Support Vector Machines in Analysis of Top Quark Production

Tony Vaiciulis
University of Rochester

Advanced Statistical Methods in Particle Physics, Durham,UK
March 21, 2002

- introduction
- hyperplane classifiers
- feature spaces and kernels
- application in top quark physics
- summary

Introduction

Common HEP problem is classification

- signal vs background event
- particle identification

no explicit method to determine correct output from input data

Devise algorithm exploiting complex patterns in input/output examples (labeled training data) to learn the solution to the problem
$\rightarrow$ "supervised learning"

map each training input/output pair onto

- two categories $\rightarrow$ "binary classification"
- more than two categories $\rightarrow$ "multi-class classification"
- continuous, real-valued output $\rightarrow$ "regression"

Introduction

possible goal: modify algorithm until all training data are classified correctly

potential problems

- noisy training data.
  $\rightarrow$ no correct underlying classification function?
  $\rightarrow$ similar training examples may be in different categories
- misclassification of unseen data
  $\rightarrow$ "overfitting"

Introduction

better goal: optimize “generalization” – ability to correctly classify unseen data

Keep it simple: extra complexity must cause significant improvement in performance.

problems:

- mistakes due to local minima
- small training set $\rightarrow$ overfitting
  (when using complex mapping function)
- large number of tunable parameters

Support Vector Machines (SVMs) address these problems.

History

history

- early 1960s: Support vector method developed to construct separating hyperplanes for pattern recognition problems (Vapnik and Lerner 1963; Vapnik and Chervonenkis 1964)
- early 1990s: generalized for constructing non-linear separating functions (but linear in feature space). (Boser et al. 1992; Cortes and Vapnik 1995)
- 1995: generalized for estimating real-valued functions (regression) (Vapnik 1995)
- development continues today

current events

- special SVM issue of Neural Computation journal. 2002.
- NATO Advanced Study Institute on Learning Theory and Practice, July, 2002.
- International Workshop on Practical Application of Support Vector Machines in Pattern Recognition, August, 2002.
- International Conference on Neural Information Processing, Special Session on Support Vector Machines, November, 2002.

SVM Overview

some applications:

- text categorization
- character recognition
- face detection
- tuberculosis diagnosis

SVM quick overview

- map training data into a high dimensional feature space
- determine decision boundary in feature space by constructing the optimal separating hyperplane
- avoid computations in feature space by using a kernel function
- use concepts from statistical learning theory to describe which factors have to be controlled for good generalization

Generalization and Capacity

Formal goal: estimate function $f : \Re^N \to \{\pm 1\}$ using input/output training data

$$(\mathbf{x}_1, y_1), ..., (\mathbf{x}_\ell, y_\ell) \in \Re^N \times \{\pm 1\}$$

such that f will correctly classify unseen examples $(\mathbf{x}, y)$, i.e. $f(\mathbf{x}) = y$.
ℓ = number of training examples.

For good generalization, restrict class of functions from which f is chosen. Minimization of training error,

$$(1/\ell) \Sigma_i \mid f(\mathbf{x}_i) - y_i \mid,$$

does not imply good generalization.

More precise: restrict class of functions to one with a "capacity" suitable for the amount of available training data.

"capacity" is richness or flexibility of the function class

$$\text{low capacity} \to \text{good generalization}$$

regardless of the dimensionality of the space (assuming function describes the data well)

Hyperplane Classifiers

Support Vector classifiers are based on the class of hyperplanes

$$(\mathbf{w} \cdot \mathbf{x}) + b = 0$$

with $\mathbf{w} \in \Re^N$, $b \in \Re$ corresponding to the decision function

$$f(\mathbf{x}) = sign[(\mathbf{w} \cdot \mathbf{x}) + b]$$

w is "weight vector", b is "threshold"
w and b: control the function, must be learned from the data.

Note: here we consider linear case

Find the (unique) optimal hyperplane

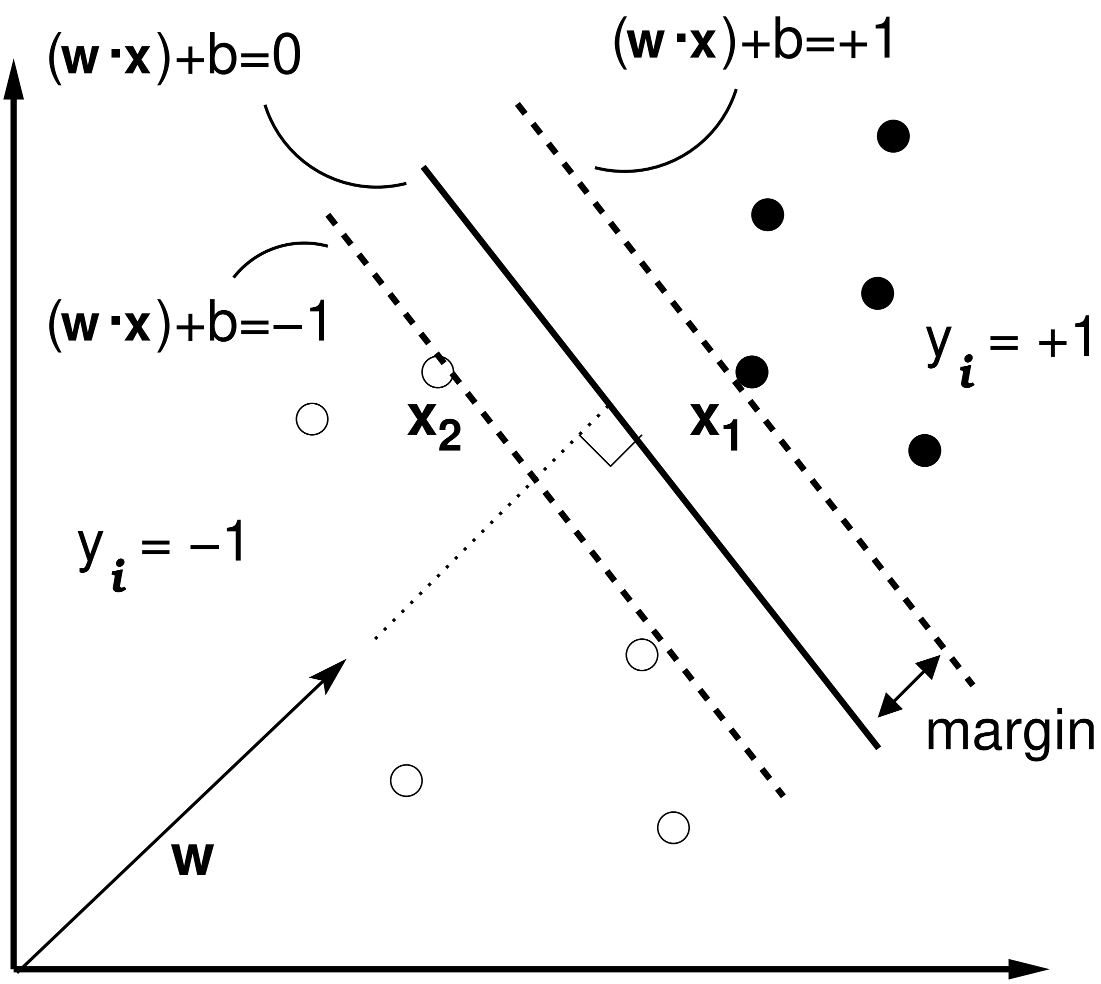
- hyperplane with maximal margin of separation between the two classes (by definition)
- has the lowest capacity of all hyperplanes
- minimizes risk of overfitting
  (different than "intuitive" way of decreasing capacity by reducing degrees of freedom)

Hyperplane Classifiers

geometric interpretation: hyperplane splits input space into two parts, each one corresponding to a different class.
margin size $= 2/||\mathbf{w}||$

To find optimal hyperplane minimize $||\mathbf{w}||^2$ subject to constraints (for $i = 1, \ldots \ell$)

$$y_i((\mathbf{w} \cdot \mathbf{x}_i) + b) \geq 1$$



Hyperplane Classifiers

$\mathbf{x}_i$ for which equality holds are "support vectors"

- carry all information about the problem
- removal would change the solution
- lie on a hyperplane defining the margin

To solve constrained quadratic optimization problem, first reformulate it in terms of a Lagrangian,

$$\mathcal{L}(\mathbf{w}, b, \alpha) = (1/2)||\mathbf{w}||^2 - \Sigma_i\, \alpha_i[y_i((\mathbf{x}_i \cdot \mathbf{w}) + b) - 1]$$

$\rightarrow$ easier to handle constraints on the Lagrange multipliers, α_i ($\alpha_i \geq 0$)

$\rightarrow$ training data only appear in dot products (to generalize to nonlinear case)

Need to minimize $\mathcal{L}$ w.r.t $\mathbf{w}$ and b (*primal variables*)
Need to maximize $\mathcal{L}$ w.r.t α_i (*dual variables*)

Hyperplane Classifiers

Solution has expansion in terms of a subset of input vectors (with $\alpha_i \neq 0$) called Support Vectors

$$\mathbf{w} = \Sigma_i \, \alpha_i y_i \mathbf{x}_i$$

$\alpha_i \sim$ difficulty in classifying the point (small α_i means easy classification)

Optimization problem in *dual* form: find α_i which maximize

$$\mathcal{L} = \Sigma_i \, \alpha_i - (1/2) \, \Sigma_{i,j} \, \alpha_i \alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j$$

subject to constraints $\alpha_i \geq 0$ and $\Sigma_i \, \alpha_i y_i = 0$

Decision function is

$$f(\mathbf{x}) = sign[\Sigma_i \, y_i \alpha_i (\mathbf{x} \cdot \mathbf{x}_i) + b]$$

Key concept: input vectors enter as dot products only. Important for nonlinear case.

Note: form of optimization problem guarantees global minimum

Feature Spaces and Kernels

Possible methods if $f(\mathbf{x})$ is nonlinear function of $\mathbf{x}$

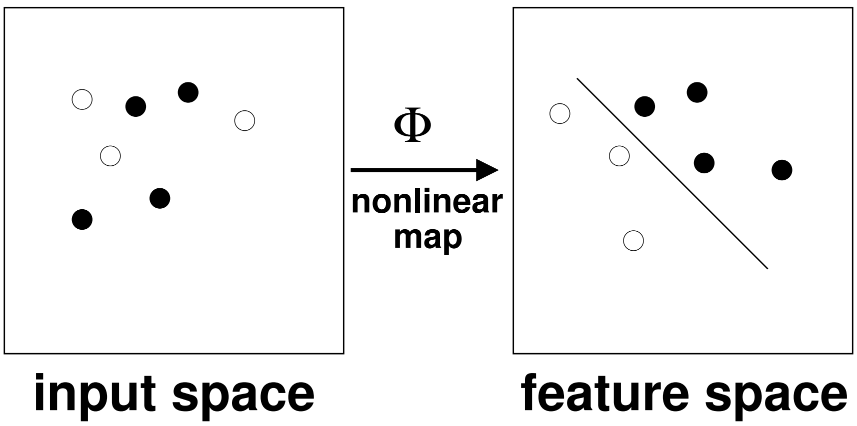
- neural network: use network of simple linear classifiers
  problems: many parameters, local minima, ...
- SVM: map input data into high dimensional feature space, $\mathcal{F}$, via nonlinear map $\Phi : R^N \rightarrow \mathcal{F}$ Use optimal hyperplane algorithm in $\mathcal{F}$.

high dimensionality $\rightarrow$ computational problem in feature space

Since input vectors appear only as dot products, we only need dot products in feature space. Find a kernel function, $\mathcal{K}$, such that

$$\mathcal{K}(\mathbf{x}_1, \mathbf{x}_2) = \Phi(\mathbf{x}_1) \cdot \Phi(\mathbf{x}_2)$$

Then we don't need to know Φ explicitly.



Feature Spaces and Kernels

How do we know a function is a kernel?

Mercer's Theorm
A function $\mathcal{K}(\mathbf{x}, \mathbf{y})$ is a kernel, i.e. there exists a mapping Φ such that

$$\mathcal{K}(\mathbf{x}_1, \mathbf{x}_2) = \Phi(\mathbf{x}_1) \cdot \Phi(\mathbf{x}_2)$$

if

$$\int \mathcal{K}(\mathbf{x}, \mathbf{y}) g(\mathbf{x}) g(\mathbf{y}) d\mathbf{x} d\mathbf{y} \geq 0$$

for all g such that $\int g(\mathbf{x})^2 d\mathbf{x}$ is finite.

But, does not tell us how to construct Φ.

Examples of kernel functions

- $\mathcal{K}(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y})^d$
  (polynomial of degree d)
- $\mathcal{K}(\mathbf{x}, \mathbf{y}) = \exp(- \parallel \mathbf{x} - \mathbf{y}^2 \parallel / (2\sigma^2))$
  (Gaussian Radial Basis Function)
- $\mathcal{K}(\mathbf{x}, \mathbf{y}) = \tanh(\kappa(\mathbf{x} \cdot \mathbf{y}) + \theta)$
  (sigmoid)

Nonlinear SVMs

SVMs

- substitute $\Phi(\mathbf{x}_i)$ for each training example $\mathbf{x}$
- substitute kernel for dot products of Φ

decision function

$$f(\mathbf{x}) = sign[\Sigma_i\, y_i \alpha_i \mathcal{K}(\mathbf{x}, \mathbf{x}_i) + b]$$

optimization problem: maximize

$$\mathcal{L} = \Sigma_i\, \alpha_i - (1/2)\, \Sigma_{i,j}\, \alpha_i \alpha_j y_i y_j \mathcal{K}(\mathbf{x}, \mathbf{x}_i)$$

subject to constraints $\alpha_i \geq 0$ and $\Sigma_i\, \alpha_i y_i = 0$

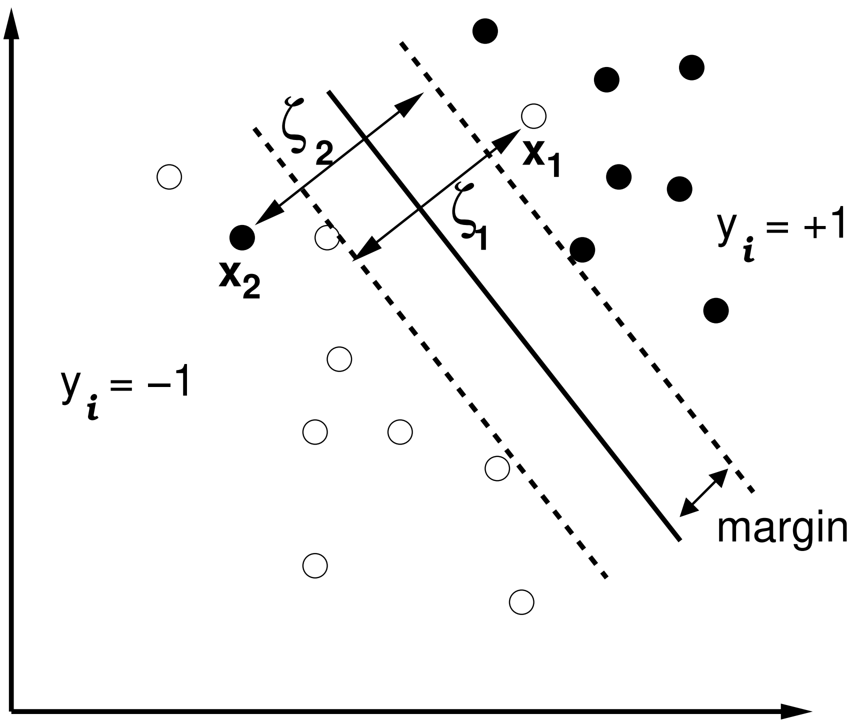
Nonseparable Data

Real world applications → nonseparable data

No linear separation in feature space (unless we use very complex kernel, leading to overfitting).

Optimization problem will not converge → Lagrangian grows arbitrarily large

Introduce slack variables ($\zeta_i \geq 0$) to allow points to violate constraints. No training error in separable case.



Nonseparable Data

The new, relaxed contraints are

$$y_i((\mathbf{w} \cdot \mathbf{x}_i) + b) \geq 1 - \zeta_i$$

$\zeta_i > 0$ means $\mathbf{x}_i$ is misclassified

Allows for some number of errors in optimization problem.

Good generalization is now achieved by controlling two things

- the capacity, as before (i.e. control margin size via $||\mathbf{w}||$)
- number of training errors

Redo Lagrangian formulation, adding a term
$C\Sigma_i \zeta_i$ where parameter C controls error penalty
(large C $\rightarrow$ larger penalty)

$\rightarrow$ same dual optimization problem and constraints as for separable case except

$$\alpha_i \geq 0 \;\rightarrow\; 0 \leq \alpha_i \leq C$$

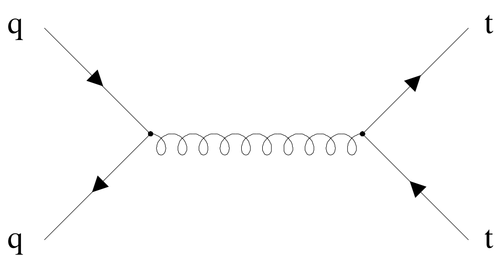
We can now handle nonlinear mappings for nonseparable data.

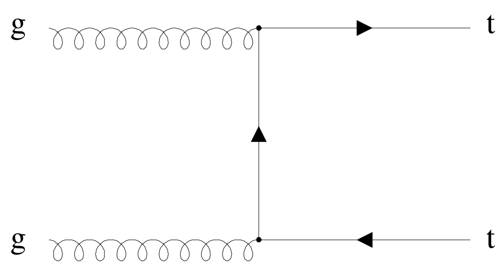
Top Quark Study

possible application of SVM in HEP:
improve signal/background discrimination in
$t\bar{t}$ dilepton channel

Note: this is a Monte Carlo study (no data)

Top production at the Tevatron: mainly $t\bar{t}$ pairs





90% 10%

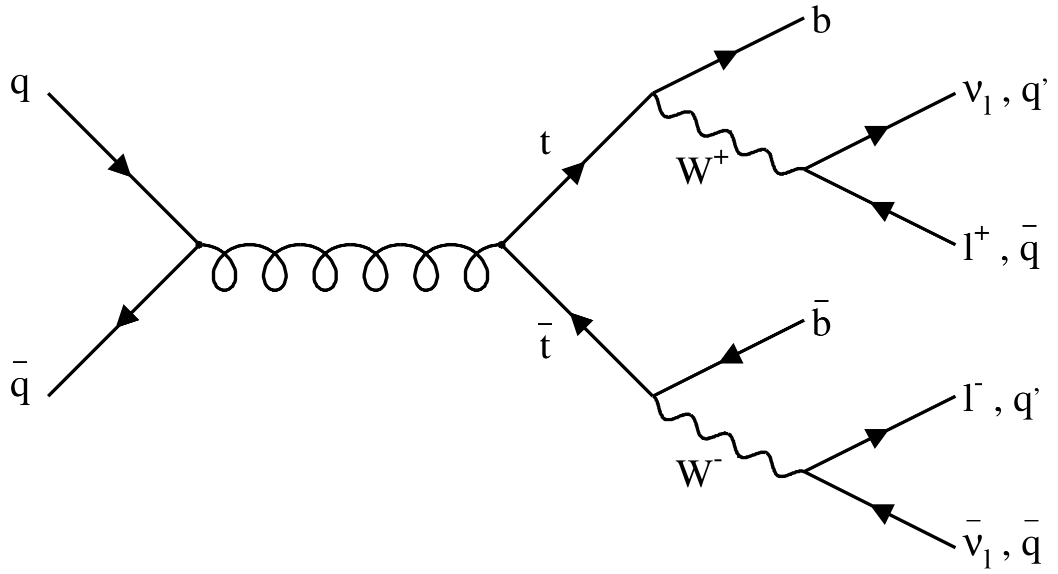
$\sigma(p\bar{p} \to t\bar{t}) = 5.0$ pb (theory)

$\rightarrow$ Small cross section!

- more than 10^{12} $p\bar{p}$ collisions in Run I data
- top measurements based on $\sim$100 events

Top Quark Study

In Standard Model, BR(t$\rightarrow W^+b$) $\sim$ 100%



Top channels distinguished by W decay modes

three event topologies:

$t\bar{t} \rightarrow b\bar{b}l\nu l\nu$	$\sim$5%	(4/81)	"dilepton"	$l = e, \mu$
$t\bar{t} \rightarrow b\bar{b}l\nu q\bar{q}$	$\sim$30%	(24/81)	"lepton + jets"	
$t\bar{t} \rightarrow b\bar{b}q\bar{q}q\bar{q}$	$\sim$44%	(36/81)	"all-hadronic"	

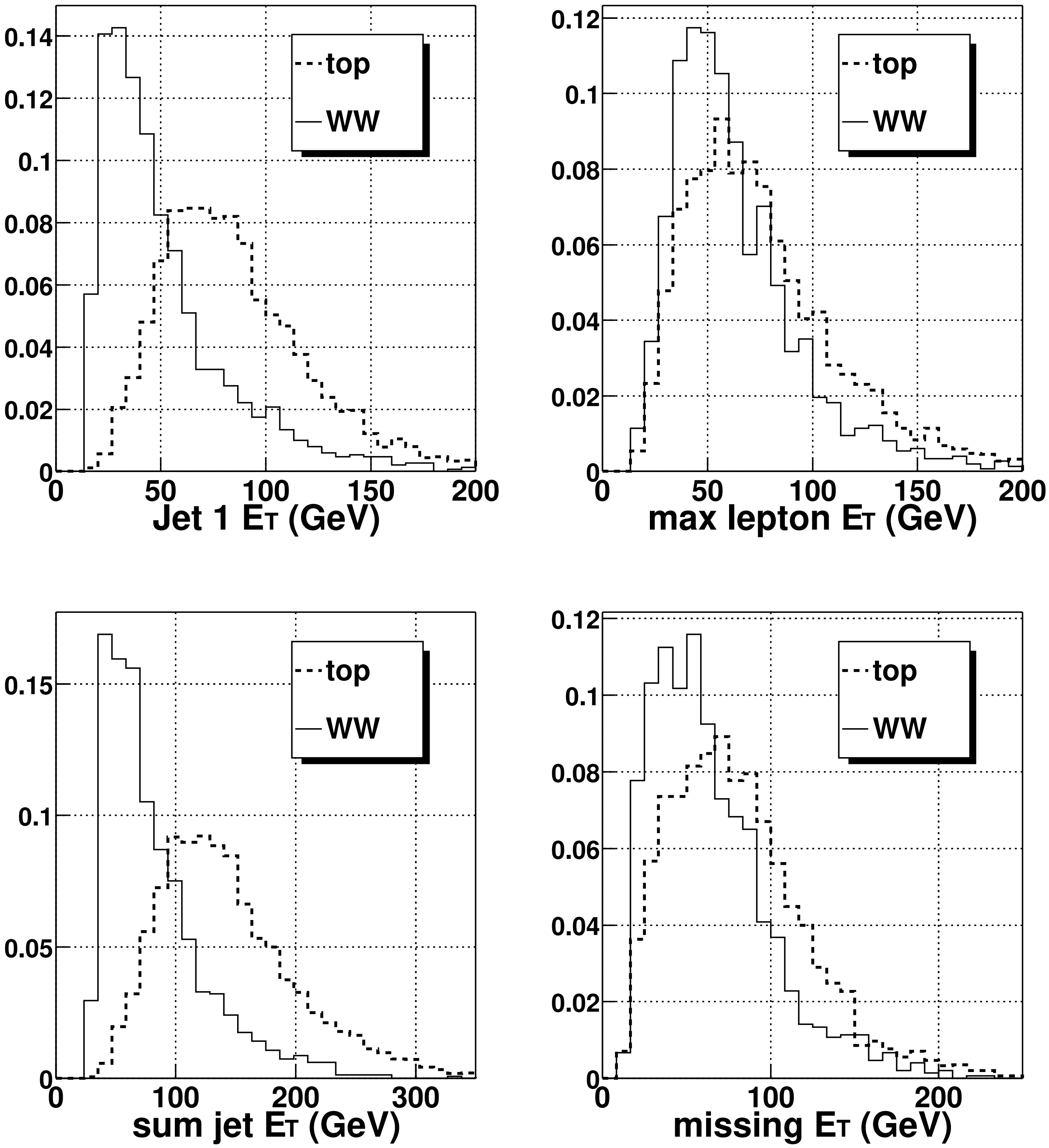
signatures

- "dilepton": 2 high-p_T isolated leptons, 2 b jets, large missing E_T
- "lepton + jets": 1 high-p_T isolated lepton, 4 jets, large missing E_T
- "all-hadronic": 6 jets

Goal: increase signal efficiency in $e\mu$ dilepton channel (low background) Backgrounds include: WW production, Z$\rightarrow \tau^+\tau^-$ Consider only WW in this study.

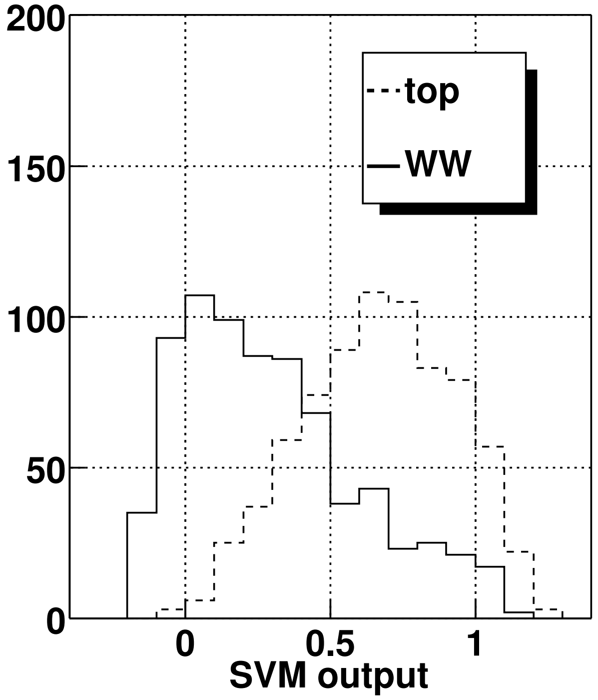
Top Quark Study

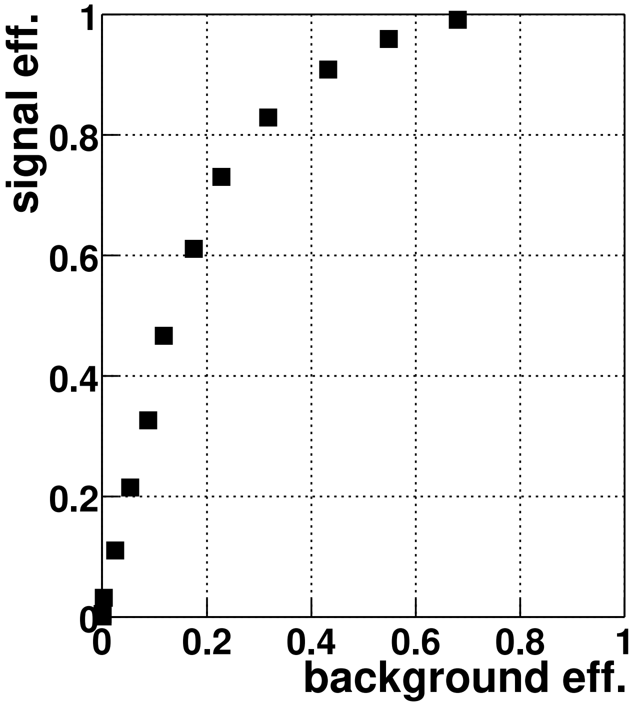
Monte Carlo: CompHEP + Pythia hadronization + PGS
First try: use some jet + lepton + missing E_T variables

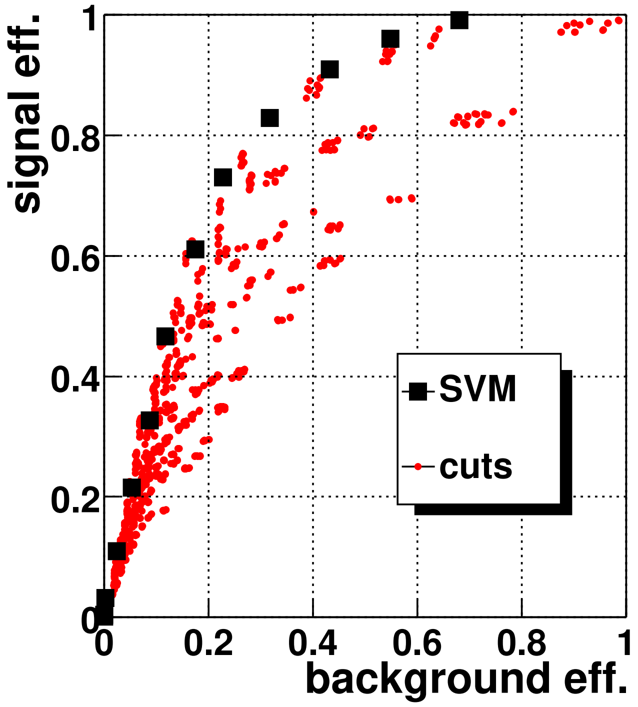


Top Quark Study

SVM performance $\sim$ performance of best cuts

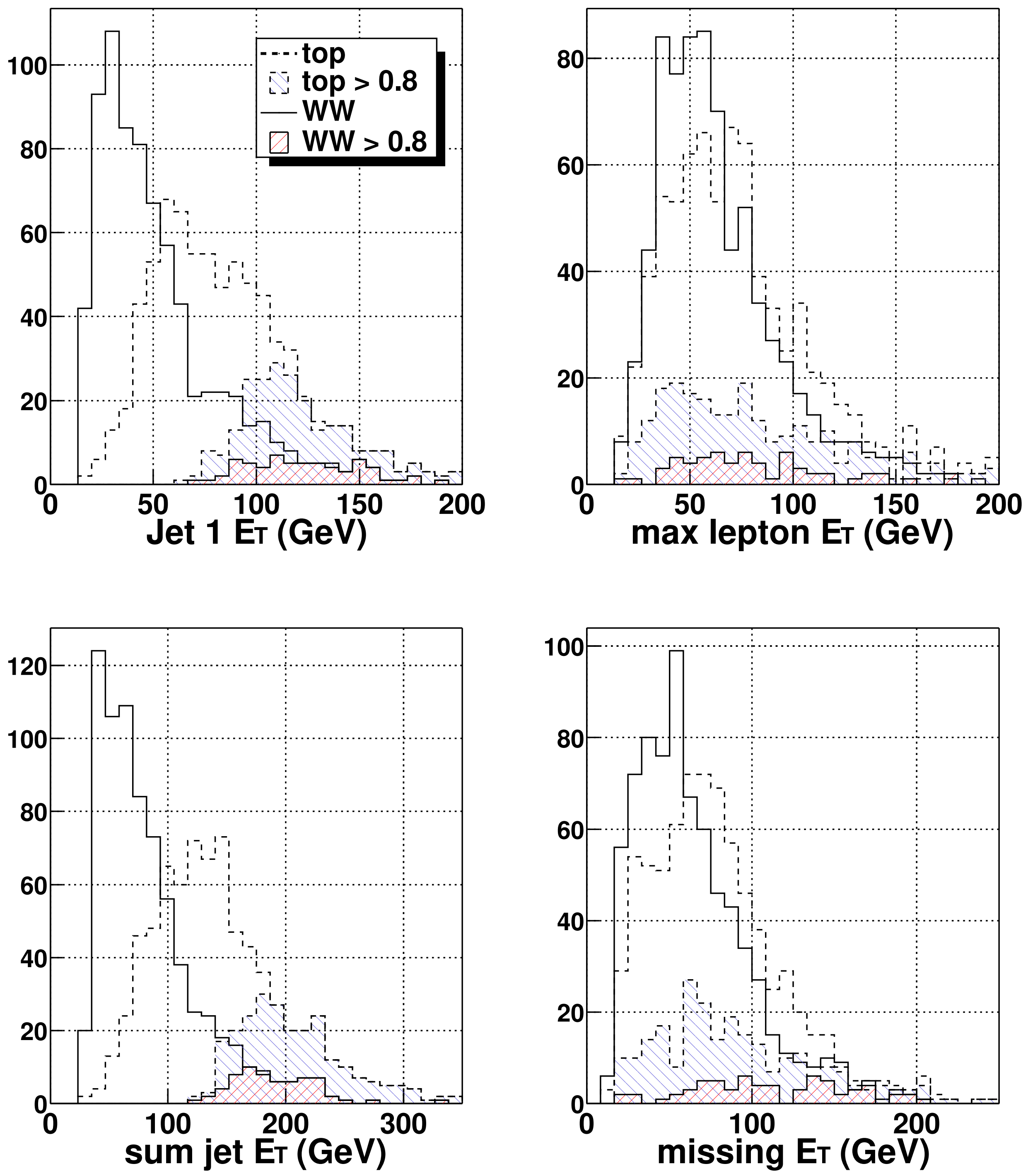






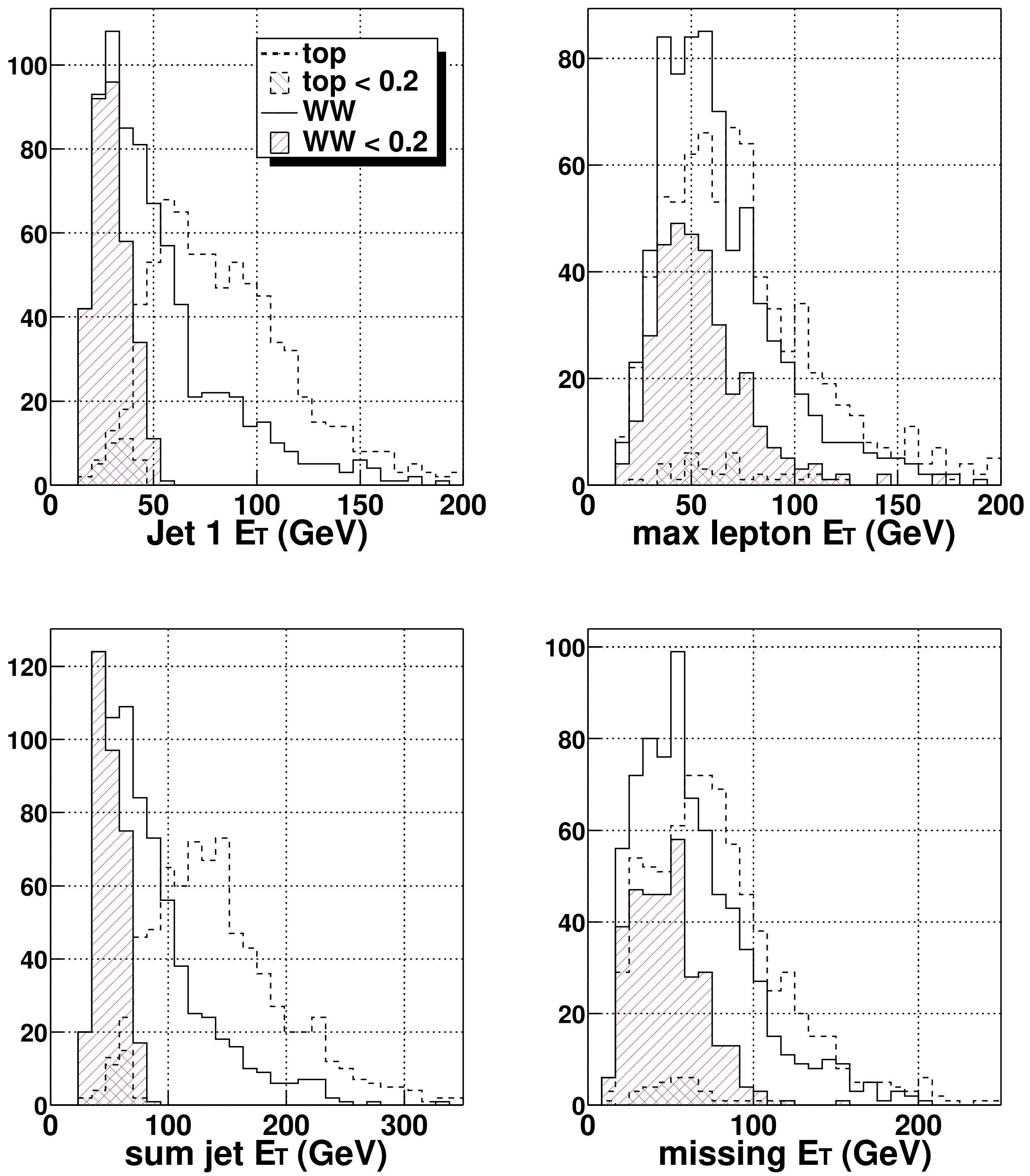
Top Quark Study

Where is SVM cut in input space? Cut on SVM output > 0.8.



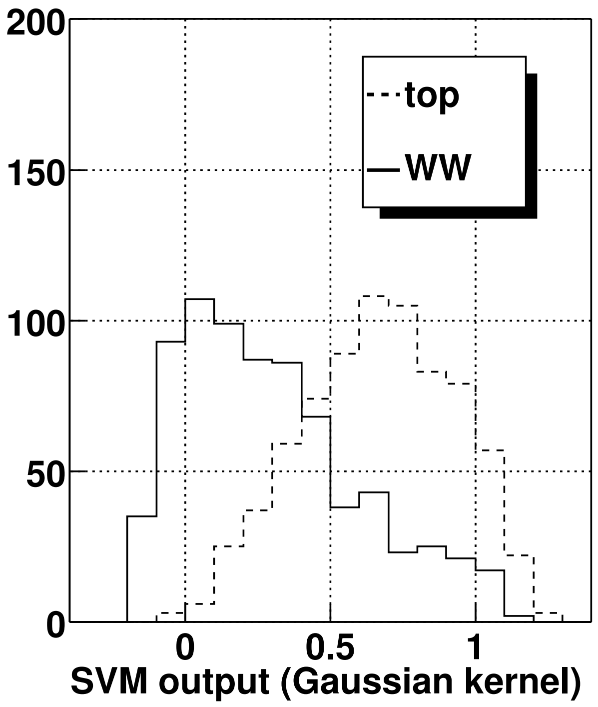
Top Quark Study

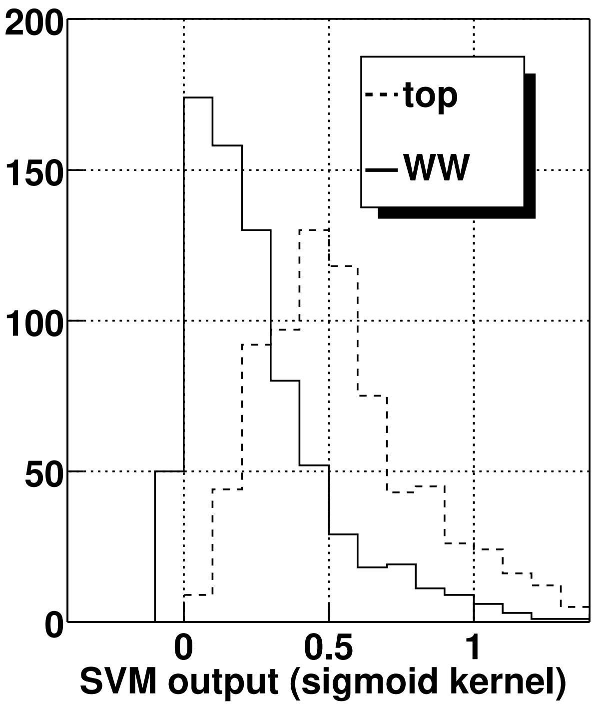
Cut on SVM output < 0.2.

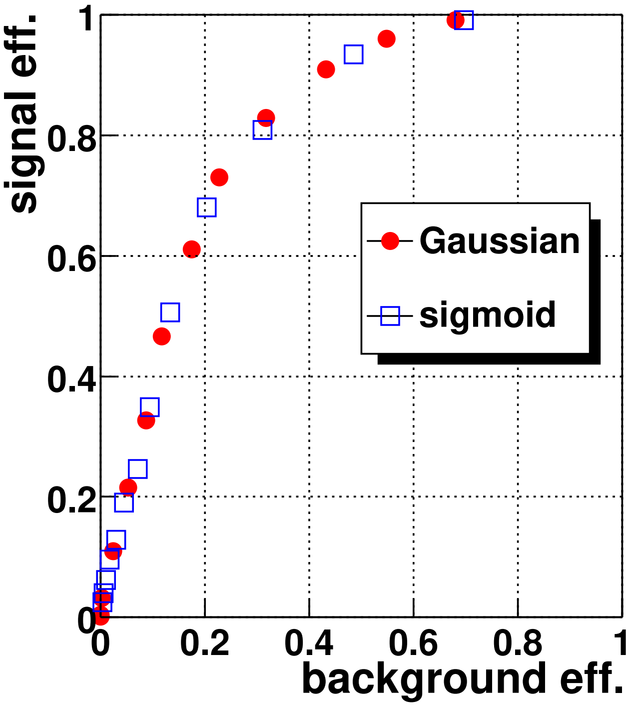


Top Quark Study

performance of Gaussian kernel $\sim$ sigmoid kernel

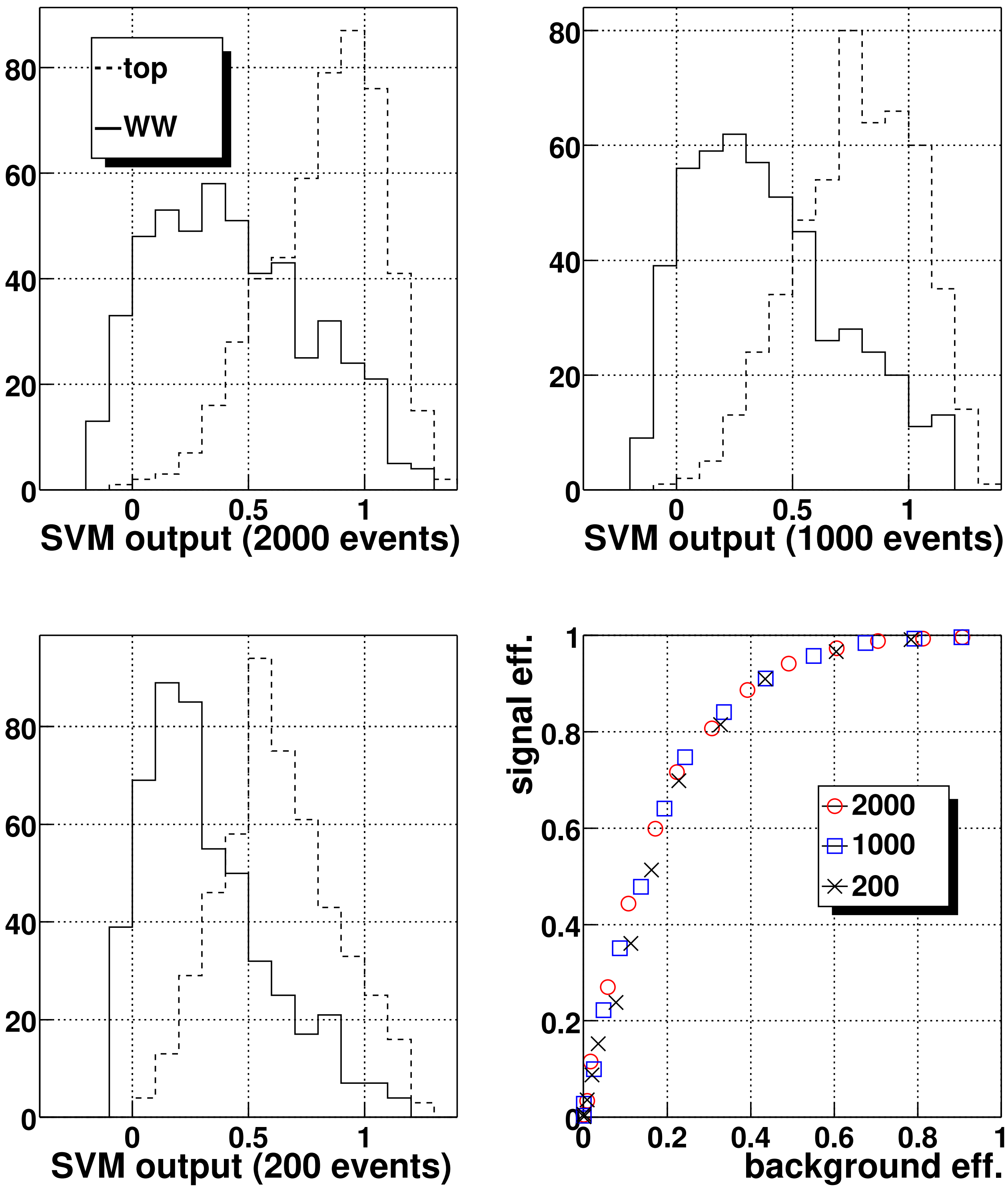






Top Quark Study

Little performance difference vs. number of training examples



Training Time

- for high statistics sample, use toy Monte Carlo $\rightarrow$ two variables (one gaussian, the other exponential)
- training done with gaussian kernel
- 1000 MHz Pentium III PC (Linux)
- LIBSVM uses a modified Sequential Minimal Optimization (SMO) algorithm – known to be a fast method to train SVMs

number of events	toy MC time (sec)	ttbar MC time (sec)
200	< 1	< 1
400	< 1	< 1
1000	< 1	1
2000	1.5	2
3000	4	6
4000	8	
8000	38	
16000	149	

For top quark study, training time was negligible.

References

- An Introduction to Support Vector Machines. Nello Cristianini and John Shawe-Taylor. Cambridge University Press. 2000.

- Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines, 2001. Software available at
  `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

- `http://www.kernel-machines.org/`

- A Tutorial on Support Vector Machines for Pattern Recognition. Christopher J.C. Burges.

- Advances in Kernel Methods. edited by Bernhard Schölkopf, Christopher J.C. Burges, Alexander J. Smola. MIT Press. 1999.

Summary

SVMs provide non-linear function approximations by mapping input vectors into a high dimensional feature space where a hyperplane is constructed to separate classes in the data.

- a new way to tackle complicated problems in HEP and other fields
- based on statistical learning theory
  $\rightarrow$ optimize generalization
  $\rightarrow$ more intuitive
- all computations performed directly in the input space (via kernels)
- correspond to a linear method in the feature space
  $\rightarrow$ theoretically easy to analyze
  $\rightarrow$ allows us to generalize some existing linear methods to non-linear methods

Summary

- only one user chosen parameter, the error penalty (aside from kernel parameters)
- well defined quadratic optimization problem – global minimum
- SVM development can build on existing work on other kernel based methods
- top quark study
  $\rightarrow$ reproduces best performance of cut based approach (only a first look)
- roughly, SVM performance $\sim$ ANN performance (not shown here)

$\rightarrow$ another method for our multivariate analysis tool box