

Introduction to Machine learning

14 September 2017 workshop

Kevin Chai Rebecca Lange

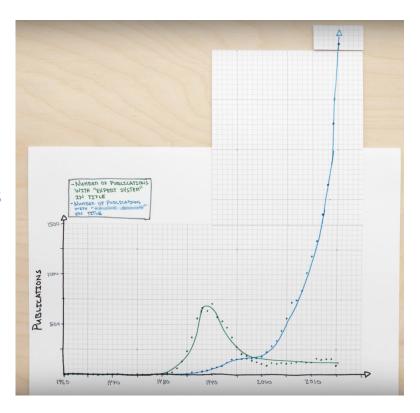
Agenda

- Introduction
- Workshop
 - Classification
 - Regression
 - Clustering
 - o Dimensionality reduction
- Reference sheets
- Deep learning
- Q&A

Machine Learning

 Algorithms that learn from examples and experiences (data)

 Performs better than hard coded rules for complex problems



Source: Welch Labs, Learning to See - Part 3, video series 2016

Examples









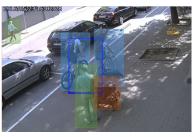


Google Photos



Examples











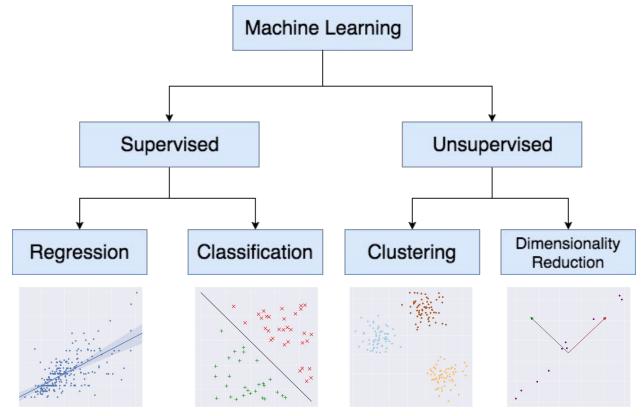






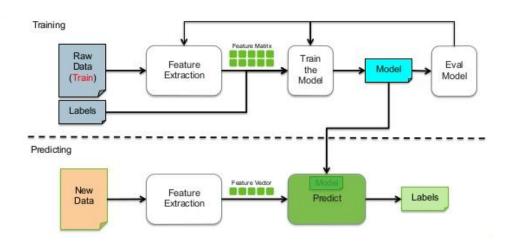


Algorithms



Supervised learning

- Trained with labelled data
- Algorithms learn in a supervised manner



Decision trees

- Classification algorithm
 - Predict discrete (category) outputs
 - Will Shiv play cricket?

Training

- Recursively split data into subsets based on a single attribute
- Stop when all subsets are pure (all yes / no)

Prediction

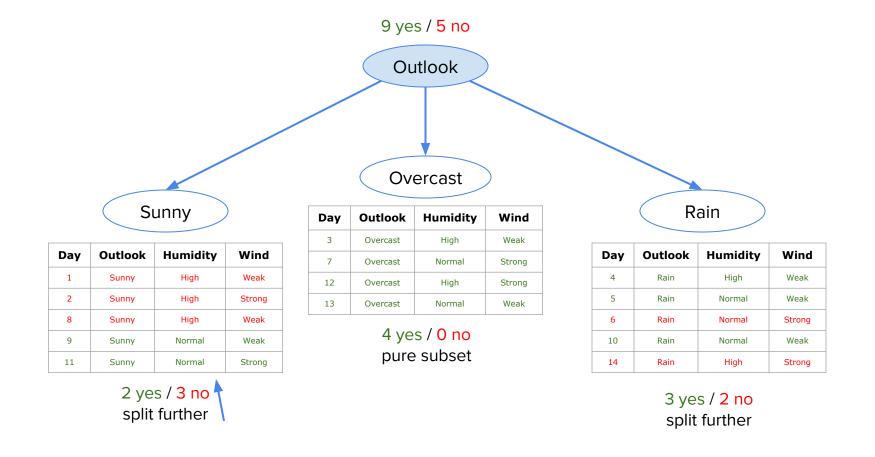
Based on subset where new data is placed

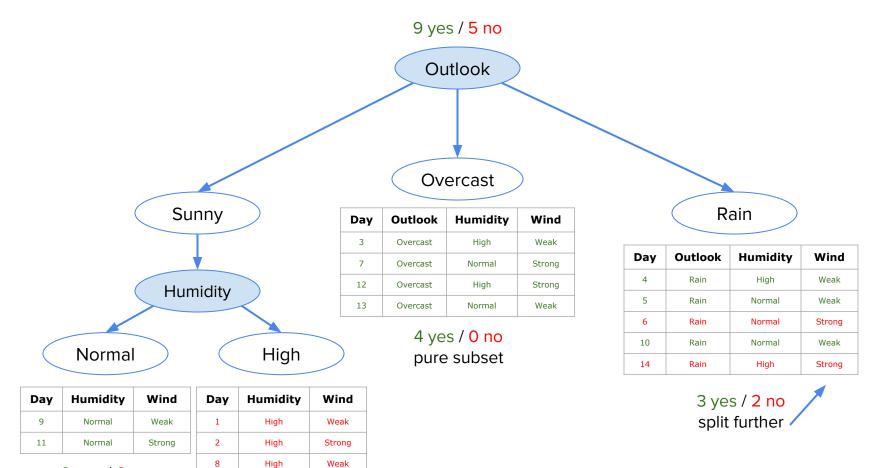
Training data: 9 yes / 5 no

	•	•		
Day	Outlook	Humidity	Wind	Cricket
1	Sunny	High	Weak	No
2	Sunny	High	Strong	No
3	Overcast	High	Weak	Yes
4	Rain	High	Weak	Yes
5	Rain	Normal	Weak	Yes
6	Rain	Normal	Strong	No
7	Overcast	Normal	Strong	Yes
8	Sunny	High	Weak	No
9	Sunny	Normal	Weak	Yes
10	Rain	Normal	Weak	Yes
11	Sunny	Normal	Strong	Yes
12	Overcast	High	Strong	Yes
13	Overcast	Normal	Weak	Yes
14	Rain	High	Strong	No

New data:

15 Rain	High	Weak	?
---------	------	------	---

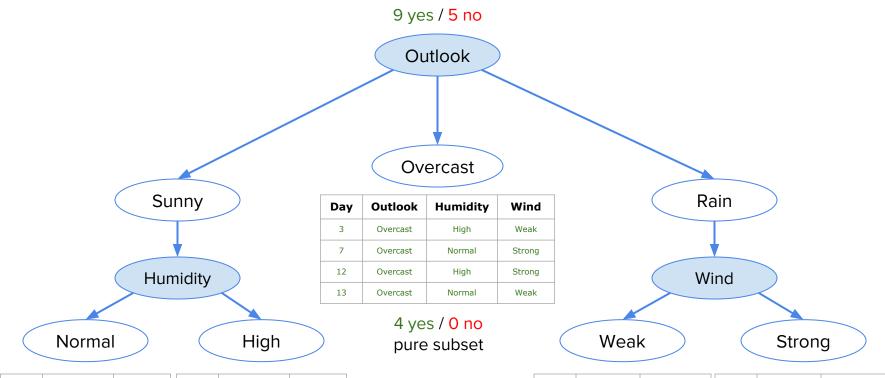




2 yes / 0 no pure subset

0 yes / 3 no pure subset

Source: Victor Lavrenko, Decision Trees, University of Edinburgh 2014.



Day	Humidity	Wind	Day	Humidity	Wind
9	Normal	Weak	1	High	Weak
11	Normal	Strong	2	High	Strong
			8	High	Weak

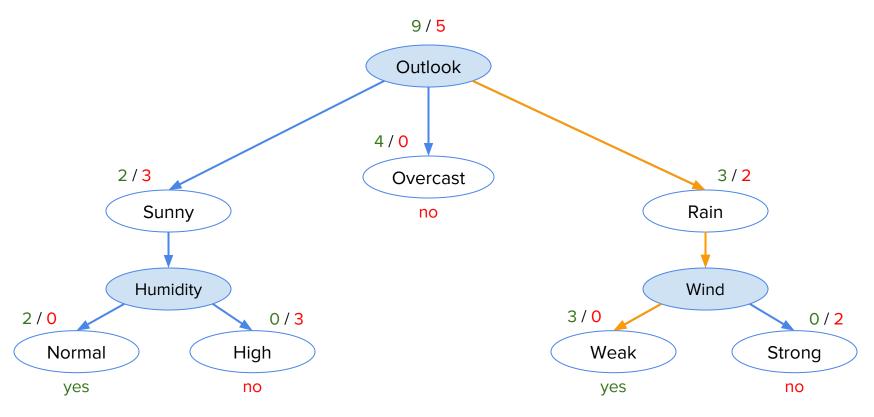
2 yes / 0 no pure subset

0 yes / 3 no pure subset

Day	Outlook	Humidity	Day	Outlook	Humidity
4	Rain	High	6	Rain	Normal
5	Rain	Normal	14	Rain	High
10	Rain	Normal			•

3 yes / 0 no pure subset

0 yes / 2 no pure subset

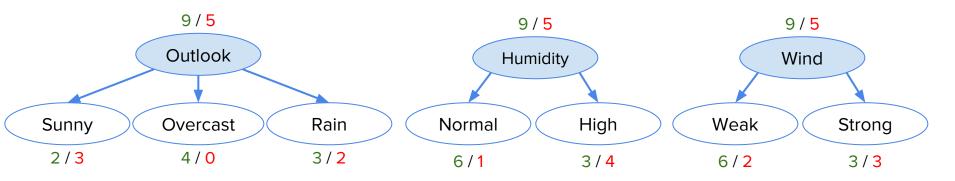


New data:

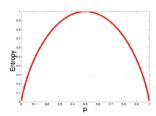
Day	Outlook	Humidity	Wind	Cricket	
15	Rain	High	Weak	?	→ yes

Decision trees

Which attribute provides the best split?



- Split attributes based on a purity measure (e.g. entropy, gini)
- Information gain = average weighted entropy of each subset
- Example: http://www.saedsayad.com/decision_tree.htm



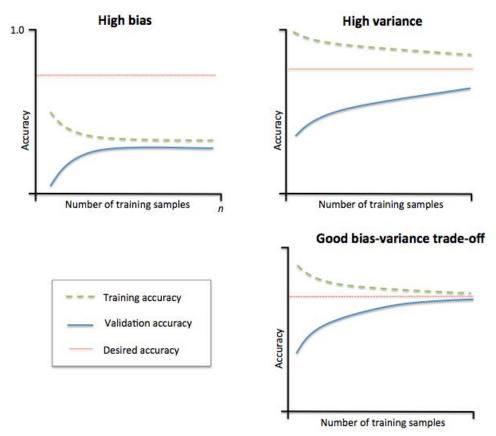
Learning curves

Underfit (high bias)

- Try more features
- Decrease regularisation

Overfit (high variance)

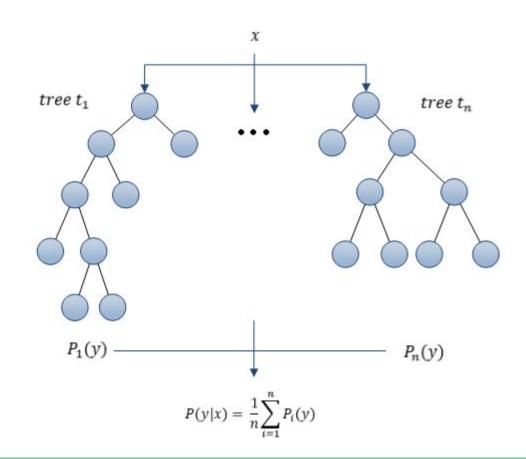
- Get more data
- Use less features
- Increase regularisation



Source: Andrew Ng, Coursera - Machine Learning. 2012

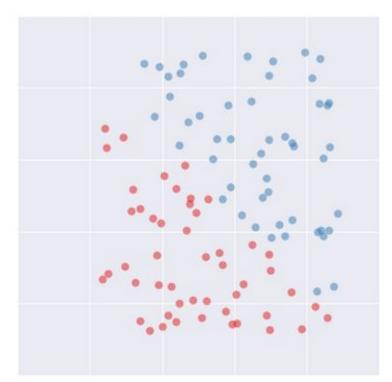
Random forests

- Train multiple decision trees (forest) based on random subsets of:
 - training examples
 - features
- Final prediction is based on the average of individual trees
- Avoids overfitting problems with single decision trees



 Decision trees split one attribute at a time

Greedy algorithm



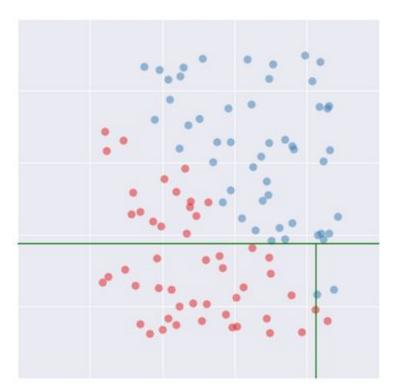
 Decision trees split one attribute at a time

Greedy algorithm



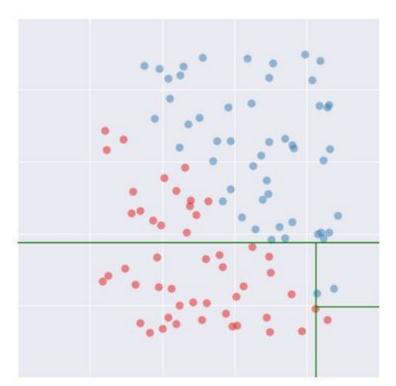
Decision trees
 split one attribute
 at a time

Greedy algorithm



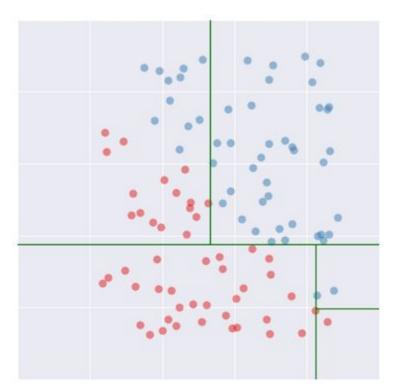
Decision trees
 split one attribute
 at a time

Greedy algorithm



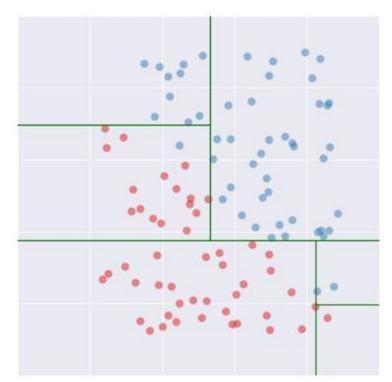
Decision trees
 split one attribute
 at a time

Greedy algorithm



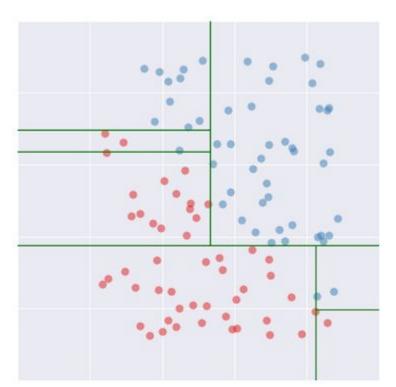
 Decision trees split one attribute at a time

Greedy algorithm



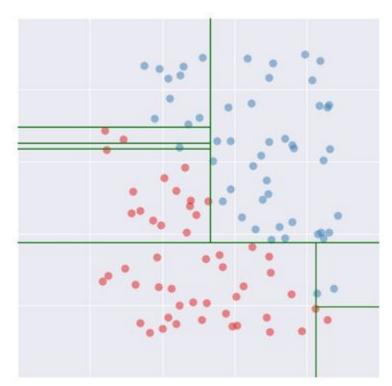
Decision trees
 split one attribute
 at a time

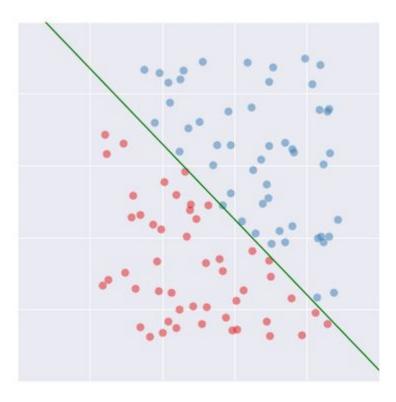
Greedy algorithm



 Decision trees split one attribute at a time

Greedy algorithm

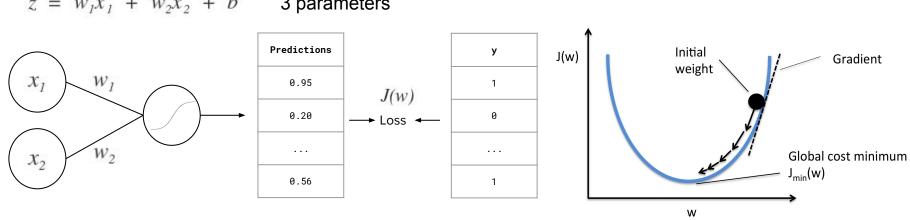




Better model

Linear model

$$z = w_1 x_1 + w_2 x_2 + b$$
 3 parameters

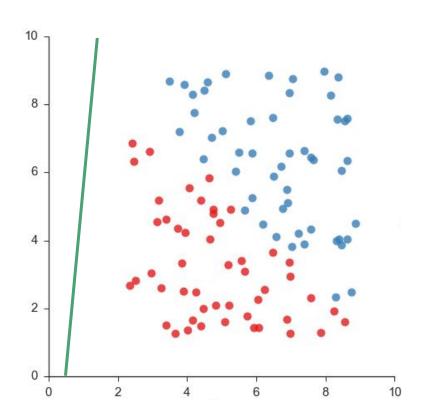


Optimisation

Iteration: 1

Model: $0.09x_1 - 0.01x_2 - 0.024$

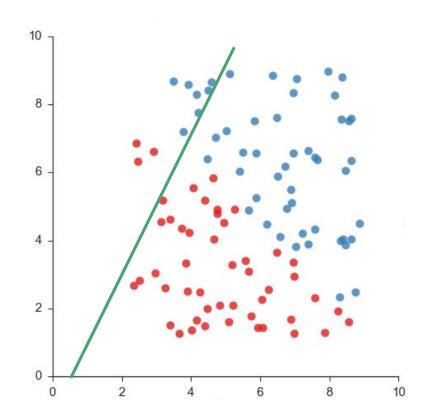
Loss: 0.641



Iteration: 2

Model: $0.17x_1 - 0.08x_2 - 0.07$

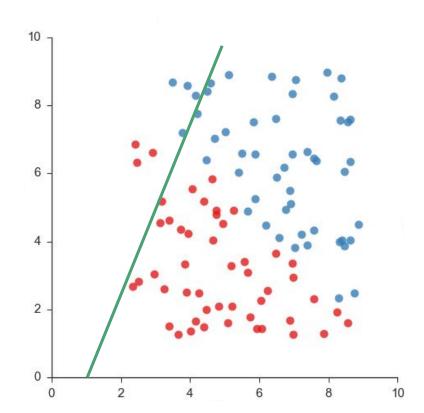
Loss: 0.623



Iteration: 5

Model: $0.42x_1 - 0.16x_2 - 0.46$

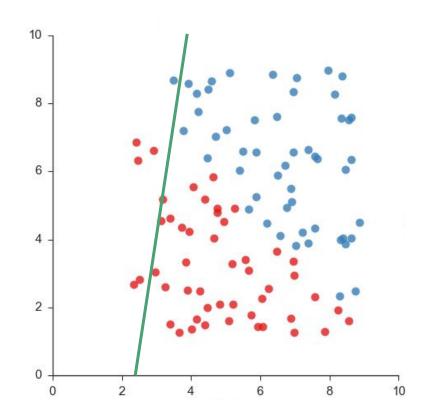
Loss: 0.583



Iteration: 7

Model: $0.64x_1 - 0.08x_2 - 1.65$

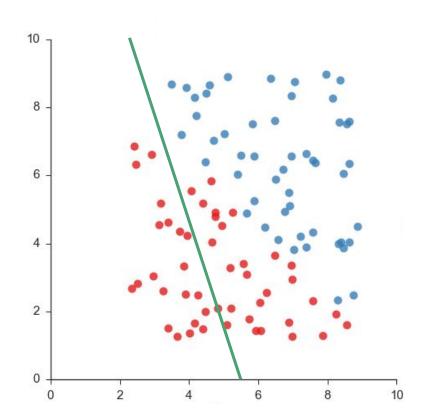
Loss: 0.527



Iteration: 9

Model: $1.15x_1 - 0.37x_2 - 6.39$

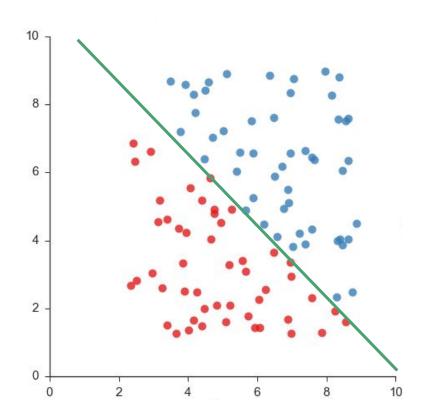
Loss: 0.351



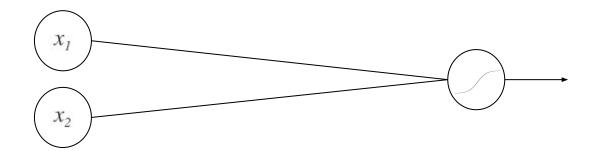
Iteration:18

Model: $2.13x_1 - 2.05x_2 - 21.88$

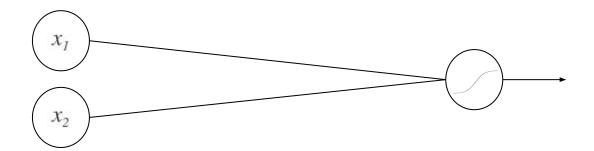
Loss: 0.091

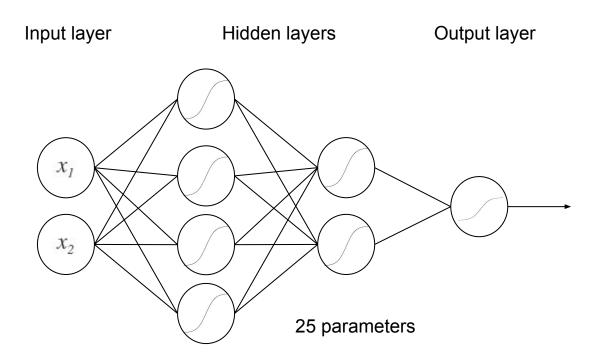


Logistic regression



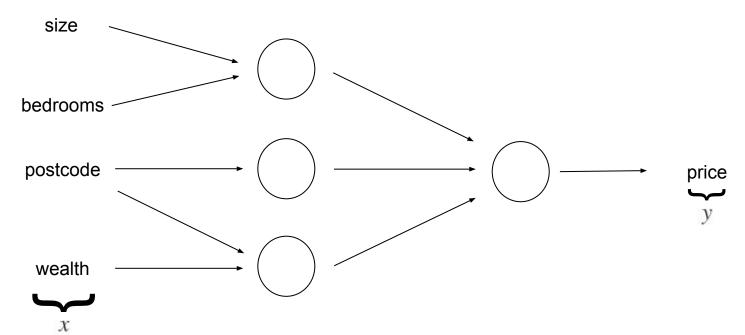
Input layer Output layer





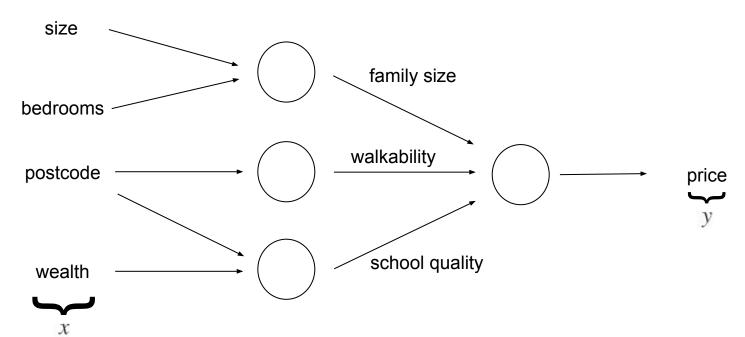
Demo: http://playground.tensorflow.org/

Housing price prediction



Source: Andrew Ng, Coursera - Neural Networks and Deep Learning. 2017

Housing price prediction



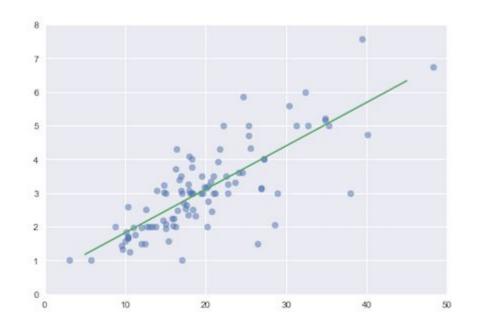
Source: Andrew Ng, Coursera - Neural Networks and Deep Learning. 2017

Linear regression

- Predict a continuous value
- Find the line of best fit
- Training process similar to logistic regression
 - different loss function
 - mean squared error

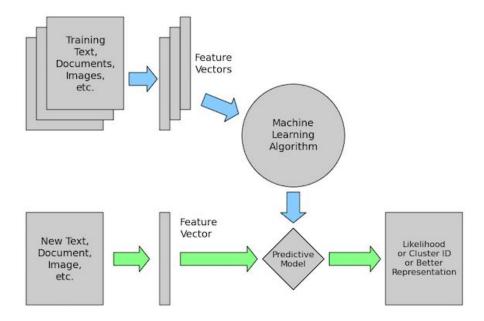
Linear model

$$y = 0.12x + 0.53$$



Unsupervised learning

- Learn underlying structure of the data
- Dimensionality reduction





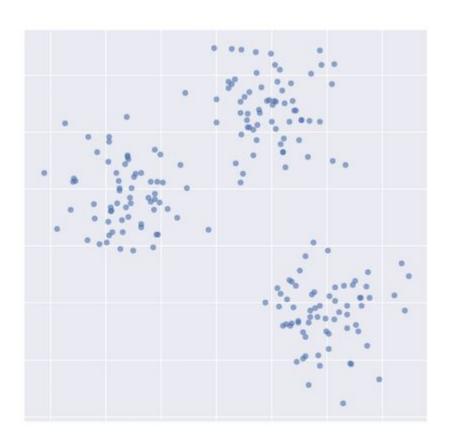
- Clustering algorithm
 - K = # of clusters

Training

- Randomly place K initial centroids
- Assign each point to closest centroid
- Update centroid position based on mean of points in assigned cluster
- Stop when centroids don't change

New examples

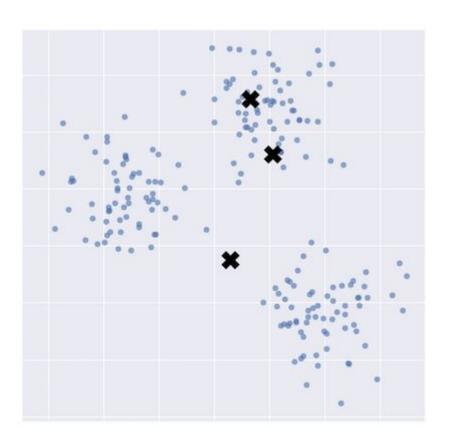
Assign to cluster with closest centroid



- Clustering algorithm
 - K = # of clusters

- Training
 - Randomly place K initial centroids
 - Assign each point to closest centroid
 - Update centroid position based on mean of points in assigned cluster
 - Stop when centroids don't change

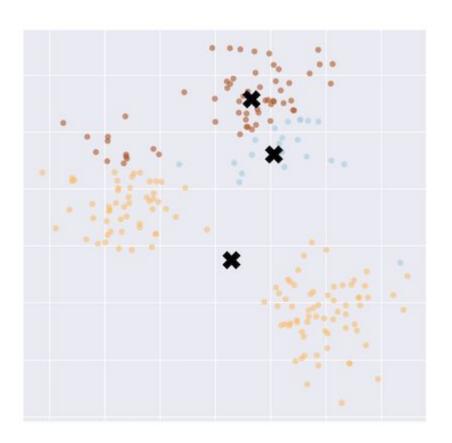
- New examples
 - Assign to cluster with closest centroid



- Clustering algorithm
 - K = # of clusters

- Training
 - Randomly place K initial centroids
 - Assign each point to closest centroid
 - Update centroid position based on mean of points in assigned cluster
 - Stop when centroids don't change

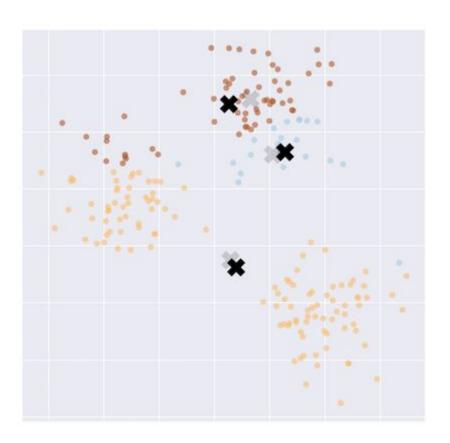
- New examples
 - Assign to cluster with closest centroid



- Clustering algorithm
 - K = # of clusters

- Training
 - Randomly place K initial centroids
 - Assign each point to closest centroid
 - Update centroid position based on mean of points in assigned cluster
 - Stop when centroids don't change

- New examples
 - Assign to cluster with closest centroid



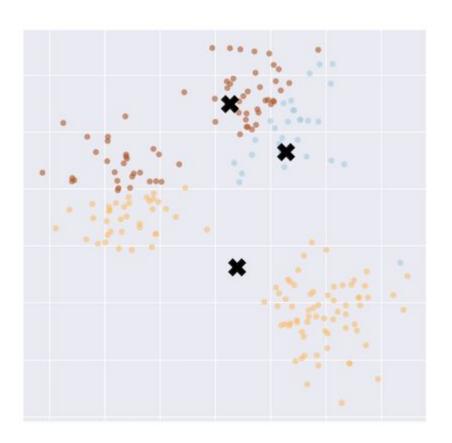
- Clustering algorithm
 - O K = # of clusters

Training

- Randomly place K initial centroids
- Assign each point to closest centroid
- Update centroid position based on mean of points in assigned cluster
- Stop when centroids don't change

New examples

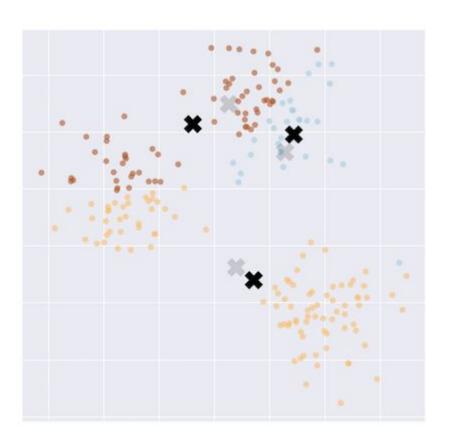
Assign to cluster with closest centroid



- Clustering algorithm
 - K = # of clusters

- Training
 - Randomly place K initial centroids
 - Assign each point to closest centroid
 - Update centroid position based on mean of points in assigned cluster
 - Stop when centroids don't change

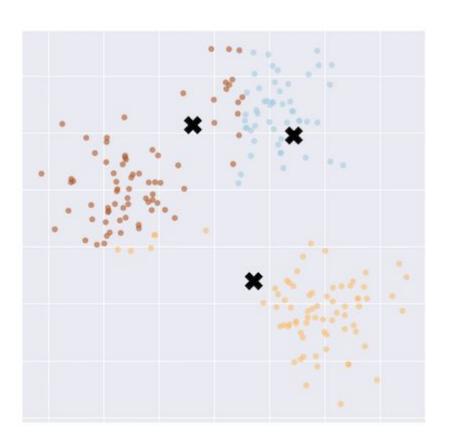
- New examples
 - Assign to cluster with closest centroid



- Clustering algorithm
 - K = # of clusters

- Training
 - Randomly place K initial centroids
 - Assign each point to closest centroid
 - Update centroid position based on mean of points in assigned cluster
 - Stop when centroids don't change

- New examples
 - Assign to cluster with closest centroid



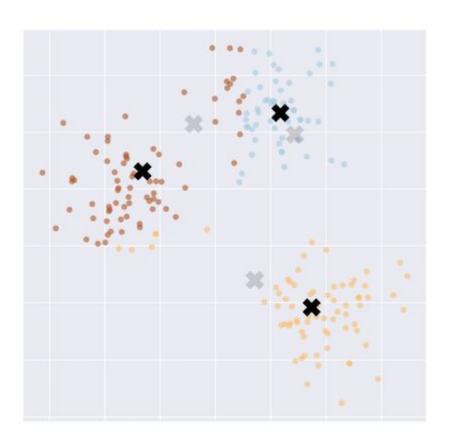
- Clustering algorithm
 - K = # of clusters

Training

- Randomly place K initial centroids
- Assign each point to closest centroid
- Update centroid position based on mean of points in assigned cluster
- Stop when centroids don't change

New examples

Assign to cluster with closest centroid



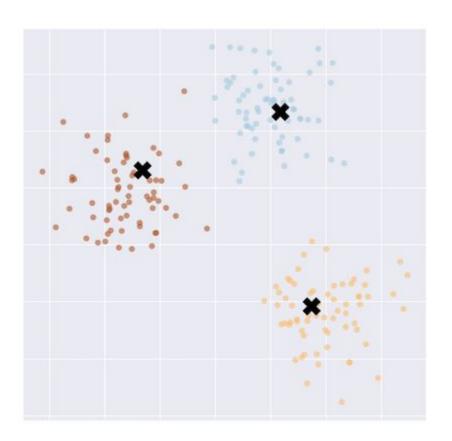
- Clustering algorithm
 - K = # of clusters

Training

- Randomly place K initial centroids
- Assign each point to closest centroid
- Update centroid position based on mean of points in assigned cluster
- Stop when centroids don't change

New examples

Assign to cluster with closest centroid

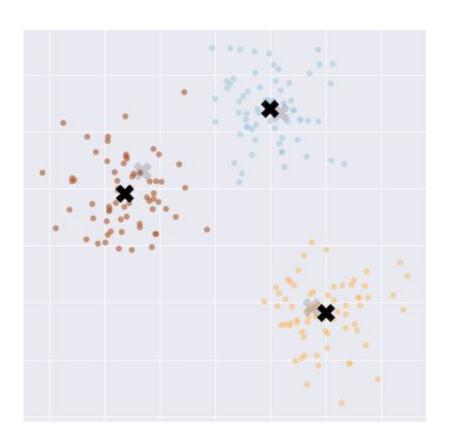


- Clustering algorithm
 - K = # of clusters

Training

- Randomly place K initial centroids
- Assign each point to closest centroid
- Update centroid position based on mean of points in assigned cluster
- Stop when centroids don't change

- New examples
 - Assign to cluster with closest centroid



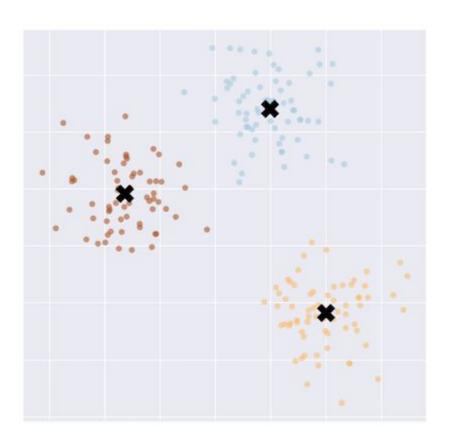
- Clustering algorithm
 - K = # of clusters

Training

- Randomly place K initial centroids
- Assign each point to closest centroid
- Update centroid position based on mean of points in assigned cluster
- Stop when centroids don't change

New examples

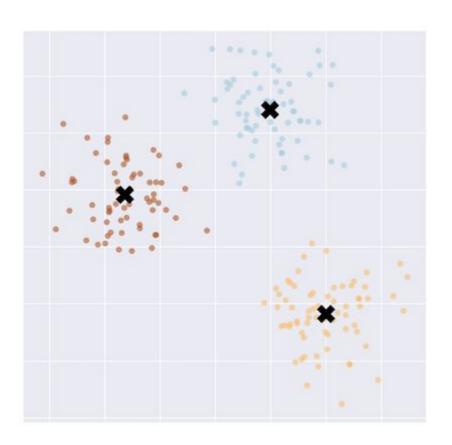
Assign to cluster with closest centroid



- Clustering algorithm
 - K = # of clusters

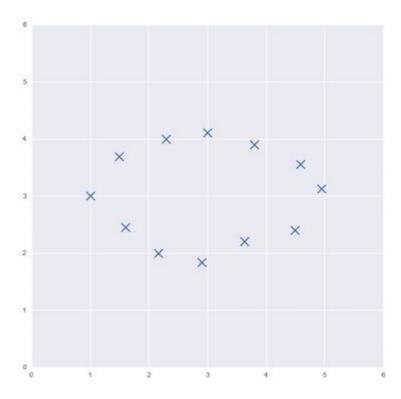
- Training
 - Randomly place K initial centroids
 - Assign each point to closest centroid
 - Update centroid position based on mean of points in assigned cluster
 - Stop when centroids don't change

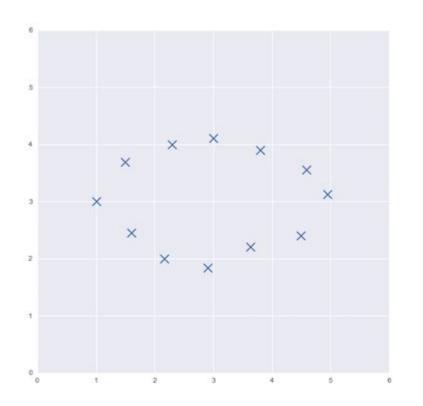
- New examples
 - Assign to cluster with closest centroid



Principal Components Analysis (PCA)

- Dimensionality reduction method
- Represent data in fewer dimensions
 - Preserve as much structure as possible
- New dimensions = principal components
 - Directions where data is most spread (variance)
 - Combination of original features
- Example
 - Reduce a dataset from 2D to 1D
 - Manually try two components

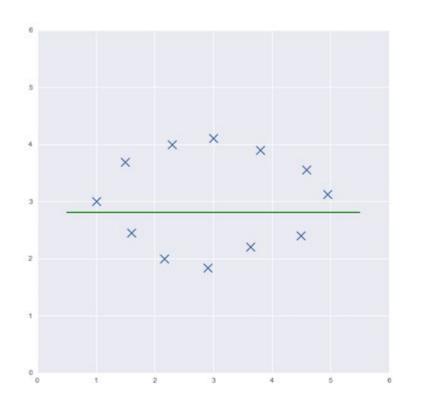




× × X 3 × × × 0 3

Component A

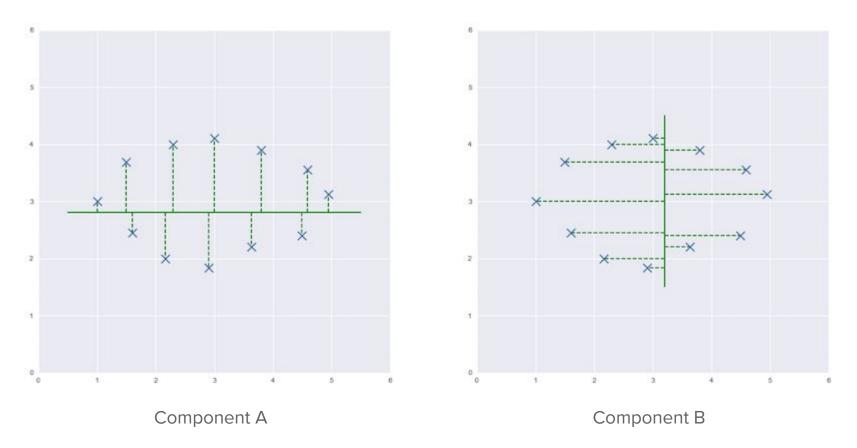
Component B



X × × X 3 × × 0 2 3

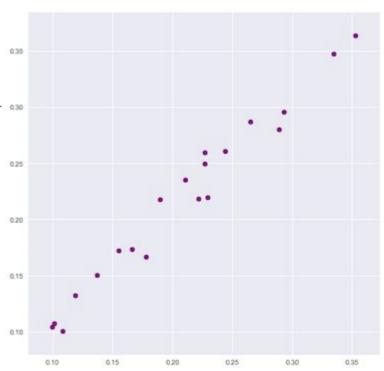
Component A

Component B

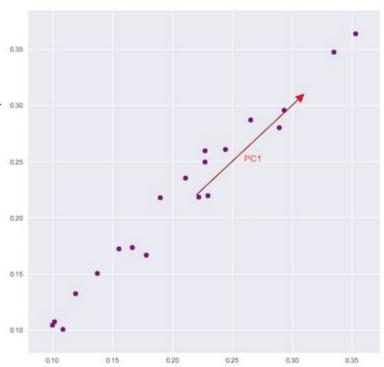


Which component represents a larger spread of the data (variance)? Component A

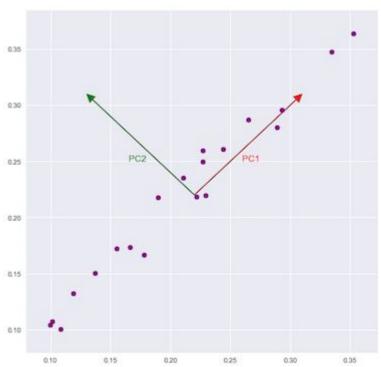
- Find a set of principal components
 - 1st: represents most variance
 - o 2nd: perpendicular to 1st, most variance of remainder
 - ... until last dimension of dataset



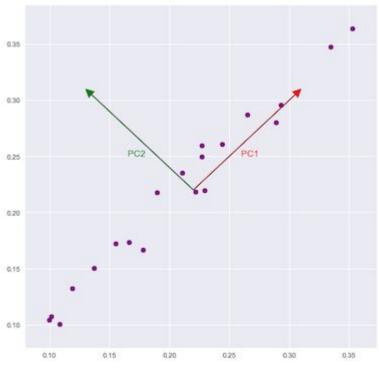
- Find a set of principal components
 - 1st: represents most variance
 - o 2nd: perpendicular to 1st, most variance of remainder
 - ... until last dimension of dataset



- Find a set of principal components
 - 1st: represents most variance
 - o 2nd: perpendicular to 1st, most variance of remainder
 - ... until last dimension of dataset

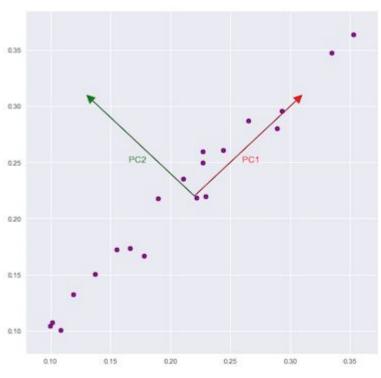


- Find a set of principal components
 - 1st: represents most variance
 - 2nd: perpendicular to 1st, most variance of remainder
 - ... until last dimension of dataset
- Reduce dataset to *k* components
 - E.g. keep top k components that represent at least
 90% of variance in the dataset
 - Transform original data to the new dimensions

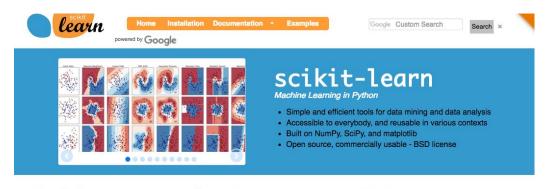


PC1 represents 99.6% of variance

- Find a set of principal components
 - 1st: represents most variance
 - 2nd: perpendicular to 1st, most variance of remainder
 - ... until last dimension of dataset
- Reduce dataset to k components
 - E.g. keep top k components that represent at least
 90% of variance in the dataset
 - Transform original data to the new dimensions
- Algorithm
 - Normalise data with zero mean
 - Calculate covariance matrix
 - Calculate:
 - Eigenvectors = principal component
 - Eigenvalues = amount of variance represented



PC1 represents 99.6% of variance



Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.

Algorithms: SVM, nearest neighbors, random forest. ... — Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.

Algorithms: SVR, ridge regression, Lasso, ...

- Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes

Algorithms: k-Means, spectral clustering, mean-shift. ... — Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency

Algorithms: PCA, feature selection, nonnegative matrix factorization. — Examples

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning

Modules: grid search, cross validation, metrics.

— Examples

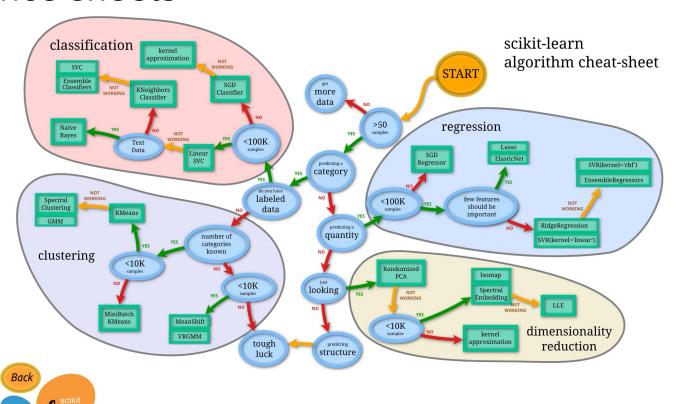
Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.

Modules: preprocessing, feature extraction.

- Examples



Comparison of supervised machine learning techniques

Factors	Decision Trees	Neural Networks	Naïve Bayes	kNN	SVM	Rule Learners
General accuracy	**	***	*	**	****	**
Learning speed	***	*	****	****	*	**
Classification speed	****	****	****	*	****	****
Tolerance to missing values	***	*	****	*	**	**
Tolerance to irrelevant features	***	*	**	**	****	**
Tolerance to redundant features	**	**	*	**	***	**
Tolerance to highly related features	**	***	*	*	***	**
Dealing with discrete, binary and continuous features	****	***	***	***	**	***
Tolerance to noise	**	**	***	*	**	*
Dealing with model overfitting	**	*	***	***	**	**
Attempts for incremental learning	**	***	***	****	**	*
Explanation of classification	****	*	****	**	*	****
Model parameter handling	***	*	***	***	*	***

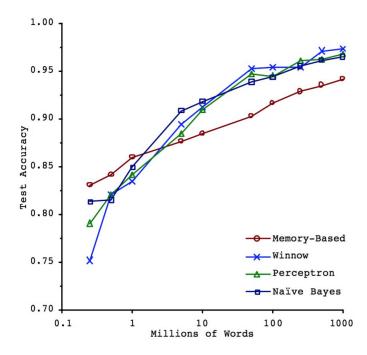
^{*} kNN = k-Nearest Neighbours

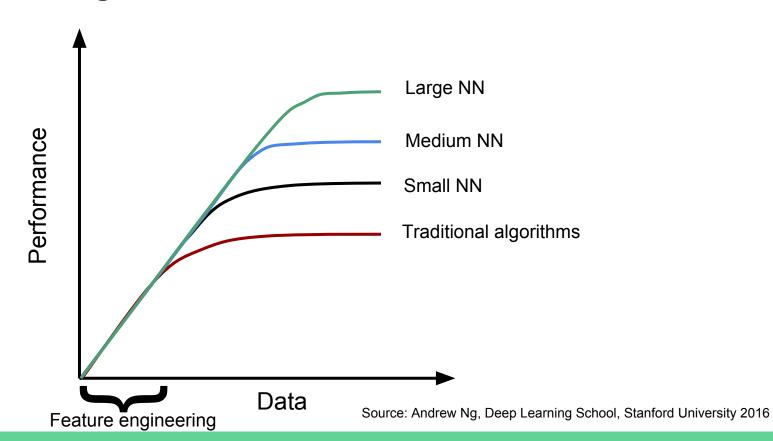
^{*} SVM = Support Vector Machines

- Supervised learning
 - http://scikit-learn.org/stable/supervised_learning.html#supervised-learning
- Performance metrics
 - http://scikit-learn.org/stable/modules/classes.html/module-sklearn.metrics
- Feature selection
 - http://scikit-learn.org/stable/modules/feature_selection.html
- Model selection (cross validation, learning curves, hyperparameter tuning)
 - http://scikit-learn.org/stable/modules/classes.html#module-sklearn.model_selection
- Clustering
 - http://scikit-learn.org/stable/modules/clustering.html#clustering
- Dimensionality reduction
 - http://scikit-learn.org/stable/modules/classes.html#module-sklearn.decomposition

- Anomaly detection
 - http://scikit-learn.org/stable/modules/outlier_detection.html
- Time series
 - http://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/
 - https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-pyth on-keras/
 - http://www.statsmodels.org/dev/generated/statsmodels.tsa.arima_model.ARIMA.html
- ML blogs:
 - https://www.analyticsvidhya.com/blog/2015/08/common-machine-learning-algorithms/
 - http://machinelearningmastery.com/a-tour-of-machine-learning-algorithms/

- Online courses
 - Machine learning
 - https://www.coursera.org/learn/machine-learning
 - https://www.udacity.com/course/intro-to-machine-learning--ud120
 - Deep learning
 - https://www.coursera.org/specializations/deep-learning
 - https://www.udacity.com/course/deep-learning--ud730
 - http://course.fast.ai/







Completed • \$16,000 • 326 teams

Galaxy Zoo - The Galaxy Challenge

Fri 20 Dec 2013 - Fri 4 Apr 2014 (2 years ago)

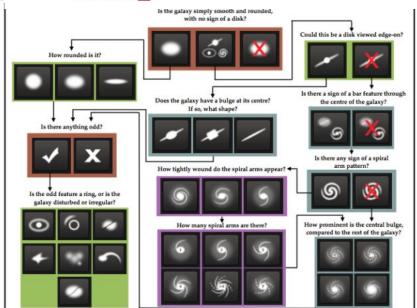
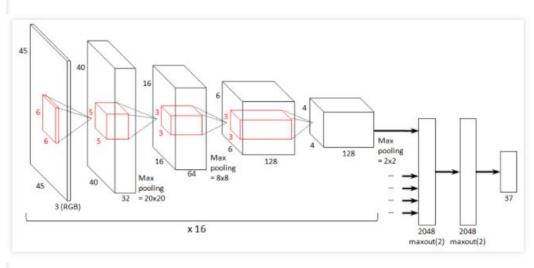


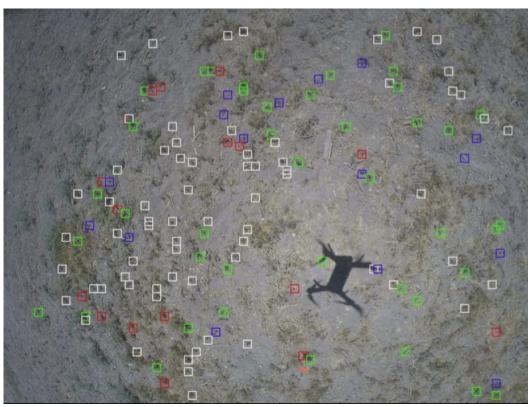
Figure 1. Flowchart of the classification tasks for GZ2, beginning at the top centre. Tasks are colour-coded by their relative depths in the decision tree. Tasks outlined in brown are asked of every galaxy. Tasks outlined in green, blue, and purple are (respectively) one, two or three steps below branching points in the decision tree. Table 2d describes the responses that correspond to the icons in this diagram.



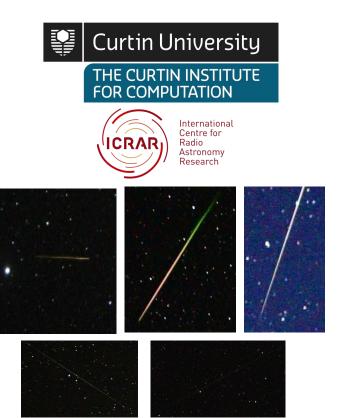
Source: Sander Dieleman 2014 - http://benanne.github.io/2014/04/05/galaxy-zoo.html

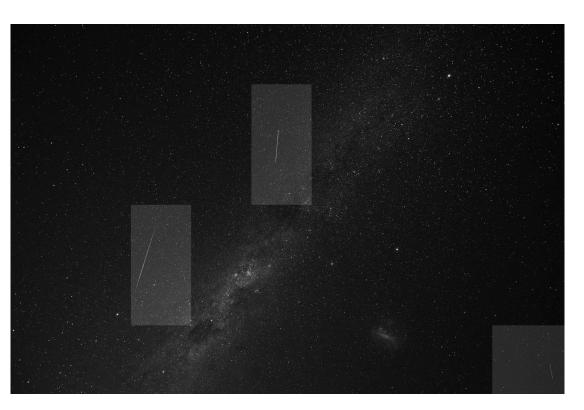






Source: Sravanthi Sinha, Deep Learning Summit, Singapore 2016

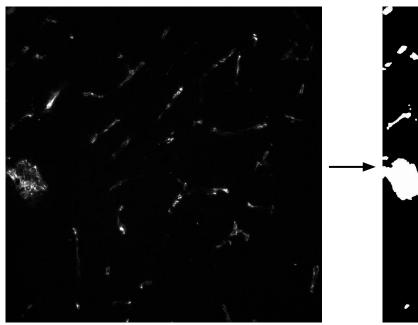




Blood vessel leakage measurement

School of Public Health, Curtin Health Innovation Research Institute (CHIRI)





Microscope images of blood vessels



Generate masks to measure leakage

Expert researcher

1 month to measure 1,000 images manually

Trained model

1 ½ minutes to measure 1,000 images automatically

