Q1

Given a contingency table of conditional and prior probabilities for a training set with 10 examples and 5 categorical features:

Swimming	Yes	No
Rain Recently=light	1/4	3/6
Rain Recently=moderate	2/4	2/6
Rain Recently=heavy	1/4	1/6
Rain Today=light	1/4	3/6
Rain Today=moderate	2/4	2/6
Rain Today=heavy	1/4	1/6
Temp=Cold	1/4	5/6
Temp=Warm	3/4	1/6
Wind=Light	2/4	2/6
Wind=Moderate	1/4	2/6
Wind=Gale	1/4	2/6
Sunshine=Some	2/4	4/6
Sunshine=None	2/4	2/6
Class Probabilities	4/10	6/10

Q1

Based on the contingency table, classify the two new examples below using Naïve Bayes.

Example	Rain Recently (RR)	Rain Today (RT)	Temp (T)	Wind (W)	Sunshine (S)	Swimming
X1	Heavy	Moderate	Warm	Light	Some	???

Example	Rain Recently (RR)	Rain Today (RT)	Temp (T)	Wind (W)	Sunshine (S)	Swimming
X2	Light	Moderate	Warm	Light	Some	???

Naïve Bayes classification steps:

1. Calculate probability of input having class *Yes*
2. Calculate probability of input having class *No*
3. Normalise probabilities (optional)

Q1: Input X1

Test input example for hypothesis 1: *Swimming=Yes*

Example	Rain Recently (RR)	Rain Today (RT)	Temp (T)	Wind (W)	Sunshine (S)	Swimming
X1	Heavy	Moderate	Warm	Light	Some	???

Identify the relevant rows in the contingency table for *Swimming=Yes*:

Swimming	Yes	No
Rain Recently=heavy	1/4	1/6
Rain Today=moderate	2/4	2/6
Temp=Warm	3/4	1/6
Wind=Light	2/4	2/6
Sunshine=Some	2/4	4/6
Class Probabilities	4/10	6/10

Apply NB for *Swimming=Yes* by calculating product of probabilities for input's feature values and class probability:

$$P = (1/4 \times 2/4 \times 3/4 \times 2/4 \times 2/4) \times 4/10$$

$$P = 0.009375$$

Q1: Input X1

Test input example for hypothesis 2: *Swimming=No*

Example	Rain Recently (RR)	Rain Today (RT)	Temp (T)	Wind (W)	Sunshine (S)	Swimming
X1	Heavy	Moderate	Warm	Light	Some	???

Identify the relevant rows in the contingency table for *Swimming=No*:

Swimming	Yes	No
Rain Recently=heavy	1/4	1/6
Rain Today=moderate	2/4	2/6
Temp=Warm	3/4	1/6
Wind=Light	2/4	2/6
Sunshine=Some	2/4	4/6
Class Probabilities	4/10	6/10

Apply NB for *Swimming=No* by calculating product of probabilities for input's feature values and class probability:

P = (1/6 x 2/6 x 1/6 x 2/6 x 4/6) x 6/10

P = .001234

Q1: Input X1

- We calculated probabilities for two hypotheses (class labels):

Yes `P(Y) = 1/4 x 2/4 x 3/4 x 2/4 x 2/4 x 4/10 = 0.009375`

No `P(N) = 1/6 x 2/6 x 1/6 x 2/6 x 4/6 x 6/10 = 0.001234`

- Normalise probabilities to sum to 1:

Yes `P(Y)' = 0.009375/(0.009375+0.001234) = .884`

No `P(N)' = 0.001234/(0.009375+0.001234) = .116`

- Output Prediction: **Swimming = Yes**

Q1: Input X2

Test input example for hypothesis 1: *Swimming=Yes*

Example	Rain Recently (RR)	Rain Today (RT)	Temp (T)	Wind (W)	Sunshine (S)	Swimming
X2	Light	Moderate	Warm	Light	Some	???

Identify the relevant rows in the contingency table for *Swimming=Yes*:

Swimming	Yes	No
Rain Recently=light	1/4	3/6
Rain Today=moderate	2/4	2/6
Temp=Warm	3/4	1/6
Wind=Light	2/4	2/6
Sunshine=Some	2/4	4/6
Class Probabilities	4/10	6/10

Apply NB for *Swimming=Yes* by calculating product of probabilities for input's feature values and class probability:

```
P = (1/4 x 2/4 x 3/4 x 2/4 x 2/4) x 4/10
P = 0.09375
```

Q1: Input X2

Test input example for hypothesis 1: *Swimming=No*

Example	Rain Recently (RR)	Rain Today (RT)	Temp (T)	Wind (W)	Sunshine (S)	Swimming
X2	Light	Moderate	Warm	Light	Some	???

Identify the relevant rows in the contingency table for *Swimming=No*:

Swimming	Yes	No
Rain Recently=light	1/4	3/6
Rain Today=moderate	2/4	2/6
Temp=Warm	3/4	1/6
Wind=Light	2/4	2/6
Sunshine=Some	2/4	4/6
Class Probabilities	4/10	6/10

Apply NB for *Swimming=No* by calculating product of probabilities for input's feature values and class probability:

```
P = (3/6 x 2/6 x 1/6 x 2/6 x 4/6) x 6/10
P = 0.003692
```

Q1: Input X2

- Calculated probabilities for two hypotheses (class labels):

Yes `P(Y) = (1/4 x 2/4 x 3/4 x 2/4 x 2/4) x 4/10 = 0.09375`

No `P(N) = (3/6 x 2/6 x 1/6 x 2/6 x 4/6) x 6/10 = 0.003692`

- Normalise probabilities to sum to 1:

Yes `P(Y)' = 0.09375/(0.093756+0.003692) = .962`

No `P(N)' = 0.0036926/(0.093756+0.003692) = .038`

- Output Prediction: **Swimming = Yes**

Q2(a)

a) Provide the contingency table of conditional and prior probabilities that would be used by Naïve Bayes to build a classifier for this dataset.

	Name	Hair	Height	Build	Lotion	Result
1	Sarah	blonde	average	light	no	sunburned
2	Dana	blonde	tall	average	yes	none
3	Alex	brown	short	average	yes	none
4	Annie	blonde	short	average	no	sunburned
5	Emily	red	average	heavy	no	sunburned
6	Pete	brown	tall	heavy	no	none
7	John	brown	average	heavy	no	none
8	Katie	brown	short	light	yes	none

Q2(a)

Construct full contingency table for all features on both classes:

Feature Value	Sunburned	None
Hair=blonde		
Hair=brown		
Hair=red		
Height=average		
Height=tall		
Height=short		
Build=light		
Build=average		
Build=heavy		
Lotion=no		
Lotion=yes		
Class Probabilities		

Q2(a)

Construct full contingency table for all features on both classes:

Feature Value	Sunburned	None
Hair=blonde	2/3	1/5
Hair=brown	0/3	4/5
Hair=red	1/3	0/5
Height=average	2/3	1/5
Height=tall	0/3	2/5
Height=short	1/3	2/5
Build=light	1/3	1/5
Build=average	1/3	2/5
Build=heavy	1/3	2/5
Lotion=no	3/3	2/5
Lotion=yes	0/3	3/5
Class Probabilities	3/8	5/8

Q2(a)

We have conditional probabilities of zero - need to use Laplace smoothing with k=1

$$P(f = v|c) = \frac{count(f = v|c) + k}{count(f|c) + (k \times |Domain(f)|)}$$

$count(f{=}v \mid c)$ is how often the feature f has value v for instances where the class is c.

$count(f \mid c)$ is how often the feature f has any value where the class is c

$|Domain(f)|$ is the number of different values feature f can have

Q2(a)

Update contingencies table with smoothed probabilities (with k=1):

Feature Value	SunB	None	#(f=v|S)	#(f=v|N)	#(f|S)	#(f|N)	|D(f)|	SunB	None
Hair=blonde	2/3	1/5	2	1	3	5	3	0.5	0.25
Hair=brown	0/3	4/5	0	4	3	5	3	0.17	0.63
Hair=red	1/3	0/5	1	0	3	5	3	0.33	0.13
Height=average	2/3	1/5	2	1	3	5	3	0.5	0.25
Height=tall	0/3	2/5	0	2	3	5	3	0.17	0.38
Height=short	1/3	2/5	1	2	3	5	3	0.33	0.38
Build=light	1/3	1/5	1	1	3	5	3	0.33	0.25
Build=average	1/3	2/5	1	2	3	5	3	0.33	0.38
Build=heavy	1/3	2/5	1	2	3	5	3	0.33	0.38
Lotion=no	3/3	2/5	3	2	3	5	2	0.8	0.43
Lotion=yes	0/3	3/5	0	3	3	5	2	0.2	0.57
Class Probabilities	3/8	5/8						3/8	5/8

Q2(b)

Use the contingency table to calculate the Naïve Bayes scores:

	Hair	Height	Build	Lotion	Result
X	blonde	average	heavy	yes	???

$$v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod_i P(f_i|v_j)$$

Result	Sunburned	None
Hair=blonde	0.5	0.25
Height=average	0.5	0.25
Build=heavy	0.33	0.38
Lotion=no	0.2	0.57
Class Probabilities	3/8	5/8

Calculate raw probabilities for two classes:

```
P(S) = 0.5 x 0.5 x 0.33 x 0.2 x (3/8)
P(S) = 0.00619
P(N) = 0.25 x 0.25 x 0.38 x 0.57 x (5/8)
P(N) = 0.00846
```

Normalise probabilities:

```
P(S)' = 0.00618/(0.00618+0.00846) = 0.422
P(N)' = 0.00846/(0.00618+0.00846) = 0.578
```

➡ **Output: None**

Q3(a)

a) Calculate the contingency table that would be used by Naïve Bayes to build a classifier using this training data.

Example	Credit History	Debt	Income	Risk
1	bad	low	10,000	high
2	bad	high	32,000	high
3	bad	low	18,000	high
4	unknown	high	46,000	high
5	unknown	high	23,000	high
6	good	high	27,500	high
7	bad	low	28,000	medium
8	unknown	low	55,000	medium
9	good	high	57,500	medium
10	unknown	low	65,000	medium
11	unknown	low	75,000	low
12	good	low	72,000	low
13	good	high	90,000	low
14	good	high	100,000	low
15	bad	low	50,000	low

Q3(a)

Income field is continuous, use equal-frequency binning to convert to a categorical feature; with 3 bins —> 5 instances in each bin

Example	Credit History	Debt	Income	Risk
14	good	high	100,000	low
13	good	high	90,000	low
11	unknown	low	75,000	low
12	good	low	72,000	low
10	unknown	low	65,000	medium
9	good	high	57,500	medium
8	unknown	low	55,000	medium
15	bad	low	50,000	low
4	unknown	high	46,000	high
2	bad	high	32,000	high
7	bad	low	28,000	medium
6	good	high	27,500	high
5	unknown	high	23,000	high
3	bad	low	18,000	high
1	bad	low	10,000	high

over60

30to60

0to30

Q3(a)

Contingency table with Income feature binned

Example	Credit History	Debt	Income	Risk
1	bad	low	0to30	high
2	bad	high	30to60	high
3	bad	low	0to30	high
4	unknown	high	30to60	high
5	unknown	high	0to30	high
6	good	high	0to30	high
7	bad	low	0to30	medium
8	unknown	low	30to60	medium
9	good	high	30to60	medium
10	unknown	low	over60	medium
11	unknown	low	over60	low
12	good	low	over60	low
13	good	high	over60	low
14	good	high	over60	low
15	bad	low	30to60	low

Q3(a)

a) Calculate the contingency table that would be used by Naïve Bayes to build a classifier using this training data.

Contingency table for each of the descriptive features across 3 classes:

Risk	high	medium	low
CH=bad			
CH=unknown			
CH=good			
Debt=low			
Debt=high			
Income=0to30			
Income=30to60			
Income=over60			
Class Probabilities			

Tutorial Q3(a)

a) Calculate the contingency table that would be used by Naïve Bayes to build a classifier using this training data.

Contingency table for each of the descriptive features across 3 classes:

Risk	high	medium	low
CH=bad	3/6	1/4	1/5
CH=unknown	2/6	2/4	1/5
CH=good	1/6	1/4	3/5
Debt=low	2/6	3/4	3/5
Debt=high	4/6	1/4	2/5
Income=0to30	4/6	1/4	0/5
Income=30to60	2/6	2/4	1/5
Income=over60	0/6	1/4	4/5
Class Probabilities	6/15	4/15	5/15

Q3(a)

b) Predict the risk level for the new loan application X below.

	Credit History	Debt	Income	Risk
X	bad	low	30to60	???

Risk	high	medium	low
CH=bad	3/6	1/4	1/5
Debt=low	1/6	2/4	2/5
Income=30to60	2/6	2/4	1/5
Class Probabilities	6/15	3/15	5/15

Calculate raw probabilities for 3 classes, using contingency table:

$$P(H) = (3/6)x(2/6)x(2/6) \ x \ (6/15) = 0.022$$

$$P(M) = (1/4)x(3/4)x(2/4) \ x \ (3/15) = 0.019$$

$$P(L) = (1/5)x(3/5)x(1/5) \ x \ (5/15) = 0.005$$

Normalise probabilities:

$$P(H)' = 0.022/(0.022+0.019+0.005) = 0.48$$

$$P(M)' = 0.019/(0.022+0.019+0.005) = 0.41$$

$$P(L)' = 0.005/(0.022+0.019+0.005) = 0.11$$

➡ **Output: Low Risk**

4(a)

4.(a) Given the nature of the `AthleteSelection` data which would be the best of the Naive Bayes options in scikit-learn for that classification task?

Gaussian Naive Bayes is possibly the only real option we have here because the features are real values - not counts or categories. The data is probably not exactly Gaussian but probably close enough.

4(b)

4.(b) A ranking classifier is a classifier that can rank a test set in order of confidence for a given classification outcome. Naive Bayes is a ranking classifier because the 'probability' can be used as a confidence measure for ranking.

1. Train a Naive Bayes classifier from the `AthleteSelection` data. Load the test data from `AthleteTest.csv` and apply the classifier.
2. Use the `predict_proba` method to find the probability of being selected.
3. Rank the test set by probability of being selected.
   3.1. Who is most likely to be selected?
   3.2. Who is least likely?

Some code for this exercise is available in the notebook `**04 Naive Bayes Lab**`. You will also need to download the test data file **'AthleteTest.csv'**.

Tutorial 4(b)

```
gnb = GaussianNB()
ath_NB = gnb.fit(X,y)

y_probs = ath_NB.predict_proba(X_test)
ath_test['Prob']=y_probs[:,1]
ath_test.sort_values(by=['Prob'], ascending=False, inplace = True)
ath_test
```

```
y_probs
Out[12]:
array([[9.58686371e-01, 4.13136290e-02],
       [8.77017219e-01, 1.22982781e-01],
       [8.80671574e-02, 9.11932843e-01],
       [8.49522335e-01, 1.50477665e-01],
       [2.00167162e-01, 7.99832838e-01],
       [2.64304710e-06, 9.99997357e-01],
       [5.48092049e-05, 9.99945191e-01],
       [2.70690822e-02, 9.72930918e-01],
       [1.45717357e-01, 8.54282643e-01],
       [2.70690822e-02, 9.72930918e-01]])
In [ ]:
```

Athlete	Speed	Agility	Prob
t6	8.1	7.8	0.999997
t7	7.7	5.2	0.999945
t8	6.1	5.5	0.972931
t10	6.1	5.5	0.972931
t3	5.5	7.2	0.911933
t9	5.5	6.0	0.854283
t5	5.5	5.2	0.799833
t4	3.8	8.8	0.150478
t2	4.5	4.5	0.122983
t1	3.3	8.2	0.041314

Tutorial 4(c)

- When a **`GaussianNB`** model is trained the model is stored in two parameters **'theta_'** and **'sigma_'.** Train a **`GaussianNB`** model and check to see if these parameters agree with your own estimates.

- Hint: this code will give you the estimated you need.

  ```
  athlete[athlete['Selected']=='No']['Agility'].describe()
  ```

- Despite the name the **'sigma_'** parameter contains the square of the standard deviation (the variance) rather than the standard deviations. You will find these figures do not agree exactly.

Question 4(c)

	Speed	Agility	Selected
Athlete			
x1	2.50	6.00	No
x2	3.75	8.00	No
x3	2.25	5.50	No
x4	3.25	8.25	No
x5	2.75	7.50	No

```
gnb.sigma_
Out[28]:
array([[0.80685764, 3.99305556],
       [1.37402344, 3.91308594]])

gnb.theta_
Out[29]:
array([[3.39583333, 5.08333333],
       [6.40625   , 6.96875   ]])

athlete[athlete['Selected']=='No']['Agility'].describe()
Out[30]:
count    12.000000
mean      5.083333
std       2.087118
min       2.000000
25%       3.625000
50%       5.125000
75%       6.375000
max       8.250000
Name: Agility, dtype: float64
```