**logistic regression**

classification, \*loss function and compare with linear regression

* Q: State Key Differences between Logistic Regression and Linear Regression
  + Ans:

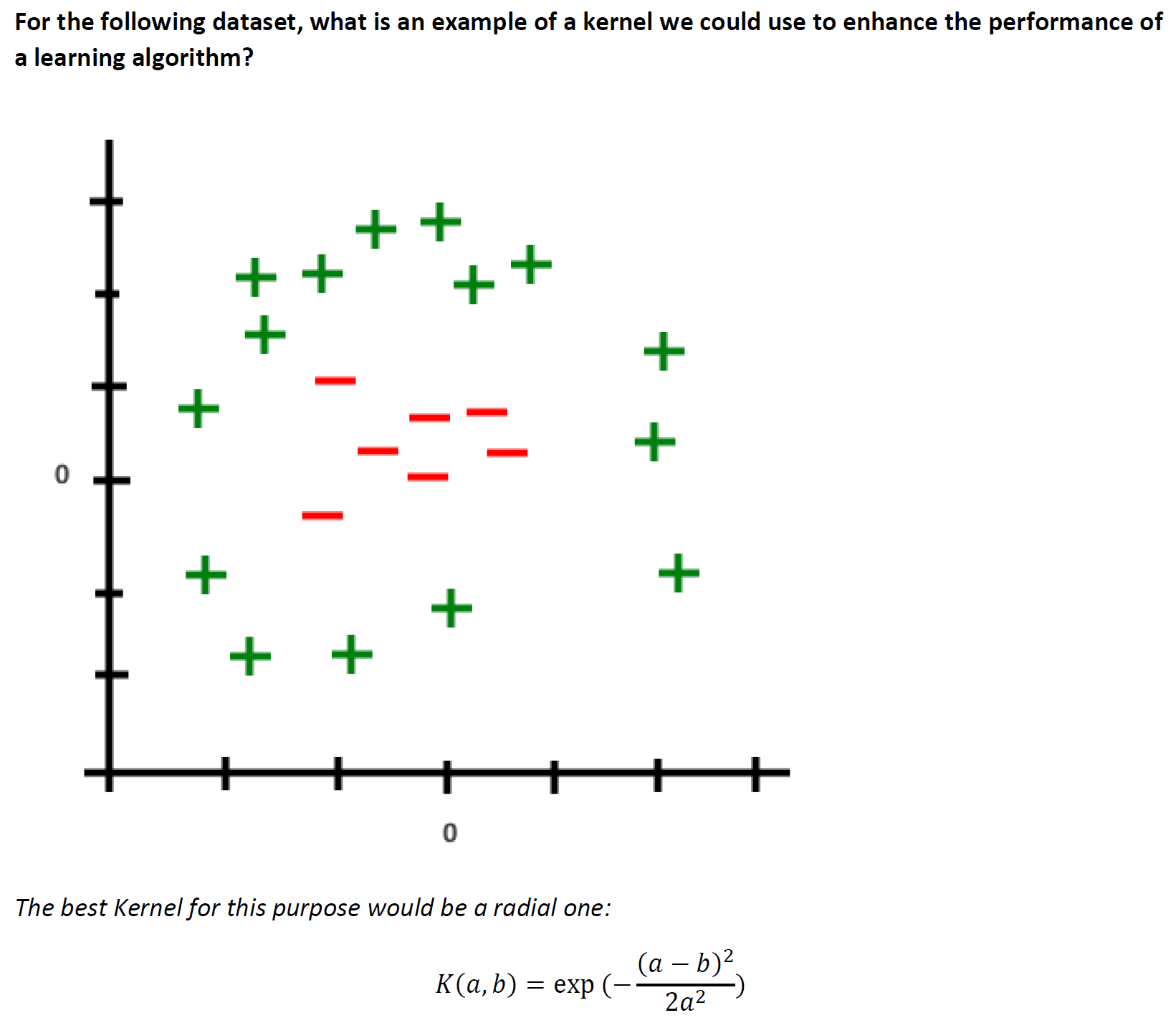
1. The Linear regression models data using continuous numeric value. Logistic regression mainly used for binary classification & models the data in the binary values.
2. Linear regression develops a linear relationship between Feature values and Label. Whereas there is no relationship developed for logistic regression. Since it’s a probabilistic model, Logistic regression predicts the probability if a data point belongs to a particular class or not.
3. The loss function used for Linear regression is to minimize Mean Squared Error. While for Logistic Regression the loss or cost function is to minimize log function which is also called as logistic function.

* Q: What is the difference between logistic regression and linear regression?
  + A: Logistic regression is a binary classifier which uses gradient descent to fit a sigmoid function ( 1 / 1−𝑒−𝑦) to a training dataset. Linear regression is a regressor which fits a line to a continuous training dataset in order to make non-binary predictions.
* Q: What’s the difference between linear regression and logistic regression?
  + A: In Linear Regression, we predict the value by an integer number/ no activation function is used/ no threshold value. In Logistic Regression, we predict the value by 1 or 0/activation function is used to convert a linear regression equation to the logistic regression equation /a threshold value is needed.
* Q: Compare logistic regression with linear regression.
  + A: For loss functions- log loss function used for logistic regression, mean-squared error is commonly used for linear regression. Linear regression is used when there is a continuous outcome variable – an output that can take an infinite number of possible values. Logistic regression classification is used for discrete categorical classes, like predicting yes/no, true/false, 1/0, etc.
* **True-False: Is it possible to apply a logistic regression algorithm on a 3-class Classification problem?**

A) TRUE  
B) FALSE

* **Solution: A**

Yes, we can apply logistic regression on 3 classification problem, We can use One Vs all method for 3 class classification in logistic regression

* Q: Why do we not use MSE (Mean Squared Error) for our cost function when optimizing Logistic Regression? (Assume we are optimizing using gradient descent)
  + A: We can not use SSE or MSE for optimizing Logistic Regression because the function would not be convex everywhere, meaning the gradient descent process would not be guaranteed to find an absolute minimum. Additionally, MSE does not accurately represent the loss of the logistic regression predictions. Because the log regression only outputs a value between 0 and 1, the maximum loss measured by MSE would only be 1 (in the case where the actual value is 1 and the predicted value is 0).
* 
  + Below are the three scatter plot(A,B,C left to right) and hand drawn decision boundaries for logistic regression.

A picture containing clock

Description automatically generated

* + **22) Which of the following above figure shows that the decision boundary is overfitting the training data?**

A) A  
B) B  
C) C  
D)None of these

* + **Solution: C**
  + Since in figure 3, Decision boundary is not smooth that means it will over-fitting the data.

**multi-layer neural networks**

tanh activation function and motivation

\*forward propagation to compute output

backward propagation to compute weights

inductive bias

\*hyperparameters and impact on underfitting, overfitting

* **Q**. What is the inductive bias for Neural Networks?
  + A: Inductive bias of neural networks is that they prefer smaller weights, as well as that they use smooth interpolation between points.
* Q: What is the inductive bias for multi-layer neural networks?
  + A: Smooth interpolation between data points. It also prefers small weight value.
* Q: What is one inductive bias of multi-layer neural networks?
  + A: The number of hidden units.
* Q: Like any other Machine Learning Algorithm, Neural Networks have an inductive bias. Name two.
  + A1: Smooth interpellation between data points, this means that any two points with the same class value will have the points between them be the same class value
  + A2: Prefer Sparse Weight vectors or small weight values. Large ones can make the network over adjust to minor differences and therefore overfit.
* Q: **Name and describe the inductive bias (s) of neural networks**
  + Same as what we saw for perceptron.
  + Smooth interpolation between data points meaning that any two points with same class value impose the same class value to other points along the function line have the same class value
  + Prefer spare weight vectors meaning that large weight values can result in over adjustments to minor changes of input and small weight values can help to eliminate factors with low impact resulting in lower chance of overfitting the data.
* Q: Multi-Layer Neural Networks are capable of doing things other simple classifiers like perceptrons can not. What is a simple example of this?
  + A: Multi-Layer Neural Networks can handle XOR
* Q: **In mathematics a Taylor series can represent an approximation of a given function. When thinking of how functions can be approximated by other functions, what model in machine learning recognizes its ability to approximate any function?**
  + Two-Layer Networks
* Q: **Describe test and training performance as you increase the number of hidden units in a neural network?**
  + The number of units in a model when increased tends to increase the performance of the training set. At some point we expect the test performance to get worse showing that increasing the number of hidden units can overfit the data.
* Q: **Why do we not initialize the weights of the neural networks input and hidden units to 0?** 
  + The evolution of the weights will tend to be uniform across weights since the weights feeding into the hidden units are the same. This generates uninteresting behavior and tends to result in local optimum that is not useful. Because of this we say that neural networks are sensitive to initialization.
* Q: How is a tanh function different than a sigmoid function? Why is this beneficial?
  + A: A Tanh function is a sigmoidal function that goes from (-1, 1) as opposed to (0, 1). This is beneficial as the negative points will be equally graphed on the negative side and the zero values will be placed closer to the median in the function.
* *Question*: Why do we use the hyperbolic tangent function to calculate the activation of a hidden node?
  + *Answer*: The hyperbolic tangent function approximates the 0/1 loss step function, but it is differentiable and so can be used to update weights using gradient descent.
* Q: Why is the hyperbolic tangent used as an activation function for neural networks?
  + A: It is differentiable everywhere and ranges from -1 to 1.
* Q: **The activation function, \_\_\_\_\_\_\_\_\_, of the neural network approximates the shape of the \_\_\_\_\_\_\_\_\_ that is used in the perceptron.**
  + Hyperbolic tangent
  + Step function (sign function by Daume)
  + Explanation: By replacing the step function with a differentiable function, like the hyperbolic tangent function, we are able to use the derivative of the activation function to perform gradient decent via backpropagation. This trains the neural network as a non-linear method.
* Q: How many layers are in a Neural network?
  + A: Trick question! There can be theoretically an infinite amount of hidden layers with a resulting one output layer and one input layer.
* Q: How many types of layers are in a Neural Network?
  + A: Three! The hidden layer(s), the input layer and the output layer.
* **Q.** What are the hyperparameters associated with Neural Networks?
  + **A.** We can tweak the number of layers, the number of hidden nodes per layer, the activation function, and the weight initialization.
* Question: What are some hyperparameters for a multi-layered perceptron?
  + Answer: Number of layers, number of nodes in hidden layers, learning rate, activation function of hidden layers, weight initialization and stopping criteria.
* Q: Name three hyperparameters for neural networks and how they can impact the model.
  + A: 1. Number of hidden layers: The more hidden layers there are, the more likely the network will be prone to overfitting a dataset. Also, each layer you add increases runtime and complexity. 2. Learning Rate: Must be chosen to where your network takes a reasonable amount of time to train, but not so high that it constantly overcorrects for incorrect data points. 3. Number of iterations: The more iterations that you run while testing, the more likely that your network will be prone to overfitting.
* Q: **Name some of the tuning parameters (hyperparameters) of the neural network**
  + Number of layers
  + Number of hidden units per layer
    - D dimensions, K hidden units,
      * (D + 2) K parameters
        + 1 for bias, 1 for weight to the output node
    - K = floor(N / D)
  + The gradient descent learning rate eta
  + Weight initialization (random small weights)
  + The stopping iteration or weight regularization
* Q: Describe one major difference between a perceptron and a multi-layer neural network.
  + A: 1) A perceptron is a single-layer network where the input is directly connected to the output, where a multi-layer neural network is a collection of perceptrons connected by hidden nodes/unit. 2) A perceptron has linear decision boundaries, whereas a neural network can express almost any geometric function with the trade-off of added complexity in tuning the parameters.
* Question: For a Perceptron with one hidden layer, in what order are the weights W and v updated?
  + Answer: All the weights, from the input layer (W) and from the output layer (v) should be updated simultaneously.
* Question: How can you modify a multi-layered perceptron to perform classification rather than regression?
  + Answer: Add multiple output nodes, one for each class value. When testing a data instance, the output node with the largest activation will correspond to the class value assigned to this data instance.
* Q: What is forward propagation and backwards propagation? How are they different?
  + A: Forward propagation is used to compute the result or output from the input whereas the backwards propagation is to train and help the NN learn from its mistakes.
* Q: Neural Network propagation forward requires a vector math equation. What is the name of this math equation and how does it work?
  + A: The name is  Dot product. The dot product is the summation of the multiplication of elements at the same index within the multiplied vectors:

[1,2,3].[4,5,6] = (1\*4)+(2\*5)+(3\*6) = 4+10+18 = 32

* Q: In a forward propagation neural network, which of the three types of nodes is responsible for calculations and does not interact directly with the data?

a) input node; b) hidden node; c) output node; d) none of the above

* + A: b, hidden node
* **Q.** Calculate the output of the following neural network using forward propagation:

Assume that the inputs from top to bottom are 2 and 3,

that weights between input and hidden layers from top to bottom are all 1,

and weights between hidden layer and output from top to bottom are 0.5, 0.6, and -1.

* **A.** Following forward propagation, we must first calculate the activation of each hidden node in the hidden layer.

**= tanh( weight \* input)**

h1 = tanh(2 \* 1 + 3 \* 1) = 0.9999

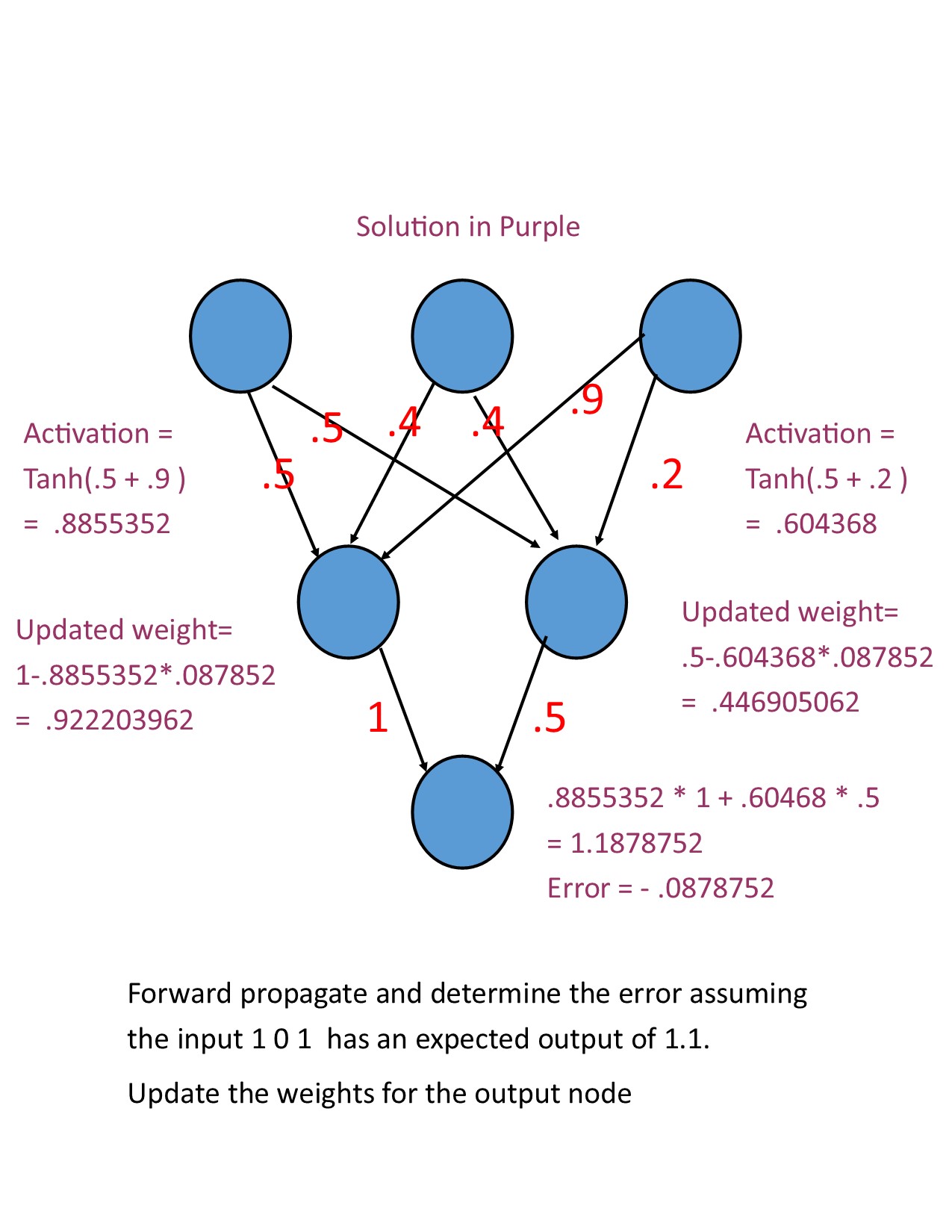
h2 = tanh(2 \* 1 + 3 \* 1) = 0.9999

h3 = tanh(2 \* 1 + 3 \* 1) = 0.9999

Using the activation values from the last step, calculate the output.

**Output =**

Output = (0.9999 \* 0.5) + (0.9999 \* 0.6) + (0.9999 \* -1) = **0.09999**

* Q: Forward propagate the data and determine the error assume the input 1 0 1 has an expected output of 1.1. Update the weights ONLY from the hidden nodes to the output node.
  + A:

**deep networks, convolutional neural networks**

convolution, pooling

* **Q.** Here we have 55\*55 convolution layer, and a filter with 5\*5. How many numbers will result from convolution?
  + **A.** (55-5)2
* Q: What is the point of pooling in a convolutional neural network?
  + A: Pooling can help the classifier ignore small differences in the picture and build an invariance to any other changes in the input.
* *Question*: What is the purpose of a convolution layer in machine learning?
  + *Answer*: It can act as a filter by reducing several features to one numerical value. It works well for images and can be used to extract significant aspects of an image. A convolution layer is also space invariant, so it can help a learner find certain features of an image regardless of where they are located in image.

**support vector machines - SVM**

constrained optimization, \*hard margin, \*soft margin

\*role of hyperparameter in overfitting and underfitting

slack parameters, size of the margin, kernels

* **Q.** What is the goal when constructing SVM?
  + **A.** Maximize margin. The bigger margin we have, the easier for SVM to classify data.
* Q: SVM's can be improved through a something that "brings a new perspective" to the data, reveals something that may not have been obvious in the initial data (data that wasn't linearly separable, now is). What is the name of this and what does it do?
  + A: This can be achieved through Kernels. Kernels transform data from one form into another by adding features using existing ones, through this the data becomes linearly separable. A simple example of this is if data points only have two features x and y, through z = x^2 + y^2, we get a third feature that will extrude far points from 0 upwards and therefore give another dimension to work with.
* **Q**. What is the inductive bias for Support Vector Machines (SVM’s)?
  + **A**. SVM’s tend to prefer a large margin. The assumption made is that distinct classes tend to be separated by wide boundaries.
* Q: What does the vector in support vector machine stand for? Why is it a vector?
  + A: The vector in support vector machine stands for the way that the points on the plane are stored, as they are used as vectors. They are vectors for the purpose of creating up to infinite-dimensional space that they can be used to plot on. This allows for an increase in complexity or size with no change to the purity of the data.
* Q: What does Zeta refer to when it is mentioned with SVM (support vector machines)? What type of SVM has zeta that is diverse?
  + A: The zeta refers to the distance the points on the data are to the margin of the decision. An SVM with a soft margin is the only type of SVM to use a Zeta with much significance since it determines the volatility of the data.
* Q: What SVM is more likely to lead to overfitting in a dataset: soft margin or hard margin?
  + A: Hard margin. This is because a single outlier can determine the boundary in a hard margin, which makes the classifier much more likely to train on the noise in the training data. Whereas a soft margin SVM is allowed to misclassify some datapoints.
* Q: What is hard margin and soft margin?
  + Hard margin works when data are linearly separable and be classified correctly.
  + Soft margin is an extension of hard margin SVM when data are not linearly separable or have noise or outliers. It introduces slack variables as a cost to move incorrect points by using hinge loss function.
* **Q**. What is the difference between a soft margin and hard margin SVMs?
  + **A.** Hard margin SVM’s only work with linearly separable data, and often can be thrown off by outliers. Soft margin SVM’s on the other hand allow for data to be not perfectly linearly separable by providing some amount of slack to each point that lies within the margin.
* Q: What is the main difference between a hard and soft margin SVM?
  + A: A hard margin SVM will only be able to classify linearly separable data while a soft margin SVM will be able to classify any set of data as it can support outliers or points that would not make the data linearly separable.
* Q:  What is the difference between a soft and a hard margin?
  + Answer: Overall, hard margins are more easily affected by noisy data, and a large number of outliers can change the margin, and lead to overfitting, where soft margins are much less susceptible to noisy data and outliers. Soft margins can ignore a certain amount of wrong classifications due to outliers, to show the most accurate separation between classes.
* **Q**. What is the different between hard margin and soft margin in SVM?
  + **A.** Hard Margin: seeks for hyperplane which do not have any data misclassified.

Soft Margin: by introducing slack , we slight tolerance would be accepted to those misclassified data.

Biggest difference: hard-margin cannot tolerate any misclassified data, but soft-margin can, in some degree depend on the slack

* Q: For SVMs, what is the relationship between hard margin, soft margin, overfitting, and underfitting?
  + A: Hard margin is used when the training data is linear separable. It is sensitive to outliers and will cause overfitting. Soft margin is used to reduce overfitting. In order to reduce overfitting. The SVM will also uses Kernel parameters.
* Q: Why do we use soft-margin SVMs instead of hard-margin SVMs?
  + A: Because minimization is NP-Hard for 0-1 loss, which is the loss function for a hard-margin SVM. With the soft-margin SVM, we get the hinge loss function which can be optimized using linear programming.
* **Q.** Understanding of slack

一張含有 時鐘, 團體, 空氣, 白色 的圖片

自動產生的描述

Here we have red line represent hyperplane, and both blue line represents different side of margin. Now, suppose green, purple, and pink dot are considered positive data.

In this case, please indicate the correspond slack value to the three dots and explain why.

* + **A.**

Green: slack = 0, green dot is classified correctly, does not need any modification.

Purple: slack < 1, although purple dot classified correctly according to hyperplane, it is still slightly out of margin.

Pink : slack > 1, pink dot classified incorrectly. Therefore, it needs a slack value bigger than one to help moving pink data to correct area.

* Q: Suppose you are building a SVM model on data X. The data X can be error prone which means that you should not trust any specific data point too much. Now think that you want to build a SVM model which has quadratic kernel function of polynomial degree 2 that uses Slack variable C as one of it’s hyper parameter. Based upon that give the answer for following question.

What would happen when you use very large value of C(C->infinity)?

Note: For small C was also classifying all data points correctly

A) We can still classify data correctly for given setting of hyper parameter C  
B) We can not classify data correctly for given setting of hyper parameter C  
C) Can’t Say  
D) None of these

* + A: A

Explanation:

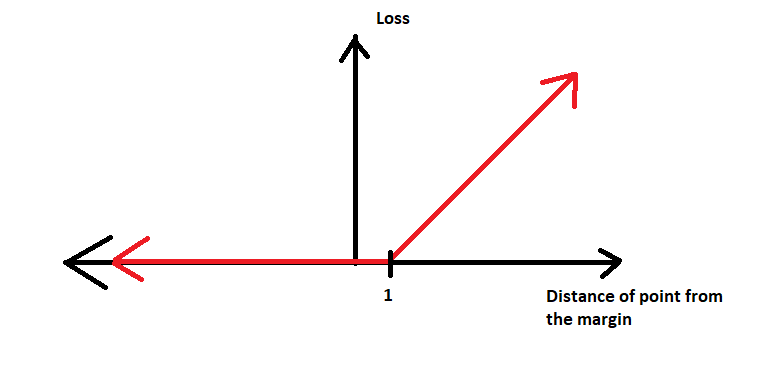
For large values of C, the penalty for misclassifying points is very high, so the decision boundary will perfectly separate the data if possible.

* Q) Explain for SVM can margin be negative? If Yes, explain. If No, explain.
  + Ans:

No, it cannot be negative. The inductive bias of SVM is to maximize the margin. Distance function is used to calculate the margin and distance is always positive.

Function: - y(w.xn +b) is always positive.

* Q) Draw the hinge loss for SVM.

Ans:

* Q: How is a kernel function able to let a SVM work on non linearly separable data?
  + A: A kernel function transforms the input data into a higher dimensional space in an attempt to create a linearly separable set of data points

**ensemble classifiers**

voting, resampling, \*bagging, \*boosting, AdaBoost, random forest

\*ways to establish diversity

* Q: What is an ensemble method and why are they useful?
  + A: An ensemble method is a technique that creates multiple models and combines them to get better results. Since multiple models are used this leads to more diversification which helps reduce variance and leads to a lower chance of overfitting.
* Q: What is the primary benefit of ensemble classifiers?
  + A: Using multiple classifiers can reduce the variance of the classifiers. As long as only a few of the classifiers make mistakes, variance can be reduced overall and better results can be achieved by using an ensemble of classifiers to solve a classification problem.
* Q: What is voting in Machine Learning? What models can it be used on?
  + A: Voting is the process of using different or the same models multiple time on a point. Each models output is counted as a vote for a classification. The class with the most votes wins and is used as the classification for that point.
* Q: Which of the following statements is true about using weak learners that are used in ensemble models?

1. Weak learners have high bias, and high variance.
2. Weak learners have high bias, and low variance.
3. Weak learners have low bias, and high variance.
4. Weak learners have low bias, and low variance.

* A: B. Weak learners tend to perform well on portions of the space. So, they usually don’t overfit and can’t solve hard learning problem.
* Q: Tree-based ensemble algorithms are resistant to
  1. Underfitting
  2. Overfitting
  3. A and b
  4. Neither

Answer: B. Tree-based ensemble algorithms use many weak learners like decision trees that underfit and combine their predictions into a strong learner. This reduces the overfitting of the model.

* Q: Majority vote is often used in ensembles of multiple classifiers. What methods may be used in ensembles of regression models?
  + A: One could take the mean or median of the predicted values from each regression model to determine the final predicted output.
* Q: What is the key concept of bagging?
  + The key concept of bagging is training the same kind of weak learners independently on different datasets that is generated by randomly selecting with replacement datapoints of the original dataset. Then combine the predictions and decide the final prediction by majority vote.
* Q: What is the key concept of boosting?
  + The key concept of boosting is training the same kind of weak learners based on the previous ones. The dataset for training the next weak learner is also dependent on the prediction of the previous weak learner, while weighting the datapoint that was misclassified higher than the correct ones. Combining the predictions by multiplying a prediction of each weak learner by its adaptive parameter, sum them up and take the sign as a final prediction.
* Q: Which best describes Bagging:

1) Bagging can be used to reduce overfitting

2) When using bagging the goal is to lower bias instead of variance

3) When using bagging the goal is to lower variance instead of bias

4) Individual learners are not dependent on each other in bagging

a) 1 & 2

b) 2 & 4

c) 1 & 4

d) 2 & 3

Answer: c

* Q: Which of these definitions best describes the concept of Boosting?

a. Using resampling to generate different variations of a training set, then using a classifier on each set and determining the optimal classifier

b. Using higher weights on training examples previously misclassified so they are focused on by the classifier in later epochs

c. Dividing training data into subsets and sampling from each of these subsets for different training data combinations for the classifier to train on

d. Dividing the labeled data into training and testing sets, and using each segment for their respective tasks

Answer: B

* Q: What are the benefits of bagging? Consequences? What about boosting?
  + A: Bagging has the tendency to overfit data occasionally however it is good at generalizing the data for smaller datasets for training. Boosting is also prone to overfitting with noisy data since the specific instances are more tuned to. Boosting however generally improves the accuracy of the algorithm more despite how it needs to be tuned very meticulously.
* Q: What’s the difference between bagging and boosting?
  + A: If the difficulty of the single model is overfitting, then Bagging is the best option because bagging can decrease variance in an overfit model. If the problem is that the single model gets a very low performance, boosting could generate a combined model with lower errors and reduces pitfalls of the single model.
* Q: What is a key difference between bagging and boosting?
  + A: Bagging can be viewed as a parallel process as each classifier can be trained on the data whenever, while boosting has to be done sequentially to adjust the weights based on the previous classifiers results.
* Q: Compare bagging and boosting.
  + A: Bagging and boosting are both ensemble methods, meaning that a set of learners are combined in order to create the best model. In bagging, ‘N’ number of training sets are generated by random sampling with replacement from the original training sets. In boosting, the observations are weighted.
* Q: What classifiers is boosting typically applied to, and how is boosting implemented?
  + A: Boosting is typically performed on weak classifiers (but better than random guess). Boosting is done by “learning from the last test you took”. You weight each data point, increasing the weights of the data points you got wrong and decreasing the weights of the data points you get correct. Done for any number of iterations. At the end of all the iterations, you have each classifier vote on every data point to get your predictions.
* Q: Which of the following options is correct regarding benefits of ensemble models?
  + - 1. Better performance
    - 2. Better interpretability
    - 3. Generalized models
    - a) 1 and 2; b) 1 and 3; c) 2 and 3; d) 1, 2, and 3
  + A: b, 1 and 3 are both benefits of the ensemble model. 2 is incorrect because with multiple models, interpretability is degraded.

**k means++**

cluster initiation, furthest first, probabilistic selection of cluster means

* Q: What is furthest first k-means initialization?
  + A: Attempts to improve the k-means algorithm by selecting a first center point, and then calculates the distance to every other point and then chooses the next center point to the point that was the farthest away from the original chosen point.
* NOTES:
* Cluster initiation:
  + Furthest first is an initialization heuristic
* Furthest first:
  + Steps:
    - 1. Pick a random data point for first mean (mew\_1)
    - For 2…k
      * Find data point m that is as far as possible from its closest previous mean (closest means either from step one or previous value from 2…k)
      * Maximize distance function ^
      * Next cluster mean will be that point ^^
  + K-means++ is similar to furthest first but when we pick the k-th mean, instead of choosing farthest possible point we choose a point with prob proportional to the squared distance
* Probabilistic selection of cluster means:
  + Like the weighted roulette wheel
  + It uses probability with the furthest first heuristic

**dimensionality reduction**

principal component analysis PCA, principal components

minimize data distance to line

maximize distance of projected points to origin

selecting components

\*visualization of first component, additional components

* PCA
  + Goal: produce dataset with high dimensionality (> 10k) to low dimensionality (< 100) while retaining the important characteristics of the data
  + Ex: data that has measurements in 2d (inches and cm, will have very high corr) can be reduced to 1d and can be represented as “length”
  + Adjust the points from higher to lower dimensions by projection
  + When adjusting from n->k, need to find k vectors u\_1, …, u\_k
* Vocab:
  + Eigenvector: 1 unit long version of the vector (also called PC or principal component)
  + Loading scores: are proportions of the original features in this new vector (also called PC or principal component)
  + Eigenvalue: Sum of Squares (SS) distances for the best fit line
  + Singular Value: square root of the eigenvalue
* Variance of PC1 (a vector) = SS(distances for PC1) / n-1
  + This is variation around the origin
  + Draw on this to determine how far we want to reduce the dimensionality
* Steps to find PC’s:
  + Center the data around the origin
  + Find best fitting line that goes through the origin (becomes PC1)
    - Best fitting means highest variance using SS distances
  + Find the next best fitting line that goes through the origin, and is perpendicular to PC1 (this becomes PC2)
  + Find the next best fitting line that goes through the origin, and is perpendicular to both PC1 and PC2 (this becomes PC3)
  + Repeat until desired dimensionality reduction
  + 🡪 # of PC’s
    - In Theory – one PC per dimension
    - In reality - # of PC is min (# dimensions, # datapoints)
* Selecting Components
  + Choose the new k to be the smallest value such that some amount (90%) of variance is retained
  + Or elbow method

**overall**

remember highlights of algorithms discussed throughout semester

NBC, linear regression, logistic regression, decision tree, knn

random forest, neural network, k means, boosting, SVM

\*compare and contrast methods

Learning rate

* **Q.** Learning rate is a crucial point in some machine learning algorithm. Please identified learning rate for following graphs, whether it is using a high learning rate or not.
  + A.一張含有 遊戲 的圖片

    自動產生的描述 Too high.

一張含有 差異, 空氣 的圖片

自動產生的描述 Too Low

* Q: We’re creating a training model to predict the winning numbers for the lottery. Using dates and their corresponding winning numbers, we train our model.

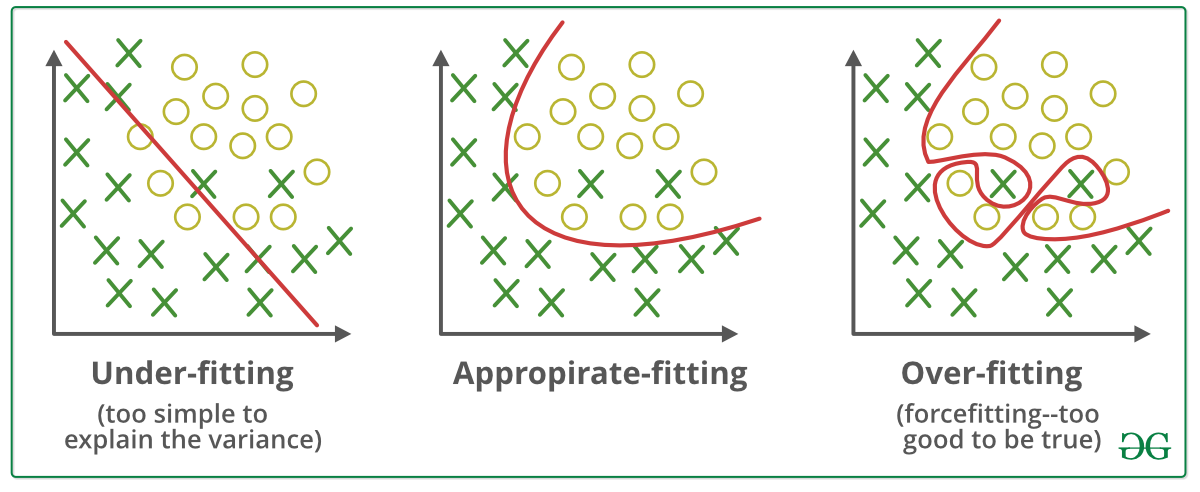
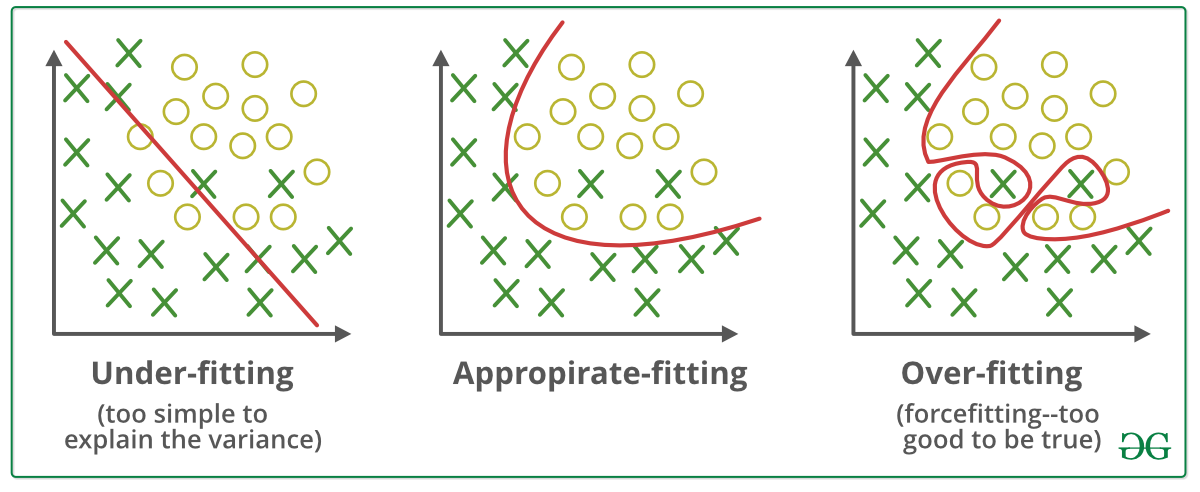
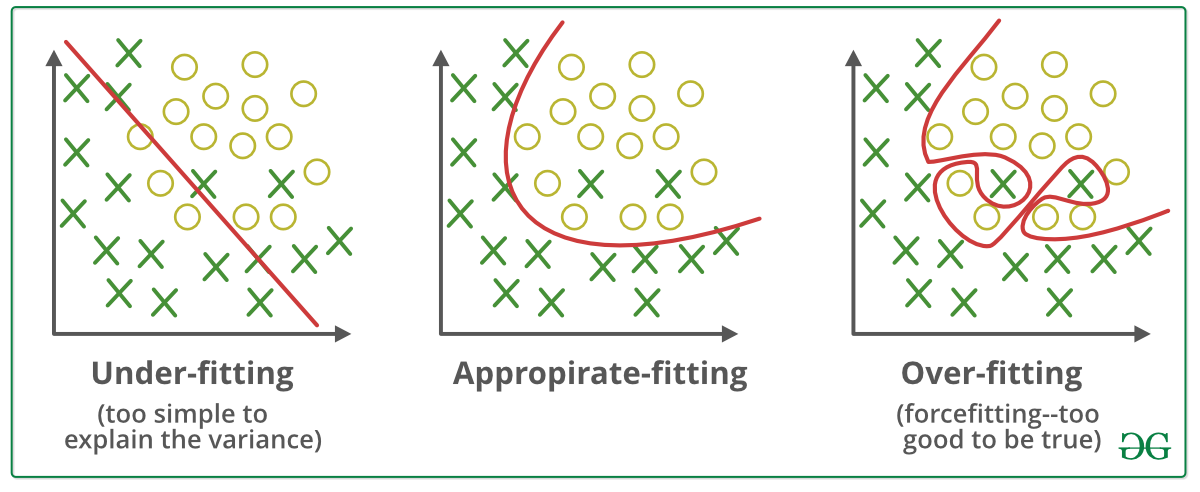
Will our model predict accurate winning numbers? Why or why not?

* + A: **No. While the model would do very well on past dates, it won’t know what to do on future dates. This is like memorizing the answers to a practice exam and expecting to get 100% on the real exam.**

Is this an example of underfitting, overfitting, or neither? Explain your answer

**Overfitting, since the model will perform well on training data but not on any unseen data**

Label each graph below as appropriate-fitting, overfitting, and underfitting:

 **APPROPRIATE UNDER OVER**

* Q: Random Forests are efficient with time series data.

    T

    F

* + Answer: F. It is difficult to maintain the order and continuity of the data while simultaneously avoiding a look-ahead bias while working with time series data and tree-based ensembles. It requires more fine-tuning by the user, so it’s recommended to use a different algorithm when handling this type of data.
* Q: **What form of graph does a neural network take on? ( I think this is an important question to reinforce CS knowledge!)**
* Directed acyclic graph (every edge is directed from earlier to later stage of the sequence)
* A close up of a map

  Description automatically generated