Supervised Learning: SVM Method + Model evaluation methods

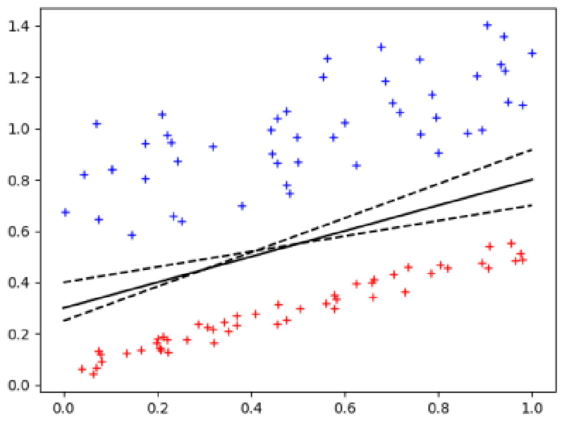
KASHTANOVA VICTORIYA

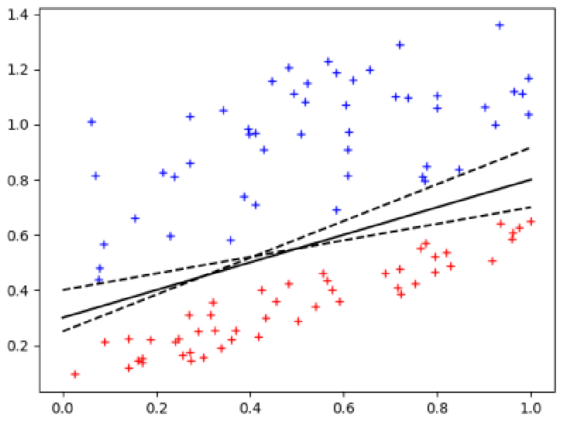
INRIA, EPIONE

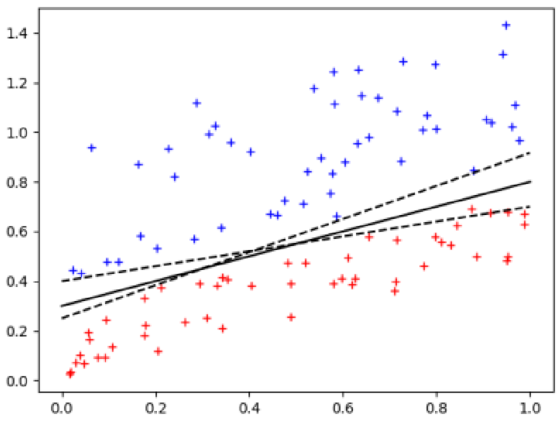
UNIVERSITÉ CÔTE D'AZUR

Inria INVENTEURS DU MONDE NUMÉRIQUE

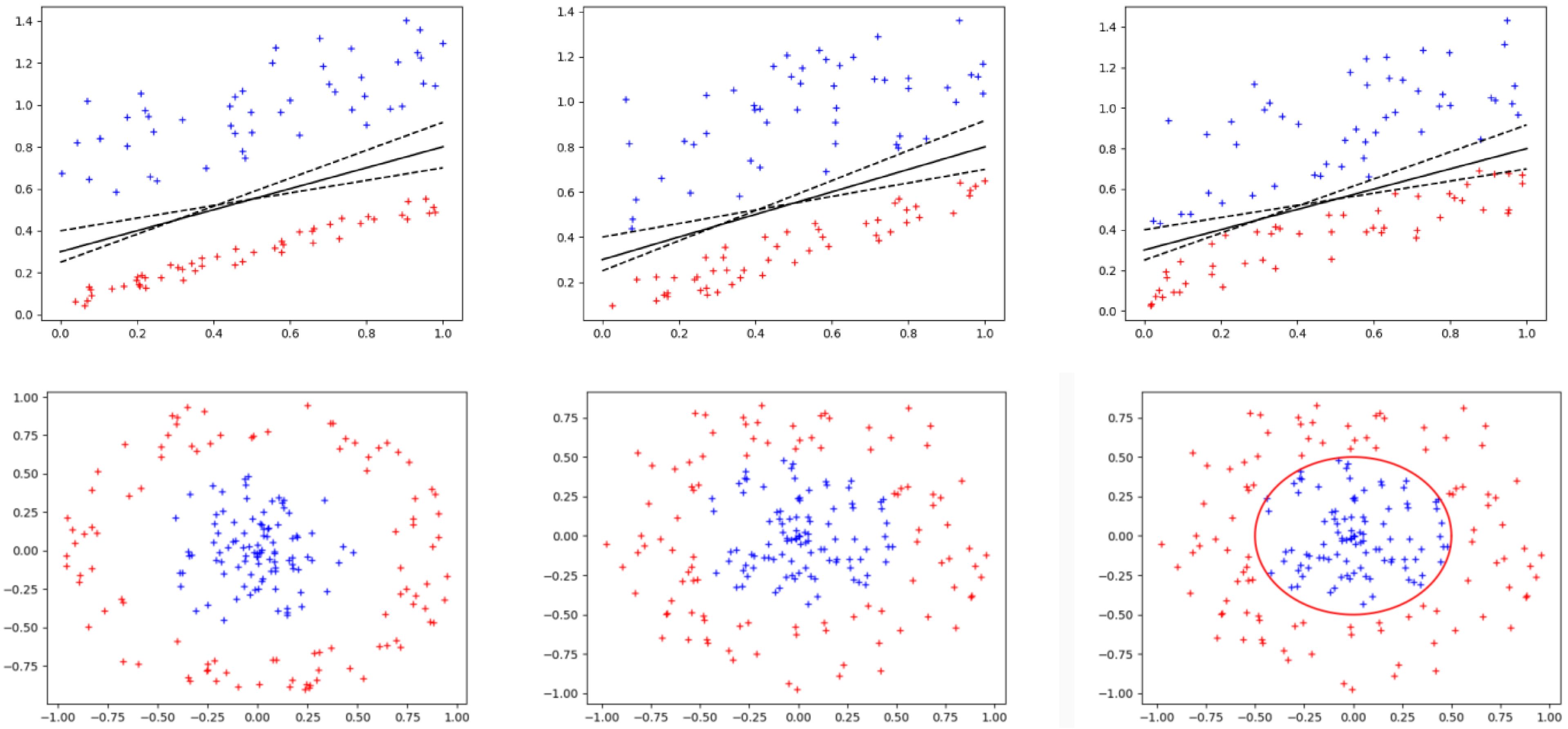
SVM (Support Vector Machine) Method







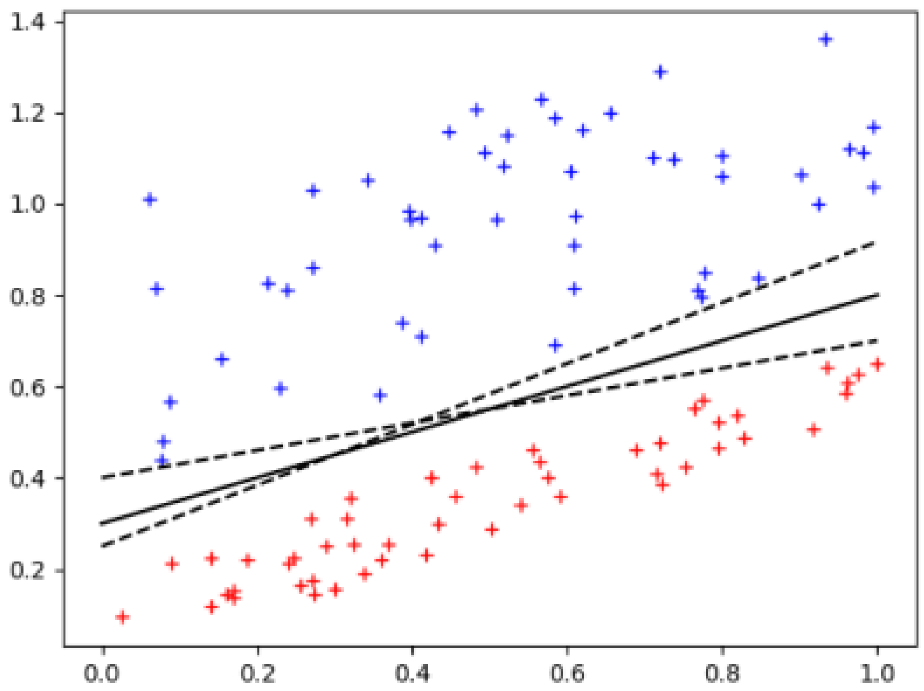
SVM (Support Vector Machine) Method



Linear SVM (Separable classes)

- Every data-sample : $x \in \mathbb{R}^D$
- Decision Boundary : $\mathcal{H} : w^T x + b = 0$
- Distance Measure of Hyper plane : $d_{\mathcal{H}}(x_0) = \frac{|w^T x + b|}{||w||_2}$

- **Goal** : $w^* = argmax_w[\min_n d_{\mathcal{H}}(x_n)]$



Linear SVM (Separable classes)

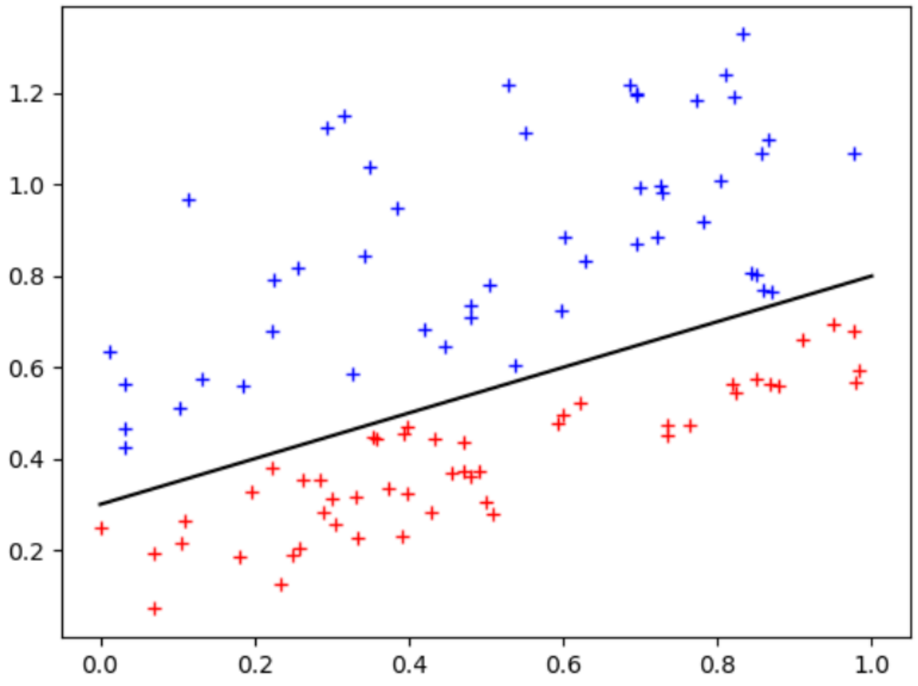
- Every data-sample : $x \in \mathbb{R}^D$
- Decision Boundary : $\mathcal{H}: \; w^T x + b = 0$
- Distance Measure of Hyper plane : $d_{\mathcal{H}}(x_0) = \frac{|w^T x + b|}{||w||_2}$
- **Goal** : $w^* = argmax_w [\min_n d_{\mathcal{H}}(x_n)]$

$$y_n[w^T x + b] = \begin{cases} \geq 0, class\ A \\ < 0, class\ B \end{cases}$$

$$w^* = argmax_w \frac{1}{||w||_2} [\min_n y_n[w^T x + b]]$$

Let $\min_n y_n[w^T x + b] = 1$

$$w^* = argmax_w \frac{1}{||w||_2}$$



Linear SVM (Separable classes)

- Every data-sample : $x \in \mathbb{R}^D$
- Decision Boundary : $\mathcal{H} : w^T x + b = 0$
- Distance Measure of Hyper plane : $d_{\mathcal{H}}(x_0) = \frac{|w^T x + b|}{||w||_2}$

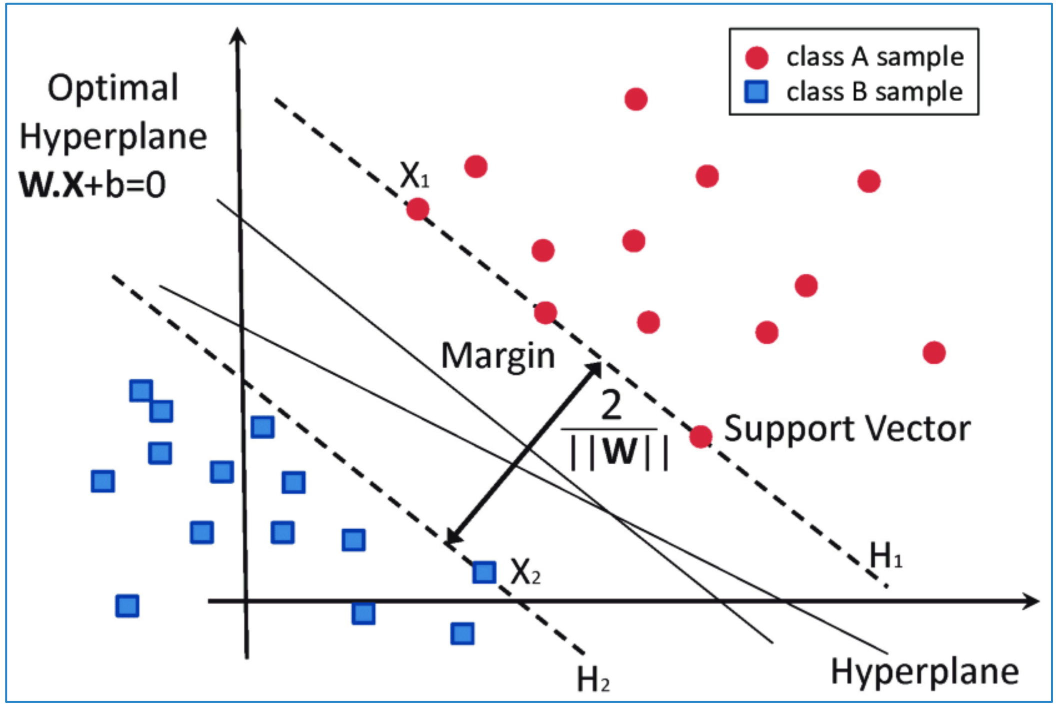
- **Goal** : $w^* = argmax_w [\min_n d_{\mathcal{H}}(x_n)]$

$$y_n[w^T x + b] = \begin{cases} \geq 0, class\ A \\ < 0, class\ B \end{cases}$$

$$w^* = argmax_w \frac{1}{||w||_2} [\min_n y_n [w^T x + b]]$$

$$y_n[w^T x + b] \geq 1, \forall n$$

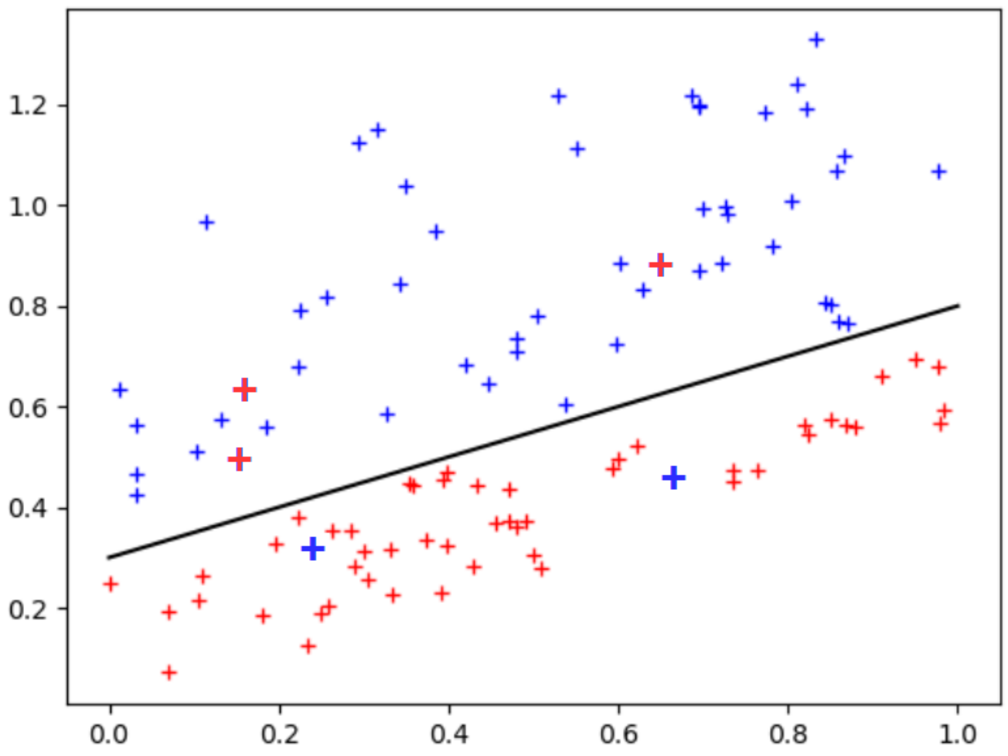
$$\min_w \frac{1}{2} ||w||_2$$



Linear SVM (General case)

Final form SVM optimization problem :

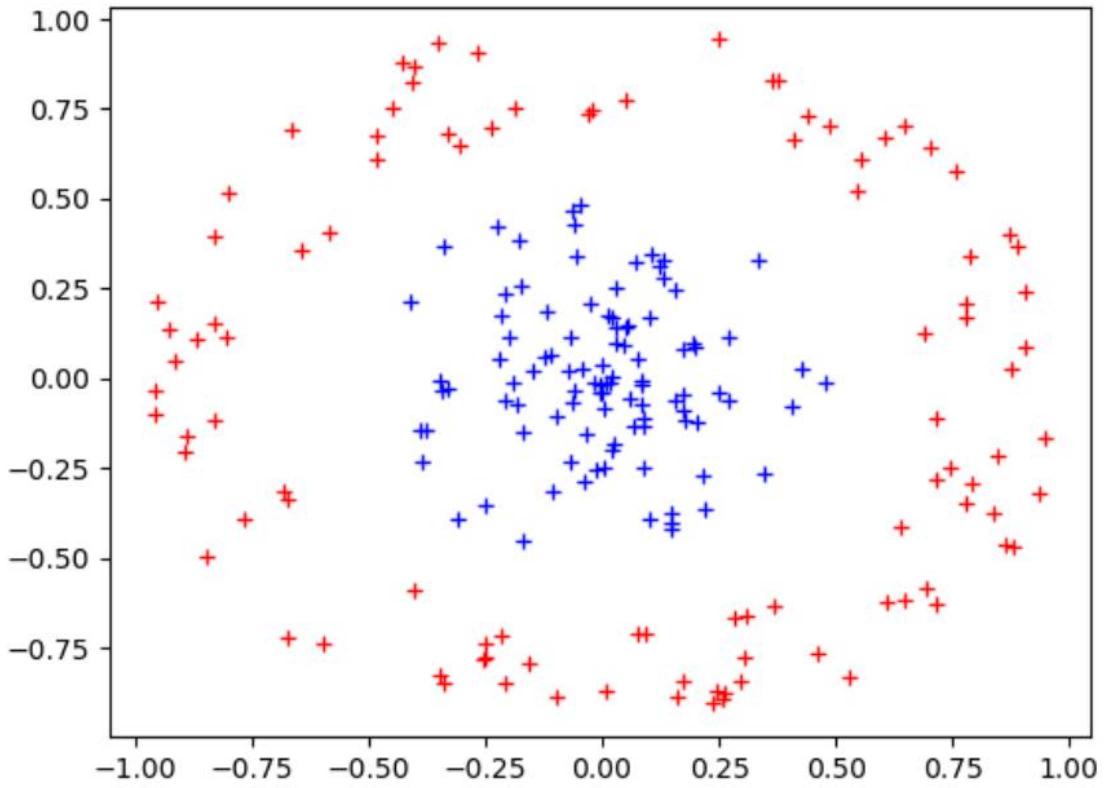
$$\min_{w} \frac{1}{2} ||w||_2 + C \sum_n \xi_n$$
$$s.t. \, y_n[w^T x + b] \geq 1 - \xi_n, \forall n$$
$$\xi_n \geq 0, \forall n$$



Linear SVM (General case)

Final form SVM optimization problem :

$$\min_{w} \frac{1}{2}||w||_2 + C \sum_{n} \xi_n$$
$$s.t.\, y_n[w^T x + b] \geq 1 - \xi_n, \forall n$$
$$\xi_n \geq 0, \forall n$$



Linear SVM (General case)

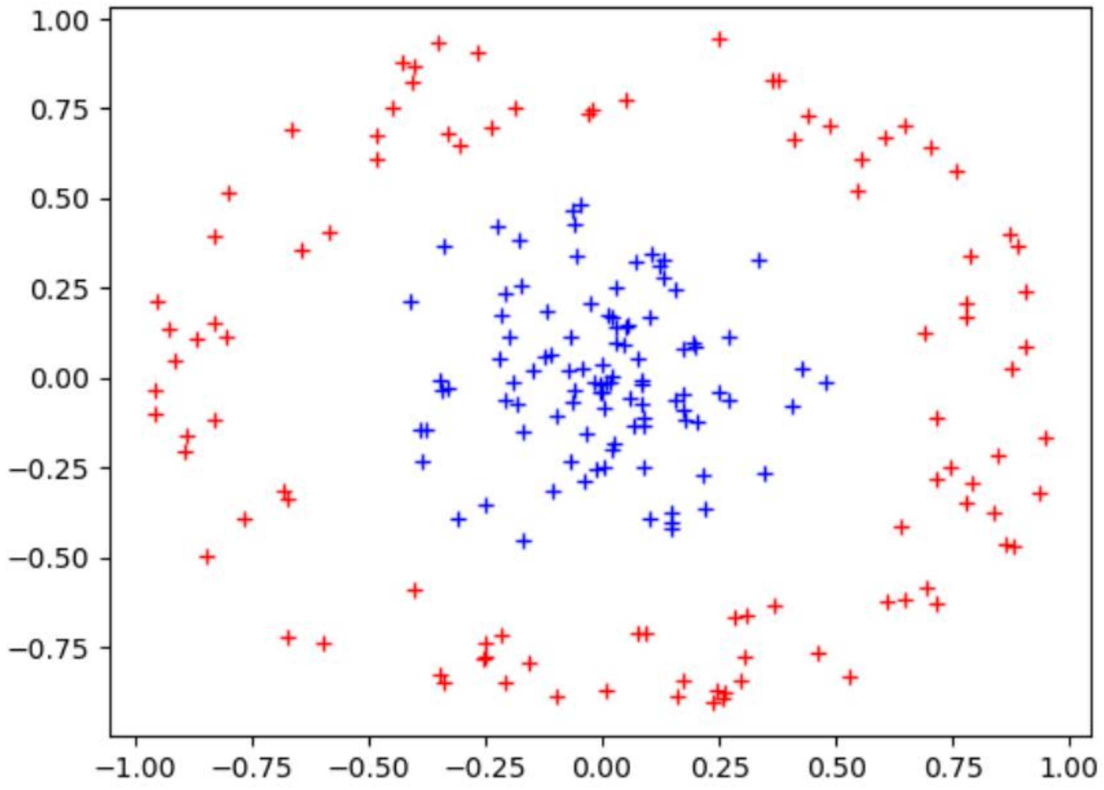
Final form SVM optimization problem :

$$\min_{w} \frac{1}{2}||w||_2 + C \sum_{n} \xi_n$$

$$s.t.\, y_n[w^T x + b] \geq 1 - \xi_n, \forall n$$

$$\xi_n \geq 0, \forall n$$

$$\boxed{\varphi(x) : \mathbb{R}^D \rightarrow \mathbb{R}^M}$$

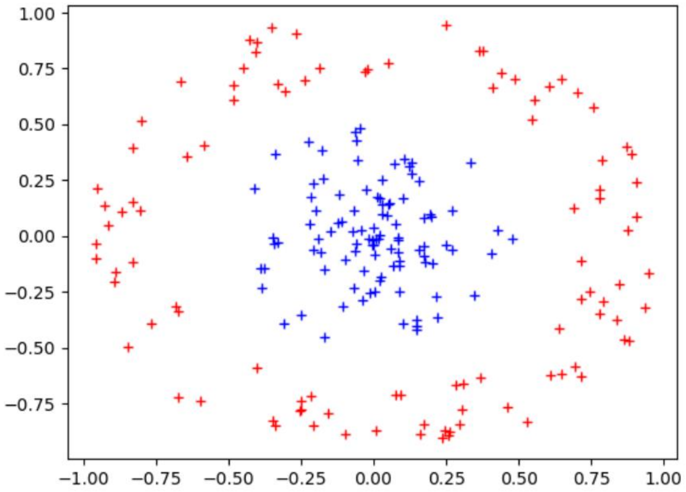


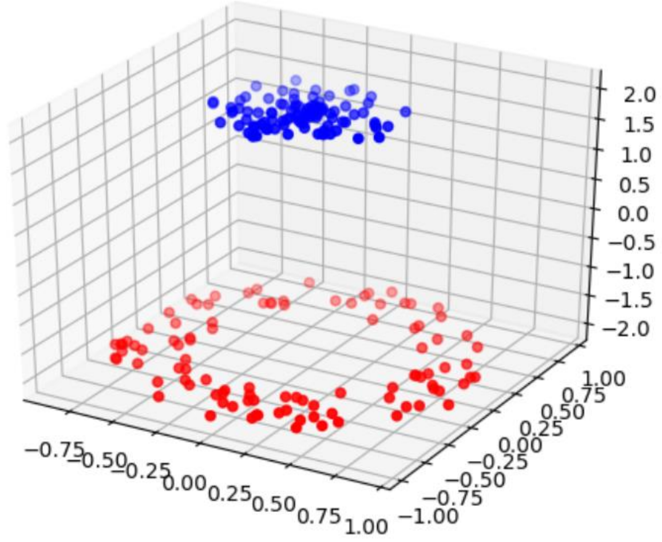
Linear SVM (General case)

Final form SVM optimization problem :

$$\min_{w} \frac{1}{2}||w||_2 + C \sum_{n} \xi_n$$
$$s.t.\, y_n[w^T x + b] \geq 1 - \xi_n, \forall n$$
$$\xi_n \geq 0, \forall n$$

$$\boxed{\varphi(x) : \mathbb{R}^D \rightarrow \mathbb{R}^M}$$

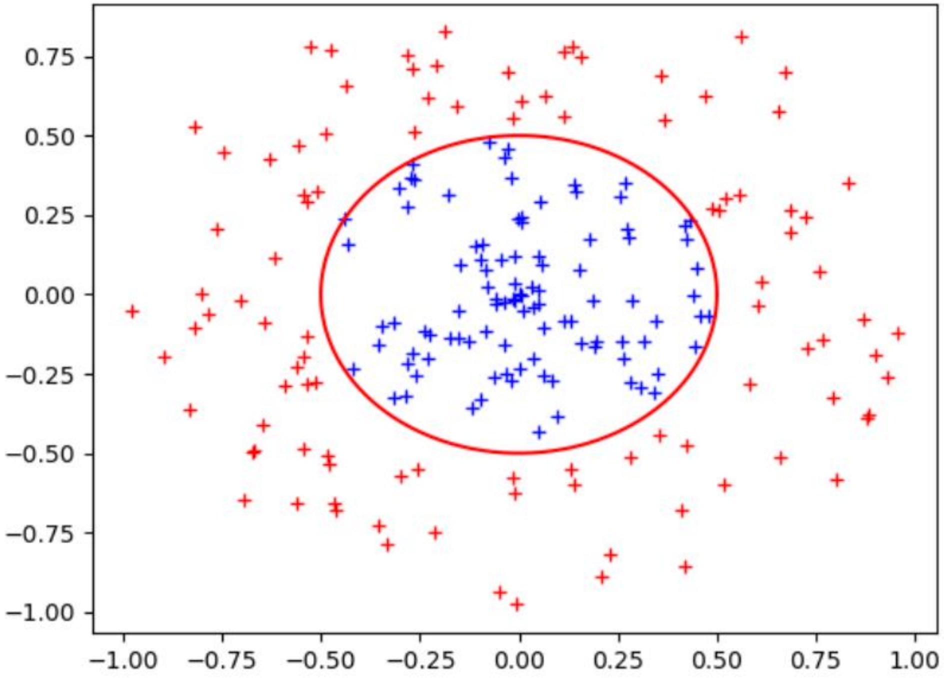




Linear SVM (General case)

The Most Final form SVM optimization problem :

$$\min_{w,\{\xi_n\}} \frac{1}{2}||w||_2 + C\sum_n \xi_n$$
$$s.t.\, y_n[w^T\varphi(x) + b] \geq 1 - \xi_n, \forall n$$
$$\xi_n \geq 0, \forall n$$

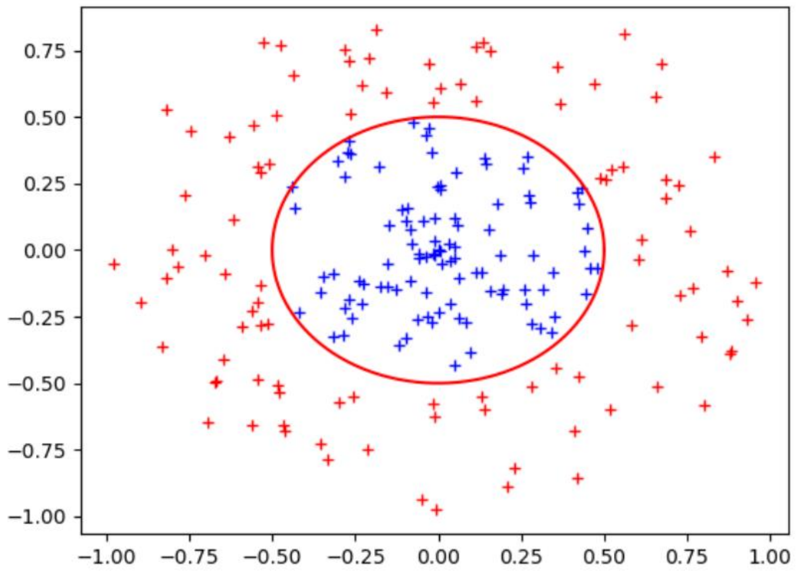


Kernel SVM

Problem : Don't know the nature of $\varphi(x)$ transformation

Idea : Rewrite Optimization problem without determining $\varphi(x)$

It's possible via applying Lagrange multipliers optimization method, Dual Form of SVM, Mercer Theorem etc.



Kernel SVM

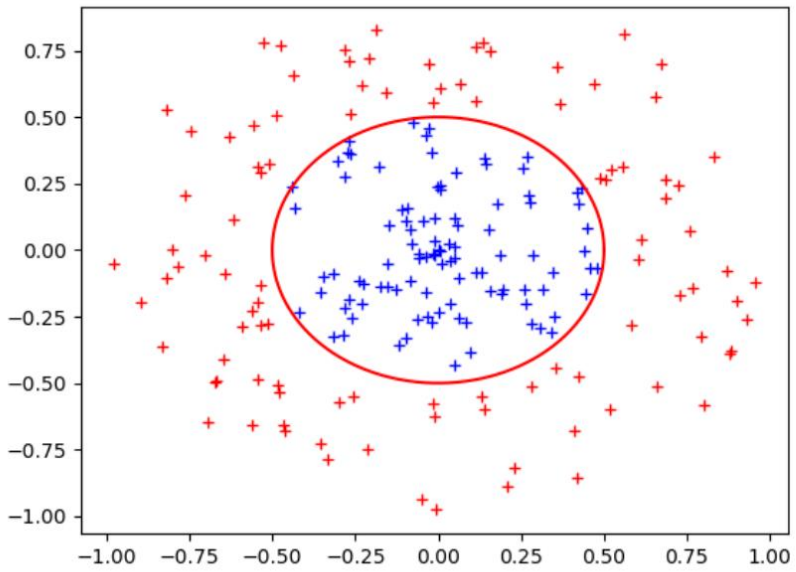
Problem : Don't know the nature of $\varphi(x)$ transformation

Idea : Rewrite Optimization problem without determining $\varphi(x)$

$$\min_{w,\{\xi_n\}} \frac{1}{2}||w||_2 + C\sum_n \xi_n$$
$$s.t.\, y_n[w^T\varphi(x) + b] \geq 1 - \xi_n, \xi_n \geq 0, \forall n$$

Dual Form : $\varphi * \varphi^T = K(x, x')$

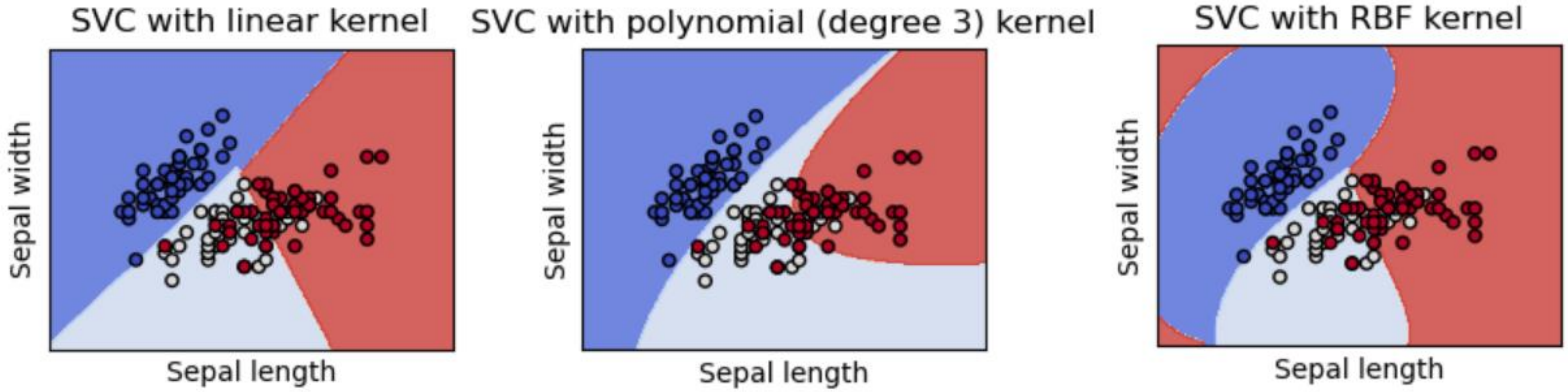
$$\begin{cases} \max L(\lambda) = \sum_{k=1}^{m} \lambda_k - \sum_{k=1}^{m}\sum_{l=1}^{m} \lambda_k \lambda_l y_k y_l \boldsymbol{K}(\boldsymbol{x_k}, \boldsymbol{x_l}) \\ s.t.\, 0 \leq \lambda_k \leq C \end{cases}$$



Kernel SVM

Kernel exemples :

- linear: $k(x, x') = x^T x'$
- polynomial:$k(x, x') = (x^T x' + 1)^d$ of different power $d = 2, 3, \dots$
- gauss-RBG: $k(x, x') = \exp(-\frac{1}{2\sigma}|x - x'|^2)$

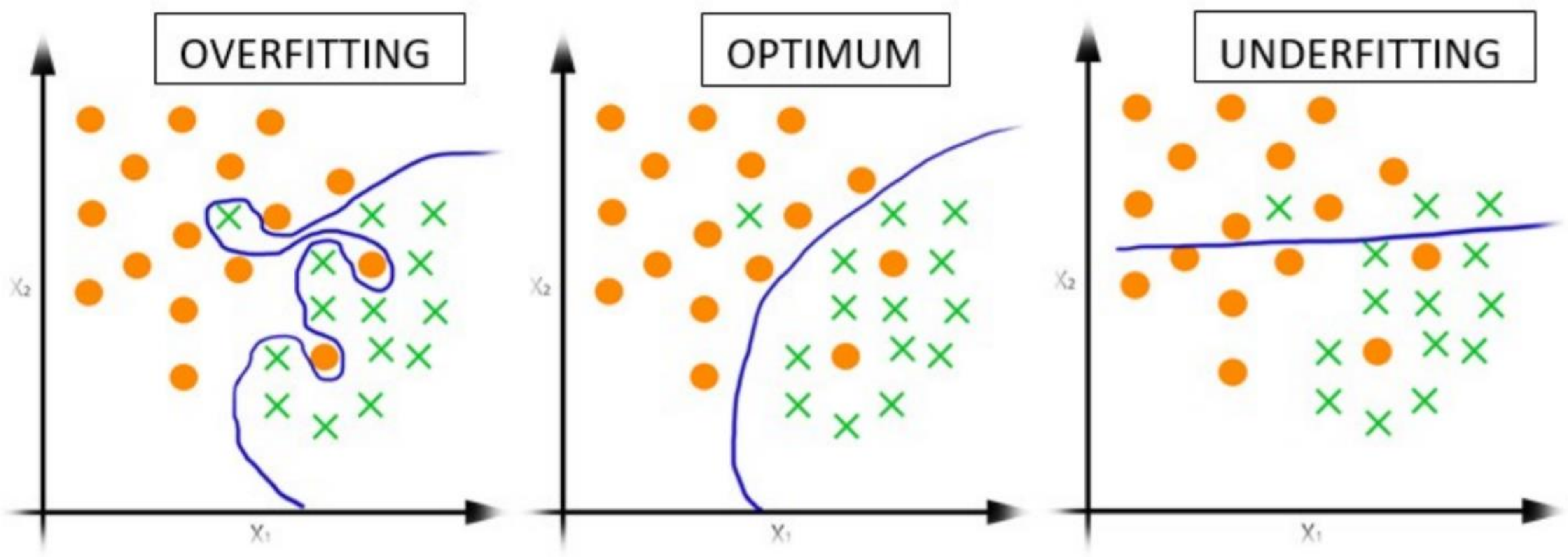


SVM : Advantages & Drawbacks

- Good interpretability of the model
- Effective in high dimensional spaces
- Memory efficient

- **High sensitivity to the noise** in input data
- Slow training on large dataset

Examples of classifiers



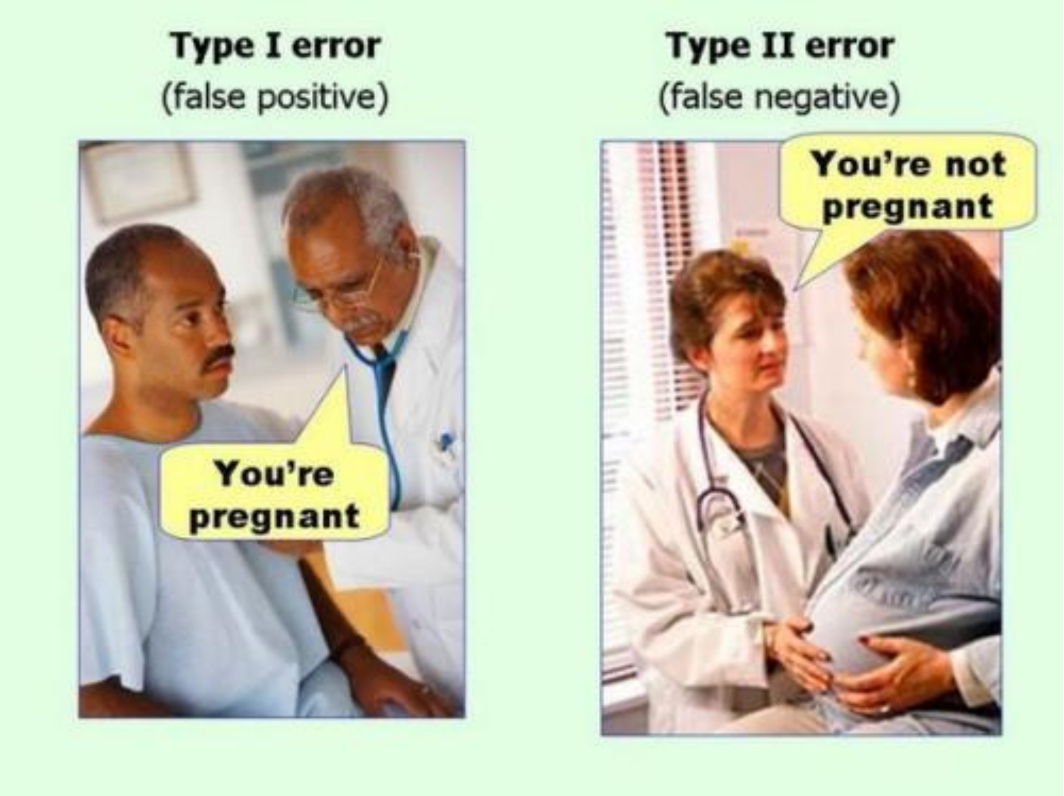
Model evaluation methods

Performance measures for Classification

- Simple Accuracy
- Precision
- Recall
- F-beta mesure
- ROC (and AUC)

Confusion matrix

		Predicted class	
		P	N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)



Metrics

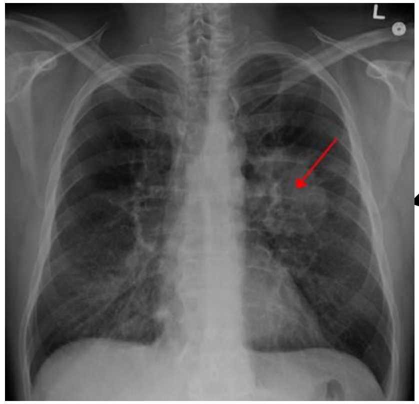
		Predicted class	
		P	*N*
Actual Class	*P*	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

- **Accuracy** $= \frac{(\mathbf{TP}+\mathbf{TN})}{(\mathbf{TP}+\mathbf{TN}+\mathbf{FP}+\mathbf{FN})} = \frac{(\mathbf{TP}+\mathbf{TN})}{\mathbf{N}}$
  (fraction of correct predictions)

Metrics

		Predicted class: P	Predicted class: N
Actual Class	P	True Positives (TP)	False Negatives (FN)
Actual Class	N	False Positives (FP)	True Negatives (TN)

- **Accuracy** $= \dfrac{(\mathrm{TP}+\mathrm{TN})}{(\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN})} = \dfrac{(\mathrm{TP}+\mathrm{TN})}{\mathrm{N}}$
  (fraction of correct predictions)



9990 Not tumor

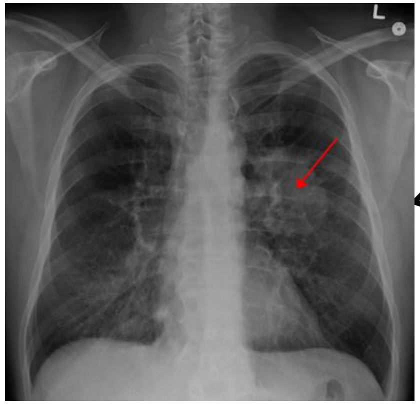
10 Tumor

Metrics

		Predicted class	
		P	*N*
Actual Class	*P*	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

- **Accuracy** $= \frac{(\mathrm{TP}+\mathrm{TN})}{(\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN})} = \frac{(\mathrm{TP}+\mathrm{TN})}{\mathrm{N}}$

  (fraction of correct predictions)



9990 Not tumor

10 Tumor

All is NOT tumor :

Accuracy = 99,9%

Metrics

		Predicted class	
		P	N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

- **Precision** $= \frac{\mathbf{TP}}{(\mathbf{TP+FP})}$
  (fraction of correctly predicted positive values to all values predicted positive)

- **Recall** $= \frac{\mathbf{TP}}{\mathbf{(TP + FN)}}$
  (completeness, fraction of correctly predicted positive values to all positive values)

Metrics

		Predicted class	
		P	N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

- **Precision** $= \frac{\mathbf{TP}}{(\mathbf{TP+FP})}$
  (fraction of correctly predicted positive values to all values predicted positive)

  => **Precision** $= \frac{\mathbf{0}}{\mathbf{0}}$

- **Recall** $= \frac{\mathbf{TP}}{\mathbf{(TP + FN)}}$
  (completeness, fraction of correctly predicted positive values to all positive values)

  => **Recall** $= \frac{\mathbf{0}}{\mathbf{0+10}} = \mathbf{0}$

Example 1&2

Predicted class

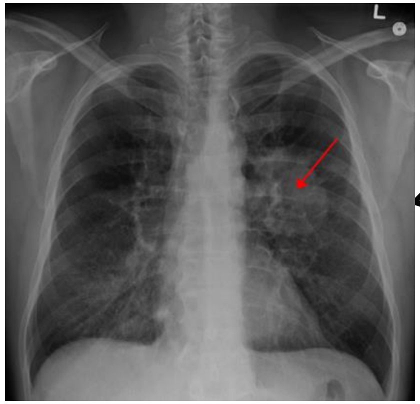
Actual Class		P	N
	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

All is NOT Tumor :

Accuracy = 99,9%

Precision $= \frac{0}{0}$

Recall $= 0$



9990 Not tumor

10 Tumor

Example 1&2

Predicted class

		P	N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

All is NOT Tumor :

Accuracy = 99,9%

$\text{Precision} = \frac{0}{0}$

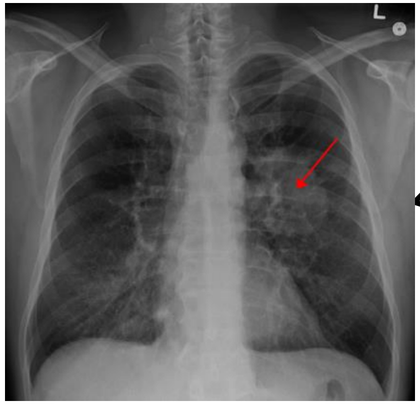
$\text{Recall} = 0$

All is Tumor :

Accuracy = 0,1%

Precision = 0,001

$\text{Recall} = 1$



9990 Not tumor

10 Tumor

Example 3

System 1 :

- Precision$= 70\%$
- Recall $= 60\%$

?

System 2 :

- Precision $= 80\%$
- Recall $= 50\%$

Metrics

- $\mathbf{F_\beta} = \dfrac{\mathbf{1}}{\left(\boldsymbol{\beta} * \frac{\mathbf{1}}{\mathbf{Precision}} + (\mathbf{1} - \boldsymbol{\beta}) * \frac{\mathbf{1}}{\mathbf{Recall}}\right)}$ (greater β, greater importance of Precision)
- $\mathbf{F_1} = \dfrac{\mathbf{2TP}}{\mathbf{(2TP + FP + FN)}}$ (harmonic mean of precision and recall, $\beta = 0.5$)

Example 3

System 1 :
- Precision$= 70\%$
- Recall $= 60\%$

?

System 2 :
- Precision $= 80\%$
- Recall $= 50\%$

$\boldsymbol{\beta = 0.5}$

$F_\beta =$

$F_\beta =$

Example 3

System 1 :

- Precision= 70%
- Recall = 60%

?

System 2 :

- Precision = 80%
- Recall = 50%

$\boldsymbol{\beta = 0.5}$

$F_\beta = 0.6461$ > $F_\beta = 0.6153$

$\boldsymbol{\beta = 0.95}$ (Cancer, pression is more important)

$F_\beta =$

$F_\beta =$

Example 3

System 1 :

- Precision$= 70\%$
- Recall $= 60\%$

?

System 2 :

- Precision $= 80\%$
- Recall $= 50\%$

$\boldsymbol{\beta = 0.5}$

$F_\beta = 0.6461$ > $F_\beta = 0.6153$

$\boldsymbol{\beta = 0.95}$ (Cancer, pression is more important)

$F_\beta = 0.6942$ < $F_\beta = 0.7766$

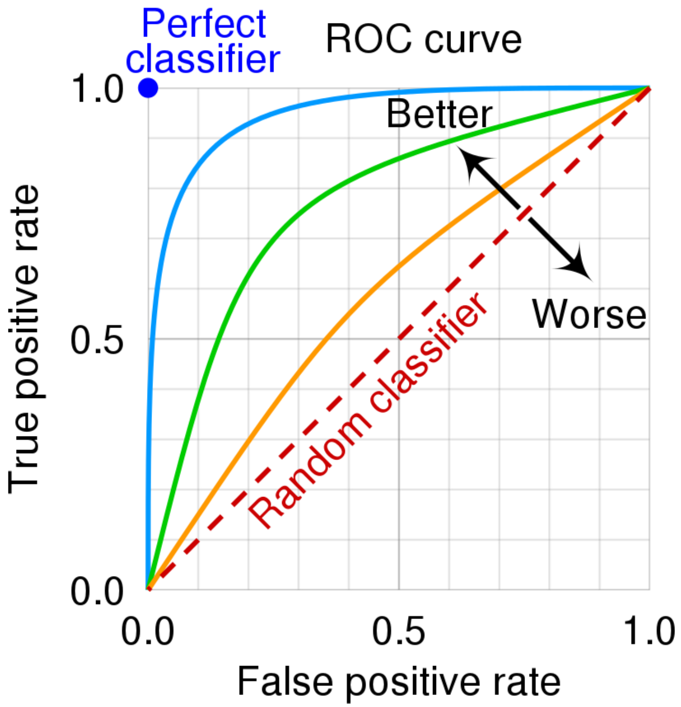
ROC (Receiver operating characteristic)

- **Sensitivity** $= \frac{\mathbf{TP}}{\mathbf{P}} = \frac{\mathbf{TP}}{\mathbf{(TP + FN)}} = \mathbf{TPR}$ (True positive rate, TPR, Recall)
- **Specificity** $= \frac{\mathbf{TN}}{\mathbf{N}} = \frac{\mathbf{TN}}{\mathbf{(FP + TN)}} = \mathbf{1 - FPR}$ (True negative rate, TNR)

		Predicted class	
		P	*N*
Actual Class	*P*	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

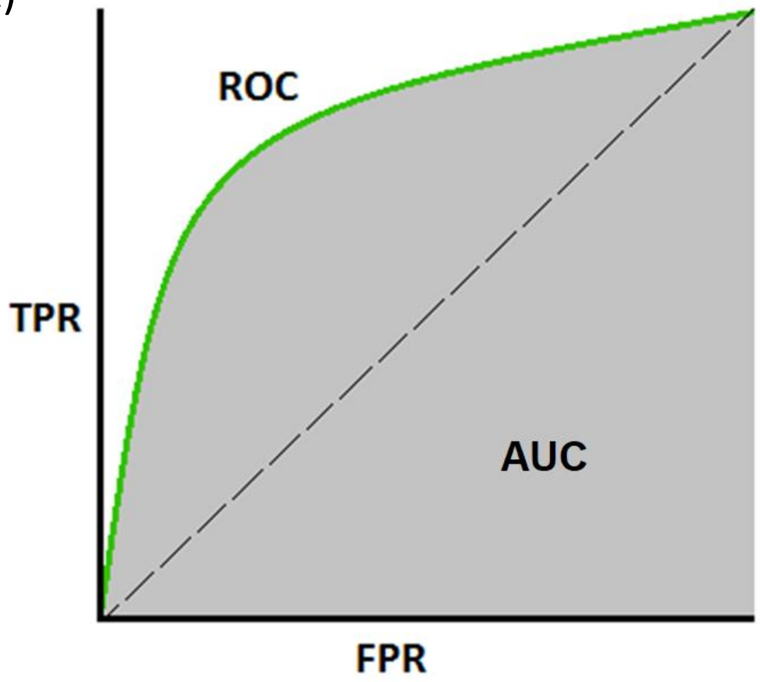
ROC (AUC)

- **Sensitivity** $= \frac{\mathbf{TP}}{\mathbf{P}} = \frac{\mathbf{TP}}{\mathbf{(TP + FN)}} = \mathbf{TPR}$ (True positive rate, TPR, Recall)
- **Specificity** $= \frac{\mathbf{TN}}{\mathbf{N}} = \frac{\mathbf{TN}}{\mathbf{(FP + TN)}} = \mathbf{1 - FPR}$ (True negative rate, TNR)



ROC (AUC)

- **Sensitivity** $= \frac{\mathbf{TP}}{\mathbf{P}} = \frac{\mathbf{TP}}{\mathbf{(TP + FN)}} = \mathbf{TPR}$ (True positive rate, TPR, Recall)
- **Specificity** $= \frac{\mathbf{TN}}{\mathbf{N}} = \frac{\mathbf{TN}}{\mathbf{(FP + TN)}} = \mathbf{1 - FPR}$ (True negative rate, TNR)



Sources

- MIT course “Introduction to Computational Thinking and Data Science” (Prof. Eric Grimson, Prof. John Guttag)
- Open Machine Learning Course (by Yury Kashnitsky, mlcourse.ai)
- YouTube lections “Algorithms and Concepts” (by CodeEmporium)