ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

M.Sc. Program in Data Science
Department of Informatics

Optimization Techniques

Applications of Convex Optimization to Machine Learning

Instructor: G. ZOIS
georzois@aueb.com

Outline

- Linear Regression
  - Learning with a linear hypothesis
  - Least square problems
  - Solving least squares: Analytic solution and gradient descent
  - Other issues: Polynomial regression and regularization
- Support Vector Machines
  - Optimal margin classifiers
  - The role of duality
  - Regularization
  - Kernel functions

Linear Regression

Linear Regression

Suppose we are given a dataset in the form

- $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}),\ldots, (x^{(m)}, y^{(m)})$
- $x^{(i)}$: typically a vector with the values of the features for the i-th data point

$$x^{(i)} = \left(x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}\right)$$

- $y^{(i)}$: the label of the i-th data point (a real number)

Goal: Learn the function that best describes the dependence of y on the features

Linear Regression: We try to learn a linear function in the form

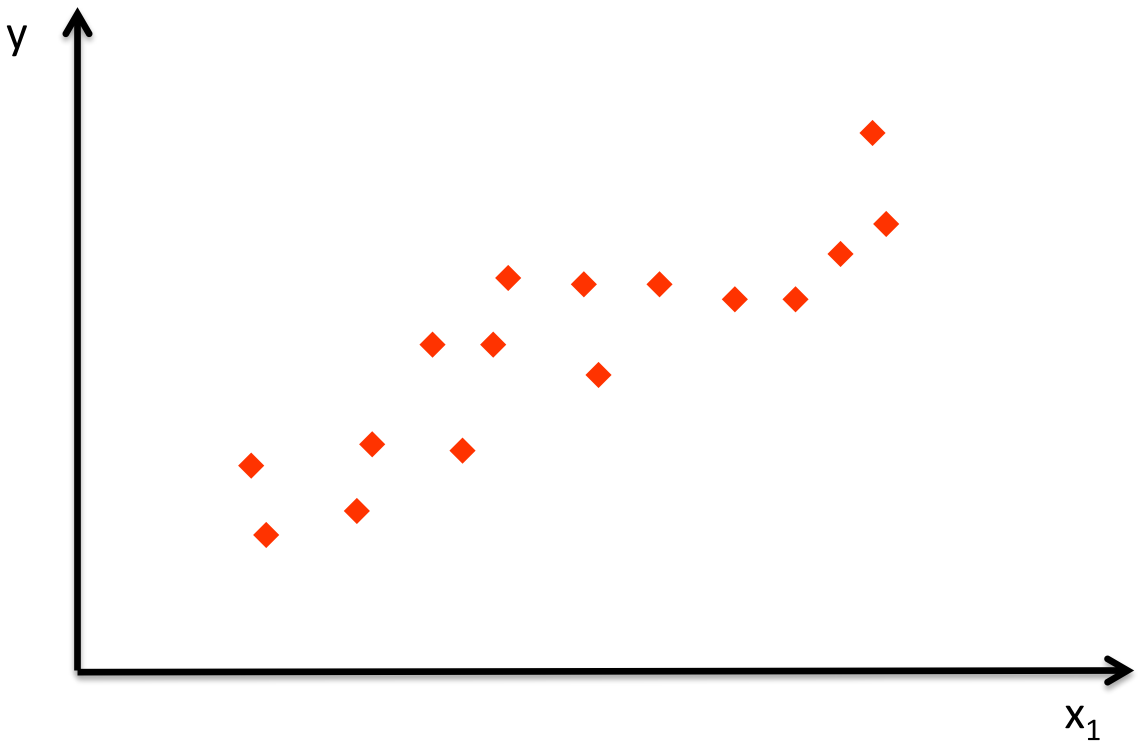
$$h(x) = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + w_0$$

- h(x) is then called a linear hypothesis

Linear Regression

A classic example

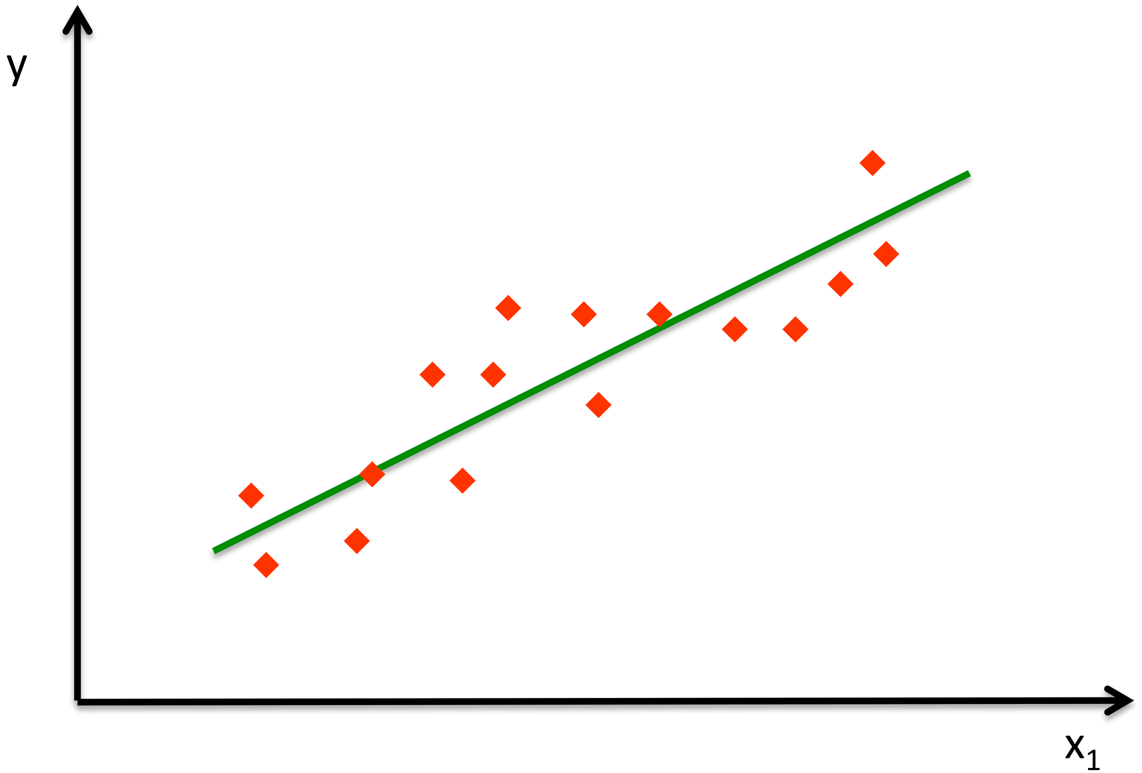
- Consider a 1-dimensional problem
- Supppose we want to predict the rent for apartments in a specific area of Athens
- x_1 = area of the apartment in sq. meters



- Dataset:
  - 1 feature (area)
  - $y^{(i)}$ = price
- We want to find a function in the form $h(x_1) = w_1 x_1 + w_0$ that best fits the data

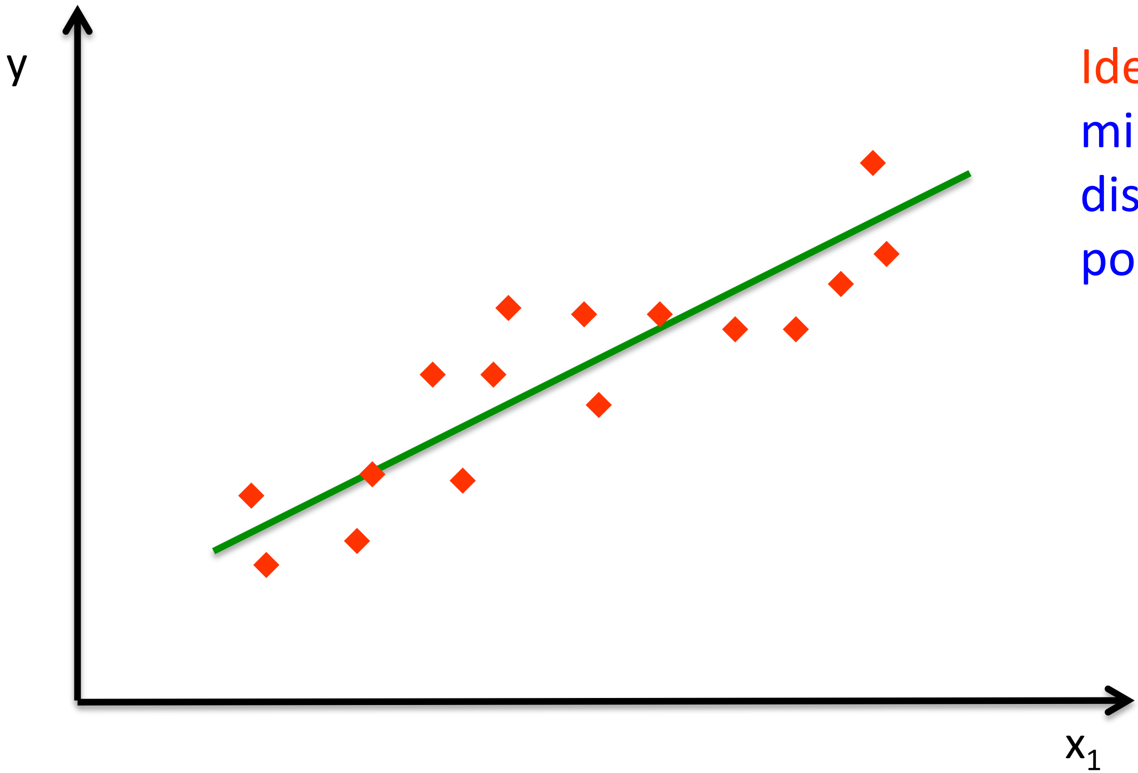
Linear Regression

How shall we decide which linear function fits best?



Linear Regression

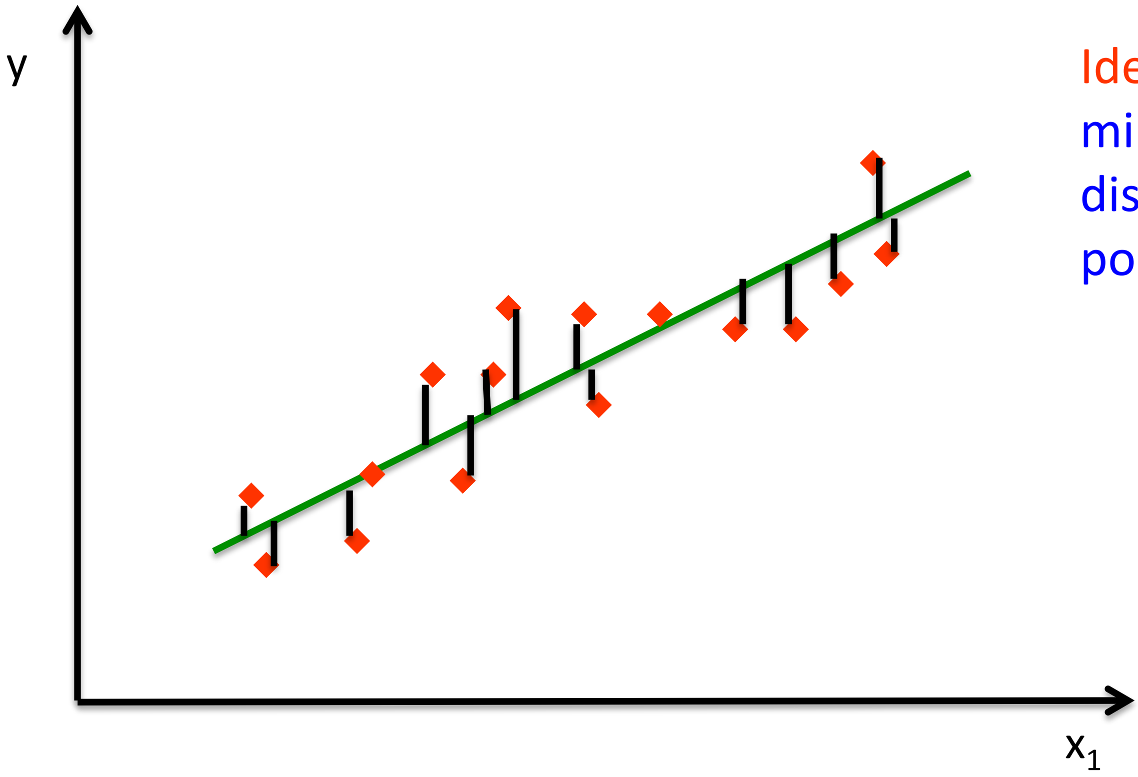
- There is no unique answer, every line will miss several points
- We need to select a loss function to evaluate the quality of the line picked



Idea: Pick the line that minimizes the (squared) distances from the data points

Linear Regression

- If the sum of the squared distances is small, we can say that we achieve a good approximation by a linear function

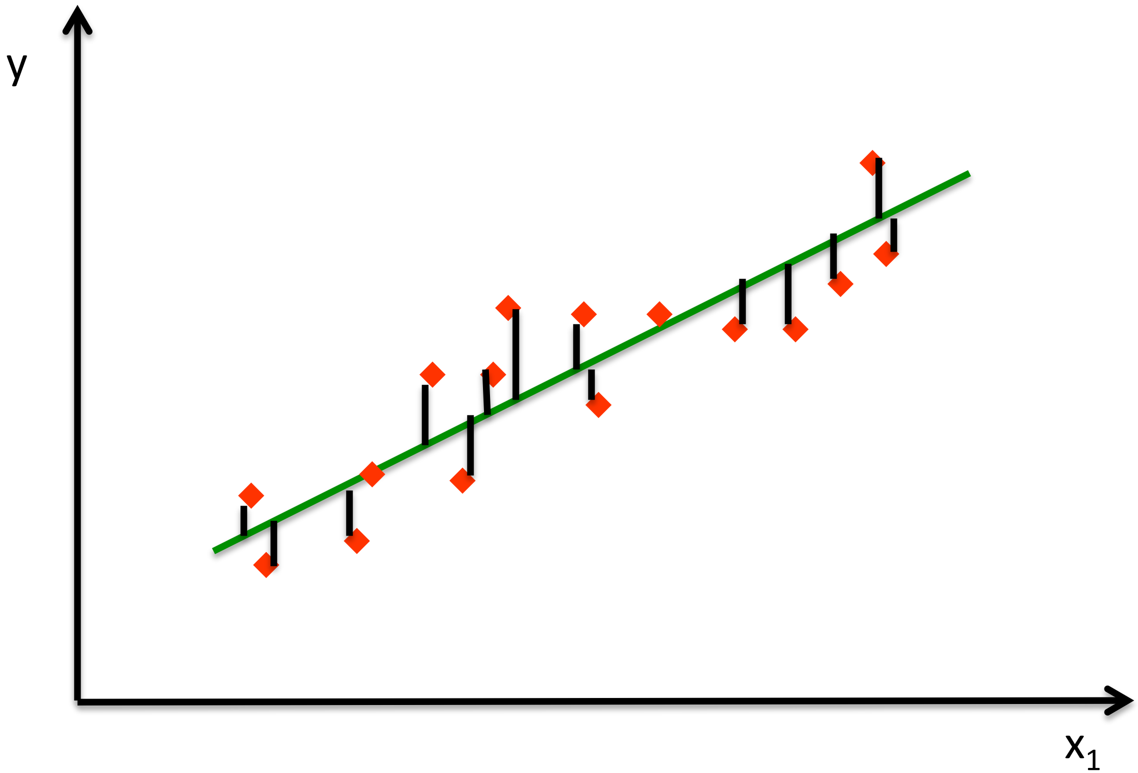


Idea: Pick the line that minimizes the (squared) distances from the data points

Linear Regression

- When we have n features (i.e. n variables), let $w = (w_0, w_1, w_2, \ldots, w_n)$
- Loss function:

$$C(w) = \frac{1}{2m} \sum_{i=1}^{m} \left[h(x^{(i)}) - y^{(i)} \right]^2$$



- We assumed m data points
- The division by 2m is for normalization

Linear Regression

This is a "least squares problem"

In more detail:

- In problems with one feature:

$$C(w) = \frac{1}{2m} \sum_{i=1}^{m} \left[w_1 x_1^{(i)} + w_0 - y^{(i)} \right]^2$$

- In problems with multiple features:

$$C(w) = \frac{1}{2m} \sum_{i=1}^{m} \left[w_1 x_1^{(i)} + w_2 x_2^{(i)} + \ldots + w_n x_n^{(i)} + w_0 - y^{(i)} \right]^2$$

- We want to find the vector w that minimizes C(w)

Least Squares Problems

- In some cases, we may have some extra constraints, e.g. some upper bound on ||w||
- If not then this is an unconstrained convex quadratic problem
    - Homework: check that C(w) is a convex function
- Analytic solution obtained by:

$$\nabla C(w) = 0$$

$$\frac{\partial C(w)}{\partial w_j} = \frac{1}{2m}\sum_{i=1}^{m} 2\left[h(x^{(i)}) - y^{(i)}\right] x_j^{(i)} = \frac{1}{m}\sum_{i=1}^{m}\left[h(x^{(i)}) - y^{(i)}\right] x_j^{(i)}$$

- The partial derivatives lead to linear equations

Least Squares Problems

In more concise form:

- For convenience, set $x_0^{(i)} = 1$ for each data point

$$x^{(i)} = \left(x_0^{(i)}, x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}\right) = \left(1, x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)}\right)$$

- Grouping together the equations
  - We can then write $h(x^{(i)})$ as $w^T x^{(i)}$
  - Let X be the matrix where the i-th row contains the i-th data point
  - Let y be the column vector with all the labels of the data points

- Then

$$\nabla C(w) = 0 \Rightarrow X^T \cdot X \cdot w = X^T \cdot y \Rightarrow w = (X^T \cdot X)^{-1} \cdot X^T \cdot y$$

Least Squares Problems

- What if the matrix $X^T \cdot X$ is not invertible?
- Of if we want to avoid solving a linear system with a large number of equations?

Gradient descent works very fast in this setting

- If the current solution is $w = (w_0, w_1, w_2, \ldots, w_n)$, then the update in iteration k for each w_j, $j=1,\ldots, n$, is (with step size α_k):

$$w_j = w_j - \frac{\alpha_k}{m} \sum_{i=1}^{m} \left[h(x^{(i)}) - y^{(i)} \right] x_j^{(i)}$$

Polynomial Regression

- In some problems a linear hypothesis does not suffice
- Next step would be to move to a polynomial hypothesis
- E.g. For one variable: we may want to search for a hypothesis of the form

$$h(x) = w_3x^3 + w_2x^2 + w_1x + w_0$$

- We can create polynomial features
- Each $x^{(i)}$ can be transformed into a new vector that includes these features
- We can apply linear regression on this transformed data set

Polynomial Regression

- If we have many variables to begin with?
- Again we can think of polynomials in all variables
- Hence, we can have features like x_1x_2 or x_2x_4 etc
- Suppose we want to fit the data with a polynomial of degree 2
- If we want to include all possible monomials, then for every data point x, we can define the transformation:

$$\phi(x) = (1, x_1, \ldots, x_n, x_1^2, x_1x_2, x_1x_3, \ldots, x_1x_n, x_2^2, x_2x_3, \ldots, x_n^2)$$

We can then do linear regression with the data set $(\phi(x^{(i)}), y^{(i)})$ for i=1,...,m

Regularized Regression

Overfitting:

- It can happen when we have too many features and small number of training examples
- Or if we use a polynomial of high degree, when a smaller one suffices

What can we do?

- It is observed that in the presence of overfitting, the parameters have very high absolute values
- Large variance
- Hence, we can “punish” large values in our objective function

Regularized Regression

New objective:

$$C(w) = \frac{1}{2m} \sum_{i=1}^{m} \left[h(x^{(i)}) - y^{(i)} \right]^2 + \frac{\lambda}{2m} ||w||^2$$

- Experimentation needed for choosing appropriate values of λ

How do we minimize the new C(w)?

- Again a convex problem
- Gradient descent still works quite well

This method is also referred to as Ridge Regression

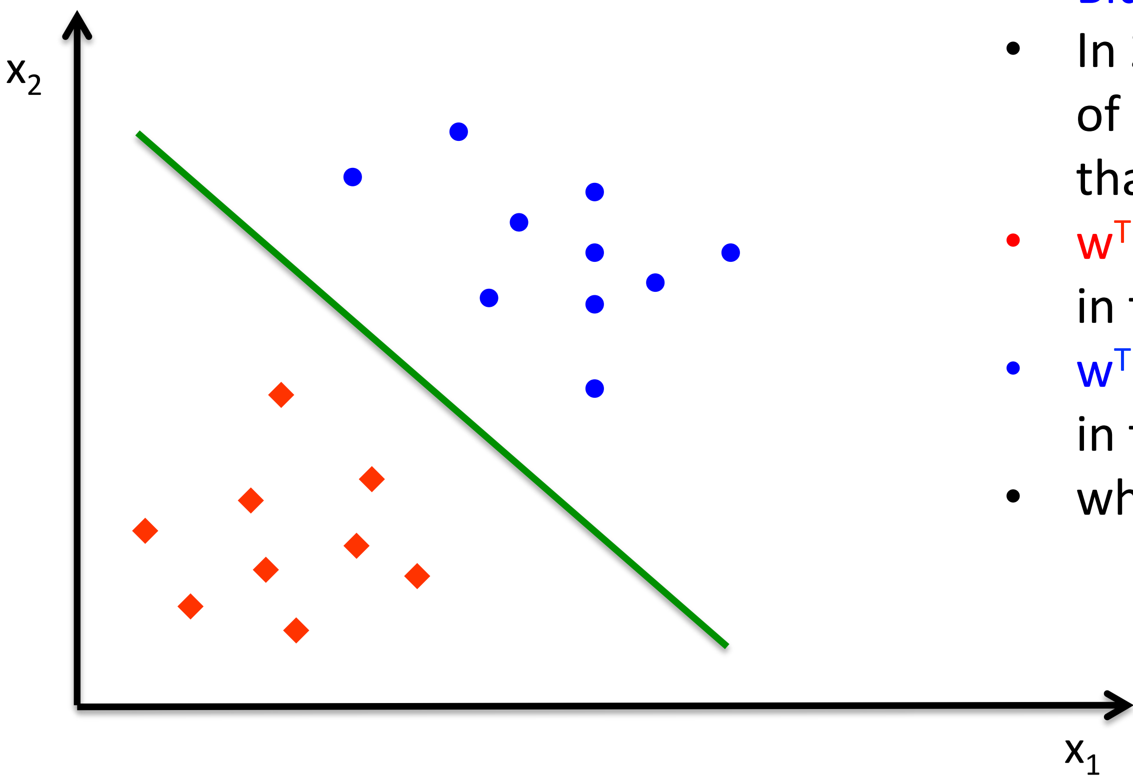
Support Vector Machines

Support Vector Machines

- One of the best families of supervised learning algorithms
- Big advantage: easily applicable in very high dimensional feature spaces
- Lagrange duality provides many insights for building SVMs

Classification Problems

- To begin with, suppose we have a linearly separable data set
- 2 labels: {-1, +1}



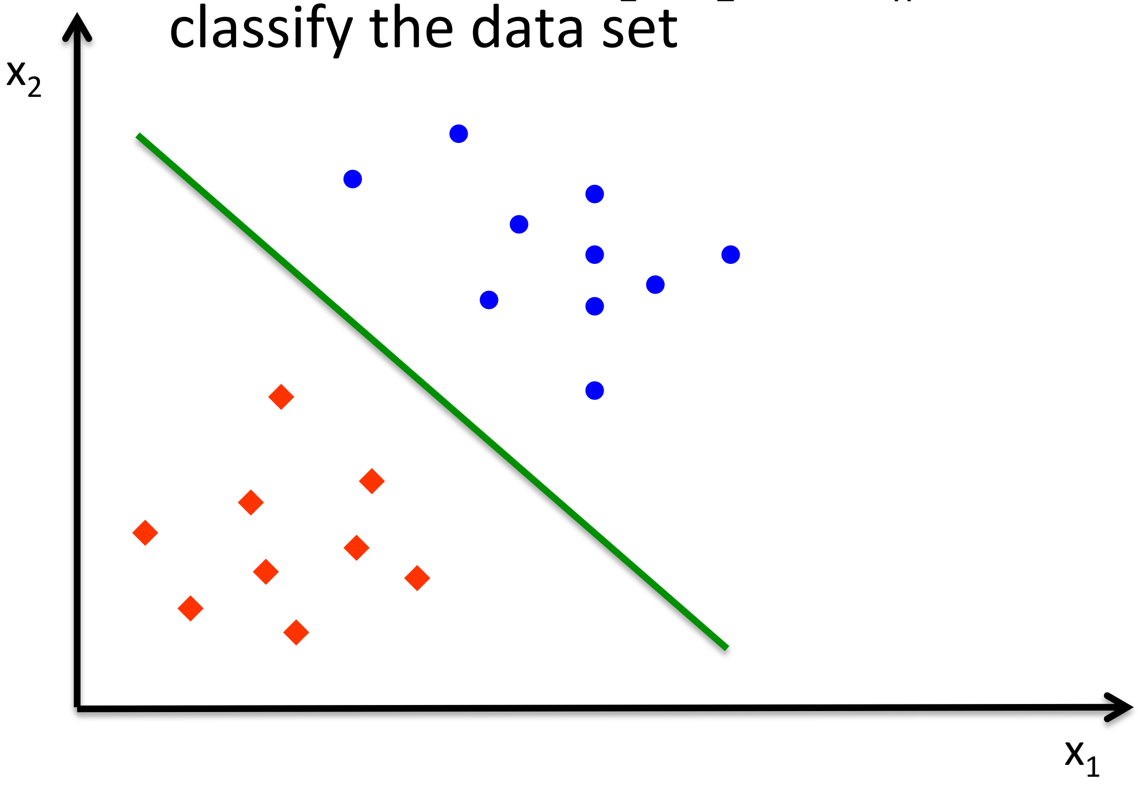
- Red class: label -1
- Blue class: label +1
- In 2 dimensions, there is a line of the form $w_1x_1 + w_2x_2 + b = 0$ that separates the 2 classes
- $w^T \cdot x + b < 0$ for every point x in the red class
- $w^T \cdot x + b > 0$ for every point x in the blue class
- where $w = (w_1, w_2)$

Classification Problems

- If each data point had n features: then there exists a hyperplane in R^n that separates the 2 classes:

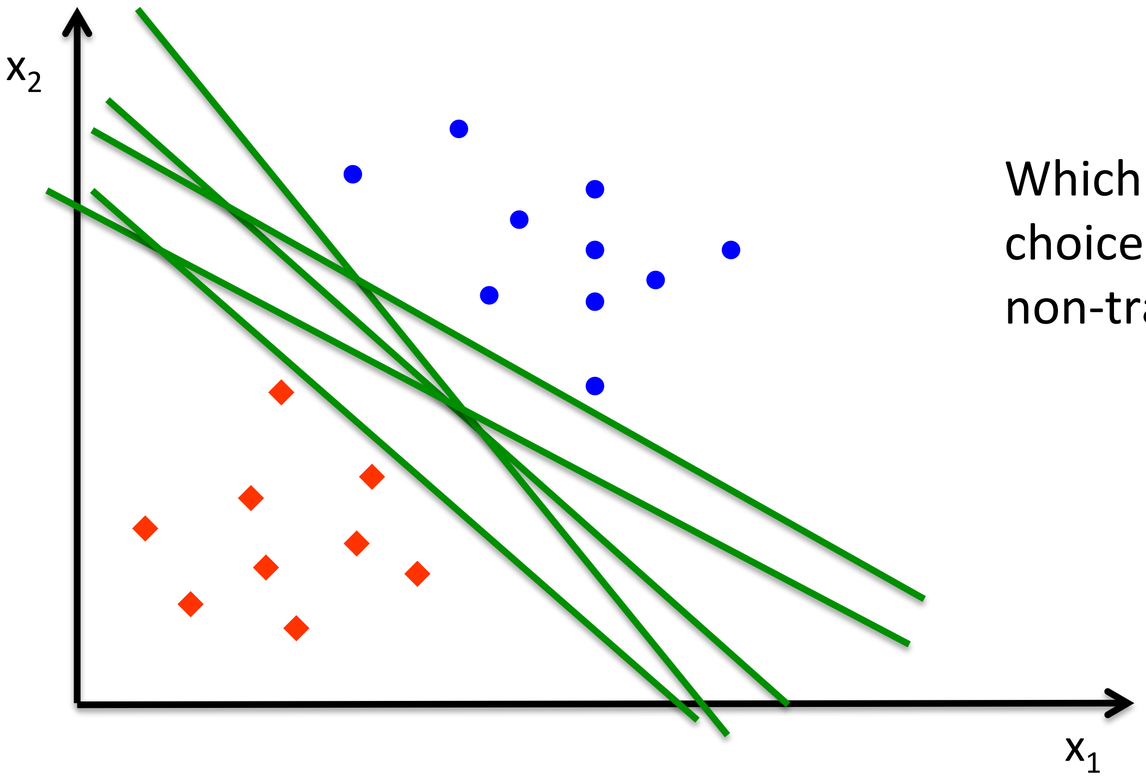
$$w_1x_1 + w_2x_2 + \ldots + w_nx_n + b = 0$$

- Goal: Find $w = (w_1, w_2, \ldots, w_n)$ and b so that we correctly classify the data set



Classification Problems

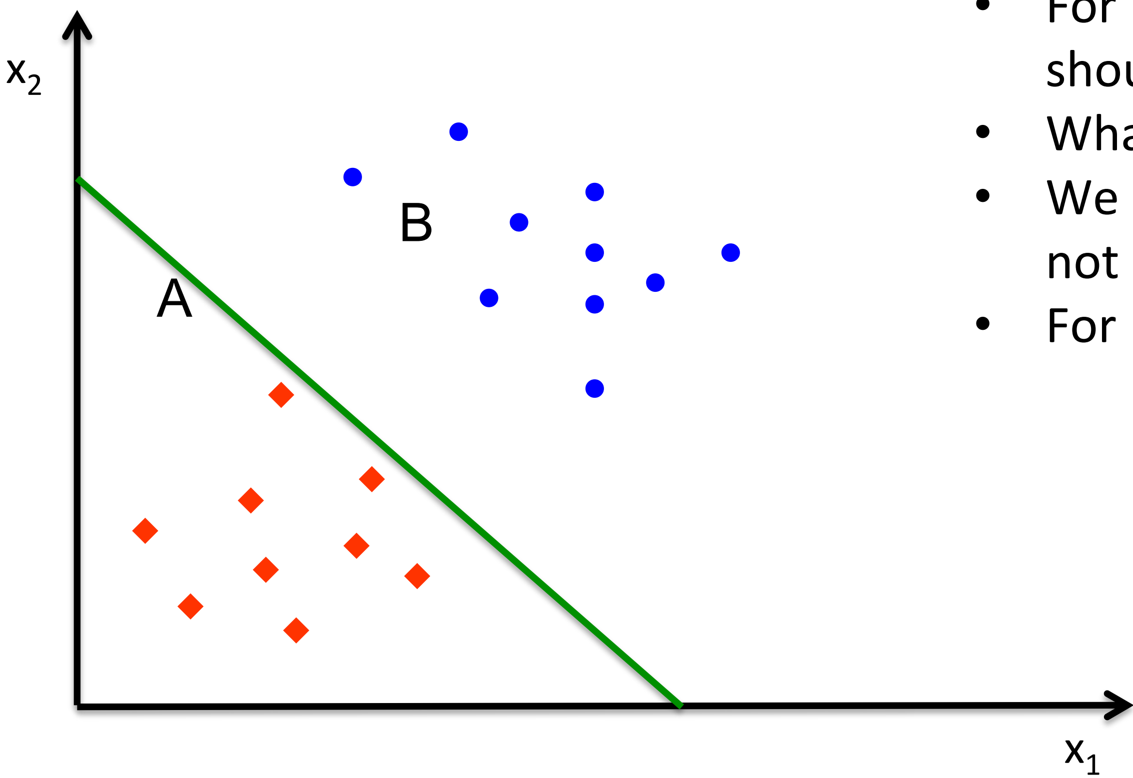
- The problem may admit many solutions
  - There can be too many lines that separate the 2 classes
- Is there a solution that is better than the others?



Which of these lines is a better choice for future predictions on non-training data?

Classification Problems

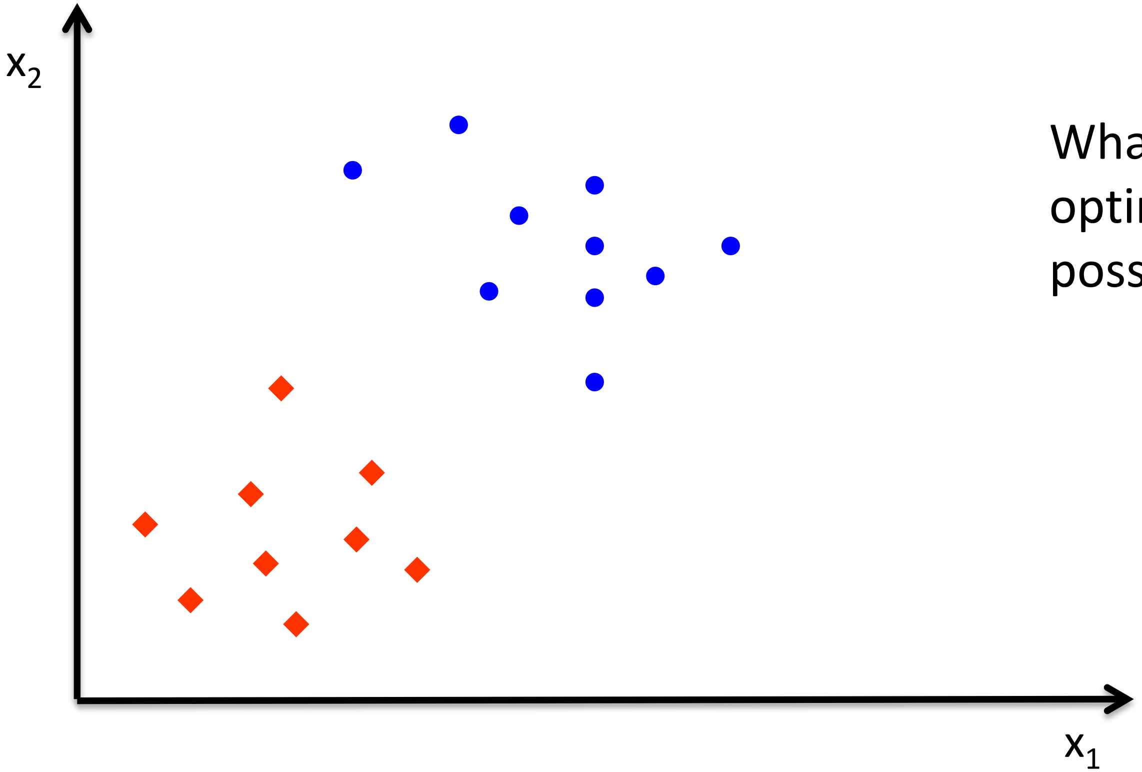
- Suppose we pick a line very close to the red class
- And suppose 2 new points, A and B, arrive for classification
  - Not part of the initial data set



- For B we can be pretty sure it should be classified as +1
- What about A?
- We can label it as -1 but we might not be sure about it
- For Point A: $w^T x + b$ is close to 0

Classification Problems

- Ideally, we would like a line, given by w, and b, such that:
- $w^T \cdot x + b << 0$ for every point x in the red class
- $w^T \cdot x + b >> 0$ for every point x in the blue class

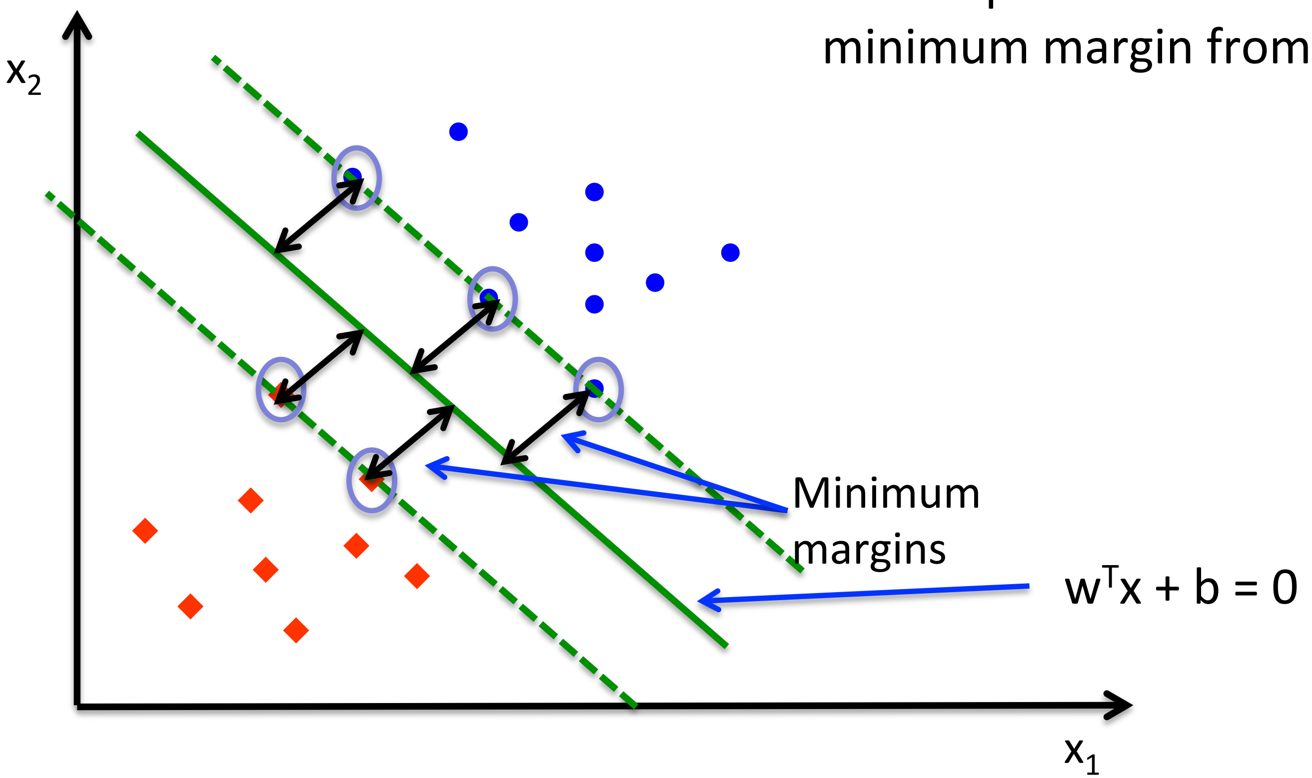


What is the criterion we should optimize to achieve the best possible results?

The Optimal Margin Classifier

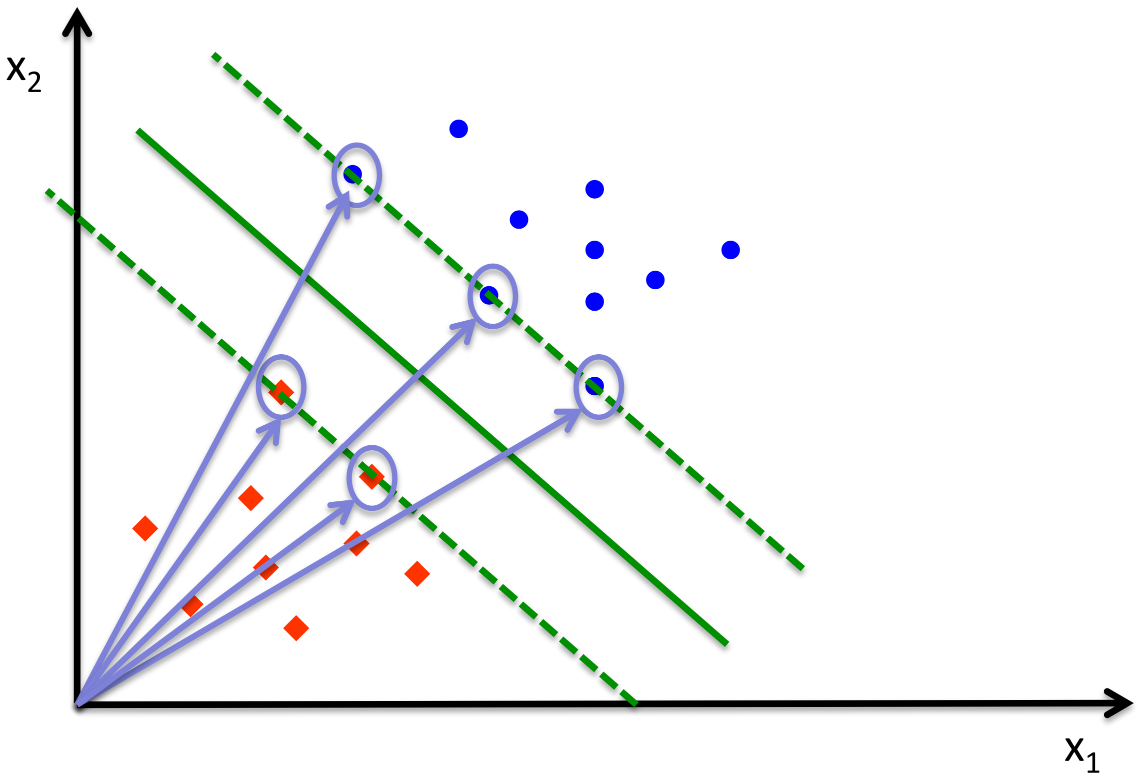
- Pick the line that maximizes the margins
- Margin of a data point: distance from the line selected

Hence: pick a line that maximizes the minimum margin from the data points



The Optimal Margin Classifier

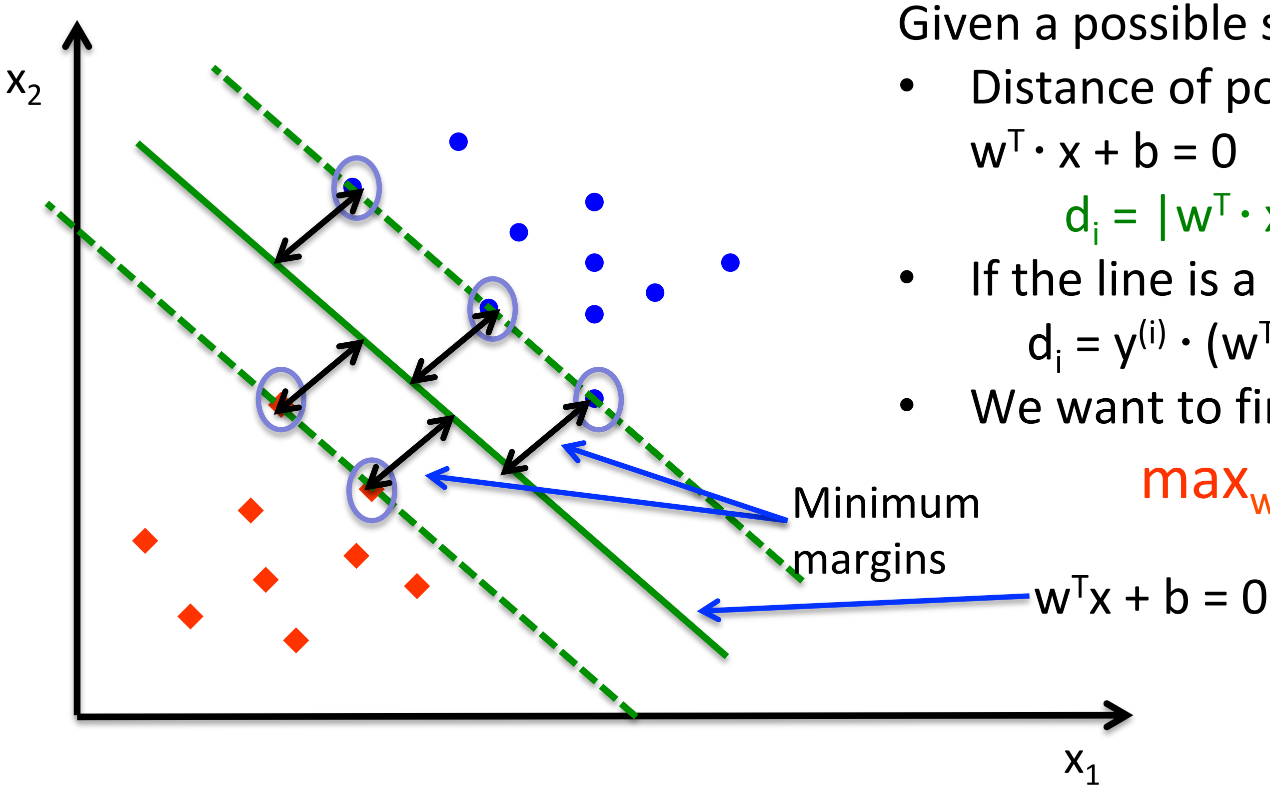
- Support vectors: The vectors formed by the data points with the minimum margins
- Will see later why they are useful



The Optimal Margin Classifier

Defining the optimization problem we care about:

- Suppose the data set is $(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$
- $y^{(i)}$ in $\{-1, +1\}$



Given a possible solution w and b:

- Distance of point $x^{(i)}$ from the line $w^T \cdot x + b = 0$

$$d_i = |w^T \cdot x^{(i)} + b| / ||w||$$

- If the line is a correct classifier

$$d_i = y^{(i)} \cdot (w^T \cdot x^{(i)} + b) / ||w||$$

- We want to find:

$$\max_{w,b} \min_i d_i$$

The Optimal Margin Classifier

First attempt to bring the problem to an amenable form:

$$\begin{aligned} &\max d \\ &\text{s.t.} \\ &d_i \geq d,\ i=1,\dots,m \end{aligned} \quad \Rightarrow \quad \begin{aligned} &\max\ r / \|w\| \\ &\text{s.t.} \\ &y^{(i)} \cdot (w^T \cdot x^{(i)} + b) / \|w\| \geq r/\|w\| \end{aligned}$$

$$\Rightarrow \quad \begin{aligned} &\max\ r / \|w\| \\ &\text{s.t.} \\ &y^{(i)} \cdot (w^T \cdot x^{(i)} + b) \geq r \end{aligned}$$

- Problem: Objective function is nasty (non-convex)
- No techniques known tailored for such functions

The Optimal Margin Classifier

Normalization:

- No need to have r as a variable, we can assume without loss of generality that r=1
- Suppose not
- Consider a solution w, b, such that $\min_i |w^T \cdot x^{(i)} + b| = a \neq 1$
- Then set w: = w/a, b:= b/a
- This is a new valid solution that satisfies what we want

Hence:

- We need to maximize 1 / ||w||
- Instead: we can minimize ||w||
- To bring the problem to a more familiar form, we will use as our objective function: $1/2 \, ||w||^2$

The Optimal Margin Classifier

$$\min \frac{1}{2}||w||^2$$

s. t.:

$$y^{(i)} \cdot (w^T \cdot x^{(i)} + b) \geq 1$$

$$\text{for } i = 1, \ldots, m$$

- Convex quadratic objective function
- Linear inequality constraints
- We can solve it with various ways
  - If we add slack variables, we have seen how to solve it using the KKT conditions
  - Otherwise interior point methods can also solve it quickly
  - There are also commercial tools specific for Quadratic Programming

The Optimal Margin Classifier

- We could consider the problem solved at this point

BUT:

- We can exploit Lagrange duality to derive the dual problem
- The dual will allow us to solve this much more efficiently
- Solving the dual works well even for very high dimensional spaces
- This also provides intuition regarding the support vectors and why it is useful that we usually have only “few” support vectors

The Dual Problem

- The Lagrange function:
  - We only have Lagrange multipliers for the inequality constraints
  - Let $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_m)$ be the vector of Lagrange multipliers

$$L(w, b;\ \alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{m} \alpha_i [y^{(i)}(w^T \cdot x^{(i)} + b) - 1]$$

- The dual function
  - We need to compute $\inf_{w,b} L(w, b; \alpha)$
  - To minimize L, we use the condition $\nabla L = 0$

The Dual Problem

- Deriving the dual function:

$$\frac{\partial L}{\partial w_j} = 0 \quad \text{for } j = 1, \ldots, n \quad \Rightarrow \quad w = \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} \qquad (1)$$

$$\frac{\partial L}{\partial b} = 0 \quad \Rightarrow \quad \sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \qquad (2)$$

- Plug in (1) into the Lagrangian function
  - After some algebraic manipulations:

$$L(w, b;\ \alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j (x^{(i)})^T \cdot x^{(j)} - b \sum_{j=1}^{m} \alpha_i y^{(i)}$$

  - By using (2), the last term vanishes

The Dual Problem

- Summarizing, we arrive at the following dual problem:

$$\max \; W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle$$

s. t.:

$$\sum_{i=1}^{m} \alpha_i y^{(i)} = 0$$

$$\alpha_i \geq 0, \;\text{for}\; i = 1, \ldots, m$$

- Notation: for convenience, we denote by $\langle x^{(i)}, x^{(j)} \rangle$ the inner product of the 2 vectors, i.e., $(x^{(i)})^T \cdot x^{(j)}$

Lessons and insights learnt from the dual

1. If we manage to solve the dual, we can easily use (1) and (2) to compute the optimal solution w* and b* for the primal
2. Why could it be easier to solve the dual?
   - Let us look at the KKT conditions
   - Because we have inequalities in the primal, we have the complementarity conditions:

     $$\alpha_i \cdot [y^{(i)} \cdot (w^T \cdot x^{(i)} + b) - 1] = 0$$

   - Hence for all data points where $y^{(i)} \cdot (w^T \cdot x^{(i)} + b) > 1 \Rightarrow \alpha_i = 0$
   - $\alpha_i > 0$ only for data points with the minimum margin
   - These are the points corresponding precisely to the support vectors!
   - In practice, we do not expect too many points to attain the minimum margin
   - Hence, even with thousands of training data, we expect to have few support vectors $\Rightarrow$ few non-zero variables in the dual

Lessons and insights learnt from the dual

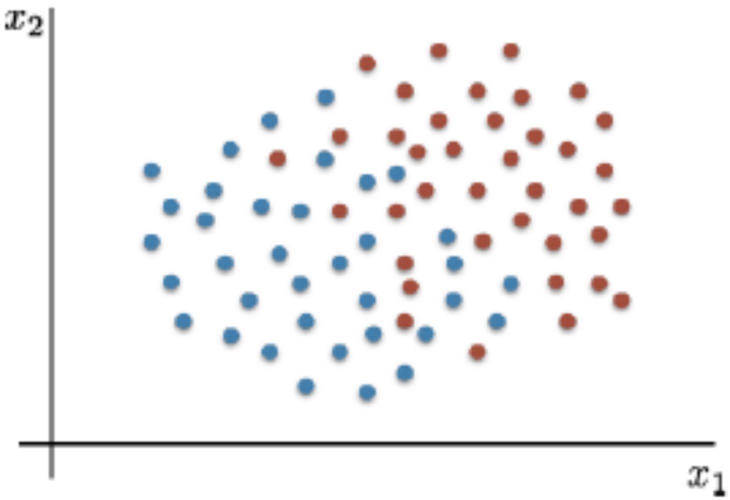
3. The dual is written in terms of the inner products
   - Suppose we solve the dual
   - Suppose also we now want to make a prediction for a new data point x
   - We should calculate $w^T x + b$ and decide which label to give
   - But by (1) this is

$$w^T \cdot x + b = \sum_{i=1}^{m} \alpha_i y^{(i)} \langle x^{(i)}, x \rangle + b$$

   - If many α_i's are zero, this needs only a few inner product calculations
   - No need to calculate w and b to make the prediction

Almost Separable Data

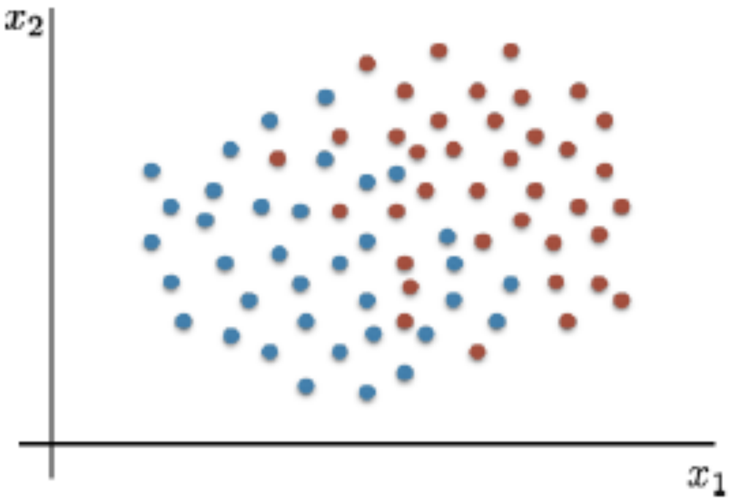
- Some times the data may not be linearly separable even though it is obvious that there are 2 separable classes of data



- In this example, the dataset is almost linearly separable
- We will treat some (few) examples as "outliers"

Almost Separable Data

- We cannot demand that $y^{(i)} \cdot (w^T \cdot x^{(i)} + b) \geq 1$
- But we can relax the constraints



- Ask for $y^{(i)} \cdot (w^T \cdot x^{(i)} + b) \geq 1 - s_i$ (slack variable $s_i \geq 0$)
- Penalize the sum

Almost Separable Data

- The new primal problem

$$\min \; \frac{1}{2}||w||^2 + C\sum_{i=1}^{m} s_i$$

s. t.:

$$y^{(i)} \cdot (w^T \cdot x^{(i)} + b) \geq 1 - s_i, \;\; i = 1, \ldots, m$$

$$s_i \geq 0, \;\; i = 1, \ldots, m$$

- And the new dual problem

$$\max \; W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle$$

s. t.:

$$\sum_{i=1}^{m} \alpha_i y^{(i)} = 0$$

$$0 \leq \alpha_i \leq C, \text{ for } i = 1, \ldots, m$$

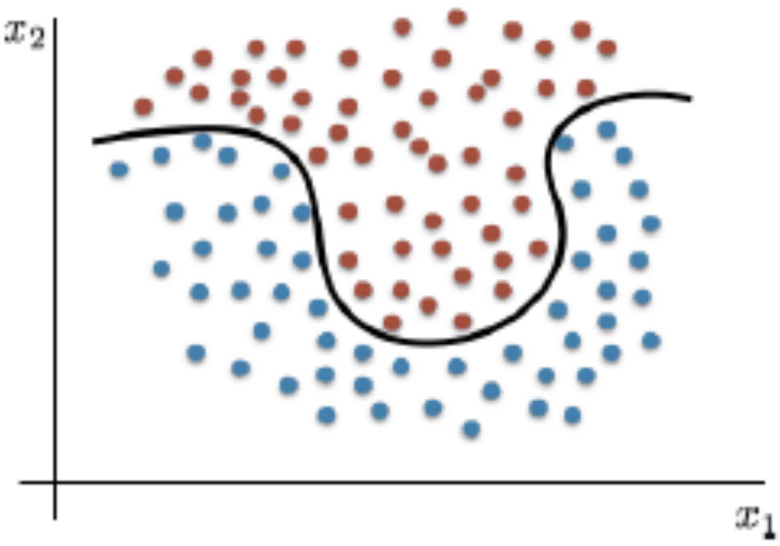
Almost Separable Data

- Lagrange duality works almost in the same way as before
- Only difference is the upper bound on each α_i
- Sanity check: Derive the new dual on your own

- Again equations (1) and (2) still valid
- Hence, again predictions on new data points can be made using inner products

Kernels

- What happens when the data are not even close to linearly separable?



Separable by a curve but not by a line

- We can try to find a polynomial that separates the 2 classes
- Similar in spirit to polynomial regression
- This is where the real power of SVMs arises

Kernels

- We can create polynomial features
- Each $x^{(i)}$ can be transformed into a new vector that includes these features, say $\phi(x^{(i)})$
- Instead of the inner products $\langle x^{(i)}, x^{(j)} \rangle$, we will now have $\langle \phi(x^{(i)}), \phi(x^{(j)}) \rangle$
- If we are careful, this can be done very efficiently

Definition: Given $\phi(x)$, a kernel is a function K such that

$$K(x, z) = \langle \phi(x), \phi(z) \rangle$$

Kernels

- It is instructive to look at some examples of kernels

1. Suppose $\phi(x) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, \sqrt{2}x_1x_2, x_2^2)$

Observation: $K(x, z) = \langle x, z \rangle^2 = (x^T z)^2$

2. Suppose now

$$\phi(x) = (x_1^2, x_1x_2, x_1x_3, x_2x_1, x_2^2, x_2x_3, x_3x_1, x_3x_2, x_3^2)$$

Again $K(x, z) = (x^T z)^2$

If we had n variables instead of 3:

- Computing $\langle \phi(x), \phi(z) \rangle$ takes $O(n^2)$ time
- Computing $K(x, z)$ takes only $O(n)$ time

Kernels

- In general we can pick our transformation so that

$$K(x, z) = [\beta \langle x, z \rangle + \gamma]^p$$

For appropriately chosen β, γ

- New objective function in the dual

$$W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j K(x^{(i)}, x^{(j)})$$

Main Conclusions:

- We can incorporate high dimensional feature spaces
- All we need is inner product computations
- No need to compute ϕ(x), we only need to compute K
- Hence: we can learn in a high dimensional feature space without the need to explicitly represent the new features

Solving the dual

- How can we actually solve the dual?
- The best approach is via the SMO algorithm (Sequential Minimal Optimization)
- Derived by Platt (1998)

Main ideas:

- A local search approach
- Suppose we keep all variables fixed and try to update a single variable α_i
- By (2) we cannot do that, if we fix m-1 variables, this fixes the last variable as well
- We do local search on pairs of variables
  - Pick a pair of variables, and keep the other m-2 variables fixed
  - Find a way to update these 2 variables so as to make progress

Reading Material

- Lecture Notes on Support Vector Machines from the machine learning course of Andrew Ng (Stanford): https://sgfin.github.io/files/notes/CS229_Lecture_Notes.pdf
- Technical report by Platt on the SMO algorithm: https://www.microsoft.com/en-us/research/uploads/prod/1998/04/sequential-minimal-optimization.pdf
- Machine Learning on Coursera by Andrew Ng also very illustrative