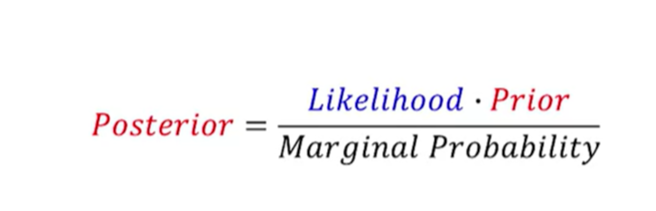
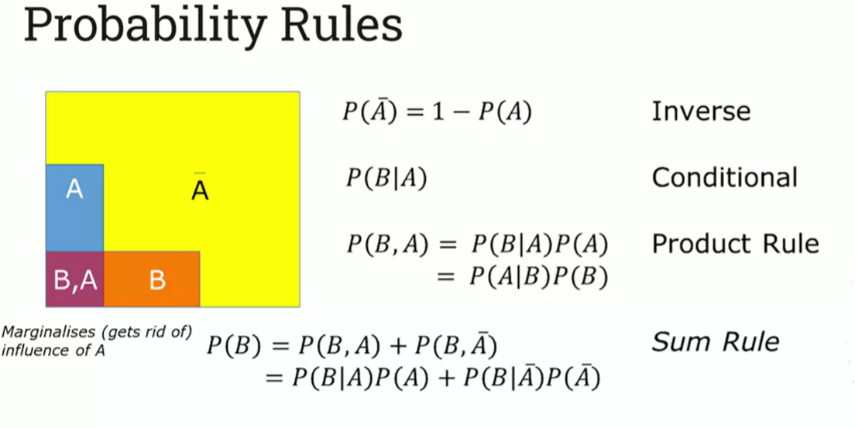
**Naïve Bayes**

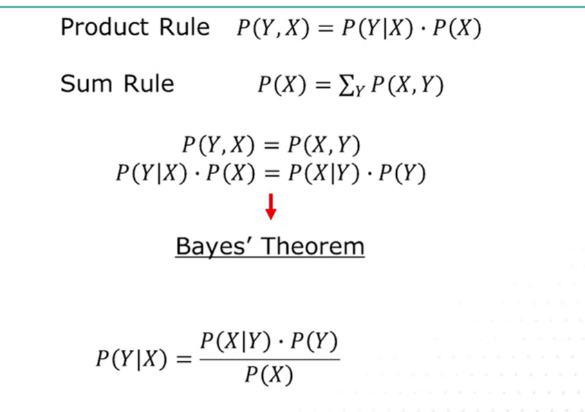
* Probabilistic model
* Outputs degrees of uncertainty
* Central concept of Bayesian modelling: new evidence can change your mind
* Prior probability = initial degree of belief
* Posterior probability = the belief *given* some evidence
* Calculating the probability of an event giving some observed evidence
* Uses a generative model that a certain outcome leads to some observation

**Bayes Theorem**

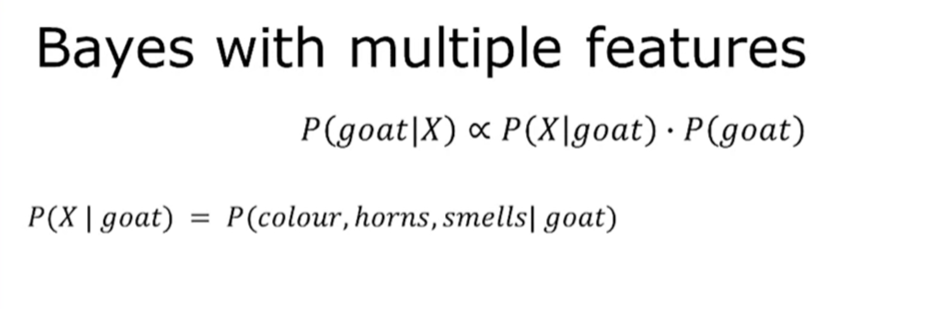




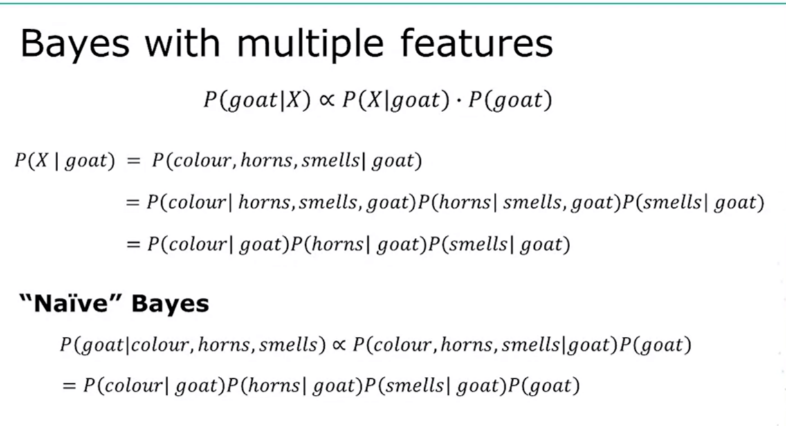


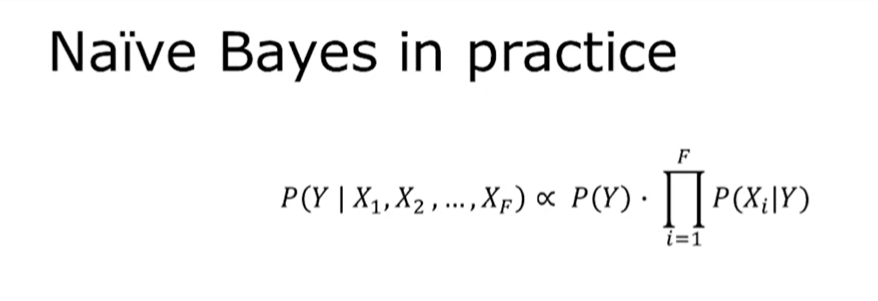


* Large nr of features/dimensions

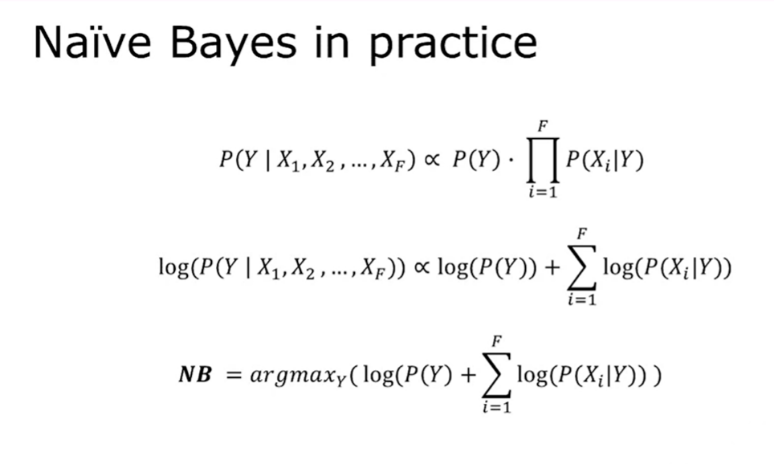


* Observations are not independent, need to look at combinations of features 🡪 chain rule make probability calcs v complicated.
* Make the assumption all the features are independent
* Reduce the product rule
* Assumption is ‘naïve’





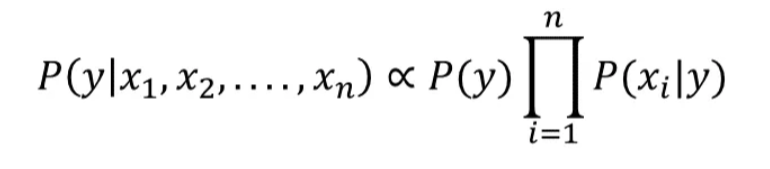
* Problem: multiplying small numbers 🡪 numbers get too small and go into numerical underflow
* Take the logarithms of the posterior, prior etc.



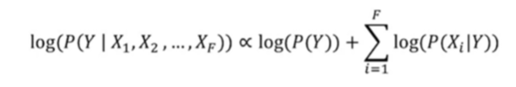
Ref: <https://www.researchgate.net/publication/220932746_Learning_Naive_Bayes_Classifiers_for_Music_Classification_and_Retrieval>

For the second classification algorithm, I have selected the Naïve Bayes Classifier, which is a model based on Bayes Theorem. The Naïve Bayes Classifier has the advantage of being fast compared to other classification algorithms such as K-Nearest Neighbour (ref: <https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/>) and easy to use because it doesn’t take in any hyperparameters. Furthermore, Naïve Bayes is highly scalable, requires a small amount of training data and is relatively easy to implement(ref: <https://medium.com/analytics-vidhya/na%C3%AFve-bayes-algorithm-5bf31e9032a2#:~:text=Naive%20Bayes%20is%20a%20classification,one%20with%20the%20highest%20probability>. ) Bayes Theorem, which provides the conceptual grounding for this classifier, (ref: <https://medium.com/analytics-vidhya/na%C3%AFve-bayes-algorithm-5bf31e9032a2#:~:text=Naive%20Bayes%20is%20a%20classification,one%20with%20the%20highest%20probability>.) calculates the conditional probability that a sample belongs to a class (in this context, is a song on a specific Taylor Swift album), *given* a set of audio features (e.g. valence, acousticness, energy) that the sample/song has. The formula for Bayes Theorem involves multiplying the *likelihood -* which is the conditional probability that a song has certain features (e.g. valence, acousticness etc.) *given* that it is a member of a certain class (album) - **by** the *prior probability*, which is defined as the overall probability that any song belongs to that album. By including the prior probability in the Bayesian calculation, this means that the final probability score for the outputted class (posterior) can account to a greater extent for the imbalanced nature and the overrepresentation of of certain album (classes) in the dataset. For example, if there are only 5 songs on an album in a dataset of 1000 songs, then the prior will be 0.005. Including this in the multiplication formula will greatly decrease the final probability of a song with those characteristics belonging to that album, thus accounting for the imbalanced proportions of different albums in the datasets. In contrast, as Kadir et al. have demonstrated, K-Nearest Neighbour is extremely ‘sensitive to the majority instances and thus perform[s] poorly for imbalanced datasets’ (ref: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7206335/pdf/978-3-030-47436-2_Chapter_6.pdf>) . Therefore, it will be interesting to compare and evaluate how Naïve Bayes performs on the album-classification task here compared to k-NN.

The Naïve Bayes classifier is called ‘naïve’ because the assumption is made that the features (audio characteristics, in this case) are independent from each other (ref: <https://medium.com/analytics-vidhya/na%C3%AFve-bayes-algorithm-5bf31e9032a2#:~:text=Naive%20Bayes%20is%20a%20classification,one%20with%20the%20highest%20probability>. ). One can assume that in reality, audio features such as ‘danceability’ and ‘tempo’ might not be independent: songs with a faster tempo are often considered more ‘danceable’. If one were to account for this lack of independence, the formula for the posterior probability would be substantially more complex, using the chain rule and many terms in the expression for the multiplication, but because of the *naïve* assumption of the variables not being connected in any way, the formula becomes very straightforward. Another assumption that we will make in the implementation of this classifier is assuming that the data follows a Gaussian (normal) distribution, with values symmetrically distributed around the mean (ref: <https://towardsdatascience.com/implementing-naive-bayes-from-scratch-df5572e042ac>) .



where, for this specific case, *y* represents the album label/class, *x* refers to the audio features and *n* refers to the number of features. The *NumPy* argmax function can be applied again in order to select the class with the highest posterior probability for a specific test sample. A final thing to note is because the values for the prior probabilities can get very small, multiplying a lot of small numbers on a computer system by each other can result in a value so tiny that it leads to numerical ‘underflow’, a situation resulting from a number being ‘too small to be represented by the CPU or memory’ (ref: <https://www.computerhope.com/jargon/u/underflo.htm>). As such, the formula used in the algorithm coded here will take logarithms of both sides (which is simple using *NumPy*) before applying *argmax* in order to address this potential issue. We will also ignore the denominator in the original Bayes’ Theorem formula, as this ensures that the result is a probability between 0 and 1, but we do not need this when our sole objective is to find the class that simply has the maximum posterior probability for that sample (ref: <https://levelup.gitconnected.com/classification-using-gaussian-naive-bayes-from-scratch-6b8ebe830266>).



As Naïve Bayes does not take in hyperparameters, cross-validation rather than nested cross-validation will suffice to fully test and validate the classifier’s performance.

Did really badly 🡪 look here as to why:

<https://www.datasciencecentral.com/comparing-classifiers-decision-trees-knn-naive-bayes/>