

Classification metrics

Plan for the video

- Accuracy
- Logarithmic loss
- Area under ROC curve
- (Quadratic weighted) Kappa

Notation

- N – is number of objects
- L – is number of classes
- y – ground truth
- \hat{y} – predictions
- $[a = b]$ – indicator function
- Soft labels (soft predictions) are classifier's scores
- Hard labels (hard predictions):
 - $\arg \max_i f_i(x)$
 - $[f(x) > b]$, b – threshold

Accuracy score

$$\text{Accuracy} = \frac{1}{N} \sum_{i=1}^N [\hat{y}_i = y_i]$$

- How frequently our class prediction is correct.

Accuracy score

$$\text{Accuracy} = \frac{1}{N} \sum_{i=1}^N [\alpha = y_i]$$

- How frequently our class prediction is correct.
- Best constant:
 - **predict the most frequent class.**

- Dataset:
 - 10 cats
 - 90 dogs

Predict always dog:
Accuracy = **0.9!**

Logarithmic loss (logloss)

- Binary:

$$\text{LogLoss} = -\frac{1}{N} \sum_{i=1}^N y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)$$
$$y_i \in \mathbb{R}, \quad \hat{y}_i \in \mathbb{R}$$

Accuracy also doesn't care how confident the classifier is in the predictions, and what soft predictions are.

It cares only about arg max of soft predictions.

And thus, people sometimes prefer to use different metrics that are first, easier to optimize. And second, these metrics work with soft predictions, not hard ones.

One of such metrics is logarithmic loss.

Logarithmic loss (logloss)

- Binary:

$$\text{LogLoss} = -\frac{1}{N} \sum_{i=1}^N y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)$$
$$y_i \in \mathbb{R}, \quad \hat{y}_i \in \mathbb{R}$$

- Multiclass:

$$\text{LogLoss} = -\frac{1}{N} \sum_{i=1}^N \sum_{l=1}^L y_{il} \log(\hat{y}_{il})$$
$$y_i \in \mathbb{R}^L, \quad \hat{y}_i \in \mathbb{R}^L$$

Logarithmic loss (logloss)

- Binary:

$$\text{LogLoss} = -\frac{1}{N} \sum_{i=1}^N y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)$$
$$y_i \in \mathbb{R}, \quad \hat{y}_i \in \mathbb{R}$$

- Multiclass:

$$\text{LogLoss} = -\frac{1}{N} \sum_{i=1}^N \sum_{l=1}^L y_{il} \log(\hat{y}_{il})$$
$$y_i \in \mathbb{R}^L, \quad \hat{y}_i \in \mathbb{R}^L$$

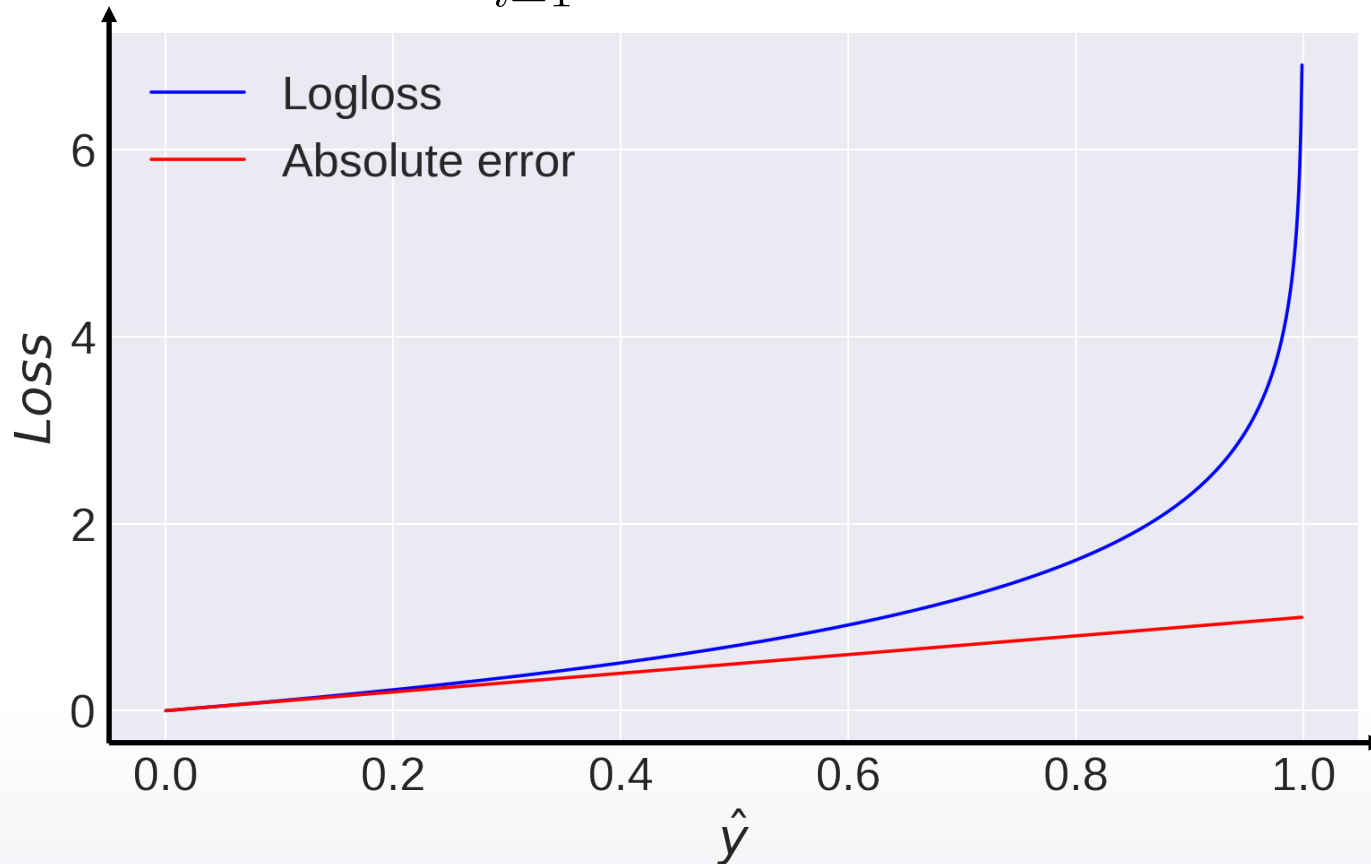
- In practice:

$$\text{LogLoss} = -\frac{1}{N} \sum_{i=1}^N \sum_{l=1}^L y_{il} \log(\min(\max(\hat{y}_{il}, 10^{-15}), 1 - 10^{-15}))$$

in practice, predictions are clipped to be not from 0 to 1, but from some small positive number to 1 minus some small positive number.

Logarithmic loss (logloss)

$$\text{LogLoss} = -\frac{1}{N} \sum_{i=1}^N y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)$$



- Logloss strongly penalizes completely wrong answers

Logarithmic loss (logloss)

$$\text{LogLoss} = -\frac{1}{N} \sum_{i=1}^N y_i \log(\alpha) + (1 - y_i) \log(1 - \alpha)$$

- Best constant:
 - **set α_i to frequency of i -th class.**
-

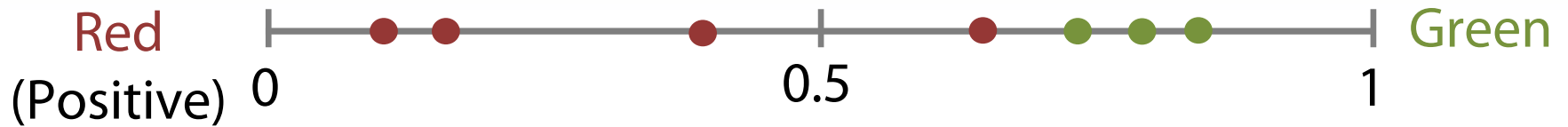
- Dataset:
 - 10 cats
 - 90 dogs

$$\alpha = [0.1, 0.9]$$

Now, what is the best constant for logarithmic loss? It turns out that you need to set predictions to the frequencies of each class in the data set. In our case, the frequencies for the cat class is 0.1, and it is 0.9 for class dog. Then the best constant is vector of those two values.

To prove it we should take a derivative with the respect to constant alpha, set it to 0, and find alpha from this equation.

Area Under Curve (AUC ROC)



$$\text{Accuracy}([\hat{y} > 0.5]) = \frac{6}{7}$$

Area Under Curve (AUC ROC)



$$\text{Accuracy}([\hat{y} > 0.7]) = 1$$

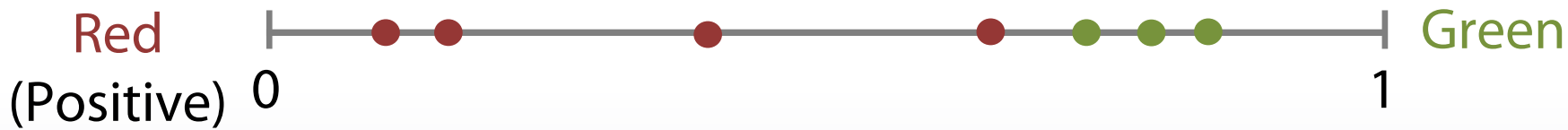
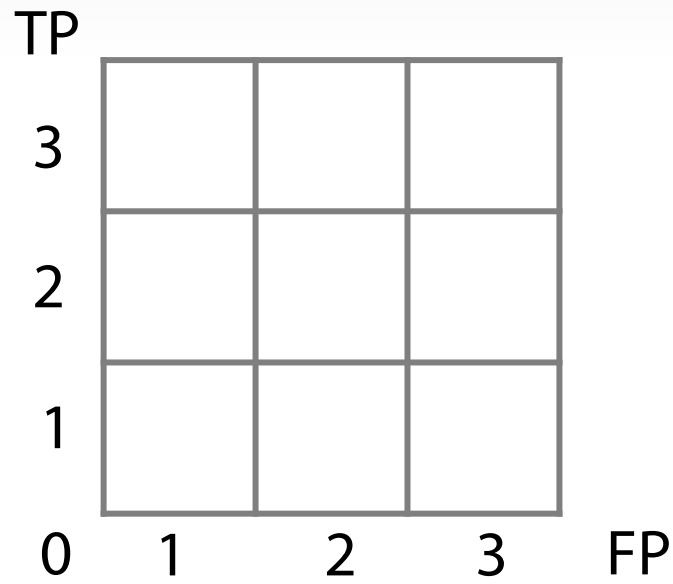
Area Under Curve (AUC ROC)



$$\text{Accuracy}([\hat{y} > 0.7]) = 1$$

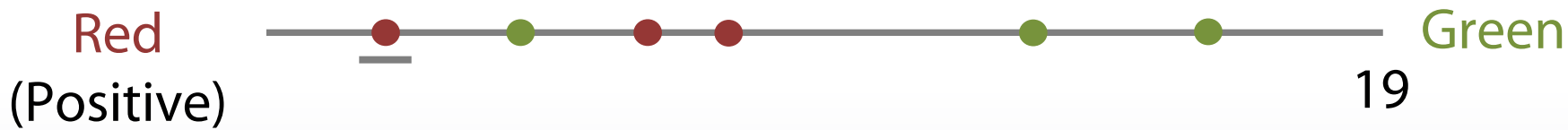
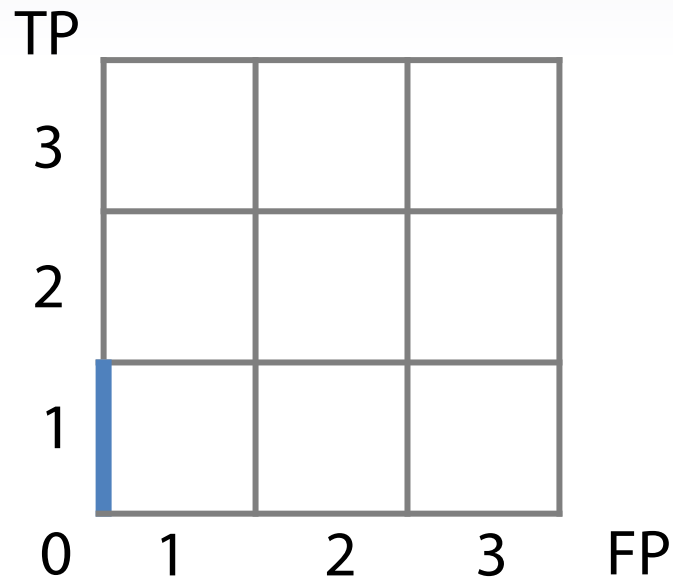
- Only for binary tasks
- Depends only on ordering of the predictions, not on absolute values
- **Several explanations**
 - 1) Area under curve
 - 2) Pairs ordering

Area Under Curve (AUC ROC)



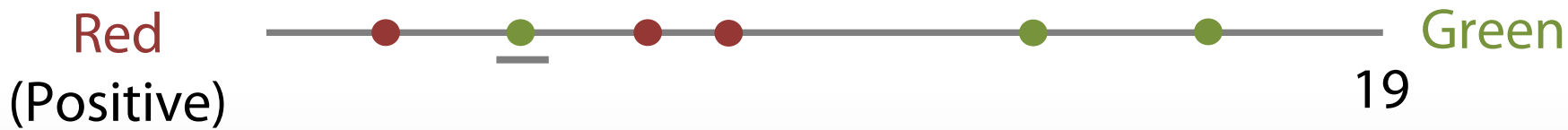
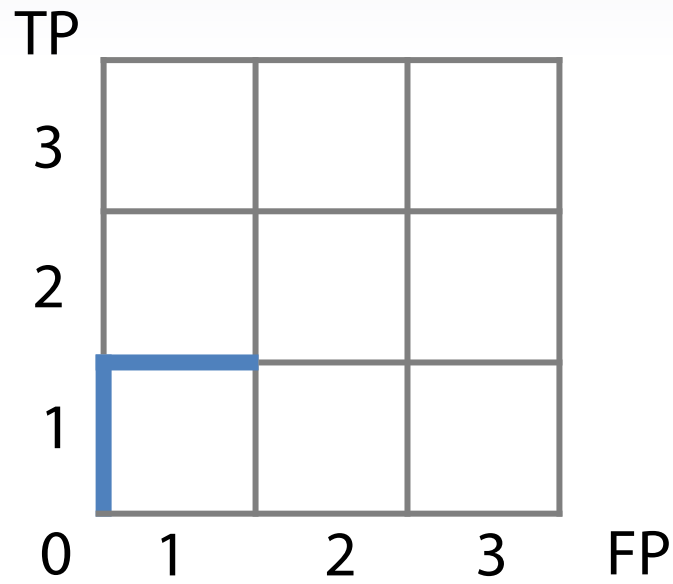
TP – true positives, **FP** – false positives

Area Under Curve (AUC ROC)



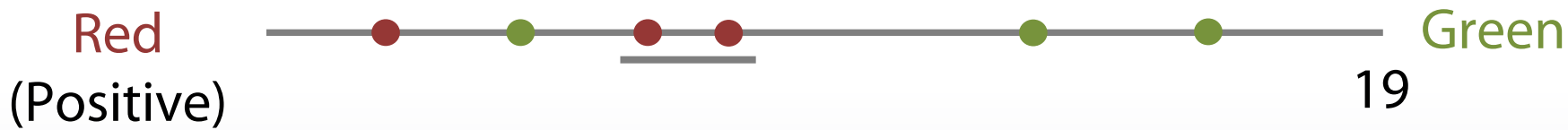
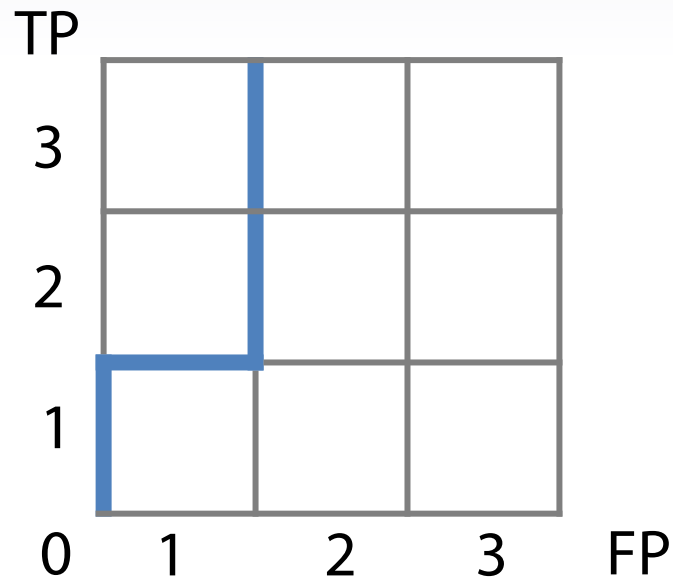
TP – true positives, **FP** – false positives

Area Under Curve (AUC ROC)



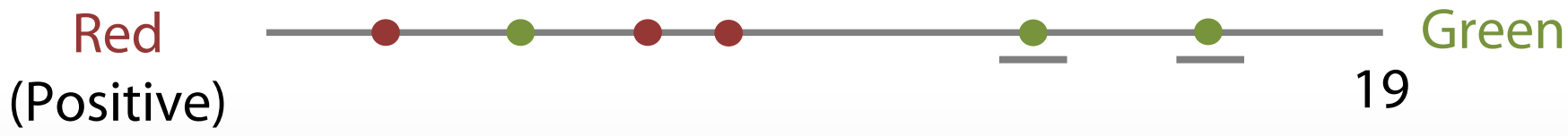
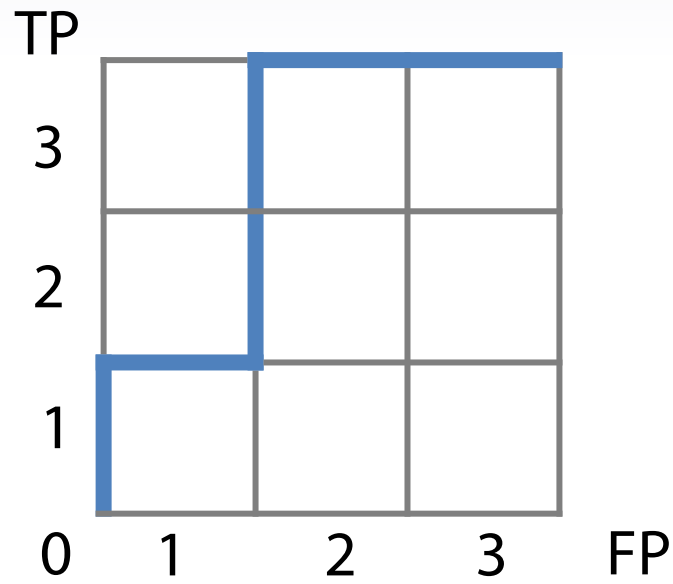
TP – true positives, **FP** – false positives

Area Under Curve (AUC ROC)



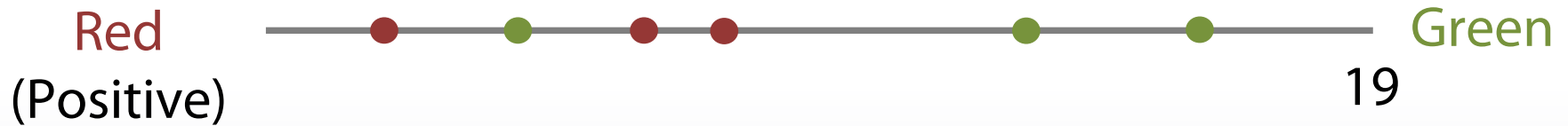
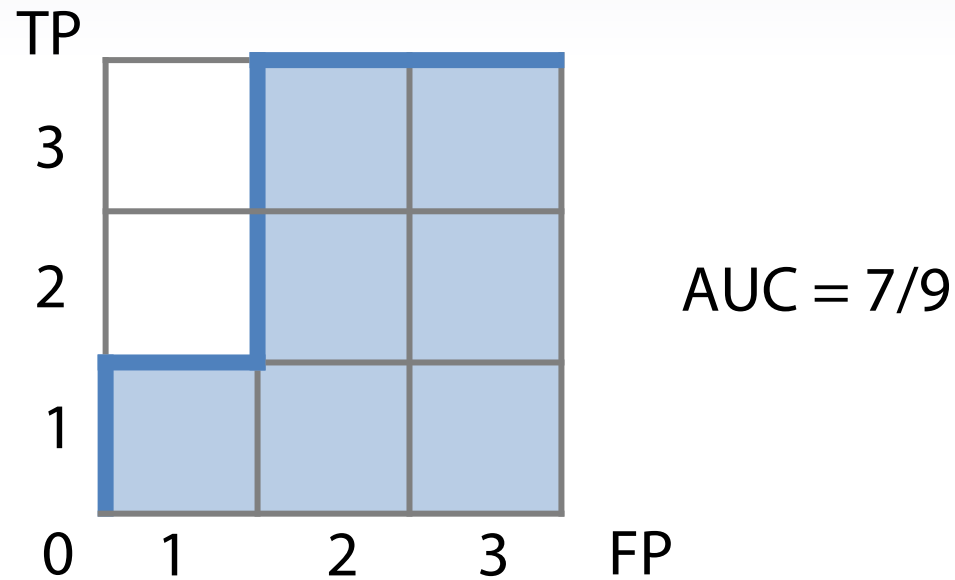
TP – true positives, **FP** – false positives

Area Under Curve (AUC ROC)



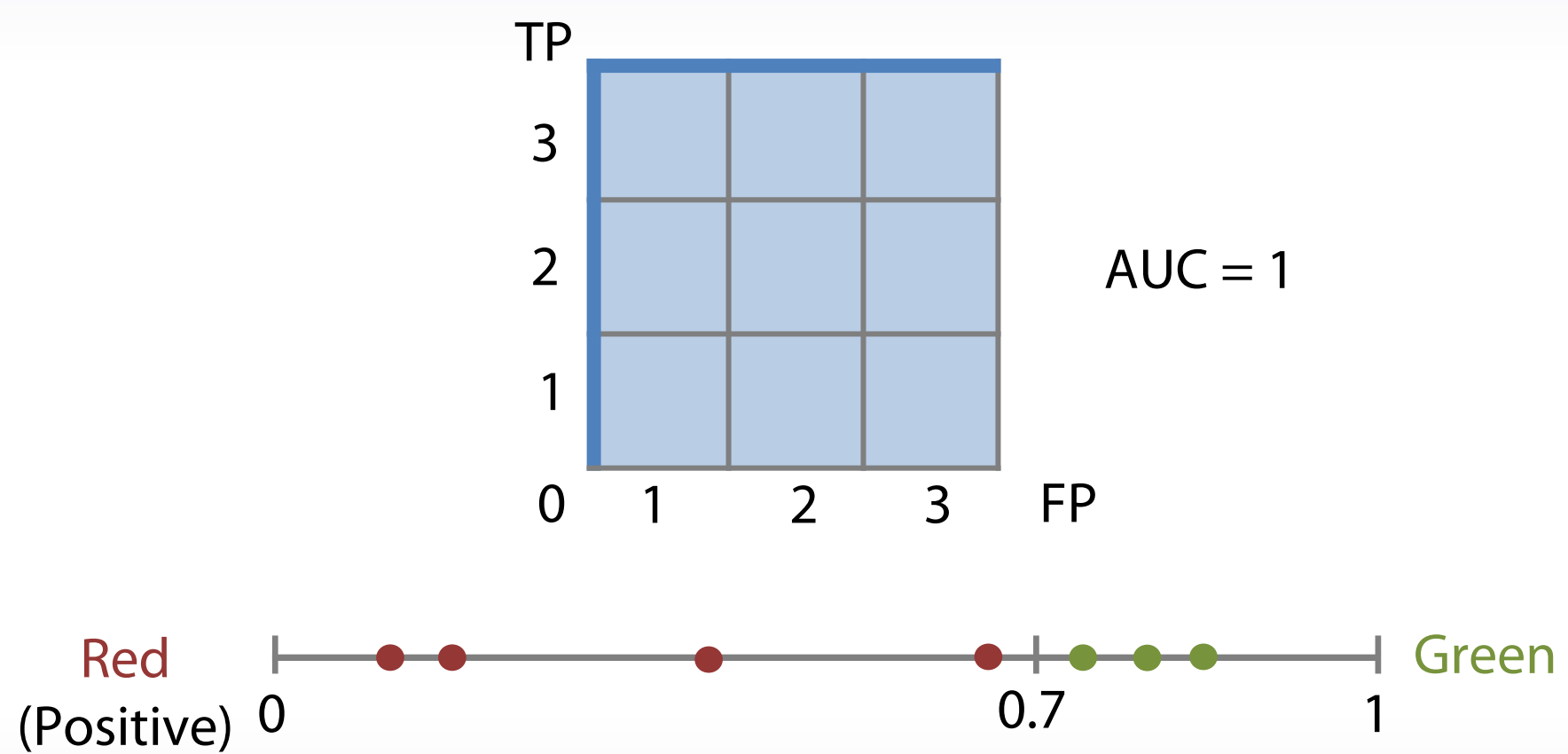
TP – true positives, **FP** – false positives

Area Under Curve (AUC ROC)



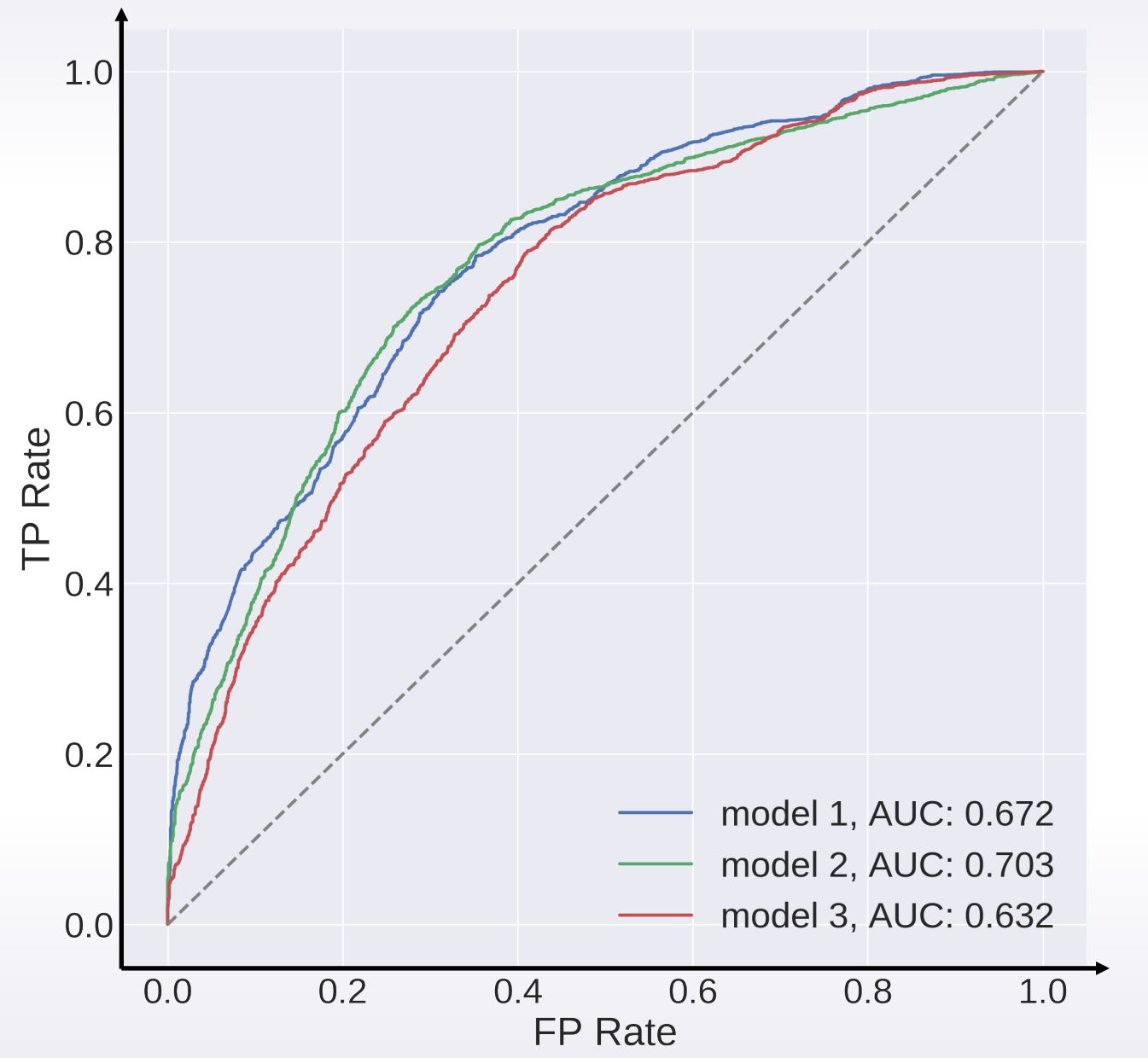
TP – true positives, **FP** – false positives

Area Under Curve (AUC ROC)



TP – true positives, **FP** – false positives

Area Under Curve (AUC ROC)



Area Under Curve (AUC ROC)

$$\begin{aligned} \text{AUC} &= \frac{\# \text{ correctly ordered pairs}}{\text{total number of pairs}} = \\ &= 1 - \frac{\# \text{ incorrectly ordered pairs}}{\text{total number of pairs}} \end{aligned}$$



pair = (red object, green object)

Area Under Curve (AUC ROC)

- **Best constant:**
 - All constants give same score
- **Random predictions lead to $AUC = 0.5$**

This is actually something that people love about AUC. It is clear what the baseline is. Of course there are flaws in AUC, every metric has some. But still AUC is metric I usually use when no one sets up another one for me.

Cohen's Kappa motivation

Dataset:

- 10 cats
- 90 dogs

Baseline accuracy = 0.9

in this example the baseline is 0.9

$$\text{my_score} = 1 - \frac{1 - \text{accuracy}}{1 - \text{baseline}}$$

- | | | |
|------------------|--|--------------|
| • accuracy = 1 |  | my_score = 1 |
| • accuracy = 0.9 |  | my_score = 0 |

Cohen's Kappa motivation

Dataset:

- 10 cats
- 90 dogs

Predict 20 *cats* and 80 *dogs* at
random: *accuracy* ~ 0.74

$$0.2 * 0.1 + 0.8 * 0.9 = 0.74$$

$$\text{Cohen's Kappa} = 1 - \frac{1 - \text{accuracy}}{1 - p_e}$$

p_e – what accuracy would be on average, if we randomly permute our predictions

$$p_e = \frac{1}{N^2} \sum_k n_{k1} n_{k2}$$

Cohen's Kappa motivation

Dataset:

- 10 cats
- 90 dogs

Predict 20 *cats* and 80 *dogs* at
random: *accuracy* ~ 0.74
error ~ 0.26

$$\text{Cohen's Kappa} = 1 - \frac{\text{error}}{\text{baseline error}}$$

Weighted error

Dataset:

- 10 cats
- 90 dogs
- 20 tigers


Error weight matrix

pred\ true	cat	dog	tiger
cat	0	1	10
dog	1	0	10
tiger	1	1	0

In our case, we set error weight to be ten times larger if we predict cat or dog, but the ground truth label is tiger.

Weighted error and weighted Kappa

Confusion matrix C



pred\ true	cat	dog	tiger
cat	4	2	3
dog	2	88	5
tiger	4	10	12

Weight matrix W

pred\ true	cat	dog	tiger
cat	0	1	10
dog	1	0	10
tiger	1	1	0

$$\text{weighted error} = \frac{1}{const} \sum_{i,j} C_{ij} W_{ij}$$

to calculate weight and error we need another matrix, confusion matrix, for the classifier's prediction.

This matrix shows how our classifier distributes the predictions over the objects.

Weighted error and weighted Kappa

Confusion matrix C

pred\ true	cat	dog	tiger
cat	4	2	3
dog	2	88	4
tiger	4	10	12

Weight matrix W

pred\ true	cat	dog	tiger
cat	0	1	10
dog	1	0	10
tiger	1	1	0

$$\text{weighted error} = \frac{1}{const} \sum_{i,j} C_{ij} W_{ij}$$

$$\text{weighted kappa} = 1 - \frac{\text{weighted error}}{\text{weighted baseline error}}$$

Quadratic and Linear Weighted Kappa

Linear weights

pred\ true	1	2	3
1	0	1	2
2	1	0	1
3	2	1	0

$$w_{ij} = |i - j|$$

Quadratic weights

pred\ true	1	2	3
1	0	1	4
2	1	0	1
3	4	1	0

$$w_{ij} = (i - j)^2$$

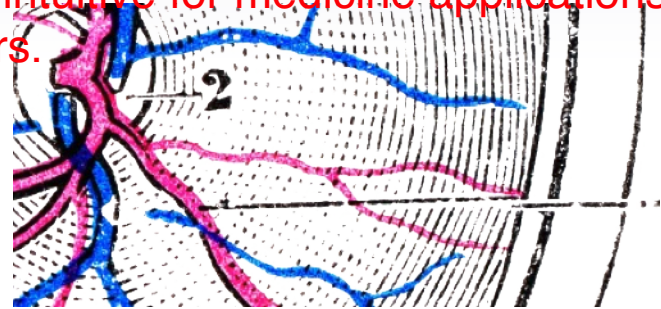
$$\text{weighted kappa} = 1 - \frac{\text{weighted error}}{\text{weighted baseline error}}$$

Quadratic Weighted Kappa

The quadratic weighted kappa has been used in several competitions on Kaggle. It is usually explained as inter-rater agreement coefficient, how much the predictions of the model agree with ground-truth raters. Which is quite intuitive for medicine applications, how much the model agrees with professional doctors.



CrowdFlower Search
Results Relevance



Diabetic Retinopathy
Detection



Prudential Life
Insurance Assessment



The Hewlett Foundation:
Automated Essay Scoring

Conclusion

- Accuracy
- Logloss
- AUC (ROC)
- (Quadratic weighted) Kappa