Understanding Naive Bayes

Supervised Naive Bayes Prediction

Marcus Birkenkrahe

March 22, 2023

Naive Bayes

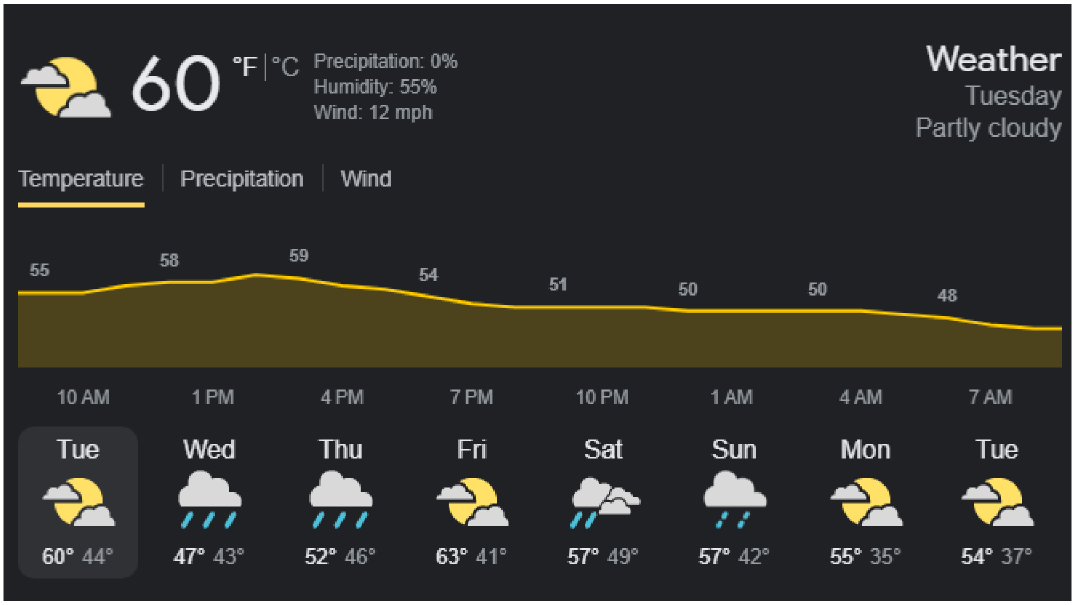
```
- Lecture notes in Markdown file (4_naive_bayes.md)
- Source: Lantz (2019), chapter 4, pp. 89-123
- DataCamp assignment: "Supervised learning with R", ch. 2
```

What you will learn

- Classification using Naive Bayes
- Bayes' theorem and naive assumptions
- Text classification use case
- R packages for text mining & visualization
- Application: SMS junk message filter

A little bit of math in here, but nothing more than basic arithmetic. The main complexity is to understand the relationship between features, class, and probability measures, which are our "similarity" measure for this type of classifier.

Probabilistic methods



60 °F | °C
Precipitation: 0%
Humidity: 55%
Wind: 12 mph

Weather
Tuesday
Partly cloudy

Tue	Wed	Thu	Fri	Sat	Sun	Mon	Tue
60° 44°	47° 43°	52° 46°	63° 41°	57° 49°	57° 42°	55° 35°	54° 37°

```
- Probabilistic methods describe uncertainty
- They use data on past events to extrapolate future events
- Such predictions are subject to many assumptions
```

- The chance of rain for example describes the proportion of prior days with similar atmospheric conditions in which in rained.
- A 70 percent chance of rain implies that in 7 out of 10 past cases with similar conditions, it rained somewhere in the area.
- The Naive Bayes algorithm uses probabilities in a similar way to a weather forecast.

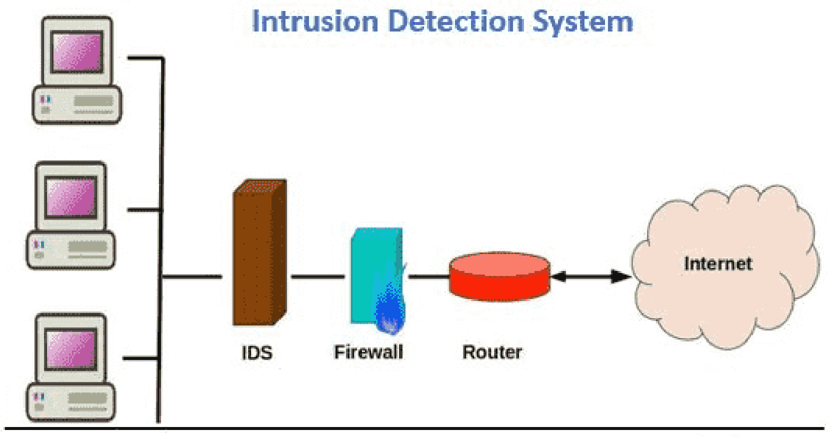
Probability

```
- A probability P is a number between 0 and 1 (0% to 100%)
- P captures the chance that an event will occur based on evidence
- P = 0 indicates that the event will definitely not occur
- P = 1 indicates that the event will occur with absolute certainty
```

Bayesian methods

```
- Training data are used to calculate outcome probability
- Evidence is provided by labeled feature values
- Classifier uses calculated probabilities to estimate class
```

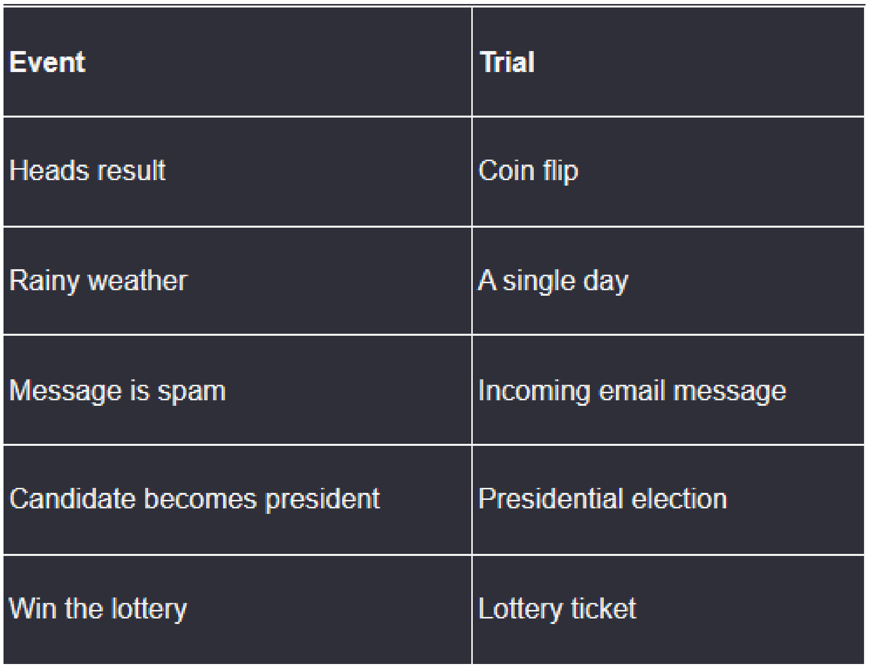
Applications



- Text classification, e.g. spam filter

- Anomaly detection in computer networks
- Diagnosing medical conditions
- Best for problems where information from numerous attributes should be considered simultaneously to estimate overall probability of an outcome.
- E.g. spam filter: various words found in an example/message instance
- Unlike other ML methods, Bayesian methods use all available evidence to make predictions.
- Even if a large number of features have minor effects, their combined impact in a Bayesian model could have a major impact.

Basic idea



Event	**Trial**
Heads result	Coin flip
Rainy weather	A single day
Message is spam	Incoming email message
Candidate becomes president	Presidential election
Win the lottery	Lottery ticket

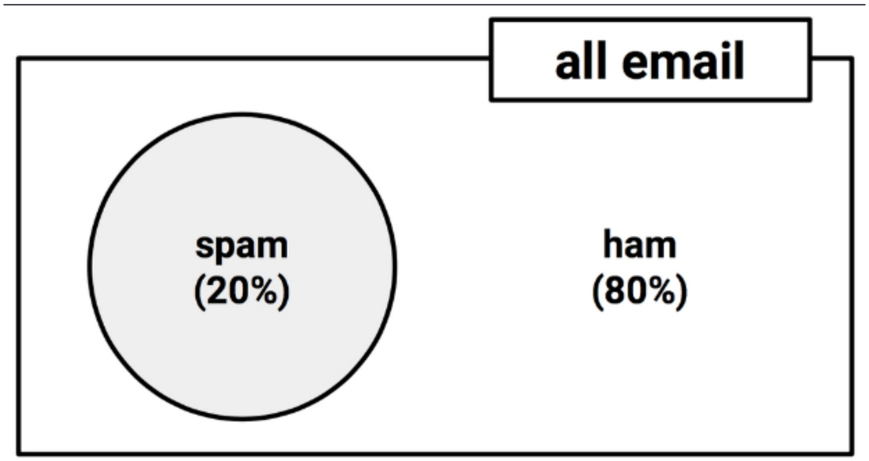
The estimated likelihood of an **event** or potential outcome is based on the evidence from multiple **trials** or opportunities for the event to occur.

The more trials the better for the accuracy of the estimate - by way of the **law of large numbers**: if you repeat an experiment independently a large number of times and average the result, your result is close to the expected value (the arithmetic mean):

- Large number of coin flips - P(head) = P(tail) = 50%
- Large number of observed days - weather averages
- Large number of email messages - certain spam prediction
- Large number of elections - certain presidential prediction
- Large number of lottery tickets - certain win

But: real events are never mathematically independent.

Spam vs. Ham

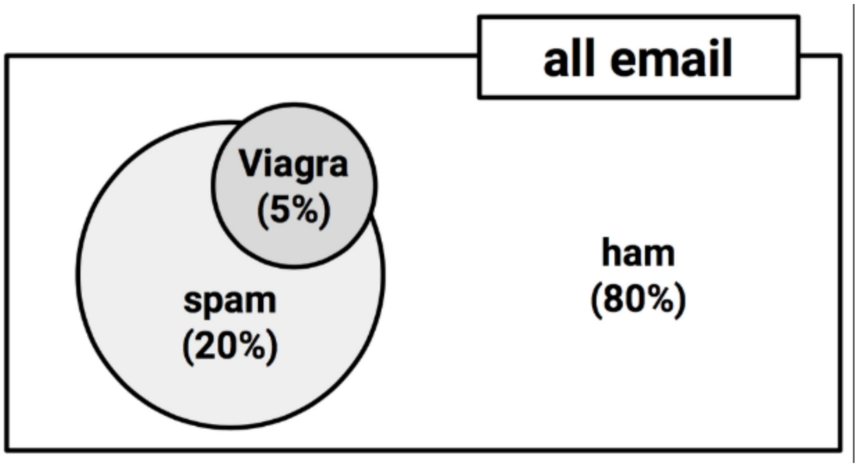


In email trials, spam and ham are mutually exclusive and exhaustive events.

- P(event) = no. of occurrences / no. of trials
- Rain on 3/10 days w/similar conditions: 30% prob today
- Adding all P => 100% of the data or $\sum$ P(event)=1 because a trial always results in an outcome.
- P(spam) + P(ham) = 1 implies that spam/ham **mutually exclusive and exhaustive**.

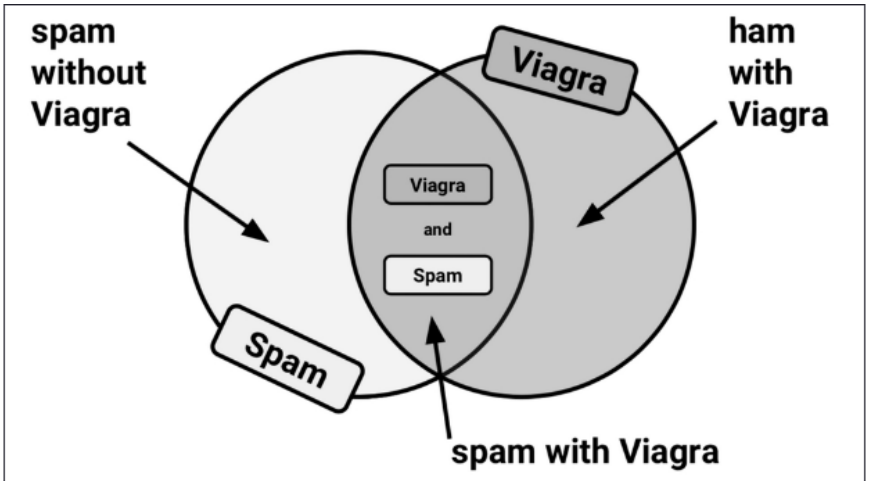
- An alternative way of saying this uses a table of records: if you record many, many instances, say 1000, you have 200 lines marked as 'spam' and 800 lines marked as 'ham'.

Joint probability



'Viagra' is a non-mutually exclusive event. Its overlap with 'spam' is larger than its overlap with 'ham'.

Venn diagrams



Calculating P(spam ∩ Viagra) depends on the joint probabilities of the two events, on their **dependency**.

- The Venn diagram illustrates instances that are only spam, only Viagra but not spam and spam with Viagra messages.
- Named after 19th century mathematician John Venn
- If the circles aren't touching, the joint prob is 0 and the events are said to be **independent**. They can still occur simultaneously.
- A ∩ B = 0: Knowing something about the outcome of A reveals nothing about the outcome of B. Hard to illustrate in the real world, but:
- The outcome of a coin flip is unlikely to depend on the weather being sunny or rainy on any given day.
- **Dependent events are the basis of predictive modeling.**
- The appearance of clouds is predictive of rain, the appearance of the word 'Viagra' is predictive of spam.

Bayes' theorem

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

- For independent events, P(A ∩ B) = P(A) * P(B)
- P(Viagra AND spam) = (5/100) * (20/100) = 0.01
- P(A|B) is the probability of A given B occurred
- P(A|B) is the probability of A conditional on B

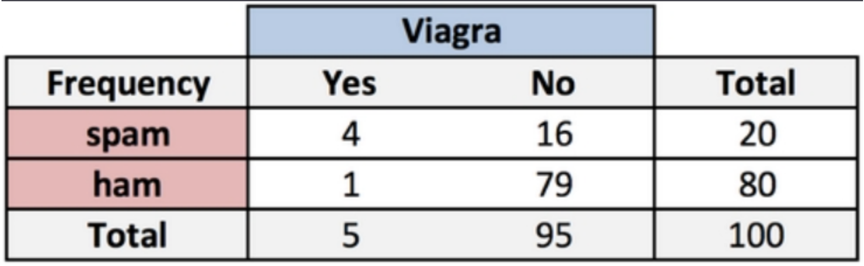
$$P(A\mid B)=\frac{P(A\cap B)}{P(B)}=\frac{P(B\mid A)P(A)}{P(B)}$$

- Recall: we're trying to predict the chance that a message that contains the word 'Viagra' (B) is spam (A).
- The formula states that the best estimate of P(A|B) is the proportion of trials in which A occurred with B, P(A ∩ B), out of all trials in which B occurred (all 'Viagra' messages).
- Extreme cases: if B is very rare, P(B) is small and the correction to P(A) is negligible (independence)
- If A and B occur together very often, P(A|B) will be high regardless of P(B).
- If Viagra and spam were independent, P(A ∩ B) = 0.05 * 0.20 = 0.01

Bayesian spam filter

$$\underset{\text{posterior probability}}{P(\text{spam}|\text{Viagra})} = \frac{\overset{\text{likelihood}}{P(\text{Viagra}|\text{spam})}\,\overset{\text{prior probability}}{P(\text{spam})}}{\underset{\text{marginal likelihood}}{P(\text{Viagra})}}$$

To calculate the components, construct a frequency table that records how often 'Viagra' appeared in 'spam' and 'ham' messages.



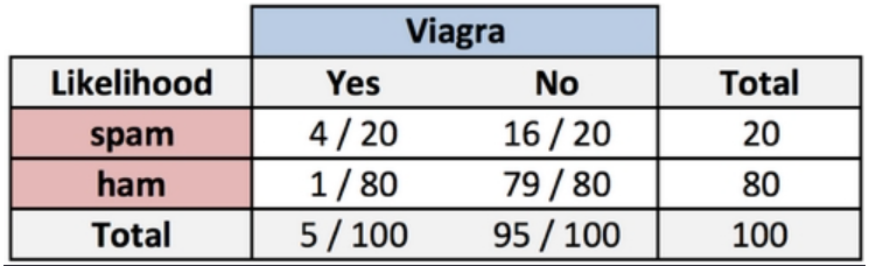
	Viagra		
Frequency	Yes	No	Total
spam	4	16	20
ham	1	79	80
Total	5	95	100

- Without knowing anything about an incoming messages, our best estimate would be P(spam) - the **prior probability** (20%)

- The chance of having any 'Viagra' in a spam message is the **marginal likelihood** - having any 'Viagra' at all is the **marginal likelihood**

- What we're after is a computation of the **posterior probability** (i.e. **after** applying the condition 'Viagra').

Likelihood table



	Viagra		
Likelihood	**Yes**	**No**	**Total**
spam	4 / 20	16 / 20	20
ham	1 / 80	79 / 80	80
Total	5 / 100	95 / 100	100

The rows of the likelihood table contain the conditional probabilities for "Viagra" (yes/no) given that an email was spam or ham:

```
P(Viagra = Yes | spam) = 4/20 = 0.20
P(spam & Viagra) = P(Viagra|spam) * P(spam) = (4/20)*(20/100) = 0.04
P(spam|Viagra) = (4/20) * (20/100) / (5/100) = 0.80
```

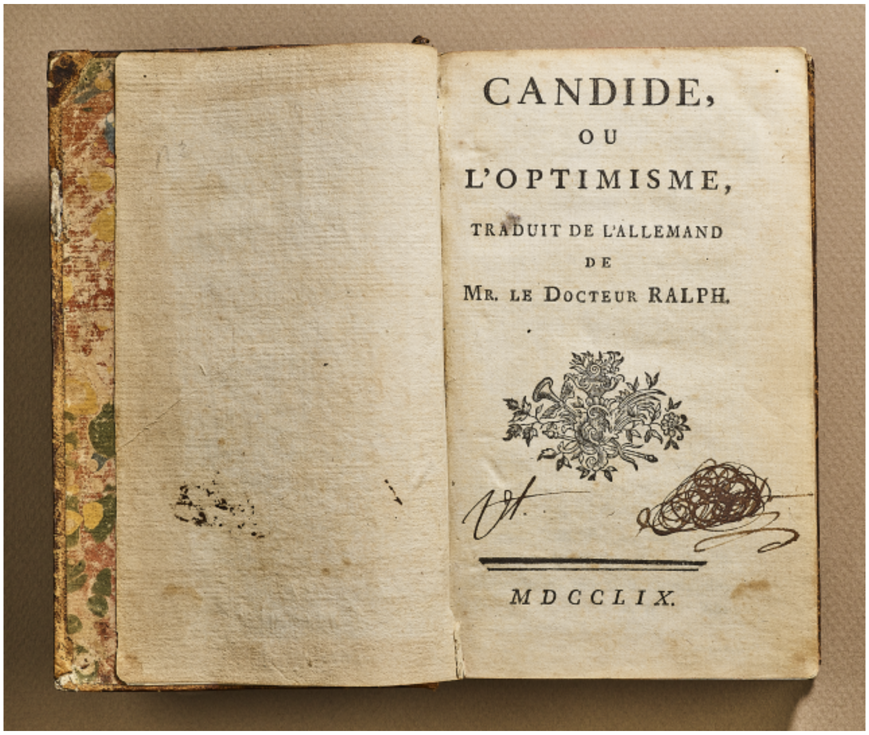
- The computed chance of getting spam AND Viagra is FOUR times as large as the chance when independence was assumed (P(Viagra)*P(spam)=0.01)

- The posterior probability that a message containing Viagra is spam is 80% - any message containing this term should be filtered.

- This is how commercial spam filters work: they consider a much larger number of words simultaneously when computing frequency and likelihood tables.

- The Naive Bayes algorithm accounts for these additional difficulties. It also relies on careful text pre-processing of the message data.

Naïvety of the algorithm



CANDIDE,
OU
L'OPTIMISME,
TRADUIT DE L'ALLEMAND
DE
MR. LE DOCTEUR RALPH.

MDCCLIX.

- All features are equally important and independent
- Is this justified in real datasets?
- Examples: spam / sentiment analysis

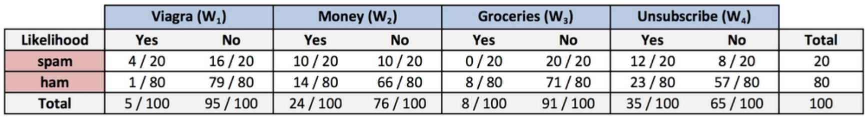
  The question is always what we want to classify: if we're after spam, some features are more **important** than others, e.g. the email sender or the subject line. Words in the message body are not **independent** from one another - e.g. "Viagra" will be accompanied by "drugs", "cash" by "free" etc.

  If we analyze for sentiment (categories: good, bad, neutral, etc.) in online reviews, then length of the review is more **important** than the (anonymous) sender. The sentiment features are not **independent** of the time of the review - it is related to the product launch time, the time of day, etc.

  NB performs well even when these assumptions are violated, even if there are strong feature dependencies, especially with smaller datasets.

The exact reason for this success is not known.

Adding more features



	Viagra (W_1)		Money (W_2)		Groceries (W_3)		Unsubscribe (W_4)		
Likelihood	Yes	No	Yes	No	Yes	No	Yes	No	Total
spam	4 / 20	16 / 20	10 / 20	10 / 20	0 / 20	20 / 20	12 / 20	8 / 20	20
ham	1 / 80	79 / 80	14 / 80	66 / 80	8 / 80	71 / 80	23 / 80	57 / 80	80
Total	5 / 100	95 / 100	24 / 100	76 / 100	8 / 100	91 / 100	35 / 100	65 / 100	100

Is the message spam given that it contains the terms "Viagra" and "unsubscribe", but not "Money" or "Groceries"?

$$P(\text{spam} \mid W_1 \cap \neg W_2 \cap \neg W_3 \cap W_4) = \frac{P(W_1 \cap \neg W_2 \cap \neg W_3 \cap W_4 \mid \text{spam}) P(\text{spam})}{P(W_1 \cap \neg W_2 \cap \neg W_3 \cap W_4)}$$

Cp. "The Ultimate List of 394 Email Spam Trigger Words to Avoid in 2023"

As new messages are received, we need to calculate the posterior probability to determine whether they are more likely spam or ham, given the likelihood of the words being found in the message text.

Computational complexity is enormous: probabilities for possible intersecting events need to be stored. Imagine the Venn diagram of four overlapping circles - in reality, we have hundreds of features.

Core assumptions

- Class-conditional independence
- Constant marginal likelihood

$$P(\text{spam} \mid W_1 \cap \neg W_2 \cap \neg W_3 \cap W_4) \propto P(W_1 \mid \text{spam}) P(\neg W_2 \mid \text{spam}) P(\neg W_3 \mid \text{spam}) P(W_4 \mid \text{spam}) P(\text{spam})$$

$$P(\text{ham} \mid W_1 \cap \neg W_2 \cap \neg W_3 \cap W_4) \propto P(W_1 \mid \text{ham}) P(\neg W_2 \mid \text{ham}) P(\neg W_3 \mid \text{ham}) P(W_4 \mid \text{ham}) P(\text{ham})$$

- Events are **independent** as long as they are conditioned on the same class value: for example, "Money" and "Unsubscribe" are considered independent when found in a spam message. Reduces the numerator term to a product of probabilities.
- Denominator does not depend on the target class (spam or ham) and is treated as **constant** and can be ignored.

- The equation has become a proportion
- To convert the likelihoods to probabilities, the denominator needs to be re-introduced (rescale likelihood of each outcome by total likelihood across all possible outcomes).

Formula

$$P\left(C_L \mid F_1,\ldots,F_n\right)=\frac{1}{Z}\,p\left(C_L\right)\prod_{i=1}^{n} p\left(F_i \mid C_L\right)$$

- Class levels L (e.g. spam vs. ham)
- Features F (e.g. "Money", "Urgent")
- Scaling factor Z

Z converts the likelihood values to probabilities.

Workflow

1. compute frequency table
2. compute likelihood table
3. multiply probabilities "naively"
4. rescale likelihood to probability

Process:

1. frequency table,
2. likelihood table,

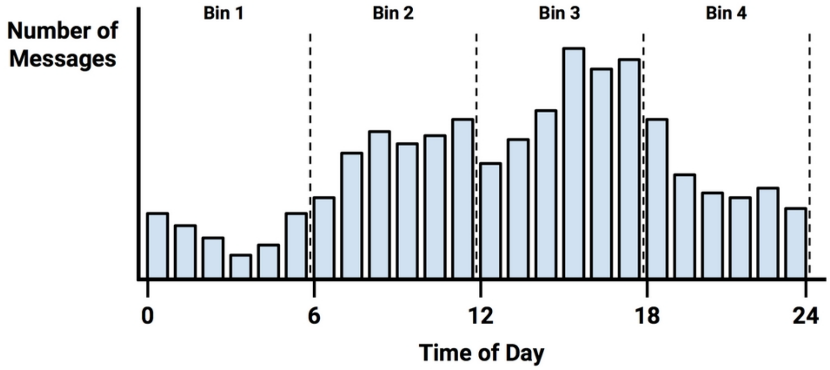
3. multiply conditional probabilities with "naive" assumption of independence,
4. divide by total likelihood to transform each class likelihood to a probability.

Laplace correction

- What if an event never occurs for one or more levels
- Joint probability P(spam|groceries) = 0%
- Laplace estimator adds small number to counts
- If an event (e.g. "groceries") has never before occurred in a spam message, the likelihood P(groceries|spam) = 0 and the chain product to compute the posterior probability P(spam|. . . .) is zero if "groceries" is suddenly found in a message.
- Add a small correction to the counts in the frequency table, so instead of $0/20$ compute $1/20$ so that each feature has a non-zero probability of occurring with each class.
- If the estimator is 1 this assumes that each class-feature combination is found in the data set at least once.

- The correction does not have to be the same for each feature. Additional assumptions about the coupling of class and feature can be built in. This is not practical for large datasets.
- Note that the prior probabilities P(spam) and P(ham) are not affected or corrected because this is our best estimate for the observations.

Numeric features



- Frequency tables require the features to be categorical
- Numeric features do not have categories of values
- The algorithm will not directly work with numeric data
- Solution: discretize numeric features into **bins** ("binning")
- Works best when there are large amounts of training data
- Practice: cut points in the distribution, e.g. for continuous time as a feature, the data could be divided into four levels.
- If binning is not obvious, you can discretize using quantiles - divide data into three bins with tertiles, four with quartiles etc.
- Binning/discretizing always reduces information: too few could obscure trends (e.g. in the diagram: 2 bins), too many increases the sensitivity to noisy data - that's naive Bayes "underfitting" and "overfitting".

Strength and Weaknesses

Strengths	Weaknesses
Simple, fast, effective	Feature independence and equity
Good for noisy, missing data	Not good for numeric features
Works few few or many samples	Unreliable estimated probabilities
Easy to obtain probability	

Strengths:
Simple, fast, effective
Good for noisy, missing data
Works few few or many samples
Easy to obtain probability for prediction

Weaknesses
Relies on feature independence and equity
Not ideal for data with many numeric features
Estimated probabilities less reliable than predictions

Summary

- NB constructs tables of probabilities to estimate the likelihood that new examples belong to various classes.
- Probabilities are calculated using Bayes' theorem, which specifies how dependent events are related.
- "Naive Bayes" makes assumptions about the independence of features to handle much larger data sets.

Glossary of terms

TERM	MEANING
Probability P	Number between 0 and 1
	Chance of an event occurring
P = 0	Event will definitely not occur
P = 1	Event will certainly occur
Evidence	Labeled feature values

Event	Something that happened or can happen
Trial	Sample of at least one event
Spam	Malicious message
Ham	Non-malicious message
Mutually exclusive events	Events cannot happen in the same trial
Exhaustive events	Events cover all possible events
Joint probability	Chance for events to happen together
Independent events	Joint probability is zero
Dependent events	Conditional probability is non-zero
Bayes' Theorem (simple)	P(A|B) = P(A AND B) / P(B)
Bayes' Theorem (complete)	P(A|B) = P(B|A) * P(A) / P(B)
	Posterior P =
	likelihood * prior P /marginal P

Glossary (cont'd)

TERM	MEANING
Conditional probability	P(A|B) Probability of A
	given B occurred
Frequency table	Counts features for each class level
Likelihood table	Feature likelihood
	conditional on class levels
Naivety of Naive Bayes	All features are equally important
	All features are independent
Naive Bayes simplified	Class conditional independence
	Constant marginal likelihood
Laplace correction	Likelihood 1 for events that never
	occur for one or more levels
Numeric Bayes	Discretized or binned feature values
Bayes' under/overfitting	Too few/many bins obscure/created trends

References

- Lantz (2019). Machine Learning with R (3e). Packt.
- Majka M (2019). naivebayes: High Performance Implementation of the Naive Bayes Algorithm in R. R package v0.9.7, URL: r-project.org.
- Photo by Dmitry Ratushny on Unsplash

- Photo by Naser Tamimi on Unsplash
- Photo by Markus Spiske on Unsplash