Week 8, Lecture 16 - Support Vector Machines

Aaron Meyer

Outline

- Administrative Issues
- Support Vector Machines
  - Linear separable
  - Nonlinear separable
  - Soft margin adjustment
- Application
- Implementation

Adapted from slides by Martin Law.

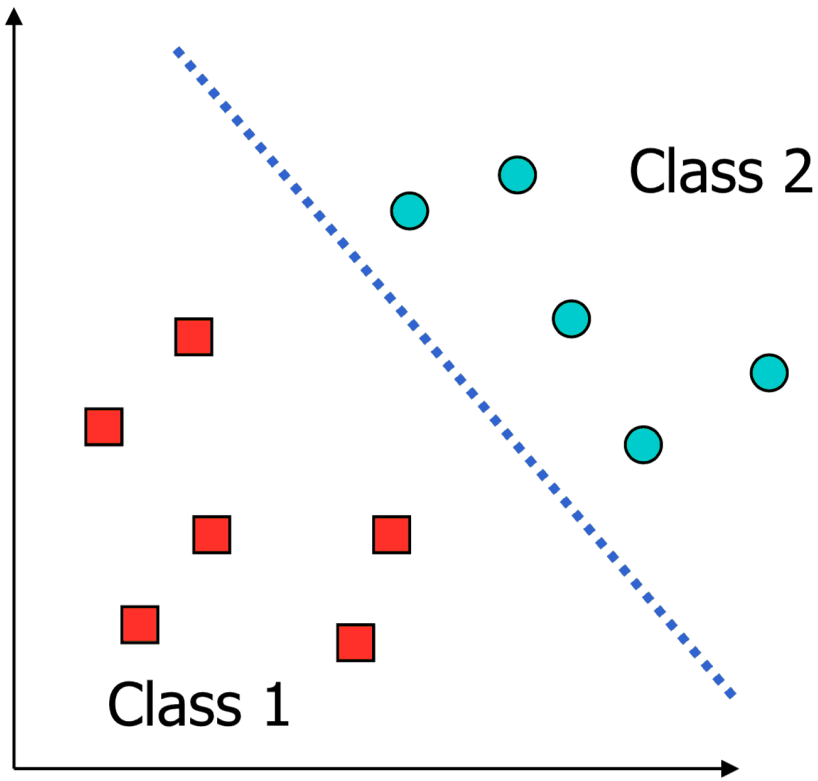
History Of SVM

- SVM is related to statistical learning theory
- SVM was first introduced in 1992
- SVM became popular because of its success in handwritten digit recognition
  - 1.1% test error rate for SVM. This is the same as the error rates of a carefully constructed neural network, LeNet 4.
  - Also used in very first self-driving cars (of 90's, not current)
  - Bested by deep learning methods", one of the key area in machine learning
- Note: the meaning of "kernel" is different from other methods

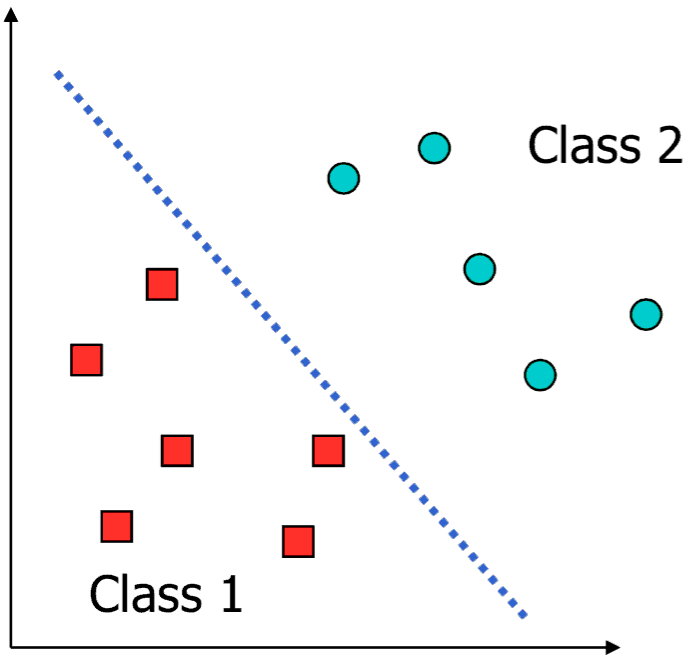
What Is A Good Decision Boundary?

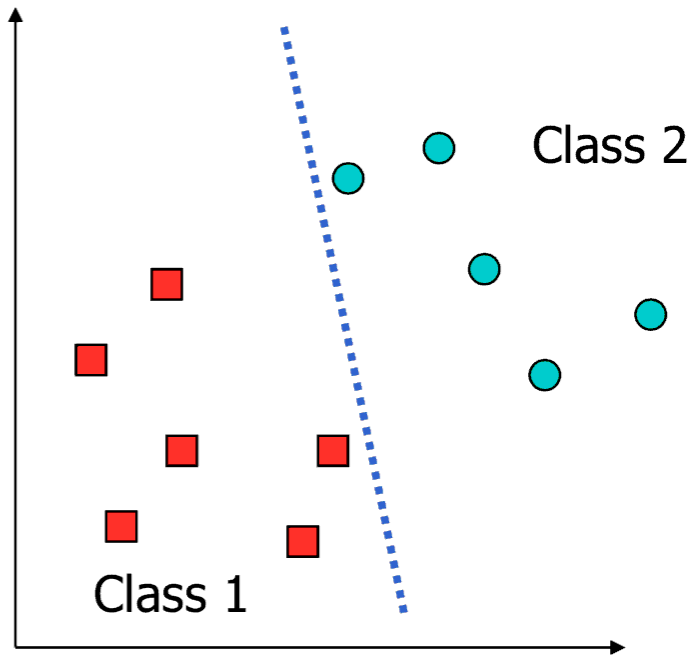
Consider a two-class, linearly separable classification problem

- Many decision boundaries!
- Are all decision boundaries equally good?



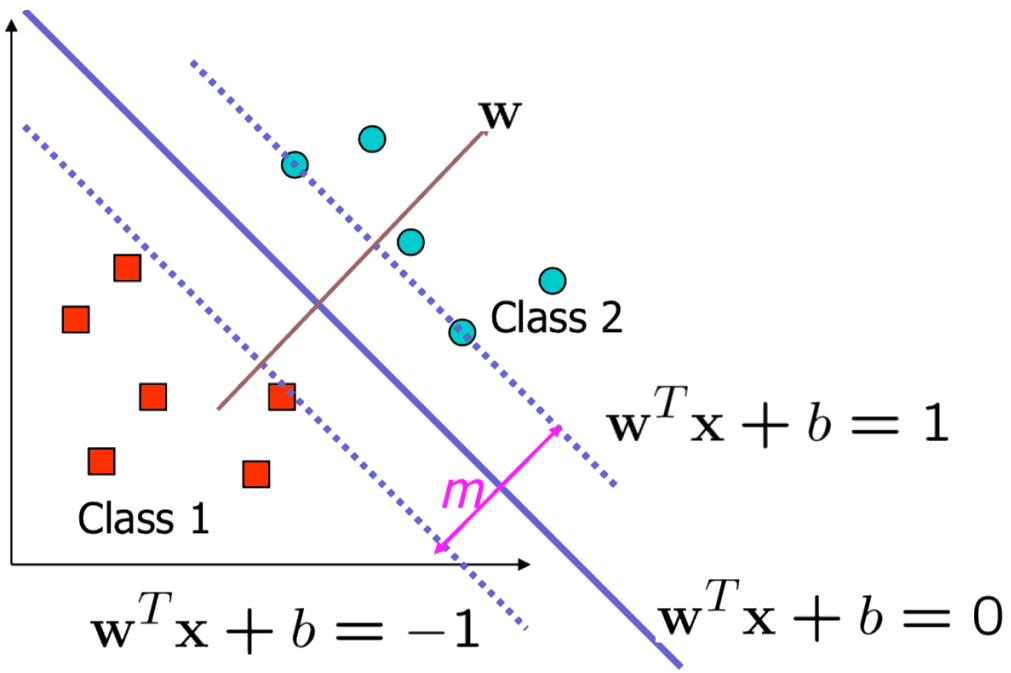
Examples Of Bad Decision Boundaries





Large-Margin Decision Boundary

- The decision boundary should be as far away from the data of both classes as possible
- We should maximize the margin, $m = 2/\|\mathbf{w}\|$
- Distance between the origin and the line $\mathbf{w}^t\mathbf{x} = k$ is $k/\|\mathbf{w}\|$



Finding The Decision Boundary

- Let $\{x_1, \ldots, x_n\}$ be our data set and let $y_i \in \{1, -1\}$ be the class label of x_i
- The decision boundary should classify all points correctly
  - $y_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) \geq 1, \quad \forall i$
- The decision boundary can be found by solving the following constrained optimization problem:
  - Minimize $\frac{1}{2} \|\mathbf{w}\|^2$
  - subject to $y_i \left(\mathbf{w}^T \mathbf{x}_i + b\right) \geq 1$
- This is a constrained optimization problem
- Quick methods to find the global optimum by quadratic programming

The Quadratic Programming Problem

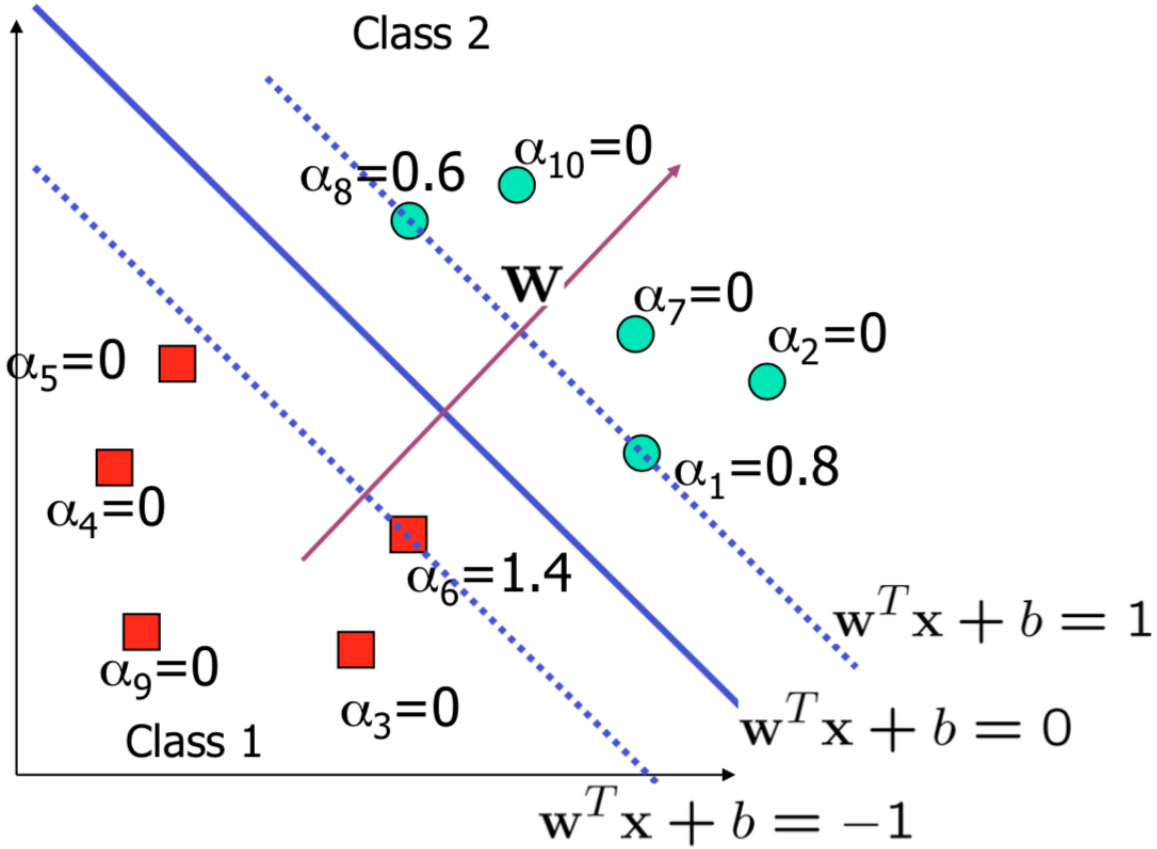
Many approaches have been proposed

- Most are "interior-point" methods
  - Start with an initial solution that can violate the constraints
  - Improve this solution by optimizing the objective function and/or reducing the amount of constraint violation
- For SVM, sequential minimal optimization (SMO) seems to be the most popular
  - A QP with two variables is trivial to solve
  - Each iteration of SMO picks a pair of (α_i, α_j) and solve the QP with these two variables; repeat until convergence
- In practice, we can just regard the QP solver as a "black-box" without bothering how it works

Characteristics Of The Solution

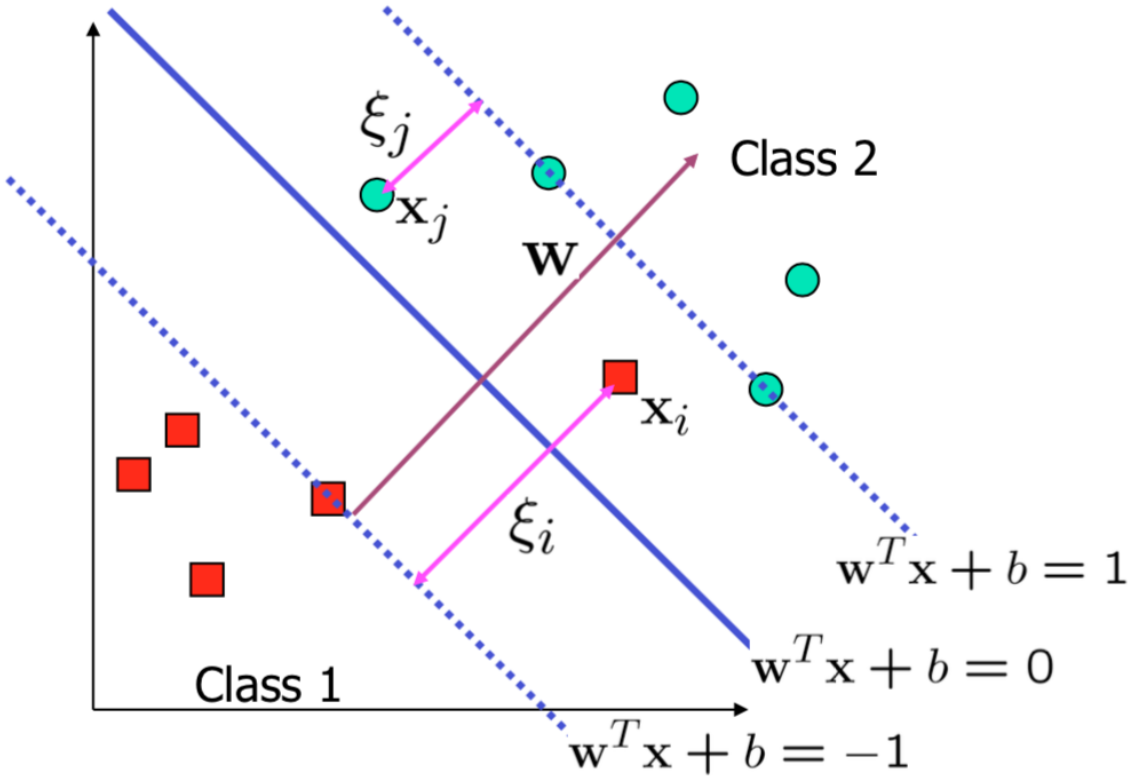
- Many of the α_i are zero
  - **w** is a linear combination of a small number of data points
  - This "sparse" representation can be viewed as data compression as in the construction of knn classifier
- x_i with non-zero α_i are called support vectors (SV)
  - The decision boundary is determined only by the SV
  - Let $t_j (j = 1, \ldots, s)$ be the indices of the s support vectors
  - We can write $\mathbf{w} = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} \mathbf{x}_{t_j}$
- For testing with a new data **z**
  - Compute $\mathbf{w}^T \mathbf{z} + b = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} \left(\mathbf{x}_{t_j}^T \mathbf{z} \right) + b$
  - Classify **z** as class 1 if the sum is positive, and class 2 otherwise
  - Note: **w** need not be formed explicitly

A Geometrical Interpretation



Non-Linearly Separable Problems

- We allow "error" ξ_i in classification; it is based on the output of the discriminant function $\mathbf{w}^T\mathbf{x} + b$
- ξi approximates the number of misclassified samples



Soft Margin Hyperplane

- If we minimize $\sum_i \xi_i$, ξ_i can be computed by
  - $(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i \quad y_i = 1$
  - $(\mathbf{w}^T \mathbf{x}_i + b) \leq -1 + \xi_i \quad y_i = 1$
  - $\xi_i \geq 0$
  - ξ_i are "slack variables" in optimization
  - Note that $\xi_i = 0$ if there is no error for $\mathbf{x}_i$
  - ξ_i is an upper bound of the number of errors
- We want to minimize:
  - $\frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i$
  - C: tradeoff parameter between error and margin
- The optimization problem becomes:
  - Minimize $\frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i$
  - subject to $y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$

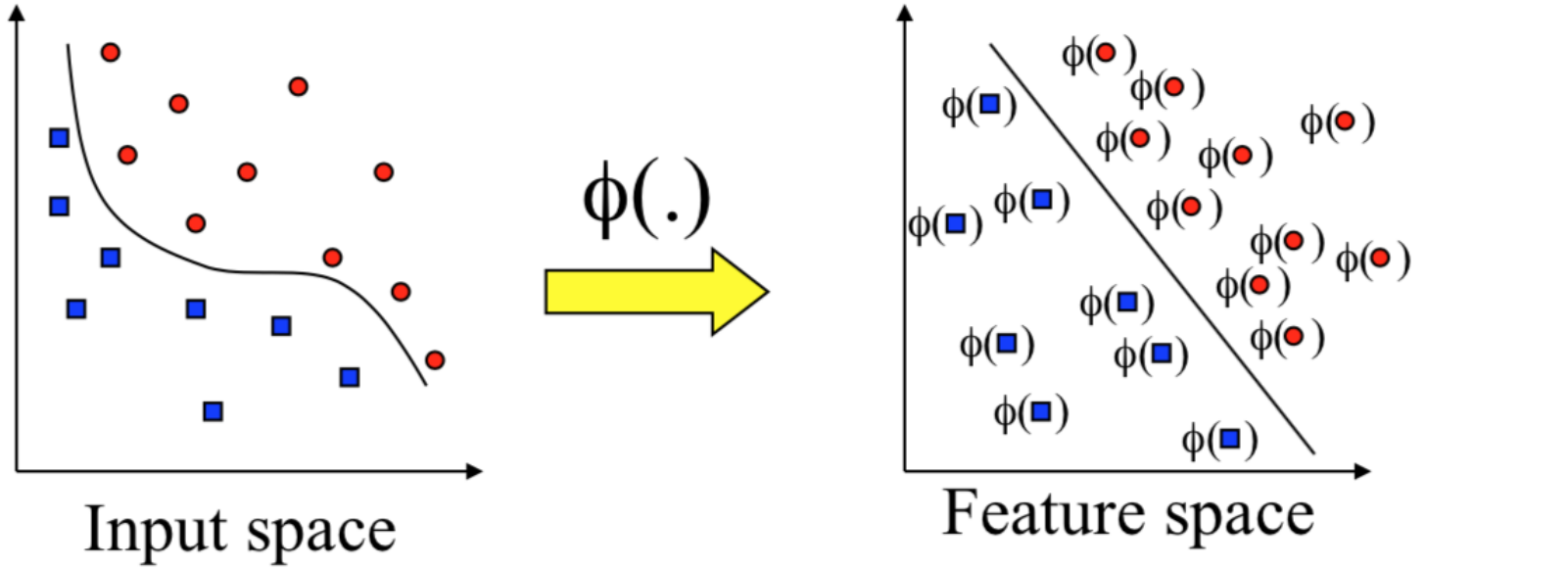
The Optimization Problem

- The dual of this new constrained optimization problem is:
  - max. $W(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$
  - subject to $C \geq \alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0$
- $\mathbf{w}$ is recovered as $\mathbf{w} = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} \mathbf{x}_{t_j}$
- This is very similar to the optimization problem in the linear separable case, except that there is an upper bound C on α_i now
- Once again, a QP solver can be used to find α_i

Extension To Non-linear Decision Boundary

- So far, we have only considered large-margin classifier with a linear decision boundary
- How to generalize it to become nonlinear?
- Key idea: transform $\mathbf{x}_i$ to a higher dimensional space to "make life easier"
  - Input space: the space the point $\mathbf{x}_i$ are located
  - Feature space: the space of $\varphi(\mathbf{x}_i)$ after transformation
- Why transform?
  - Linear operation in the feature space is equivalent to non-linear operation in input space
  - Classification can become easier with a proper transformation.

Transforming The Data



- Computation in the feature space can be costly because it is high dimensional
  - The feature space is typically infinite-dimensional!
- The kernel trick comes to rescue

The Kernel Trick

- Recall the SVM optimization problem
- The data points only appear as the inner product
  - max. $W(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$
  - subject to $C \geq \alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0$
- As long as we can calculate the inner product in the feature space, we do not need the mapping explicitly
- Many common geometric operations (angles, distances) can be expressed by inner products
- Define the kernel function K by:

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

Kernel Functions

- In practical use of SVM, the user specifies the kernel function; the transformation $\varphi(.)$ is not explicitly stated
- Given a kernel function $K(x_i, x_j)$, the transformation $\varphi(.)$ is given by its eigenfunctions (a concept in functional analysis)
  - Eigenfunctions can be difficult to construct explicitly
  - This is why people only specify the kernel function without worrying about the exact transformation
- Another view: kernel function, being an inner product, is really a similarity measure between the objects

Examples of Kernel Functions

- Polynomial kernel with degree d
  - $K(\mathbf{x}, \mathbf{y}) = \left(\mathbf{x}^T \mathbf{y} + 1\right)^d$
- Radial basis function kernel with width σ
  - $K(\mathbf{x}, \mathbf{y}) = \exp(-\left\|\mathbf{x} - \mathbf{y}\right\|^2 / (2\sigma^2))$
  - Closely related to radial basis function neural networks
  - The feature space is infinite-dimensional
- Sigmoid with parameter κ and θ
  - $K(\mathbf{x}, \mathbf{y}) = \tanh(\kappa \mathbf{x}^T \mathbf{y} + \theta)$

Modification Due to Kernel Function

- Change all inner products to kernel functions
- For training:
  - Original: max. $W(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$
  - With kernel function: max. $W(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$
  - Both: subject to $C \geq \alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0$

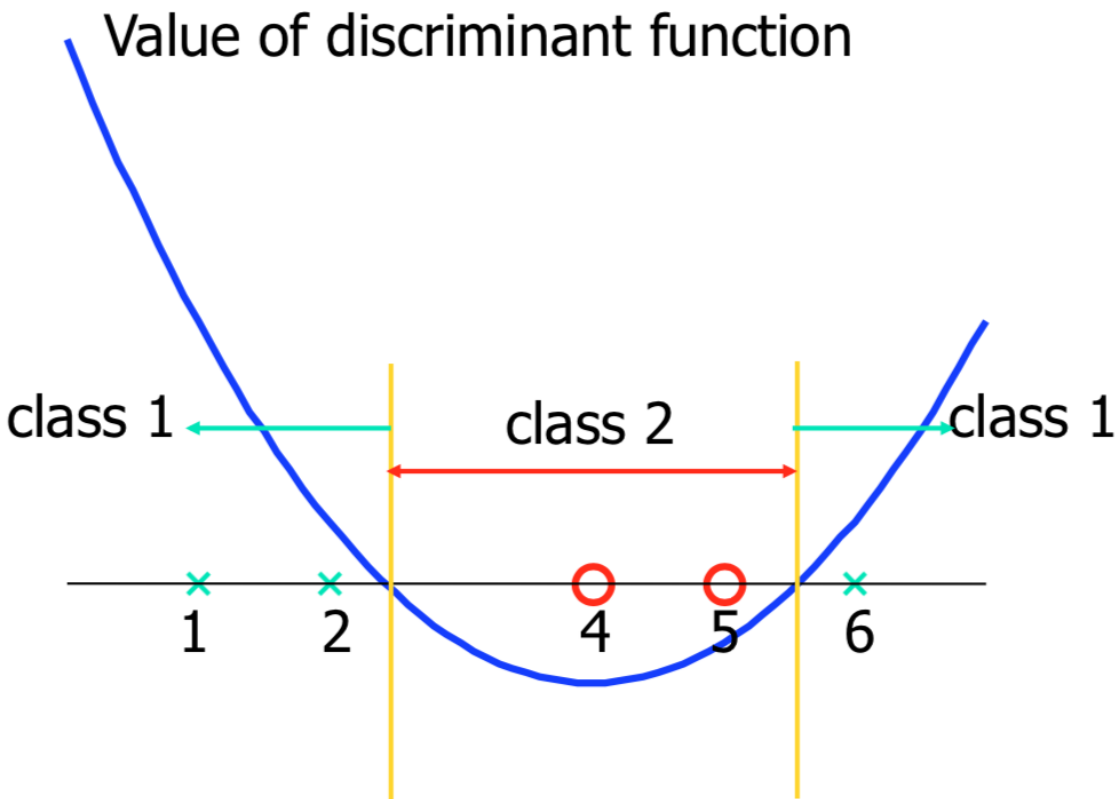
Modification Due to Kernel Function

- For testing, the new data $\mathbf{z}$ is classified as class 1 if $f \geq 0$, and class 2 if $f < 0$
- Original:
  - $\mathbf{w} = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} \mathbf{x}_{t_j}$
  - $f = \mathbf{w}^T \mathbf{z} + b = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} \left(\mathbf{x}_{t_j}^T \mathbf{z} \right) + b$
- With kernel function:
  - $\mathbf{w} = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} \phi \left(\mathbf{x}_{t_j} \right)$
  - $f = \langle \mathbf{w}, \phi (\mathbf{z}) \rangle + b = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} K \left(\mathbf{x}_{t_j}^T, \mathbf{z} \right) + b$

More on Kernel Functions

- Since the training of SVM only requires the value of $K(\mathbf{x}_i, \mathbf{x}_j)$, there is no restriction of the form of $\mathbf{x}_i$ and $\mathbf{x}_j$
  - $\mathbf{x}_i$ can be a sequence or a tree, instead of a feature vector
- $K(\mathbf{x}_i, \mathbf{x}_j)$ is just a similarity measure comparing $\mathbf{x}_i$ and $\mathbf{x}_j$
- For a test object $\mathbf{z}$, the discrimination function essentially is a weighted sum of the similarity between $\mathbf{z}$ and a pre-selected set of objects (the support vectors):
  - $f(\mathbf{z}) = \sum_{\mathbf{x}_i \in S} \alpha_i y_i K(\mathbf{z}, \mathbf{x}_i) + b$
  - S: the set of support vectors

Example of Non-Linear Transformation



Justification of SVM

- Large margin classifier
- Ridge regression: the term $\frac{1}{2}\|w\|^2$ "shrinks" the parameters towards zero to avoid overfitting
- The term the term $\frac{1}{2}\|w\|^2$ can also be viewed as imposing a weight-decay prior on the weight vector

Choosing the Kernel Function

- Probably the most tricky part of using SVM
- The kernel function is important because it creates the kernel matrix, which summarizes all the data
- In practice, a low degree polynomial kernel or RBF kernel with a reasonable width is a good initial try

Other Aspects of SVM

- How to use SVM for multi-class classification?
  - One can change the QP formulation to become multi-class
  - More often, multiple binary classifiers are combined
  - One can train multiple one-versus-all classifiers, or combine multiple pairwise classifiers "intelligently"
- How to interpret the SVM discriminant function value as probability?
  - By performing logistic regression on the SVM output of a set of data (validation set) that is not used for training

Summary: Steps For Classification

- Prepare the pattern matrix
- Select the kernel function to use
- Select the parameter of the kernel function and the value of C
  - You can use the values suggested by the SVM software, or you can set apart a validation set to determine the values of the parameter
- Execute the training algorithm and obtain the α_i
- Unseen data can be classified using the α_i and the support vectors

Strengths And Weaknesses Of SVM

- Strengths
  - Training is relatively easy
    - No local optimal, unlike in neural networks
  - It scales relatively well to high dimensional data
  - Tradeoff between classifier complexity and error can be controlled explicitly
  - Non-traditional data like strings and trees can be used as input to SVM, instead of feature vectors
- Weaknesses
  - Need to choose a "good" kernel function.

Other Types Of Kernel Methods

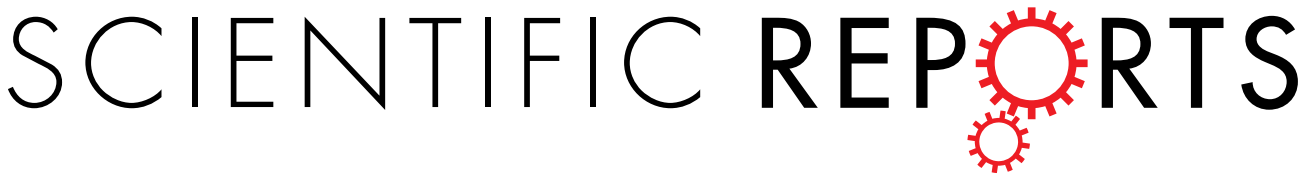
- A lesson learnt in SVM: a linear algorithm in the feature space is equivalent to a non-linear algorithm in the input space
- Standard linear algorithms can be generalized to its non-linear version by going to the feature space
  - Kernel principal component analysis, kernel independent component analysis, kernel canonical correlation analysis, kernel k-means, 1-class SVM are some examples

Multi-Class Classification

- SVM is basically a two-class classifier
- One can change the QP formulation to allow multi-class classification
- More commonly, the data set is divided into two parts "intelligently" in different ways and a separate SVM is trained for each way of division
- Multi-class classification is done by combining the output of all the SVM classifiers
  - Majority rule
  - Error correcting code
  - Directed acyclic graph

Application - Circulating Tumor Cells

www.nature.com/scientificreports



OPEN

Multiparameter mechanical and morphometric screening of cells

Mahdokht Masaeli[1,2,3], Dewal Gupta[1], Sean O'Byrne[1,2], Henry T. K. Tse[1,2,4], Daniel R. Gossett[1,2,4], Peter Tseng[1,2], Andrew S. Utada[1,2], Hea-Jin Jung[5], Stephen Young[5,6,7], Amander T. Clark[8,9] & Dino Di Carlo[1,2]

Received: 22 October 2015
Accepted: 01 November 2016
Published: 02 December 2016

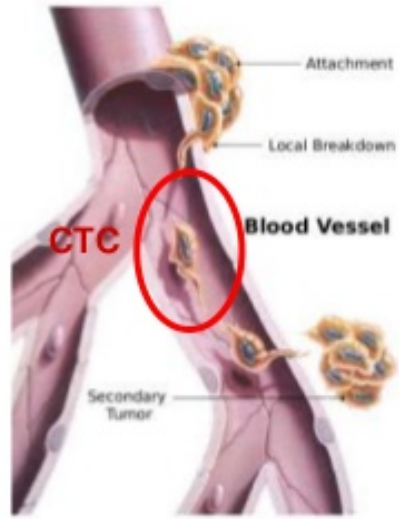
We introduce a label-free method to rapidly phenotype and classify cells purely based on physical properties. We extract 15 biophysical parameters from cells as they deform in a microfluidic stretching flow field via high-speed microscopy and apply machine-learning approaches to discriminate different cell types and states. When employing the full 15 dimensional dataset, the technique robustly classifies individual cells based on their pluripotency, with accuracy above 95%. Rheological and morphological properties of cells while deforming were critical for this classification. We also show the application of this method in accurate classifying cells based on their viability, drug screening and detecting populations of malignant cells in mixed samples. We show that some of the extracted parameters are not linearly independent, and in fact we reach maximum classification accuracy by using only a subset of parameters. However, the informative subsets could vary depending on cell types in the sample. This work shows the utility of an assay purely based on intrinsic biophysical properties of cells to identify changes in cell state. In addition to a label-free alternative to flow cytometry in certain applications, this work, also can provide novel intracellular metrics that would not be feasible with labeled approaches (i.e. flow cytometry).

Intrinsic physical properties of cells that reflect underlying molecular structure are indicators of cell state associated with a number of processes including cancer progression, stem cell differentiation, and drug response[1–3]. Nuclear and cytoplasmic structure or morphology have been one of the main tools for histological detection

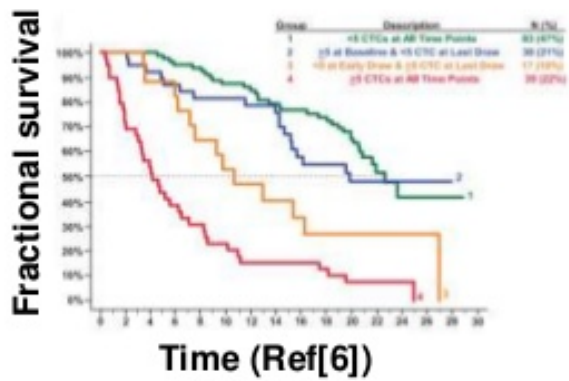
CTCs Are A Useful Resource for Cancer Analysis

Capture of circulating tumor cells enables assessment of patient metastatic state and chemotherapeutic response

- **Circulating Tumor Cells (CTCs)**
  - Found in the blood of metastatic cancer patients[1,2]
- **Approximately 1 CTC per billion blood cells**
- **CTCs have great clinical value**
  - Liquid biopsy
  - Genetic information[3]
  - Prognostic[4,5]



National Cancer Institute



[1] Allard+, Clin Canc Res 2004
[2] Danila+,Clin Canc Res 2007
[3] Scher+, Lancet Onc 2009
[4] Paris+, Cancer Lett 2009
[5] Okegawa+, J of Urology 2009
[6] Hayes+ CCR 2006

Erica Pratt

Application - Separating CTCs From Other Blood Cells

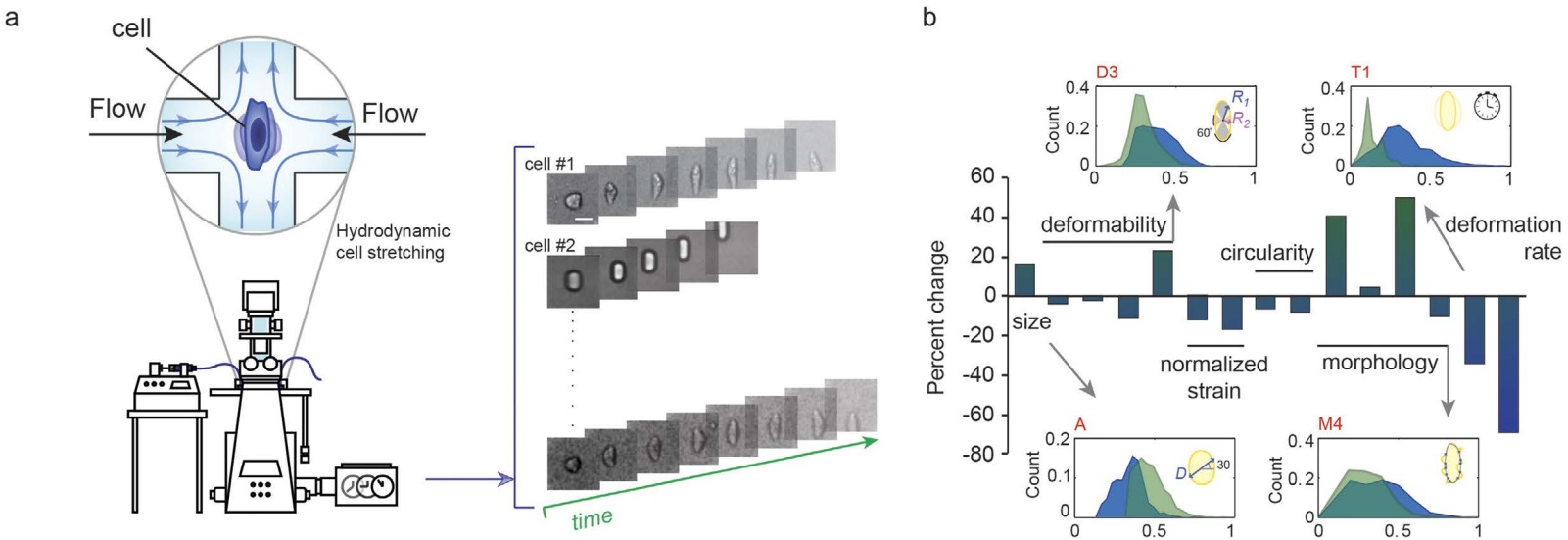
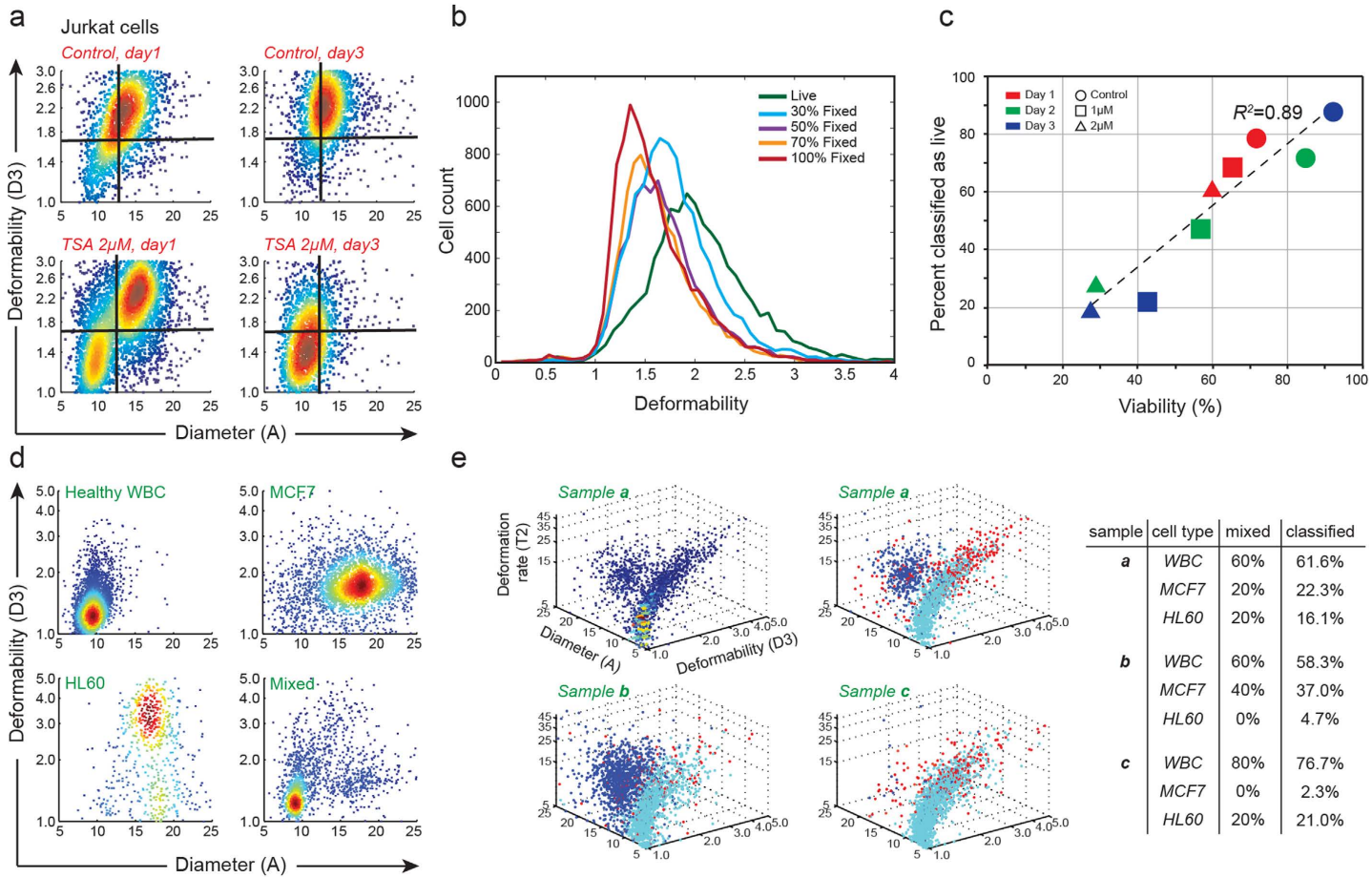


Figure 1. Comprehensive high-throughput quantification of cell physical properties. (**a**) Single cells in suspension are injected into a microfluidic device and are imaged while passing through a hydrodynamic stretching region. Several parameters are extracted from the image series captured from each cell. (**b**) The percent changes in median values of populations of hESCs (blue) and 14 day differentiated hESCs (green) are plotted for 15 parameters extracted from the image series. Histograms of >1000 single cells per condition are shown for select parameters, indicating substantial overlap in population characteristics when only considering a single parameter. (D3: Maximum deformability at the junction, A: Initial cell size, T1: Total deformation time, M4: Morphology metric extracted during deformation defined by the number of intersections of the trace and the moving average or the cell border).

Application - Separating CTCs From Other Blood Cells



sample	cell type	mixed	classified
a	*WBC*	60%	61.6%
	MCF7	20%	22.3%
	HL60	20%	16.1%
b	*WBC*	60%	58.3%
	MCF7	40%	37.0%
	HL60	0%	4.7%
c	*WBC*	80%	76.7%
	MCF7	0%	2.3%
	HL60	20%	21.0%

Figure 5. (**a**) Density profiles of Jurkat cells after 1–3 days of treatment with TSA (1, 2 μM) show that at higher dosages and treatment times, a portion of the population becomes stiff and small. (**b**) The profile of spiked and pure samples of live and fixed Jurkat cells. (**c**) Data shows high correlation between predicted viability and viability calculated using a standard cytotoxicity assay. (**d**) 2D profiles for healthy leukocytes (WBC), MCF7, and HL60 cells. (**e**) Three-dimensional profiles of diameter (A), deformability (D3), and deformation rate (T2) for three differently mixed samples of WBCs (cyan), HL60s (red), and MCF7s (blue). Classification using all 15 parameters led to $< 5\%$ error in classification of three differently mixed samples.

Implementation

sklearn has implementations for a variety of SVM methods:

- `sklearn.svm.SVC`
  - Performs single or multi-class classification
    - Multi-class is through one-vs-one scheme
  - Multiple kernels available
    - `linear`: $\langle x, x' \rangle$
    - `polynomial`: $(\gamma \langle x, x' \rangle + r)^d$
    - `rbf`: $\exp(-\gamma \|x - x'\|^2)$
    - `sigmoid`: $\tanh(\gamma \langle x, x' \rangle + r)$
- Alternative implementations
  - `sklearn.svm.NuSVC`
    - Additionally provides parameter to control number of support parameters
  - `sklearn.svm.LinearSVC`
    - Only support for linear kernel, with better scaling/options
    - For example can provide l1 or l2 regularization
    - Scales better for many samples
- Similar implementation in TensorFlow

Further Reading

- MIT 6.034: Support Vector Machines
- Computer Age Statistical Inference: Chapter 19
- sklearn: Support Vector Machines
- Linear Digressions Podcast
  - Maximal Margin Classifiers
  - The Kernel Trick and Support Vector Machines