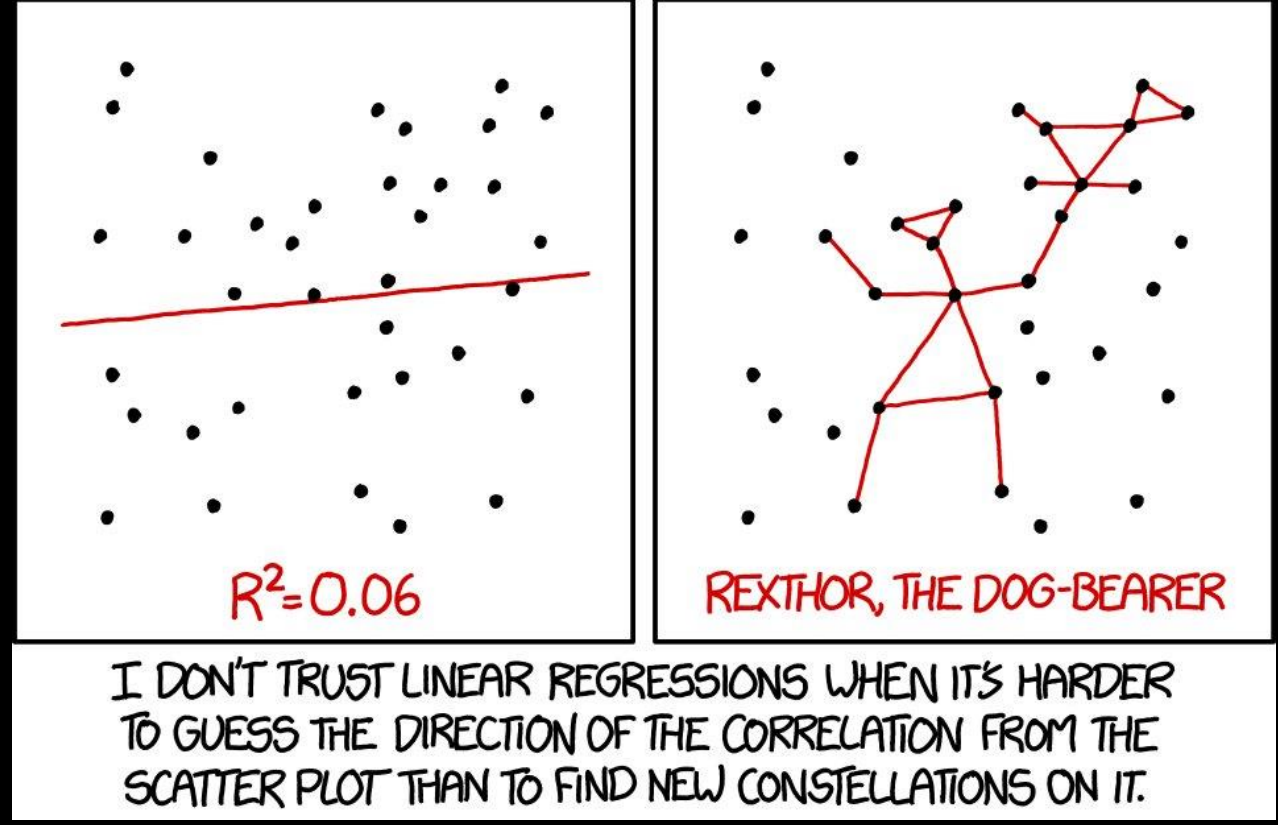


INFO 251: Applied Machine Learning

Supervised Learning Summary

Announcements

- Home stretch! PS6 posted now, PS7 posted soon
- Please provide feedback on ML Bias lectures

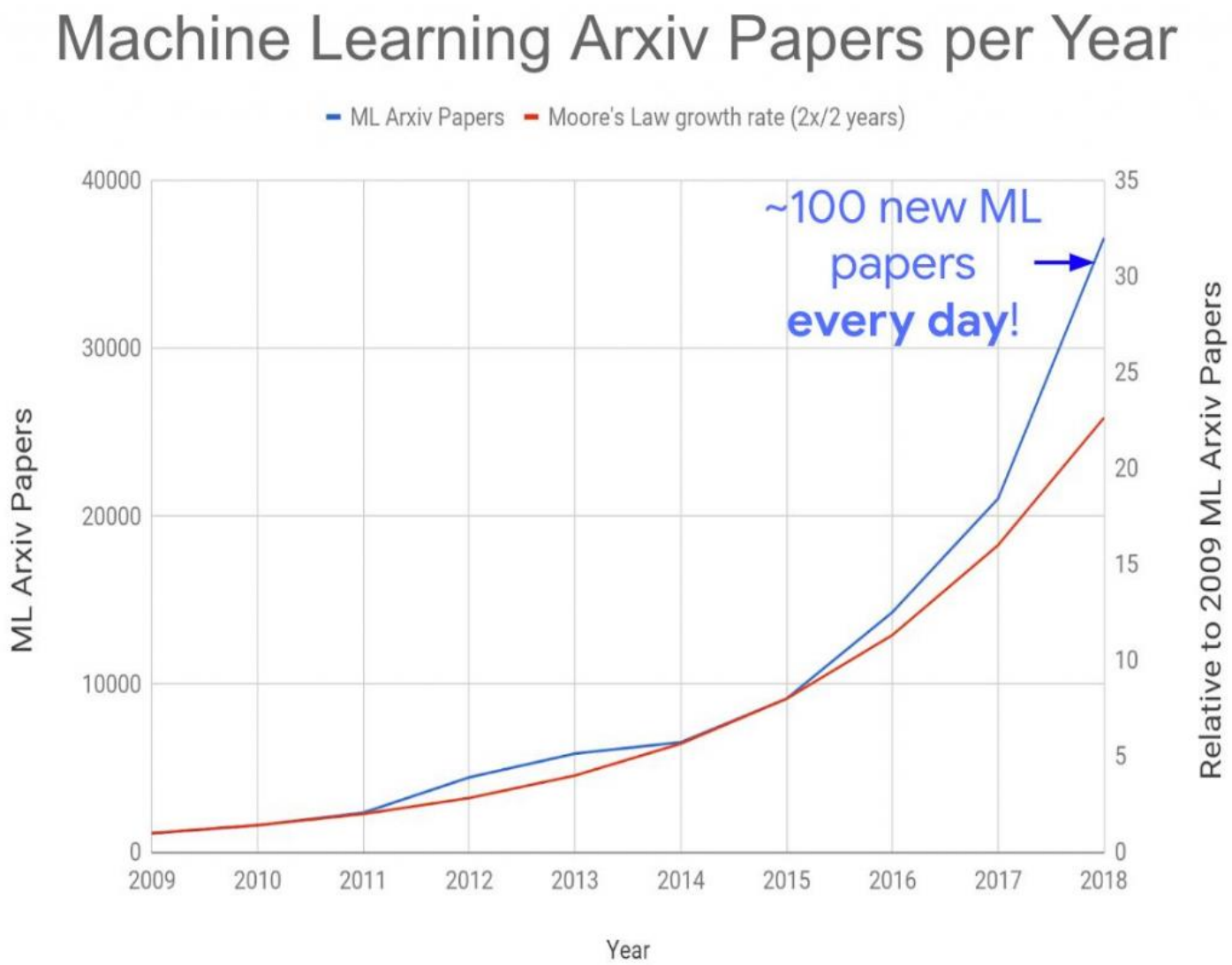
Course Outline

- Causal Inference and Research Design
  - Experimental methods
  - Non-experiment methods
- Machine Learning
  - Design of Machine Learning Experiments
  - Linear Models and Gradient Descent
  - Non-linear models
  - Neural models
  - Fairness and Bias
  - **Practicalities and summary**
  - Unsupervised Learning
- Special topics

Outline

- **Algorithm inventory**
- Comparing classifiers
- Guiding principles
- Consulting/Mock Interview Question

Machine Learning: Context

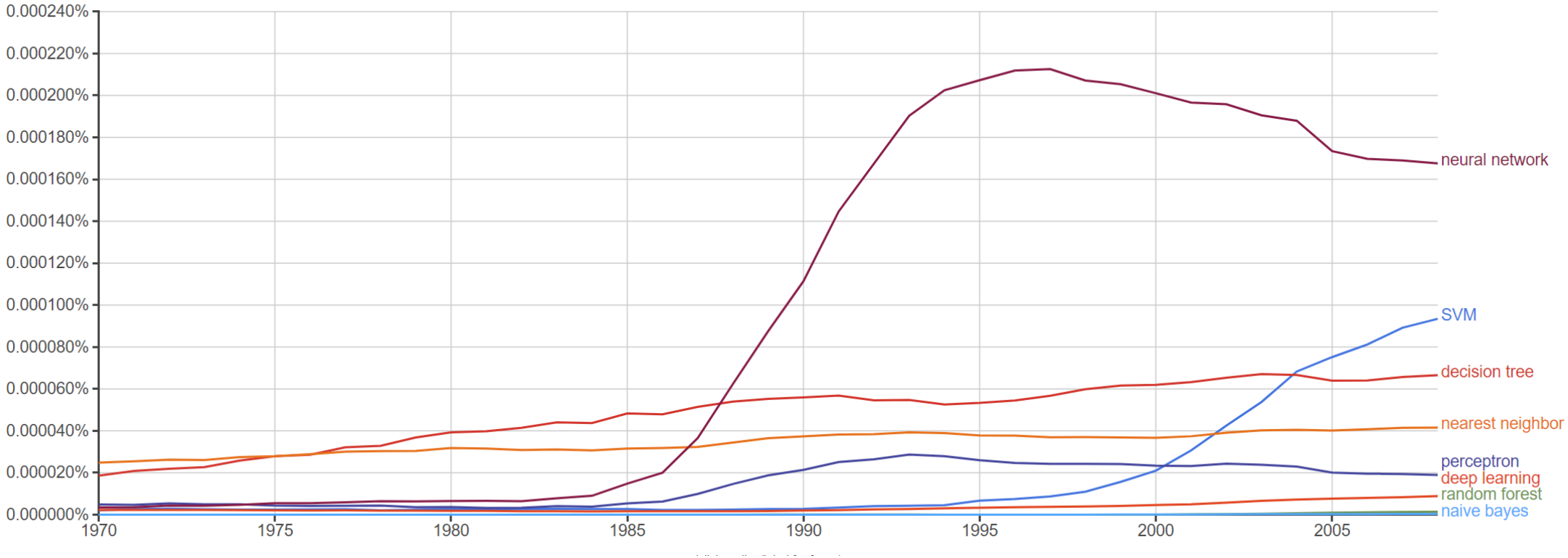


We've focused on these algorithms:

- Nearest Neighbor
- *k*-Nearest Neighbors
- Linear Regression
- Logistic Regression
- LASSO / Ridge
- Naive Bayes
- Decision Trees
- Regression Trees
- Random Forests
- Boosted trees
- Perceptrons
- Neural Networks
- ConvNets
- LSTM's

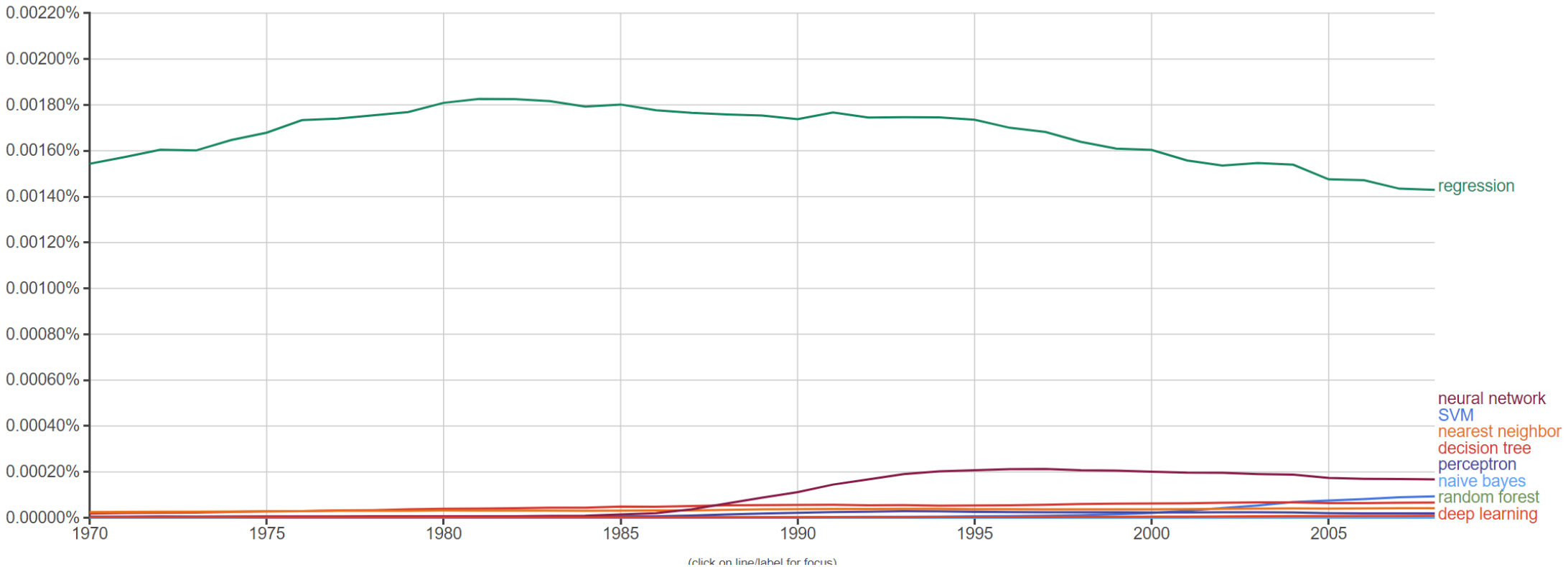
Modeling trends

- Google books ngram viewer, through 2010
- What's missing?



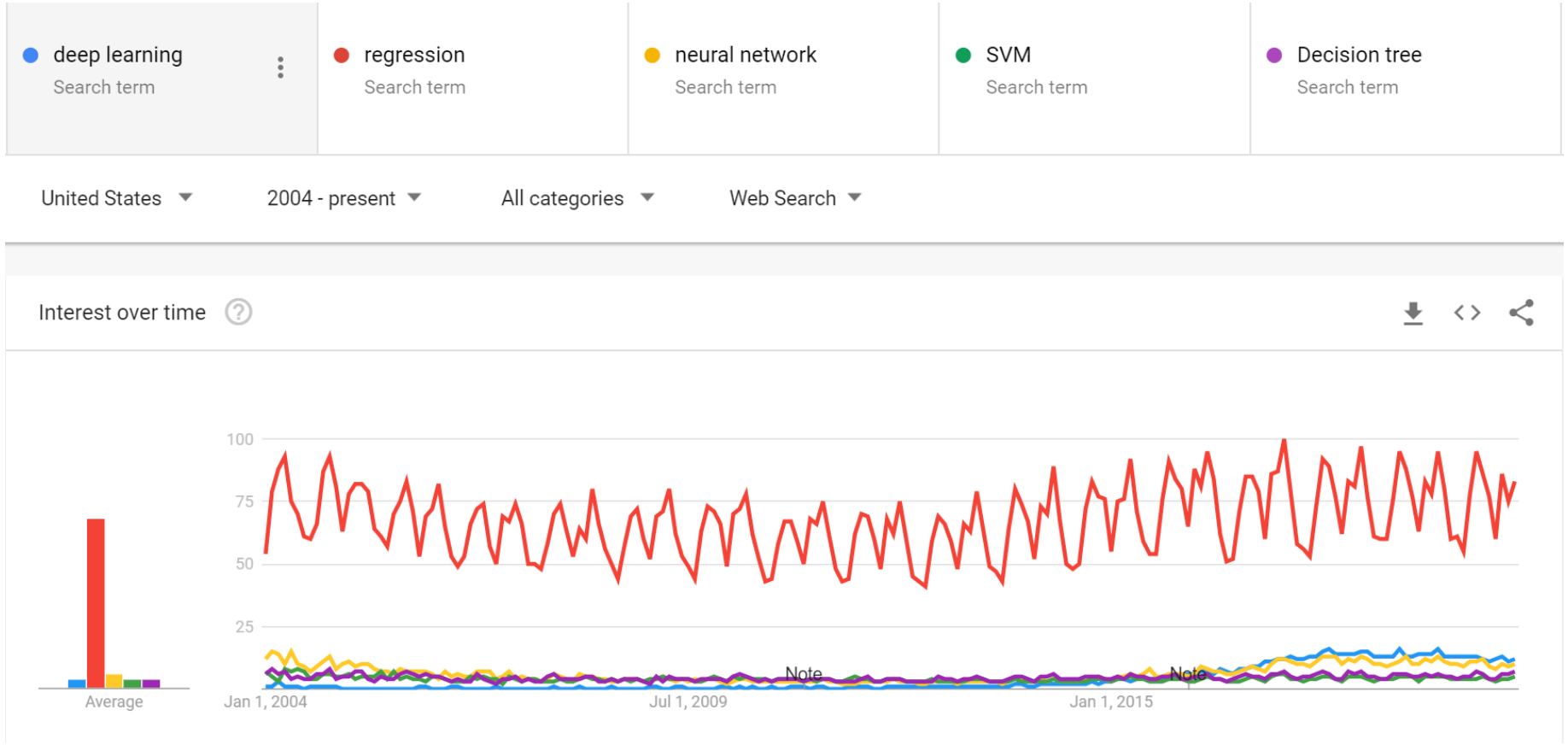
Modeling trends

- Google books ngram viewer, including "regression"

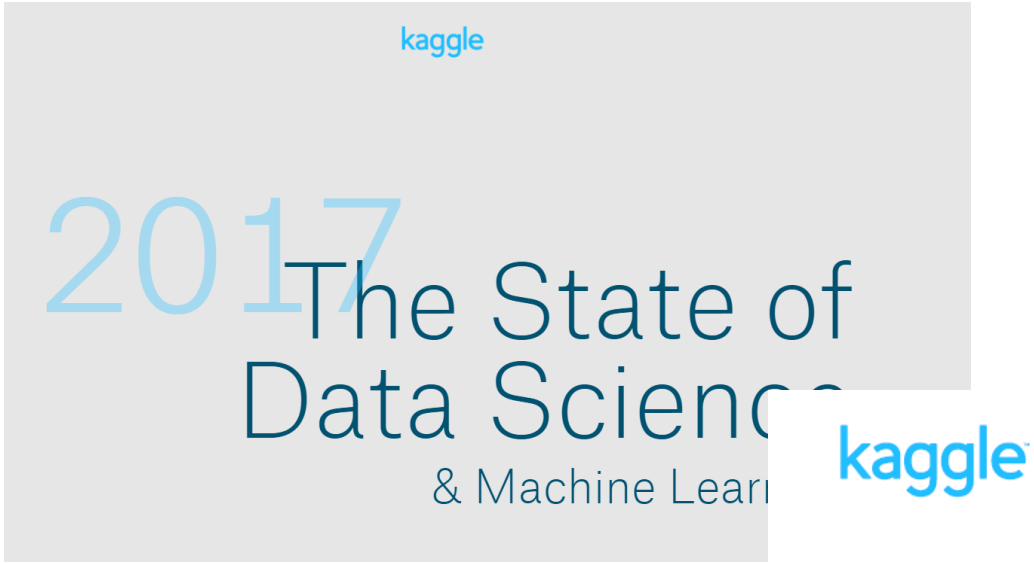


Modeling trends

- Google Trends, 2004 - present



Machine Learning Trends



Kaggle's State of Data Science and Machine Learning 2019

Enterprise Executive Summary

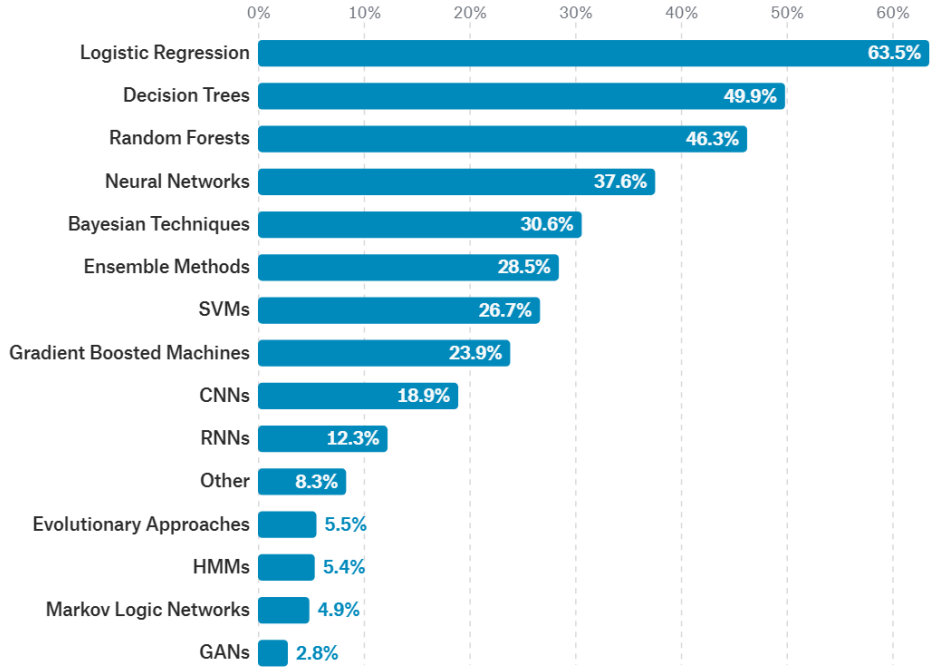
Machine Learning Trends

2017

What data science methods are used at work?

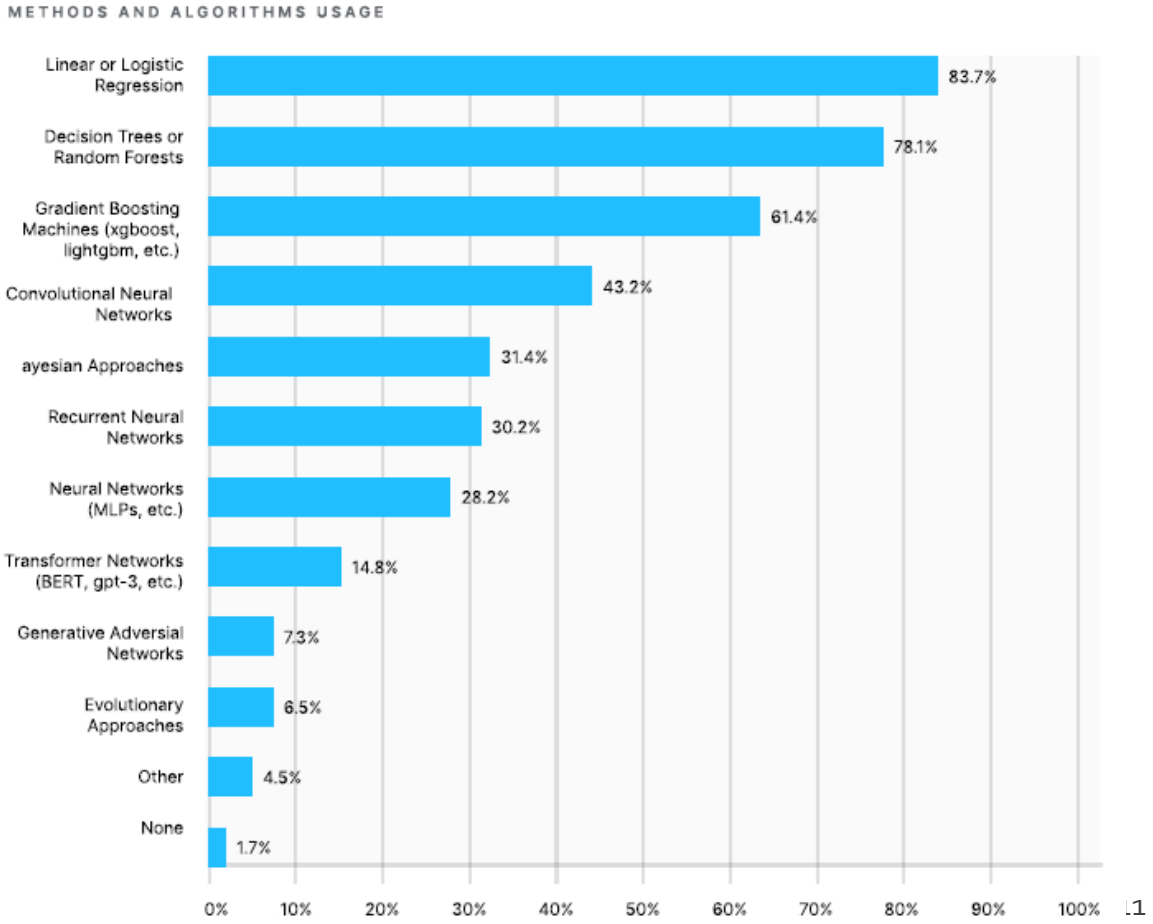
Logistic regression is the most commonly reported data science method used at work for all industries *except* Military and Security where Neural Networks are used slightly more frequently.

Company Size ▾ | Industry ▾ | Job Title ▾

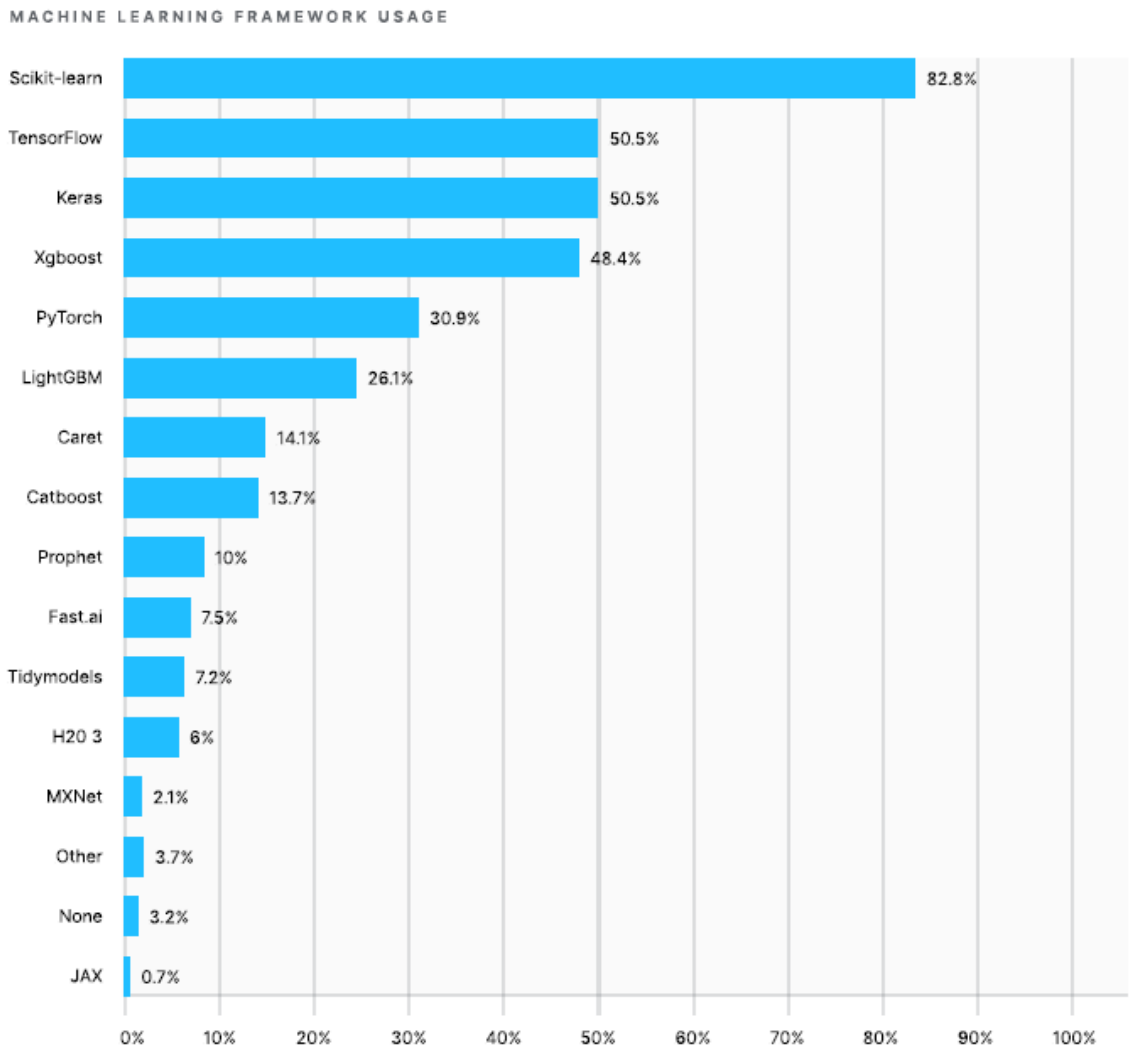


7,301 responses

2020



Machine Learning Trends

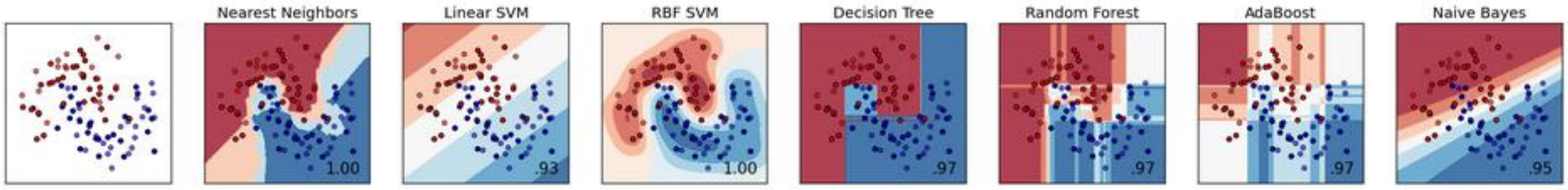


Outline

- Algorithm inventory
- **Comparing classifiers**
- Guiding principles
- Consulting/Mock Interview Question

Comparing classifiers

- *k*-Nearest Neighbors
- Naive Bayes
- Decision Trees
- Logistic Regression
- Neural Net



Comparing classifiers

- ***k*-Nearest Neighbors**
  - PRO: simple, intuitive, no training
  - CON: overfitting, large "model", slow inference
  - AND: must choose a good distance metric
- Naive Bayes
- Decision Trees
- Logistic Regression
- Neural Net

Comparing classifiers

- *k*-Nearest Neighbors
- **Naive Bayes**
  - PRO: simple, generative model, fast training, scalable
  - CON: independence assumption, unrealistic DGP
  - AND: feature selection and smoothing are important
- Decision Trees
- Logistic Regression
- Neural Net

Comparing classifiers

- *k*-Nearest Neighbors
- Naive Bayes
- **Decision Trees**
  - PRO: logical inference, small models, non-linear
  - CON: single tree suboptimal; forest hard to interpret
  - AND: usually improved performance with boosting
- Logistic Regression
- Neural Net

Comparing classifiers

- *k*-Nearest Neighbors
- Naive Bayes
- Decision Trees
- **Logistic Regression**
  - PRO: calibrated probabilities, scalable, interpretable
  - CON: linear models
  - AND: widely used for prediction everywhere
- Neural Net

Comparing classifiers

- *k*-Nearest Neighbors
- Naive Bayes
- Decision Trees
- Logistic Regression
- SVM
- **Neural Net**
  - PRO: can give state-of-the-art performance, non-linear
  - CON: uninterpretable, computationally demanding
  - AND: lots of current ML research in this area

Mini-Exercise 1

- Rank these algorithms from "least" to "most" interpretable
  1. *k*-Nearest Neighbors
  2. Linear Regression
  3. Logistic Regression
  4. Naive Bayes
  5. Decision/Regression Tree
  6. Random Forests
  7. Boosted RF (adaboost, GB)
  8. Perceptron
  9. Neural Network

Mini-Exercise 2

- I'll create 9 breakout room groups
- I will assign each breakout room an algorithm
- Spend 8 minutes preparing a 2-minute summary of the algorithm
- Present to class (presenter will be chosen randomly by Prof)

Mini-Exercise 2

- Algorithm
  1. *k*-Nearest Neighbors
  2. Linear Regression
  3. Logistic Regression
  4. Naive Bayes
  5. Decision/Regression Tree
  6. Random Forests
  7. Boosted RF (adaboost, GB)
  8. Perceptron
  9. Neural Network

- How does it work?
  - What is the model?
  - Continuous or discrete output?
- Parameters
  - What (hyper)parameters are specified?
  - What parameters does the model learn?
- Decision Boundary
  - How to represent? Linear or non-linear?
- Interpretability
  - Easy or difficult? How to interpret?
- Computation
  - Training vs. Run-time
- Overfitting
  - How to reduce overfitting?
- What other pros/cons?

Outline

- Algorithm inventory
- Comparing classifiers
- **Guiding principles**
- Consulting/Mock Interview Question

Some guiding principles

- If you have about 1k - 10k training examples, it's worth trying a few linear and non-linear models
- Once you have Google-scale data, the model may not matter so much -- ridge regression is a good starting point
- Neural nets can win by discovering useful features in complex inputs (images, speech)
- Deep learning requires >~50k training examples

Some guiding principles

- Accuracy is not all that matters
  - Interpretability
  - Model size
  - Training/inference speed
- "No Free Lunch Theorem"
  - There is no universally best classifier

Some guiding principles

"The fundamental goal of machine learning is to generalize beyond the examples in the training set"

- Cross-validation is your friend
- Regularization and smoothing are your friends
- Over-fitting is your enemy
- Ensembles work!
- Features matter *a lot*
- Machine learning is an art!
- It's easy for bias to creep into your training data
- Never, ever train on your test data

One more thing: Build-then-refine

- In a real-world environment
  - Assemble the full pipeline first
  - Then make iterative improvements
  - Invest cycles where the marginal returns are largest
- Where to invest cycles?
  - Collecting more data/labels
  - Feature engineering
  - Feature selection
  - Missing data
  - Imbalanced data
  - Model choice
  - Hyperparameter tuning
  - Evaluating performance
  - Interpreting models
  - Ensembles

What if it's not working?

- Is your model able to fit your *training* data?
  - No: Problem is with representation
    - Model not expressive enough
    - Need better features
    - Are you optimizing what you should be? Check the model's loss function (e.g., log-likelihood), instead of your metric (e.g., predictive accuracy)
    - Is your implementation correct? Try it on a dataset on which you know it should succeed
    - Try adding a "cheat" feature that you know should be able to predict your response
    - Not enough data
    - Conduct error analysis!

What if it's not working?

- Is your model able to fit your *training* data?
  - Yes: Problem is with generalization
    - Training data not representative of test data
    - Model is too complicated
    - Too many features
    - Not enough data

Outline

- Algorithm inventory
- Comparing classifiers
- Guiding principles
- **Consulting/Mock Interview Question**

Data Science consulting exercise

- You have recently been hired by LinkedIn
- You get assigned to a marketing team that is trying to get inactive customers to re-engage with LinkedIn
- A key strategy used by this team is to send email blasts to inactive users, which are designed to encourage those subscribers to re-engage
- Over the past year, the team has sent out 10 such email blasts, each of which has targeted 100,000 inactive subscribers. Each group of 100k was chosen randomly
- They are currently preparing to send another email blast to another list of 100,000 users

Data Science consulting exercise

- The team wants your advice on how to choose those 100,000 individuals. Specifically, to justify your new job, they want you to pick a group of 100,000 users that will be more likely to respond than any of the previous groups they targeted
- Your goal:
  - Design a method to identify the 100,000 individuals in the full list of 100M inactive subscribers who are most likely to respond to the marketing email
  - You cannot run a new experiment, but you have access to all historical data from LinkedIn, including the data from the most recent email campaigns (described above)

Data Science consulting exercise

- **Goal:** Design a method to identify the 100,000 individuals in the full list of 100M inactive subscribers who are most likely to respond to the marketing email
  1. Start simple. Without machine learning, how might you quickly and intuitively select those 100,000?
  2. How can you frame this as a supervised learning problem? What are you trying to predict? What features would you use? What model(s) would be most appropriate and why?
  3. Describe how you will evaluate the performance of this email campaign