

# Introduction to Machine Learning

John Maxwell - NLTT

# Topics of This Talk

- Support Vector Machines
- Log-linear models
- Decision trees

# Basic Goal of Machine Learning

- Predict unseen data from seen data

Temperature	Humidity	Rain
55	10%	no
40	40%	yes
95	20%	no
75	45%	yes

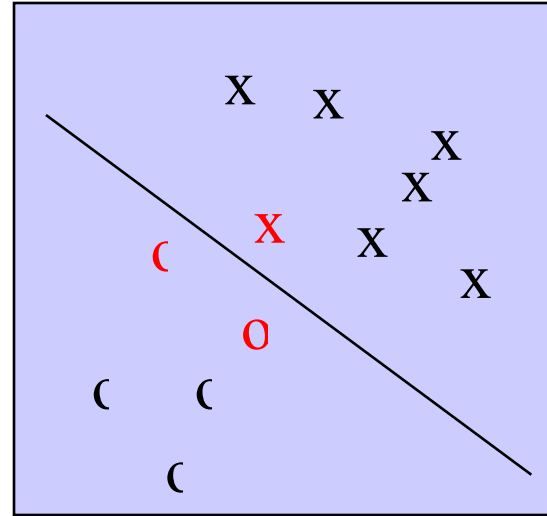
- Will it rain at 70 degrees and 30% humidity?

# Basic Goal of Machine Learning (2)

- Seen data is *labeled*
- Unseen data does not match seen data
- Goal is categorization

# Basic Strategy of Machine Learning

- Draw a line:



- *Large margin* = line with most separation
- *support vectors* = nearest points to line
- For more dimensions, choose hyperplane

# Support Vector Machines (SVMs)

- Explicitly maximize the margin
- Minimize the number of support vectors
- Represent line using support vectors
- Work in data space rather than feature space
- Data space is dual of feature space

# The Curse of Dimensionality

- Many dimensions => many degrees of freedom
- Many dimensions => easy to *overtrain*
- Overtrain = good results on training data, bad results on test data

# The Curse of Dimensionality (2)

Many dimensions => sparse data problem

Data points	Dimensions (10 units per dim.)	Density
1000	1	100/unit
1000	3	1/unit
1000	9	1/million units

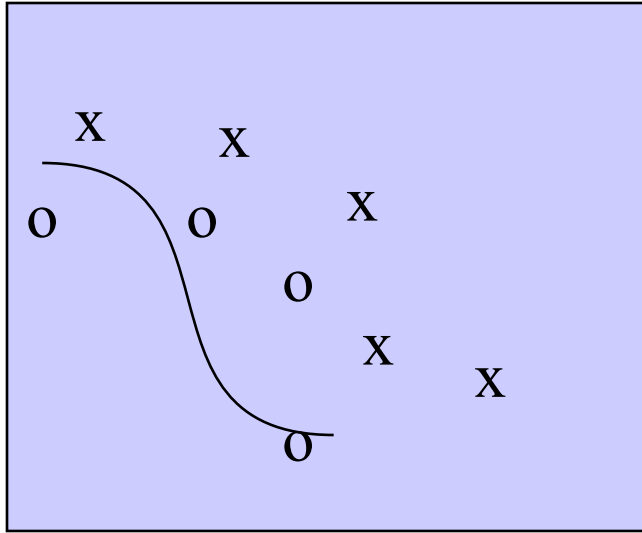
# SVMs and Dimensionality

- Degrees of freedom  $\leq$  number support vectors
- Number support vectors  $\leq$  #dimensions
- Number support vectors  $\leq$  #data points
- $\Rightarrow$  SVMs tend not to overtrain

# What If You Can't Draw a Line?

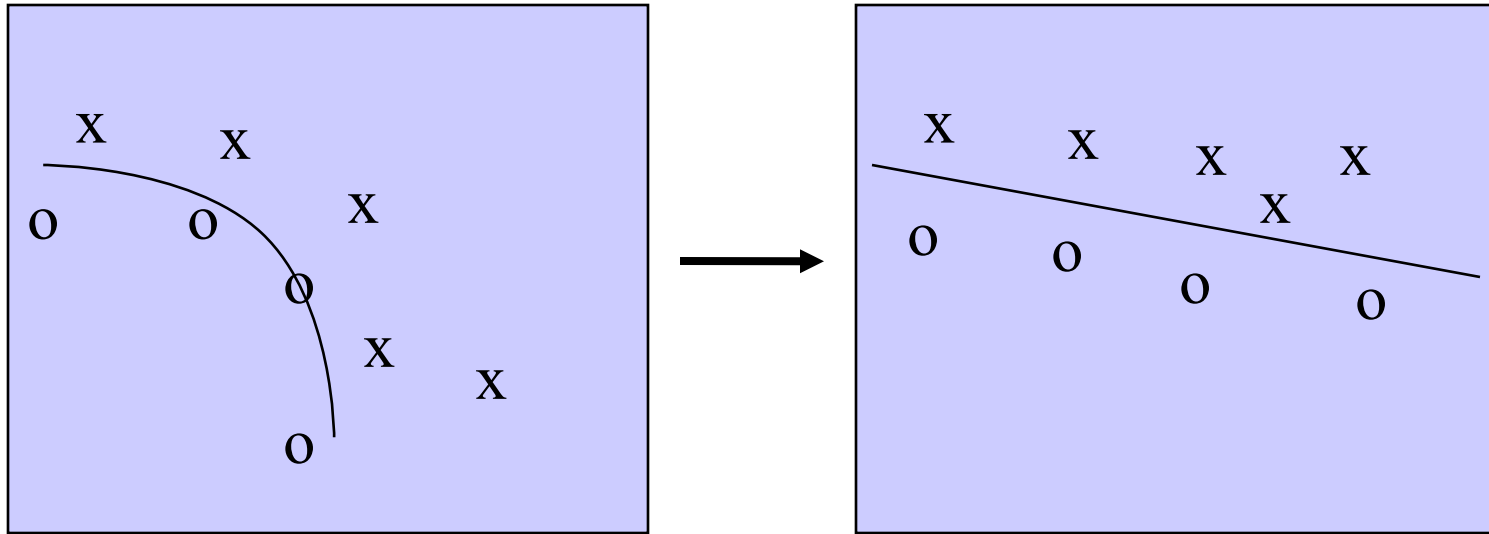
- Use a higher-order function
- Map to a space where you can draw a line
- Ignore some of the data points
- Add new features (dimensions) and draw a hyperplane

# Using a Higher-order Function



- Adds degrees of freedom
- Same as mapping to higher-order space
- Same as adding higher-order features

# Mapping to another space

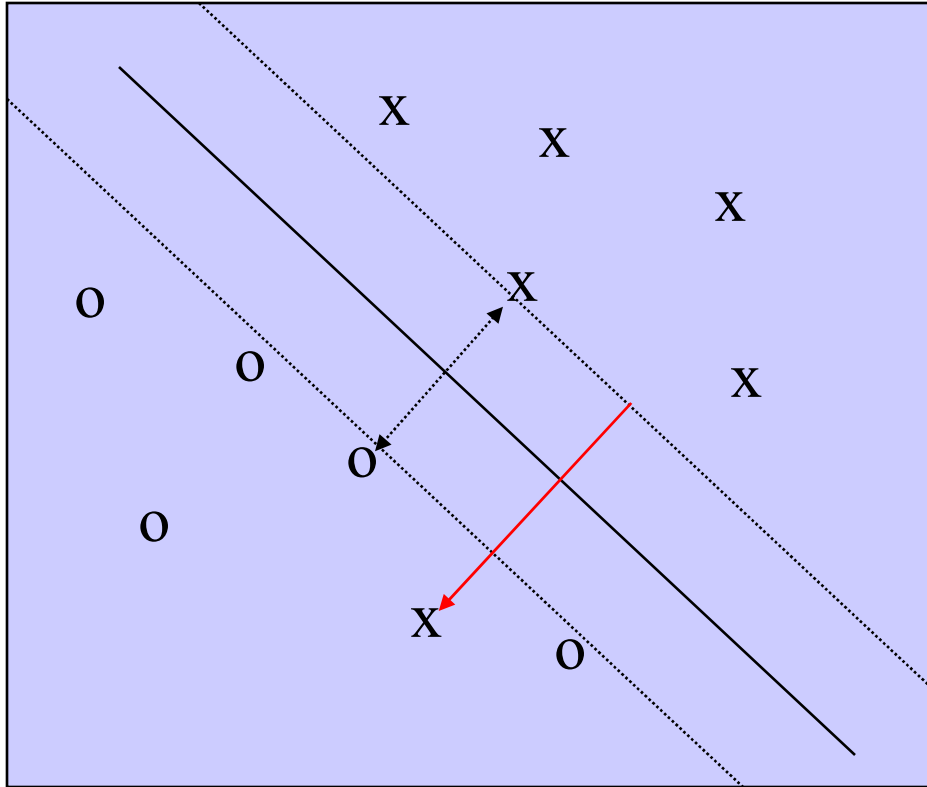


- May add degrees of freedom
- You must choose mapping in advance

# Kernel Functions

- Used by SVMs
- Work in data space, not feature space
- Implicit mapping of a large number of features
- Sometimes infinite number of features
- Don't have to compute the features

# Ignoring Some Data Points



- SVMs ignore points using **slack** variables

# Using Slack Variables

- Maximize margin with a penalty for slack
- Useful when data is noisy
- Use with separable data to get a larger margin
- The penalty weight is a *hyperparameter*
- More weight => less slack
- The best weight can be chosen by *cross-validation*
- Produces a *soft-margin* classifier

# Adding Features

- Features can be new properties of data
- Features can be combinations of old features
- General form:  $\text{function}(f_1(x), f_2(x), f_3(x), \dots)$
- Example:  $n(x) = \text{sine}(f_1(x))$
- Example:  $n(x) = f_1(x) * f_2(x)$
- $ax^2 + bx + c =$  three features  $(x^2, x, 1)$

# Searching For a Solution

- Margin solution space is convex
- There is one maximum
- Solution can be found by hill-climbing
  
- Hyperparameter space is not convex
- There can be more than one maximum
- Hill-climbing may get the wrong hill

# PAC Bounds for SVMs

- PAC = Provably Approximately Correct
- Bounds the error: “The probability that the training data set gives rise to a hypothesis with large error on the test set is small.”
- Doesn't assume anything about distribution
- Doesn't assume right function can be learned
- Assumes training distribution = test distribution
- Called *distribution-free*

# PAC Bounds: Problems

- The PAC bound is often loose
- The bound on error rate is  $>100\%$  unless the training set is large
- Often, training distribution  $\approx$  test distribution
- In spite of this, SVMs often work well

# Appeal of SVMs

- Intuitive geometric interpretation
- Distribution-free
- Novel PAC bounds
- Works well

# Log-linear Models

- Also known as logistic-regression, exponential models, Markov Random Fields, softmax regression, maximum likelihood, and maximum entropy
- Probability-based approach (Bayesian)
- $\text{score}_i(x)$  = weighted sum of features
- $\text{Prob}(i|x) = e^{\text{score}_i(x)} / \sum_j e^{\text{score}_j(x)}$
- Maximizes the likelihood of the data
- Maximizes the entropy of what is unknown
- Results in good feature weights

# Regularized Log-linear Models

- Regularization = smoothing
- Unregularized log-linear models overtrain
- SVMs do better
- L1 regularization adds a linear penalty for each feature weight (Laplacian prior)
- L2 regularization adds a quadratic penalty for each feature weight (Gaussian prior)

# Regularized Log-linear Models (2)

- Both L1 and L2 regularization multiply weight penalty by a constant
- This constant is a hyperparameter
- A large constant is good for noisy data
- Constant can be chosen by cross-validation

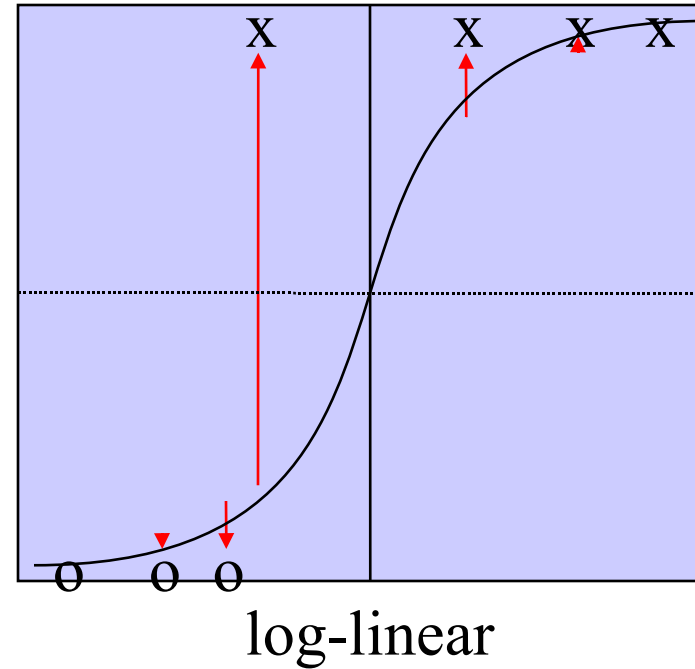
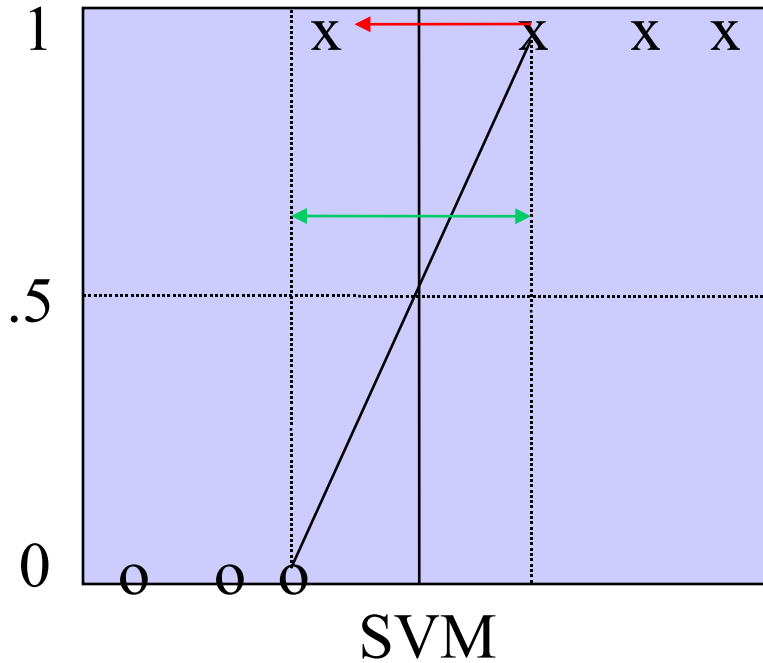
# L1 vs. L2 Regularization

- L1 ignores irrelevant features (Ng 2004)
- L2 and SVMs do not
- L1 sets many feature weights to zero
- L2 and SVMs do not
- L1 produces sparse models
- L1 produces human-understandable models
- L1 often produces better results because it reduces the degrees of freedom

# Log-linear Models vs. SVMs

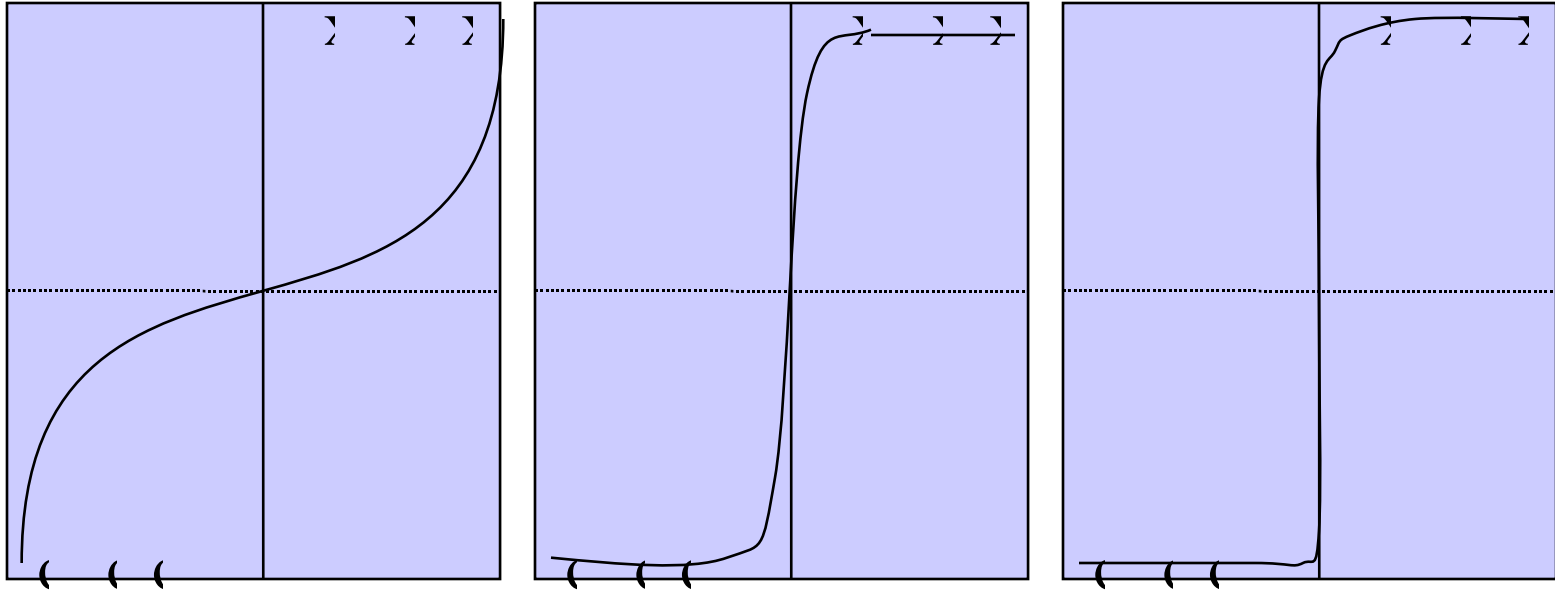
- Different loss functions
- Different statistical bounds (Law of Large Numbers vs. PAC Bounds)
- Log-linear models are probability models
- Log-linear models are optimized for Gaussian distribution, SVMs are distribution-free
- Log-linear models work in feature space, SVMs work in the dual space of data points

# Different Loss Functions



- 1 dimensional case
- $x$  = feature,  $y$  = class (0 = o class, 1 = x, .5 = split)
- Maximize **margin**, minimize **loss**

# Different Loss Functions (2)



- Separable case
- As feature penalty goes to zero, log-linear maximizes margin
- Unregularized log-linear models are large margin classifiers (Rosset et. al. 2003)

# Different Statistical Bounds

- PAC bounds can be derived for log-linear models (*PAC-Bayes*)
- Log-linear model PAC bounds are better than SVM PAC bounds (Graepel et. al. 2001)

# Probability Models

- Log-linear models compute the probability that a data point is a member of a class
- This can be useful (e.g. credit risk)
- It is easy to generalize log-linear models to more than two classes (MAP rule)
- It is not as easy to generalize SVMs to more than two classes (one-vs-rest, pairwise, others)

# Different Distribution Assumptions

- Log-linear models are optimal for Gaussian distributions (nothing can do better)
- Log-linear models are not that sensitive to the Gaussian assumption

# Feature Space vs. Data Space

- Kernelized support vectors can be features
- This allows data space and feature space to be compared
- Features produce larger margins than kernelized support vectors (Krishnapuram et. al. 2002)
- Larger margins => better results

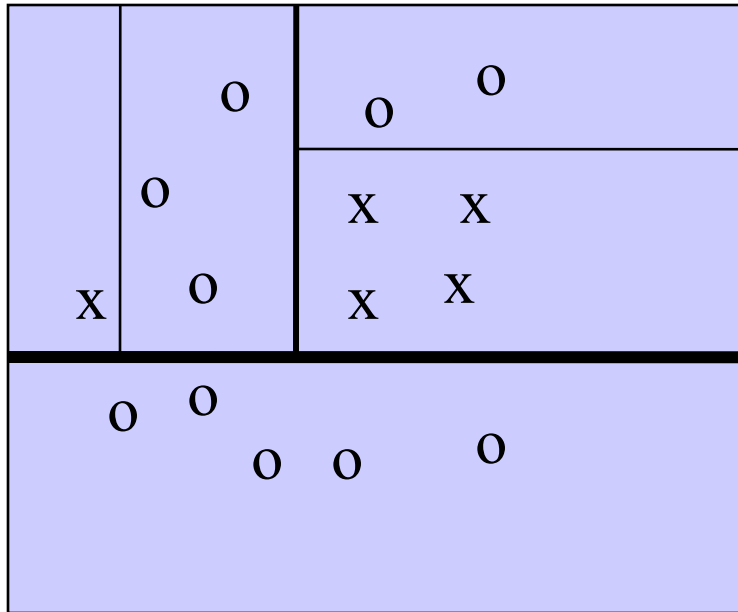
# Model Selection

- Kernel must be chosen in advance in SVMs (though the kernel can be very general)
- Log-linear models let you add models as features
- Correct model is selected by feature weights
- L1 regularization => many zero weights
- L1 regularization => understandable model

# Advantages of Log-linear Models

- Better PAC bound
- Larger margins
- Better results
- Probability model
- Better for multi-class case
- Optimal for Gaussian distributions
- Model selection
- Faster runtime

# Decision Trees



- Decision Trees (C4.5) successively subdivide the feature space

# Decision Trees (2)

- Decision Trees overtrain
- *Bagging* and *boosting* are techniques for regularizing Decision Trees
- Boosting is equivalent to unregularized log-linear models (Lebanon and Lafferty 2001)

# Decision Trees (3)

- Decision Trees keep improving with more data
- Log-linear models plateau
- If there is enough data, Decision Trees are better than log-linear models (Perlich et. al. 2003)
- Decision Trees grow their own features
- Given enough data, you don't need to regularize

# Degrees of Freedom

- If you have too few degrees of freedom for a data set, you need to add features
- If you have too many degrees of freedom for a data set, you need to regularize
- Decision Trees grow degrees of freedom
- Log-linear models can grow degrees of freedom by taking combinations of features, like Decision Trees

# Hyperparameters

- The hyperparameter for L1 log-linear models can be found efficiently using cross-validation (Park and Hastie 2006)
- The hyperparameter for L1 regularization is known if the data is assumed Gaussian (but the data isn't always Gaussian)

# Local Hyperparameters

- Local hyperparameters: one per feature weight
- Can be found efficiently by examining the optimization matrix (Chen 2006)
- Produce models with fewer features
- Do not need cross-validation
- Make use of all of the data
- Space of solutions is not convex
- Multiple solutions are possible

# Conclusions

- Log-linear models seem better than SVMs
- Log-linear models should add feature combinations
- Local hyperparameters may improve log-linear models

# References

- Chen, S. "Local Regularization Assisted Orthogonal Least Squares Regression". Neurocomputing 69(4-6) pp. 559-585. 2006.
- Christiani, N. and Shawe-Taylor, J. "An Introduction to Support Vector Machines". Cambridge University Press. 2000.
- Graepel, T. and Herbrich, R. and Williamson, R.C. "From Margin to Sparsity". In Advances in Neural Information System Processing 13. 2001.
- Krishnapuram, B. and Hartemink, A. and Carin, L. "Applying Logistic Regression and RVM to Achieve Accurate Probabilistic Cancer Diagnosis from Gene Expression Profiles". GENSIPS: Workshop on Genomic Signal Processing and Statistics, October 2002.
- Lebanon, G. and Lafferty, J. "Boosting and Maximum Likelihood for Exponential Models". In Advances in Neural Information Processing Systems, 15, 2001.
- Ng, A. "Feature selection, L1 vs. L2 regularization, and rotational invariance". In Proceedings of the Twenty-first International Conference on Machine Learning. 2004.
- Park, M. and Hastie, T. "L1 Regularization Path Algorithm for Generalized Linear Models". 2006.
- Perlich, C. and Provost, F. and Simonoff, J. "Tree Induction vs. Logistic Regression: A Learning-Curve Analysis". Journal of Machine Learning Research 4 211-255. 2003.
- Quinlan, R. C4.5: Programs for Machine Learning. Morgan Kaufman, 1993.
- Rosset, S. and Zhu, J. and Hastie, T. "Margin Maximizing Loss Functions" In Advances in Neural Information Processing Systems (NIPS) 15. MIT Press, 2003.