Introduction to Machine Learning
Lecture 3: Linear Classification

Harbour.Space University
February 2021

Nikolay Karpachev

Outline

1. Recap: Linear regression
2. Linear classification
   - Problem statement
   - Linear classification model
   - Loss functions and training procedure
3. Logistic regression
4. Measuring the quality in classification: metrics

Linear regression problem statement:

- Dataset $\mathcal{L} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathbb{R}^n$, $y_i \in \mathbb{R}$.

Linear regression problem statement:

- Dataset $\mathcal{L} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathbb{R}^n, \quad y_i \in \mathbb{R}$.

Examples:

- Housing price prediction
- Person age
- Photo background depth estimation

Linear regression problem statement:

- Dataset $\mathcal{L} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathbb{R}^n, \quad y_i \in \mathbb{R}$.
- The model is linear:

$$\hat{y} = w_0 + \sum_{k=1}^{p} x_k \cdot w_k = //\mathbf{x} = [1, x_1, x_2, \ldots, x_p]// = \mathbf{x}^T\mathbf{w}$$

Linear regression problem statement:

- Dataset $\mathcal{L} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathbb{R}^n, \quad y_i \in \mathbb{R}$.
- The model is linear:

$$\hat{y} = w_0 + \sum_{k=1}^{p} x_k \cdot w_k = //\mathbf{x} = [1, x_1, x_2, \dots, x_p]// = \mathbf{x}^T \mathbf{w}$$

where $\mathbf{w} = (w_0, w_1, \dots, w_n)$, w_0 is bias term

Linear regression problem statement:

- Dataset $\mathcal{L} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathbb{R}^n, \quad y_i \in \mathbb{R}$.
- The model is linear:

$$\hat{y} = w_0 + \sum_{k=1}^{p} x_k \cdot w_k = //\mathbf{x} = [1, x_1, x_2, \ldots, x_p]// = \mathbf{x}^T \mathbf{w}$$

where $\mathbf{w} = (w_0, w_1, \ldots, w_n)$, w_0 is bias term

we added an additional column of 1’s to the design matrix to simplify the formulas

Linear regression problem statement:

- Dataset $\mathcal{L} = \{\mathbf{x}_i, y_i\}_{i=1}^{N}$, where $\mathbf{x}_i \in \mathbb{R}^n, \quad y_i \in \mathbb{R}$.
- The model is linear:

$$\hat{y} = w_0 + \sum_{k=1}^{p} x_k \cdot w_k = //\mathbf{x} = [1, x_1, x_2, \ldots, x_p]// = \mathbf{x}^T \mathbf{w}$$

where $\mathbf{w} = (w_0, w_1, \ldots, w_n)$, w_0 is bias term.

- Least squares method (MSE minimization) provides a solution:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \|Y - \hat{Y}\|_2^2 = \arg\min_{\mathbf{w}} \|Y - X\mathbf{w}\|_2^2$$

Analytical solution:

$$\hat{\mathbf{w}} = (X^T X)^{-1} X^T Y$$

In practice:

Gradient descent optimization

$$\mathbf{w}_{n+1} = \mathbf{w}_n - \gamma \nabla ||\mathbf{Y} - X\mathbf{w}_n||^2$$

From regression to classification

From regression to classification

Task: predict a label from a fixed predefined set

- **Binary classification (labels -1, +1)**
  - Sentiment analysis
  - Credit scoring
  - etc.
- **Multiclass classification (labels 0, 1, …, n)**
  - Image classification
  - Robotics
  - Text generation, reinforcement learning, etc.

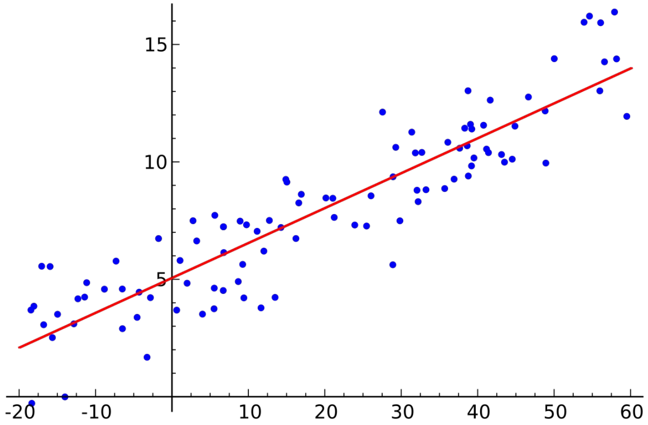
From regression to classification

Regression:

$$\hat{y} = f(\mathbf{x}) = \mathbf{x}^T \mathbf{w}$$

$$Q = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

How can we use the same technique to solve the *classification* problem?



From regression to classification

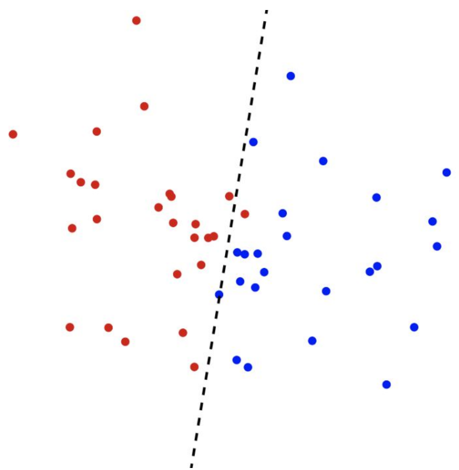
Classification:

$$\hat{y} = f(\mathbf{x}) = \mathbf{x}^T \mathbf{w}$$

$$a(\mathbf{x}) = +1 \text{ if } f(\mathbf{x}) > 0$$
$$a(\mathbf{x}) = -1 \text{ if } f(\mathbf{x}) < 0$$

$$Q = \frac{1}{N} \sum_{i=1}^{N} L(y_i, \hat{y}_i)$$

Let's say we predict the class label now

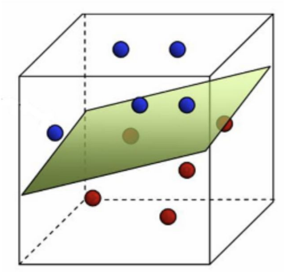


Linear classification

$$a(\mathbf{x}) = +1 \ \text{ if } f(\mathbf{x}) > 0$$
$$a(\mathbf{x}) = -1 \ \text{ if } f(\mathbf{x}) < 0$$

$$f(\mathbf{x}) = \mathbf{x}^T \mathbf{w}$$

Geometrical interpretation:
Linearly separable case



Linear classification

Classification:

$$\hat{y} = f(\mathbf{x}) = \mathbf{x}^T \mathbf{w}$$

$$a(\mathbf{x}) = +1 \ \text{ if } f(\mathbf{x}) > 0$$
$$a(\mathbf{x}) = -1 \ \text{ if } f(\mathbf{x}) < 0$$

$$Q = \frac{1}{N} \sum_{i=1}^{N} \boxed{L(y_i, \hat{y}_i)}$$

What about loss function?
To train our model, we actually need some target

Linear classification

Classification:

$$\hat{y} = f(\mathbf{x}) = \mathbf{x}^T \mathbf{w}$$

$$a(\mathbf{x}) = +1 \text{ if } f(\mathbf{x}) > 0$$
$$a(\mathbf{x}) = -1 \text{ if } f(\mathbf{x}) < 0$$

$$Q = \frac{1}{N} \sum_{i=1}^{N} \boxed{L(y_i, \hat{y}_i)}$$

What about loss function? To train our model, we actually need some target

Idea #0: Loss function could be just number of misclassifications

Margin

Our linear classification algorithm: $a(\mathbf{x})$

Let’s call $M_i = y_i a(\mathbf{x}_i)$ algorithm ***margin*** on object $\mathbf{x}_i$.

$$M_i \leq 0 \iff y_i \neq a(\mathbf{x}_i)$$
$$M_i > 0 \iff y_i = a(\mathbf{x}_i)$$

Margin Loss

- Margin loss (number of classification errors)

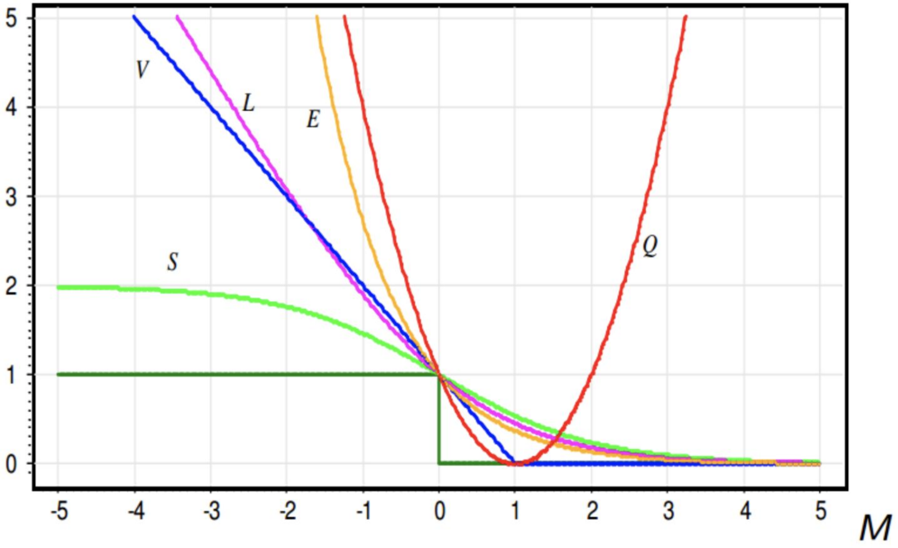
$$Q = \frac{1}{N}\sum_{i=1}^{N}[M_i \leq 0]$$

- Problem: this function is not differentiable
- And we can not find optimal solution with gradient descent

Loss functions in classification

$$Q = \frac{1}{N}\sum_{i=1}^{N}[M_i \le 0] \le \widetilde{Q} = \frac{1}{N}\sum_{i=1}^{N} L(M_i)$$

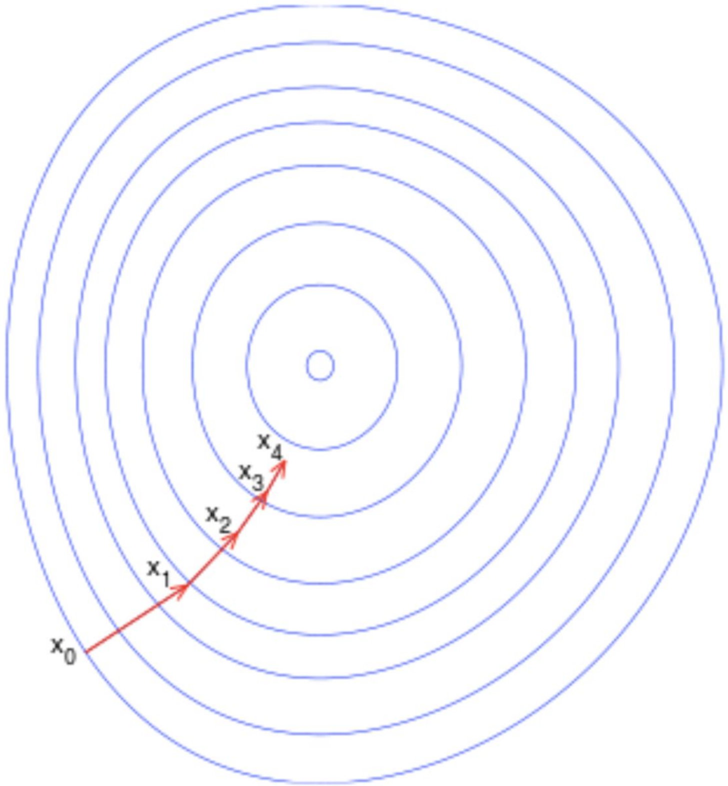
$$\widetilde{Q} \longrightarrow \min \implies Q \longrightarrow \min$$



$Q(M) = (1 - M)^2$
$V(M) = (1 - M)_+$
$S(M) = 2(1 + e^M)^{-1}$
$L(M) = \log_2(1 + e^{-M})$
$E(M) = e^{-M}$

Loss functions

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \gamma_n \nabla F(\mathbf{x}_n),\ n \geq 0.$$



$$\nabla_w \tilde{Q} = \sum_{i=1}^{l} \nabla L(M_i)$$

$$\nabla \tilde{Q} = \sum_{i=1}^{l} L'(M_i) \frac{\partial M_i}{\partial w}$$

$$\frac{\partial M_i}{\partial w} = y_i x_i$$

$$\nabla \tilde{Q} = \sum_{i=1}^{l} y_i x_i L'(M_i)$$

$$\boxed{w_{n+1} = w_n - \gamma_n \sum_{i=1}^{l} y_i x_i L'(M_i)}$$

Time for your questions and a coffee break.

Linear classification from the other side

So far: optimization point of view:

- Define a loss function
- Minimize it with numerical methods

Linear classification from the other side

Probabilistic point of view:

- Suppose our data comes from a certain distribution: $X \times Y$

$$p(x, y|w) = P(y|x, w)p(x).$$

- Maximum likelihood estimation principle
- We want to find the most likely parameters ***w*** that could generate the data that we have

Linear classification from the other side

Maximum likelihood estimation

Log-likelihood maximization method:

$$L(w) = \sum_{i=1}^{\ell} \log P(y_i | x_i, w) \to \max_w .$$

Log-Loss

Or (equivalently), log-loss minimization method:

$$-\log P(y_i | x_i, w) = \boxed{\mathscr{L}(y_i g(x_i, w))}.$$

Logistic regression

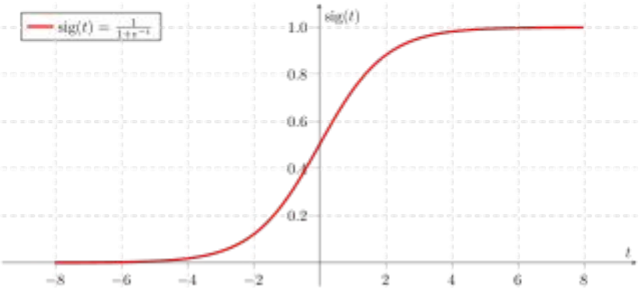
Linear classification model:

$$a(x) = \mathrm{sign}\langle w, x\rangle, \quad x, w \in \mathbb{R}^n.$$

Logistic regression:

$$p_i = \underline{\sigma(\langle w, x_i\rangle)} = \frac{1}{1 + e^{-\langle w, x_i\rangle}} = P(y = 1|x)$$

Sigmoid function



Logistic regression

$$y_i \in \{0, 1\} \qquad Q = -\sum_{i=1}^{\ell} y_i \ln p_i + (1 - y_i) \ln(1 - p_i) \to \min_w$$

$$p_i = \sigma(\langle w, x_i \rangle) = \frac{1}{1 + e^{-\langle w, x_i \rangle}} = P(y = 1 | x)$$

logistic loss

The class labels are described as Bernoulli-distributed data with parameters *p* inferred from the model

Logistic regression optimization problem

$$Q = -\sum_{i=1}^{\ell} y_i \ln \frac{1}{1 + e^{-\langle w, x_i \rangle}} + (1 - y_i) \ln \frac{1}{1 + e^{\langle w, x_i \rangle}} \to \min_w$$

$$-y_i \ln \frac{1}{1 + e^{-\langle w, x_i \rangle}} - (1 - y_i) \ln \frac{1}{1 + e^{\langle w, x_i \rangle}} = \begin{cases} \ln\left(1 + e^{-\langle w, x_i \rangle}\right), y_i = 1 \\ \ln\left(1 + e^{\langle w, x_i \rangle}\right), y_i = 0 \end{cases}$$

$$Q = \sum_{i=1}^{\ell} \underbrace{\ln\left(1 + e^{-y_i \langle w, x_i \rangle}\right)}_{L(M) = \ln(1 + e^{-M_i})} \to \min_w \qquad y_i \in \{-1, 1\}$$

Measuring the quality in classification

Quality functions in classification

- Accuracy
- Precision
- Recall
- F-score
- ROC-curve, ROC-AUC
- PR-curve

Accuracy

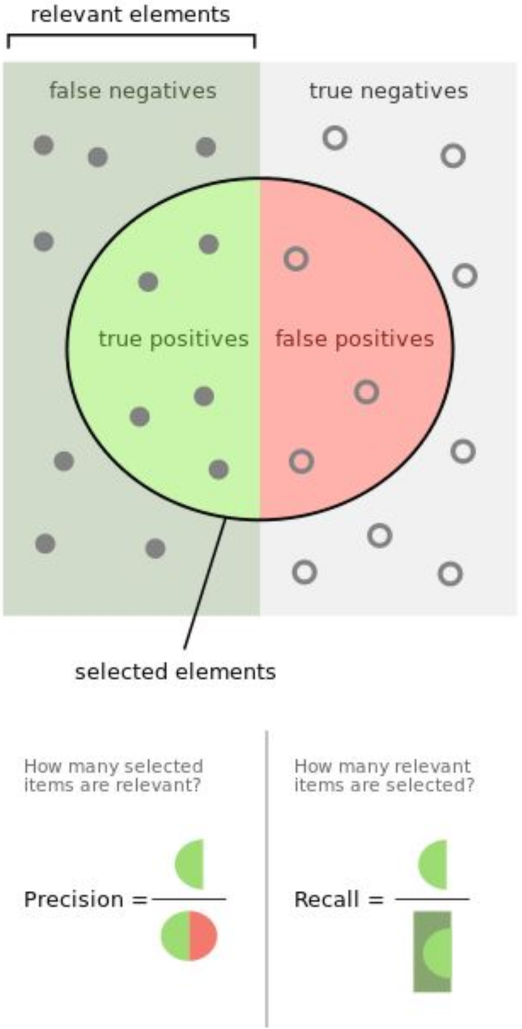
Number of right classifications

target: 1 0 1 0 0 0 0 1 0 0

predicted: 0 0 1 0 0 0 0 1 1 0

accuracy = 8/10 = 0.8

Precision and recall



		Actual Class	
		Yes	No
Predicted Class	Yes	**T**rue **P**ositive	**F**alse **P**ositive
	No	**F**alse **N**egative	**T**rue **N**egative

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

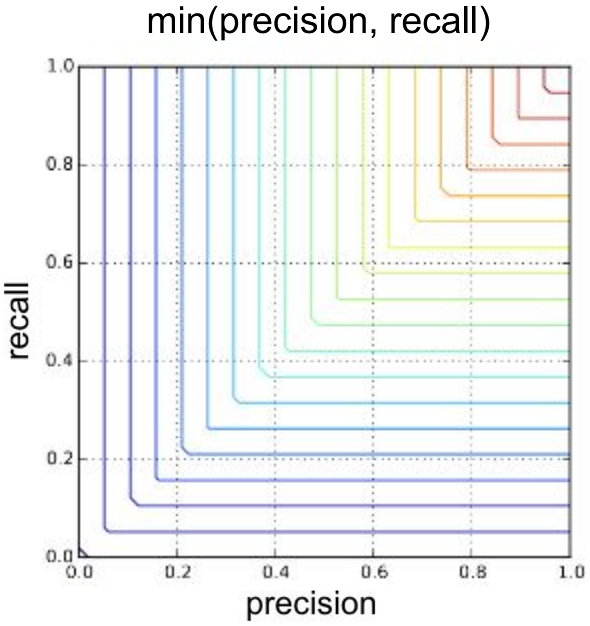
F-score

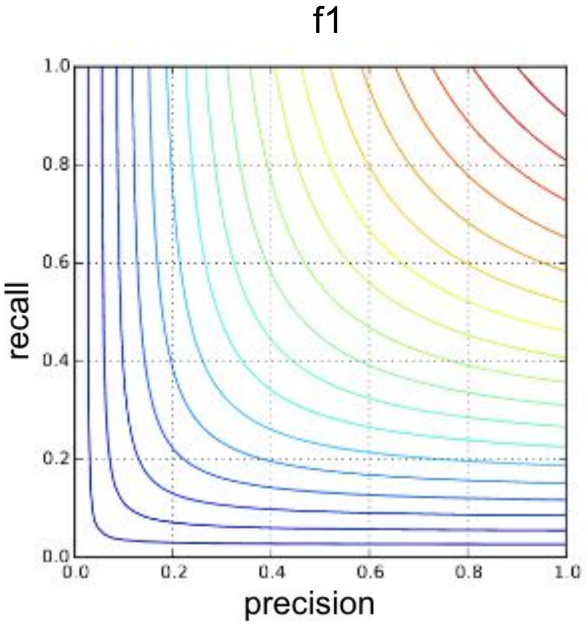
Harmonic mean of precision and recall.
Closer to the smallest one.

$$F_1 = \left(\frac{\mathrm{recall}^{-1} + \mathrm{precision}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

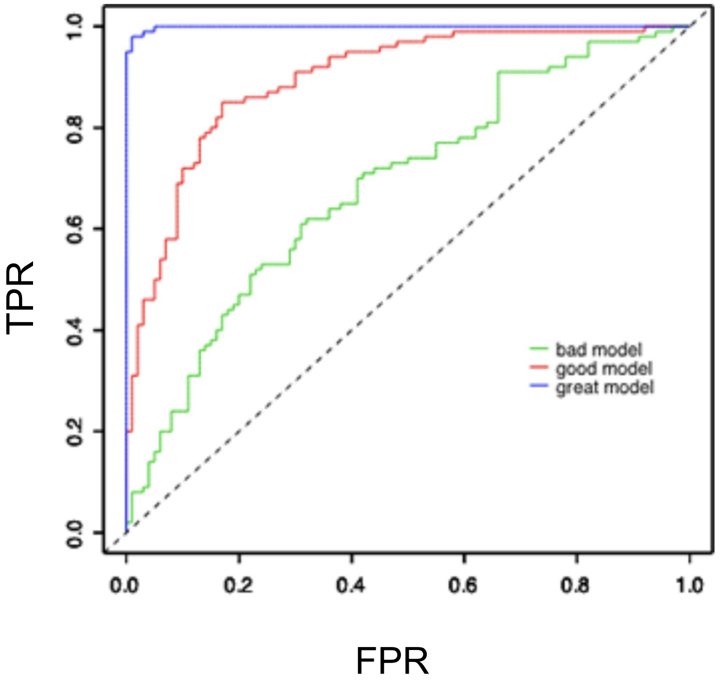
$$F_\beta = (1 + \beta^2) \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{(\beta^2 \cdot \mathrm{precision}) + \mathrm{recall}}$$

F-score


min(precision, recall)
1.0
0.8
0.6
0.4
0.2
0.0
recall
0.0
0.2
0.4
0.6
0.8
1.0
precision


f1
1.0
0.8
0.6
0.4
0.2
0.0
recall
0.0
0.2
0.4
0.6
0.8
1.0
precision

ROC - receiver operating characteristic

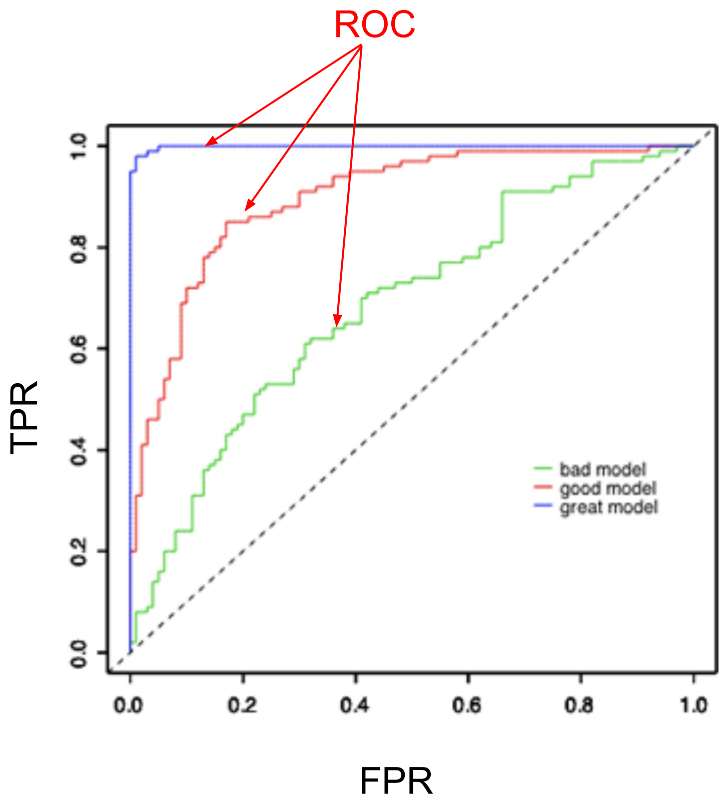


		Actual Class	
		Yes	No
Predicted Class	Yes	**T**rue **P**ositive	**F**alse **P**ositive
	No	**F**alse **N**egative	**T**rue **N**egative

$$TPR = \frac{True\ positives}{True\ positives + False\ negatives}$$

$$FPR = \frac{False\ positives}{False\ positives + True\ negatives}.$$

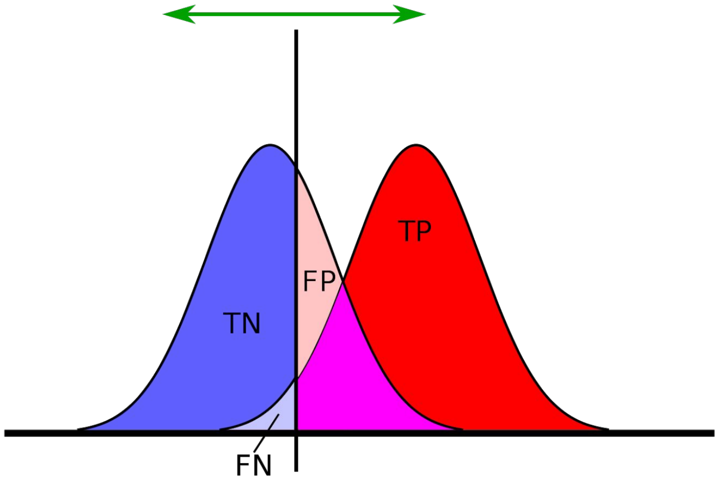
ROC

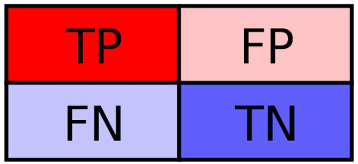


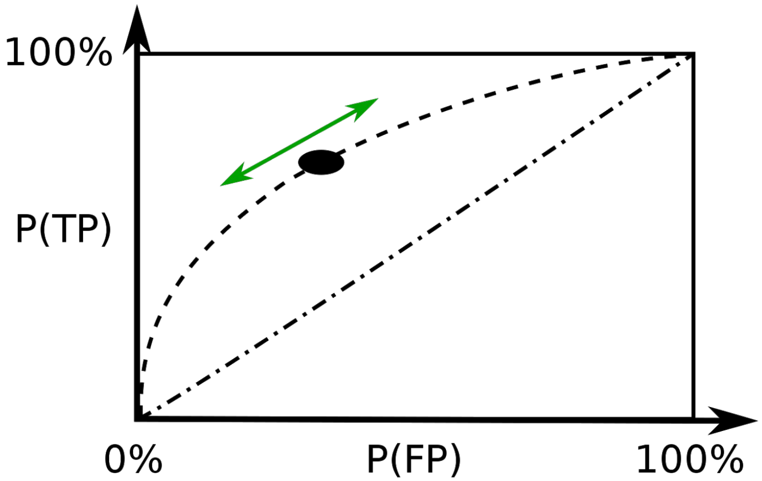
		Actual Class	
		Yes	No
Predicted Class	Yes	**T**rue **P**ositive	**F**alse **P**ositive
	No	**F**alse **N**egative	**T**rue **N**egative

$$TPR = \frac{True\ positives}{True\ positives + False\ negatives}$$

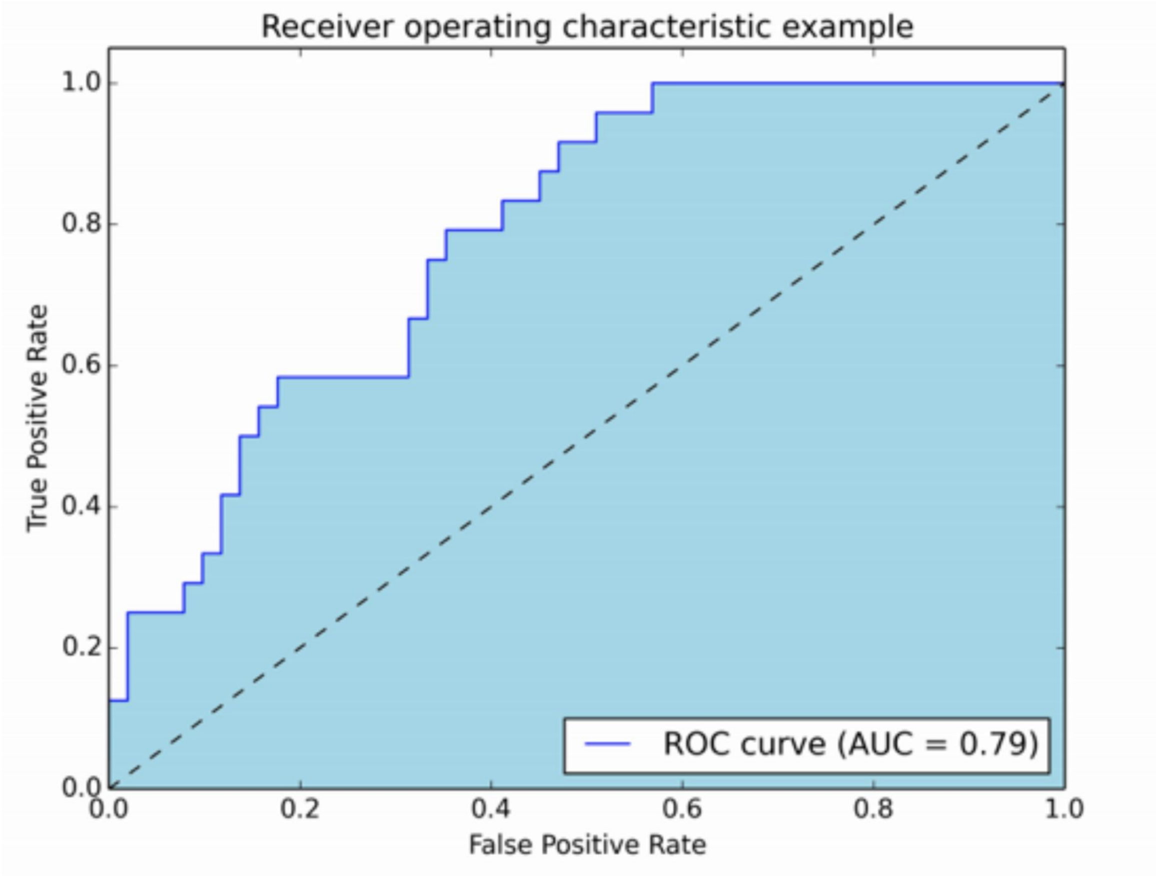
$$FPR = \frac{False\ positives}{False\ positives + True\ negatives}.$$


TP
FP
TN
FN

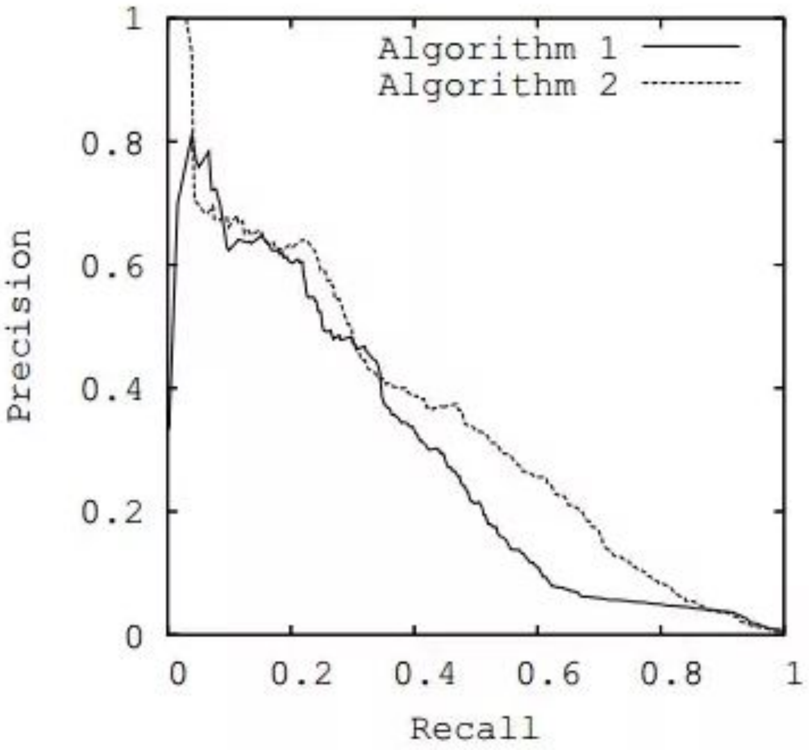

TP
FP
FN
TN


100%
P(TP)
0%
P(FP)
100%

ROC-AUC - area under curve



PR-curve



$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

Thanks for your attention!