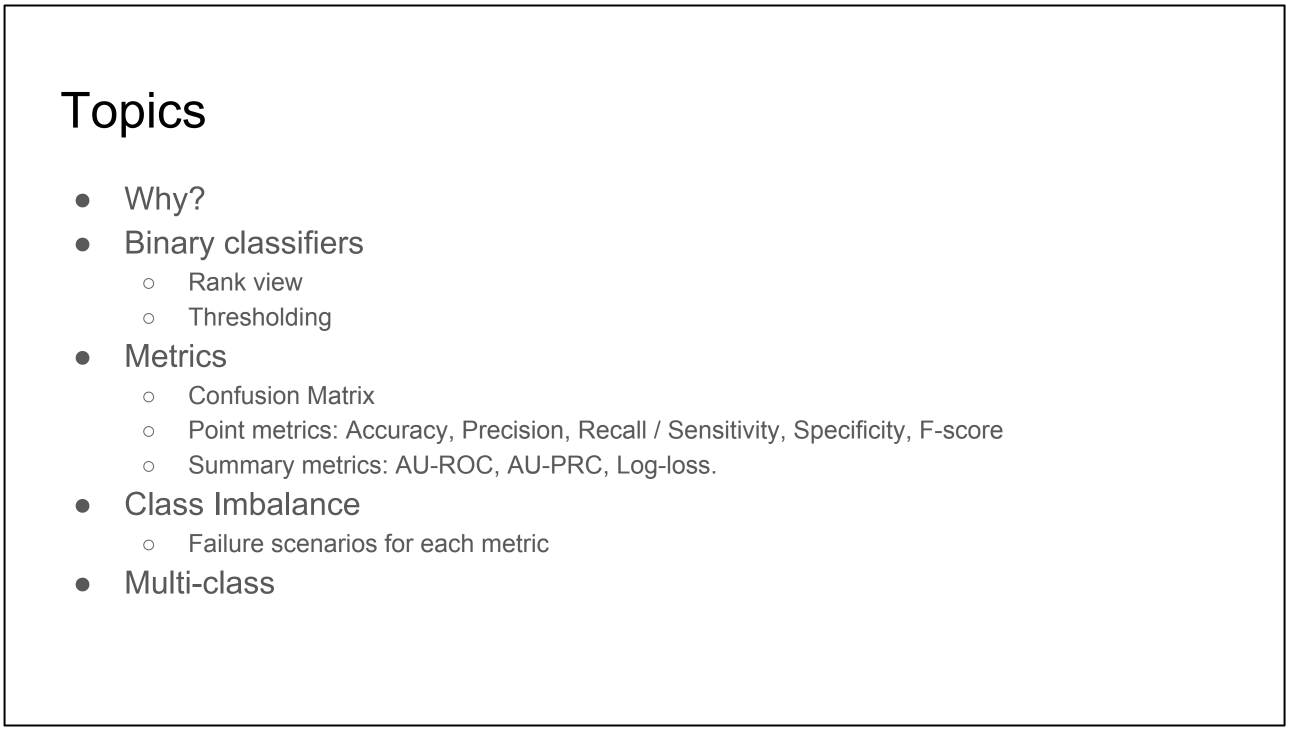
Evaluation Metrics

(Classifiers)

CS229 Section

Anand Avati



Topics

- Why?
- 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.
- Class Imbalance
  - Failure scenarios for each metric
- Multi-class

Only math necessary: Counting and ratios!

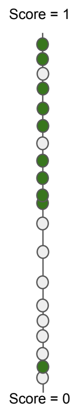
Why are metrics important?

- Training objective (cost function) is only a proxy for real world objective.
- 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 (No Free Lunch Theorem).
- Useful for lower level tasks and debugging (like 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 Yhat = 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



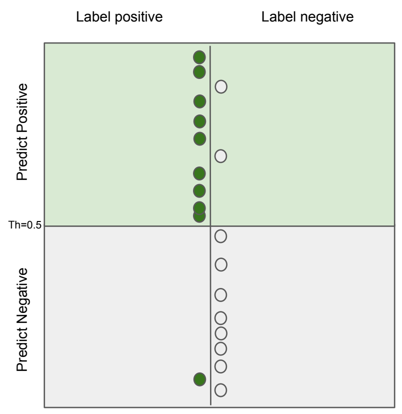
●	Positive labelled example
○	Negative labelled example

$$\text{Prevalence} = \frac{\#\text{positives}}{\#\text{positives} + \#\text{negatives}}$$

Dots are examples: Green = positive label, gray = negative label.
Consider score to be probability output of Logistic Regression.
Position of example on scoring line based on model output. For most of the metrics, only the relative rank / ordering of positive/negative matter.
Prevalence decides class imbalance.
If too many examples, plot class-wise histogram instead.

Rank view useful to start error analysis (start with the topmost negative example, or bottom-most positive example).

Score based models : Classifier

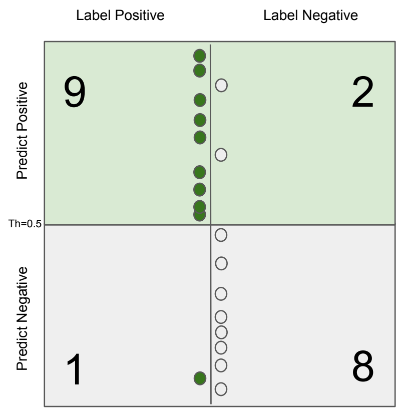


Th
0.5

Only after picking a threshold, we have a classifier.

Once we have a classifier, we can obtain all the Point metrics of that classifier.

Point metrics: Confusion Matrix



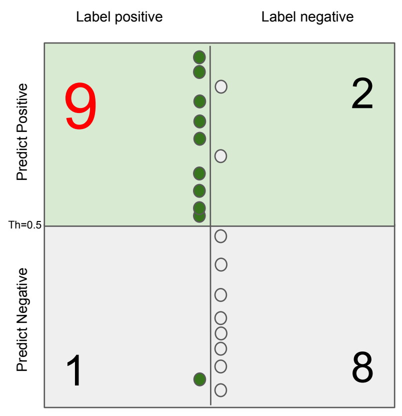
Th
0.5

All point metrics can be derived from the confusion matrix. Confusion matrix captures all the information about a classifier performance, but is not a scalar!

Properties:

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

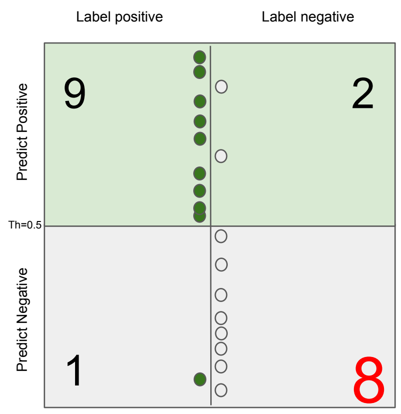
Point metrics: True Positives



Th	TP
0.5	9

TP = Green examples correctly identified as green.

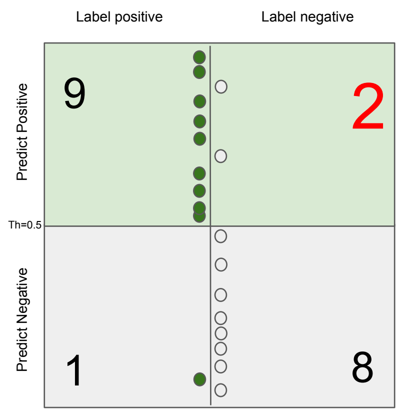
Point metrics: True Negatives



Th	TP	TN
0.5	9	8

TN = Gray examples correctly identified as gray. “Trueness” is along the diagonal.

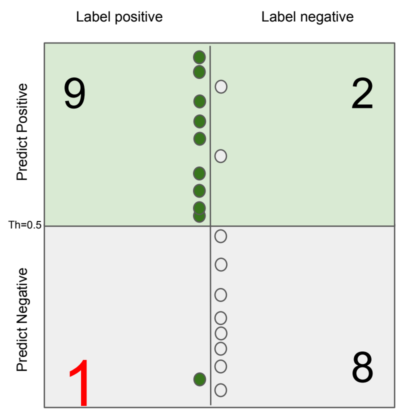
Point metrics: False Positives



Th	TP	TN	FP
0.5	9	8	2

FP = Gray examples falsely identified as green.

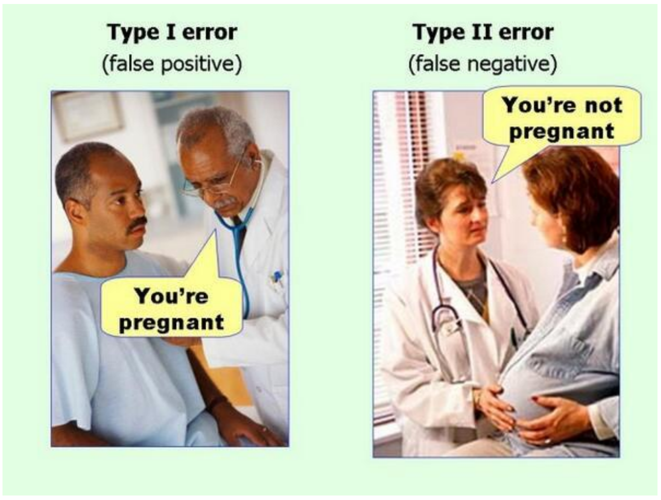
Point metrics: False Negatives



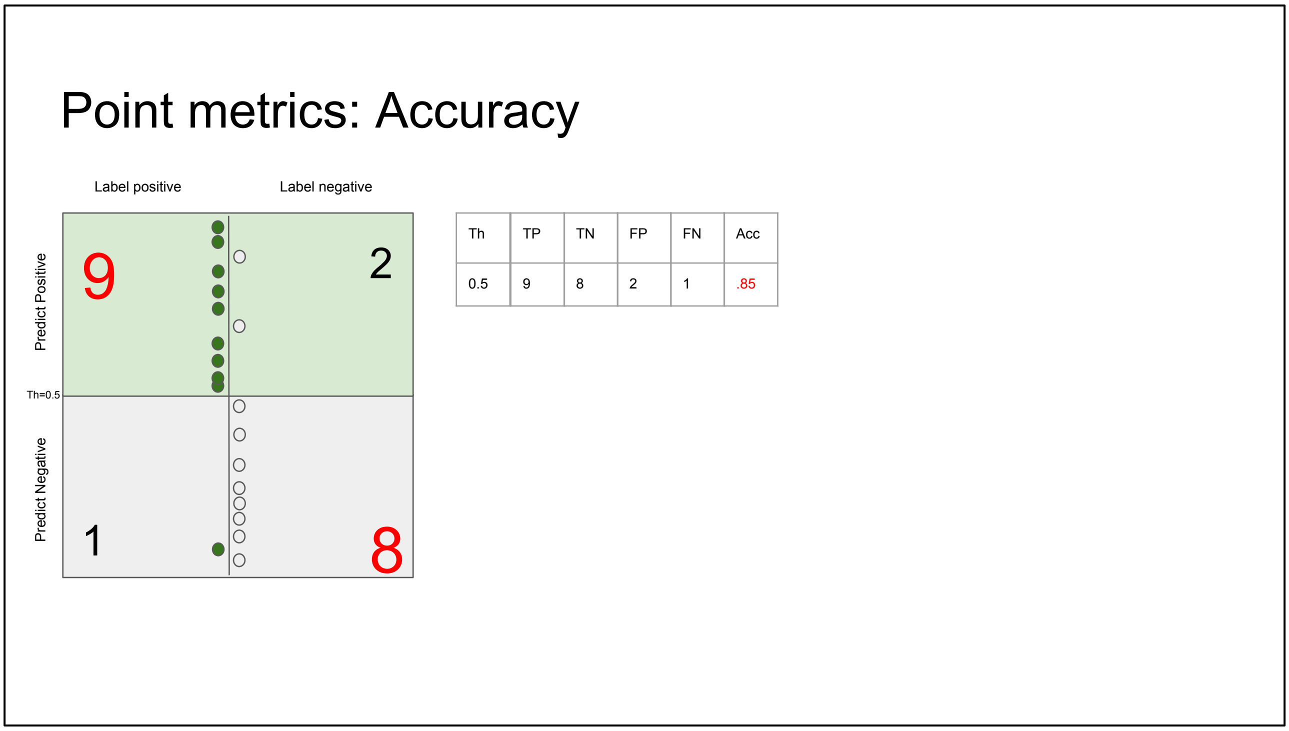
Th	TP	TN	FP	FN
0.5	9	8	2	1

FN = Green examples falsely identified as gray.

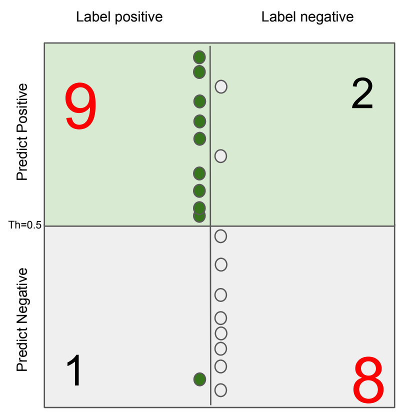
FP and FN also called Type-1 and Type-2 errors



Could not find true source of image to cite



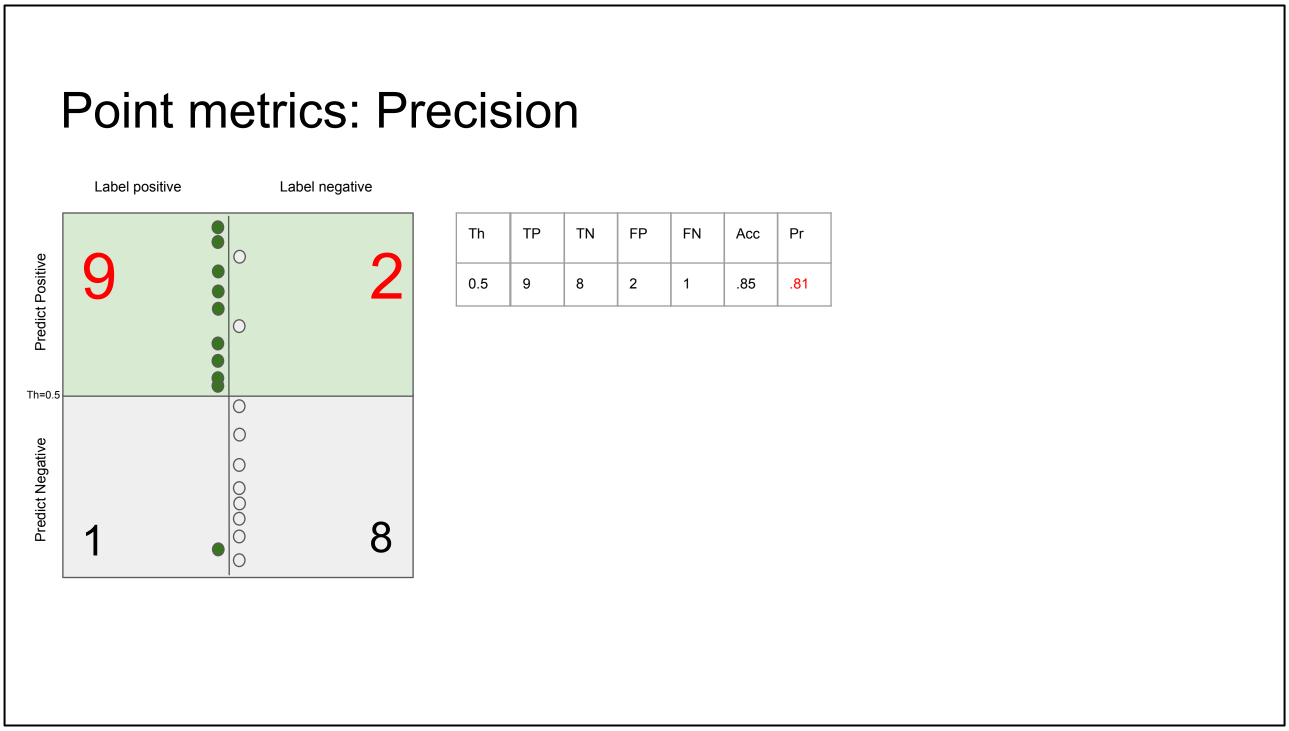
Point metrics: Accuracy



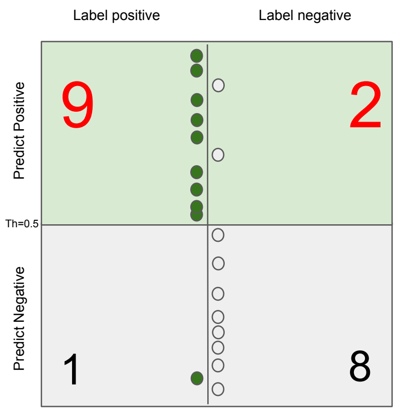
Th	TP	TN	FP	FN	Acc
0.5	9	8	2	1	.85

Accuracy = what fraction of all predictions did we get right?

Improving accuracy is equivalent to having a 0-1 Loss function.



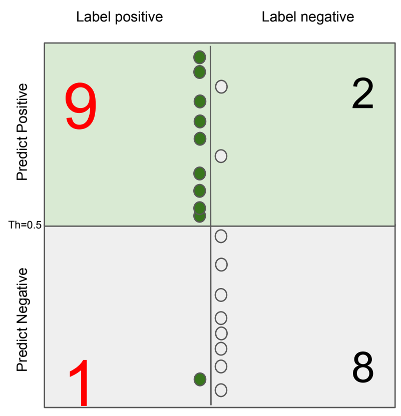
Point metrics: Precision



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

Precision = quality of positive predictions.
Also called PPV (Positive Predictive Value).

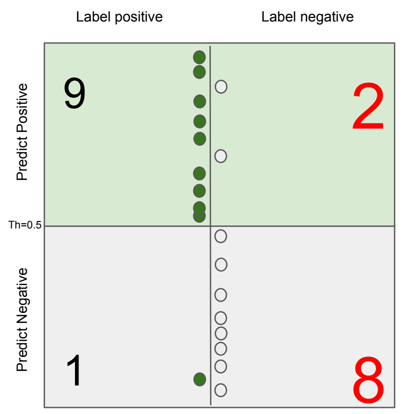
Point metrics: Positive Recall (Sensitivity)



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

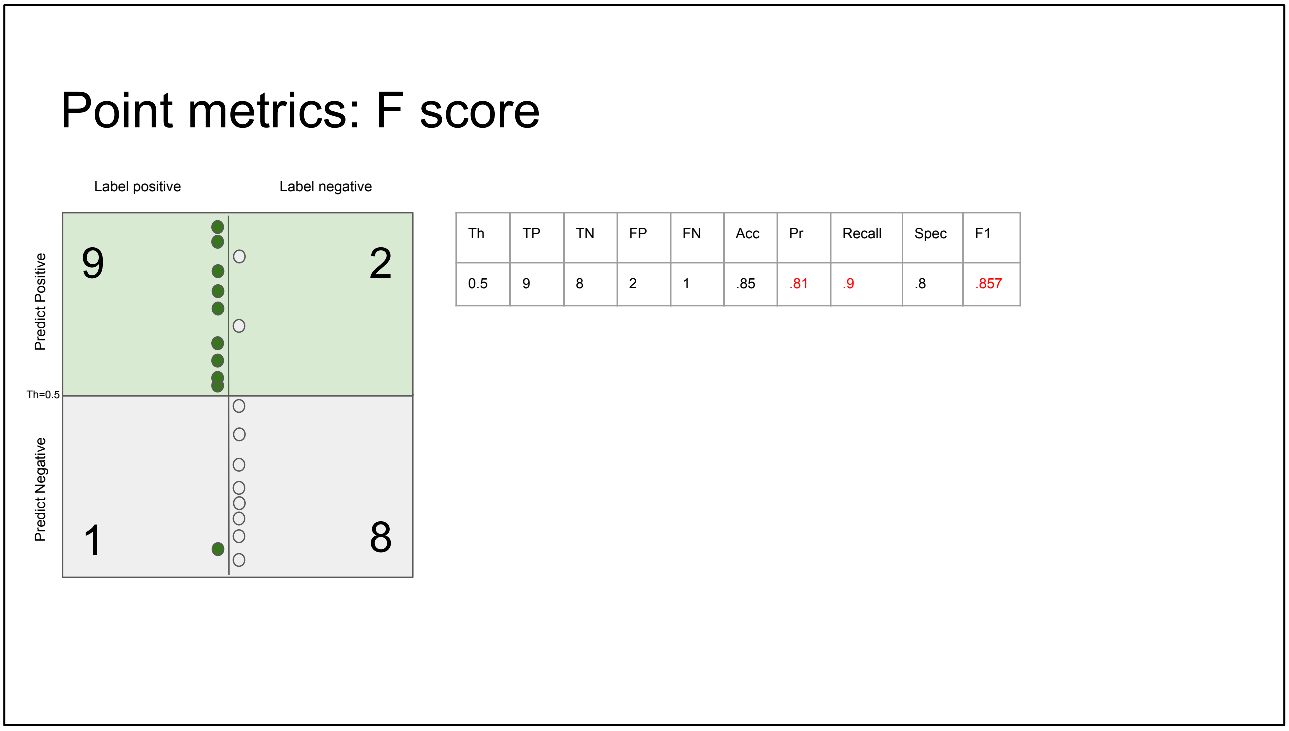
Recall (sensitivity) - What fraction of all greens did we pick out? Terminology from lab tests: how sensitive is the test in detecting disease?
Somewhat related to Precision (both recall and precision involve TP)
Trivial 100% recall = pull everybody above the threshold.
Trivial 100% precision = push 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 most gray as low as possible in the ranking.

Point metrics: Negative Recall (Specificity)

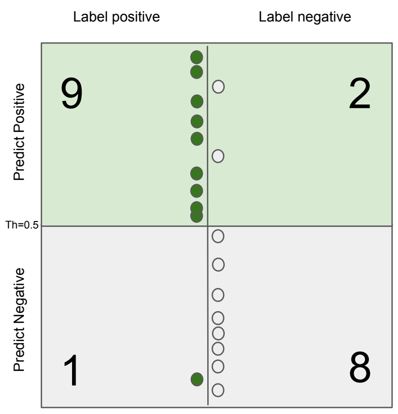


Th	TP	TN	FP	FN	Acc	Pr	Recall	Spec
0.5	9	8	2	1	.85	.81	.9	0.8

Negative Recall (Specificity) - What fraction of all negatives in the population were we able to keep away?
Note: Sensitivity and specificity operate in different sub-universes.



Point metrics: F score

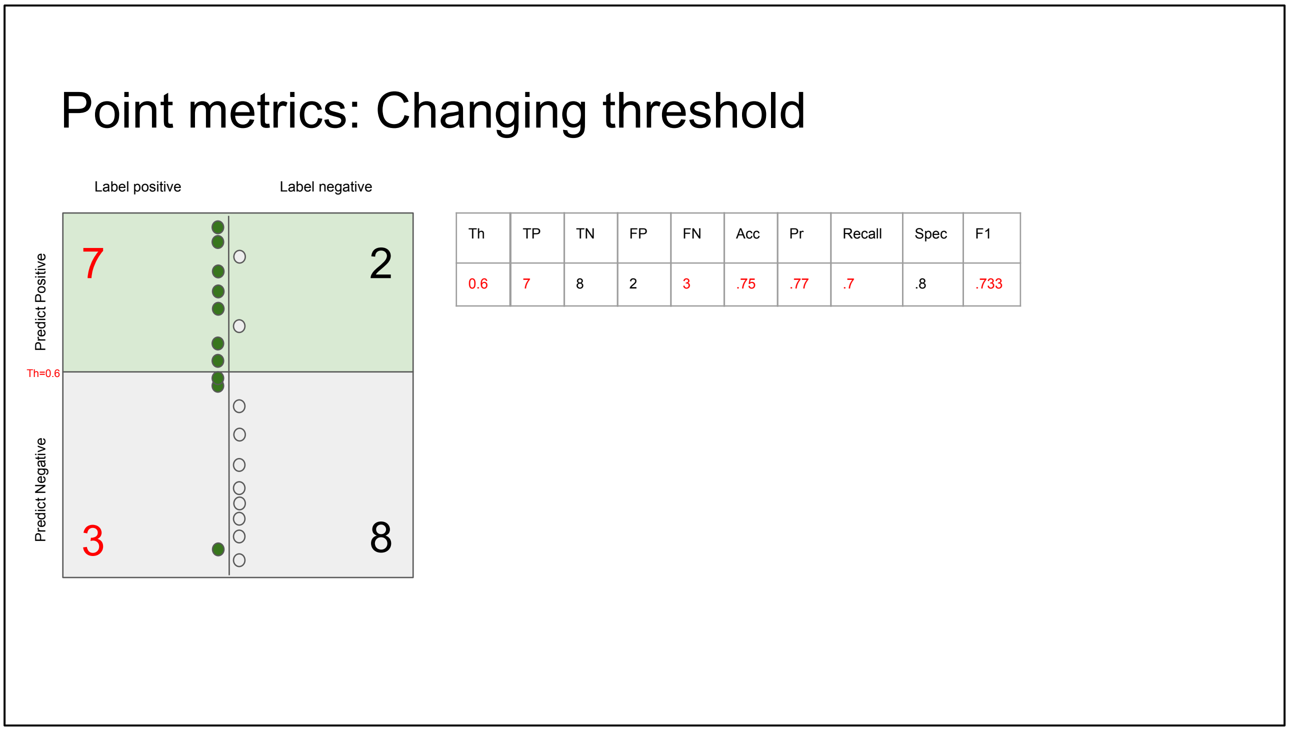


Th	TP	TN	FP	FN	Acc	Pr	Recall	Spec	F1
0.5	9	8	2	1	.85	.81	.9	.8	.857

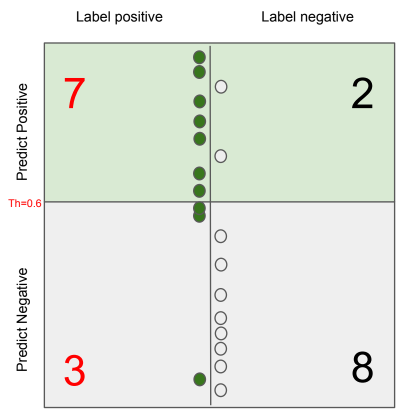
Summarize precision and recall into single score

F1-score: Harmonic mean of PR and REC

G-score: Geometric mean of PR and REC



Point metrics: Changing threshold

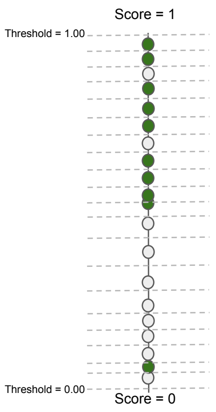


Th	TP	TN	FP	FN	Acc	Pr	Recall	Spec	F1
0.6	7	8	2	3	.75	.77	.7	.8	.733

Changing the threshold can result in a new confusion matrix, and new values for some of the metrics.

Many threshold values are redundant (between two consecutively ranked examples). Number of effective thresholds = M + 1 (# of 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

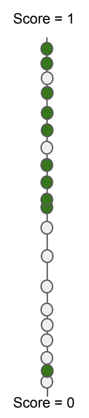
As we scan through all possible effective thresholds, we explore all the possible values the metrics can take on for the given model.
Each row is specific to the threshold.
Table is specific to the model (different model = different effective thresholds).
Observations: Sensitivity monotonic, Specificity monotonic in opposite direction.
Precision somewhat correlated with specificity, and not monotonic. Tradeoff between Sensitivity and Specificity, Precision and Recall.

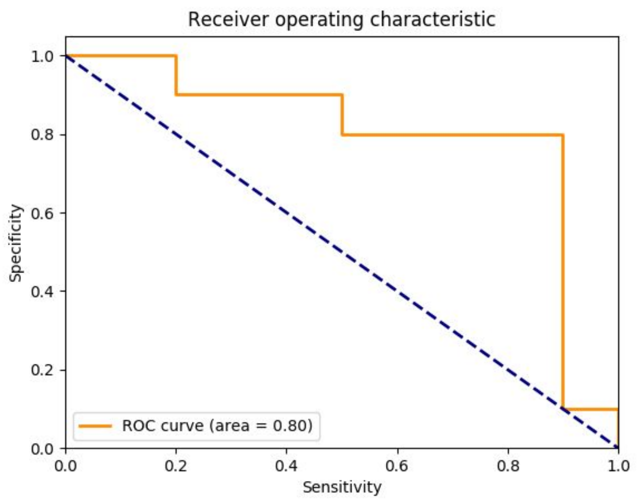
How to summarize the trade-off?

{Precision, Specificity}   vs   Recall/Sensitivity

We may not want to pick a threshold up front, but still want to summarize the overall model performance.

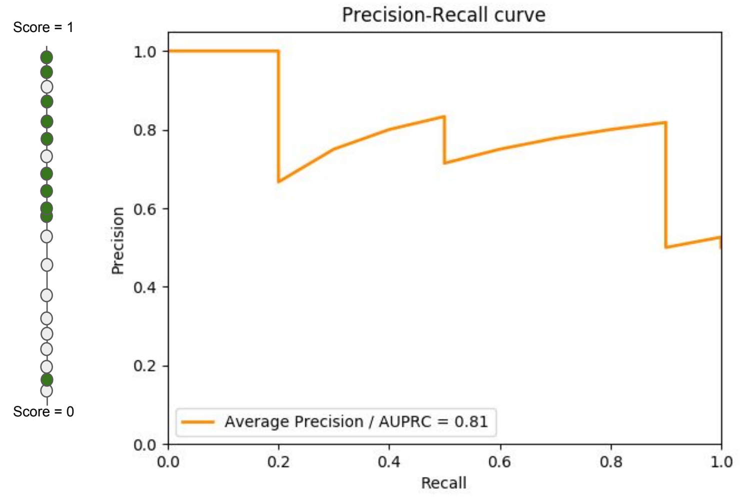
Summary metrics: ROC (rotated version)





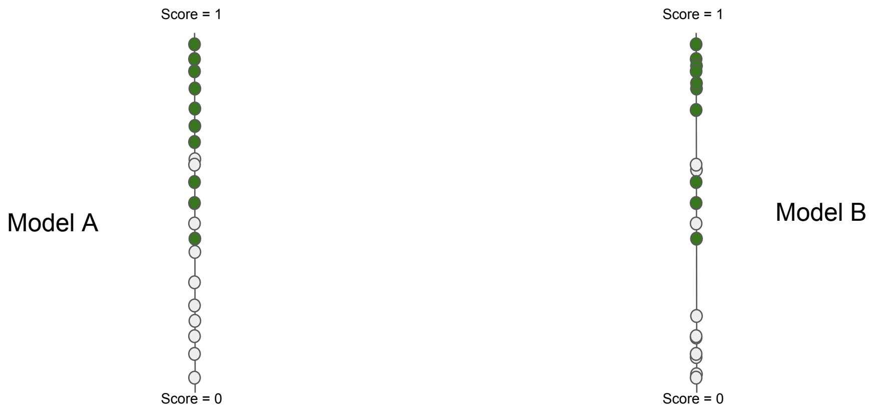
AUC = Area Under Curve. Also called C-Statistic (concordance score). Represents how well the results are ranked. If you pick a positive e.g by random, and negative e.g by random, AUC = probability that positive eg is ranked > negative example. Measure of quality of discrimination.
Some people plot TPR (= Sensitivity) vs FPR (= 1 - Specificity).
Thresholds are points on this curve. Each point represents one set of point metrics.
Diagonal line = random guessing.

Summary metrics: PRC



Precision Recall curve represents a different tradeoff. Think search engine results ranked.
End of curve at right cannot be lower than prevalence.
Area under PRC = Average Precision. Intuition: By randomly picking the threshold, what’s the expected precision?
While sensitivity monotonically decreased with increasing recall (the number of correctly classified negative examples can only go down), precision can be “jagged”. Getting a bunch of positives in the sequence increases both precision and recall (hence curve climbs up slowly), but getting a bunch of negatives in the sequence drops the precision without increasing recall. Hence, by definition, curve has only slow climbs and vertical drops.

Summary metrics: Log-Loss motivation

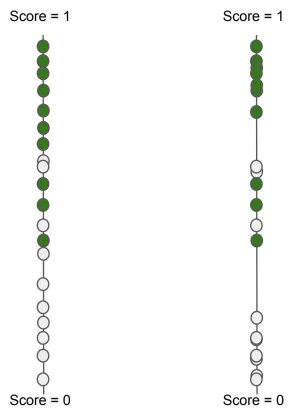


Two models scoring the same data set. Is one of them better than the other?

Somehow the second model feels “better”.

Summary metrics: Log-Loss

- These two model outputs have same ranking, and therefore the same AU-ROC, AU-PRC, accuracy!
- Gain = p(x)*y + (1-p(x))*(1-y)
- Log loss rewards confident correct answers and heavily penalizes confident wrong answers.
- exp(E[log-loss]) is G.M. of gains, in [0,1].
- Gaining popularity as an evaluation metric (Kaggle)



The metrics so far fail to capture this.
Accuracy optimizes for: Model gets $1 for correct answer, $0 for wrong.
Log-loss: Model makes a bet for $0.xy dollars that it’s prediction is correct. If correct, gets $0.xy dollars, else gets $(1-0.xy) dollar. Incentive structure is well suited to produce calibrated models. Take home is the GM of earned amounts.
Low-bar for exp(E[log-loss]) is 0.5
Log-loss on validation set measure how much did this “betting engine” earn.

Class Imbalance: Problems

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)

Scenarios: Mostly retrieval-like. Generally one of the patterns:

- High precision is hard constraint, optimize recall (e.g search engine results, grammar correction) -- intolerant to FP
- High recall is hard constraint, optimize precision (e.g medical test) -- intolerant to FN

Class Imbalance: Metrics (pathological cases)

Accuracy: Blindly predict majority class.

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.

- What kind of interpolation? AUCNPR?

In general: Accuracy << AUROC << AUPRC

For accuracy, prevalence of majority class should be the low bar!

AUROC: 10% prevalence. top-10% are all negatives, next K are all positives, followed by rest of the negatives. AUPRC = 0.9 even though practically useless.

Multi-class (few remarks)

- Confusion matrix will be NxN (still want heavy diagonals, light off-diagonals)
- Most metrics (except accuracy) generally analysed as multiple 1-vs-many.
- Class imbalance is common (both in absolute, and relative sense)
- Cost sensitive learning techniques (also helps in Binary Imbalance)
  - Assign $$ value for each block in the confusion matrix, and incorporate those into the loss function.

Thank You!