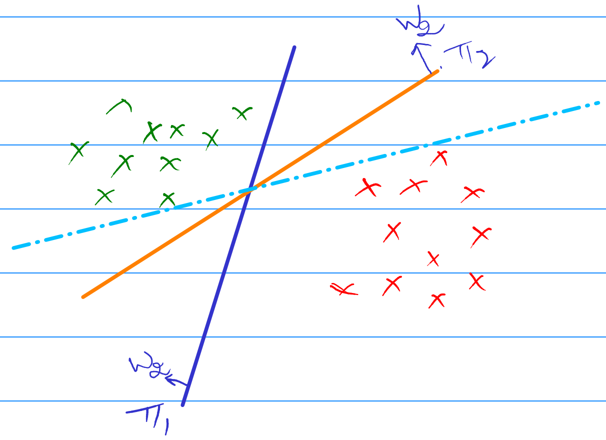
Notes by,
Bhargav

Last updated 16/06/20

Support Vector Machine: (SVM)

Both Classification and Regression

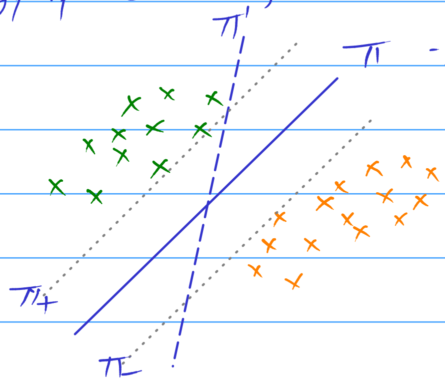
Geometric Intuition:



Many planes / Hyperplanes seperate the two classes.

- Seperates +ve pts from −ve points as widely as possible.

points that are close to hyperplane have less probability of belonging to a class that hyperplane classifies than pts. that are farther to the hyperplane.



π - Seperates points as farther as possible than π'

π_+, π_- touch the 1st point in their respective class.

π – margin maximising hyperplane.

Let π_+ be a plane // to π
π_- be a plane // to π

$dist(\pi_+, \pi_-) = d$, is Constant
↳ Margin

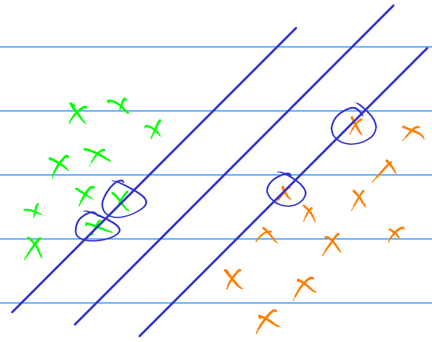
SVM: Tries to maximizes the margin

Margin ↑ ; Generalization accuracy ↑
↳ Accuracy on unseen Data; New data

Positive Hyperplane ← π_+ → Margin Maximising Hyperplane π

Support Vector:

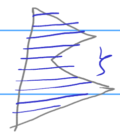
π_- → Negative hyperplane



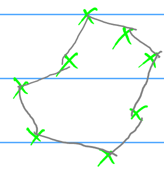
Points through which π_+, π_- pass are called "Support vectors"

Alternative Geometric Intution of SVM:

① Draw a Convex Hull for +ve points & -ve points seperately.



NonConvex polygon

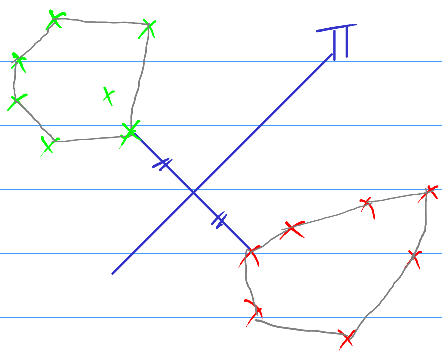


Convex polygon; ConvexHull

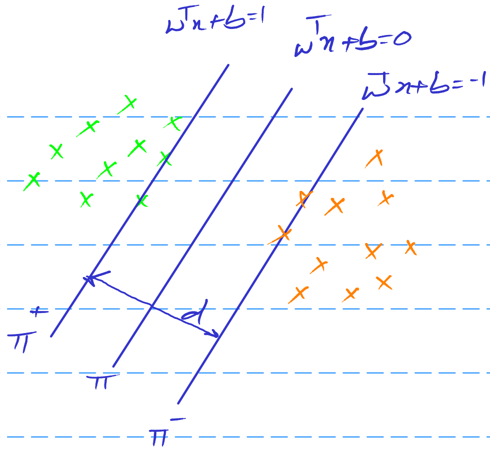
Smallest polygon so that all points lie inside or on the polygon.

② Draw shortest line connecting both the Hulls.

③ Bisect the line; the Hyperplane Bisecting the line will be the Margin maximising plane



Mathematical formulation of SVM:



π : Margin maximising Hyperplane

Let,

$$\pi : w^T x + b = 0$$

if,

$$\pi^+ : w^T x + b = 1$$

Margin $d = \frac{2}{||w||}$ → L_2 Norm

$$\pi^- : w^T x + b = -1$$

$$(w^*, b^*) = \underset{w,b}{argmax} \frac{2}{||w||}$$ = Margin maximising

Such that

$$y_i(w^T x + b) \geq 1 \;\; \forall x_i$$

n constrains for all n points

Hard Margin SVM

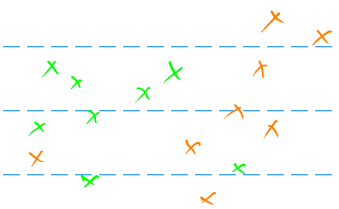
$$w^*, b^* = \underset{w,b}{argmax} \frac{2}{||w||}$$

such that $\forall i$, $y_i(w^T x_i + b) \geq 1$

constrained optimisation problem.

Assumes data is strictly linearly seperable.

What if Data is not linearly seperable?



Almost linearly seperable data →

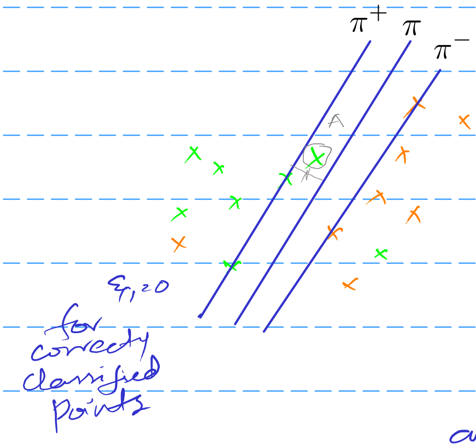
In such a case there will be no w^*, b^* that satisfies optimisation problem.

So Above optimisation problem is called Hard Margin SVM

$\pi : w^T x + b = 0$

$\pi^+ : w^T x + b = +1$

$\pi^- : w^T x + b = -1$



1. Create a New variable

ζ_i (zeta i)

$\zeta_i = 0$, if $y_i(w^T x + b) \geq 1$

↳ Correctly classified

$\zeta_i > 0$ it is equal to some units of distance away from the hyperplane in incorrect direction.

For example point A,

$y_i(w^T x + b) = 0.5$ 0.5 Distance units away from hyperplane π

$= \underbrace{1-(0.5)}_{\zeta_i}$

Soft margin SVM :- Regularization

$$(w^*, b^*) = \underset{w,b}{argmin} \underbrace{\frac{||w||}{2}}_{1/(\text{margin})} + \overset{\text{Hyperparameter}}{C} . \underbrace{\frac{1}{n} \sum_{i}^{n} \zeta_i}$$

Loss (Hinge) or → Avg distance of misclassified points from Correct π

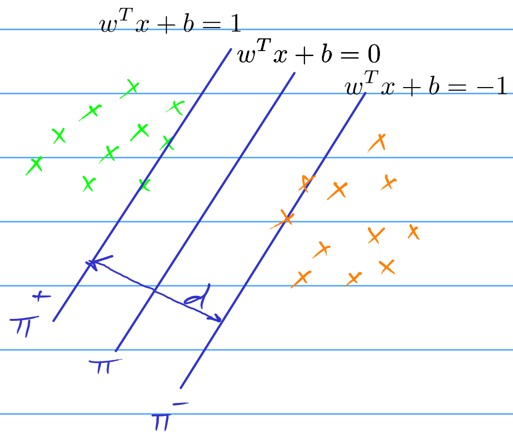
Such that
$$\begin{cases} y_i(w^T x_i + b) \geq 1 - \zeta_i \;\; \forall i \\ \zeta_i \geq 0 \end{cases}$$

$\min_{w,b}$ C. loss + regularization

C ↑ ; Tendency to make mistakes on train data ↓ (or) Overfit
High variance

C ↓ underfit ; High bias

Why +1 and -1 for π^+ and π^- :



* w could be any vector and w does not have to be a unit vector

$||w|| \neq 1$

$$w^*, b^* = \underset{w,b}{argmax} \frac{2}{||w||}$$

① π^+ :- $w^T x + b = k$ $k > 0$

π^- :- $w^T x + b = -k$ k is a constant

$$\text{margin} = \frac{2k}{||w||}$$

$$\underset{w,b}{\text{argmax}} \frac{2}{||w||} = \underbrace{\underset{w,b}{\text{argmax}} \frac{2k}{||w||}}_{\text{Same optimisation problem as } k \text{ is a constant}}$$

② $w^T x + b = k$

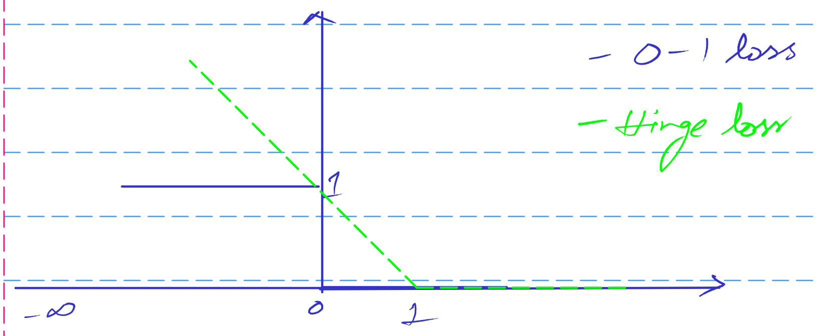
$$\left(\frac{w}{k}\right)^T x + \frac{b}{k} = 1$$

You can have any value of k but it does not occur in optimisation problem.

This is possible only because w is not a unit vector and $\frac{2}{||w||}$ is not a constant

Loss minimization: Hinge - loss:

Logistic Regression: Logistic Loss + Regularization
Linear Regression: Linear loss + Regularization
SVM :- Hinge loss + Regularization



$z_i = y_i(w^T x + b)$

Case 1: $z_i \geq 1$, $1 - z_i$ is -ve $\Rightarrow \max(\ \ ) = 0$

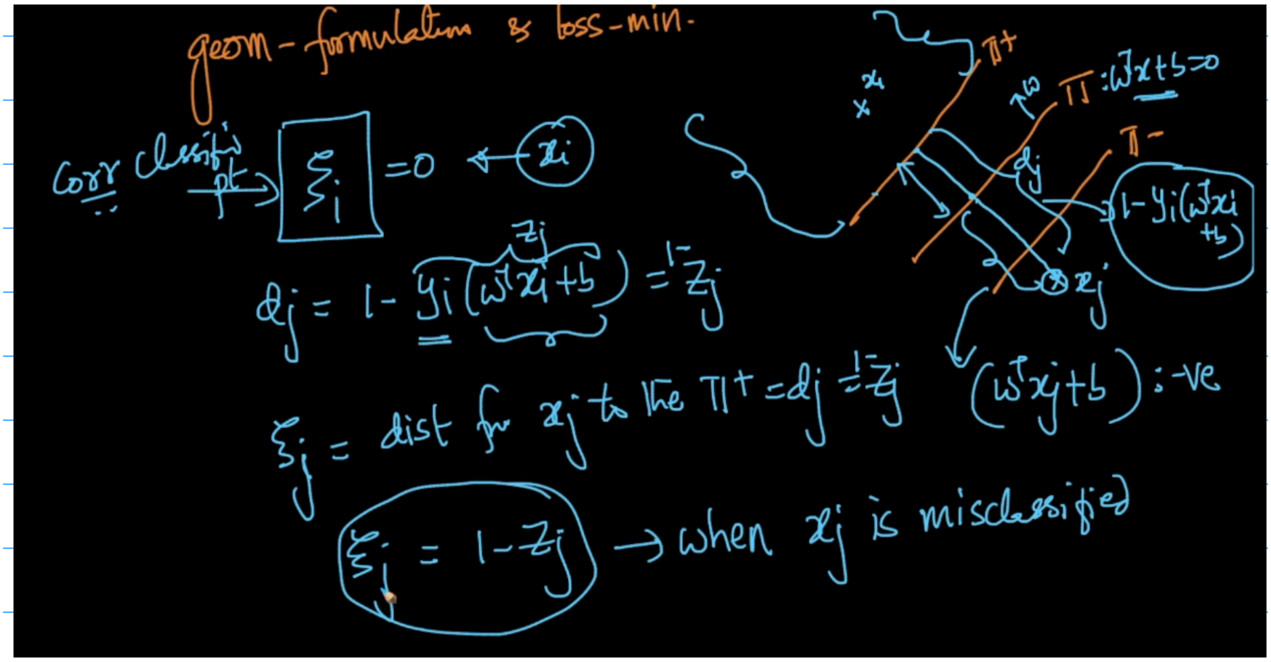
Case 2: $z_i < 1$, $1 - z_i > 0 \Rightarrow \max(\ \ ) = 1 - z_i$

$z_i \geq 1$; hinge loss $= 0$

$z_i < 1$; hinge loss $= 1 - z_i$

(or) $\max(0, 1 - z_i)$

Geometric formulation & loss minimization:



Soft SVM:

$C \uparrow \Rightarrow$ overfit

$C \downarrow \Rightarrow$ underfit

Primal of SVM

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

such that $(1 - y_i(w^T x_i + b)) \geq \xi_i \quad \forall i$

$\xi_i \geq 0$

Loss min:- $\min_{w,b} \sum_{i=1}^{n} \max(0, 1 - y_i(w^T x_i + b)) + \lambda \|w\|^2$

$\lambda \uparrow \Rightarrow$ underfit

$\lambda \downarrow \Rightarrow$ overfit

Equivalent

Dual form of SVM:

$$\max_{\alpha_i} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \, \alpha_j \, y_i \, y_j \, \underbrace{x_i^T x_j}_{K(x_i, x_j)}$$

Kernel function

such that $0 \leq \alpha_i \leq C$

$\sum_{i=1}^{n} \alpha_i y_i = 0$

$\alpha_i = 0$ for non support vectors

$\alpha_i > 0$ for support vectors

$x_i^T x_j = x_i \cdot x_j = \text{Cosine sim}(x_i, x_j)$ if $\|x_i\| = \|x_j\| = 1$

① for every $x_i \rightarrow \alpha_i$

② x_i's only occurs in form $x_i^T x_j$

③ $f(x_q) = \sum_{i=1}^{n} \alpha_i y_i x_i^T x_q + b$ $\rightarrow$ for a query point x_q

④ $\alpha_i > 0$ only for support vectors

$\alpha_i = 0$ for non support vectors

SVM only points that matter are support vectors since $\alpha_i = 0$ for non support vectors.

Kernel Trick:

[Similarity matrix & similarity between pairs of points]

one kind of similarity is a kernel function

$K(x_i, x_j)$; K - Kernel function

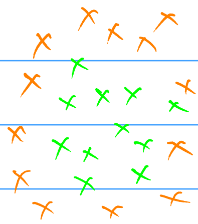
If no Kernel function is used it is called → Linear SVM

Else → Kernel SVM

Linear SVM: (x_i, x_j)

Kernel SVM: $K(x_i, x_j)$

Linear SVM is logistic regression Except for Margin maximization



If dataset looks like this,

Linear SVM ✗ fail

Logistic Regression ✗ fail

Logistic Regression + Feature Transform ✓ Succeed

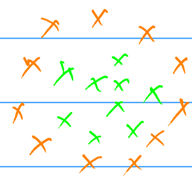
Kernel SVM with right kernel ✓ Succeed

Kernel SVM also does similar like feature transforms

$x_i \rightarrow x_i'$

Polynomial Kernel:

Kernalization



$$K(x_1, x_2) = (x_1^T x_2 + c)^d$$

Ex: $K(x_1, x_2) = (1 + x_1^T x_2)^2$; Quadratic kernel

$$= (1 + x_{11}x_{21} + x_{12}x_{22})^2$$

$x_1 = \langle x_{11}, x_{12} \rangle$

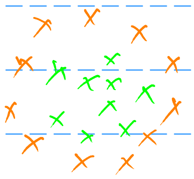
$x_2 = \langle x_{21}, x_{22} \rangle$

$$= 1 + x_{11}^2 x_{21}^2 + x_{12}^2 x_{22}^2 + 2x_{11}x_{21} + 2x_{12}x_{22} + 2x_{11}x_{21}x_{12}x_{22}$$

$$= (x_1')^T (x_2') \quad \text{Where,}$$

$$\left.\begin{aligned} x_1' &= [1, x_{11}^2, x_{12}^2, \sqrt{2}x_{11}, \sqrt{2}x_{12}, \sqrt{2}x_{11}x_{12}] \\ x_2' &= [1, x_{21}^2, x_{22}^2, \sqrt{2}x_{21}, \sqrt{2}x_{22}, \sqrt{2}x_{21}x_{22}] \end{aligned}\right\} \text{From 2D to 6D}$$

Kernalization: $\underset{\text{dimensions}}{d} \xrightarrow[\text{internally implicit transformation}]{\text{Feature transform}} d'$ where d' usually $> d$



For this dataset, polynomial kernel with degree 2 will work

Kernalization vs Feature Transforms

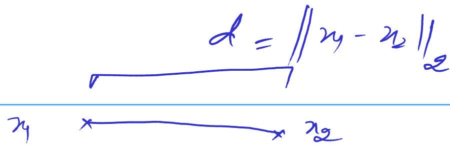
Kernalization	Feature Transforms
Similarity btwn 2 points	Transformation
Implicit	Explicit
Defined on Data points pairs of data points	Defined on feature

The beauty of kernels is that we don't obtain the explicit representation of points in high-dimensions.

We simply obtain the similarity between the high dimensional points through the kernel function without the need to obtain the points explicitly.

High-dimensional data is not always bad and is sometimes preferred especially in the case of classification using hyperplanes as higher-dimensions allow for easy splitting of data-points using a hyper-plane than lower-dimensional data.

Higher dimensional data certainly poses problems with visualization, but if our goal is to classify data, why bother too much about visualization.



Radial Basis Function (RBF):

$$K_{RBF}(x_1, x_2) = \exp\left(\frac{-||x_1 - x_2||^2}{2\sigma^2}\right) \quad ; \ \sigma \text{ - hyper parameter}$$

$$= \exp\left(\frac{-d_{12}^2}{2\sigma^2}\right)$$

$$\frac{1}{e^{(d^2/2\sigma^2)}}$$

① $d_{12} \uparrow \ ; \ K(x_1, x_2) \downarrow$

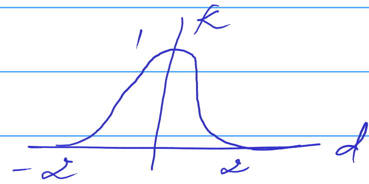
Like similarity

② Impact of σ :

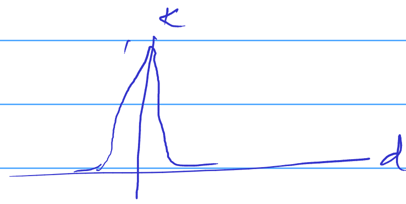
* $\sigma = 1;$ $\quad d = 0$

$k = 1$

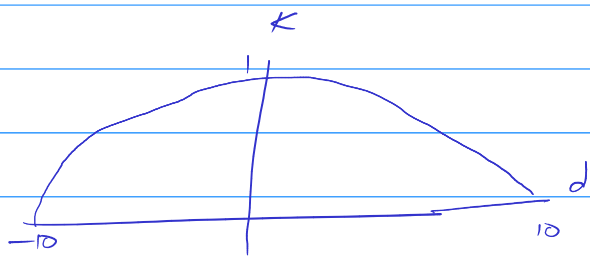
* $d \uparrow \ ; \ k \downarrow$ exponentially



* $\sigma = 0.1 \ ; \ \sigma^2 = 0.01$



* $\sigma = 10 \ ; \ \sigma^2 = 100$



KNN & RBF-Kernel :-

$\sigma \uparrow \implies k \uparrow$ in KNN ; Popular due to its similarity with KNN

RBF is a good approximation of KNN ; KNN requires high space Complexity and time Complexity.

Domain Specific Kernels:-

- String Kernels
- Genome Kernels

Feature Transformation is hard
↳ partially replaced by Right Kernel

RBF is a General purpose Kernel

Train & RunTime Complexity:

Train: → SGD
↳ specialized algorithm (Dual) → Sequential minimal optimization (SMO)

libSVM - library for training SVM

Training time: $\sim O(n^2)$ for Kernel SVMs

SVM is typically not used when n is Large

RunTime:

$$f(xq) = \sum_{i=1}^{n} \sigma_i \, y_i \, k(x_i, x_q) + b$$

No. of support vectors = K

We have no control over K

$O(Kd)$

nu-SVM: Control errors and support vectors: ν - SVM

Original formulation of SVM → C-SVM $C \geq 0$

Alternative formulation of SVM:-

nu-SVM :- $0 \leq \nu \leq 1$

nu ≥ fraction of errors
nu ≤ fraction of support vectors

If nu = 0.01 ⇒ % of errors = 1%
Support vectors ≥ 1% of n (n-datapoints)

n = 100,000

Support vectors k ≥ 1000

SVM Regression:-

$$f(x) = w^T x_i + b$$
$$= \hat{y}_i$$

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

Such that
$$y_i - (w^T x_i + b) \leq \epsilon$$
$$(w^T x_i + b) - y_i \leq \epsilon$$
$$\epsilon \geq 0$$

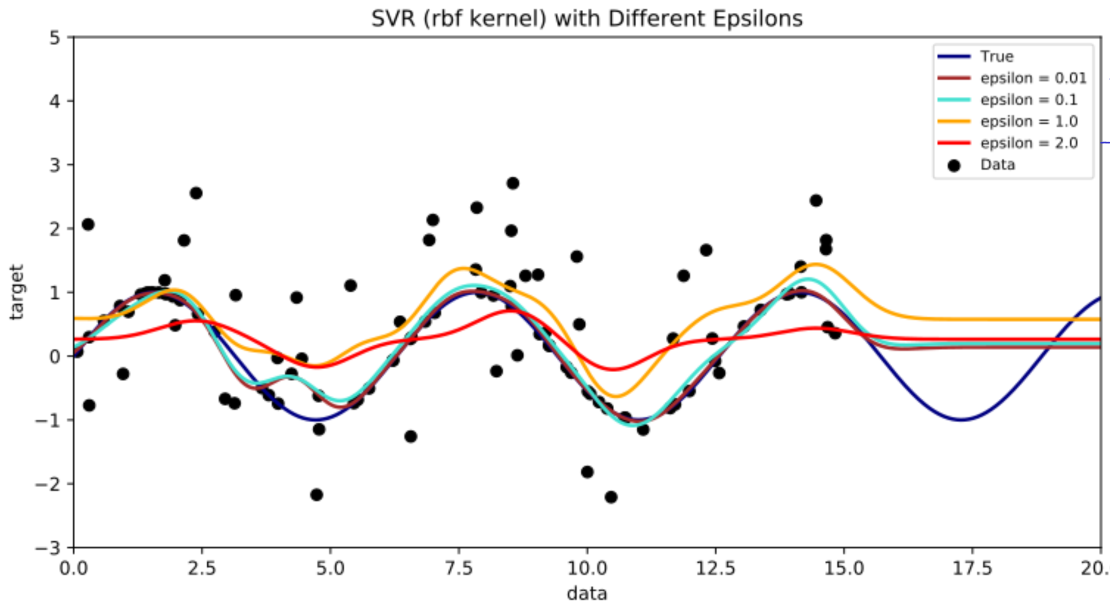
Linear SVR

ϵ - Hyperparameter

(or) $|y_i - \hat{y}| \leq \epsilon$ and $\epsilon \geq 0$

$\epsilon \downarrow$ ⇒ Errors are low on Train data ↓ ⇒ overfitting ↑
$\epsilon \uparrow$ ⇒ Errors are Train Data ↑ ⇒ underfitting ↑

RBF SVR Behaves similar to KNN:-



→ Could overfit

→ Could underfit

Cases: (SVM)

* Feature Engg & Feature Transform : Kernel design, Finding right kernel

* Decision Surface :- Linear SVM :- Hyperplane
  Kernel SVM :- d, x_i → non linear surface
  ↓
  d', x_i' → Linear Surface

* Similarity function / Distance fn
  ↳ Kernel

* Interpretability & Feature Importance → Linear SVM is Interpretable; Kernel SVM no interpretability
  ↳ Forward feature selection
  ↳ Can't get feature importance directly

* outliers :- – very little impact (Because only support vectors matter)
  – RBF with small σ will have outlier effect since it behaves like a KNN with small K

* Bias - Variance :- $C \uparrow \Rightarrow$ overfit $\Rightarrow$ High variance

$C \downarrow \Rightarrow$ underfit $\Rightarrow$ High bias

For RBF SVM :- C, σ 2 hyperparameters $\rightarrow$ Grid search

Large $D \uparrow \rightarrow$ Good for SVM

* Best Cases :- Right kernel

Worst Cases :- n is large $\rightarrow$ Low latency not possible

k is large