Bayesian Models

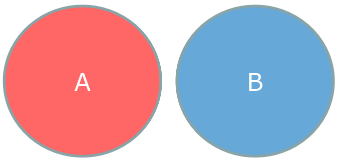
Payam Barnaghi
Department of Brain Sciences &
School of Convergence Science (Human and Artificial Intelligence)
Imperial College London

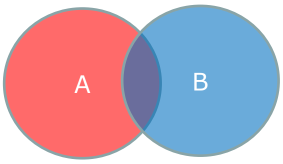
January 2026

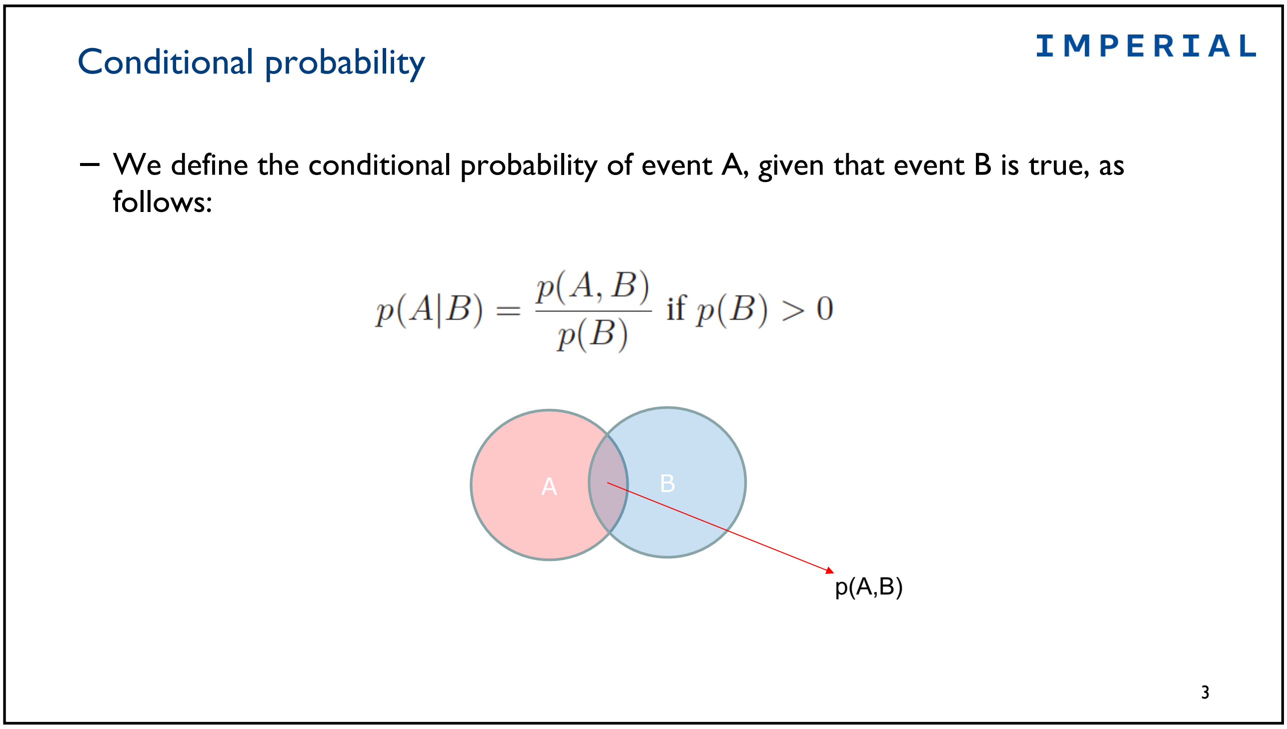
Probability of a union of two events

- Given two events, A and B, we define the probability of A or B as follows:

$$\begin{aligned} p(A \vee B) &= p(A) + p(B) - p(A \wedge B) \\ &= p(A) + p(B) \text{ if } A \text{ and } B \text{ are mutually exclusive} \end{aligned}$$







Conditional probability

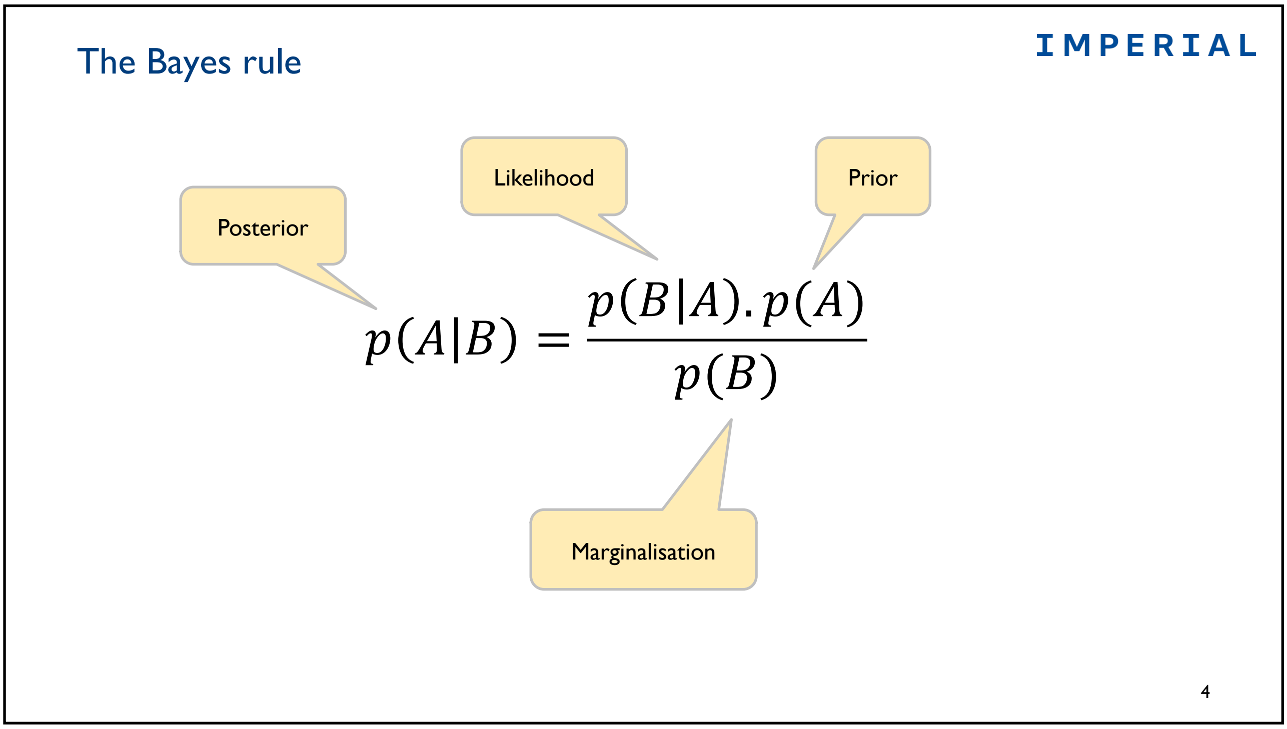
IMPERIAL

- We define the conditional probability of event A, given that event B is true, as follows:

$$p(A|B) = \frac{p(A,B)}{p(B)} \text{ if } p(B) > 0$$

3

3



The Bayes rule

IMPERIAL

$$p(A|B) = \frac{p(B|A).p(A)}{p(B)}$$

4

4

Bayes theorem*

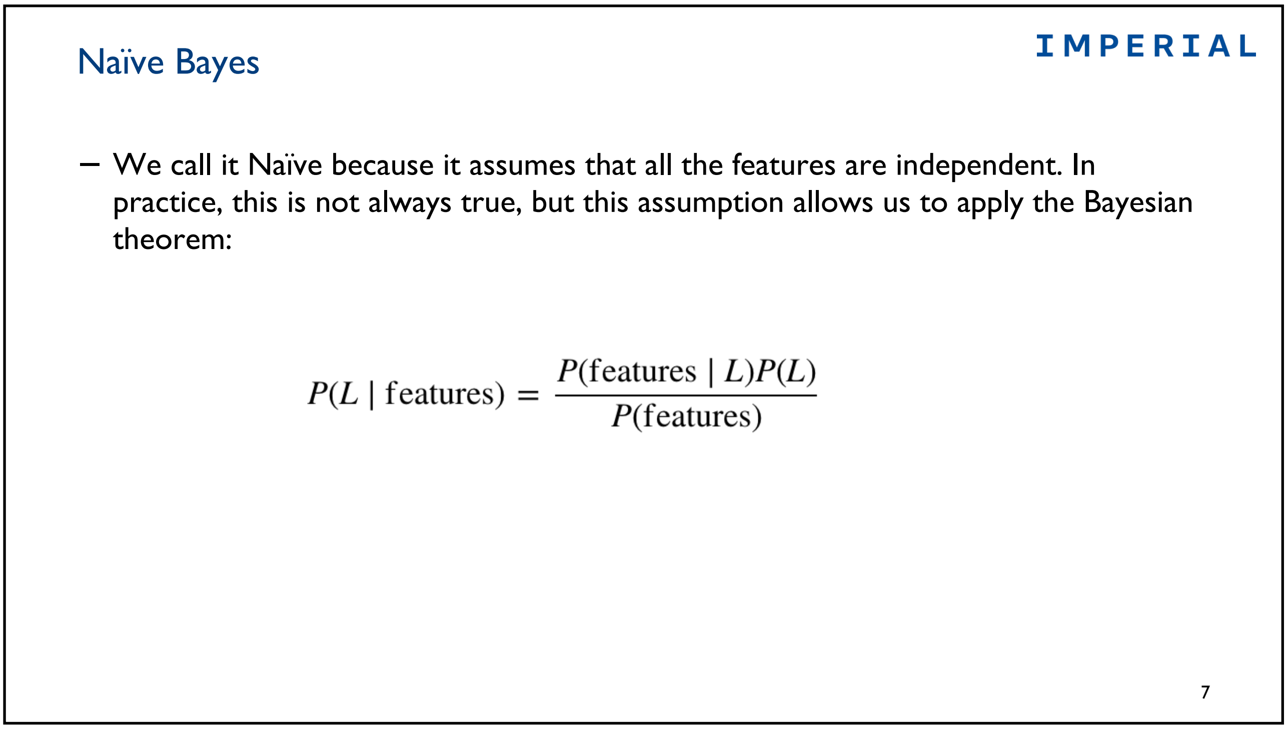
- Combining the definition of conditional probability with the product and sum rules yields the Bayes rule, also called the Bayes theorem:

$$p(X = x|Y = y) = \frac{p(X = x, Y = y)}{p(Y = y)} = \frac{p(X = x)p(Y = y|X = x)}{\sum_{x'} p(X = x')p(Y = y|X = x')}$$

Bayesian Classification

- Naive Bayes classifiers are built on Bayesian classification methods.
- These rely on Bayes's theorem, which describes the relationship of conditional probabilities of statistical quantities.
- In Bayesian classification, we are interested in finding the probability of a label given some observed features, which we can write as
  P(*label* | *features*).

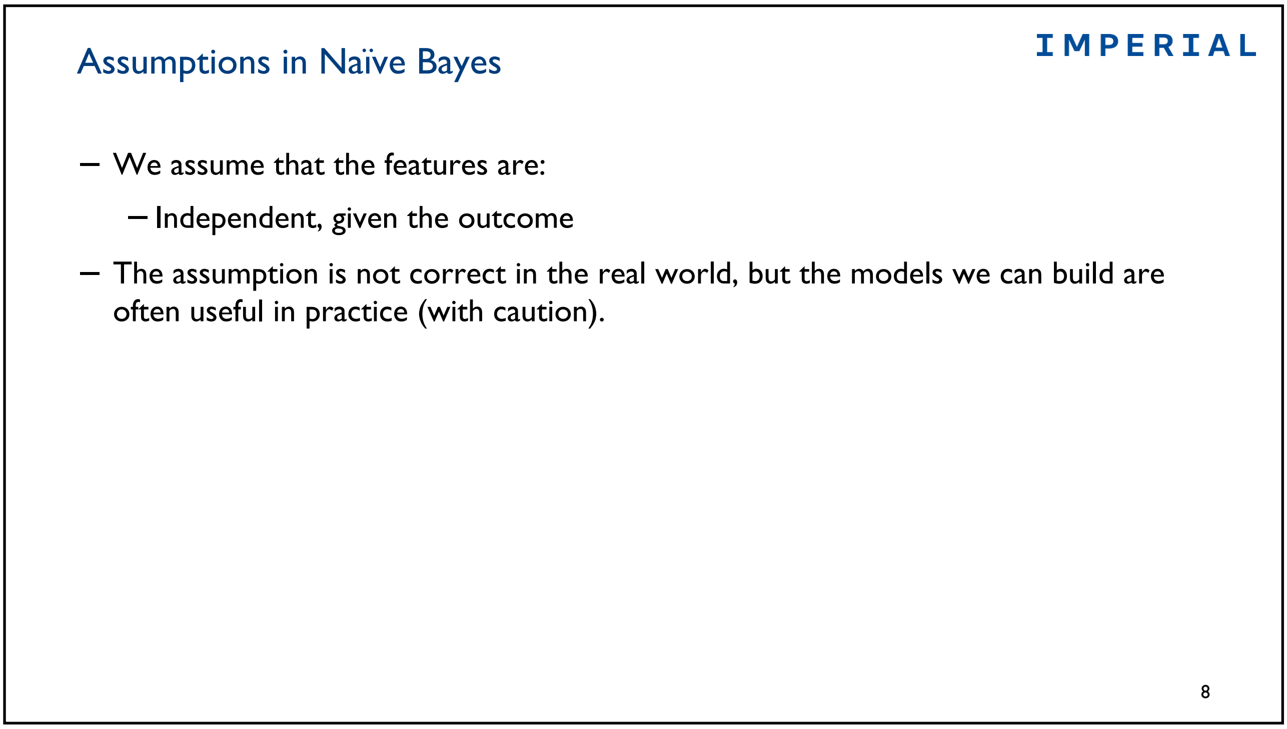
Source: Python Data Science Handbook by Jake VanderPlas, O'Reilly Media, 2016.



Naïve Bayes

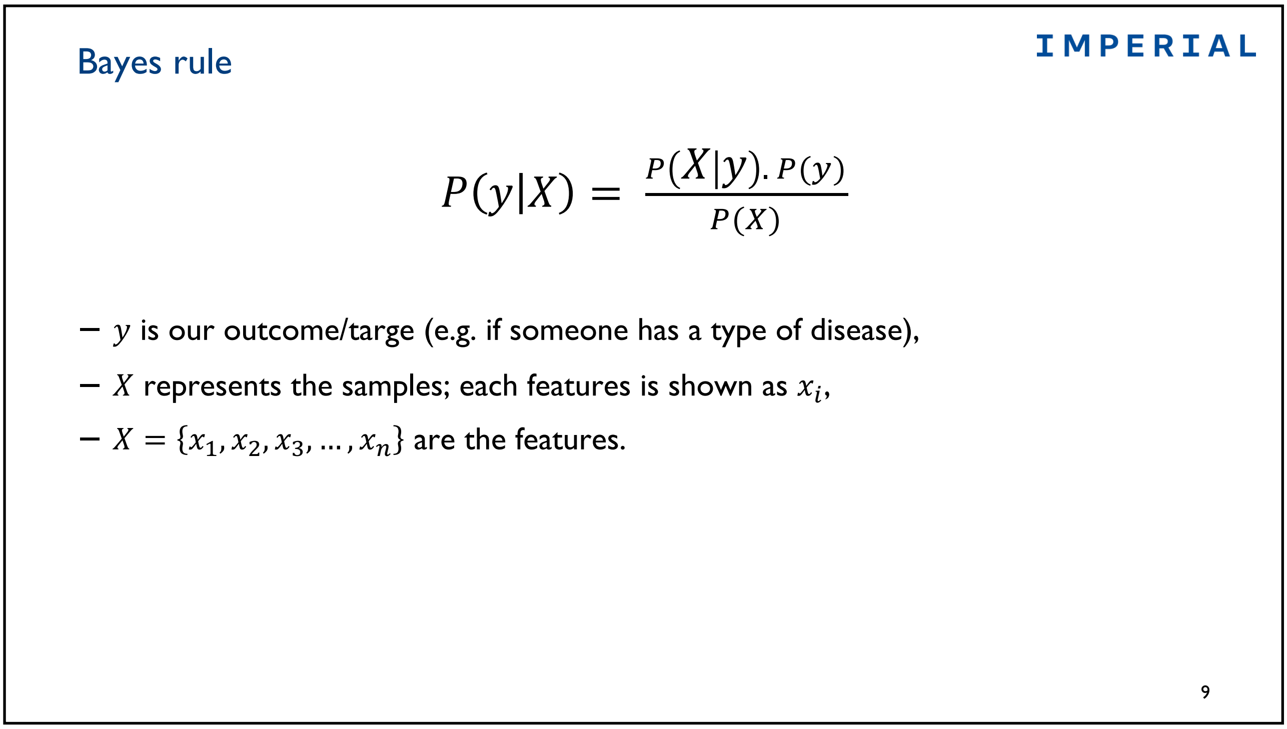
- We call it Naïve because it assumes that all the features are independent. In practice, this is not always true, but this assumption allows us to apply the Bayesian theorem:

$$P(L \mid \text{features}) = \frac{P(\text{features} \mid L)P(L)}{P(\text{features})}$$



Assumptions in Naïve Bayes

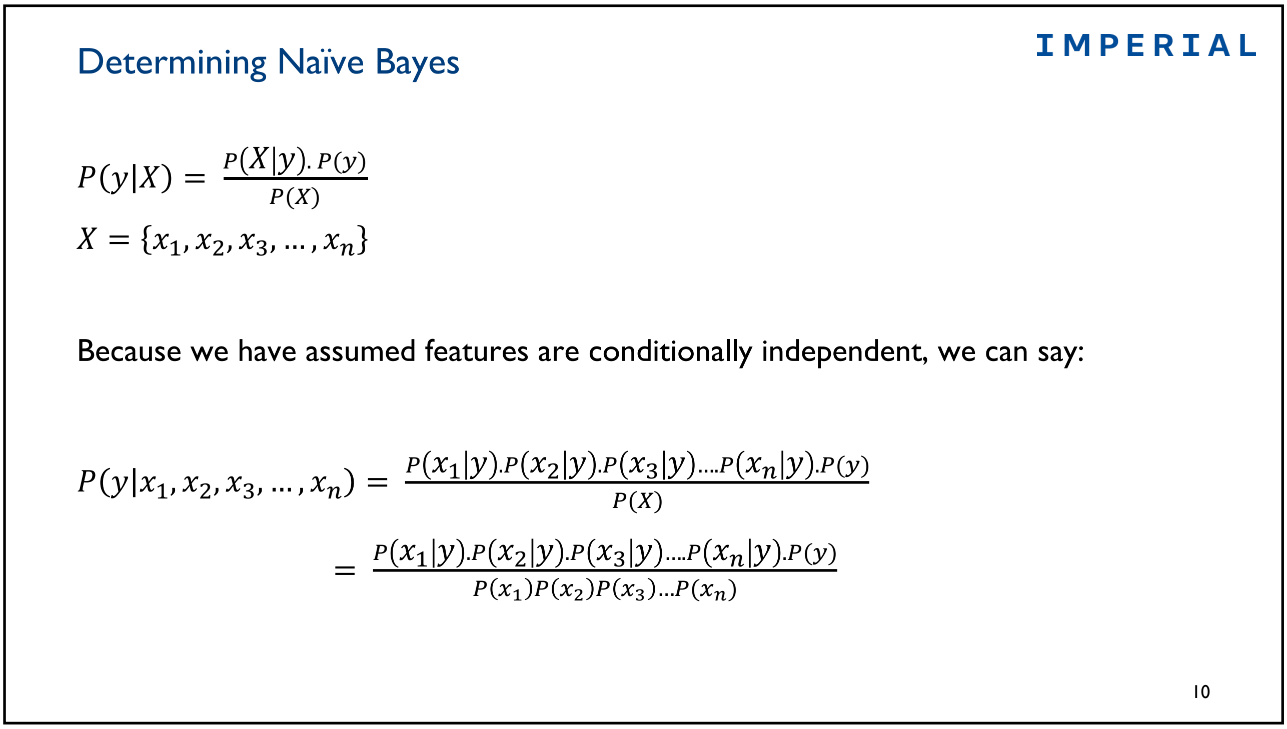
- We assume that the features are:
  - Independent, given the outcome
- The assumption is not correct in the real world, but the models we can build are often useful in practice (with caution).



Bayes rule

$$P(y|X) = \frac{P(X|y).\,P(y)}{P(X)}$$

- y is our outcome/targe (e.g. if someone has a type of disease),
- X represents the samples; each features is shown as x_i,
- $X = \{x_1, x_2, x_3, \dots, x_n\}$ are the features.



Determining Naïve Bayes

$$P(y|X) = \frac{P(X|y).\,P(y)}{P(X)}$$

$$X = \{x_1, x_2, x_3, \dots, x_n\}$$

Because we have assumed features are conditionally independent, we can say:

$$P(y|x_1, x_2, x_3, \dots, x_n) = \frac{P(x_1|y).P(x_2|y).P(x_3|y)....P(x_n|y).P(y)}{P(X)}$$

$$= \frac{P(x_1|y).P(x_2|y).P(x_3|y)....P(x_n|y).P(y)}{P(x_1)P(x_2)P(x_3)...P(x_n)}$$

Let’s simplify

IMPERIAL

$$p(y, \mathbf{x}) = p(y) \prod_{j=1}^{D} p(x_j|y)$$

This represents the **joint probability** of a class label y and a feature vector $\mathbf{x} = (x_1, x_2, \ldots, x_D)$ under the **Naïve Bayes** assumption.

- p(y,x): Joint probability of the label y and all features $\mathbf{x}$.

– $p(y)$: Prior probability of the class y.

- $\prod_{j=1}^{D} p\,(x_j \mid y)$: Product of conditional probabilities of each feature given the class.

11

11

Example

IMPERIAL

- Imagine predicting whether a patient has Alzheimer’s (y) based on features below:
  - x_1: Age
  - x_2: APOE-4 status
  - x_3: CSF tau level
  - x_4: MRI hippocampal volume
- Under Naïve Bayes:
  - $p(y, \mathbf{x}) = p(y) \cdot p(Age \mid y) \cdot p(APOE \mid y) \cdot p(Tau \mid y) \cdot p(MRI \mid y)$
  - p(y): Prior probability of the class y

12

12

Naïve Bayes: example

IMPERIAL

P (disease =1) given [0,1]:

$$P(Disease = 1 \mid Feature1 = 0, Feature2 = 1) = \frac{P(Feature1 = 0 \mid Disease = 1) \cdot P(Feature2 = 1 \mid Disease = 1) \cdot P(Disease = 1)}{P(Feature1 = 0, Feature2 = 1)}$$

	High BP	TBI	Disease
Patient #01	1	0	0
Patient #02	1	1	1
Patient #03	1	0	0
Patient #04	0	0	0
Patient #05	1	0	1
Patient #06	0	0	1
Patient #07	0	0	0
Patient #08	0	1	1
Patient #09	0	1	0
Patient #10	1	0	0

TBI: History of Traumatic Brain Injury; BP: Bloop Pressure

Algorithm for Naïve Bayes (with binary features)

IMPERIAL

```
$N_c = 0$, $N_{jc} = 0$;
for $i = 1 : N$ do
    $c = y_i$ // Class label of $i$'th example;
    $N_c := N_c + 1$ ;
    for $j = 1 : D$ do
        if $x_{ij} = 1$ then
            $N_{jc} := N_{jc} + 1$
$\hat{\pi}_c = \frac{N_c}{N}$, $\hat{\theta}_{jc} = \frac{N_{jc}}{N}$
```

What about continuous variables?

- For real-valued features, we typically model them using a Gaussian (normal) distribution:

$$p(\mathbf{x}|y = c, \boldsymbol{\theta}) = \prod_{j=1}^{D} \mathcal{N}(x_j|\mu_{jc}, \sigma_{jc}^2)$$

where μ_{jc} is the mean of feature j in objects of class c, and σ_{jc}^2 is its variance.

Bayesian decision methods

- Suppose we have an input vector x with a corresponding vector y of target variables.

- Our goal is to predict y given a new value for x.
- For a regression problem, y will contain continuous (real-valued) variables, and for a classification problem, y will represent class labels.

Joint probability

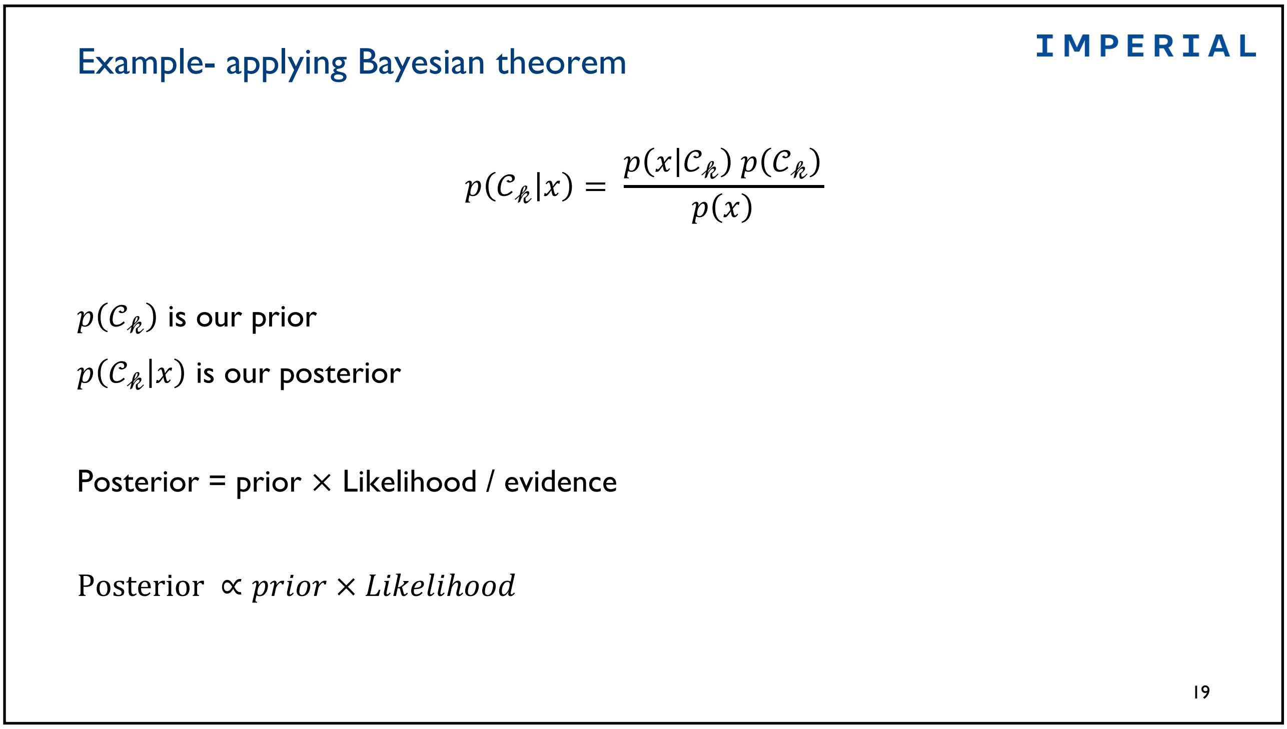
IMPERIAL

- The joint probability distribution p(x,y) provides a summary of the uncertainty associated with these variables.
- Determining p(x,y) from a set of training data is also referred to as *inference*.
- Consider a medical diagnosis example: We have performed an MRI scan of a patient to determine whether the patient has a specific neurological disorder.

Medical diagnosis example

IMPERIAL

- In the previous example, the input vector x is the set of pixels (or engineered features) from the scan.
- y will represent the presence of the disease; for the presence of the disease, we show it, for example, with C_1, and for the absence of the disease, we note it as $C_{2.}$
- The general inference problem then involves determining the join distribution p(x, C_K)



Example- applying Bayesian theorem

IMPERIAL

$$p(\mathcal{C}_k|x) = \frac{p(x|\mathcal{C}_k)\,p(\mathcal{C}_k)}{p(x)}$$

$p(\mathcal{C}_k)$ is our prior

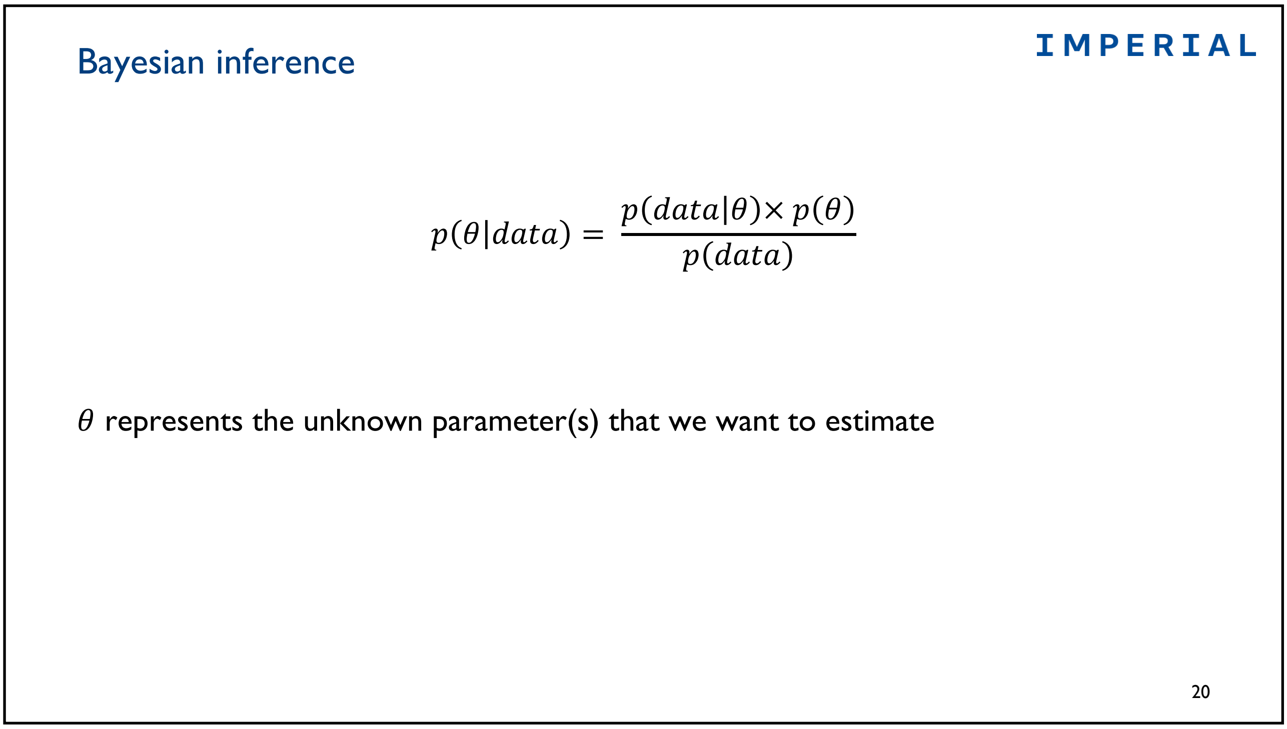
$p(\mathcal{C}_k|x)$ is our posterior

Posterior = prior × Likelihood / evidence

$\text{Posterior} \;\propto prior \times Likelihood$

19

19



Bayesian inference

IMPERIAL

$$p(\theta|data) = \frac{p(data|\theta)\times p(\theta)}{p(data)}$$

θ represents the unknown parameter(s) that we want to estimate

20

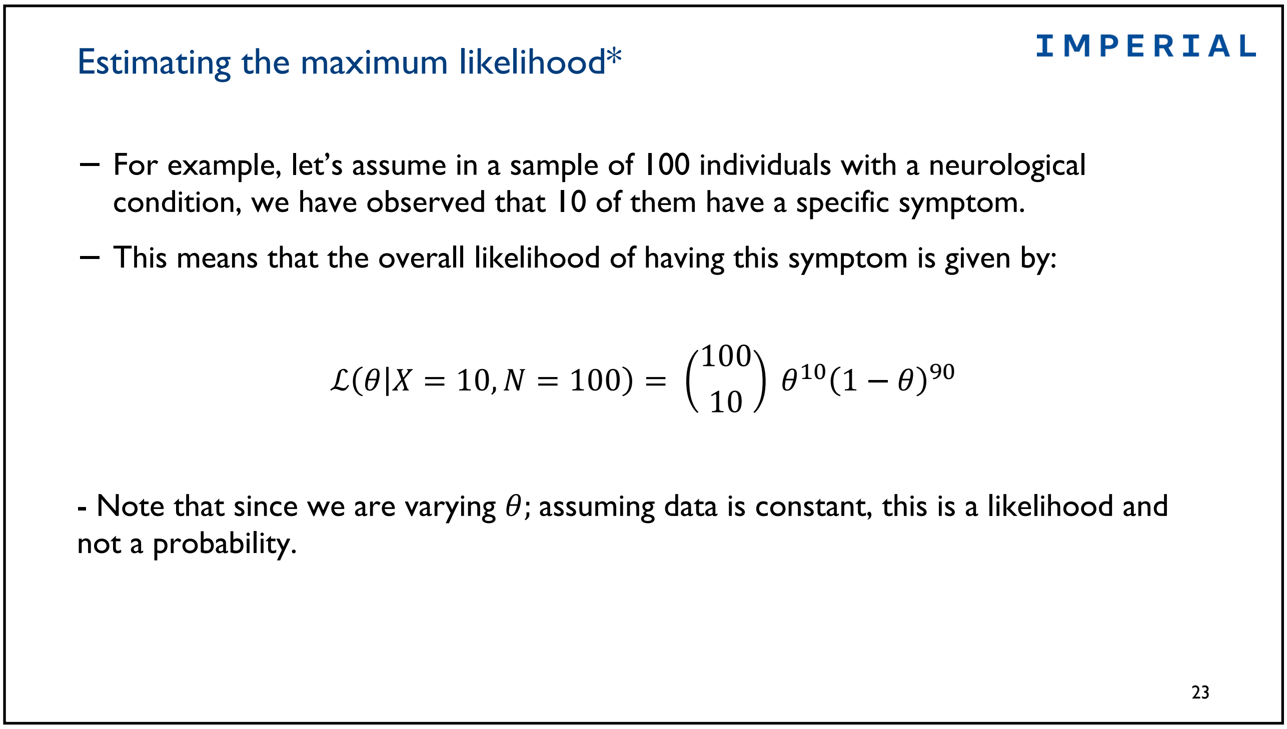
20

Likelihood

- $p(data|\theta)$ is called likelihood.
- This means the probability of generating a particular sample if the parameter of our model was equal to θ.
- For example, if our data includes samples from a population with features that indicate if they had high blood pressure and if they had a history of TBI (X) associated with the target variable as having Alzheimer's disease or not (y):
  - Then the likelihood would present, if someone had Alzheimer's disease, what is the probability that they had high blood pressure and/or a history of TBI (depending on what likelihood we are interested in).

Maximum likelihood*

- In principle, we define the likelihood function and then calculate the parameter values that maximise the likelihood of observing our data.
- In other words, in the previous example, p(data | θ), we assumed we knew the parameter θ, but what if we used a method to identify the value(s) of θ that maximise the likelihood of obtaining the data?
- For example, in the case of a disease, we may not know its prevalence in the population in advance.



Estimating the maximum likelihood*

IMPERIAL

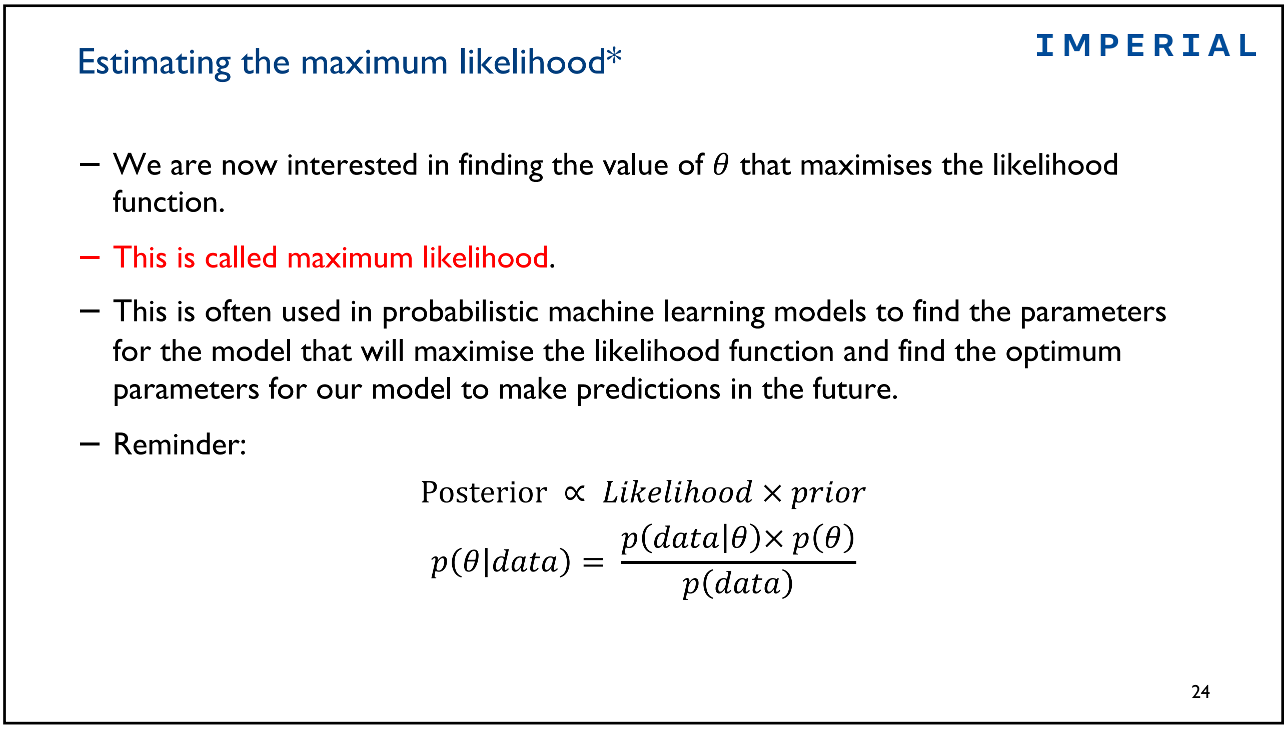
- For example, let’s assume in a sample of 100 individuals with a neurological condition, we have observed that 10 of them have a specific symptom.
- This means that the overall likelihood of having this symptom is given by:

$$\mathcal{L}(\theta|X = 10, N = 100) = \binom{100}{10}\,\theta^{10}(1-\theta)^{90}$$

- Note that since we are varying θ; assuming data is constant, this is a likelihood and not a probability.

23

23



Estimating the maximum likelihood*

IMPERIAL

- We are now interested in finding the value of θ that maximises the likelihood function.
- This is called maximum likelihood.
- This is often used in probabilistic machine learning models to find the parameters for the model that will maximise the likelihood function and find the optimum parameters for our model to make predictions in the future.
- Reminder:

$$\text{Posterior} \,\propto\, Likelihood \times prior$$

$$p(\theta|data) = \frac{p(data|\theta)\times p(\theta)}{p(data)}$$

24

24

Repeating our example*

- For example, let’s assume in a sample of 100 individuals with a neurological condition, we have observed that 10 of them have a specific symptom.
- This means that the overall likelihood of having this symptom is given by:

$$\mathcal{L}(\theta|X = 10, N = 100) = \binom{100}{10}\,\theta^{10}(1-\theta)^{90}$$

- How to find what value for theta (θ) will maximise this?

Log likelihood*

- Before we continue with the next few slides:
- I don’t expect you to know or use the maths discussed here- this is just added for your information and for those who might be interested in pursuing Bayesian methods further.
- Most modern libraries will automatically train/learn your model and optimise its parameters, including the models we will use in our lab experiments.

Log likelihood*

IMPERIAL

$$\mathcal{L}(\theta|X = 10, N = 100) = \binom{100}{10}\,\theta^{10}(1-\theta)^{90}$$

- How to find what value for theta will maximise this?

- These three rules are going to help us:

$$\log(ab) = \log(a) + \log(b)$$
$$\log\left(a^b\right) = \mathrm{b}\log(a)$$
$$\frac{d}{dx}(\log_a x) = \frac{1}{x\ln a}$$

Maximising the Log likelihood*

IMPERIAL

- To maximise the likelihood, calculate the log-likelihood, take the derivative and then set it equal to zero.
- We calculate the log-likelihood for simplicity of working with sums (+) instead of multiplications (x) [needed for the step below]
- We take the first derivative and set it to zero because the first derivative will give us the slope of change;
- And setting it to zero will give you the point that the point that the function is maximised.

Why do we take the first derivative and set it to zero?

IMPERIAL

- Imagine the likelihood function as a hill. The top of the hill is where the likelihood is highest.
- That's the parameter value we want. To find the top, we look at the slope of the hill. When you're climbing up, the slope is positive; when you're going down, it's negative. At the very peak, the slope is flat, zero.
- The first derivative tells us the slope. So, by taking the derivative of the log-likelihood and setting it to zero, we're finding the point where the slope is flat, the maximum. This is why solving
  - $\frac{d\ell(\theta)}{d\theta} = 0$
  - gives us the Maximum Likelihood Estimate.

29

29

Maximising the Log likelihood*

IMPERIAL

$$\mathcal{L}(\theta|X = 10, N = 100) = \binom{100}{10}\,\theta^{10}(1-\theta)^{90}$$

$$\log(ab) = \log(a) + \log(b)$$

$$\log\left(a^{b}\right) = \mathrm{b}\log(a)$$

$$\frac{d}{dx}(\log_{a} x) = \frac{1}{x\ln a}$$

$$\log\mathcal{L}(\theta|X = 10, N = 100) = \log\binom{100}{10} + \log\theta^{10} + \log(1-\theta)^{90}$$

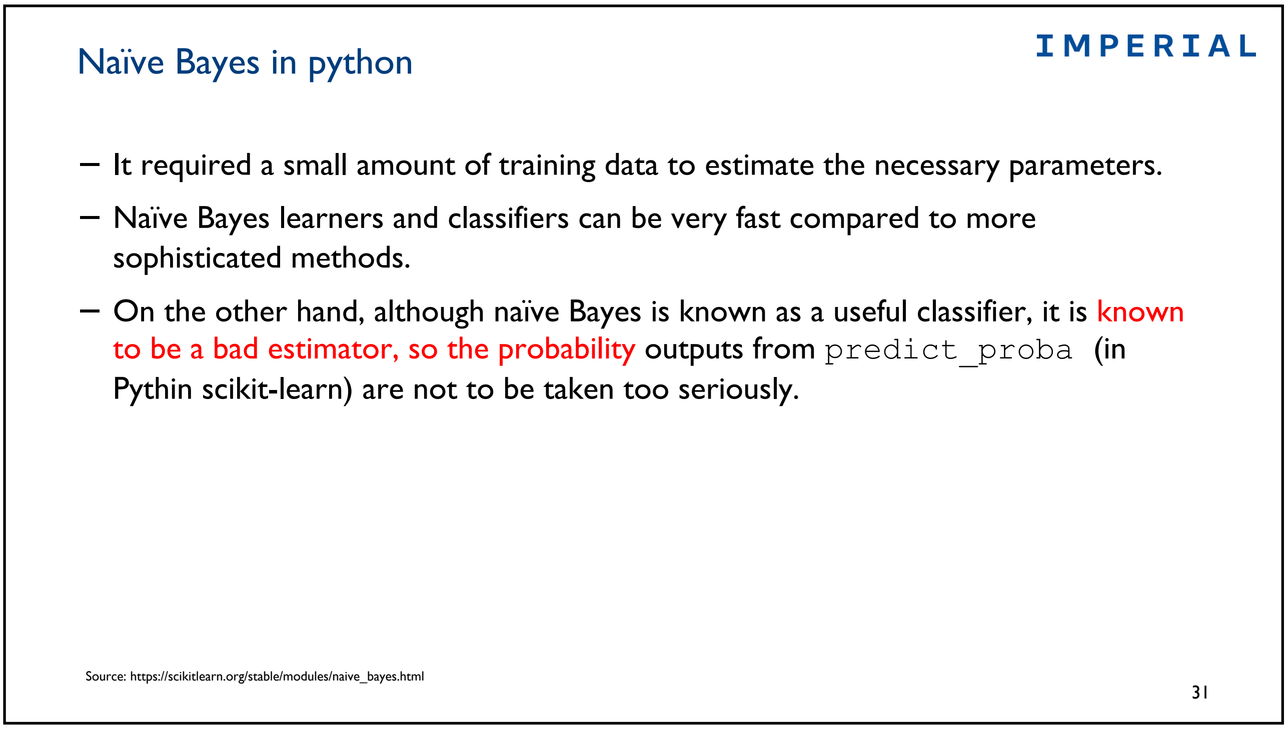
$$\frac{\partial l}{\partial \theta} = \mathrm{lna}\left(\frac{10}{\hat{\theta}} - \frac{90}{1-\hat{\theta}}\right) = 0$$

$$\hat{\theta} = 1/10$$

** We did not define any specific base for the logarithm as it is not important here; overall, $\mathrm{Log}_{b}x$ = ln(x)/ln(b)

30

30

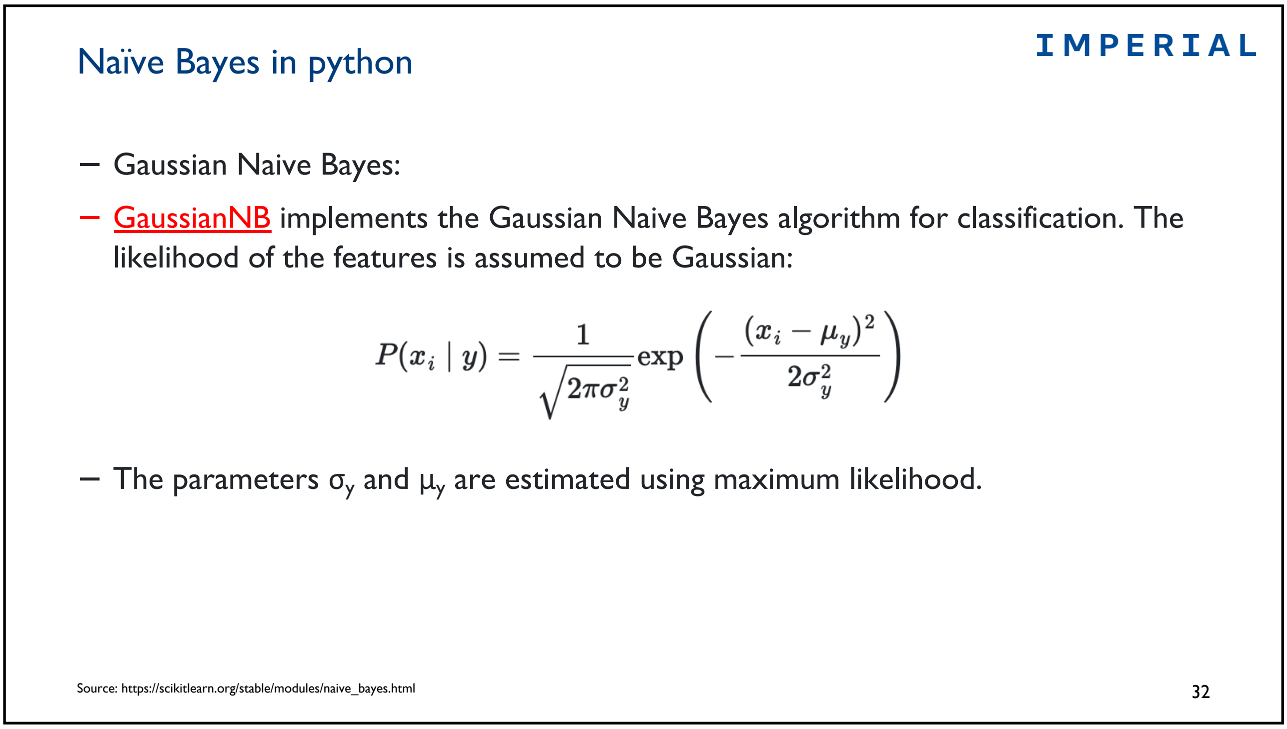


Naïve Bayes in python

IMPERIAL

- It required a small amount of training data to estimate the necessary parameters.
- Naïve Bayes learners and classifiers can be very fast compared to more sophisticated methods.
- On the other hand, although naïve Bayes is known as a useful classifier, it is known to be a bad estimator, so the probability outputs from `predict_proba` (in Pythin scikit-learn) are not to be taken too seriously.

Source: https://scikitlearn.org/stable/modules/naive_bayes.html



Naïve Bayes in python

IMPERIAL

- Gaussian Naive Bayes:
- GaussianNB implements the Gaussian Naive Bayes algorithm for classification. The likelihood of the features is assumed to be Gaussian:

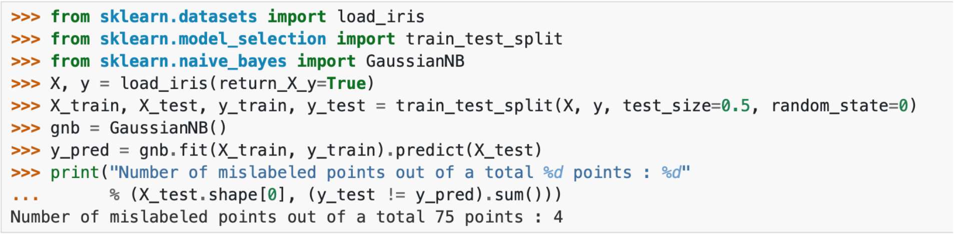
$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

- The parameters σ_y and μ_y are estimated using maximum likelihood.

Source: https://scikitlearn.org/stable/modules/naive_bayes.html

Gaussian Naïve Bayes

IMPERIAL



```
>>> from sklearn.datasets import load_iris
>>> from sklearn.model_selection import train_test_split
>>> from sklearn.naive_bayes import GaussianNB
>>> X, y = load_iris(return_X_y=True)
>>> X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5, random_state=0)
>>> gnb = GaussianNB()
>>> y_pred = gnb.fit(X_train, y_train).predict(X_test)
>>> print("Number of mislabeled points out of a total %d points : %d"
...       % (X_test.shape[0], (y_test != y_pred).sum()))
Number of mislabeled points out of a total 75 points : 4
```

Source: https://scikitlearn.org/stable/modules/naive_bayes.html

Multinomial Naïve Bayes

IMPERIAL

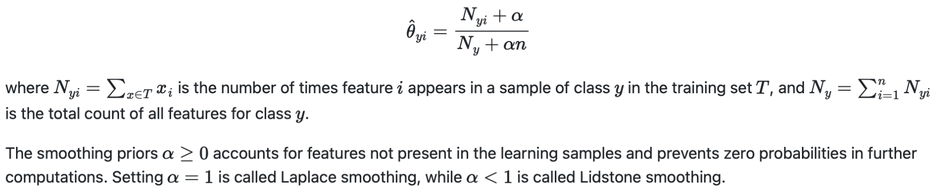
- MultinomialNB implements the naïve Bayes algorithm for multinomially distributed data and is one of the two classic naïve Bayes variants used in text classification.
- The distribution is parametrised by vectors $\theta y=(\theta_{y1},\ldots,\theta_{yn})$ for each class y, where n is the number of features and θyi is the probability $P(x_i|y)$ of feature i appearing in a sample belonging to class y.
- The parameters θy are estimated by a smoothed version of maximum likelihood, i.e. relative frequency counting.

Source: https://scikitlearn.org/stable/modules/naive_bayes.html

Multinomial Naïve Bayes*

IMPERIAL

– The parameters θy is estimated by a smoothed version of maximum likelihood, i.e. relative frequency counting:



$$\hat{\theta}_{yi} = \frac{N_{yi} + \alpha}{N_y + \alpha n}$$

where $N_{yi} = \sum_{x \in T} x_i$ is the number of times feature i appears in a sample of class y in the training set T, and $N_y = \sum_{i=1}^{n} N_{yi}$ is the total count of all features for class y.

The smoothing priors $\alpha \geq 0$ accounts for features not present in the learning samples and prevents zero probabilities in further computations. Setting $\alpha = 1$ is called Laplace smoothing, while $\alpha < 1$ is called Lidstone smoothing.

Source: https://scikitlearn.org/stable/modules/naive_bayes.html

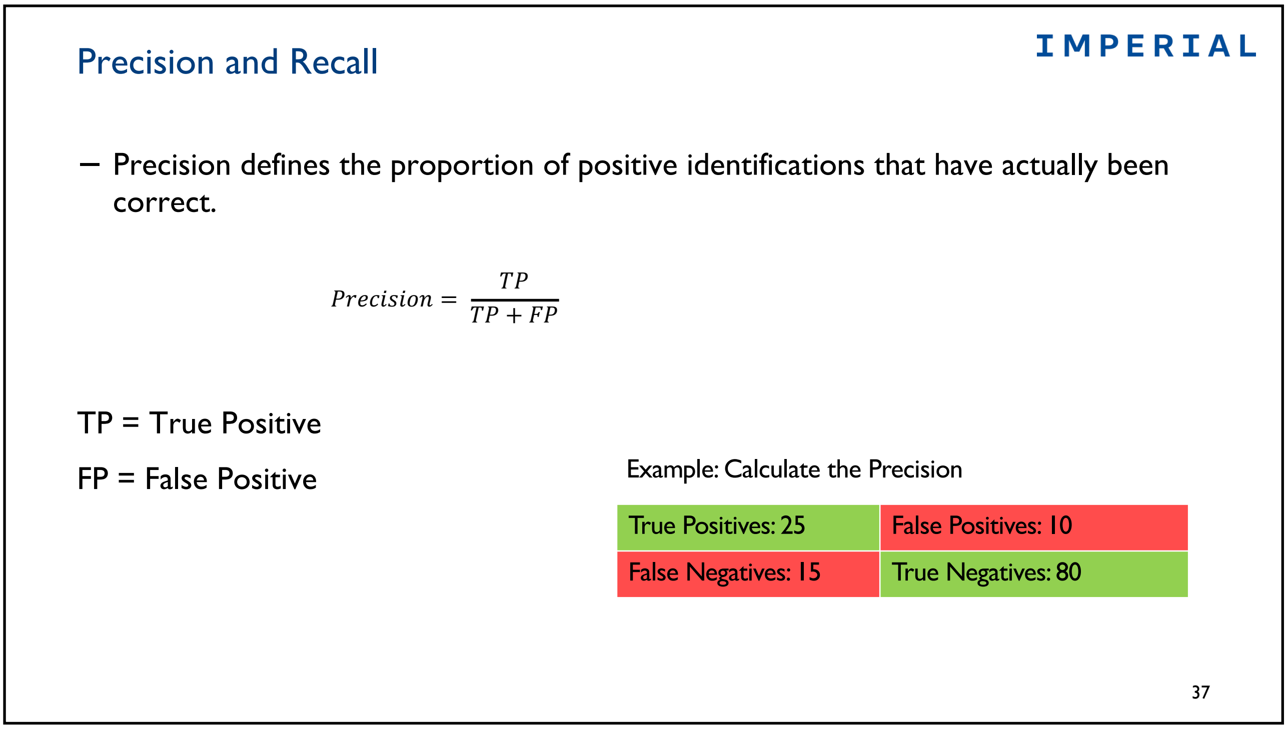
35

35

IMPERIAL

Evaluation metrics

36



Precision and Recall

IMPERIAL

– Precision defines the proportion of positive identifications that have actually been correct.

$$Precision = \frac{TP}{TP + FP}$$

TP = True Positive

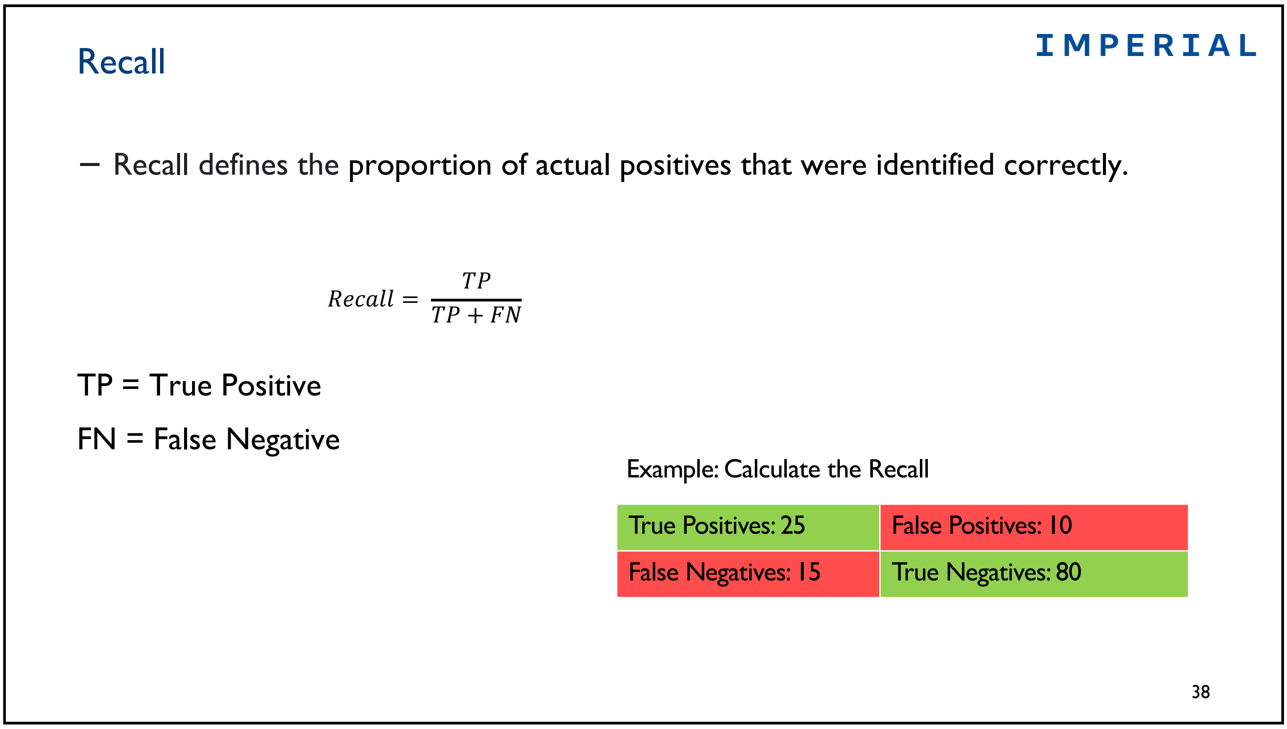
FP = False Positive

Example: Calculate the Precision

True Positives: 25	False Positives: 10
False Negatives: 15	True Negatives: 80

37

37



Recall

IMPERIAL

– Recall defines the proportion of actual positives that were identified correctly.

$$Recall = \frac{TP}{TP + FN}$$

TP = True Positive

FN = False Negative

Example: Calculate the Recall

True Positives: 25	False Positives: 10
False Negatives: 15	True Negatives: 80

38

38

Precision and Recall

IMPERIAL

How the decision threshold could affect Precision and Recall.

This can be very important in medical applications.

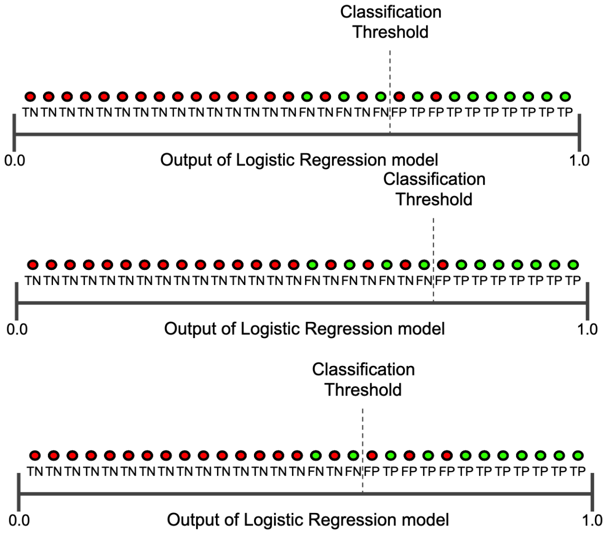

Classification
Threshold
TN TN TN TN TN TN TN TN TN TN TN TN TN TN TN FN TN FN TN FN FP TP FP TP TP TP TP TP TP TP
0.0
Output of Logistic Regression model
1.0
Classification
Threshold
TN TN TN TN TN TN TN TN TN TN TN TN TN TN TN FN TN FN TN FN TN FN FP TP TP TP TP TP TP TP
0.0
Output of Logistic Regression model
1.0
Classification
Threshold
TN TN TN TN TN TN TN TN TN TN TN TN TN TN TN FN TN FN FP TP FP TP FP TP TP TP TP TP TP TP
0.0
Output of Logistic Regression model
1.0

Image source: https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall

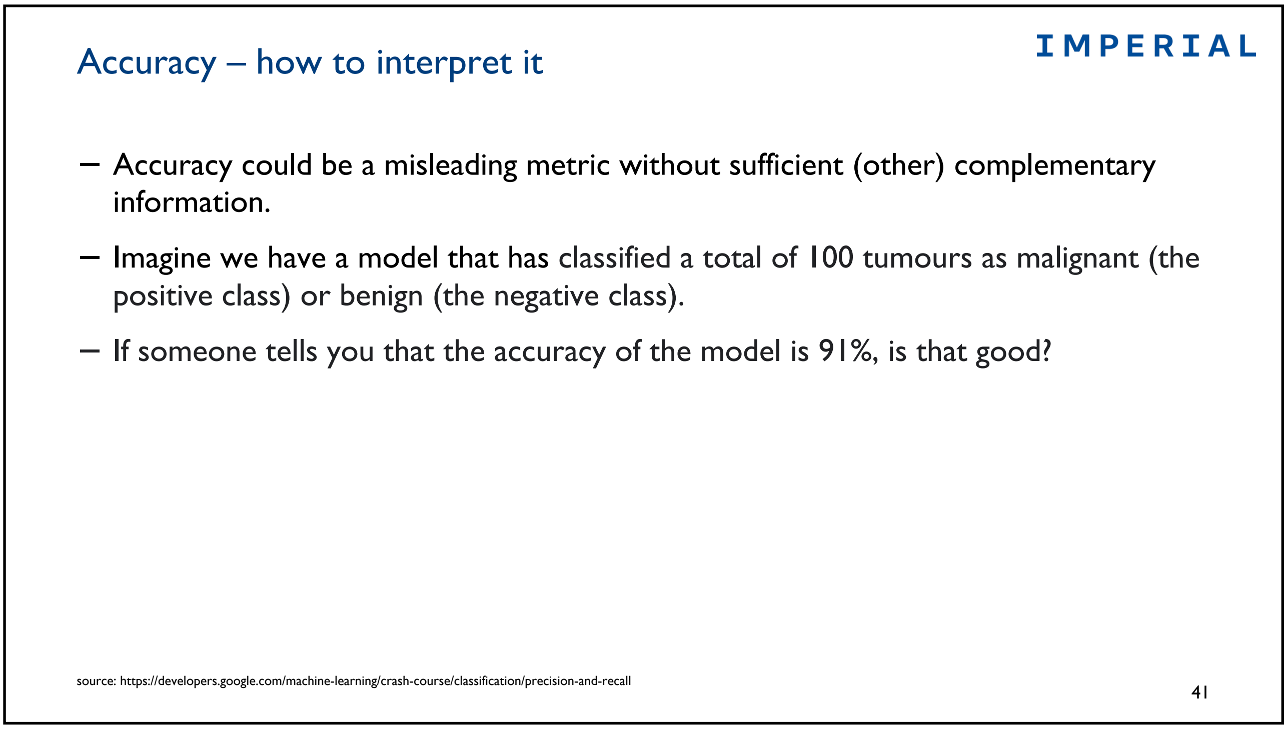
Accuracy

IMPERIAL

- Accuracy is a metric to evaluate classification models.
- It shows the fraction of predictions that the model has made correctly.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

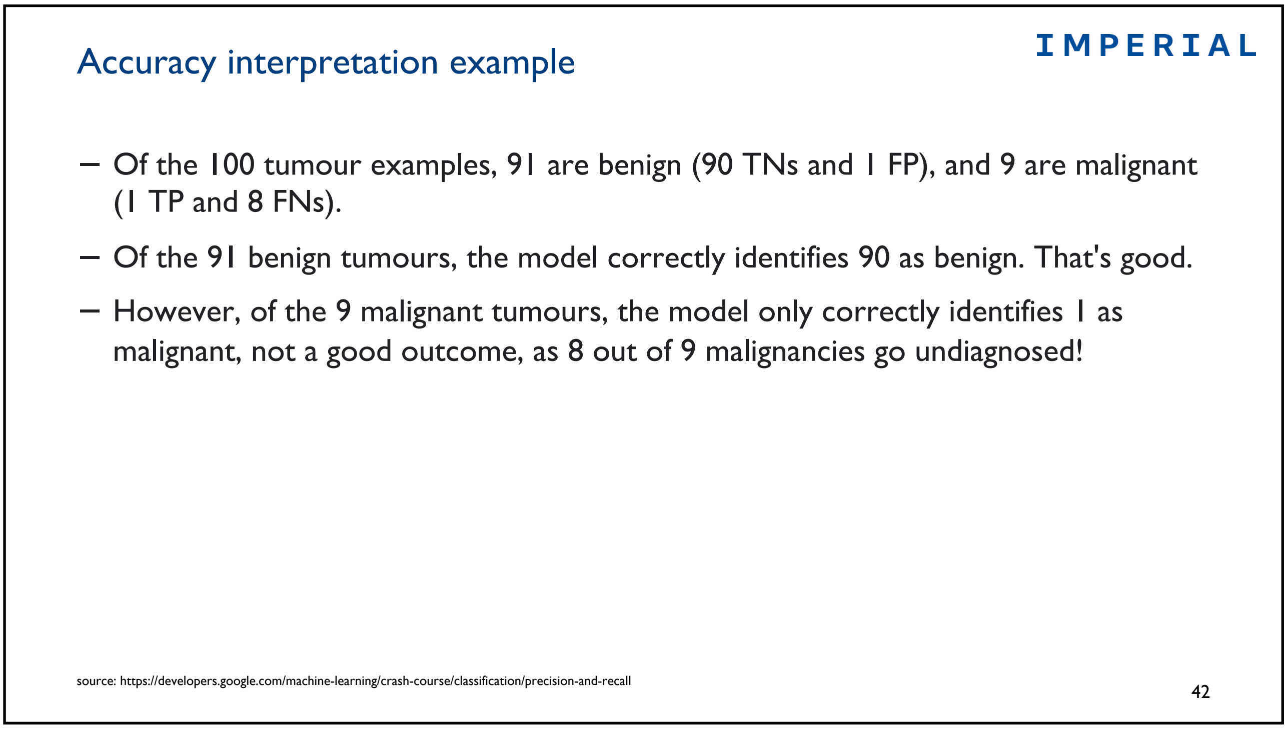
$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$



Accuracy – how to interpret it

- Accuracy could be a misleading metric without sufficient (other) complementary information.
- Imagine we have a model that has classified a total of 100 tumours as malignant (the positive class) or benign (the negative class).
- If someone tells you that the accuracy of the model is 91%, is that good?

source: https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall



Accuracy interpretation example

- Of the 100 tumour examples, 91 are benign (90 TNs and 1 FP), and 9 are malignant (1 TP and 8 FNs).
- Of the 91 benign tumours, the model correctly identifies 90 as benign. That's good.
- However, of the 9 malignant tumours, the model only correctly identifies 1 as malignant, not a good outcome, as 8 out of 9 malignancies go undiagnosed!

source: https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall

F1 Score

IMPERIAL

– F1 score is the harmonic mean of precision and recall.

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision\ + Recall}$$

Example: Calculate the F1 score

True Positives: 25	False Positives: 10
False Negatives: 15	True Negatives: 80

ROC curve (receiver operating characteristic curve)

IMPERIAL

– A ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

  – True Positive Rate (TPR)

  – False Positive Rate (FPR)

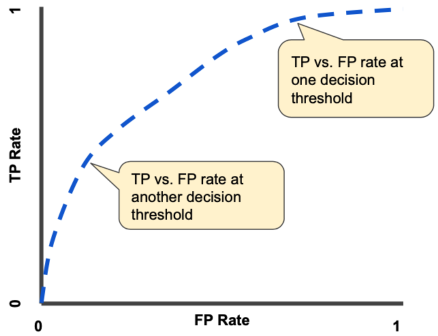
$$TPR = \frac{TP}{TP + FN}$$

Does this look familiar?

source: https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall

ROC curve

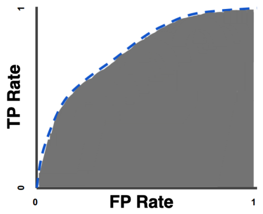
- A ROC curve plots TPR vs. FPR at different classification thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives.



source: https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall

AUC: Area Under the ROC Curve

- AUC stands for "Area under the Curve."
- In this case, AUC measures the entire two-dimensional area underneath the entire ROC curve.
- AUC provides an aggregate measure of performance across all possible classification thresholds.



- Please note that there is also an AUC for Precision-Recall Curve

source: https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall

PR AUC vs ROC AUC

- PR AUC:

- Evaluating performance based on positive class
- Better suited for imbalanced data,
- Precision vs. Recall

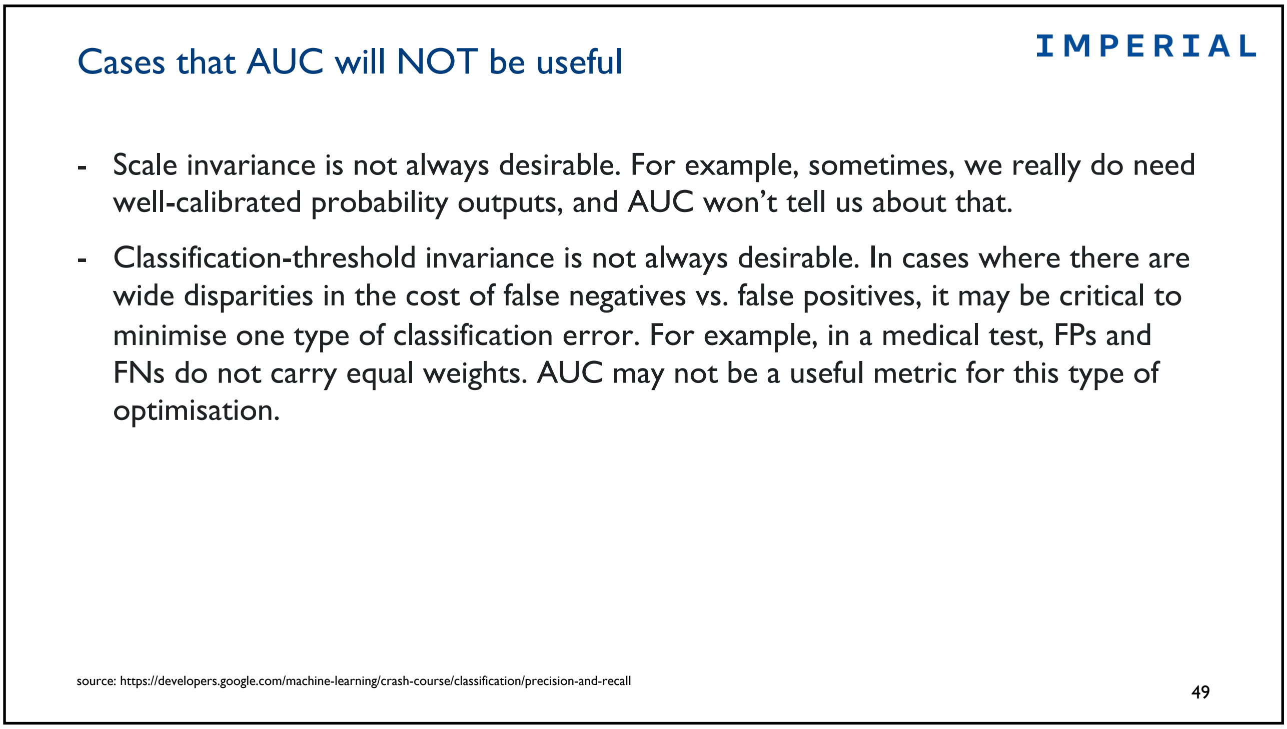
- ROC AUC:

- Evaluating overall classification performance,
- It can be misleading for imbalanced data,
- TPR vs. FPR

Cases that AUC could be useful

- AUC is desirable for the following two reasons:
  - AUC is scale-invariant. It measures how well predictions are ranked rather than their absolute values.
  - AUC is classification-threshold invariant. It measures the quality of the model's predictions regardless of the classification threshold chosen.

source: https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall

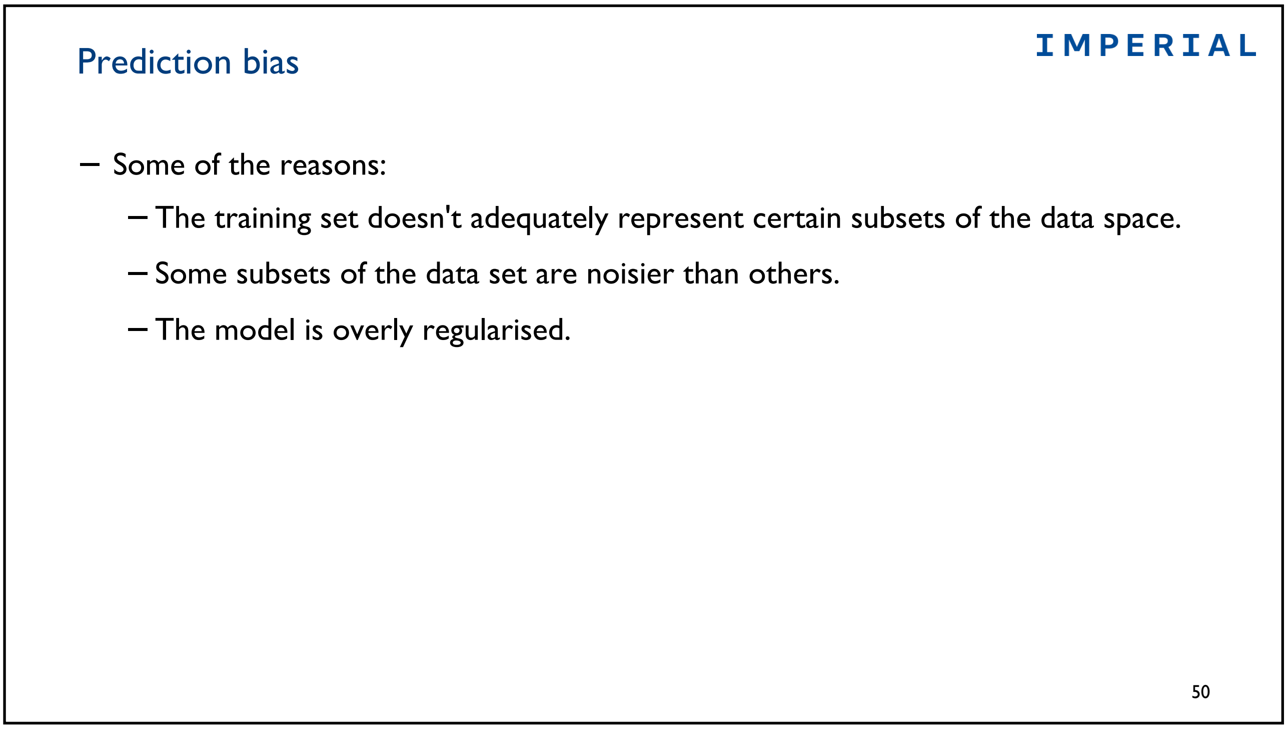


Cases that AUC will NOT be useful

- Scale invariance is not always desirable. For example, sometimes, we really do need well-calibrated probability outputs, and AUC won’t tell us about that.
- Classification-threshold invariance is not always desirable. In cases where there are wide disparities in the cost of false negatives vs. false positives, it may be critical to minimise one type of classification error. For example, in a medical test, FPs and FNs do not carry equal weights. AUC may not be a useful metric for this type of optimisation.

source: https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall

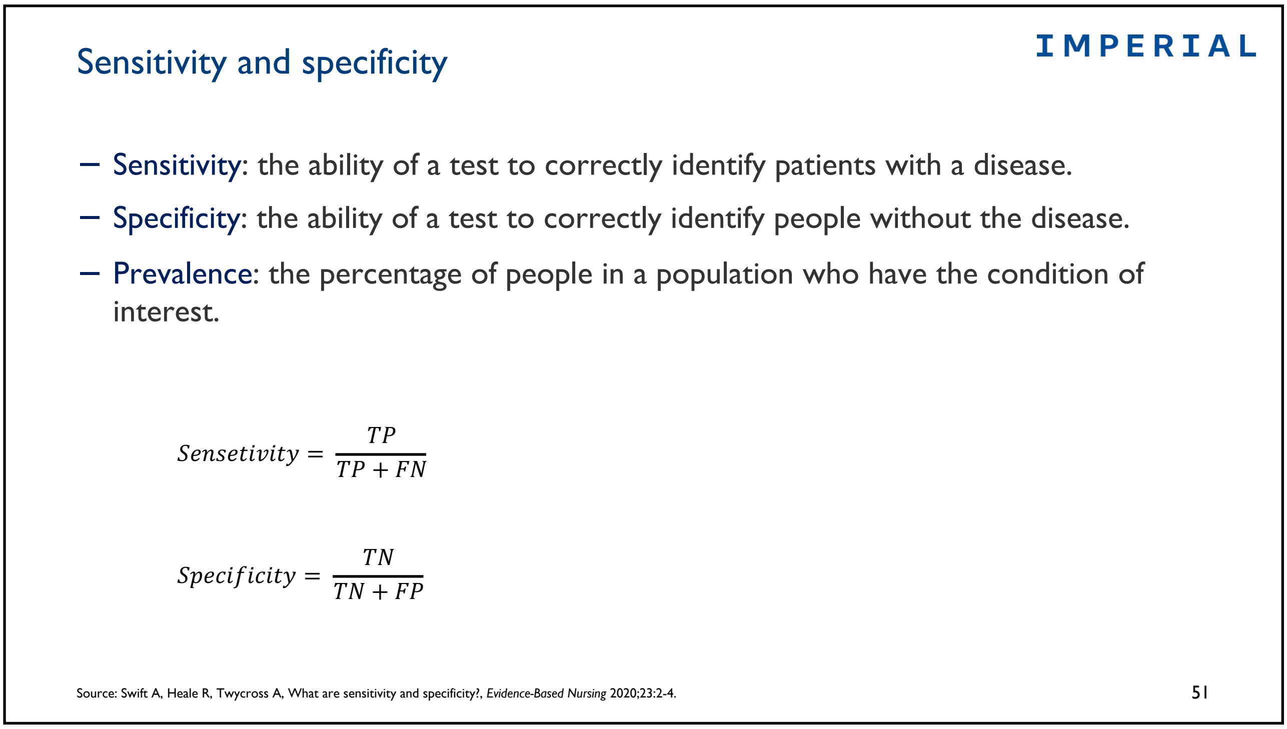
49



Prediction bias

- Some of the reasons:
  - The training set doesn't adequately represent certain subsets of the data space.
  - Some subsets of the data set are noisier than others.
  - The model is overly regularised.

50



Sensitivity and specificity

- **Sensitivity**: the ability of a test to correctly identify patients with a disease.
- **Specificity**: the ability of a test to correctly identify people without the disease.
- **Prevalence**: the percentage of people in a population who have the condition of interest.

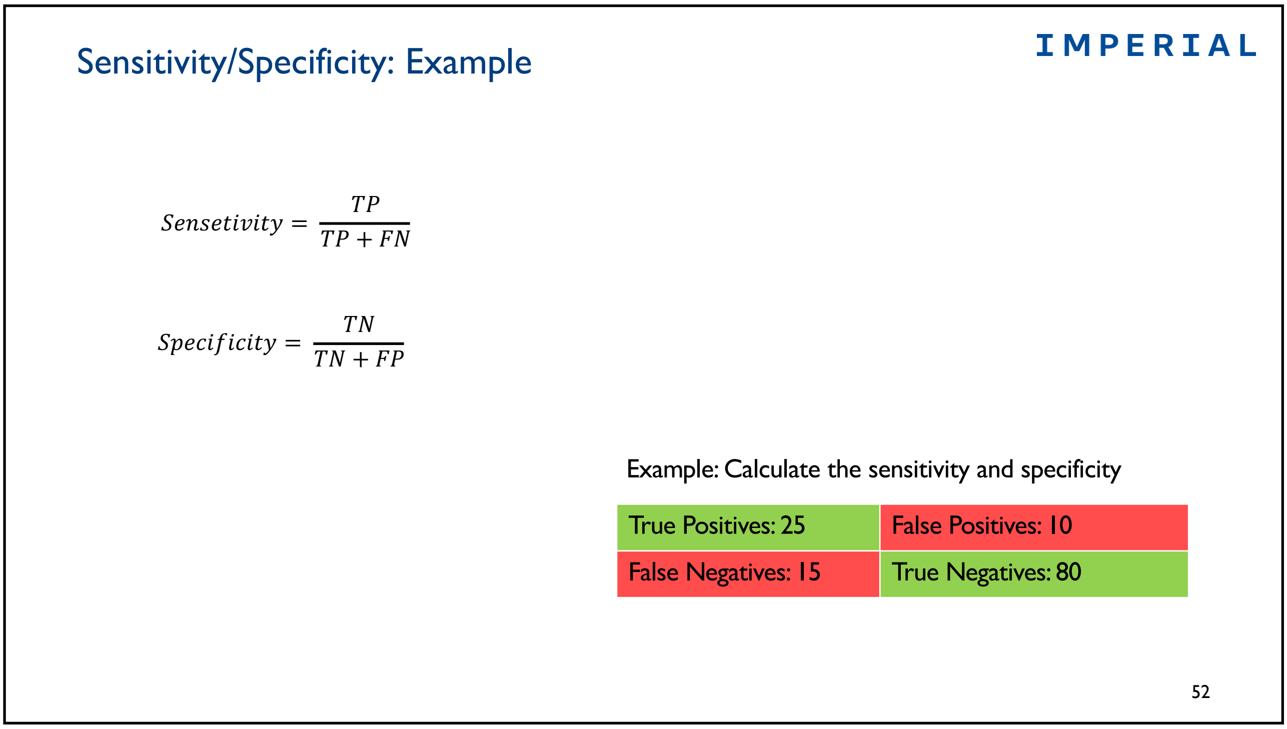
$$Sensetivity = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP}$$

Source: Swift A, Heale R, Twycross A, What are sensitivity and specificity?, *Evidence-Based Nursing* 2020;23:2-4.

51

51



Sensitivity/Specificity: Example

$$Sensetivity = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP}$$

Example: Calculate the sensitivity and specificity

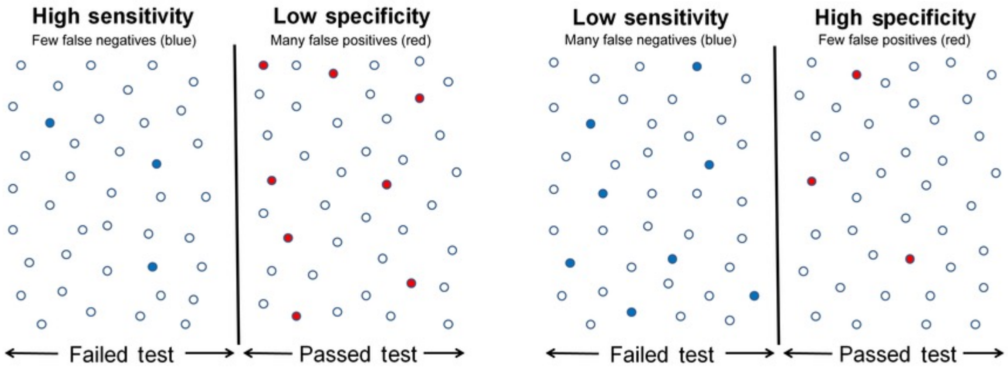
True Positives: 25	False Positives: 10
False Negatives: 15	True Negatives: 80

52

52

Specificity and Sensitivity

IMPERIAL



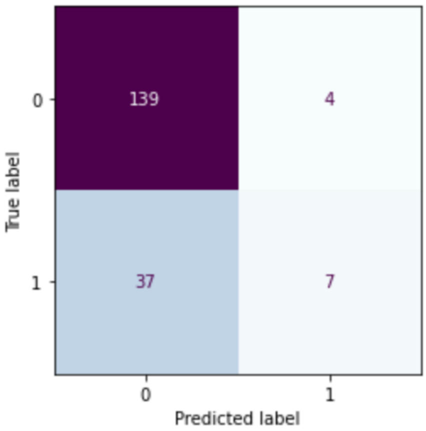
Source: https://en.wikipedia.org/wiki/Sensitivity_and_specificity

53

53

Confusion matrix

IMPERIAL



True label \ Predicted label	0	1
0	139	4
1	37	7

54

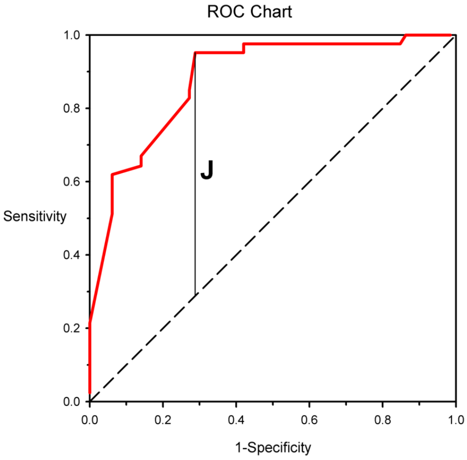
54

Youden’s index

- Captures the “informed-ness” measure.

$$J = \mathit{Sensetivity} + \mathit{Specificity} - 1$$

- The classification thresholds can be selected to maximise informedness.



Solid red: ROC curve; Dashed line: Chance level; Vertical line (J) maximum value of Youden's index for the ROC

Source: https://en.wikipedia.org/wiki/Youden%27s_J_statistic

Probability calibration

- Sometimes, in classification, you may want to predict the class label as well as the probability of the label for that class.
- This probability will provide an estimate/measure of the prediction.
- You need to be aware that some models can produce poor estimates of class probabilities (e.g., Naïve Bayes).
- Calibrating a classifier consists of fitting a regressor (called a calibrator) that maps the output of the classifier to a calibrated probability in [0, 1].
- Denoting the output of the classifier for a given sample by f_i, the calibrator tries to predict $p(y_i=1|f_i)$.
- See: https://scikit-learn.org/stable/modules/calibration.html

All in one place

IMPERIAL

$$Precision = \frac{TP}{TP + FP}$$

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision\ + Recall}$$

$$Recall = \frac{TP}{TP + FN}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Sensetivity = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP}$$

Open discussion

IMPERIAL

- Imagine a hospital deploying an AI model to predict which patients are likely to be readmitted within 30 days. The goal is to allocate extra care resources to high-risk patients.
- Initial Metric: The team optimises for overall accuracy and achieves 92%. Everyone celebrates.
- Hidden Problem: When they break down performance by demographic groups:
  - Accuracy for younger patients: 95%
  - Accuracy for older patients: 88%
- Was accuracy the right metric?

IMPERIAL

Review Questions

59

59

IMPERIAL

Q1

- Imagine you come across a model for which its accuracy is specified as 91%; is that good?

menti code will be provided.

60

60

Q2

IMPERIAL

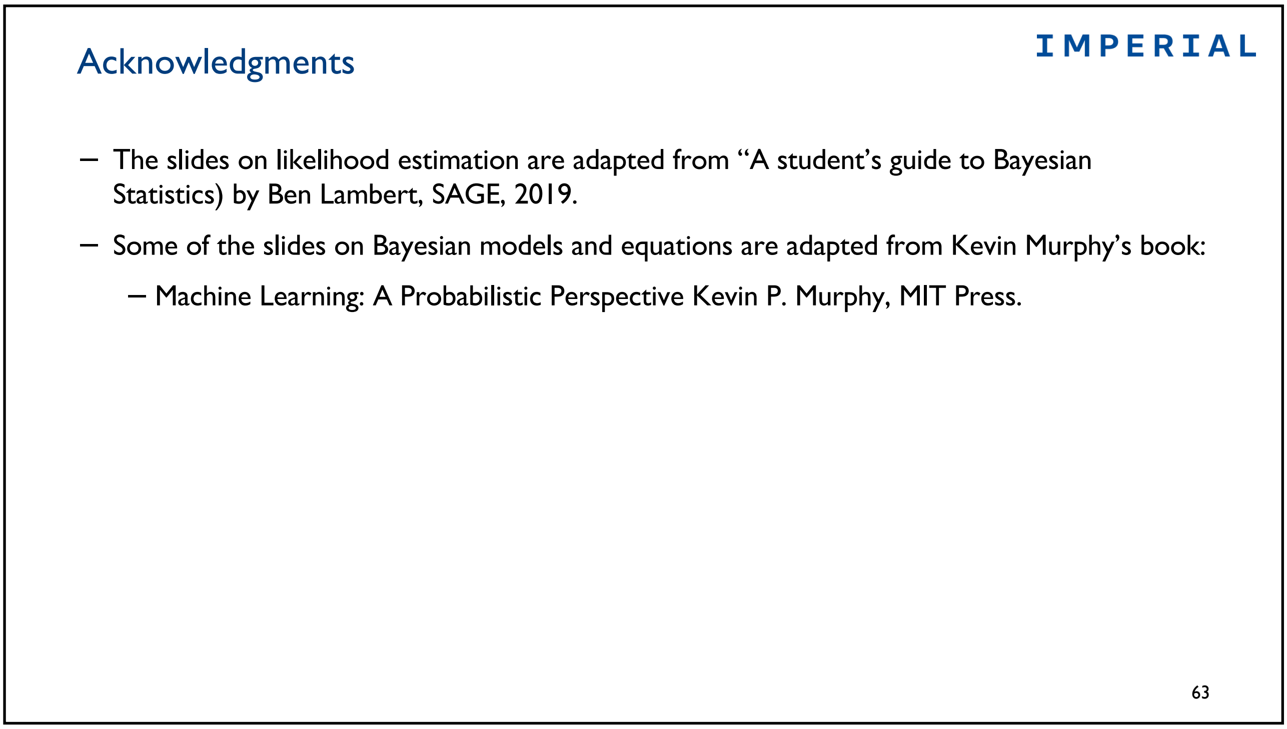
- Imagine you are developing a screening tool for the risk of Alzheimer's disease using electronic healthcare records (EHR) data. You will implement an ML model to process historical medical records. Which metric will give you a higher weight (if any)?

Metric in Scikit-learn

IMPERIAL

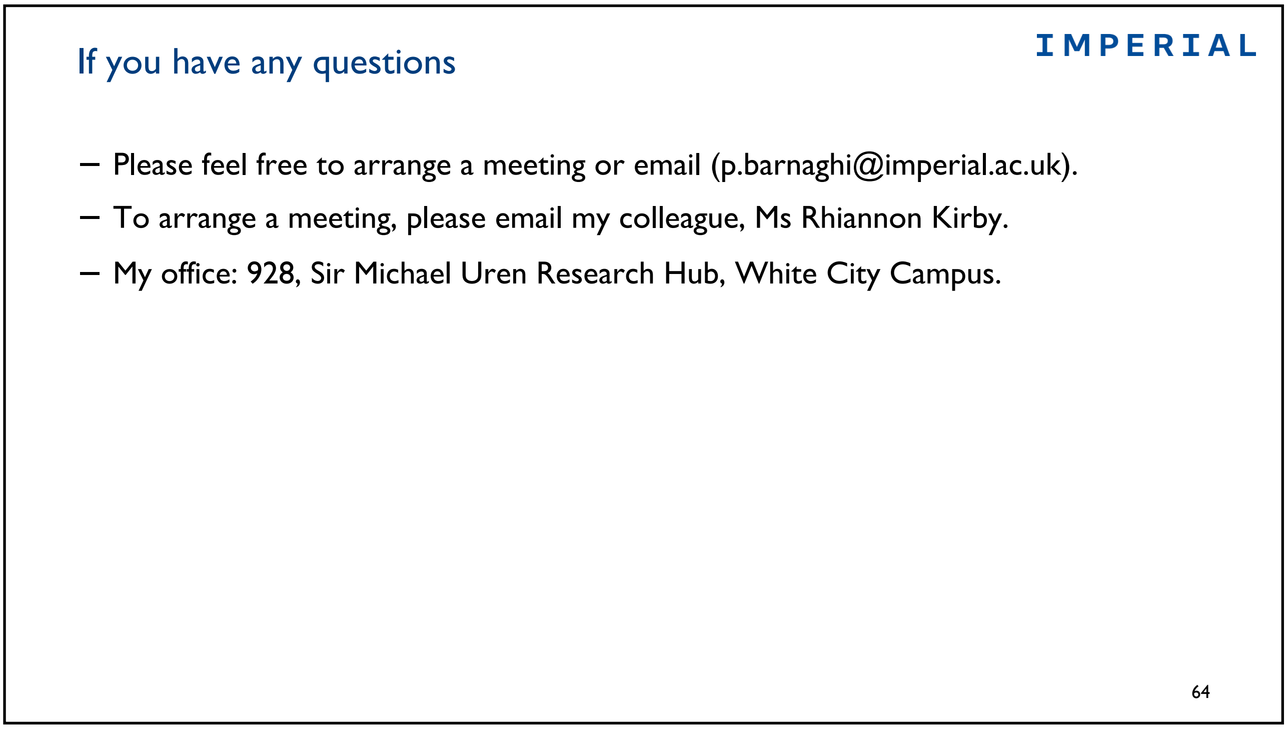
- Metrics and scoring:

https://scikit-learn.org/stable/modules/model_evaluation.html#confusion-matrix



Acknowledgments

- The slides on likelihood estimation are adapted from “A student’s guide to Bayesian Statistics) by Ben Lambert, SAGE, 2019.
- Some of the slides on Bayesian models and equations are adapted from Kevin Murphy’s book:
  - Machine Learning: A Probabilistic Perspective Kevin P. Murphy, MIT Press.



If you have any questions

- Please feel free to arrange a meeting or email (p.barnaghi@imperial.ac.uk).
- To arrange a meeting, please email my colleague, Ms Rhiannon Kirby.
- My office: 928, Sir Michael Uren Research Hub, White City Campus.