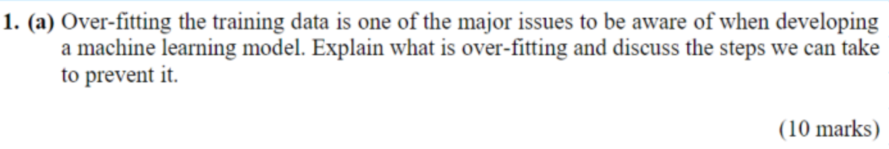
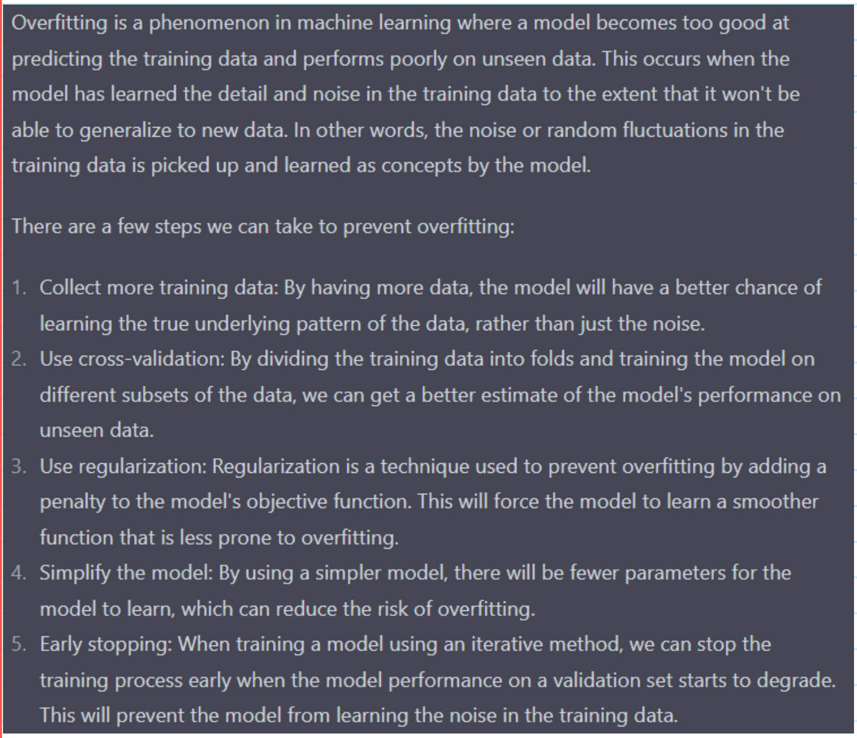
2021-22

Sunday 8 January 2023 20:19



1. (a) Over-fitting the training data is one of the major issues to be aware of when developing a machine learning model. Explain what is over-fitting and discuss the steps we can take to prevent it.

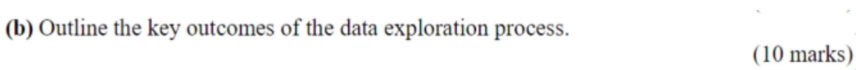
(10 marks)



Overfitting is a phenomenon in machine learning where a model becomes too good at predicting the training data and performs poorly on unseen data. This occurs when the model has learned the detail and noise in the training data to the extent that it won't be able to generalize to new data. In other words, the noise or random fluctuations in the training data is picked up and learned as concepts by the model.

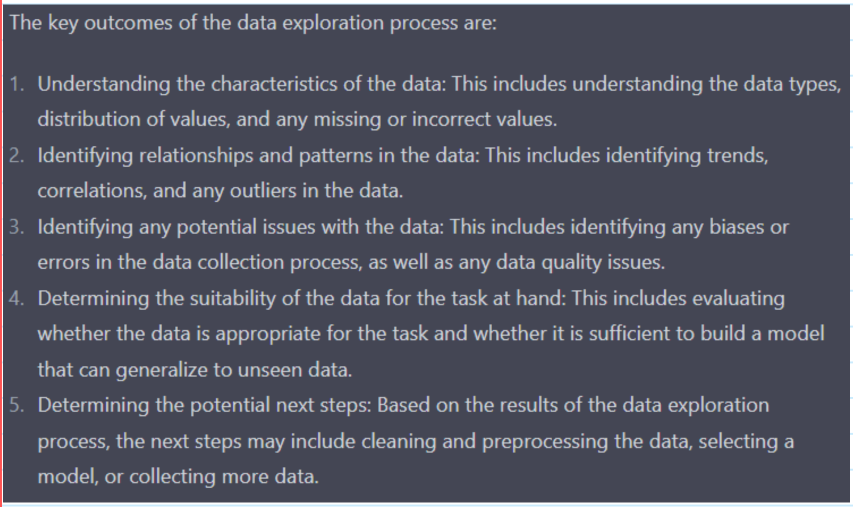
There are a few steps we can take to prevent overfitting:

1. Collect more training data: By having more data, the model will have a better chance of learning the true underlying pattern of the data, rather than just the noise.
2. Use cross-validation: By dividing the training data into folds and training the model on different subsets of the data, we can get a better estimate of the model's performance on unseen data.
3. Use regularization: Regularization is a technique used to prevent overfitting by adding a penalty to the model's objective function. This will force the model to learn a smoother function that is less prone to overfitting.
4. Simplify the model: By using a simpler model, there will be fewer parameters for the model to learn, which can reduce the risk of overfitting.
5. Early stopping: When training a model using an iterative method, we can stop the training process early when the model performance on a validation set starts to degrade. This will prevent the model from learning the noise in the training data.



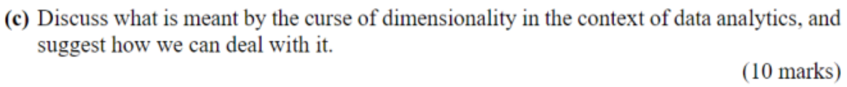
(b) Outline the key outcomes of the data exploration process.

(10 marks)



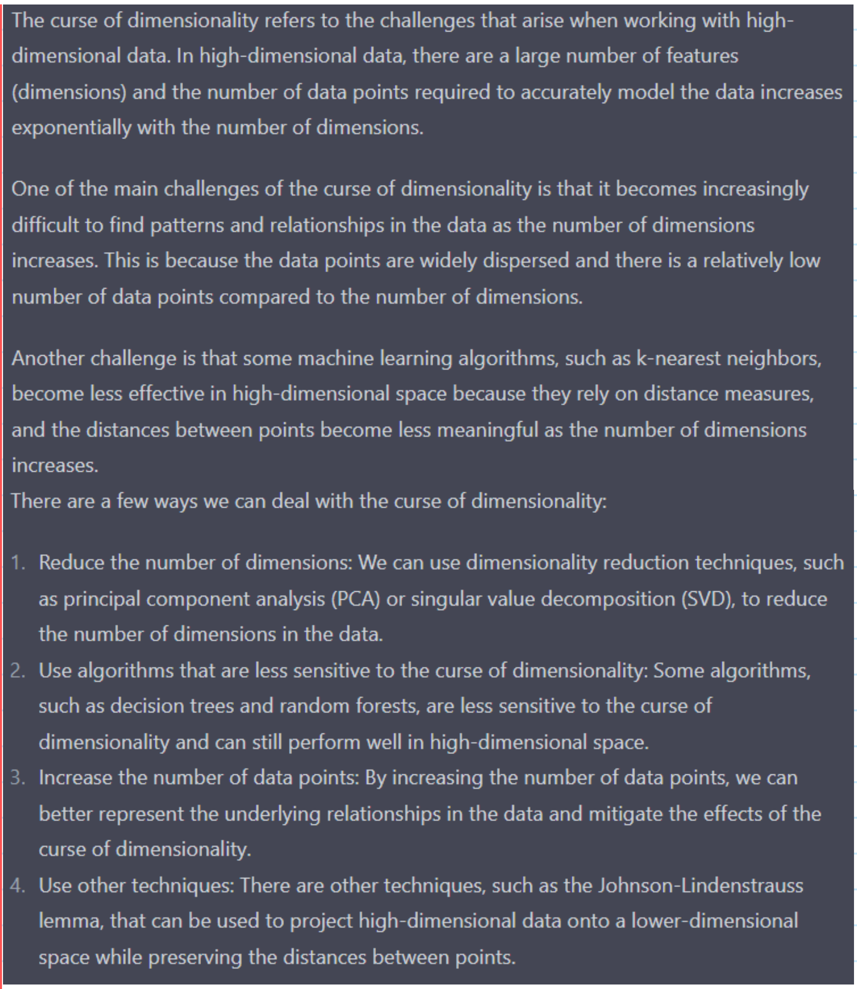
The key outcomes of the data exploration process are:

1. Understanding the characteristics of the data: This includes understanding the data types, distribution of values, and any missing or incorrect values.
2. Identifying relationships and patterns in the data: This includes identifying trends, correlations, and any outliers in the data.
3. Identifying any potential issues with the data: This includes identifying any biases or errors in the data collection process, as well as any data quality issues.
4. Determining the suitability of the data for the task at hand: This includes evaluating whether the data is appropriate for the task and whether it is sufficient to build a model that can generalize to unseen data.
5. Determining the potential next steps: Based on the results of the data exploration process, the next steps may include cleaning and preprocessing the data, selecting a model, or collecting more data.



(c) Discuss what is meant by the curse of dimensionality in the context of data analytics, and suggest how we can deal with it.

(10 marks)



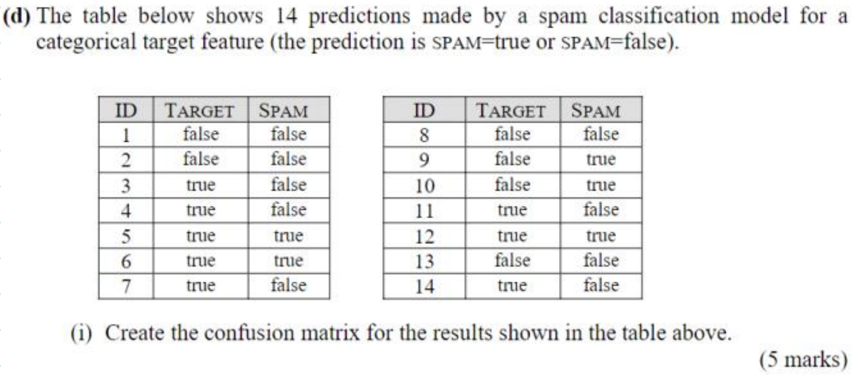
The curse of dimensionality refers to the challenges that arise when working with high-dimensional data. In high-dimensional data, there are a large number of features (dimensions) and the number of data points required to accurately model the data increases exponentially with the number of dimensions.

One of the main challenges of the curse of dimensionality is that it becomes increasingly difficult to find patterns and relationships in the data as the number of dimensions increases. This is because the data points are widely dispersed and there is a relatively low number of data points compared to the number of dimensions.

Another challenge is that some machine learning algorithms, such as k-nearest neighbors, become less effective in high-dimensional space because they rely on distance measures, and the distances between points become less meaningful as the number of dimensions increases.

There are a few ways we can deal with the curse of dimensionality:

1. Reduce the number of dimensions: We can use dimensionality reduction techniques, such as principal component analysis (PCA) or singular value decomposition (SVD), to reduce the number of dimensions in the data.
2. Use algorithms that are less sensitive to the curse of dimensionality: Some algorithms, such as decision trees and random forests, are less sensitive to the curse of dimensionality and can still perform well in high-dimensional space.
3. Increase the number of data points: By increasing the number of data points, we can better represent the underlying relationships in the data and mitigate the effects of the curse of dimensionality.
4. Use other techniques: There are other techniques, such as the Johnson-Lindenstrauss lemma, that can be used to project high-dimensional data onto a lower-dimensional space while preserving the distances between points.



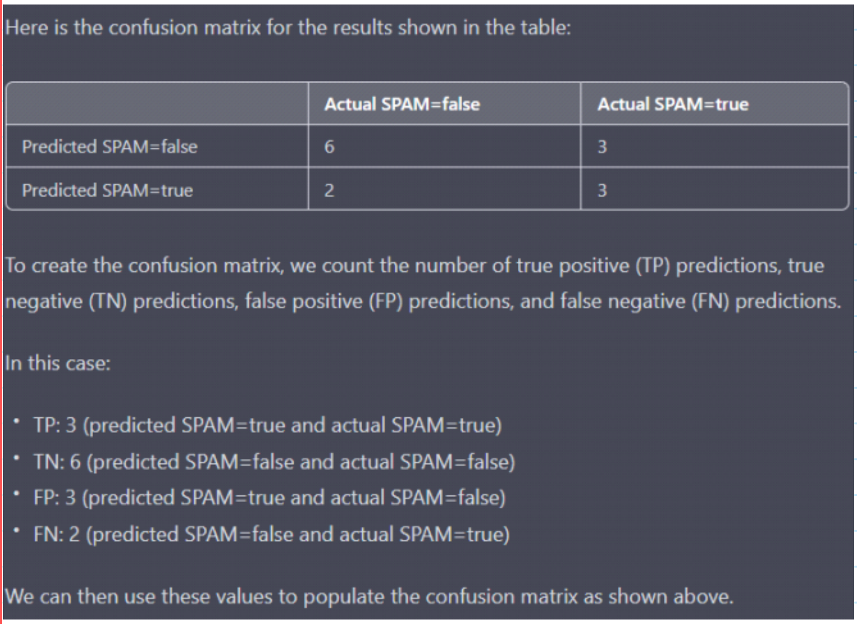
(d) The table below shows 14 predictions made by a spam classification model for a categorical target feature (the prediction is SPAM=true or SPAM=false).

ID	TARGET	SPAM
1	false	false
2	false	false
3	true	false
4	true	false
5	true	true
6	true	true
7	true	false

ID	TARGET	SPAM
8	false	false
9	false	true
10	false	true
11	true	false
12	true	true
13	false	false
14	true	false

(i) Create the confusion matrix for the results shown in the table above.

(5 marks)



Here is the confusion matrix for the results shown in the table:

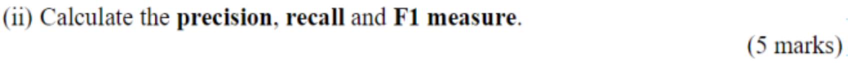
	Actual SPAM=false	Actual SPAM=true
Predicted SPAM=false	6	3
Predicted SPAM=true	2	3

To create the confusion matrix, we count the number of true positive (TP) predictions, true negative (TN) predictions, false positive (FP) predictions, and false negative (FN) predictions.

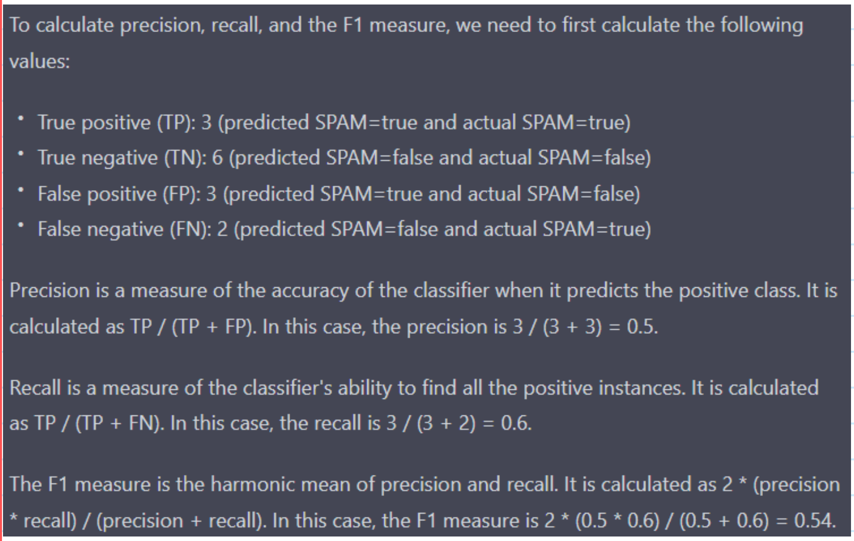
In this case:

- TP: 3 (predicted SPAM=true and actual SPAM=true)
- TN: 6 (predicted SPAM=false and actual SPAM=false)
- FP: 3 (predicted SPAM=true and actual SPAM=false)
- FN: 2 (predicted SPAM=false and actual SPAM=true)

We can then use these values to populate the confusion matrix as shown above.


(ii) Calculate the **precision**, **recall** and **F1 measure**.

(5 marks)

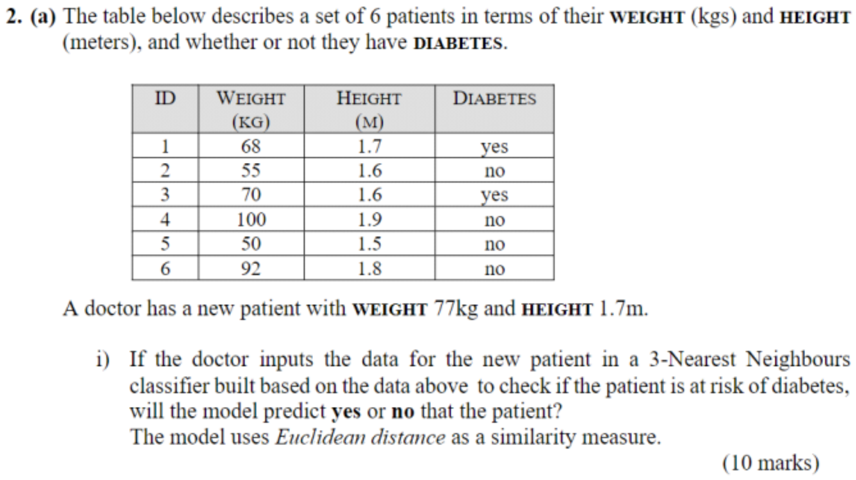

To calculate precision, recall, and the F1 measure, we need to first calculate the following values:

- True positive (TP): 3 (predicted SPAM=true and actual SPAM=true)
- True negative (TN): 6 (predicted SPAM=false and actual SPAM=false)
- False positive (FP): 3 (predicted SPAM=true and actual SPAM=false)
- False negative (FN): 2 (predicted SPAM=false and actual SPAM=true)

Precision is a measure of the accuracy of the classifier when it predicts the positive class. It is calculated as TP / (TP + FP). In this case, the precision is 3 / (3 + 3) = 0.5.

Recall is a measure of the classifier's ability to find all the positive instances. It is calculated as TP / (TP + FN). In this case, the recall is 3 / (3 + 2) = 0.6.

The F1 measure is the harmonic mean of precision and recall. It is calculated as 2 * (precision * recall) / (precision + recall). In this case, the F1 measure is 2 * (0.5 * 0.6) / (0.5 + 0.6) = 0.54.

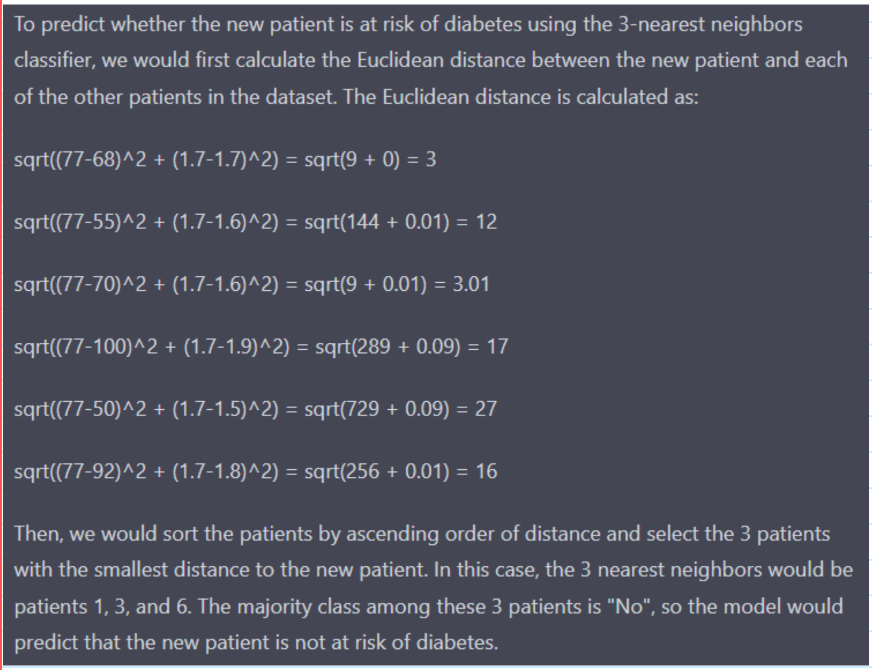

2. (a) The table below describes a set of 6 patients in terms of their **WEIGHT** (kgs) and **HEIGHT** (meters), and whether or not they have **DIABETES**.

ID	WEIGHT (KG)	HEIGHT (M)	DIABETES
1	68	1.7	yes
2	55	1.6	no
3	70	1.6	yes
4	100	1.9	no
5	50	1.5	no
6	92	1.8	no

A doctor has a new patient with **WEIGHT** 77kg and **HEIGHT** 1.7m.

i) If the doctor inputs the data for the new patient in a 3-Nearest Neighbours classifier built based on the data above to check if the patient is at risk of diabetes, will the model predict **yes** or **no** that the patient?
The model uses *Euclidean distance* as a similarity measure.

(10 marks)


To predict whether the new patient is at risk of diabetes using the 3-nearest neighbors classifier, we would first calculate the Euclidean distance between the new patient and each of the other patients in the dataset. The Euclidean distance is calculated as:

sqrt((77-68)^2 + (1.7-1.7)^2) = sqrt(9 + 0) = 3

sqrt((77-55)^2 + (1.7-1.6)^2) = sqrt(144 + 0.01) = 12

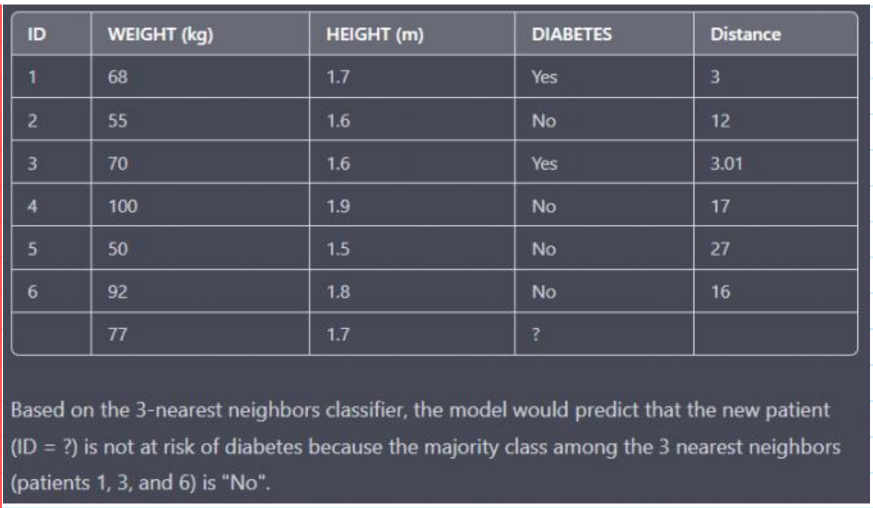
sqrt((77-70)^2 + (1.7-1.6)^2) = sqrt(9 + 0.01) = 3.01

sqrt((77-100)^2 + (1.7-1.9)^2) = sqrt(289 + 0.09) = 17

sqrt((77-50)^2 + (1.7-1.5)^2) = sqrt(729 + 0.09) = 27

sqrt((77-92)^2 + (1.7-1.8)^2) = sqrt(256 + 0.01) = 16

Then, we would sort the patients by ascending order of distance and select the 3 patients with the smallest distance to the new patient. In this case, the 3 nearest neighbors would be patients 1, 3, and 6. The majority class among these 3 patients is "No", so the model would predict that the new patient is not at risk of diabetes.



ID	WEIGHT (kg)	HEIGHT (m)	DIABETES	Distance
1	68	1.7	Yes	3
2	55	1.6	No	12
3	70	1.6	Yes	3.01
4	100	1.9	No	17
5	50	1.5	No	27
6	92	1.8	No	16
	77	1.7	?	

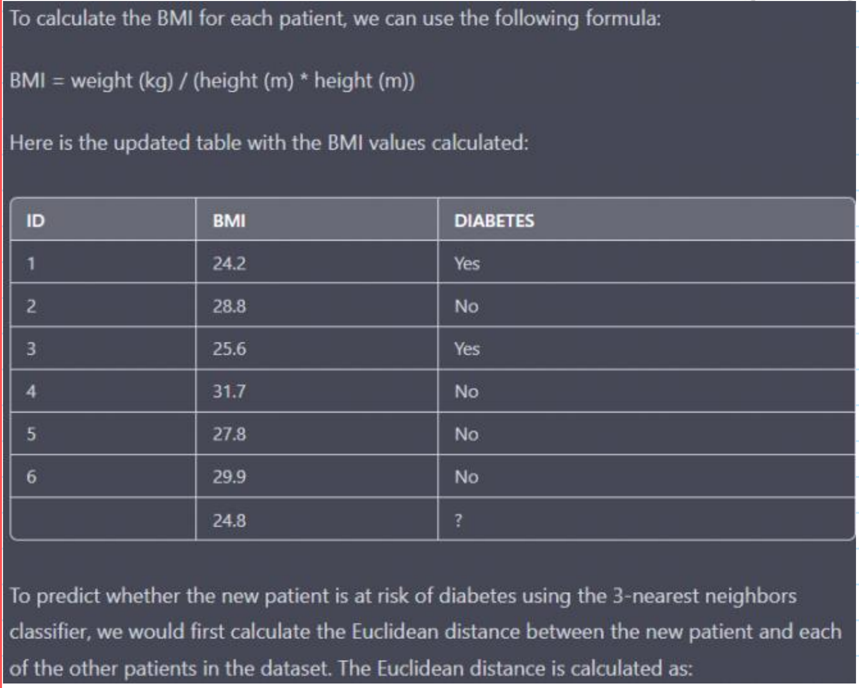
Based on the 3-nearest neighbors classifier, the model would predict that the new patient (ID = ?) is not at risk of diabetes because the majority class among the 3 nearest neighbors (patients 1, 3, and 6) is "No".

ii) Some medical professionals use BMI as a combined metrics for a patient's **WEIGHT** and **HEIGHT**. BMI is calculated as

$$BMI = \frac{weight\ (kg)}{height\ (m) * \ height\ (m)}$$

Assuming the data for the patients in table above has been updated, and **WEIGHT** and **HEIGHT** features replaced with **BMI** feature instead, what prediction would a 3-Nearest Neighbours classifier return for the new patient based on the updated data.

(10 marks)



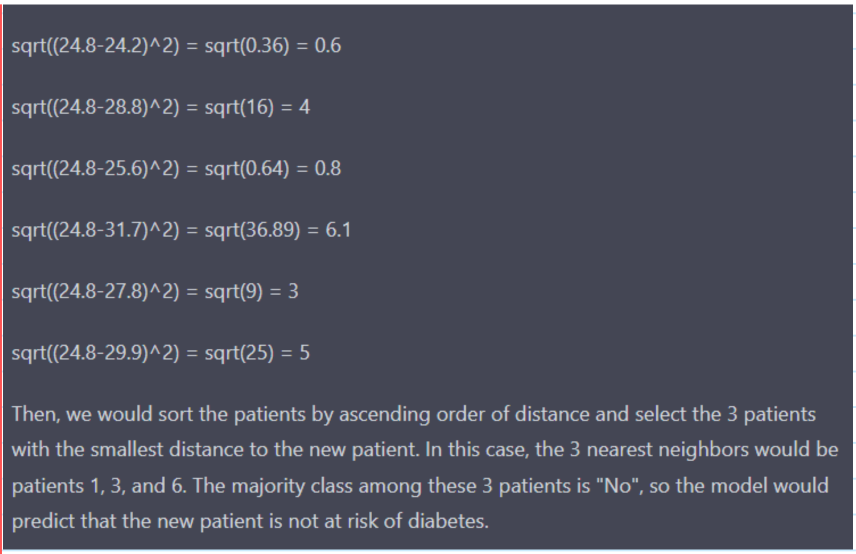
To calculate the BMI for each patient, we can use the following formula:

BMI = weight (kg) / (height (m) * height (m))

Here is the updated table with the BMI values calculated:

ID	BMI	DIABETES
1	24.2	Yes
2	28.8	No
3	25.6	Yes
4	31.7	No
5	27.8	No
6	29.9	No
	24.8	?

To predict whether the new patient is at risk of diabetes using the 3-nearest neighbors classifier, we would first calculate the Euclidean distance between the new patient and each of the other patients in the dataset. The Euclidean distance is calculated as:



sqrt((24.8-24.2)^2) = sqrt(0.36) = 0.6

sqrt((24.8-28.8)^2) = sqrt(16) = 4

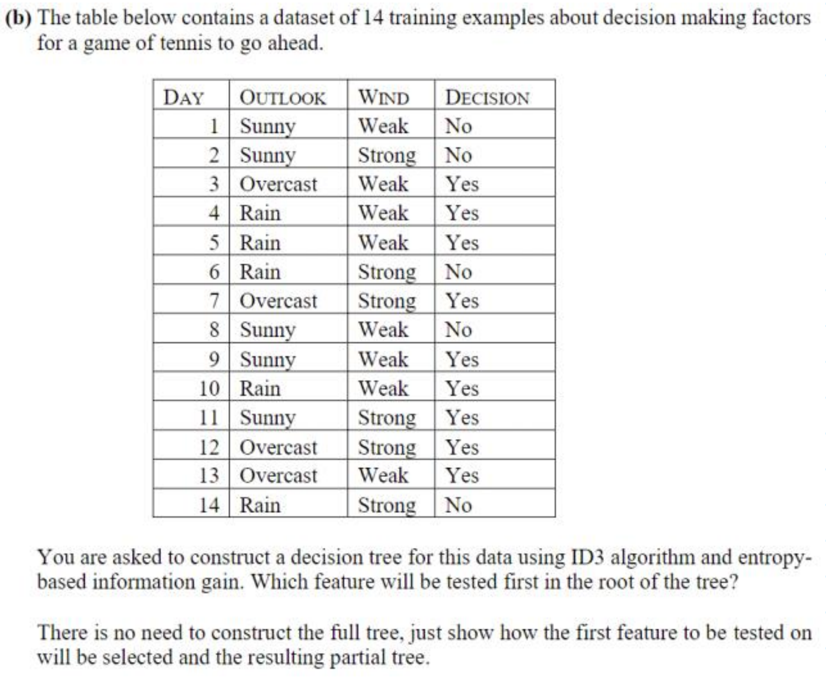
sqrt((24.8-25.6)^2) = sqrt(0.64) = 0.8

sqrt((24.8-31.7)^2) = sqrt(36.89) = 6.1

sqrt((24.8-27.8)^2) = sqrt(9) = 3

sqrt((24.8-29.9)^2) = sqrt(25) = 5

Then, we would sort the patients by ascending order of distance and select the 3 patients with the smallest distance to the new patient. In this case, the 3 nearest neighbors would be patients 1, 3, and 6. The majority class among these 3 patients is "No", so the model would predict that the new patient is not at risk of diabetes.

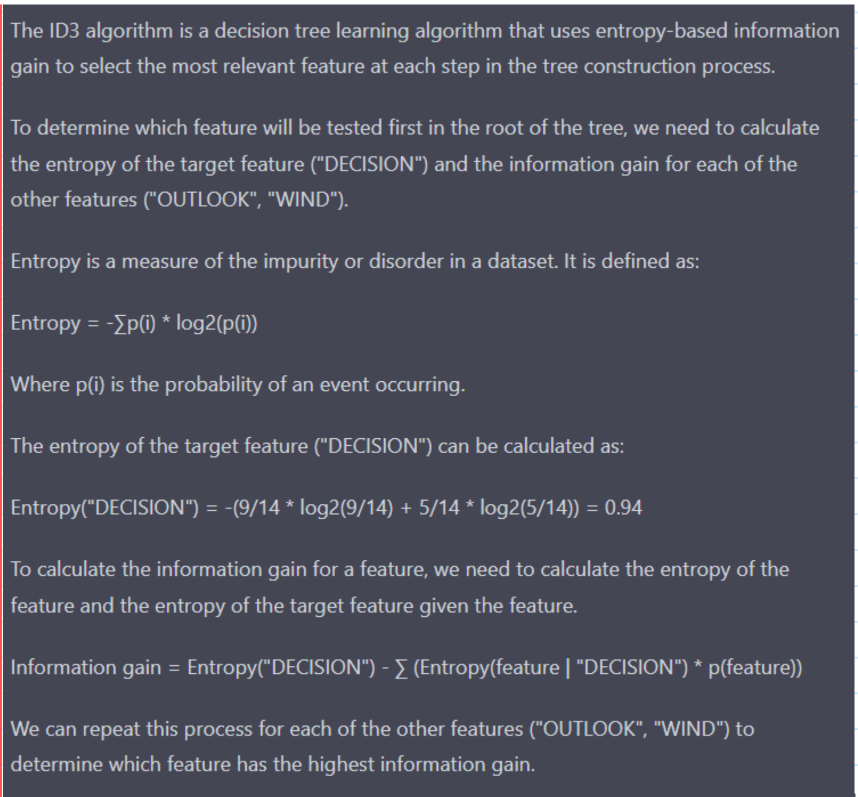


(b) The table below contains a dataset of 14 training examples about decision making factors for a game of tennis to go ahead.

DAY	OUTLOOK	WIND	DECISION
1	Sunny	Weak	No
2	Sunny	Strong	No
3	Overcast	Weak	Yes
4	Rain	Weak	Yes
5	Rain	Weak	Yes
6	Rain	Strong	No
7	Overcast	Strong	Yes
8	Sunny	Weak	No
9	Sunny	Weak	Yes
10	Rain	Weak	Yes
11	Sunny	Strong	Yes
12	Overcast	Strong	Yes
13	Overcast	Weak	Yes
14	Rain	Strong	No

You are asked to construct a decision tree for this data using ID3 algorithm and entropy-based information gain. Which feature will be tested first in the root of the tree?

There is no need to construct the full tree, just show how the first feature to be tested on will be selected and the resulting partial tree.



The ID3 algorithm is a decision tree learning algorithm that uses entropy-based information gain to select the most relevant feature at each step in the tree construction process.

To determine which feature will be tested first in the root of the tree, we need to calculate the entropy of the target feature ("DECISION") and the information gain for each of the other features ("OUTLOOK", "WIND").

Entropy is a measure of the impurity or disorder in a dataset. It is defined as:

Entropy = -∑p(i) * log2(p(i))

Where p(i) is the probability of an event occurring.

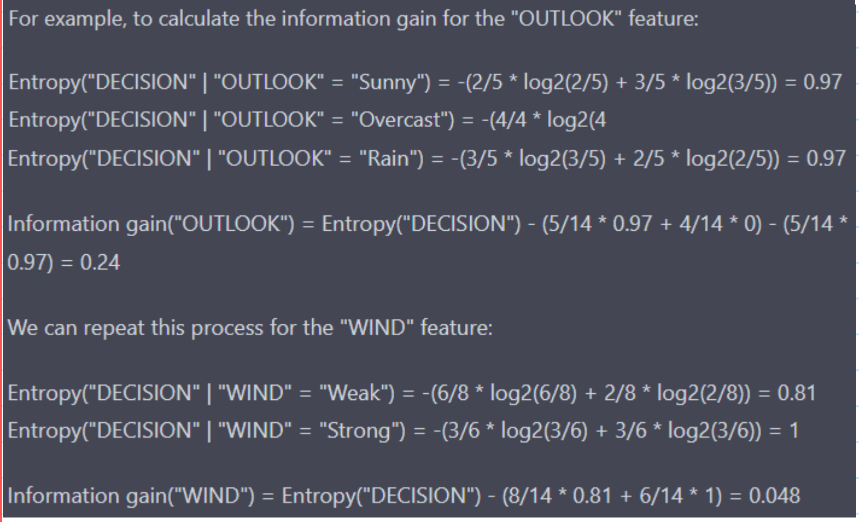
The entropy of the target feature ("DECISION") can be calculated as:

Entropy("DECISION") = -(9/14 * log2(9/14) + 5/14 * log2(5/14)) = 0.94

To calculate the information gain for a feature, we need to calculate the entropy of the feature and the entropy of the target feature given the feature.

Information gain = Entropy("DECISION") - ∑ (Entropy(feature | "DECISION") * p(feature))

We can repeat this process for each of the other features ("OUTLOOK", "WIND") to determine which feature has the highest information gain.



For example, to calculate the information gain for the "OUTLOOK" feature:

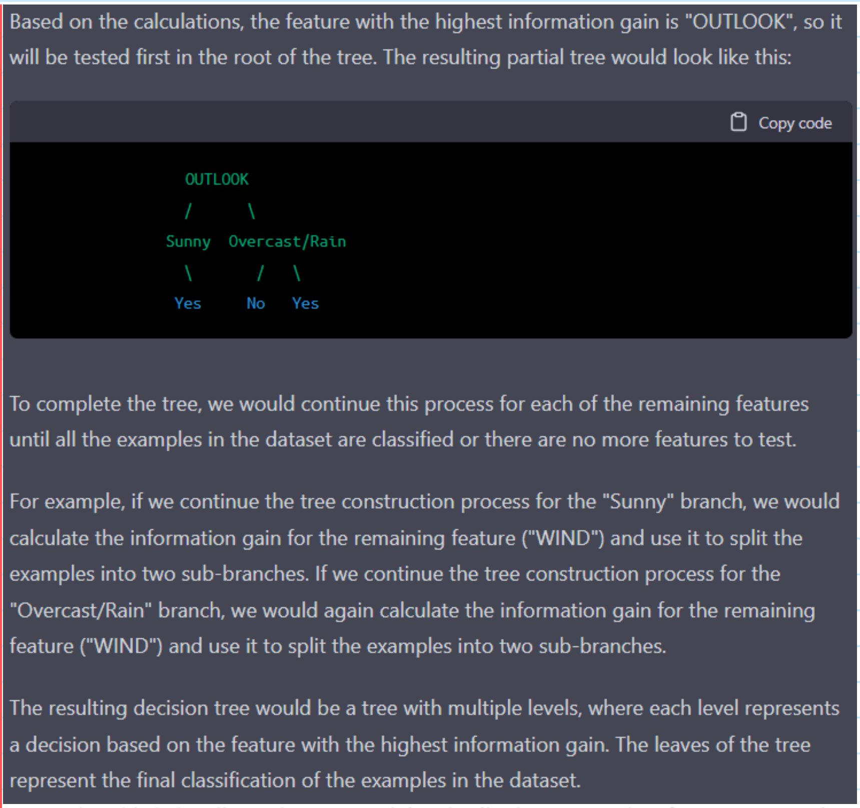
Entropy("DECISION" | "OUTLOOK" = "Sunny") = -(2/5 * log2(2/5) + 3/5 * log2(3/5)) = 0.97
Entropy("DECISION" | "OUTLOOK" = "Overcast") = -(4/4 * log2(4
Entropy("DECISION" | "OUTLOOK" = "Rain") = -(3/5 * log2(3/5) + 2/5 * log2(2/5)) = 0.97

Information gain("OUTLOOK") = Entropy("DECISION") - (5/14 * 0.97 + 4/14 * 0) - (5/14 * 0.97) = 0.24

We can repeat this process for the "WIND" feature:

Entropy("DECISION" | "WIND" = "Weak") = -(6/8 * log2(6/8) + 2/8 * log2(2/8)) = 0.81
Entropy("DECISION" | "WIND" = "Strong") = -(3/6 * log2(3/6) + 3/6 * log2(3/6)) = 1

Information gain("WIND") = Entropy("DECISION") - (8/14 * 0.81 + 6/14 * 1) = 0.048



Based on the calculations, the feature with the highest information gain is "OUTLOOK", so it will be tested first in the root of the tree. The resulting partial tree would look like this:

```
  OUTLOOK
  /      \
Sunny  Overcast/Rain
  \        /   \
 Yes      No    Yes
```

To complete the tree, we would continue this process for each of the remaining features until all the examples in the dataset are classified or there are no more features to test.

For example, if we continue the tree construction process for the "Sunny" branch, we would calculate the information gain for the remaining feature ("WIND") and use it to split the examples into two sub-branches. If we continue the tree construction process for the "Overcast/Rain" branch, we would again calculate the information gain for the remaining feature ("WIND") and use it to split the examples into two sub-branches.

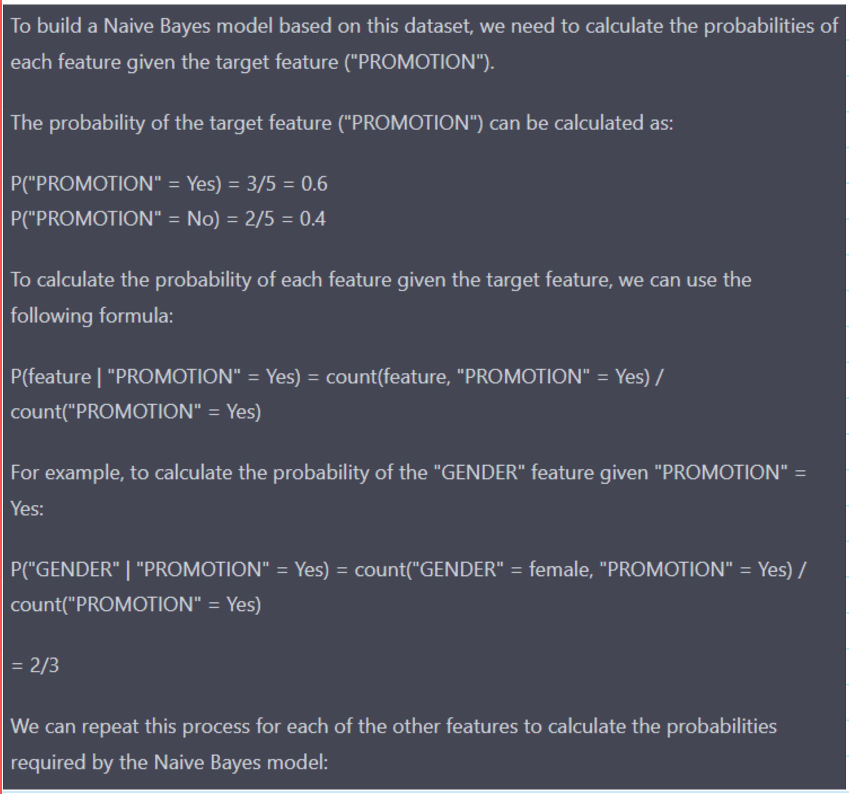
The resulting decision tree would be a tree with multiple levels, where each level represents a decision based on the feature with the highest information gain. The leaves of the tree represent the final classification of the examples in the dataset.

3. (a) The table below lists a dataset containing details about an uptake of a recent promotion (Yes or No) by 5 customers of an airline. The descriptive features included in the table are gender (GENDER), whether the customer is a frequent flyer (FREQUENT FLYER) and whether the customer has purchased a flight in the last 12 months (FLIGHT 12 MONTHS).

ID	GENDER	FREQUENT FLYER	FLIGHT 12 MONTHS	PROMOTION
1	female	true	true	Yes
2	male	false	false	Yes
3	male	false	true	No
4	female	true	true	Yes
5	female	false	false	No

(i) Calculate the probabilities required by a **Naïve-Bayes model** built based on this dataset.

(10 marks)



To build a Naive Bayes model based on this dataset, we need to calculate the probabilities of each feature given the target feature ("PROMOTION").

The probability of the target feature ("PROMOTION") can be calculated as:

P("PROMOTION" = Yes) = 3/5 = 0.6
P("PROMOTION" = No) = 2/5 = 0.4

To calculate the probability of each feature given the target feature, we can use the following formula:

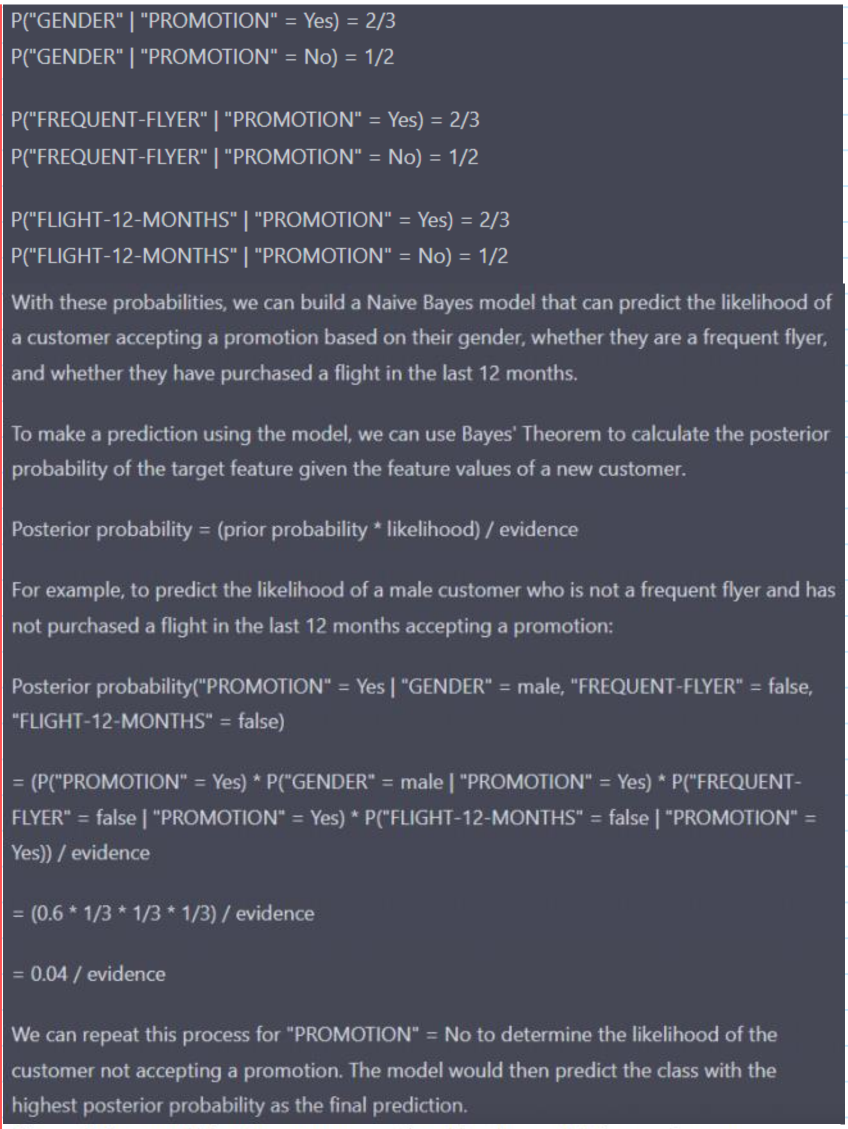
P(feature | "PROMOTION" = Yes) = count(feature, "PROMOTION" = Yes) / count("PROMOTION" = Yes)

For example, to calculate the probability of the "GENDER" feature given "PROMOTION" = Yes:

P("GENDER" | "PROMOTION" = Yes) = count("GENDER" = female, "PROMOTION" = Yes) / count("PROMOTION" = Yes)

= 2/3

We can repeat this process for each of the other features to calculate the probabilities required by the Naive Bayes model:



P("GENDER" | "PROMOTION" = Yes) = 2/3
P("GENDER" | "PROMOTION" = No) = 1/2

P("FREQUENT-FLYER" | "PROMOTION" = Yes) = 2/3
P("FREQUENT-FLYER" | "PROMOTION" = No) = 1/2

P("FLIGHT-12-MONTHS" | "PROMOTION" = Yes) = 2/3
P("FLIGHT-12-MONTHS" | "PROMOTION" = No) = 1/2

With these probabilities, we can build a Naive Bayes model that can predict the likelihood of a customer accepting a promotion based on their gender, whether they are a frequent flyer, and whether they have purchased a flight in the last 12 months.

To make a prediction using the model, we can use Bayes' Theorem to calculate the posterior probability of the target feature given the feature values of a new customer.

Posterior probability = (prior probability * likelihood) / evidence

For example, to predict the likelihood of a male customer who is not a frequent flyer and has not purchased a flight in the last 12 months accepting a promotion:

Posterior probability("PROMOTION" = Yes | "GENDER" = male, "FREQUENT-FLYER" = false, "FLIGHT-12-MONTHS" = false)

= (P("PROMOTION" = Yes) * P("GENDER" = male | "PROMOTION" = Yes) * P("FREQUENT-FLYER" = false | "PROMOTION" = Yes) * P("FLIGHT-12-MONTHS" = false | "PROMOTION" = Yes)) / evidence

= (0.6 * 1/3 * 1/3 * 1/3) / evidence

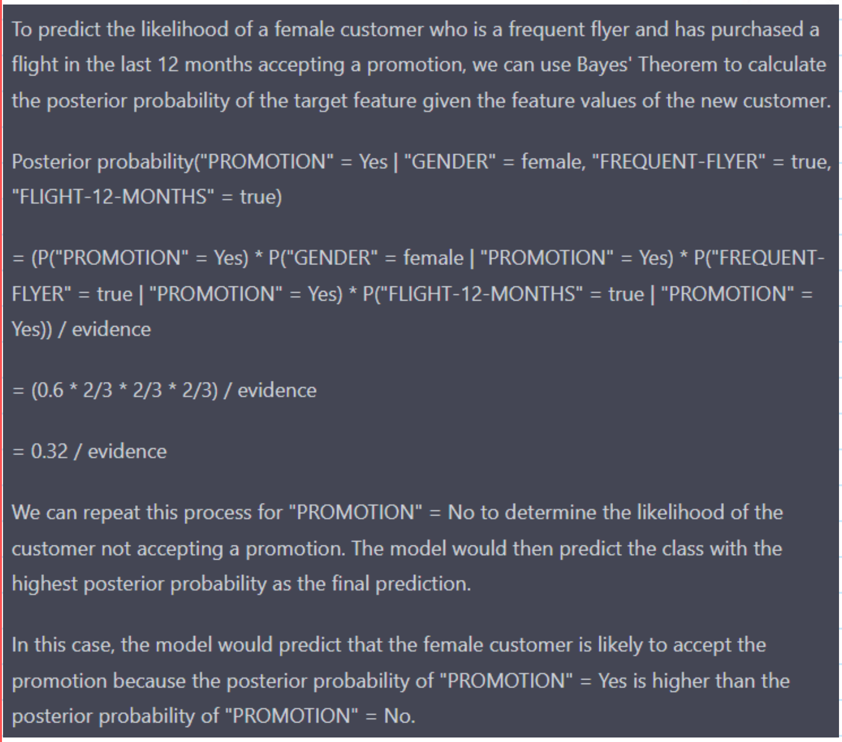
= 0.04 / evidence

We can repeat this process for "PROMOTION" = No to determine the likelihood of the customer not accepting a promotion. The model would then predict the class with the highest posterior probability as the final prediction.

(ii) What would be the outcome predicted by the model for new instance

Gender = female, Frequent flyer = true, Flight 12 months = true

(5 marks)



To predict the likelihood of a female customer who is a frequent flyer and has purchased a flight in the last 12 months accepting a promotion, we can use Bayes' Theorem to calculate the posterior probability of the target feature given the feature values of the new customer.

Posterior probability("PROMOTION" = Yes | "GENDER" = female, "FREQUENT-FLYER" = true, "FLIGHT-12-MONTHS" = true)

= (P("PROMOTION" = Yes) * P("GENDER" = female | "PROMOTION" = Yes) * P("FREQUENT-FLYER" = true | "PROMOTION" = Yes) * P("FLIGHT-12-MONTHS" = true | "PROMOTION" = Yes)) / evidence

= (0.6 * 2/3 * 2/3 * 2/3) / evidence

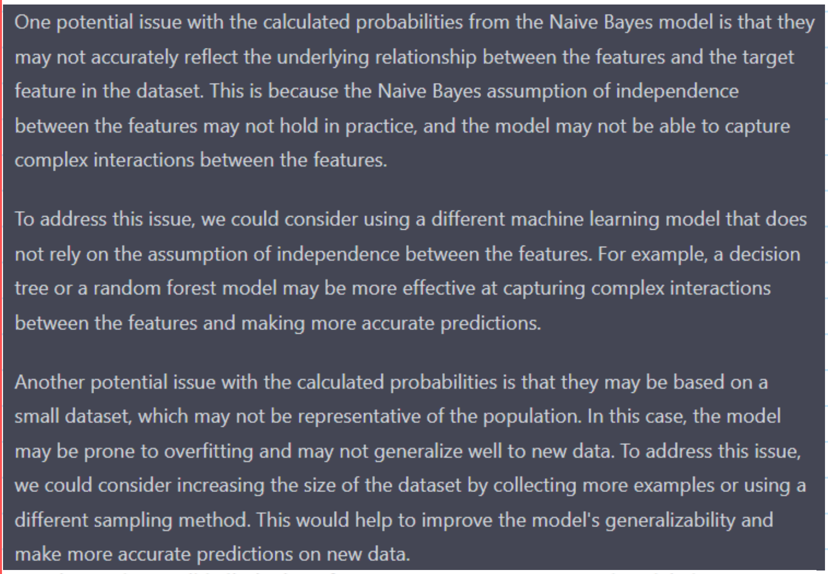
= 0.32 / evidence

We can repeat this process for "PROMOTION" = No to determine the likelihood of the customer not accepting a promotion. The model would then predict the class with the highest posterior probability as the final prediction.

In this case, the model would predict that the female customer is likely to accept the promotion because the posterior probability of "PROMOTION" = Yes is higher than the posterior probability of "PROMOTION" = No.

(iii) Do you notice any issues with the calculated probabilities in part (i)? Discuss how you could address them.

(5 marks)



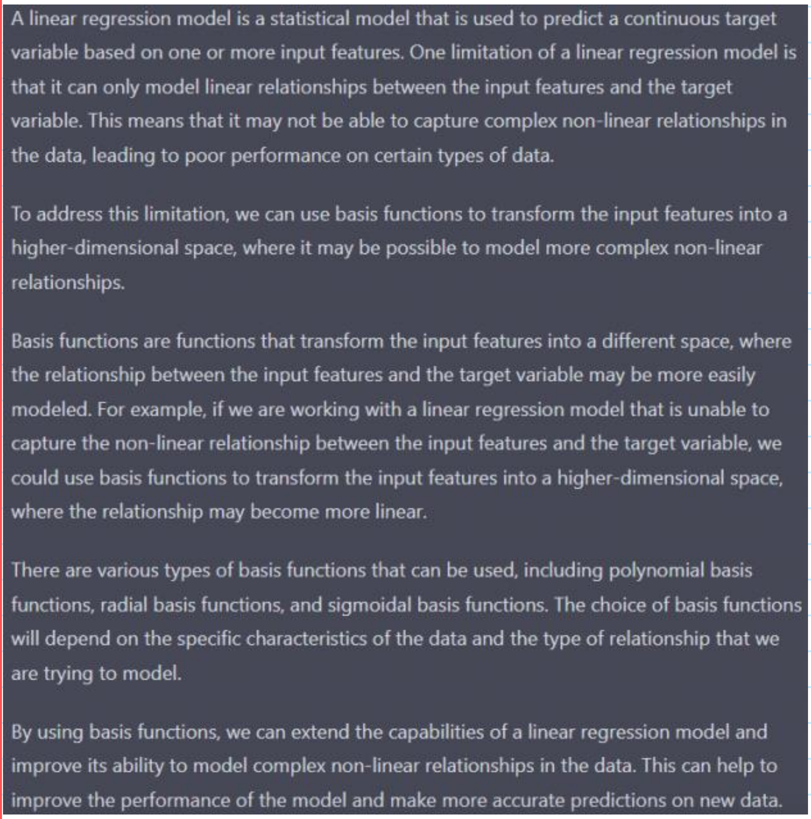
One potential issue with the calculated probabilities from the Naive Bayes model is that they may not accurately reflect the underlying relationship between the features and the target feature in the dataset. This is because the Naive Bayes assumption of independence between the features may not hold in practice, and the model may not be able to capture complex interactions between the features.

To address this issue, we could consider using a different machine learning model that does not rely on the assumption of independence between the features. For example, a decision tree or a random forest model may be more effective at capturing complex interactions between the features and making more accurate predictions.

Another potential issue with the calculated probabilities is that they may be based on a small dataset, which may not be representative of the population. In this case, the model may be prone to overfitting and may not generalize well to new data. To address this issue, we could consider increasing the size of the dataset by collecting more examples or using a different sampling method. This would help to improve the model's generalizability and make more accurate predictions on new data.

(b) Discuss the possible limitation of a **linear regression model**, and explain how **basis functions** could be used to address it.

(10 marks)



A linear regression model is a statistical model that is used to predict a continuous target variable based on one or more input features. One limitation of a linear regression model is that it can only model linear relationships between the input features and the target variable. This means that it may not be able to capture complex non-linear relationships in the data, leading to poor performance on certain types of data.

To address this limitation, we can use basis functions to transform the input features into a higher-dimensional space, where it may be possible to model more complex non-linear relationships.

Basis functions are functions that transform the input features into a different space, where the relationship between the input features and the target variable may be more easily modeled. For example, if we are working with a linear regression model that is unable to capture the non-linear relationship between the input features and the target variable, we could use basis functions to transform the input features into a higher-dimensional space, where the relationship may become more linear.

There are various types of basis functions that can be used, including polynomial basis functions, radial basis functions, and sigmoidal basis functions. The choice of basis functions will depend on the specific characteristics of the data and the type of relationship that we are trying to model.

By using basis functions, we can extend the capabilities of a linear regression model and improve its ability to model complex non-linear relationships in the data. This can help to improve the performance of the model and make more accurate predictions on new data.