Question 1

a. Predicting whether a patient has COVID or not.
Classification.
b. Grouping similar news together.
Clustering.
c. Predicting the genre of a book.
Classification.
d. Predicting your percent grade in this course.
Regression.
e. Recognizing English characters in an image.
Classification.
f. Folding the laundry automatically.
Reinforcement Learning.
g. Predicting if the weather will be cold, warm, or hot tomorrow.
Classification.
h. Playing chess
Reinforcement Learning.

Question 2

ML is not needed where we don't have the data. Another domain is the Markov Decision Process. They have the potential to grow computationally huge!
Another problem is that of data cleaning. We don't need ML to do that.

Question 3

Classification accuracy for a random classifier is:

Accuracy = 1/k (k is the number of classes)

In binary classification, k=2:
So, the accuracy will be: 1/2 = 50%

Question 4

Because we have to stop after our maximum depth, the last node becomes the terminal node which determines the outcomes!

```
1: terminal(guess)
2:     outcomes = [row[-1] for row in guess] //we are selecting the most common class value
3:       set= (set(outcomes), key=outcomes.count) //we are creating a set of different
outcomes: yes or no. and then the key corresponds to the count of each outcome
4:     return max(set) //the outcome with most count is our final prediction
```

```
DecisionTreeTrain(data,remaining features)
1:
2:     if the labels in data are unambiguous then
3:         return terminal(guess) // base case: no need to split further
4:     else if remaining features is empty then
5:         return terminal(guess)
6:     else if depth==3
7:         return terminal(guesses) //we might not have a single guess value.
8:     else if depth<3 // we need to query more features
9:         for all f ∈ remaining features do // we are selecting the best fetaure
10:            NO ← the subset of data on which f=no
11:            YES ← the subset of data on which f=yes
12:            score[f] ← # of majority vote answers in NO
13:    + # of majority vote answers in YES// the accuracy we would get if we only queried on f
14:        end for
15:    f ← the feature with maximal score(f)
16:    NO ← the subset of data on which f=no
17:    YES ← the subset of data on which f=yes
18:    depth= depth+1
19:    left ← DecisionTreeTrain(NO, remaining features ¥ {f })
20:    right ← DecisionTreeTrain(YES, remaining features ¥ {f })
18:    return Node(f, left, right)
19:    end i
```

b.

We don’t have to change anything because we have a pre-made tree which will give us the answer!

Question 5

Outlook	Temperature	Humidity	Wind	Play
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

We have five 'YES' and four 'NO'. So, we calculate the probability of our output variable that is 'PlayTennis.' It is (-0.36)log(0.36)-0.64log2(0.64). It comes to be 0.94. So we need to have more Information Gain.
So, let's select the first splitting attribute:
Humidity..
Entropy of High where we have 3+ and 4-, so our entropy will be 0.985
Entropy of Normal where we have 6+ and 1-, so our entropy will be 0.592
So, Humidity gain is: Gain(P, Humidity) = 0.94-(7/14)*0.985-(7/14)*(0.592) = 0.151

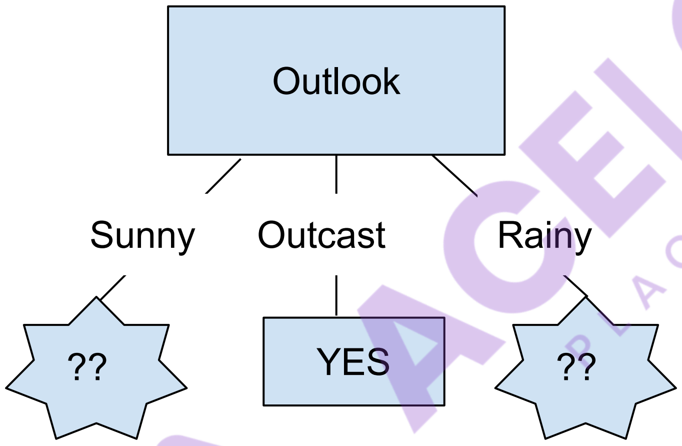
Wind..
Entropy of Weak where we have 6+ and 2-, so our entropy will be 0.811
Entropy of Strong where we have 3+ and 3-, so our entropy will be 1.0
So, Wind gain is: Gain(P, Wind) = 0.94-(8/14)*0.811-(6/14)*(1.0) = 0.048

Outlook..

Entropy of Sunny where we have 2+ and 3-, so our entropy will be 0.971
Entropy of Overcast where we have 4+ and 0-, so our entropy will be 0.0
Entropy of Rain where we have 3+ and 2-, so our entropy will be 0.971
So, Outlook gain is: Gain(P, Outlook) = 0.94-(5/14)*0.811- 0 -(5/14)*(1.0) = 0.247

Temperature..
Entropy of Hot where we have 2+ and 2-, so our entropy will be 1.
Entropy of Cold where we have 3+ and 1-, so our entropy will be 0.811
Entropy of Rain where we have 4+ and 2-, so our entropy will be 0.918
So, Temperature gain is: Gain(P, Temperature) = 0.94 - 0.811*(4/14) - 1*(4/14) - 0.918*(6/14) = 0.029

The one with the maximum split is **OUTLOOK**.



Now, We have to decide for the appropriate feature at the question marks.
Let's start with the sunny!
When it comes to sunny, we have 2+ and 3-. These belong to 1, 2, 8, 9, 11 data instances.
Entropy of Sunny is -⅖ log2 ⅖ -⅗ log2 ⅗ = 0.97.
Temperature:

Entropy of S_hot is 0.0(0+,2-)
Entropy of S_cold is 0.0(1+,0-)
Entropy of S_mild is 1.0(1+,1-)
So, Gain(Sunny, Temp)= Entropy(Sunny)-⅖(S_hot)-⅕(S_cold)-⅖ (S_mild)= 0.570.

Humidity:

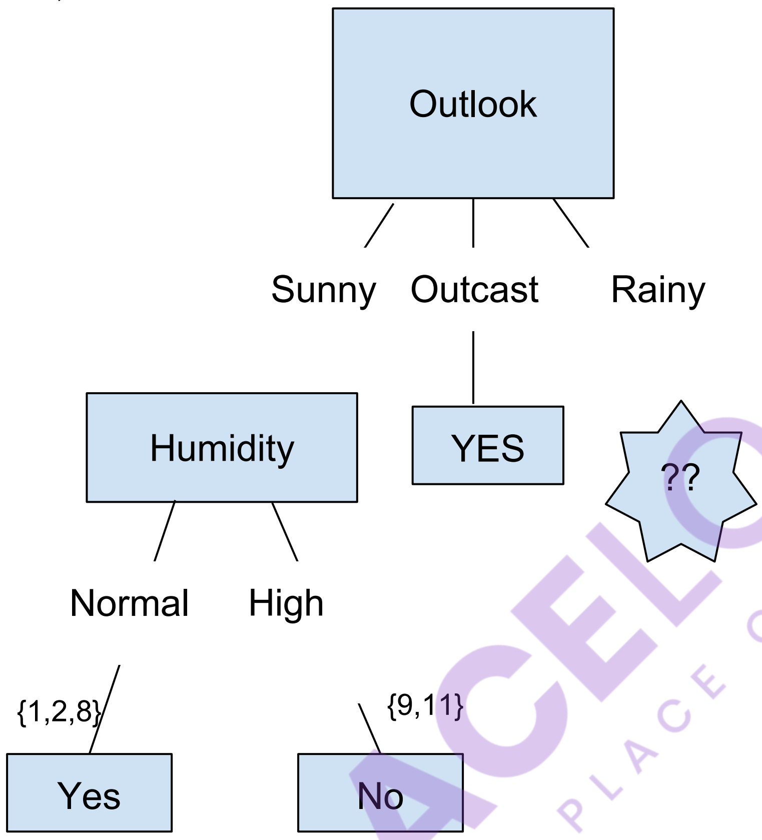
Entropy of S_high is 0.0(0+,3-)
Entropy of S_normal is 0.0(2+,0-)
So, Gain(Sunny, Humidity)= Entropy(Sunny)-⅗ (S_high)-⅖ (S_normal)= 0.970.

Wind:

Entropy of S_strong is 1.0(1+,1-)
Entropy of S_weak is 0.9183(1+,2-)
So, Gain(Sunny, Wind)= Entropy(Sunny)-⅗ (S_weak)-⅖ (S_strong)= 0.0192.

So, Humidity Will get selected!!!

Now, our tree becomes



When it comes to rain, we have 3+, 2-.
Entropy of Rain is -⅖ log2 ⅖ -⅗ log2 ⅗ = 0.97.
Temperature:

Entropy of S_hot is 0.0(0+,0-)
Entropy of S_mild is 0.9183(2+,1-)
Entropy of S_cold is 1.0(1+,1-)
So, Gain(Sunny, Temp)= Entropy(Sunny)- 0/5 (S_hot)-⅖ (S_cold)-⅗ (S_mild)= 0.0192.

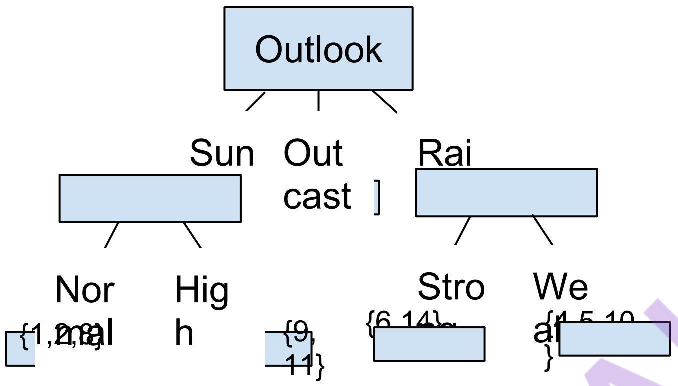
Humidity:

Entropy of S_high is 1.0(1+,1-)
Entropy of S_normal is 0.9183(2+,1-)
So, Gain(Sunny, Humidity)= Entropy(Sunny)-⅗ (S_normal)-⅖ (S_highl)= 0.0192.

Wind:

Entropy of S_strong is 1.0(0+,2-)
Entropy of S_weak is 0.9183(3+,0-)
So, Gain(Sunny, Wind)= Entropy(Sunny)-⅗ (S_weak)-⅖ (S_strong)= 0.970.

So, Wind Will get selected!!!



This is our final decision tree!

Question 6

It is because it is non-differentiable and discontinuous. So, the traditional optimization functions that we use for regression problems are not applicable to it.

Question 7

It is conceptually different because, without this type of bias, induction will not be possible at all. That's because a problem can be generalized into many domains. Without inductive bias, the model will be able to classify the examples it has previously seen, but wouldn't be able to generalize well to the new, unseen data.

Question 8

If some of the models are performing exceptionally well and have a large enough dataset for training, we can just perform the spot-checking of these algorithms and select the best out of them. This approach is used most of the time. There is no need to retrain the model. If a hyperparameter tuning did the best, it doesn't suddenly imply that the model itself performed really well.
For example, Naive Bayes can perform well in the case of independent features. However, when it comes to dependent features, it doesn't perform very well. So, if our features are highly correlated, we might make the performance better by changing the hyperparameters. But that doesn't automatically make Naive Bayes fit for our problem.

Question 9

Once you have tuned your model, that’s the best you could achieve. Now, we are moving forward to test the generalization of our model on “unseen”, “test” data. If you temper with that, it will no longer be indicative of your model’s real performance.