Introduction to Machine Learning

Foundations and Applications

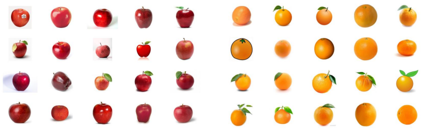
Paul J. Atzberger

University of California Santa Barbara

Support Vector Machines

Support Vector Machines: Motivations

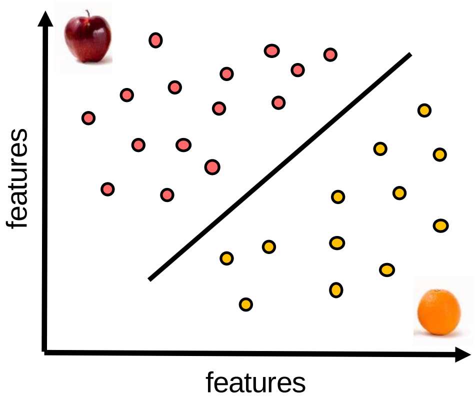
Image Database



Consider data: $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, yn)\}$, with features x, labels y.

Example: $x \in \mathbb{R}^N$, $y \in \{-1,+1\}$, with x=image, y = +1→ Apple, y = -1→ Orange.

Task: Find hyperplane that separates points x_i having different labels y_i.

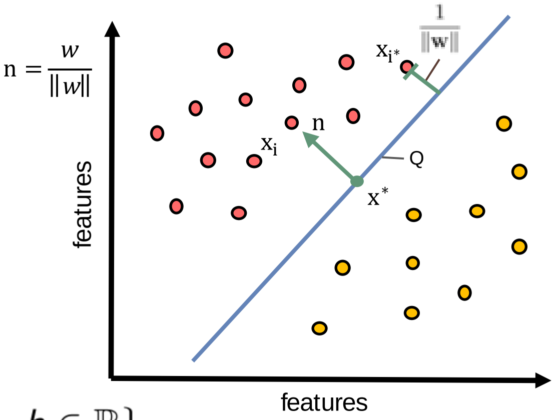


Challenges:

- **What algorithms** can be used to find hyperplanes from data?
- **Many hyperplanes are possible. Which may have the best generalization?**
- **What if the data is not separable?**
- **How do we precisely** define "separation" and the classification task?

Approach: Support Vector Machines + Kernel Methods.

Support Vector Machines



SVM: Optimization Problem (Primal $\mathcal{P}$)

$$\begin{cases} \min_{\mathrm{w},b} \frac{1}{2}\|\mathrm{w}\|^2 \\ \text{subject to } y_i(\mathrm{w}\cdot\mathrm{x}_i + b) \geq 1. \end{cases}$$

Separable Case: $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^m, \ \ \mathcal{H} = \{h \mid h(x) = \mathrm{sign}(w^T x + b), \ w \in \mathbb{R}^N, \ b \in \mathbb{R}\}.$

Definition: The data is **separable** if there exists $h \in \mathcal{H}$ so that $h(x_i) = \mathrm{sign}(w^T x_i + b) = y_i, \ i \in \{1, 2, \ldots, m\}$.

Hyperplane: $Q = \{x \mid w_0^T x + b_0 = 0\} = \{x \mid c^{-1} w_0^T x + c^{-1} b_0 = 0\}$.

Require: $\min_{x_i} |w^T x_i + b| = 1$ for the w used for a given data set $\mathcal{S}$.

Result: We will always have $y_i(w^T x + b) \geq 1$.

Definition: The **geometric margin** $\rho(x)$ of a point x is the distance to the hyperplane Q.

Let x^* be s.t. $-w^T x^* = b$, then $w^T x + b = w^T(x - x^*)$.

$$\Rightarrow \left| \frac{w}{\|w\|}(x - x^*) \right| = \frac{|w^T x + b|}{\|w\|} = \rho(x).$$

Consequence:, the closest data point x_{i^*} has distance $\rho(x_{i^*}) = 1/\|w\|$.

SVM Separable Case: Summary

Consider a data set $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, yn)\}$, where x denotes features and y denotes labels.

Example: $x \epsilon RN, y \epsilon \{-1, +1\}$,
with x=image, y = +1→ Apple, y = -1→ Orange.

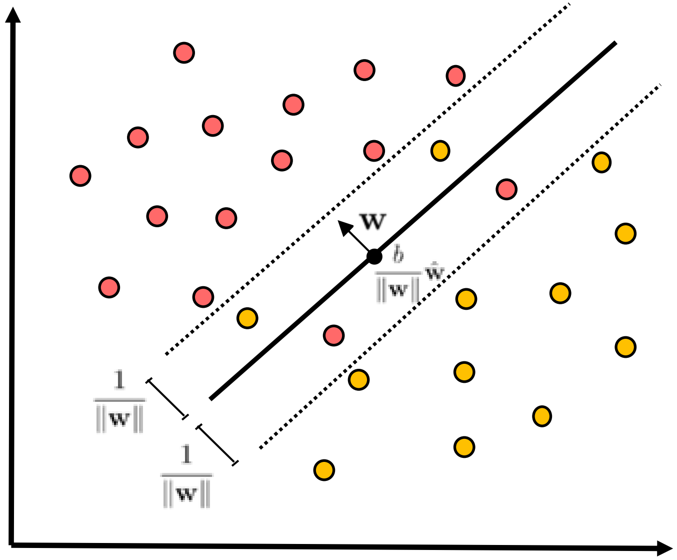
Find hyperplane separating points x_i having different labels y_i and with the **"greatest margin"** (helps with generalization).

Find parameters w, b that optimize

$$\min_{\mathbf{w},b} \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

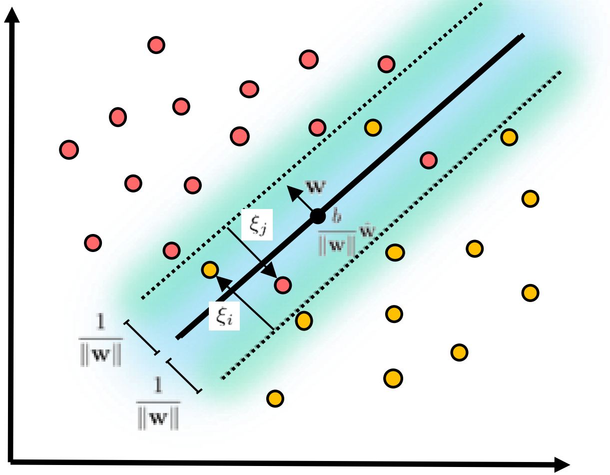
subject to $y_i\left(\mathbf{w}^T\phi(\mathbf{x}_i) + b\right) \geq 1$ (for now, $\phi(\mathbf{x}_i) = \mathbf{x}_i$)

This **assumes** data is **separable.** Minimizing w maximizes the margin. Classifier h(x) = sign $\left(\mathbf{w}^T\phi(\mathbf{x}) + b\right)$



What if data is not separable?

Summary: SVM Non-Separable Case



Case of data that is not separable?

Find hyperplane and with **biggest margin** that minimizes extent of misclassifications.

Introduce **"slack variables"** ξ for the constraint.

Find parameters w, b, ξ that optimize

$$\min_{\mathbf{w},b,\xi_i} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}\xi_i$$

subject to $y_i\left(\mathbf{w}^T\phi(\mathbf{x}_i) + b\right) \geq 1 - \xi_i \quad \xi_i \geq 0$ (for now, $\phi(\mathbf{x}_i) = \mathbf{x}_i$)

Tries to find hyperplane and margin that minimizes the total amount training data points violate the constraint.

C is crucial regularization parameter determining penalty for violating the constraint.

Insights into generalization using results from optimization (duality).

Can apply more generally using kernel methods.

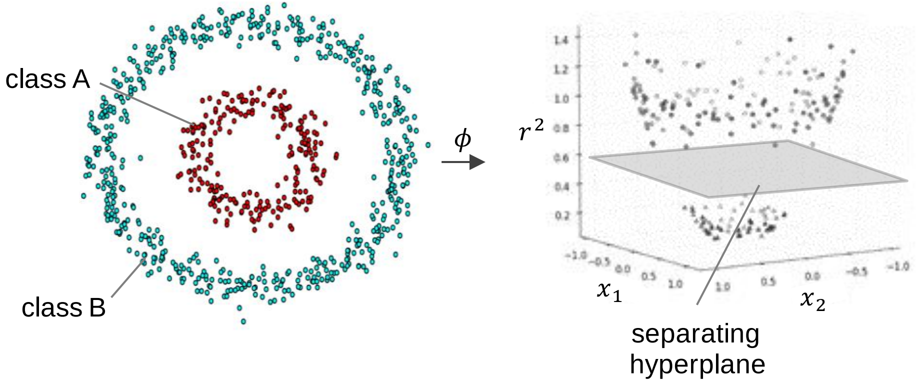
Kernel Methods Overview

Support Vector Machines (Kernel Trick)

Data is often not separable.

Mapping points to higher dimensional spaces they can become separable.

Example:



map ϕ

$$\tilde{x} = \phi(x) = (x_1, x_2, r^2)$$
$$r^2 = x_1^2 + x_2^2$$

Kernel: associated to map ϕ is an inner-product $K(x_i, xj) = \langle \phi(x_i), \phi(x_j) \rangle$.

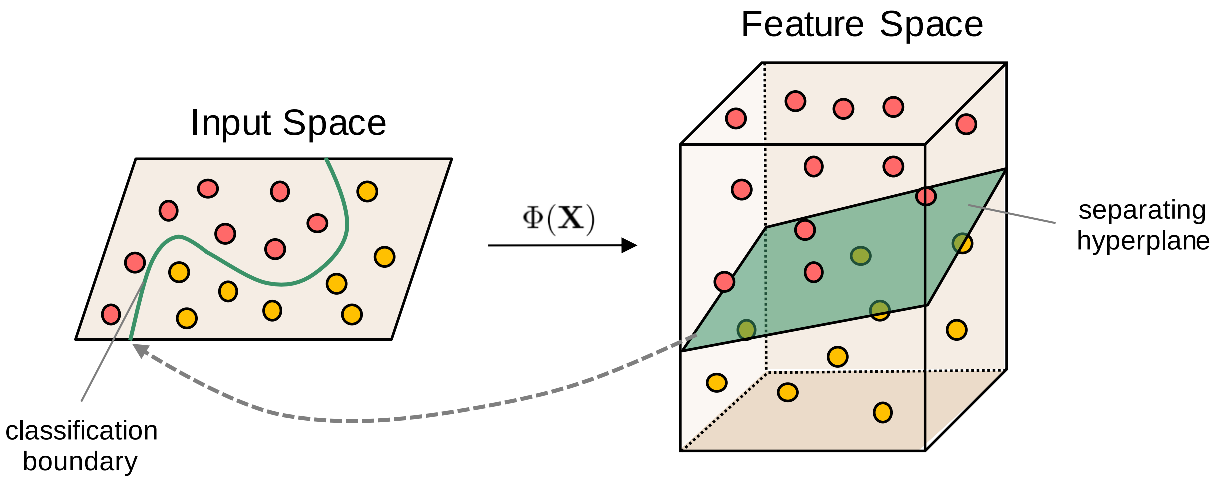
Example: $K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle = x_{i,1}x_{j,1} + x_{i,2}x_{j,2} + r_i^2 r_j^2$

More generally…

Support Vector Machines (Kernel Trick)

Data is often not separable.

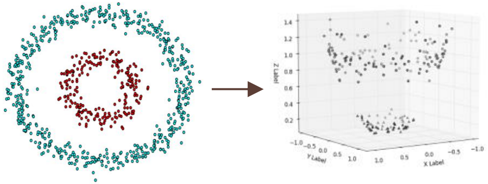
Mapping points to higher dimensional spaces they can become separable.



Kernel: associated to map ϕ is an inner-product $K(x_i, xj) = \langle \phi(x_i), \phi(x_j) \rangle$.

Example: $K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle = x_{i,1}x_{j,1} + x_{i,2}x_{j,2} + r_i^2 r_j^2$

Support Vector Machines (Kernel Trick)



Data is often not separable but can become **separable in higher dimensional spaces.**

Inner-products can be replaced by **kernel** $\langle x_i, x_j \rangle \rightarrow K(x_i, xj)$.

Kernel should be **symmetric, positive definite**, L^2: $\sum_{i,j=1}^{N} a_i a_j K(x_i, x_j) \geq 0$, $\int_X \int_X K(x,t)^2 d\mu(x) d\mu(t) < \infty$

Consider: $L_K : L^2_\mu(X) \rightarrow L^2_\mu(X)$, $L_K f(x) = \int_X K(x,t) f(t) d\mu(t)$ has countable set of non-negative eigenvalues $\{\lambda_k\}_{k=1}^{\infty}$.

Theorem (Mercer 1909): An L^2 kernel K(x,t) that is symmetric positive definite can be represented as the product

$$K(x,t) = \sum_{k=1}^{\infty} \lambda_k \phi_k(x) \phi_k(t)$$

Let $\mathbf{\Phi}(x) = [\Phi_k(x)]$, then $K(x,t) = <\mathbf{\Phi}(x), \mathbf{\Phi}(t)>$.

Consequence: $K(x_i, xj) = <\mathbf{\Phi}(xi), \mathbf{\Phi}(x_j)>$, so $k(v, x_i) = w^T \mathbf{\Phi}(x_i)$ as appears in the SVM constraints.

Note, only action K(x,y) is needed in SVM, so no need to map explicitly to feature space $\mathbf{\Phi}$(x).

Kernel Methods: Hilbert-Schmidt and Mercer Theorem

Theorem (Hilbert-Schmidt): For L_K a **self-adjoint compact operator** there is a **countable complete orthonormal basis** $\{\phi_i\}$ for $L^2_\mu(\mathcal{Z})$ so that $L_K\phi_i = \lambda_i\phi_i$ with $\lambda_i \to 0$.

Theorem (Mercer 1909): An **L^2-kernel** K(x,t) that is **symmetric positive definite** can be represented as the product

$$K(x,t) = \sum_{k=1}^{\infty} \lambda_k \phi_k(x)\phi_k(t) \quad \text{with } \lambda_i > 0, \lambda_1 \geq \lambda_2 \geq \lambda_3 \ldots, \text{and } \lambda_i \to 0.$$

Remark: We can interpret the **Mercer Theorem** as stating **there exists a non-linear transformation** $z = \Phi(x)$ related to the kernel **K** as follows. Let $[\Phi(x)]_k = \sqrt{\lambda_k}\Phi_k(x)$ then $K(x,t) = \langle \mathbf{\Phi}(x), \mathbf{\Phi}(t)\rangle_{\ell^2}$. Also can represent using Reproducing Kernel Hilbert Space (RKHS) (later).

Consequence: This shows that **if a kernel is L^2 and symmetric positive definite** then we can interpret it as being the **inner-product** associated with **some non-linear transformation Φ of the data!** For instance, $\mathbb{R}^N \to \ell^2$ or later $\mathbb{R}^N \to RKHS$.

Remark: To compute the inner-product we do **not** need to use $\langle \mathbf{\Phi}(x), \mathbf{\Phi}(t)\rangle_{\ell^2}$ which could be expensive, instead **we only need to evaluate kernel** $K(x,t)$. **This called the kernel trick!**

Support Vector Machines (Kernel Trick)

Kernels provide sensitivity to different features of the data, K(x,t) = <**Φ**(x),**Φ**(t)>.

Popular Kernels:

Linear:

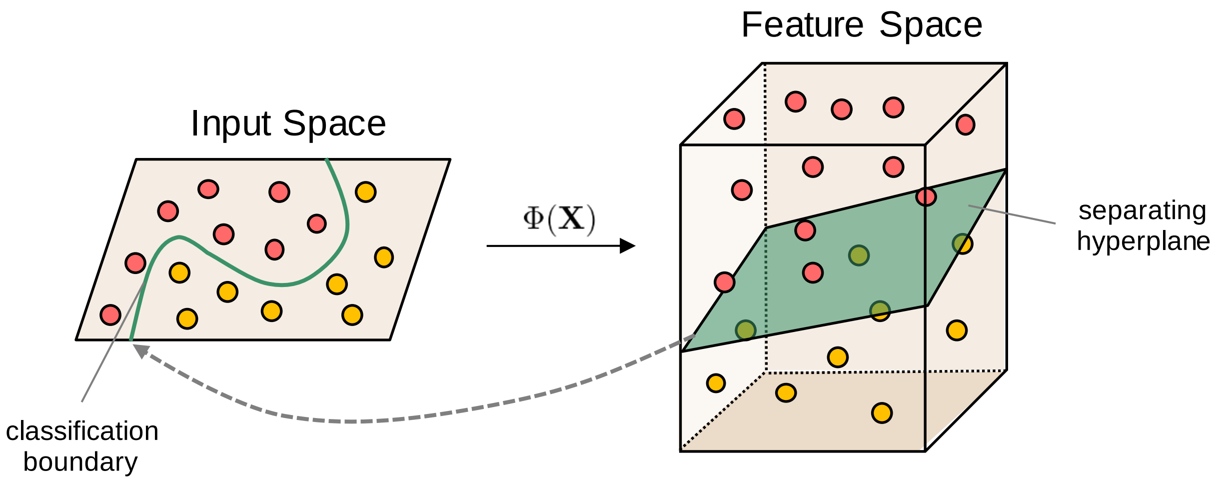
$$K(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}, \mathbf{y} \rangle$$

Radial Basis Function (RBF):

$$K(\mathbf{x}, \mathbf{y}) = \exp\left[-\gamma \|\mathbf{x} - \mathbf{y}\|^2\right]$$

Polynomial (degree d):

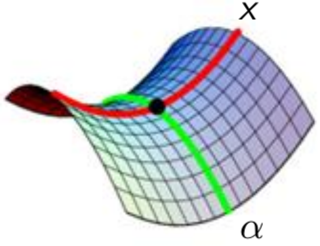
$$K(\mathbf{x}, \mathbf{y}) = (\gamma \langle \mathbf{x}, \mathbf{y} \rangle + r)^d$$



Lots of other choices possible.

Optimization Theory

Optimization



Constrained Optimization Problem (Primal $\mathcal{P}$)

$$\begin{cases} \min_{x\in\mathcal{X}} f(x) \\ \text{subject to } g_i(x) \leq 0. \end{cases}$$

Definition: The **Lagrangian** $\mathcal{L}$ of $\mathcal{P}$ is

$$\mathcal{L}(x,\alpha) = f(x) + \sum_{i=1}^{m} \alpha_i g_i(x) \;\rightarrow\; \mathcal{L}(x,\alpha) = f(x) + \boldsymbol{\alpha}\cdot \mathrm{g}(x), \;\; \text{where } \boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_M \end{bmatrix}, \;\; \mathrm{g}(\mathrm{x}) = \begin{bmatrix} g_1(\mathrm{x}) \\ \vdots \\ g_M(\mathrm{x}) \end{bmatrix}.$$

Definition: A saddle-point $(x^*, \boldsymbol{\alpha}^*)$ of the Lagrangian $\mathcal{L}$ is a point satisfying
$\mathcal{L}(\mathrm{x}^*, \boldsymbol{\alpha}) \leq \mathcal{L}(\mathrm{x}^*, \boldsymbol{\alpha}^*) \leq \mathcal{L}(\mathrm{x}, \boldsymbol{\alpha}^*)$, holding for $\forall \mathrm{x} \in \mathcal{X}$, $\boldsymbol{\alpha} \geq 0$.

Theorem

For **constrained optimization problem** $\mathcal{P}$, a **saddle-point** $(x^*, \boldsymbol{\alpha}^*)$ of the Lagrangian $\mathcal{L}$ is a solution of $\mathcal{P}$.

Optimization

Constrained Optimization Problem (Primal $\mathcal{P}$)

$$\begin{cases} \min_{x \in \mathcal{X}} f(x) \\ \text{subject to } g_i(x) \leq 0. \end{cases}$$

Lagrangian

$$\mathcal{L}(x, \alpha) = f(x) + \boldsymbol{\alpha} \cdot \mathrm{g}(x)$$

Theorem

For **constrained optimization problem** $\mathcal{P}$, a **saddle-point** $(x^*, \boldsymbol{\alpha}^*)$ of the Lagrangian $\mathcal{L}$ is a solution of $\mathcal{P}$.

Proof: From $\mathcal{L}(\mathrm{x}^*, \boldsymbol{\alpha}) \leq \mathcal{L}(\mathrm{x}^*, \boldsymbol{\alpha}^*) \Rightarrow \boldsymbol{\alpha} \cdot \mathrm{g}(\mathrm{x}^*) \leq \boldsymbol{\alpha}^* \cdot \mathrm{g}(\mathrm{x}^*), \forall \boldsymbol{\alpha} \geq 0, \Rightarrow \mathrm{g}(\mathrm{x}^*) \leq 0.$

If $g_i(\mathrm{x}^*) > 0$ then take α_i large so that $\alpha_i \cdot g_i(\mathrm{x}^*) > c_i$ for any given $c_i \in \mathbb{R}$. Let $c = \boldsymbol{\alpha}^* \cdot \mathrm{g}(\mathrm{x}^*)$.

Furthermore, $\boldsymbol{\alpha}^* \cdot \mathrm{g}(\mathrm{x}^*) = 0$. Consider $\boldsymbol{\alpha} \to 0$ then $0 \leq \boldsymbol{\alpha}^* \cdot \mathrm{g}(\mathrm{x}^*) \leq 0, \Rightarrow \boldsymbol{\alpha}^* \cdot \mathrm{g}(\mathrm{x}^*) = 0$.

From $\mathcal{L}(\mathrm{x}^*, \boldsymbol{\alpha}^*) \leq \mathcal{L}(\mathrm{x}, \boldsymbol{\alpha}^*), \ \forall \mathrm{x} \Rightarrow f(\mathrm{x}^*) \leq f(\mathrm{x}) + \boldsymbol{\alpha}^* \cdot \mathrm{g}(\mathrm{x})$ for all x s.t. $\mathrm{g}(\mathrm{x}) \leq 0$.

We have $f(\mathrm{x}^*) \leq f(\mathrm{x})$ so $(\mathrm{x}^*, \boldsymbol{\alpha}^*)$ solves $\mathcal{P}$. $\blacksquare$.

Optimization

Definition: **Strong Constraint Qualification (Slater's Condition)**
$\exists \bar{x} \in \text{interior}(\mathcal{X}), \quad \forall i \in \{1, 2, \ldots, m\}, \quad g_i(\bar{x}) < 0.$

Definition: **Weak Constraint Qualification (Weak Slater's Condition)**
$\exists \bar{x} \in \text{interior}(\mathcal{X}), \quad \forall i \in \{1, 2, \ldots, m\}, \quad (g_i(\bar{x}) < 0) \vee (g_i(\bar{x}) = 0 \wedge g_i(\bar{x}) = a\bar{x} + b \text{ (affine)}).$

Theorem: (when saddle point is necessary w/ strong slater)

Let f, g be **convex functions** with **strong slater condition** holding.
If x^* is a solution to $\mathcal{P}$ then $\exists \boldsymbol{\alpha}^* \geq 0$ s.t. $(\mathrm{x}^*, \boldsymbol{\alpha}^*)$ satisfies the saddle condition for $\mathcal{L}$.

Theorem: (when saddle point is necessary w/ weak slater)

Let f, g be **convex** and **differentiable** functions with **weak slater condition** holding.
If x^* is a solution to $\mathcal{P}$ then $\exists \boldsymbol{\alpha}^* \geq 0$ s.t. $(\mathrm{x}^*, \boldsymbol{\alpha}^*)$ satisfies the saddle condition for $\mathcal{L}$.

Optimization:

Theorem: Karuch-Kuhn-Tucker (KKT) Conditions

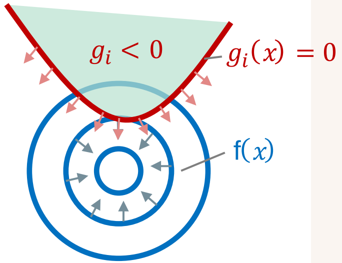
Let f, g_i be **convex** and **differentiable** functions where the **weak constraint qualification** is satisfied.

A $\bar{x}$ is a solution to the **constrained optimization problem** $\mathcal{P}$ if and only if $\exists \bar{\alpha} \geq 0$ s.t.

$$\nabla_x \mathcal{L}(\bar{x}, \bar{\boldsymbol{\alpha}}) = \nabla_x f + \bar{\boldsymbol{\alpha}} \cdot \nabla_x g(\bar{x}) = 0.$$

$$\nabla_{\boldsymbol{\alpha}} \mathcal{L}(\bar{x}, \bar{\boldsymbol{\alpha}}) = g(\bar{x}) \leq 0.$$

$$\bar{\boldsymbol{\alpha}} \cdot g(\bar{x}) = \sum_{i=1}^{m} \bar{\alpha}_i g_i(\bar{x}) = 0.$$

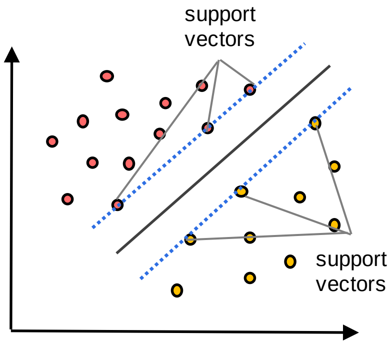


SVM Separable Case (KKT): $\mathcal{L}(w, b, \boldsymbol{\alpha}) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i \left[y_i \left(w \cdot x_i + b\right) - 1\right].$

$$\nabla_w \mathcal{L} = w - \sum_{i=1}^{m} \alpha_i y_i x_i = 0, \quad \Rightarrow \quad w = \sum_{i=1}^{m} \alpha_i y_i x_i.$$

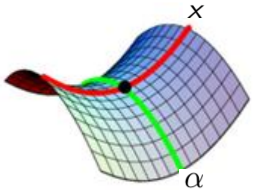
$$\nabla_b \mathcal{L} = -\sum_{i=1}^{m} \alpha_i y_i = 0, \quad \Rightarrow \quad \sum_{i=1}^{m} \alpha_i y_i = 0$$

$$\forall i, \alpha_i \left[y_i \left(w \cdot x_i + b\right) - 1\right] = 0, \quad \Rightarrow \quad \alpha_i = 0 \vee y_i \left(w \cdot x_i + b\right) = 1.$$



SVM
Dual Formulations

Optimization:



Definition: Dual Function
$\forall \boldsymbol{\alpha} \geq 0, \quad F(\boldsymbol{\alpha}) = \inf_{x \in \mathcal{X}} \mathcal{L}(\mathrm{x}, \boldsymbol{\alpha}).$

Definition: Dual Optimization Problem $\mathcal{P}^*$

$$\begin{cases} \max_{\boldsymbol{\alpha} \in \mathbb{R}^M} F(\boldsymbol{\alpha}) \\ \text{subject to } \alpha_i \geq 0. \end{cases}$$

SVM Separable Case (Dual Form): $\mathcal{L}(\mathrm{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathrm{w}\|^2 - \sum_{i=1}^m \alpha_i \left[y_i \left(\mathrm{w} \cdot \mathrm{x}_i + b\right) - 1\right].$

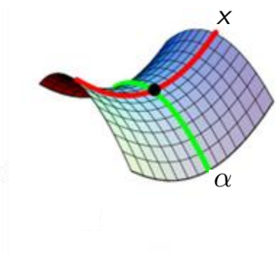
From KKT we have $\mathrm{w} = \sum_{i=1}^m \alpha_i y_i \mathrm{x}_i$ and $\sum_{i=1}^m \alpha_i y_i = 0$. This gives

Dual Function: $F(\boldsymbol{\alpha}) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m \alpha_i \alpha_j y_i y_j \left(\mathrm{x}_i \cdot \mathrm{x}_j\right)$

Dual Optimization Problem $\mathcal{P}^*$:

$$\begin{cases} \max_{\boldsymbol{\alpha}} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m \alpha_i \alpha_j y_i y_j \left(\mathrm{x}_i \cdot \mathrm{x}_j\right) \\ \text{subject to } \alpha_i \geq 0 \wedge \sum_{i=1}^m \alpha_i y_i = 0, \quad \forall i \in \{1, 2, \ldots, m\} \end{cases}$$

SVM Dual Formulation



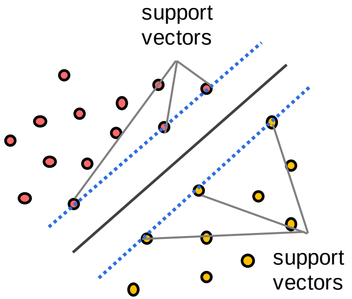
SVN Non-Separable Case (Primal):

$$\begin{cases} \min\limits_{\mathbf{w},b,\boldsymbol{\xi}} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m} \xi_i^p \\ \text{subject to} \quad y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 - \xi_i \;\wedge\; \xi_i \geq 0, i \in [1,m] \end{cases}$$

Lagrangian:

$$\mathcal{L}(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta}) = \underbrace{\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i}_{\text{"}f(x)\text{"}} \underbrace{- \sum_{i=1}^{m}\alpha_i[y_i(\mathbf{w}\cdot\mathbf{x}_i + b) - 1 + \xi_i] - \sum_{i=1}^{m}\beta_i\xi_i}_{\text{"}\alpha\cdot g(x)\text{"}}$$

$\mathrm{x} = (\mathrm{w}, b, \xi)$



KKT Conditions:

$$\nabla_{\mathbf{w}}\mathcal{L} = \mathbf{w} - \sum_{i=1}^{m}\alpha_i y_i \mathbf{x}_i = 0 \implies \mathbf{w} = \sum_{i=1}^{m}\alpha_i y_i \mathbf{x}_i$$

$$\nabla_b\mathcal{L} = -\sum_{i=1}^{m}\alpha_i y_i = 0 \implies \sum_{i=1}^{m}\alpha_i y_i = 0$$

$$\nabla_{\xi_i}\mathcal{L} = C - \alpha_i - \beta_i = 0 \implies \alpha_i + \beta_i = C \quad \to \alpha_i \leq C \;\; \beta_i \leq C$$

$$\forall i, \alpha_i[y_i(\mathbf{w}\cdot\mathbf{x}_i + b) - 1 + \xi_i] = 0 \implies \alpha_i = 0 \vee y_i(\mathbf{w}\cdot\mathbf{x}_i + b) = 1 - \xi_i$$

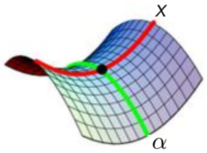
$$\forall i, \beta_i\xi_i = 0 \implies \beta_i = 0 \vee \xi_i = 0\,.$$

Dual Function F(α,β):

$$\mathcal{L} = \underbrace{\frac{1}{2}\|\sum_{i=1}^{m}\alpha_i y_i \mathbf{x}_i\|^2 - \sum_{i,j=1}^{m}\alpha_i\alpha_j y_i y_j(\mathbf{x}_i\cdot\mathbf{x}_j)}_{-\frac{1}{2}\sum_{i,j=1}^{m}\alpha_i\alpha_j y_i y_j(\mathbf{x}_i\cdot\mathbf{x}_j)} \underbrace{- \sum_{i=1}^{m}\alpha_i y_i b}_{0} + \sum_{i=1}^{m}\alpha_i \longrightarrow \mathcal{L} = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m}\alpha_i\alpha_j y_i y_j(\mathbf{x}_i\cdot\mathbf{x}_j)$$

same as separable case!

SVM Dual Formulation



SVM Non-Separable Case:

KKT Conditions:

$$\nabla_{\mathbf{w}}\mathcal{L} = \mathbf{w} - \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i = 0 \implies \mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i$$

$$\nabla_b \mathcal{L} = -\sum_{i=1}^{m} \alpha_i y_i = 0 \implies \sum_{i=1}^{m} \alpha_i y_i = 0$$

$$\nabla_{\xi_i}\mathcal{L} = C - \alpha_i - \beta_i = 0 \implies \alpha_i + \beta_i = C$$

$$\forall i, \alpha_i[y_i(\mathbf{w}\cdot\mathbf{x}_i + b) - 1 + \xi_i] = 0 \implies \alpha_i = 0 \vee y_i(\mathbf{w}\cdot\mathbf{x}_i + b) = 1 - \xi_i$$

$$\forall i, \beta_i \xi_i = 0 \implies \beta_i = 0 \vee \xi_i = 0\,.$$

Dual Function F(α, β): $\mathcal{L} = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} \alpha_i\alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j)$

SVM Non-Separable Case (Dual $\mathscr{P}$*):

$$\begin{cases} \max_{\boldsymbol{\alpha}} \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} \alpha_i\alpha_j y_i y_j \underbrace{(\mathbf{x}_i \cdot \mathbf{x}_j)}_{\text{kernel here}} \\ \text{subject to: } 0 \le \alpha_i \le C \wedge \sum_{i=1}^{m} \alpha_i y_i = 0, i \in [1, m]. \end{cases}$$

Inner-products of $(\mathbf{x}_i \cdot \mathbf{x}_j)$ only appear. **Kernel Method:** $\tilde{x} = \phi(x)$ holds for $\tilde{x}_i \cdot \hat{x}_j = \phi(x_i)\cdot\phi(x_j) = k(x_i, x_j)$.

Dimension of dual problem is **m. Primal problem** has dimension N.

Regularization C in primal problem $\mathscr{P}$ becomes constraint in dual problem $\mathscr{P}^*$.

Provides alternative ways to solve the optimization problem.

Generalization Error Bounds for Support Vector Machines

VC-Dimension: Hyperplanes

Example: Learning separating hyperplane in $\mathbb{R}^N$ (related to SVM). For data $\{(x_i,y_i)\}$ with $x_i \in \mathbb{R}^N$ and $y_i \in \{-1,1\}$. Ideally, find **w**, b so that $\text{sign}(\boldsymbol{w}^T \boldsymbol{x}_i + b) = y_i$.

Hypothesis class:

$\mathcal{H} = \{h: h(\mathbf{x}) = \text{sign}(\mathbf{w}^T\mathbf{x} + b)$ with $\boldsymbol{w} \in \mathbb{R}^N$, $b \in \mathbb{R}\}$.

What is the $VCdim(\mathcal{H})$?

Claim: $VCdim(\mathcal{H}) = N + 1$

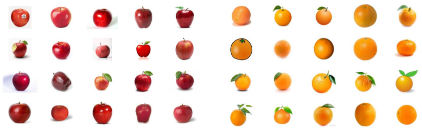
In separable case we have bound on generalization error (pr > 1 - δ)

$$R(h) \leq \widehat{R}(h) + \sqrt{\frac{2(N+1)\log\frac{em}{N+1}}{m}} + \sqrt{\frac{\log\frac{1}{\delta}}{2m}}$$

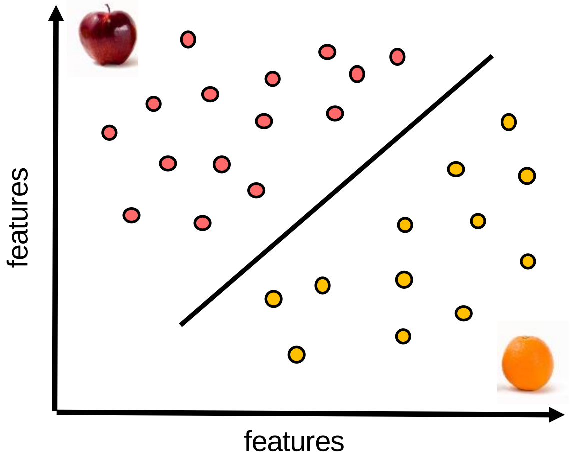
We can do even better in bounding sampling complexity using special structure of SVM.

We want bounds independent of **feature dimension N** so can handle large N or even N= ∞.

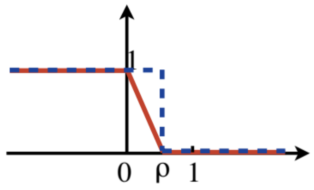
Image Database



Linear Classifier



SVM: Generalization Error



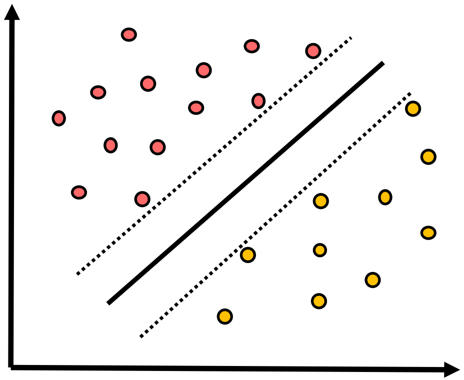
Definition: The **geometric margin** of a data point **x** is

$$\rho(x) = \frac{y\,(\mathbf{w}\cdot\mathbf{x} + b)}{\|\mathbf{w}\|}$$

$y \in \{-1,1\}$ depending on side of hyperplane

Definition: The **margin** of linear classifier h(x) = sign(w · x + b) for data set $\mathcal{S} = (x_1, x_2, \ldots, x_m)$ is

$$\rho = \min_{1\leq i\leq m} \frac{y_i\,(\mathbf{w}\cdot\mathbf{x}_i + b)}{\|\mathbf{w}\|}$$



Definition: We define a **marginal loss function** using

$$\Phi_\rho(x) = \begin{cases} 0 & \text{if } \rho \leq x \\ 1 - x/\rho & \text{if } 0 \leq x \leq \rho \\ 1 & \text{if } x \leq 0\,. \end{cases}$$

Definition: We define **empirical marginal loss** as

$$\widehat{R}_\rho(h) = \frac{1}{m}\sum_{i=1}^{m} \Phi_\rho(y_i h(x_i))$$

SVM: Generalization Error

Theorem (Margin bound for binary classification): For any fixed $\rho > 0$ and $\delta > 0$, we have with probability $1-\delta$ that the generalization error for marginal loss function is

$$R(h) \leq \widehat{R}_\rho(h) + \frac{2}{\rho}\mathfrak{R}_m(H) + \sqrt{\frac{\log \frac{1}{\delta}}{2m}}$$

$$R(h) \leq \widehat{R}_\rho(h) + \frac{2}{\rho}\widehat{\mathfrak{R}}_S(H) + 3\sqrt{\frac{\log \frac{2}{\delta}}{2m}}$$

Key idea: obtain bounds using the Rademacher Complexity of $\mathcal{H} = \{h : h(\mathbf{x}) = \mathrm{sign}(\mathbf{w}^T\mathbf{x} + b) \text{ with } \boldsymbol{w} \in \mathbb{R}^N, b \in \mathbb{R}\}$.

Notational convention: Suppress the b term by using $\tilde{\boldsymbol{x}} = \begin{bmatrix} x \\ 1 \end{bmatrix}, \tilde{w} = \begin{bmatrix} w \\ b \end{bmatrix}$, $\mathcal{H} = \{h : h(\tilde{\boldsymbol{x}}) = \mathrm{sign}(\tilde{\boldsymbol{w}}^T\tilde{\boldsymbol{x}}) \text{ with } \tilde{\boldsymbol{w}} \in \mathbb{R}^{N+1}\}$.

SVM: Generalization Error

Theorem: (Radamacher Complexity of Constrained Hyperplanes for Bounded Data $\mathcal{S}$)

Let $\mathcal{S} \subseteq \{\mathrm{x} : \|\mathrm{x}\| \leq r\}$ be a sample of size m and let $\mathcal{H} = \{h \mid h(\mathrm{x}) = \mathrm{sign}(\mathrm{w} \cdot \mathrm{x}) \mid \|\mathrm{w}\| \leq \Lambda\}$.
The Rademacher complexity satisfies
$\hat{R}_{\mathcal{S}}(\mathcal{H}) \leq \sqrt{\frac{r^2\Lambda^2}{m}}$.

Proof:

$$\begin{aligned}
\hat{R}_{\mathcal{S}}(\mathcal{H}) &= \frac{1}{m} E_{\sigma}\left[\sup_{\|\mathrm{w}\|\leq\Lambda} \sum_{i=1}^{m} \sigma_i \mathrm{w} \cdot \mathrm{x}_i\right] = \frac{1}{m} E_{\sigma}\left[\sup_{\|\mathrm{w}\|\leq\Lambda} \mathrm{w} \cdot \sum_{i=1}^{m} \sigma_i \mathrm{x}_i\right] \\
&\leq \frac{\Lambda}{m} E_{\sigma}\left[\left\|\sum_{i=1}^{m} \sigma_i \mathrm{x}_i\right\|\right] \leq \frac{\Lambda}{m} E_{\sigma}\left[\left\|\sum_{i=1}^{m} \sigma_i \mathrm{x}_i\right\|^2\right]^{1/2} \\
&\leq \frac{\Lambda}{m} E_{\sigma}\left[\sum_{i,j=1}^{m} \sigma_i \sigma_j \mathrm{x}_i \cdot \mathrm{x}_j\right]^{1/2} = \frac{\Lambda}{m}\left[\sum_{i=1}^{m} \|\mathrm{x}_i\|^2\right]^{1/2} \leq \frac{\Lambda\sqrt{mr^2}}{m} = \sqrt{\frac{r^2\Lambda^2}{m}} \quad \blacksquare.
\end{aligned}$$

Cauchy-Swartz Lemma:
$\mathrm{a} \cdot \mathrm{b} \leq \|\mathrm{a}\|\|\mathrm{b}\|$.

Jensen Inequality:
$\phi(E[X]) \leq E[\phi(X)]$

$(E[X])^2 \leq E[X^2]$

Radamacher Random Variables:
$E[\sigma_i\sigma_j] = E[\sigma_i]\,E[\sigma_j] = 0$.

Talagrand's Lemma:

Let $\Phi : \mathbb{R} \to \mathbb{R}$ be an ℓ-Lipschitz function $\|\Phi(\mathrm{x}) - \Phi(\mathrm{y})\| \leq \ell|\mathrm{x} - \mathrm{y}|$, then
$\hat{R}_{\mathcal{S}}(\Phi \circ \mathcal{H}) \leq \ell\hat{R}_{\mathcal{S}}(\mathcal{H})$.

SVM: Generalization Error

Theorem (Margin bound for binary classification): For any $\rho > 0$ and $\delta > 0$, we have with probability $1-\delta$ that the generalization error for marginal loss function is

$$R(h) \leq \widehat{R}_\rho(h) + \frac{2}{\rho}\mathfrak{R}_m(H) + \sqrt{\frac{\log\frac{1}{\delta}}{2m}}$$

$$R(h) \leq \widehat{R}_\rho(h) + \frac{2}{\rho}\widehat{\mathfrak{R}}_S(H) + 3\sqrt{\frac{\log\frac{2}{\delta}}{2m}}$$

Key idea: obtain bounds using the Rademacher Complexity of $\mathcal{H} = \{h : h(\mathbf{x}) = \mathrm{sign}(\mathbf{w}^T\mathbf{x} + b)$ with $\boldsymbol{w} \in \mathbb{R}^N, b \in \mathbb{R}\}$.

Notational convention: Suppress the b term by using $\tilde{\boldsymbol{x}} = \begin{bmatrix} x \\ 1 \end{bmatrix}, \tilde{w} = \begin{bmatrix} w \\ b \end{bmatrix}$, $\mathcal{H} = \{h : h(\tilde{\boldsymbol{x}}) = \mathrm{sign}(\tilde{\boldsymbol{w}}^T\tilde{\boldsymbol{x}})$ with $\tilde{\boldsymbol{w}} \in \mathbb{R}^{N+1}\}$.

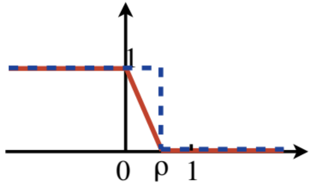
SVM: Generalization Error

Theorem (Margin bound for binary classification): For any fixed $\rho > 0$ and δ > 0, we have with probability 1- δ that the generalization error for marginal loss function is

$$R(h) \leq \widehat{R}_\rho(h) + 2\sqrt{\frac{r^2\Lambda^2/\rho^2}{m}} + \sqrt{\frac{\log\frac{1}{\delta}}{2m}}$$

$$\mathcal{H} = \{h \mid h(\mathrm{x}) = \mathrm{sign}(\mathrm{w}\cdot\mathrm{x}) \mid \|\mathrm{w}\| \leq \Lambda\}$$

marginal loss function



$$h(\mathrm{x}_i) = \mathrm{sign}(\mathrm{w}\cdot\mathrm{x}_i)$$

$$\hat{R}_\rho(h) = \frac{1}{m}\sum_{i=1}^m \Phi_\rho(y_i h(\mathrm{x}_i))$$

We have the **marginal loss** is bounded by **hinge loss**

$\Phi_1(\mathrm{x}) \leq \max(1 - \mathrm{x}, 0)$

Theorem: For any fixed δ > 0, we have with probability 1- δ that the generalization error for marginal loss function is

$$R(h) \leq \underbrace{\frac{1}{m}\sum_{i=1}^m \xi_i}_{\text{empirical risk}} + \underbrace{2\sqrt{\frac{r^2\Lambda^2}{m}}}_{\text{class complexity (regularization)}} + \underbrace{\sqrt{\frac{\log\frac{1}{\delta}}{2m}}}_{\text{sampling confidence}}$$

$$\begin{aligned}\Phi_1(y_i h(\mathrm{x}_i)) &\leq \max\{1 - y_i h(\mathrm{x}_i), 0\}\\ &= \max\{1 - y_i \mathrm{sign}(\mathrm{w}\cdot\mathrm{x}_i), 0\}\\ &= \xi_i.\end{aligned}$$

$$\hat{R}_\rho(h) \leq \frac{1}{m}\sum_{i=1}^m \xi_i.$$

SVM Objective:

$$\sum_{i=1}^m \xi_i + \frac{1}{2}\|\mathrm{w}\|^2.$$

Key Result: The SVM objective function → makes small the RHS bound!

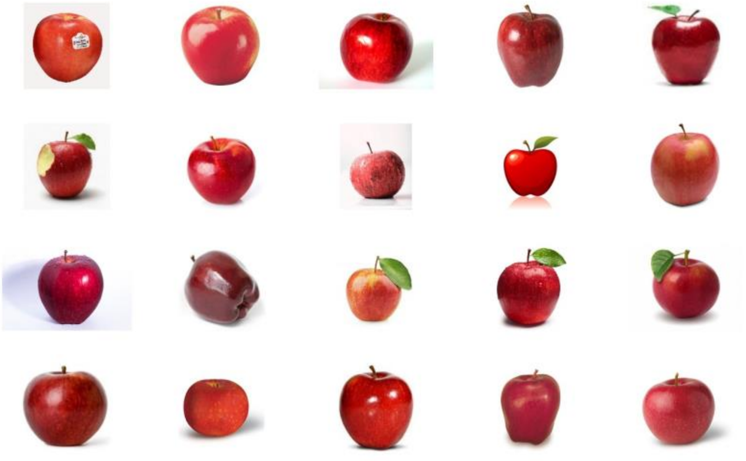
Trade-off: make **slack variables small** while making **margin** ρ = 1/||w|| **large. Allows for** $N = \infty$.

Regularization: Make $\|w\|^2 \leq \Lambda^2$ **small** → serves as **regularization** term (controlled by $\Lambda = \Lambda(C^{-1})$).

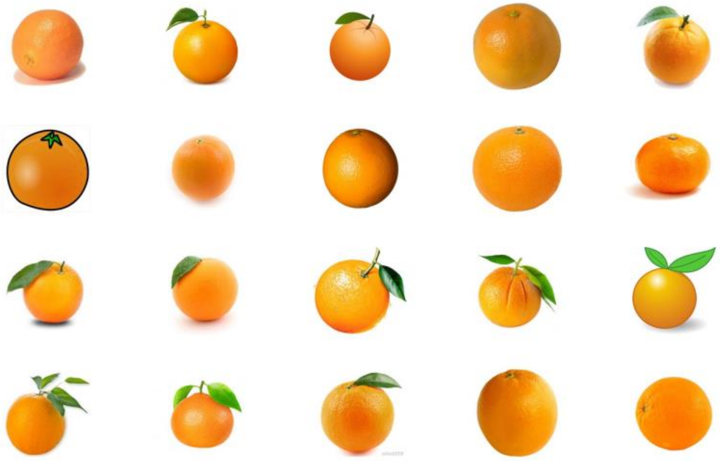
Example of SVM
Classifying Apples & Oranges

Support Vector Machines (Apples and Oranges Training Data)

Apple Image Set



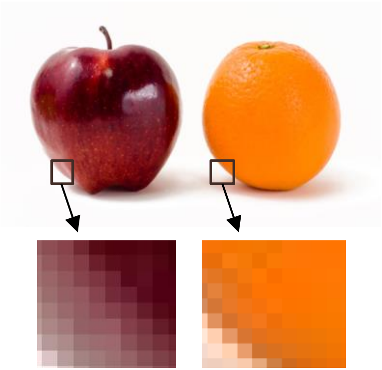
Orange Image Set



Average Apple

Average Orange

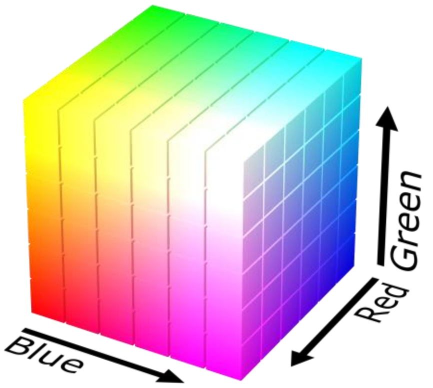
Supervised Learning (Apples and Oranges)



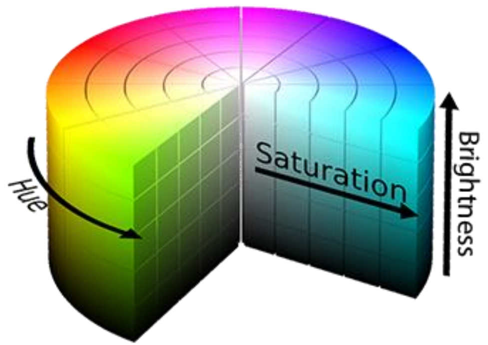
What features to use to distinguish apples and oranges?

- Natural to use the colors of the objects in the images.
- However, many different color spaces can be used (RGB, HLS, ...)
- Does the choice matter?

Red, Green, Blue: [Pixel (RGB)]

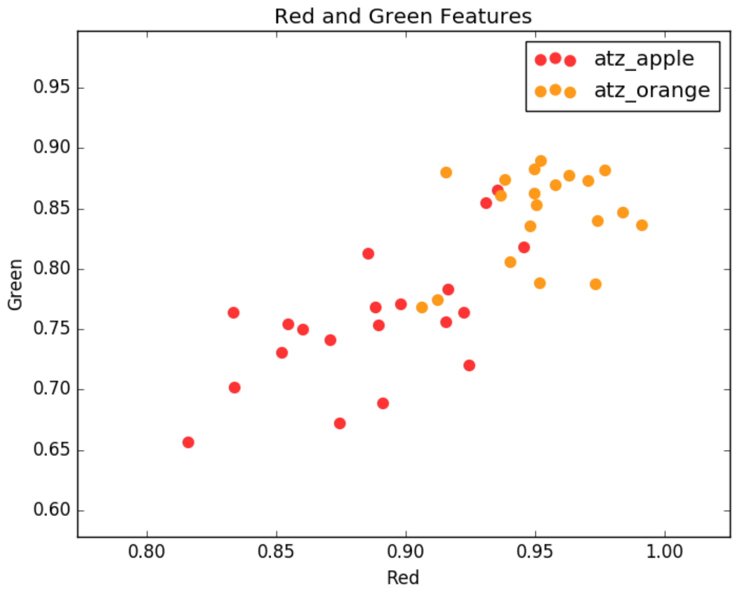


Hue, Luminance, Saturation: [Pixel (HLS)]

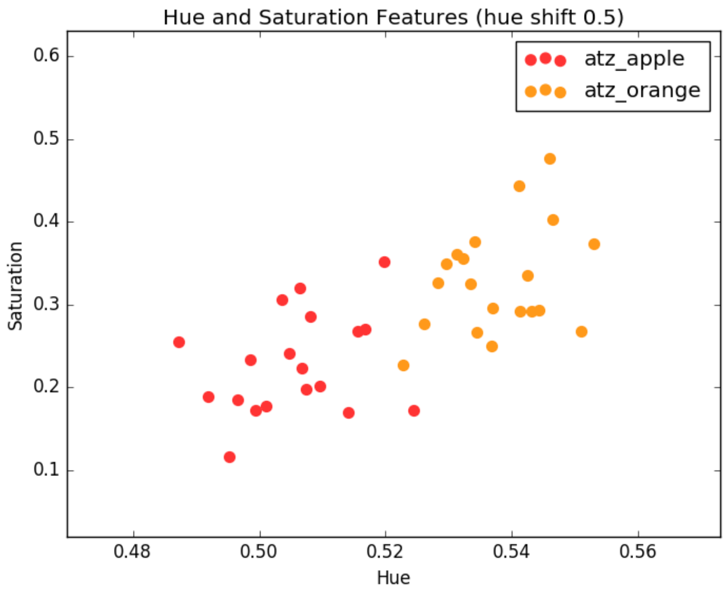


Support Vector Machines (Apples and Oranges Training Data)

RGB Training Data



HLS Training Data



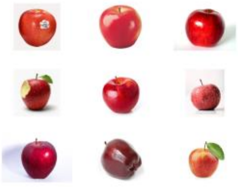
SVM Performance

Linear: $K(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}, \mathbf{y} \rangle$

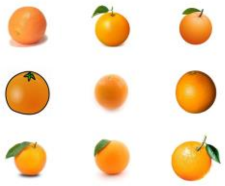
RBF: $K(\mathbf{x}, \mathbf{y}) = \exp\left[-\gamma \|\mathbf{x} - \mathbf{y}\|^2\right]$

Polynomial: $K(\mathbf{x}, \mathbf{y}) = (\gamma \langle \mathbf{x}, \mathbf{y} \rangle + r)^d$

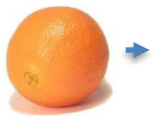
Apple Images



Orange Images



Item

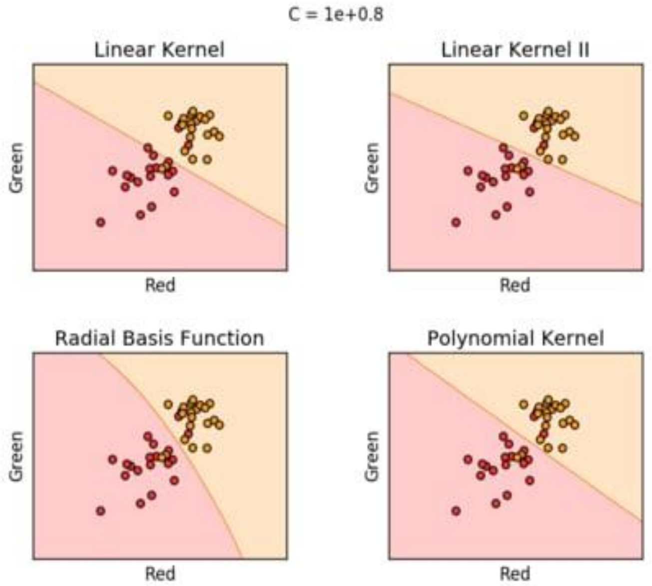


Features

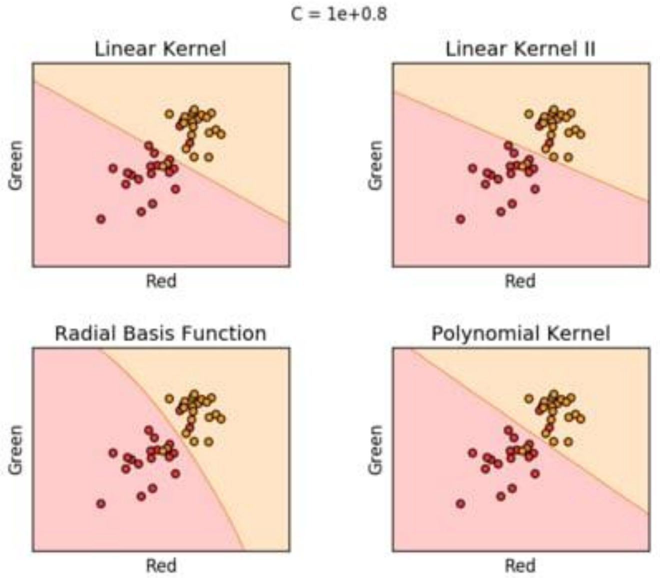
Feature	Value
Roundness	0.8
Sweetness	0.9
Redness	0.1
Greenness	0.3

SVM Results:

RGB Features



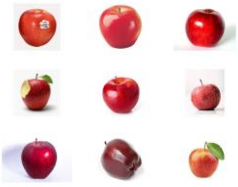
HLS Features



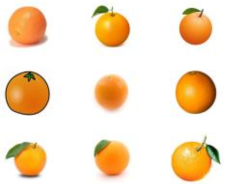
SVM Performance

Importance of features used?
Importance of regularization C?
How does training set generalize?

Apple Images



Orange Images



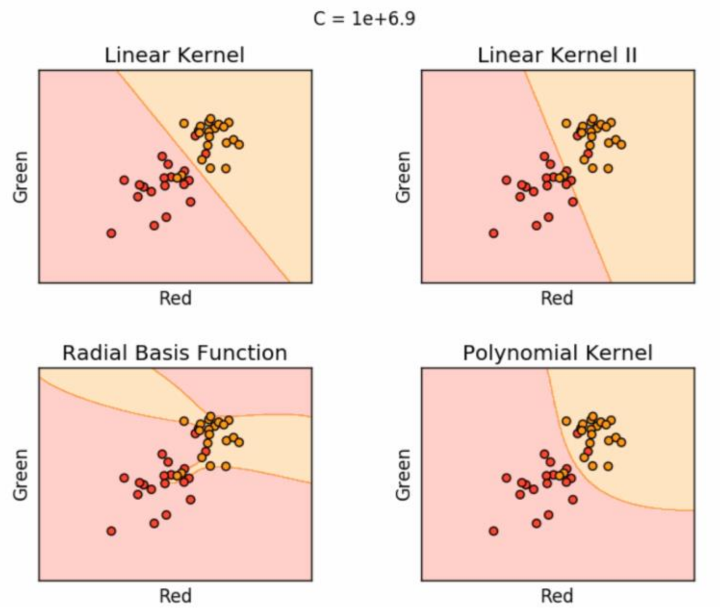
Item

Features

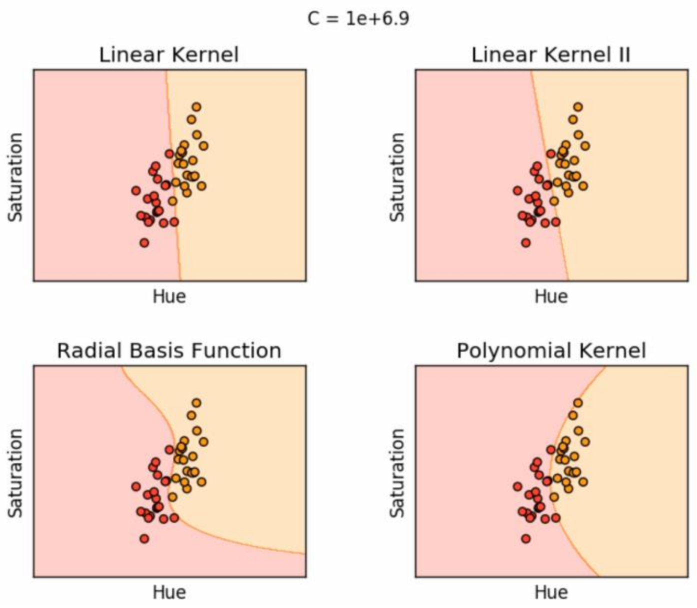
Feature	Value
Roundness	0.8
Sweetness	0.9
Redness	0.1
Greenness	0.3

SVM Results:

RGB Features



HLS Features

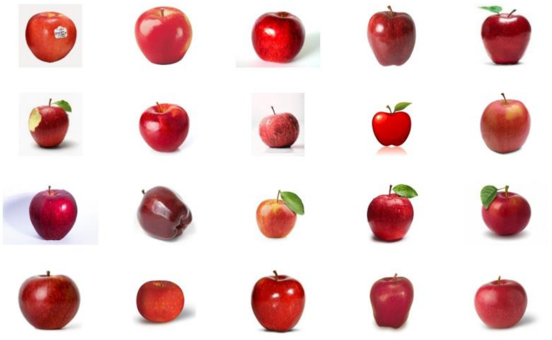


SVM Example
Apples vs Oranges vs Blueberries

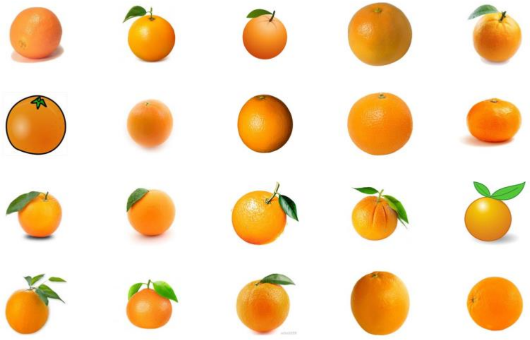
Support Vector Machines (Apples and Oranges and Blueberries)

How might we train on more than two data sets?
Three data sets: Apples, Oranges, and Blueberries.

Apple Image Set



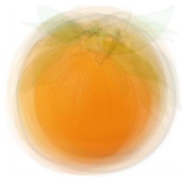
Orange Image Set



Blueberry Image Set

Average Apple

Average Orange



Average Blueberry

SVM Multi-class Classification

Support Vector Machines (Multi-Class Case)

Many classification involve multiple classes.

Problem: From data x learn for k classes $C_1, C_2, \ldots, C_k$ a classifying function $f(x) \rightarrow y \in \{C_1, C_2, \ldots C_k\}$.

Binary classification: $f(x) \rightarrow y \in \{-1,1\}$ which corresponds to k = 2 classes $C_1 = -1, C_2 = 1$.

Multi-class classification: $f(x) \rightarrow y \in \{1,2,\ldots,k\}$ corresponds to k classes $C_1 = 1, C_2 = 2, \ldots, C_k = k$.

How can we extend linear classifier methods to handle multiple classes?

Two common approaches:

One vs All (OvA): Reduce to a collection of k binary classification problems to determine one category labeled +1 vs rest of the data labeled -1. Pick classification with the greatest margin.

One vs One (OvO): Reduce to a collection of $\binom{k}{2} = k(k-1)/2$ binary classification problems to determine one category labeled +1 vs one other category labeled -1. Consider each classifier as a voter and pick class with the most number of votes.

Above heuristics do not always work well in practice. Alternatives: optimization formulations (more expensive).

Support Vector Machines (Multi-Class Classification)

Optimization of Maximum Margin (OMM): Classes $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$, with $C_\ell = \ell$, data (x_i, y_i) with $y_i \in \mathcal{C}$.

$\min_{\boldsymbol{W},\xi} \frac{1}{2} \sum_{l=1}^{k} \|\boldsymbol{w}_l\|^2 + C \sum_{i=1}^{m} \xi_i$
subject: $\forall i \in [1, m], \forall l \in \mathcal{C} \setminus \{yi\}$
$\boldsymbol{w}_{y_i} \cdot \boldsymbol{\Phi}(\boldsymbol{x}_i) \geq \boldsymbol{w}_l \cdot \boldsymbol{\Phi}(\boldsymbol{x}_l) + 1 - \xi_i, \quad \xi_i \geq 0$

Classifier obtained: $h(\boldsymbol{x}) = \mathrm{argmax}_{l \in \mathcal{C}}\, \boldsymbol{w}_l \cdot \boldsymbol{\Phi}(\boldsymbol{x}_l)$, where $\boldsymbol{\Phi}(\boldsymbol{x})$ is transformation of the data.

Dual Optimization Problem (Keneralization): $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \boldsymbol{\Phi}(\boldsymbol{x}_i), \boldsymbol{\Phi}(\boldsymbol{x}_j) \rangle$

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^{m \times k}} \sum_{i=1}^{m} \boldsymbol{\alpha}_i \cdot \mathbf{e}_{y_i} - \frac{1}{2} \sum_{i=1}^{m} (\boldsymbol{\alpha}_i \cdot \boldsymbol{\alpha}_j) K(x_i, x_j)$$

subject to: $\forall i \in [1, m], (0 \leq \alpha_{iy_i} \leq C) \wedge (\forall j \neq y_i, \alpha_{ij} \leq 0) \wedge (\boldsymbol{\alpha}_i \cdot \mathbf{1} = 0)$

Generalization Bounds: $R(h) = \mathop{\mathrm{E}}_{x \sim D}[1_{h(x) \neq f(x)}]$ $\quad \mathcal{H} = \left\{ h(\boldsymbol{x}) = \mathrm{argmax}_{l \in \mathcal{C}}\, \boldsymbol{w}_l \cdot \boldsymbol{\Phi}(\boldsymbol{x}) \,\middle|\, \boldsymbol{W} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_k)^T, \sum_{l=1}^{k} \|\boldsymbol{w}_l\|^2 \leq \Lambda^2 \right\}$

$$R(h) \leq \frac{1}{m} \sum_{i=1}^{m} \xi_i + 2k^2 \sqrt{\frac{r^2 \Lambda^2}{m}} + \sqrt{\frac{\log \frac{1}{\delta}}{2m}}$$

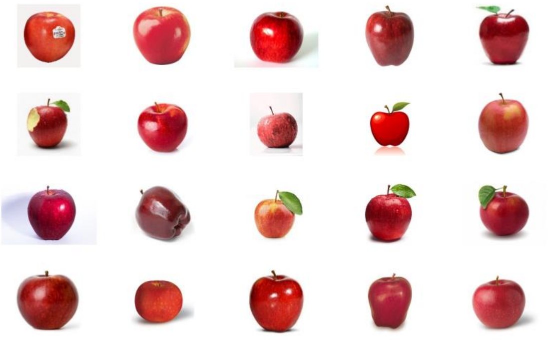
(for any $\delta > 0$ holds with probability $1 - \delta$)

where $\xi_i = \max\left(1 - [\mathbf{w}_{y_i} \cdot \boldsymbol{\Phi}(x_i) - \max_{y' \neq y_i} \mathbf{w}_{y'} \cdot \boldsymbol{\Phi}(x_i)], 0\right)$ for all $i \in [1, m]$

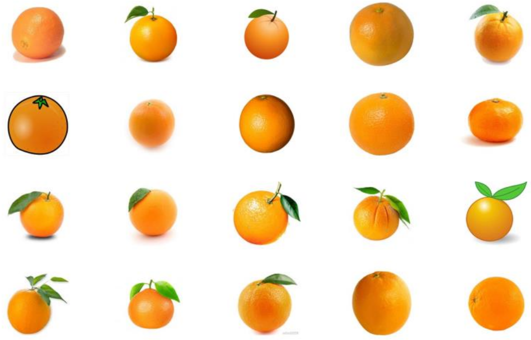
Support Vector Machines (Apples and Oranges and Blueberries)

How might we train on more than two data sets?
Three data sets: Apples, Oranges, and Blueberries.

Apple Image Set



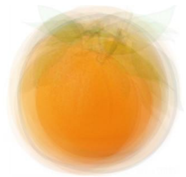
Orange Image Set



Blueberry Image Set

Average Apple

Average Orange

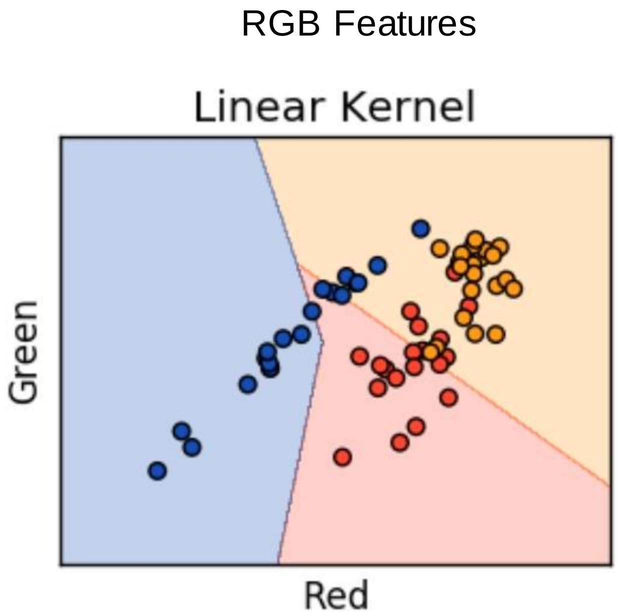


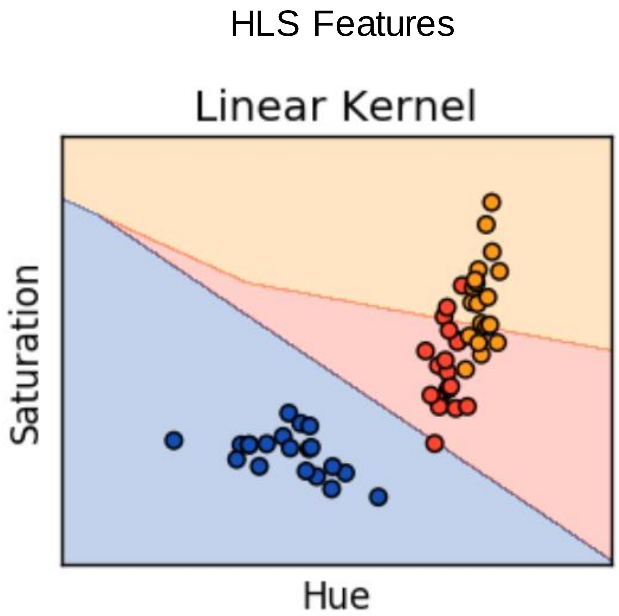
Average Blueberry

Supervised Learning (SVM Results: Apples, Oranges and Blueberries)

How does SVM distinguish between three data sets?
Importance of features used.
Importance of regularization C.
How does training set generalize?

SVM Results:





C = 1e-4.0

Supervised Learning (SVM Results: Apples, Oranges and Blueberries)

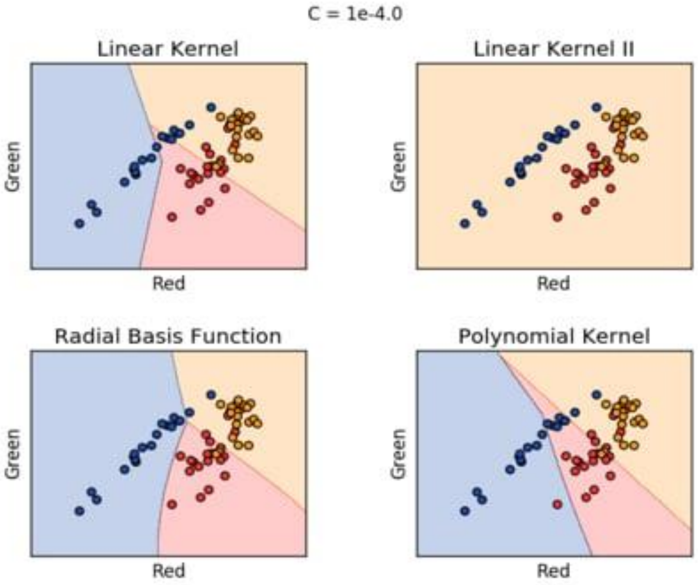
Linear: $K(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}, \mathbf{y} \rangle$

RBF: $K(\mathbf{x}, \mathbf{y}) = \exp\left[-\gamma \|\mathbf{x} - \mathbf{y}\|^2\right]$

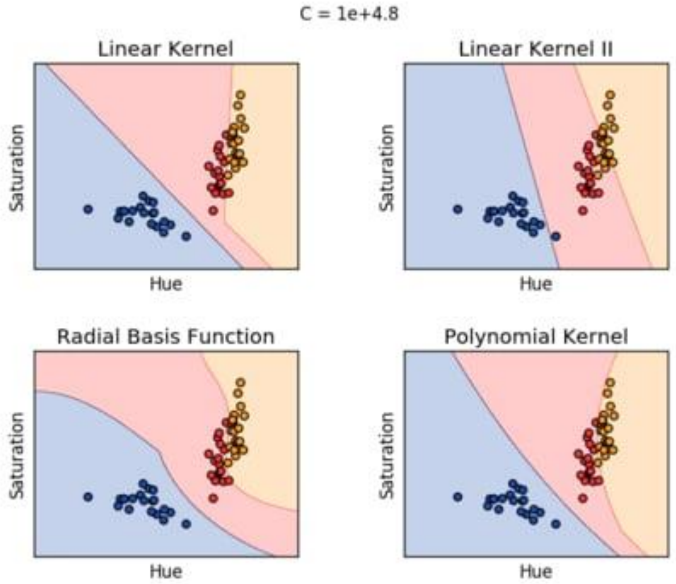
Polynomial: $K(\mathbf{x}, \mathbf{y}) = (\gamma \langle \mathbf{x}, \mathbf{y} \rangle + r)^d$

SVM Results:

RGB Features



HLS Features

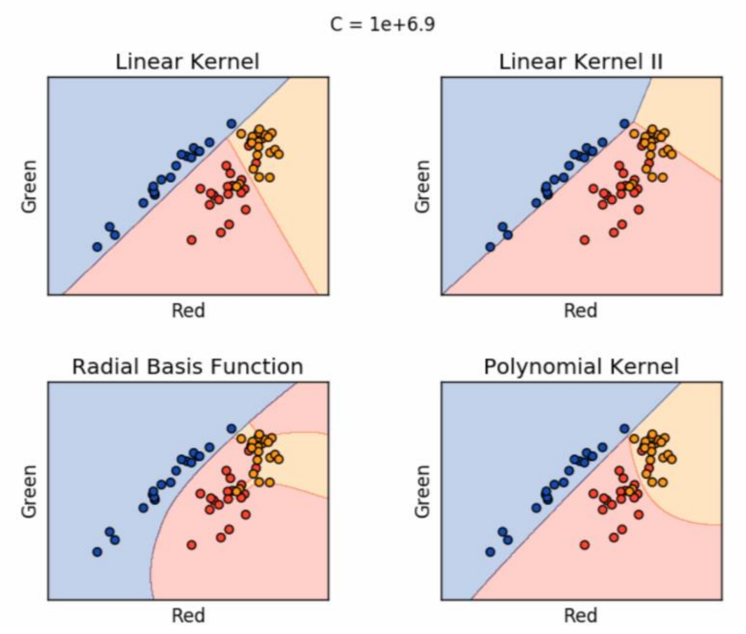


Supervised Learning (SVM Results: Apples, Oranges and Blueberries)

How does SVM distinguish between three data sets?
Importance of features used.
Importance of regularization C.
How does training set generalize?

SVM Results:

RGB Features



HLS Features