1 General questions (20 points)

1: What is machine learning?

Machine learning is a type of artificial intelligence, in which computer algorithms can improve based on data and experience. They are very much reliant on data. Their are 3 main types of machine learning algorithms: supervised learning, unsupervised learning, and reinforcement learning. The basics for most learning algorithms are as follows:

- Start with a random piece of data or some kind of null element
- get answer from computer algorithm
- get feedback on answer
- change algorithm based on feedback
- repeat until performance is satisfactory

2: What is supervised, unsupervised learning?

When the programmer has access to labeled training data, i.e. the answers are theoretically known by the programmer, this is called supervised learning.
In unsupervised learning, you are given just data examples and not their corresponding labels, so it is the job of the programmer to make sense of that data without knowing their labels.

3: What are features and target?

Features are the columns of data that give some characterization to the data set. Features can be categorical, binary, or numerical.
The target is the output of the machine learning algorithm based on the data given from the features.

4: What is the difference categorical and numerical features/target

A categorical feature/target has a pre-determined set of possible values, lets call its cardinality V, that can be mapped to V-many binary indicator features.
Numerical features/targets output anything that can be represented numerically with integers or decimal numbers and do not have to be limited to a pre-determined set.

5: What does it mean for a feature to be relevant?

For it feature to be relevant, its output for each data point in the set is somehow (through a machine learning algorithm) related to the final output/target of the data point i.e. The target variable is dependent on the feature; the two are not independent.

6: What is overfitting?

Overfitting is when a machine learning algorithm pays too much attention to idiosyncracies of the training data, and isnt able to generalize well. Often this means that your model is fitting noise, rather than whatever it is supposed to fit.

7: What is error?

Error is a measure of how accurately a machine learning algorithm is able to predict the target for previously unseen data (the test set).

8: What is distance measure?

A distance measure is an objective score that summarizes the relative difference between two objects in a problem domain, these objects are usually rows in a data set that have defined features that describe a subject or event.

9: What are some common error measures? - give definitions/formulae (2 points)

Mean Absolute Error:
The Mean Absolute Error measures the average of all absolute errors. Its formula where n is the number of errors is below:

$$\frac{1}{n}\sum_{i=1}^{n}|x_i - x|$$

Root Mean Squared Error:
Root Mean Squared Error is the square root of the average of squared errors. Consequently, RMSE is sensitive to outliers. RMSE can give you and good idea of how concentrated the data is around the line of best fit. It's forumal, where n is the number of predictions is below:

$$\sqrt{\frac{\sum_{i=1}^{n}(\hat{x}_i - x_i)^2}{n}}$$

10: What are some common distance measures? - give definitions/formulae (2 points)

Euclidean Distance:
The Euclidean distance between two points in either the plane or 3-dimensional space measures the length of a segment connecting the two points.
a and b are two points on a real line, the Cartesian coordinates for a and b are $(a_1, a_2, ..., a_i)$ and $(b_1, b_2, ..., b_i)$

$$d(a,b) = \sqrt{\sum_i (a_i - b_i)^2}$$

Manhattan Distance:

Manhattan distance is calculated by taking the sum of the absolute differences between the two vectors.
The Manhattan distance between two points (x_1, y_1) and (x_2, y_2) is given by the following formula:

$$|x_1 - x_2| + |y_1 - y_2|$$

11: What is decision boundary?

A decision boundary is a visualization technique the separates data into particular regions, signifying a difference in class for data points in different regions. This is helpful after training a machine learning model, because it can easily visualize classification of data-points in a feature space(usually a scatter plot).

12: Explain gradient descent (2 points)

Gradient descent is an optimization algorithm that is meant to find the values of parameters of a function that minimize the cost of the function itself.
The goal is the numerically find a local minimum to the function (i.e. f' = 0). Basically how the this optimization algorithm works is we choose a random point along the function, if our gradient(slope) is negative than we move our point in the positive direction, and if our gradient(slope) is positive we move in the negative direction over the x-axis. If our gradient is 0 we check that the points is at a minimum and that is our optimizing parameter. The length of the steps in which you move one direction or another depends on the steepness of our slope, i.e. the closer the slope is to 0 the closer we are to our local minimum.

13: Explain expectation maximization algorithm (2 points)

Expectation Maximization algorithm is a method used in estimating a joint probability distribution for a data set with the assumption that our data has latent variables(variables that affect our data but are hidden).
It involves two main steps:

- Estimate any missing variables that may have effect on the classification of our data points
- Maximize or optimize the parameters of the model in the presence of this data to best explain the data set.

Its commonly used in unsupervised learning problems.

14: What is bias?

Bias is an error from incorrect assumptions in the learning algorithm and represents the accuracy of our algorithms predictions. Bias is the accuracy of our model prediction, in other words a high bias means the a prediction will most likely be inaccurate and cause underfitting.

15: What are parameters of a model? What are hyper parameters?

A parameter of a model is a configurations variable within the model whose value can be estimated from data. They are a requirement for the model when making predictions and are often saved as a part of the learned model.
A hyperparameter of a model is a configuration variable external to the model whose value cannot be estimated from data. They are often used in processes to help estimate model parameters and are usually set by the programmer. They are also usually tuned for the specific predictive model.

16: Explain training and test sets

The training set is the data set on which your algorithm is expected to learn. Based on our training set, our learning algorithm induces a function f that will map a new example to a corresponding prediction.
The test set is the collection of examples on which we will evaluate our algorithm. We do this by testing our function f on the test set and evaluate the error of the output predictions

2 Naive Bayes (15 points)

1: State Bayes theorem

The following theorem describes the probability of an event, based on prior knowledge of conditions that might be related to the event:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

2: What is the "naive" assumption in Naive Bayesian algorithm?

The "naive" assumption is that the features are independent, so say we are comparing the output S of two features A and B, Naive Bayesian algorithm says:

$$P(S|B,A) = \frac{P(B,A|S)P(S)}{P(B,A)} = \frac{P(A|S)P(B|S)P(S)}{P(A)P(B)}$$

3: Is it justified? If so, how? If not, why is naive baysian used?

When the "naive" assumption holds, i.e. no features are related to another feature in the data set, than the algorithm performs very well with the need of even less data. However, in real life models this is hardly ever the case, so it isn't the best predictor for many models. It is a simple to understand model that is good for very specific data sets, so in some was its assumption is justified.

4: Give the naive baysian algorithm.

First, convert the data into a frequency table, where the number of outcomes is labeled for each output type in a feature.
Then, create a likelihood table (or a summary table) that tells you the probabilities for these feature's classes.
Then use Baysians theorem to calculate the posterior probability for each class. The class with the highest probability is the predicted outcome for the particular data entry.

5: For 3 categorical features (f1,f2,f3) with f1,f2 boolean and f3 being able to take values a,b and c, and boolean target (t) describe the table you would need, and what values go in each cell.

The data table would look something like the one below:

f_1	f_2	f_3	t
(T or F)	(T or F)	(a or b or c)	(T or F)
...	...	...	...

The frequency table would look like the one below:

Features	t = T	T = F
f1 = T	frequency	frequency
f1 = F	frequency	frequency
f2 = T	frequency	frequency
f2 = F	frequency	frequency
f3 = a	frequency	frequency
f3 = b	frequency	frequency
f3 = c	frequency	frequency

And from our frequency table we make our summary table like so:

Features	t = T	t = F
f1	$\frac{frequency_{f1=T}+frequency_{f1=F}}{total-entries}$	$\frac{frequency_{f1=T}+frequency_{f1=F}}{total-entries}$
f1 = T	$\frac{frequency}{frequency+frequency_{f1=F}}$	$\frac{frequency}{frequency+frequency_{f1=F}}$
f1 = F	$\frac{frequency}{frequency+frequency_{f1=T}}$	$\frac{frequency}{frequency+frequency_{f1=T}}$
f2	$\frac{frequency_{f2=T}+frequency_{f2=F}}{total-entries}$	$\frac{frequency_{f2=T}+frequency_{f2=F}}{total-entries}$
f2 = T	$\frac{frequency}{frequency+frequency_{f2=F}}$	$\frac{frequency}{frequency+frequency_{f2=F}}$
f2 = F	$\frac{frequency}{frequency+frequency_{f2=T}}$	$\frac{frequency}{frequency+frequency_{f2=T}}$
f3	$\frac{frequency_{f3=a}+frequency_{f3=b}=frequency_{f3=c}}{total-entries}$	$\frac{frequency_{f3=a}+frequency_{f3=b}+frequency_{f3=c}}{total-entries}$
f3 = a	$\frac{frequency}{frequency+frequency_{f3=b}+frequency_{f3=c}}$	$\frac{frequency}{frequency+frequency_{f3=b}+frequency_{f3=c}}$
f3 = b	$\frac{frequency}{frequency+frequency_{f3=a}+frequency_{f3=c}}$	$\frac{frequency}{frequency+frequency_{f3=a}+frequency_{f3=c}}$
f3 = c	$\frac{frequency}{frequency+frequency_{f3=a}+frequency_{f3=b}}$	$\frac{frequency}{frequency+frequency_{f3=a}+frequency_{f3=b}}$

6: For 3 numerical features (f1,f2,f3) and boolean target (t) describe the table you would need, and what values go in each cell

Our data table will look something like the one below:

f1	f2	f3	t
(numerical value)	(numerical value)	(numerical value)	(T or F)
. . .	. . .	. . .	. . .

Our ”frequency” table would look something like this with sets of numerical values:

Features	t = T	t = F
f1	(numerical value), (numerical value), . . .	(numerical value), (numerical value), . . .
f2	(numerical value), (numerical value), . . .	(numerical value), (numerical value), . . .
f3	(numerical value), (numerical value), . . .	(numerical value), (numerical value), . . .

Then we can calculate the mean and standard from our set of values in the frequency table:

Features	$\mu_{t=T}$	$\sigma_{t=T}$	$\mu_{t=F}$	$\sigma_{t=F}$
f1	$\frac{1}{n}\sum_{i=1}^{n} x_i$	$(\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\mu))^{\frac{1}{2}}$	$\frac{1}{n}\sum_{i=1}^{n} x_i$	$(\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\mu))^{\frac{1}{2}}$
f2	$\frac{1}{n}\sum_{i=1}^{n} x_i$	$(\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\mu))^{\frac{1}{2}}$	$\frac{1}{n}\sum_{i=1}^{n} x_i$	$(\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\mu))^{\frac{1}{2}}$
f3	$\frac{1}{n}\sum_{i=1}^{n} x_i$	$(\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\mu))^{\frac{1}{2}}$	$\frac{1}{n}\sum_{i=1}^{n} x_i$	$(\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\mu))^{\frac{1}{2}}$

Once we have this table we can use the formula for normal distribution for any new numerical value of x as a given feature:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}e^{\frac{(x-\mu)^2}{2\sigma^2}}$$

use this formula for both t=T and t=F and the largest probability will be the chosen outcome for x.

7: Research to find out what laplacian smoothing is, and how it is applied to naive baysian.

Laplacian smoothing is a smoothing technique that helps tackle the problem of zero probability in a Naive Baysian algorithm. Using a smoothing paramenter α, we are able to move the likelihoods closer to 0.5 instead of dealing with zero probability.

3 K nearest neighbors (10 points)

1: Give the basic k NN

K Nearest Neighbors algorithm works under the assumption that similar things exist in close proximity to one another, to do this we use various distance measurements between points on a graph (usually Euclidean distance).
The algorithm works like so:

- Initialize a value for k that is the chosen number of neighbors for your data set
- To get a prediction classification, iterate from 1 to the total number of points in the training set
  - Calculate the distance between each data point in the test set to the training set using whatever distance measurement is relevant to this data
  - then sort the calculated distances in ascending order

  - Get top k rows from the sorted data
  - Get the most Frequent classification of these rows and return that as the predicted class
- run the algorithm again over several different values of k to choose a k that reduces the number of errors we encounter while the algorithm still gives accurate predictions.

2: How is it modified to include categorical vs numerical output

Since KNN is very reliant on distance formulas, it makes dealing with categorical output difficult. KNN thrives on numerical features and output, so converting categorical features or output INTO a(or several) numerical feature(s)/output(s). There are several ways to do this, creating new variables whose value is 1 for the correct categorical value and 0 otherwise. You can also normalize the data by giving each categorical value a unique ID. These methods are never absolute solutions to this problem but can still make the algorithm perform well with various data sets.

3: Why should you normalize features when using knn?

Normalization is the process of converting data values between the minimum 0 and the maximum 1. This process allows us to compare features, because computing distances for non-normalized data will assign higher weights to features with wider ranges of values.

4: How can you help the model with prior knowledge of feature importance?

We can use this prior knowledge of feature importance to assign a score for features based on their importance to predict our target. Features that are more responsible for predicting the data targets will have a higher score and will make our data more interpretable.

5: How would you modify the knn to indicate the distance of various neighbors?

You can add an additional weight to the algorithm, to indicate that the closer in distance that one data point to another using any preferred distance metric. The more likely it is to be classified to a specified target and to be neighbors.

6: How would you modify knn to include all neighbors within a distance

Instead of using k as our metric to determine neighbors, we could simply use distance as our metric to determine all neighbors of a certain data point within a defined distance of it.

7: What behaviour do you expect as k increases? In training set vs test set

Their is an optimal solution for k, and as k increases past this, the less likely we are to get any meaningful categorization of data in our training set, and the more likely we are to get a high error percentage for the test set.

8: http://vision.stanford.edu/teaching/cs231n-demos/knn/ (Links to an external site.) Run the program. Describe the behaviour as you vary the parameters. (2 points)

This algorithm works best we data set where related points are clustered very close to one another. A lower number of points gives each point a heavier weight when it comes to classification of neighbors. As k increases, the space of data that can be unclassified seems to increase, and this increases with an increased number of points as well.

9: Research "lazy learning" - why is knn a lazy method?

K-NN is a lazy learning method because "memorizes" the training dataset instead of learning a classification function from the training data.

4 Decision trees (10 points)

1: Describe the basic decision tree algorithms with categorical features and categorical output

Choose the "best" decision attribute, lets call it X, to be be your root value. Then, for each categorical value of X create a new descendant node. Then sort the training examples to leaf nodes, if they are classified purely and without ambiguity, stop, else, recurse algorithm over new leaf nodes.

2: Give the formula for entropy - what does it measure. When is it maximum and when is it minimum

Entropy is the measure of hidden information or uncertainty.
X is a discrete random variable with pmf $p_X(x)$, the formula for its entropy is as follows:

$$H(X) = -\sum_x p(x)log(p(x))$$

Entropy has a minimum value of 0. This happens when their is only one output for an event i.e. its probability is 100%
Entropy has a maximum value of 1. This means that the data set would have an equal probability for each output in the feature

3: Give the formula for mutual information - what does it measure? When is it maximum and when is it minimum

In regard to two random variables, mutual information measures the amount of information that one variable contains about the other
The formula for the mutual information between X and Y is as follows:
$MI(Y, X) = MI(X, Y) = H(Y) - H(Y|X) = H(X) - H(X|Y)$, where H is entropy.

Mutual information has a minimum of 0, which means that the two variables are independent from one another.
Mutual information has a maximum of $\min[H(var1), H(var2)]$ or the smallest entropy between the two features. This means that one feature determines the other.

4: Why is mutual information problematic to use if a feature has lots of values that it can take? How do you fix this?

5: Give the formula for Gini index - what does it measure

Gini Index $= 1 - \sum_j p_j^2$
The gini index measures the frequency at which any element of the dataset will be mislabelled when it is randomly labeled.

6: What changes do you make when the target is numerical

We can separate numerical data into bins, where each bin is a specified range of numbers. So our target answer would be a bin categorization instead of numerical value. We do this because decision tree's very heavily rely on categorical data.

7: What changes do you make when the features are numerical

We can use the bin method on numerical features as well in order to have them in some sort of categorical data form.

8: What is the bias used in decision trees? What is the justification

The decision tree algorithm makes the most optimal decision at each node of the tree, which means it is making locally optimal decisions recursively in the assumption that this will lead to the best solution. The algorithm is recursive so only works on one subset of the tree at a time, and only considers the best result for subset. If it is not clear which attribute to choose for the next nodes of a sub-tree, than the choice is considered arbitrary.

9: What is pruning? Describe how to prune a tree.

First, make a full binary decision tree based on the training data. Then use the decision tree to test the test data and find its output error. Then randomly choose a node that is not a leaf and cut the tree at that node. Then use the test data to find the new cut trees output error. If the new tree performs better (has a lower error) than keep the new tree and continue, if not than don't cut.

10: What makes the algorithm greedy? How can you make it less greedy?

The algorithm is recursive so only works on one subset of the tree at a time, and only considers the best result for subset. In other words, it uses a divide and conquer approach, making it a greedy algorithm. A non-greedy algorithm would have the goal of finding the most optimal decision tree. You could do this by running the decision tree algorithm several times and finding the tree with the best accuracy.

5 K-means clustering (7 points)

1: Describe the k-means clustering algorithm

This algorithm uses clusters, which are a collection of data points aggregated together because of similarities. A target K refers to the number of centroids needed to classify the data. A centroid is the imaginary or real location representing the center of the cluster. The algorithm starts with a first group of randomly selected centroids, which are used as the beginning points for every cluster. We then do this iteratively until we can optimize the location of the centroids. We stop interating once either the centroids have no change to their values or a defined number of iterations have happened.

2: What are the parameters and hyper parameters?

The parameters and hyper parameters for K Means Clustering are as follows: the number of clusters(k) we want to separate the data into, some type of planned or random generation (more common) to determine where centroids will be tested, the maximum number of iterations required, and the number of iterations performed.

3: How is the goodness of clusters measured? How does this measure change with k?

A good cluster is usually one that has a centroid that is stabilized, meaning the centroids have no change to their values and are taking up the smallest amount of space possible. As the value of K increases, there will be fewer elements in the cluster and the average distortion of the algorithm will decrease. The lesser number of elements means closer to the centroid. Too many centroids will not give us enough information to classify anything and not enough centroids will have a higher error probability.

4: Describe the elbow method for choosing the number of clusters

The elbow method for determining value of k where the average distortion declines the most, giving us the most optimal value of k. We can visually determine this by graphing error against number of clusters, and finding our lowest sharp turn on the graph.

5: K-means clustering is greedy and will find the local minimum. What can be done to combat this?

There are several ways to combat finding local minimums and trying to find a global minimum instead, the easiset would be to iterate the algorithm over mutliple times and find the most optimal minimum given from these iterations.

6: Research and outline an argument stating that k-means clustering run will converge (2 points)

I had a lot of trouble trying to find information on this.

6 (3points)

1: Assume you have developed an ml system to predict tornadoes.
Out of 100 instances of tornadoes, it correctly identifies 95.
Out of 1000000 instances of non-tornadoes, it incorrectly claims 1000 are tornadoes.
If it predicts a tornado tomorrow, will you send out a tornado warning? Why or why not

With our data set of 100 tornadoes, the probability of it correctly identifying a tornado is $\frac{95}{100} = 95\%$ and its probability of not identifying a tornado is $\frac{5}{100} = 5\%$ which are both pretty accurate readings. Ideally we'd like this data set to be a lot bigger to give us more accuracy but these odds are still very good
With our 1000000 non-tornadoes data set, the probability of it correctly identifying non-tornadoes is $\frac{1000-1000000}{1000000} = 99.9\%$, and its probability of giving a false tornado claim is $\frac{1000}{1000000} = 0.1\%$ which tells us its very unlikely to give a false claim.
With these probabilities I'd confidently say that I'd send out a tornado warning, because it is very unlikely to identify a tornado falsely. I think it's also important to note the context of a situation like this, where the safest thing would be to release a tornado warning so that people are prepared for a natural disaster in case of one.