

# Big Data Engineering

## Introduction to Machine Learning

Julie Weeds

March 2019



© Paul Fremantle 2015. This work is licensed under a Creative Commons  
Attribution-NonCommercial-ShareAlike 4.0 International License  
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

# Contents

- Definitions and terminology
- The overall process
- Main techniques
- Algorithms and examples
- Big Data Machine Learning
- Spark MLLib



© Paul Fremantle 2015. This work is licensed under a Creative Commons  
Attribution-NonCommercial-ShareAlike 4.0 International License  
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

# Definition of Machine Learning

- Algorithms that can learn from data



© Paul Fremantle 2015. This work is licensed under a Creative Commons  
Attribution-NonCommercial-ShareAlike 4.0 International License  
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

# Definition of Machine Learning

- Algorithms that can learn from data

Ok that was a circular definition 😊



© Paul Fremantle 2015. This work is licensed under a Creative Commons  
Attribution-NonCommercial-ShareAlike 4.0 International License  
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

# Definition take 2

- Algorithms that can analyse a set of data to find patterns and then make predictions when new data comes in



© Paul Fremantle 2015. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License  
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

# Uses of Machine Learning

- Fraud Detection
  - Spam emails, fake reviews, credit card fraud
- Personalization
  - Recommendations
- Targeted Marketing
  - Predictive preferences, cross-selling
- Content Classification
  - Document classification, sentiment analysis
- Customer Support
  - Social media analysis
- Many others



# Learning phase

AKA **encode** phase or **training** phase or **model fitting** phase



# Usage phase

AKA **decode** phase or **testing** phase or **predicting** phase

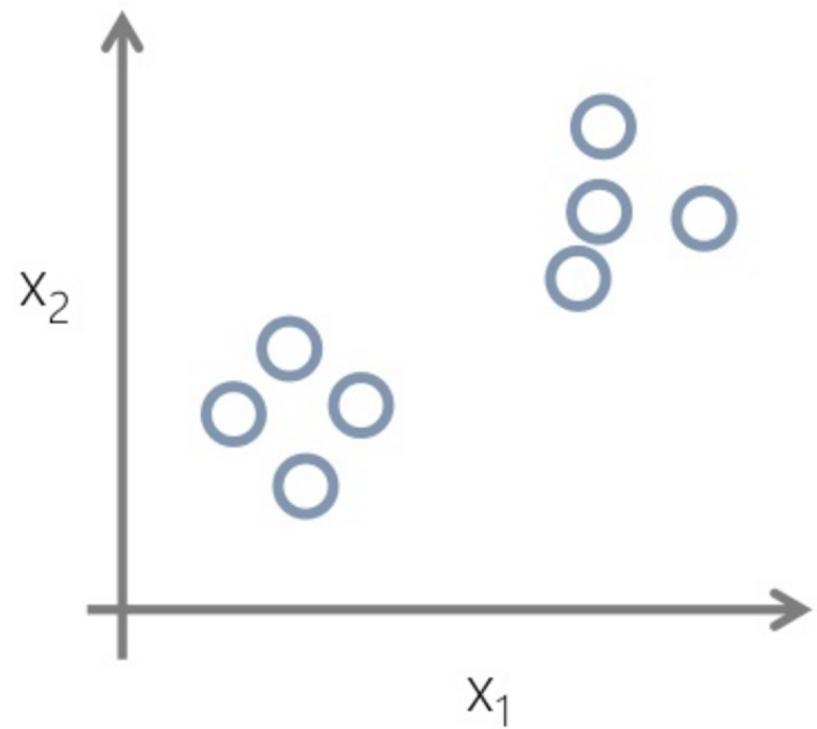
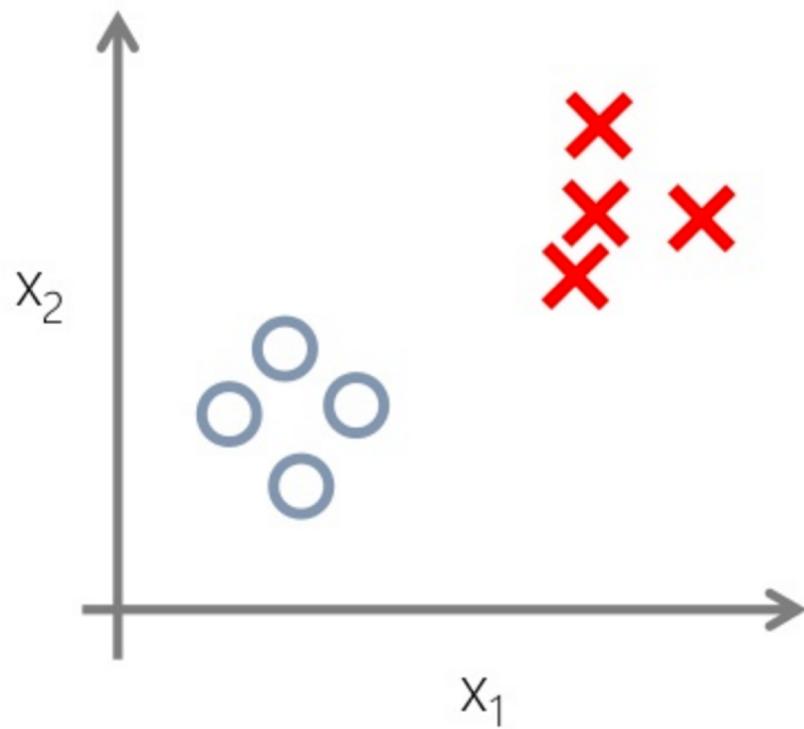


# Terminology

- Sample
  - Some incoming data to be analysed
  - E.g. a JPG picture
- Feature
  - Some quantifiable data from the sample
  - E.g. colour, height, width, pixel data, etc
- Label
  - Some useful information about the sample that we wish to categorise:
    - E.g. looking at a picture this is a person
- Model
  - The output of some learning algorithm
  - The parameterization of an algorithm that can be run against new data



# Supervised vs Unsupervised



© Paul Fremantle 2015. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License  
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

Source: <http://www.slideshare.net/damirdobric>

# Types of learning

- Supervised
  - The required labels are known
  - Aiming to find an algorithm that correctly identifies these
  - Iterative exploration and refinement
  - Useful for prediction
- Unsupervised
  - The labels are not known
  - The system identifies new classifications
  - Exploring the past, better understanding it
- Reinforcement
  - Learning as you go
  - E.g. learning to play chess while playing chess



# Types of machine learning

- Classification
- Regression / Prediction
- Clustering
- Recommendation and Collaborative Filtering
- Frequent Pattern mining



# Classification

birds

Jul 27

May 13

▼

▼

BY NC SA See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

The image shows a user interface for a mobile application designed for identifying birds. At the top, there is a search bar with the word "birds" typed into it. To the right of the search bar is a close button (an "X"). Below the search bar is a horizontal scroll bar. The main content area displays several images of birds. On the far left, there are two images of birds in flight against a dark, cloudy sky. The first of these has a checkmark icon in its top-left corner. To the right of these are two images of seagulls perched on wooden posts, with a scenic background of water and distant land under a cloudy sky. To the right of these four images is a fifth image of a swallow perched on a wooden post in a grassy field. To the right of the swallow image is a sixth image of a pheasant standing on a low wall. At the bottom of the screen, there is a large, semi-transparent gray rectangular overlay. The overall layout suggests a grid-based image viewer with a focus on bird photography.

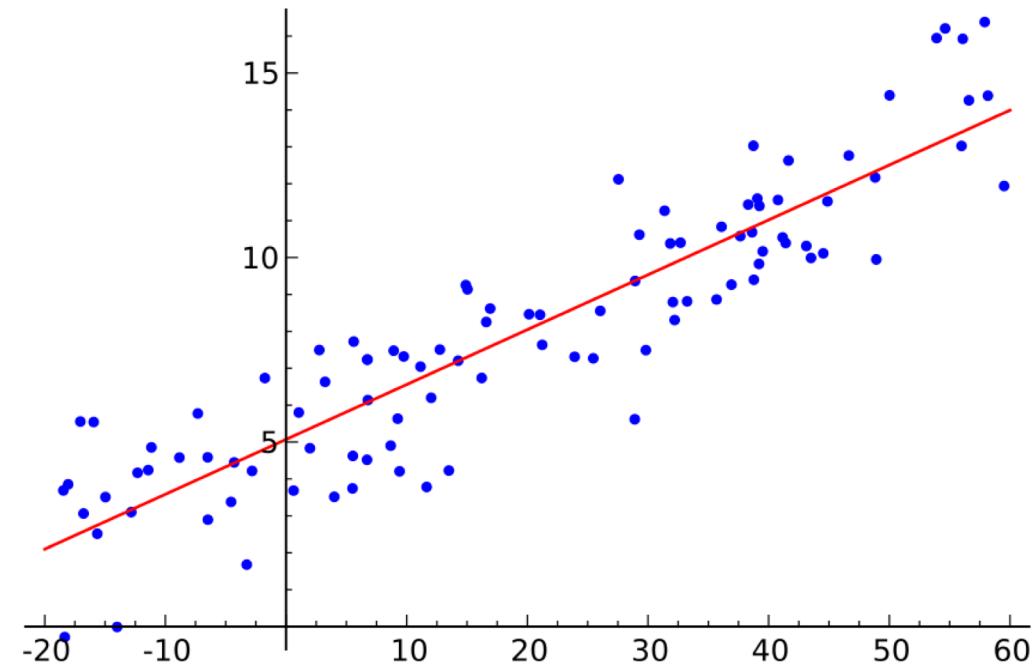
# Classification

- Identifying a class into which this sample fits
  - E.g. look at a picture and decide if it contains a bird
  - A key part of artificial intelligence
  - Also deeply useful for making sense of big data



# Regression

- Applying a model based on previous data
  - Allows prediction of future state
- Many statistical techniques

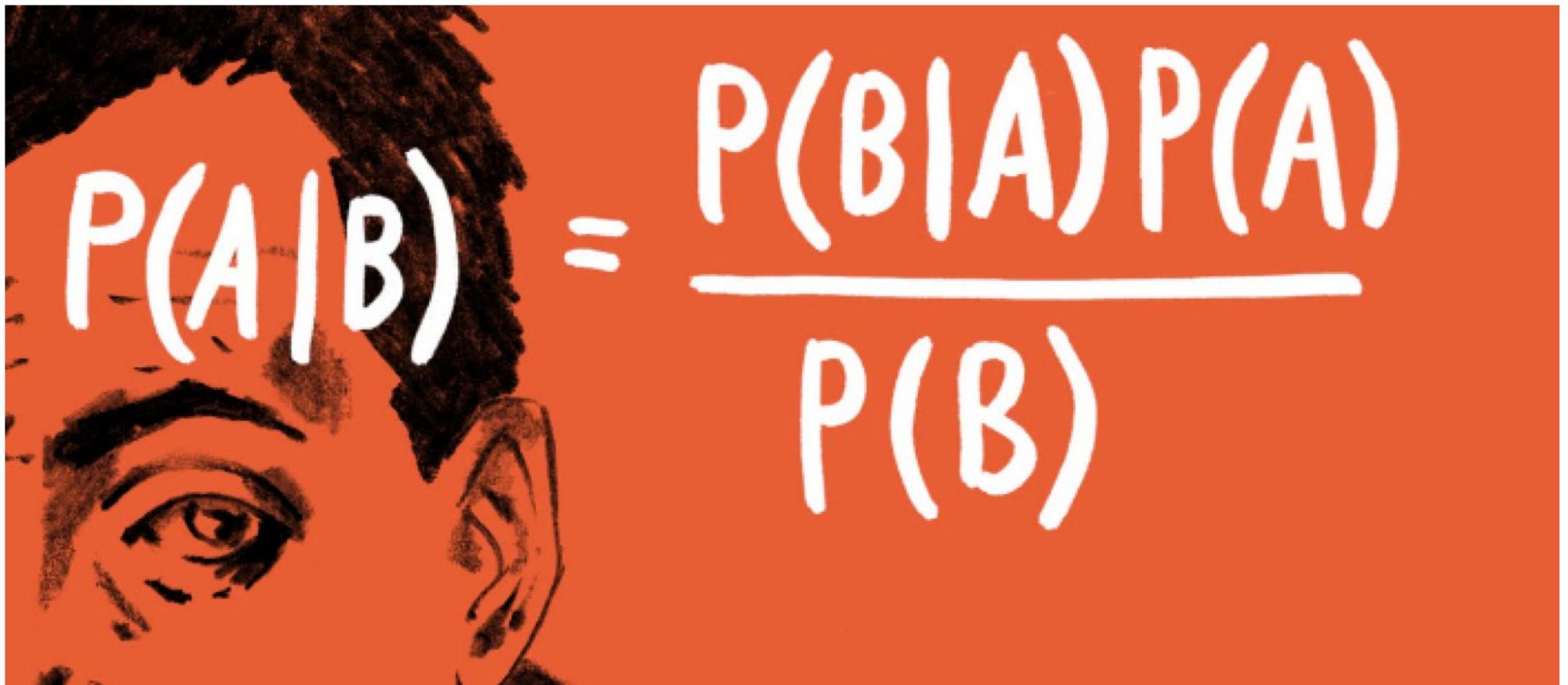


# Regression vs Classification

- Regression produces a real number or numbers
  - i.e. a continuously varying answer or answers
- Classification identifies a set or element of a set
  - E.g. False, Blue, Person, High-Value Customer



# Bayes Theorem



$P(A|B)$  is the probability of A given B

$P(A)$  is the probability of A without regard to B

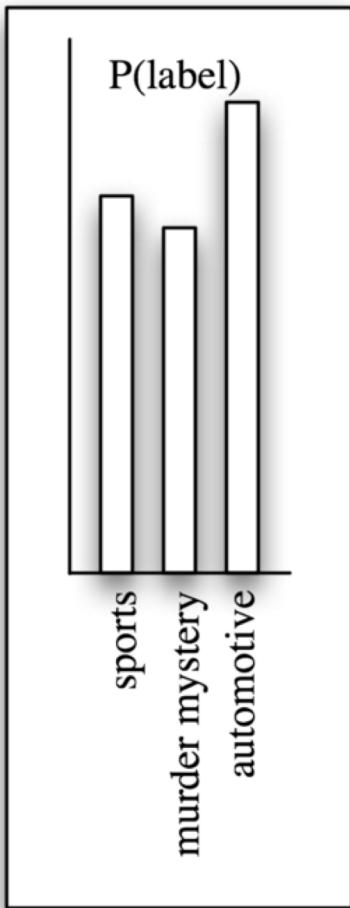


© Paul Fremantle 2015. This work is licensed under a Creative Commons  
Attribution-NonCommercial-ShareAlike 4.0 International License  
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

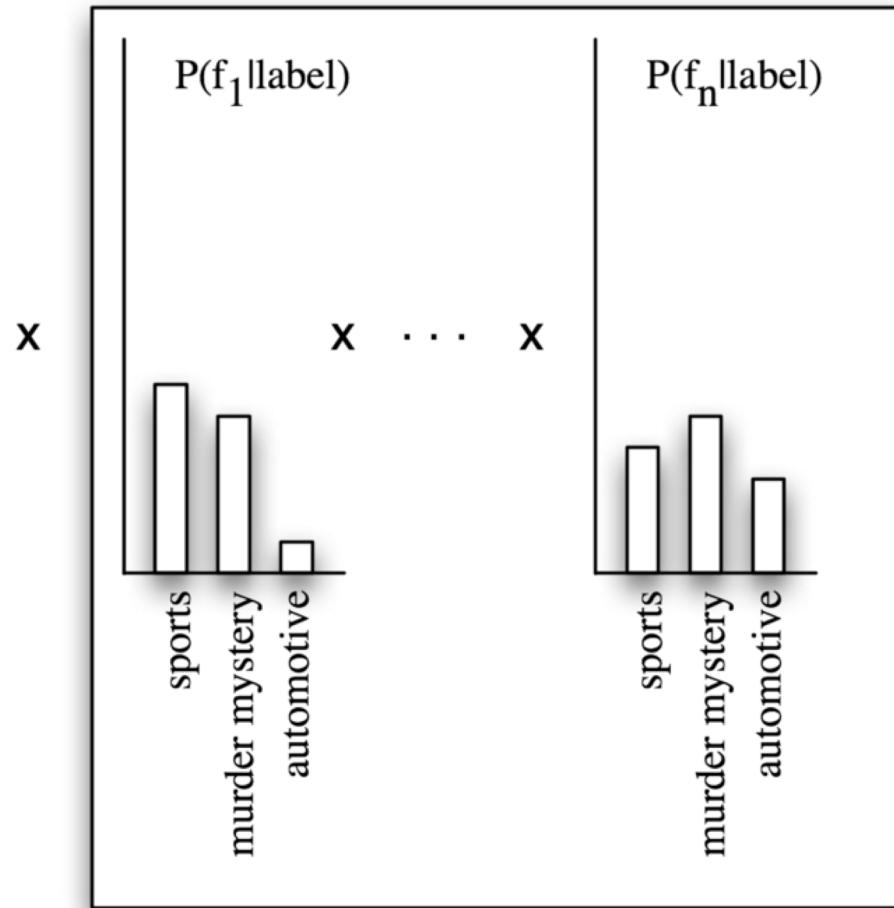
# Classification Algorithms

## Naïve Bayes

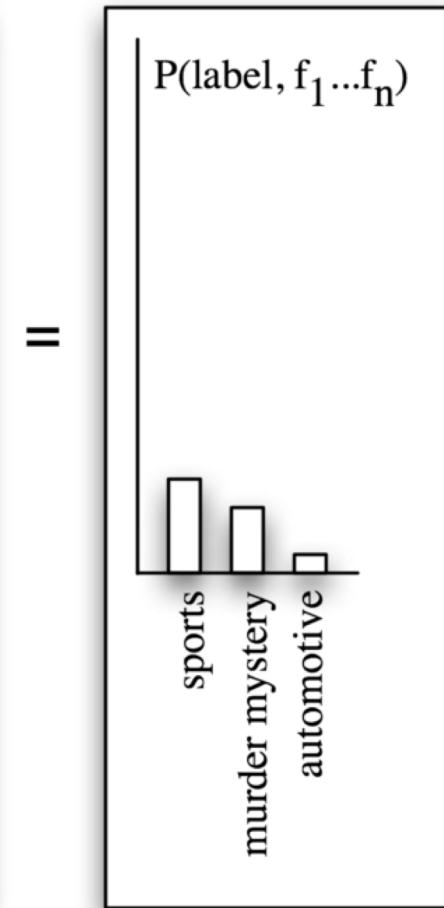
Prior Probabilities



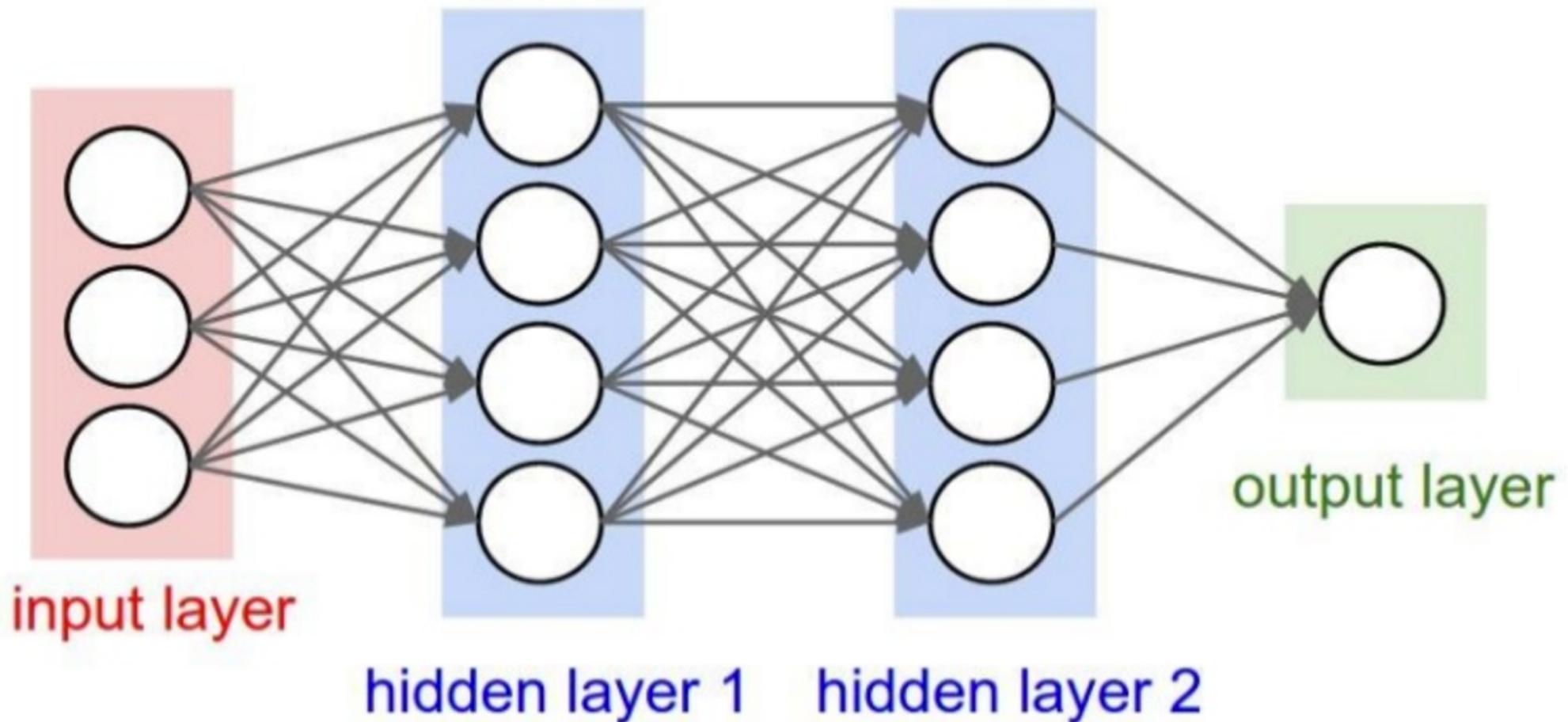
Feature Contributions



Label Likelihoods



# Deep Learning



# Google Tensor Flow

## TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems (Preliminary White Paper, November 9, 2015)

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng

Google Research\*

### Abstract

TensorFlow [1] is an interface for expressing machine learning algorithms, and an implementation for executing such algorithms. A computation expressed using TensorFlow can be executed with little or no change on a wide variety of heterogeneous systems, ranging from mobile devices such as phones and tablets up to large-scale distributed systems of hundreds of machines and thousands of computational devices such as GPU cards. The system is flexible and can be used to express a wide variety of algorithms, including training and inference algorithms for deep neural network models, and it has been used for conducting research and for deploying machine learning systems into production across more than a dozen areas of computer science and other fields, including speech recognition, computer vision, robotics, information retrieval, natural language processing, geographic information extraction, and computational drug discovery. This paper describes the TensorFlow interface and an implementation of that interface that we have built at Google. The TensorFlow API and a reference implementation were released as an open-source package under

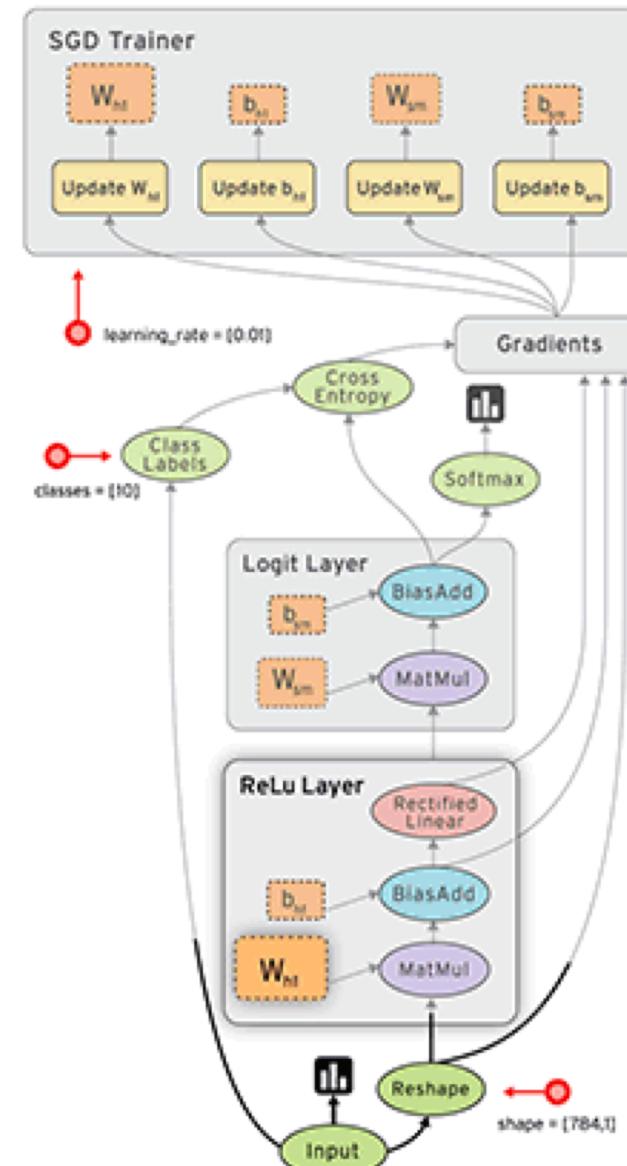
sequence prediction [47], move selection for Go [34], pedestrian detection [2], reinforcement learning [38], and other areas [17, 5]. In addition, often in close collaboration with the Google Brain team, more than 50 teams at Google and other Alphabet companies have deployed deep neural networks using DistBelief in a wide variety of products, including Google Search [11], our advertising products, our speech recognition systems [50, 6, 46], Google Photos [43], Google Maps and StreetView [19], Google Translate [18], YouTube, and many others.

Based on our experience with DistBelief and a more complete understanding of the desirable system properties and requirements for training and using neural networks, we have built TensorFlow, our second-generation system for the implementation and deployment of large-scale machine learning models. TensorFlow takes computations described using a dataflow-like model and maps them onto a wide variety of different hardware platforms, ranging from running inference on mobile device platforms such as Android and iOS to modest-



# TensorFlow

- Announced and open sourced in Nov 2015
  - Strong adoption in the meantime
- CPU and GPU support with no coding changes
- Neural networks plus arbitrary dataflows
- [www.tensorflow.org](http://www.tensorflow.org)



# Spark MLLib's algorithms

Problem Type	Supported Methods
Binary Classification	linear SVMs, logistic regression, decision trees, random forests, gradient-boosted trees, naive Bayes
Multiclass Classification	decision trees, random forests, naive Bayes
Regression	linear least squares, Lasso, ridge regression, decision trees, random forests, gradient-boosted trees, isotonic regression



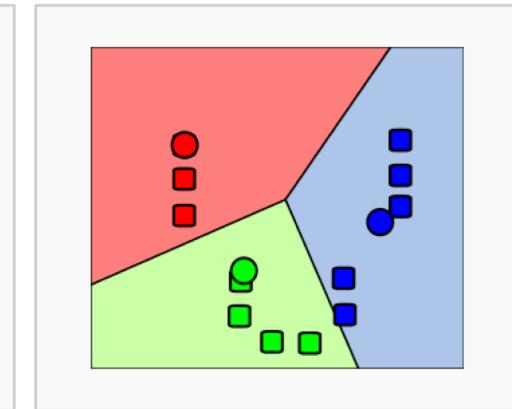
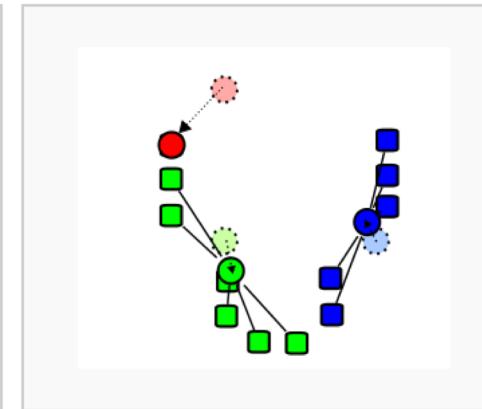
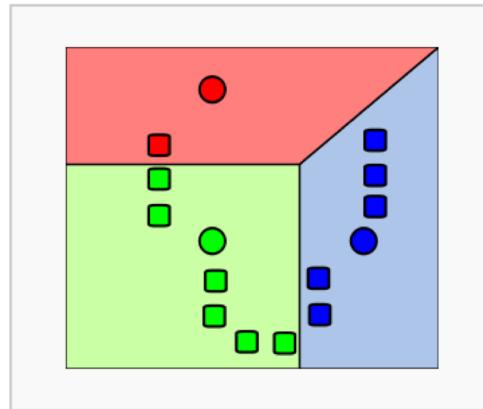
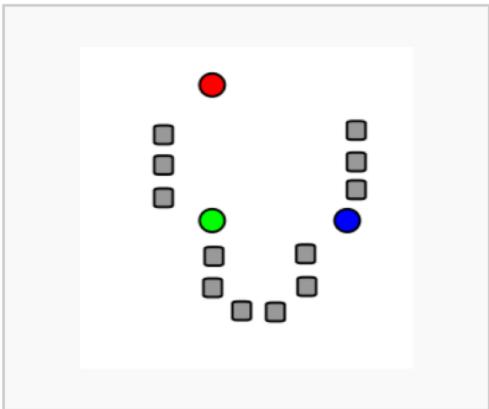
# Clustering

- Grouping items into clusters
  - Where items in a cluster are more similar to each other than to items in other clusters
- Basically creating the classifications from the data rather than applying them a priori



# K-Means Clustering

Demonstration of the standard algorithm



1.  $k$  initial "means" (in this case  $k=3$ ) are randomly generated within the data domain (shown in color).

2.  $k$  clusters are created by associating every observation with the nearest mean. The partitions here represent the **Voronoi diagram** generated by the means.

3. The **centroid** of each of the  $k$  clusters becomes the new mean.

4. Steps 2 and 3 are repeated until convergence has been reached.



# MLLib's clustering

- K-means
- Gaussian mixture
- Power iteration clustering (PIC)
- Latent Dirichlet allocation (LDA)
- Streaming k-means



© Paul Fremantle 2015. This work is licensed under a Creative Commons  
Attribution-NonCommercial-ShareAlike 4.0 International License  
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

# Recommendation and Collaborative Filtering

- Given a user's interaction with items, what else are they likely to prefer

## Large-scale Parallel Collaborative Filtering for the Netflix Prize

Yunhong Zhou, Dennis Wilkinson, Robert Schreiber and Rong Pan

HP Labs, 1501 Page Mill Rd, Palo Alto, CA, 94304  
{yunhong.zhou, dennis.wilkinson, rob.schreiber, rong.pan}@hp.com

**Abstract.** Many recommendation systems suggest items to users by utilizing the techniques of collaborative filtering (CF) based on historical records of items that the users have viewed, purchased, or rated. Two major problems that most CF approaches have to resolve are scalability and sparseness of the user profiles. In this paper, we describe *Alternating-Least-Squares with Weighted- $\lambda$ -Regularization* (ALS-WR), a parallel algorithm that we designed for the Netflix Prize, a large-scale collaborative filtering challenge. We use parallel Matlab on a Linux cluster

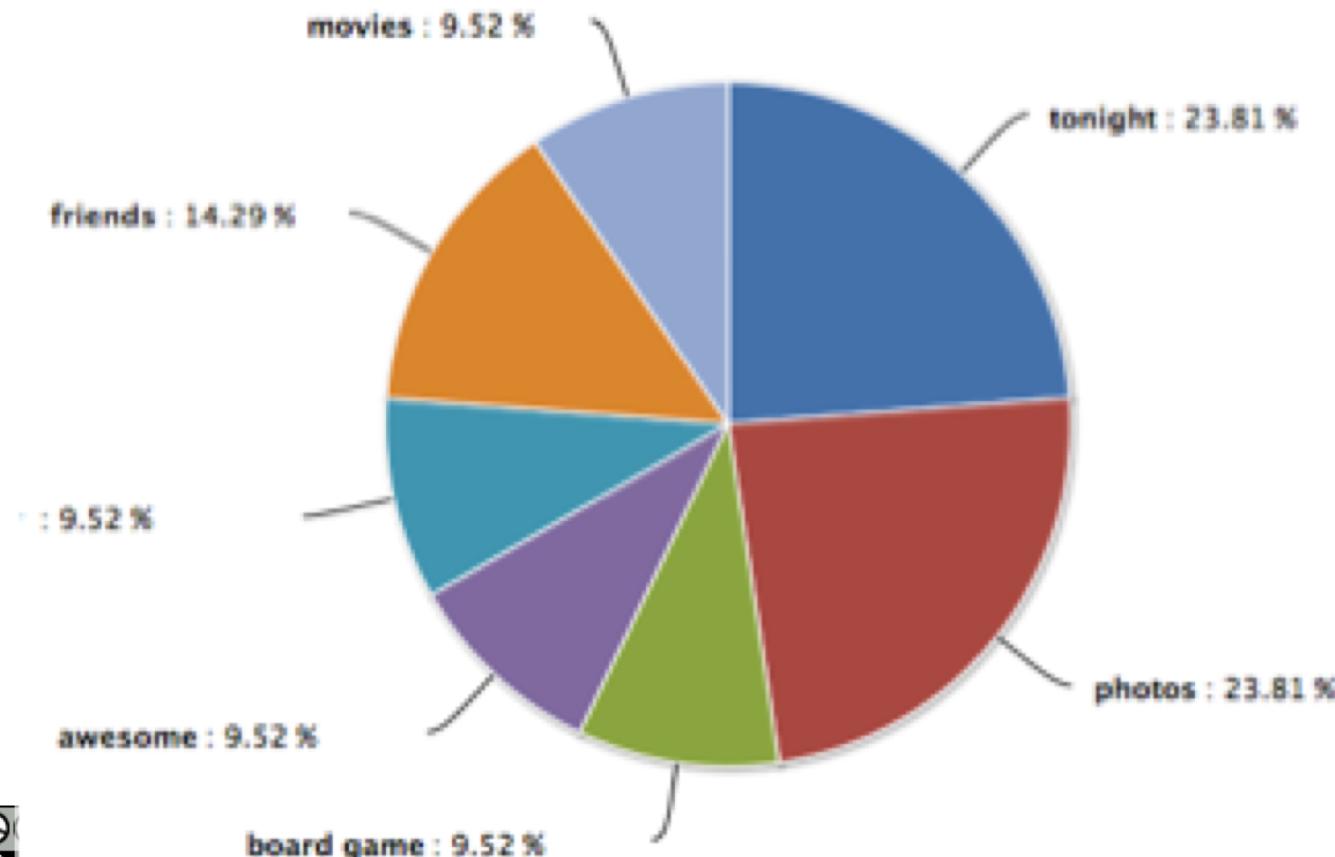


# Frequent Pattern Mining

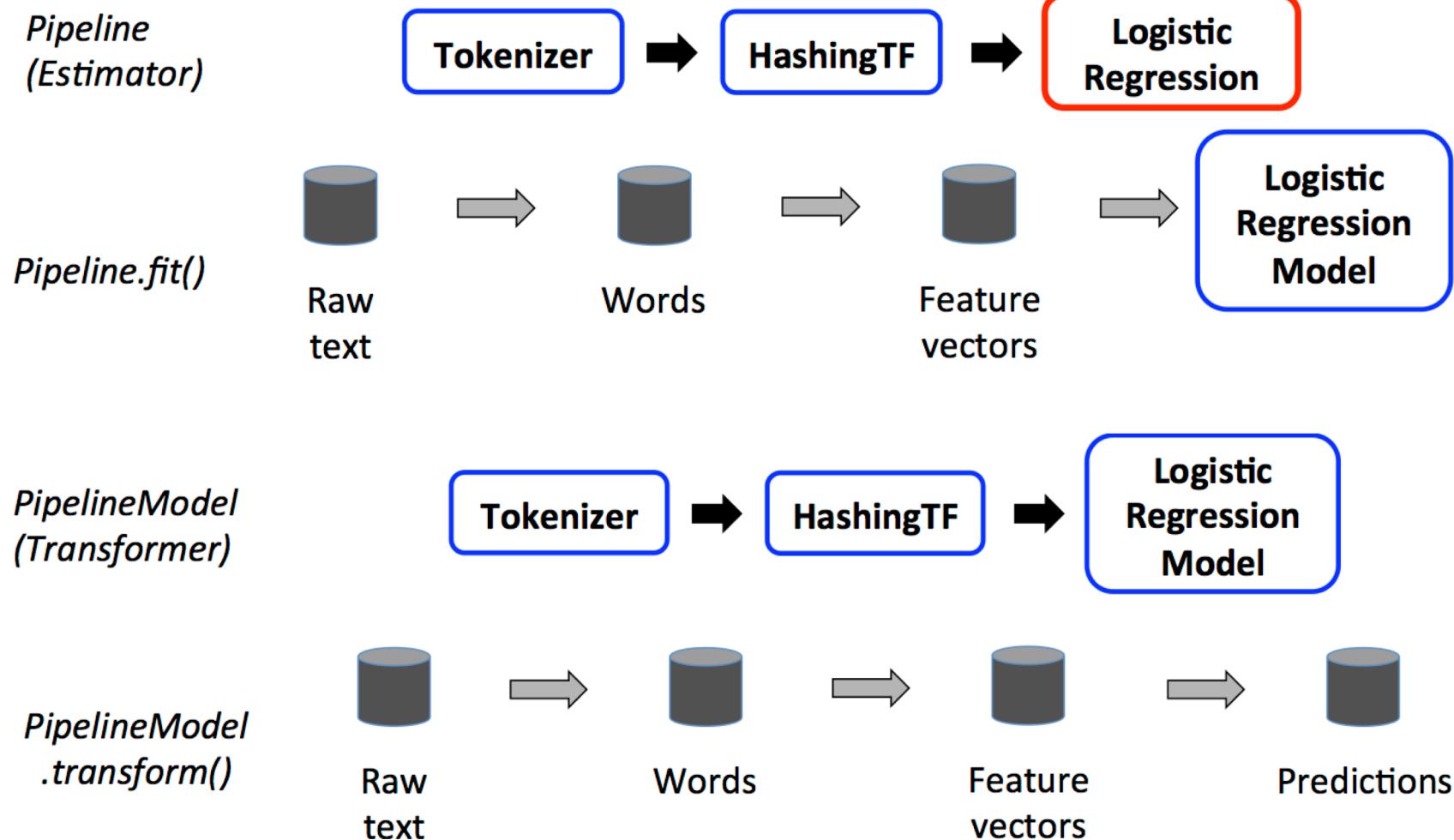
Related topics:  @cassiomelo Your last post was about a hangout. These are the topics you relate to hangout: [tonight](#), [movies](#), [board game](#), [friends](#), [awesome](#), [photos](#), [bar](#) and [NBA](#).



Related words for: "hangout"



# Spark MLLib Pipelines



TRAINING

USAGE



© Paul Fremantle 2015. This work is licensed under a Creative Commons  
Attribution-NonCommercial-ShareAlike 4.0 International License  
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

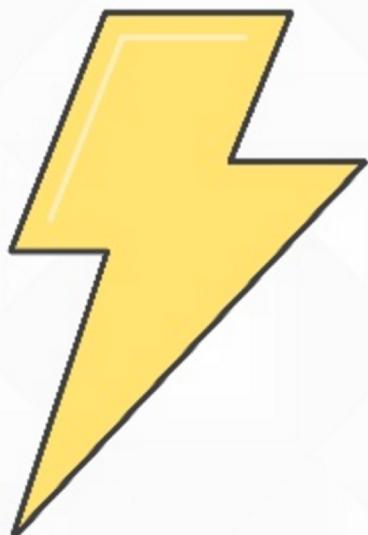
# Big Data ML

- Obviously we can learn more insights with more data
- Many examples
  - Netflix competition (2009), Kaggle competitions
  - Google, Facebook, Twitter etc are all doing big data ML
- Obviously we want the right algorithms:
  - E.g. Kmeans++ is a parallelizable version of Kmeans
- MLLib and Mahout come pre-built with these



# Amazon Machine Learning

## Powerful machine learning technology



Based on Amazon's battle-hardened internal systems

Not just the algorithms:

- Smart data transformations
- Input data and model quality alerts
- Built-in industry best practices

Grows with your needs

- Train on up to 100 GB of data
- Generate billions of predictions
- Obtain predictions in batches or real-time



# Amazon SageMaker

## How it works



Label



Build



Train

Set up and manage labeling jobs for highly accurate training datasets within Amazon SageMaker, using active learning and human labeling

Connect to other AWS services and transform data in Amazon SageMaker notebooks

Use Amazon SageMaker's algorithms and frameworks, or bring your own, for distributed training



Tune



Deploy

Amazon SageMaker automatically tunes your model by adjusting multiple combinations of algorithm parameters

Once training is completed, models can be deployed to Amazon SageMaker endpoints, for real-time predictions



# Benefits of Amazon SageMaker

## Benefits and features

### Labeling raw data with active learning

Amazon SageMaker Ground Truth uses a machine learning model to automatically attempt to label training data. Any data that can't be confidently labeled by the model is automatically sent to human labelers. The human labeled data is provided back to the model so that it can continuously learn and improve.

### Fully-managed notebook instances

For training data exploration and preprocessing, Amazon SageMaker provides fully managed instances running Jupyter notebooks that include example code for common model training and hosting exercises.

### One-click training

When you're ready to train in Amazon SageMaker, simply indicate the type and quantity of instances you need and initiate training with a single click.

### Highly accurate training datasets

Active learning models from Amazon SageMaker Ground Truth provide a very high level of consistency and accuracy for training datasets, without the burden of audits and asking for the same data to be labeled multiple times to remove outliers.

### Highly-optimized machine learning algorithms

Amazon SageMaker installs high-performance, scalable machine learning algorithms optimized for speed, scale, and accuracy, to run on extremely large training datasets.

### Deployment without engineering effort

After training, SageMaker provides the model artifacts and scoring images to you for deployment to Amazon EC2 or anywhere else.



# Recap

- Machine Learning is a powerful way of gaining insight and value from big data
  - Recommendation
  - Classification and prediction
  - Clustering and understanding
- Many coding and deployment options
- Built into Spark, Hadoop and AWS



# Questions?



© Paul Fremantle 2015. This work is licensed under a Creative Commons  
Attribution-NonCommercial-ShareAlike 4.0 International License  
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>