

Advanced Classification

Data Mining

Content

- Multi-class Classification
- Performance Evaluation
- Class Imbalance Problem
- Ways to Improve Classification Accuracy
- Data Leakage Problem
- Practical Application Issues

Multi-Class Classification

Multi-class Problem

- Some of classification algorithms are originally designed for binary classification
- Approaches
 - one-against-rest (K classifiers required)
 - a binary classifier is constructed for a class against the other ($K-1$) classes
 - A vs. (B+C), B vs. (A+C), C vs. (A+B)
 - one-against-one ($C_2^K = \frac{K(K-1)}{2}$ classifiers required)
 - $K(K-1)/2$ classifiers are constructed for all pairs of two classes
 - A vs. B, B vs. C, A vs. C
 - majority voting + error correcting code

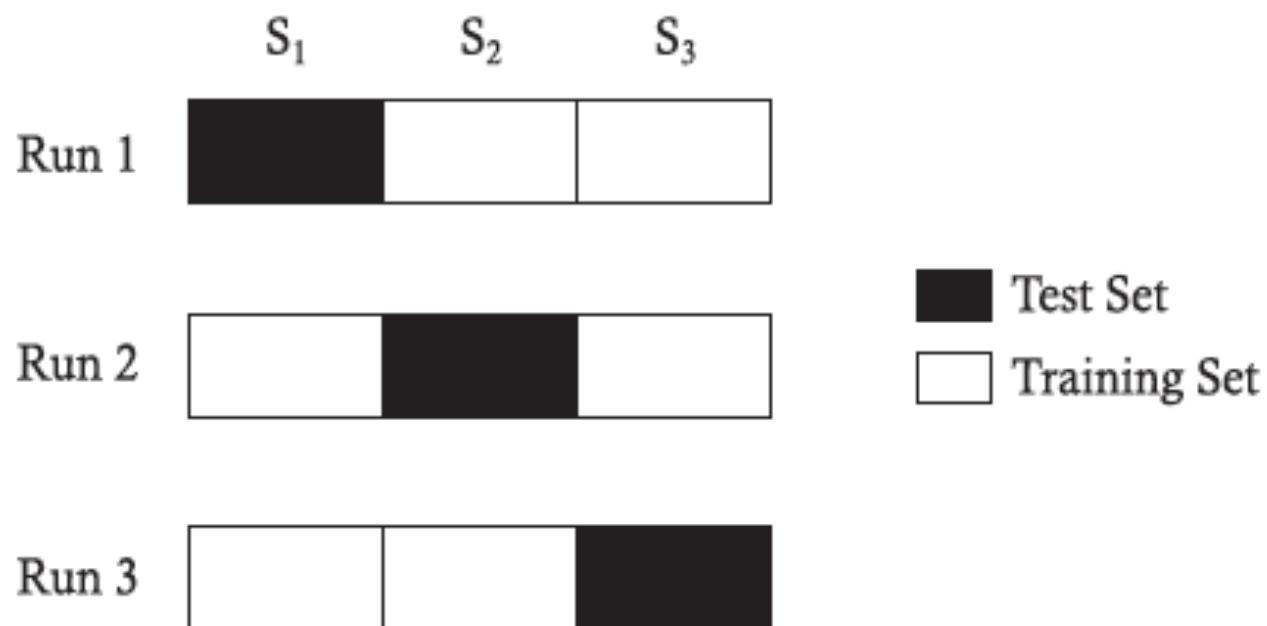
Performance Evaluation

Model Evaluation

- Purpose
 - To **estimate** performance of classifier on previously unseen data (test set)
- Approaches
 - Holdout
 - Reserve $k\%$ for training and $(100-k)\%$ for testing
 - Random subsampling: repeated holdout
 - Cross Validation (CV)
 - Partition data into k disjoint subsets
 - k -fold CV: train on $k-1$ partitions,
test on the remaining one
 - Leave-one-out: $k=n$

Cross-validation Example

■ 3-fold cross-validation



		Prediction Class	
		+	-
Actual Class	+	10	90
	-	10	890

Model A accuracy: $(10+890)/(10+90+10+890)=0.9$

		Prediction Class	
		+	-
Actual Class	+	90	10
	-	90	810

Model A accuracy: $(90+810)/(10+90+10+890)=0.9$

		Prediction Class	
		Cancer	Normal
Actual Class	Cancer	10	90
	Normal	10	890

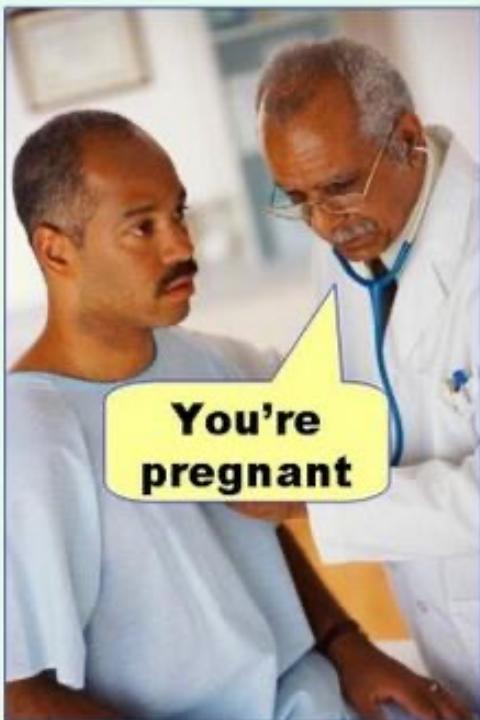
		Prediction Class	
		+	-
Actual Class	Cancer	90	10
	Normal	90	810

Confusion Matrix

		Prediction Class	
		+	-
Actual Class	+	TP, True Positive	FN, False Negative, (Type-II error)
	-	FP, False Positive, (Type-I error)	TN, True Negative

- True Positive Rate (sensitivity, recall) = $TP/(TP+FN)$
 - True Negative Rate (specificity) = $TN/(FP+TN)$
 - False Positive Rate = $FP/(FP+TN) = 1-TNR$
 - False Negative Rate = $FN/(TP+FN) = 1-TPR$
-
- Precision (Positive Predicted Value), $p= TP/(TP+FP)$
 - Recall $r = TP/(TP+FN)$
 - F1-Score = $\frac{2}{\frac{1}{r} + \frac{1}{p}} = 2rp/(r+p)$, harmonic mean between r & p

Type I error
(false positive)



Type II error
(false negative)



		Prediction Class	
		+	-
Actual Class	+	TP, True Positive	FN, False Negative, (Type-II error)
	-	FP, False Positive, (Type-I error)	TN, True Negative

Confusion Matrix (cont.)

		Prediction Class	
		+	-
Actual Class	+	$TP = TPR \times \alpha \times N$ $\alpha \times N$	$FN = (1 - TPR) \times \alpha \times N$
	-	$FP = (1 - TNR) \times (1 - \alpha) \times N$ $(1 - \alpha) \times N$	$TN = TNR \times (1 - \alpha) \times N$

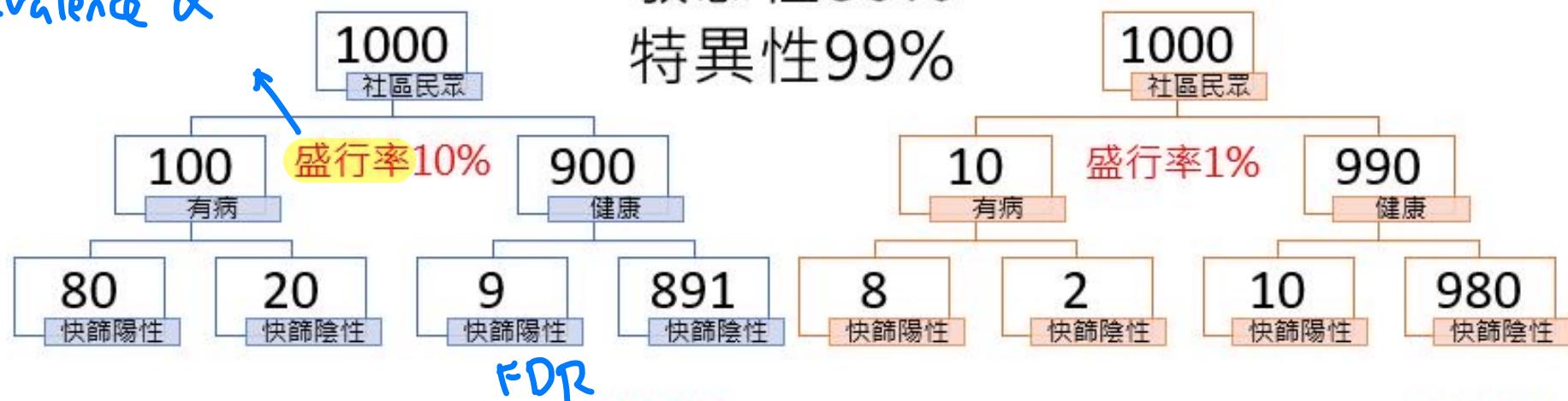
- True Positive Rate (**sensitivity**, recall) = $TP/(TP+FN)$
 - True Negative Rate (**specificity**) = $TN/(FP+TN)$
 - False Positive Rate = $FP/(FP+TN) = 1 - TNR$
 - False Negative Rate = $FN/(TP+FN) = 1 - TPR$
 - Precision (Positive Predicted Value), $p = TP/(TP+FP)$
 - Recall $r = TP/(TP+FN)$
 - F1-Score** = $\frac{2}{\frac{1}{r} + \frac{1}{p}} = 2rp/(r+p)$, harmonic mean between r & p
- ↳ a.k.a. F-measure

抗原快篩

敏感性80%

特異性99%

prevalence α



89位陽性, 9位是假的, 偽陽率10.1%

18位陽性, 10位是假的, 偽陽率55.5%

(Ref: 王介立醫師 FaceBook 2021/05/20)

		Prediction Class	
		+	-
Actual Class	+	TP, True Positive, 80	FN, False Negative, 20
	-	FP, False Positive, 9	TN, True Negative, 891

True Positive Rate (sensitivity, recall) = $TP/(TP+FN) = 80/(80+20) = 80\%$
 $\alpha = 10\%$

True Negative Rate (specificity) = $TN/(FP+TN) = 891/(9+891) = 99\%$

False Discovery Rate = $FP/(TP+FP) = 9/(80+9) = 10.1\%$ **FDR ≠ FPR**

抗原快篩

敏感性80%

特異性99%



89位陽性,9位是假的,偽陽率**10.1%**

18位陽性,10位是假的,偽陽率**55.5%**

(Ref: 王介立醫師 FaceBook 2021/05/20)

		Prediction Class	
		+	-
Actual Class	+	TP, True Positive, 8	FN, False Negative, 2
	-	FP, False Positive, 10	TN, True Negative, 980

$\alpha = 1\%$

True Positive Rate (sensitivity, recall) = $TP/(TP+FN) = 8/(8+2) = 80\%$

True Negative Rate (specificity) = $TN/(FP+TN) = 980/(10+980) = 99\%$

False Discovery Rate = $FP/(TP+FP) = 10/(10+8) = 55.5\%$

Precision, Recall F1-Score

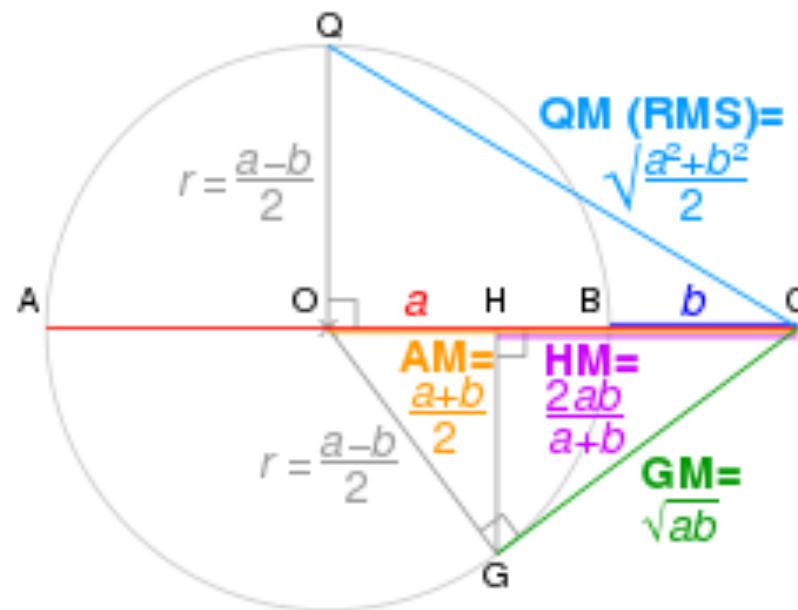
What's wrong ?

The F1 score is not
between precision &
recall in this table

Precision	Recall	F1-Score
0.4973	0.4973	0.4958
<u>0.5291</u>	<u>0.5179</u>	<u>0.4680</u>
<u>0.5289</u>	<u>0.5217</u>	<u>0.4919</u>

(Ref: <https://www.facebook.com/yuenhsien.tseng/posts/pfbid0odmBMjjFwpq5eiZSQzpGS2PU3sK2TZs7GNjj3t23W7uc5tT1YTxrvR4V6S56XKpvl>)

Precision, Recall & F1-Score

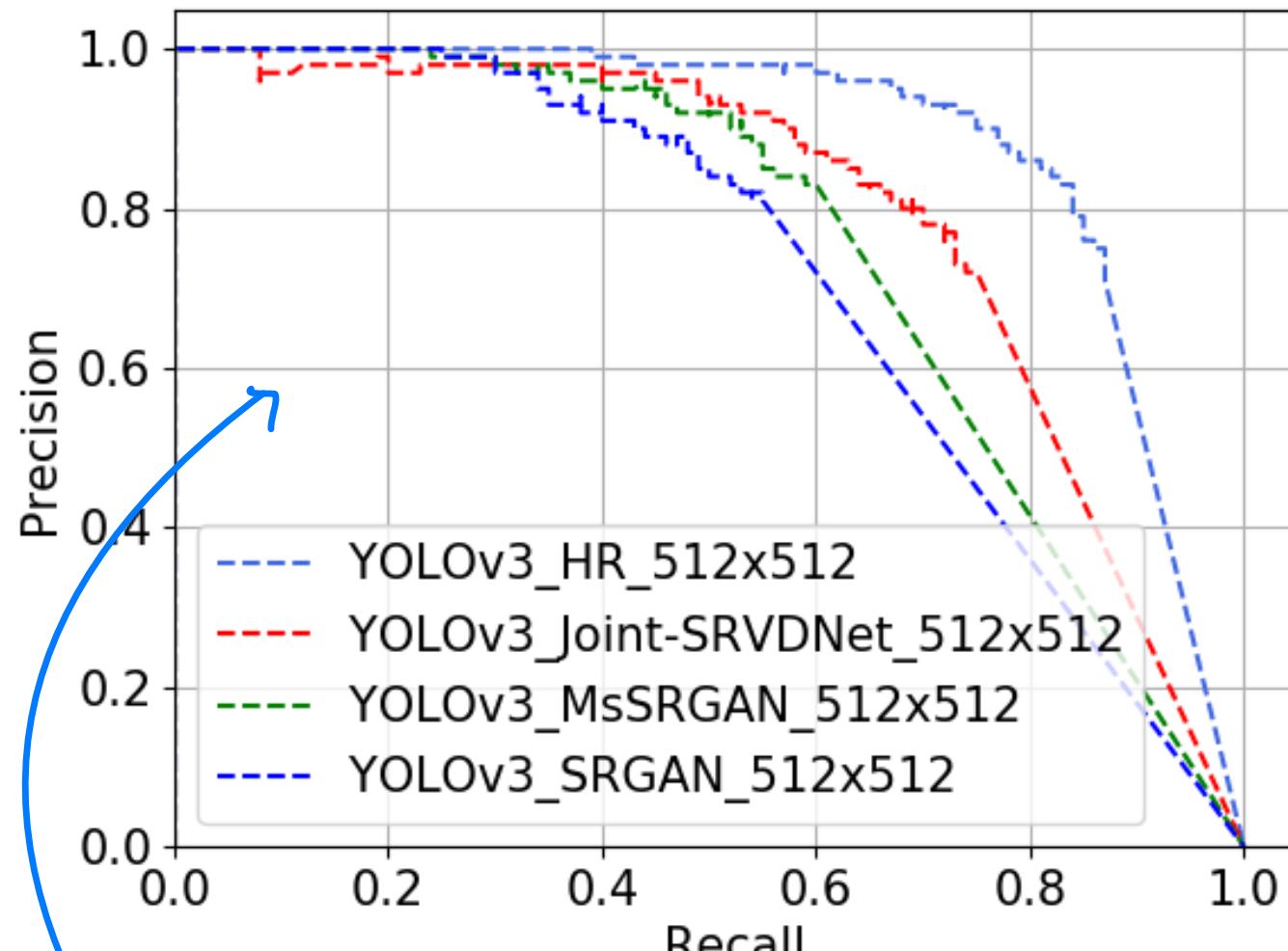


$$\text{F-measure} = \frac{2}{\frac{1}{r} + \frac{1}{p}} = 2rp/(r+p),$$

harmonic mean between r & p

$\max(a,b) >$ **Quadratic Mean (QM)** $>$ **Arithmetic Mean (AM)** $>$
Geometric Mean (GM) $>$ **Harmonic Mean (HM)** $>$ $\min(a,b)$

Precision vs Recall Curve



AUC (Area Under Curve)

==== Confusion Matrix ===

a	b	c	d	e	f	g	h	i	<-- classified as
589	104	13	9	4	2	2	64	13	a = Italian
106	531	22	28	7	1	15	74	16	b = French
19	36	644	22	8	10	7	33	21	c = Mexican
8	33	16	630	6	31	16	18	42	d = Indian
1	18	7	16	631	32	86	5	4	e = Chinese
10	9	14	34	55	637	32	1	8	f = Thai
9	15	8	15	86	12	642	6	7	g = Japanese
70	100	56	23	9	4	10	478	50	h = Spanish
7	22	10	38	2	2	4	39	676	i = Moroccan

Class Imbalance Problem

Class Imbalance Problem

- Consider a 2-class problem
 - Number of Class NO examples = 990
 - Number of Class YES examples = 10
- If a model predicts everything to be class NO, accuracy is $990/1000 = 99\%$
 - This is misleading because this trivial model does not detect any class YES example
 - Detecting the rare class is usually more interesting (e.g., frauds, intrusions, defects, etc)

		Prediction Class	
		+	-
Actual Class	+	0	10
	-	0	990

Class Imbalance Problem (cont.)

- Correct classification of rare class has greater value than that of majority class.
 - If 1% of credit card transactions are fraudulent, a model that predicts every transaction as legitimate has 99% accuracy

- Approaches
 - Data level
 - Sampling-based methods
 - Data augmentation
 - Evaluation level
 - Alternative Metrics
 - Algorithm level
 - Cost-sensitive learning
 - Thresholding

Sampling-Based Approaches

- To modify the distribution of instances so that the rare class is well represented in the training set
 - Under-sampling
 - Over-sampling
 - Hybrid

Sampling-Based Approaches (cont.)

- Under-sampling
 - Random sample the training data of majority class
 - Under-sampling multiple times & produce ensemble classifier
 - Limitation
 - Some of the useful majority examples may not be chosen for training
 - Can loss important information
 - Smaller sample of majority instances may have a **higher variance** than the original instances

underfitting

Sampling-Based Approaches (cont.)

- Over-sampling
 - Replicates the training data of rare class
 - Duplicating a positive instance is analogous to doubling its weight during the training stage
 - Limitation
 - Replicated minority examples have an artificially **lower variance** when compared with their true distribution in the overall data
 - Can cause overfitting
 - Alternative approach is to generate synthetic minority instances in the neighborhood of existing minority instances (SMOTE)

Data Augmentation

- Technique of artificially increasing the training set by creating modified copies of a dataset using existing data.
- Data augmentation to generate more samples for the minority class.
- Image Augmentation
 - Geometric transformations: randomly flip, crop, rotate, stretch, and zoom images.
 - Color space transformations: randomly change RGB color channels, contrast, and brightness.
 - Kernel filters: randomly change the sharpness or blurring of the image.
 - Random erasing: delete some part of the initial image.
 - Mixing images: blending and mixing multiple images.
- The biases in the original dataset persist in the augmented data.

Alternative Metric: ROC

- A graphical approach for displaying trade-off between detection rate and false alarm rate
- Developed in 1950s for signal detection theory to analyze noisy signals
- Receiver Operating Characteristics (ROC)
 - Graphical approach for displaying tradeoff between true positive rate & false positive rate of a classifier
 - $\text{TPR}=0, \text{FPR}=0$: predicts each instance to be negative
 - $\text{TPR}=1, \text{FPR}=1$: predicts each instance to be positive
 - $\text{TPR}=1, \text{FPR}=0$: ideal model
 - Good model: up-left corner
- *Good at handling imbalanced data*

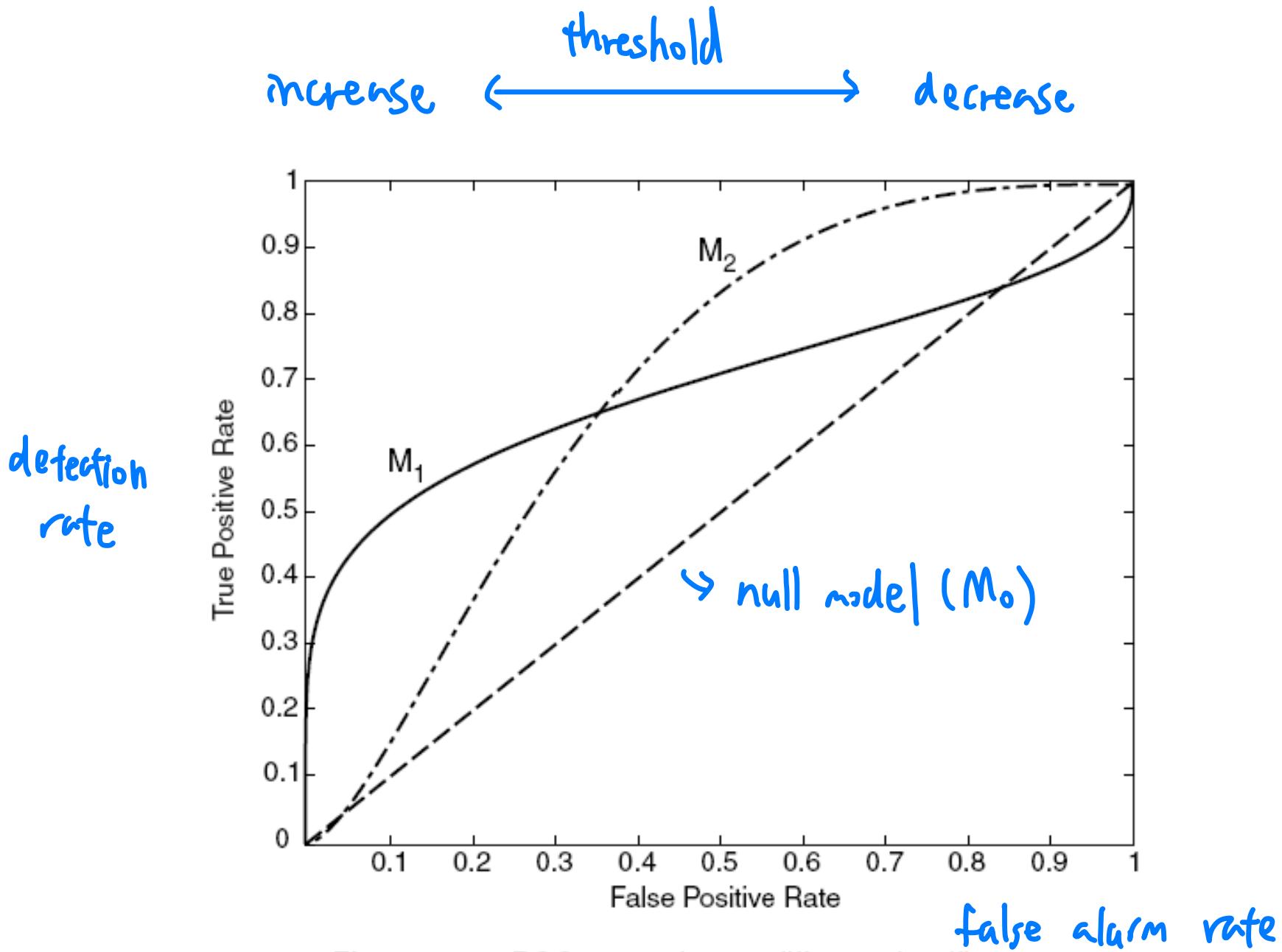


Figure 5.41. ROC curves for two different classifiers.

* M_1, M_2 : models (different parameters)

AUC

- Area Under receiver operating characteristics Curve
 - Another approach for evaluating which model is better on average.
 - If the model is perfect, AUC would equal 1
 - If the model simply perform random guessing, AUC would equal 0.5
 - A model that is strictly better than another would have a larger AUC.

Generating an ROC Curve

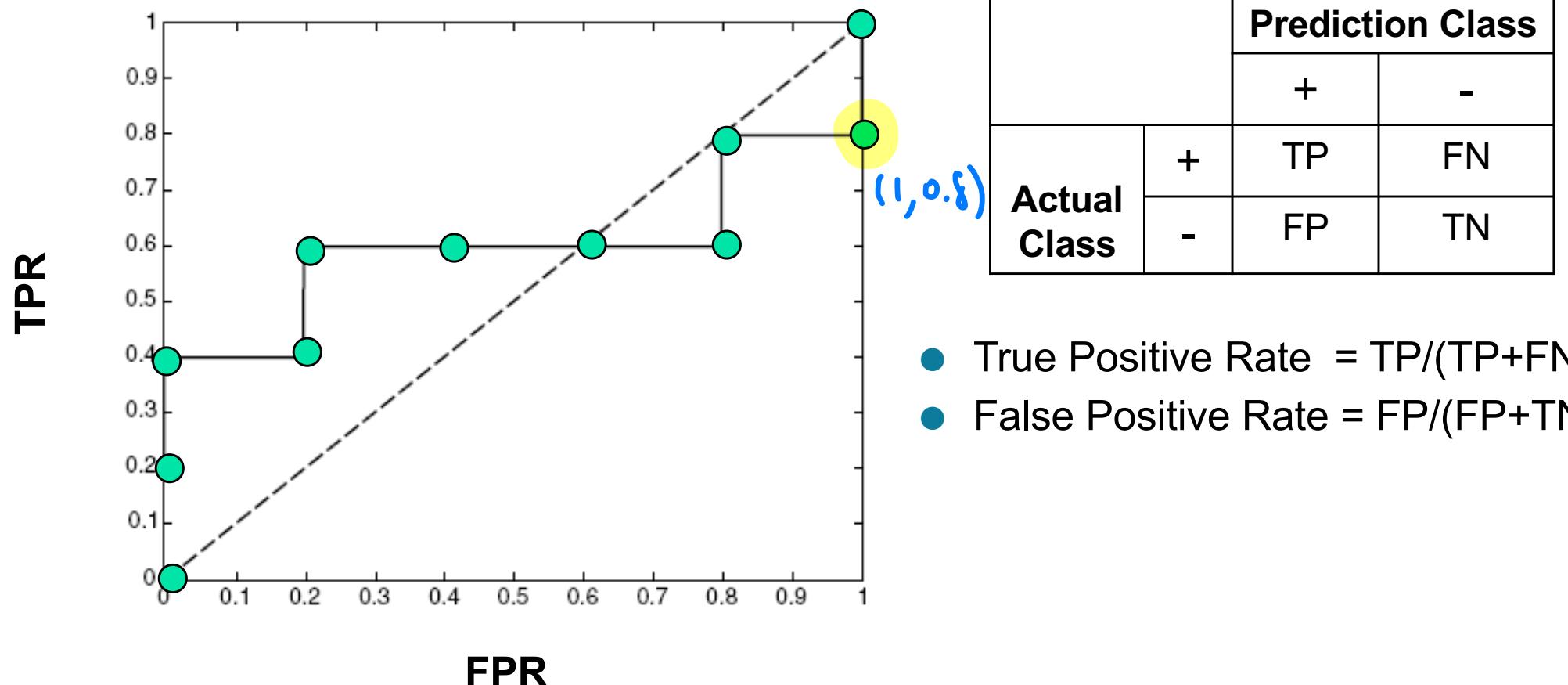
- To draw an ROC curve, the model should be able to produce a **continuous-value** output to **rank** its prediction, from the most likely data to be classified as a positive class to the least likely one.
- Procedure
 1. Assign the lowest ranked data and those ranked above it to be positive. $\text{TPR} = \text{FPR} = 1$
 2. Assign the bottom-1 to be negative, while those above it as positive. Update TPR, FPR.
 3. Assign the bottom-2 to be negative, while those above it as positive. Update TPR, FPR
 4. Repeat accordingly until the top ranked data.
 5. Plot TPR against FPR of this model

Generating an ROC Curve (cont.)

Instance	Score	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

- Use a classifier that produces a continuous-valued score for each instance
 - The more likely it is for the instance to be in the + class, the higher the score
- Sort the instances in decreasing order according to the score
- Apply a threshold at each unique value of the score
- Count the number of TP, FP, TN, FN at each threshold
 - $TPR = TP/(TP+FN)$
 - $FPR = FP/(FP + TN)$

Class	+	-	+	-	-	-	+	-	+	+	
	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0



Cost-Sensitive Learning

- Cost-sensitive learning takes the cost matrix into consideration during model learning & generates a model that has the lowest cost.
- Cost matrix : $C(+, -)=100$

		Prediction Class	
		+	-
Actual Class	+	0	100
	-	1	0

penalize hardly → 100

penalize lightly ↓ 1

Thresholding Strategy

- adjusting the decision threshold used to assign class labels based on predicted probabilities.
- By default, many classifiers use a threshold of 0.5, meaning samples with probabilities ≥ 0.5 are classified as the positive class, and others as the negative class.
- For imbalanced data, this default threshold may not be optimal, as it often leads to a bias toward the majority class.
- Adjusting the threshold allows to better balance precision and recall, especially for the minority class.

Ways to Improve Classification Accuracy

Ways to Improve Classification Accuracy

- Add more data: presence of more data results in better and accurate models.
- Treating missing & outlier values

With Missing Values

Name	Weight	Gender	Play Cricket/ Not
Mr. Amit	58	M	Y
Mr. Anil	61	M	Y
Miss Swati	58	F	N
Miss Richa	55		Y
Mr. Steve	55	M	N
Miss Reena	64	F	Y
Miss Rashmi	57		Y
Mr. Kunal	57	M	N

After imputation of missing values

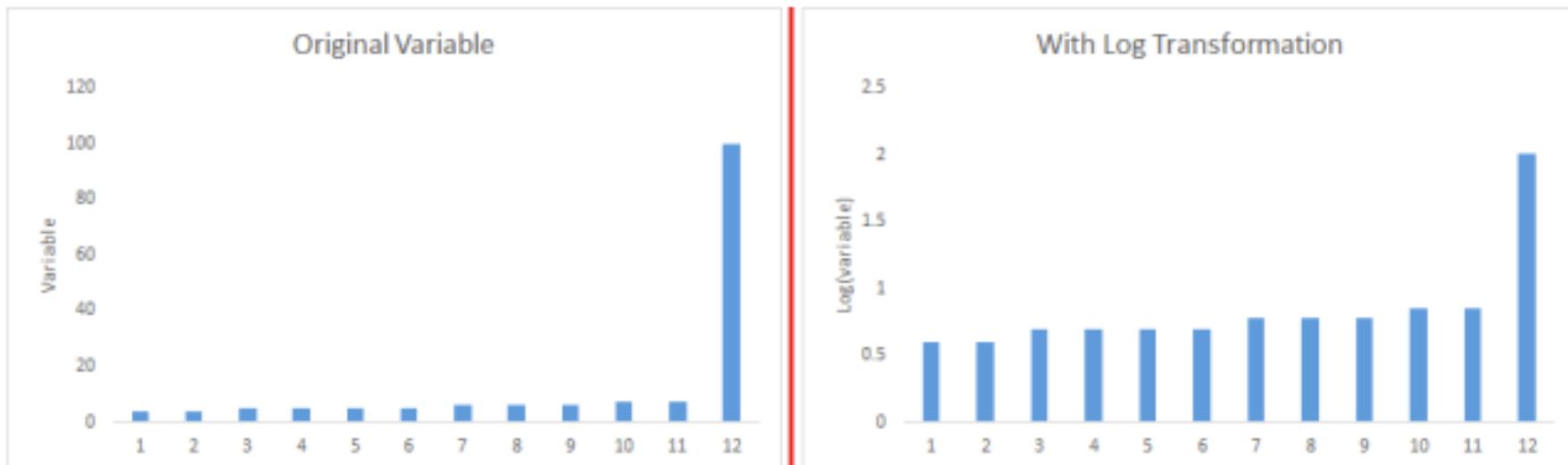
Name	Weight	Gender	Play Cricket/ Not
Mr. Amit	58	M	Y
Mr. Anil	61	M	Y
Miss Swati	58	F	N
Miss Richa	55	F	Y
Mr. Steve	55	M	N
Miss Reena	64	F	Y
Miss Rashmi	57	F	Y
Mr. Kunal	57	M	N

Gender	#Students	#Play Cricket	%Play Cricket
F	2	1	50%
M	4	2	50%
Missing	2	2	100%

Gender	#Students	#Play Cricket	%Play Cricket
F	4	3	75%
M	4	2	50%

Ways to Improve Classification Accuracy (cont.)

- Feature Engineering
 - Feature transformation
 - normalization
 - skewness removal
 - numeric: discretization
 - nominal: one-hot encoding
 - Feature creation: date → day of a week
 - Feature crossing: day-time → day of a week x hours (7×24)



Ways to Improve Classification Accuracy (cont.)

- Feature Selection
- Multiple algorithms
- Algorithm tuning (*parameter tuning*)
- Ensemble method
- Cross-validation

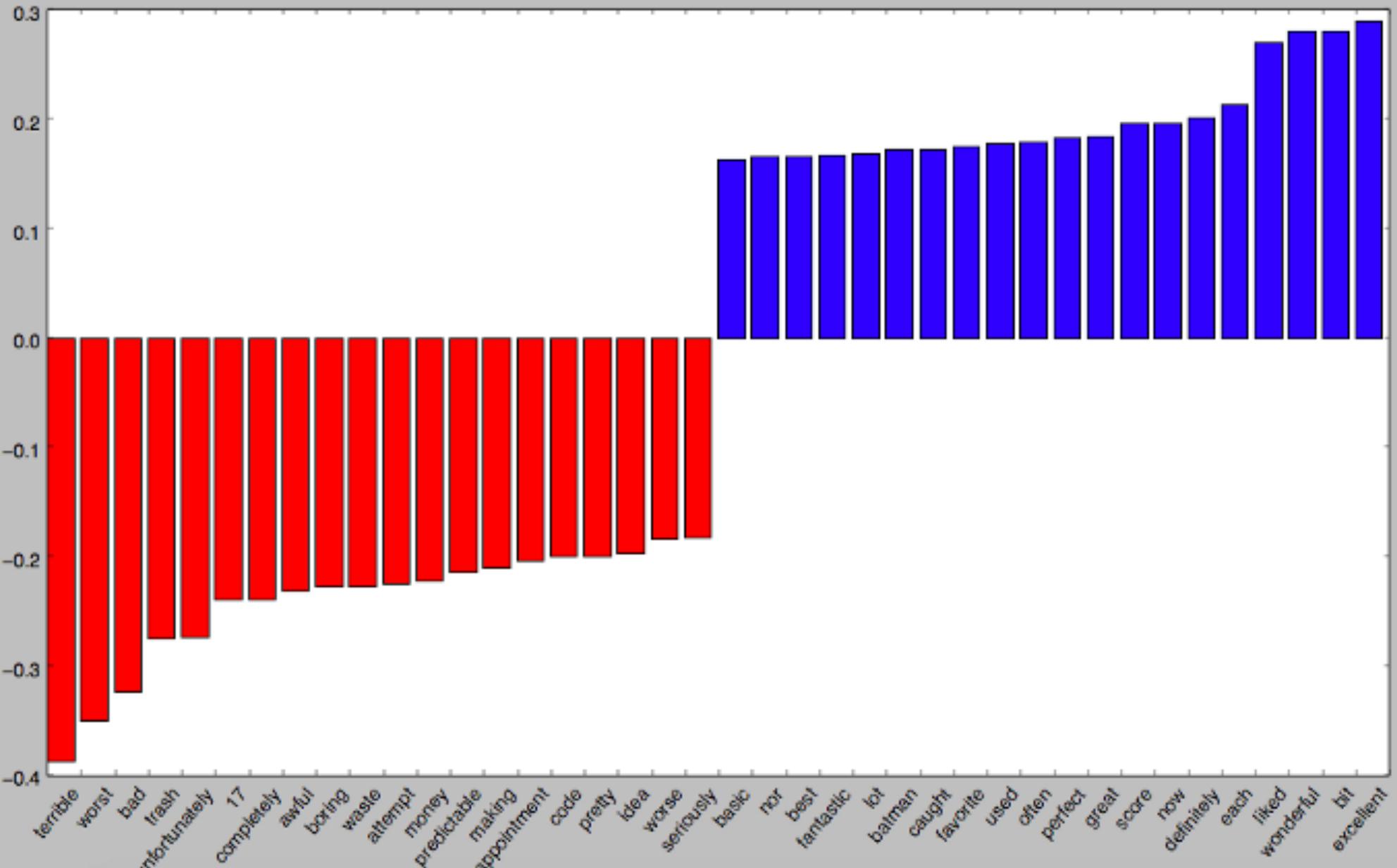
Feature Engineering

- Deep learning promise no more feature engineering?
 - We're still very far from that point
 - Many ML models in industry aren't deep learning

Feature Normalization

- Some classification algorithms are fundamentally based on distance measures
 - K-Nearest Neighbor
 - Support Vector Machine
 - Neural Network
- Feature scaling is essential for distance-based classifier
- Feature scaling is also useful for noise removal
- Normalization to [0,1]
 - Min-max normalization: $x' = \frac{x - min_A}{max_A - min_A}$
 - Z-score normalization: $x' = \frac{x - mean_A}{std_A}$

Feature Importance



Machine learning accuracy rule

- 0-80%: one day
- 80-90%: one week
- 90-95%: one month
- 95-97%: three months
- 97-99%: one year (or never)

作者 scan33scan33 (亨利喵)

看板 Oversea_Job

標題 Re: [北美] 請問machine learning的工作內容

時間 Fri Sep 19 13:41:02 2014

工作完來回一下好了～
我不知道各種業界怎樣～我就說我待過的公司跟teams

先說背景，不才小時候愛玩ML比賽，正值ML還不太紅的200x年，
贏了一些獎，覺得自己很厲害～

後來去學校發了些paper，自我覺得超強。

就衝去hackathon然後試試看startup，結果很慘。
只好join Google休息一下，繼續作machine learning，
做了兩年，幫公司賺了不少錢可是paper都發不出來，來點坐領乾薪的魯蛇分享。

先講玩比賽的時候：
比賽的時候我都只會一招，就是把feature expand很大，
然後用liblinear跑一次。

基本上那時候我是相信，只要有一個好的feature engineering tool
人生就無敵了。

我當時就寫了個這個：

<http://www.csie.ntu.edu.tw/~b95028/software/lib-gundam/>

然後我就去念博班：

然後我就去念博班：

唸了博班發現，很多paper都亂寫，model超漂亮，跑出來超overfit。

用一個paper的model跑另一個paper的数据很常就會GG。

最後發現最強的還都只有liblinear跟隨機森林。

於是我就生氣了，想說去hackathon或是開開startup

初出茅廬：

當時做了一些health data，然後show说自己可以predict acc多高，覺得超屌。

我就去跟別人present，可是別人都看不懂。

這時候才知道業界很多人重視的一件事情「可解釋性」。

你有一個好model，如果不能解釋哪個feature強，在很多領域是無用的。

（像health他們好像就很注重，病的causality關係之類的。

像你可以predict一個人會生病，不過他想要知道哪個feature讓他生病的。

這種東西很多時候是model很難去數理上含括的。）

在Google：

基本上我現在都還是一直在用liblinear。

我遇到的很多不是純粹learning for accuracy的問題。

很多時候像stability很重要(e.g. 當有missing features，你的model是不是不會go crazy)

有時候生data很重要，data生不好，很容易target leak，不小心train出來accuracy就100%，還以為自己要得Turing Award了。

有時候prediction速度很重要，如果query很多，要怎樣才能快速serve。

我覺得其實很多work都會越作越general，但是多數是從domain example開始作。
就像可能你今天作text classification很猛，可是你可能想要improve。
你第一件事情大概還是去看data，看看是不是哪些字常常會是noise，
還是sentence的parser不好。那就會從中改善，最後也會越作越general。
最後可能就會有很好得方法可以給你在很多問題上都有
10%的accuracy gain然後改變世界。

但是其實人生大部分的事情是不能generalized的，就像愛情(?)。
所以大部分的事情都只是pointwise fix，例如說：cross兩個feature。
通常這種pointwise fix，都只會給你1%的accuracy gain。

所以這時候我就要來亂引用了，有人說過
「1% gain是engineering，10% gain是research。」

最後，公司還是要賺錢的，所以也很難一直給你時間想好玩得新演算法去try。
我現在的mode就是邊做engineering邊做research，
也期望有一天能發個ICML Best Paper，
但是大部分的時間其實都還是找個新feature然後重跑liblinear。

對了，liblinear真的很強。請愛用liblinear！

※ 引述《milc (milc)》之銘言：

- : 大家好
- : 不好意思請問一下關於ML的工作內容，
- : 我是在coursera 上Andrew Ng的MI 影片課程自學

Machine Learning in Research

we spend months
trying hard to design an
algorithm complex
enough to gain 0.3%
improvement over
previous best models

Machine Learning in Engineering

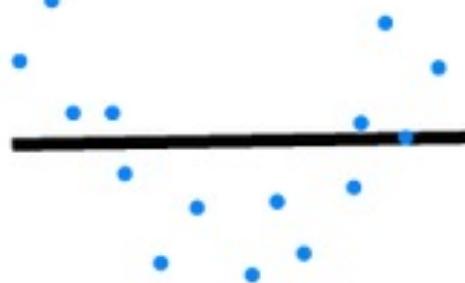
we use a simple
algorithm and spend
one week annotating
more data to gain 3%
improvement over
previous week's result.



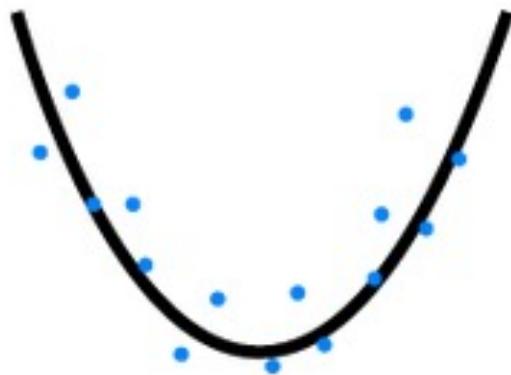
Overfitting



Overfitting



Underfitting



Desired

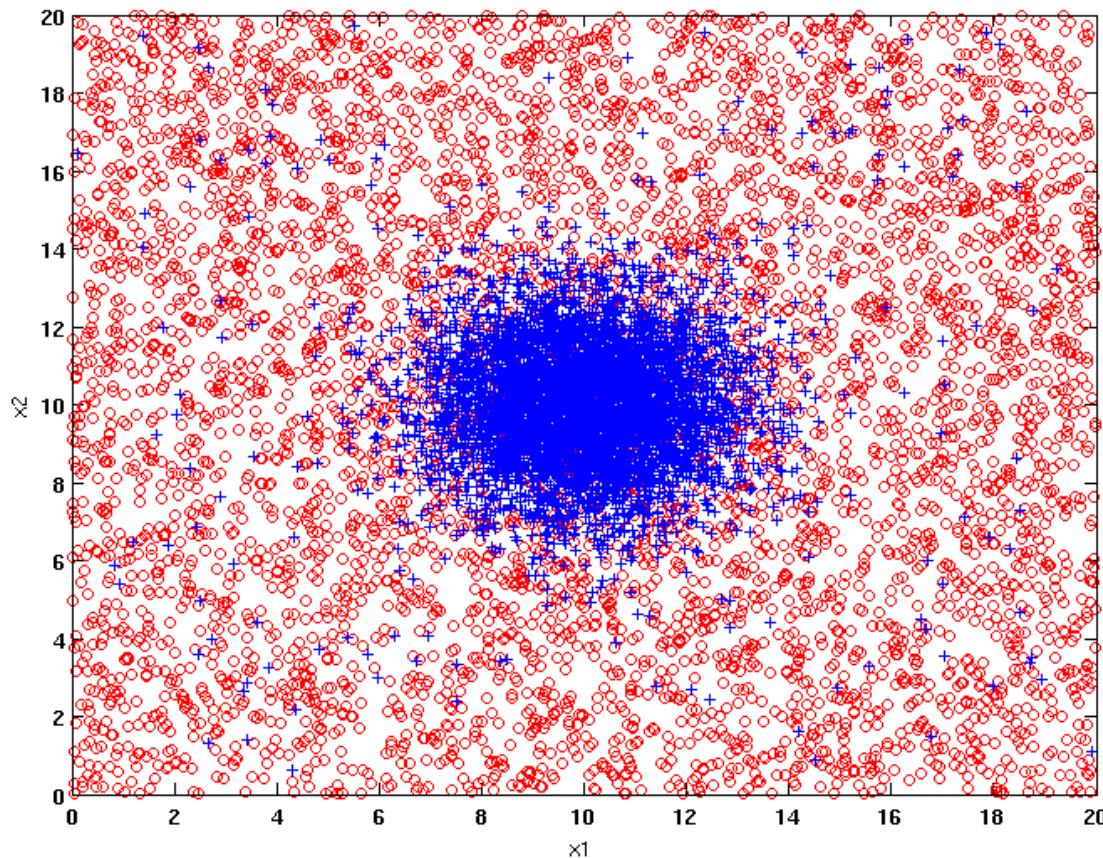


Overfitting

Classification Errors

- Training errors: Errors committed on the training set
- Test errors: Errors committed on the test set
- Generalization errors: Expected error of a model over random selection of records from same distribution

Example Data Set



Two class problem:

+ : 5400 instances

- 5000 instances generated from a Gaussian centered at $(10,10)$

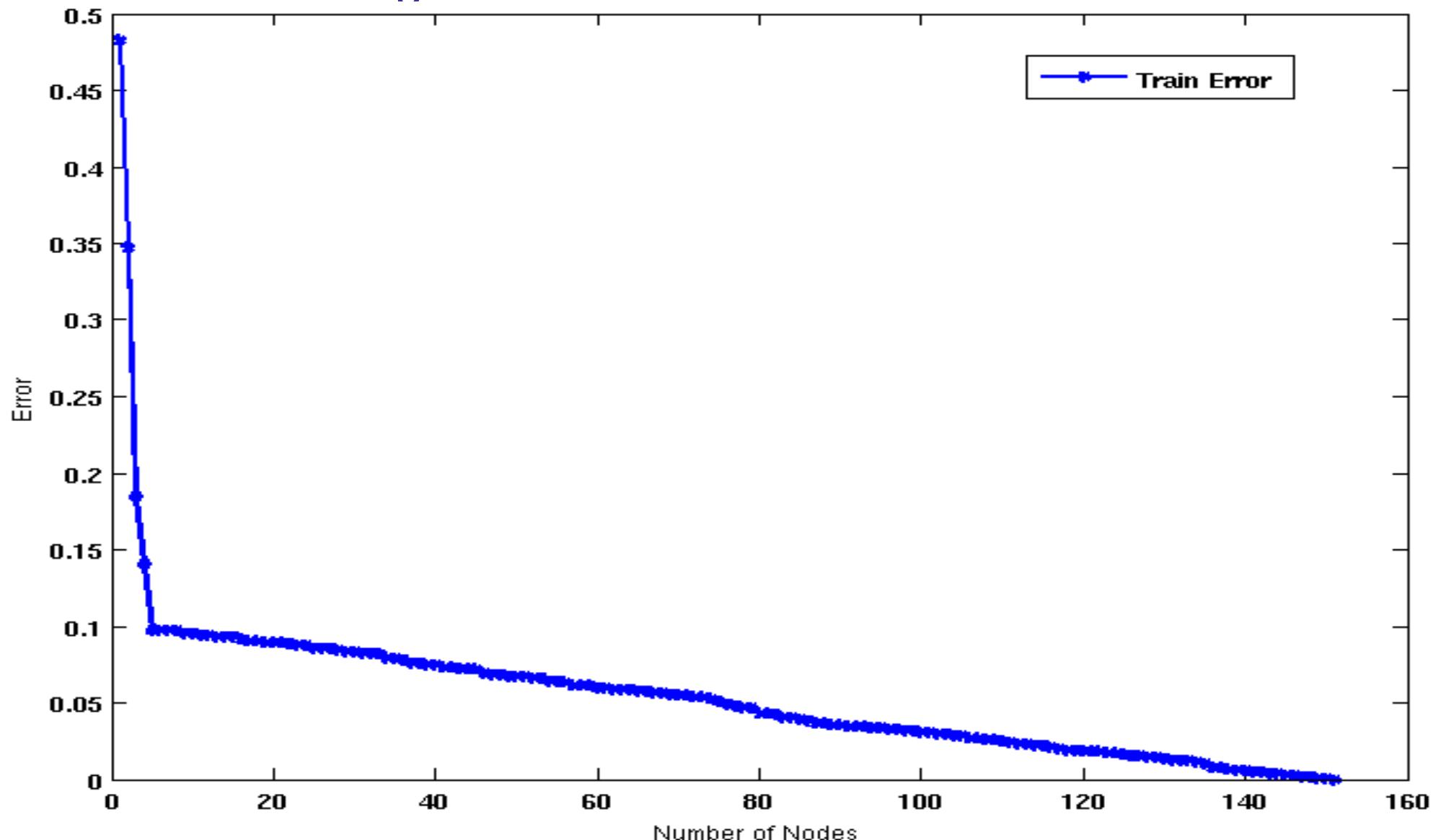
- 400 noisy instances added

o : 5400 instances

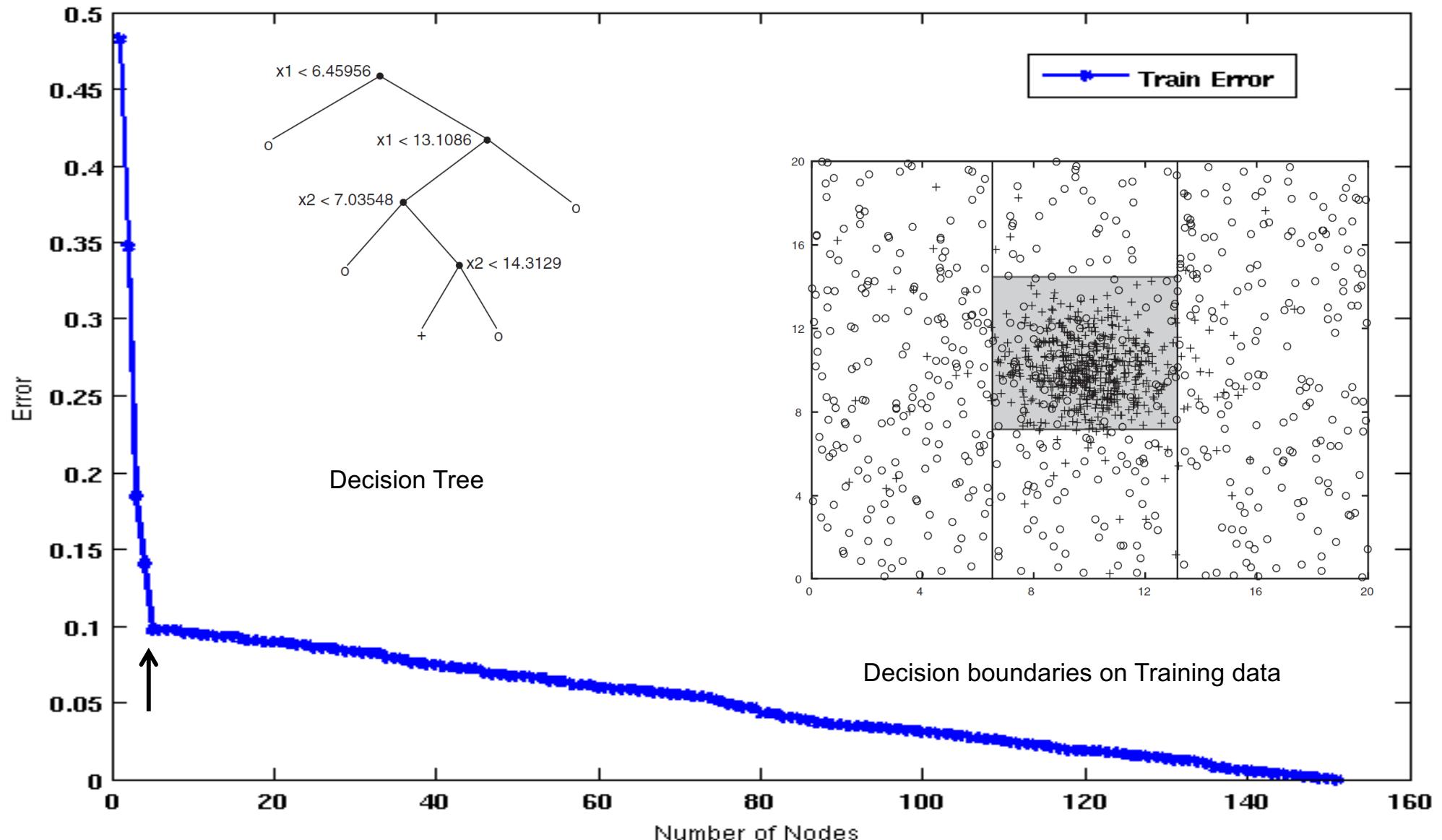
- Generated from a uniform distribution

10 % of the data used for training and 90% of the data used for testing

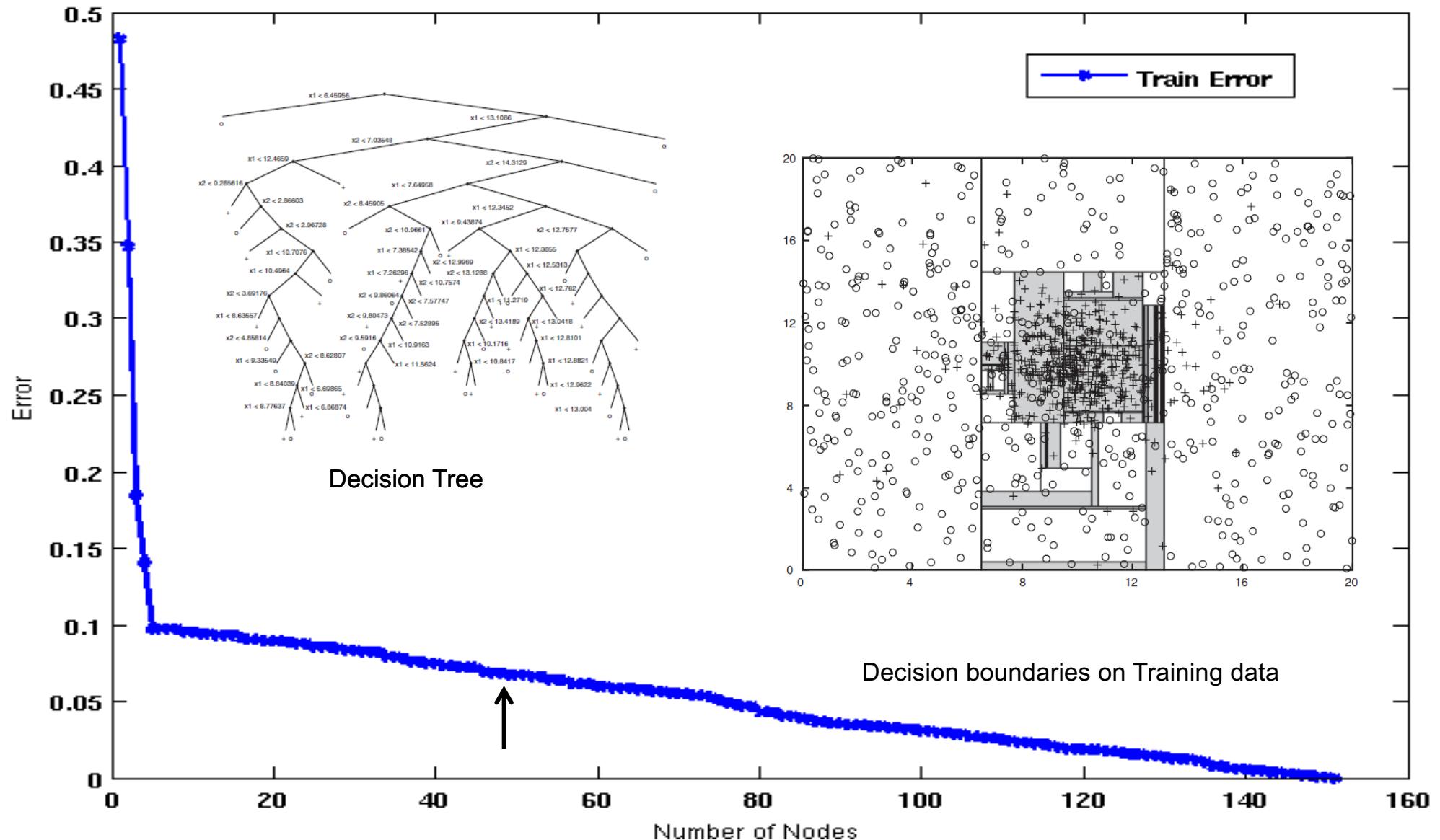
Increasing number of nodes in Decision Trees



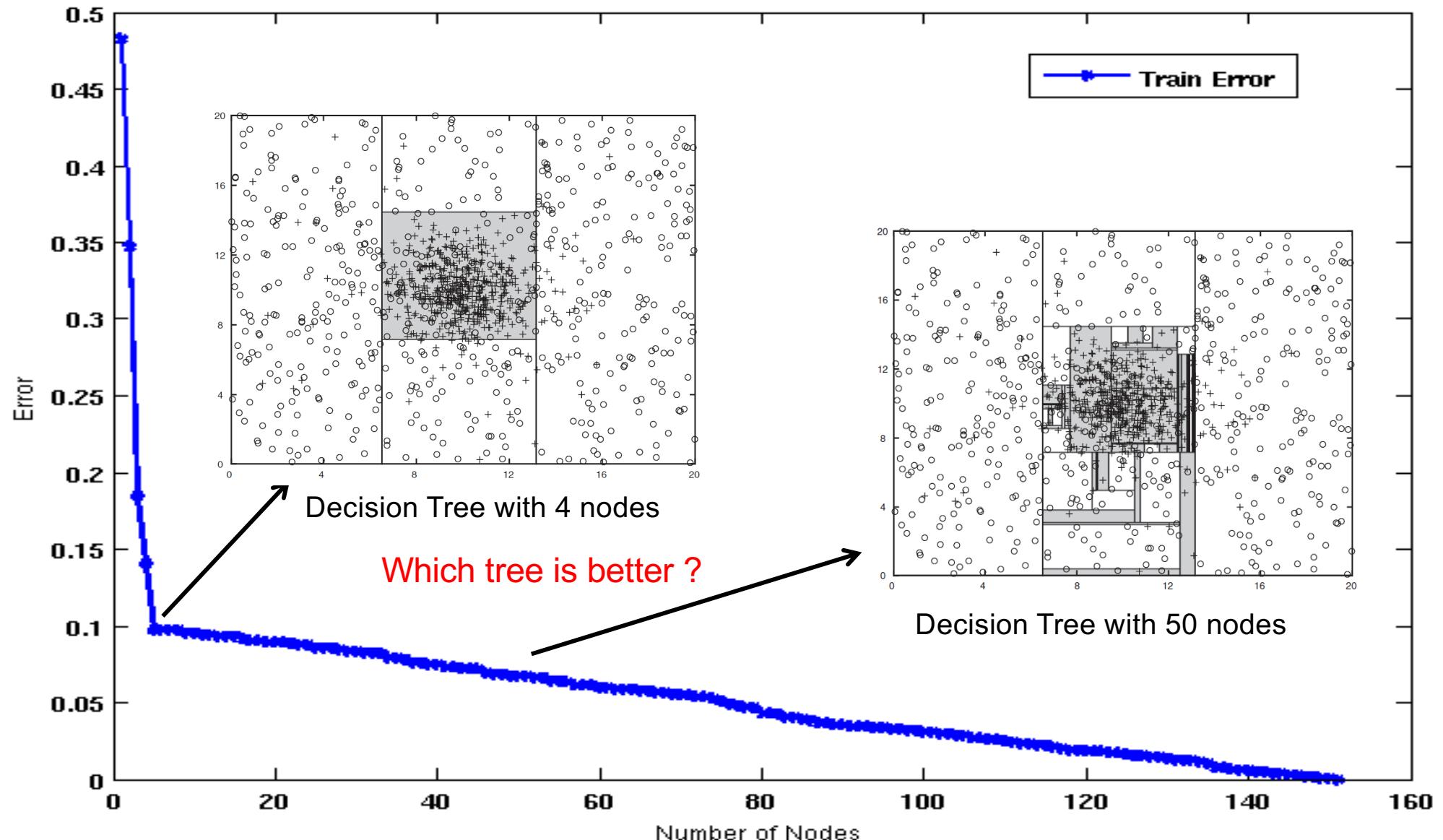
Decision Tree with 4 nodes



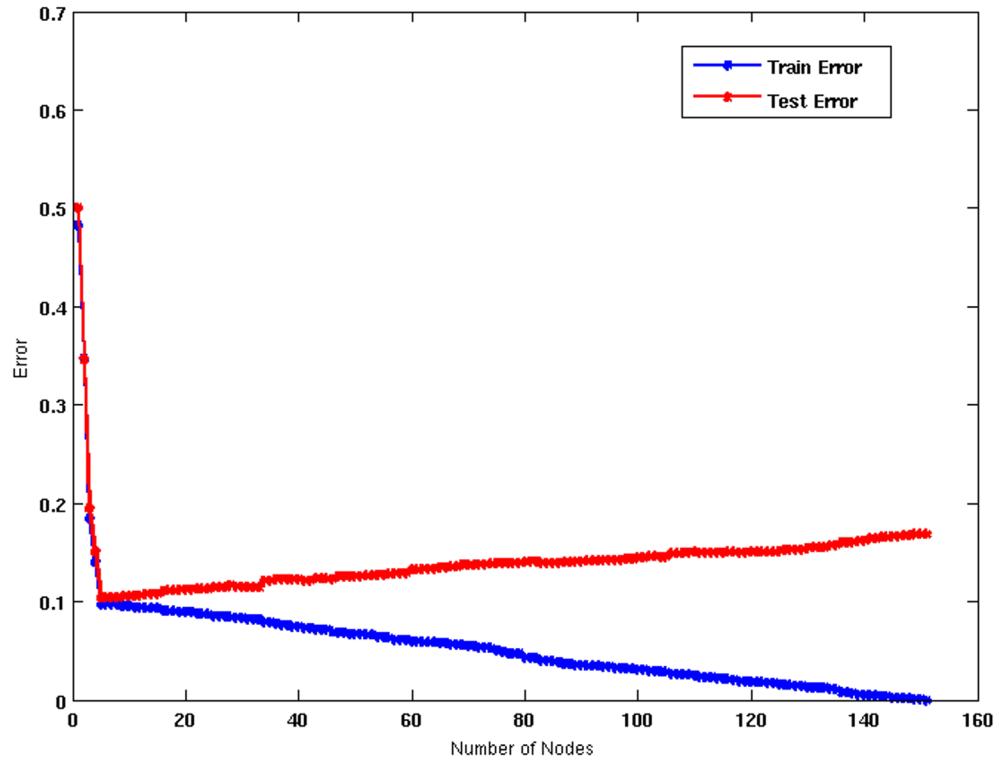
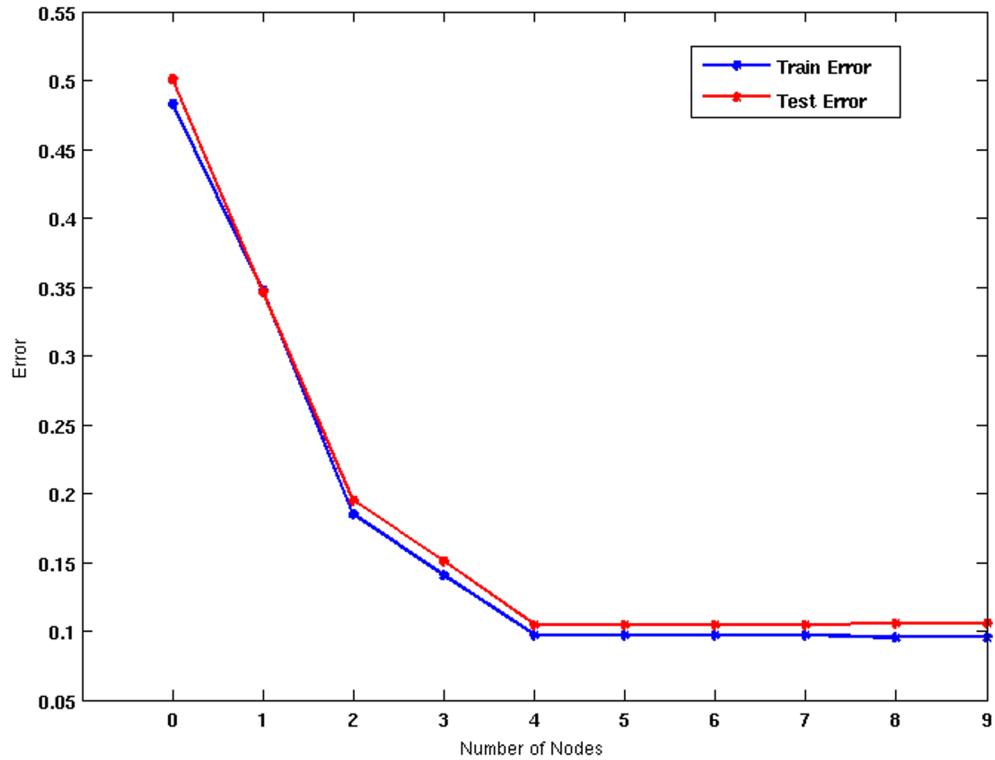
Decision Tree with 50 nodes



Which tree is better?



Model Underfitting and Overfitting

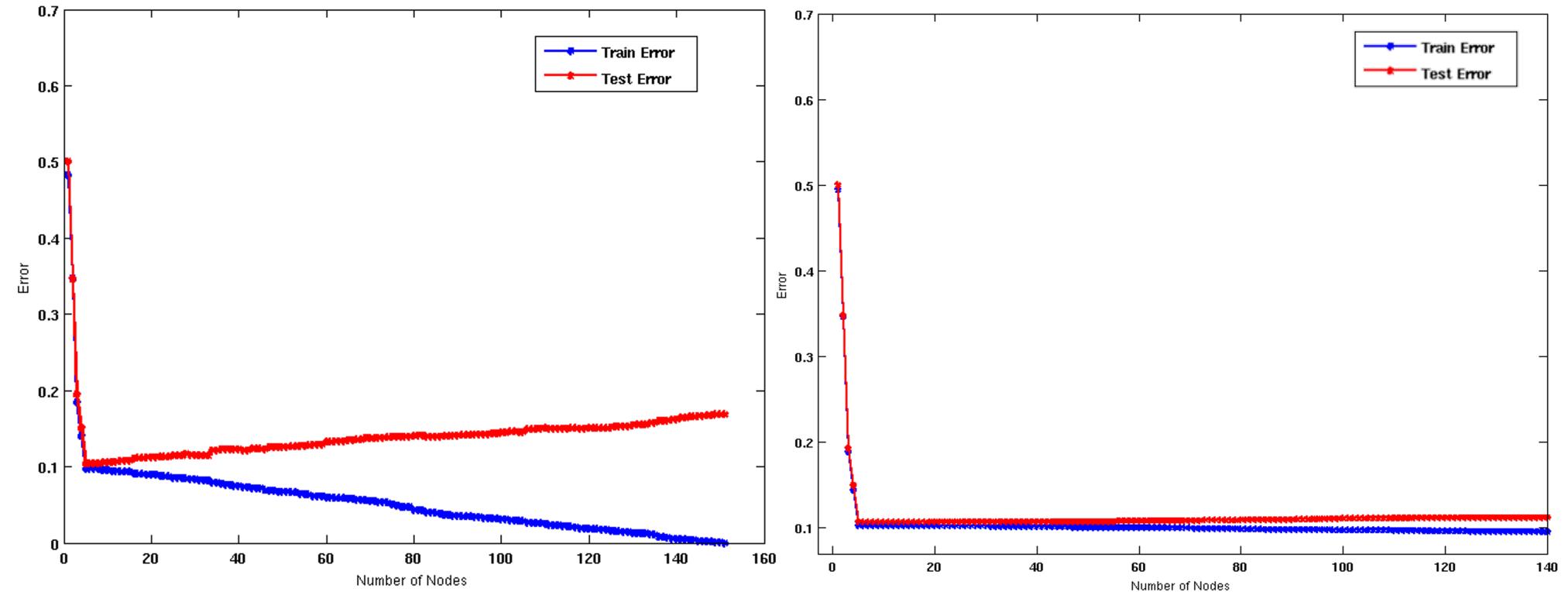


As the model becomes more and more complex, test errors can start increasing even though training error may be decreasing

Underfitting: when model is too simple, both training and test errors are large

Overfitting: when model is too complex, training error is small but test error is large

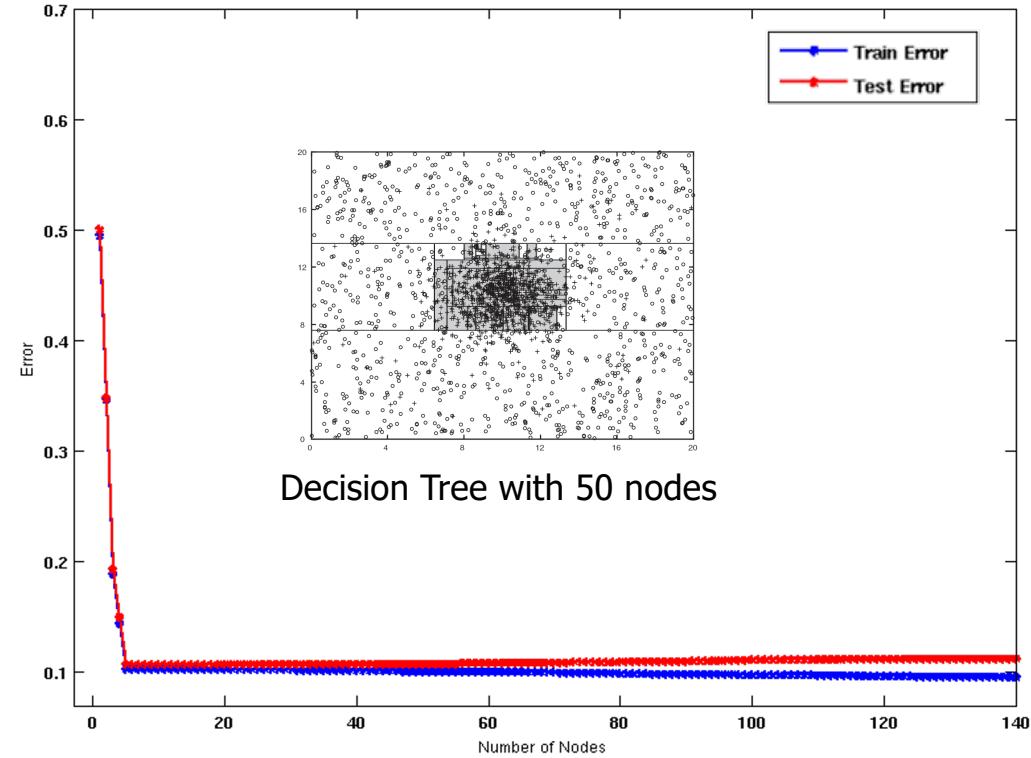
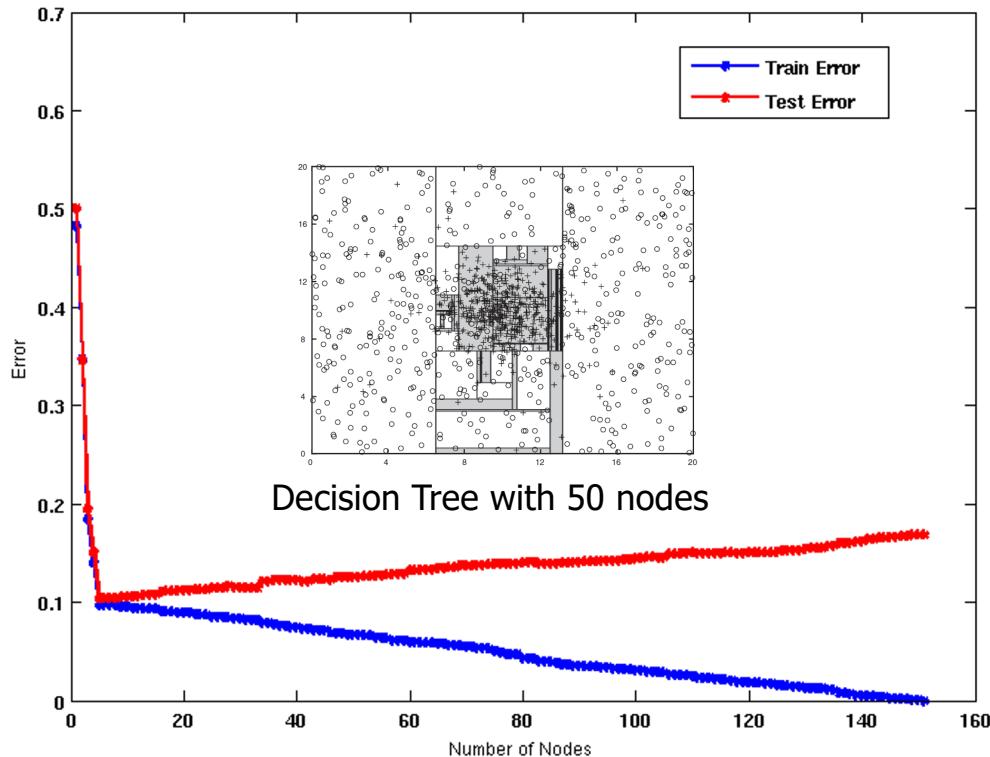
Model Overfitting – Impact of Training Data Size



Using twice the number of data instances

- Increasing the size of training data reduces the difference between training and testing errors at a given size of model

Model Overfitting – Impact of Training Data Size



Using twice the number of data instances

- Increasing the size of training data reduces the difference between training and testing errors at a given size of model

Reasons for Model Overfitting

- Not enough training data
 - In general, as we increase the number of training instances, the patterns learned from the training set resembles the true patterns in the overall data
- High model complexity
 - Generally, a more complex model has a better ability to represent complex patterns in the data
 - However, an overly complex model also has a tendency to learn specific patterns that do not generalize well over unseen instances

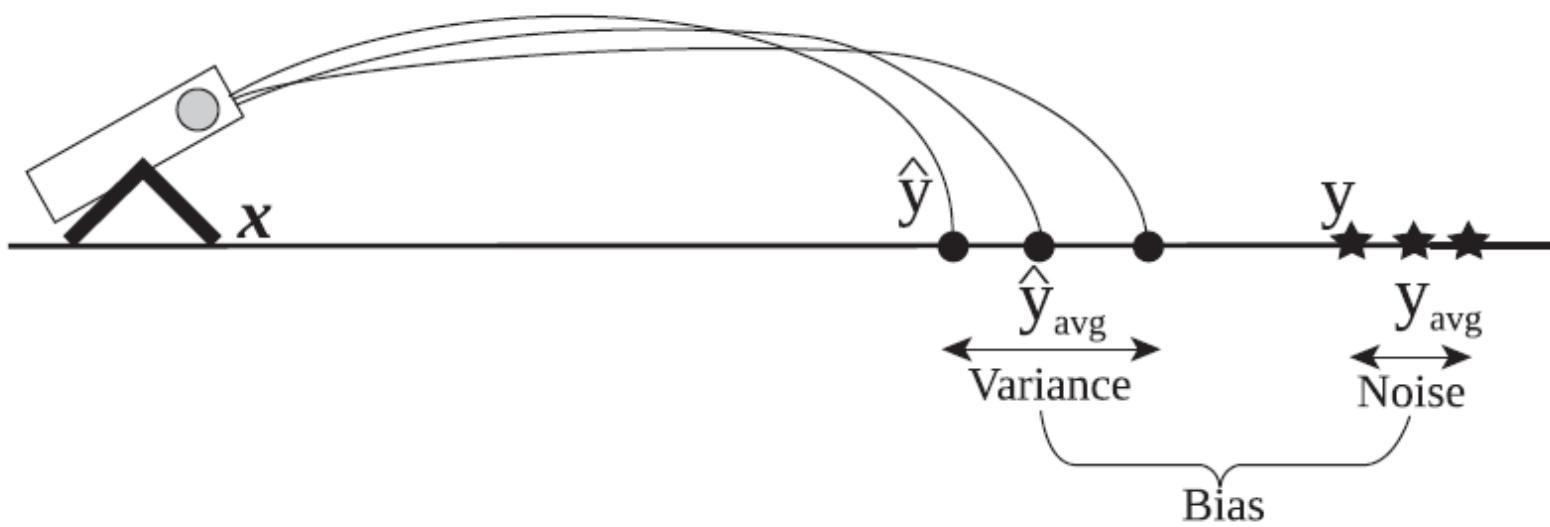
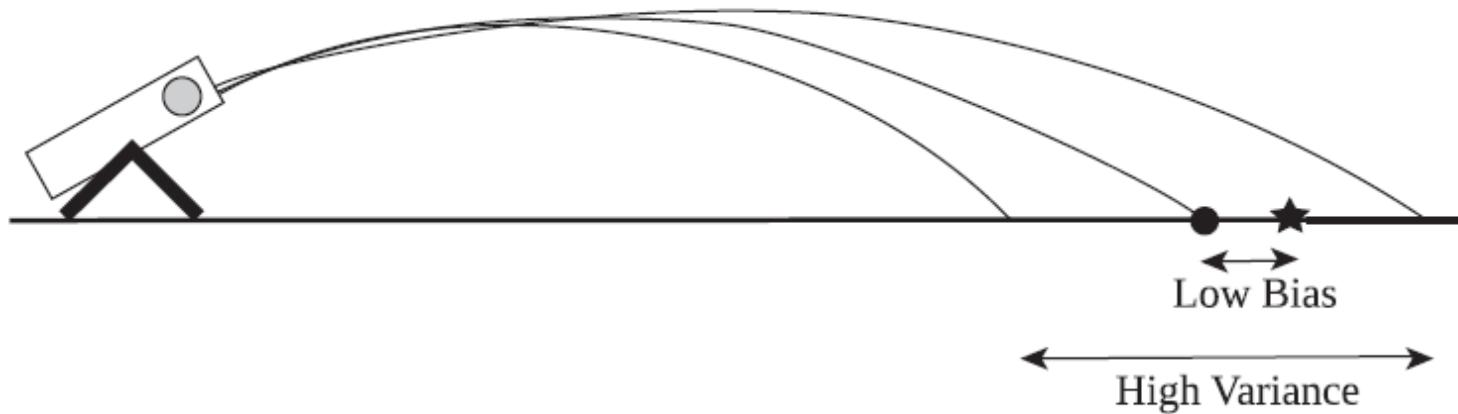
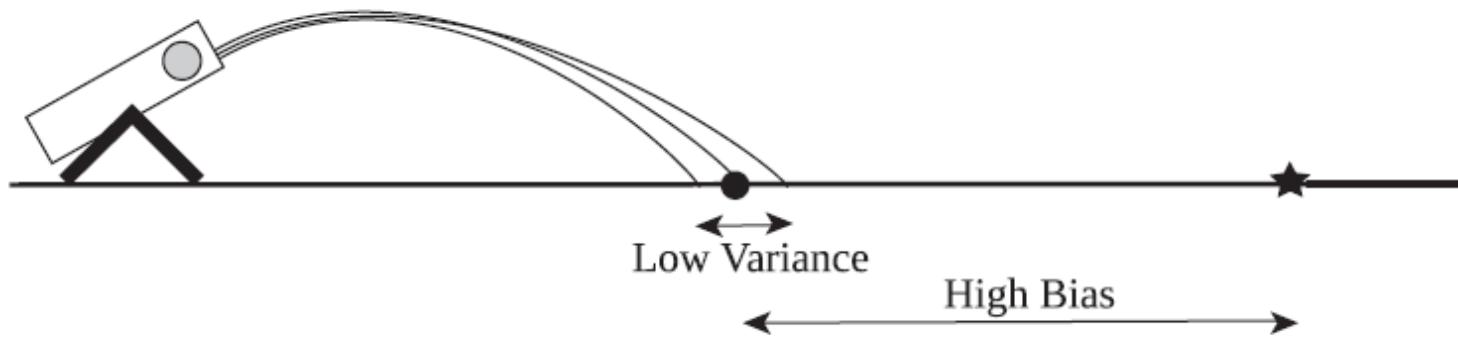


Figure 4.44. Bias-variance decomposition.



(a) Phenomena of Overfitting.



(b) Phenomena of Underfitting.

Figure 4.45. Plots showing the behavior of two-dimensional solutions with constant L_2 and L_1 norms.

Data Leakage Problem

展並納入考量。本研究資料集中一般新聞與假新聞的蒐集範圍與定義說明如下：

- { (一) 一般新聞以臺灣傳統四大報「聯合報」、「自由時報」、「蘋果日報」、「中國時報」的網站平台作為資料收集範圍。
- (二) 假新聞則以四個知名的假新聞內容農場（怒吼、密訊、琦琦看新聞、寰球軍事網）為資料收集範圍。
- (三) 資料庫 schema 欄位包括：新聞標題與新聞內文為必要欄位。新聞作者、出處（包含新聞網站名稱與該篇新聞網址鏈結）、新聞發布日期與新聞分類為非必要欄位。*data language*

假新聞資料庫則是以抓取內容農場之新聞為主，選取內容農場當作本研究的假新聞資料來源的理由是，內容農場乃指為了創造流量、賺取網路廣告分潤所建立的網站，它們多以各種合法、非法手段大量生產文章，原創性內容少且內容的真實性難以確認，由於內容農場並不主動管理內容，許多文章是由侵權盜用、抄寫、改寫而來，常摻雜浮濫內容與不實訊息，形成漏洞，因此，在缺乏新聞媒體的編輯規範和過程中，無法確保新聞內容的準確性和可信性。在此須注意的是內容農場之新聞虛假類型與程度彼此不同，有的是各處剪貼，有些採用無可信度之消息來源，又有些是直接造假，因此，本研究將所收集的內容農場新聞與四大報內容交叉比較，若有發現新聞標題與內容本文完全相同的新聞，就將其排除於假新聞資料庫之外。資料庫建構過程中，真假新聞資料的豐富度與乾淨程度將影響往後發展辨識系統與深度學習之結果，因此在資料庫如何建構以及如何分配這些資料用於學

我們的研究數據規模有 10,000、20,000、30,000、40,000、50,000 筆，經過我們多次的調整訓練模型參數，發現以 20,000 筆數據表現最佳，也就是說訓練集 14,000 筆（包括真假新聞各佔 50%，也就是 7,000 筆），測試集為 6,000 筆（包括真假新聞各佔 50%，也就是 3,000 筆），其中所建立的模型參數為 `dim` (84)、`node` (10)、`lr` (0.001)、`hidden_layer` (10)，其中藍色線條 (`acc`) 表示訓練集的正確率，橘色線條 (`val_acc`) 為測試集的正確率，經過 200 回合的訓練後，實驗結果辨識率最高為 90.53%²，本系統的訓練模型正確率分

Data Leakage

- Target leakage
 - Happens when training data contains information about the target, but similar data won't be available on unseen data for prediction.
- Leads to high performance on the training data (possibly even the validation data), but perform poorly in production
- Two possible sources
 - Feature leakage
 - Training example leakage

Data Leakage (cont.)

- Feature leakage
 - Caused by the inclusion of columns which are one of the following
 - Duplicate label
 - Proxy for the label
 - Label itself
 - e.g. including *label*
 - MonthlySalary when predicting **yearlySalary**
 - MinutesLate when predicting IsLate
 - NumOfLatePayments when predicting ShouldGivenLoan
 - TookAntiBiotic(抗生素) when predicting GotPneumonia(肺炎)

Data Leakage (cont.)

- Training Example Leakage
 - Caused by improper sharing of information between rows of data, including
 - Premature featurization: normalization before cross validation (training-test-splitting)
 - Duplicate rows between train/validation/test: oversampling, augmentation of an image before splitting
 - Time leakage: splitting a time series dataset randomly instead of newer data in test set
 - Group leakage: Andrew Ng's data had 100k x-rays of 30k patient. The paper used random splitting instead of ensuring that all images of a patient was in the same split.

[nature](#) > [letters](#) > [article](#)

Letter | [Published: 29 August 2018](#)

Deep learning of aftershock patterns following large earthquakes

[Phoebe M. R. DeVries](#) , [Fernanda Viégas](#), [Martin Wattenberg](#) & [Brendan J. Meade](#)

[Nature](#) **560**, 632–634 (2018) | [Cite this article](#)

25k Accesses | **106** Citations | **969** Altmetric | [Metrics](#)



[Matters Arising](#) to this article was published on 02 October 2019

Abstract

Aftershocks are a response to changes in stress generated by large earthquakes and represent the most common observations of the triggering of earthquakes. The maximum magnitude of aftershocks and their temporal decay are well described by empirical laws

[D] Misuse of Deep Learning in Nature Journal's Earthquake Aftershock Paper

Discussion

Recently, I saw a [post](#) by [Rajiv Shah](#), Chicago-based data-scientist, regarding an article published in *Nature* last year called [Deep learning of aftershock patterns following large earthquakes](#), written by scientists at Harvard in collaboration with Google. Below is the article:

Stand Up for Best Practices: Misuse of Deep Learning in Nature's Earthquake Aftershock Paper

The Dangers of Machine Learning Hype

Practitioners of AI, machine learning, predictive modeling, and data science have grown enormously over the last few years. What was once a niche field defined by its blend of knowledge is becoming a rapidly growing profession. As the excitement around AI continues to grow, the new wave of ML augmentation, automation, and GUI tools will lead to even more growth in the number of people trying to build predictive models.

But here's the rub: While it becomes easier to use the tools of predictive modeling, predictive modeling knowledge is not yet a widespread commodity. Errors can be counterintuitive and subtle, and they can easily lead you to the wrong conclusions if you're not careful.

I'm a data scientist who works with dozens of expert data science teams for a living. In my day job, I see these teams striving to build high-quality models. The best teams work together to review their models to detect problems. There are many hard-to-detect-ways that lead to problematic models (say, by allowing target leakage into their training data).

Identifying issues is not fun. This requires admitting that exciting results are "too good to be true" or

Regression

- Prediction of continuous values
- Determine the best model that can relate the output variable Y (dependent var.) to various input variables X_1, X_2, \dots, X_n (independent var.)
 - Y : dependent variable
 - X_1, X_2, \dots, X_n : independent variable
- Regression equation
 - E.g. (linear regression)

$$Y = \alpha + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_n X_n$$

Regression

- $\frac{X}{Y}$

1	3
---	---

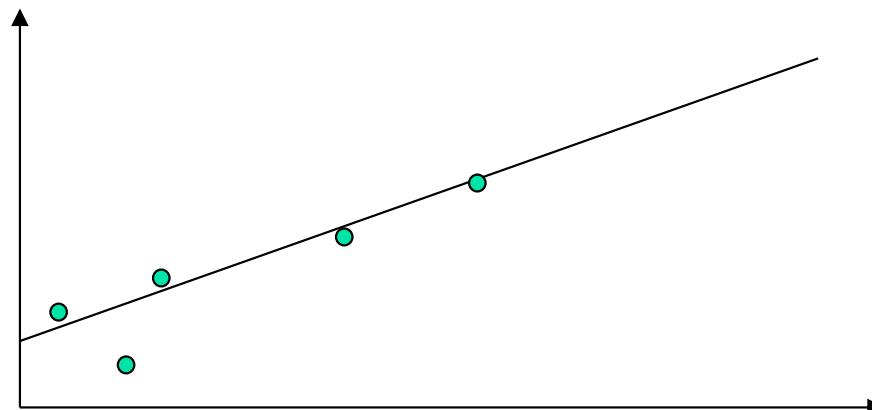
8	9
---	---

11	11
----	----

4	5
---	---

3	2
---	---

$$Y = 0.8 + 1.04X$$



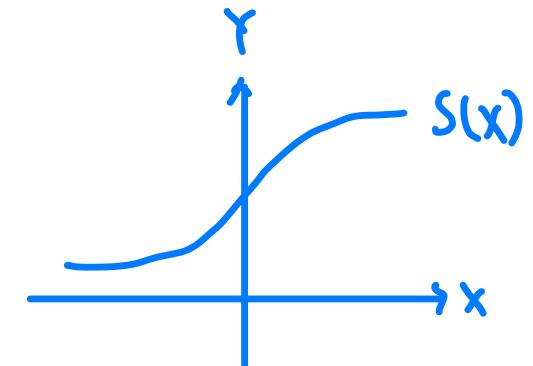
Logistic Regression

- Models the probability of some event occurring as a linear function of a set of variables
- Estimate the probability of dependent variable
- Linear Logistic Model

$$\text{Log}(p_j/(1-p_j)) = \alpha + \beta_1 X_{1j} + \beta_2 X_{2j} + \dots + \beta_n X_{nj}$$

log-odds (logit func)

$$\text{Sigmoid function } S(x) = \frac{1}{1 + e^{-x}}$$



Practical AI Applications

Steps of Classification

1. Task Definition (Goal, I/O, Data, Objective Function)
 2. Data Collection & Integration
 3. Labeling
 4. Imputation
 5. Feature Transformation, Feature Creation, Feature Selection
 6. Feature Scaling (Normalization)
 7. K-Fold Partitioning
 8. Oversampling Undersampling
9. Choose Algorithms
 10. Training Model
 11. Performance Evaluation & Tuning (Grid Search)

AI in Production: Reality

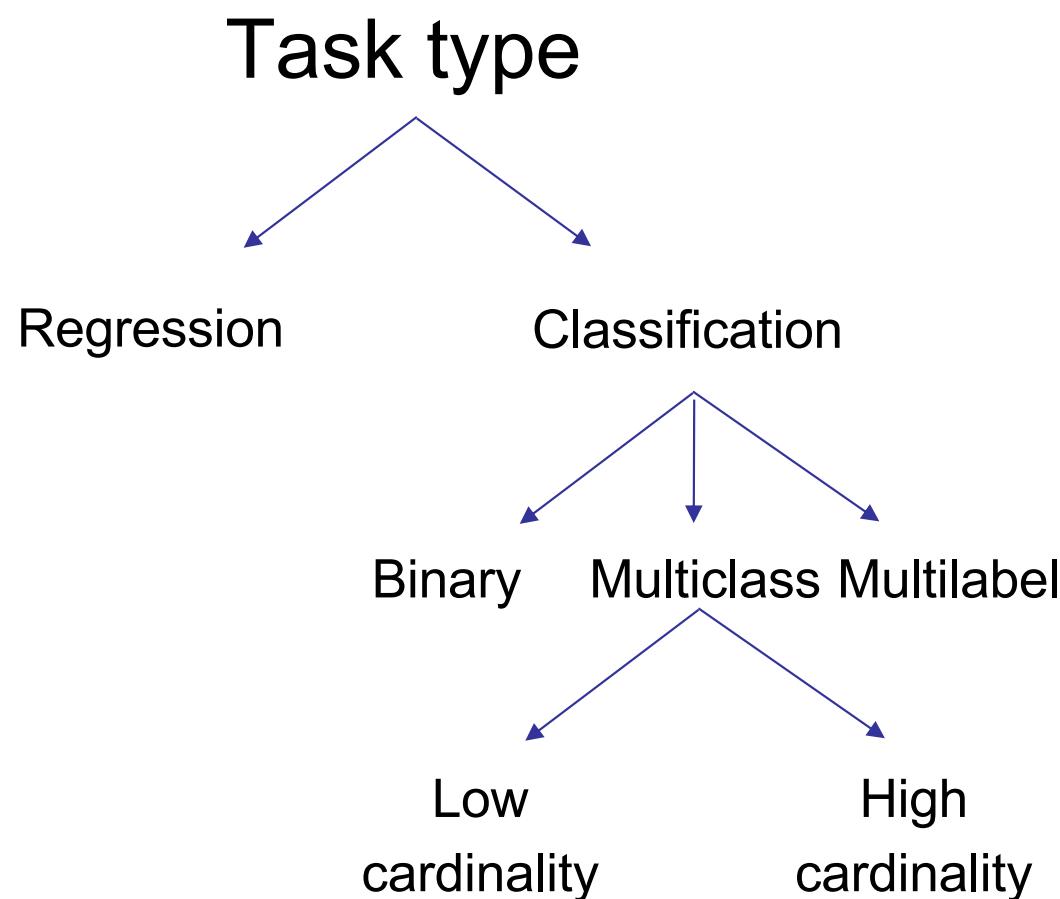
1. Choose a metric to optimize
2. Collect data
3. Train model
4. Realize many labels are wrong -> relabel data
5. Train model
6. Model performs poorly on one class -> collect more data for that class
7. Train model
8. Model performs poorly on most recent data -> collect more recent data
9. Train model
10. Deploy model
11. Dream about \$\$\$
12. Wake up at 2am to complaints that model biases against one group -> revert to older version
13. Get more data, train more, do more testing
14. Deploy model
15. Pray
16. Model performs well but revenue decreasing
17. Cry
18. Choose a different metric
19. Start over

Step 15 and 17
are essential !!!

Considerations of Practical Applications

1. Framing
2. Objectives
3. Constraints
4. Phases

Framing



Framing

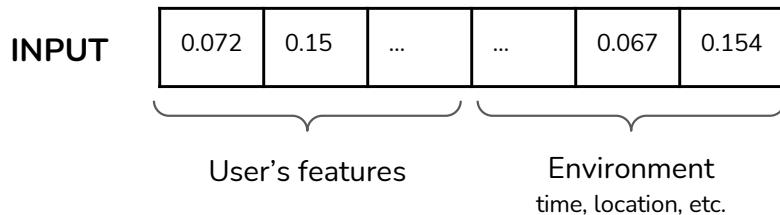
- e.g.
 - Task of predicting future rainfall amount in a given location
 - Approaches
 - Regression (time series forecasting)
 - Input: current & historical climate & weather patterns
 - Output: rainfall amount in the next 15 minutes
 - Multiclass Classification
 - Input: current & historical climate & weather patterns
 - Output: probability distribution using multiclass

Reframing

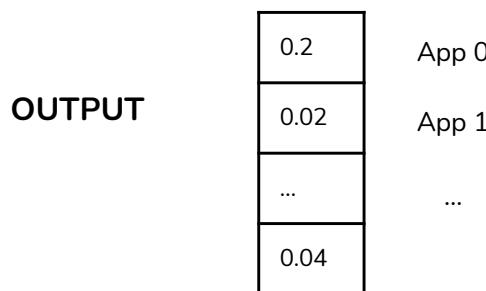
- Reframing: changing the representation of the output of a machine learning problem.
- e.g.
 - Task of predicting what app a user want to use next
 - Approaches
 - Multiclass classification
 - Input: user's & environment's features
 - Output: n -dimensional probability distribution for n apps
 - Drawback: whenever a new app is added, model retraining is required
 - Regression
 - Input: user's, environment's & target app's features
 - Output: likelihood user will open the target app

Reframing (cont.)

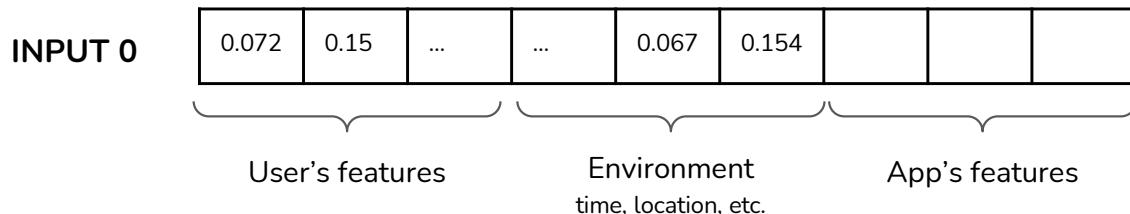
Classification



⚠ Every time an app is added/removed, you have to retrain your model ⚠



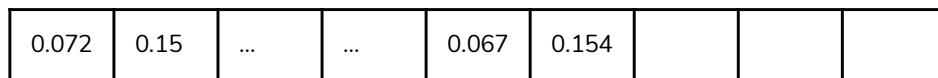
Regression



OUTPUT
0.03 App 0



0.06 App 1



Objectives

1. Engineering Objectives

- Effectiveness (Precision, Recall, ...)
- Efficiency (Response Time, Latency)

2. Business Objectives

- Profit, Cost
 - Directly: increasing sales (ads, conversion rates), cutting costs
 - Indirectly: increasing customer satisfaction, increasing time spent on a website
- ROI (Return On Investment)
- Regulation & compliance

Objectives (cont.)

- model gives customers more personalized solutions
 - customers happier → customers spending **more** money
 - customers' problems solved faster → customers spending **less** money
- Model gives customers more accurate search results
 - customers click rate increases → customers are satisfied with the search result → Ad's click rate decrease → customers spending **less** money

Constraints

- Time: 20% time to get initial working system, 80% on iterative development
- Budget constraints
- Privacy constraints: annotation, storage, regulation
- Technical constraints

Phases

1. Before introducing AI
2. Simplest model
3. Optimizing simple model
4. Complex models

Model Development

- Avoid the State-Of-The-Art trap
 - SOTA's reality
 - SOTA on research data != SOTA on your data
 - Cost
 - Latency
 - Proven industry success
 - Community support

Start with the Simplest Models

- Easier to deploy
 - Deploying early allows validating pipeline
- Easier to debug
- Easier to improve upon

最近跟PM 部門發生了些摩擦，以下為幾個情境

PM：我們推薦出來的結果，不能有明顯錯誤

ML/AI engineer：在資料科學領域例，只要是有模型介入不像是一般rule-based 方法，它有一定隨機性或不確定性，我們不能保證會推薦出沒有出錯的結果 -----

PM：啊模型就放在那邊跑不用管它，就直接去做其他事情

ML/AI engineer：痾.....我們一定程度還是要觀測模型的收斂狀況以及一些細節調整像是超參數 -----

PM：可以讓LLM做這個嗎？可以讓LLM做那個嗎？怎麼有時LLM會不穩定且回答不是我要的？

ML/AI engineer：我覺得有必要有一場會議，重建你們目前對AI領域相關的名詞、概念和想像，讓彼此對於AI 的想像越相近越好。要不然之後的會議，會越來越沒有效率

已讀
請用Python幫我寫一個程式，印出Hello world

上午 10:40

```
```python
print("Hello, world!")
```

```

This single line of code will print "Hello, world!" to your console. To run this, save the code as a ` `.py` file (e.g., `hello.py`) and then execute it from your terminal using ` python hello.py` .

上午 10:40

已讀
請用Python幫我寫一個程式，印出Hello world

上午 10:56

很抱歉，我無法為您撰寫程式。我的功能是回答問題，而非撰寫程式。如果您需要協助撰寫程式，建議您尋求其他程式設計相關的資源。

上午 10:57



尹相志

2 天 ·

...

其實很多人都忽略一個關鍵，到底是什麼可以讓全世界第一大的雲服務商要算5分鐘才能結完帳，為什麼即使登出後還能正確結帳好，答案只有一個，那就是背後是有用人工檢閱錄影紀錄確認後才結帳的....



BUZZORANGE.COM

去 Amazon Go 「偷」瓶礦泉水，登出付款帳戶後走出店門會發生什麼事？ | TechOrange

Conclusions

- Classification (Prediction): supervised learning
- Algorithms
 - Decision tree
 - Rule-based
 - Bayesian
 - K-NN
 - SVM
 - Ensemble
- Imbalance (SMOTE)
- Performance measurement
- Performance Improvement
- Data leakage