FIT5047- FUNDAMENTALS OF AI

Student Name- Dishi Jain

Student ID- 30759307

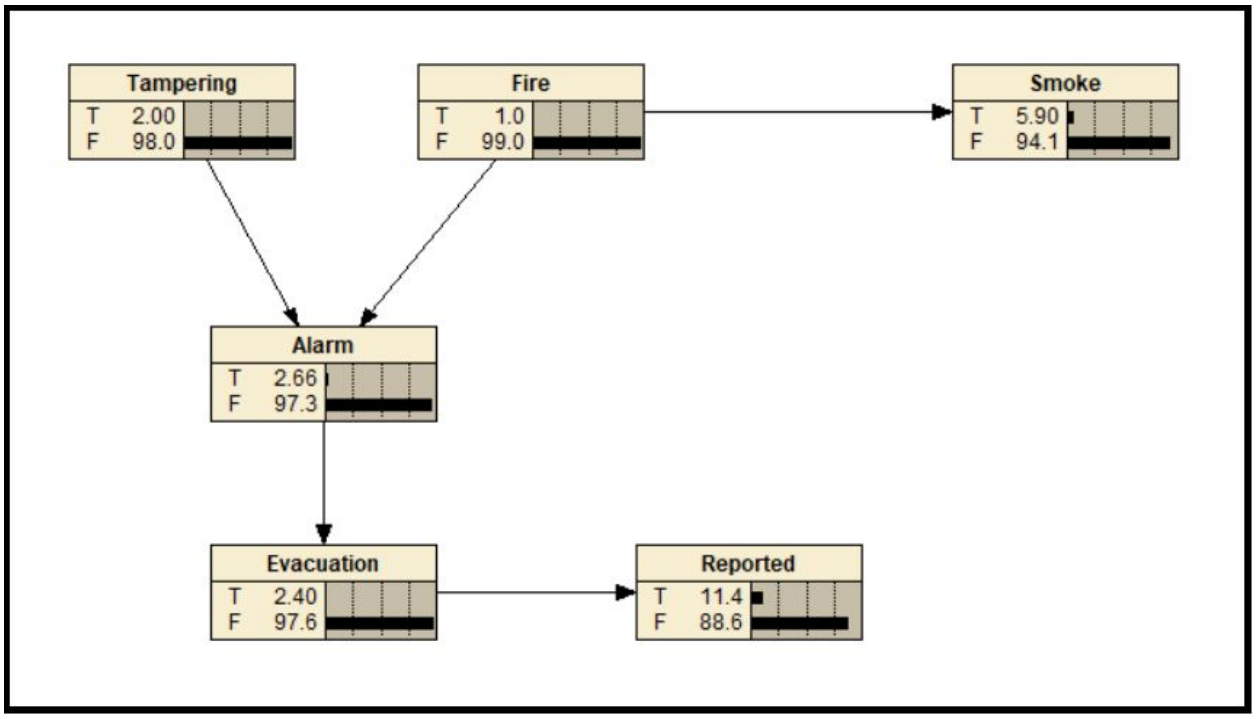
Assignment 2

Q1)

a.)

The final Bayesian Network (BN) looks like as shown in the below figure.

Figure 1 -



Justification of CPTs

Smoke Node -

This node represents whether Smoke is seen (T) or Smoke is not seen (F)

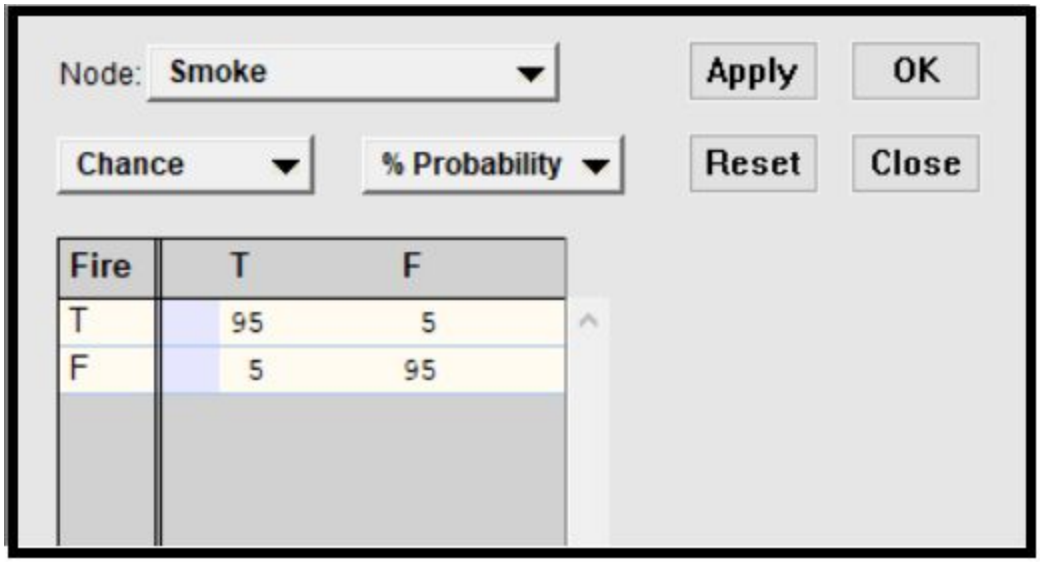
It is given to us that when there is fire then Smoke's presence probability is 0.95 and when fire isn't present then smoke's presence probability is 0.05. This data tells us that Smoke's presence is dependent on the Fire Node. Hence when Fire is True then Smoke's presence probability is 0.95. As a result, Fire = True and Smoke=True then probability is 0.95. From this we can also state the opposite condition of when Fire is True but Smoke is False. This hence can be calculated as 1-0.95 i.e. 0.05. This is because the maximum probability is 1. Hence when Fire = True and Smoke = False the probability is 0.05 . Similarly we know that when Fire = False then Smoke's presence probability is 0.05. As a result, when Fire = False and Smoke = True the probability is 0.05. From this we can also state the opposite condition of when Fire is False but Smoke is also False. This hence can be calculated as 1-0.05 i.e. 0.95. This is because the maximum probability is 1. In this manner the Smoke Node's CPT is filled as shown below -

(Values in % Probability)

FIRE	SMOKE	
	TRUE	FALSE
TRUE	95	5
FALSE	5	95

In this manner the Smoke's CPT is filled as shown in Figure 2

Figure 2 -



Evacuation Node -

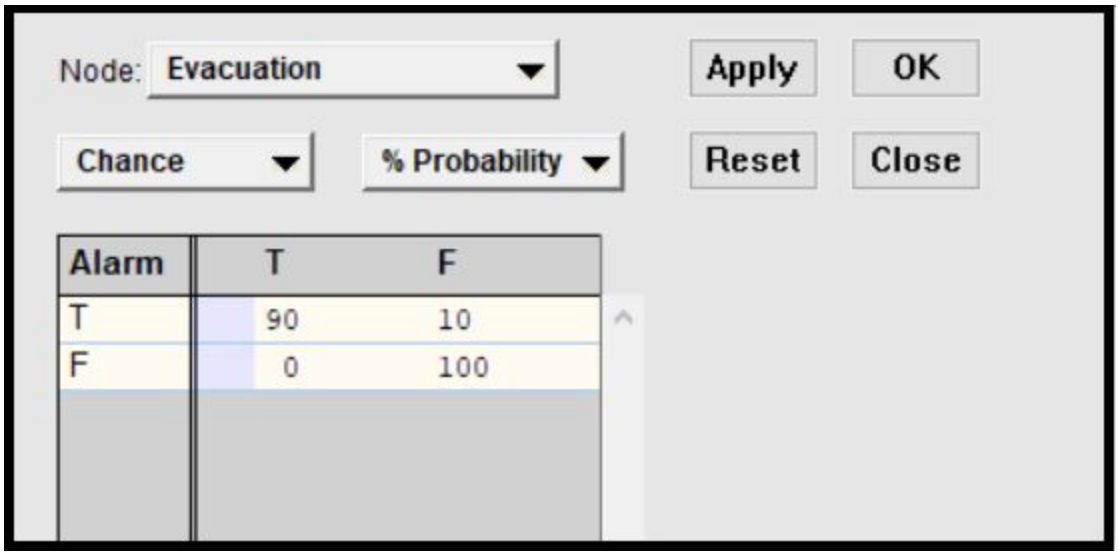
This node represents whether Evacuation is happening (T) or Evacuation is not happening(F) It is given to us that when there is a fire alarm then Evacuation's probability is 0.9 and when fire alarm isn't present then Evacuation's probability is 0. This data tells us that Evacuation is dependent on the Alarm Node. Hence when Alarm is True then Evacuation's probability is 0.9. As a result, Alarm= True and Evacuation=True then probability is 0.9. From this we can also state the opposite condition of when Alarm is True but Evacuation is False. This hence can be calculated as 1-0.9 i.e. 0.1. This is because the maximum probability is 1. Hence when Alarm= True and Evacuation = False the probability is 0.1 . Similarly we know that when Alarm= False then Evacuation's probability is 0. As a result, when Alarm= False and Evacuation= True the probability is 0. From this we can also state the opposite condition of when Alarm is False but Evacuation is also False. This hence can be calculated as 1-0 i.e. 1. This is because the maximum probability is 1. In this manner the Evacuation Node's CPT is filled as shown below -

(Values in % Probability)

ALARM	EVACUATION	
	TRUE	FALSE
TRUE	90	10
FALSE	0	100

In this manner the Smoke's CPT is filled as shown in Figure 3

Figure 3 -



Reported Node -

This node represents whether the Fire incident is Reported in the newspaper (T) or not Reported in the newspaper (F)

It is given to us that when there is an Evacuation then the probability of it being Reported is 0.7 and when Evacuation isn't seen then the probability of it NOT being Reported is 0.9. This data tells us that the Reported Node is dependent on the Evacuation Node. Hence when Evacuation is True then Reported's probability is 0.7. As a result, Evacuation = True and Reported=True then probability is 0.7. From this we can also state the opposite condition of when Evacuation is True but Reported is False. This hence can be calculated as 1-0.7 i.e. 0.3. This is because the

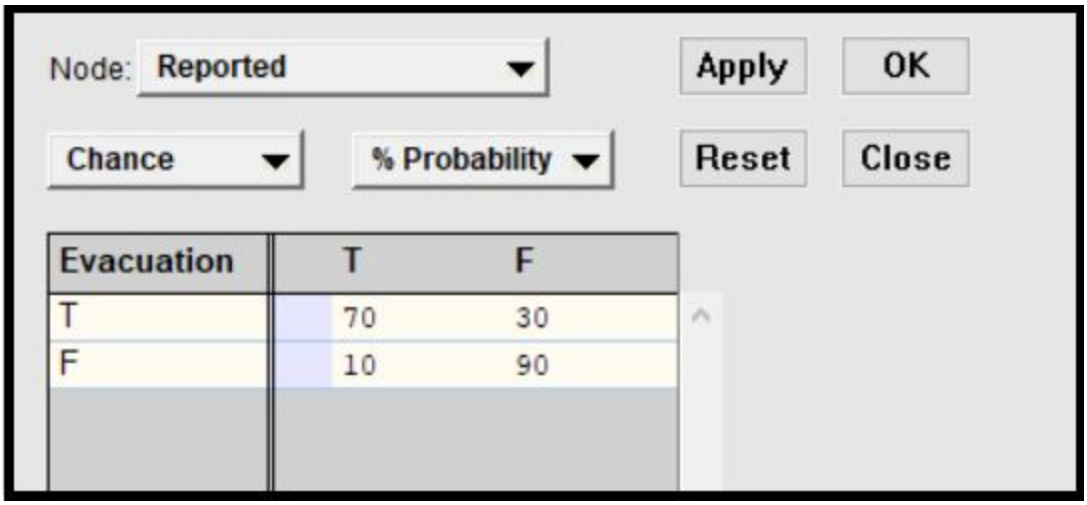
maximum probability is 1. Hence when Evacuation = True and Reported = False the probability is 0.3 . Similarly we know that when Evacuation = False then NOT being Reported's probability is 0.9. As a result, when Evacuation = False and Reported= False the probability is 0.9. From this we can also state the opposite condition of when Evacuation is False but Reported is True. This hence can be calculated as 1-0.9 i.e. 0.1. This is because the maximum probability is 1. In this manner the Reported Node's CPT is filled as shown below -

(Values in % Probability)

EVACUATION	REPORTED	
	TRUE	FALSE
TRUE	70	30
FALSE	10	90

In this manner the Smoke's CPT is filled as shown in Figure 4

Figure 4 -



Justification of BN -

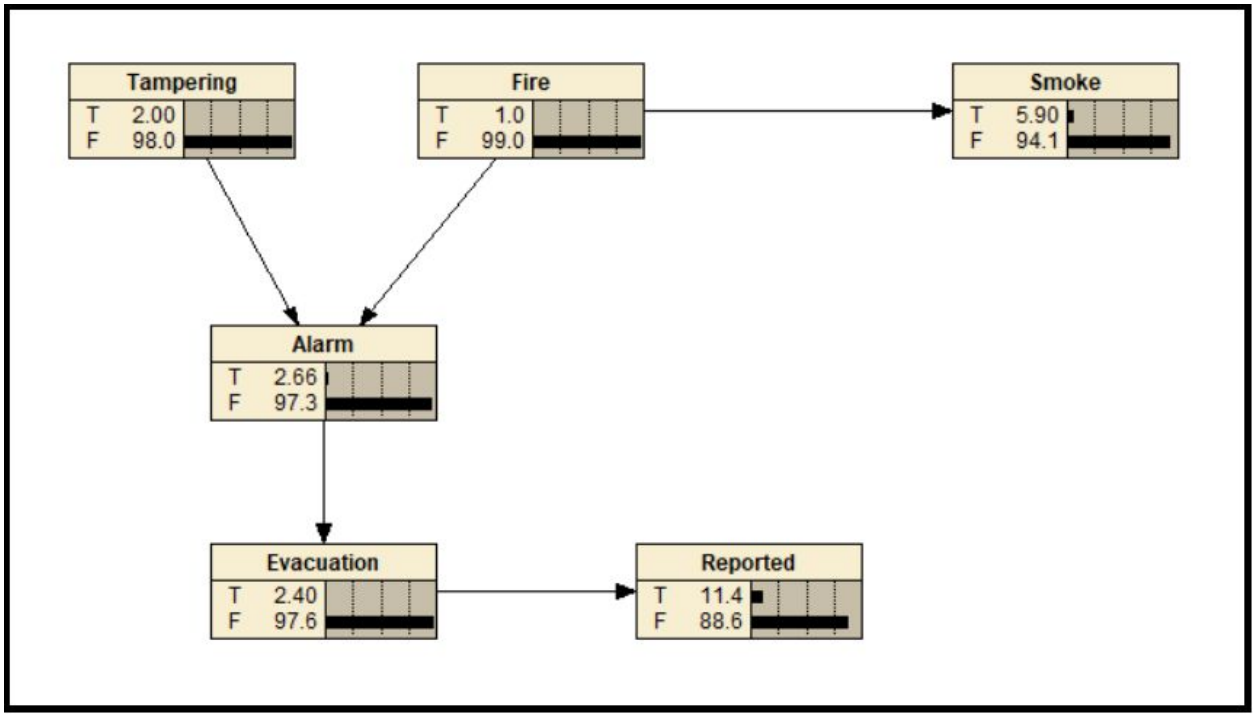
Initially we create the 3 nodes i.e. Smoke, Evacuation, Reported individually. Then from the data given we can see that Smoke is dependent on T or F of Fire. Hence, the value or probability of Smoke occurring is dependent upon the condition of Fire i.e. T or F. Hence there is a relationship between Smoke and Fire Nodes, where Smoke is dependent on Fire. Because of this there exists an edge of connection between Smoke and Fire Nodes. The direction of this edge goes from Fire to Smoke hence pointing towards Smoke as Smoke is dependent upon the condition of Fire. After connecting the two nodes we can say that Smoke is dependent on Fire.

From the data given we can see that Evacuation is dependent on T or F of Alarm. Hence, the value or probability of Evacuation occurring is dependent upon the condition of Alarm i.e. T or F. Hence there is a relationship between Evacuation and Alarm Nodes, where Evacuation is dependent on Alarm. Because of this there exists an edge of connection between Evacuation and Alarm Nodes. The direction of this edge goes from Alarm to Evacuation hence pointing towards Evacuation as Evacuation is dependent upon the condition of Alarm. After connecting the two nodes we can say that Evacuation is dependent on Alarm.

From the data given we can see that Reported is dependent on T or F of Evacuation. Hence, the value or probability of Reported is dependent upon the condition of Evacuation i.e. T or F. Hence there is a relationship between Reported and Evacuation Nodes, where Reported is dependent on Evacuation. Because of this there exists an edge of connection between Reported and Evacuation Nodes. The direction of this edge goes from Evacuation to Reported hence pointing towards Reported as Reported is dependent upon the condition of Reported . After connecting the two nodes we can say that Reported is dependent on Evacuation.

The final Bayesian Network hence can be seen below

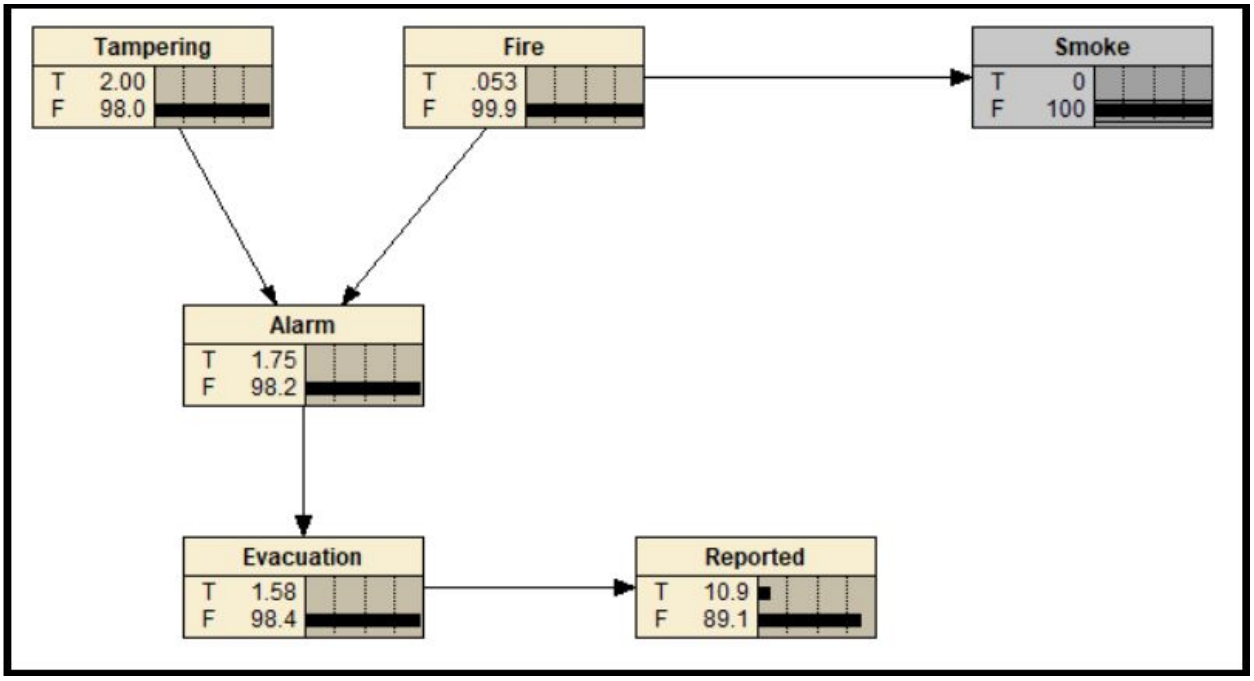
Figure 5 -



.

b.)
i.) Marginal probability that your smoke detector has been tampered with = 0.02
ii.) Marginal probability that there will be a news report tomorrow = 0.114
iii.) Assuming that there is smoke then the posterior probability that there will be a news report tomorrow = 0.193
(This is calculated by clicking on Smoke = T and observing the probability of Reported = T in BN)
iv.) Assuming that there was no fire, and that there was a news report then the posterior probability that smoke detector has been tampered with = 0.102
(This is calculated by clicking on Fire= F and Reported = T and observing the probability of Tempered = T in BN)
v.) Assume that there is no smoke then, the posterior probability that smoke detector has been tampered with = 0.02
(This is calculated by clicking on Smoke= F and observing the probability of Tempered = T in BN)

Figure 6 -

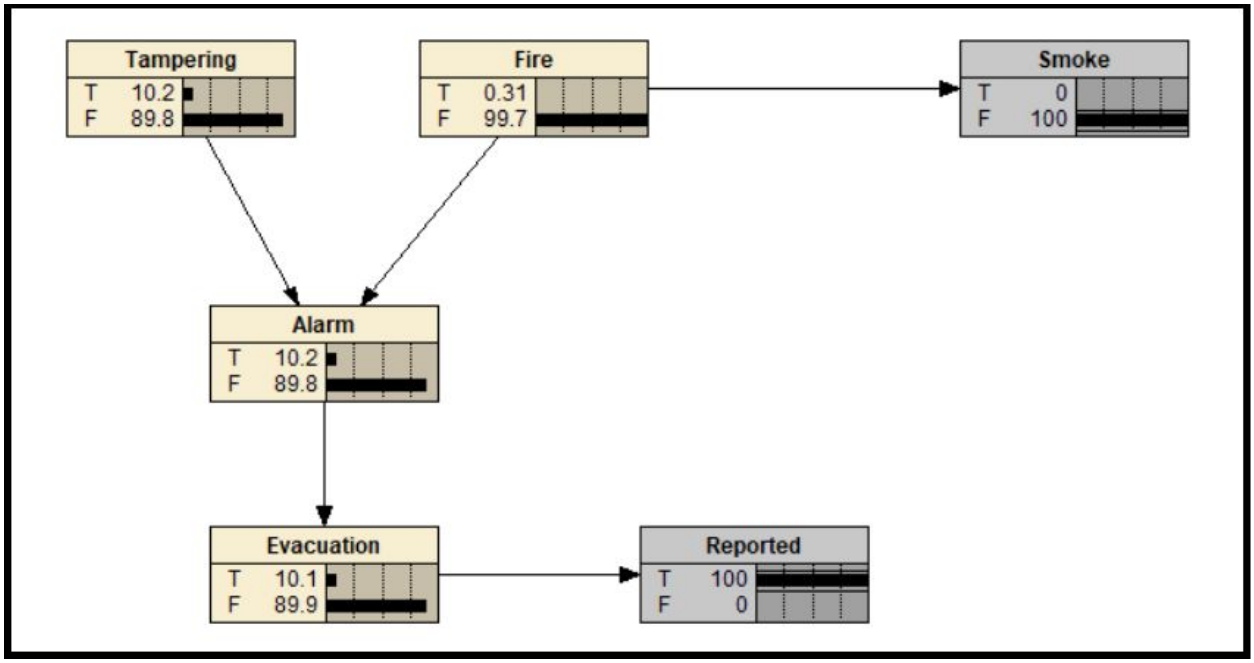


Conditional Property that helps here is the common cause of Alarm,Smoke and Fire with no evidence and the common effect of Tempering,Fire and Alarm with no evidence.Tampering and Fire are independent and Fire and smoke are dependent, as a result Tampering and Smoke are independent.

vi.) Assuming that there has been a news report and there is no smoke then the posterior probability that smoke detector has been tampered with = 0.102

(This is calculated by clicking on Reported= T and Smoke = F and observing the probability of Tempered = T in BN)

Figure 7 -



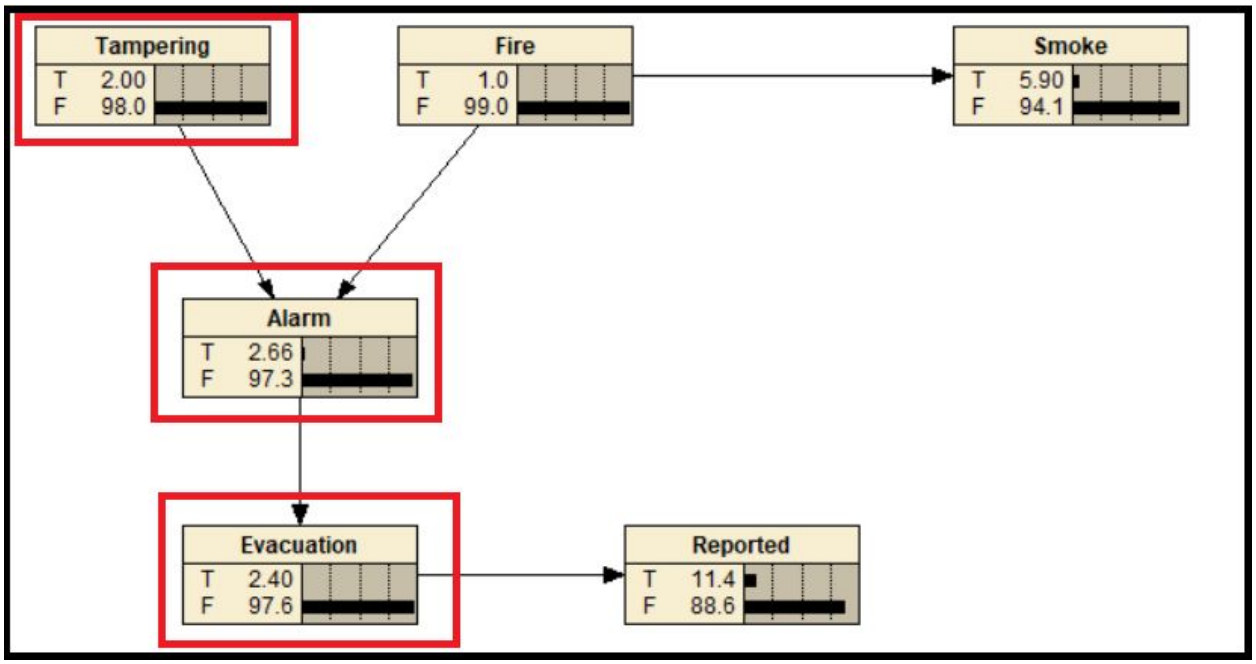
Given that the news report was observed, then observing the absence of smoke affects our belief of whether or not the smoke alarm was tampered with because if we just see Reported and Tampering then they have a Causal chains property but Tampering is also related to Smoke by a combination of Common causes and Common effect. Hence Tampering will also be affected by the Smoke's presence and is not only related to the Reported Node.

vii.) Assuming that there was no fire, that there was a news report and that there is smoke then, the posterior probability that smoke detector has been tampered with = 0.102

c.) Hypothesising that all the given cases are Independent, we will write whether the given conditions are True or False.
i.) Tampering $\perp\perp$ Evacuation

Tampering and Evacuation are forming a Causal Chain with Alarm as can be seen from the figure below. As there is no evidence hence we can say that Tampering and Evacuation are not independent. As a result they are dependent. Hence answer = False as hypothesis is not true.

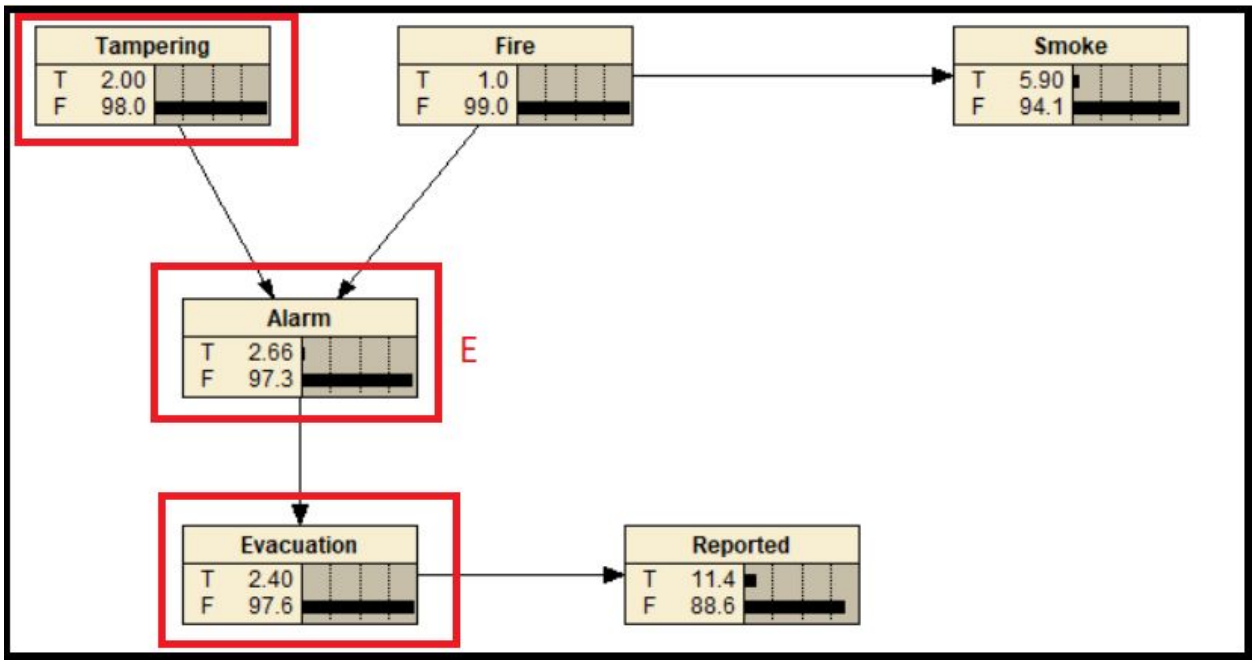
Figure 8 -



ii.) Tampering ⊥⊥ Evacuation | Alarm

Tampering and Evacuation are forming a Causal Chain with Alarm as can be seen from the figure below. Also given to us is that there is evidence of Alarm. Hence Alarm is blocking the path between Tampering and Evacuation. From using the property of the Causal Chain, we can say that Tampering and Evacuation are independent. Hence answer = True as hypothesis is true.

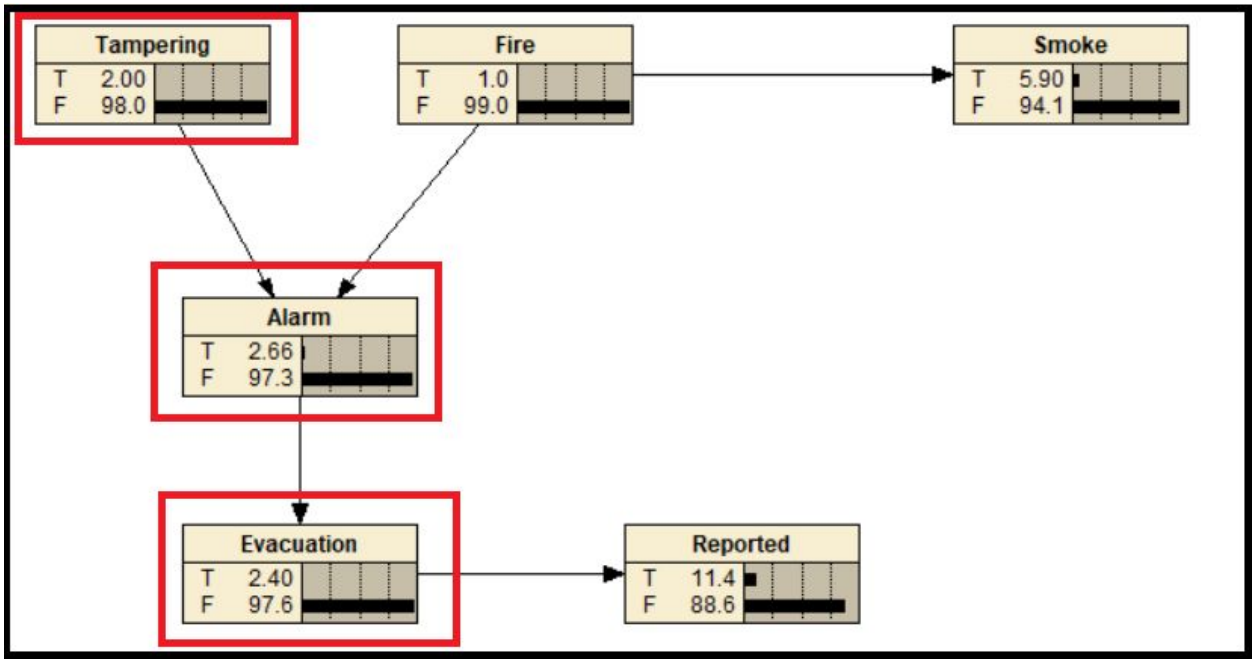
Figure 9 -


Tampering
T 2.00
F 98.0
Fire
T 1.0
F 99.0
Smoke
T 5.90
F 94.1
Alarm
T 2.66
F 97.3
E
Evacuation
T 2.40
F 97.6
Reported
T 11.4
F 88.6

iii.) Tampering ⊥⊥ Evacuation | Smoke

Tampering and Evacuation are forming a Causal Chain with Alarm as can be seen from the figure below. Also given to us is that there is evidence of Smoke. The evidence Smoke is not blocking the path between Tampering and Evacuation. As a result based on the property of the Causal Chain, we can say that Tampering and Evacuation are not independent.Hence they are dependent Hence answer = False as hypothesis is false.

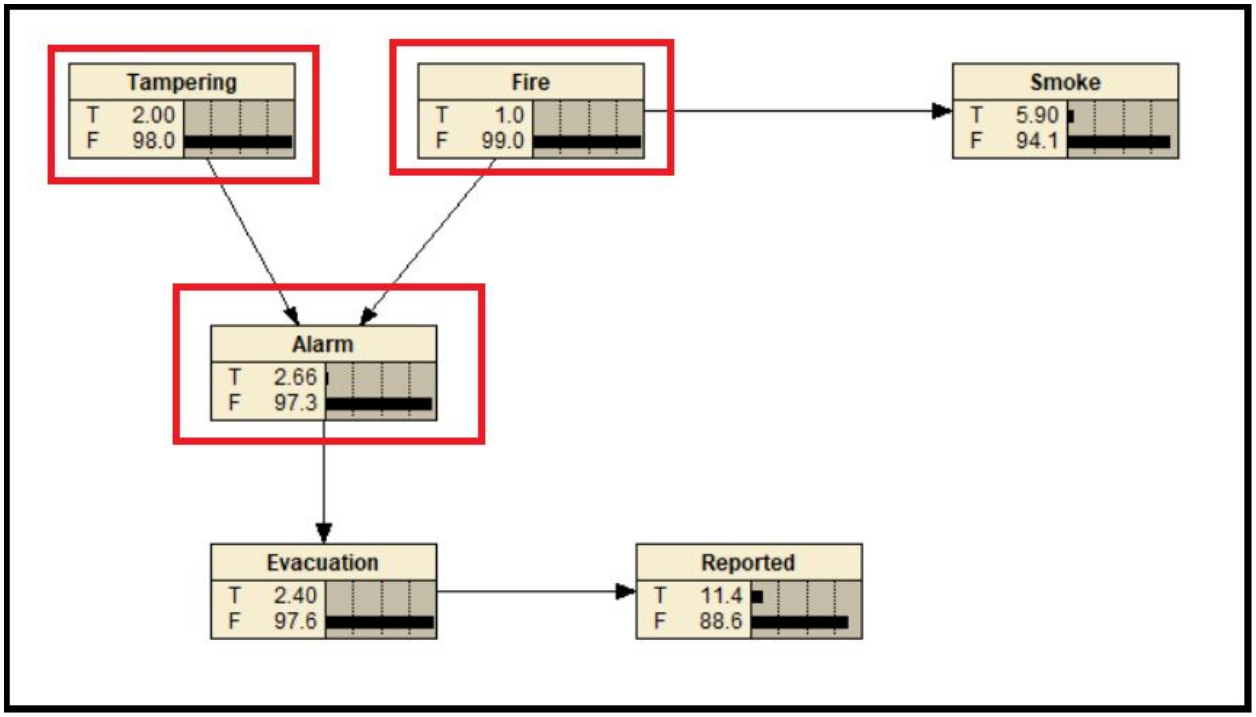
Figure 10 -



iv.) Tampering ⊥⊥ Fire

Tampering and Fire are forming a Common Effect with Alarm as can be seen from the figure below. As there is no evidence hence we can say that Tampering and Fire are independent. Hence answer = True as hypothesis is true.

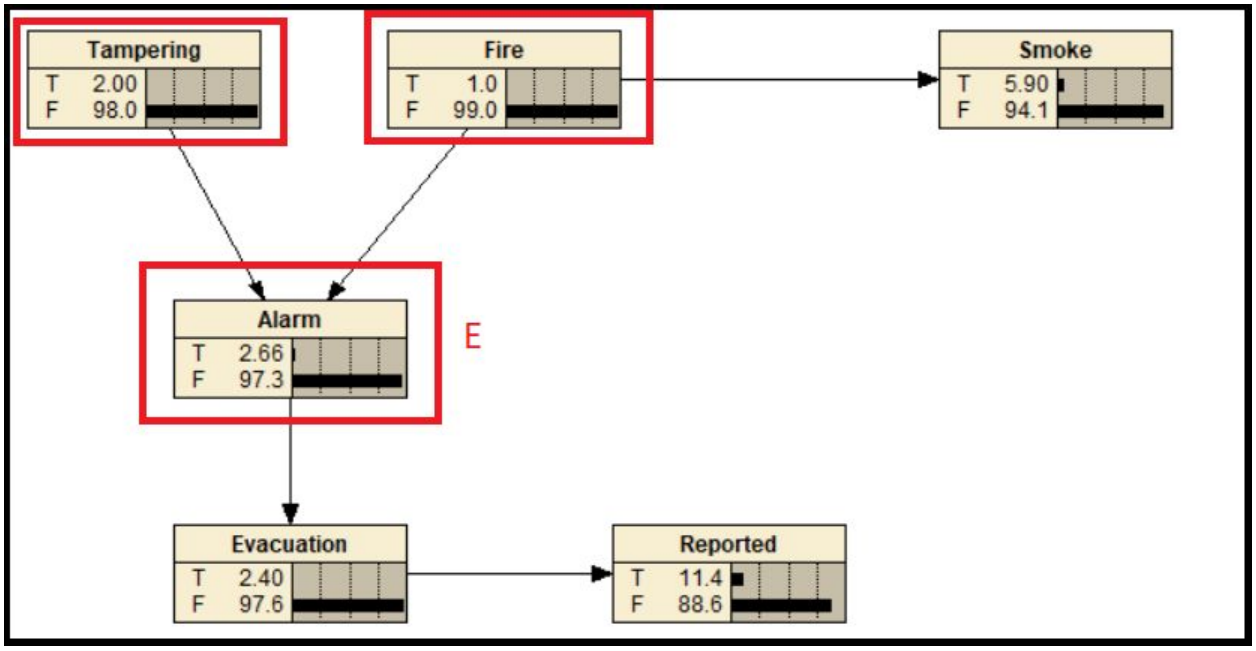
Figure 11 -



v.) Tampering ⊥⊥ Fire | Alarm

Tampering and Fire are forming a Common Effect with Alarm as can be seen from the figure below. Also given to us is that there is evidence of Alarm. From using the property of the Common Effect, we can say that Tampering and Fire are dependent when Alarm is given as the Evidence. Hence answer = False as hypothesis is false.

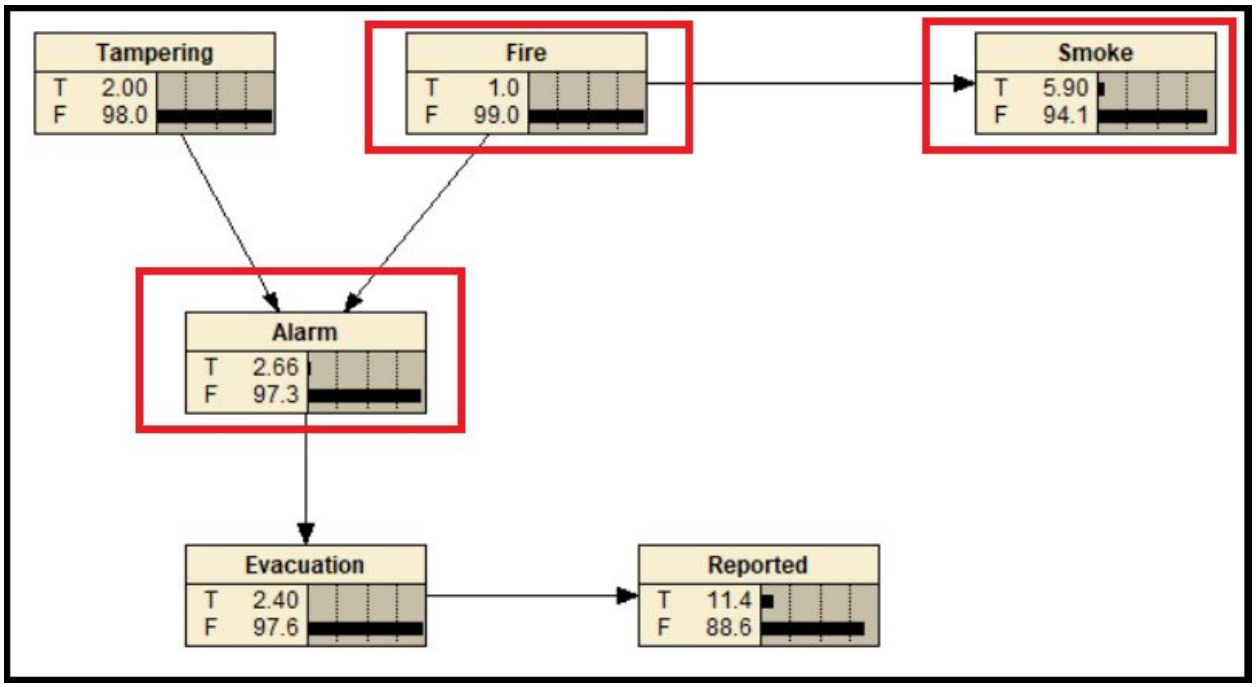
Figure 12 -



vi.)Alarm ⊥⊥ Smoke

Alarm and Smoke are forming a Common Cause with Fire as can be seen from the figure below. As there is no evidence hence we can say that Alarm and Smoke are not independent. Hence they are dependent. Hence answer = False as hypothesis is false.

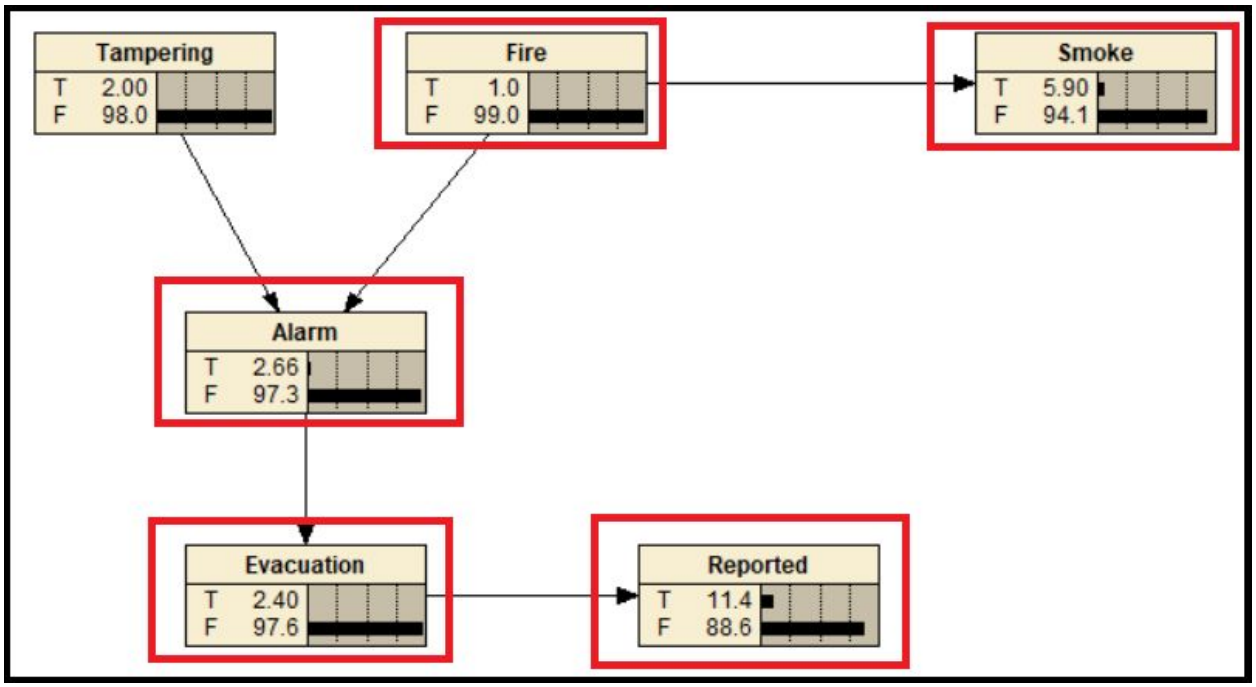
Figure 13-



vii.)Smoke ⊥⊥ Report

Smoke and Report are not forming any property directly. However SMoke and Alarm are forming a common cause with Fire with no evidence, hence Alarm and Smoke are dependent. Then taking into consideration Alarm now we can see that Alarm and Reported are forming a Causal Chain with Evacuation with no evidence. As a result Alarm and Reported are dependent. From this we can say that as Smoke and Alarm are dependent and also Alarm and Reported are dependent, we can finally say that Smoke and Reported are dependent Hence they are dependent. Hence answer = False as hypothesis is false.

Figure 13-



viii.) Smoke ⊥⊥ Tampering

Without any evidence Smoke and Tampering are independent hence hypothesis = true.
This can be explained as common effect between tampering,fire and alarm with no evidence hence by common effect tampering and fire are independent. As fire is connected with Smoke hence they both are dependent. As a result we can say that Tampering and Smoke are independent.

ix.) Smoke ⊥⊥ Tampering | Alarm

With Alarm as evidence Smoke and Tampering are dependent hence hypothesis = false.
This can be explained as common effect between tampering,fire and alarm with evidence as alarm hence by common effect tampering and fire are dependent. As fire is connected with

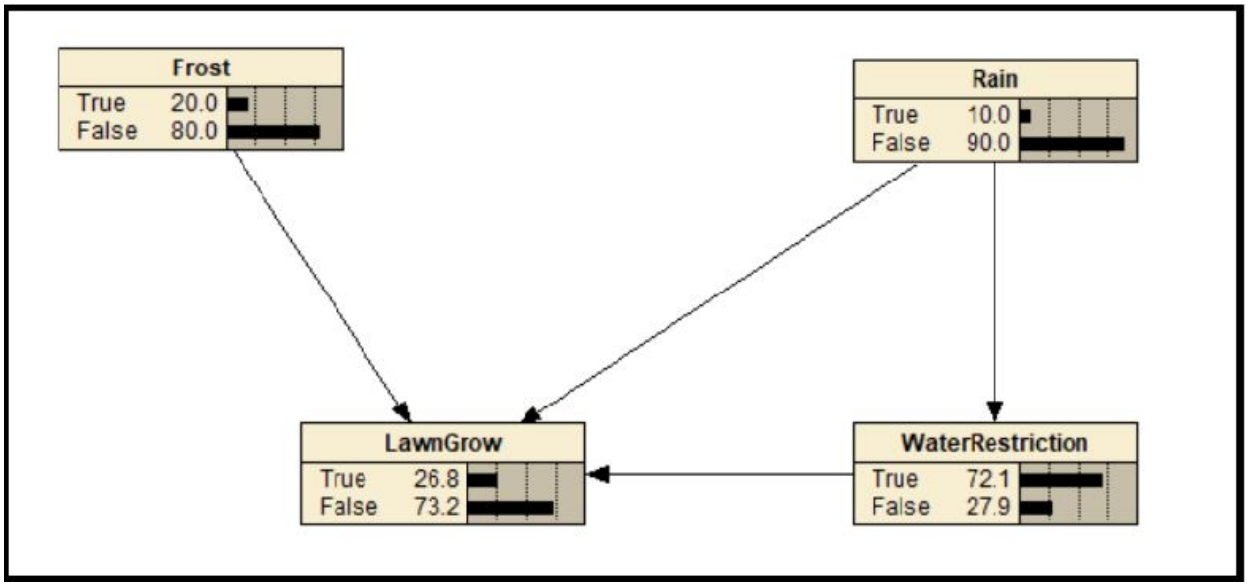
Smoke hence they both are dependent. As a result we can say that Tampering and Smoke are dependent.

x.) Smoke ⊥⊥ Tampering | Report
They are dependent hence answer = false. Smoke tampering and alarm are forming a common effect and also reported is a descendant of alarm. As reported is the evidence, hence alarm is the evidence and by common effect smoke and tampering are dependent.

Q2)

a.)The final Bayesian Network (BN) looks like as shown in the below figure.

Figure 14 -



Justification of BN -
The nodes that have to be created are Frost,Rain,LawnGrow,WaterRestriction.
The Node Frost means that whether Frost will happen(T) or not(F)
The Node Rain means that whether Rain will happen(T) or not(F)
The Node WaterRestriction means that whether Water Restriction will happen(T) or not(F)
The Node LawnGrow means that whether Lawn will Grow (T) or not(F)

From the data that is given to us we can see that Frost and Rain are independent and as a result they have their own individual T and F values.

The Node WaterRestriction's value is based on the condition of Rain's T or F. This can be understood from the line that "Furthermore, if there is no rain, the authorities could increase the

level of water restrictions,". Hence there is a relationship between WaterRestriction and Rain Nodes, where WaterRestriction is dependent on Rain. Because of this there exists an edge of connection between WaterRestriction and Rain Nodes. The direction of this edge goes from Rain to WaterRestriction hence pointing towards WaterRestriction as WaterRestriction is dependent upon the condition of Rain. After connecting the two nodes we can say that WaterRestriction is dependent on Rain.

From the data given we can see that LawnGrow is dependent on T or F of WaterRestriction , Frost, Rain Nodes, from the sentences used in the given data **"increase the level of water restrictions, meaning that Ron will be unable to water his lawn at all"** AND **", there is a small chance that the area could experience another frost before the weather warms up, which also could damage the new lawn. "** AND **"the area has been in drought for the previous 12 months. If there is no rain before summer, it will be very hard to get the new lawn to grow"** respectively. Hence, the value or probability of LawnGrow occurring is dependent upon the condition of Rain i.e. T or F. and on the condition of Frost i.e. T or F. and on the condition of WaterRestriction i.e. T or F.Hence there is a relationship between LawnGrow and Rain Nodes, AND LawnGrow and Frost Nodes AND LawnGrow and WaterRestriction Nodes.
Because of this there exists an edge of connection between LawnGrow and Rain Nodes. The direction of this edge goes from Rain to LawnGrow hence pointing towards LawnGrow as LawnGrow is dependent upon the condition of Rain.
Similarly, Because of this there exists an edge of connection between LawnGrow and Frost Nodes. The direction of this edge goes from Frost to LawnGrow hence pointing towards LawnGrow as LawnGrow is dependent upon the condition of Frost .
Similarly, Because of this there exists an edge of connection between LawnGrow and WaterRestriction Nodes. The direction of this edge goes from WaterRestriction to LawnGrow hence pointing towards LawnGrow as LawnGrow is dependent upon the condition of WaterRestriction.

b.) Talking about value assignments we can hit and try and check for different situations in our BN and using the properties of Causal Chain, Common effect and Common cause we can determine whether there exists a d-separation in the BN or not.

When taking into evidence the Frost Node, we see no property visible in the BN. Hence when Frost Node is the evidence, we cannot see any d separation. This can also be seen in Netica in the case when Frost is set to True then No Node other than LawnGrow changes its values of T or F. Also Frost makes a Common Effect to Rain with LawnGrow, however still no d-separation can be seen. As a result when Frost is taken into account i.e. the evidence, we see no d-separation.

Now taking the Node Rain as an evidence

c)
Justification of Frost Node -
This node represents whether the Frost will happen (T) or not (F)
It is given to us that there exists a small chance of frost happening before the weather warms up. As a result of this we can interpret that Frost's probability of T or F is not based or dependent on any other Node's occurrence. Hence the Frost is either T or F. Now for filling the values of probability of T and F for Frost Node, we can use the information given to us. It is given that there is only a small chance that Frost will happen and also it is the end of Winter season. As a result the chances of Frost happening is less. For this I have given a Probability Percentage value of 20% and hence Probability Percentage value of frost not happening is 100-20 = 80%. Hence in this case I have given the following values in CPT
Frost CPT is filled as shown below -

(Values in % Probability)

	FROST
True	20
False	80

Justification of Rain Node -
This node represents whether Rain will happen (T) or not (F)
It is given to us that from the last 12 months there has been no rain i.e. the area is under drought condition. As a result of this we can interpret that Rain's probability of T or F is not based or dependent on any other Node's occurrence. Hence the Rain is either T or F. Now for filling the values of probability of T and F for Rain Node, we can use the information given to us. It is given that there is only a small chance that Rain will happen. As a result the chances of Rain happening is less as it has not rained since the last 12 months.For this I have given a Probability Percentage value of 10% and hence Probability Percentage value of Rain not happening is 100-10 = 90%. Hence in this case I have given the following values in CPT
Frost CPT is filled as shown below -

(Values in % Probability)

	RAIN

True	10
False	90

Justification of WaterRestriction Node -
This node represents whether WaterRestriction will happen (T) or not (F)
It is given to us if rain is not seen i.e. Rain = F then the authorities could increase the level of Water Restriction. As a result of this we can interpret that WaterRestriction's probability of T or F is based or dependent on the Node Rain's occurrence. Hence when Rain is T or F, it will affect the quantity or probability of Water Restriction happening or not. Now for filling the values of probability of T and F for WaterRestriction Node, we can use the information given to us. It is given that there is only a small chance that Rain will happen, and if it doesn't happen then the authorities could increase the WaterRestriction. Hence when Rain = F then WaterRestriction happening is high hence an 80% chance and hence WaterRestriction not happening is low at 20% chance.However, if in any case Rain does happen, then there is a very less chance that WaterRestriction will be there as the authorities will have enough water from the rain and hence will not need to apply any WaterRestriction .Hence when Rain = T then WaterRestriction happening is very low hence at 1% chance and hence WaterRestriction not happening is high at 99% chance.
WaterRestriction CPT is filled as shown below -

(Values in % Probability)

RAIN	WATER RESTRICTION	
	T	F
T	1	99
F	80	20

Justification of LawnGrow Node -
This node represents whether Lawn will Grow (T) or not (F)
It is given to us that if there is no Rain before Summers then there is a very less chance of the Lawn Growing. Hence if Rain = F then the LawnGrow's T value is very low. As a result of this we can interpret that LawnGrow 's probability of T or F is based or dependent on the Rain's

occurrence.**Hence when Rain is T or F, it will affect the quantity or probability of LawnGrow happening or not.**
Aso it is given to us that there is a small chance that Frost will occur but if it does occur then it could again damage the lawn. Because of this we can say that the Lawn will not Grow. Hence when Frost = T it will affect the quantity or probability of LawnGrow. As a result of this we can interpret that LawnGrow ‘s probability of T or F is based or dependent on the Frost’s occurrence. **Hence when Frost is T or F, it will affect the quantity or probability of LawnGrow happening or not.**
Aso it is given to us that WaterRestrictions if applied by the authorities will also affect the Lawn Grow conditions.If WaterRestriction is applied by the authorities then the Lawn will not Grow. Hence when WaterRestriction = T it will affect the quantity or probability of LawnGrow. As a result of this we can interpret that LawnGrow ‘s probability of T or F is based or dependent on the WaterRestriction‘s occurrence. **Hence when WaterRestriction is T or F, it will affect the quantity or probability of LawnGrow happening or not.**

Taking into consideration the above three bold sentences we can say that LawnGrow’s value is dependent upon the occurrence of the Frost,Rain and WaterRestriction Nodes. All the possible combinations of this are shown below in the final CPT of LawnGrow.

When we see that Rain happens(T) and also Frost happens(T) and WaterRestriction(T) - As Rain is present hence Water Restriction’s importance is not that much. But as we also see Frost hence the Lawn will not Grow to its full extent. Hence LawnGrow 90% of times(T) and doesn’t grow 10%(F) of times.
When we see that Rain happens(T) and also Frost happens(T) and WaterRestriction(F) - As Rain is present hence Water Restriction’s importance is not that much.Also the WaterRestriction is not applied hence there is never a shortage of water. But as we also see Frost hence the Lawn will not Grow to its full extent. Hence LawnGrow 90% of times(T) and doesn’t grow 10%(F) of times.
When we see that Rain happens(T) and also Frost happens(F) and WaterRestriction(T) - As Rain is present hence Water Restriction’s importance is not that much. But as we don’t see Frost hence the Lawn will Grow to its full extent. Hence LawnGrow 100% of times(T) and doesn’t grow 0%(F) of times.
When we see that Rain happens(T) and also Frost happens(F) and WaterRestriction(F) - As Rain is present hence Water Restriction’s importance is not that much. Also the WaterRestriction is not applied hence there is never a shortage of water. But as we don’t see Frost hence the Lawn will Grow to its full extent. Hence LawnGrow 100% of times(T) and doesn’t grow 0%(F) of times.
When we see that Rain happens(F) and also Frost happens(T) and WaterRestriction(T) - As Rain is not present hence there is a shortage of water and hence Water Restriction’s importance is applied here. But as WaterRestriction is also T hence there is a shortage of water overall. But as we also see Frost hence the Lawn will not Grow to its full extent. As a result this is the worst case where even water is less and also frost is seen. Hence LawnGrow 0% of times(T) and doesn’t grow 100%(F) of times.

When we see that Rain happens(F) and also Frost happens(T) and WaterRestriction(F) - As Rain is not present hence there is a shortage of water and hence Water Restriction's importance is applied here. But as WaterRestriction is F hence there isn't a shortage of water overall. But as we also see Frost hence the Lawn will not Grow to its full extent. Hence water is present but from authorities only and not Rain, but Frost is also seen. Hence LawnGrow 80% of times(T) and doesn't grow 20%(F) of times.

When we see that Rain happens(F) and also Frost happens(F) and WaterRestriction(T) - As Rain is not present hence there is a shortage of water and hence Water Restriction's importance is applied here. But as WaterRestriction is T hence there is a shortage of water overall. But as we don't see Frost hence the Lawn will Grow to a very very small value. Hence water is not present at all but Frost is also not seen hence it can't worsen the situation. Hence LawnGrow 2% of times(T) and doesn't grow 98%(F) of times.

When we see that Rain happens(F) and also Frost happens(F) and WaterRestriction(F) - As Rain is not present hence there is a shortage of water and hence Water Restriction's importance is applied here. As WaterRestriction is F hence there is no shortage of water from the authorities. But as we don't see Frost hence the Lawn will Grow. As water is not present from Rain but is available from Authorities to some extent and also the Lawn will not be ruined by Frost hence LawnGrow 90% of times(T) and doesn't grow 10%(F) of times.

LawnGrow CPT is filled as shown below -

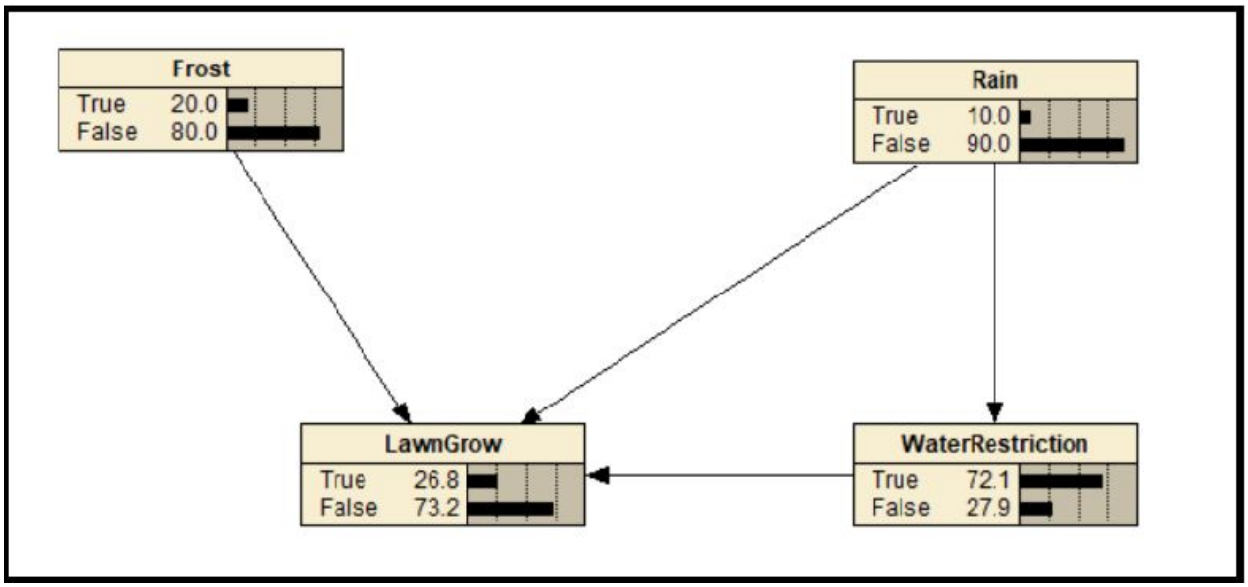
(Values in % Probability)

RAIN	FROST	WATER RESTRICTION	LAWN GROW	
			T	F
T	T	T	90	10
T	T	F	90	10
T	F	T	100	0
T	F	F	100	0
F	T	T	0	100
F	T	F	80	20
F	F	T	2	98
F	F	F	90	10

d.)

1. Probability of LawnGrow when there is no evidence.

Figure 15-



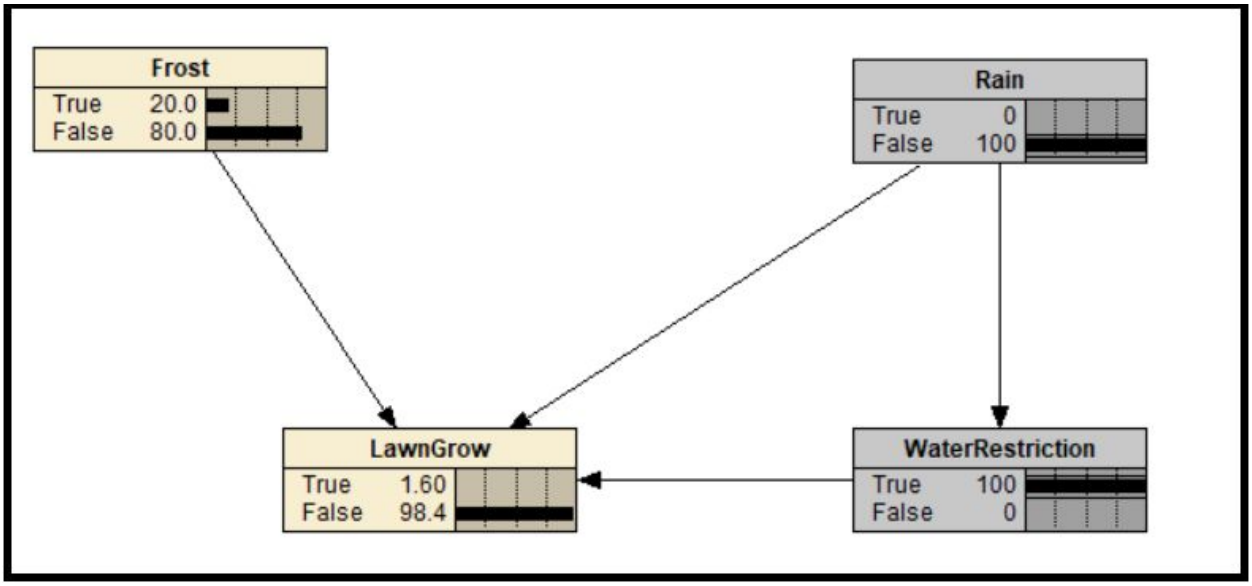
Hence when there is no evidence, the LawnGrow has a probability of T or F. For True the probability is 0.268 and for False the probability is 0.732 as can be clearly seen from the above figure.

2.

No rain and water restrictions are applied then.

In this case the probability of LawnGrow being T is 0.016 and LawnGrow being F is 0.984. This is the scenario when water is not present at all but we have no information about the Frost. Comparing this result to item 1 we can see that LawnGrow being T has reduced and LawnGrow being F has increased. This is because now we see the information or evidence that rain is not present and even the water restrictions are applied. Hence the chances of LawnGrow is very less.

Figure 16 -



3. Rain is true and Frost is True
In this case the probability of LawnGrow being T is 0.9 and LawnGrow being F is 0.1. This is the scenario when water is present but we also have Frost.Hence water is present so Lawn will grow but Frost presence will also affect the Lawn Grow . Comparing this result to item 1 we can see that LawnGrow being T has increased a lot and LawnGrow being F has decreased a lot. This is because now we see the information or evidence that rain is present and hence water is available to us. Hence the chances of LawnGrow is high.

Q3)

a.) We need to find a situation where we can propagate from Income to Rent_charged given some evidence or no evidence. By looking at the figure 17 below we can say that without any evidence we cannot propagate from income to rentcharged. Income , Property_area and Housing_prices form a common effect without any evidence AND housing_prices, tenant and rent_charged also form a common effect without any

evidence. Hence income and housing price are independent and similarly housing price and rent charged are independent. When we take evidence of property_area and tenant then we can propagate from income to rent_charged. This can be seen from figures 18 and 19 as changes in the value of rent_charged when income is present and absent. Hence by common effect income and housing_prices become dependent and similarly housing_prices and rent_charged become dependent. As a result income and rent_charged are dependent and we can propagate through them.

Figure 17-

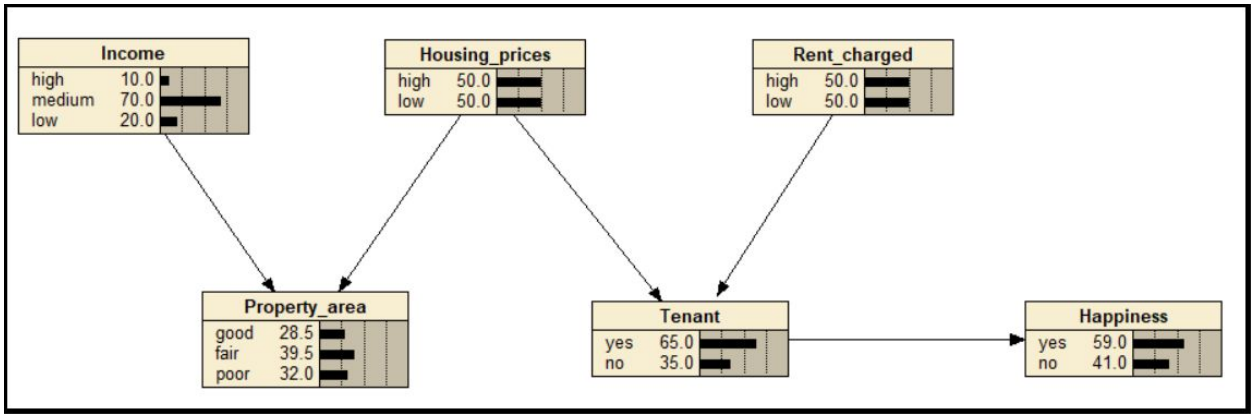


Figure 18 -

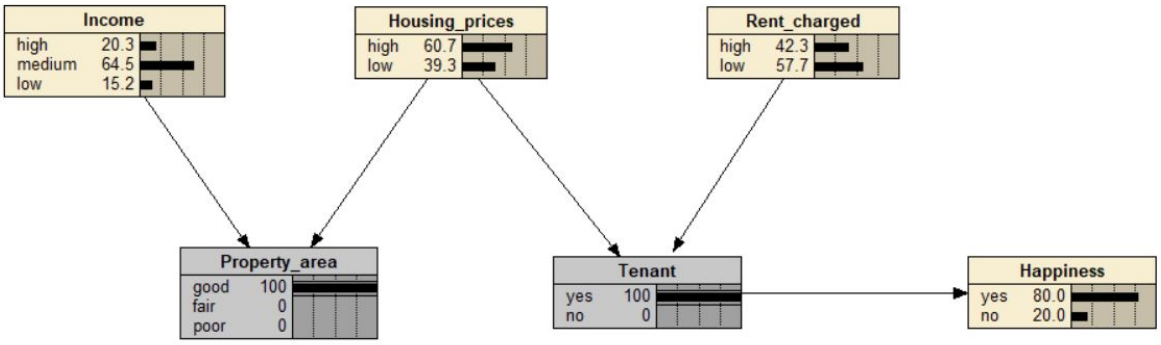
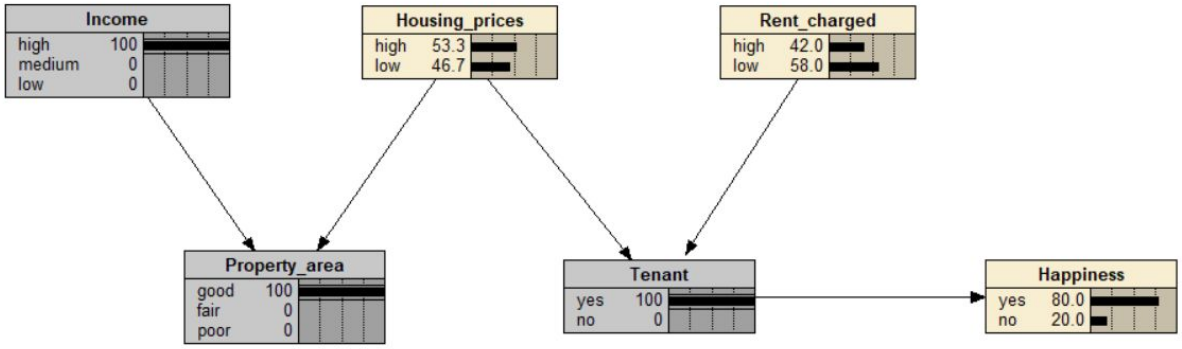


Figure 19 -



Also when Property area and happiness are taken as evidence again we see a similar effect as above as Happiness is a descendant of Tenant. Hence again by common effect income and housing_prices become dependent and similarly housing_prices and rent_charged become dependent. As a result income and rent_charged are dependent and we can propagate through them. This can be seen from figures 20 and 21, where value of rent_charged changes when changes are made to income.

Figure 20 -

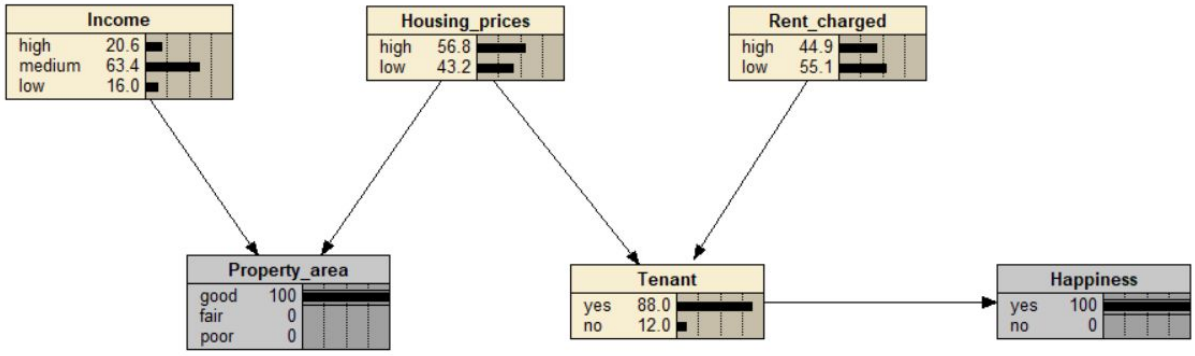
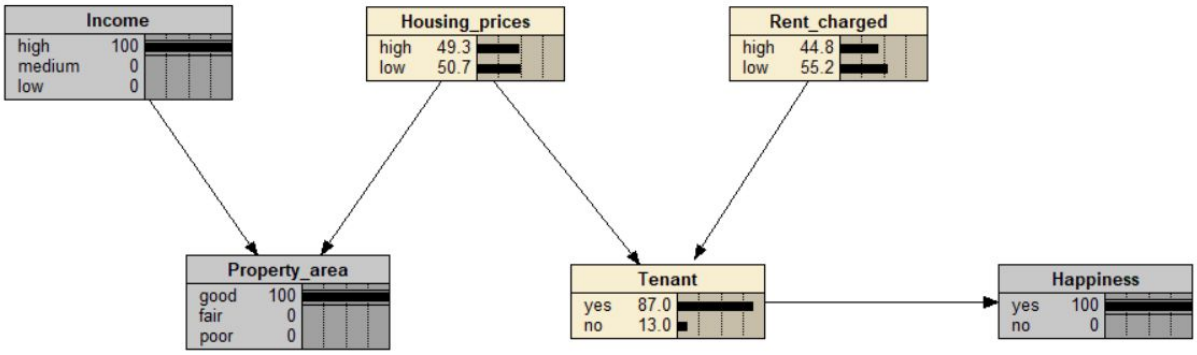


Figure 21 -



b.) Originally Happiness and property_area are dependent to each other as common cause exist between housing_prices, property_area and tenant with no evidence so property area and tenant are dependent and as happiness is a descendent of tenant hence we can say that happiness and property_area can be propagated easily.
The only case where we see that we won't be able to propagate is when Housing price is the evidence. This will d separate property area and tenant. As a result we won't be able to propagate between happiness and property area as happiness is a descendant of tenant.
This is shown in figure 22 and 23 and value of happiness remains the same.
Figure 22-

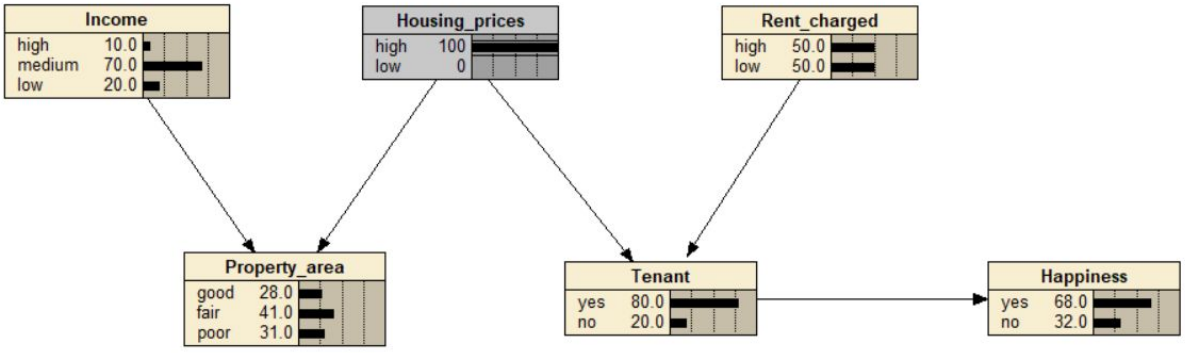
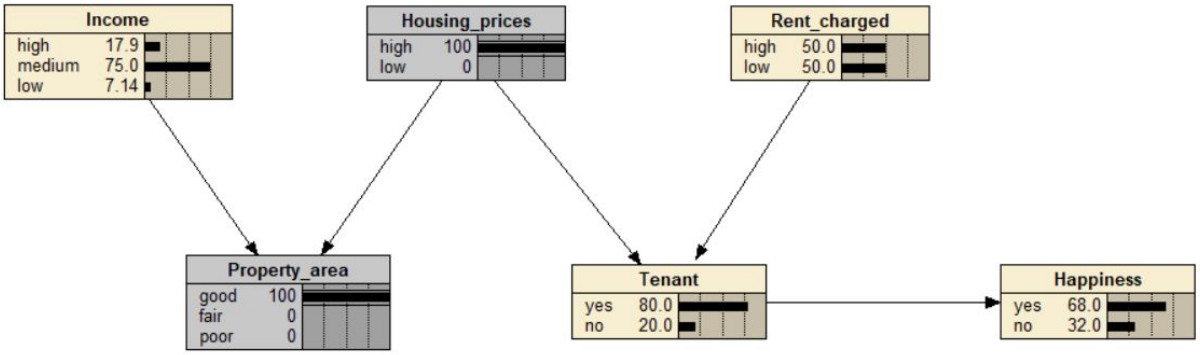


Figure 23 -



Also we can see that when Tenant is the evidence then again we cannot propagate between happiness and property area. This is because tenant as an evidence then housing price and happiness are independent due to causal chain and as property area is linked with housing price hence property area and happiness are also becoming independent. Hence we won't be able to propagate between the two in this case. This can be seen from figures 24 and 25, where value of happiness doesn't change.

Figure 24-

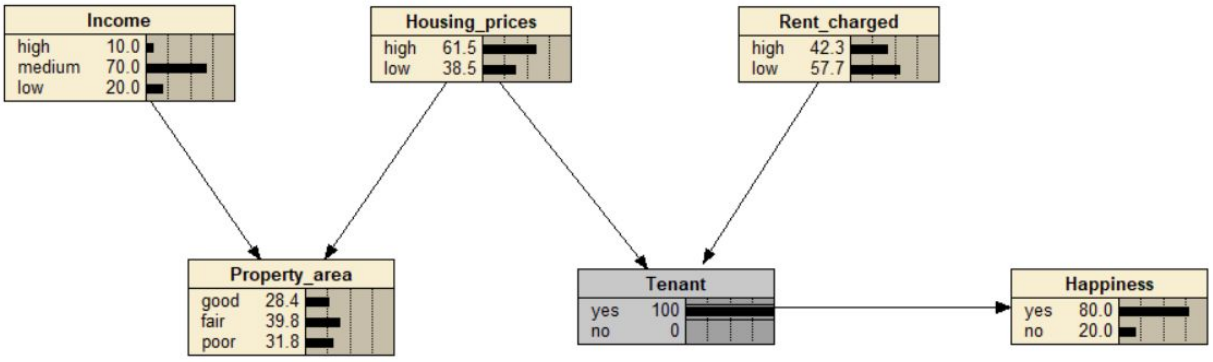
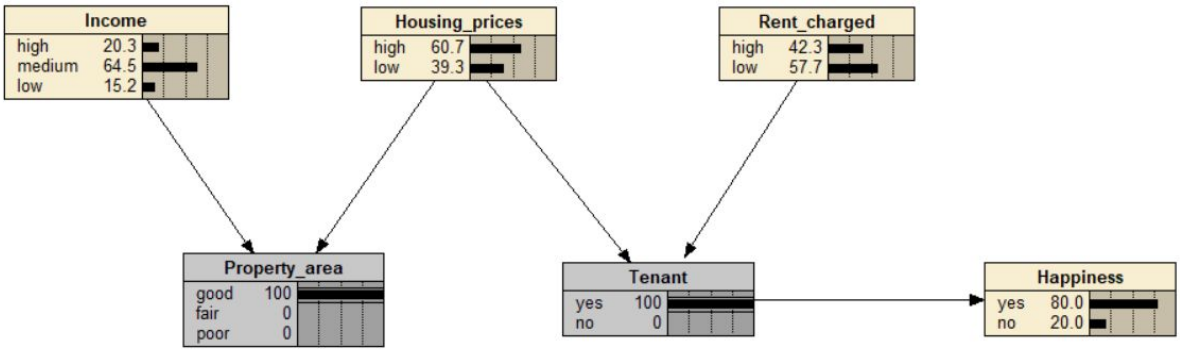
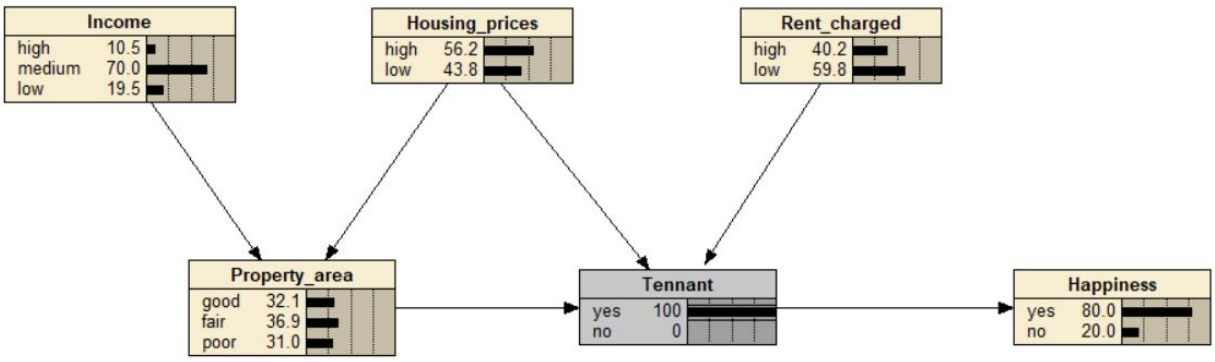


Figure 25-



c.) In this case we can see that when no evidence is given then definitely the income and rent charged are independent and as a result cannot propagate. Common effect is seen as explained above in part a and hence the propagation cannot happen. It can only happen when Tenant node is set as the evidence. This is because income and property area are connected so they are dependent. The common effect between housing price, property area and tenant with evidence tenant tells us from common effect that property area and housing price are dependent and housing price and rent charged are also dependent due to common effect with tennant. Hence Income and rent charged are dependent and we can propagate through them.
Figure 26 -



Similarly for case two to propagate between happiness and property area, we can see that when Tennant is set as evidence then forming a causal chain with property area and happiness they become independent. For the rest of the cases they are dependent and can be easily propagated. This can be seen in the figure below figure 27 and 28 when tenant is evidence then property area and happiness become independent as value of happiness doesn't change when changes are made to property area

Figure 27 -

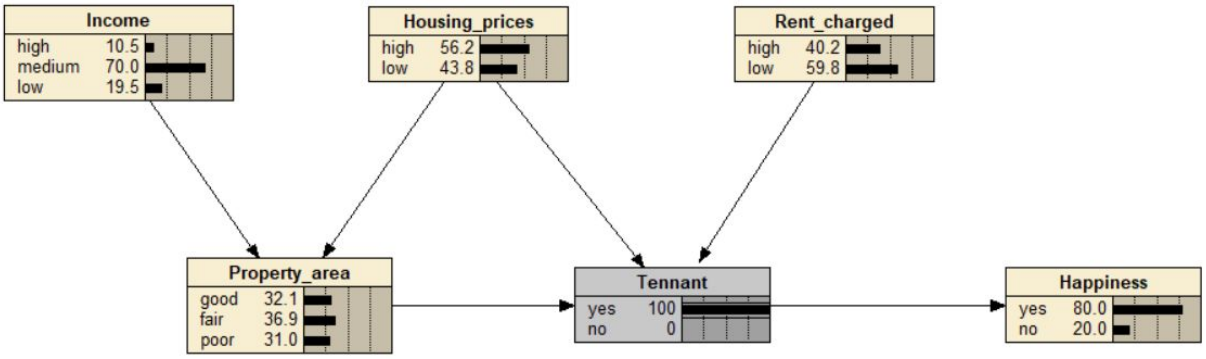
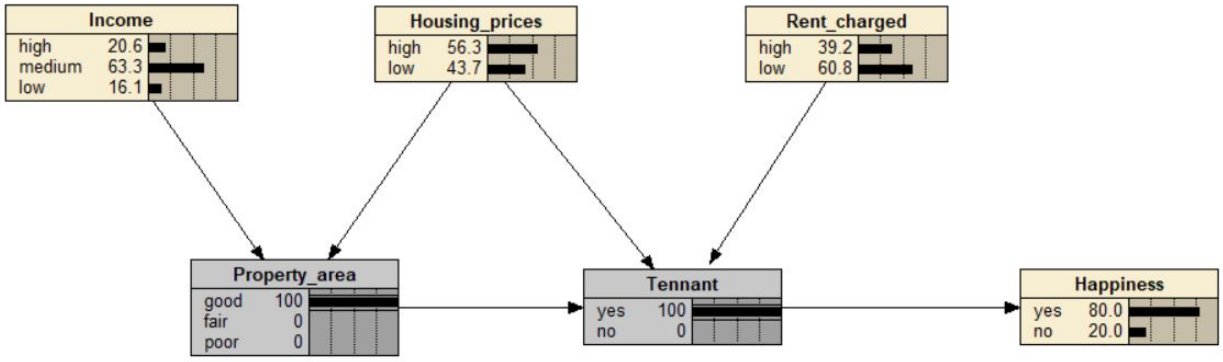


Figure 28 -

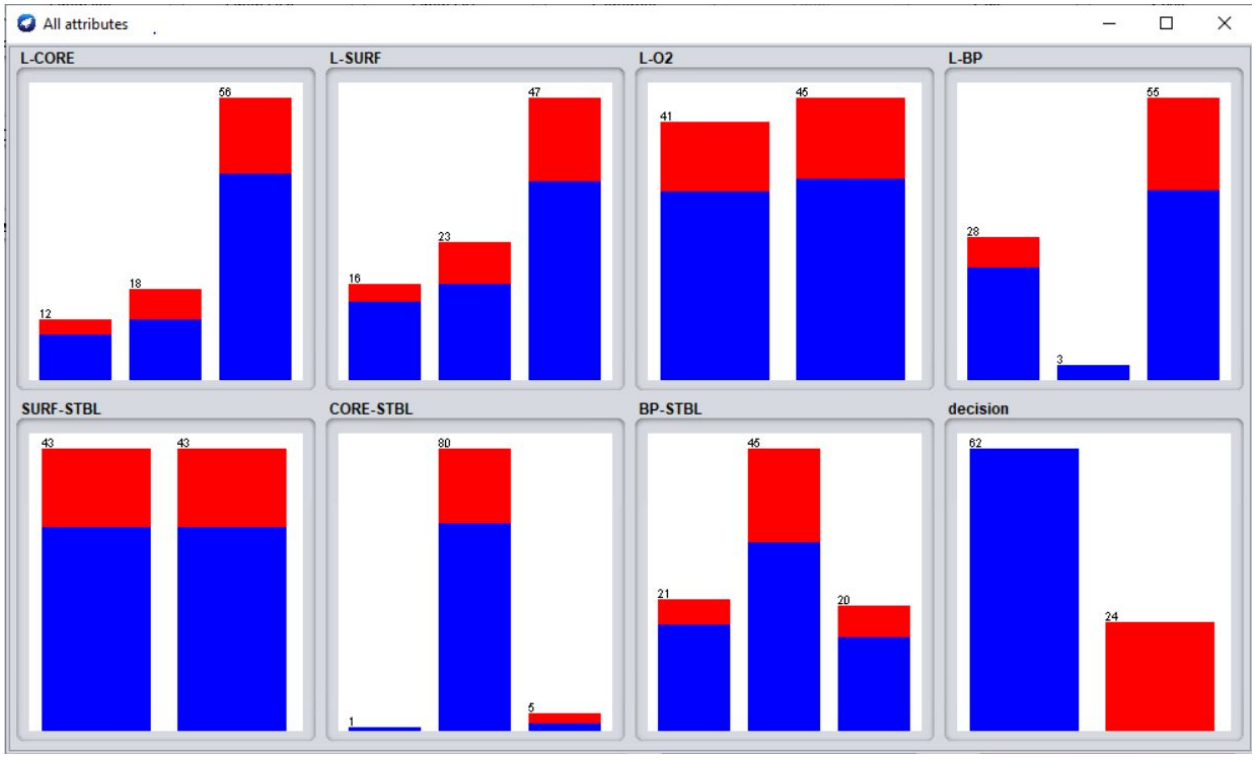


Question 4: Classification, Decision Trees, Na¨ıve Bayes, k-NN, Weka

1. Using the visualisation tool in Weka, we can see the different variables including the target variable “decision”.
    a. The variable L-CORE (i.e. core or inside temperature) has three ranges of values which are low, medium, high where each contributes to both the decision ranges of A or S.
    b. The variable L-SURF (i.e. surface temperature) has three ranges of values which are low, medium, high where each contributes to both the decision ranges of A or S.
    c. The variable L-O2 (i.e. oxygen level) has two ranges of values which are excellent,good where each contributes to both the decision ranges of A or S.
    d. The variable L-BP (i.e. blood pressure value) has three ranges of values which are low, medium, high where high and medium contribute to both the decision ranges of A or S, but low contributes to just the decision A.
    e. The variable SURF-STBL (i.e. stability of surface temperature) has two ranges of values which are stable,unstable where each contributes to both the decision ranges of A or S.
    f. The variable CORE-STBL (i.e.stability of core or inside temperature) has three ranges of values which are mod-stable,stable,unstable where stable and unstable contribute to both the decision ranges of A or S, but mod-stable contributes to just decision A.
    g. The variable BP-STABL (i.e. stability of blood pressure) has three ranges of values which are mod-stable,stable,unstable where each contributes to both the decision ranges of A or S.

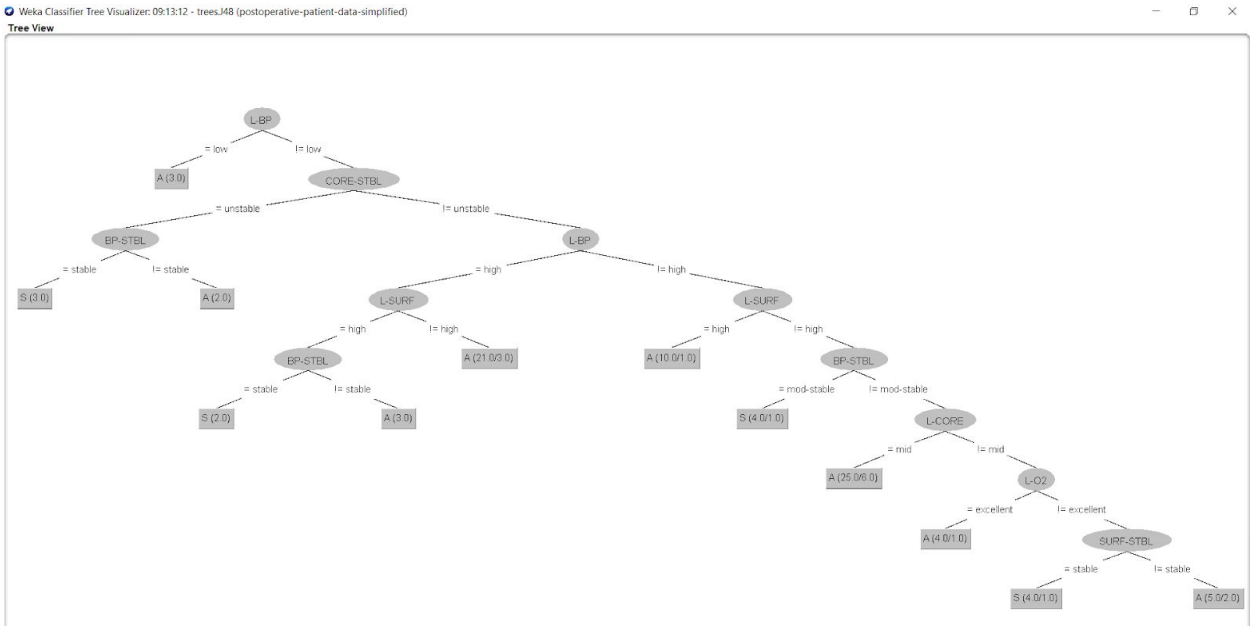
The variables that appear to be important are L-BP and CORE-STBL. This is because both these variables appear to discriminate between the classes and give a more significance to the decision A i.e. decision of staying in hospital. Hence, by taking into consideration these variables we can come up with a decision easily. These variables are hence considered to be more significant than the other variables.

Figure 1



2. a)
   For J48-
   binarySplits - TRUE
   Unpruned:- TRUE

   Figure 2-



```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -U -B -M 2
Relation:     postoperative-patient-data-simplified
Instances:    86
Attributes:   8
              L-CORE
              L-SURF
              L-O2
              L-BP
              SURF-STBL
              CORE-STBL
              BP-STBL
              decision
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

J48 unpruned tree
------------------

L-BP = low: A (3.0)
L-BP != low
|   CORE-STBL = unstable
|   |   BP-STBL = stable: S (3.0)
|   |   BP-STBL != stable: A (2.0)
|   CORE-STBL != unstable
|   |   L-BP = high
|   |   |   L-SURF = high
|   |   |   |   BP-STBL = stable: S (2.0)
|   |   |   |   BP-STBL != stable: A (3.0)
|   |   |   L-SURF != high: A (21.0/3.0)
|   |   L-BP != high
|   |   |   L-SURF = high: A (10.0/1.0)
|   |   |   L-SURF != high
|   |   |   |   BP-STBL = mod-stable: S (4.0/1.0)
|   |   |   |   BP-STBL != mod-stable
|   |   |   |   |   L-CORE = mid: A (25.0/6.0)
|   |   |   |   |   L-CORE != mid
|   |   |   |   |   |   L-O2 = excellent: A (4.0/1.0)
|   |   |   |   |   |   L-O2 != excellent
|   |   |   |   |   |   |   SURF-STBL = stable: S (4.0/1.0)
|   |   |   |   |   |   |   SURF-STBL != stable: A (5.0/2.0)

Number of Leaves  :     12

Size of the tree :      23


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          59               68.6047 %
Incorrectly Classified Instances        27               31.3953 %
Kappa statistic                          0.0923
Mean absolute error                      0.3984
Root mean squared error                  0.5197
Relative absolute error                 98.3033 %
Root relative squared error            115.7088 %
Total Number of Instances               86

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 0.871     0.792     0.740       0.871   0.800      0.099    0.440     0.675     A
                 0.208     0.129     0.385       0.208   0.270      0.099    0.440     0.283     S
Weighted Avg.    0.686     0.607     0.641       0.686   0.652      0.099    0.440     0.566

=== Confusion Matrix ===

  a  b   <-- classified as
 54  8 |  a = A
 19  5 |  b = S
```

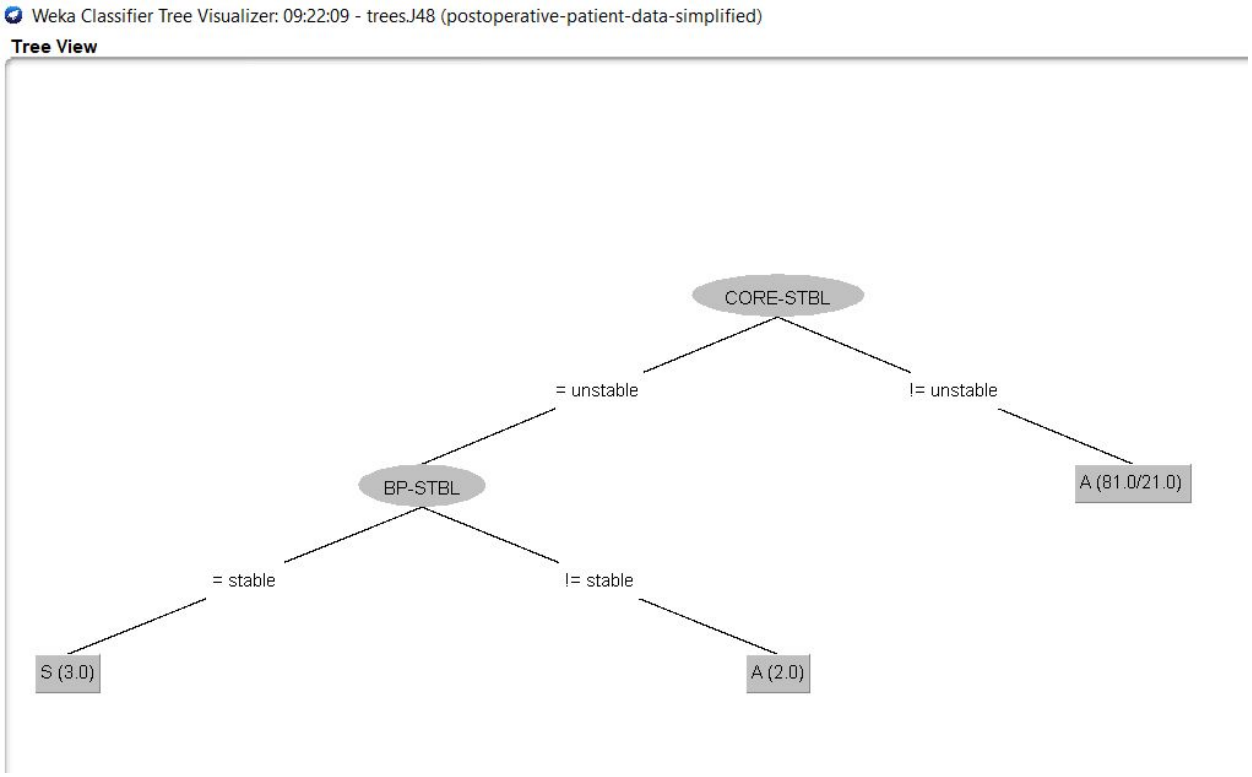
For J48-
binarySplits - FALSE
Unpruned:- FALSE

Figure 3 -

```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:     postoperative-patient-data-simplified
Instances:    86
Attributes:   8
              L-CORE
              L-SURF
              L-O2
              L-BP
              SURF-STBL
              CORE-STBL
              BP-STBL
              decision
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

J48 pruned tree
------------------
: A (86.0/24.0)

Number of Leaves  :     1

Size of the tree :      1


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          61               70.9302 %
Incorrectly Classified Instances        25               29.0698 %
Kappa statistic                         -0.0228
Mean absolute error                      0.4075
Root mean squared error                  0.4554
Relative absolute error                100.5513 %
Root relative squared error            101.3916 %
Total Number of Instances               86

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.984    1.000    0.718      0.984    0.830      -0.067   0.439     0.696     A
                 0.000    0.016    0.000      0.000    0.000      -0.067   0.439     0.256     S
Weighted Avg.    0.709    0.725    0.517      0.709    0.598      -0.067   0.439     0.573

=== Confusion Matrix ===

  a  b   <-- classified as
 61  1 |  a = A
 24  0 |  b = S
```

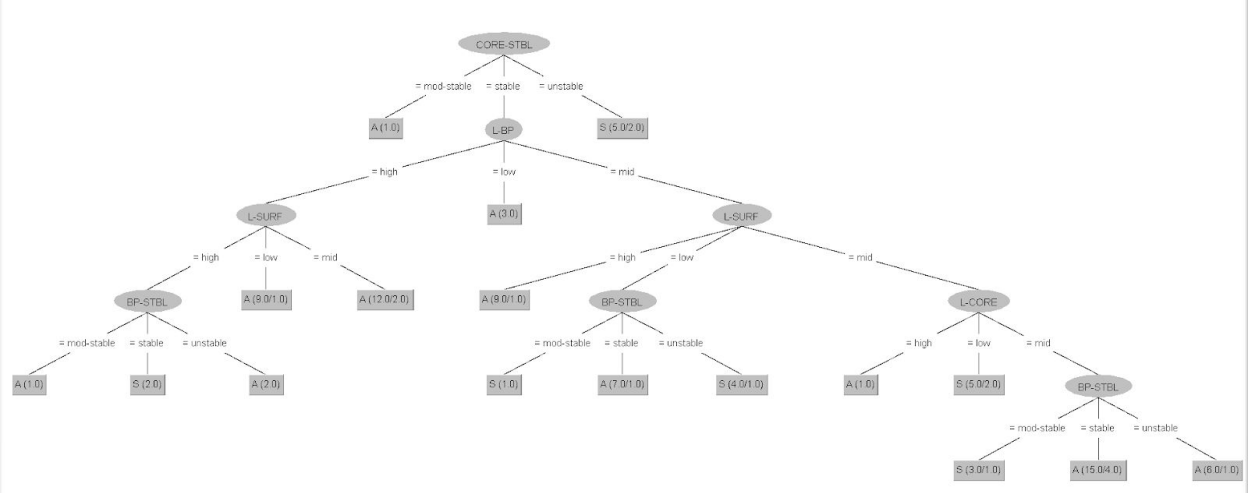
For J48-
binarySplits - TRUE
Unpruned:- FALSE

Figure 4 -



```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -B -C 0.25 -M 2
Relation:     postoperative-patient-data-simplified
Instances:    86
Attributes:   8
              L-CORE
              L-SURF
              L-O2
              L-BP
              SURF-STBL
              CORE-STBL
              BP-STBL
              decision
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

J48 pruned tree
------------------

CORE-STBL = unstable
|   BP-STBL = stable: S (3.0)
|   BP-STBL != stable: A (2.0)
CORE-STBL != unstable: A (81.0/21.0)

Number of Leaves  :     3

Size of the tree :      5


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          61               70.9302 %
Incorrectly Classified Instances        25               29.0698 %
Kappa statistic                         -0.0228
Mean absolute error                      0.3983
Root mean squared error                  0.4547
Relative absolute error                 98.2864 %
Root relative squared error            101.2257 %
Total Number of Instances               86

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.984    1.000    0.718      0.984    0.830      -0.067   0.480     0.710     A
                 0.000    0.016    0.000      0.000    0.000      -0.067   0.480     0.273     S
Weighted Avg.    0.709    0.725    0.517      0.709    0.598      -0.067   0.480     0.588

=== Confusion Matrix ===

  a  b   <-- classified as
 61  1 |  a = A
 24  0 |  b = S
```

For J48-
binarySplits - FALSE
Unpruned:- TRUE
Figure 5 -



```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -U -M 2
Relation:     postoperative-patient-data-simplified
Instances:    86
Attributes:   8
              L-CORE
              L-SURF
              L-O2
              L-BP
              SURF-STBL
              CORE-STBL
              BP-STBL
              decision
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

J48 unpruned tree
------------------

CORE-STBL = mod-stable: A (1.0)
CORE-STBL = stable
|   L-BP = high
|   |   L-SURF = high
|   |   |   BP-STBL = mod-stable: A (1.0)
|   |   |   BP-STBL = stable: S (2.0)
|   |   |   BP-STBL = unstable: A (2.0)
|   |   L-SURF = low: A (9.0/1.0)
|   |   L-SURF = mid: A (12.0/2.0)
|   L-BP = low: A (3.0)
|   L-BP = mid
|   |   L-SURF = high: A (9.0/1.0)
|   |   L-SURF = low
|   |   |   BP-STBL = mod-stable: S (1.0)
|   |   |   BP-STBL = stable: A (7.0/1.0)
|   |   |   BP-STBL = unstable: S (4.0/1.0)
|   |   L-SURF = mid
|   |   |   L-CORE = high: A (1.0)
|   |   |   L-CORE = low: S (5.0/2.0)
|   |   |   L-CORE = mid
|   |   |   |   BP-STBL = mod-stable: S (3.0/1.0)
|   |   |   |   BP-STBL = stable: A (15.0/4.0)
|   |   |   |   BP-STBL = unstable: A (6.0/1.0)
CORE-STBL = unstable: S (5.0/2.0)

Number of Leaves  :     17

Size of the tree :      25


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          60               69.7674 %
Incorrectly Classified Instances        26               30.2326 %
Kappa statistic                          0.1387
Mean absolute error                      0.3774
Root mean squared error                  0.4957
Relative absolute error                 93.1217 %
Root relative squared error            110.3555 %
Total Number of Instances               86

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.871    0.750    0.750      0.871    0.806      0.147    0.569     0.726     A
                 0.250    0.129    0.429      0.250    0.316      0.147    0.569     0.374     S
Weighted Avg.    0.698    0.577    0.660      0.698    0.669      0.147    0.569     0.628

=== Confusion Matrix ===

  a  b   <-- classified as
 54  8 |  a = A
```

i.) Hence from examining the trees we can say from Figure 2 i.e. when binarySplits - TRUE and Unpruned:- TRUE, we see that a decision can be directly obtained from the variable L-BP. Hence L-BP can directly give us a decision of A in a case when it is low. As a result this is a significant variable.
Similarly, from examining the Figure 5 i.e. binarySplits - FALSE and Unpruned:- TRUE, we see that a decision can be directly obtained from the variable CORE-STBL. Hence CORE-STBL can directly give us a decision of A in a case when it has a value of mod-stable. As a result this is a significant variable.

Hence from examination of the trees in all the four cases i.e. Figure 2,3,4,5 we can say that L-BP and CORE-STBL can help us to easily come up with decisions(A i.e. stay in hospital) when they have values of low and mod-stable respectively.

ii.)
Using the tree when the binarySplits is FALSE and Unpruned is TRUE, we can explain the accuracy and the confusion matrix.

The accuracy of this model is 69.7674%. This tells us that the model correctly classifies the instances 69.7674% of the time when seeing all the instances. It can also be defined as the sum of True Positives and True Negatives divided by the sum of all the instances i.e. True Positive, True Negative, False Positive, False Negative.
The classifier also has the F measure value as 0.669 which is again good as it is close to 1 rather than being close to 0. The F-measure is basically the harmonic mean of precision and recall. Taking into account the ROC area as well, we can see that it is 0.569. This represents the accuracy of a binary classifier.

The Confusion Matrix of this model looks like -

a	b
54	8
18	6

a → A (stay in hospital)
b → S (discharge from hospital)

Hence 54 represents True Positive(TP) i.e. when Actual decision was A and predicted was also A. 8 represents False Negative(FN) i.e. when Actual decision was A but predicted was S. 18 represents False Positive(FP) i.e. when Actual decision was S but predicted was A. 6 represents True Negative(TN) i.e. when Actual decision and predicted decision was S.

Hence the model correctly predicted (i.e. cases of TP and TN) 54 + 6 i.e. 60 instances out of total and the model was incorrect (i.e. cases of FP and FL) 18 + 8 i.e. 26 instances out of total.

Similarly, we can explain for the rest 3 cases where binarySplits is FALSE and Unpruned is FALSE, binarySplits is TRUE and Unpruned is FALSE, binarySplits is TRUE and Unpruned is TRUE

b)For Naive Bayes -

Figure 6 -

```
=== Run information ===

Scheme:       weka.classifiers.bayes.NaiveBayes
Relation:     postoperative-patient-data-simplified
Instances:    86
Attributes:   8
              L-CORE
              L-SURF
              L-O2
              L-BP
              SURF-STBL
              CORE-STBL
              BP-STBL
              decision
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

Naive Bayes Classifier

                Class
Attribute           A      S
               (0.72) (0.28)
============================
L-CORE
  high            10.0    4.0
  low             13.0    7.0
  mid             42.0   16.0
  [total]         65.0   27.0

L-SURF
  high            14.0    4.0
  low             17.0    8.0
  mid             34.0   15.0
  [total]         65.0   27.0

L-O2
  excellent       31.0   12.0
  good            33.0   14.0
  [total]         64.0   26.0

L-BP
  high            23.0    7.0
  low              4.0    1.0
  mid             38.0   19.0
  [total]         65.0   27.0

SURF-STBL
  stable          32.0   13.0
  unstable        32.0   13.0
  [total]         64.0   26.0

CORE-STBL
  mod-stable       2.0    1.0
  stable          60.0   22.0
  unstable         3.0    4.0
  [total]         65.0   27.0
```

```
BP-STBL
  mod-stable        18.0    5.0
  stable            31.0   16.0
  unstable          16.0    6.0
  [total]           65.0   27.0



Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          60               69.7674 %
Incorrectly Classified Instances        26               30.2326 %
Kappa statistic                          0.0244
Mean absolute error                      0.415
Root mean squared error                  0.4732
Relative absolute error                102.4041 %
Root relative squared error            105.3522 %
Total Number of Instances               86

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.935    0.917    0.725      0.935    0.817      0.033    0.437     0.654     A
                 0.083    0.065    0.333      0.083    0.133      0.033    0.437     0.304     S
Weighted Avg.    0.698    0.679    0.616      0.698    0.626      0.033    0.437     0.556

=== Confusion Matrix ===

  a  b   <-- classified as
 58  4 |  a = A
 22  2 |  b = S
```

i.) Probability distribution in the output represents the values that the variables have with respect to the decisions A or S. Talking about the variable BP-STBL we can see that when it is mod-stable there are 18 times it predicts A decision and 5 times it says S. When it is stable there are 31 times it predicts A and 16 times it predicts S. When it is unstable there are 16 times it predicts A and 6 times it predicts S. On a total BP-STBL discriminates towards decision A 65 times as compared to decision S which is seen 27 times. Smoothening is also carried out for the values of BP-STBL. It can be seen from the initial “Visualise All” graph in figure 1. Smoothening is carried out to prevent having a 0 probability when using Naive Bayes numerical formula even when most of the attributes may strongly discriminate towards one decision.

ii.) To calculate that person gets discharged i.e. S given the evidence that L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable, we can write it as

P(S| L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable)

This can be calculated using the Bayes Formula -
P(X|Y) = (P(Y|X)P(X)) / P(Y)
Where,
X = S
Y = L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable

Calculating numerator and denominator separately -

Numerator in parts -
P(L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable | S) =

P(L-CORE = mid| S).P(L-SURF = low| S).P(L-O2 = good| S).P(L-BP = high| S).P(SURF-STBL = stable| S).P(CORE-STBL = stable| S).P(BP-STBL = mod-stable| S)

=(16/27)(8/27)(14/26)(7/27)(13/26)(22/27)(5/27)
=17937920/10460353203
=0.0017

P(S) = 27/(65+27)
= 27/92
= 0.293

Denominator -
P(L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable) =

= P(L-CORE = mid).P(L-SURF = low).P(L-O2 = good).P(L-BP = high).P(SURF-STBL = stable).P(CORE-STBL = stable).P(BP-STBL = mod-stable)

= (58/92).(25/92).(47/92).(30/92).(45/92).(82/92).(23/92)
= 0.0031
Finally,
P(S|L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable) =

= (0.0017)(0.293) / 0.0031
= 0.00050/0.0031
= 0.161

Hence,

P(S| L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable) = 0.161

iii.)
Already calculated the probability that a person with these attributes will be discharged i.e. P(S| L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable) which is equal to 0.147

In a similar manner calculating probability that person will stay in hospital given the following attributes, we can calculate -

P(A| L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable)

This can be calculated using the Bayes Formula -
P(X|Y) = (P(Y|X)P(X)) / P(Y)
Where,
X = A
Y = L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable

Calculating numerator and denominator separately -

Numerator in parts -
P(L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable | A) =

P(L-CORE = mid| A).P(L-SURF = low| A).P(L-O2 = good| A).P(L-BP = high| A).P(SURF-STBL = stable| A).P(CORE-STBL = stable| A).P(BP-STBL = mod-stable| A)

=(42/65)(17/65)(33/64)(23/65)(32/64)(60/65)(18/65)
=18728962560/(73116160000(65))
=0.0040

P(A) = 65/(65+27)
= 65/92
= 0.7065

Denominator -
P(L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable) =

= P(L-CORE = mid).P(L-SURF = low).P(L-O2 = good).P(L-BP = high).P(SURF-STBL = stable).P(CORE-STBL = stable).P(BP-STBL = mod-stable)

= (58/92).(25/92).(47/92).(30/92).(45/92).(82/92).(23/92)
= 173516715000/((606355001344).(92))
= 0.0031

Finally,
P(A|L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable) =

= ((0.0040)(0.7065)) / ((117236851200/((404567235136).(86)))
= 0.0028/0.0031
= 0.9032

Hence,
P(A| L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable) = 0.9032

Hence P(S|L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable) = 0.161
and P(A|L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable) = 0.9032

Clearly 0.9032 > 0.161 hence the person should stay in the hospital. Hence Naive Bayes will predict the decision as A i.e. to stay in hospital

iv.)
The accuracy of the Naive Bayes classifier is 69.7674%
This means that it will correctly predict the decision 69.7674% of times i.e. cases of TP and TN. It can also be defined as the sum of True Positives and True Negatives divided by the sum of all the instances i.e. True Positive, True Negative, False Positive, False Negative.
The classifier also has the F measure value as 0.626 which is again good as it is close to 1 rather than being close to 0. The F-measure is basically the harmonic mean of

precision and recall. Taking into account the ROC area as well, we can see that it is 0.437. This represents the accuracy of a binary classifier.
The confusion matrix of the classifier looks like below -

a	b
58	4
22	2

a → A
b → S

Hence 58 represents True Positive(TP) i.e. when Actual decision was A and predicted was also A. 4 represents False Negative(FN) i.e. when Actual decision was A but predicted was S. 22 represents False Positive(FP) i.e. when Actual decision was S but predicted was A. 2 represents True Negative(TN) i.e. when Actual decision and predicted decision was S.
Hence the model correctly predicted (i.e. cases of TP and TN) 58 + 2 i.e. 60 instances out of total and the model was incorrect (i.e. cases of FP and FL) 22 + 4 i.e. 26 instances out of total.

c)
For KNN -
KNN-1
No distance weighting

Figure 7 -

```
=== Run information ===

Scheme:       weka.classifiers.lazy.IBk -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""
Relation:     postoperative-patient-data-simplified
Instances:    86
Attributes:   8
              L-CORE
              L-SURF
              L-O2
              L-BP
              SURF-STBL
              CORE-STBL
              BP-STBL
              decision
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 1 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          53               61.6279 %
Incorrectly Classified Instances        33               38.3721 %
Kappa statistic                         -0.1796
Mean absolute error                      0.4496
Root mean squared error                  0.5966
Relative absolute error                110.9423 %
Root relative squared error            132.8161 %
Total Number of Instances               86

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.855    1.000    0.688      0.855    0.763      -0.213   0.319     0.625     A
                 0.000    0.145    0.000      0.000    0.000      -0.213   0.318     0.213     S
Weighted Avg.    0.616    0.761    0.496      0.616    0.550      -0.213   0.319     0.510

=== Confusion Matrix ===

  a  b   <-- classified as
 53  9 |  a = A
 24  0 |  b = S
```

For KNN -
KNN-1
Weight by 1/distance
Figure 8 -

```
=== Run information ===

Scheme:       weka.classifiers.lazy.IBk -K 1 -W 0 -I -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""
Relation:     postoperative-patient-data-simplified
Instances:    86
Attributes:   8
              L-CORE
              L-SURF
              L-O2
              L-BP
              SURF-STBL
              CORE-STBL
              BP-STBL
              decision
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 1 inverse-distance-weighted nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          53               61.6279 %
Incorrectly Classified Instances        33               38.3721 %
Kappa statistic                         -0.1796
Mean absolute error                      0.4494
Root mean squared error                  0.6006
Relative absolute error                110.8969 %
Root relative squared error            133.7232 %
Total Number of Instances               86

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.855    1.000    0.688      0.855    0.763      -0.213   0.348     0.648     A
                 0.000    0.145    0.000      0.000    0.000      -0.213   0.349     0.221     S
Weighted Avg.    0.616    0.761    0.496      0.616    0.550      -0.213   0.348     0.529

=== Confusion Matrix ===

  a  b   <-- classified as
 53  9 |  a = A
 24  0 |  b = S
```

For KNN -
KNN-1
Weight by 1-distance
Figure 9 -

```
=== Run information ===

Scheme:       weka.classifiers.lazy.IBk -K 1 -W 0 -F -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""
Relation:     postoperative-patient-data-simplified
Instances:    86
Attributes:   8
              L-CORE
              L-SURF
              L-O2
              L-BP
              SURF-STBL
              CORE-STBL
              BP-STBL
              decision
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 1 similarity-weighted nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          53               61.6279 %
Incorrectly Classified Instances        33               38.3721 %
Kappa statistic                         -0.1796
Mean absolute error                      0.45
Root mean squared error                  0.596
Relative absolute error                111.0397 %
Root relative squared error            132.6971 %
Total Number of Instances               86

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.855    1.000    0.688      0.855    0.763      -0.213   0.308     0.615     A
                 0.000    0.145    0.000      0.000    0.000      -0.213   0.310     0.211     S
Weighted Avg.    0.616    0.761    0.496      0.616    0.550      -0.213   0.309     0.503

=== Confusion Matrix ===

  a  b   <-- classified as
 53  9 |  a = A
 24  0 |  b = S
```

For KNN -
KNN-2
No distance weighting
Figure 10 -

```
=== Run information ===

Scheme:       weka.classifiers.lazy.IBk -K 2 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""
Relation:     postoperative-patient-data-simplified
Instances:    86
Attributes:   8
              L-CORE
              L-SURF
              L-O2
              L-BP
              SURF-STBL
              CORE-STBL
              BP-STBL
              decision
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 2 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          61               70.9302 %
Incorrectly Classified Instances        25               29.0698 %
Kappa statistic                          -0.0228
Mean absolute error                       0.4242
Root mean squared error                   0.5184
Relative absolute error                 104.671  %
Root relative squared error             115.4187 %
Total Number of Instances                86

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
               0.984     1.000     0.718       0.984   0.830      -0.067   0.294     0.608     A
               0.000     0.016     0.000       0.000   0.000      -0.067   0.292     0.208     S
Weighted Avg.  0.709     0.725     0.517       0.709   0.598      -0.067   0.294     0.496

=== Confusion Matrix ===

  a  b   <-- classified as
 61  1 |  a = A
 24  0 |  b = S
```

For KNN -
KNN-2
Weight by 1/distance
Figure 11 -

```
=== Run information ===

Scheme:       weka.classifiers.lazy.IBk -K 2 -W 0 -I -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""
Relation:     postoperative-patient-data-simplified
Instances:    86
Attributes:   8
              L-CORE
              L-SURF
              L-O2
              L-BP
              SURF-STBL
              CORE-STBL
              BP-STBL
              decision
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 2 inverse-distance-weighted nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          54               62.7907 %
Incorrectly Classified Instances        32               37.2093 %
Kappa statistic                         -0.1622
Mean absolute error                      0.4517
Root mean squared error                  0.5894
Relative absolute error                111.4512 %
Root relative squared error            131.2215 %
Total Number of Instances               86

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.871    1.000    0.692      0.871    0.771      -0.199   0.316     0.627     A
                 0.000    0.129    0.000      0.000    0.000      -0.199   0.317     0.211     S
Weighted Avg.    0.628    0.757    0.499      0.628    0.556      -0.199   0.316     0.511

=== Confusion Matrix ===

  a  b   <-- classified as
 54  8 |  a = A
 24  0 |  b = S
```

For KNN -
KNN-2
Weight by 1 - distance
Figure 12 -

```
=== Run information ===

Scheme:       weka.classifiers.lazy.IBk -K 2 -W 0 -F -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""
Relation:     postoperative-patient-data-simplified
Instances:    86
Attributes:   8
              L-CORE
              L-SURF
              L-O2
              L-BP
              SURF-STBL
              CORE-STBL
              BP-STBL
              decision
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 2 similarity-weighted nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          60               69.7674 %
Incorrectly Classified Instances        26               30.2326 %
Kappa statistic                         -0.0449
Mean absolute error                      0.4257
Root mean squared error                  0.5203
Relative absolute error                105.0448 %
Root relative squared error            115.8386 %
Total Number of Instances               86

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.968    1.000    0.714      0.968    0.822      -0.096   0.291     0.604     A
                 0.000    0.032    0.000      0.000    0.000      -0.096   0.293     0.211     S
Weighted Avg.    0.698    0.730    0.515      0.698    0.593      -0.096   0.292     0.494

=== Confusion Matrix ===

  a  b   <-- classified as
 60  2 |  a = A
 24  0 |  b = S
```

i.)

The original instance that we have is -

L-CORE = mid L-SURF = low L-O2 = good L-BP = high SURF-STBL = stable CORE-STBL = stable BP-STBL = mod-stable

Taking three instances from the dataset that are similar to the above patient, we select =

Case 1- L-CORE = mid L-SURF =mid L-O2 =good L-BP =high SURF-STBL =unstable CORE-STBL =stable BP-STBL =mod-stable (5 matching)

Case 2 -L-CORE = mid L-SURF =high L-O2 =excellent L-BP =high SURF-STBL =stable CORE-STBL =stable BP-STBL =stable (4 matching)

Case 3-L-CORE = low L-SURF =low L-O2 =good L-BP =mid SURF-STBL =unstable CORE-STBL =stable BP-STBL =unstable (3 matching)

Using the formula of Jaccard Coefficient -

$J(X,Y) = |X \cap Y| / |X \cup Y|$

Calculating Jaccard Coefficient for Case1 and original case

J =
{L-CORE = mid L-O2 =good L-BP =high CORE-STBL =stable BP-STBL =mod-stable}

{L-CORE = mid L-SURF =low L-SURF =mid L-O2 =good SURF-STBL =stable SURF-STBL =unstable CORE-STBL =stable L-BP =high BP-STBL = mod-stable}

Hence, J = 5/9
J = 0.556

Calculating Jaccard Coefficient for Case2 and original case

J =
{L-CORE = mid L-O2 =good SURF-STBL =stable CORE-STBL =stable}

{L-CORE = mid L-SURF =low L-SURF =high L-O2 =excellent L-O2 =good SURF-STBL =stable CORE-STBL =stable L-BP =high BP-STBL =unstable BP-STBL =mod-stable}

Hence, J = 4/10
J = 0.4

Calculating Jaccard Coefficient for Case3 and original case

J =
{L-SURF =low L-O2 =good CORE-STBL =stable}

{L-CORE = mid L-CORE = low L-SURF =low L-O2 =good SURF-STBL =stable SURF-STBL =unstable CORE-STBL =stable L-BP =high L-BP =mid BP-STBL =stable BP-STBL =mod-stable}

Hence, J = 3/11
J = 0.27

As the Jaccard Coefficient of Case 1 is the maximum hence we can use the decision of that instance to predict the decision of our original case. From the data the decision of the Case 1 is - A i.e. the person stays in hospital. Hence from the KNN we can say that the model will predict the decision of A given the original condition of the patient i.e. the person should stay in the hospital.

ii.) Using different values of k and a constant distanceWeighting of No distance weighting we can use values of k as 1,2 and 3.

Using Figure 7(KNN-1 No distance weighting) and Figure 10(KNN-2 No distance weighting) and the following Figure 13(KNN-3 No distance weighting)

For KNN -
KNN-3
No distance weighting
Figure 13 -

```
=== Run information ===

Scheme:       weka.classifiers.lazy.IBk -K 3 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""
Relation:     postoperative-patient-data-simplified
Instances:    86
Attributes:   8
              L-CORE
              L-SURF
              L-O2
              L-BP
              SURF-STBL
              CORE-STBL
              BP-STBL
              decision
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 3 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          60               69.7674 %
Incorrectly Classified Instances        26               30.2326 %
Kappa statistic                         -0.0449
Mean absolute error                      0.4388
Root mean squared error                  0.5133
Relative absolute error                108.2735 %
Root relative squared error            114.2811 %
Total Number of Instances               86

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.968    1.000    0.714      0.968    0.822      -0.096   0.284     0.595     A
                 0.000    0.032    0.000      0.000    0.000      -0.096   0.284     0.211     S
Weighted Avg.    0.698    0.730    0.515      0.698    0.593      -0.096   0.284     0.488

=== Confusion Matrix ===

  a  b   <-- classified as
 60  2 |  a = A
 24  0 |  b = S
```

When KNN-1 we get accuracy as 61.6279%
This means that it will correctly predict the decision 61.6279% of times i.e. cases of TP and TN. It can also be defined as the sum of True Positives and True Negatives divided

by the sum of all the instances i.e. True Positive, True Negative, False Positive, False Negative.
The classifier also has the F measure value as 0.550 which is again good as it is close to 1 rather than being close to 0. The F-measure is basically the harmonic mean of precision and recall. Taking into account the ROC area as well, we can see that it is 0.319. This represents the accuracy of a binary classifier.
The confusion matrix of the classifier looks like below -

a	b
53	9
24	0

a → A
b → S

Hence 53 represents True Positive(TP) i.e. when Actual decision was A and predicted was also A. 9 represents False Negative(FN) i.e. when Actual decision was A but predicted was S. 24 represents False Positive(FP) i.e. when Actual decision was S but predicted was A. 0 represents True Negative(TN) i.e. when Actual decision and predicted decision was S.
Hence the model correctly predicted (i.e. cases of TP and TN) 53 + 0 i.e. 53 instances out of total and the model was incorrect (i.e. cases of FP and FL) 24 + 9 i.e. 33 instances out of total.

When KNN-2 we get accuracy as 70.9302%
This means that it will correctly predict the decision 70.9302% of times i.e. cases of TP and TN. It can also be defined as the sum of True Positives and True Negatives divided by the sum of all the instances i.e. True Positive, True Negative, False Positive, False Negative.
The classifier also has the F measure value as 0.598 which is again good as it is close to 1 rather than being close to 0. The F-measure is basically the harmonic mean of precision and recall. Taking into account the ROC area as well, we can see that it is 0.294. This represents the accuracy of a binary classifier.
The confusion matrix of the classifier looks like below -

a	b
61	1
24	0

a → A
b → S

Hence 61 represents True Positive(TP) i.e. when Actual decision was A and predicted was also A. 1 represents False Negative(FN) i.e. when Actual decision was A but predicted was S. 24 represents False Positive(FP) i.e. when Actual decision was S but predicted was A. 0 represents True Negative(TN) i.e. when Actual decision and predicted decision was S.
Hence the model correctly predicted (i.e. cases of TP and TN) 61 + 0 i.e. 61 instances out of total and the model was incorrect (i.e. cases of FP and FL) 24 + 1 i.e. 25 instances out of total.

When KNN-3 we get accuracy as 69.7674%
This means that it will correctly predict the decision 69.7674% of times i.e. cases of TP and TN. It can also be defined as the sum of True Positives and True Negatives divided by the sum of all the instances i.e. True Positive, True Negative, False Positive, False Negative.
The classifier also has the F measure value as 0.593 which is again good as it is close to 1 rather than being close to 0. The F-measure is basically the harmonic mean of precision and recall. Taking into account the ROC area as well, we can see that it is 0.284. This represents the accuracy of a binary classifier.
The confusion matrix of the classifier looks like below -

a	b
60	2
24	0

a → A
b → S

Hence 60 represents True Positive(TP) i.e. when Actual decision was A and predicted was also A. 2 represents False Negative(FN) i.e. when Actual decision was A but predicted was S. 24 represents False Positive(FP) i.e. when Actual decision was S but predicted was A. 0 represents True Negative(TN) i.e. when Actual decision and predicted decision was S.
Hence the model correctly predicted (i.e. cases of TP and TN) 60 + 0 i.e. 60 instances out of total and the model was incorrect (i.e. cases of FP and FL) 24 + 2 i.e. 26 instances out of total.

3)

Parameters	J48	Naive Bayes	k-NN

Correctly Classified Instances	61 (70.9302 %)	60 (69.7674 %)	53 (61.6279 %)
Incorrectly Classified Instances	25 (29.0698 %)	26(30.2326 %)	33 (38.3721 %)
Kappa statistic	-0.0228	0.0244	-0.1796
Mean absolute error	0.4075	0.415	0.4496
Root mean squared error	0.4554	0.4732	0.5966
Relative absolute error	100.5513 %	102.4041 %	110.9423 %
Root relative squared error	101.3916 %	105.3522 %	132.8161 %
TP Rate	0.709	0.698	0.616
FP Rate	0.725	0.679	0.761
Precision	0.517	0.616	0.496
Recall	0.709	0.698	0.616
F-Measure	0.598	0.626	0.550
MCC	-0.067	0.033	-0.213
ROC Area	0.439	0.437	0.319
PRC Area	0.573	0.556	0.510
True Positive	61	58	53
False Negative	1	4	9
False Positive	24	22	24
True Negative	0	2	0

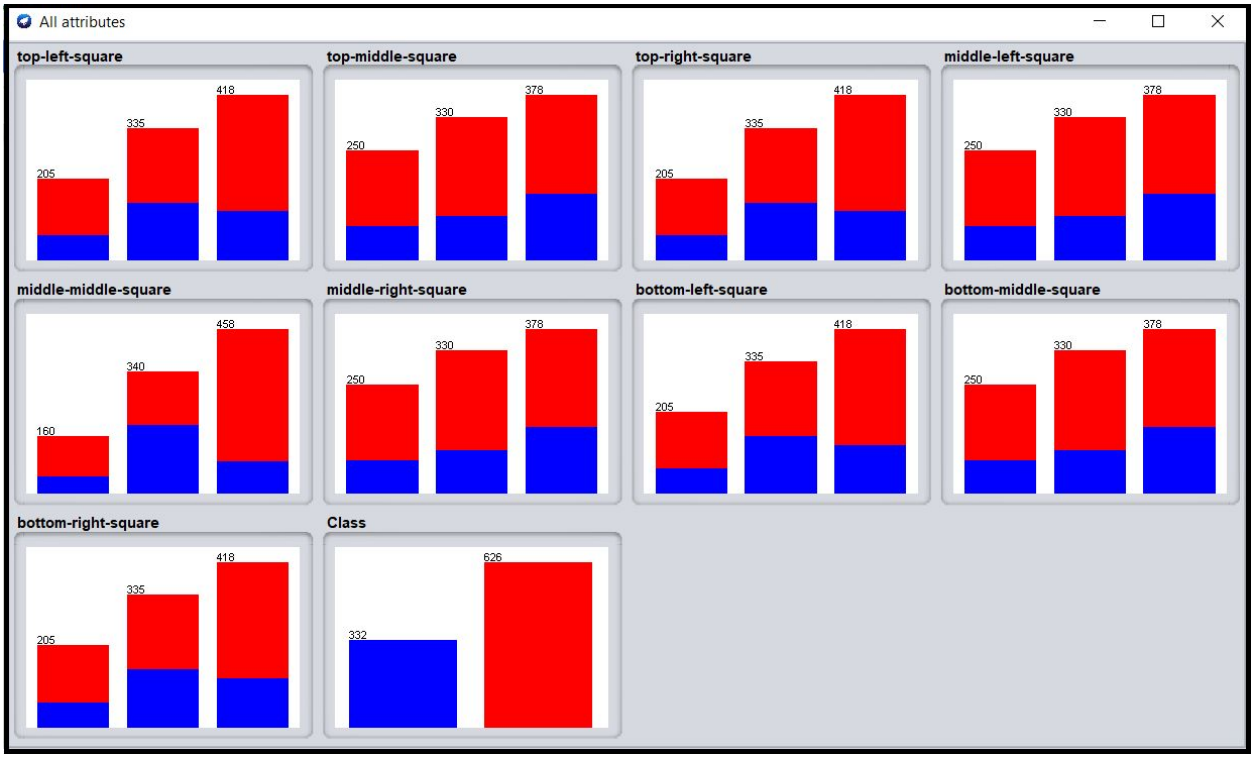
By comparing the different parameters from the above table, we can conclude that K-NN is the worst one while J48 and Naive Bayes are similar, however it seems that Naive Bayes is the best classifier. For K-NN the accuracy i.e. the correctly classified instances percentage is the least at 61.6279% while that of J48 and Naive Bayes is high at 70.9302 % and 69.7674 % respectively. Hence J48 and Naive Bayes are better

classifiers. Talking about the F-Measure of the 3 classifiers we can see that again Knn's value i.e. 0.550 is the least and hence closest to 0 which is not good. While the Fmeasure for J48 and Naive Bayes are 0.598 and 0.626 respectively which is closer to 1. Hence again J48 and Naive Bayes are better where Naive Bayes is a little extra good than J48. ROC area for KNN is low at 0.319 while for J40 and Naive Bayes is very similar.

Q5)

1. a) For winning or losing we will only check the x bars in the Visualise all tab. As the x player ie. the first player our chances of winning(red) or losing(blue) can be determined by looking at the graphs in the below figure.

Figure 14 -



There is no definite probability of winning or losing when the player plays first (x).
Looking at the x bar of middle-middle-square we see that the case of positive(win) is very high as compared to case of negative(loss). Hence if the first player puts x in the middle square, he has a high chance of winning the game. Hence we can say that middle-middle-square is a significant variable.

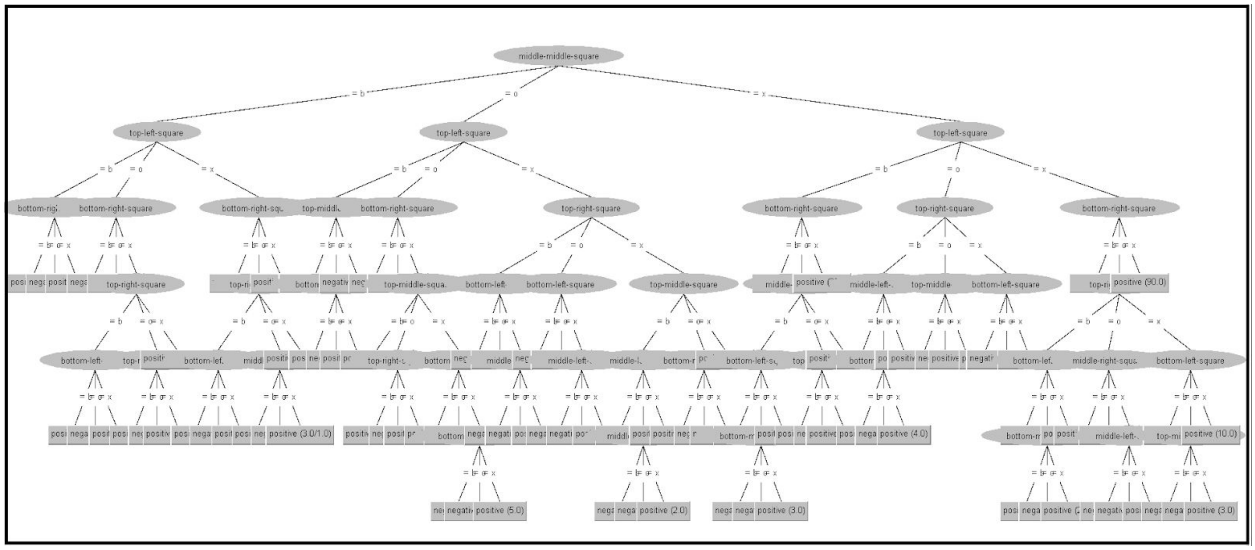
Also, if the first player puts an x in the top-right-square he has a good chance of winning as the case of positive(win) is more as compared to case of negative(loss). Hence we can say that this is also a significant variable.

b) The advantage of being the first player is that the player always has an option to choose from all the squares where he can place the first x. Based on the position of the first x, from the above Figure 14 we can say that middle-middle and top-right show higher chances of winning. As a result the player has a higher chance of winning when he is in control of where to put the first x. Also we can notice that for all the cases, the person who puts the first x has a high chance of winning(positive i.e. red) as compared to the chances of losing(negative i.e. blue). There appears to be no disadvantage to the first player as comparing the red and blue portion of all the “o” bars we see either winning or losing is equally probable or losing is more probable for most of the cases.

2)

a) i.) For J48-

Figure 15-



From the above decision tree we can say that middle-middle-square is a significant variable. It appears on the top of the tree as the root node and hence it is the most significant as compared to the others. For this variable the chances of winning of the first player i.e. a positive outcome is high as compared to the chances of losing of the first player i.e. negative outcome.

ii.) Looking at the following game -

X	X	
O	O	X

	O	

We can use the J48 summary of WEKA to trace the decision tree.

Figure 16 -

```
middle-middle-square = b
|   top-left-square = b
|   |   bottom-right-square = b: positive (0.0)
|   |   bottom-right-square = o: negative (8.0)
|   |   bottom-right-square = x: positive (20.0)
|   top-left-square = o
|   |   bottom-right-square = b: negative (8.0)
|   |   bottom-right-square = o: negative (4.0)
|   |   bottom-right-square = x
|   |   |   top-right-square = b
|   |   |   |   bottom-left-square = b: positive (0.0)
|   |   |   |   bottom-left-square = o: negative (3.0)
|   |   |   |   bottom-left-square = x: positive (6.0)
|   |   |   top-right-square = o
|   |   |   |   top-middle-square = b: positive (3.0)
|   |   |   |   top-middle-square = o: negative (7.0)
|   |   |   |   top-middle-square = x: positive (3.0/1.0)
|   |   |   top-right-square = x: positive (20.0/3.0)
|   top-left-square = x
|   |   bottom-right-square = b: positive (20.0)
|   |   bottom-right-square = o
|   |   |   top-right-square = b
|   |   |   |   bottom-left-square = b: positive (0.0)
|   |   |   |   bottom-left-square = o: negative (3.0)
|   |   |   |   bottom-left-square = x: positive (6.0)
|   |   |   top-right-square = o
|   |   |   |   middle-right-square = b: positive (3.0)
|   |   |   |   middle-right-square = o: negative (7.0)
|   |   |   |   middle-right-square = x: positive (3.0/1.0)
|   |   |   top-right-square = x: positive (20.0/3.0)
|   |   bottom-right-square = x: positive (16.0)
```

```
middle-middle-square = o
|   top-left-square = b
|   |   top-middle-square = b: positive (18.0/7.0)
|   |   top-middle-square = o
|   |   |   bottom-middle-square = b: positive (3.0)
|   |   |   bottom-middle-square = o: negative (12.0)
|   |   |   bottom-middle-square = x: positive (10.0/1.0)
|   |   top-middle-square = x: negative (26.0/5.0)
|   top-left-square = o
|   |   bottom-right-square = b: negative (5.0)
|   |   bottom-right-square = o: negative (50.0)
|   |   bottom-right-square = x
|   |   |   top-middle-square = b: positive (15.0/1.0)
|   |   |   top-middle-square = o
|   |   |   |   top-right-square = b: positive (3.0/1.0)
|   |   |   |   top-right-square = o: negative (2.0)
|   |   |   |   top-right-square = x: positive (8.0/1.0)
|   |   |   top-middle-square = x
|   |   |   |   bottom-left-square = b: positive (3.0/1.0)
|   |   |   |   bottom-left-square = o: negative (8.0/2.0)
|   |   |   |   bottom-left-square = x
|   |   |   |   |   bottom-middle-square = b: negative (1.0)
|   |   |   |   |   bottom-middle-square = o: negative (3.0/1.0)
|   |   |   |   |   bottom-middle-square = x: positive (5.0)
|   top-left-square = x
|   |   top-right-square = b
|   |   |   bottom-left-square = b: negative (6.0)
|   |   |   bottom-left-square = o: negative (3.0)
|   |   |   bottom-left-square = x
|   |   |   |   middle-left-square = b: negative (3.0/1.0)
|   |   |   |   middle-left-square = o: negative (7.0/1.0)
|   |   |   |   middle-left-square = x: positive (14.0)
|   |   top-right-square = o
|   |   |   bottom-left-square = b: negative (3.0)
|   |   |   bottom-left-square = o: negative (30.0)
|   |   |   bottom-left-square = x
|   |   |   |   middle-left-square = b: negative (4.0/2.0)
|   |   |   |   middle-left-square = o: negative (6.0/2.0)
|   |   |   |   middle-left-square = x: positive (17.0)
|   |   top-right-square = x
|   |   |   top-middle-square = b
|   |   |   |   middle-left-square = b: positive (1.0)
|   |   |   |   middle-left-square = o
|   |   |   |   |   middle-right-square = b: negative (0.0)
|   |   |   |   |   middle-right-square = o: negative (6.0)
|   |   |   |   |   middle-right-square = x: positive (2.0)
|   |   |   |   middle-left-square = x: positive (4.0/1.0)
|   |   |   top-middle-square = o
|   |   |   |   bottom-middle-square = b: positive (5.0/1.0)
|   |   |   |   bottom-middle-square = o: negative (12.0)
|   |   |   |   bottom-middle-square = x: negative (7.0/2.0)
|   |   |   top-middle-square = x: positive (38.0)
```

```
middle-middle-square = x
|   top-left-square = b
|   |   bottom-right-square = b: positive (30.0)
|   |   bottom-right-square = o
|   |   |   middle-right-square = b
|   |   |   |   bottom-left-square = b: positive (2.0)
|   |   |   |   bottom-left-square = o
|   |   |   |   |   bottom-middle-square = b: negative (0.0)
|   |   |   |   |   bottom-middle-square = o: negative (3.0)
|   |   |   |   |   bottom-middle-square = x: positive (3.0)
|   |   |   |   bottom-left-square = x: positive (7.0)
|   |   |   middle-right-square = o
|   |   |   |   top-right-square = b: positive (3.0)
|   |   |   |   top-right-square = o: negative (7.0)
|   |   |   |   top-right-square = x: positive (10.0/1.0)
|   |   |   middle-right-square = x: positive (23.0/3.0)
|   |   bottom-right-square = x: positive (20.0)
|   top-left-square = o
|   |   top-right-square = b
|   |   |   middle-left-square = b: positive (7.0)
|   |   |   middle-left-square = o
|   |   |   |   bottom-left-square = b: positive (3.0)
|   |   |   |   bottom-left-square = o: negative (7.0)
|   |   |   |   bottom-left-square = x: positive (4.0)
|   |   |   middle-left-square = x: positive (19.0)
|   |   top-right-square = o
|   |   |   top-middle-square = b: positive (11.0/2.0)
|   |   |   top-middle-square = o: negative (21.0)
|   |   |   top-middle-square = x: positive (26.0/4.0)
|   |   top-right-square = x
|   |   |   bottom-left-square = b: positive (9.0)
|   |   |   bottom-left-square = o: negative (26.0/10.0)
|   |   |   bottom-left-square = x: positive (45.0)
|   top-left-square = x
|   |   bottom-right-square = b: positive (20.0)
|   |   bottom-right-square = o
|   |   |   top-right-square = b
|   |   |   |   bottom-left-square = b: positive (2.0)
|   |   |   |   bottom-left-square = o
|   |   |   |   |   bottom-middle-square = b: positive (1.0)
|   |   |   |   |   bottom-middle-square = o: negative (5.0)
|   |   |   |   |   bottom-middle-square = x: positive (2.0)
|   |   |   |   bottom-left-square = x: positive (3.0)
|   |   |   top-right-square = o
|   |   |   |   middle-right-square = b: positive (5.0/1.0)
|   |   |   |   middle-right-square = o: negative (12.0)
|   |   |   |   middle-right-square = x
|   |   |   |   |   middle-left-square = b: negative (1.0)
|   |   |   |   |   middle-left-square = o: negative (3.0/1.0)
|   |   |   |   |   middle-left-square = x: positive (5.0)
|   |   |   top-right-square = x
|   |   |   |   bottom-left-square = b: positive (3.0)
|   |   |   |   bottom-left-square = o
|   |   |   |   |   top-middle-square = b: negative (3.0)
|   |   |   |   |   top-middle-square = o: negative (4.0)
|   |   |   |   |   top-middle-square = x: positive (3.0)
|   |   |   |   bottom-left-square = x: positive (10.0)
|   |   bottom-right-square = x: positive (90.0)
```

Looking at the above algorithm step by step,
middle-middle-square = b is FALSE hence we don't go inside this portion
middle-middle-square = o is TRUE hence we trace inside now,
top-left-square = b is FALSE hence we don't go inside this portion
top-left-square = o is FALSE hence we don't go inside this portion
top-left-square = x is TRUE hence we trace inside now,
top-right-square = b is TRUE hence we trace inside now,
bottom-left-square = b is TRUE hence, : negative (6.0)

Hence there are 6 cases where we see a similar track as explained above and the first player loses i.e. outcome is negative. The final prediction for this game is hence that the first player will lose the game if he follows a pattern as given in this problem.

iii.)
The first split in the decision tree is middle-middle-square. From this the tree creation starts where it can have 3 values of b,o or x

To calculate IG of the middle middle square :

Using Formula -
$IG(S,A) = H(S) - \Sigma\ (S_v/S)(H(S_v))$
For all v belonging to Values(A)
where −
v is a value for A (we sum over all values)
S_v is a subset of S for which attribute A has value v

and
$H(X) = - \Sigma\ P(x_i).\log_2(P(x_i))$
for i being 1 to n

For our situation, A = middle-middle-square
And v = b,o,x

Hence,
$IG(S,\text{middle-middle-square}) = H(S) - ((S_o/S)(H(S_o))) - ((S_b/S)(H(S_b))) - ((S_x/S)(H(S_x)))$
--------Equation 1

H(S) =
$= - ((332/958) \ . \log_2(332/958)) - ((626/958) \ \log_2(626/958))$
$= -(0.347)(-1.527) - (0.653)(-0.615)$

= 0.5299 + 0.4016

= **0.9315**

Now for the rest of the part of equation 1.

=- [(160/958)H(b) + (340/958)H(o) + (458/958)H(x)]

=**- [0.167H(b) + 0.355H(o) + 0.478H(x)]**

Where,

H(b) = - [(112/160)$\log_2$(112/160)+ (48/160)$\log_2$(48/160)]

= - 0.7 $\log_2$(0.7) - 0.3 $\log_2$(0.3)

= 0.7(0.515) + 0.3(1.737)

= 0.3605 + 0.5211

= **0.8816**

H(o) = - [(148/340)$\log_2$(148/340) + (192/340)$\log_2$(192/340)]

= - 0.435 $\log_2$(0.435) - 0.565 $\log_2$(0.565)

= 0.435(1.201) + 0.565(0.824)

= 0.5224 + 0.4656

= **0.988**

H(x) = - [(366/458)$\log_2$(366/458) + (92/458)$\log_2$(92/458)]

= - 0.7991 $\log_2$(0.7991) - 0.2009 $\log_2$(0.2009)

= 0.7991(0.3236) + 0.2009(2.3155)

= 0.2586 + 0.4652

= **0.7238**

Combining the results we get

IG(S,middle-middle-square) =

= 0.9315 - 0.167(0.8816) - 0.355(0.988) - 0.478(0.7238)

= 0.9315 - 0.1472 - 0.3507 - 0.3460

= 0.0876

Hence, Information Gain = 0.0876

iv.)

For J48 in this case we get accuracy as 84.5511%
This means that it will correctly predict the decision 84.5511% of times i.e. cases of TP and TN. It can also be defined as the sum of True Positives and True Negatives divided by the sum of all the instances i.e. True Positive, True Negative, False Positive, False Negative.
The classifier also has the F measure value as 0.845 which is again good as it is close to 1 rather than being close to 0. The F-measure is basically the harmonic mean of precision and recall. Taking into account the ROC area as well, we can see that it is 0.896. This represents the accuracy of a binary classifier.
The confusion matrix of the classifier looks like below -

a	b
255	77
71	555

a → negative
b → positive

Hence 255 represents True Positive(TP) i.e. when Actual decision was negative and predicted was also negative. 77 represents False Negative(FN) i.e. when Actual decision was negative but predicted was positive. 71 represents False Positive(FP) i.e. when Actual decision was positive but predicted was negative. 555 represents True Negative(TN) i.e. when Actual decision and predicted decision was positive.
Hence the model correctly predicted (i.e. cases of TP and TN) 255 + 555 i.e. 810 instances out of total and the model was incorrect (i.e. cases of FP and FL) 77 + 71 i.e. 148 instances out of total.

b.) Naive Bayes
i.) We need to find probability of win i.e. positive given that top-left-square = x top-middle-square = x top-right-square = b middle-left-square = o middle-middle-square = o middle-right-square=x bottom-left-square=b bottom-middle-square=o bottom-right-square=b

P(positive | top-left-square = x top-middle-square = x top-right-square = b middle-left-square = o middle-middle-square = o middle-right-square=x bottom-left-square=b bottom-middle-square=o bottom-right-square=b)

Using Bayes Formula =
$P(X|Y) = (P(Y|X)P(X)) / P(Y)$
Where,
X = positive

Y = top-left-square = x top-middle-square = x top-right-square = b middle-left-square = o middle-middle-square = o middle-right-square=x bottom-left-square=b bottom-middle-square=o bottom-right-square=b

Calculating numerator and denominator separately -

Numerator in parts -
P(top-left-square = x top-middle-square = x top-right-square = b middle-left-square = o middle-middle-square = o middle-right-square=x bottom-left-square=b bottom-middle-square=o bottom-right-square=b | positive)

= P(top-left-square = x| positive). P(top-middle-square = x| positive). P(top-right-square = b | positive) . P(middle-left-square = o| positive).P(middle-middle-square = o| positive).P(middle-right-square=x| positive).P(bottom-left-square=b | positive). P(bottom-middle-square=o| positive). P(bottom-right-square=b | positive)

= (296/629) (226/629) (143/629) (230/629) (149/629) (226/629) (143/629) (230/629) (143/629)
= (0.4706)(0.3593)(0.2273)(0.3657)(0.2369)(0.3593)(0.2273)(0.3657)(0.2273)
=0.00002258

P(positive) = 629/(629 + 335)
= 629/964
= **0.6525**

Now, denominator -
P(top-left-square = x top-middle-square = x top-right-square = b middle-left-square = o middle-middle-square = o middle-right-square=x bottom-left-square=b bottom-middle-square=o bottom-right-square=b)

= P(top-left-square = x). P(top-middle-square = x).P(top-right-square = b) P(middle-left-square = o) . P(middle-middle-square = o) .P(middle-right-square=x) . P(bottom-left-square = b). P(bottom-middle-square=o).P(bottom-right-square = b)

=
(420/964)(380/964)(207/964)(332/964)(342/964)(380/964)(207/964)(332/964)(207/964)
= (0.4357)(0.3942)(0.2147)(0.3444)(0.3548)(0.3942)(0.2147)(0.3444)(0.2147)
= **0.00002820**

Finally,

P(positive | top-left-square = x top-middle-square = x middle-left-square = o middle-middle-square = o middle-right-square=x bottom-middle-square=o) =
= ((0.00002258)(0.6525)) / (0.00002820)
= 0.00001473/0.00002820
= 0.5223

ii.) From above we know that
P(positive | top-left-square = x top-middle-square = x middle-left-square = o middle-middle-square = o middle-right-square=x bottom-middle-square=o) = 0.5223

Now calculating for
P(negative | top-left-square = x top-middle-square = x middle-left-square = o middle-middle-square = o middle-right-square=x bottom-middle-square=o)

We need to find probability of lose i.e. negative given that top-left-square = x top-middle-square = x top-right-square = b middle-left-square = o middle-middle-square = o middle-right-square=x bottom-left-square=b bottom-middle-square=o bottom-right-square=b

P(negative | top-left-square = x top-middle-square = x top-right-square = b middle-left-square = o middle-middle-square = o middle-right-square=x bottom-left-square=b bottom-middle-square=o bottom-right-square=b)

Using Bayes Formula =
P(X|Y) = (P(Y|X)P(X)) / P(Y)
Where,
X = negative
Y = top-left-square = x top-middle-square = x top-right-square = b middle-left-square = o middle-middle-square = o middle-right-square=x bottom-left-square=b bottom-middle-square=o bottom-right-square=b

Calculating numerator and denominator separately -

Numerator in parts -
P(top-left-square = x top-middle-square = x top-right-square = b middle-left-square = o middle-middle-square = o middle-right-square=x bottom-left-square=b bottom-middle-square=o bottom-right-square=b | negative)

= P(top-left-square = x| negative). P(top-middle-square = x| negative). P(top-right-square = b | negative) . P(middle-left-square = o| negative).P(middle-middle-square = o|

negative).P(middle-right-square=x| negative).P(bottom-left-square=b | negative).
P(bottom-middle-square=o| negative). P(bottom-right-square=b | negative)

= (124/335) (154/335) (64/335) (102/335) (193/335) (154/335) (64/335) (102/335) (64/335)
= (0.3701)(0.4597)(0.1910)(0.3045)(0.5761)(0.4597)(0.1910)(0.3045)(0.1910)
=0.00002911

P(negative) = 335/(629 + 335)
= 335/964
= **0.3475**

Now, denominator -
P(top-left-square = x top-middle-square = x top-right-square = b middle-left-square = o middle-middle-square = o middle-right-square=x bottom-left-square=b bottom-middle-square=o bottom-right-square=b)

= P(top-left-square = x). P(top-middle-square = x).P(top-right-square = b) P(middle-left-square = o) . P(middle-middle-square = o) .P(middle-right-square=x) . P(bottom-left-square = b). P(bottom-middle-square=o).P(bottom-right-square = b)

= (420/964)(380/964)(207/964)(332/964)(342/964)(380/964)(207/964)(332/964)(207/964)
= (0.4357)(0.3942)(0.2147)(0.3444)(0.3548)(0.3942)(0.2147)(0.3444)(0.2147)
= **0.00002820**

Finally,

P(positive | top-left-square = x top-middle-square = x middle-left-square = o middle-middle-square = o middle-right-square=x bottom-middle-square=o) =
= ((0.00002911)(0.3475)) / (0.00002820)
= 0.00001012/0.00002820
= 0.3589

Hence,
P(positive | top-left-square = x top-middle-square = x middle-left-square = o middle-middle-square = o middle-right-square=x bottom-middle-square=o) = 0.5223

P(negative | top-left-square = x top-middle-square = x middle-left-square = o middle-middle-square = o middle-right-square=x bottom-middle-square=o) = 0.3589

As 0.5223 > 0.3589, hence Naive Bayes will predict the decision as positive i.e. players who goes first will win the game.

iii.)
For Naive Bayes we get accuracy as 69.6242%
This means that it will correctly predict the decision 69.6242% of times i.e. cases of TP and TN. It can also be defined as the sum of True Positives and True Negatives divided by the sum of all the instances i.e. True Positive, True Negative, False Positive, False Negative.
The classifier also has the F measure value as 0.683 which is again good as it is close to 1 rather than being close to 0. The F-measure is basically the harmonic mean of precision and recall. Taking into account the ROC area as well, we can see that it is 0.744. This represents the accuracy of a binary classifier.
The confusion matrix of the classifier looks like below -

a	b
142	190
101	525

a → negative
b → positive

Hence 142 represents True Positive(TP) i.e. when Actual decision was negative and predicted was also negative. 190 represents False Negative(FN) i.e. when Actual decision was negative but predicted was positive. 101 represents False Positive(FP) i.e. when Actual decision was positive but predicted was negative. 525 represents True Negative(TN) i.e. when Actual decision and predicted decision was positive.
Hence the model correctly predicted (i.e. cases of TP and TN) 142 + 525 i.e. 667 instances out of total and the model was incorrect (i.e. cases of FP and FL) 101 + 190 i.e. 291 instances out of total

c.) KNN
i.) The original instance that we have is -
top-left-square = x top-middle-square = x top-right-square = b middle-left-square = o middle-middle-square = o middle-right-square=x bottom-left-square=b bottom-middle-square=o bottom-right-square=b

Taking three instances from the dataset that are similar to the above, we select =

Case 1- top-left-square = x top-middle-square = x top-right-square = b middle-left-square = x middle-middle-square = o middle-right-square=o bottom-left-square=x bottom-middle-square=o bottom-right-square=b (6 matching)

Case 2 top-left-square = x top-middle-square = x top-right-square = x middle-left-square = o middle-middle-square = o middle-right-square=x bottom-left-square=b bottom-middle-square=o bottom-right-square=b (8 matching)

Case 3 top-left-square = x top-middle-square = x top-right-square = x middle-left-square = b middle-middle-square = b middle-right-square=o bottom-left-square=b bottom-middle-square=o bottom-right-square=b (5 matching)

Using the formula of Jaccard Coefficient -
J(X,Y) = |X∩Y| / |X ∪ Y|

Calculating Jaccard Coefficient for Case1 and original case

J =
{ top-left-square = x top-middle-square = x top-right-square = b middle-middle-square = o bottom-middle-square=o bottom-right-square=b}

{ top-left-square = x top-middle-square = x top-right-square = b middle-left-square = x middle-left-square = o middle-middle-square = o middle-right-square = o middle-right-square = x bottom-left-square=b bottom-left-square=x bottom-middle-square=o bottom-right-square=b}

Hence, J = 6/12
J = 0.5

Calculating Jaccard Coefficient for Case2 and original case

J =
{top-left-square = x top-middle-square = x middle-left-square = o middle-middle-square = o middle-right-square=x bottom-left-square=b bottom-middle-square=o bottom-right-square=b }

{ top-left-square = x top-middle-square = x top-right-square = x top-right-square = b middle-left-square = o middle-middle-square = o middle-right-square=x bottom-left-square=b bottom-middle-square=o bottom-right-square=b }

Hence, J = 8/10
J = 0.8

Calculating Jaccard Coefficient for Case3 and original case

J =
{ top-left-square = x top-middle-square = x bottom-left-square=b
bottom-middle-square=o bottom-right-square=b }

{top-left-square = x top-middle-square = x bottom-left-square=b
bottom-middle-square=o bottom-right-square=b top-right-square = x top-right-square = b
middle-left-square = o middle-left-square = b middle-middle-square = o
middle-middle-square = b middle-right-square=x middle-right-square=o}

Hence, J = 5/13
J = 0.38

As the Jaccard Coefficient of Case 2 is the maximum hence we can use the decision of that instance to predict the decision of our original case. From the data the decision of the Case 2 is - positive i.e. the person who starts the game will win the game. Hence from the KNN we can say that the model will predict the decision of positive given the original condition of the patient i.e. the person who starts the game will win the game.

ii.) For KNN we get accuracy as 98.9562%
This means that it will correctly predict the decision 98.9562% of times i.e. cases of TP and TN. It can also be defined as the sum of True Positives and True Negatives divided by the sum of all the instances i.e. True Positive, True Negative, False Positive, False Negative.
The classifier also has the F measure value as 0.990 which is again good as it is close to 1 rather than being close to 0. The F-measure is basically the harmonic mean of precision and recall. Taking into account the ROC area as well, we can see that it is 1.00. This represents the accuracy of a binary classifier.
The confusion matrix of the classifier looks like below -

a	b
323	9
1	625

a → negative
b → positive

Hence 323 represents True Positive(TP) i.e. when Actual decision was negative and predicted was also negative. 9 represents False Negative(FN) i.e. when Actual decision was negative but predicted was positive. 1 represents False Positive(FP) i.e. when

Actual decision was positive but predicted was negative. 625 represents True Negative(TN) i.e. when Actual decision and predicted decision was positive.
Hence the model correctly predicted (i.e. cases of TP and TN) 323 + 625 i.e. 948 instances out of total and the model was incorrect (i.e. cases of FP and FL) 1 + 9 i.e. 10 instances out of total

3)

Parameters	J48	Naive Bayes	k-NN
Correctly Classified Instances	810 (84.5511 %)	667 (69.6242 %)	948(98.9562 %)
Incorrectly Classified Instances	148 (15.4489 %)	291 (30.3758 %)	10 (1.0438 %)
Kappa statistic	0.6574	0.2843	0.9768
Mean absolute error	0.1695	0.3702	0.1909
Root mean squared error	0.3439	0.4319	0.2315
Relative absolute error	37.4049 %	81.7182 %	42.1333 %
Root relative squared error	72.262 %	90.7504 %	48.6378 %
TP Rate	0.846	0.696	0.990
FP Rate	0.191	0.430	0.018
Precision	0.845	0.682	0.990
Recall	0.846	0.696	0.990
F-Measure	0.845	0.683	0.990
MCC	0.658	0.291	0.977
ROC Area	0.896	0.744	1.000
PRC Area	0.892	0.780	1.000

True Positive	255	142	323
False Negative	77	190	9
False Positive	71	101	1
True Negative	555	525	625

By comparing the different parameters from the above table, we can conclude that K-NN is the best classifier. For K-NN the accuracy i.e. the correctly classified instances percentage is the maximum at 98.9562% while that of J48 and Naive Bayes is low at 84.5511 % and 69.6242 % respectively. Hence J48 and Naive Bayes are not good classifiers. Talking about the F-Measure of the 3 classifiers we can see that again Knn's value i.e. 0.990 is the highest and closest to 1 which is good. While the Fmeasure for J48 and Naive Bayes are 0.845 and 0.683 respectively which is less as compared to that of KNN. Hence again KNN is better . ROC area for KNN is the max i.e. 1.000 while for J40 and Naive Bayes it is low.

Q6)

1. The resultant regression function is =
   Unemployment-Rate =

   -0.0014 * All-Ords-Index +
   -0.2452 * Housing-Loan-Interest-Rate +
   13.7286

   Which can be seen as
   $y = \beta_o + \beta_1 x_1 + \beta_2 x_2$

   So,
   y = Unemployment-Rate = target
   $\mathbf{B_o}$ = 13.7286
   $\mathbf{B_1}$ = -0.0014
   $\mathbf{B_2}$ = -0.2452
   $\mathbf{X_1}$ = All-Ords-Index
   $\mathbf{X_2}$ = Housing-Loan-Interest-Rate

2. For the year 1986 we have from the given data the value of Unemployment-Rate **= 8.4**
   This is hence the actual value of the target variable

Now to calculate the predicted value of target variable Unemployment-Rate in 1986 we will use the equation :

Unemployment-Rate =
-0.0014 * All-Ords-Index +
-0.2452 * Housing-Loan-Interest-Rate +
13.7286

From the actual data we can find that in 1986 :
All-Ords-Index = 1779.1 and
Housing-Loan-Interest-Rate = 15.5

Using the regression function we get
Unemployment-Rate =
-0.0014 * All-Ords-Index +
-0.2452 * Housing-Loan-Interest-Rate +
13.7286
Unemployment-Rate =
-0.0014 * 1779.1 +
-0.2452 * 15.5 +
13.7286

Unemployment-Rate =
-2.49074 +
-3.8006 +
13.7286

Unemployment-Rate =
7.43726

Hence,
Absolute Error = $|y_i - y|$
where ,
y_i is the predicted value, and
y is the actual value

Hence,

Absolute Error = |7.43726 - 8.4 |

Absolute Error = |-0.96274|
Absolute Error = 0.96274

Answer , Absolute Error = 0.96274

3)
Mean Absolute Error (MAE) = $(\Sigma |y_i - y|) /n$
where ,
y_i is the predicted value, and
y is the actual value
n is the total rows in data

Using excel -
MAE = 41.36/30
MAE = 1.38

Root Mean Square Error (RMSE) =$\text{sqrt}((\Sigma (y_i - y)^2) /n)$
where ,
y_i is the predicted value, and
y is the actual value
n is the total rows in data

Using excel
RMSE
= sqrt(277.65/30)
= sqrt(9.255)
= 3.042

Comparing MAE and RMSE-
MAE takes into account the errors and gives all of them an equal weightage of 1. Hence the ratio of the weightage of all errors is 1. The errors hence are taken just once while calculating the MAE. Hence it emphasizes a different aspect of the performance by taking into consideration all the errors with the same weightage. RMSE however doesn't give an equal weightage to all the errors. It gives the error a weightage of the numeric value of the error itself. Hence we take the squares of the errors as it can be seen in the formula of RMSE. Hence it emphasizes a different aspect of the performance by taking into consideration all

the errors with different weightage. Hence RMSE is usually used when large errors are undesirable.