Precision and Recall

Lecture 6 - DAMLF | ML1

Ingredients:

Task *to perform*

Type of **Experience**

Performance *measure*

What is a Learning Algorithm?

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E"

–Tom Mitchell

Ingredients:

Task *to perform*

Type of **Experience**

Performance *measure*

Performance

- Numerical Performance Measures:
  - Cost functions used to fit training data
  - Cost functions used to assess generalization error

Ingredients:

Task *to perform*

Type of **Experience**

Performance *measure*

Performance

- Numerical Performance Measures:
  - Cost functions used to fit training data
  - Cost functions used to assess generalization error
- Other measures to assess generalization error

Banknote Authentication

Suppose there are 1000 examples in the **cross-validation** set ($m_{cv} = 1000$), and your algorithm misclassifies 200 forged Euro banknotes as genuine.

What can you do?

Banknote Authentication

1. **EDA**: Exploratory Data Analysis.
   Are specific types (denomination) of notes are misclassified?

   5, 10, 20, 50, etc.

[1]See UCI repository https://archive.ics.uci.edu/ml/datasets/banknote+authentication

Banknote Authentication

1. **EDA**: Exploratory Data Analysis.
   Are specific types (denomination) of notes are misclassified?

   5, 10, 20, 50, etc.

2. **Feature Engineering**: What features (cues) do **you** think would have helped the algorithm classify them correctly?[1]

   variance of Wavelet
   skewness of Wavelet
   curtosis of Wavelet
   entropy of image

[1]See UCI repository https://archive.ics.uci.edu/ml/datasets/banknote+authentication

Numerical Evaluation

It is important to incorporate a **numerical evaluation** of your algorithm, such as cross-validation error, in order to evaluate your algorithm's performance as you make changes to it.

However, care and thought is required in picking a metric.

Numerical evaluation and Skewed Classes

To illustrate why numerical evaluation requires **practical** judgment, consider the case of **skewed classes**.

Skewed Classes

Cancer Classification

Suppose you train a **logistic regression** model to detect malignant tumors, where

$y = 1$, if the tumor is malignant

$y = 0$, if the tumor is benign

and suppose you got a **1% error** on your test set, (i.e., 99% correct diagnoses).

Cancer Classification

Suppose you train a **logistic regression** model to detect malignant tumors, where

$y = 1$, if the tumor is malignant

$y = 0$, if the tumor is benign

and suppose you got a **1% error** on your test set, (i.e., 99% correct diagnoses).

However, suppose **only 0.5%** of patients have cancer (a malignant tumor).

Cancer Classification

Suppose you train a **logistic regression** model to detect malignant tumors, where

$y = 1$, if the tumor is malignant

$y = 0$, if the tumor is benign

and suppose you got a **1% error** on your test set, (i.e., 99% correct diagnoses).

However, suppose **only 0.5%** of patients have cancer (a malignant tumor).

In this situations, our 1% error no longer looks impressive:

Cancer Classification

Suppose you train a **logistic regression** model to detect malignant tumors, where

$y = 1$, if the tumor is malignant

$y = 0$, if the tumor is benign

and suppose you got a **1% error** on your test set, (i.e., 99% correct diagnoses).

However, suppose **only 0.5%** of patients have cancer (a malignant tumor).

In this situations, our 1% error no longer looks impressive:

```
def easyPredictCancer(x):
    y = 0  # ignore x
    return y
```

The non-learning `easyPredictCancer` has a 0.5% error rate!

Skewed Classes

Skewed classes appear when you have a lot more instances of some classes than of others.

Example: The occurrence of *malignant tumors*,

95% ($y = 0$)

5% ($y = 1$)

is an example of a skewed class.

Example: The occurrence of *credit card fraud*,

99.7% ($y = 0$)

0.3% ($y = 1$)

is an example of a skewed class.

Limits of Classification Accuracy

Numerical Accuracy Scores

It is useful to have a single numerical "score" to evaluate learning algorithms.

However, moving from

99.3% accuracy (0.70% error)

to

99.7% accuracy (0.30% error)

is **not always** an indication of an improvement to your classifier.

Precision and Recall

Consider two new quantities for making numerical evaluations.

Precision and Recall

Consider two new quantities for making numerical evaluations.

We are interested in detecting $y = 1$ for rare classes.

		Actual Class	
		1	0
Predicted Class	1		
	0		

Precision and Recall

Consider two new quantities for making numerical evaluations.

We are interested in detecting $y = 1$ for rare classes.

		Actual Class	
		1	0
Predicted Class	1	**true positive**	
	0		

Precision and Recall

Consider two new quantities for making numerical evaluations.

We are interested in detecting $y = 1$ for rare classes.

		Actual Class	
		1	0
Predicted Class	1	**true positive**	
	0		**true negative**

Precision and Recall

Consider two new quantities for making numerical evaluations.

We are interested in detecting $y = 1$ for rare classes.

		Actual Class	
		1	0
Predicted Class	1	**true positive**	**false positive**
	0		**true negative**

Precision and Recall

Consider two new quantities for making numerical evaluations.

We are interested in detecting $y = 1$ for rare classes.

		Actual Class	
		1	0
Predicted Class	1	**true positive**	**false positive**
	0	**false negative**	**true negative**

Precision and Recall

Confusion Matrix

		Actual Class: 1	**Actual** Class: 0
Predicted Class	1	true positive	false positive
	0	false negative	true negative

Precision

Of all the patients where we **predicted positive** $y = 1$, what fraction actually has cancer?

Recall

Of all the patients that actually has cancer, what fraction did we **correctly** detect as having cancer?

Precision and Recall

Confusion Matrix

		Actual Class: 1	**Actual** Class: 0
Predicted Class	1	true positive	false positive
	0	false negative	true negative

Precision

Of all the patients where we **predicted positive** $y = 1$, what fraction actually has cancer?

$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Recall

Of all the patients that actually has cancer, what fraction did we **correctly** detect as having cancer?

Precision and Recall

Confusion Matrix

		Actual Class: 1	**Actual** Class: 0
Predicted Class	1	true positive	false positive
	0	false negative	true negative

Precision

Of all the patients where we **predicted positive** $y = 1$, what fraction actually has cancer?

$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Recall

Of all the patients that actually has cancer, what fraction did we **correctly** detect as having cancer?

$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Precision & Recall

Precision: the frequency with which predictions are correct.
Other names for Precision:

- Positive Predictive Value (PPV)
- True Positive Accuracy (TPA)

Recall: the frequency with relevant elements are retrieved (recalled) by a system.

Other names for Recall:

- True Positive Rate (TPR)
- Sensitivity

Let's look again at

```
def easyPredictCancer(x):
    y = 0  # ignore x
    return y
```

[2]Note: true positive ($y = 1$) is the occurrence of the rare class we wish to detect.

Let's look again at

```
def easyPredictCancer(x):
    y = 0  # ignore x
    return y
```

Since $(y^{(i)} = 0)$ for all $x^{(i)}$, **true positive** $(y = 1) = 0$.[2]

So, both
Precision:

$$\frac{\textbf{True Positives}}{\textbf{True Positives} + \textbf{False Positives}} := \frac{\textbf{True Positives}}{\#\text{ of Predicted Positives}} = 0$$

Recall:

$$\frac{\textbf{True Positives}}{\textbf{True Positives} + \textbf{False Negatives}} := \frac{\textbf{True Positives}}{\#\text{ of Actual Positives}} = 0$$

[2]Note: true positive $(y = 1)$ is the occurrence of the rare class we wish to detect.

Intuitively, **precision** and **recall** catch what is wrong with `easyPredictCancer(x)`.

Intuitively, **precision** and **recall** catch what is wrong with `easyPredictCancer(x)`.

But a *numerical evaluation* promises the ability to make **meaningful comparisons**.

Intuitively, **precision** and **recall** catch what is wrong with `easyPredictCancer(x)`.

But a *numerical evaluation* promises the ability to make **meaningful comparisons**.

How does this work for **precision** and **recall**?

Trading off precision and recall

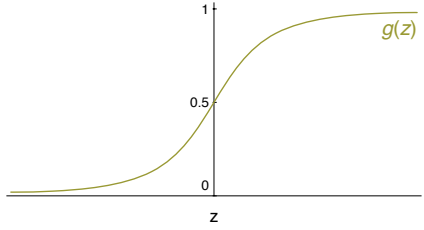
Precision and Recall

In practice there is a **trade off** between **precision** and **recall**.

Precision and Recall Trade-off

Logistic Regression: $0 \leqslant h(x;\theta) \leqslant 1$

Predict $(y = 1)$ if $h(x;\theta) \geqslant 0.5$
Predict $(y = 0)$ if $h(x;\theta) < 0.5$



Precision:

$$\frac{\textbf{True Positives}}{\text{\# of Predicted Positive}}$$

Recall:

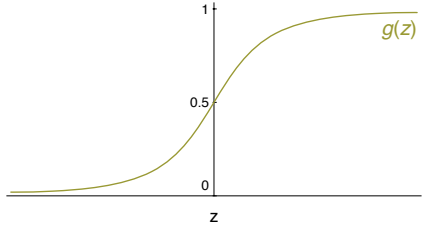
$$\frac{\textbf{True Positives}}{\text{\# of Actual Positive}}$$

Precision and Recall Trade-off

Logistic Regression: $0 \leqslant h(x;\theta) \leqslant 1$

Predict $(y = 1)$ if $h(x;\theta) \geqslant 0.5$
Predict $(y = 0)$ if $h(x;\theta) < 0.5$

Suppose we want to predict $y = 1$
only if we are very confident.



Precision:

$$\frac{\text{True Positives}}{\text{\# of Predicted Positive}}$$

Recall:

$$\frac{\text{True Positives}}{\text{\# of Actual Positive}}$$

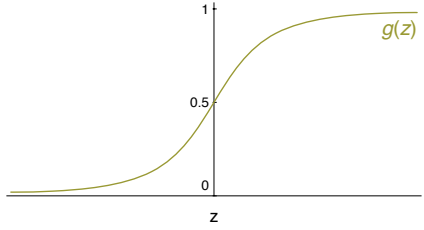
Precision and Recall Trade-off

Logistic Regression: $0 \leqslant h(x;\theta) \leqslant 1$

Predict $(y = 1)$ if $h(x;\theta) \geqslant 0.5$
Predict $(y = 0)$ if $h(x;\theta) < 0.5$

Suppose we want to predict $y = 1$
only if we are very confident.

We could do this by **modifying the threshold** points.



Precision:

$$\frac{\text{True Positives}}{\text{\# of Predicted Positive}}$$

Recall:

$$\frac{\text{True Positives}}{\text{\# of Actual Positive}}$$

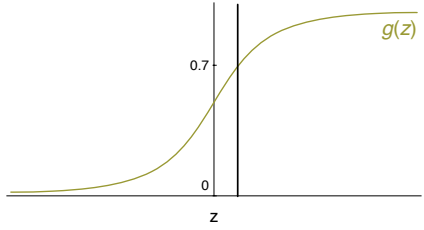
Precision and Recall Trade-off

Logistic Regression: $0 \leqslant h(x;\theta) \leqslant 1$

Predict $(y = 1)$ if $h(x;\theta) \geqslant 0.7$
Predict $(y = 0)$ if $h(x;\theta) < 0.7$

Suppose we want to predict $y = 1$
only if we are very confident.

» **Higher Precision** / **Lower Recall**



Precision:

$$\frac{\text{True Positives}}{\text{\# of Predicted Positive}}$$

Recall:

$$\frac{\text{True Positives}}{\text{\# of Actual Positive}}$$

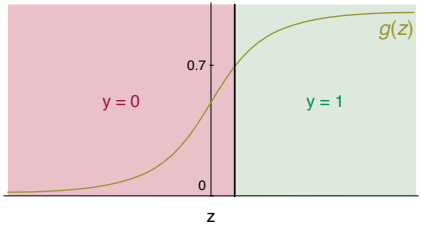
Precision and Recall Trade off

Logistic Regression: $0 \leqslant h(x;\theta) \leqslant 1$

Predict $(y = 1)$ if $h(x;\theta) \geqslant 0.7$
Predict $(y = 0)$ if $h(x;\theta) < 0.7$

Suppose we want to predict $y = 1$
only if we are very confident.

» **Higher Precision** / **Lower Recall**



Precision:

$$\frac{\text{True Positives}}{\text{\# of Predicted Positive}}$$

Recall:

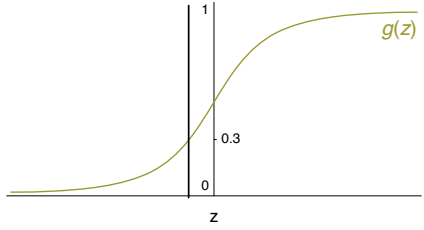
$$\frac{\text{True Positives}}{\text{\# of Actual Positive}}$$

Precision and Recall Trade off

Logistic Regression: $0 \leqslant h(x;\theta) \leqslant 1$

Predict $(y = 1)$ if $h(x;\theta) \geqslant 0.3$
Predict $(y = 0)$ if $h(x;\theta) < 0.3$

Suppose we want to predict $y = 1$
but **avoid missing** too many cases .



Precision:

$$\frac{\text{True Positives}}{\text{\# of Predicted Positive}}$$

Recall:

$$\frac{\text{True Positives}}{\text{\# of Actual Positive}}$$

Precision and Recall Trade off

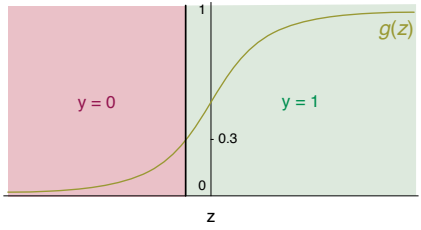
Logistic Regression: $0 \leqslant h(x;\theta) \leqslant 1$

Predict $(y = 1)$ if $h(x;\theta) \geqslant 0.3$
Predict $(y = 0)$ if $h(x;\theta) < 0.3$

Suppose we want to predict $y = 1$
but **avoid missing** too many cases .

» **Higher Recall** / **Lower Precision**

When in doubt, predict $(y = 1)$



Precision:

$$\frac{\text{True Positives}}{\text{\# of Predicted Positive}}$$

Recall:

$$\frac{\text{True Positives}}{\text{\# of Actual Positive}}$$

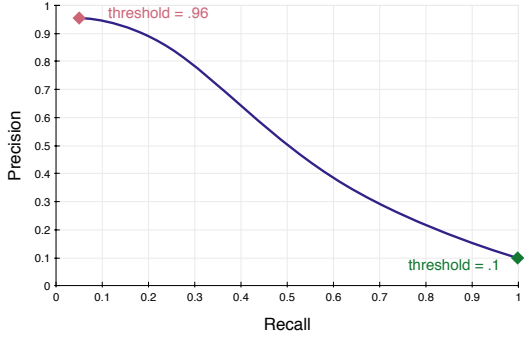
Precision and Recall Trade off

Logistic Regression: $0 \leqslant h(x;\theta) \leqslant 1$

Predict $(y = 1)$ if $h(x;\theta) \geqslant 0.5$
Predict $(y = 0)$ if $h(x;\theta) < 0.5$

Upshot:

Predict 1 if $h(x;\theta) \geqslant$ **threshold**.



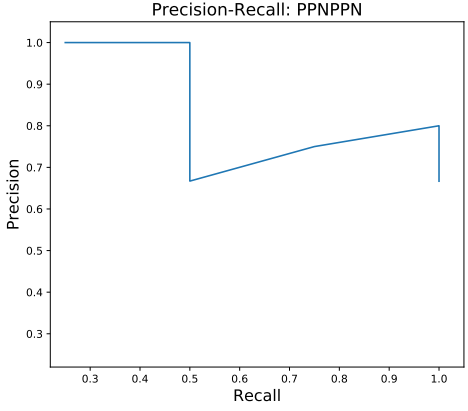
Precision:

$$\frac{\text{True Positives}}{\text{\# of Predicted Positive}}$$

Recall:

$$\frac{\text{True Positives}}{\text{\# of Actual Positive}}$$

Precision and Recall Trade-off



Suppose I have labeled data:

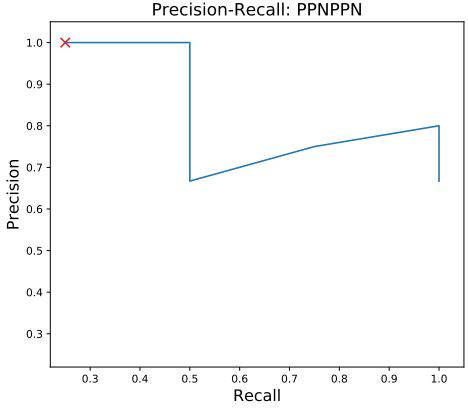
P: positive class (y = 1)

N: negative class (y = 0)

Ordered by $p(y = 1|x; \theta)$, max to min:

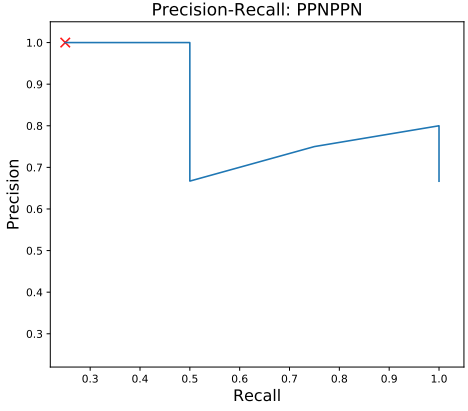
PPNPPN

Precision and Recall Trade-off



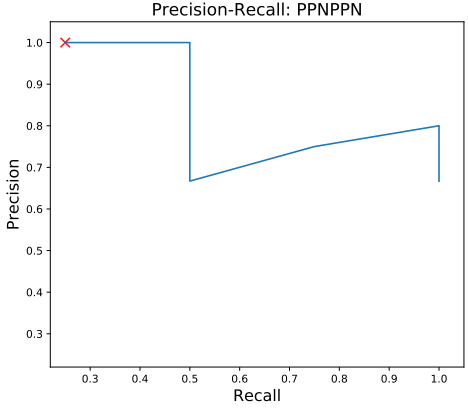
$$\underbrace{\text{predict } y{=}1}_{\text{P}} \quad \wedge \quad \underbrace{\text{predict } y{=}0}_{\text{PNPPN}}$$

Precision and Recall Trade-off



$P_{\wedge}$PNPPN

Precision and Recall Trade-off

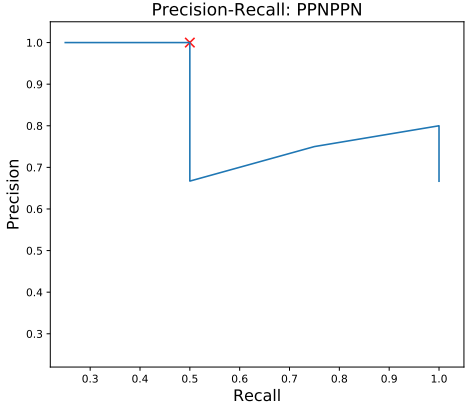


$P_{\wedge}$ PNPPN

Precision: 1

Recall: 1/4

Precision and Recall Trade-off

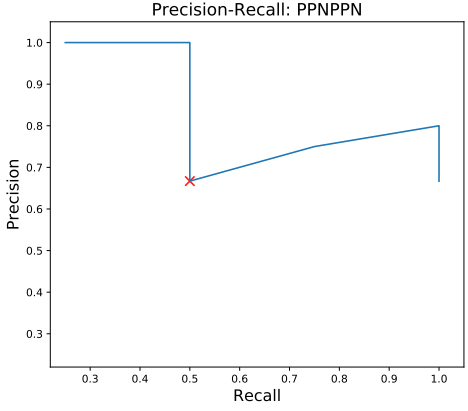


$PP_{\wedge}NPPN$

Precision: 1

Recall: ½

Precision and Recall Trade-off

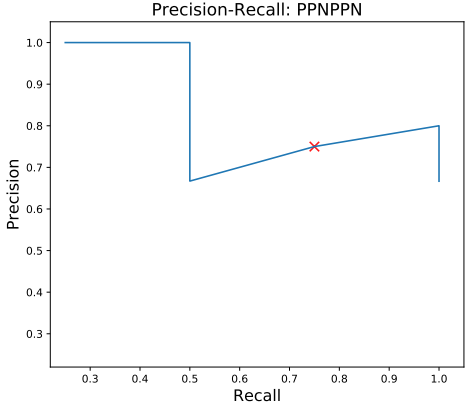


PPN ^ PPN

Precision: 2/3

Recall: 1/2

Precision and Recall Trade-off

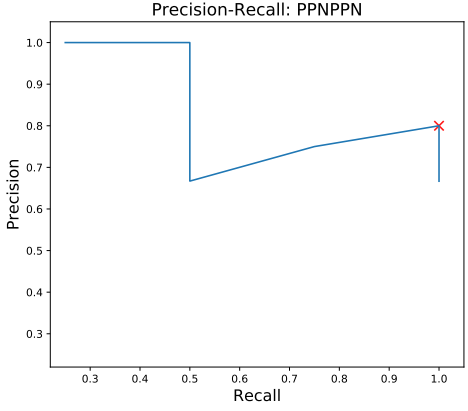


PPNP^PN

Precision: 3/4

Recall: 3/4

Precision and Recall Trade-off

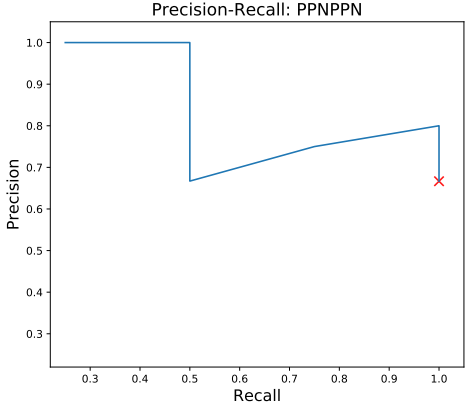


PPNPP^N

Precision: 4/5

Recall: 1

Precision and Recall Trade-off



PPNPPN^

Precision: 2/3

Recall: 1

How to choose thresholds?

Is there a way to choose threshold values automatically?

How to choose thresholds?

How ought we compare precision/recall values?
1. We might try **averaging**.

	Precision (P)	**Recall** (R)
Algorithm 1	0.5	0.4
Algorithm 2	0.7	0.1
Algorithm 3	0.02	1.0

How to choose thresholds?

How ought we compare precision/recall values?
1. We might try **averaging**.

	Precision (P)	**Recall** (R)	**Average** $\frac{P+R}{2}$
Algorithm 1	0.5	0.4	.45
Algorithm 2	0.7	0.1	.4
Algorithm 3	0.02	1.0	.51

How to choose thresholds?

How ought we compare precision/recall values?
1. We might try **averaging**.

	Precision (P)	**Recall** (R)	**Average** $\frac{P+R}{2}$
Algorithm 1	0.5	0.4	.45
Algorithm 2	0.7	0.1	.4
Algorithm 3	0.02	1.0	.51

» Algorithm 3 has highest average but predicts $y = 1$ all the time!

How to choose thresholds?

How ought we compare precision/recall values?
1. We might try **averaging**.

	Precision (P)	**Recall** (R)	**Average** $\frac{P+R}{2}$
Algorithm 1	0.5	0.4	.45
Algorithm 2	0.7	0.1	.4
Algorithm 3	0.02	1.0	.51

» Algorithm 3 has highest average but predicts $y = 1$ all the time!

So, averaging is a **bad idea**.

F_1 Score

How ought we compare precision/recall values?
2. We might try the **harmonic mean** by computing the **F_1 Score**.

	Precision (P)	**Recall** (R)	**F_1 Score**
Algorithm 1	0.5	0.4	
Algorithm 2	0.7	0.1	
Algorithm 3	0.02	1.0	

F_1 Score

How ought we compare precision/recall values?
2. We might try the **harmonic mean** by computing the **F_1 Score**.

	Precision (P)	**Recall** (R)	**F_1 Score** $2\frac{PR}{P+R}$
Algorithm 1	0.5	0.4	.444
Algorithm 2	0.7	0.1	.175
Algorithm 3	0.02	1.0	.0392

F_1 Score

How ought we compare precision/recall values?
2. We might try the **harmonic mean** by computing the **F_1 Score**.

	Precision (P)	**Recall** (R)	**F_1 Score** $2\frac{PR}{P+R}$
Algorithm 1	0.5	0.4	.444
Algorithm 2	0.7	0.1	.175
Algorithm 3	0.02	1.0	.0392

» Algorithm 1 has highest F_1 score

F_1 Score

How ought we compare precision/recall values?
2. We might try the **harmonic mean** by computing the **F_1 Score**.

	Precision (P)	**Recall** (R)	**F_1 Score** $2\frac{PR}{P+R}$
Algorithm 1	0.5	0.4	.444
Algorithm 2	0.7	0.1	.175
Algorithm 3	0.02	1.0	.0392

» Algorithm 1 has highest F_1 score

If $P = 0$ or $R = 0$, then F_1 score = 0.
If $P = 1$ and $R = 1$, then F_1 score = 1.

F_1 Score

The **harmonic mean** is often used to average *rates* or *frequencies*.

The class of *F*-scores allows differential weighting of Precision and Recall;

The F_1 assigns **equal** weight.

Limits of the F_1 Score

Although it is nice to have a single number to evaluate your threshold value, keep in mind that **Precision** and **Recall** are two different quantities.

Limits of the F_1 Score

Although it is nice to have a single number to evaluate your threshold value, keep in mind that **Precision** and **Recall** are two different quantities.

Just because they are each on the same scale (i.e., both are real numbers), it does not follow that they are necessarily comparable quantities.

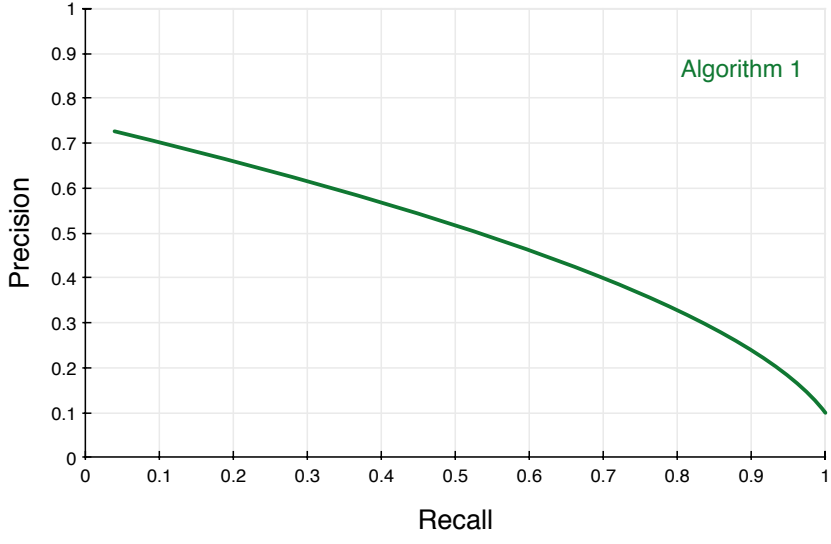
Limits of the F_1 Score

Although it is nice to have a single number to evaluate your threshold value, keep in mind that **Precision** and **Recall** are two different quantities.

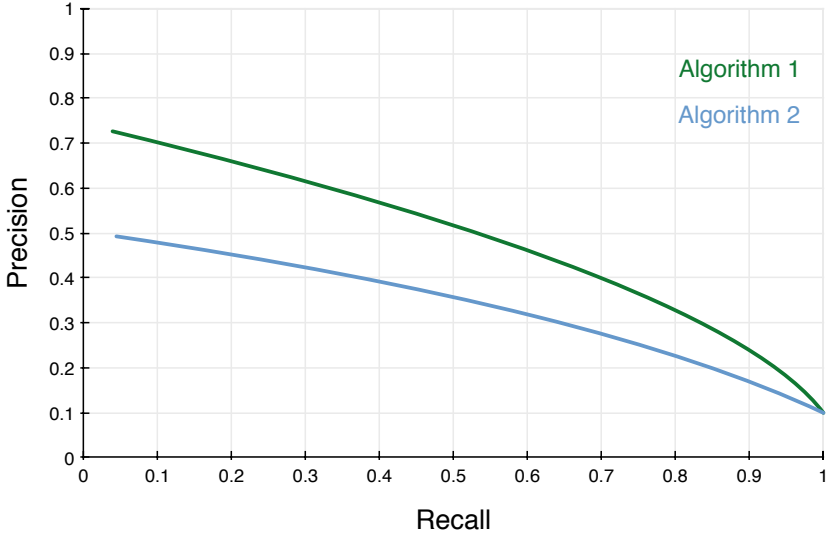
Just because they are each on the same scale (i.e., both are real numbers), it does not follow that they are necessarily comparable quantities.

Musical pitches and frequencies of light are both representable by real numbers. But, it does not follow that "***Royal Blue is closer to Middle C than Canary Yellow is***" is a meaningful comparison.

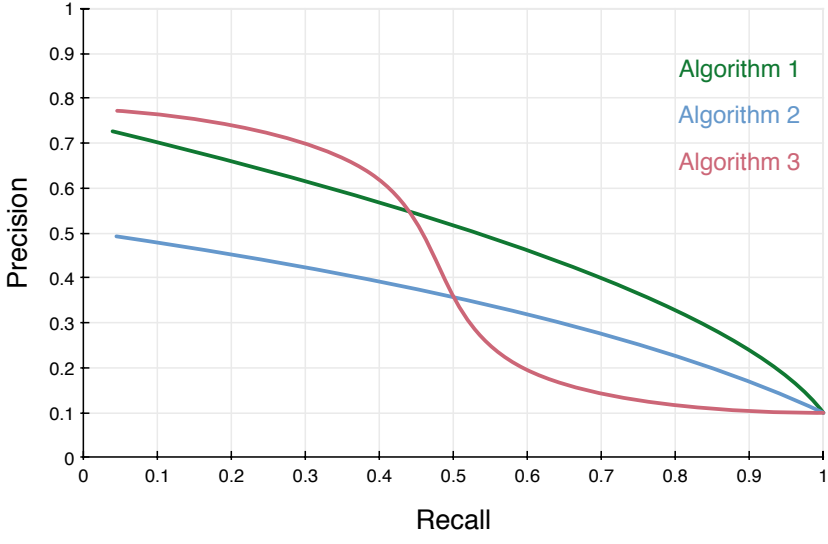
Limits of the F_1 Score



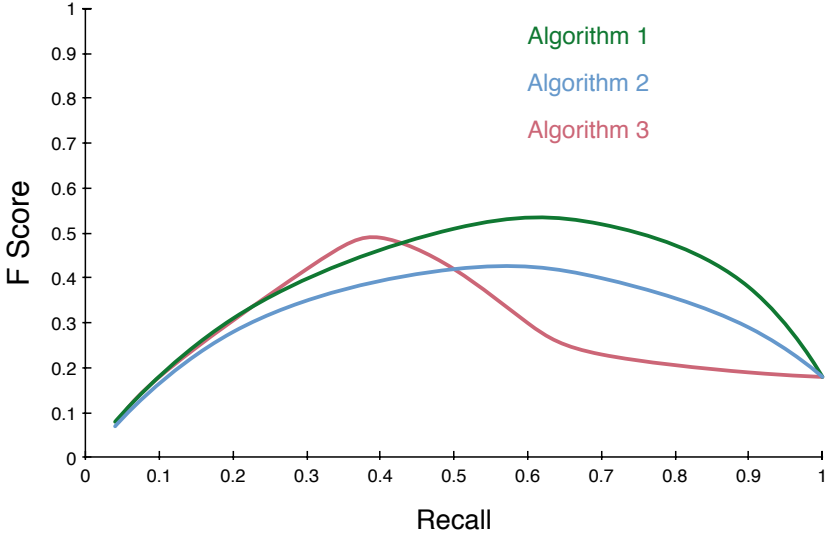
Limits of the F_1 Score



Limits of the F_1 Score



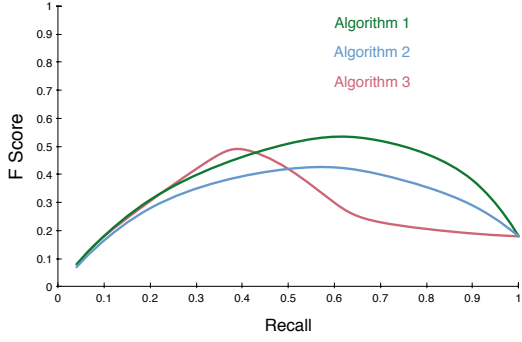
Limits of the F_1 Score



Limits of the F_1 Score

```
» 2 * (r .* p) ./ (r+p)
```

Computes a vector of F-scores given vectors of recall values (`r`) and vectors of precision values (`p`).



Limits of the F_1 Score

	Average F_1 Score	Max F_1 Score	
Algorithm 1	0.33	0.54	at 0.61 R
Algorithm 2	0.27	0.43	at 0.58 R
Algorithm 3	0.29	0.49	at 0.38 R

» According to either metric, Algorithm 3 is second best!

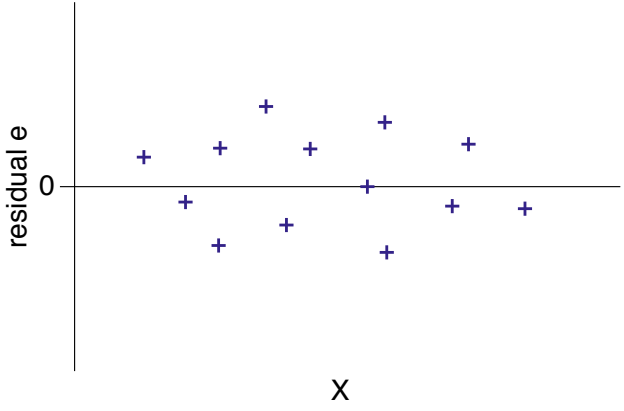
Residuals

Residuals

A **residual** e is the difference between the observed value of the dependent variable y and the predicted value $h(x; \boldsymbol{\theta})$:

$$e = y - h(x; \boldsymbol{\theta})$$

Residual Plots



$$y = \underbrace{\theta^T x}_{\text{structural component}} + \underbrace{\epsilon}_{\text{error component}}$$

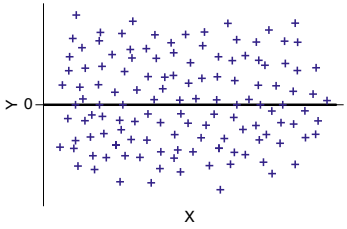
Some Properties of Residuals

Homoscedasticity: A vector of random variables is **homoscedastic** if all random variables in the sequence have the **same** variance.

Some Properties of Residuals

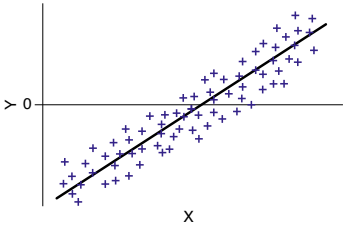
Homoscedasticity: A vector of random variables is **homoscedastic** if all random variables in the sequence have the **same** variance.

Heteroscedasticity: A vector of random variables is **heteroscedastic** if **not** all random variables in the sequence have the same variance.



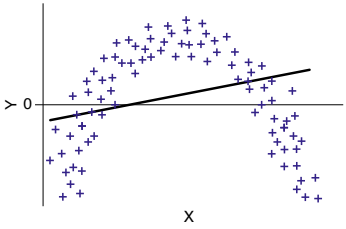
Homoscedastic &

unbiased



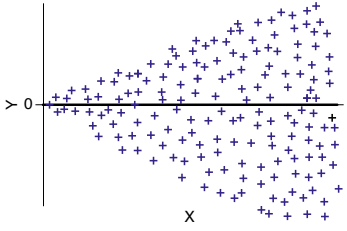
Homoscedastic &

unbiased



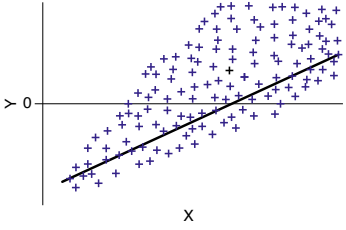
Homoscedastic &

biased



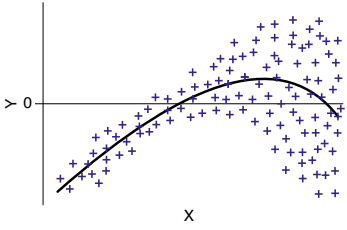
Heteroscedastic &

unbiased



Heteroscedastic

biased



Heteroscedastic

unbiased

Limits of Residual Plots

In higher-dimension models (i.e, with >5 variables), interactions between variables can go undetected by residual plots.

Two Cultures in Statistical Modeling

Two goals of data analysis

1. **Prediction**: To predict responses *y* from future input *x*
2. **Information**: To extract information about how **Nature** associates input *x* and response *y*.

Two goals of data analysis

1. **Prediction**: To predict responses *y* from future input *x*
2. **Information**: To extract information about how **Nature** associates input *x* and response *y*.

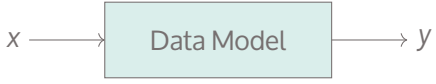
There are two different approaches:

Data Modeling Culture

Algorithmic Modeling Culture

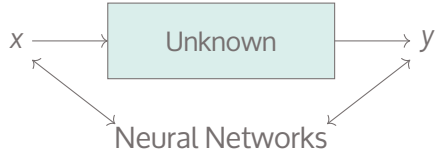
Breiman's Two Cultures

Data Modeling Culture



Data Model: Examples: *linear regression*, *logistic regression*

Algorithmic Modeling Culture



The inside of the box is complicated and unknown. The goal is to find some function $h(x; \boldsymbol{\theta})$ that operates on x to predict responses y.

Goal: Predictive accuracy.

Statistics	Machine Learning
response variables (y) = *f*(predictor variables (x), parameters (β) random noise (ϵ)	hypothesis (h) = *f*(features (x) parameters/weights (θ/ω) random noise (ϵ)

A Comparison

Statistics

Quantiative

Define theoretical / operational terms

Design Data Model

Select Variables and Measurement Scale

Parameter Selection

Model Selection

Check Model Fit

Refine

Machine Learning

Quantiative

Wrangle data (most of the effort)

Quick and dirty **implementation**.

Plot **learning curves** to guide your next step

Perform **error analysis** to search for systematic errors that could become a feature.

A Comparison

Statistics

Quantiative

Define theoretical / operational terms

Design Data Model

Select Variables and Measurement Scale

Parameter Selection

Model Selection

Check Model Fit

Refine

*What **principles** warrant these **decisions**?*

Machine Learning

Quantiative

Wrangle data (most of the effort)

Quick and dirty **implementation**.

Plot **learning curves** to guide your next step

Perform **error analysis** to search for systematic errors that could become a feature.

A Comparison

Statistics

Quantiative

Define theoretical / operational terms

Design Data Model

Select Variables and Measurement Scale

Parameter Selection

Model Selection

Check Model Fit

Refine

*What **principles** warrant these **decisions**?*

Machine Learning

Quantiative

Wrangle data (most of the effort)

Quick and dirty **implementation**.

Plot **learning curves** to guide your next step

Perform **error analysis** to search for systematic errors that could become a feature.

*Empirical performance **without** (much) theoretical justification.*

No Free Lunch Theorem

Why so many models?

All models are wrong, but some models are useful.

– George Box

Why so many models?

No Free Lunch Theorem

There is no single best model that is optimal for all possible problems (Wolpert and Macready 1997).

Why so many models?

No Free Lunch Theorem

In other words, a set of assumptions that performs well in one domain may perform poorly in another.

Why so many models?

No Free Lunch Theorem

In other words, a set of assumptions that performs well in one domain may perform poorly in another.

Hence the need to develop many **different types of models** to cover the wide variety of data that occurs in the real world.

References

Wolpert, D. H. and W. G. Macready (1997).
No free lunch theorems for optimization.
IEEE Transactions on Evolutionary Computation 1(1), 67–82.