1. Consider a Naive Bayes model trained using the following familiar dataset:

ID	*Outl*	*Temp*	*Humi*	*Wind*	PLAY
A	s	h	n	F	N
B	s	h	h	T	N
C	o	h	h	F	Y
D	r	m	h	F	Y
E	r	c	n	F	Y
F	r	c	n	T	N

Suppose that you made additional observations of days and their features. But you don't have the label for the PLAY in these days:

ID	*Outl*	*Temp*	*Humi*	*Wind*	PLAY
G	o	m	n	T	?
H	s	m	h	F	?

How could you incorporate this information into your Naïve Bayes model? If necessary, recompute your model parameters.

Self-training ⇒ get more training data

model

1. Train the learner on currently-labelled instances
2. Use the learner to predict the label of the unlabelled instances
3. Where the learner is very confident (e.g. high probability), add newly-labelled (predicted) to the training set.
4. Repeat (from step 1) until all instances are labelled or no instances can be labelled confidently.

Current NB:

For step (a) let's train our model using the labelled instances. In NB we need the probability of each label (the prior probabilities):

$P(Play = Y) = \frac{1}{2}$ $P(Play = N) = \frac{1}{2}$

And all the conditional probabilities between the labels of class (PLAY) and other attribute values.

$P(Outl = s \mid N) = \frac{2}{3}$ $P(Outl = o \mid N) = 0$ $P(Outl = r \mid N) = \frac{1}{3}$

$P(Outl = s \mid Y) = 0$ $P(Outl = o \mid Y) = \frac{1}{3}$ $P(Outl = r \mid Y) = \frac{2}{3}$

$P(Temp = h \mid N) = \frac{2}{3}$ $P(Temp = m \mid N) = 0$ $P(Temp = c \mid N) = \frac{1}{3}$

$P(Temp = h \mid Y) = \frac{1}{3}$ $P(Temp = m \mid Y) = \frac{1}{3}$ $P(Temp = c \mid Y) = \frac{1}{3}$

$P(Humi = n \mid N) = \frac{2}{3}$ $P(Humi = h \mid N) = \frac{1}{3}$

$P(Humi = n \mid Y) = \frac{1}{3}$ $P(Humi = h \mid Y) = \frac{2}{3}$

$P(Wind = T \mid N) = \frac{2}{3}$ $P(Wind = F \mid N) = \frac{1}{3}$

$P(Wind = T \mid Y) = 0$ $P(Wind = F \mid Y) = 1$

Predict G: (with Laplace smoothing)

class N: $P(N) \times P(Outl=o|N) \times P(Temp=m|N) \times P(Humi=n|N) \times P(Wind=T|N)$

$= \frac{1}{2} \times \frac{0+1}{3+3} \times \frac{0+1}{3+3} \times \frac{2+1}{3+2} \times \frac{2+1}{3+2} = 0.005$

class Y: $P(Y) \times P(Outl=o|Y) \times P(Temp=m|Y) \times P(Humi=n|Y) \times P(Wind=T|Y)$

$= \frac{1}{2} \times \cdots = 0.004$

⇒ Predict N with 0.005 confidence

Predict H: (with Laplace smoothing)

class N: $P(N) \times P(Outl=s|N) \times P(Temp=m|N) \times P(Humi=h|N) \times P(Wind=F|N)$

$= 0.007$

class Y: $P(Y) \times P(Outl=s|Y) \times P(Temp=m|Y) \times P(Humi=h|Y) \times P(Wind=F|Y)$

$= 0.013$

⇒ Predict Y with 0.013 confidence

Assume confidence threshold = 0.01:

G: $0.005 < 0.01$ ⇒ not confident

H: $0.013 > 0.01$ ⇒ add to training set! (label: Y)

Repeat!

New NB params:

This time our prior probabilities will be

$P(Play = Y) = \frac{4}{7}$ $P(Play = N) = \frac{3}{7}$

Our conditional probabilities will also change to

$P(Outl = s \mid N) = \frac{2}{3}$	$P(Outl = o \mid N) = 0$	$P(Outl = r \mid N) = \frac{1}{3}$
$P(Outl = s \mid Y) = \frac{1}{4}$	$P(Outl = o \mid Y) = \frac{1}{4}$	$P(Outl = r \mid Y) = \frac{2}{4}$
$P(Temp = h \mid N) = \frac{2}{3}$	$P(Temp = m \mid N) = 0$	$P(Temp = c \mid N) = \frac{1}{3}$
$P(Temp = h \mid Y) = \frac{1}{4}$	$P(Temp = m \mid Y) = \frac{2}{4}$	$P(Temp = c \mid Y) = \frac{1}{4}$
$P(Humi = n \mid N) = \frac{2}{3}$	$P(Humi = h \mid N) = \frac{1}{3}$	
$P(Humi = n \mid Y) = \frac{1}{4}$	$P(Humi = h \mid Y) = \frac{3}{4}$	
$P(Wind = T \mid N) = \frac{2}{3}$	$P(Wind = F \mid N) = \frac{1}{3}$	
$P(Wind = T \mid Y) = 0$	$P(Wind = F \mid Y) = 1$	

Predict G: (with Laplace smoothing)

class N: $P(N) \times P(Outl=o|N) \times P(Temp=m|N) \times P(Humi=n|N) \times P(wind=T|N)$

$= \frac{3}{7} \times \frac{0+1}{3+3} \times \frac{0+1}{3+3} \times \frac{2+1}{3+2} \times \frac{2+1}{3+2} = 0.0042$

class Y: $P(Y) \times P(Outl=o|Y) \times P(Temp=m|Y) \times P(Humi=n|Y) \times P(wind=T|Y)$

$= \frac{4}{7} \times \cdots = 0.0038$

⇒ Predict N with 0.0042 confidence

G: 0.005 < 0.01 ⇒ not confident

⇒ Stop!

(active learning)

2. One of the strategies for Query sampling was query-by-committee (QBC), where a suite of classifiers is trained over a fixed training set, and the instance that results in the highest disagreement amongst the classifiers, is selected for querying. Using the equation below, which captures vote entropy, determine the instance that our active learner would select first.

$$x^*_{VE} = \underset{x}{\text{argmax}}\left(-\sum_{y_i} \frac{V(y_i)}{C} log_2 \frac{V(y_i)}{C}\right)$$

Respectively y_i, $V(y_i)$, *and* C are the possible labels, the number of "votes" that a label receives from the classifiers, and the total number of classifiers.

classifier	Instance 1			Instance 2			Instance 3		
	y_1	y_2	y_3	y_1	y_2	y_3	y_1	y_2	y_3
C_1	0.2	0.7	0.1	0.2	0.7	0.1	0.6	0.1	0.3
C_2	0.1	0.3	0.6	0.2	0.6	0.2	0.21	0.21	0.58
C_3	0.8	0.1	0.1	0.05	0.9	0.05	0.75	0.01	0.24
C_4	0.3	0.5	0.2	0.1	0.8	0.1	0.1	0.28	0.62
$V(\cdot)$	1	2	1	0	1	0	2	0	2

⇒ Compute voting entropy: $H(X) = -\sum_i p_i \log_2 p_i$

Instance 1: $-\left\{\frac{1}{4}\log_2\frac{1}{4} + \frac{2}{4}\log_2\frac{2}{4} + \frac{1}{4}\log_2\frac{1}{4}\right\} = 1.5$ ✓

Instance 2: 0

Instance 3: $-\left\{\frac{2}{4}\log_2\frac{2}{4} + \frac{2}{4}\log_2\frac{2}{4}\right\} = 1$

⇒ Select instance 1 for querying (highest disagreement)

(Most difficult to classify ⇒ learn more about it by querying this instance)

3. Given the following univariate dataset, calculate a statistical model based on the assumption that your data is coming from a normal distribution. Determine whether the instance x=1.2 is anomalous or not if we use the boxplot test?

$$X = \{2, 2.5, 2.6, 3, 3.1, 3.2, 3.4, 3.7, 4, 4.1, 4.8\}$$

Anomaly detection: learn a model → fit the dataset & identify the objs in low prob regions. (as anomalies)
(statistical)

Normal dist: $X \sim N(\mu, \sigma^2)$

$\hat{\mu} = \frac{1}{n}\sum_i x_i = \frac{1}{11}(2 + 2.5 + \cdots + 4.8) = 3.31$

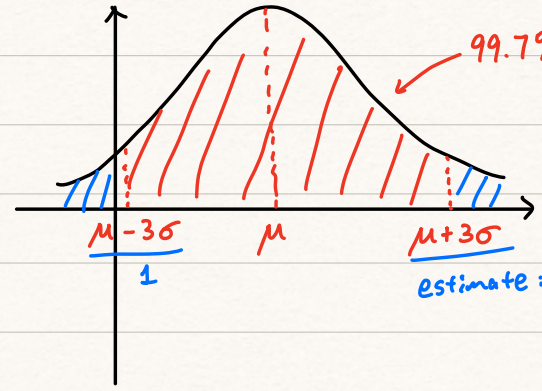
$\hat{\sigma}^2 = \frac{1}{n}\sum_i (x_i - \hat{\mu})^2 = \cdots = 0.59$

$\Rightarrow \hat{\sigma} = 0.77$

Boxplot (z-test):

If $X \sim N(\mu, \sigma^2)$: ~95% data lies in $\mu \pm 2\sigma$

~99.7% data lies in $\mu \pm 3\sigma$



$1 < 1.2 < 5.62$ ⇒ within 3σ ⇒ not outlier
(x)

(OR z-score $= \frac{|x - \mu|}{\sigma} = \frac{|1.2 - 3.31|}{0.77} = 2.74 < 3$)

4. Given the following univariate dataset, determine the outlier score for instances (x=0.5) and (x=4) using the Inverse Relative density using 2-NN (Manhattan distance) strategy.

Dataset = {1,1.05, 1.1, 1.15, 1.2, 1.21, 1.3, 1.4, 1.45, 1.5, 4.55, 5.6, 6.8, 7.58, 8.6, 9.7, 10.3, 11.4, 12.3,13.5}

Density based ⇒ outliers are in low density regions.

Relative density: "compactness" of each cluster of objects

$$\text{relative density}(x,k) = \frac{\text{density}(x,k)}{\frac{1}{k}\sum_{y \in N(x,k)} \text{density}(y,k)}$$ ← avg density of neighbors

$$\text{density}(x,k) = \left(\frac{1}{k}\sum_{y \in N(x,k)} \text{distance}(x,y)\right)^{-1}$$

inv of (average dist from k neighbors)

⇒ penalize instance if its nearest neighbors in high density region

① $x = 0.5$ (neighbours: 1, 1.05)

$$\text{density}(x=0.5, k=2) = \left(\frac{1}{2}(\underbrace{|0.5-1|}_{\text{dist}} + \underbrace{|0.5-1.05|}_{\text{dist}})\right)^{-1} = 1.9$$

neighbours: density($x=1, k=2$) = 13.3 (neighbors: 1.05, 1.1)

density($x=1.05, k=2$) = 20 (neighbors: 1, 1.1)

$$\Rightarrow \text{relative density} = \frac{1.9}{\frac{1}{2}(13.3+20)} = 0.11$$

$$\text{IRD} = \frac{1}{0.11} = 9.1 \quad \text{(high outlier score)}$$

② $x = 4$ (neighbours: 4.55, 5.6)

$$\text{RD} = \frac{0.93}{\frac{1}{2}(0.61+0.89)} = 1.24$$

$$\text{IRD} = \frac{1}{1.24} = 0.81 \quad \text{(low outlier score)}$$

$\chi = 0.5$ penalise more : nearest cluster is very compact

$\chi = 4$: close to low density clusters.

5. We have a dataset with 101 instances. Each instance corresponds to an animal and is characterized by 16 features. These animals are categorized into 7 groups (mammal, bird, reptile, fish, amphibian, insect, invertebrate). Suggest a suitable method to detect anomalies between them. Would you use a supervised, semi-supervised or unsupervised approach? Can you think of a way to make anomaly detection more reliable?

Animal dataset : 101 instances of animals (16 features)

↑ 7 groups (classes) : bird, reptile...

Anomalies : instances that don't match the characteristic of any of the groups very well.

E.g. Platypus → anomaly (or outlier)

↑ some features of birds & amphibians & mammal

Assume < we have a model trained to classify the animals into 7 groups
< platypus was not part of the training set

① Supervised :

Use it when we have access to label for "normal" data & "anomalies".

In our case, we've labels for animal categories but no labels for "normal" & "not normal" instances

⇒ Can't use supervised anomaly detection method.

② Semi-Supervised

Train a model on "normal" instances → use it to indirectly detect outliers

1. Assume that the labelled dataset includes only "normal" instances.
2. Train a supervised model on "normal" data (e.g. NB, LR, ...)

NB: outlier could be one having posterior probs with high entropy (evenly distributed among several classes)

3. Platypus features → expect the classifier to assign similar probs for bird & mammal & amphibian.

Problem: this approach can lead to high FP rate (there could be high entropy distribution for noisy or just slightly atypical instances)

To be more confident about the outcome ⇒ ensemble (several different classifiers)

③ Unsupervised

Useful when we are not quite confident that the given dataset is good representative of "normal" instances (OR have no access to labels y)

⇒ treat dataset as unsupervised set

E.g. Cluster-based outlier detection (& ensemble method could lead to more reliable predictions)

k-means with different seeds or "k"...

- We can decide on a threshold → outlier or not

  e.g. outlier as top-n instances with furthest dist (relative dist) from any centroid

- Clustering → make binary decision (Outlier or not) ⇒ voting

Alternatively, clustering algorithm → return relative distance values directly

⇒ ensemble { Average dist
"max" dist
"min" dist

⇒ threshold the value