Introduction to the Naive Bayes Classifier

THE KNOWLEDGE HOUSE

Agenda - Schedule

1. Kahoot
2. Naive Bayes Classifier
3. Break
4. Naive Bayes & Probability Distributions

It's stats all the way down

Agenda - Goals

- ...

Kahoot

Week 5 Kahoot - Classification

Naive Bayes Classifiers

Naive Bayes Classifier

We can utilize our formula in a **supervised learning multiclass classifier** called the **naive bayes classifier**.

The goal of naive bayes classification is to calculate **the probability a new sample belongs to a certain class** using **historical data**.

We use this probability to calculate our “confidence” of a sample belonging to a class.

We can take this a step further and **assume shapes of our probability distribution.**

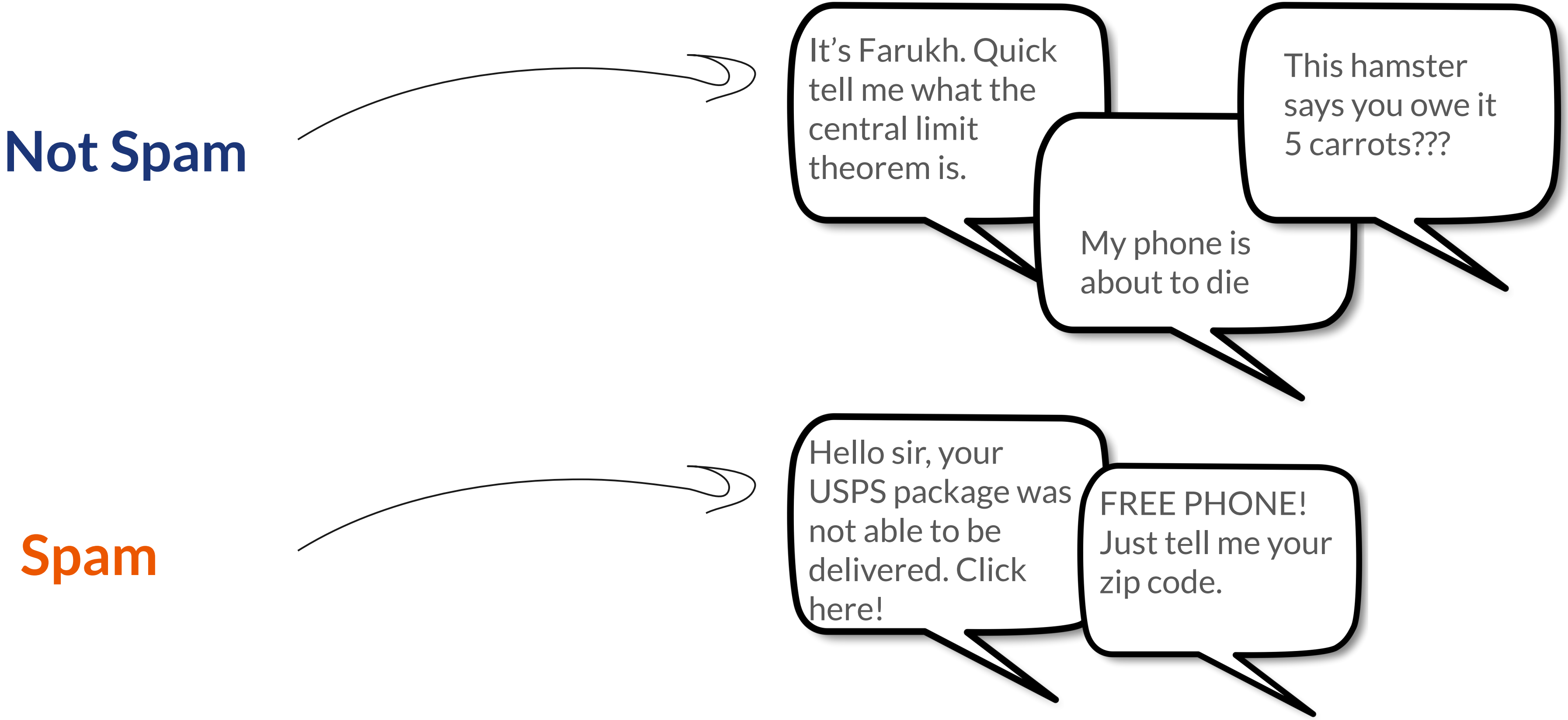
Naive Bayes Classifier

With our logistic and linear models, we had a **formula** to model our data.

However this does not apply to the **naive bayes classifier**.

Instead, we will **simply compute probabilities and likelihoods** using the **ratios we observe in our dataset.**

Let's take a look at the formula that we use, and then a "spam-text" example.



You have **5 texts**. 2 out of those messages are spam messages trying to steal your card info. The other 3 are human.

Word	Spam	Ratio
Farukh	Yes	P(Farukh|Spam)
Farukh	No	P(Farukh|Not Sp)
Phone	Yes	P(Phone|Spam)
Phone	No	P(Phone|Not Sp)

Using the **number of times a word appears in a type of message, divided by the total number of a specific type of message** we can calculate the **likelihood** of a text belonging to the spam or non-spam class!

How many times does the word "Farukh" appear in spam?

How many spam messages do we have?

Word	Spam	Ratio
Farukh	Yes	P(Farukh|Spam)
Farukh	No	P(Farukh|Not Sp)
Phone	Yes	P(Phone|Spam)
Phone	No	P(Phone|Not Sp)

P(Farukh|Spam) = Frequency of Farukh & Spam / Frequency of Spam

How many times does the word "Farukh" appear in spam? = 0

How many spam messages do we have? = 2
0/2 = 0

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	???
Phone	Yes	???
Phone	No	???

$$P(\text{Farukh}|\text{Spam}) = 0 / 2$$

How many times does the word “Farukh” appear in non-spam?

How many non-spam messages do we have?

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	???
Phone	Yes	???
Phone	No	???

P(Farukh|Not Spam) = Frequency of Farukh & Not-Spam / Frequency of Not-Spam

How many times does the word "Farukh" appear in non-spam? = 1

How many non-spam messages do we have? = 3
⅓ =0.333

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	???
Phone	No	???

P(Farukh|Not Spam) = 1 / 3

How many times does the word “Phone” appear in spam? = 1

How many spam messages do we have? = 2
½ = 0.5

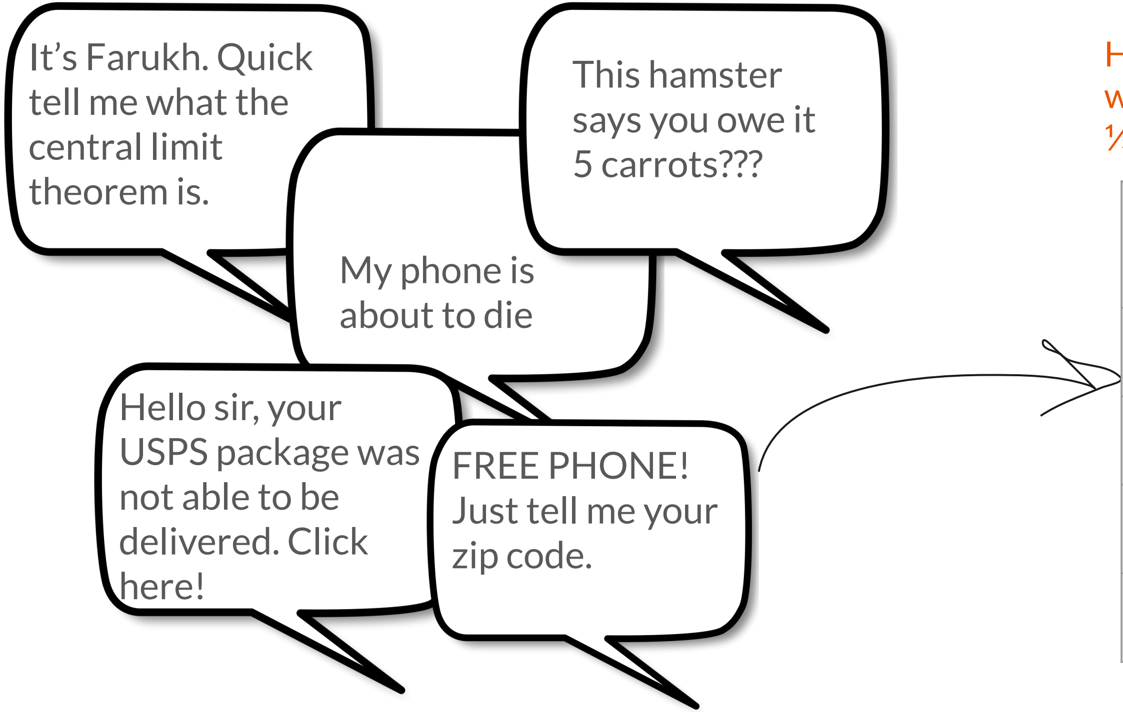
Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	???

P(Phone|Spam) = Frequency of Phone & Spam / Frequency of Spam

How many times does the word "Phone" appear in non-spam? = 1

How many non-spam messages do we have? = 3

⅓ = 0.33



Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

P(Phone|Not-Spam) = Frequency of Phone & Not-Spam / Frequency of Not-Spam

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

The last thing we need to consider is the probability of getting a spam text and the probability of getting a non-spam-text. AKA our **PRIOR**.

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

P(S) = ???

P(NS) = ???

We simply calculate this by getting the ratio of spam texts and the ratio of non-spam texts. Can anyone figure this out?

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

$P(S) = 2/(3+2) = \frac{2}{5} = 0.4$

$P(NS) = 3/(2+3) = \frac{3}{5} = 0.6$

$$P(H|E) = \frac{P(H)\ P(E|H)}{P(E)}$$

Now that we have all these calculations, let's bring back Bayes theorem. The likelihood of a **hypothesis given an event** is our **previous evidence** multiplied by the **probability of the event occurring ASSUMING OUR HYPOTHESIS IS TRUE**.

…

$$P(H|E) = \frac{P(H)\ P(E|H)}{P(E)}$$

Now that we have all these calculations, let's bring back Bayes theorem. The likelihood of a **hypothesis given an event** is our **previous evidence** multiplied by the **probability of the event occurring ASSUMING OUR HYPOTHESIS IS TRUE**.

Divided by **the probability of the event occurring when the hypothesis is true or not true!**

$$P(H|E) = \frac{P(H)\ P(E|H)}{P(E)}$$

So! Using the simple rule of "**more likelihood means more confidence**" we could say that the class that results in the **highest likelihood is the class that the sample belongs to!**

We'll try all available classes and see which class gives the highest number!

“argmax” simply means, select the argument (*y*) that gives me the maximum value

$$\hat{y} = \text{argmax} \frac{P(Y)\ P(X|Y)}{P(X)}$$

This could be formalized into the above formula. Choose the “Y” (class) that maximizes the probability given the “X” evidence.

$$\hat{y} = \operatorname{argmax} P(Y)\ P(X|Y)$$

The reasoning isn't obvious **yet**. However, we can actually eliminate the denominator from this calculation. This will become clear why once we go through an example.

$$\hat{y} = \text{argmax} \quad P(Y) \; P(X1|Y) \; P(X2|Y) \ldots P(Xn|Y)$$

Furthermore, we multiply conditional probabilities for **each event that occurs**.

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Using all these values, we can calculate the **proportional probability** that a **new sample belongs to spam (or not spam)!** Let’s say we get a new text: “Phone”

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

To figure out the probability that this is a spam text, we utilize our bayes theorem formula and see which “score” is higher. **We select the class that results in the highest score!**

Phone

$P(S) = 0.4$

$P(NS) = 0.6$

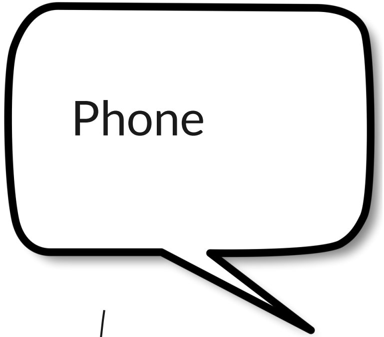
Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Not spam

$$P(NS|\text{"Phone"}) = \frac{P(NS)\ P(\text{"Phone"}|NS)}{P(\text{"Phone"})}$$

* I keep the denominator for now to show you why we don't need it later

Let's first assume "**not spam**". What is the probability of no spam in our dataset?



P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Not spam

$$P(NS|\text{"Phone"}) = \frac{0.6\ P(\text{"Phone"}|NS)}{P(\text{"Phone"})}$$

So far, the only "event" we have is the word "Phone." What is the probability of the word "Phone" given the text is not spam?

$P(S) = 0.4$

$P(NS) = 0.6$

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Phone

Not spam

$$P(NS|\text{"Phone"}) = \frac{0.6 * 0.33}{P(\text{"Phone"})}$$

Multiplying this, we get a score of …

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Phone

Not spam

P(NS|”Phone”) =0.198

P(“Phone”)

Multiplying this, we get a score of … 0.198.

Phone

P(S) = 0.4

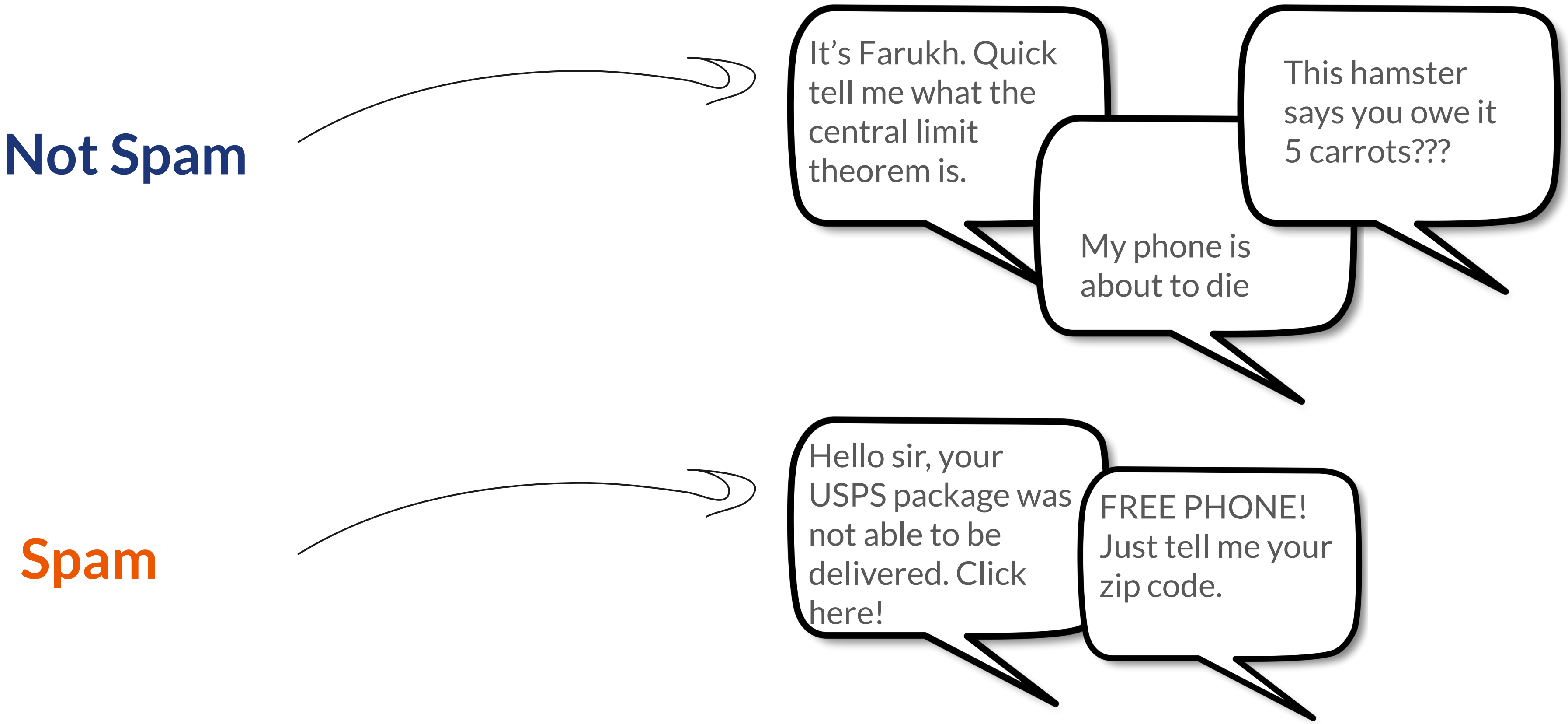
P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

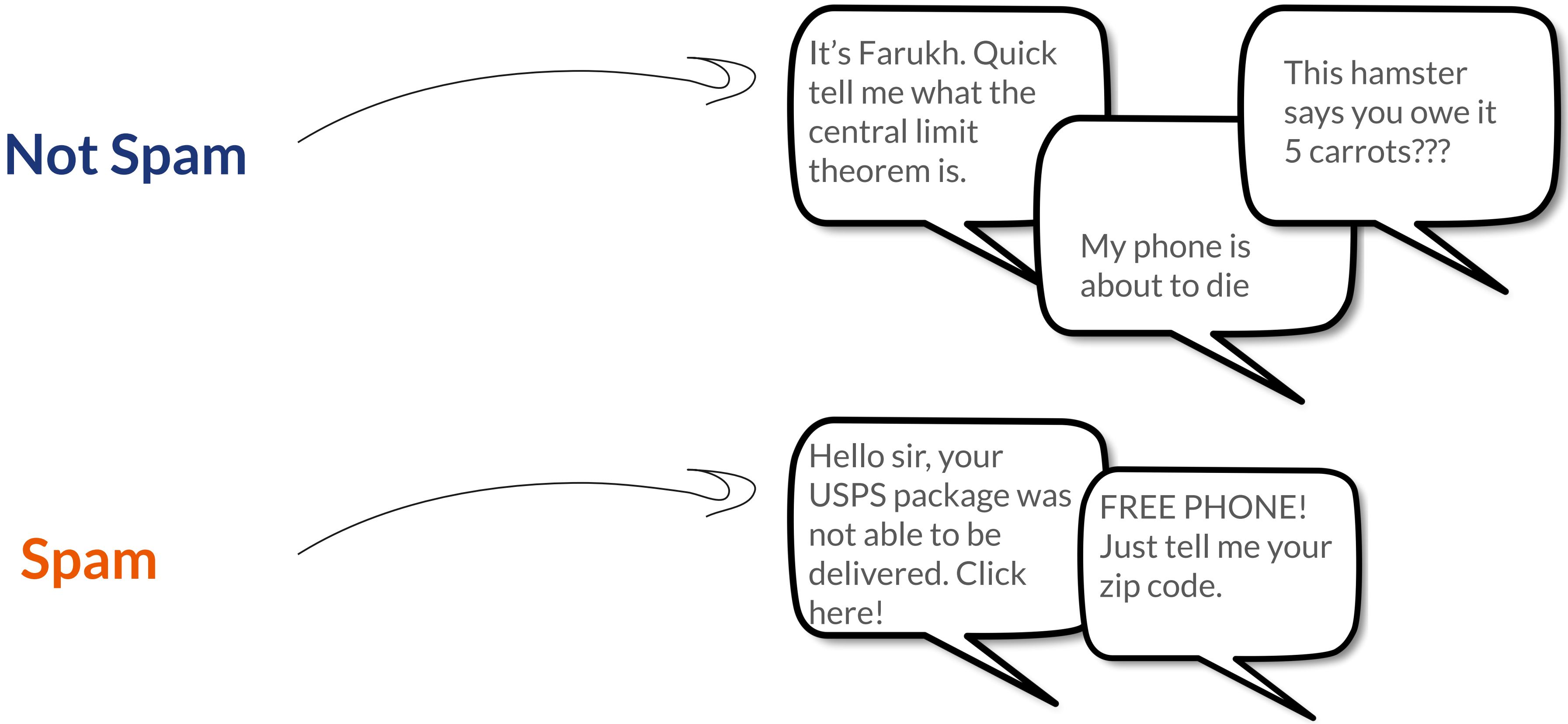
Not spam

$$P(NS|\text{"Phone"}) = \frac{0.198}{P(\text{"Phone"})}$$

Next, let's find out what the overall probability is of getting the word "Phone" in all our texts.



Here we look **across all classes** and calculate how many texts contain the word "phone". Looking at our dataset, **what is this proportion?**



Here we look **across all classes** and calculate how many texts contain the word "phone". Looking at our dataset, **what is this proportion? = ⅖**

Phone

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Not spam

$$P(NS|\text{"Phone"}) = \frac{0.198}{0.4}$$

Dividing these two numbers we get…

Phone

$P(S) = 0.4$

$P(NS) = 0.6$

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Not spam

$$P(NS|\text{"Phone"}) = \frac{0.198}{0.4}$$

$P(NS|\text{"Phone"}) = 0.495$

Dividing these two numbers we get…0.495

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Phone

Not spam

$$P(NS|\text{"Phone"}) = \frac{0.198}{0.4}$$

P(NS|"Phone") = 0.495

Spam

$$P(S|\text{"Phone"}) = \frac{P(S)\ P(\text{"Phone"}|S)}{P(\text{"Phone"})}$$

Next, let's check the probability that this message is "spam." What is the probability of **spam** in our dataset?

Phone

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Not spam

$$P(NS|\text{"Phone"}) = \frac{0.198}{0.4}$$

$$P(NS|\text{"Phone"}) = 0.495$$

Spam

$$P(S|\text{"Phone"}) = \frac{0.4 \; P(\text{"Phone"}|S)}{P(\text{"Phone"})}$$

What is the probability of the word "Phone" given that this message is "spam."

Phone

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Not spam

$$P(NS|\text{"Phone"}) = \frac{0.198}{0.4}$$

P(NS|"Phone") = 0.495

Spam

$$P(S|\text{"Phone"}) = \frac{0.4 * 0.5}{P(\text{"Phone"})}$$

What is the probability of the word "Phone" given that this message is "spam."

Phone

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Not spam

$$P(NS|\text{"Phone"}) = \frac{0.198}{0.4}$$

P(NS|"Phone") = 0.495

Spam

$$P(S|\text{"Phone"}) = \frac{0.4 * 0.5}{P(\text{"Phone"})}$$

Multiplying this together we get...

Phone

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Not spam

P(NS|”Phone”) =0.198

0.4

P(NS|”Phone”) = 0.495

Spam

P(S|”Phone”) = 0.2

P(“Phone”)

Multiplying this together we get…0.2

$P(S) = 0.4$

$P(NS) = 0.6$

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Phone

Not spam

$$P(NS|\text{"Phone"}) = \frac{0.198}{0.4}$$

$P(NS|\text{"Phone"}) = 0.495$

Spam

$$P(S|\text{"Phone"}) = \frac{0.2}{P(\text{"Phone"})}$$

What is the probability that we have the word “Phone” in our texts?

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Phone

Not spam

$$P(NS|\text{"Phone"}) = \frac{0.198}{0.4}$$

$$P(NS|\text{"Phone"}) = 0.495$$

Spam

$$P(S|\text{"Phone"}) = \frac{0.2}{0.4}$$

Again, **0.4.** Finally, what does this ratio evaluate to?

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Phone

Not spam

$$P(NS|\text{"Phone"}) = \frac{0.198}{0.4}$$

P(NS|"Phone") = 0.495

Spam

$$P(S|\text{"Phone"}) = \frac{0.2}{0.4}$$

P(S|"Phone") = 0.50

$\hat{y}$ = argmax

We get 0.5! Now, we have to take a look at these two values and decide, which class is **more confident**?

$P(S) = 0.4$

$P(NS) = 0.6$

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Phone

Not spam

$$P(NS|\text{"Phone"}) = \frac{0.198}{0.4}$$

$$P(NS|\text{"Phone"}) = 0.495$$

Spam

$$P(S|\text{"Phone"}) = \frac{0.2}{0.4}$$

$$P(S|\text{"Phone"}) = 0.50$$

$$\hat{y} = \text{argmax}$$

Since 0.5 > 0.495, we must state that this text message belongs to the **"spam" category.**

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Phone

Not spam

$$P(NS|\text{"Phone"}) = \frac{0.198}{0.4}$$

P(NS|"Phone") = 0.495

Spam

$$P(S|\text{"Phone"}) = \frac{0.2}{0.4}$$

P(S|"Phone") = 0.50

But! Let's take a look at some **unnecessary steps**. Given the P("Phone") is the same across classes, we are essentially taking an extra step that could be removed.

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33

Phone

Not spam

P(NS|"Phone") =0.198

Spam

P(S|"Phone") = 0.2

*As technologists, we are always in the mindset of conversing of steps, computational power, and memory

By removing division, we can get the same output with less steps.

If I do not divide by 0.4, I still come away with the same inequality across both classes:
0.2 > 0.198 aka **"Spam" > "Not Spam"**

P(S) = 0.4

P(NS) = 0.6

Phone USPS

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33
USPS	Yes	0.5
USPS	No	0

Let’s say we get the text “Phone USPS.” We also update our frequency table to observe the ratio of the word “USPS” in both spam & non-spam messages.

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33
USPS	Yes	0.5
USPS	No	0

Phone USPS

Not spam

P(NS|”Phone USPS”) =P(NS)*P(“Phone”|NS)*P(“USPS”|NS)

Can anyone calculate this?

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33
USPS	Yes	0.5
USPS	No	0

Phone USPS

Not spam

P(NS|"Phone USPS") = 0

This becomes "0." This might become a problem later...

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33
USPS	Yes	0.5
USPS	No	0

Phone USPS

Not spam

Spam

P(NS|”Phone USPS”) = 0

P(S|”Phone USPS”) =P(S)*P(“Phone”|S)*P(“USPS”|S)

Next, let’s calculate the proportional probability of spam.

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33
USPS	Yes	0.5
USPS	No	0

Phone USPS

Not spam

Spam

P(NS|"Phone USPS") = 0

P(S|"Phone USPS") = 0.1

Observing these two levels of confidence.
Which class will be **selected**?

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33
USPS	Yes	0.5
USPS	No	0

Phone USPS

Not spam

Spam

P(NS|"Phone USPS") = 0

P(S|"Phone USPS") = 0.1

This will be classified as a **Spam** text.

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33
USPS	Yes	0.5
USPS	No	0

Farukh
Phone Phone
Phone Phone

Not spam

Spam

P(NS|"Farukh...") = ...

P(S|"Farukh...") = ...

Let's say hackers have discovered my name and implant it in their text messages to you. **What will we always get for the "spam" category?**

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33
USPS	Yes	0.5
USPS	No	0

Farukh
Phone Phone
Phone Phone

Not spam

Spam

P(NS|"Farukh...") = 0.002

P(S|"Farukh...") = 0

We will always get **0!** **This is a problem** as values that are definitely spam (such as the word "Phone") will be ignored (or overpowered) by the presence of one non-spam text.

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0
Farukh	No	0.33
Phone	Yes	0.5
Phone	No	0.33
USPS	Yes	0.5
USPS	No	0

Farukh
Phone Phone
Phone Phone

Not spam

Spam

P(NS|"Farukh...") = 0.002

P(S|"Farukh...") = 0

To prevent this we implement a technique called **Laplace Smoothing**. We choose some "*alpha*" to add to **all of our frequencies** so that our predictions are never zeroed out. **We typically use 1**, but we can also use other values.

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	
Farukh	No	
Phone	Yes	
Phone	No	
USPS	Yes	
USPS	No	

(0+1)/(2+3)

(1+1)/(3+3)

(1+1)/(2+3)

(1+1)/(3+3)

(1+1)/(2+3)

(0+1)/(3+3)

Farukh
Phone Phone
Phone Phone

Not spam

P(NS|"Farukh...") = ...

Spam

P(S|"Farukh...") = ...

We add "1" to each original frequency. **Keep in mind we also need to update our denominator**, as we are adding more words to each category.

Farukh
Phone Phone
Phone Phone

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0.2
Farukh	No	0.33
Phone	Yes	0.4
Phone	No	0.33
USPS	Yes	0.4
USPS	No	0.16

1/5
2/6
2/5
2/6
2/5
1/6

Not spam

Spam

P(NS|"Farukh...") = ...

P(S|"Farukh...") = ...

However, we do not have to update our priors! This smoothing technique only applies to the frequency of each word, and **not the sample size.**

Laplace Smoothing - Summary

$$\hat{\theta}_i = \frac{x_i + \alpha}{N + \alpha d}$$

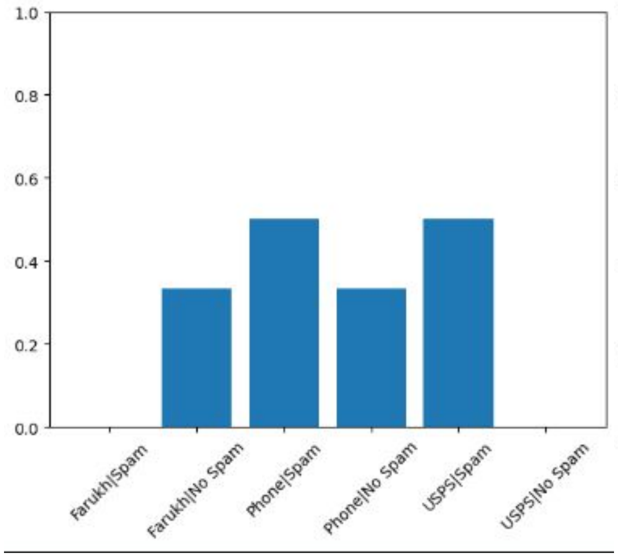
Laplace smoothing is a technique to “smooth” out frequencies and therefore eliminate 0 occurrences from our count.

This introduces a hyperparameter of “alpha” to our Naive Bayes Classifier

We add “*alpha*” occurrences to our count

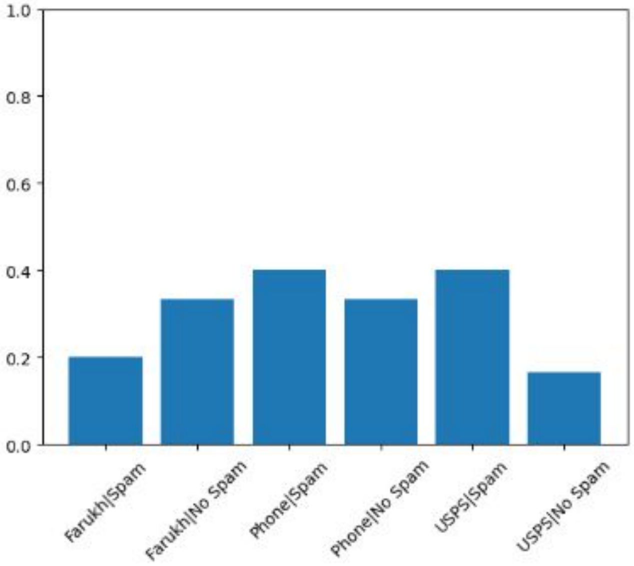
We add “*alpha*” * *d* occurrences to our denominator, where *d* is the number of unique words (aka dimensions) that we consider.

What do you think happens as we increase alpha?

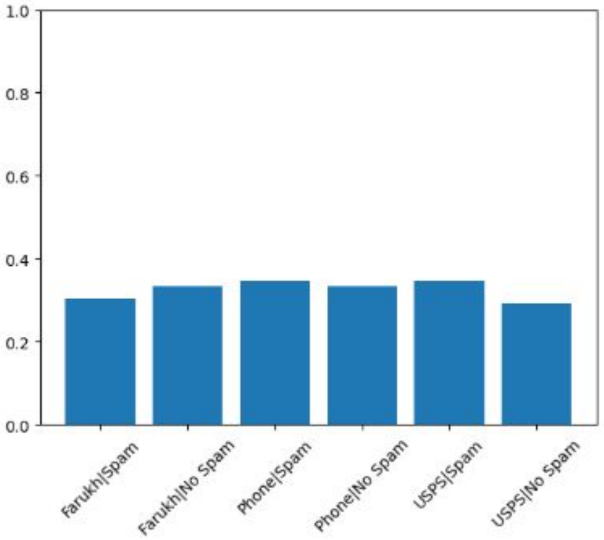


alpha = 0

(No laplace smoothing)

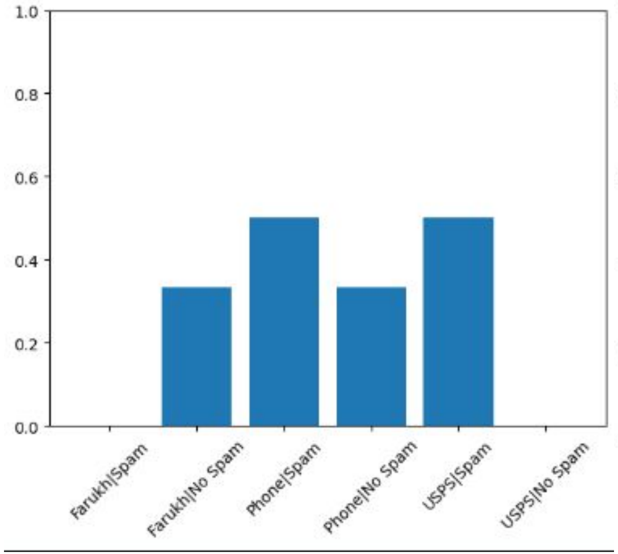


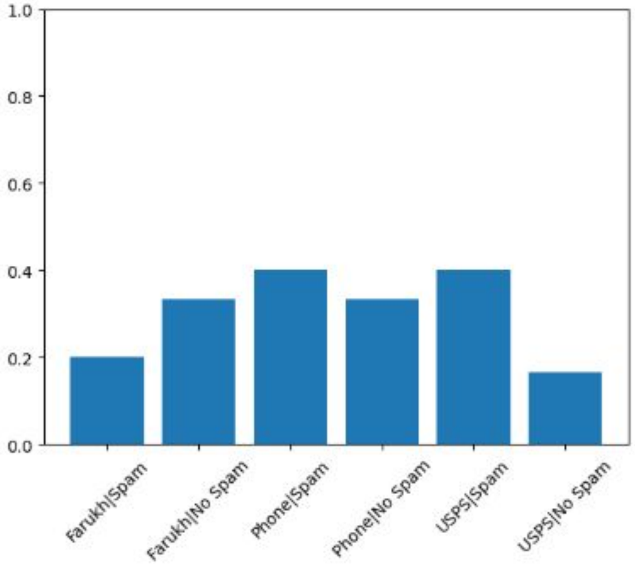
alpha = 2

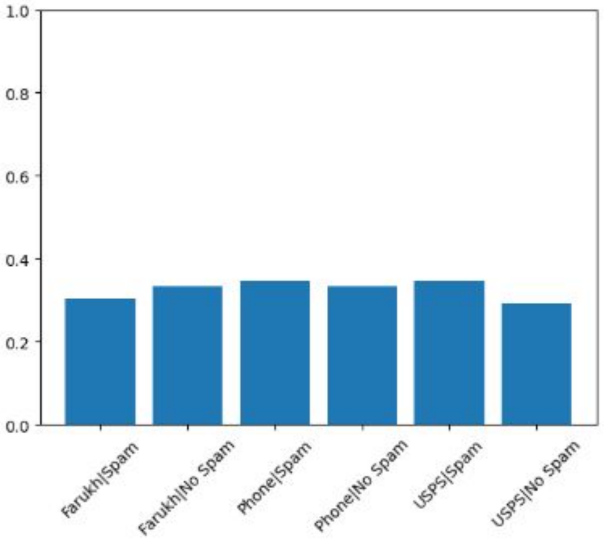


alpha = 7

Whenever you want a question answered, try it out yourself and see what happens. **What do you notice is happening to our probabilities as we increase alpha?**







alpha = 0

(No laplace smoothing)

alpha = 2

alpha = 7

Our probabilities become equal (**aka uniform distribution**). Hence the name laplace *smoothing*.

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0.2
Farukh	No	0.33
Phone	Yes	0.4
Phone	No	0.33
USPS	Yes	0.4
USPS	No	0.16

Farukh
Phone Phone
Phone Phone

Not spam

Spam

P(NS|"Farukh...") = ...

P(S|"Farukh...") = ...

Now that I have new frequencies to work with, what will be my new calculations?

P(S) = 0.4

P(NS) = 0.6

Word	Spam	Ratio
Farukh	Yes	0.2
Farukh	No	0.33
Phone	Yes	0.4
Phone	No	0.33
USPS	Yes	0.4
USPS	No	0.16

Farukh
Phone Phone
Phone Phone

Not spam

Spam

P(NS|"Farukh...") = 0.00391

P(S|"Farukh...") = 0.005

Amazing, we can now better predict spam messages through this simple transformation. Thank you Laplace.

The Meaning of “Naive”

Naive Bayes Classifier

In maths, we call something **naive** when we simply assume something to be true without making room for nuance.

This is different from the more common definition of naive, which means "lacking experience, judgement, or wisdom."

Instead, we make assumptions to reveal **positive qualities.**

Naive Bayes Classifier

The **Naive Bayes Classifier** is naive because it does not consider dependence between predictors (something very important for natural language!). Consider if these two statements are the same:

- *“They served chicken to the guests.”*
- *“They served guests to the chicken.”*

Naive Bayes Classifier

- *"They served chicken to the guests."*
- *"They served guests to the chicken."*

The naive bayes classifier states that both these statements are the same! This is the primary assumption of naive bayes classifiers:

"All predictors are independent"

However, we can still get **excellent classification results** for **cheap**.

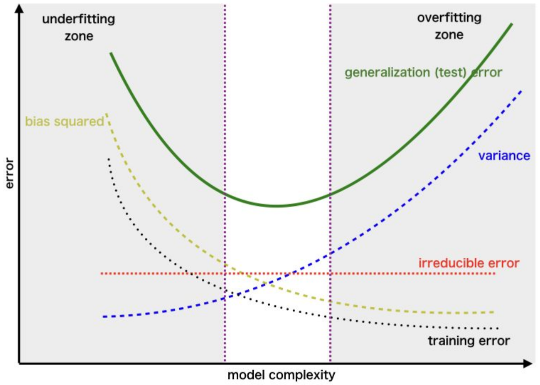
This makes it ideal for large datasets, but less than ideal when working with datasets that contain complex relationships entailing dependence.

Naive Bayes Classifier

This means that naive bayes classifier **introduces bias**, but also **reduces variance**.

This leads to a classifier that operates quite *well as a result of the bias-variance tradeoff*.

This makes it an ideal “**baseline**” comparative classifier.



Multiclass Classification

Naive Bayes Classifier

We can utilize our formula in a **multiclass supervised learning classifier** called the **naive bayes classifier**.

You might be wondering:

“Hey Anil/Farukh, you told me that Naive Bayes is multiclass! I only saw 2 classes (binary).”

P(J) = 0.33

P(M) = 0.33

P(F) = 0.33

Want to grab tea?

Word	Class	Ratio
Coffee	Jonnathan	0.5
Coffee	Mickal	0.2
Coffee	Farukh	0.1
Tea	Jonnathan	0.2
Tea	Mickal	0.3
Tea	Farukh	0.4

We can include **any amount of classes** in our dataset. As long as we **compute the conditional probability for each class**, we can **estimate if a sample belongs to a specific point.**

Word	Class	Ratio
Coffee	Jonnathan	0.5
Coffee	Mickal	0.2
Coffee	Farukh	0.1
Tea	Jonnathan	0.2
Tea	Mickal	0.3
Tea	Farukh	0.4

P(F|”Tea?”) = …

P(M|”Tea?”) = …

P(J|”Tea?”) = …

Can anyone calculate the probability of this text being sent from Jonnathan, Mickal, or Farukh?

Want to grab tea?

Word	Class	Ratio
Coffee	Jonnathan	0.5
Coffee	Mickal	0.2
Coffee	Farukh	0.1
Tea	Jonnathan	0.2
Tea	Mickal	0.3
Tea	Farukh	0.4

P(F|”Tea?”) = P(J)P(“Tea”|F)

P(F|”Tea?”) = 0.33 * 0.4

P(F|”Tea?”) = 0.132

P(M|”Tea?”) = P(M)P(“Tea”|M)

P(M|”Tea?”) = 0.33 * 0.3

P(M|”Tea?”) = 0.099

P(J|”Tea?”) = P(J)P(“Tea”|J)

P(J|”Tea?”) = 0.33 * 0.1

P(J|”Tea?”) = 0.033

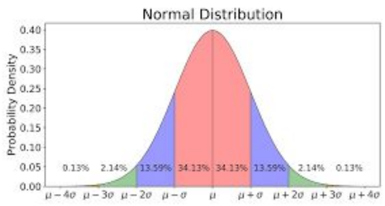
Based on these values, who is the likely source of this text?

Distributions & Naive Bayes

Distributions & Naive Bayes

We can take this a step further and classify datasets that contain different types of predictors by assuming distributions, this includes:

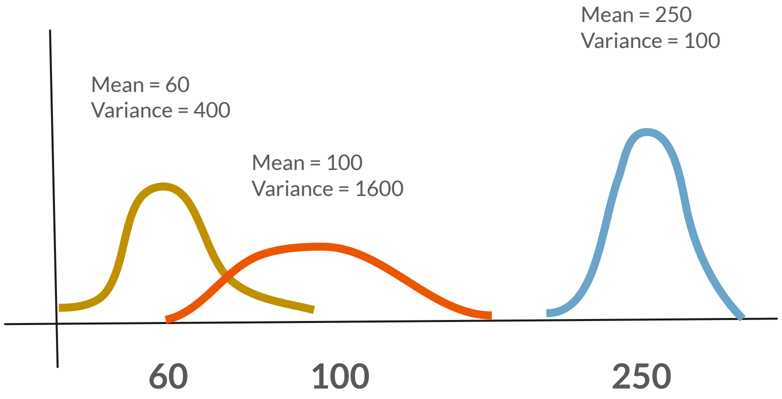
- ***Gaussian Naive Bayes*** : assume predictor draws from **normal distribution**
- ***Multinomial Naive Bayes*** : the “text message” example we just worked through



kibble_grams	noise_dB	animal
200	40	cat
250	60	dog
115	45	cat
300	80	dog
50	75	hamster

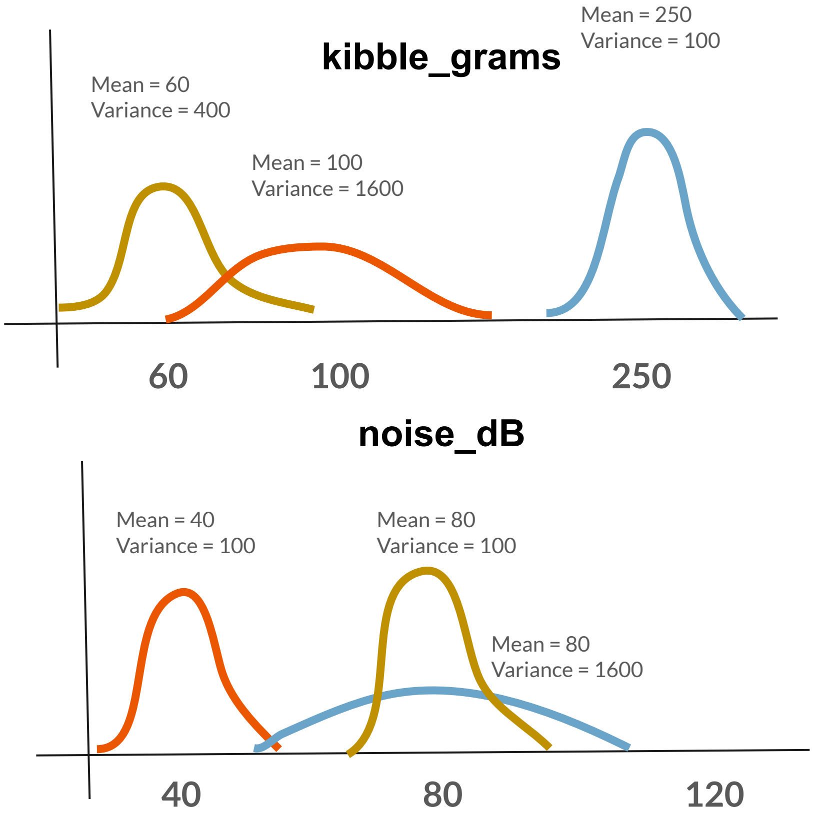
Instead of counting frequency and proportion, we assume that each predictor originates from an **independent gaussian distribution** with its own **mean** and **variance** across all dimensions.

kibble_grams	noise_dB	animal
200	40	cat
250	60	dog
115	45	cat
300	80	dog
50	75	hamster



For example, for "**kibble_grams**", cats, dogs, and hamsters all have a different normal distribution. We **estimate their mean and variance using the maximum likelihood estimate.** We'll skip over this for now for simplicity.

kibble_grams	noise_dB	animal
200	40	cat
250	60	dog
115	45	cat
300	80	dog
50	75	hamster



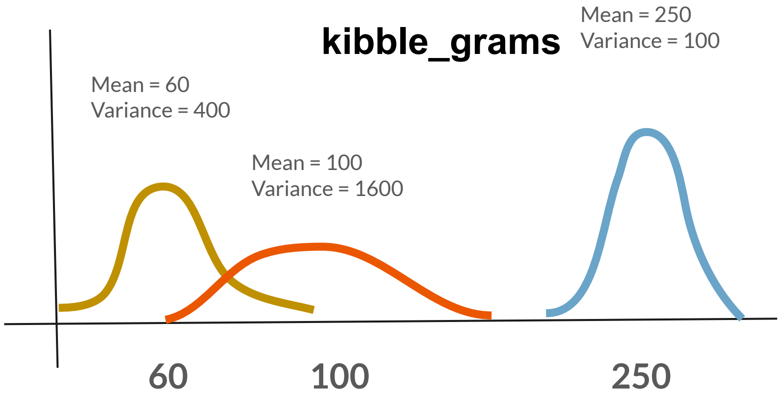
As well as “**noise_DB.**” These graphs are super rough and their only purpose is for light exploration.

unknown animal

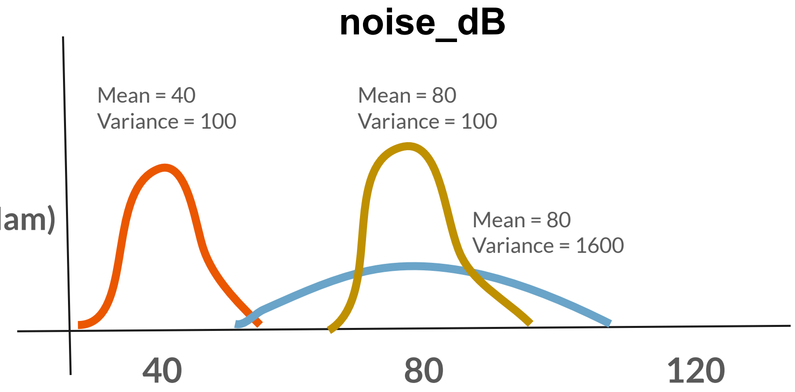
kibble_grams = 80

noise_DB = 35

P(Cat|kibb=80; noise=35) = P(Cat) P(kibb=80| Cat) P(noise=35| Cat)



P(Dog|kibb=80; noise=35) = P(Dog) P(kibb=80| Dog) P(noise=35| Dog)



P(Ham|kibb=80; noise=35) = P(Hamst) P(kibb=80| Ham) P(noise=35| Ham)

When we receive a new test observation. We can utilize bayes theorem once again to calculate the conditional probability that this is a dog, cat, or hamster.

unknown animal

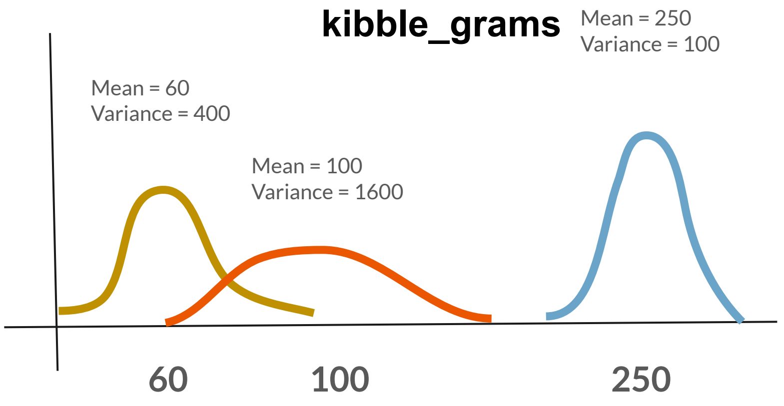
kibble_grams = 80

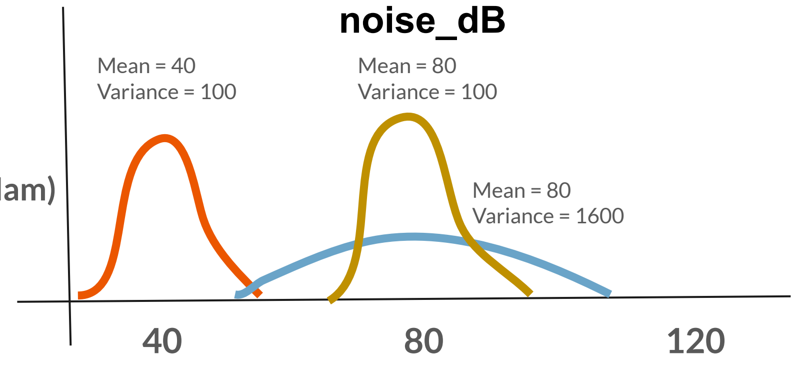
noise_DB = 35

P(Cat|kibb=80; noise=35) = P(Cat) P(kibb=80| Cat) P(noise=35| Cat)

P(Dog|kibb=80; noise=35) = P(Dog) P(kibb=80| Dog) P(noise=35| Dog)

P(Ham|kibb=80; noise=35) = P(Hamst) P(kibb=80| Ham) P(noise=35| Ham)





But wait, how do we calculate P(Cat), P(Dog), or P(Hamst)? Think back to the text example...

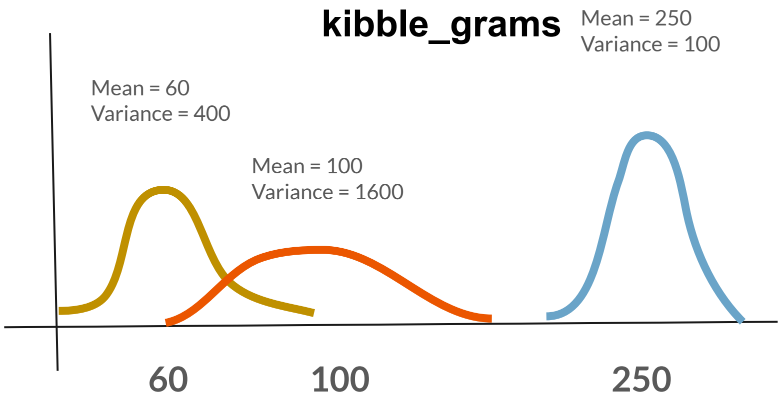
unknown animal

kibble_grams = 80

noise_DB = 35

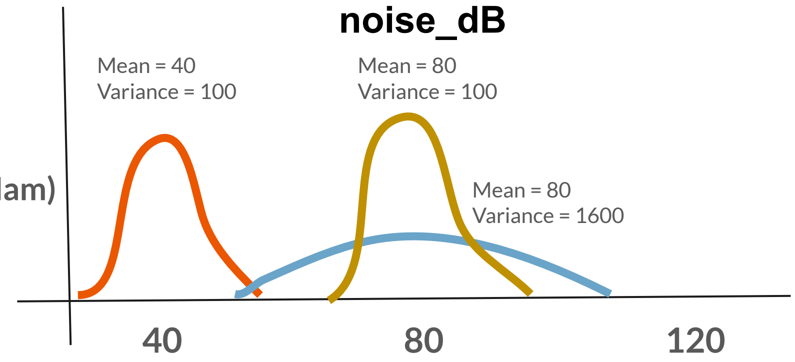
P(Cat|kibb=80; noise=35) = P(Cat) P(kibb=80| Cat) P(noise=35| Cat)

= (⅖) P(kibb=80| Cat) P(noise=35| Cat)



P(Dog|kibb=80; noise=35) = P(Dog) P(kibb=80| Dog) P(noise=35| Dog)

= (⅖) P(kibb=80| Dog) P(noise=35| Dog)



P(Ham|kibb=80; noise=35) = P(Hamst) P(kibb=80| Ham) P(noise=35| Ham)

= (⅕) P(kibb=80| Ham) P(noise=35| Ham)

Keep in mind that we simply get the ratio of cats, or dogs, or hamsters in our current dataset, and use that as our **prior belief. Looking back to our dataset** we ⅖ cats, ⅖ dogs, and ⅕ hamsters.

unknown animal

kibble_grams = 80

noise_DB = 35

P(Cat|kibb=80; noise=35) = P(Cat) P(kibb=80| Cat) P(noise=35| Cat)

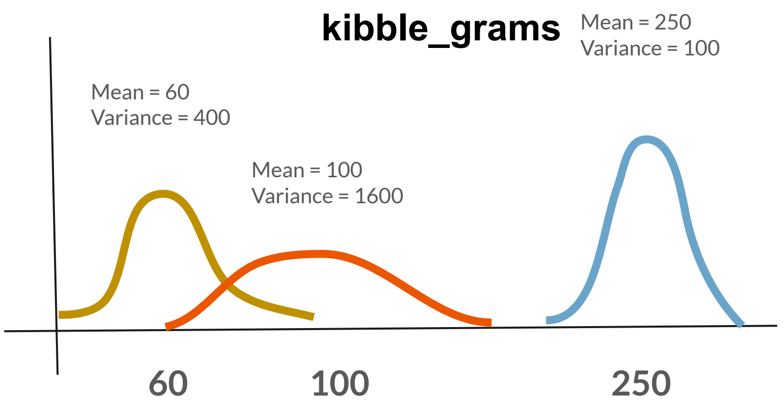
= (2/5) P(kibb=80| Cat) P(noise=35| Cat)

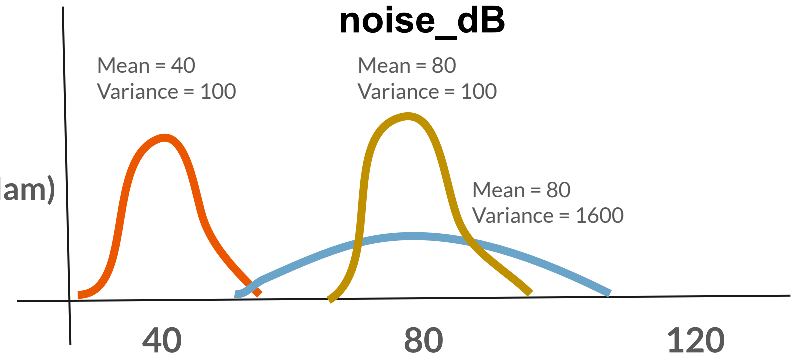
P(Dog|kibb=80; noise=35) = P(Dog) P(kibb=80| Dog) P(noise=35| Dog)

= (2/5) P(kibb=80| Dog) P(noise=35| Dog)

P(Ham|kibb=80; noise=35) = P(Hamst) P(kibb=80| Ham) P(noise=35| Ham)

= (1/5) P(kibb=80| Ham) P(noise=35| Ham)





For this next part, we estimate something called "likelihood" using a **probability density function** that comes with the **assumption of normal distributions**.

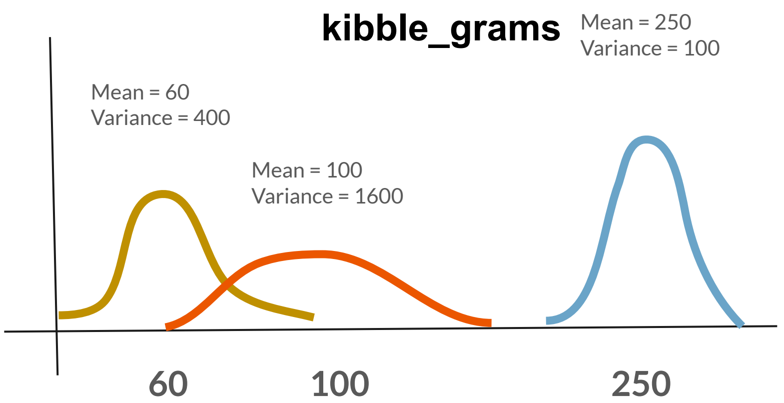
unknown animal

kibble_grams = 80

noise_DB = 35

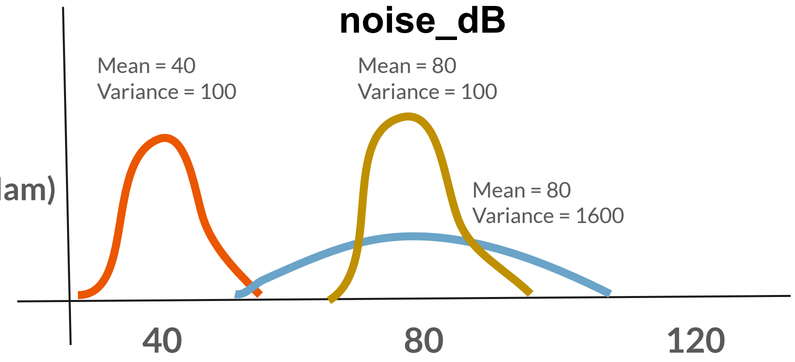
P(Cat|kibb=80; noise=35) = P(Cat) P(kibb=80| Cat) P(noise=35| Cat)

= (2/5) P(kibb=80| Cat) P(noise=35| Cat)



P(Dog|kibb=80; noise=35) = P(Dog) P(kibb=80| Dog) P(noise=35| Dog)

= (2/5) P(kibb=80| Dog) P(noise=35| Dog)



P(Ham|kibb=80; noise=35) = P(Hamst) P(kibb=80| Ham) P(noise=35| Ham)

= (1/5) P(kibb=80| Ham) P(noise=35| Ham)

Don't get lost in the details. Notice that we only need mean and variance to calculate this value. Similar to logistic regression, we use **MLE to find the mean and variance which maximizes this value.**

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

unknown animal

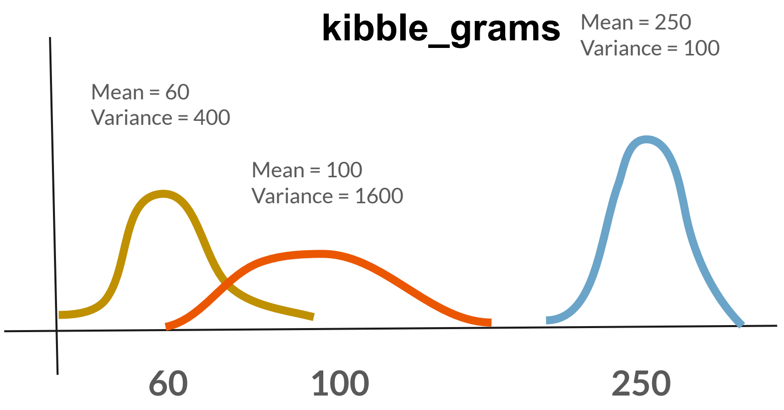
kibble_grams = 80

noise_DB = 35

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

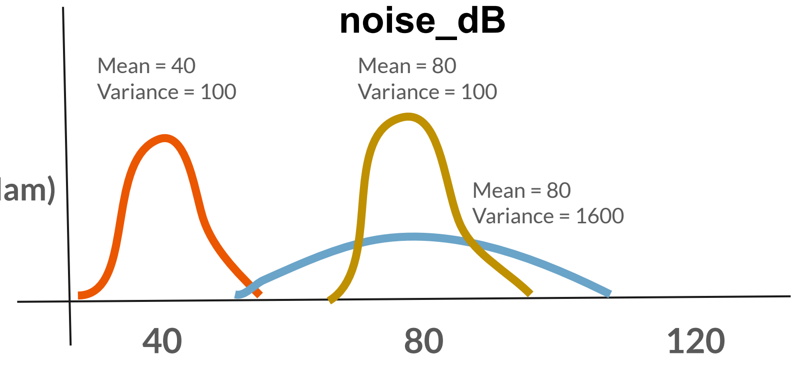
P(Cat|kibb=80; noise=35) = P(Cat) P(kibb=80| Cat) P(noise=35| Cat)

= (2/5) P(kibb=80| Cat) P(noise=35| Cat)



P(Dog|kibb=80; noise=35) = P(Dog) P(kibb=80| Dog) P(noise=35| Dog)

= (2/5) P(kibb=80| Dog) P(noise=35| Dog)



P(Ham|kibb=80; noise=35) = P(Hamst) P(kibb=80| Ham) P(noise=35| Ham)

= (1/5) P(kibb=80| Ham) P(noise=35| Ham)

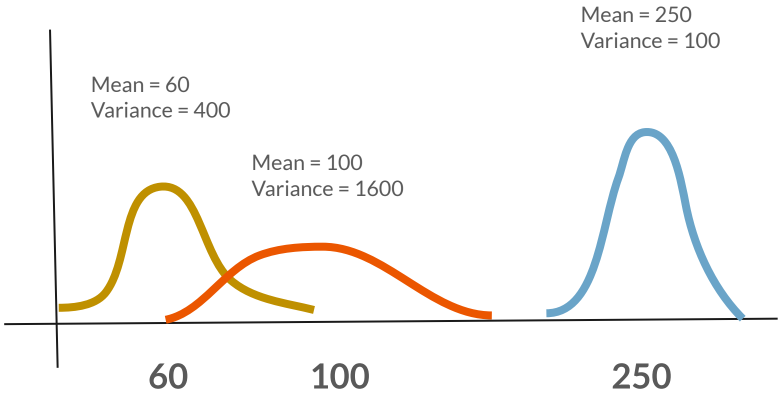
From a more abstract perspective, this calculates the **corresponding y-value** of our probability distribution graph for each class. This is called **likelihood**.

unknown animal

kibble_grams = 80

noise_DB = 35

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



P(Cat|kibb=80; noise=35) = P(Cat) P(kibb=80| Cat) P(noise=35| Cat)

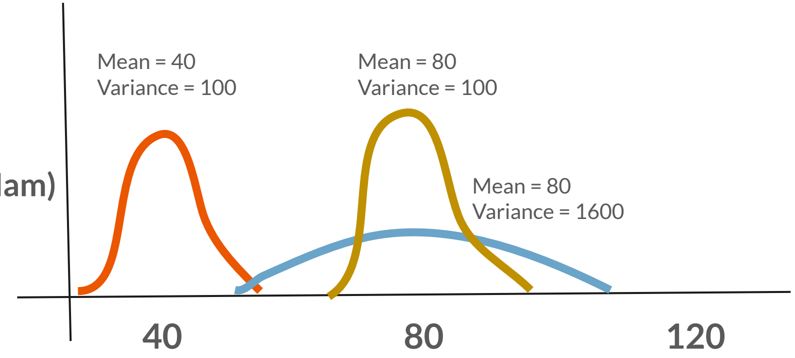
= (⅖) P(kibb=80| Cat) P(noise=35| Cat)

= (⅖) (0.0088) (0.035)

P(Dog|kibb=80; noise=35) = P(Dog) P(kibb=80| Dog) P(noise=35| Dog)

= (⅖) P(kibb=80| Dog) P(noise=35| Dog)

= (⅖) (0.0) (0.005)



P(Ham|kibb=80; noise=35) = P(Hamst) P(kibb=80| Ham) P(noise=35| Ham)

= (⅕) P(kibb=80| Ham) P(noise=35| Ham)

= (⅕) (0.012) (0.000001)

For simplicity, I use this calculator: https://www.danielsoper.com/statcalc/calculator.aspx?id=54

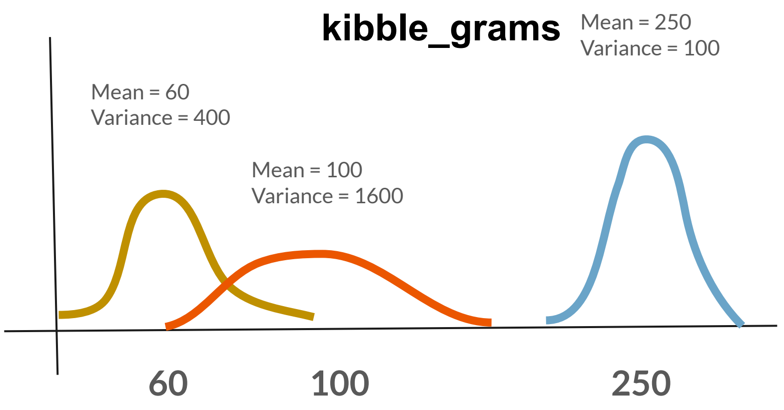
We get the values above.

unknown animal

kibble_grams = 80

noise_DB = 35

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



P(Cat|kibb=80; noise=35) = P(Cat) P(kibb=80| Cat) P(noise=35| Cat)

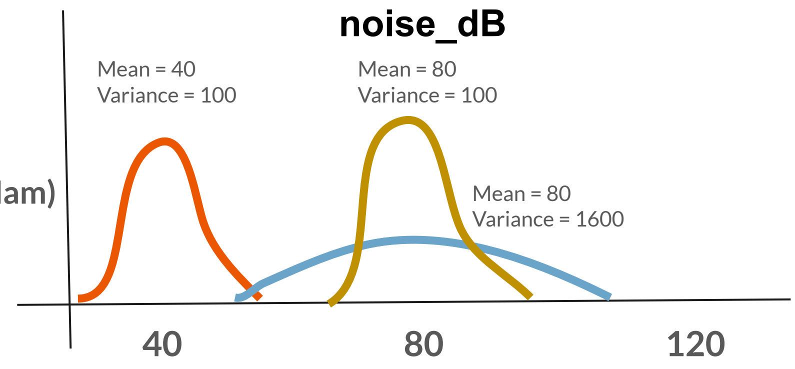
= (2/5) P(kibb=80| Cat) P(noise=35| Cat)

= (2/5) (0.0088) (0.035)

P(Dog|kibb=80; noise=35) = P(Dog) P(kibb=80| Dog) P(noise=35| Dog)

= (2/5) P(kibb=80| Dog) P(noise=35| Dog)

= (2/5) (0.000...) (0.005)



P(Ham|kibb=80; noise=35) = P(Hamst) P(kibb=80| Ham) P(noise=35| Ham)

= (1/5) P(kibb=80| Ham) P(noise=35| Ham)

= (1/5) (0.012) (0.000001)

Now here's a very real practical problem that we need to deal with. Computers are finite beings. This introduces the problems of floating point numbers and **underflow**? There is a very real possibility that the values we get will be "0.000000000000001." **This will result in an error.**

unknown animal

kibble_grams = 80

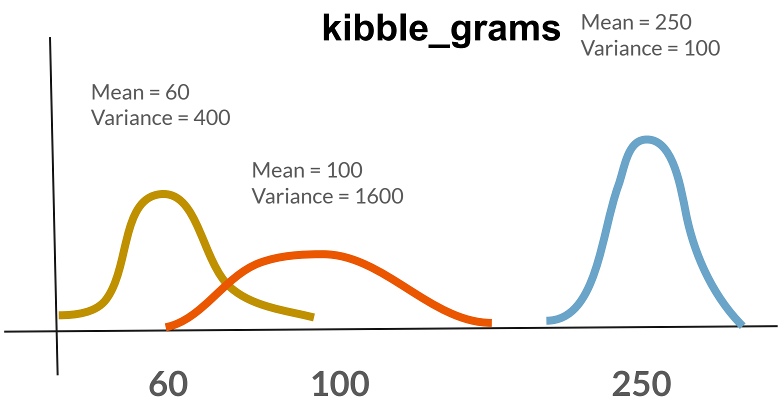
noise_DB = 35

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

P(Cat|kibb=80; noise=35) = P(Cat) P(kibb=80| Cat) P(noise=35| Cat)

= (⅖) P(kibb=80| Cat) P(noise=35| Cat)

= (⅖) (0.0088) (0.035)



P(Dog|kibb=80; noise=35) = P(Dog) P(kibb=80| Dog) P(noise=35| Dog)

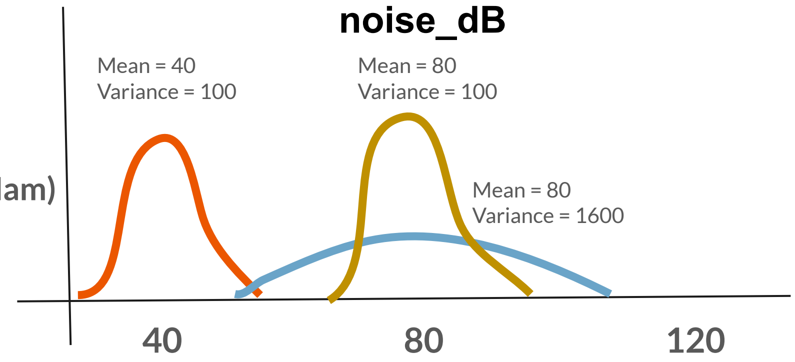
= (⅖) P(kibb=80| Dog) P(noise=35| Dog)

= (⅖) (0.000...) (0.005)

P(Ham|kibb=80; noise=35) = P(Hamst) P(kibb=80| Ham) P(noise=35| Ham)

= (⅕) P(kibb=80| Ham) P(noise=35| Ham)

= (⅕) (0.012) (0.000001)



Therefore, we utilize the "natural log" *ln()* to convert these values into "manageable" values.

unknown animal

kibble_grams = 80

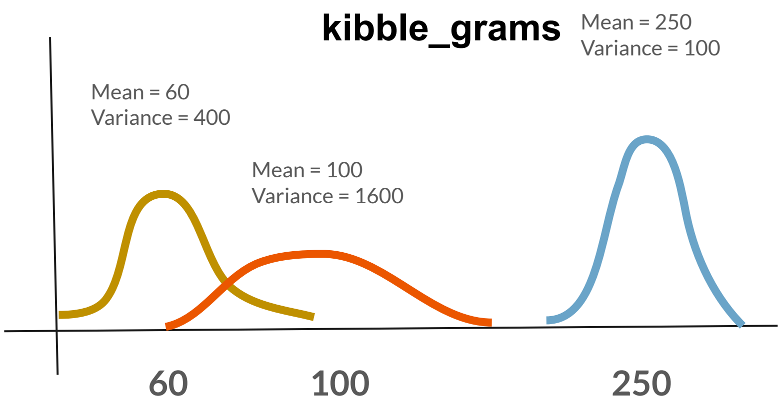
noise_DB = 35

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

P(Cat|kibb=80; noise=35) = P(Cat) P(kibb=80| Cat) P(noise=35| Cat)

= (2/5) P(kibb=80| Cat) P(noise=35| Cat)

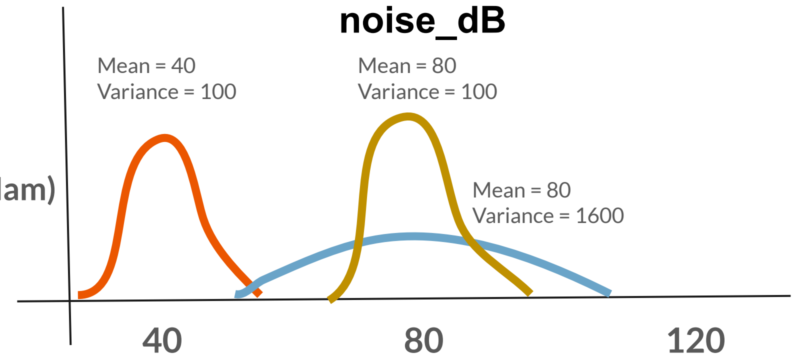
= ln((2/5) (0.0088) (0.035))



P(Dog|kibb=80; noise=35) = P(Dog) P(kibb=80| Dog) P(noise=35| Dog)

= (2/5) P(kibb=80| Dog) P(noise=35| Dog)

= ln((2/5) (0.000...) (0.005))



P(Ham|kibb=80; noise=35) = P(Hamst) P(kibb=80| Ham) P(noise=35| Ham)

= (1/5) P(kibb=80| Ham) P(noise=35| Ham)

= ln((1/5) (0.012) (0.000001))

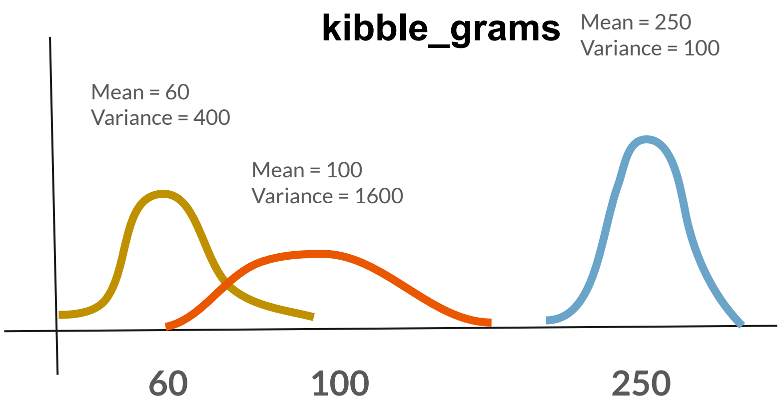
Therefore, we utilize the “natural log” *ln()* to convert these values into “manageable” values.

unknown animal

kibble_grams = 80

noise_DB = 35

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



P(Cat|kibb=80; noise=35) = P(Cat) P(kibb=80| Cat) P(noise=35| Cat)

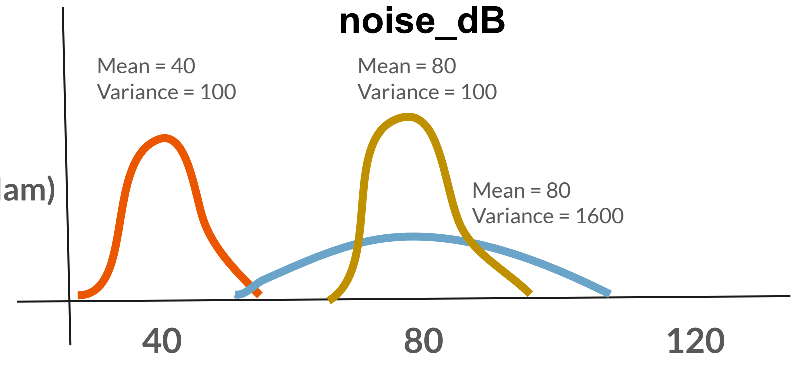
= (⅖) P(kibb=80| Cat) P(noise=35| Cat)

= ln(⅖) + ln(0.0088) + ln(0.035)

P(Dog|kibb=80; noise=35) = P(Dog) P(kibb=80| Dog) P(noise=35| Dog)

= (⅖) P(kibb=80| Dog) P(noise=35| Dog)

= ln(⅖) + ln (0.000...) + ln(0.005)



P(Ham|kibb=80; noise=35) = P(Hamst) P(kibb=80| Ham) P(noise=35| Ham)

= (⅕) P(kibb=80| Ham) P(noise=35| Ham)

= ln(⅕) + ln(0.012) + ln(0.000001)

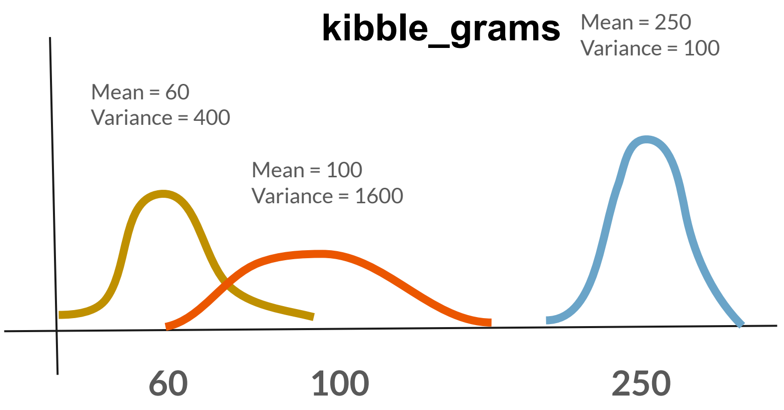
According to “log” rules, this becomes **addition**.

unknown animal

kibble_grams = 80

noise_DB = 35

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



P(Cat|kibb=80; noise=35) = P(Cat) P(kibb=80| Cat) P(noise=35| Cat)

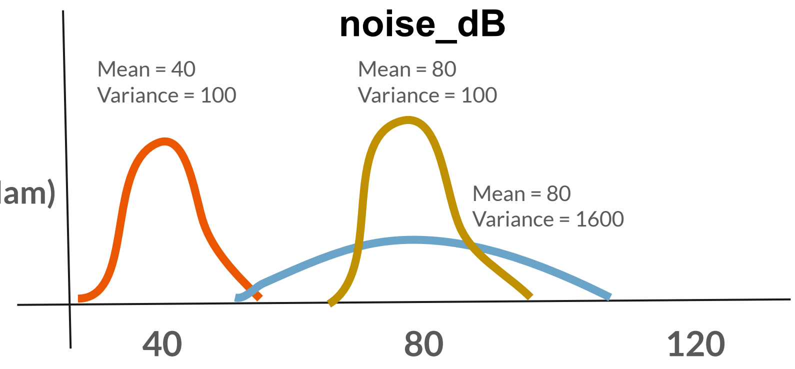
= (⅖) P(kibb=80| Cat) P(noise=35| Cat)

= -0.91 + -4.73 + -3.35

P(Dog|kibb=80; noise=35) = P(Dog) P(kibb=80| Dog) P(noise=35| Dog)

= (⅖) P(kibb=80| Dog) P(noise=35| Dog)

= -0.91+ -25.32 + -5.29



P(Ham|kibb=80; noise=35) = P(Hamst) P(kibb=80| Ham) P(noise=35| Ham)

= (⅕) P(kibb=80| Ham) P(noise=35| Ham)

= -1.60 + -4.42 +-13.81

We evaluate these values. Alright, we did the hard part. **Someone else please add this up.**

unknown animal

kibble_grams = 80

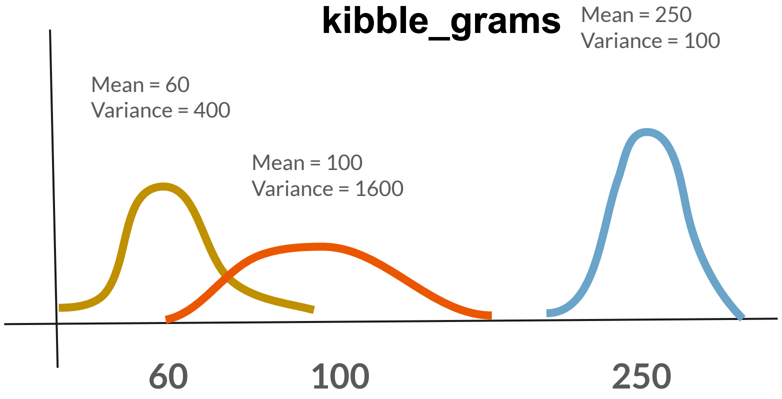
noise_DB = 35

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

P(Cat|kibb=80; noise=35) = P(Cat) P(kibb=80| Cat) P(noise=35| Cat)

= (2/5) P(kibb=80| Cat) P(noise=35| Cat)

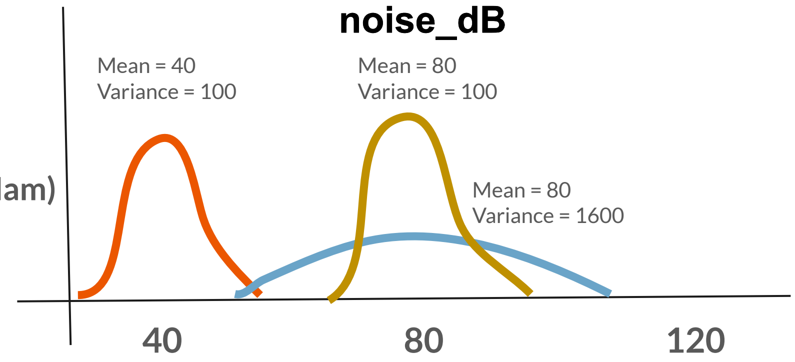
= -8.99



P(Dog|kibb=80; noise=35) = P(Dog) P(kibb=80| Dog) P(noise=35| Dog)

= (2/5) P(kibb=80| Dog) P(noise=35| Dog)

= -31.52



P(Ham|kibb=80; noise=35) = P(Hamst) P(kibb=80| Ham) P(noise=35| Ham)

= (1/5) P(kibb=80| Ham) P(noise=35| Ham)

=-19.83

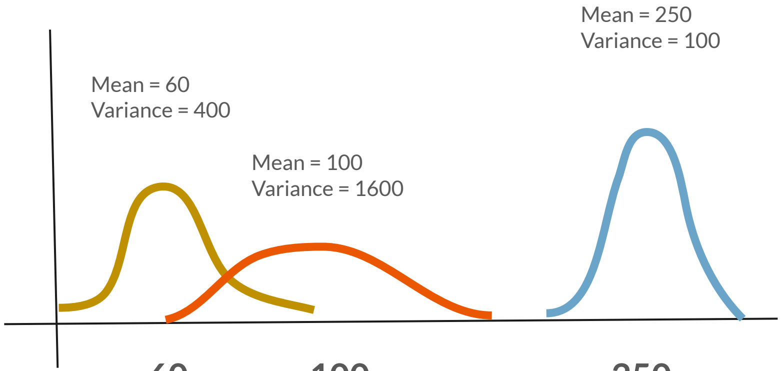
We choose the largest value as our class. **Which one is the largest value?????**

unknown animal

kibble_grams = 80

noise_DB = 35

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



P(Cat|kibb=80; noise=35) = P(Cat) P(kibb=80| Cat) P(noise=35| Cat)

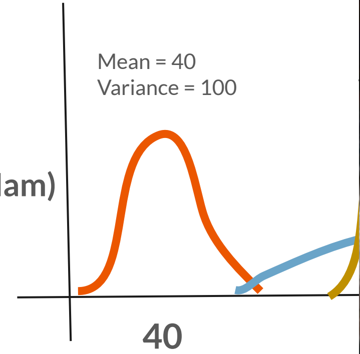
= (⅖) P(kibb=80| Cat) P(noise=35| Cat)

= -8.99

P(Dog|kibb=80; noise=35) = P(Dog) P(kibb=80| Dog) P(noise=35| Dog)

= (⅖) P(kibb=80| Dog) P(noise=35| Dog)

= -31.52



P(Ham|kibb=80; noise=35) = P(Hamst) P(kibb=80| Ham) P(noise=35| Ham)

= (⅕) P(kibb=80| Ham) P(noise=35| Ham)

=-19.83

We choose the largest value as our class. **Which one is the largest value?????**

Naive Bayes Theorem

To conclude our conversation on the supervised learning classifier Naive Bayes, it is a **powerful parametric supervised learning algorithm that utilizes bayes theorem to classify samples**.

Pros

- No **optimization** involved
- Works well with **large** datasets
- Excellent **baseline** classifier
- **Assumption of independence** simplifies training

Cons

- Sensitive to **class imbalance**
- Need to store **entire training dataset for prediction**
- **Assumption of independence** might miss out on predictive capabilities

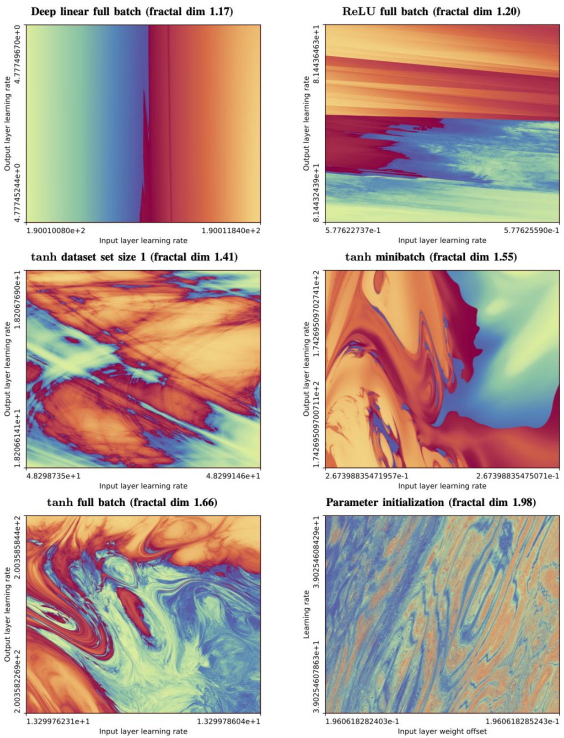
Tomorrow

K-Nearest Neighbors

- Who is my neighbor?
- What is a manhattan distance?

kNN Hyperparameters

- What happens if we increase/decrease k?
- Where does variance/bias exist in kNN?



[*Neural network training makes beautiful fractals*]()