UNIVERSITY OF TORONTO
SCHOOL OF CONTINUING STUDIES

3253 - Analytic Techniques and Machine Learning

Module 6: Support Vector Machines

Course Plan

Module Titles
Module 1 – Introduction to Machine Learning
Module 2 – End to End Machine Learning Project
Module 3 – Classification
Module 4 – Clustering and Unsupervised Learning
Module 5 – Training Models and Feature Selection
Module 6 – Current Focus: Support Vector Machines
Module 7 – Decision Trees and Ensemble Learning
Module 8 – Dimensionality Reduction
Module 9 – Introduction to TensorFlow
Module 10 – Introduction to Deep Learning and Deep Neural Networks
Module 11 – Distributing TensorFlow, CNNs and RNNs
Module 12 – Final Assignment and Presentations (no content)

Learning Outcomes for this Module

- Apply linear and non-linear Support Vector Machine (SVM) to classification problems
- Use SVM for regression problems
- Explore how SVM works

Topics for this Module

- **6.1** Introduction to Support Vector Machines (SVM)
- **6.2** Linear classification with SVM
- **6.3** Non-linear classification
- **6.4** SVM regression
- **6.5** How SVM works
- **6.6** Resources and Wrap-up

UNIVERSITY OF TORONTO
SCHOOL OF CONTINUING STUDIES

Module 6 – Section 1

Introduction to Support Vector Machines (SVM)

SVM

- An ML algorithm for either classification or regression
- It can model both linear and non-linear patterns
- Most appropriate for small or mid-size data sets
- Training points (x, y) have labels +1 or -1

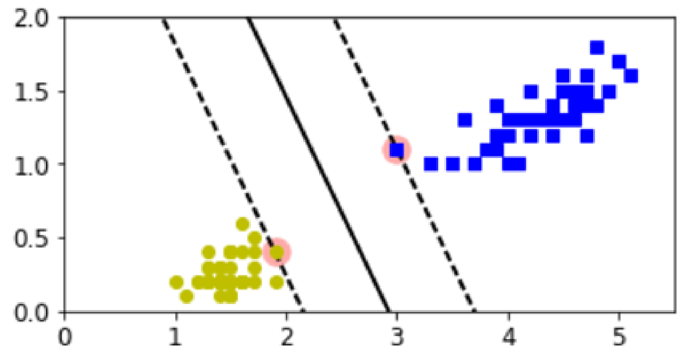
UNIVERSITY OF TORONTO
SCHOOL OF CONTINUING STUDIES

Module 6 – Section 2

Linear Classification with SVM

Linear Classification

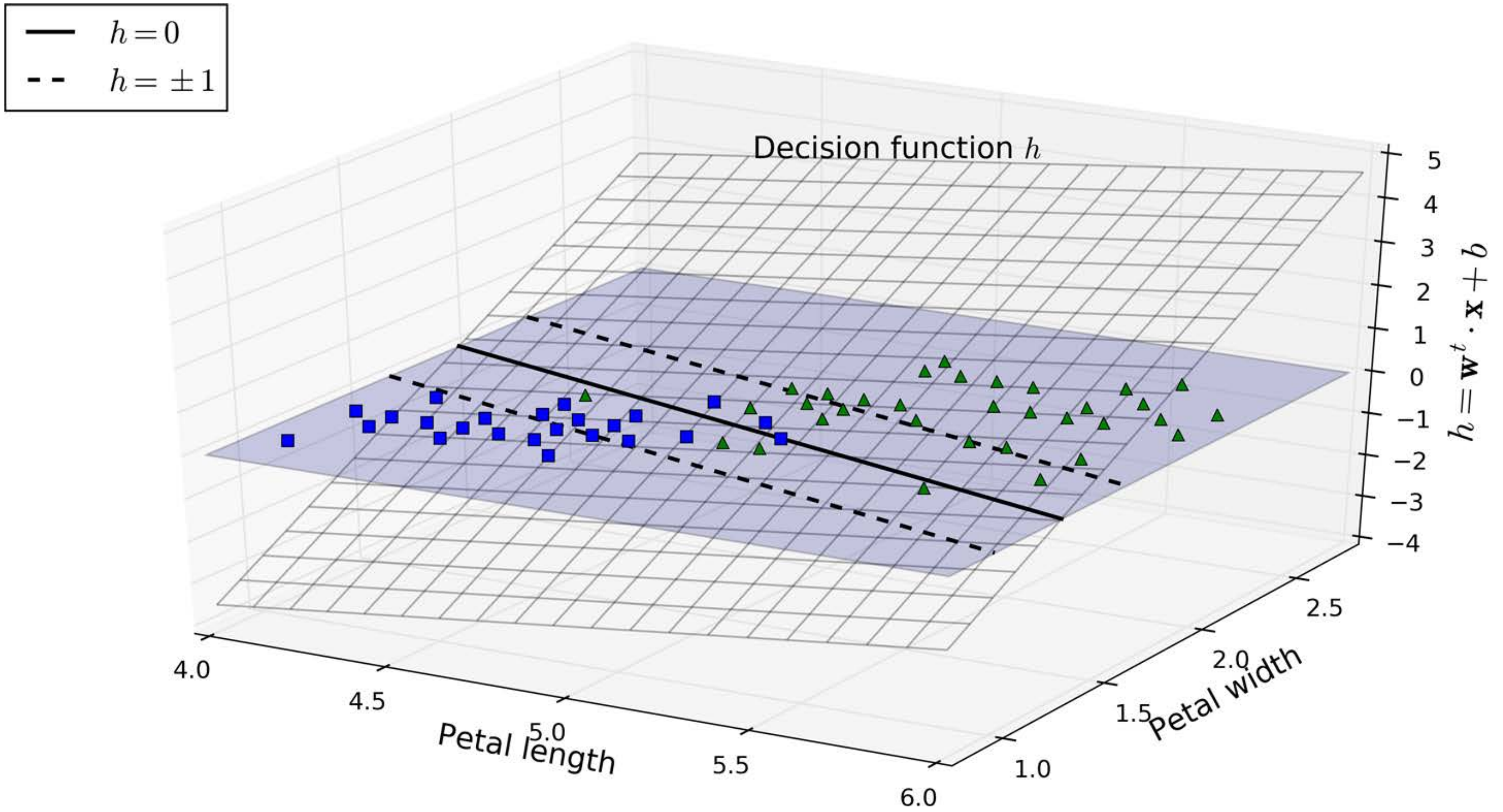
- A training set is linearly separable if a hyperplane leaves class +1 on one side and class -1 on the other side
- SVM finds the hyperplane that leaves the largest possible margin on both sides, until the first examples are found



Linear Classification (cont'd)

- A hyperplane in the plane is a line
- The points (x_1, x_2) in the line satisfy the equation $w_1 x_1 + w_2 x_2 + b = 0$ with w_1, w_2, b constants
- (w_1, w_2) is $\perp$ to the line
- A hyperplane in 3D space is a plane described as $w_1 x_1 + w_2 x_2 + w_3 x_3 + b = 0$
- In n-dimensions (features), a hyperplane is $w_1 x_1 + \ldots + w_n x_n + b = w^T \cdot x + b = 0$

Linear Classification (cont'd)



Linear Classification (cont'd)

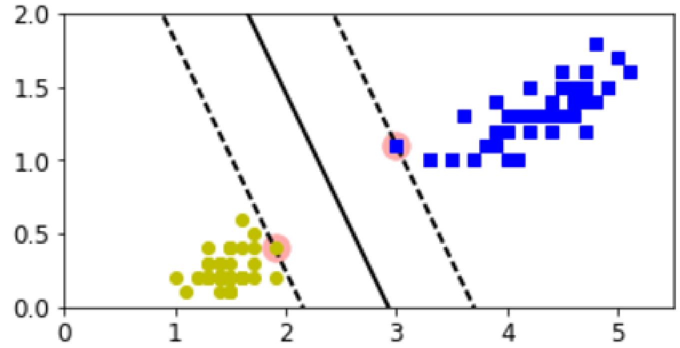
- If the data is linearly separable, the SVM algorithm finds w and b, so that class +1 satisfies $w^T \cdot x + b \geq 1$ and class -1 satisfies $w^T \cdot x + b \leq -1$

- Once w, b are known, new points x are classified by:

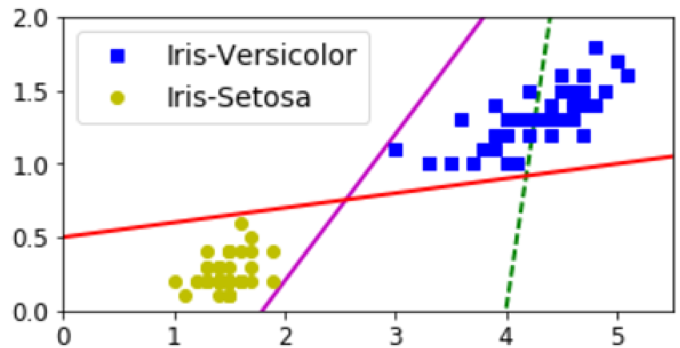
$$y = \begin{cases} +1, & if\ w^T \cdot x + b > 0 \\ -1, & if\ w^T \cdot x + b < 0 \end{cases}$$

Linear Classification (cont'd)

- What does an SVM solution look like?

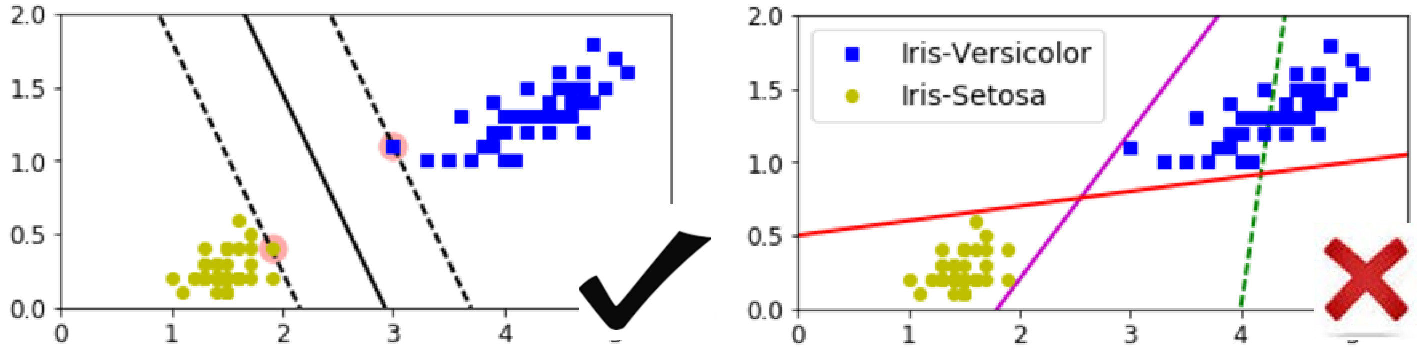


- What does it *not* look like?



Linear Classification (cont'd)

- What does an SVM solution look like?



- The vectors on the dashed lines are the *support vectors,* and determine the solution
- Neither Setosa examples to the far left, or Versicolor examples to the far right, determine the solution

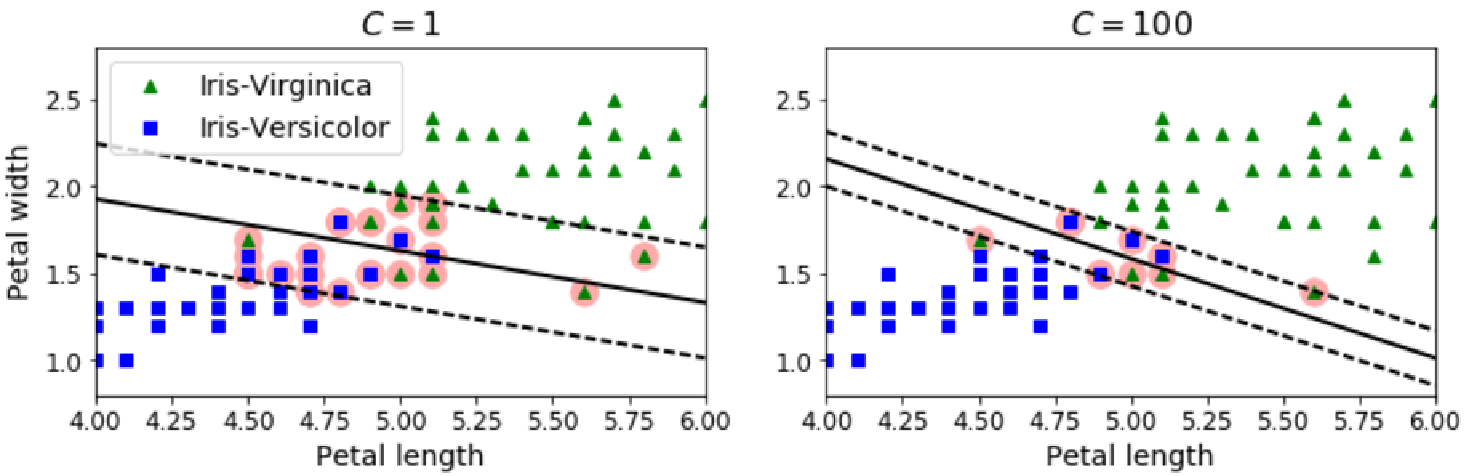
UNIVERSITY OF TORONTO
SCHOOL OF CONTINUING STUDIES

Linear Classification (cont'd)

- SVM described above is *hard margin*, no examples between dashed lines
- *Soft margin* SVM allows violations of this: examples within the dashed lines
- The solution finds a balance between wide margin and number of violations
- This is managed by a hyperparameter C
- Low C ⇔ big margin & more violations

Linear Classification (cont'd)

- C can be used for regularization
- If SVM overfits, decrease C. It allows violations, less aggressively fitting the data



- Violations may not be misclassifications

UNIVERSITY OF TORONTO
SCHOOL OF CONTINUING STUDIES

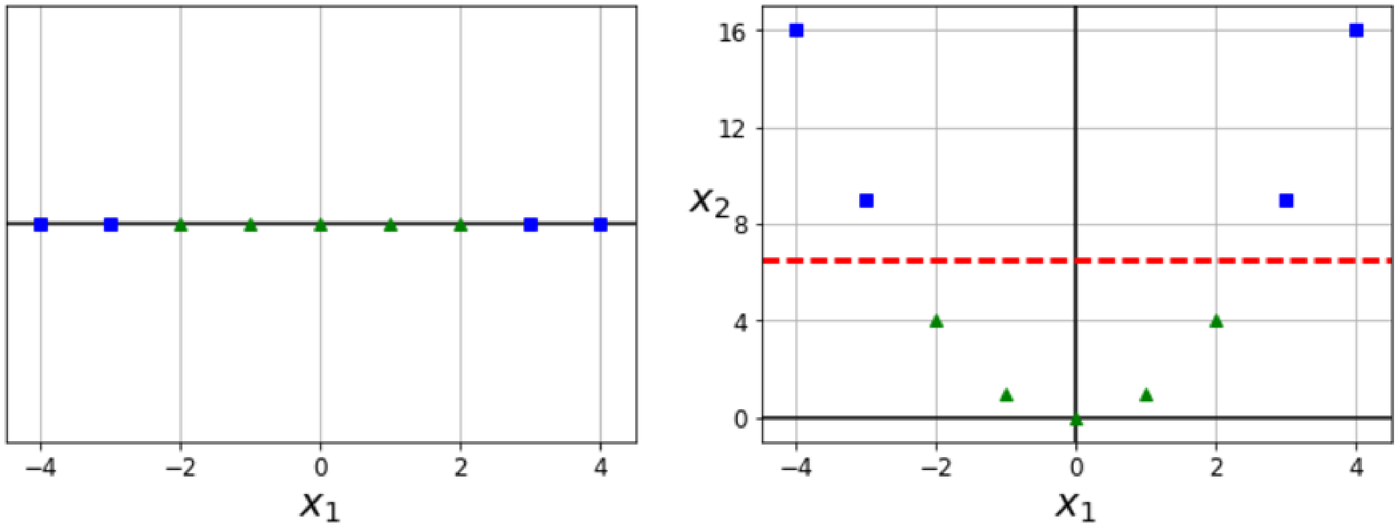
UNIVERSITY OF TORONTO
SCHOOL OF CONTINUING STUDIES

Module 6 – Section 3

Non-Linear Classification

Non-Linear Classification

- Non linearly-separable datasets can become so by adding new features
- A non linearly separable 1-feature set can become separable by transforming the feature (in this example, by squaring it)



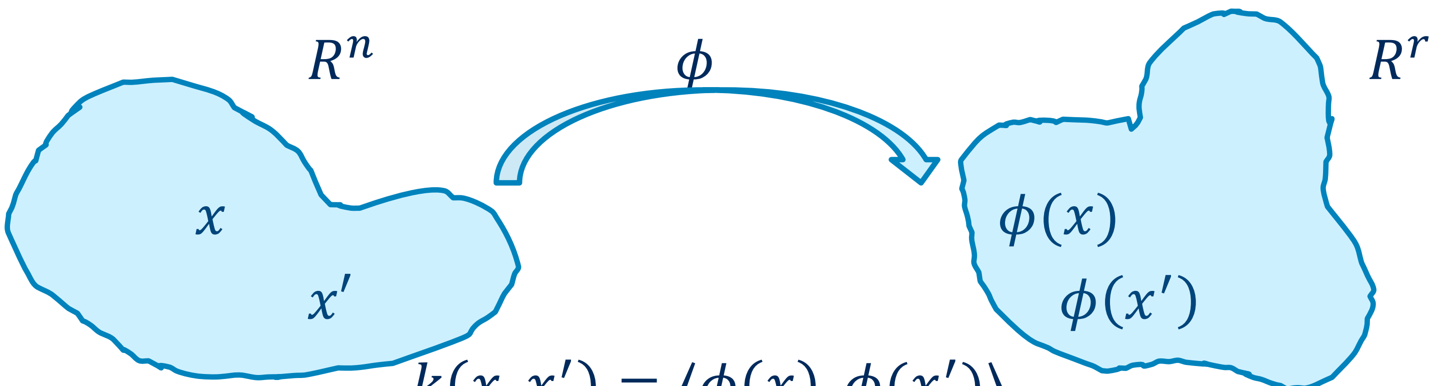
Non-Linear Classification (cont'd)

- Adding non-linear features manually is not the best approach. There's a better way!
- The algorithm works as if the original features are transformed non-linearly into a high dimensional space $\phi: R^n \rightarrow R^r, r \gg n$
- In R^r the dataset (hopefully) becomes separable, and linear SVM is applied
- The transformation ϕ is unknown. The algorithm does not actually use it

Non-Linear Classification (cont'd)

- All that is needed to do linear SVM is to apply inner product, with which geometry can be done, and w, b be found
- To do linear SVM is the transformed space we only need inner product $\langle \phi(x), \phi(x') \rangle$
- Kernel trick: only need a kernel in the original space $k: R^n \times R^n \rightarrow R$. Mercer's Thm. assures there exists $\phi: R^n \rightarrow R^r$ (for some r) such that $k(x, x') = \langle \phi(x), \phi(x') \rangle$

Non-Linear Classification (cont'd)



$$\underbrace{k(x, x')}_{\text{Doable}} = \underbrace{\langle \phi(x), \phi(x') \rangle}_{\text{Wanted}}$$

Non-linear SVM requires specifying the kernel

There are many possible kernels (hyperparameter)

Popular kernels: polynomial, gaussian, linear

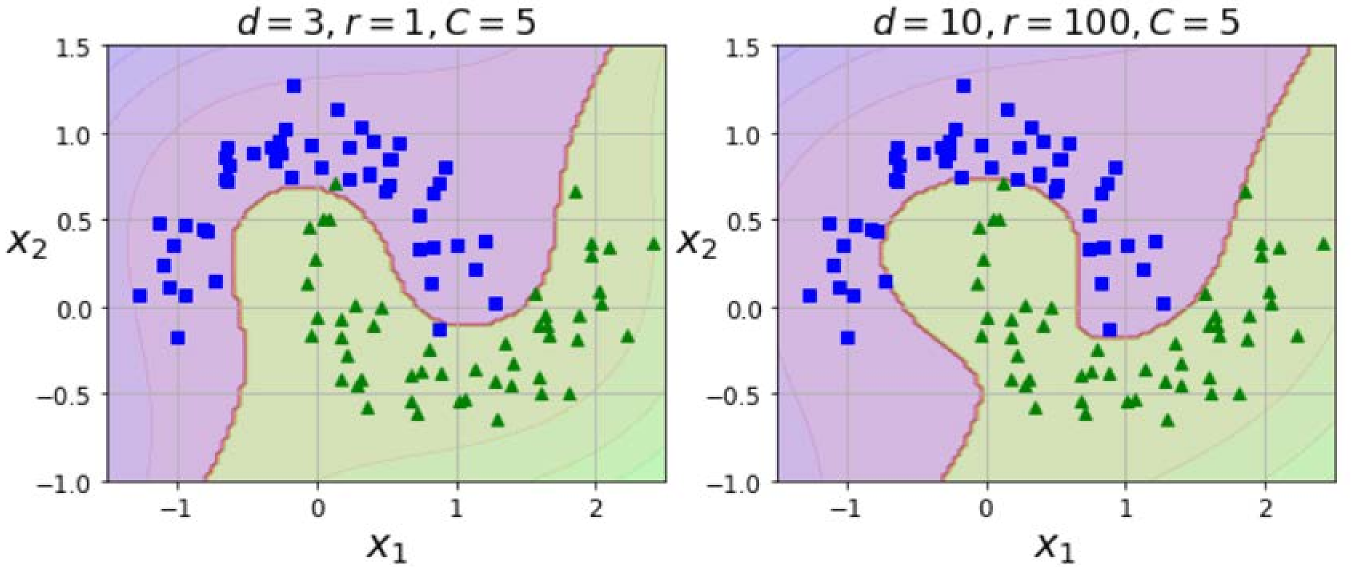
Non-Linear Classification (cont'd)

- Linear: $k(x, x') = \langle x, x' \rangle$
- Polynomial: $k(x, x') = (\langle x, x' \rangle + r)^d$
- Gaussian: $k(x, x') = \exp\left(-\gamma \left|\left|x - x'\right|\right|^2\right) = \exp(-\gamma\langle x -$

Non-Linear Classification (cont'd)

- Polynomial kernel is analogous to replacing the original features by many polynomial transformations of them

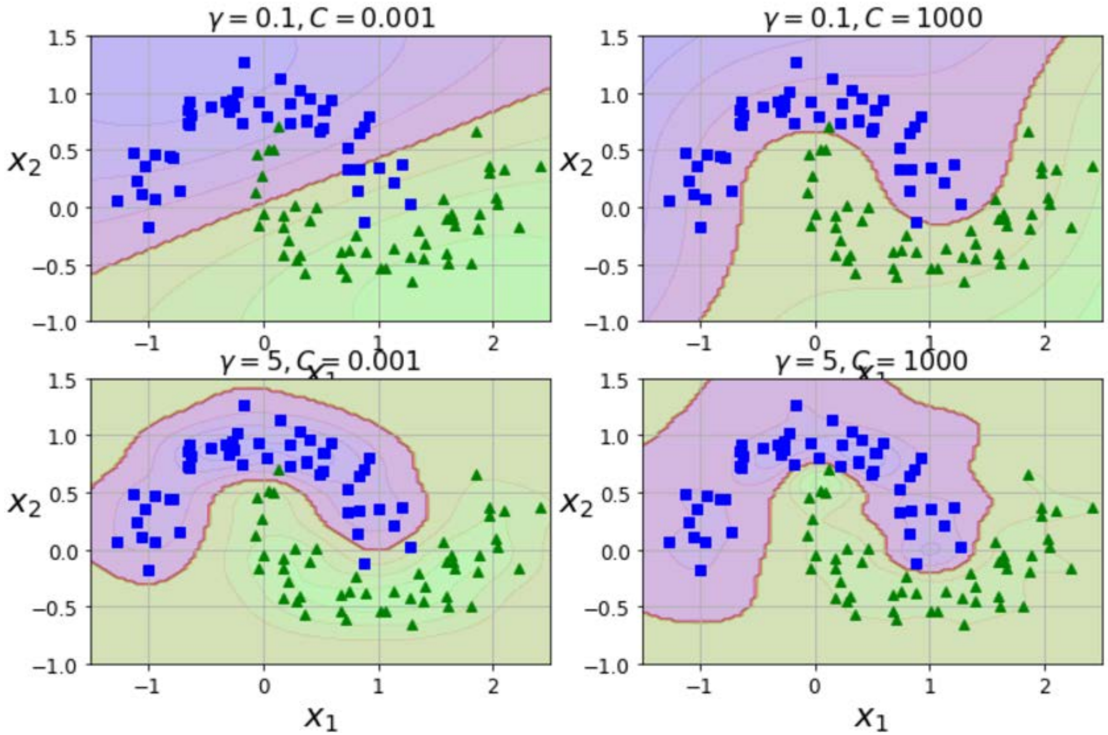
$$k(x, x') = (\langle x, x' \rangle + r)^d$$



Non-Linear Classification (cont'd)

- Gaussian Kernel

$$k(x, x') = \exp(-\gamma \left|\left| x - x' \right|\right|^2)$$



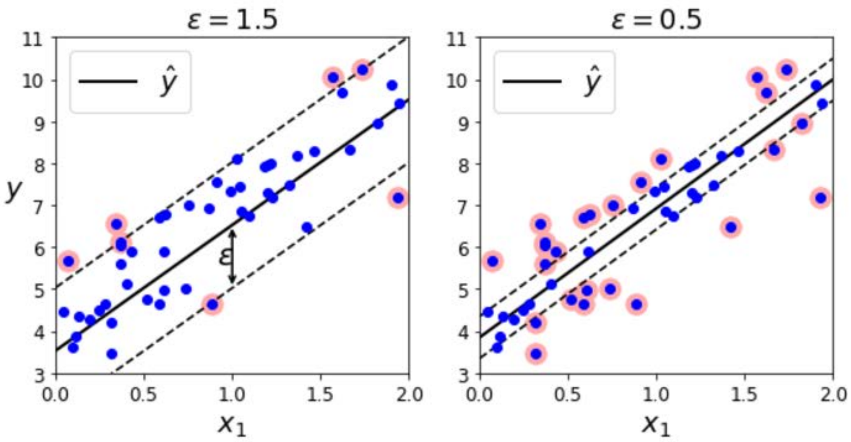
UNIVERSITY OF TORONTO
SCHOOL OF CONTINUING STUDIES

Module 6 – Section 4

SVM Regression

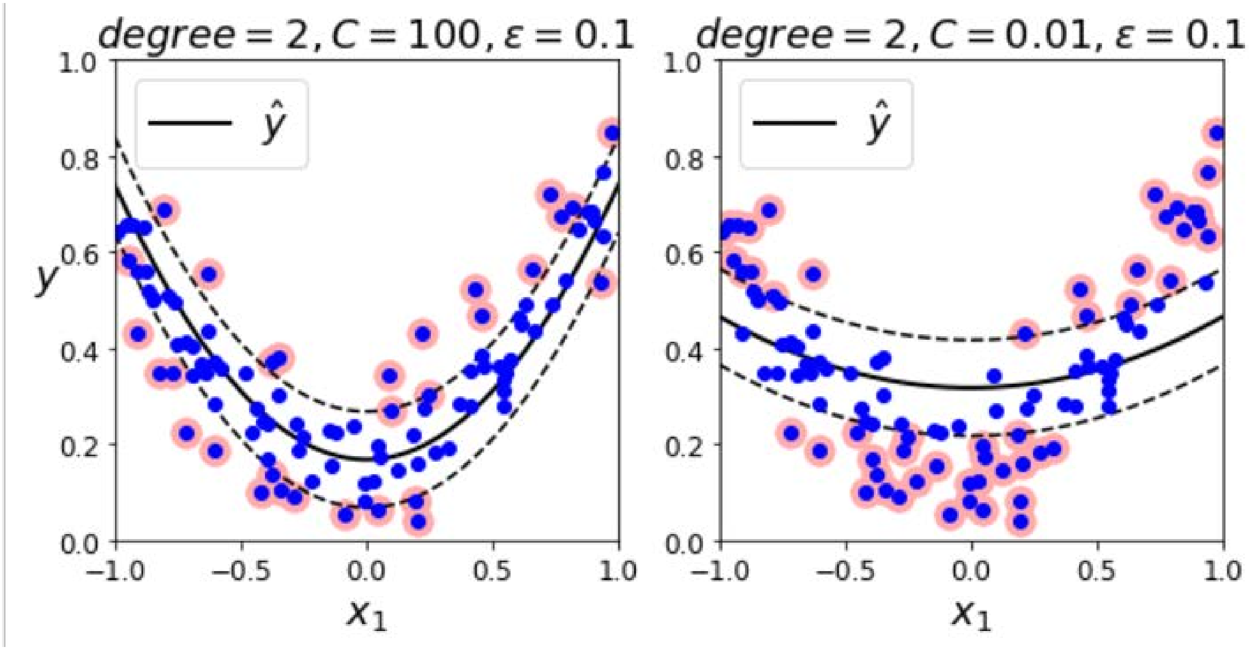
SVM Regression

- Classification goal: find parallel dashed lines as far as possible and containing few points between them
- Regression goal: find parallel dashed lines that are close to each other and contain as many points as possible
- Instead of C the regularization hyperparameter is ϵ



SVM Regression (cont'd)

- Non linear SVM Regression, same kernels
- E.g., polynomial kernels:



UNIVERSITY OF TORONTO
SCHOOL OF CONTINUING STUDIES

Module 6 – Section 5

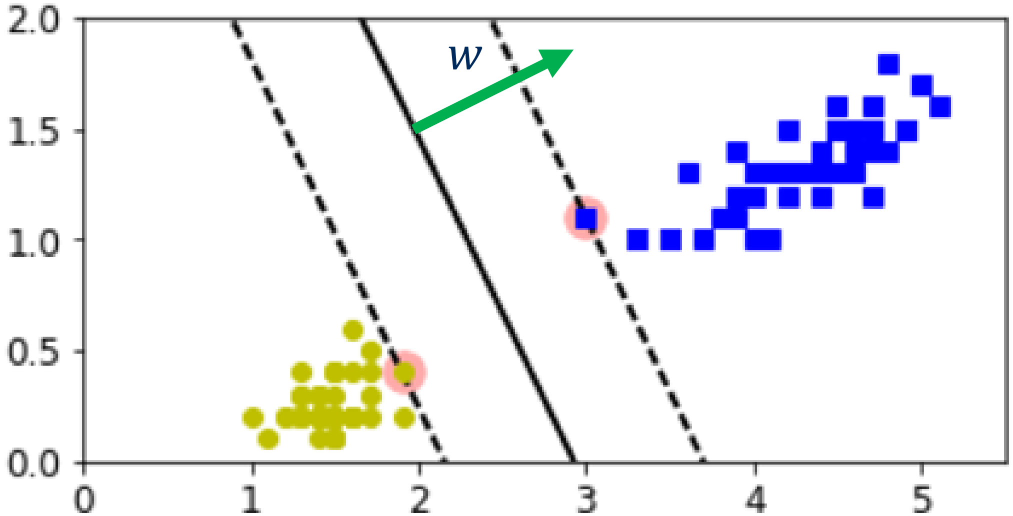
How SVM Works

How SVM works

- Linear SVM classifies points by:

$$y = \begin{cases} +1, & if\ w^T \cdot x + b > 0 \\ -1, & if\ w^T \cdot x + b < 0 \end{cases}$$

- The algorithm finds w $(\perp\ to\ line)$ and b



UNIVERSITY OF TORONTO
SCHOOL OF CONTINUING STUDIES

How SVM works (cont'd)

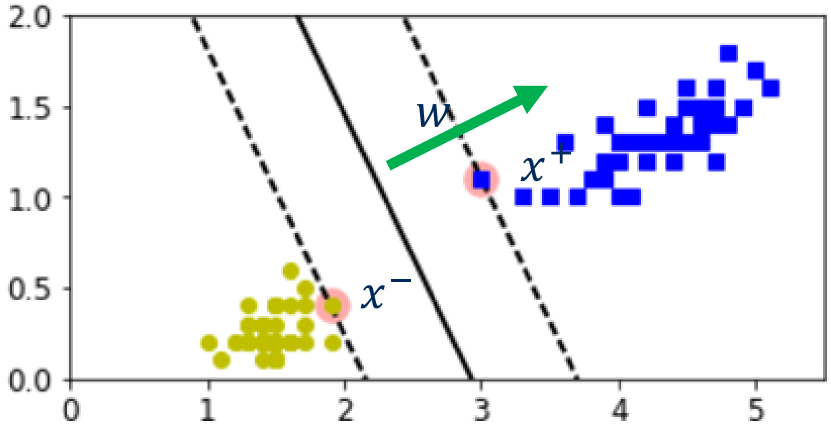
- Linear SVM classifies points by:

$$y = \begin{cases} +1, & if\ w^T \cdot x + b > 0 \\ -1, & if\ w^T \cdot x + b < 0 \end{cases}$$

- The algorithm finds w ($\perp\ to\ line$) and b

- Distance between margins: $\frac{w}{||w||}(x^+ - x^-) = \frac{2}{||w||}$

x^+, x^- are supports



How SVM works (cont'd)

- Optimization problem that yields w, b

$$\begin{cases} minimize: \dfrac{1}{2} w^T \cdot w \\ subject\ to: \ y^{(i)}\left(w^T \cdot x^{(i)} + b\right) \geq 1, for\ all\ i \end{cases}$$

- For soft margin: add slack variables

$$\begin{cases} minimize: \dfrac{1}{2} w^T \cdot w + C \displaystyle\sum_i \xi^{(i)} \\ subject\ to: y^{(i)}\left(w^T \cdot x^{(i)} + b\right) \geq 1 - \xi^{(i)} \end{cases}$$

How SVM works (cont'd)

- These formulations (*Primal*) are cases of Quadratic Programming

$$\begin{cases} minimize\ (variable\ u)\text{: } \dfrac{1}{2}u^T \cdot H \cdot u + f^T \cdot u \\ subject\ to\text{: } A \cdot u \leq d \end{cases}$$

where: $u, f \in R^{q\times 1}, H\ symmetrix\ q \times q, d \in R^{m\times 1}, A \in R^{m\times q}$

- With appropriate values for H, f, A, d, the primal formulation can be solved as QP problems (well studies)

How SVM works (cont'd)

- There is an equivalent formulation (same solution) to the Primal, called *Dual*

$$\begin{cases} minimize\ (variable\ \alpha): \sum_{i=1}^{m} \alpha_i y^{(i)} - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y^{(i)} y^{(j)} \langle x^{(i)}, x^{(j)} \rangle \\ subject\ to: \alpha_i \geq 0, \sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \end{cases}$$

- Once α is found, it can be used to obtain w and b from the Primal formulation, and build the classifier

$$b = \frac{1}{n_s} \sum\nolimits_{i:\alpha_i > 0} (1 - y^{(i)} \langle w, x^{(i)} \rangle)$$

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

How SVM works (cont'd)

- The Dual formulation is also applicable for the non linear case, using any kernel: replace $\langle _, _ \rangle$by k

$$\begin{cases} minimize\ (variable\ \alpha)\text{:} \sum_{i=1}^{m} \alpha_i y^{(i)} - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y^{(i)} y^{(j)} (k(x^{(i)}, x^{(j)})) \\ subject\ to\text{:} \alpha_i \geq 0, \sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \end{cases}$$

- Once α is found, b can be found, and the classifier can be computed $y = w^T \cdot \phi(x) + b$:

$$b = \frac{1}{n_s} \sum_{i:\alpha_i > 0} \left(1 - y^{(i)} \sum_{j=1}^{m} \alpha_j y^{(j)} k(x^{(j)}, x^{(i)}) \right)$$

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

UNIVERSITY OF TORONTO
SCHOOL OF CONTINUING STUDIES

Module 6 – Section 6

Resources and Wrap-up

Resources

- Hands-On Machine Learning with Scikit-Learn and Tensorflow

Assessment

- See Jupyter Notebook

Next Class

- Decision Trees

Follow us on social

Join the conversation with us online:

facebook.com/uoftscs

@uoftscs

linkedin.com/company/university-of-toronto-school-of-continuing-studies

@uoftscs

UNIVERSITY OF TORONTO
SCHOOL OF CONTINUING STUDIES

Any questions?

UNIVERSITY OF TORONTO
SCHOOL OF CONTINUING STUDIES

Thank You

Thank you for choosing the University of Toronto School of Continuing Studies