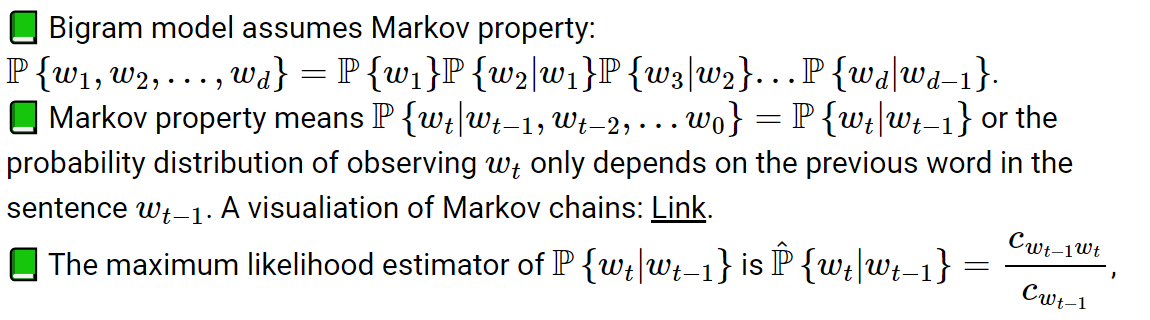
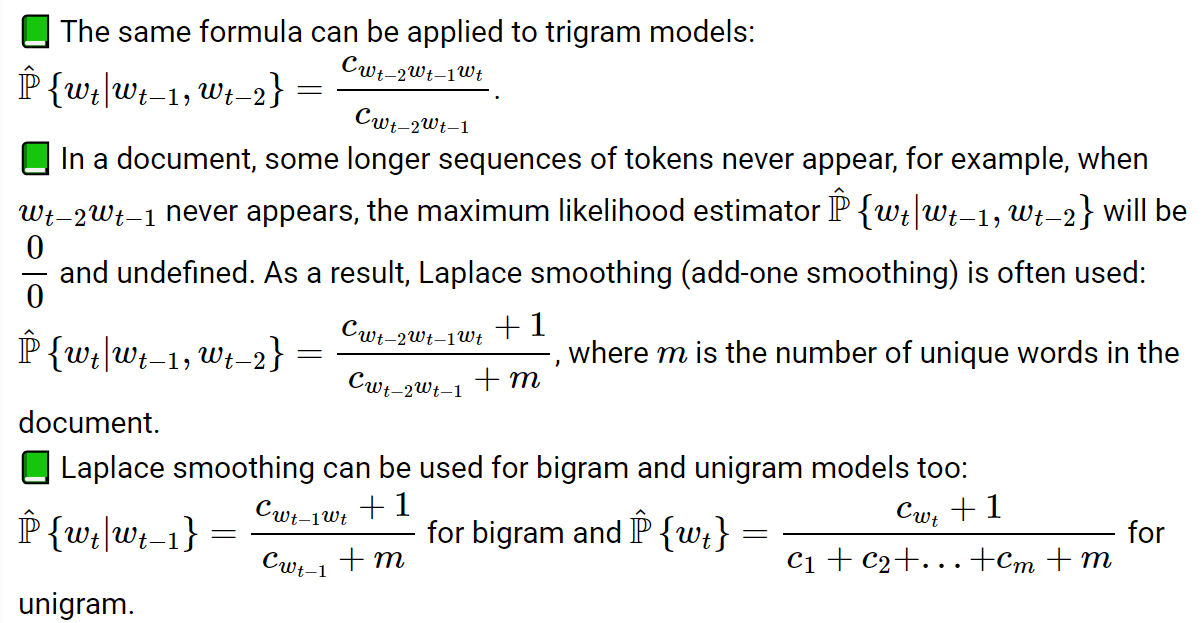
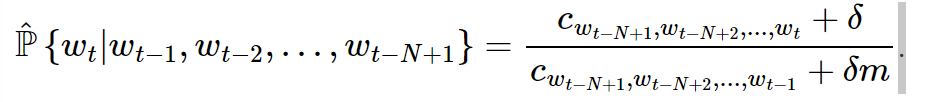
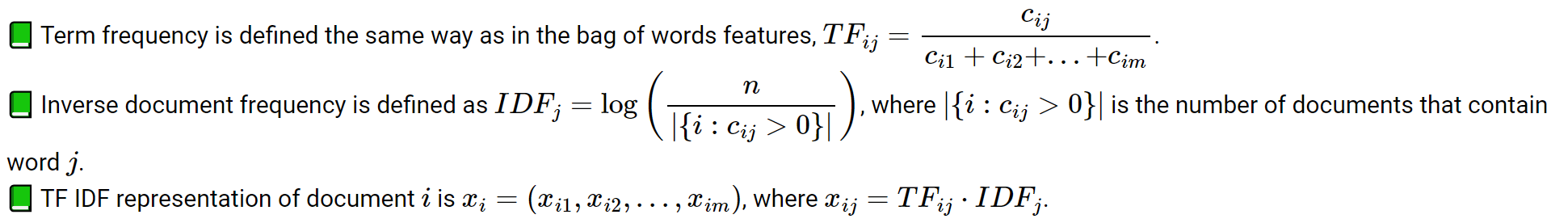
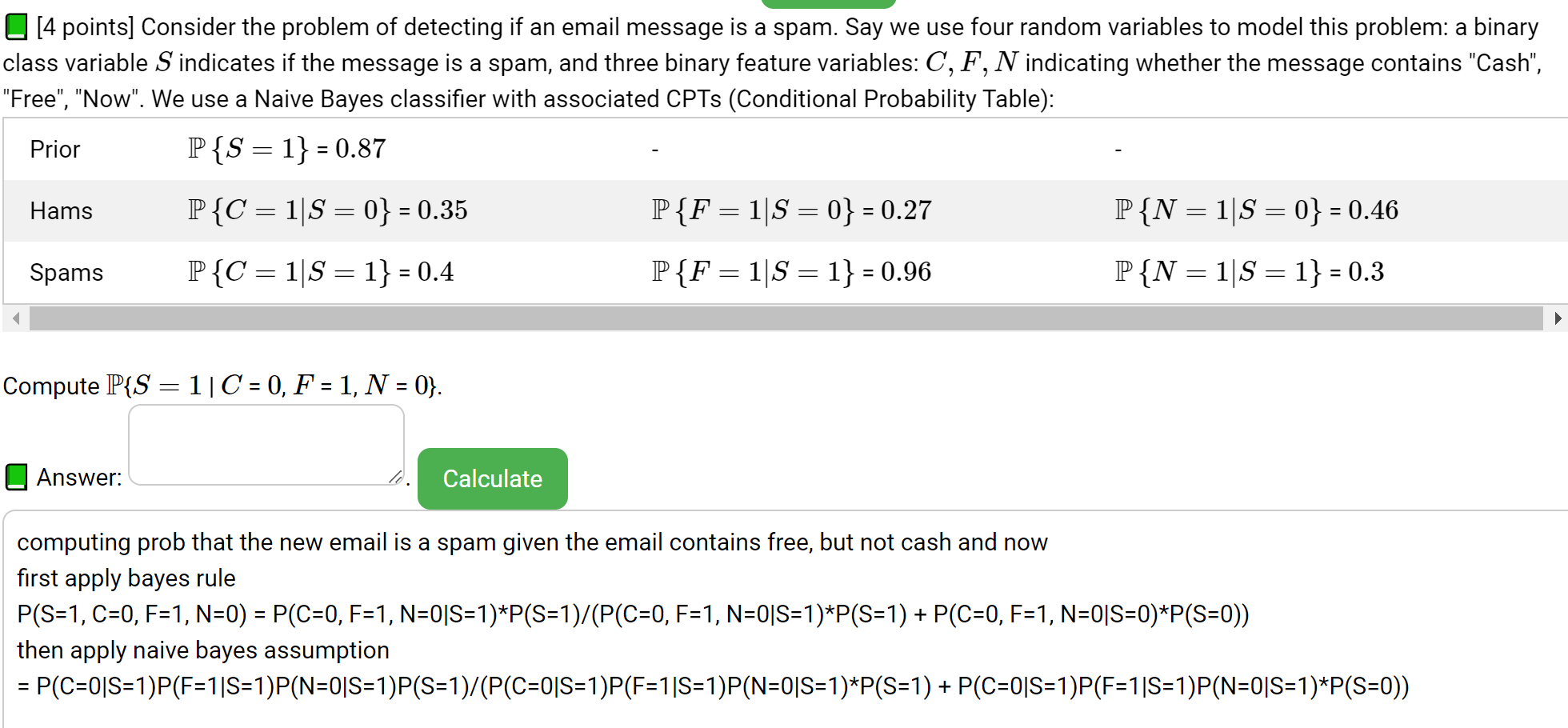
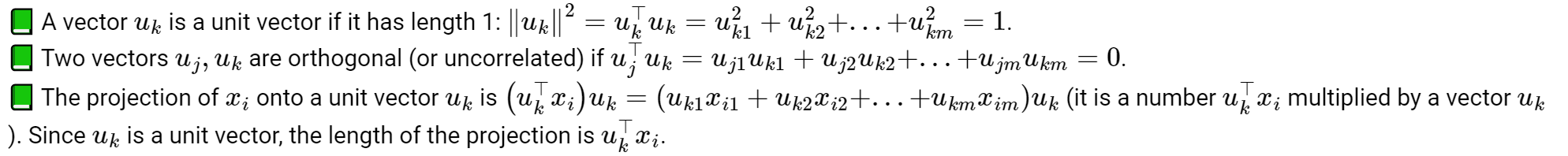
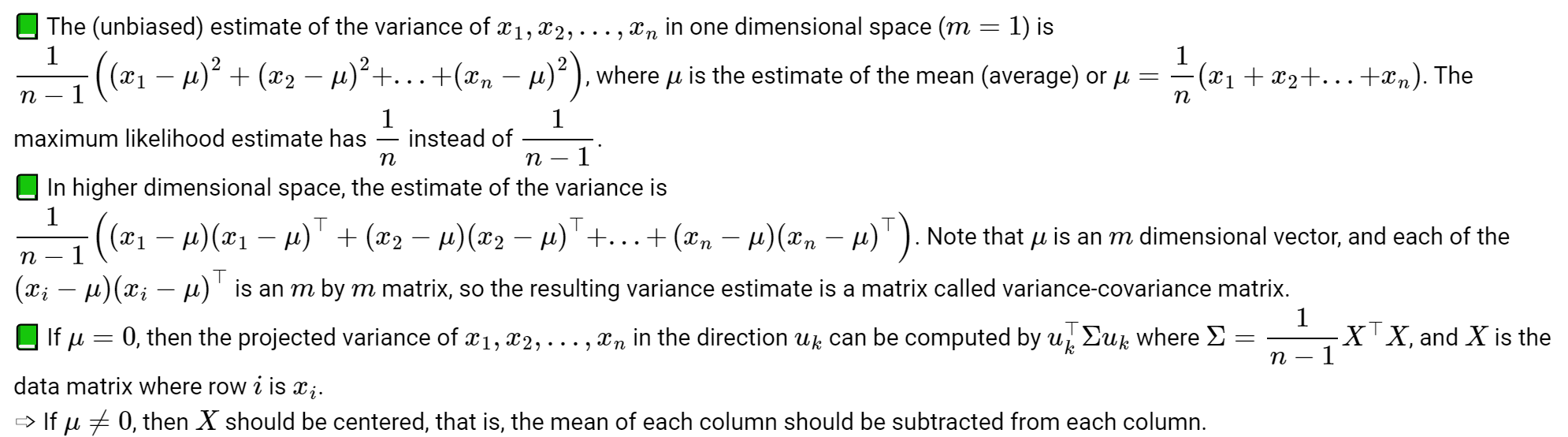
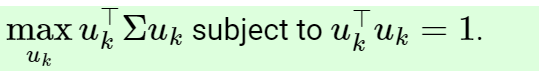
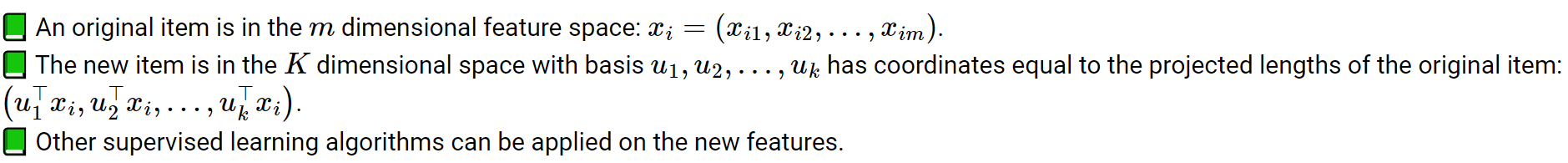
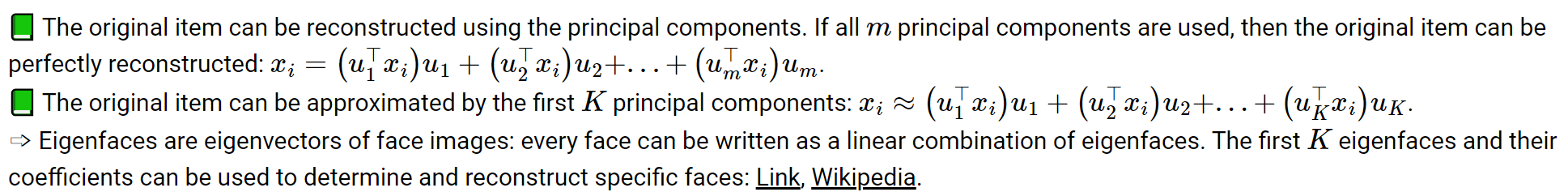
9/5

* N-gram model
  + A sentence is a sequence of words (tokens). Each unique word token is called a word type. The set of word types is called the vocabulary.
  + Assumes that the word at i is dependent of the n previous words
* Unigram model
  + Distribution of each word is not depend on any previous words
  + A unigram model generate the likelihood for each word in the vocabulary, choose the one with the maximum likelihood
  + Prob of observing a sequence is P1 \* P2 \* P3
* Bigram model
  + 
  + P(wt | wt-1) = # of time wt-1wt appears / number of time wt-1 appears
* Maximum likelihood estimation
  + Given a training set, P is estimated by P(wt) = cwt/(c1 + c2 + … + cm)
    - Number of times a word appear in the training set / total words in the training set
* Transition matrix
  + The bigram probabilities can be stored in a transition matrix of markov chain
* Trigram model
  + 
  + Laplace smoothing
    - advantages:
      * deal with 0/0 cases
      * Able to generate new phrases
    - General laplace smoothing
      * 
      * As delta increase, more likely to generate new phrases

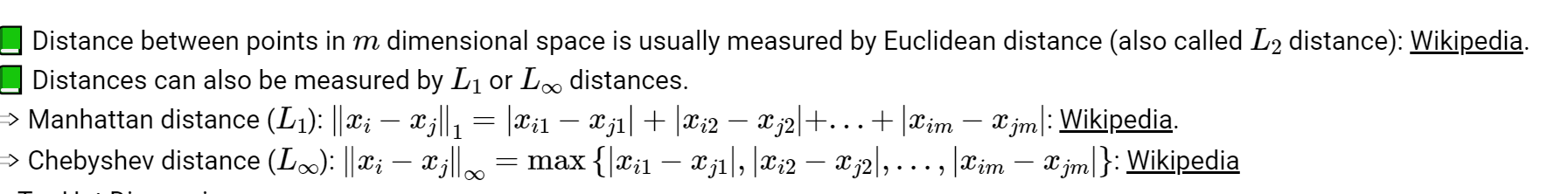
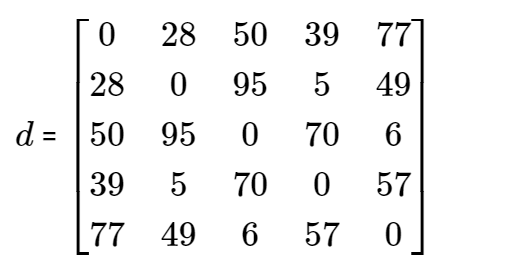
9/10

* Natural language processing
  + When processing language data, documents need to be first turned into sequences of word tokens
  + Stemming (lemmatization) - looks, looked, looking → look
  + Tokenization
    - Bag of words feature
      * One numerical vector for each document
      * Length = size of the vocabulary
      * xj = xj/(x1+...+xm)
        + x is the # appearance of a word in the document
    - TF IDF features
      * Term frequency - inverse document frequency
      * 
      * 
* Supervised learning examples
  + Given emails, predict whether they are spam or ham
  + Given comments, oredict whether they are offensive or not
  + Given reviews, predict whether they are positive/negative
* Discriminative vs generative models
* Conditional prob
  + P(B|A) = P(A, B)/P(A)
* Joint prob
  + P(A, B)
* Marginal prob
  + P(A) = P(x=1, y=1) + P(x=1, y=0)
* Bayes rule
  + P(A|B) = P(B|A)\*P(A)/P(B)
  + P(B|A) = P(A, B)/P(A)
  + Prior probability
    - P(A) - the probability of a specific label
  + Likelihood
    - P(B|A) - chance for the
* Naive bayes classifier
  + Simple Bayesian network that assumes the features are independent
  + The key assumption is the independence assumption
    - 
* Multinomial naive bayes
  + 
* In this class, log = loge
* Other naive bayes models
  + Multinomial naive bayes
    - Assume that follow multinomial distribution
  + Gaussian naive bayes
    - If the features are continuous, assume the features follow gaussian distribution
  + Bayesian network
    - A network of relationship between features, so they do not depend only on labels

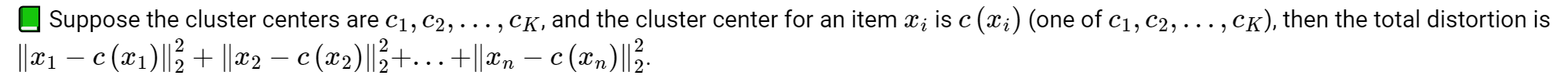
9/12

* For this course, use 0.5 as the classification threshold
* Unsupervised learning
  + Data are not labeled
  + Clustering - separates items into groups
  + Novelty(outlier) detection - finds items that are different (two groups)
  + Dimensionality reduction - represents each item by a lower dimensional feature vector while maintaining key characteristics
  + Applications
    - Google news
    - Image segmentation
    - Text processing
    - Data visualization
* Principal component analysis
  + - math
    - 
    - 
  + Find the direction that maximize the projected variance
    - 
  + Pick the largest eigenvalue and the corresponding eigenvector
  + How to pick the principle dimension K
    - Number of non-zero eigenvalues
    - Selected based on prior knowledge
    - Number of eigenvalues larger than some threshold
  + Reduced feature space
    - Image the original data into the new vector space defined by the remaining eigenvectors
    - 
    - u are unit vectors
  + Reconstruction
    - 

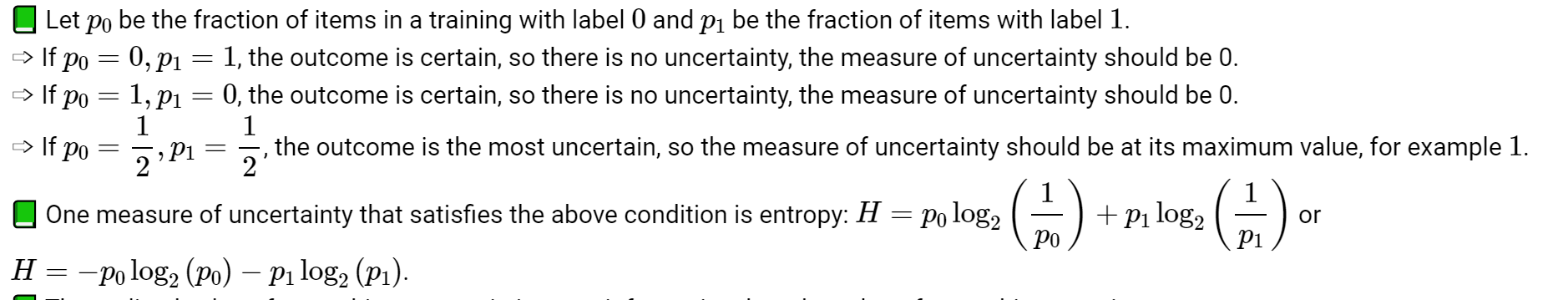
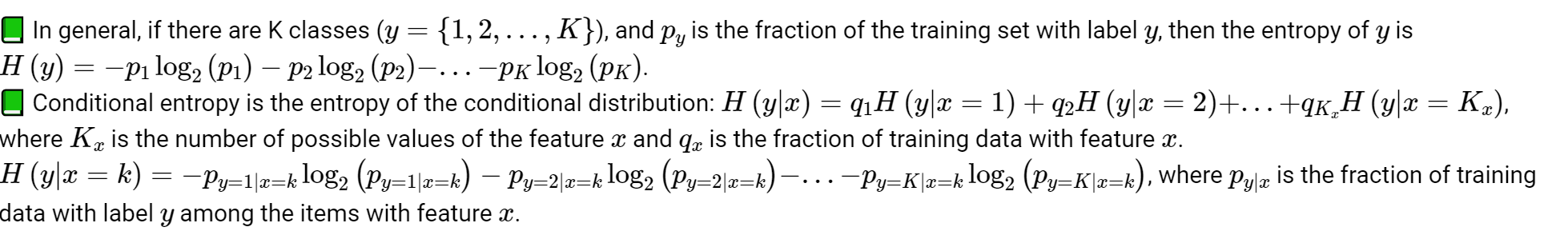
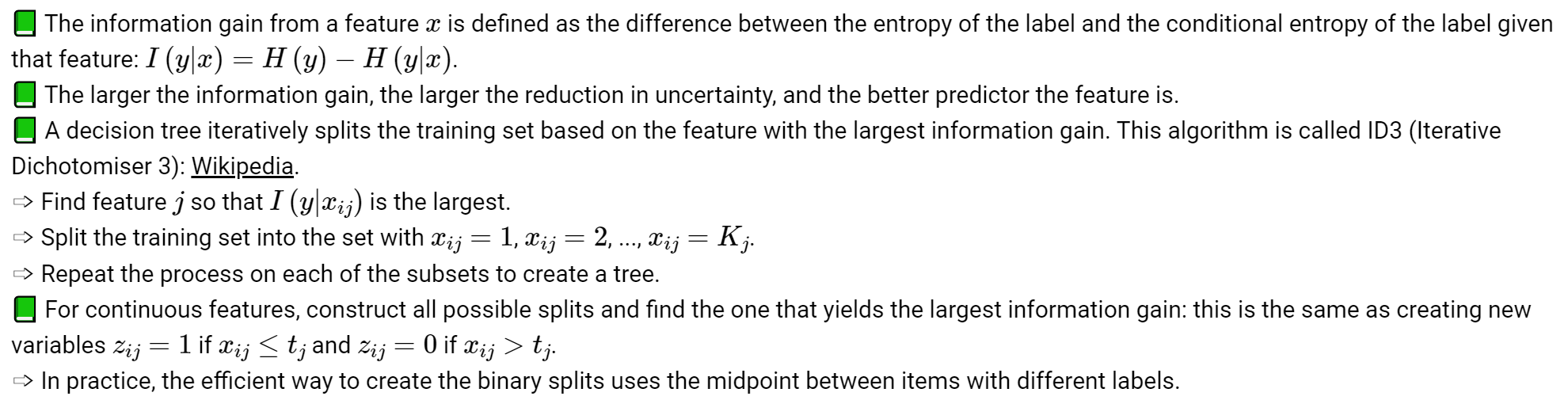
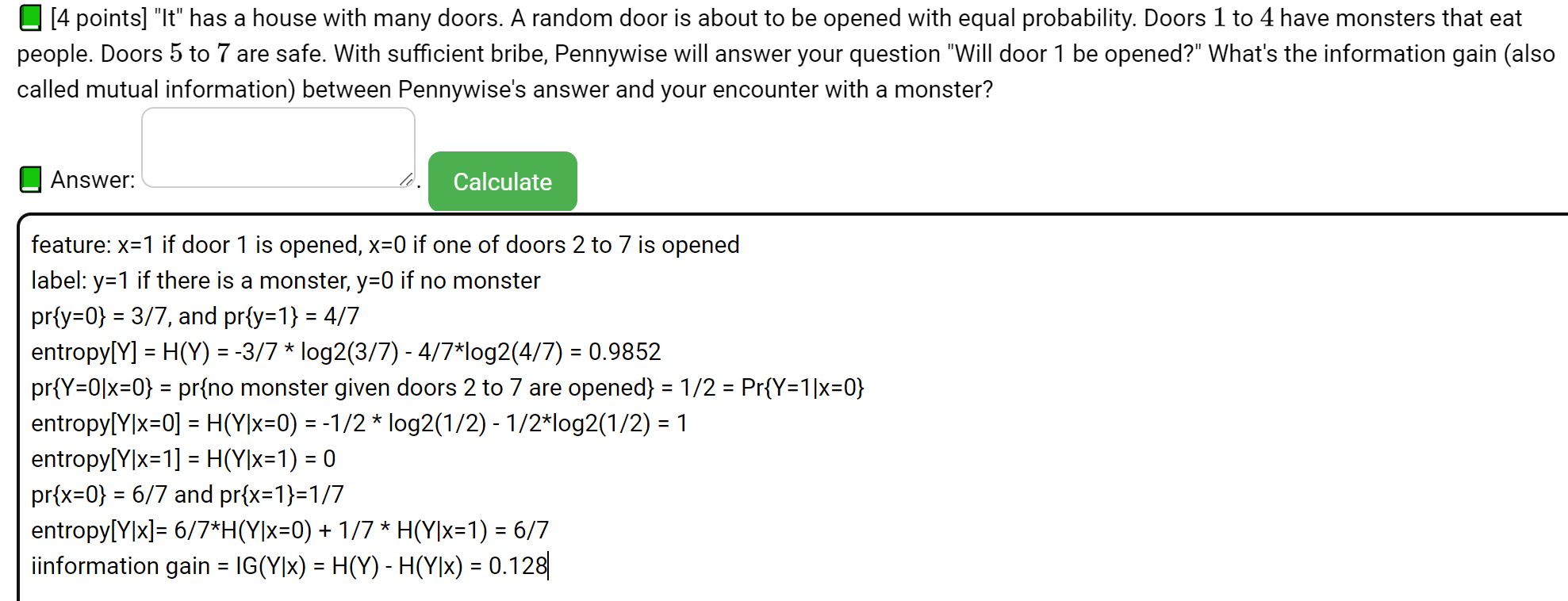
9/17

* Hierarchical clustering
  + Items of similar feature belongs to the same group
  + Every point start as its own clustering, drag one point to another point to merge them into one cluster
* Distance between points
  + 
  + Single linkage distance
    - Shortest distance from any item in one cluster to any item in another cluster
  + Complete linkage distance
    - the longest distance from any item in one cluster to any item in the other cluster
  + Average linkage distance
    - the average distance from any item in one cluster to any item in the other cluster (average of distances, not distance between averages)
  + Cluster by distance matrix
    - 
    - Step1 - find the two points closest to each other - point2 and point4
    - Step2 - merge {2, 4} into one cluster, recompute the pairwise distance table
    - Then repeat the algorithm
* Choosing number of clusters
  + The number of clusters should be chosen based on prior knowledge about the dataset.
  + The algorithm can also stop merging as soon as all the between-cluster distances are larger than some fixed threshold.
  + The binary tree generated by hierarchical clustering is often called dendrogram

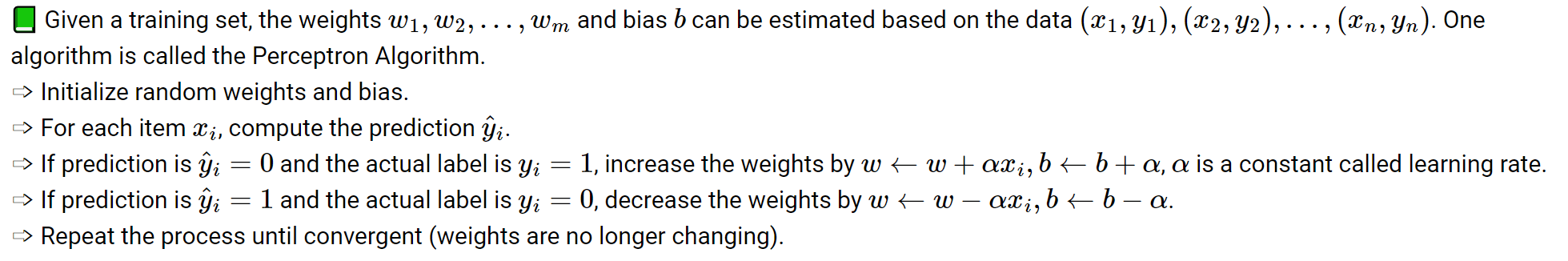
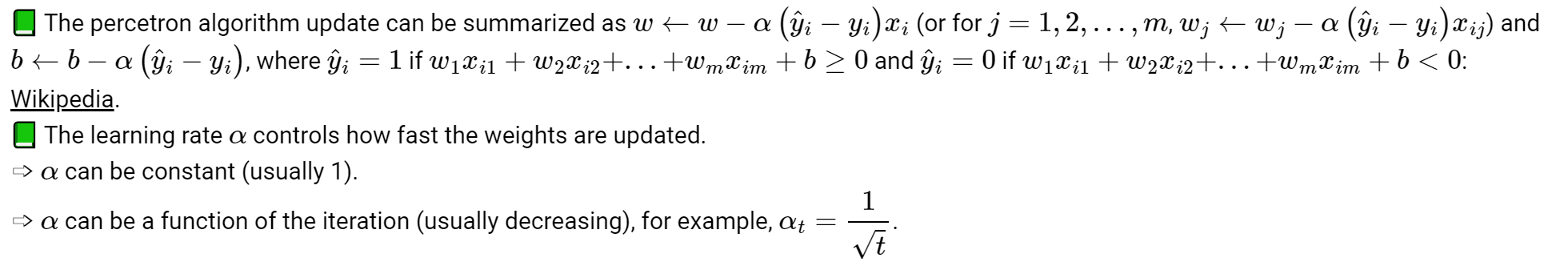
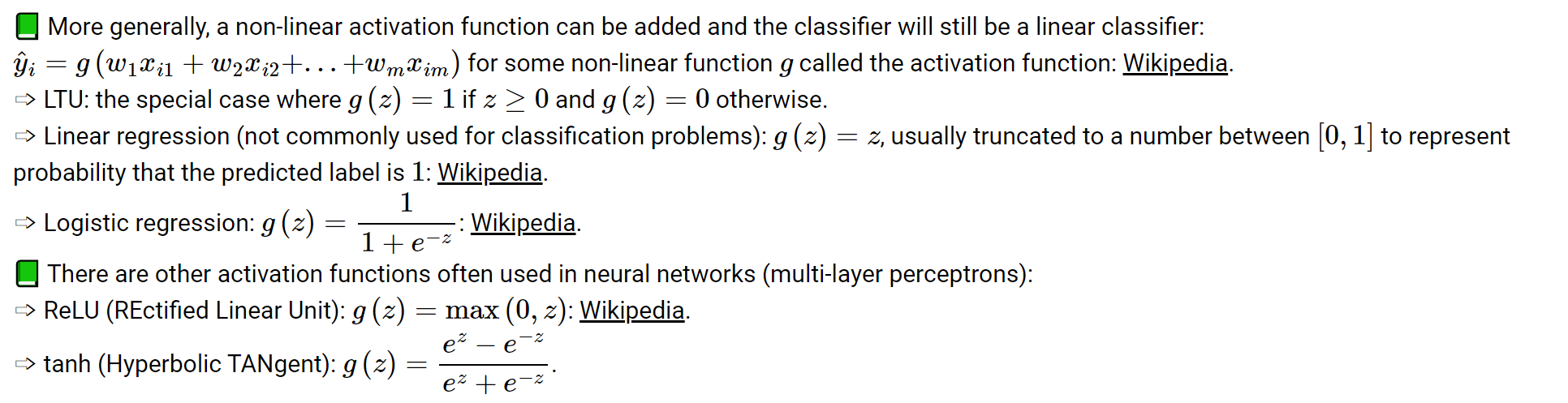
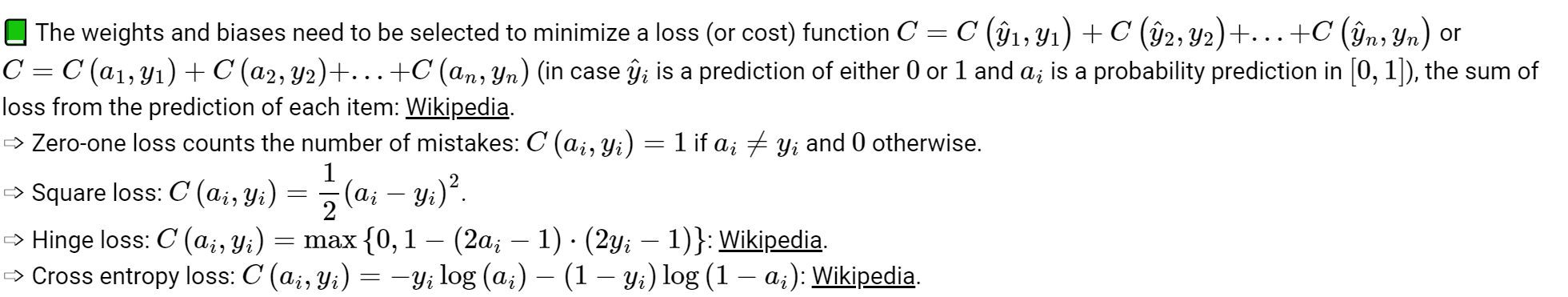
9/19

* K means clustering
  + Start with a fixed k
  + Start from k random cluster centers
  + Compare distance from each point to the cluster centers, assign cluster based on the shortest distance
  + Relocate the centers to the actual center of the clusters
  + Re-iterate the algorithm until it converges
* Total distortion
  + K means clustering try to minimize total distortion
  + 
  + Minimizing total distortion is similar to gradient descent
* Number of clusters
  + Based on prior knowledge
  + Cannot be chosen by minimizing total distortion since the total distortion is always minimized at 0 when k=n (number of clusters = number of training items)
  + K can be chosen by minimizing total distortion plus some regularizer, for e.g., c\*mKlog(n) where c is a fixed constant and m is the number of features for each item
    - C large if lower k
    - C small if higher k
* Initial clusters
  + The initial cluster centers can be randomly chosen in the domain.
  + The initial cluster centers can be randomly chosen as K distinct items.
  + The first cluster center can be a random item, the second cluster center can be the item that is the farthest from the first item, the third cluster center can be the item that is the farthest from the first two items, …
* K nearest neighbor
  + The K Nearest Neighbor algorithm (not related to K Means) is a simple supervised learning algorithm that uses the items from the training set that is the closest to a new item to predict the label of the new item
  + 1 nearest neighbor copies the label of the closest item.
  + 3 nearest neighbors find the majority label of the three closest items.
  + N nearest neighbor uses the majority label of the training set (of size N) to predict the label of every new item
* Training set accuracy
  + For 1NN, the accuracy of the prediction on the training set is always 100 percent.
  + When comparing the accuracy of KNN for different values of K (called hyperparameter tuning), training set accuracy is not a great measure.
  + K fold cross validation is often used instead to measure the performance of a supervised learning algorithm on the training set.
    - The training set is divided into K groups (K can be different from the K in KNN).
    - Train the model on K - 1 groups and compute the accuracy on the remaining 1 group.
    - Repeat this process K times.
  + K fold cross validation with K=n is called Leave One Out Cross Validation (LOOCV).

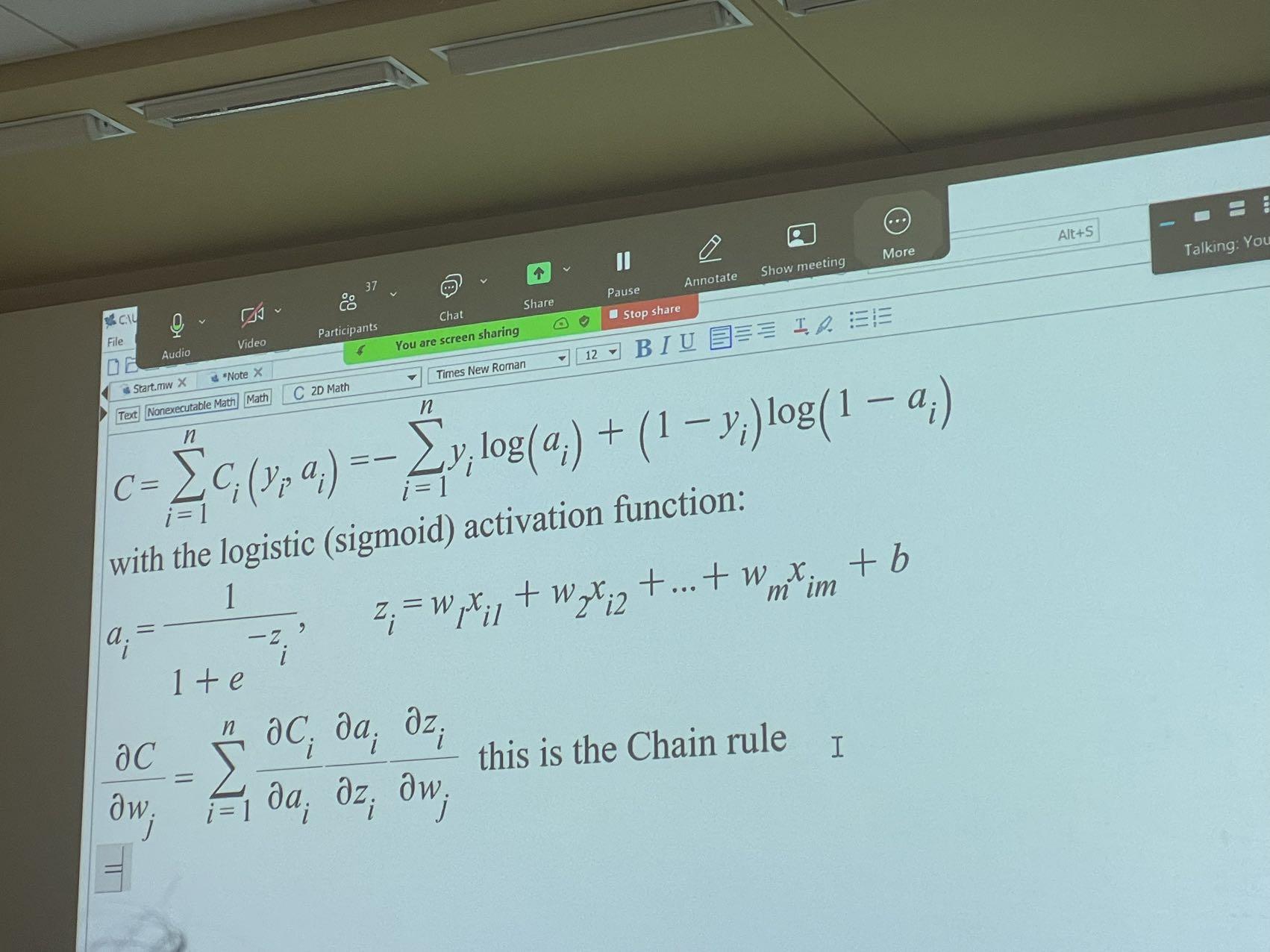
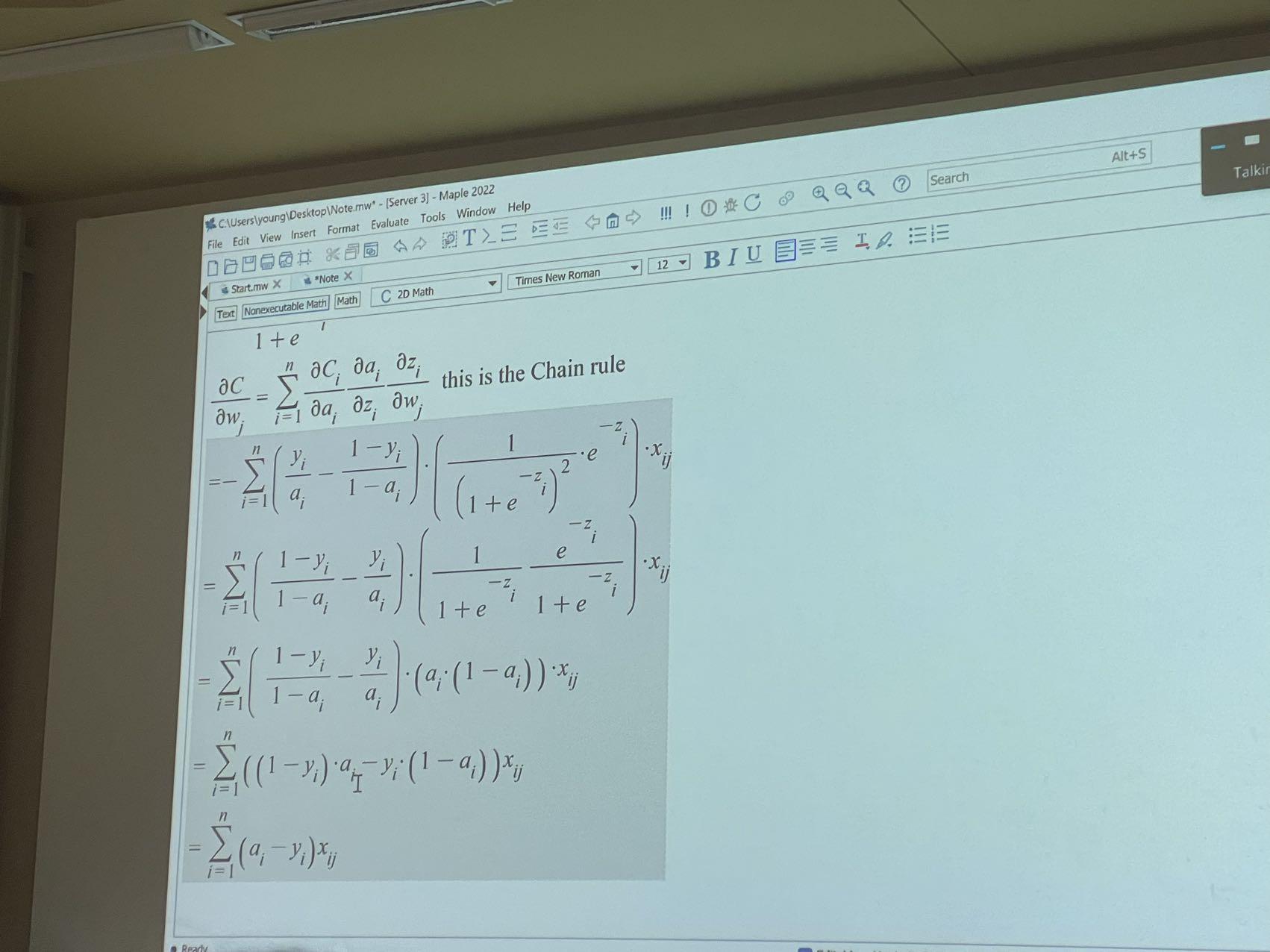
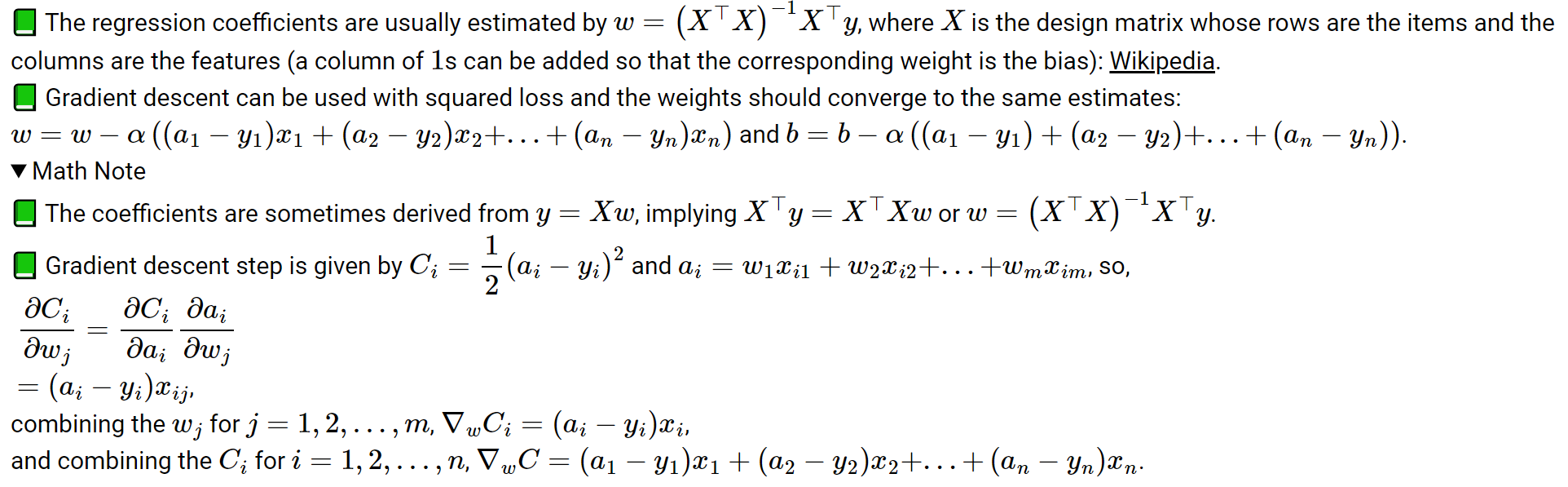
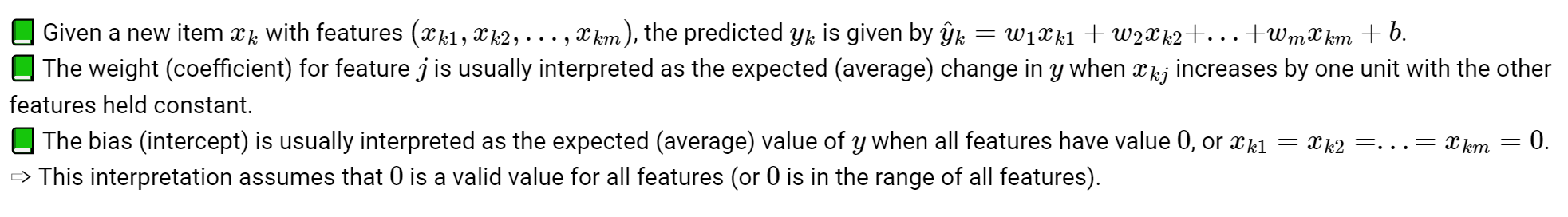
9/24

* Decision tree
  + We want to make the tree in a way to minimize the uncertainty
  + We want to maximize information gain
  + Find the feature that is most informative
  + Split the training set into subsets based on this feature
  + Repeat on each of the subset recursively until all features or labels in the subset are the same.
* Uncertainty
  + 
    - The entropy formula is the only place we use log2x
    - Max at 1
* Entropy
  + 
* Information gain
  + 
  + 
  + We assume log0 = 0
* Pruning
  + Decision trees can be pruned by replacing a subtree by a leaf when the accuracy on a validation set with the leaf is equal or higher than the accuracy with the subtree. This method is called Reduced Error Pruning
  + A validation set is a subset of the training set that is set aside when training the decision tree and only used for pruning the tree.
  + The items used to train the decision tree cannot be used to prune the tree.
* Random forest
  + Smaller training sets can be created by sampling from the complete training set, and different decision trees can be trained on these smaller training sets (and only using a subset of the features). This is called bagging (or Bootstrap AGGregatING)
    - Training items are sampled with replacement.
    - Features are sampled without replacement.
  + The label of a new item can be predicted based on the majority vote from the decision trees training on these smaller training sets. These trees form a random forest
* Adaptive boosting
  + Decision trees can also be trained sequentially. The items that are classified incorrectly by the previous trees are made more important when training the next decision tree.
  + Each training item has a weight representing how important they are when training each decision tree, and the weights can be updated based on the error made by the previous decision trees. This is called AdaBoost (ADAptive BOOSTing)

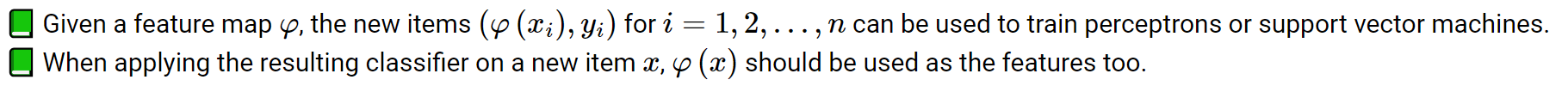
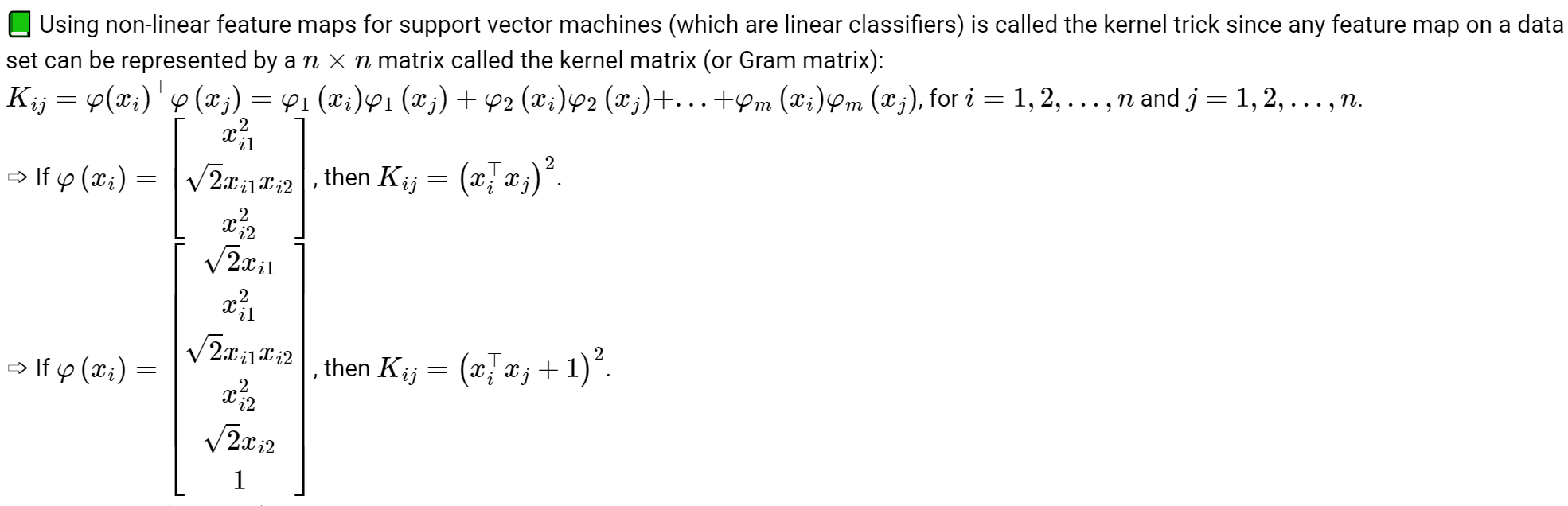
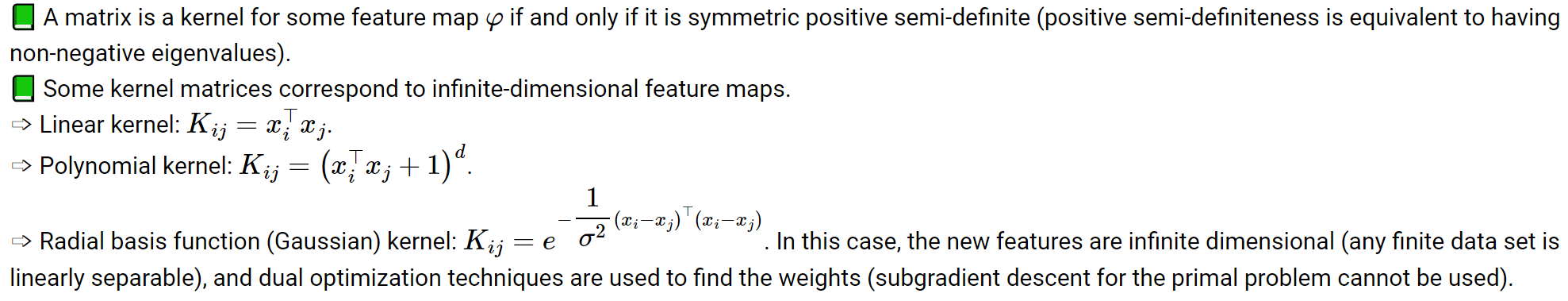
9/26

* Linear classifier
  + Linear threshold unit
    - W1xi1 + w2xi2 + … wmxim + b >=0
  + 
* Perceptron algorithm
  + 
* Activation function
  + 
* Loss function
  + 

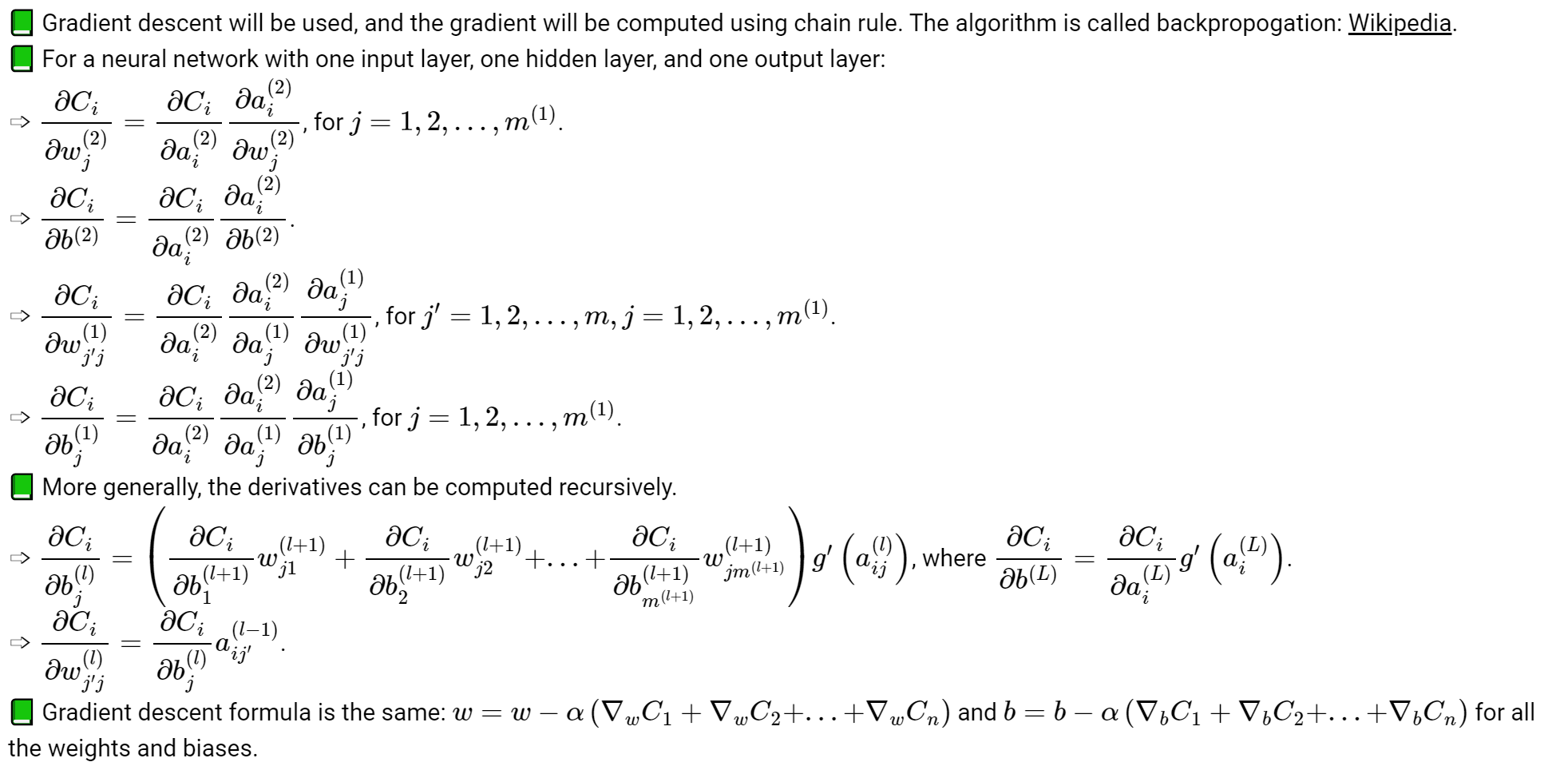
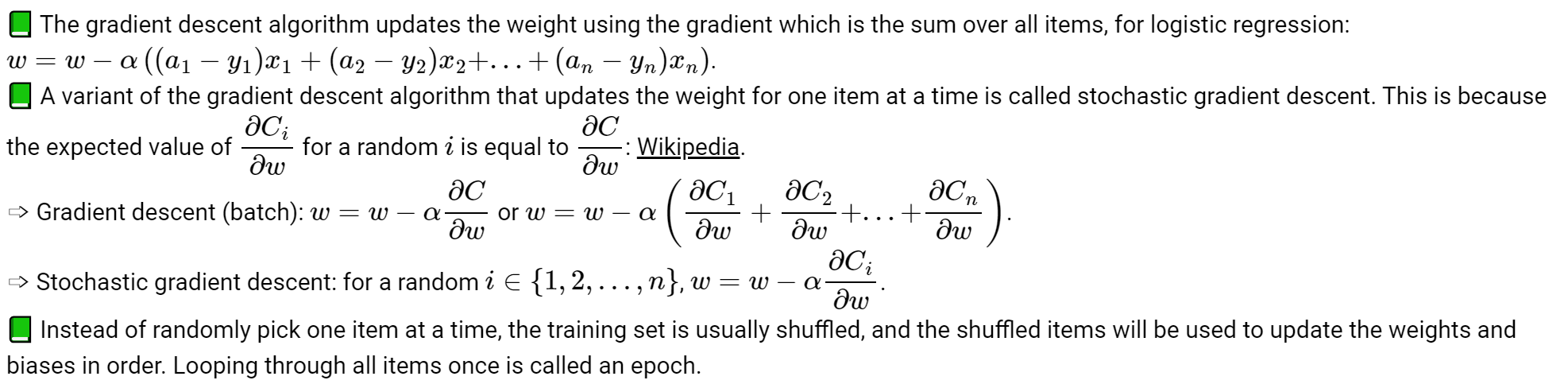
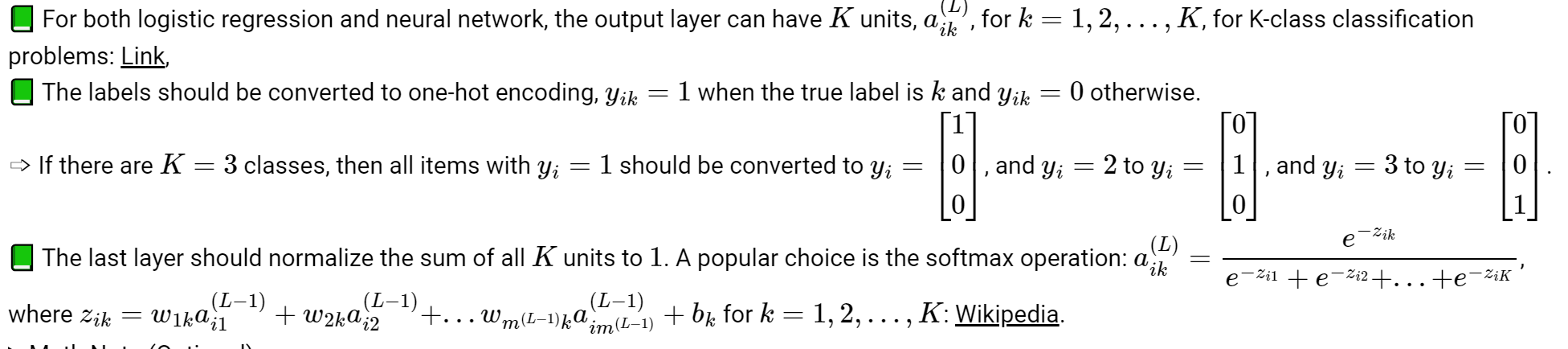
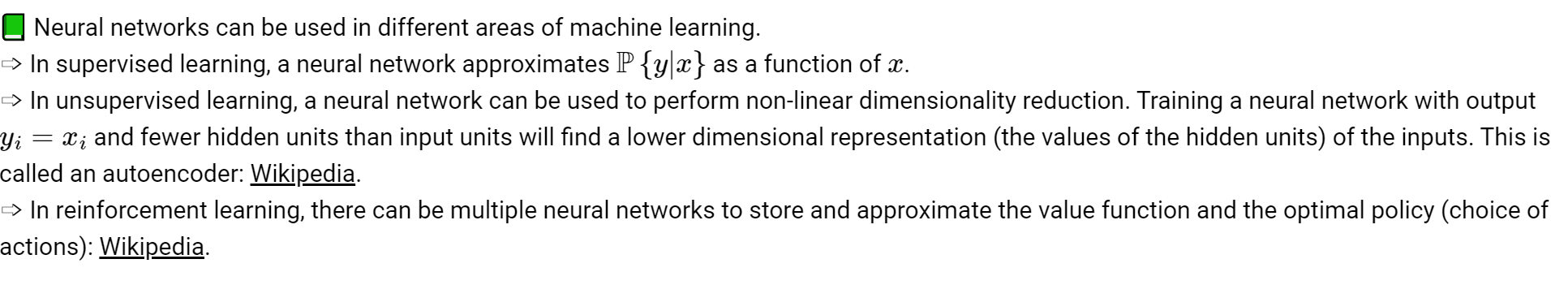
10/1

* Gradient descent example
  + 
  + 
* Linear regression
  + 
  + Do not use linear regression for classify problems
    - Loss function does not make sense
* Model interpretation
  + 
* Margin and support vectors
* Multi-class SVM
  + Train multiple SVMs
  + One vs one - (1/2)K(K - 1) classifiers
  + One vs all - K classifiers (treat one as a class and the rest as a class)

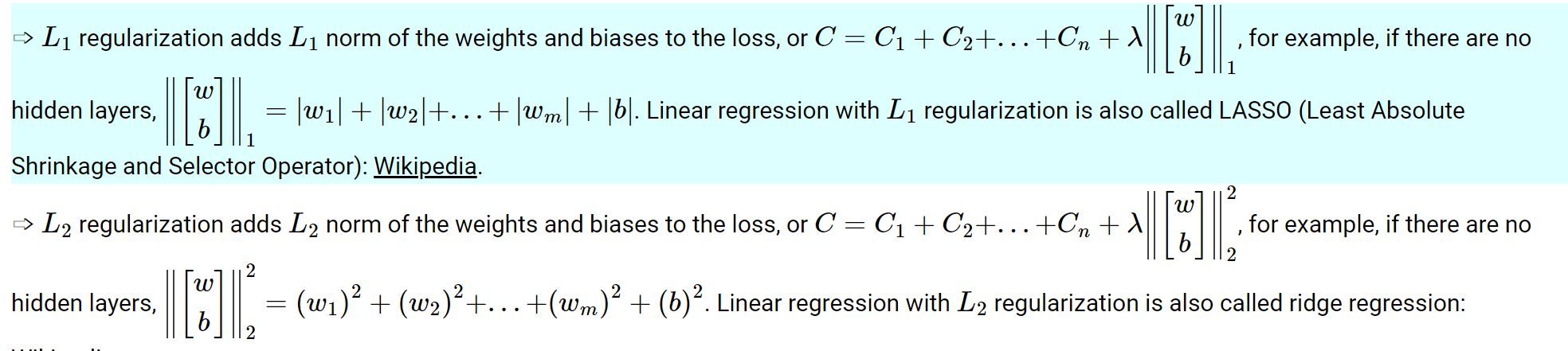
10/3

* Feature map
  + If the classes are not separable, more features can be created so that in higher dimensions the items might be linear separable
  + 
* Kernel trick
  + 
* Kernel matrix
  + 
* MLP
  + Multiple layers of logistic regressions
  + Output is non-linear in the original features
* Neuro network
  + A 2 layer network can approximate any continuous function arbitrarily closely with enough hidden units
  + A 3-layer network can approximate any function arbitrarily closely with enough hidden units

10/8

* Gradient descent
  + 
* Stochastic gradient descent
  + 
* Softmax layer
  + 
* Function approximator
  + 

10/10

* Generalization error
  + With a large number of hidden layers and units, a neural network can overfit a training set perfectly. This does not imply the performance on new items will be good
  + More data can be created for training using generative models or unsupervised learning techniques
  + A validation set can be used to train the network until the loss of the validation set begins to increase
  + Dropout
    - randomly omit units (random pruning of weights) during training so the rest of the units will have a better performance
* Regularization
  + A simpler model (with fewer weights, or many weights set to 0) is usually more general and would not overfit the training set as much. A way to achieve that is to include an additional cost for non-zero weights during training.
  + 
  + 