Aalto University
School of Science
and Technology

# Molecular classification

Hongyu Su

Department of Information and Computer Science
Aalto University, School of Science and Technology
hongyu.su@aalto.fi

March 21, 2015

# Content

- Learning for classification
  - Statistic learning
  - Empirical risk minimization
  - Regularization
  - Support vector machines for single label classification
  - Kernel methods
- Molecular classification problems
  - Predicting anti-cancer potentials of molecules
  - Hierarchical molecular classification
- Multilabel classification with structured output learning model
  - Max-margin conditional random field
  - Random spanning tree approximation

# Learning for classification

## Example: dog vs. cat?

- We have 5000 pictures of dog and 5000 pictures of cat.


- Each picture is digitalized into $100 \times 100$ pixels, $\mathbf{x} \in R^{10000}$.
- Given a new picture, we want to answer: is it a dog or a cat?
- This is a Kaggle competition (https://www.kaggle.com/c/dogs-vs-cats)
- The idea is to test the CAPTCHA system is safe from machine learning attack.
- Simple for human, dog, or cat. It used to be difficult for machines, with 82.7% accuracy (Golle, 2008) before competition and 98.9% afterwards.

# Statistical learning

- Two random variables $\mathbf{x} \in \boldsymbol{\mathcal{X}}$ and $y \in \mathcal{Y}$ are jointly distributed according to some fixed but unknown distribution $P(\mathbf{x}, y)$.
- $\boldsymbol{\mathcal{X}} = \mathbb{R}^d$ is an input (instance) space, and $\mathcal{Y} = \{-1, +1\}$ is an output (label) space.
- Examples are given in pair $(\mathbf{x}, y) \in \boldsymbol{\mathcal{X}} \times \mathcal{Y}$ generated by sampling from distribution $P(\mathbf{x}, y)$.
- A hypothesis class $\mathcal{H}$ is a set of function to be explored, for example, a set of hyperplanes in the feature space $f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b = 0, \mathbf{w} \in \mathbb{R}^d$.
- The goal of statistical learning is to provide an estimator $f \in \mathcal{H} : \boldsymbol{\mathcal{X}} \to \mathcal{Y}$ that predicts the best output $y$ for an input $\mathbf{x}$.

# Loss function

- Loss function measures the goodness of an estimator $f(\mathbf{x})$ on a single training example $(\mathbf{x}_i, y_i)$

$$\ell(f(\mathbf{x}_i), y_i) : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+.$$

- It is a monotonic bounded function defined on an estimation and the ground truth.
- The simplest one is hamming loss, which return 1 if the estimation is the same as the ground truth, 0 otherwise.

$$\ell_{hamming}(f(\mathbf{x}_i), y_i) = \mathbf{1}_{\{\mathbf{y}_i = f(\mathbf{x}_i)\}}$$

- Many other loss functions: hinge loss in support vector machines (Cortes and Vapnik, 1995), 0/1 loss in structured SVM (Tsochantaridis et al., 2004), squared loss in ridge regression (Hoerl and Kennard, 2000), exponential loss in ADABOOST (Schapire and Singer, 1999), logistic loss in logistic regression (Chen and Rosenfeld, 1999).

# Empirical risk minimization

- In the end, we want to minimize the true risk of an estimator over a domain $\mathcal{X} \times \mathcal{Y}$

$$R_{true}(f) = \int_{(\mathbf{x},y)\in\mathcal{X}\times\mathcal{Y}} \ell(y, f(\mathbf{x}))P(\mathbf{x}, y)\, d_{\mathbf{x}} d_y. \quad \text{Test error}$$

- It is impossible to compute the true risk directly, as $P(\mathcal{X}, \mathcal{Y})$ is known.
- We are given a set of $m$ training examples $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$.
- The empirical risk of an estimator $f \in \mathcal{H}$ over $\mathcal{S}$ is

$$R_{emp}(f) = \frac{1}{m}\sum_{i=1}^{m} \ell(y_i, f(\mathbf{x}_i)). \quad \text{Training error}$$

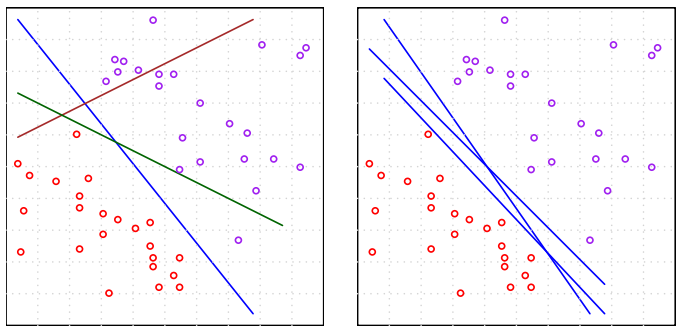
- Empirical risk minimization strategy (Vapnik, 1992) suggests we should search for an estimator that minimizes the empirical risk with regards to the training data.
- Vapnik (1992) showed that true risk is upper bounded by empirical risk

$$R_{true}(f) \leq R_{emp}(f) + \sqrt{\frac{h(\log(\frac{2m}{h}+1) - \log(\frac{\eta}{4}))}{m}}.$$

# Empirical risk minimization

## An example

- Training examples from two classes are located in 2 dimensional space.
- We are searching for a linear function (hyperplane) to separate two classes.
- Empirical minimization principle is applied.



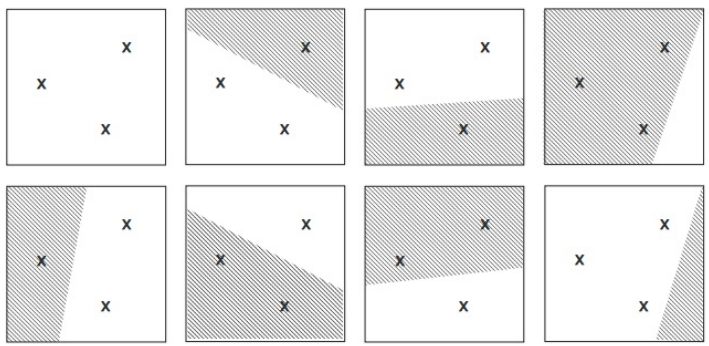
- We will not get a unique solution!

# Regularized learning

- Empirical risk minimization is
  - ill-posed, no unique solution.
  - overfitting, high dimensional feature space, small number of examples.
- Regularization theory (Evgeniou et al., 1999, 2002) aims to minimize

$$\mathcal{J}(f) = \underbrace{\mathcal{R}_{emp}(f)}_{\text{empirical risk}} + \underbrace{\lambda \cdot \Omega(f)}_{\text{regularization}},$$

  the latter part controls the complexity (capacity) of the function $f$.
- The complexity is defined by its VC dimension as the maximum number of points that can be separated by all possible way by the set of functions.
- For example, the VC dimension of a hyperplane in $\mathbb{R}^2$ is 3.



# Regularizer

- Vapnik (1992) shown that, the set of decision functions $f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b = 0$ defined on $\boldsymbol{\mathcal{X}}$ such that $||\mathbf{w}|| \leq A$ has a VC dimension $h \leq R^2 A^2$.
- This suggests we should minimize the norm of the feature weight $||\mathbf{w}||$ in order to control the complexity of the linear function class.
- Popular regularizers to control the complexity
  - $L_2$-norm regularizer, ridge regression (Hoerl and Kennard, 2000), logistic regression (Chen and Rosenfeld, 2000), and support vector machines (Cortes and Vapnik, 1995)

$$\Omega_{L_2}(f) = ||\mathbf{w}||_2 = \left( \sum_{i=1}^{d} |\mathbf{w}[i]|^2 \right)^{\frac{1}{2}}.$$

  - $L_1$-norm regularizer, LASSO (Tibshirani, 1994)

$$\Omega_{L_1}(f) = ||\mathbf{w}||_1 = \sum_{i=1}^{d} |\mathbf{w}[i]|.$$

- Other regularizer: graph regularizer (Zhou et al., 2004).

# Support Vector Machines

## History

- Support Vector Machines (SVMs) was introduced by Boser et al. (1992); Cortes and Vapnik (1995).
- SVM is firmly rooted from statistical learning theory (Vapnik, 1998).
- Good empirical performance in many applications (e.g., bioinformatics, computer vision, text classification).
- A large and diverse communities: e.g., machine learning, optimization, learning theory, statistics, applied computer science.
- Centralized website at http://www.kernel-machines.org
- Text books e.g.,
  - An introduction to support vector machines and other kernel-based learning methods by Cristinini and Shawe-Taylor (2000).
  - Kernel methods for pattern analysis by Shawe-Taylor and Cristianini (2004).

# Support Vector Machine

## Preliminaries

- We focus on classification, though SVMs can do more than classification.
- Examples are given in pairs $(\mathbf{x}, y) \in \boldsymbol{\mathcal{X}} \times \mathcal{Y}$, where $\mathbf{x} \in \boldsymbol{\mathcal{X}}$ is an object and $y \in \mathcal{Y}$ is a class label.
- $\boldsymbol{\mathcal{X}}$ is an input space $\boldsymbol{\mathcal{X}} = \mathbb{R}^d$.
- We consider binary classification for now, setting $\mathcal{Y} = \{-1, +1\}$.
- A training set of $m$ training example $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$.
- We want to learn a mapping function $f : \boldsymbol{\mathcal{X}} \to \mathcal{Y}$.
- The mapping function is a hyperplane in the $d$-dimension feature space $\boldsymbol{\mathcal{F}}$

$$f(x) = \langle \mathbf{w}, \mathbf{x} \rangle + b = 0.$$

- For an unknown example $\mathbf{x}_0$, the predicted label is given by

$$\mathbf{sign}(\langle \mathbf{w}, \mathbf{x}_0 \rangle + b).$$

# Support Vector Machines

## Regularized risk minimization

- We want to minimize a function

$$\mathcal{J}(f) = \underbrace{\mathcal{R}_{emp}(f)}_{\text{empirical risk}} + \underbrace{\lambda \cdot \Omega(f)}_{\text{regularization}}$$

- Use hamming loss $\ell_{hamming}$ and $L_2$ regularizer

$$\mathcal{J}(f) = \underbrace{\frac{1}{m}\sum_{i=1}^{m} \ell_{hamming}(f(\mathbf{x}_i), y_i)}_{\text{empirical risk}} + \underbrace{||\mathbf{w}||^2}_{\textit{regularization}} .$$

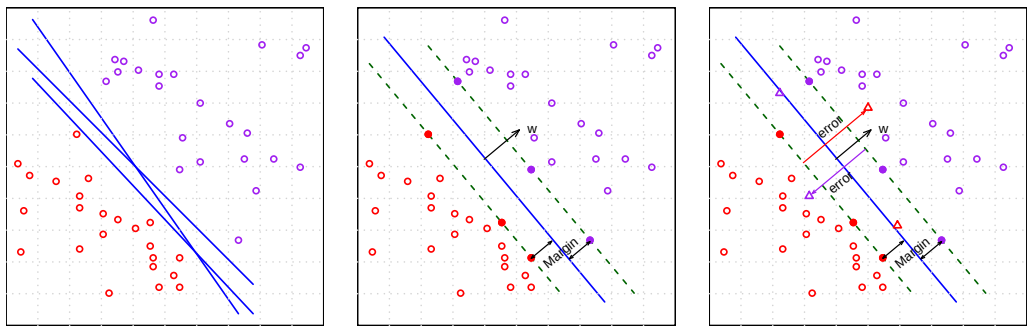
- $h_{hamming}(f(\mathbf{x}_i), y_i) = 1$ if $f(\mathbf{x}_i) \neq y_i$, 0 otherwise.
- Assume we can perfectly separate the data, we can rewrite the optimization problem as

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}||\mathbf{w}||^2 \\ \text{s.t.} \quad & y_i(\langle \mathbf{w}_i, \mathbf{x}_i \rangle + b) \geq 1, \forall i \in \{1, \cdots, m\}. \end{aligned}$$

- This is known as hard-margin SVMs.

# Geometric interpretation

- We are actually looking for a hyperplane with maximum margin between examples of two classes.
- Max-margin principle
  - Robustness: small perturbation in training data does not change the classification, decision boundary only depends on the support vectors.
  - Performance: large margin lead to low error on unseen data.
- What if there is no perfect separation?



# Soft-margin SVMs

- For non-separable case, we write the optimization problem as

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{m}\xi_i \\ \text{s.t.} \quad & y_i\left(\langle \mathbf{w}, \mathbf{x}_i\rangle + b\right) \geq 1 - \xi_i, i \in \{1, \cdots, m\}, \xi \geq 0. \end{aligned}$$

- This is known as soft-margin SVMs.
- It is the same as the original objective

$$\mathcal{J}(f) = \underbrace{\frac{1}{m}\sum_{i=1}^{m}\ell_{hinge}(f(\mathbf{x}_i), y_i)}_{\text{empirical risk}} + \underbrace{||\mathbf{w}||^2}_{\textit{regularization}} .$$

- Hamming loss is replaced by hinge loss

$$\ell_{hinge}(f(\mathbf{x}_i), y_i) = \mathbf{max}(0, 1 - y_i(f(\mathbf{x}_i) + b)).$$

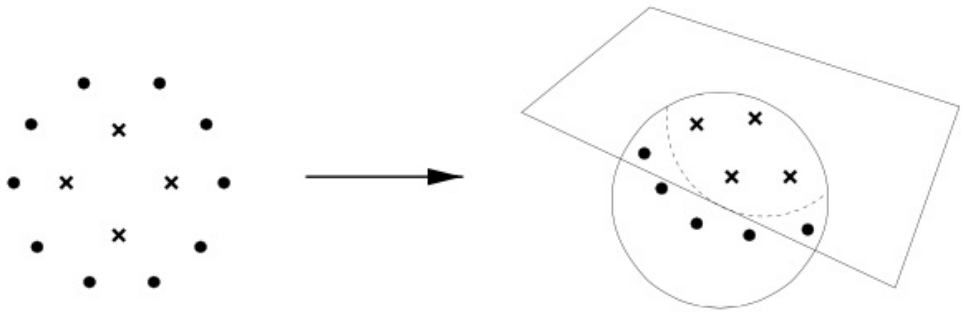
# Non-linear SVMs

- Data are not linearly separable in the original feature space.
- Linear functions are sometimes not powerful enough.
- SVMs solution: map data into a richer feature space including non-linear features, then construct a hyperplane in that space.
- Define a input feature map

$$\mathbf{x} \in \boldsymbol{\mathcal{X}} = \mathbb{R}^d \xrightarrow{\varphi} \boldsymbol{\varphi}(\mathbf{x}) \in \boldsymbol{\mathcal{H}} = \mathbb{R}^{d'}.$$

- Learn the map from $\boldsymbol{\varphi}(\mathbf{x})$ to $y$

$$f(\mathbf{x}) = \langle \mathbf{w}, \boldsymbol{\varphi}(\mathbf{x}) \rangle + b, \mathbf{w} \in \mathbb{R}^{d'}$$



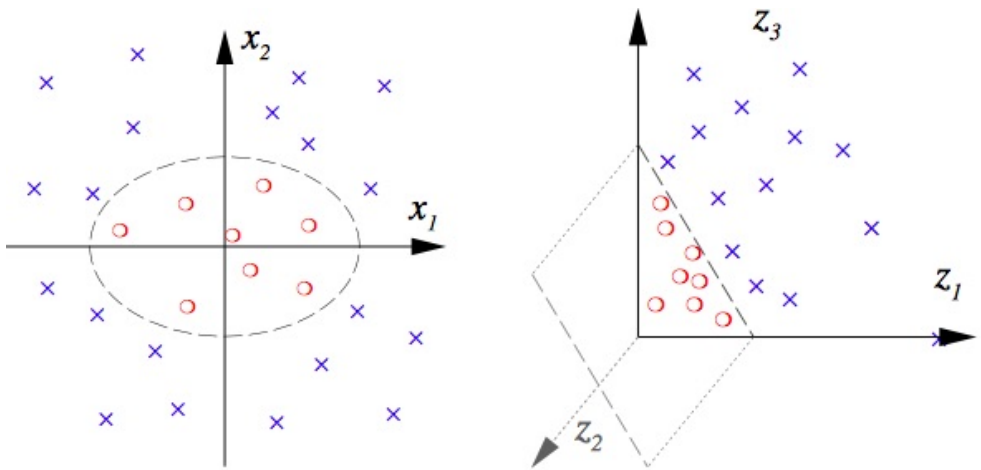
# Example: polynomial feature map

- Data points locate in two dimensional space, $\mathbf{x} \in \mathbb{R}^2$.
- We cannot separate the data point with a line.
- Define a feature map:

$$\varphi : \mathbb{R}^2 \to \mathbb{R}^3$$

$$\mathbf{x} = (x_1, x_2) \to \varphi(\mathbf{x}) \stackrel{\text{def}}{=} (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

- We can separate the data points with a hyperplane in the high dimensional feature space $\mathbb{R}^3$.



# Kernel methods

- Dual form of SVMs

$$\max_{\alpha} \quad \sum_{i=1}^{m} \alpha_i - \sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \underbrace{\langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j) \rangle}_{k(\mathbf{x}_i, \mathbf{x}_j)}$$

$$\text{s.t.} \quad 0 \leq \alpha_i \leq C, \forall i \in \{1, \cdots, m\}.$$

- We do not need to know $\varphi(\mathbf{x}_i)$ and $\varphi(\mathbf{x}_j)$, we only need the results of the inner product $\langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j) \rangle$.
- Kernel is a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j) \rangle,$$

  $\varphi : \mathcal{X} \to \mathcal{H}$ is the input feature map.
- Kernel function is used to make (implicit) nonlinear feature map.
- Kernel function can be evaluated in original space $\mathcal{X}$ without working explicitly in high the dimensional space $\mathcal{H}$.

# An example of kernel function

- Degree two polynomial kernel

$$k(\mathbf{x}, \mathbf{z}) = (\langle \mathbf{x}, \mathbf{z} \rangle)^2, \mathbf{x}, \mathbf{z} \in \mathbb{R}^2.$$

- Evaluate kernel function in original feature space $\mathbb{R}^2$

$$\begin{aligned} k(\mathbf{x}, \mathbf{z}) &= (\langle (x_1, x_2), (z_1, z_2) \rangle)^2 \\ &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 x_2 z_1 z_2 \end{aligned}$$

- Explicit feature map

$$\mathbf{x} = (x_1, x_2) \in \mathbb{R}^2 \xrightarrow{\boldsymbol{\varphi}} \boldsymbol{\varphi}(\mathbf{x}) = (x_1^2, \sqrt{2} x_1 x_2, x_2^2) \in \mathbb{R}^3$$
$$\mathbf{z} = (z_1, z_2) \in \mathbb{R}^2 \xrightarrow{\boldsymbol{\varphi}} \boldsymbol{\varphi}(\mathbf{z}) = (z_1^2, \sqrt{2} z_1 z_2, z_2^2) \in \mathbb{R}^3$$

- Evaluate kernel function with explicit feature map

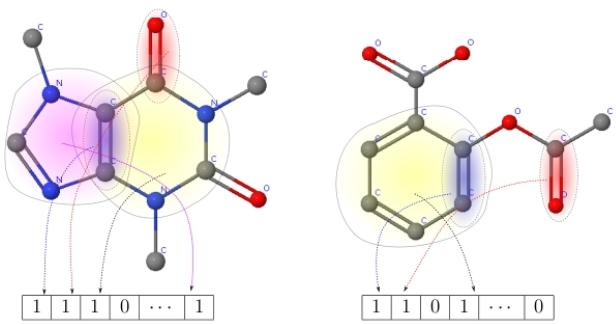
$$\begin{aligned} \langle \boldsymbol{\varphi}(\mathbf{x}), \boldsymbol{\varphi}(\mathbf{z}) \rangle &= \langle (x_1^2, \sqrt{2} x_1 x_2, x_2^2), (z_1^2, \sqrt{2} z_1 z_2, z_2^2) \rangle \\ &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 x_2 z_1 z_2 \end{aligned}$$

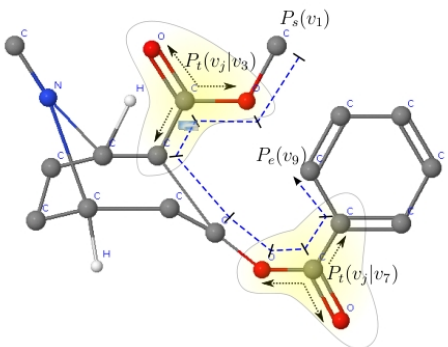
# Popular kernels

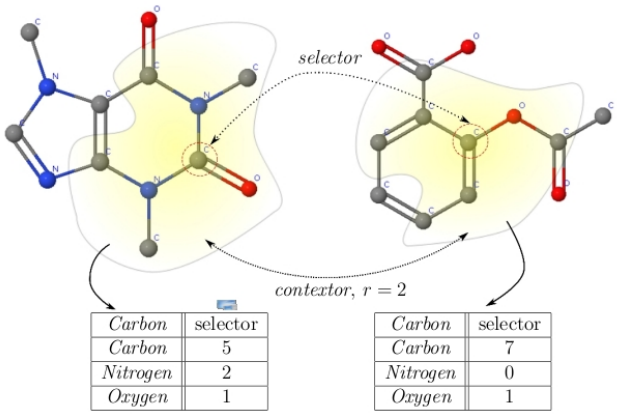
- Linear kernel, $k_{linear}(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle, \mathbf{x}_i, \mathbf{x}_j \in \boldsymbol{\mathcal{X}}$.
- Polynomial kernel, $k_{poly}(\mathbf{x}_i, \mathbf{x}_j) = (\langle \mathbf{x}_i, \mathbf{x}_j \rangle + b)^d, \mathbf{x}_i, \mathbf{x}_j \in \boldsymbol{\mathcal{X}}$.
- Gaussian kernel, $k_{gaussian}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\delta^2}\right), \mathbf{x}_i, \mathbf{x}_j \in \boldsymbol{\mathcal{X}}$.
- Graph kernel, measures the similarity between two graphs.
- String kernel, measure the similarity between strings.

# Graph kernels

- Figureprint kernel: map molecule into a binary bit vector of predefined fingerprints
- Walk kernel: counting the number of same walks on two molecules.
- Decomposition kernel: compare similar regions on two molecules.



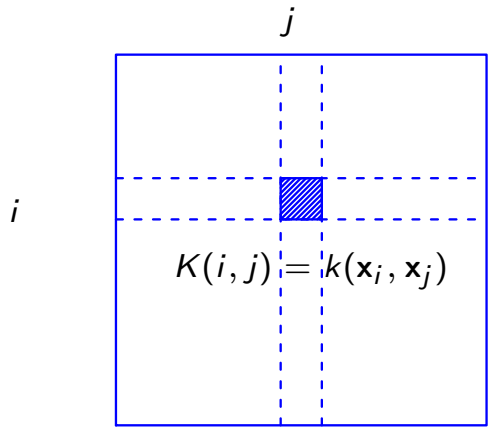




| *Carbon* | selector |
|---|---|
| *Carbon* | 5 |
| *Nitrogen* | 2 |
| *Oxygen* | 1 |

| *Carbon* | selector |
|---|---|
| *Carbon* | 7 |
| *Nitrogen* | 0 |
| *Oxygen* | 1 |

# Kernel matrix

- Given a dataset $S = \{\mathbf{x}_1, \cdots, \mathbf{x}_m\}$ of arbitrary choice of $m$ examples sampled from $\mathcal{X}$ and a kernel function $k$ on $S$, an $m \times m$ matrix $K = (k(\mathbf{x}_i, \mathbf{x}_j))_{i,j}$ is called the kernel matrix (gram matrix) of kernel $k$ with respect to $S$.




- Kernel matrix can be seen as a matrix of pairwise similarities.

# Kernel centering and normalization

- Kernel matrix is often centered

$$K(i,j) \stackrel{\text{def}}{=} K(i,j) - \frac{1}{m}\sum_{i=1}^{m} K(i,j) - \frac{1}{m}\sum_{j=1}^{m} K(i,j) + \frac{1}{m^2}\sum_{i,j=1}^{m} K(i,j)$$

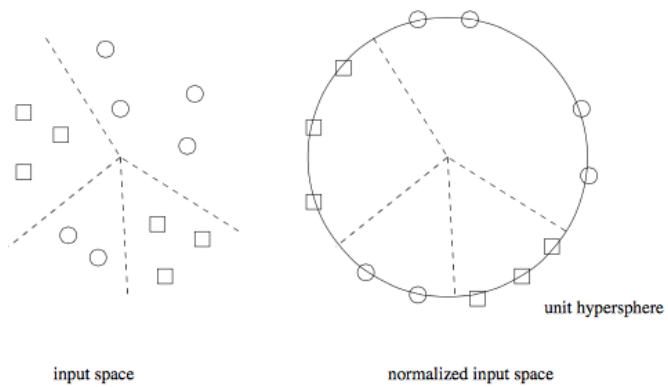
$$= \langle \varphi(\mathbf{x}_i) - \underset{\mathbf{x}_i}{E}[\varphi], \varphi(\mathbf{x}_j) - \underset{\mathbf{x}_j}{E}[\varphi] \rangle,$$

  which corresponds to center feature space.

- Kernel matrix is often normalized

$$K(i,j) \stackrel{\text{def}}{=} \frac{K(i,j)}{\sqrt{K(i,i)K(j,j)}} = \langle \frac{\varphi(\mathbf{x}_i)}{||\varphi(\mathbf{x}_i)||}, \frac{\varphi(\mathbf{x}_j)}{||\varphi(\mathbf{x}_j)||} \rangle,$$

  which corresponds to map data points into a unit hypersphere.



# SVMs Softwares

- LibSVM, in C++, by Chang and Lin (2011)
- SVMLight, in C, by Joachims (1998)
- Torch (http://bengio.abracadoudou.com/SVMTorch.html)
- Spider (http://people.kyb.tuebingen.mpg.de/spider/)
- Weka (http://www.cs.waikato.ac.nz/ml/weka/index.html)

# Two applications

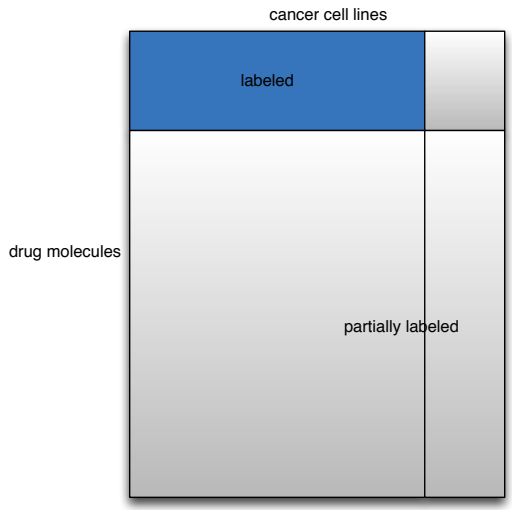
- Predict anti-cancer potentials of drug molecules.
- Predict the classification hierarchies of molecules.

# Predicting anti-cancer potentials

- Data is from NCI-60 screening project, which is a part of developmental therapeutical program (DTP-NCI) from national cancer institute.
- Goal of DTP is to identify and evaluate potential drug molecules that kills cancer cells.
- DTP includes *in vitro* (dead) and *in vivo* (alive) compound screen for pure compounds.
- NCI-60 is a part of *in vitro* screen, fully operation in April 1990.
- NCI-60 screens up to 3000 compounds per year, in 60 different human tumor cell lines, representing leukemia, melanoma and cancers of the lung, colon, brain, ovary, breast, prostate, and kidney (overview).
- Monitor e.g., the growth inhibition of 50% (GI50) which is the drug concentration resulting in a 50% reduction in the net protein increase in control cell during the drug incubation.
- Compute activity score and activity outcome of the drug molecules, basically compounds with LogGI50 less than $-6$ were considered as active. Activity score was based on increasing values of -LogGI50.

# Activity data

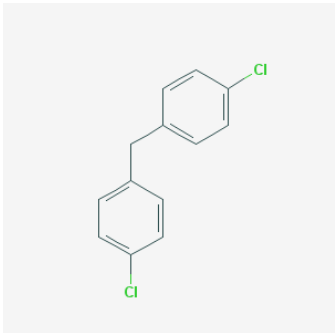
- Activity data is extracted from PubChem, a chemical compound database.
- An example of the data in a single cell line (link).
- Collect data from different cell lines:
  - 60 human tumor cell lines, 13 other cell lines.
  - 43889 molecules, 43340 with structural information available.
  - About 4000 molecules with activities in all 60 cell lines.
- Organize molecular activity data into a matrix




- For now, we focus on the labeled part of the activity matrix.

# Structural data of molecules

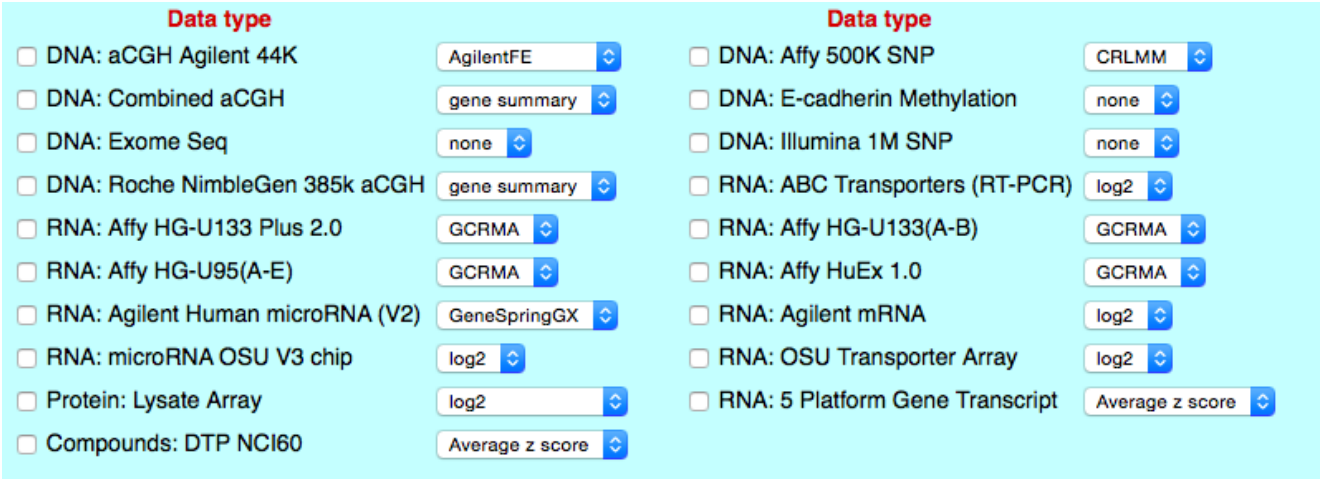
- Each molecule is represented by a small graph, with vertices correspond to chemical atoms and edges correspond to covalent bonds.
- Molecular graph is coded by a SDF file.




```
25 26  0     0  0  0  0  0  0999 V2000
   -5.0312    0.0021    1.5782 Cl  0  0  0  0  0  0  0  0  0  0  0  0
    5.0284    0.0022    1.5783 Cl  0  0  0  0  0  0  0  0  0  0  0  0
   -0.0014   -0.0023   -1.6972 C   0  0  0  0  0  0  0  0  0  0  0  0
   -1.2517   -0.0012   -0.8840 C   0  0  0  0  0  0  0  0  0  0  0  0
    1.2691   -0.0013   -0.8710 C   0  0  0  0  0  0  0  0  0  0  0  0
   -1.8363    1.2072   -0.5054 C   0  0  0  0  0  0  0  0  0  0  0  0
    1.8337   -1.2086   -0.5022 C   0  0  0  0  0  0  0  0  0  0  0  0
   -1.8358   -1.2087   -0.5014 C   0  0  0  0  0  0  0  0  0  0  0  0
    1.8332    1.2074   -0.5046 C   0  0  0  0  0  0  0  0  0  0  0  0
   -3.0051    1.2083    0.2560 C   0  0  0  0  0  0  0  0  0  0  0  0
    3.0024   -1.2076    0.2593 C   0  0  0  0  0  0  0  0  0  0  0  0
   -3.0047   -1.2077    0.2599 C   0  0  0  0  0  0  0  0  0  0  0  0
    3.0021    1.2085    0.2567 C   0  0  0  0  0  0  0  0  0  0  0  0
   -3.5894    0.0009    0.6386 C   0  0  0  0  0  0  0  0  0  0  0  0
    3.5868    0.0009    0.6387 C   0  0  0  0  0  0  0  0  0  0  0  0
    0.0043   -0.8680   -2.3743 H   0  0  0  0  0  0  0  0  0  0  0  0
    0.0043    0.8618   -2.3765 H   0  0  0  0  0  0  0  0  0  0  0  0
   -1.3924    2.1554   -0.7970 H   0  0  0  0  0  0  0  0  0  0  0  0
    1.3867   -2.1546   -0.7930 H   0  0  0  0  0  0  0  0  0  0  0  0
   -1.3913   -2.1576   -0.7898 H   0  0  0  0  0  0  0  0  0  0  0  0
    1.3858    2.1526   -0.7974 H   0  0  0  0  0  0  0  0  0  0  0  0
   -3.4495    2.1576    0.5438 H   0  0  0  0  0  0  0  0  0  0  0  0
    3.4468   -2.1562    0.5497 H   0  0  0  0  0  0  0  0  0  0  0  0
   -3.4488   -2.1563    0.5509 H   0  0  0  0  0  0  0  0  0  0  0  0
    3.4462    2.1578    0.5452 H   0  0  0  0  0  0  0  0  0  0  0  0
 1 14  1  0  0  0  0
 2 15  1  0  0  0  0
 3  4  1  0  0  0  0
 3  5  1  0  0  0  0
 3 16  1  0  0  0  0
 3 17  1  0  0  0  0
 4  6  2  0  0  0  0
 4  8  1  0  0  0  0
 5  7  2  0  0  0  0
 5  9  1  0  0  0  0
 6 10  1  0  0  0  0
 6 18  1  0  0  0  0
 7 11  1  0  0  0  0
 7 19  1  0  0  0  0
 8 12  2  0  0  0  0
 8 20  1  0  0  0  0
 9 13  2  0  0  0  0
 9 21  1  0  0  0  0
10 14  2  0  0  0  0
10 22  1  0  0  0  0
11 15  2  0  0  0  0
11 23  1  0  0  0  0
12 14  1  0  0  0  0
12 24  1  0  0  0  0
```

# Auxiliary data

- There are auxiliary data to describe the similarities between cancer cell lines.
  - Data comes from CellMiner project.
  - Microarray experiments on 60 cell lines (DNA, RNA, Protein).



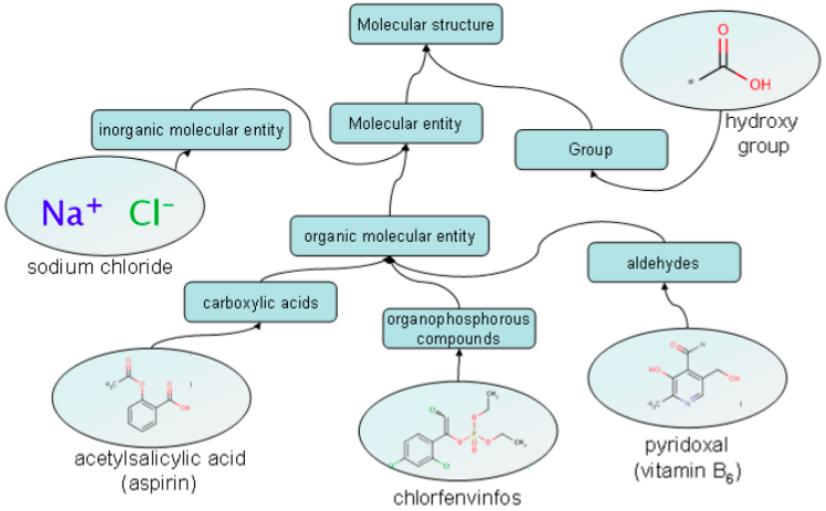
| Data type | | Data type | |
|---|---|---|---|
| DNA: aCGH Agilent 44K | AgilentFE | DNA: Affy 500K SNP | CRLMM |
| DNA: Combined aCGH | gene summary | DNA: E-cadherin Methylation | none |
| DNA: Exome Seq | none | DNA: Illumina 1M SNP | none |
| DNA: Roche NimbleGen 385k aCGH | gene summary | RNA: ABC Transporters (RT-PCR) | log2 |
| RNA: Affy HG-U133 Plus 2.0 | GCRMA | RNA: Affy HG-U133(A-B) | GCRMA |
| RNA: Affy HG-U95(A-E) | GCRMA | RNA: Affy HuEx 1.0 | GCRMA |
| RNA: Agilent Human microRNA (V2) | GeneSpringGX | RNA: Agilent mRNA | log2 |
| RNA: microRNA OSU V3 chip | log2 | RNA: OSU Transporter Array | log2 |
| Protein: Lysate Array | log2 | RNA: 5 Platform Gene Transcript | Average z score |
| Compounds: DTP NCI60 | Average z score | | |

  - As building block of some learning algorithms.
- Task of predicting anti-cancer potentials of molecules
  - Given a set of molecules $\{\mathbf{x}_i\}_{i=1}^m$ with activities in 60 cancer cell line $\{(y_{i,1}, \cdots, y_{i,60})\}_{i=1}^m$, similarities between cell lines.
  - Build a model which predicts the activities of an unknown molecule.

$$\mathbf{x} \xrightarrow{f} (y_1, \cdots, y_{60})$$

# Hierarchical molecular classification

- There is hierarchical classification system for molecules.




- Complex classification system in PubChem (link).
- Task of molecular classification
  - Given a set of molecules $\{\mathbf{x}_i\}_{i=1}^m$ with known classifications $\{(y_{1,1}, \cdots, y_{i,k})\}_{i=1}^m$, and the structure of the classification system.
  - Build a model which predicts the classification of an unknown molecule.

$$\mathbf{x} \xrightarrow{f} (y_1, \cdots, y_k)$$

# Apply single label classification

- For both predicting anti-cancer potential and hierarchical molecular classification, we want to predict multiple outputs given an input.

$$\mathbf{x} \xrightarrow{f} (y_1, \cdots, y_k)$$

- We can learn a collection of single label learners, one for each output

$$\mathbf{x} \xrightarrow{f_1} (y_1, \cdots, \cdots)$$

$$\cdots$$

$$\mathbf{x} \xrightarrow{f_j} (\cdots, y_j, \cdots)$$

$$\mathbf{x} \xrightarrow{f_k} (\cdots, \cdots, y_k)$$

- Different output variables are interdependent.
- The dependency structure is given
  - Explicitly, as the hierarchical structure.
  - Implicitly, by the auxiliary data.
- Can we do better than single label classification?

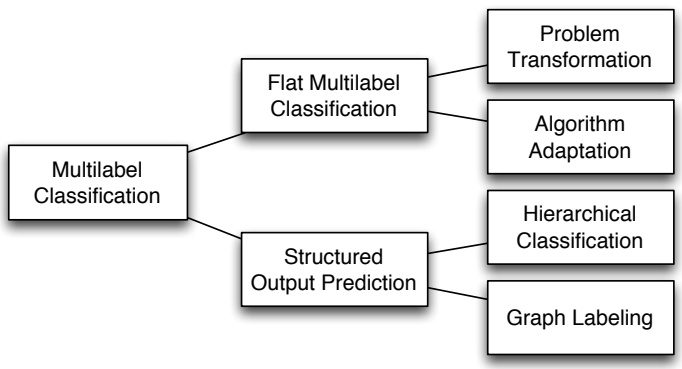
# Multilabel classification

- Multilabel classification is an important research field in machine learning.
  - For example, a document can be classified as "science", "genomics", and "drug discovery".
  - Each input variable is simultaneously associated with multiple output variables.
  - The goal is to reliably predict the values of multiple output variables given an input variable.
- The central problems in multilabel classification is
  - Exponential search space, $\mathbf{x} \to y \in \{-1, +1\}$ vs. $\mathbf{x} \to \mathbf{y} \in \{-1, +1\}^k$.
  - Output variables are interdependent.

# Multilabel classification

## Approaches

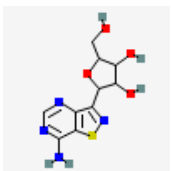
- Flat multilabel classification
  - Consider multiple output variables as a flat vector.
  - Does not take into account label correlations.
  - For example, ML-KNN, ADABOOST, MTL.
- Structured output learning
  - Assume there is structure describing label correlations.
  - The structure can be observed or hidden.
  - Hierarchical structure, SSVM
  - General graph structure, $M^3N$, CRF, MMCRF, RTA



# Max-margin conditional random field

## Preliminaries

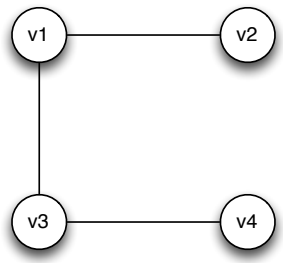
- Training examples come in pairs $(\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathcal{Y}$ sampled from some unknown joint distribution $P(\mathbf{x}, \mathbf{y})$.
- $\mathbf{x} \in \mathcal{X}$ is an instant from input domain $\mathcal{X}$ (e.g., a document, a molecule).
- $\mathbf{y} = (y_1, \cdots, y_k) \in \mathcal{Y}$ is a vector of labels from output domain (e.g., categories of a documents, potentials of a molecules, annotations of a picture).
- The output domain $\mathcal{Y} = \mathcal{Y}_1 \times \cdots \times \mathcal{Y}_k$, $\mathcal{Y}_j = \{-1, +1\}$ and $\mathcal{Y} = \{-1, +1\}^k$.
- $\mathbf{y}$ is called a multilabel, $y_i$ is called a microlabel of the multilabel.
- We are given a set of training examples $S = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^m$.
- A pair $(\mathbf{x}_i, \mathbf{y})$ is called pseudo example, $\mathbf{y} \in \mathcal{Y}$.
- The goal is to obtain a function $f : \mathcal{X} \to \mathcal{Y}$.



$\rightarrow \qquad \mathbf{y} = (y_1, y_2, y_3, y_4) = (+1, -1, +1, +1)$

# Output graph and feature maps

- Observe an output graph $G = (E, V)$, a vertex $v_i \in V$ corresponds to a microlabel $y_i$, an edge $(v_i, v_j) \in E$ corresponds to the pairwise correlation of the microlabel $y_i$ and $y_j$.




- $\varphi(\mathbf{x})$ is the input feature map, $\varphi(\mathbf{x}) \in \mathbb{R}^d$.
- $\psi(\mathbf{y})$ is the output feature map that maps the multilabel $\mathbf{y}$ into a set of edges and labels, $\psi(\mathbf{y}) \in \mathbb{R}^{4|E|}$.

$$\mathbf{y} = (y_1, y_2, y_3, y_4) = (+1, -1, +1, +1)$$

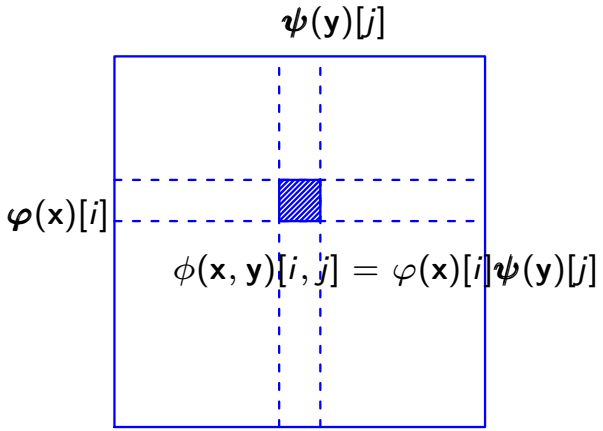
$$\psi(\mathbf{y}) = (\underbrace{\underbrace{0}_{--}, \underbrace{0}_{-+}, \underbrace{1}_{+-}, \underbrace{0}_{++}}_{(v_1,v_3)}, \underbrace{\underbrace{0}_{--}, \underbrace{0}_{-+}, \underbrace{0}_{+-}, \underbrace{1}_{++}}_{(v_1,v_2)}, \underbrace{\underbrace{0}_{--}, \underbrace{0}_{-+}, \underbrace{0}_{+-}, \underbrace{1}_{++}}_{(v_3,v_4)})$$

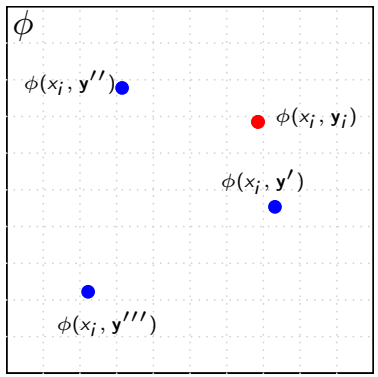
# Compatibility score

- Joint feature map put input and output feature into a joint feature space

$$\boldsymbol{\phi}(\mathbf{x}, \mathbf{y}) = \varphi(\mathbf{x}) \otimes \psi(\mathbf{y}) = (\boldsymbol{\varphi}(\mathbf{x})[i]\,\boldsymbol{\psi}(\mathbf{y})[j])_{i,j},$$

  $\otimes$ is the Kronecker product. $\boldsymbol{\phi}(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{d \times 4|E|}$




- For now we assume we already have optimal $\mathbf{w}$, compatibility score will assign high value to $(\mathbf{x}_i, \mathbf{y}_i)$ compared to $(\mathbf{x}_i, \mathbf{y}), \mathbf{y} \in \mathcal{Y}/\mathbf{y}_i$

$$f(\mathbf{x}_i, \mathbf{y}_i; \mathbf{w}) = \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}_i, \mathbf{y}_i) \rangle.$$

- Prediction is by solving the **argmax** in exponential space $|\mathcal{Y}| = 2^k$

$$\mathbf{y}^* = \operatorname*{\mathbf{argmax}}_{\mathbf{y} \in \mathcal{Y}} f(\mathbf{x}, \mathbf{y}) = \operatorname*{\mathbf{argmax}}_{\mathbf{y} \in \mathcal{Y}} \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}, \mathbf{y}) \rangle.$$

# Obtain w by regularized learning

- $L_2$ norm regularization on $\mathbf{w}$, $\Omega(f) = ||\mathbf{w}||^2$.
- Risk term
  - $f$ will not make mistake if

$$f(\mathbf{x}_i, \mathbf{y}_i) - f(\mathbf{x}_i, \mathbf{y}) \geq 0, \forall \mathbf{y} \in \mathcal{Y}/\mathbf{y}_i, i \in \{1, \cdots, m\}.$$

  - We only need to evaluate the score of best incorrect label

$$f(\mathbf{x}_i, \mathbf{y}_i) - \max_{\mathbf{y} \in \mathcal{Y}/\mathbf{y}_i} f(\mathbf{x}_i, \mathbf{y}) \geq 0, \forall i \in \{1, \cdots, m\}.$$

  - Further push away "bad" multilabels

$$f(\mathbf{x}_i, \mathbf{y}_i) - \max_{\mathbf{y} \in \mathcal{Y}/\mathbf{y}_i} f(\mathbf{x}_i, \mathbf{y}) \geq \ell(\mathbf{y}_i, \mathbf{y}), \forall i \in \{1, \cdots, m\}.$$

- Regularization theory suggests we minimize

$$\mathcal{J}(f) = \underbrace{||\mathbf{w}||^2}_{\text{regularization}} + \underbrace{\sum_{i=1}^{m} \max_{\mathbf{y} \in \mathcal{Y}/\mathbf{y}_i} f(\mathbf{x}_i, \mathbf{y}) + \ell(\mathbf{y}_i, \mathbf{y}) - f(\mathbf{x}_i, \mathbf{y}_i)}_{\text{risk term}}$$

# The optimization problem

- The optimization problem is defined as

$$\begin{aligned} \min_{\mathbf{w},\xi} \quad & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{m}\xi_i \\ \text{s.t.} \quad & f(\mathbf{x}_i,\mathbf{y}_i) - \max_{\mathbf{y}\in\mathcal{Y}/\mathbf{y}_i} f(\mathbf{x}_i,\mathbf{y}) \geq \ell(\mathbf{y}_i,\mathbf{y}) - \xi_i, \\ & \xi_i \geq 0, \forall i \in \{1,\cdots,m\}. \end{aligned}$$

- In order to solve MMCRF we need to solve similar problems both in training and in prediction

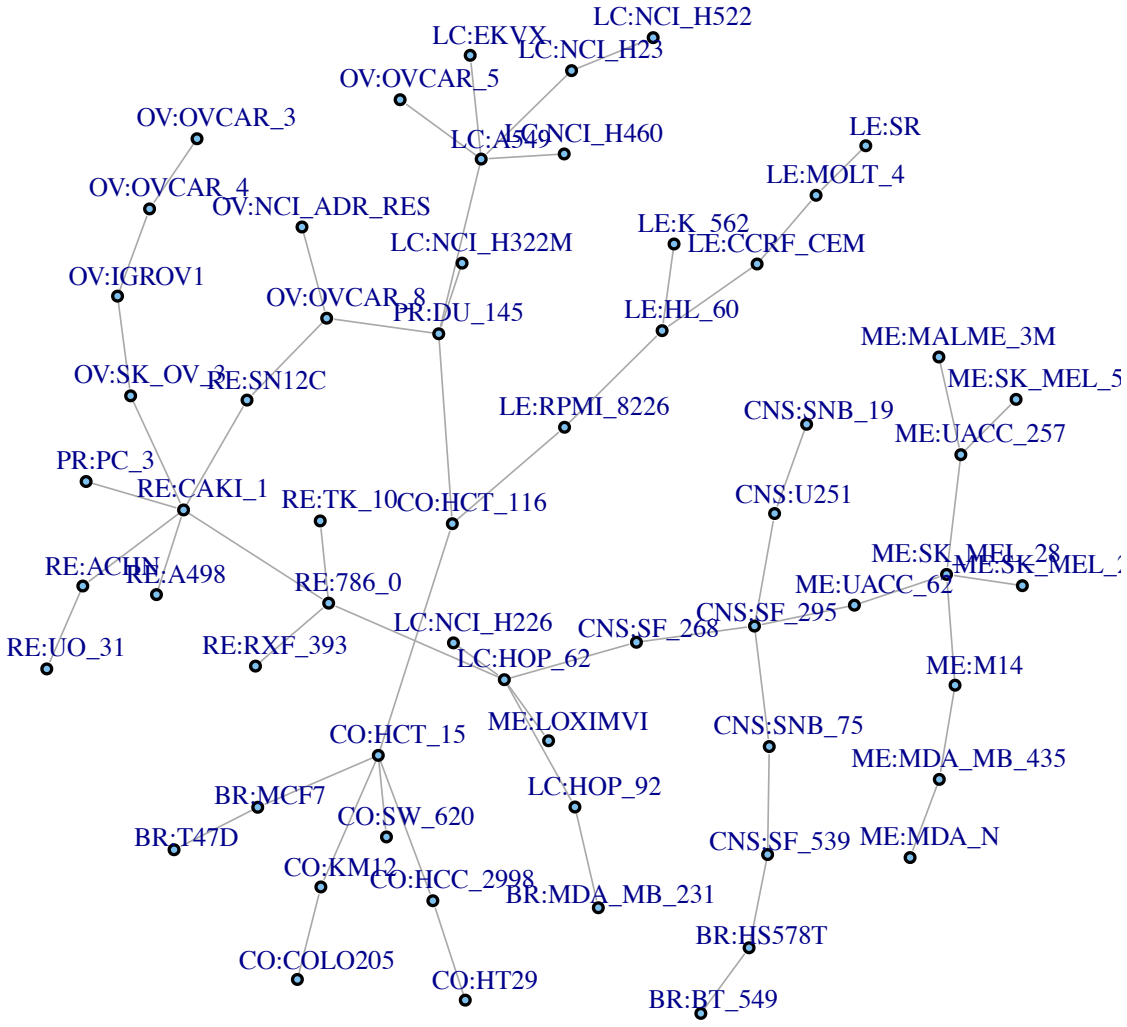
$$\mathbf{y}^* = \operatorname*{argmax}_{\mathbf{y}\in\mathcal{Y}} f(\mathbf{x},\mathbf{y}) = \operatorname*{argmax}_{\mathbf{y}\in\mathcal{Y}} \langle \mathbf{w}, \phi(\mathbf{x},\mathbf{y})\rangle.$$

- It is an instantiation of maximize a-posteriori (MAP) problem defined on the output graph $G$.
  - Simple structure (e.g., grid, tree, sequence): polynomial size algorithms (e.g., message-passing) to get exact solutions.
  - Complex structure: $\mathcal{NP}$-hard problem, approximation algorithms (e.g., loopy belief propagation (Wainwright and Jordan, 2003)).
- The output graph need to be observed *apriori*.

# What if the output graph is not observed?

- In many applications, the output graph is not directly observed.
  - In the task of predicting anti-cancer potentials, the output graph describing the correlations between different cancer cell lines is unknown.
- We now focus on the problem in structured output learning where the output graph is not observed.
  - We can extract the correlation graph from some external data sources if then are available.
  - "Ensemble" methods
    - Generate a set random output graphs $\{G_1, \cdots, G_n\}$
    - Build a collection of structured output learners $\{f_{G_1}(\mathbf{x}, \mathbf{y}; \mathbf{w}_{G_1}), \cdots, f_{G_1}(\mathbf{x}, \mathbf{y}; \mathbf{w}_{G_n})\}$
    - Combine the predictions.
- Can we do better than these?

# Example: cancer cell lines



# Complete graph as output graph

- Assume a complete set of pairwise correlations is sufficient to describe the dependencies between multiple labels.
- We use a complete graph $G$ as the output graph (have enough power of expression).

$$|V| = |\mathbf{y}| = k, |E| = \frac{1}{2}k(k-1)$$

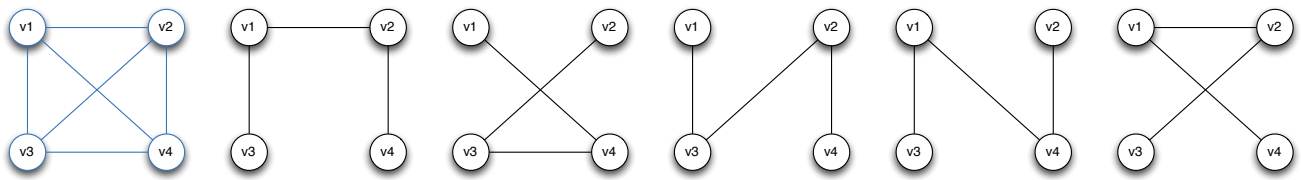
- Compatibility score defined on the complete graph

$$f_G(\mathbf{x}, \mathbf{y}; \mathbf{w}_G) = \langle \mathbf{w}_G, \boldsymbol{\phi}_G(\mathbf{x}, \mathbf{y}) \rangle.$$

  - Parameter space, $\Theta(k^2)$.
  - Inference problem is $\mathcal{NP}$-hard, $\mathbf{y}^* = \underset{\mathbf{y} \in \boldsymbol{\mathcal{Y}}}{\mathbf{argmax}} \langle \mathbf{w}_G, \boldsymbol{\phi}_G(\mathbf{x}, \mathbf{y}) \rangle$

# Random spanning tree approximation (RTA)

- We want to maintain a small parameter space, and feasible inference.
- When the output graph is a tree $T = (E_T, V), |V| = k, |E| = k - 1$
  - $f_T(\mathbf{x}, \mathbf{y}; \mathbf{w}_T) = \langle \mathbf{w}_T, \phi_T(\mathbf{x}, \mathbf{y}) \rangle$
  - $|\mathbf{w}_T| = d \times 4(k-1), \Theta(k)$
  - Linear time $\Theta(k)$ to compute $\mathbf{y}^* = \underset{\mathbf{y} \in \mathcal{Y}}{\mathbf{argmax}} \langle \mathbf{w}_T, \phi_T(\mathbf{x}, \mathbf{y}) \rangle$
- Sample all spanning tree from the complete graph $G$, which gives $k^{k-2}$ unique random spanning trees $\mathcal{T} = \{T_1, \cdots, T_{k^{(k-2)}}\}$.




- Rewrite the compatibility score of a complete graph $G$ in terms of all spanning trees

$$f_G(\mathbf{x}, \mathbf{y}; \mathbf{w}_G) = \frac{2k^{(k-2)}}{k} \sum_{i=1}^{k^{k-2}} f_{T_i}(\mathbf{x}, \mathbf{y}; \mathbf{w}_{T_i})$$

- This is not inviting, summing over exponential number of spanning trees.

# Regularized learning for RTA

- Numeric values vs. algorithmic properties
- It is proved that properties (e.g., margin, generalization error) achieved by a complete graph can also be preserved by a small set of its random spanning tree, $n$ is bounded by $\Theta(k^2)$.

$$f_G(\mathbf{x}, \mathbf{y}; \mathbf{w}_G) \approx \frac{1}{n} \sum_{t=1}^{n} f_{T_i}(\mathbf{x}, \mathbf{y}; \mathbf{w}_{T_i}) = \frac{1}{n} \sum_{t=1}^{n} \langle \mathbf{w}_{T_i}, \phi_{T_i}(\mathbf{x}, \mathbf{y}) \rangle$$

- The optimization problem is

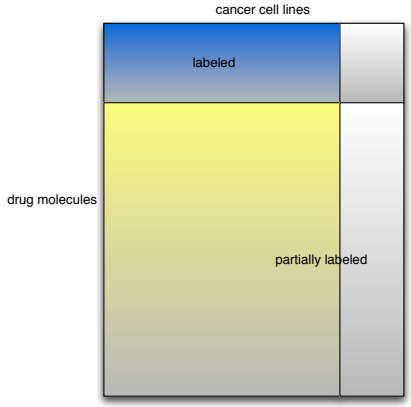
$$\min_{\mathbf{w}, \xi} \quad \underbrace{\frac{1}{2} \sum_{t=1}^{n} ||\mathbf{w}_{T_t}||^2}_{\text{regularization}} + \underbrace{C \sum_{i=1}^{m} \xi_i}_{\textit{risk}}$$

$$\textbf{s.t.} \quad \sum_{t=1}^{n} f_{T_t}(\mathbf{x}_i, \mathbf{y}_i; \mathbf{w}_{T_t}) - \max_{\mathbf{y} \in \mathcal{Y}/\mathbf{y}_i} \sum_{t=1}^{n} f_{T_t}(\mathbf{x}_i, \mathbf{y}; \mathbf{w}_{T_t}) \geq \ell(\mathbf{y}_i, \mathbf{y}) - \xi_i, \xi_i \geq 0. \forall i.$$

- The inference problem can be solved by polynomial time algorithm in $\Theta(n\kappa k)$

# Partially labeled data

- We have huge amount of partially labeled data.




- Can we do better?
- Semi-supervised learning

$$\min_{\mathbf{w}} \underbrace{\sum_{i=1}^{m} \ell(f(\mathbf{x}_i)\mathbf{y}_i)}_{\text{risk}} + \underbrace{\lambda_1 \Omega(f)}_{\text{regularization}} \underbrace{+ \lambda_2 \sum_{i,j=1}^{m+u} W_{i,j} ||f(\mathbf{x}_i) - f(\mathbf{x}_j)||^2}_{\text{consistency regularizer}}$$

  - Consistency regularizer required similar input patterns generate similar output patterns.

# To sum up

- Application fields of classification learning:
  - Predicting anti-cancer potentials of molecules
  - Hierarchical molecular classification
- Start from single label classification
  - Risk minimization, regularization
  - Support vector machines, kernel methods
- Multilabel classification
  - Flat multilabel classification, structured output prediction
  - Output graph as *apriori*: Max-margin conditional random field
  - Unobserved output graph: ensemble methods, random spanning tree approximation
- Max-margin regression

# Bibliography


Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the 5th Annual Workshop on Computational Learning Theory*, pages 144–152. ACM.

Chang, C.-C. and Lin, C.-J. (2011). Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27.

Chen, S. and Rosenfeld, R. (1999). *A Gaussian Prior for Smoothing Maximum Entropy Models*. PhD thesis, Computer Science Department, Carnegie Mellon University. Technical Report CMU-CS-99-108.

Chen, S. and Rosenfeld, R. (2000). A survey of smoothing techniques for me models. *IEEE Transactions on Speech and Audio Processing*, 8(1):37–50.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Cristinini, N. and Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press.

# Bibliography (cont.)


Evgeniou, T., Poggio, T., Pontil, M., and Verri, A. (2002). Regularization and statistical learning theory for data analysis. *Comput. Stat. Data Anal.*, 38(4):421–432.

Evgeniou, T., Pontil, M., and Poggio, T. (1999). A unified framework for regularization networks and support vector machines. Technical report, Cambridge, MA, USA.

Golle, P. (2008). Machine learning attacks against the asirra captcha. In *Proceedings of the 15th ACM Conference on Computer and Communications Security*, CCS '08, pages 535–542, New York, NY, USA. ACM.

Hoerl, A. E. and Kennard, R. W. (2000). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 42(1):80–86.

Joachims, T. (1998). Making large-scale svm learning practical. LS8-Report 24, Universität Dortmund, LS VIII-Report.

Schapire, R. and Singer, Y. (1999). Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336.

Shawe-Taylor, J. and Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press.

# Bibliography (cont.)


Tibshirani, R. (1994). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288.

Tsochantaridis, I., Hofmann, T., Joachims, T., and Altun, Y. (2004). Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the 21th International Conference on Machine Learning (ICML 2004)*, pages 823–830. ACM.

Vapnik, V. (1992). Principles of risk minimization for learning theory. In Moody, J., Hanson, S., and Lippmann, R., editors, *Advances in Neural Information Processing Systems 4*, pages 831–838. Morgan-Kaufmann.

Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley-Interscience.

Wainwright, M. J. and Jordan, M. I. (2003). Graphical models, exponential families, and variational inference. *Foundation and Trends in Machine Learning*, 1(1-2):1–305.

Zhou, D., Bousquet, O., Lal, T. N., Weston, J., and Schölkopf, B. (2004). Learning with local and global consistency. In Thrun, S., Saul, L., and Schölkopf, B., editors, *Advances in Neural Information Processing Systems 16*, pages 321–328. MIT Press.