Project

Project details

Use BRFSS dataset or your own data.

Define the key issue, question or hypothesis you investigated. Explain the algorithm(s) you used and why you chose it/them, including any problems or difficulties and how you handled them. Provide a summary of relevant results in tabular or graphical form and a discussion of what they indicate about your question of interest. Include a brief conclusion. Discuss limitations to the analysis including possible sources of bias and overfitting. **The text should be 1500 words or less and the project should have no more than 3 tables or figures.** Include a title and references but no abstract.

See also: Project_Guidelines_Biostat216.docx

BRFSS project

The objective of the BRFSS is to collect uniform, state-specific data on preventive health practices and risk behaviors that are linked to chronic diseases, injuries, and preventable infectious diseases in the adult population. Factors assessed by the BRFSS include tobacco use, health care coverage, HIV/AIDS knowledge or prevention, physical activity, and fruit and vegetable consumption. Data are collected from a random sample of adults (one per household) through a telephone survey.

The Behavioral Risk Factor Surveillance System (BRFSS) is the nation's premier system of health-related telephone surveys that collect state data about U.S. residents regarding their health-related risk behaviors, chronic health conditions, and use of preventive services. Established in 1984 with 15 states, BRFSS now collects data in all 50 states as well as the District of Columbia and three U.S. territories. BRFSS completes more than 400,000 adult interviews each year, making it the largest continuously conducted health survey system in the world.

BRFSS project

- Each year contains a few hundred columns. Please see one of the original codebooks, e.g. https://www.cdc.gov/brfss/annual_data/2015/pdf/codebook15_llcp.pdf for complete details.
- The .Rdata file was converted from a SAS data format using pandas (a Python environment for statistical analysis); there may be some data artifacts as a result. In addition, they were further curated by me in R and then subset to a much smaller size for the purpose of the project (5,000 data points for each of years 2014 and 2015).

BRFSS project

Step 1: Load the data with load("datbrfss.RData")

Step 2: Use visualization/plotting tools to examine the datasets (both training and independent datasets). Are there extreme outliers? Are there any missing data?

Step 3: Determine a question that you want to examine. You can treat this as a regression, classification or clustering/data reduction problem.

Step 4: Fit a classification, regression, or clustering model/algorithm to the training dataset (2014). Determine the classification accuracy or cluster separation within the training dataset. Did you try any other models/algorithm? Why did you choose the particular model/algorithm that you did? Were there any parameters that you needed to tune? How were the parameter values chosen?

Step 5: Evaluate the regression/classification accuracy or cluster reliability of the model/algorithm fitted (as appropriate) on the independent test dataset (2015) and compare to that of the training dataset.

Choosing a Machine Learning Method

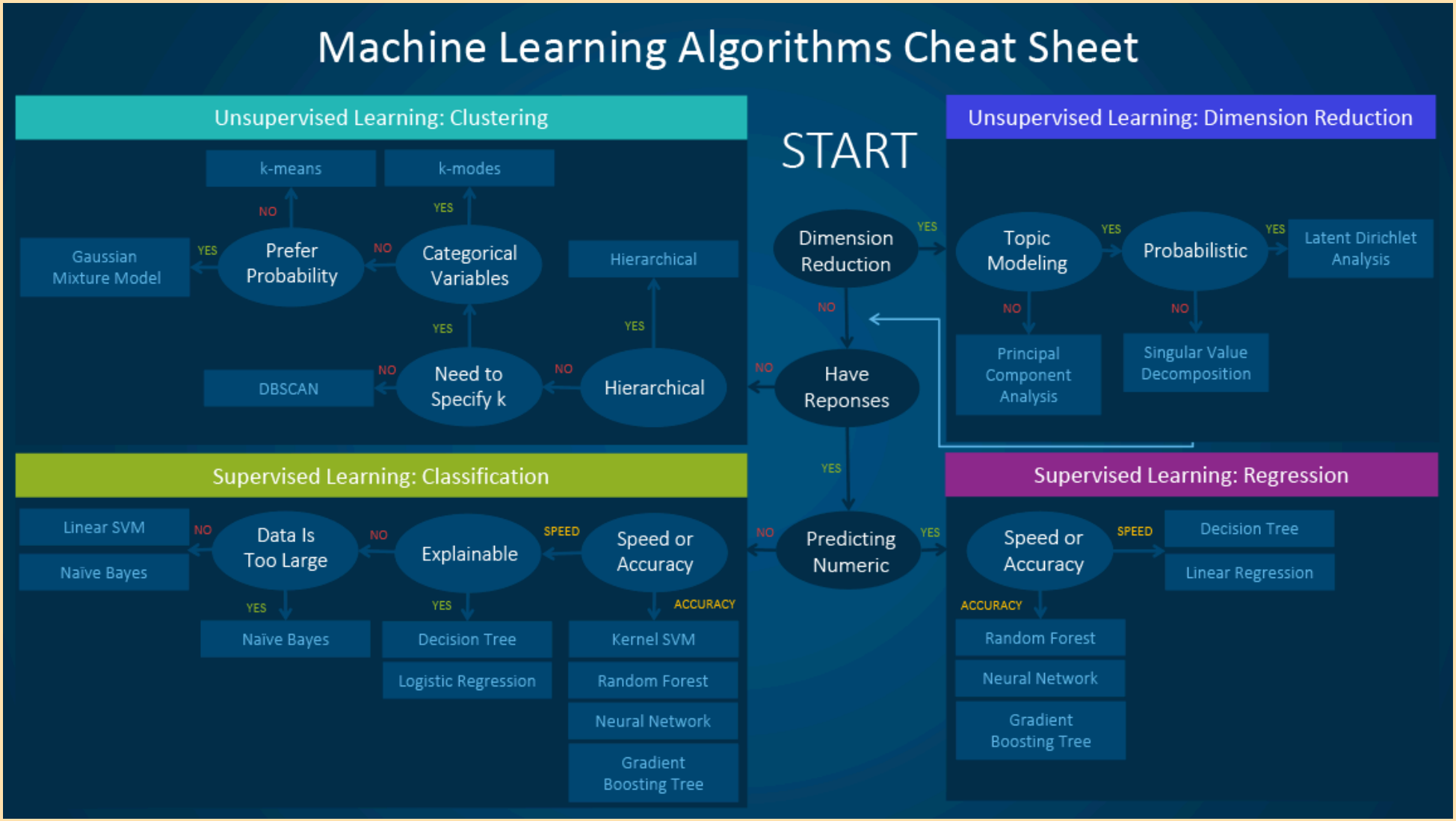
- Common question: “Which machine learning algorithm should I use?”
- Answer: It depends,
  1. Do you have supervised or unsupervised data?
  2. If supervised, do you have continuous, binary class, or multi-class outcome(s)?
  3. If unsupervised, are you interested in looking for latent groupings or data reduction?
  4. Do you have predictors that are continuous, categorical, ordinal, text-based, or some combination of the above?
  5. How large is your training set?
  6. Do you need a fast algorithm?
  7. Do you have time to explore multiple approaches?
  8. Do you need an interpretable model?
  9. For classification, do you need class probabilities, or will “hard” classes suffice?
  10. Do you need to deal with missing data?

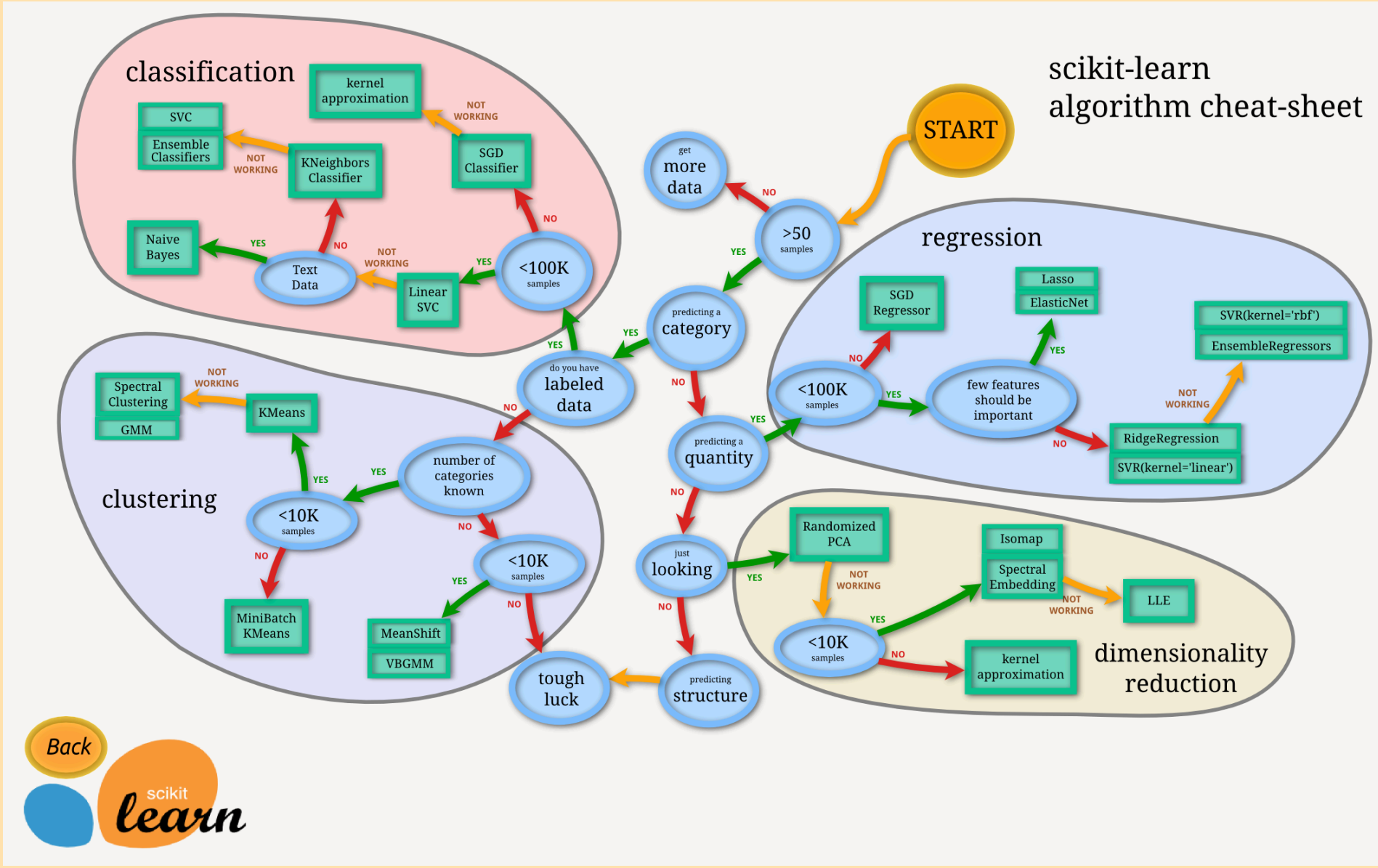
1. Do you have supervised or unsupervised data?
   - Supervised data → use classification or regression methods
   - Unsupervised data → use clustering or data reduction algorithms
2. If supervised, do you have continuous, binary class, or multi-class outcomes?
   - Continuous data → use regression methods (e.g. multiple linear regression)
   - Class data → use classification methods (e.g. recursive partitioning regression trees); possible choices of method will be more limited for problems with more than 2 classes for the outcome.
3. If unsupervised, are you interested in looking for natural groupings, or data reduction/simplification?
   - To group data into similar sets → clustering algorithms, e.g. K-nearest neighbors (kNN)
   - To reduce data → use data reduction algorithms (e.g. principal components analysis)

4. Do you have predictors that are continuous, categorical, ordinal, text-based, or some combination of the above?
   - Some algorithms will not work with certain types of predictors
5. How large is your training set?
   - Small data → "high bias/low variance" methods (e.g. Naïve Bayes); higher level of assumptions – converges faster if assumptions hold
   - Large data → "low bias/high variance" methods (e.g. kNN); lower level of assumptions – over-fits with small datasets, but lower asymptotic error (i.e. less error for super-large datasets)
6. Do you need a fast algorithm? (e.g. for real-time work)
   - More complex algorithms can be relatively slow, especially if there are algorithm parameters to be optimized (e.g. Support Vector Machines) – not likely to be an issue in most medicine datasets
7. Do you have time to explore multiple approaches?
   - If so, you may as well try many and see which does "best" for your data (making sure you are dutifully evaluating on independent data) or use ensemble methods

8. Do you need an interpretable model?
9. For classification, do you need class probabilities, or will “hard” classes suffice?
    - Some classification methods don’t provide class probabilities, so if you want probabilities your options are less
10. Do you have missing data?
    - Some classification and regression methods do a better job of dealing with missing data – e.g. classification and regression trees

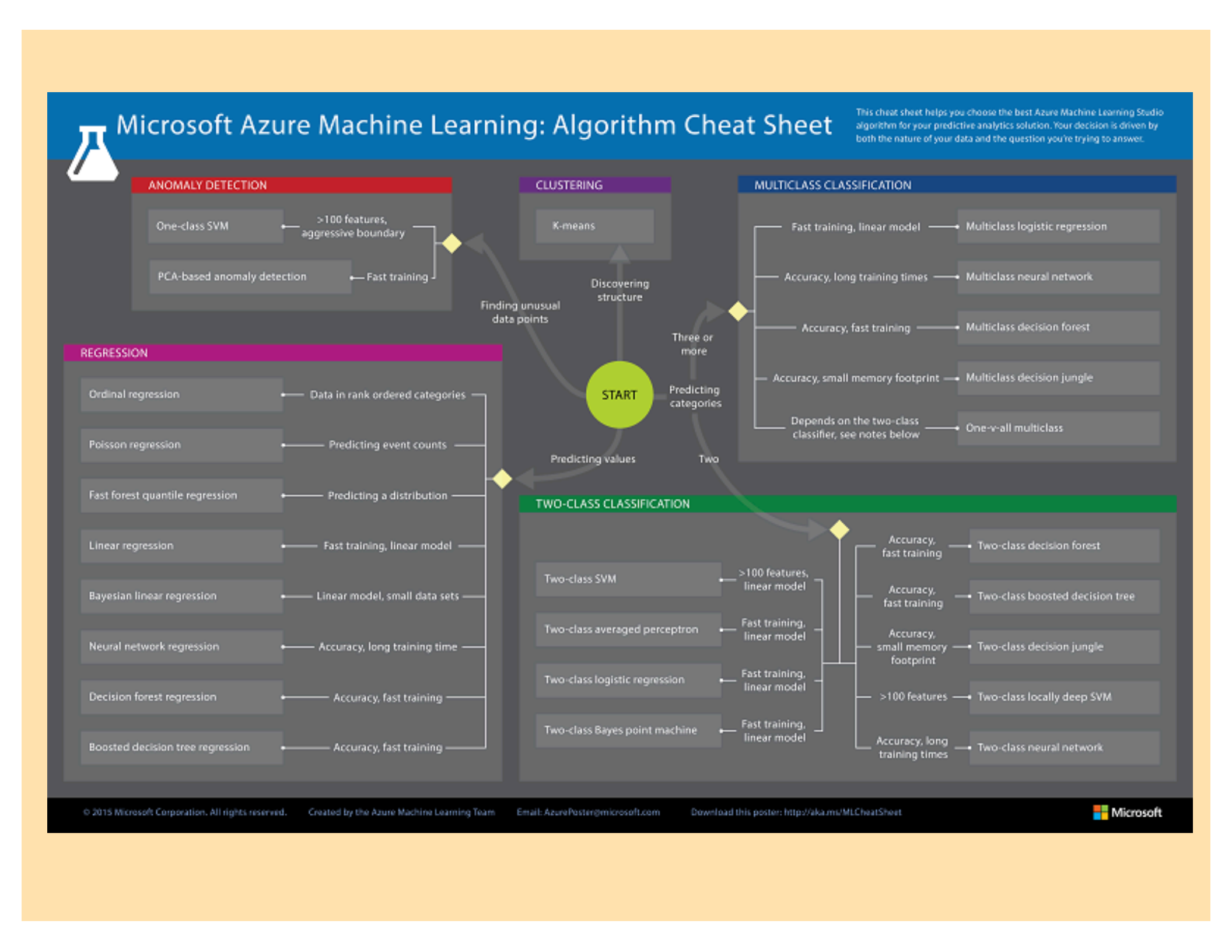
There are also many “cheat sheets” for choosing machine learning algorithms on the internet, but you can see there is a lot of variability…


Machine Learning Algorithms Cheat Sheet
START
Unsupervised Learning: Clustering
k-means
k-modes
NO
YES
Gaussian Mixture Model
YES
Prefer Probability
NO
Categorical Variables
Hierarchical
YES
YES
DBSCAN
NO
Need to Specify k
NO
Hierarchical
Dimension Reduction
YES
NO
Have Reponses
NO
YES
Unsupervised Learning: Dimension Reduction
Topic Modeling
YES
Probabilistic
YES
Latent Dirichlet Analysis
NO
NO
Principal Component Analysis
Singular Value Decomposition
Supervised Learning: Classification
Linear SVM
NO
Data Is Too Large
NO
Explainable
SPEED
Speed or Accuracy
Naïve Bayes
YES
YES
ACCURACY
Naïve Bayes
Decision Tree
Logistic Regression
Kernel SVM
Random Forest
Neural Network
Gradient Boosting Tree
NO
Predicting Numeric
YES
Supervised Learning: Regression
Speed or Accuracy
SPEED
Decision Tree
Linear Regression
ACCURACY
Random Forest
Neural Network
Gradient Boosting Tree


scikit-learn
algorithm cheat-sheet
START
>50 samples
NO
get more data
YES
predicting a category
YES
NO
do you have labeled data
YES
NO
predicting a quantity
YES
NO
just looking
YES
NO
predicting structure
tough luck
classification
kernel approximation
NOT WORKING
SVC
Ensemble Classifiers
NOT WORKING
KNeighbors Classifier
SGD Classifier
NO
Naive Bayes
YES
Text Data
NO
NOT WORKING
Linear SVC
YES
<100K samples
YES
regression
SGD Regressor
Lasso
ElasticNet
SVR(kernel='rbf')
EnsembleRegressors
NO
<100K samples
YES
few features should be important
YES
NOT WORKING
NO
RidgeRegression
SVR(kernel='linear')
clustering
Spectral Clustering
GMM
NOT WORKING
KMeans
YES
number of categories known
YES
NO
<10K samples
NO
MiniBatch KMeans
<10K samples
YES
NO
MeanShift
VBGMM
dimensionality reduction
Randomized PCA
YES
NOT WORKING
Isomap
Spectral Embedding
NOT WORKING
LLE
YES
<10K samples
NO
kernel approximation
Back
scikit
learn

Microsoft Azure Machine Learning: Algorithm Cheat Sheet

This cheat sheet helps you choose the best Azure Machine Learning Studio algorithm for your predictive analytics solution. Your decision is driven by both the nature of your data and the question you're trying to answer.



Created by the Azure Machine Learning Team Email: AzurePoster@microsoft.com
Download this poster: http://aka.ms/MLCheatSheet

Microsoft

Our own quick guide

Supervised learning methods

Interpretability	Model type	Outcome type: Numeric	Binary	Category
	Regression	Stepwise linear regression	Stepwise logistic regression	Stepwise multinomial regression
More		LASSO linear regression	LASSO logistic regression	LASSO multinomial regression
	Bayes	--	Naïve Bayes	Naïve Bayes
			Tree augmented naïve Bayes	Tree augmented naïve Bayes
		--	Bayesian Networks	Bayesian Networks
	Tree	Regression tree	Classification tree	Classification tree
		Averaged regression tree (bagging, boosting, random forest)	Averaged classification tree (bagging, boosting, random forest)	Averaged classification tree (bagging, boosting, random forest)
	Other	--	Linear support vector machine*	--
Less		--	Support vector machine*	--
		--	Neural Network*	--

*Only gives predicted categories, not predicted outcome probabilities

Ensemble Methods

- Run a set of classifiers and allow them to “vote” on predictions.
- **Advantage:**
  - improvement in predictive accuracy.
- **Disadvantages:**
  - more computation
  - it is difficult to understand an ensemble of classifiers
- Examples: bagging, boosting, and stacking

Bagging

- Simplest way of combining predictions that belong to the same type.
- Combine via voting or averaging
- Each model receives equal weight
- Basic idea of bagging:
  - Re-sample several training sets of size n from the dataset (instead of just just using the one training set of size n)
  - Build a classifier for each training set
  - Combine the classifier's predictions (majority vote)
- This improves performance in almost all cases if learning scheme is *unstable* (e.g. decision trees depend heavily on where the first cut occurs – random forests)

Boosting

- Also uses voting/averaging but models are weighted based on their performance
- Iterative procedure: new models are influenced by performance of previously built ones
  - New model is encouraged to become expert for instances classified incorrectly by earlier models
  - Intuitive justification: models should be experts that complement each other
- There are many variants of boosting, one of the most celebrated is *AdaBoost*

Stacking

- Uses a "*meta-learner*" instead of voting to combine predictions across models
  - Predictions of initial models (*level-0 models*) are used as input for meta-learner (*level-1 model)*
- Individual initial models can be of many different types
- Complex methodology to build meta-learner by learning again from the initial set of models (machine learning on top of machine learning)

On Friday, May 17th, there will be a Friday Science Journal Club meeting.

We will discuss a recent opinion piece touting the success of machine learning to design personalized diets and the underlying scientific paper.

https://www.nytimes.com/2019/03/02/opinion/sunday/diet-artificial-intelligence-diabetes.html

https://www.cell.com/cell/fulltext/S0092-8674(15)01481-6#secsectitle0025

Cheese and fruit will be served. We hope to see you there!

When & Where

Friday, May 17th

2:00 – 3:00 pm

2nd Floor - Room 2700

Mission Hall

550 16th Street

San Francisco, CA 94158