Machine Learning

An overview of supervised methods

William La Cava
Postdoctoral Researcher
Computational Genetics Laboratory
lacava@upenn.edu

October 9, 2018

Outline

Outline

Outline

Outline

Outline

Outline

Statistics vs. Machine Learning

Statistics

- Interpretable models
- Precision and Uncertainty
- Exposure vs. adjustments
- Distributions

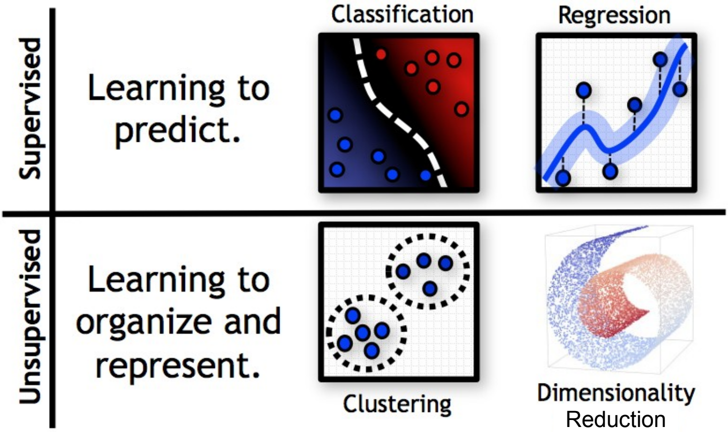
Machine Learning

- Larger, often black-box models
- Prediction
- Features
- Optimization, Algorithms

Fields converge, especially as datasets grow

Machine Learning

Tasks



Supervised Learning

- Training data: $\mathcal{T} = \{(\mathbf{x_i}, y_i), i = 1 \ \ldots \ N\}$ with d attributes $\mathbf{x}$
- Classification: $y \in \mathcal{Y} = \{1 \ \ldots \ K\}$
- Regression: $y \in \mathbb{R}$ (real values)

Supervised Learning

- Training data: $\mathcal{T} = \{(\mathbf{x_i}, y_i), i = 1 \ \ldots \ N\}$ with d attributes $\mathbf{x}$
- Classification: $y \in \mathcal{Y} = \{1 \ \ldots \ K\}$
- Regression: $y \in \mathbb{R}$ (real values)

Definition (Classification)

Given a set of examples $\mathcal{T}$, learn a function $f(\mathbf{x}) \rightarrow \mathcal{Y}$ that accurately predicts the class label y, for any feature vector $\mathbf{x}$.

Supervised Learning

- Training data: $\mathcal{T} = \{(\mathbf{x_i}, y_i), i = 1 \ \ldots \ N\}$ with d attributes $\mathbf{x}$
- Classification: $y \in \mathcal{Y} = \{1 \ \ldots \ K\}$
- Regression: $y \in \mathbb{R}$ (real values)

Definition (Classification)

Given a set of examples $\mathcal{T}$, learn a function $f(\mathbf{x}) \rightarrow \mathcal{Y}$ that accurately predicts the class label y, for any feature vector $\mathbf{x}$.

Definition (Regression)

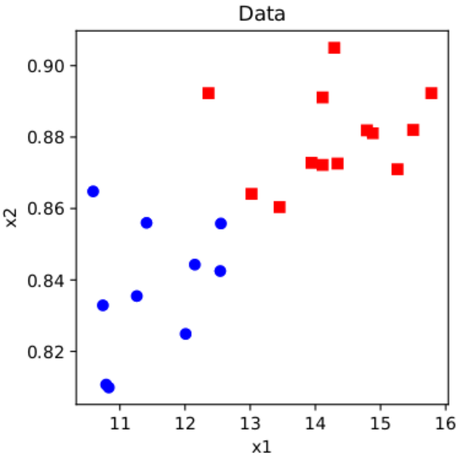
Given a set of examples $\mathcal{T}$, learn a function $f(\mathbf{x}) \rightarrow \mathbb{R}$ that accurately predicts the value of y, for any feature vector $\mathbf{x}$.

Support Vector Machines

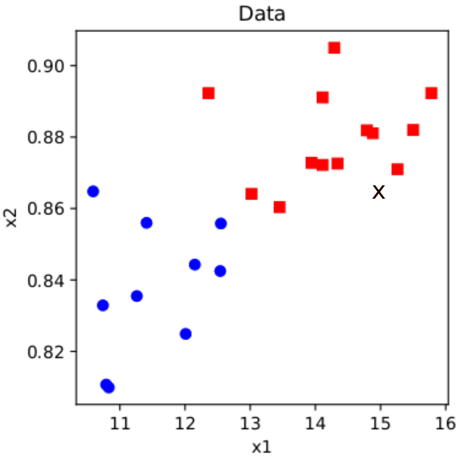
- A binary support vector machine is a linear classifier that takes labels in the set {-1,1}.
- decision function:

$$f_{SVM}(\mathbf{x}) = \text{sign}(\mathbf{w}^T\mathbf{x} + b)$$

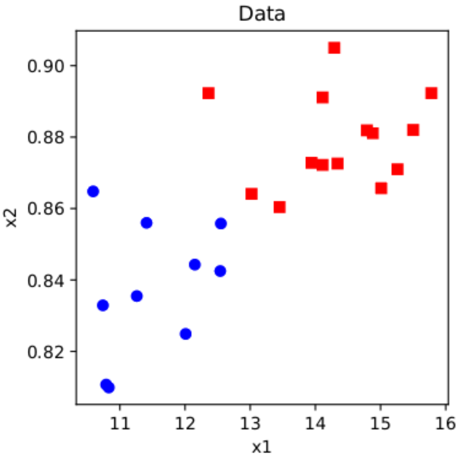
Support Vector Machines



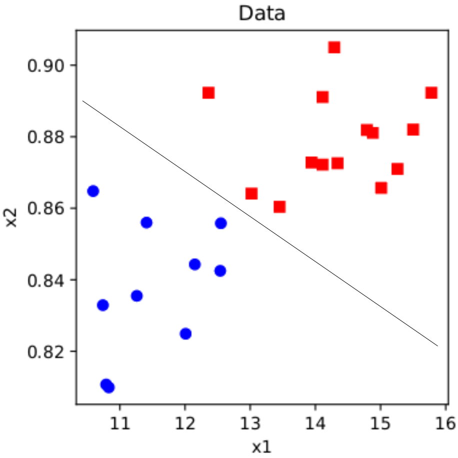
Support Vector Machines



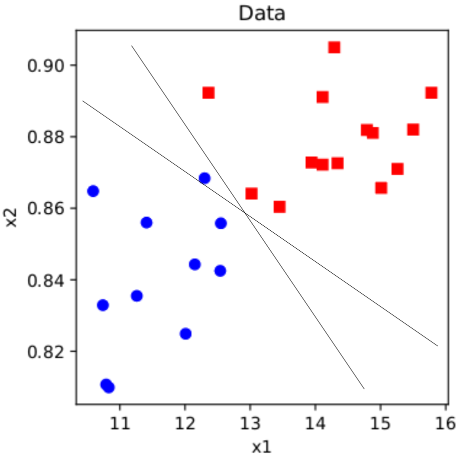
Support Vector Machines



Support Vector Machines

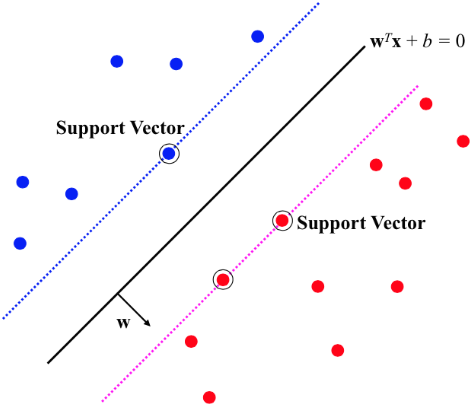


Support Vector Machines



Support Vector Machines

SVMs choose parameters that give the "maximum margin" property.



Support Vector Machines

- A binary support vector machine is a linear classifier that takes labels in the set {-1,1}.
- decision function:

$$f_{SVM}(\mathbf{x}) = \mathrm{sign}(\mathbf{w}^T\mathbf{x} + b)$$

Support Vector Machines

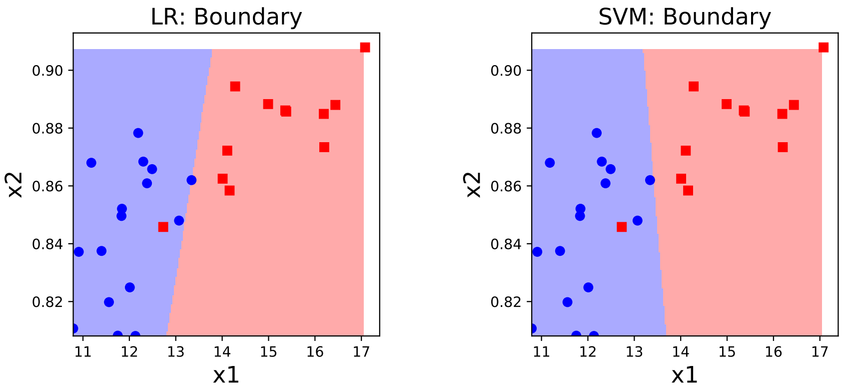
- A binary support vector machine is a linear classifier that takes labels in the set {-1,1}.
- decision function:

$$f_{SVM}(\mathbf{x}) = \text{sign}(\mathbf{w}^T\mathbf{x} + b)$$

- the decision boundary for logistic regression can be written the same way

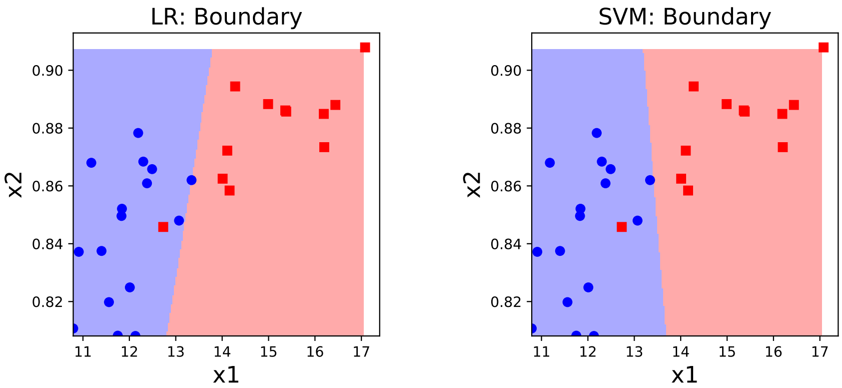
Support Vector Machines

and Logistic Regression



Support Vector Machines

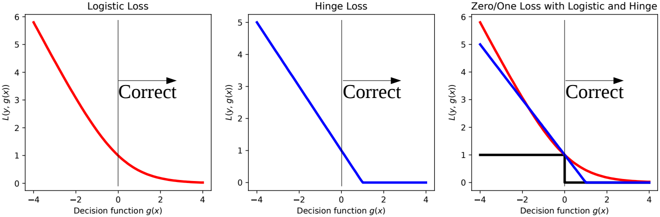
and Logistic Regression



So how do SVM and LR differ?

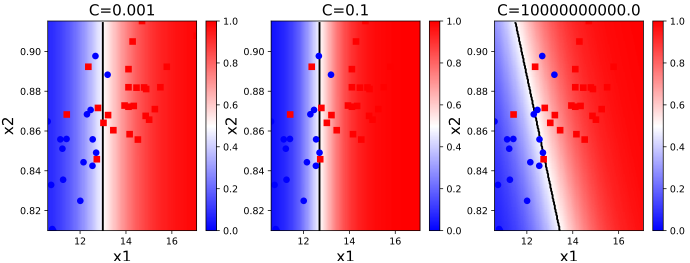
Support Vector Machines

They optimize different loss functions.



Support Vector Machines

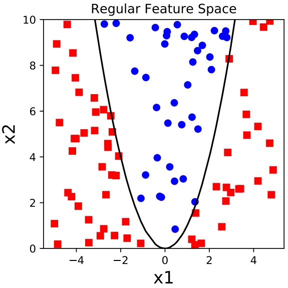
C: relaxation parameter allows some points to fall in the margin.

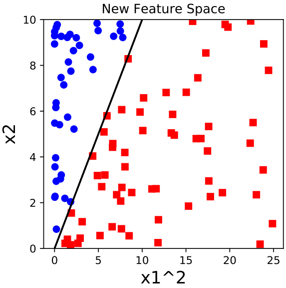


C	0.001	0.1	1e10
Train Accuracy	0.595	0.929	0.888
Test Accuracy	0.459	**0.905**	0.867

Support Vector Machines

- What if the decision boundary isn't linear?
- We can non-linear transform **x**, then train in the (hopefully linear) transformed space
- SVMs use Kernels to do this.





Support Vector Machines

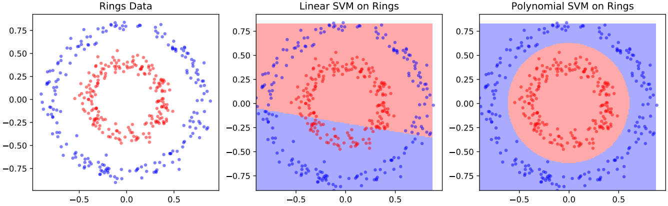
The SVM function can be written as an inner product:

$$\mathbf{w}^T\mathbf{x}_i + b = \sum_j^N \alpha_j \mathbf{x}_j^T \mathbf{x}_i + b$$

$$K(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\mathbf{x}'$$

- B-order Polynomial Kernel: $K_p(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^T\mathbf{x}' + 1)^B$
- Gaussian/RBF Kernel: $K_G(\mathbf{x}, \mathbf{x}') = exp(-\gamma||\mathbf{x} - \mathbf{x}'||_2^2)$
- "Kernel trick" makes this quick

Support Vector Machines



Support Vector Machines

Conclusions

- SVMs use of the hinge loss function can make them generalize better than Logistic Regression.
- SVMs use Kernel functions to make non-linear boundaries.
- The C parameter must be tuned to generate reliable predictions.

Decision Trees

- Classifies data cases using a conjunction of rules organized into a binary tree structure.

Decision Trees

- Classifies data cases using a conjunction of rules organized into a binary tree structure.
- each node contains a rule: $(x_d < t)$ or $(x_d == t)$

Decision Trees

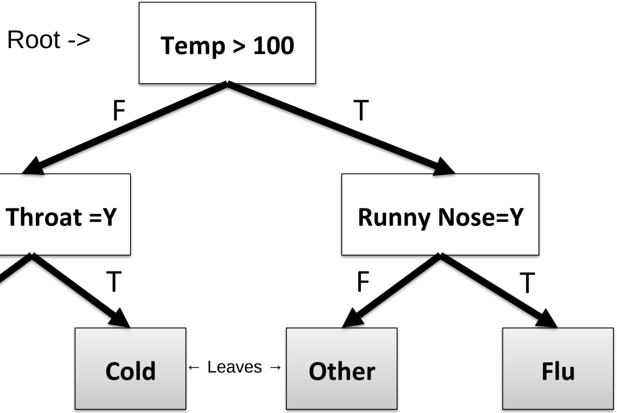
- Classifies data cases using a conjunction of rules organized into a binary tree structure.
- each node contains a rule: $(x_d < t)$ or $(x_d == t)$
- each training example is routed left or right down the tree based on the rule.

Decision Trees

- Classifies data cases using a conjunction of rules organized into a binary tree structure.
- each node contains a rule: $(x_d < t)$ or $(x_d == t)$
- each training example is routed left or right down the tree based on the rule.
- leaf nodes label examples with a class or average value.

Decision Trees

Example



Decision Trees

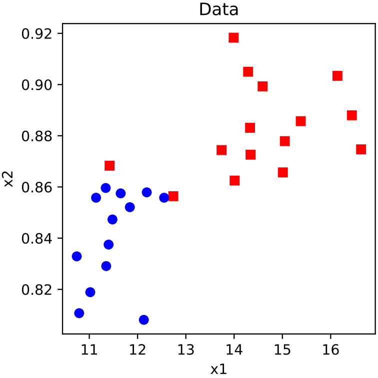
Learning

- Recursively finds a variable that best divides data into two outcomes
- "best variable" is determined heuristically
  - Gini impurity (CART)
  - Information Gain (ID3, C4.5)
- Heuristics: produce splits as homogenous as possible in terms of labels
- stop criterion: max depth, purity of labels in each leaf

Decision Trees

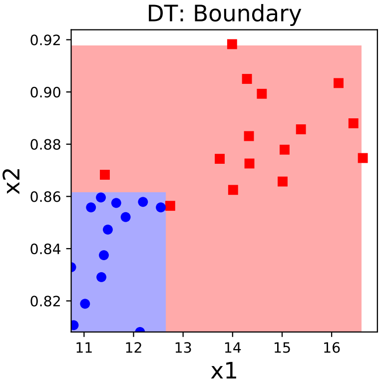
Boundary

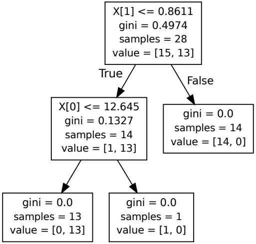
What does a decision tree boundary look like?



Decision Trees
Boundary

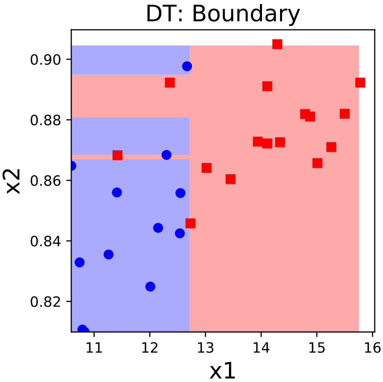
What does a decision tree boundary look like?

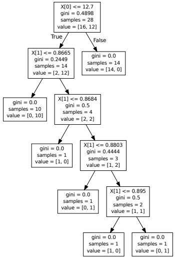




Decision Trees
Boundary

What does a decision tree boundary look like?





Decision Trees

Other Properties

- Splitting on singe variables can require very large trees to accurately model decision boundaries
- Greedy (optimal trees are hard to find)
- Given sufficient depth, a decision tree can approximate any classification function to arbitrary accuracy

Ensembles

- An *ensemble* is simply a collection of models trained on the same task.

Ensembles

- An *ensemble* is simply a collection of models trained on the same task.
- Many versions of same model, different types of models

Ensembles

- An *ensemble* is simply a collection of models trained on the same task.
- Many versions of same model, different types of models
- Final output: weighted average or vote

Ensembles

- An ensemble of different models that all achieve similar generalization performance often outperforms any of the individual models.

Ensembles

- An ensemble of different models that all achieve similar generalization performance often outperforms any of the individual models.
- how?

Ensembles

- Suppose we have a set of binary classifiers

Ensembles

- Suppose we have a set of binary classifiers
- Each classifier has the same average error rate that is better than randomly guessing

Ensembles

- Suppose we have a set of binary classifiers
- Each classifier has the same average error rate that is better than randomly guessing
- Assume the errors they make are *independent*

Ensembles

- Suppose we have a set of binary classifiers
- Each classifier has the same average error rate that is better than randomly guessing
- Assume the errors they make are *independent*
- Intuition: the majority of the classifiers will be correct on many examples where any individual classifier makes an error.

Ensembles

- Suppose we have a set of binary classifiers
- Each classifier has the same average error rate that is better than randomly guessing
- Assume the errors they make are *independent*
- Intuition: the majority of the classifiers will be correct on many examples where any individual classifier makes an error.
- A simple majority vote can improve classification performance by *decreasing variance* in this setting.

Ensembles

- Suppose we have a set of binary classifiers
- Each classifier has the same average error rate that is better than randomly guessing
- Assume the errors they make are *independent*
- Intuition: the majority of the classifiers will be correct on many examples where any individual classifier makes an error.
- A simple majority vote can improve classification performance by *decreasing variance* in this setting.
- How do we train such an ensemble?

Bagging

- Bootstrap aggregation
- Attempts to train independent classifiers by sampling the training set.
- Sample $\mathcal{T}$ k times with replacement
- Train k classifiers $f_1(\mathbf{x})$ ... $f_k(\mathbf{x})$ on subsets
- Useful for high-variance, high-capacity models (i.e. decision trees)
- Internal error estimate: use the portion of data that wasnt built to create model as a test set (out of bag data)

Random Forests

- Random forests are a successful extension of bagged trees

Random Forests

- Random forests are a successful extension of bagged trees
- Extension: only consider random sub-set of features when deciding which variables to split on

Random Forests

- Random forests are a successful extension of bagged trees
- Extension: only consider random sub-set of features when deciding which variables to split on
- When given new data, pass it to all trees in forest, and estimate class based on most popular outcome

Random Forests

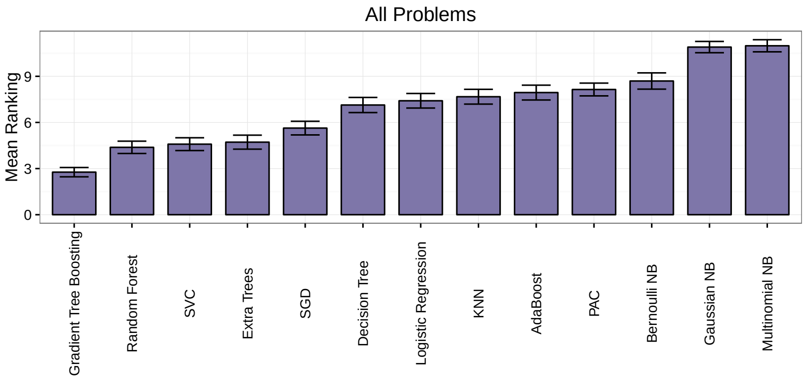
- Good predictive performance and avoids overfitting

Random Forests

- Good predictive performance and avoids overfitting
- Importance: estimate of single variable contribution to classification

Ensemble Examples

Analysis of 14 methods on 165 open-source classification datasets (PSB 2018)



Conclusions

- We covered SVM, Decision Trees, Random Forests
- Other supervised learning algorithms
  - K-Nearest Neighbors
  - Neural Networks / Deep Learning
  - Naïve Bayes
- Thursday: Unsupervised Learning
  - K-Means
  - Heirarchical Clustering
  - PCA