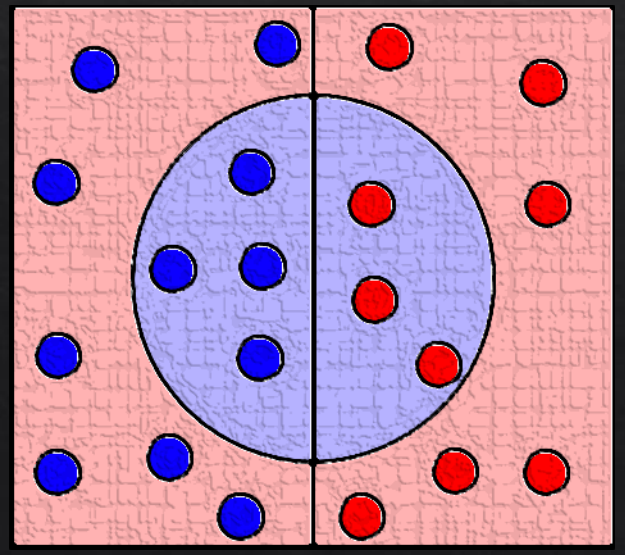
Supervised Learning – Performance Evaluation

Machine Learning Methods – Lecture 4



June 2021

fixel your pixels

fixelalgorithms.gitlab.io

Or Yair

Multiclass Classification (K–NN)

- Consider the multi-class classification task:

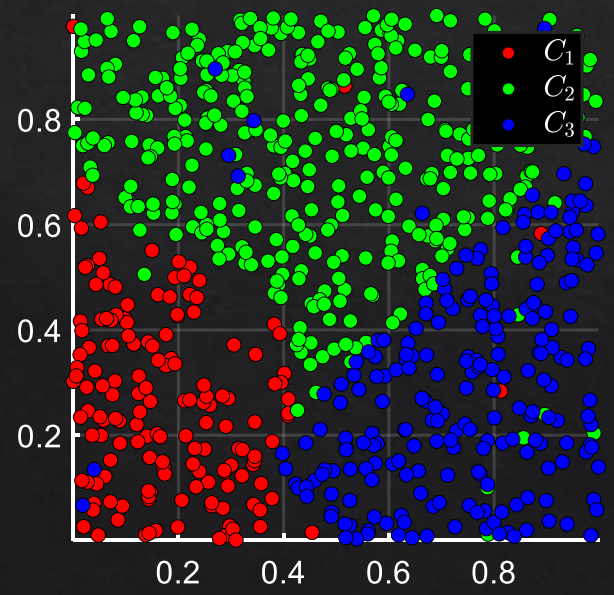
$$\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}, \qquad y_i \in \mathcal{Y} = \{C_1, C_2, \ldots, C_K\}$$

- K–NN and other classifiers can be immediately generalized into multiclass classification.

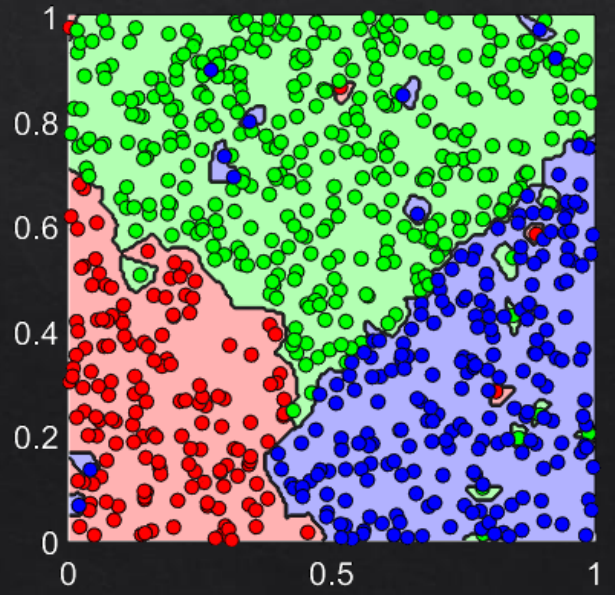
K–NN

$$f_{1\text{–NN}}(\boldsymbol{x}) = y_{\hat{i}(\boldsymbol{x})}$$

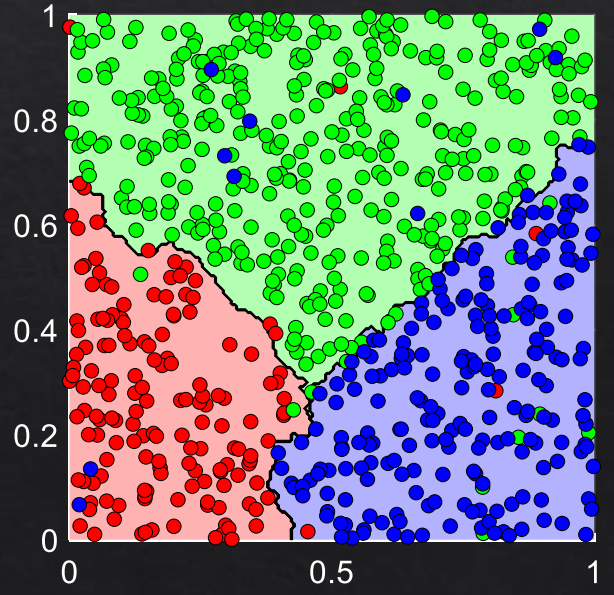
$$\hat{i}(\boldsymbol{x}) = \arg\min_i d(\boldsymbol{x}, \boldsymbol{x}_i)$$



1–NN



5–NN

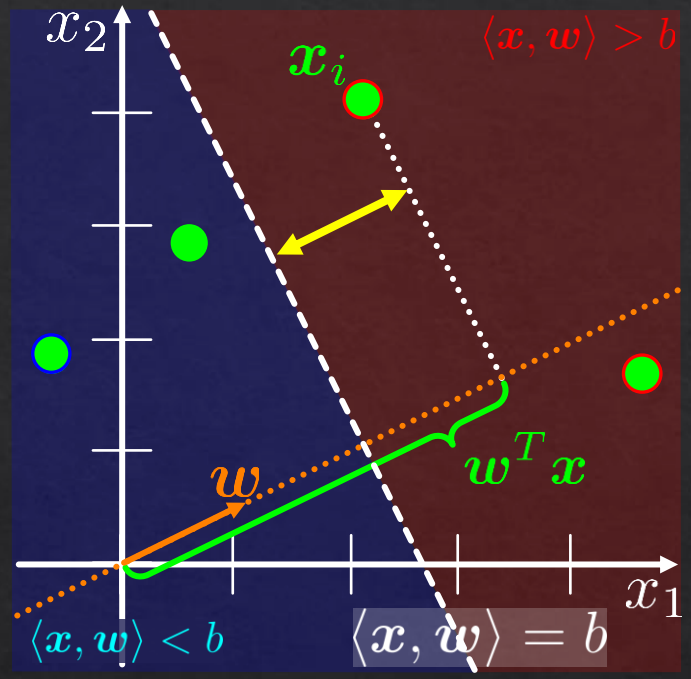


What about binary linear classifiers (such as SVM)?

Classification Score (Confidence)

- A linear classifier:

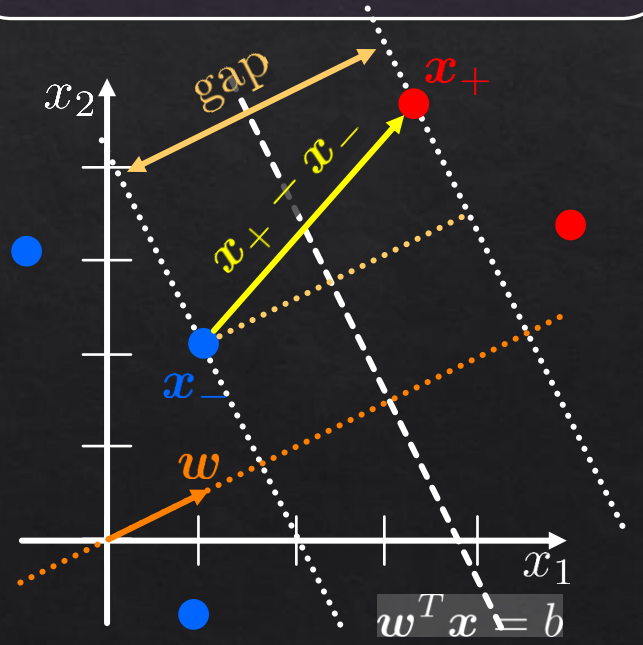
$$f(\boldsymbol{x}) = \text{sign}\,\underbrace{(\boldsymbol{w}^T\boldsymbol{x} - b)}_{\text{distance from the decision boundary}}$$



$$\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$$
$$y_i \in \mathcal{Y} = \{C_1, C_2, \ldots, C_K\}$$

SVM

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_i \xi_i$$
$$\xi_i := \max\{0, 1 - y_i(\boldsymbol{w}^T\boldsymbol{x}_i - b)\}$$

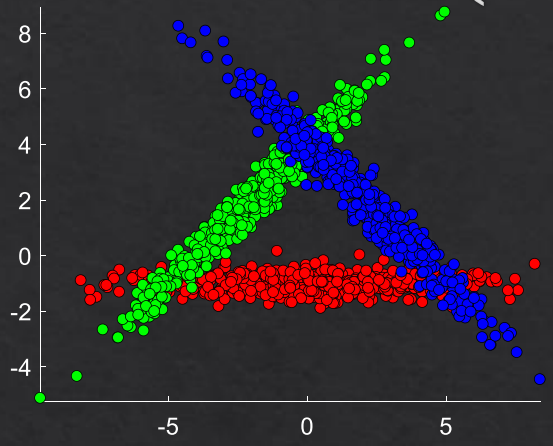


- The classification score of linear classifiers:

$$\text{classification score} := \boldsymbol{w}^T\boldsymbol{x} - b$$

- Most classifiers has some classification score (confidence).

Multiclass Classification (Coding)

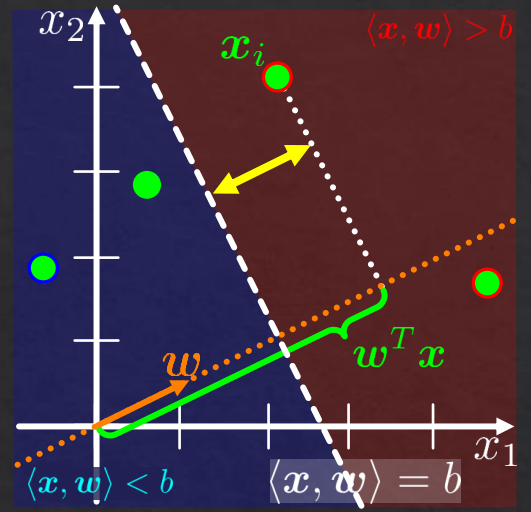


$$y_i \in \mathcal{Y} = \{C_1, C_2, \ldots, C_K\}$$

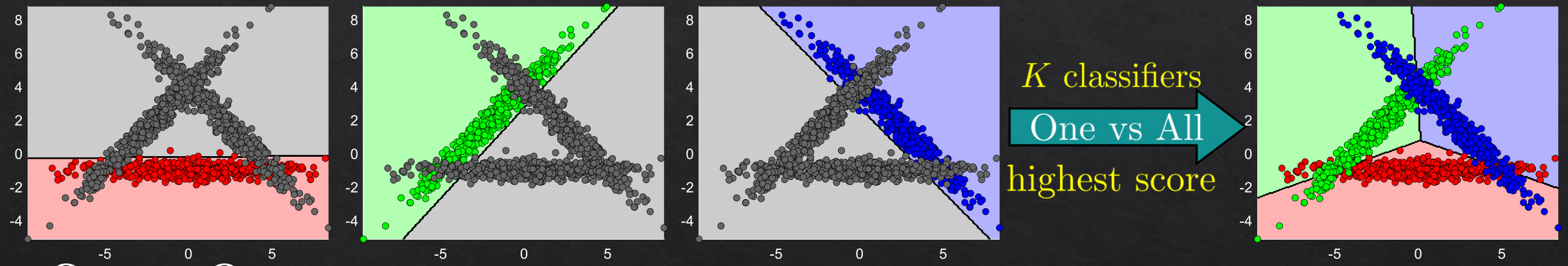
$$f(\boldsymbol{x}) = \text{sign}\underbrace{\left(\boldsymbol{w}^T\boldsymbol{x} - b\right)}$$

distance from the decision boundary

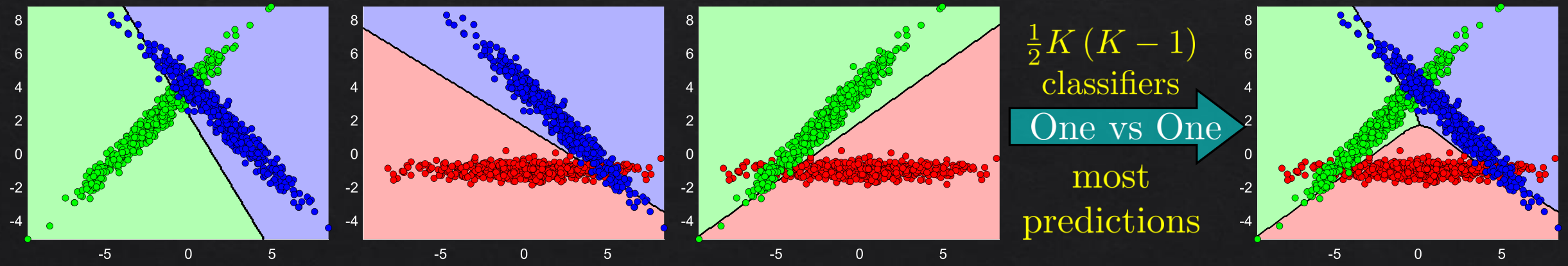
classification score $:= \boldsymbol{w}^T\boldsymbol{x} - b$



- **One vs All:**



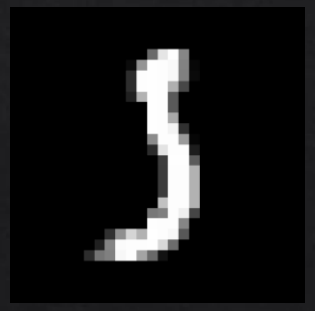
- **One vs One:**



MNIST

- The MNIST data set:

$$\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{60{,}000} \qquad \mathcal{D}_{\text{test}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{10{,}000}$$



$$\in \mathbb{R}^{28\times 28} = \mathbb{R}^{784}$$

$$\boldsymbol{x}_i \in \mathbb{R}^{784}$$

$$y_i \in \{0, 1, 2, \ldots, 9\}$$

Confusion Matrix

- Training an SVM classifier on $N = 3,000$ samples with $C = 0.0004$:

Train accuracy: 80.8333%

True Class \ Predicted Class	0	1	2	3	4	5	6	7	8	9		
0	278		1	3		7	12	1	14	3	87.1%	12.9%
1		344	1	2			1		3	1	97.7%	2.3%
2	2	12	238	4	6		8	7	21	4	78.8%	21.2%
3	2	9	12	247	1	4	3	2	19	7	80.7%	19.3%
4	1	7			144		5			105	55.0%	45.0%
5	3	28		41	1	142	10	1	13	13	56.3%	43.7%
6		11	4			10	275	1	4		90.2%	9.8%
7	1	11	2		2			257	2	38	82.1%	17.9%
8	1	15	3	16			5	1	234	20	79.3%	20.7%
9	2	8	1	7	4	2		1	3	266	90.5%	9.5%
	95.9%	77.3%	90.8%	77.2%	91.1%	86.1%	86.2%	94.8%	74.8%	58.2%		
	4.1%	22.7%	9.2%	22.8%	8.9%	13.9%	13.8%	5.2%	25.2%	41.8%		

Test accuracy: 79.9333%

True Class \ Predicted Class	0	1	2	3	4	5	6	7	8	9		
0	253		5	2		14	9	1	6	1	86.9%	13.1%
1		331							8		97.6%	2.4%
2	2	34	223	7	4		7	4	20	4	73.1%	26.9%
3		4	6	250		10	1	7	17	4	83.6%	16.4%
4	1	7	1		164		3		3	103	58.2%	41.8%
5	3	23	1	57	1	137	16	4	13	19	50.0%	50.0%
6	4	5	8		3	9	260	2		1	89.0%	11.0%
7	1	22	7	1				269	6	25	81.3%	18.7%
8	3	10		17		3	4	3	243	14	81.8%	18.2%
9	3	4	3	2	3			3	4	268	92.4%	7.6%
	93.7%	75.2%	87.8%	74.4%	93.7%	79.2%	86.7%	91.8%	75.9%	61.0%		
	6.3%	24.8%	12.2%	25.6%	6.3%	20.8%	13.3%	8.2%	24.1%	39.0%		

SVM

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_i \xi_i$$

$$\xi_i := \max\left\{0, 1 - y_i\left(\boldsymbol{w}^T\boldsymbol{x}_i - b\right)\right\}$$

K-NN

$$f_{1\text{–NN}}(\boldsymbol{x}) = y_{\hat{i}(\boldsymbol{x})}$$

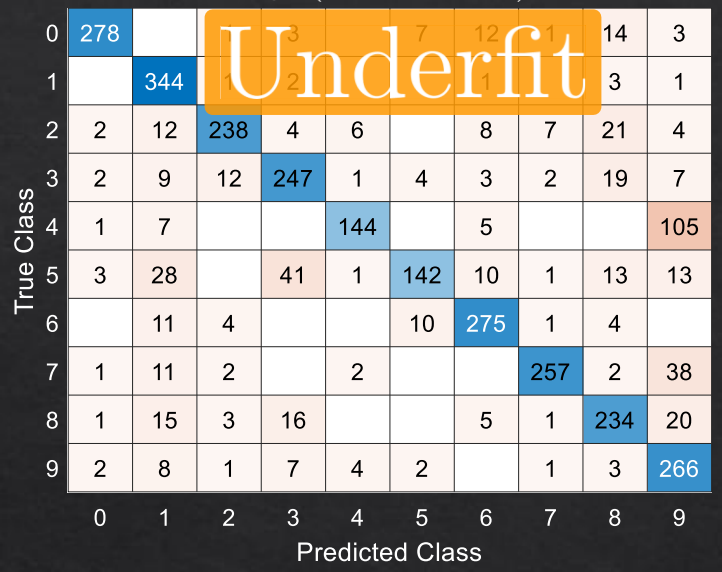
$$\hat{i}(\boldsymbol{x}) = \arg\min_i d(\boldsymbol{x}, \boldsymbol{x}_i)$$

C and K are hyper-parameters

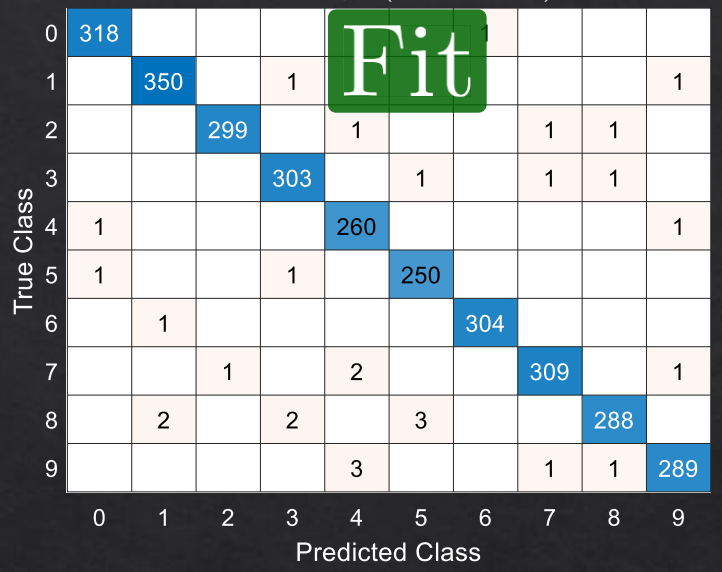
What is the optimal value of C?

Confusion Matrix (SVM and the hyperparameter C)

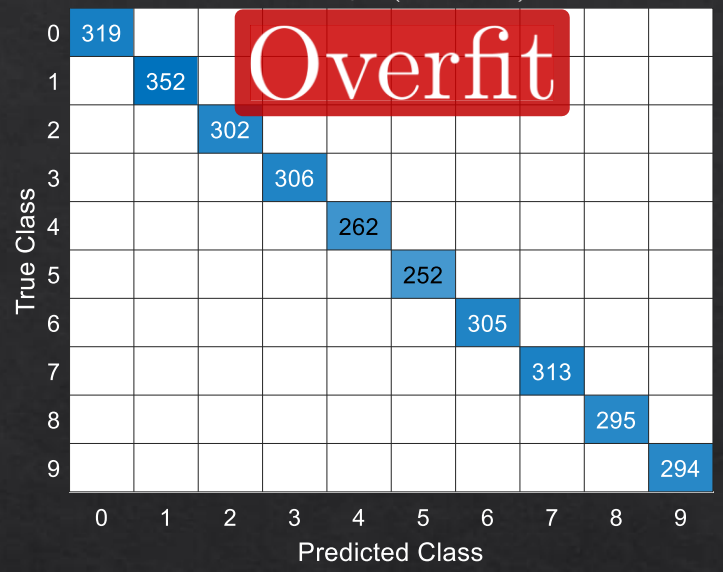
Train accuracy ($C = 0.004$): 80.8333%



Train accuracy ($C = 0.1$): 99%



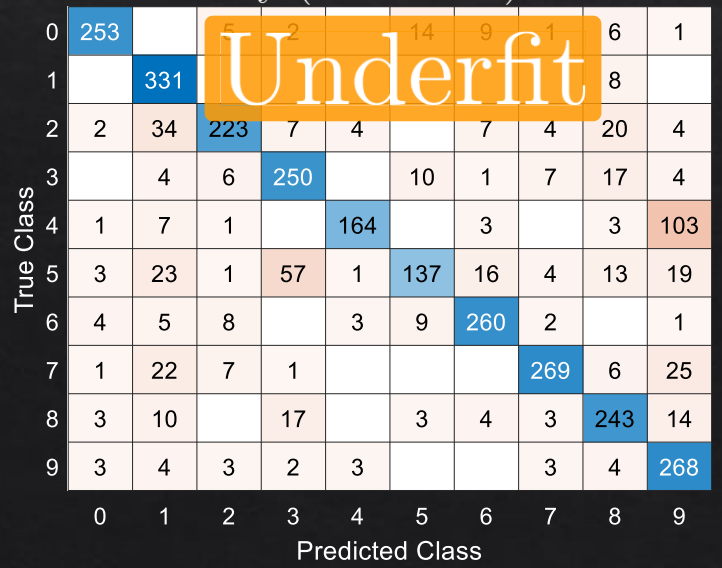
Train accuracy ($C = 1$): 100%



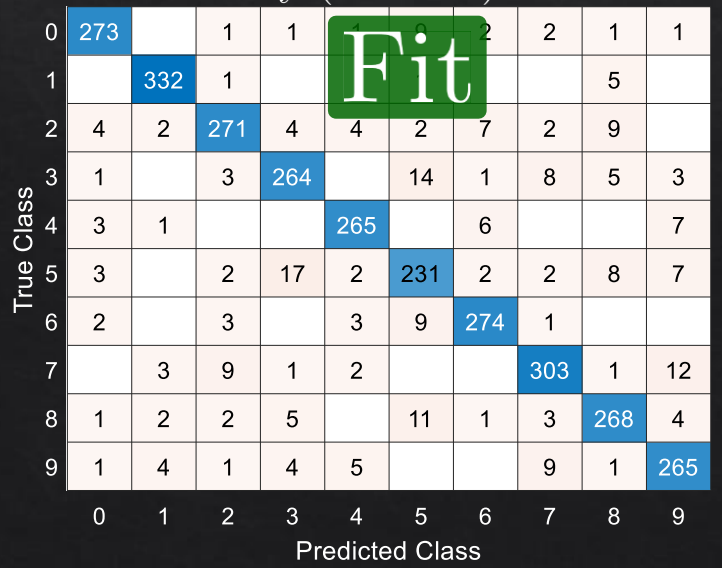
What is the optimal value of C?

good training performance $\Longrightarrow$ good test performance

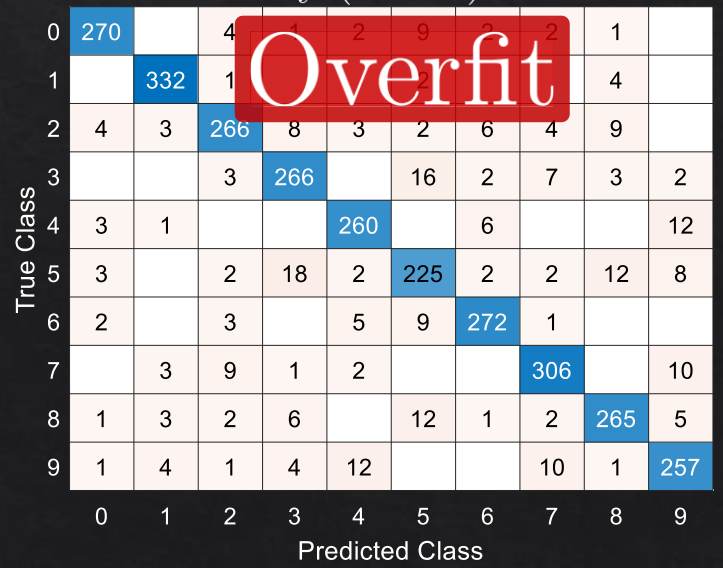
Test accuracy ($C = 0.004$): 79.9333%



Test accuracy ($C = 0.1$): 91.5333%

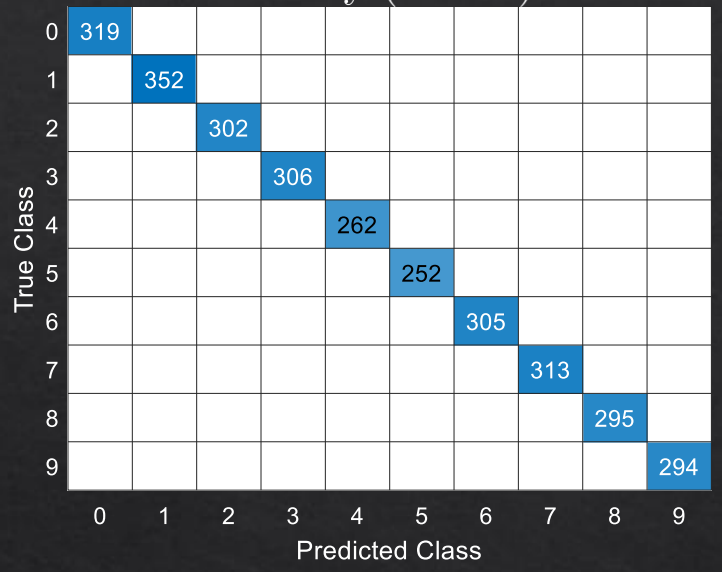


Test accuracy ($C = 1$): 90.6333%

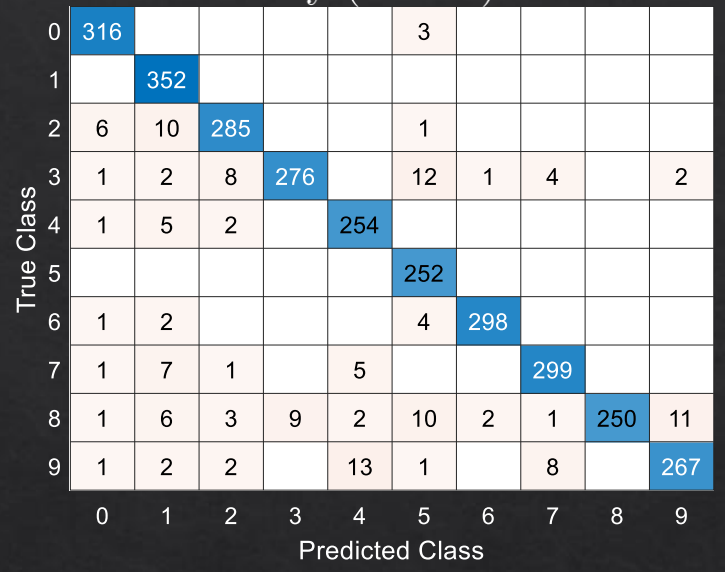


Confusion Matrix (K–NN and the hyperparameter K)

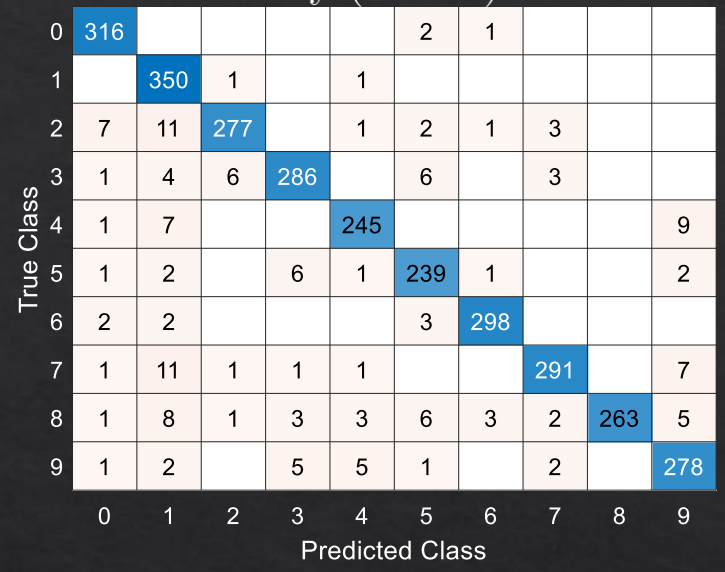
Train accuracy ($K = 1$): 100%



Train accuracy ($K = 2$): 94.9667%



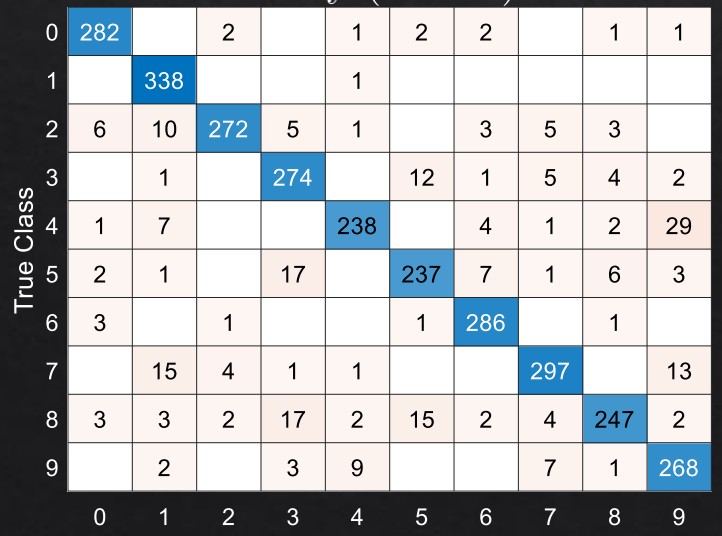
Train accuracy ($K = 5$): 94.7667%



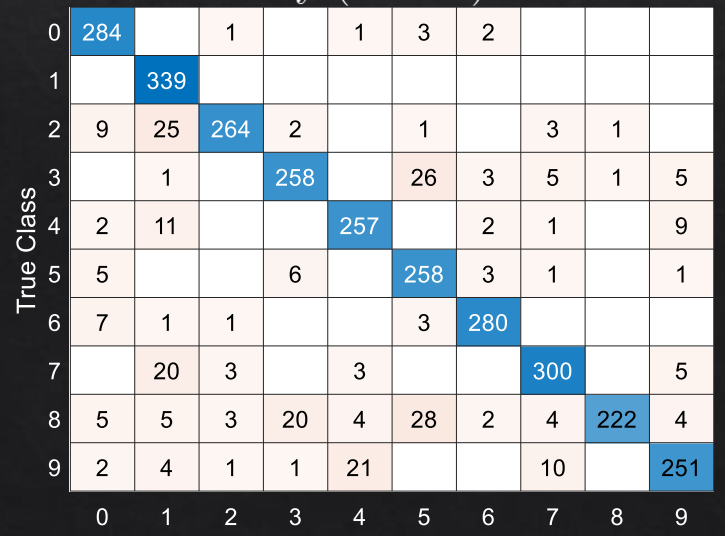
What is the optimal value of K?

good training performance $\Longrightarrow$ good test performance

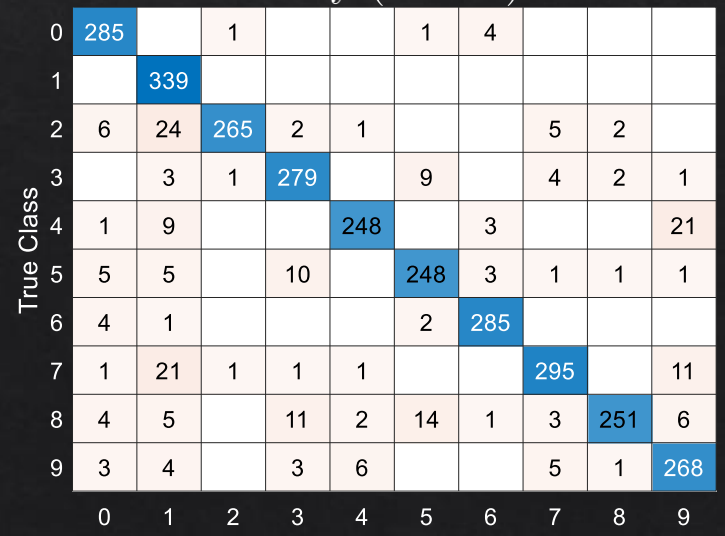
Test accuracy ($K = 1$): 91.3%



Test accuracy ($K = 2$): 90.4333%



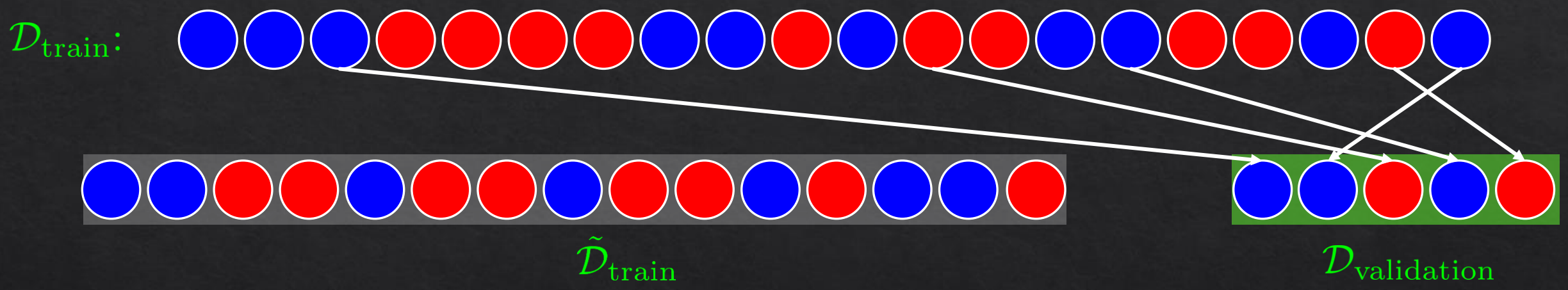
Test accuracy ($K = 5$): 92.1%



Validation Set

- A good practice is to split the training set into:
  - A (smaller) training set $\tilde{\mathcal{D}}_{\text{train}}$.
  - A validation set $\mathcal{D}_{\text{validation}}$.

$$\tilde{\mathcal{D}}_{\text{train}} \sqcup \mathcal{D}_{\text{validation}} = \mathcal{D}_{\text{train}}$$

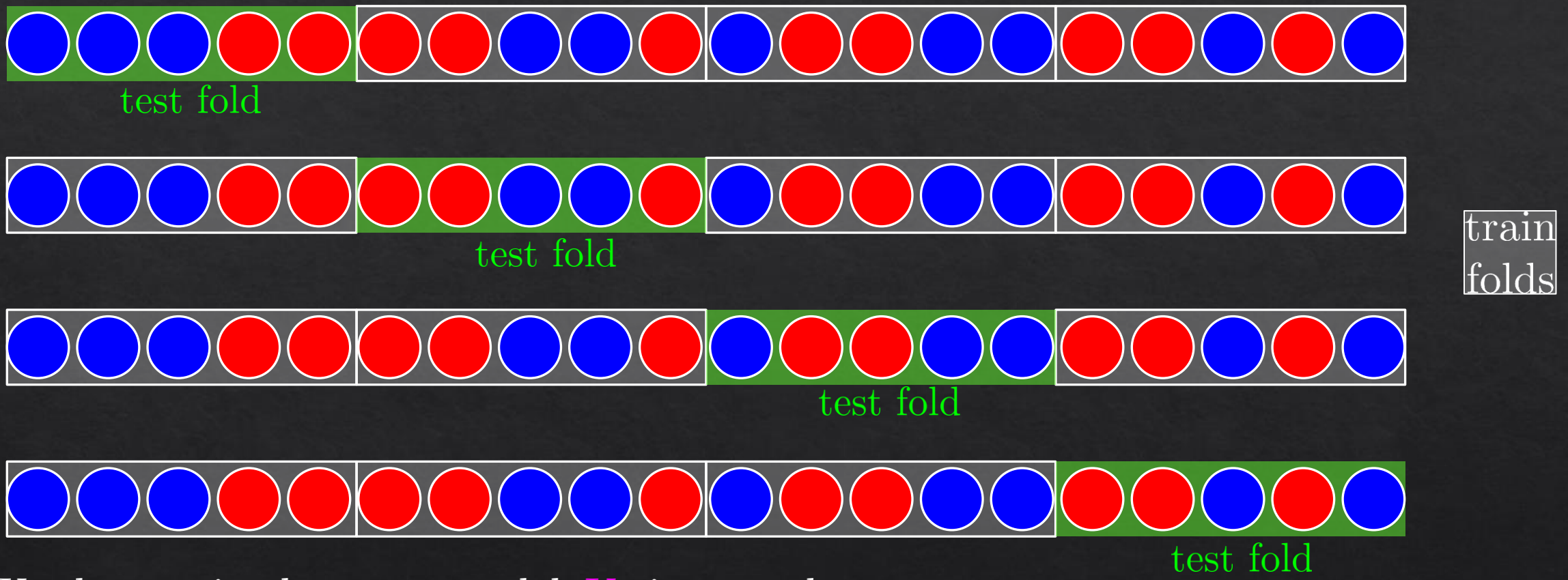


- Then, we check how different classifiers and different values of hyperparameters perform on the validation set.

- Make sure that $\tilde{\mathcal{D}}_{\text{train}}$ and $\mathcal{D}_{\text{validation}}$ are shuffled and balanced.

Cross Validation (K Fold)

- We can also split the data into K folds:



- We then train the same model K times and compute the average performance of the K folds.

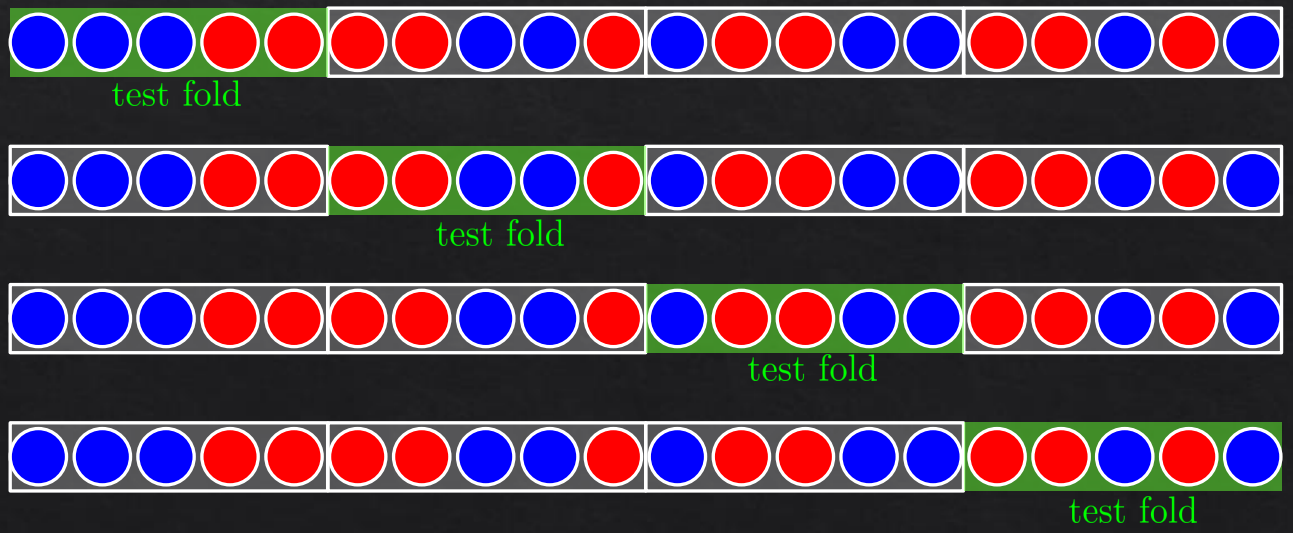
- The case $K = N$ is known as "leave one out cross validation".

Confusion Matrix and Cross-Validation – Example

Confusion and Cross-validation

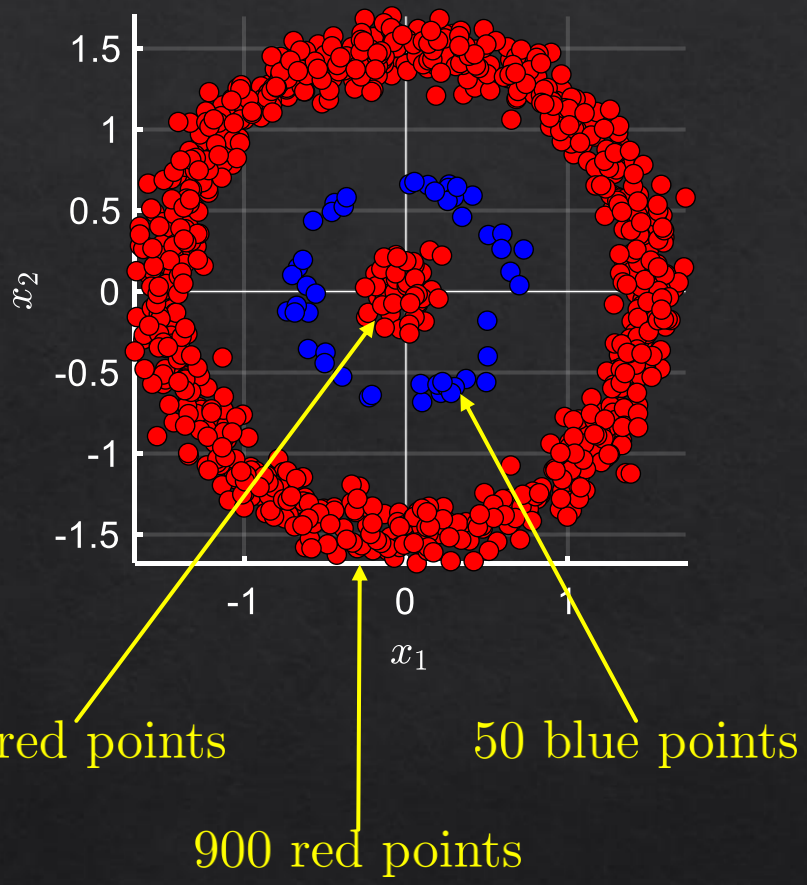
Test accuracy: 79.9333%

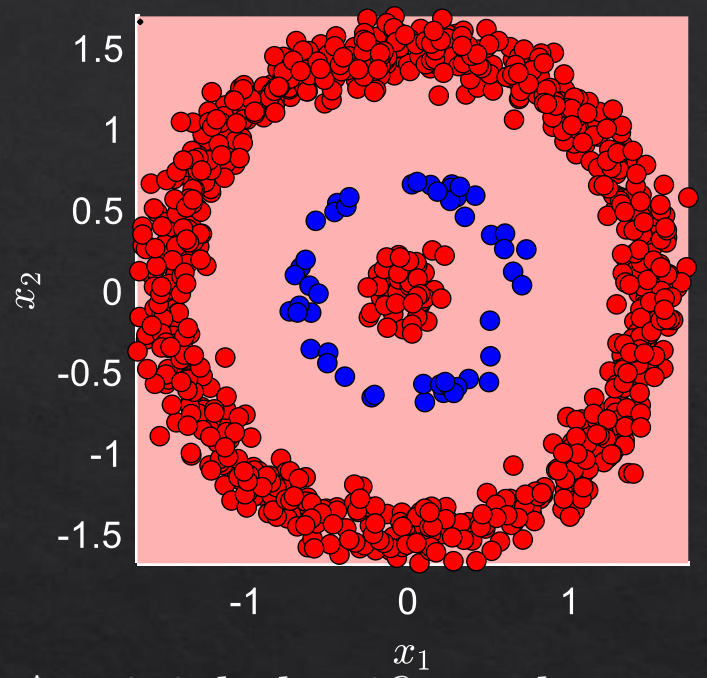
True Class \ Predicted Class	0	1	2	3	4	5	6	7	8	9	
0	253		5	2		14	9	1	6	1	86.9% 13.1%
1		331							8		97.6% 2.4%
2	2	34	223	7	4		7	4	20	4	73.1% 26.9%
3		4	6	250		10	1	7	17	4	83.6% 16.4%
4	1	7	1		164		3		3	103	58.2% 41.8%
5	3	23	1	57	1	137	16	4	13	19	50.0% 50.0%
6	4	5	8		3	9	260	2		1	89.0% 11.0%
7	1	22	7	1				269	6	25	81.3% 18.7%
8	3	10		17		3	4	3	243	14	81.8% 18.2%
9	3	4	3	2	3			3	4	268	92.4% 7.6%
	93.7%	75.2%	87.8%	74.4%	93.7%	79.2%	86.7%	91.8%	75.9%	61.0%	
	6.3%	24.8%	12.2%	25.6%	6.3%	20.8%	13.3%	8.2%	24.1%	39.0%	



Precision and Recall – Introduction

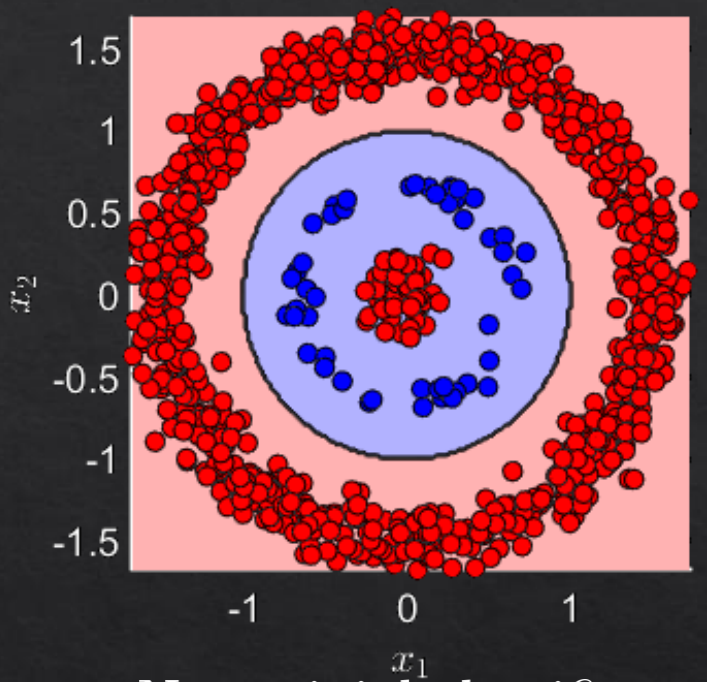
- Consider the following imbalanced data ($N = 1,000$ points):





A trivial classifier: always red

accuracy: 95%

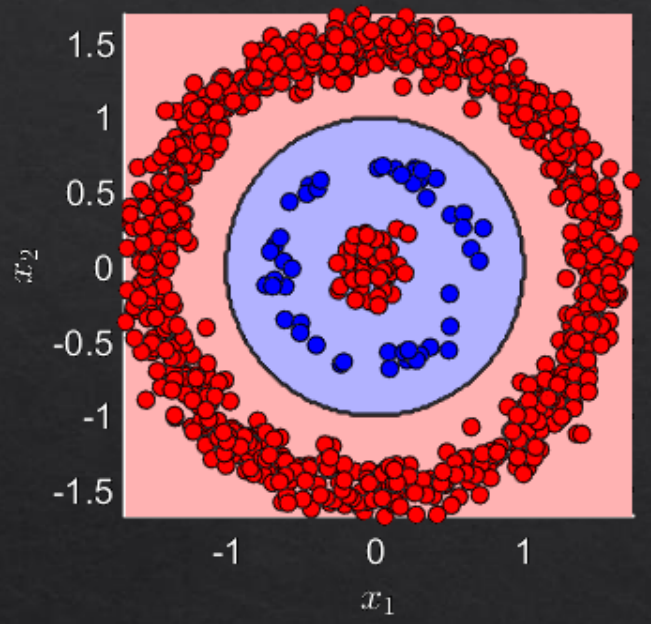


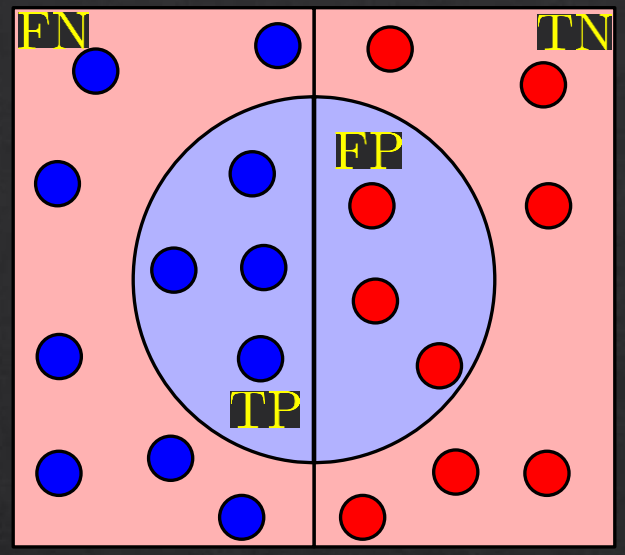
Non trivial classifier

accuracy: 95%

- For imbalanced data, accuracy might be misleading.
- Precision and recall might be more informative.

Precision and Recall





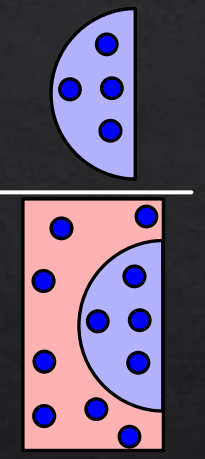
positive

negative

TP – True Positive
FN – False Negative
FP – False Positive
TN – True Negative

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{50}{50 + 0} = 1$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} =$$



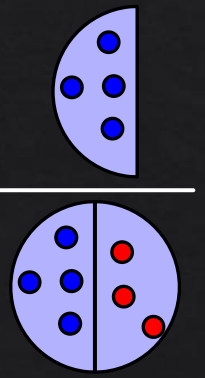
how many relevant items are selected

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{50}{50 + 50} = \frac{1}{2}$$

- The F_1 measure, combines the two into a single number:

$$F_1 = 2\frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} =$$

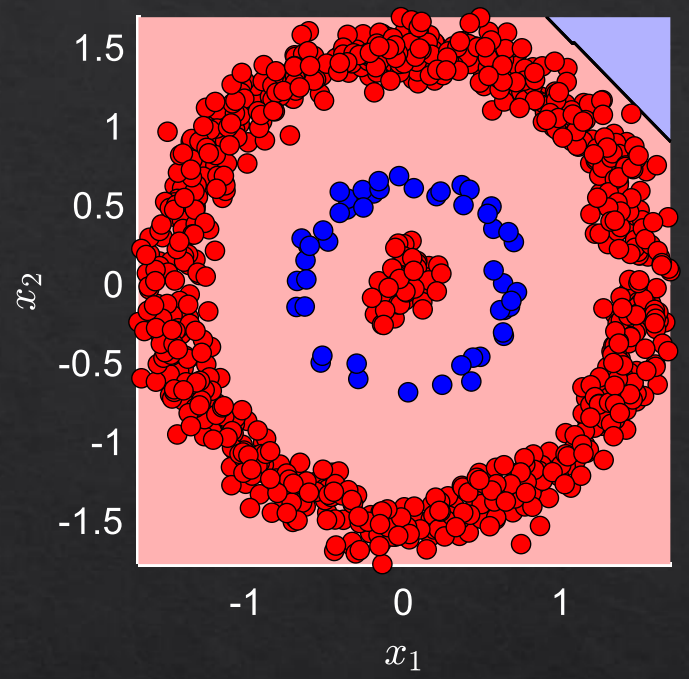


how many selected items are relevant

Receiver Operating Characteristic (ROC) – Introduction

- A linear classifier:

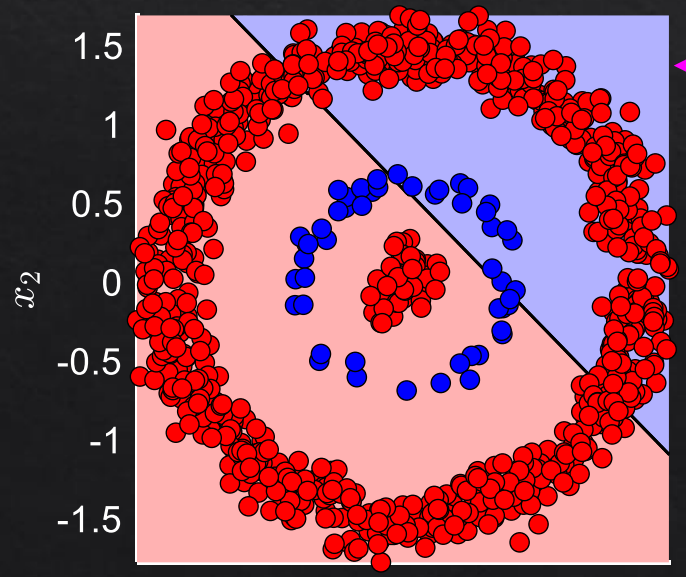
$$f(\boldsymbol{x}) = \text{sign}\underbrace{(\boldsymbol{w}^T\boldsymbol{x} - b)}_{\text{distance from the decision boundary}} = \begin{cases} 1 & \boldsymbol{w}^T\boldsymbol{x} - b \geq 0 \\ -1 & \boldsymbol{w}^T\boldsymbol{x} - b < 0 \end{cases}$$



- We can control the sensitivity of the classifier by changing the threshold:

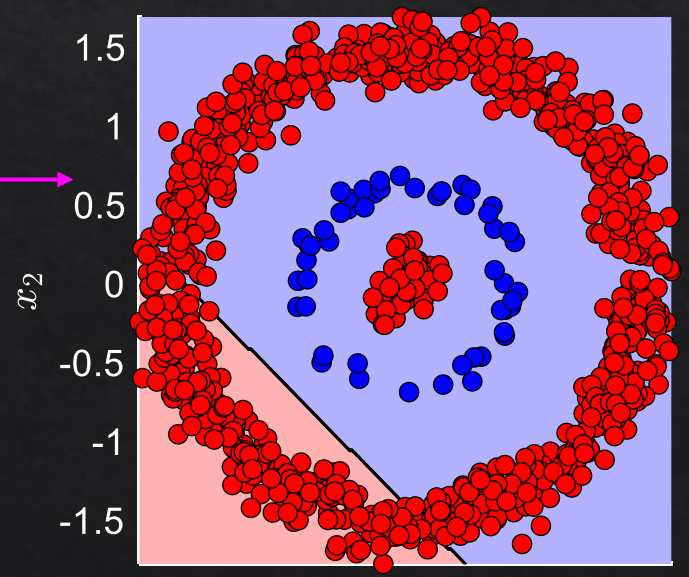
$$\tilde{f}(\boldsymbol{x}) = \begin{cases} 1 & \boldsymbol{w}^T\boldsymbol{x} - b \geq \alpha \\ -1 & \boldsymbol{w}^T\boldsymbol{x} - b < \alpha \end{cases}$$

$$\alpha \in (-\infty, \infty)$$
$$\alpha = \infty \implies \tilde{f}(\boldsymbol{x}) = -1$$
$$\alpha = -\infty \implies \tilde{f}(\boldsymbol{x}) = 1$$



$\tilde{f}(\boldsymbol{x})$ with $\alpha = -2$

$\tilde{f}(\boldsymbol{x})$ with $\alpha = -4$

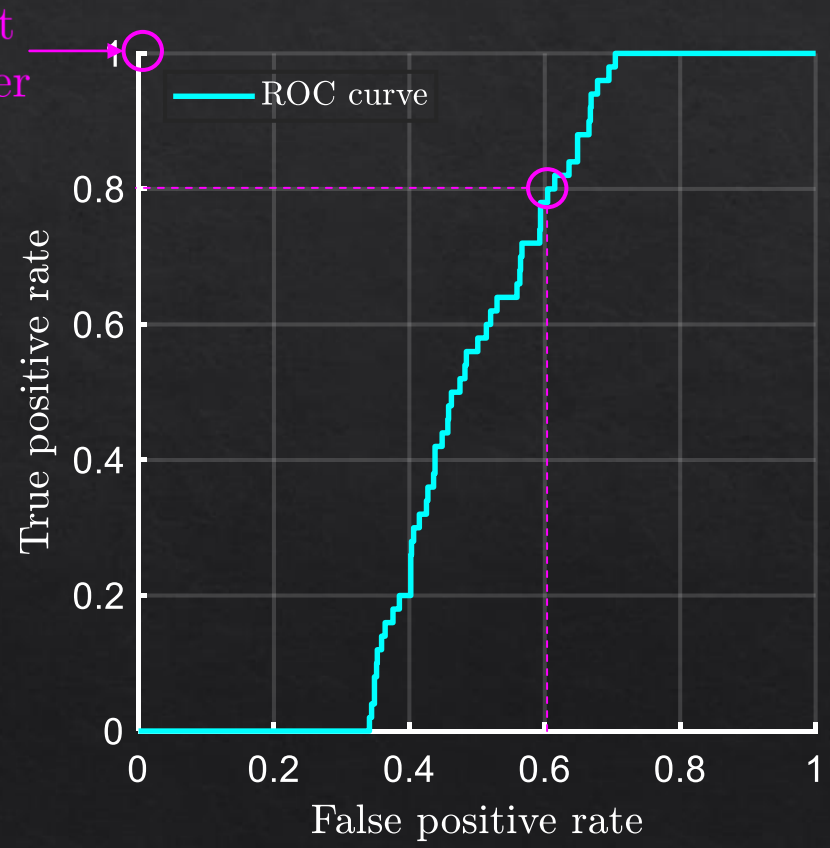


α controls the trade-off between false positive rate and true positive rate

ROC Curve

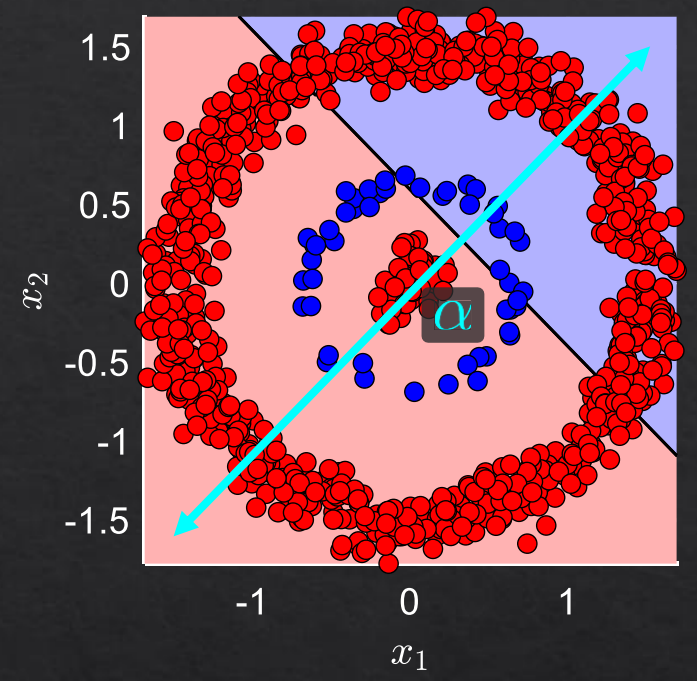
$$\tilde{f}(\boldsymbol{x}) = \begin{cases} 1 & \boldsymbol{w}^T\boldsymbol{x} - b \geq \alpha \\ -1 & \boldsymbol{w}^T\boldsymbol{x} - b < \alpha \end{cases}$$

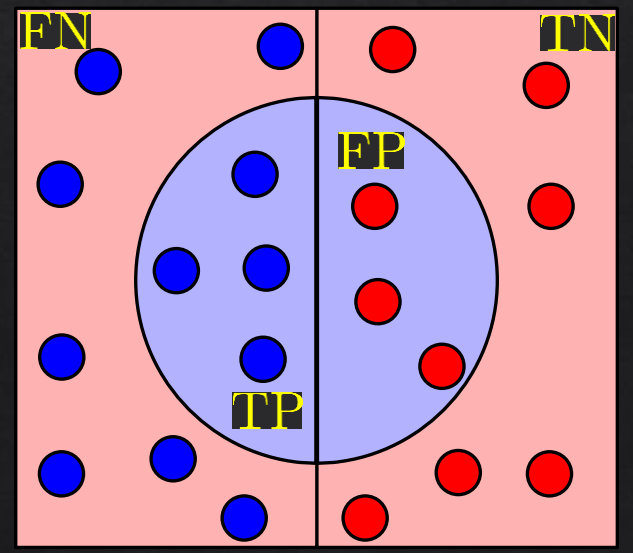
$$\alpha \in (-\infty, \infty)$$
$$\alpha = \infty \implies \tilde{f}(\boldsymbol{x}) = -1$$
$$\alpha = -\infty \implies \tilde{f}(\boldsymbol{x}) = 1$$



$$\text{TP rate} = \frac{\text{TP}}{\text{TP+FN}} = \text{Recall}$$

$$\text{FP rate} = \frac{\text{FP}}{\text{FP+TN}}$$

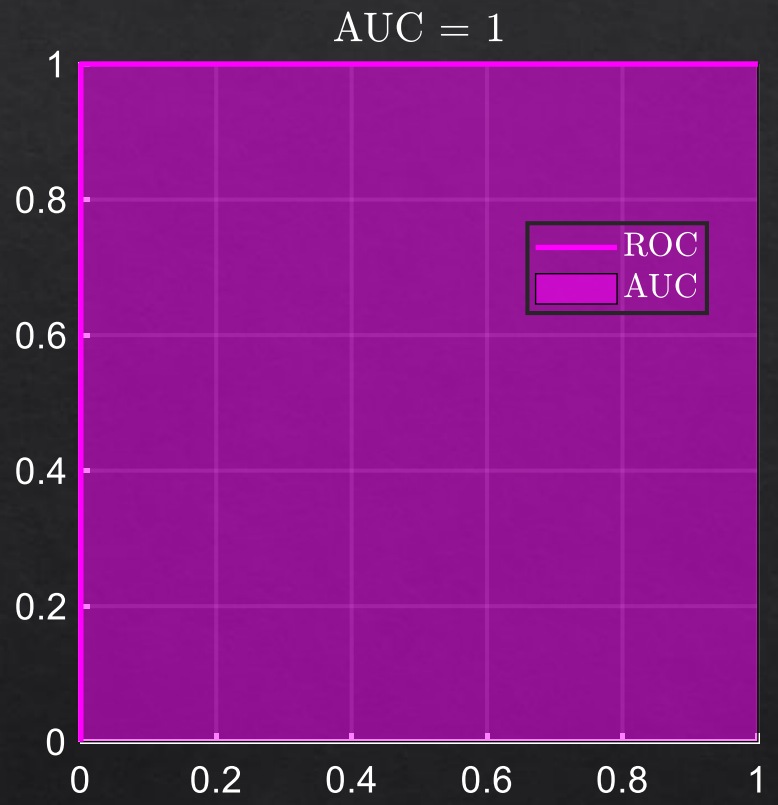




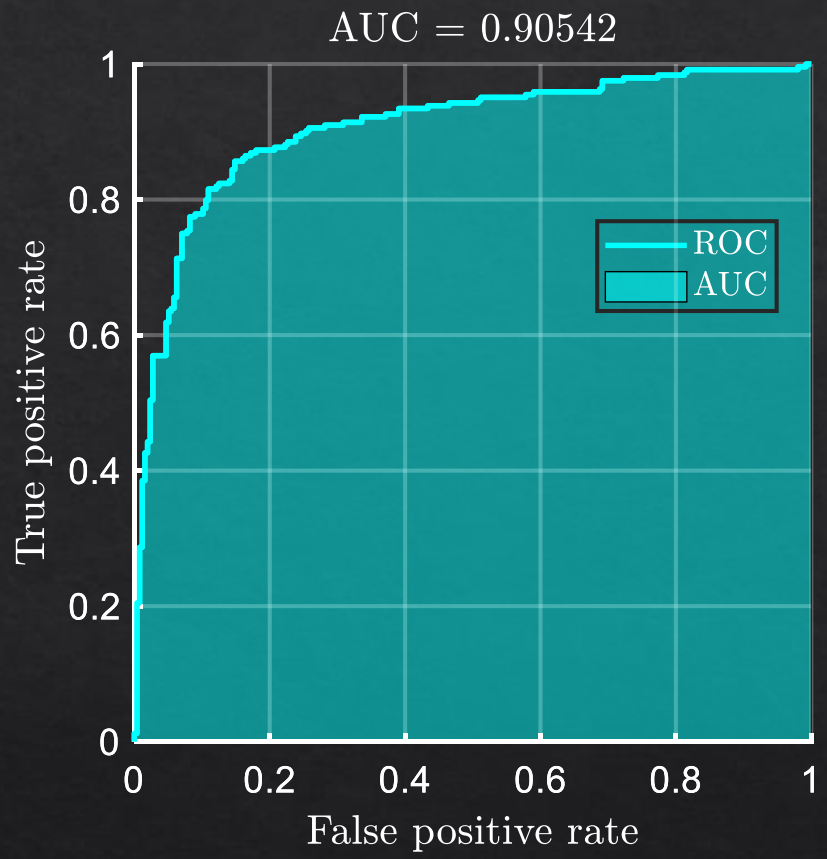
the curve is obtain by swiping over $\alpha \in (-\infty, \infty)$

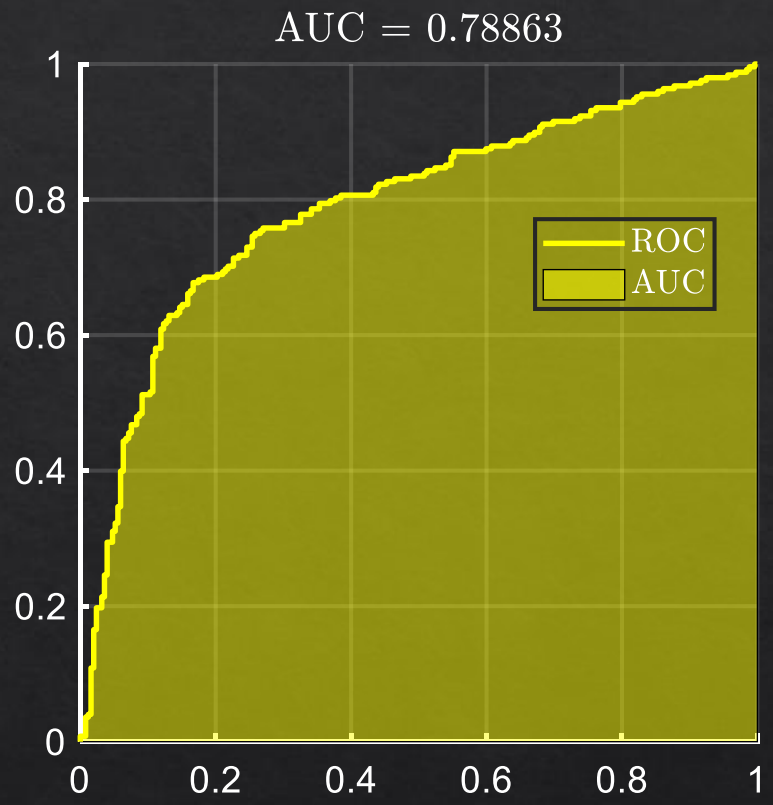
Area Under the Curve (AUC)

- The AUC is the area under the ROC curve:



optimal classifier



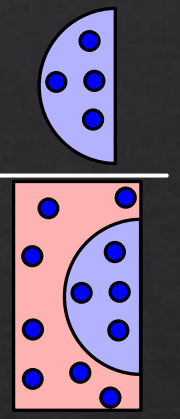


a poorer model

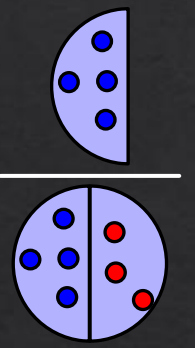
- We can use the AUC as a score for a model.

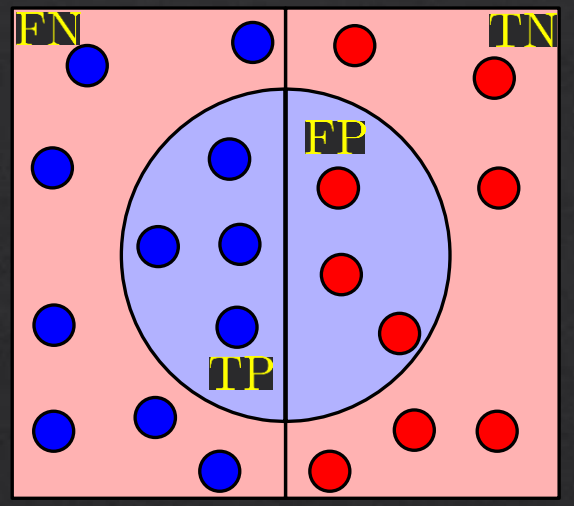
Precision, Recall, ROC, and AUC – Example

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} =$$

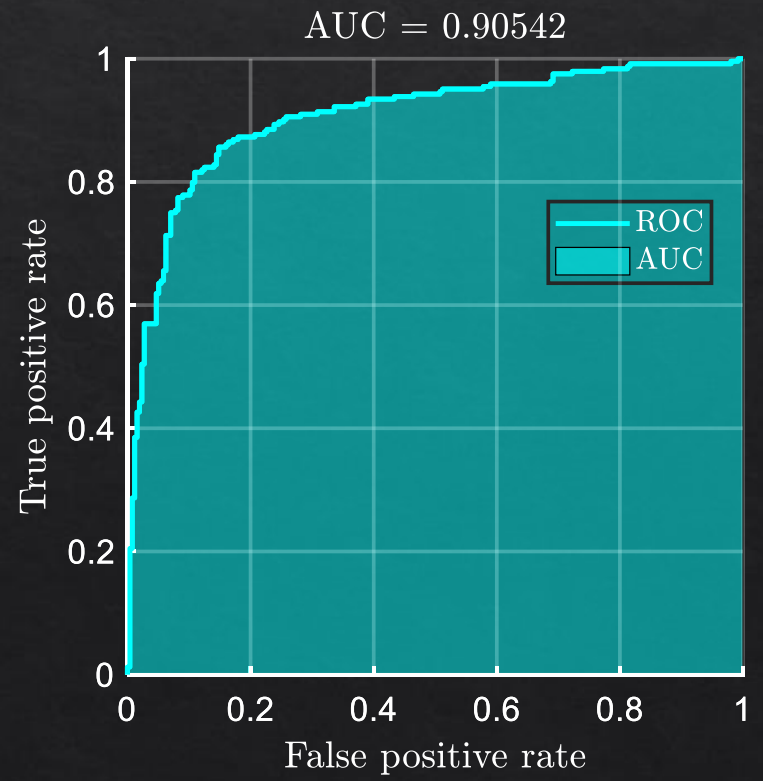


$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} =$$





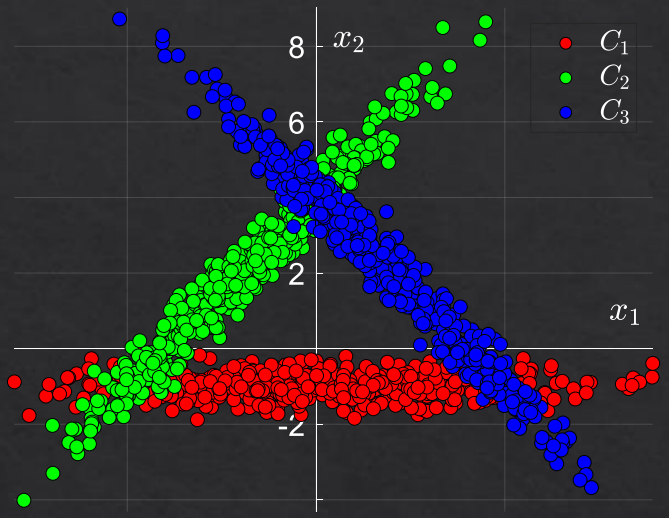
Precision, Recall, ROC, and AUC



Weighted Loss (Cost)

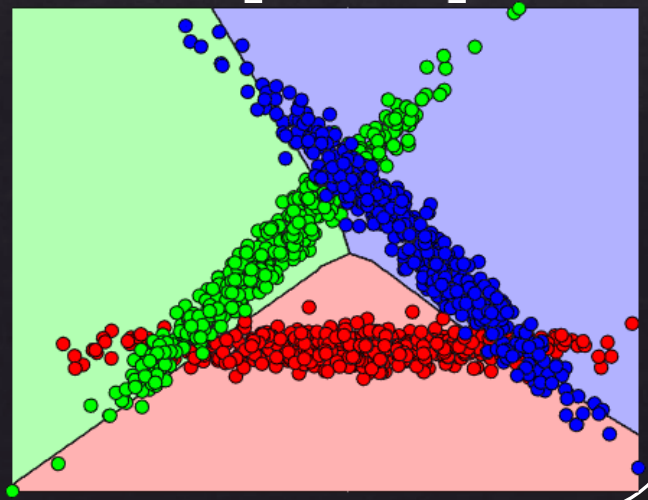
- (Reminder) Loss function: $\ell\left(\hat{y}_i, y_i\right) \geq 0$
- Matrix form: $\boldsymbol{L} = \begin{bmatrix} 0 & \ell\left(\hat{C}_1, C_2\right) & \ell\left(\hat{C}_1, C_3\right) \\ \ell\left(\hat{C}_2, C_1\right) & 0 & \ell\left(\hat{C}_2, C_3\right) \\ \ell\left(\hat{C}_3, C_1\right) & \ell\left(\hat{C}_3, C_2\right) & 0 \end{bmatrix}$

$$\boldsymbol{L}[i,j] = \ell\left(\hat{C}_i, C_j\right)$$



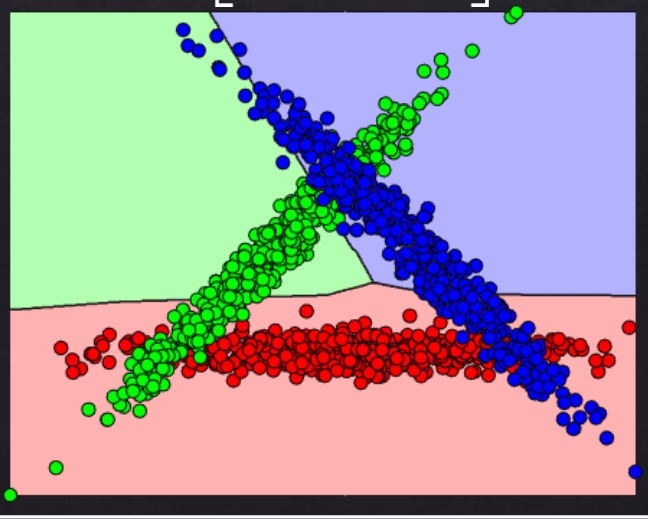
Hamming $0-1$ loss:

$$\boldsymbol{L} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$



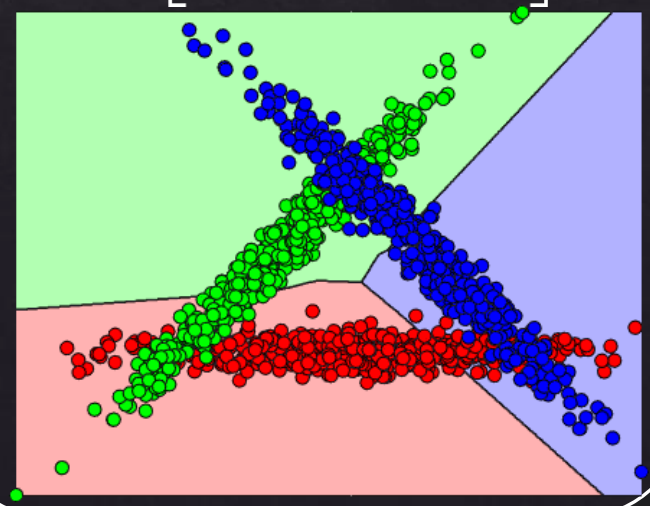
high cost for mislabeling C_1:

$$\boldsymbol{L} = \begin{bmatrix} 0 & 1 & 1 \\ 100 & 0 & 1 \\ 100 & 1 & 0 \end{bmatrix}$$

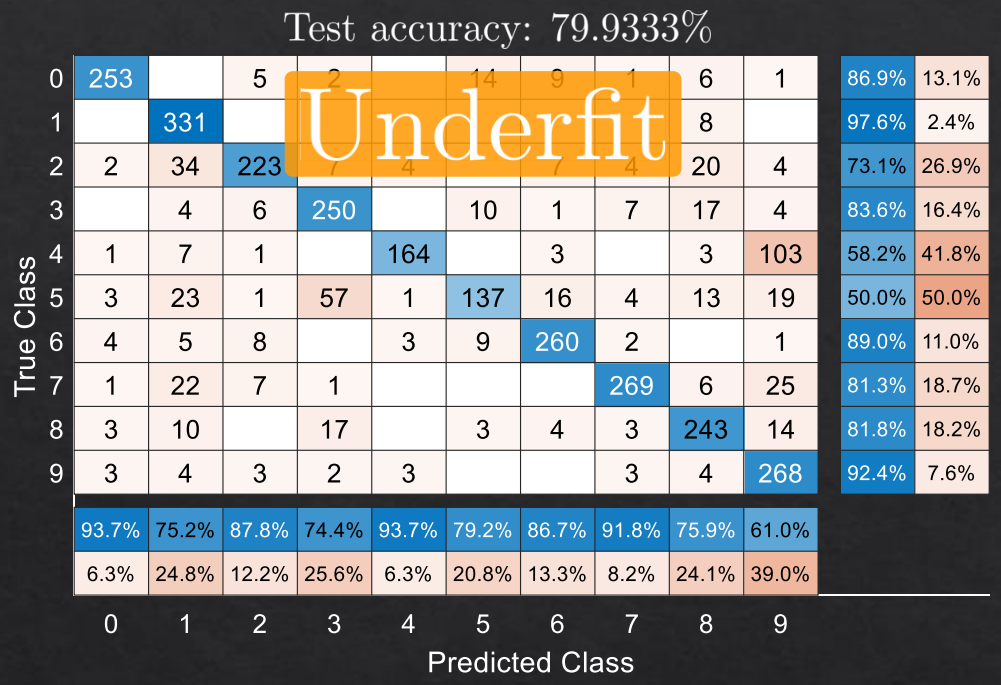


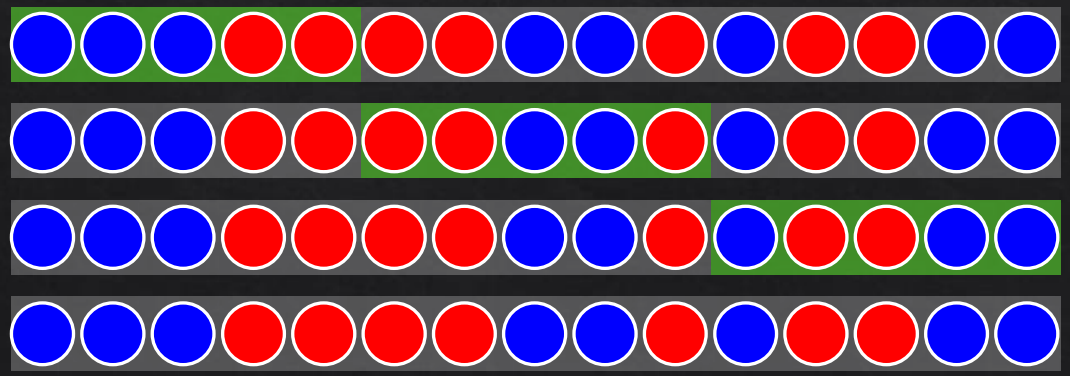
cyclic cost

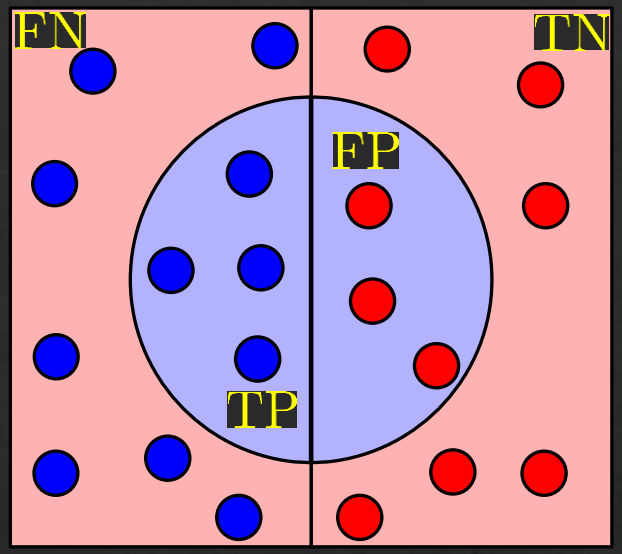
$$\boldsymbol{L} = \begin{bmatrix} 0 & 1 & 100 \\ 100 & 0 & 1 \\ 1 & 100 & 0 \end{bmatrix}$$



Questions

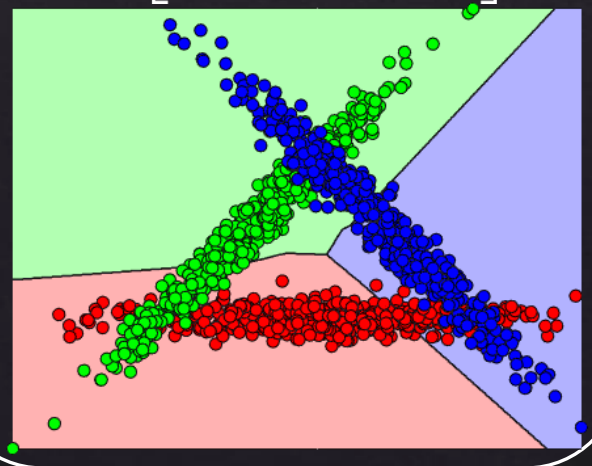






cyclic cost

$$\boldsymbol{L} = \begin{bmatrix} 0 & 1 & 100 \\ 100 & 0 & 1 \\ 1 & 100 & 0 \end{bmatrix}$$



fixelalgorithms.gitlab.io