Evaluation Metrics
CS229

Yining Chen
(Adapted from slides by Anand Avati)
May 1, 2020

Topics

- Why are metrics important?
- Binary classifiers
  - Rank view, Thresholding
- Metrics
  - Confusion Matrix
  - Point metrics: Accuracy, Precision, Recall / Sensitivity, Specificity, F-score
  - Summary metrics: AU-ROC, AU-PRC, Log-loss.
- Choosing Metrics
- Class Imbalance
  - Failure scenarios for each metric
- Multi-class

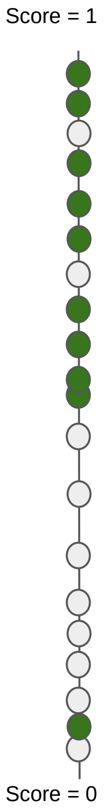
Why are metrics important?

- Training objective (cost function) is only a proxy for real world objectives.
- Metrics help capture a business goal into a quantitative target (not all errors are equal).
- Helps organize ML team effort towards that target.
    - Generally in the form of improving that metric on the dev set.
- Useful to quantify the “gap” between:
    - Desired performance and baseline (estimate effort initially).
    - Desired performance and current performance.
    - Measure progress over time.
- Useful for lower level tasks and debugging (e.g. diagnosing bias vs variance).
- Ideally training objective should be the metric, but not always possible. Still, metrics are useful and important for evaluation.

Binary Classification

- x is input
- y is binary output (0/1)
- Model is $\hat{y} = h(x)$
- Two types of models
  - Models that output a categorical class directly (K-nearest neighbor, Decision tree)
  - Models that output a real valued score (SVM, Logistic Regression)
    - Score could be margin (SVM), probability (LR, NN)
    - Need to pick a threshold
    - We focus on this type (the other type can be interpreted as an instance)

Score based models

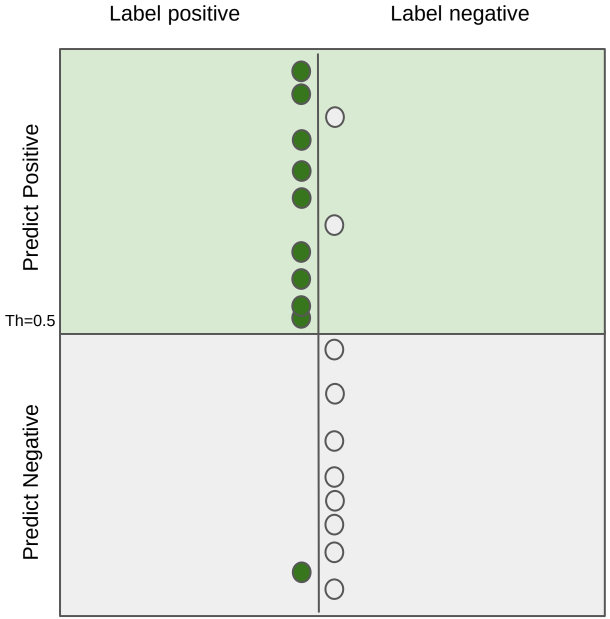


●	Positive example
○	Negative example

Example of Score: Output of logistic regression.
For most metrics: Only ranking matters.
If too many examples: Plot class-wise histogram.

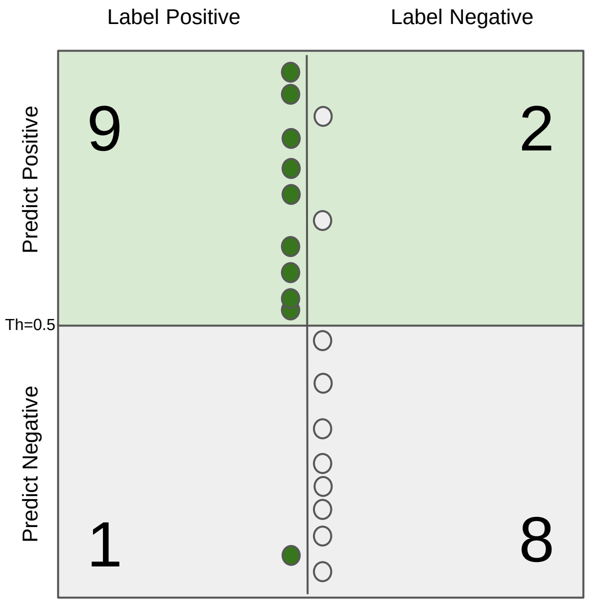
$$\text{Prevalence} = \frac{\#\text{ positive examples}}{\#\text{ positive examples} + \#\text{ negatives examples}}$$

Threshold -> Classifier -> Point Metrics



Th
0.5

Point metrics: Confusion Matrix

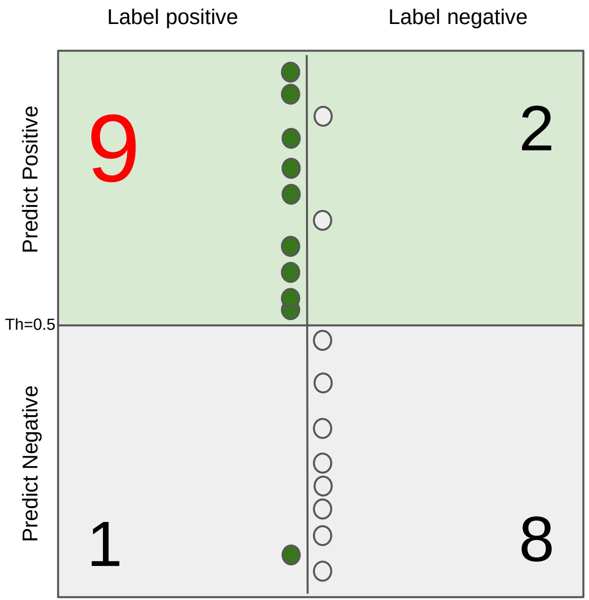


Th
0.5

Properties:

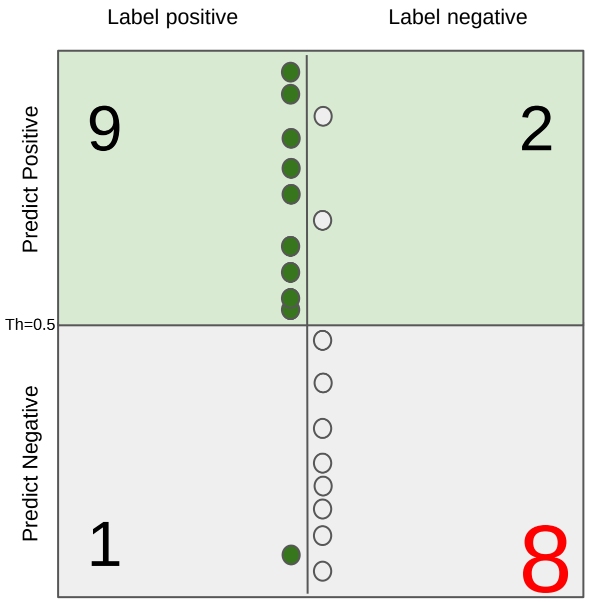
- Total sum is fixed (population).
- Column sums are fixed (class-wise population).
- Quality of model & threshold decide how columns are split into rows.
- We want diagonals to be “heavy”, off diagonals to be “light”.

Point metrics: True Positives



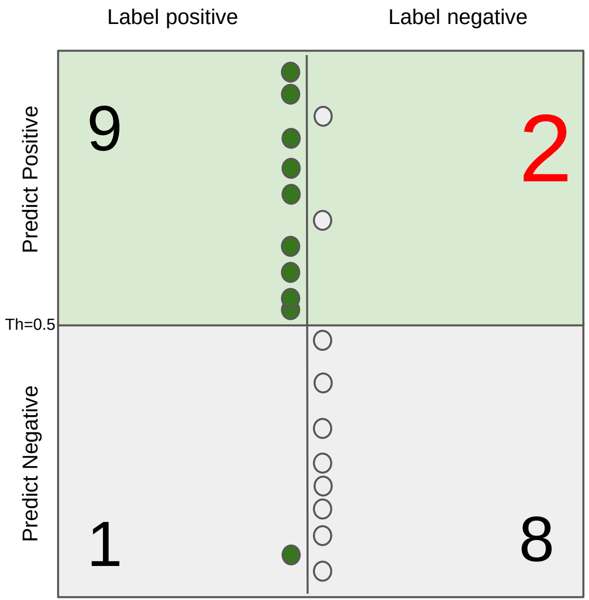
Th	TP
0.5	9

Point metrics: True Negatives



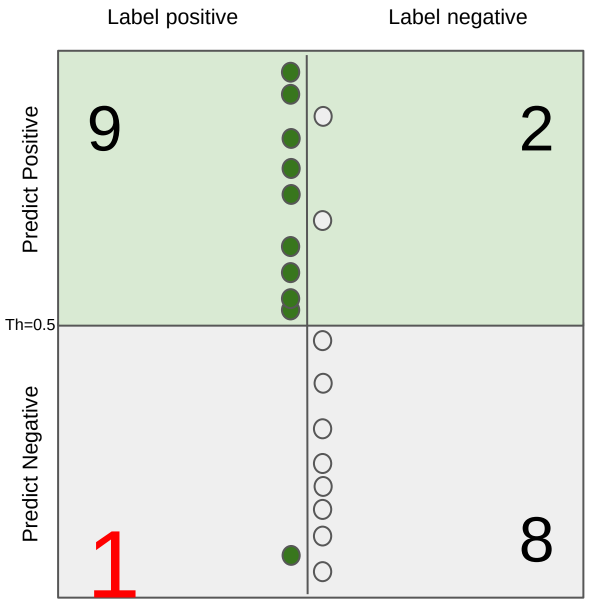
Th	TP	TN
0.5	9	8

Point metrics: False Positives



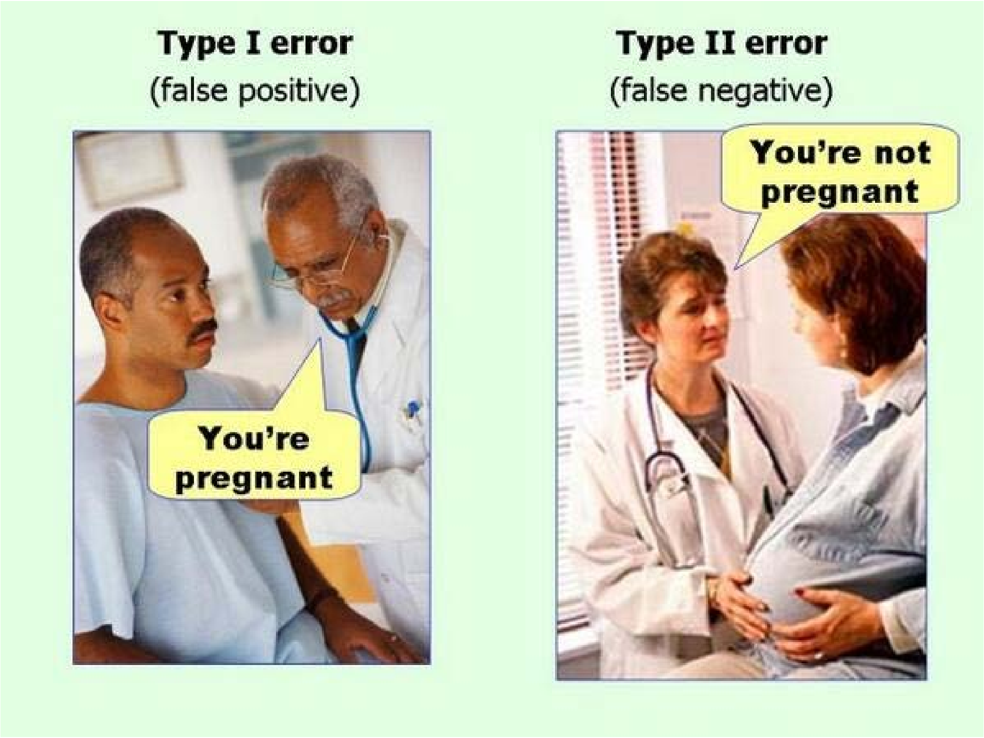
Th	TP	TN	FP
0.5	9	8	2

Point metrics: False Negatives



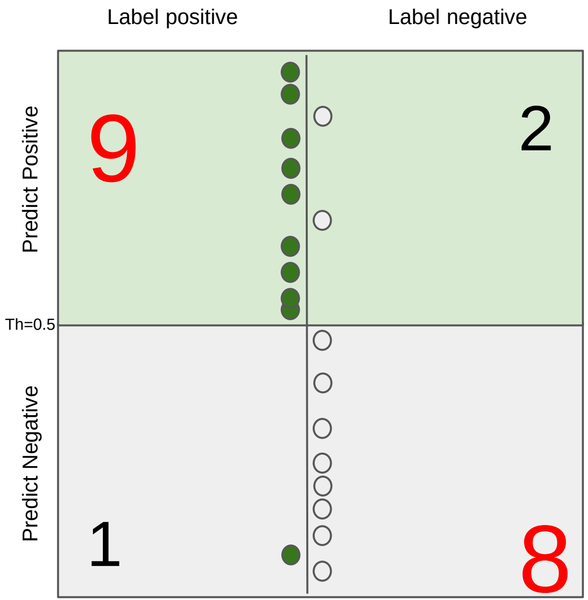
Th	TP	TN	FP	FN
0.5	9	8	2	1

FP and FN also called Type-1 and Type-2 errors



Could not find true source of image to cite

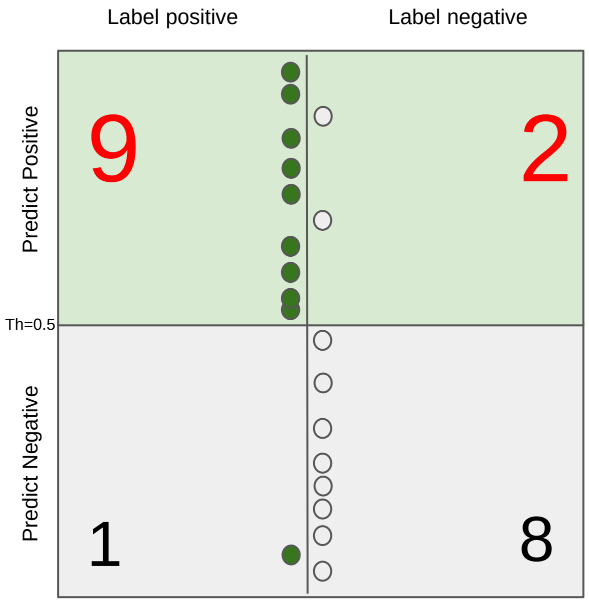
Point metrics: Accuracy



Th	TP	TN	FP	FN	Acc
0.5	9	8	2	1	.85

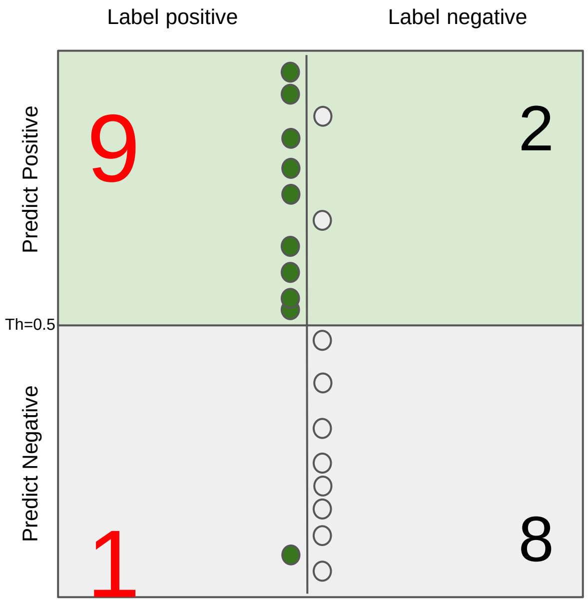
Equivalent to 0-1 Loss!

Point metrics: Precision



Th	TP	TN	FP	FN	Acc	Pr
0.5	9	8	2	1	.85	.81

Point metrics: Positive Recall (Sensitivity)



Th	TP	TN	FP	FN	Acc	Pr	Recall
0.5	9	8	2	1	.85	.81	.9

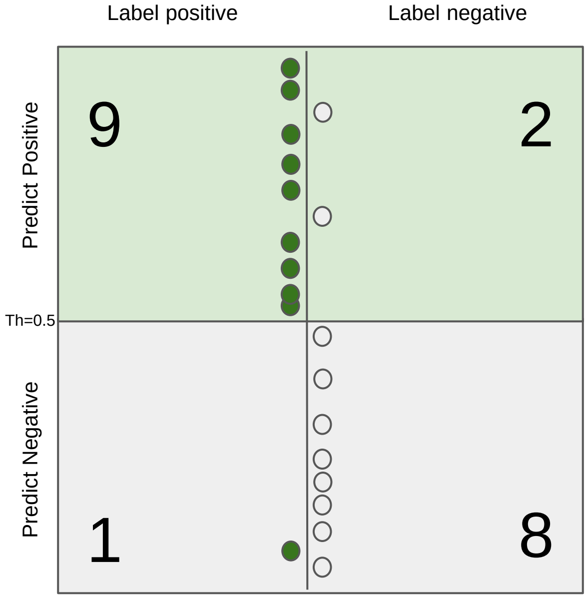
Trivial 100% recall = pull everybody above the threshold.
Trivial 100% precision = push everybody below the threshold except 1 green on top.
(Hopefully no gray above it!)

Striving for good precision with 100% recall =
pulling up the lowest green as high as possible in the ranking.
Striving for good recall with 100% precision =
pushing down the top gray as low as possible in the ranking.

Point metrics: Negative Recall (Specificity)

Th	TP	TN	FP	FN	Acc	Pr	Recall	Spec
0.5	9	8	2	1	.85	.81	.9	0.8

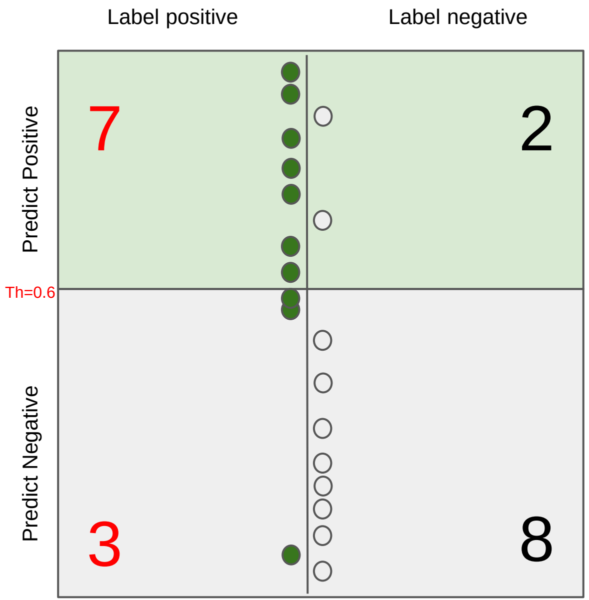
Point metrics: F1-score



Th	TP	TN	FP	FN	Acc	Pr	Recall	Spec	F1
0.5	9	8	2	1	.85	.81	.9	.8	.857

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

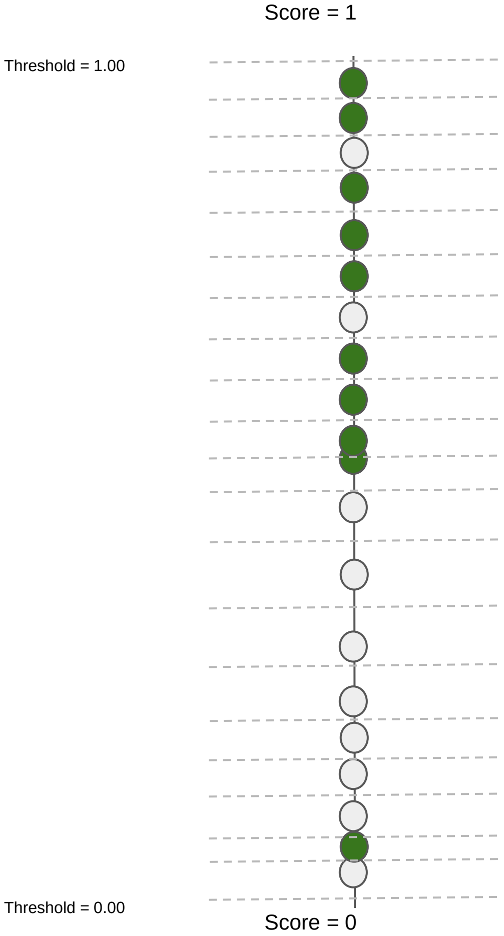
Point metrics: Changing threshold



Th	TP	TN	FP	FN	Acc	Pr	Recall	Spec	F1
0.6	7	8	2	3	.75	.77	.7	.8	.733

effective thresholds = # examples + 1

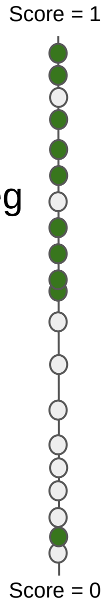
Threshold Scanning

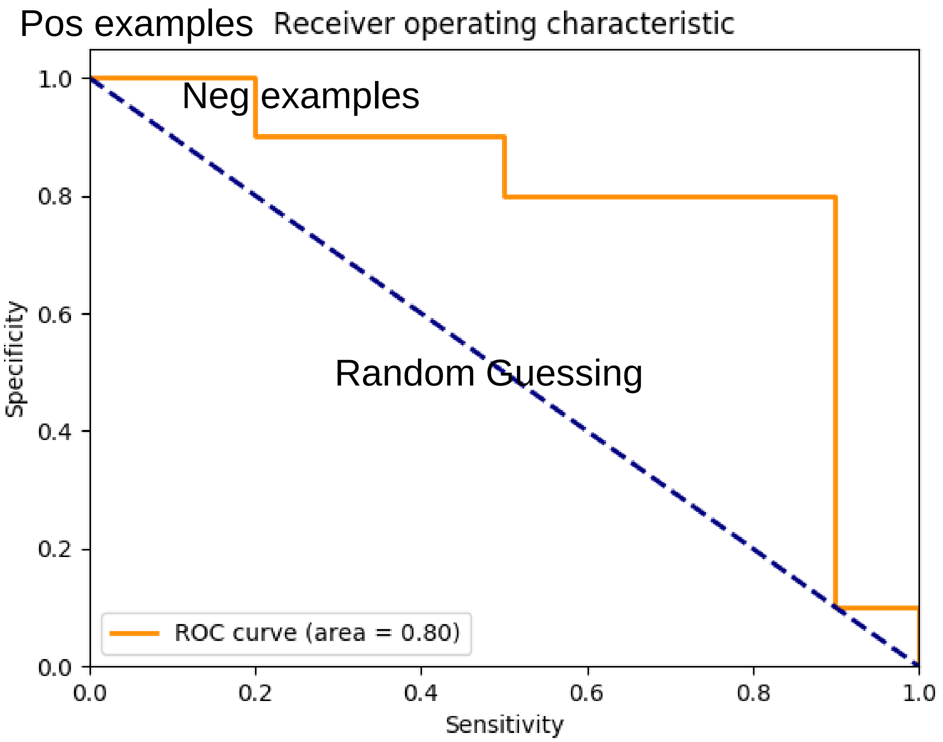


Threshold	TP	TN	FP	FN	Accuracy	Precision	Recall	Specificity	F1
1.00	0	10	0	10	0.50	1	0	1	0
0.95	1	10	0	9	0.55	1	0.1	1	0.182
0.90	2	10	0	8	0.60	1	0.2	1	0.333
0.85	2	9	1	8	0.55	0.667	0.2	0.9	0.308
0.80	3	9	1	7	0.60	0.750	0.3	0.9	0.429
0.75	4	9	1	6	0.65	0.800	0.4	0.9	0.533
0.70	5	9	1	5	0.70	0.833	0.5	0.9	0.625
0.65	5	8	2	5	0.65	0.714	0.5	0.8	0.588
0.60	6	8	2	4	0.70	0.750	0.6	0.8	0.667
0.55	7	8	2	3	0.75	0.778	0.7	0.8	0.737
0.50	8	8	2	2	0.80	0.800	0.8	0.8	0.800
0.45	9	8	2	1	0.85	0.818	0.9	0.8	0.857
0.40	9	7	3	1	0.80	0.750	0.9	0.7	0.818
0.35	9	6	4	1	0.75	0.692	0.9	0.6	0.783
0.30	9	5	5	1	0.70	0.643	0.9	0.5	0.750
0.25	9	4	6	1	0.65	0.600	0.9	0.4	0.720
0.20	9	3	7	1	0.60	0.562	0.9	0.3	0.692
0.15	9	2	8	1	0.55	0.529	0.9	0.2	0.667
0.10	9	1	9	1	0.50	0.500	0.9	0.1	0.643
0.05	10	1	9	0	0.55	0.526	1	0.1	0.690
0.00	10	0	10	0	0.50	0.500	1	0	0.667

Summary metrics: Rotated ROC (Sen vs. Spec)

Specificity
= True Neg / Neg





AUROC = Area Under ROC

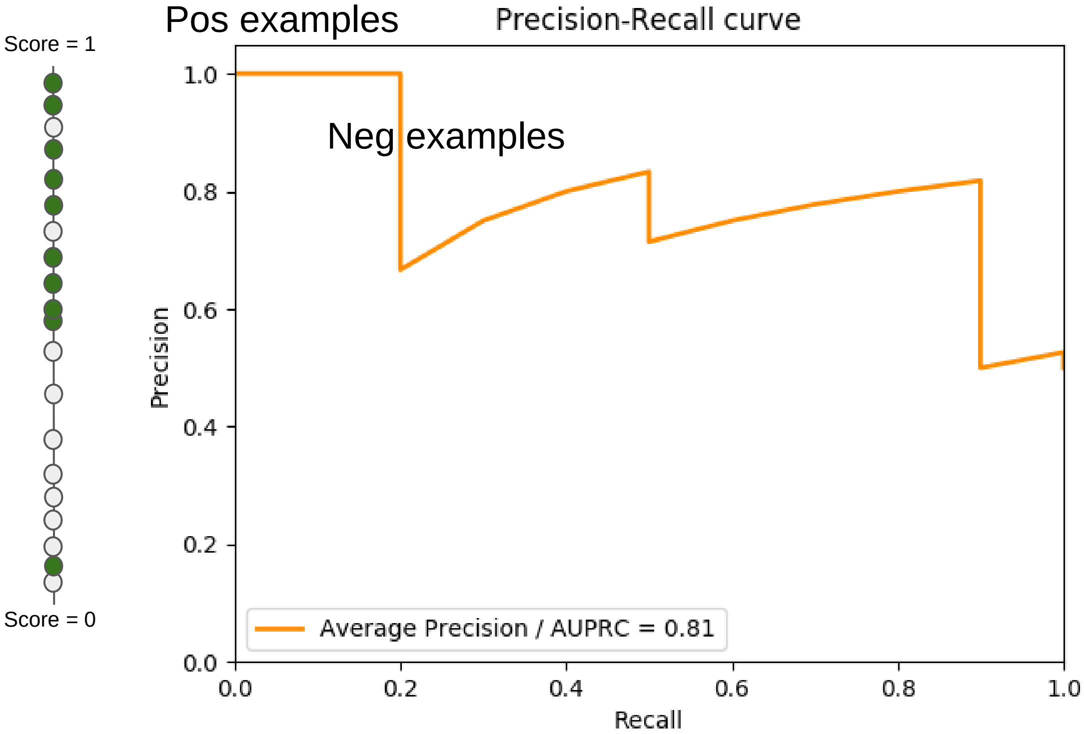
= Prob[Random Pos ranked higher than random Neg]

Agnostic to prevalence!

Sensitivity = True Pos / Pos

Summary metrics: PRC (Recall vs. Precision)

Precision
= True Pos /
Predicted Pos



AUPRC = Area Under PRC

= Expected precision for
Random threshold

When threshold = 0:
Precision = prevalence

Recall = Sensitivity = True Pos / Pos

Summary metrics:

Two models scoring the same data set. Is one of them better than the other?

Summary metrics: Log-Loss vs Brier Score

- Same ranking, and therefore the same AUROC, AUPRC, accuracy!

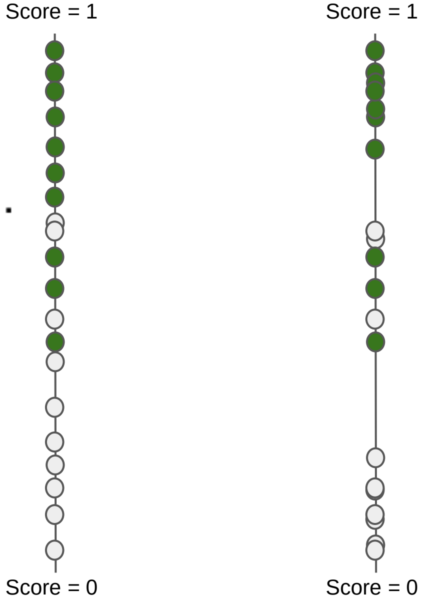
$$\text{Log Loss} = \frac{1}{N}\sum_{i=1}^{N} -y_i \log \hat{y}_i - (1-y_i)\log(1-\hat{y}_i).$$

- Rewards confident correct answers, heavily penalizes confident wrong answers.
- One perfectly confident wrong prediction is fatal.

-> Well-**calibrated** model

- **Proper** scoring rule: Minimized at $\hat{y} = y$

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



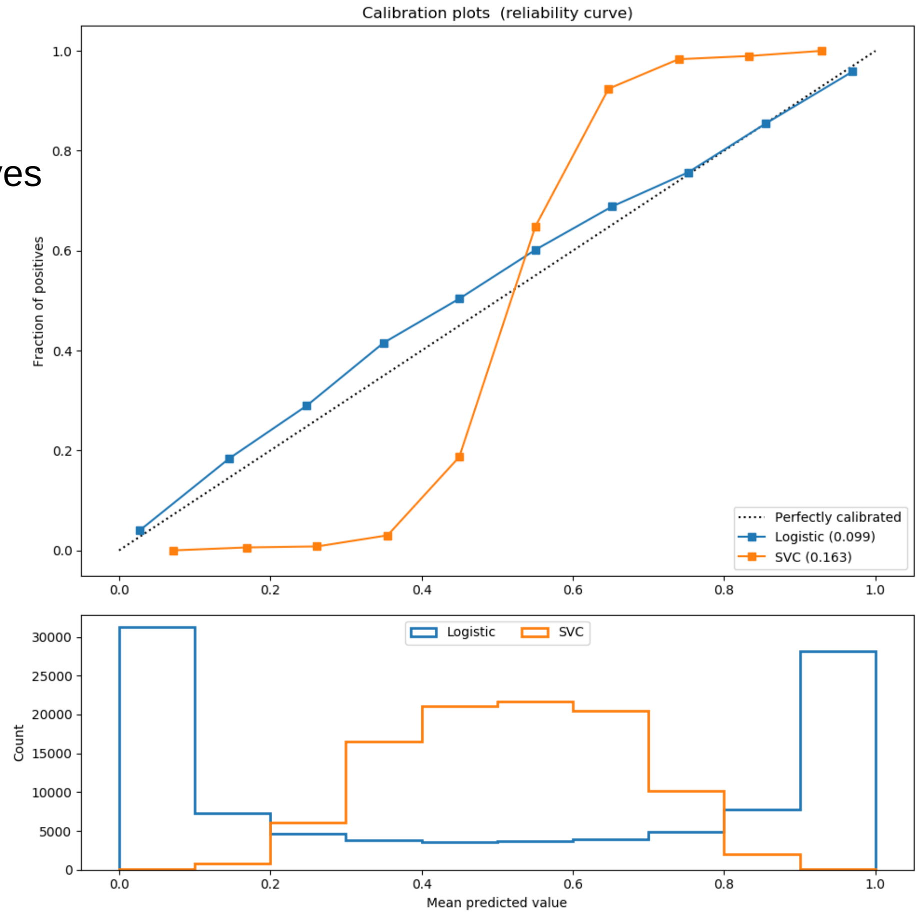
Calibration vs Discriminative Power

Logistic (th=0.5):
  Precision: 0.872
  Recall: 0.851
  F1: 0.862
  Brier: 0.099

SVC (th=0.5):
  Precision: 0.872
  Recall: 0.852
  F1: 0.862
  Brier: 0.163

Fraction of Positives

Histogram



Output

Unsupervised Learning

- Log P(x) is a measure of fit in Probabilistic models (GMM, Factor Analysis)
  - High log P(x) on training set, but low log P(x) on test set is a measure of overfitting
  - Raw value of log P(x) hard to interpret in isolation
- K-means is trickier (because of fixed covariance assumption)

Class Imbalance

Symptom: Prevalence < 5% (no strict definition)

Metrics: May not be meaningful.

Learning: May not focus on minority class examples at all

(majority class can overwhelm logistic regression, to a lesser extent SVM)

What happen to the metrics under class imbalance?

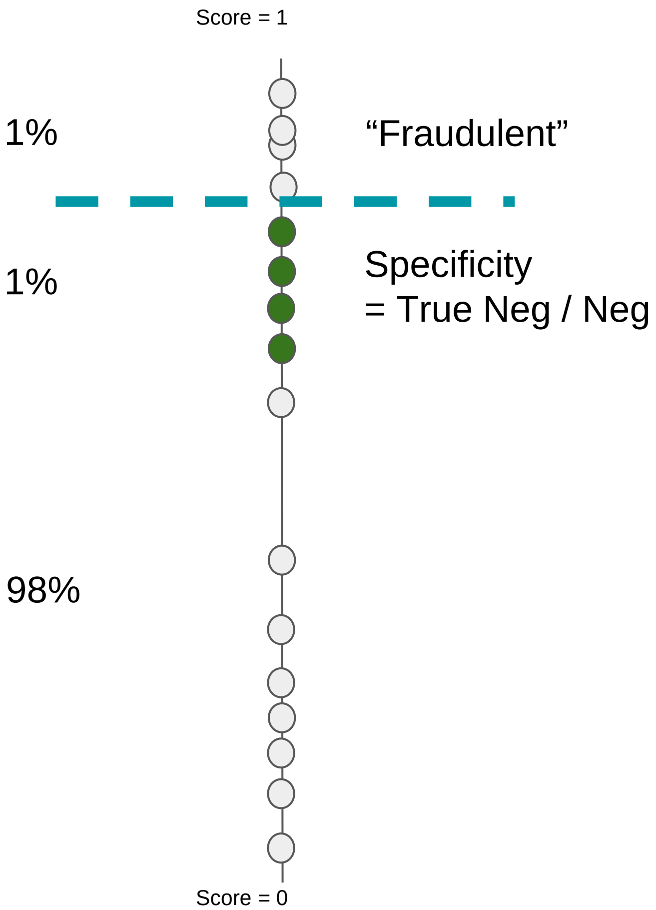
Accuracy: Blindly predicts majority class -> prevalence is the baseline.

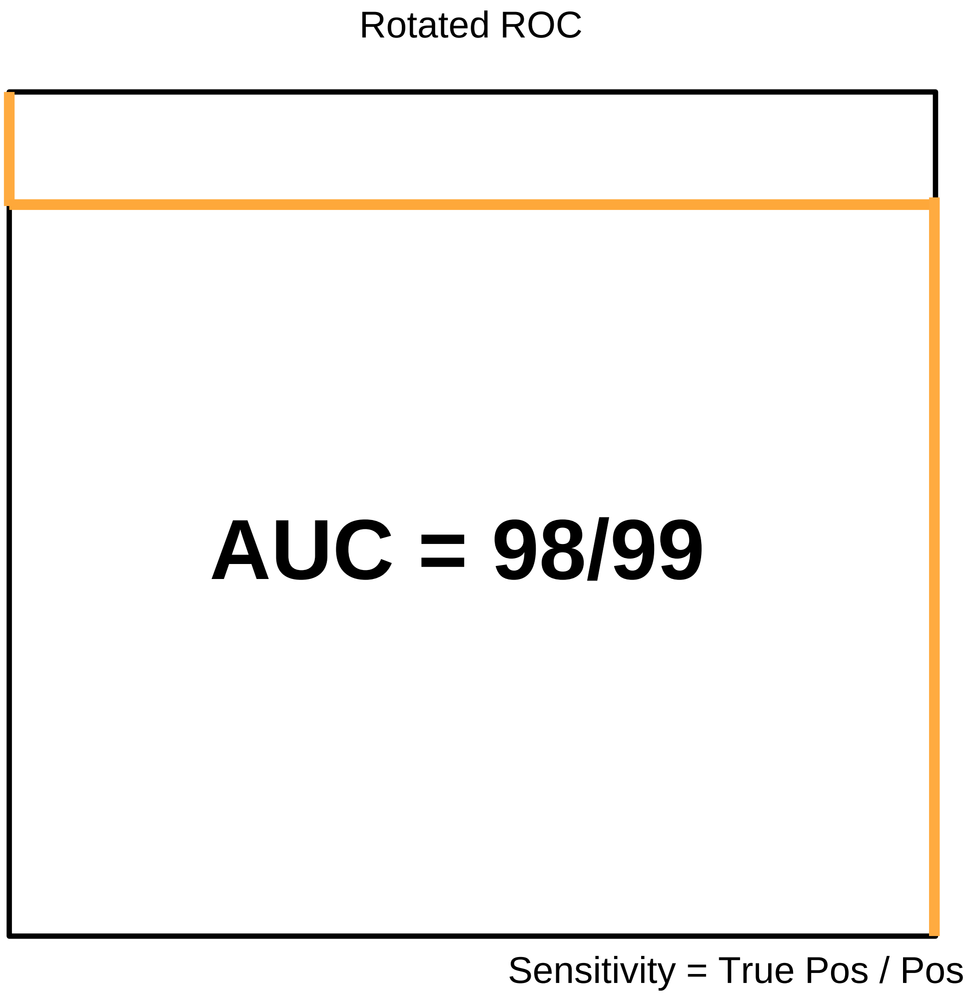
Log-Loss: Majority class can dominate the loss.

AUROC: Easy to keep AUC high by scoring most negatives very low.

AUPRC: Somewhat more robust than AUROC. But other challenges.

In general: Accuracy < AUROC < AUPRC


Score = 1
1%
"Fraudulent"
1%
Specificity
= True Neg / Neg
98%
Score = 0


Rotated ROC
AUC = 98/99
Sensitivity = True Pos / Pos

Multi-class

- Confusion matrix will be N * N (still want heavy diagonals, light off-diagonals)
- Most metrics (except accuracy) generally analyzed as multiple 1-vs-many
- Multiclass variants of AUROC and AUPRC (micro vs macro averaging)
- Class imbalance is common (both in absolute and relative sense)
- Cost sensitive learning techniques (also helps in binary Imbalance)
  - Assign weights for each block in the confusion matrix.
  - Incorporate weights into the loss function.

Choosing Metrics

Some common patterns:

- High precision is hard constraint, do best recall (search engine results, grammar correction): Intolerant to FP
    - Metric: Recall at Precision = XX %
- High recall is hard constraint, do best precision (medical diagnosis): Intolerant to FN
    - Metric: Precision at Recall = 100 %
- Capacity constrained (by K)
    - Metric: Precision in top-K.
- ……

Thank You!