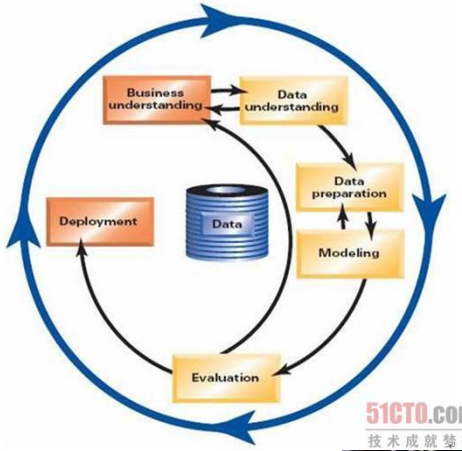
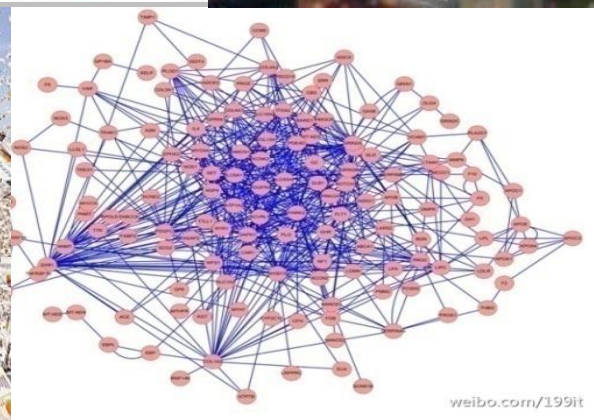




Chapter 3 Evaluation

2025 Autumn

Lei Sun

01 Why

02 Confusion matrix

03 Metrics based on CM

04 ROC curve

05 Bias and Variance

Why

- ✓ Multiple methods are available to classify or predict. For each method, multiple choices are available for settings.
- ✓ To choose the best model, need to assess each model's performance
- ✓ When there is an uneven class distribution (skewed data) in a dataset, confusion matrix(CM混淆矩阵) is especially helpful in evaluating a model's performance beyond basic accuracy metrics.
  - Tax fraud Credit default Predicting delayed flights Detecting electronic
- ✓ In above cases, we may be willing to tolerate greater over error, in return for better accuracy in identifying the important class.

Confusion matrix

- ✓ A matrix that summarizes the performance of a machine learning model on a set of test data.

Out of those cases predicted as p (or n), how many are actual p (or n) ?

Out of those cases predicted as p (or n), how many are no actually p (or n)?

Proportion of actual n cases that are falsely predicted to be p ?

Actual	Predicted	
	Yes	No
Yes	TURE+	FALSE-
No	FALSE+	TRUE-

Confusion matrix

Actual	Predicted	
	Yes	No
Yes	TURE+	FALSE-
No	FALSE+	TRUE-

TP = True Positive: Case was positive and predicted positive

TN = True Negative: Case was negative and predicted negative

FP = False Positive: Case was negative but predicted positive

FN = False Negative: case was positive and predicted negative

Confusion matrix

True

Prediction

Dog	Dog	Dog	No	Dog	No	Dog	No	Dog	Dog
×	√	√		×		√		×	√

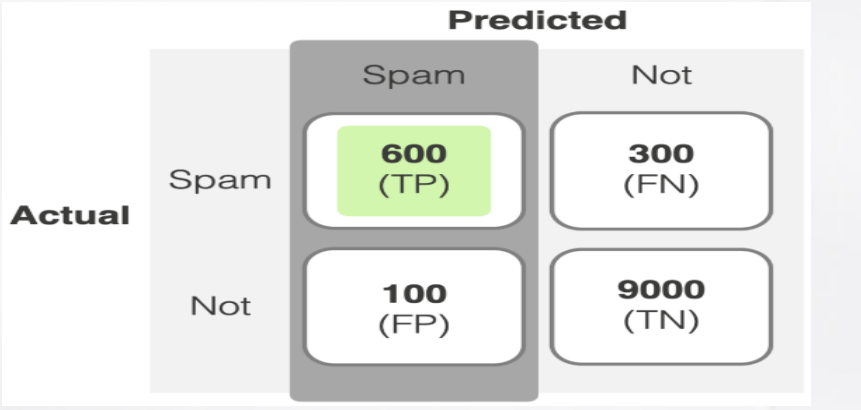
TP=4

FP=3

TN=1

FN=2

Metrics based on CM



① Accuracy（准确度）

the ratio of Total correct instances to the total instances.

accuracy =(9000+600)/10000 = 0.96.

The model was correct in 96% of cases.

$$\text{Accuracy} = \frac{\text{True predictions (TP + TN)}}{\text{All predictions (TP + TN + FP + FN)}}$$

Problem:

Misleads for imbalanced datasets when one class has significantly more samples.

- 9100 out of 10000 are regular emails, only 900 is spam（垃圾邮件）
- If model got 91% correctness for non-spam, is it our objectives?

The overall model “correctness” is heavily skewed to reflect how well the model can identify those non-spam emails.

The accuracy number is not very informative if you are interested in catching spam.

Metrics based on CM

When we see an imbalanced example like the spam example above, it is very intuitive to suggest different approaches to model evaluation that overcomes the limitation of accuracy:

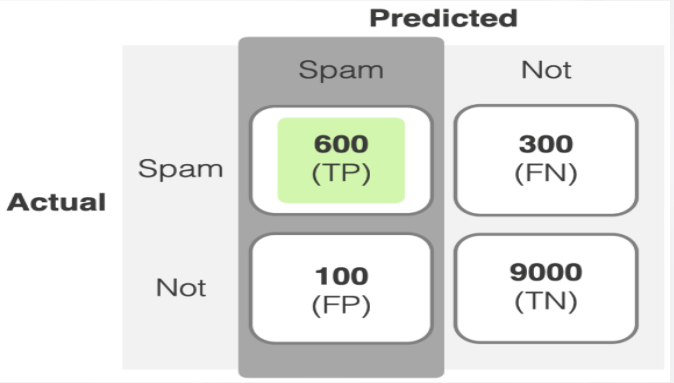
we do not need the “overall” correctness, we want to find spam emails after all!

Can we focus on how well we find and detect them specifically?

Precision and recall are the two metrics that help with that.

$$\text{Precision} = \frac{\text{Actual spam (TP)}}{\text{Predicted spam (TP + FP)}}$$

Precision= 600/(600+100)= 0.86



Out of all the items labeled as positive, how many were truly positive?”

② **Precision(精确度)** *positive predicted value* (PPV)

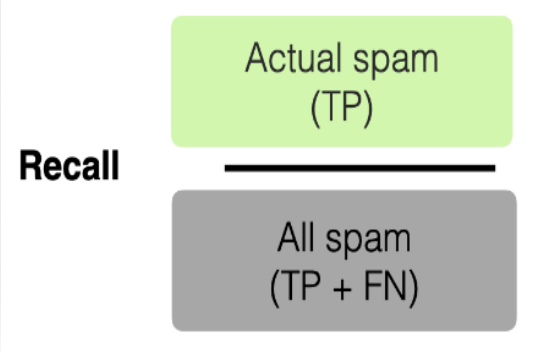
the proportion of positive class predictions that actually belong to the class

e.g. how many emails flagged as spam are classified as spam

- Precision’s denominator includes FP, emphasizing prediction accuracy

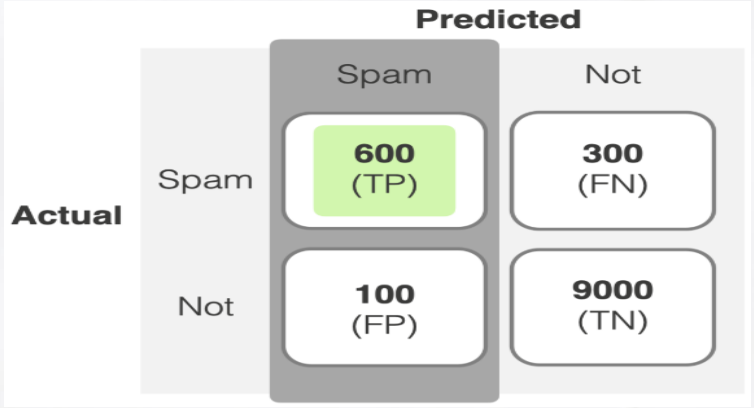
- **True Positives (TP)** is 600 referring the 600 spam emails correctly identified as spam
- **False Positives (FP)** refer to the 100 legitimate emails incorrectly marked as spam

The precision is 86% means that 86% of the emails labeled as spam are indeed spam, while the remaining 14% are not. In other words, the 14% is false positive by model.



• Recall's denominator includes FP, emphasizing **completeness** of detection

recall =600/(600+300)= 0.67



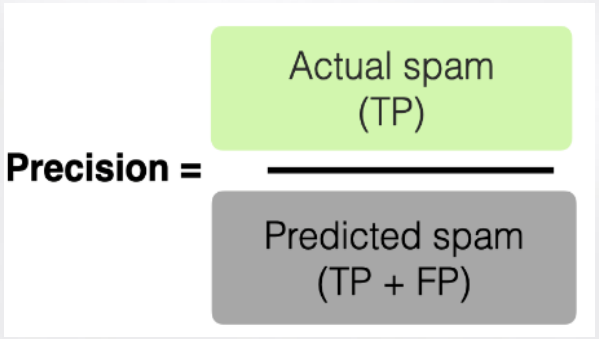
Out of all the truly positive items, how many did we correctly identify?

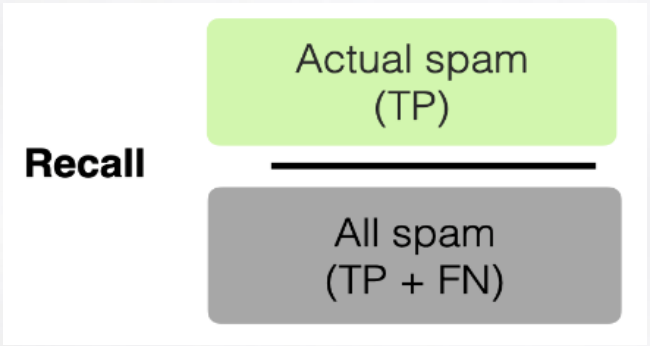
③ **Recall** （召回率）*sensitivity* 灵敏度or *true positive rate* (TPR)

a metric evaluating the ability of a model to correctly identify all of the actual positive instances

- **True Positives** is 600 referring the spam emails correctly identified as spam
- **FN** is 300 referring to the spam emails that not identified as spam

The spam filter's recall is 67%. This means the model successfully identifies 67% of the spam emails but misses 33%. In other words, these 300 emails are false negatives by the model out of all positive samples.





- **Precision**: minimize FP, emphasizing positive prediction accuracy
- **Recall:** minimize FN, emphasizing completeness of positive detection

When Precision Matters More:

- Email spam detection: You don't want legitimate emails marked as spam
- Medical diagnoses: False positives can lead to unnecessary treatments
- Financial fraud detection: False alarms can inconvenience customers

When Recall Matters More:

- Disease screening: Missing actual cases can be life-threatening
- Security threat detection: Failing to identify real threats poses risks
- Quality control: Missing defective products can damage reputation

Metrics based on CM

④ **Specificity**（特异度）*true negative rate* (TNR)

It measures the ability of a model to correctly identify negative instances.

the proportion of correct negative predictions out of actual non-instances of a given class.

$$Specificity=TN/(TN+FP)$$

$$Sensitivity=TP/(TP+FN)$$

Specificity is the complement to sensitivity, or the true negative rate, and summarizes how well the negative class was predicted.

Metrics based on CM

Numerator: labeled healthy worker （negative）

Denominator: All workers who are healthy in actuality, regardless of whether they are classed as positive or negative

How many people who are healthy did we accurately predict?

A. precision　　B. recall　　C. specificity

Metrics based on CM

Which two performance metric are similar ?

A. Precision and Recall.

B. Recall and Specificity.

C. Recall and Sensitivity.

D. Precision and Sensitivity.

Metrics based on CM

In which case, the occurrence of false negatives is undesirable ?

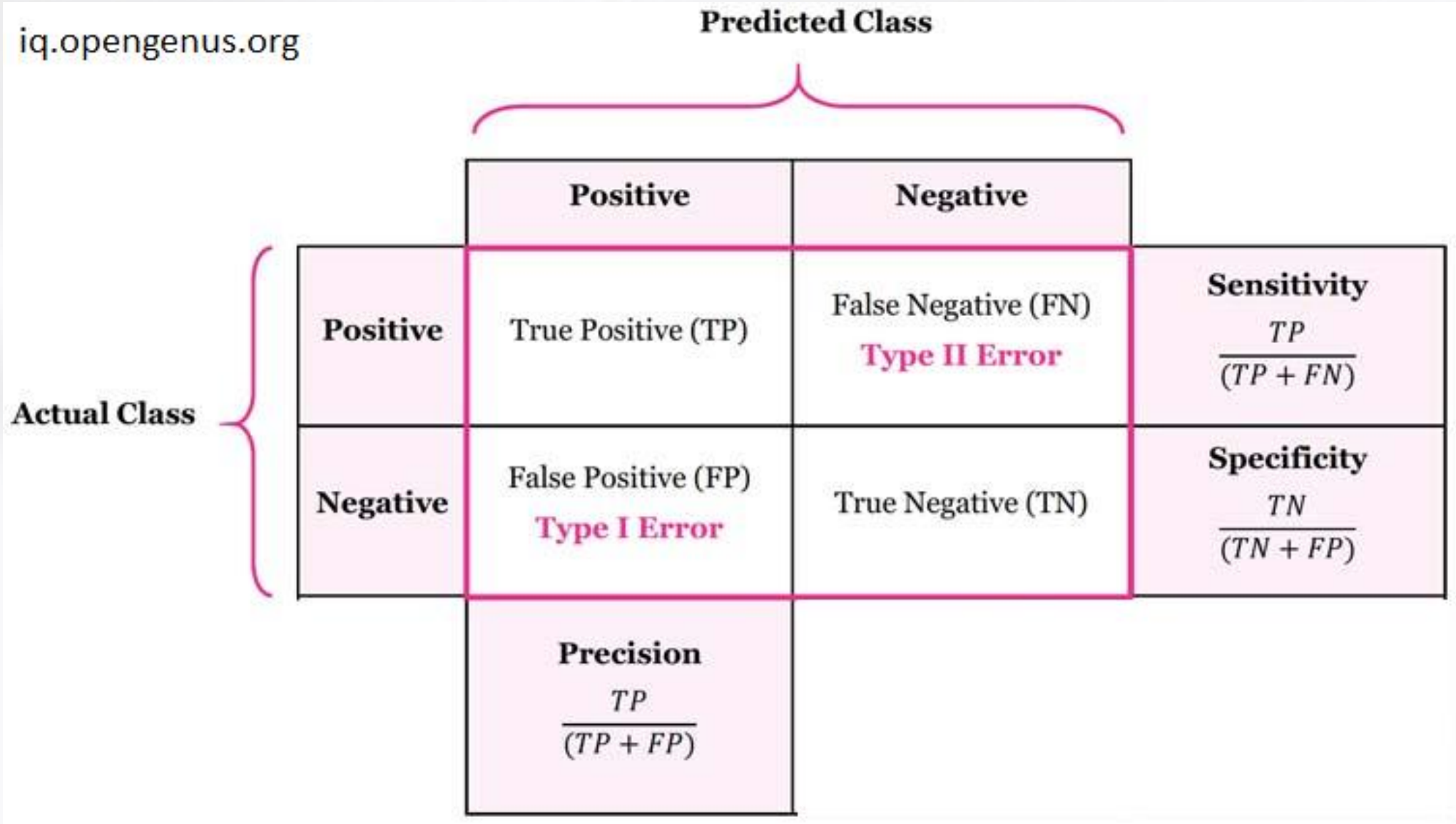
A. Precision B. Specificity C. Sensitivity

Why False Negatives are undesirable here?

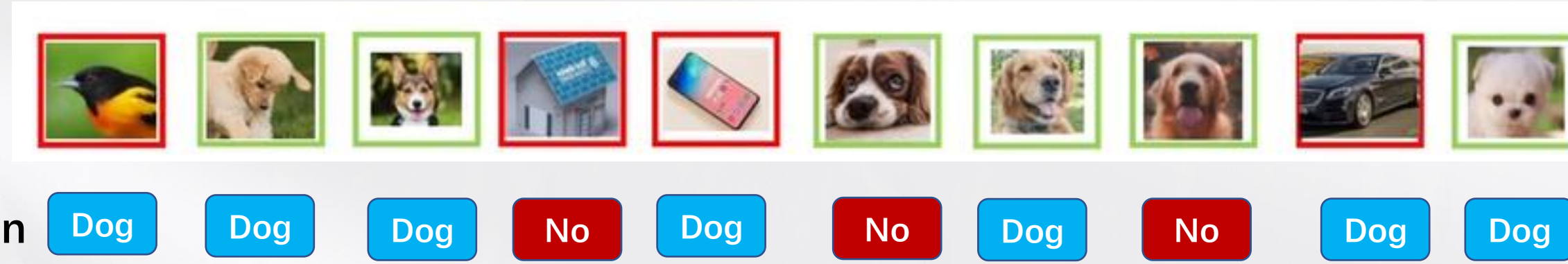
Sensitivity focuses on correctly identifying all the actual positive cases.

→ A high false negative rate means you're missing many actual positive cases, which is bad, especially in critical applications like:

iq.opengenus.org



Metrics based on CM



Precision is out of all dog predictions how many you got it right?

Precision=4/7

Recall is out of all dog truth how many you got it right? Recall=4/6

Specificity is out of all nodog truth how many you got it right?

Specificity=1/4

	公式	说明
准确率 ACC	Accuracy=（TP+TN）/ (TP+TN+FP+FN)	分类模型所有判断正确的结果占总观测的比重
精确率PPV	Precision=TP/(TP+FP)	在模型预测是Positive的所有结果中，模型预测对的比重
灵敏度 TPR	Sensitivity=Recall=TP/（TP+FN） 正类中预测正确的概率	在真实值是Positive的所有结果中，模型预测对的比重。TPR的值越大，说明预测的正类中实际的正类越多。
FPR	FPR=FP/(FP+TN) 负类错误预测为正类的概率	在真实值是Negative的所有结果中，有多少被预测成正类。FPR的值越大，说明预测的正类中实际的负类越多。
特异度 TNR	Specificity=TN/(TN+FP) 负类中预测正确概率	在真实值是Negative的所有结果中，模型预测对的比重。
FNR	FNR=FN/(TP+FN) 正类预测为负类的概率	所有正类中，有多少被预测成负类。
RPP	RPP= (TP+FP) / (N+P)	分类模型所有判断为Positive的结果占总观测的比重。

Metrics based on CM

⑤ F-score

An AI model with 95% accuracy might look impressive—until you realize it missed every single fraud transaction.

In 2025, businesses don't care about vanity metrics(虚荣指标). They care about outcomes: caught frauds, diagnosed diseases, prevented churn.

This is where F1 Score steps in—especially when you're dealing with imbalanced datasets and real-world consequences.

It balances precision (how many predicted positives are actually correct) with recall (how many actual positives you captured).

Metrics based on CM

⑤ F-score

Inverse Relationship: When one increases, the other tends to decrease. This reflects the inherent trade-off between precision and recall.

harmonic mean(调和平均) of precision and recall—combines precision and recall to represent a model's total class-wise accuracy.

$$H_n = \frac{1}{\frac{1}{n}\sum_{i=1}^{n}\frac{1}{x_i}} = \frac{n}{\sum_{i=1}^{n}\frac{1}{x_i}}$$

$$F - Sore = (1+\beta^2)\frac{Precision \times Recall}{\beta^2 Precision + Recall}$$

$$F_1 = \frac{1}{\frac{1}{2}\times\frac{1}{precision}+\frac{1}{2}\times\frac{1}{recall}} = 2\times\frac{precision \times recall}{precision + recall}$$

Metrics based on CM

⑤ **F-score**

F1 Score: the most common, which gives equal weight to precision and recall.

F-beta score (Fβ): a generalized form of the F-Score

introduces a parameter, β, to control the trade-off between precision and recall.

When β is greater than 1, recall is weighted more heavily,

while a β less than 1 gives more weight to precision.

$$\beta = 0$$

$$F_0 = (1+0)\frac{Precision * Recall}{0 * Precision + Recall} = \frac{Precision * Recall}{Recall} = Precision$$

$$F_\beta = (1+N^2)\frac{Precision * Recall}{N^2 * Precision + Recall} = \frac{Precision * Recall}{\frac{N^2}{1+N^2} Precision + \frac{1}{1+N^2} Recall}$$

$$N \to \infty, \qquad \frac{1}{1+N^2} \to 0, \qquad \frac{N^2}{1+N^2} \to 1$$

$$\lim_{N\to\infty} F_N = \frac{Precision * Recall}{1 * Precision + 0 * Recall} = Recall$$

- $\beta = 0$， Get Precision, $\beta \to 0$, F-score closes to Precision.
- $\beta = \infty$， Get Recall, β is very big, F-score closes to Recall.
- $\beta = 1$， F-score is harmonic mean of Precision and Recall. P and R are same important.

Metrics based on CM

⑤ **F-score**

Use F1 Score When:

- You want a balanced view between false positives and false negatives
- You're working with imbalanced datasets
- Neither precision nor recall alone tells the full story

⑤ F-score

The **F1 Score** always falls between **0 and 1**, but what do those numbers actually mean in real-world data mining?

F1	Precision	Recall	interpretation
1.0	1.0	1.0	All predictions are correct
0.67	1.0	0.5	Missing some positive examples
0.67	0.5	1.0	Predict all possible positive examples（误报多）
0.5	0.5	0.5	Prediction is completely random
0	0	0	Totally wrong prediction

F1 Score	Interpretation
0.0	Model failed completely—no useful prediction
0.5	Precision and recall are both modest
0.7	Acceptable for many real-world applications
0.8+	Strong model with balanced performance
1.0	Perfect precision and recall (rare in practice)

Metrics based on CM

Metrics in multi-class

① **Accuracy in multi-class**

measures the proportion of correctly classified cases from the total number of objects in the dataset.

$$\text{Accuracy} = \frac{\text{Correct predictions}}{\text{All predictions}}$$

② **Precision for a given class in multi-class**

Is the fraction of instances correctly classified as belonging to a specific class out of all instances the model predicted to belong to that class.

$$\text{Precision}_{\text{Class A}} = \frac{TP_{\text{Class A}}}{TP_{\text{Class A}} + FP_{\text{Class A}}}$$

Metrics based on CM

Metrics in multi-class

③ **Recall in multi-class**

Is the fraction of instances in a class that the model correctly classified out of all instances in that class.

$$\text{Recall}_{\text{Class A}} = \frac{TP_{\text{Class A}}}{TP_{\text{Class A}} + FN_{\text{Class A}}}$$

ROC curve- Receiver Operating Characteristic

Since to compare two different models it is often more convenient to have a single metric from the confusion matrixrather than several ones:

- **True positive rate (TPR),** a.k.a. sensitivity, hit rate, and recall, which is defined as TP/(TP+FN). It corresponds to the proportion of positive data points that are correctly considered as positive, with respect to all positive data points. In other words, the higher TPR, the fewer positive data points we will miss.

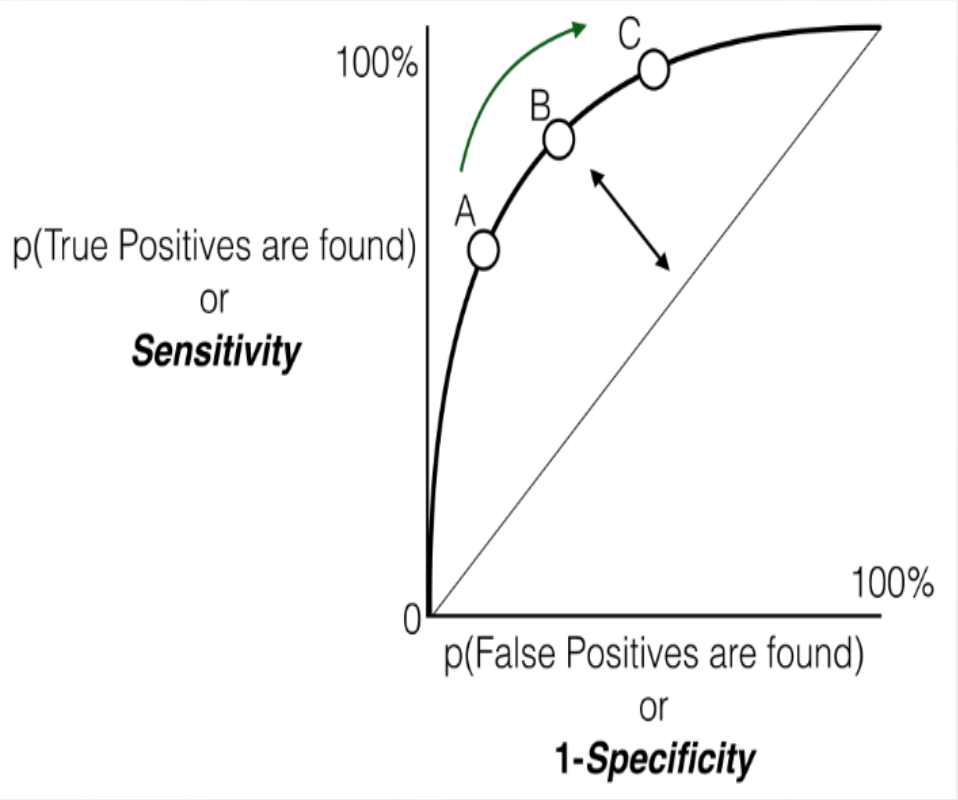
- **False positive rate (FPR)**, a.k.a. false alarm rate, fall-out or 1 - specificity, which is defined as FP/(FP+TN). Intuitively this metric corresponds to the proportion of negative data points that are mistakenly considered as positive, with respect to all negative data points. In other words, the higher FPR, the more negative data points will be missclassified.

ROC curve- Receiver Operating Characteristic

- ✓ A plot of the true positive rate (TPR) on the y-axis against the false positive rate (FPR) on the x-axis at various threshold settings.

$$FPR=FP/(TN+FP)$$

$$Specificity=TN/(TN+FP)=1-FPR$$



True positive rate (TPR, sensitivity, hit rate, recall) the proportion of positive data points that are correctly considered as positive, with respect to all positive data points. In other words, the higher TPR, the fewer positive data points we will miss.

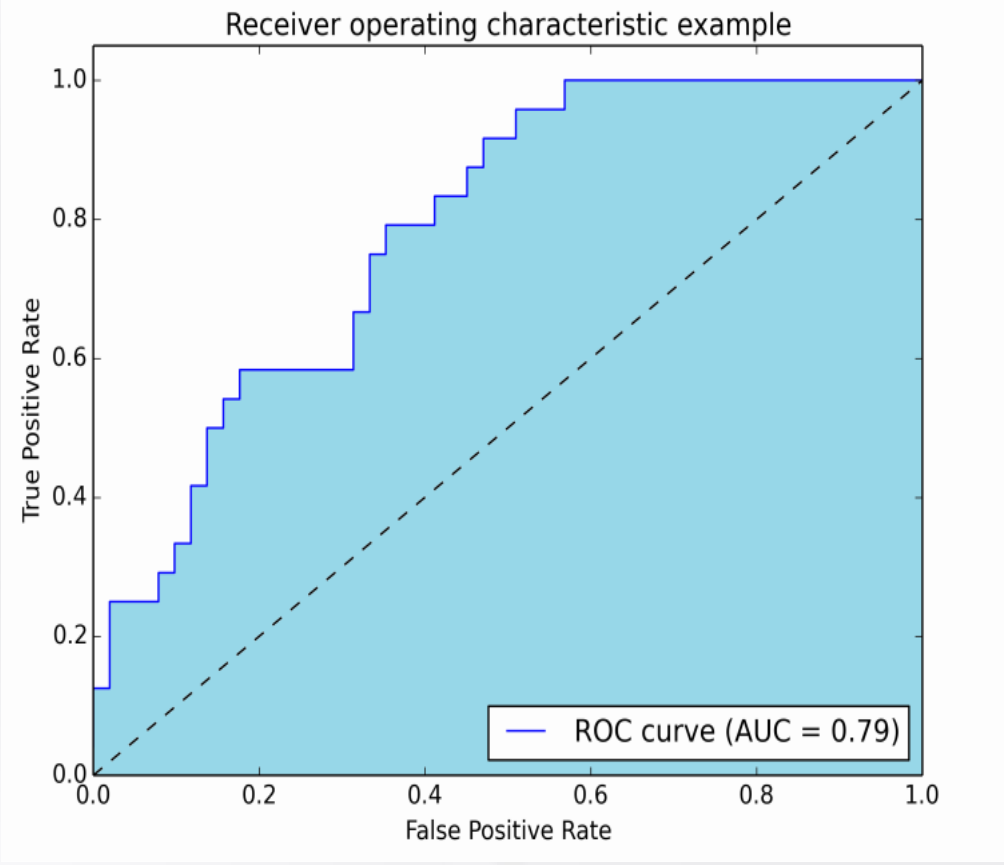
ROC curve- Receiver Operating Characteristic

To combine the FPR and the TPR into one single metric:

- we first compute the two former metrics with many different threshold (for example 0.00,0.01,0.02,⋯,1.00),
- then plot them on a single graph, with the FPR values on the x-axis and the TPR values on the y-axis.

Threshold values from 0 to 1 are decided based on the number of samples in the dataset.

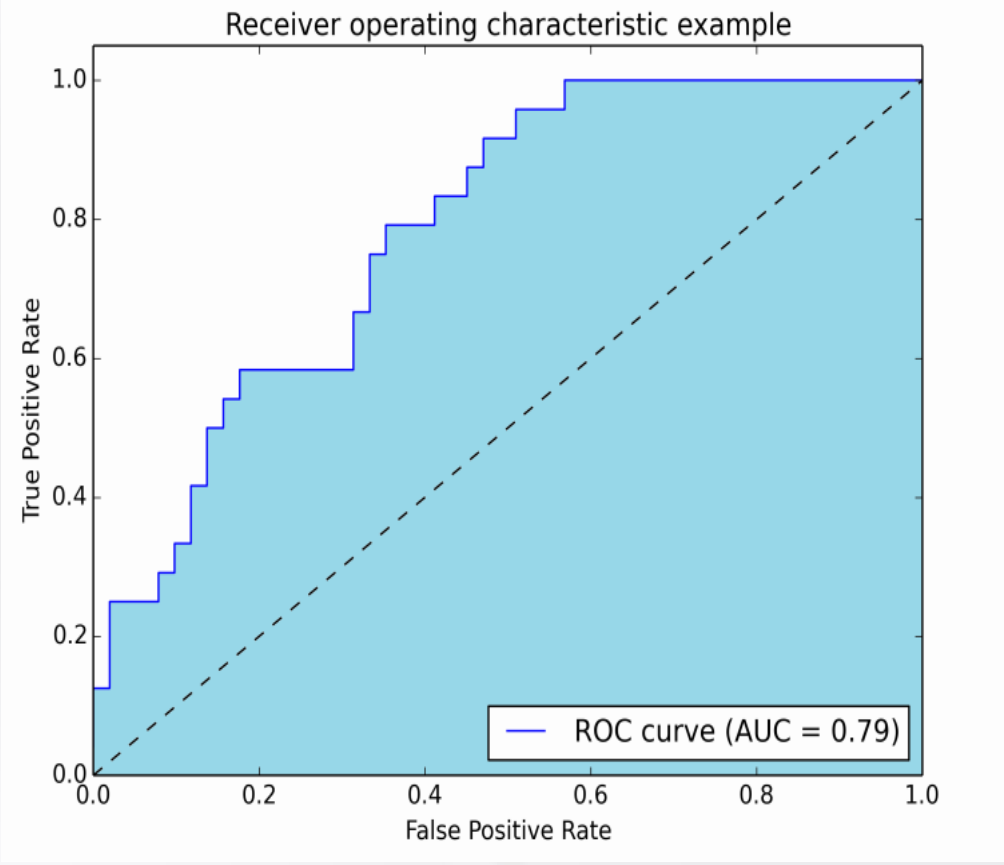
The resulting curve is called ROC curve, and the metric we consider is the AUC of this curve.



ROC curve- Receiver Operating Characteristic

In this figure, the blue area corresponds to the Area Under the curve of the ROC. The higher the area under the ROC curve, the better the classifier.

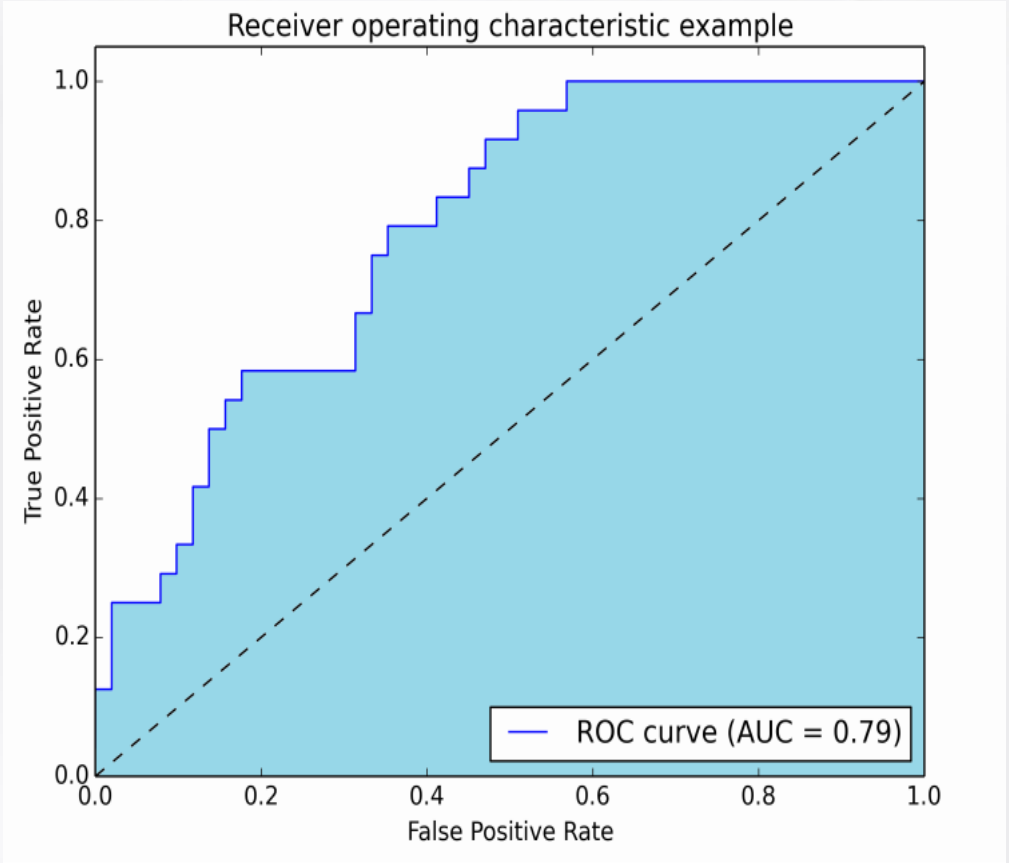
The diagonal line y=x (dashed line) represents the strategy of randomly guessing a class. For example, if a classifier randomly guesses the positive class half the time, it can be expected to get half the positives and half the negatives correct; this yields the point (0.5, 0.5) in ROC space.



The closer the curve comes to the 45-degree diagonal random line, the less accurate the test.

- If you got an AUC value of 0.9-1, it is an excellent model, A-class model.
- If it is 0.8-0.9, it is a good model, B-class model.
- If it is 0.7-0.8, it is a fair model, C-class model. 0.6-0.7 is poor model, D-class model.

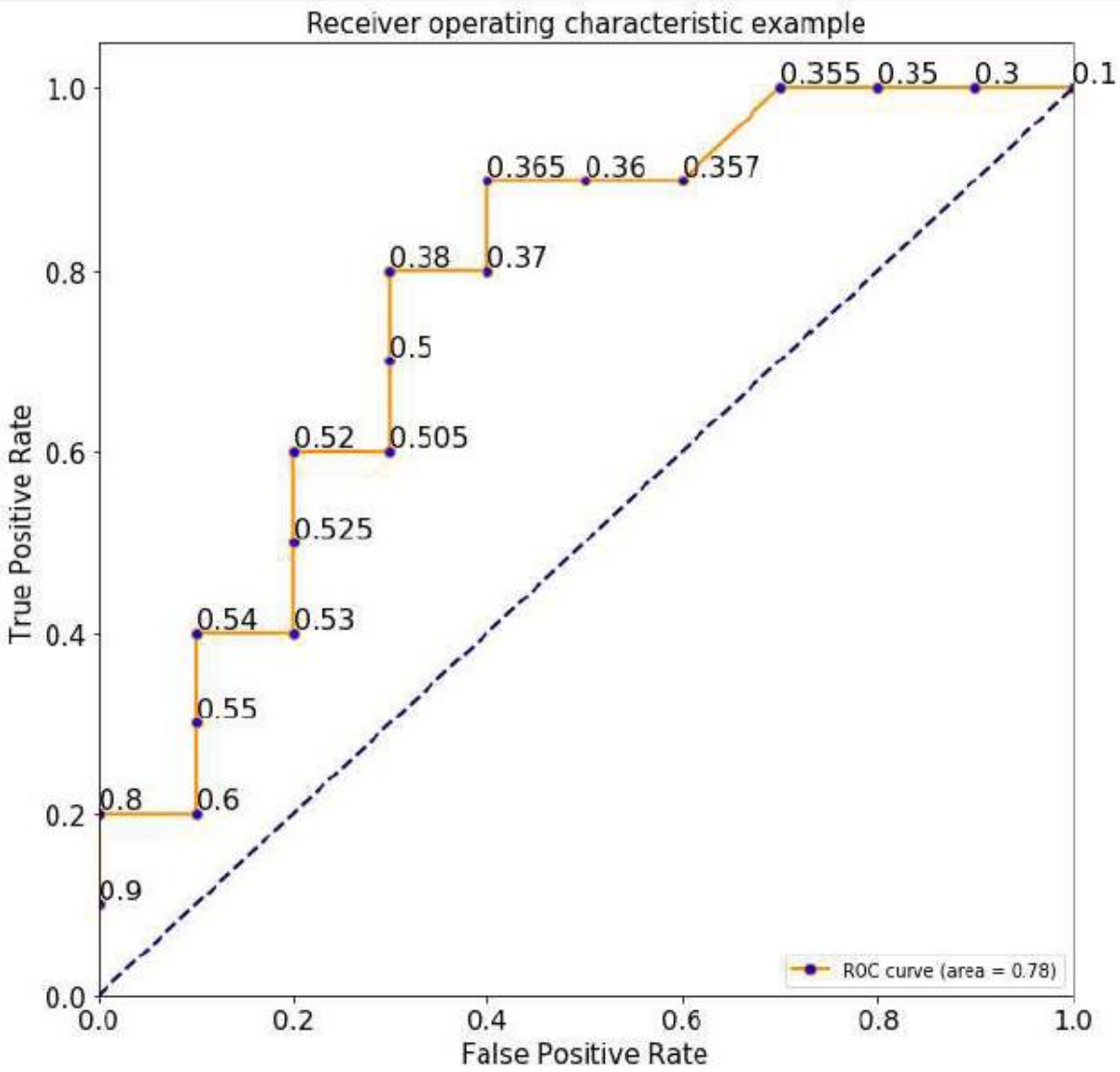
Anything less than 0.6, is a failure, an F, with no discrimination.



ID	actual	Prob.	TP	FP	TN	FN	TPR	FPR
1	T	0.9	1	0	10	9	0.1	0
2	T	0.8	2	0	10	8	0.2	0
3	F	0.6	2	1	9	8	0.2	0.1
4	T	0.55	3	1	9	7	0.3	0.1
5	T	0.54	4	1	9	6	0.4	0.1
6	F	0.53	4	2	8	6	0.4	0.2
7	T	0.525	5	2	8	5	0.5	0.2
8	T	0.52	6	2	8	4	0.6	0.2
9	F	0.505	6	3	7	4	0.6	0.3
10	T	0.5	7	3	7	3	0.7	0.3
11	T	0.38	8	3	7	2	0.8	0.3
12	F	0.37	8	4	6	2	0.8	0.4
13	T	0.365	9	4	6	1	0.9	0.4
14	F	0.36	9	5	5	1	0.9	0.5
15	F	0.357	9	6	4	1	0.9	0.6
16	F	0.355	10	7	3	0	1	0.7
17	T	0.355	10	7	3	0	1	0.7
18	F	0.35	10	8	2	0	1	0.8
19	F	0.3	10	9	1	0	1	0.9
20	F	0.1	10	10	0	0	1	1

04

ROC curve- Receiver Operating Characteristic

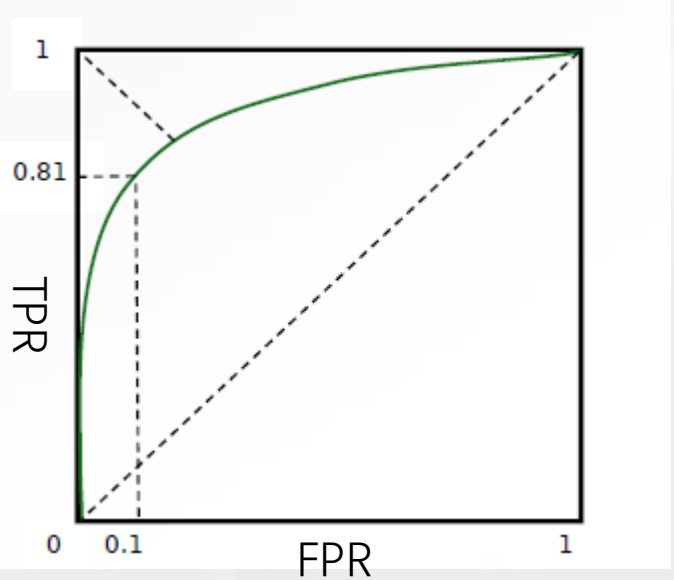


ROC curve- Receiver Operating Characteristic

FPR=FP/(TN+FP) TPR=TP/(TP+FN)

✓ Four points and one line of ROC

- (0,1): FPR=0, TPR=1. Correctly classify all the samples.
- (1,0): FPR=1,TPR=0. The classifier avoids all correct answers
- (0,0): FPR=TPR=0. The model classify all the samples as negative.
- (1,1): FPR=TPR=1. The model classify all the samples as positive.
- y=x: The model classifies half of the samples as positive and half as negative

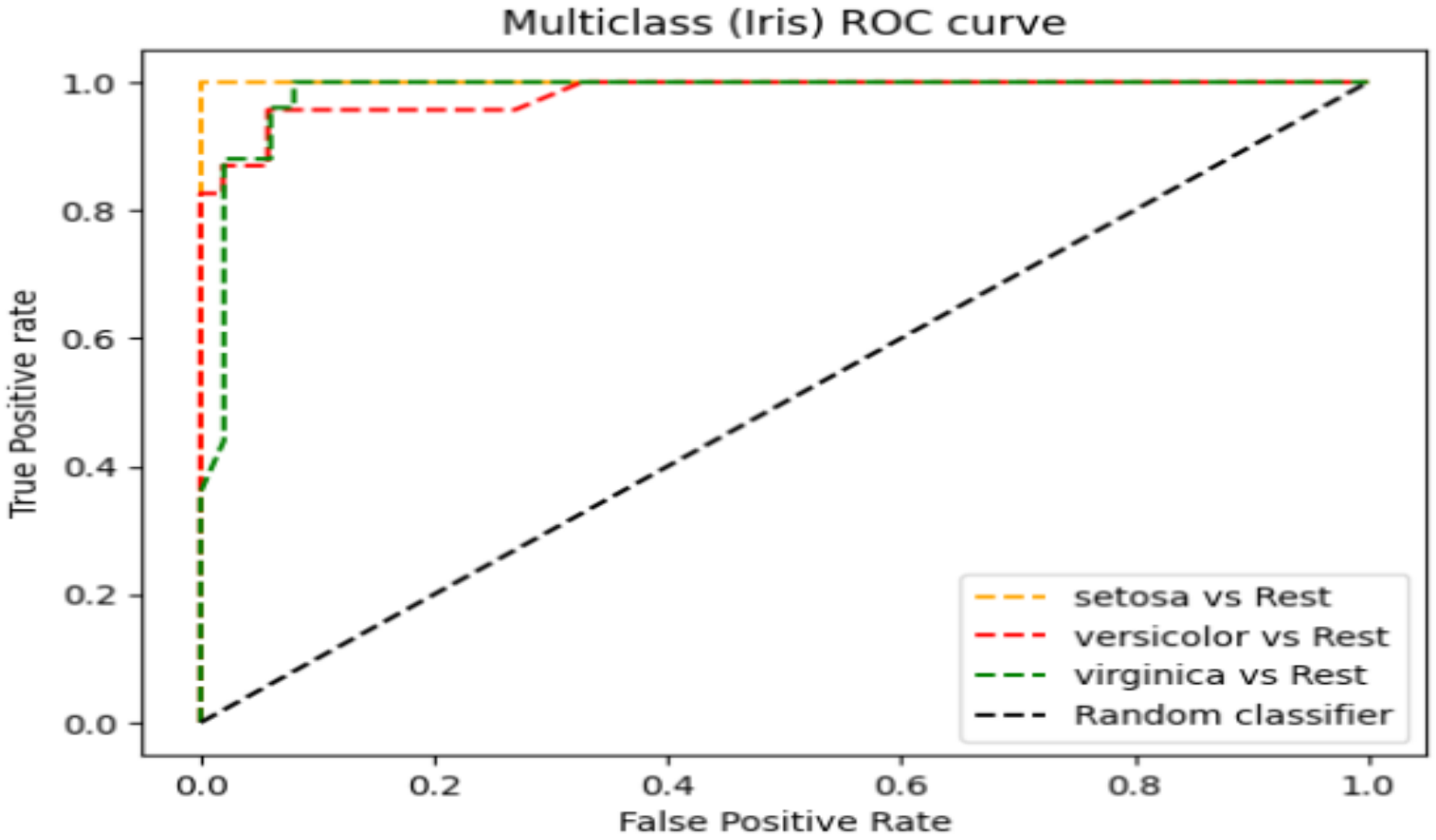


The closer the ROC curve is to the top-left corner, the better the classifier performs.

04

ROC curve- Receiver Operating Characteristic

ROC curve for Multi-Class Classifications



ROC curve for OneVSRest Multiclass Classifications

PR curve (precision-recall)

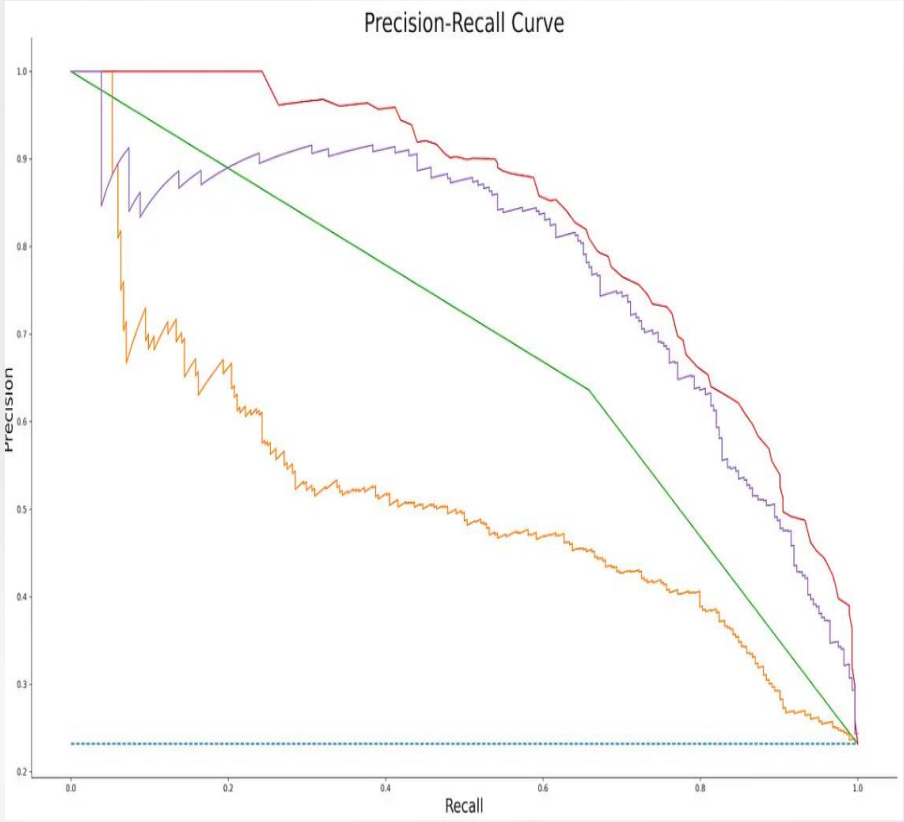
- Precision = TP / (TP + FP)
- Recall = TP / (TP + FN)

✓ Both the precision and the recall are focused on the positive class and are unconcerned with the true negatives.

✓ A plot of the precision (y-axis) against the recall (x-axis), for different classification thresholds.

✓ As the recall increases, the precision may show a downward trend.

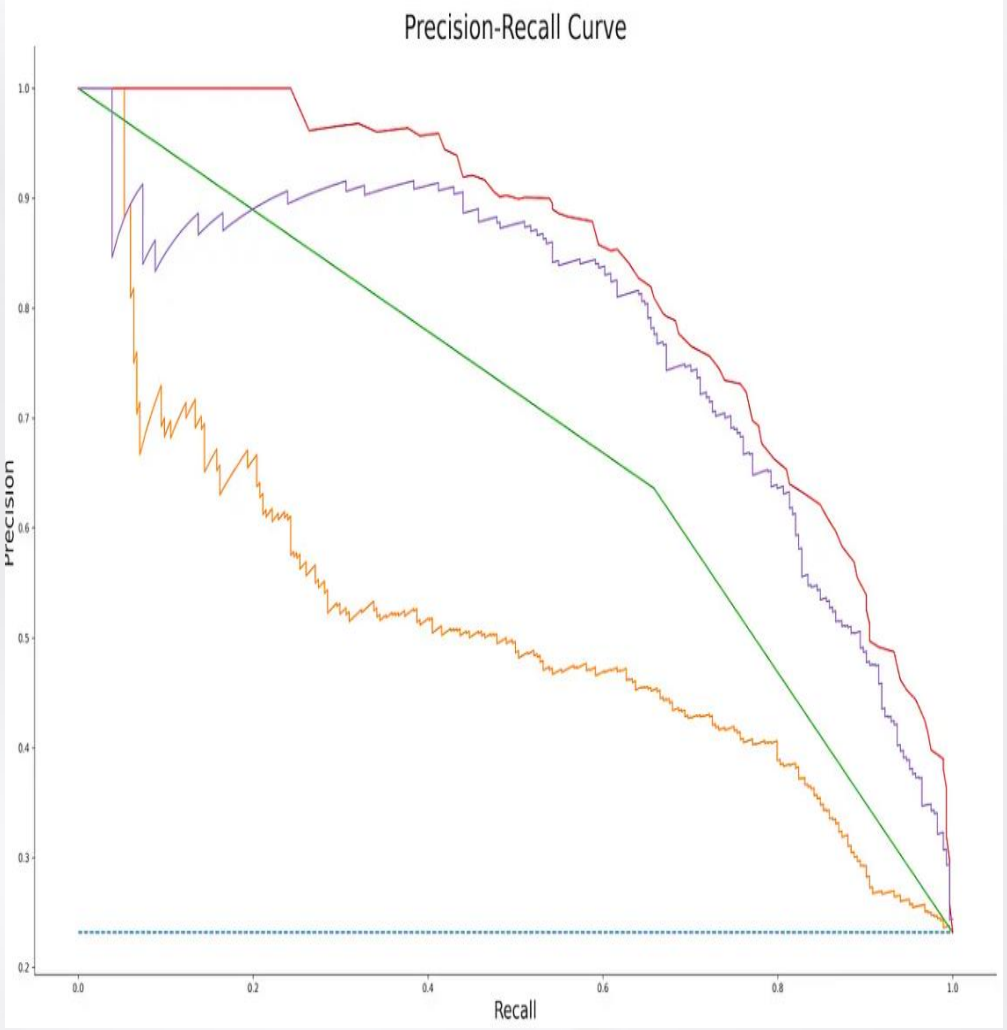
✓ A good PR curve has greater AUC (area under the curve).



PR curve (precision-recall)

To summarize, we should visualize precision-recall curves any time you want to visualize the **tradeoff between false positives and false negatives.** A high number of false positives leads to low precision, and a high number of false negatives leads to low recall.

We should aim for high-precision and high-recall models, but in reality, one metric is more important, so you can always optimize for it.



Bias vs Variance

estimate a model $\hat{f}(x)$ of $f(X)$ by ML algorithms

Consider $Y = f(x) + \epsilon$ where $\epsilon \sim N(0, \sigma_e)$.

Components of prediction error for model $\hat{f}(x)$:

$$\text{Error}(x) = E\left(Y - \hat{f}(x)\right)^2$$

$$= \underbrace{\left(E\left(\hat{f}(x)\right) - f(x)\right)^2}_{\text{bias}^2} + \underbrace{E\left(\hat{f}(x) - E\left(\hat{f}(x)\right)\right)^2}_{\text{variance}} + \underbrace{\sigma_e^2}_{\text{error}}$$

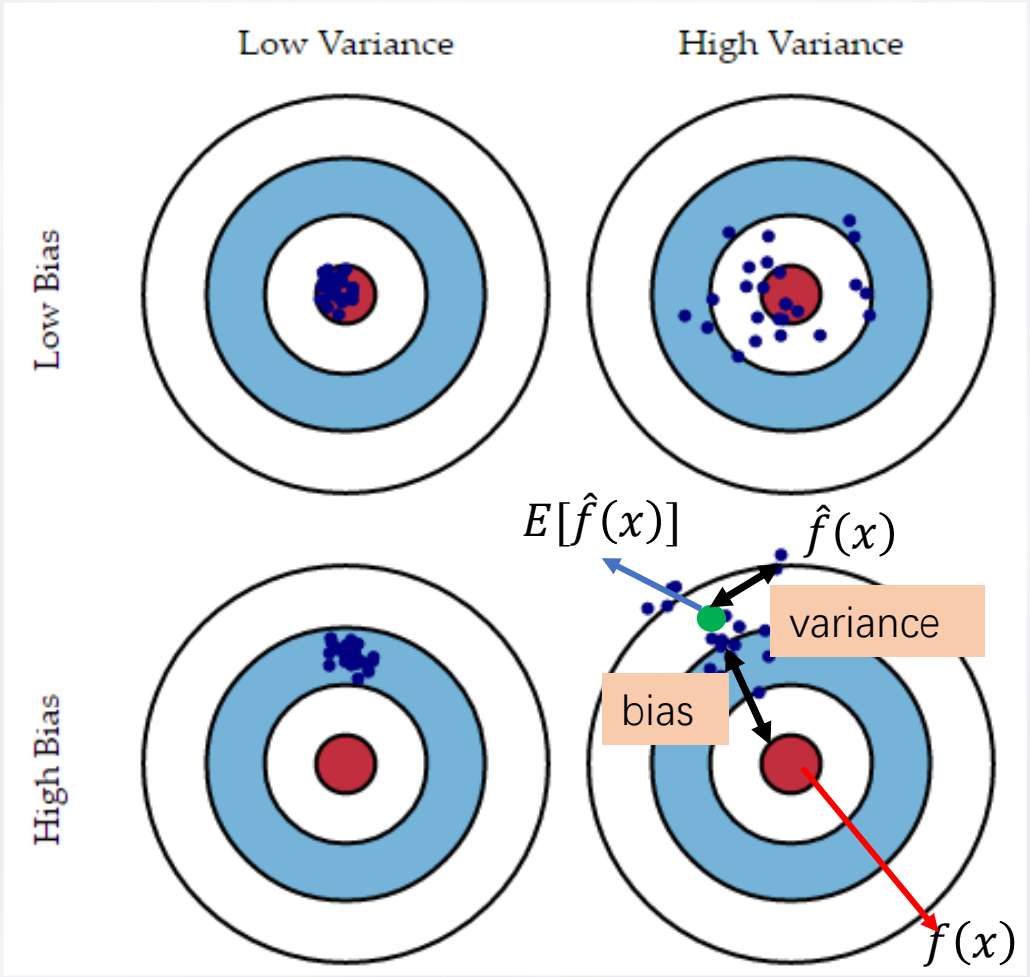
the expected squared prediction error at a point x is:

irreducible error, is the noise term in the true relationship that cannot fundamentally be reduced by any model.

Bias vs Variance

the center is a model that perfectly predicts the correct values. As we move away from the bulls-eye(靶心), our predictions get worse and worse.

We can repeat our entire model building process to get a number of separate hits on the target. Sometimes we will get a good distribution of training data so we predict very well (close to the bulls-eye), while sometimes poorer predictions. These different realizations result in a scatter of hits on the target.



Bias-variance tradeoff: Lower variance models have high bias, and visa versa

Bias vs Variance

There are two main types of errors present in any machine learning model:

Reducible

errors are those errors whose values can be further reduced to improve a model.

Bias and Variance

Irreducible

errors which will always be present in a machine learning model, because of unknown variables, and whose values cannot be reduced. E.g. random noise

Bias vs Variance

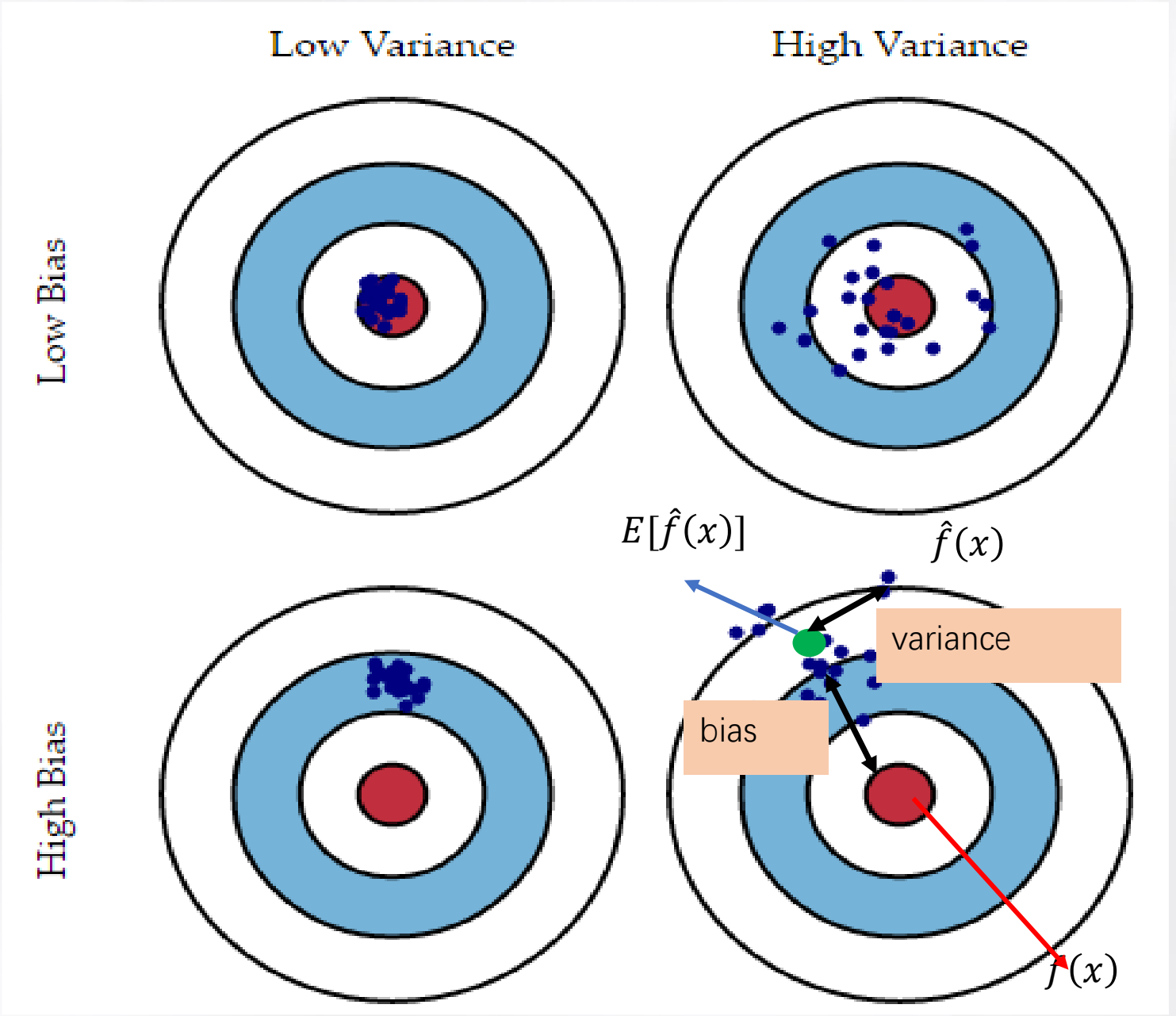
Bias is simply defined as the inability of the model because of that there is some difference or error occurring between the model's predicted value and the actual(target) value. These differences between actual or expected values and the predicted values are known as error or bias error or error due to bias.

$$Bias(\hat{f}(x)) = E[\hat{f}(x)] - f(x) \longrightarrow \text{Fitting ability}$$

where $E[\hat{f}(x)]$is the expected value of the estimator $f(x)$.
It is the measurement of the model that how well it fits the data.

Bias-variance tradeoff:

Lower variance models have high bias, and visa versa



05

Bias vs Variance

✓ **Low Bias**

Low bias value means fewer assumptions are taken to build the target function. In this case, the model will closely match the training dataset.

✓ **High Bias**

High bias value means more assumptions are taken to build the target function. In this case, the model will not match the training dataset closely.

Bias vs Variance

Ways to reduce high bias in Machine Learning

- **Use a more complex model:** One of the main reasons for high bias is the very simplified model. it will not be able to capture the complexity of the data.
- **Increase the number of features:** By adding more features to train the dataset will increase the complexity of the model. And improve its ability to capture the underlying patterns in the data.
- **Increase the size of the training data:** Not good
- **Reduce Regularization:** reduce the strength of regularization or removing it altogether can help to improve its performance.

Bias vs Variance

Variance: The amount by which the performance of a predictive model changes when it is trained on different subsets of the training data. More specifically, variance is the variability of the model that how much it is sensitive to another subset of the training dataset. i.e. how much it can adjust on the new subset of the training dataset.

$$Variance = E\left[\left(\hat{f}(x) - E\left(\hat{f}(x)\right)^2\right]\right. \longrightarrow \text{stability}$$

Let $f(x)$ be the actual values of the target variable, and $\hat{f}(x)$ be the predicted values of the target variable.

$E\left(\hat{f}(x)\right)$ is the expected value of the predicted values. Expected value is averaged over all the training data.

05

Bias vs Variance

✓ Low variance

The model is less sensitive to changes in the training data and can produce consistent estimates of the target function with different subsets of data from the same distribution.

✓ High variance:

The model is very sensitive to changes in the training data and can result in significant changes in the estimate of the target function when trained on different subsets of data from the same distribution. This is the case of overfitting when the model performs well on the training data but poorly on new, unseen test data.

Bias vs Variance

Ways to Reduce the Variance

- **Cross-validation:** By splitting the data into training and testing sets multiple times, cross-validation can help identify if a model is overfitting or underfitting.
- **Feature selection:** By choosing the only relevant feature will decrease the model's complexity.
- **Regularization:** We can use L1 or L2 regularization to reduce variance.
- **Ensemble methods:** It will combine multiple models to improve generalization performance. RandomForest and boosting are common ensemble methods that can help reduce variance and improve generalization performance.

Bias vs Variance

Ways to Reduce the Variance

- **Simplifying the model:** Reducing the complexity of the model, such as decreasing the number of parameters or layers in a neural network, can also help reduce variance and improve generalization performance.
- **Early stopping:** Early stopping is a technique used to prevent overfitting by stopping the training of the deep learning model when the performance on the validation set stops improving.
- **Increase the size of the training data:**

05

Bias vs Variance

Questions

① If your model cannot fit your training data, then you have large bias.

A. underfitting　　B. overfitting

② If your model can fit your training data, but large error on your testing data, then you probably have large variance.

A. underfitting　　B. overfitting

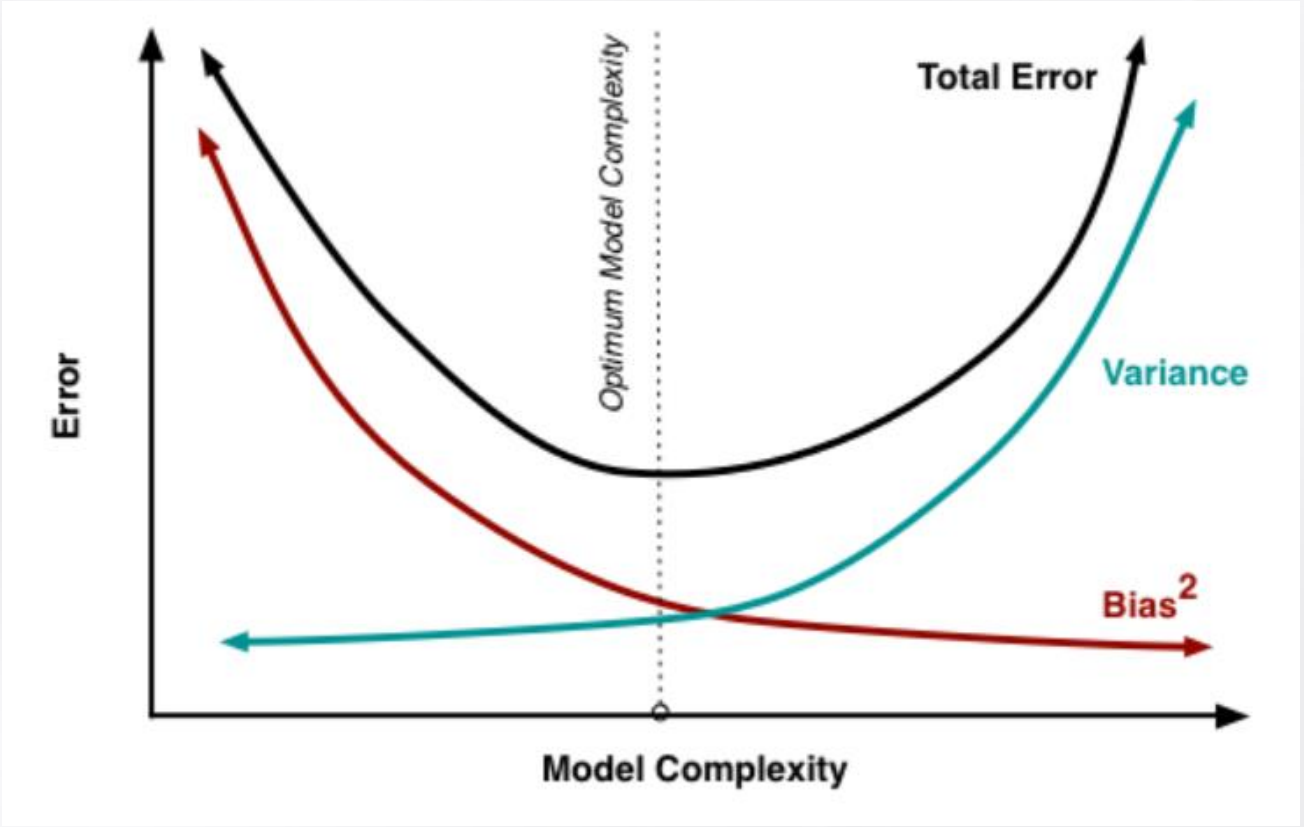
③ Is adding more data useful for dealing with large biars?

A. Yes　　B. No

Bias vs Variance

At its root, dealing with bias and variance is really about dealing with over- and under-fitting. Bias is reduced and variance is increased in relation to model complexity.

As more and more parameters are added to a model, the complexity of the model rises and variance becomes our primary concern while bias steadily falls.



Bias and variance contributing to total error.