**1 - Supervised Classification**

**Notation. We consider the probabilistic and statistical framework of supervised classification where X is a random vector on R^d, d >= 1 and Y is a binary random variable with values in {-1, +1}. A random samples S\_n = {(X\_1, Y\_1), ..., (X\_n, Y\_n)} contains n independent copies of the pair (X, Y) of joint probability distribution P.**

1. **Define a classifier.**

A classifier is a function that takes an input vector (or instance) x and assigns it to one of several predefined categories or classes, based on a set of features or attributes associated with the input vector. In the context of supervised learning, a classifier is a model that learns to map input vectors to their corresponding classes based on a labeled training dataset. The output of a classifier is typically a predicted class label or a probability distribution over the class labels.

1. **Recall the definition of risk.**

the risk of a classifier f, denoted as R(f), is the expected value of a loss function L(Y, f(X)) over the joint distribution of the input-output pair (X, Y):

R(f) = E[L(Y, f(X))]

1. **Define the problem of supervised binary classification (in ideal conditions) using the definition of risk**

Supervised binary classification is the task of learning a function that maps an input vector to one of two possible classes, based on labeled training data. The risk of a classifier is the expected error of the classifier over the joint distribution of input-output pairs. The goal of learning is to find a classifier that minimizes this risk.

1. **Define the empirical risk of a classifier calculated using S\_n. Explain the principle of empirical risk minimization.**

The empirical risk of a classifier is the average loss of the classifier over the training set. The principle of empirical risk minimization suggests choosing the classifier that minimizes the empirical risk, with the assumption that the training set is representative of the true underlying distribution of input-output pairs. However, this assumption may not always hold.

1. **Why empirical risk minimization may rise issues? Which approach to propose to address this issue?**

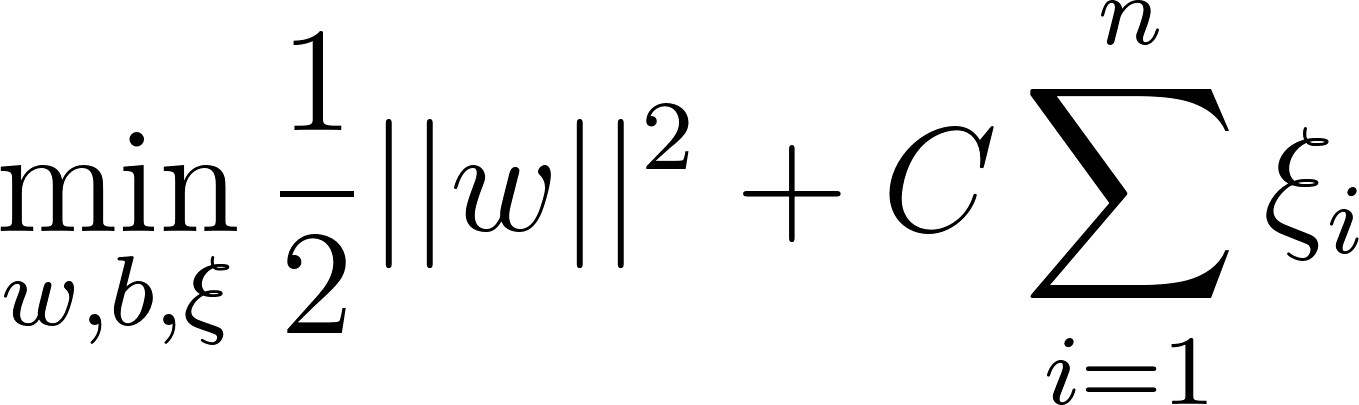
ERM may give rise to issues because it tries to minimize the empirical risk, which is the average loss over the training set. However, the empirical risk may not always reflect the true underlying pattern in the data or the true generalization error of the model. These issues can be addressed by using techniques such as regularization, choosing more flexible model classes, using different optimization algorithms or initialization strategies, using robust optimization or label smoothing, and using techniques such as domain adaptation or transfer learning.

1. **What is overfitting? What is the underlying idea of methods designed to avoid it?**

Overfitting occurs when a model fits the noise or idiosyncrasies of the training data, rather than the underlying pattern that generalizes to new data. As a result, an overfitted model is likely to have high variance and low bias, which means that it will perform well on the training data but poorly on new, unseen data. The underlying idea of methods designed to avoid overfitting is to control the complexity of the model by using a simpler model or to use additional techniques such as regularization, early stopping, or data augmentation to prevent the model from fitting the noise in the data.

**2. Support Vector Machines**

1. **What optimization problem do we need to solve in the primal space to find the optimal margin hyperplane, e.g. a linear SVM, when data are noisy?**

[****](https://www.codecogs.com/eqnedit.php?latex=%20%5Cmin_%7Bw%2Cb%2C%5Cxi%7D%20%5Cfrac%7B1%7D%7B2%7D%20%7C%7Cw%7C%7C%5E2%20%2B%20C%20%5Csum_%7Bi%3D1%7D%5E%7Bn%7D%20%5Cxi_i%20#0)

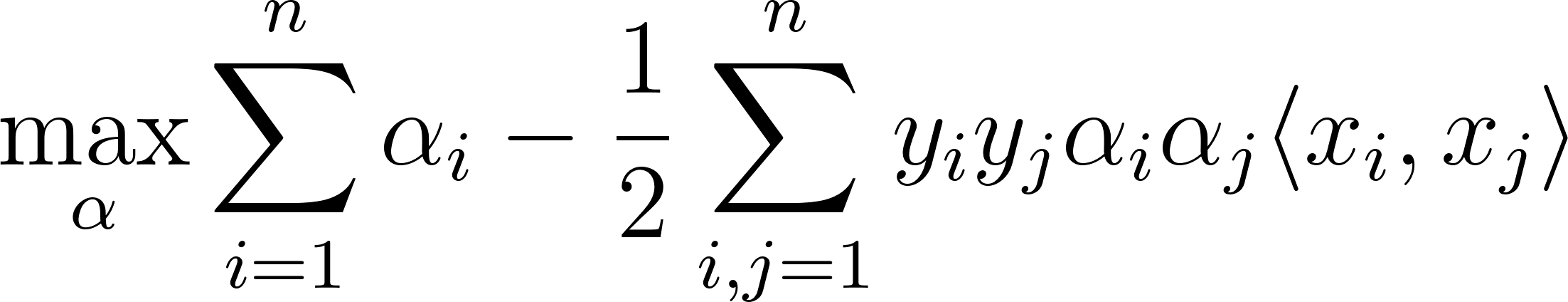
**subject to the constraints:**

[****](https://www.codecogs.com/eqnedit.php?latex=%20y_i(w%20%5Ccdot%20x_i%20%2B%20b)%20%5Cgeq%201%20-%20%5Cxi_i%2C%20%5Cquad%20%5Cxi_i%20%5Cgeq%200%2C%20%5Cquad%20i%3D1%2C2%2C%5Cdots%2Cn%20#0)

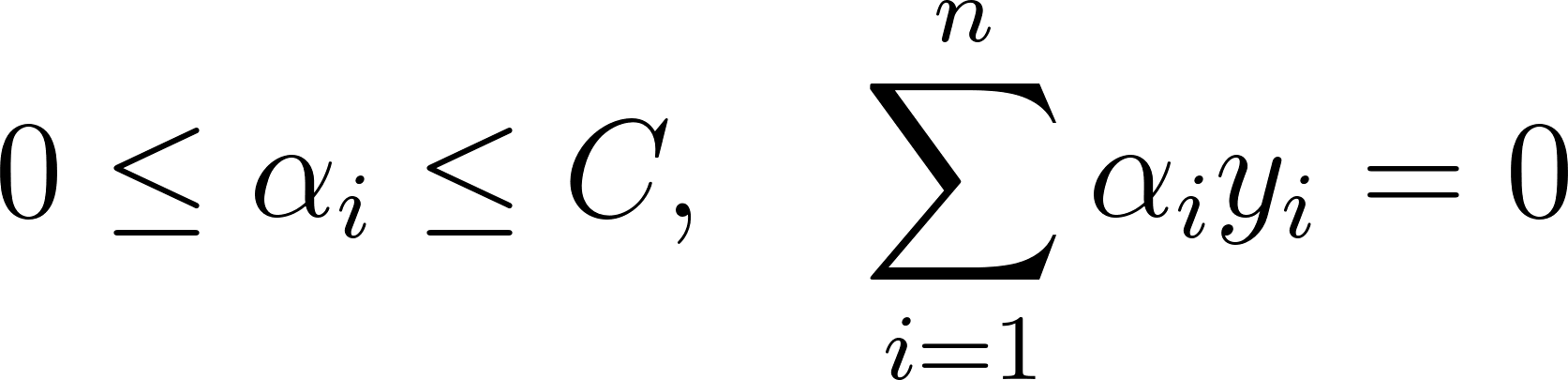
where $w$ is the weight vector, $b$ is the bias term, $C$ is a regularization parameter, $n$ is the number of training examples, $x\_i$ is the feature vector of the $i$th training example, $y\_i$ is the corresponding label in ${-1, +1}$, and $\xi\_i$ is a non-negative slack variable that allows for misclassification of noisy examples.

1. **Write the dual formulation of this problem.**

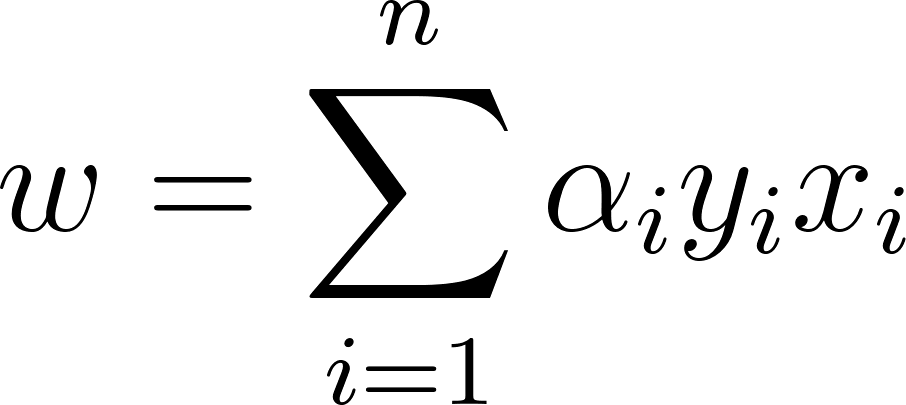
The dual formulation of the optimization problem for finding the optimal margin hyperplane with noisy data is obtained by applying the Lagrange duality method to the primal problem, and it is given by:

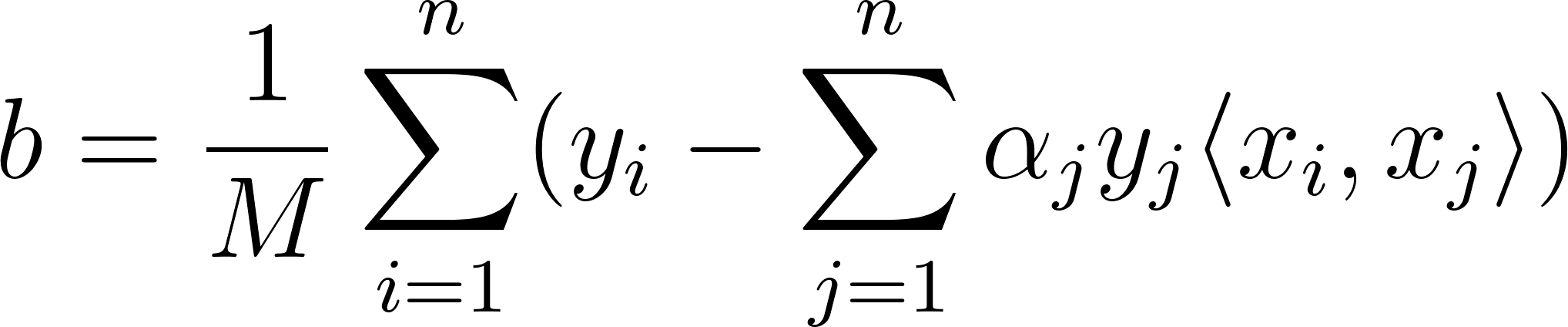
[](https://www.codecogs.com/eqnedit.php?latex=%20%5Cmax_%7B%5Calpha%7D%20%5Csum_%7Bi%3D1%7D%5E%7Bn%7D%20%5Calpha_i%20-%20%5Cfrac%7B1%7D%7B2%7D%20%5Csum_%7Bi%2Cj%3D1%7D%5E%7Bn%7D%20y_i%20y_j%20%5Calpha_i%20%5Calpha_j%20%5Clangle%20x_i%2C%20x_j%20%5Crangle%20#0)

subject to the constraints:

[](https://www.codecogs.com/eqnedit.php?latex=%200%20%5Cleq%20%5Calpha_i%20%5Cleq%20C%2C%20%5Cquad%20%5Csum_%7Bi%3D1%7D%5E%7Bn%7D%20%5Calpha_i%20y_i%20%3D%200%20#0)

where $\alpha$ is a vector of Lagrange multipliers, and $\langle x\_i, x\_j \rangle$ is the inner product of the feature vectors $x\_i$ and $x\_j$. The weight vector $w$ and the bias term $b$ of the hyperplane can be expressed in terms of the Lagrange multipliers as:

[](https://www.codecogs.com/eqnedit.php?latex=%20w%20%3D%20%5Csum_%7Bi%3D1%7D%5E%7Bn%7D%20%5Calpha_i%20y_i%20x_i%20#0)

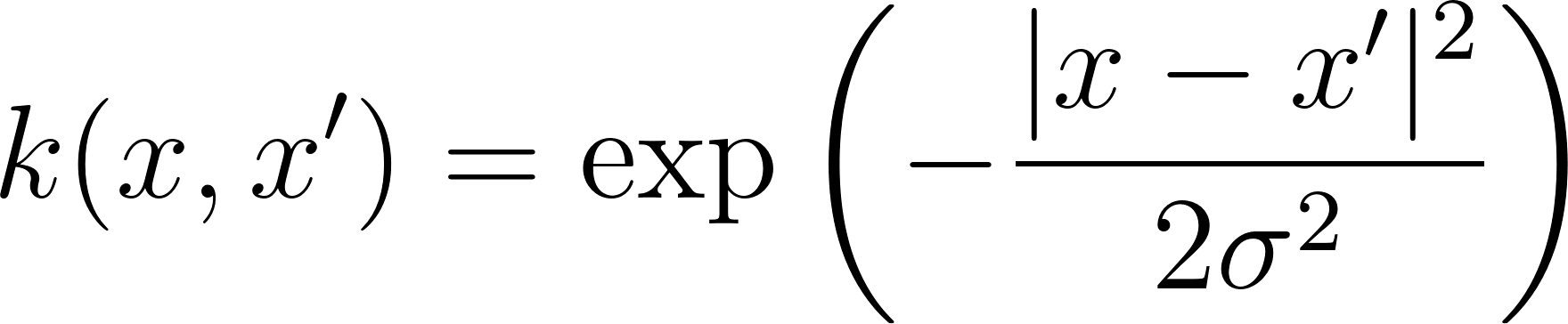
[](https://www.codecogs.com/eqnedit.php?latex=%20b%20%3D%20%5Cfrac%7B1%7D%7BM%7D%20%5Csum_%7Bi%3D1%7D%5E%7Bn%7D%20(y_i%20-%20%5Csum_%7Bj%3D1%7D%5E%7Bn%7D%20%5Calpha_j%20y_j%20%5Clangle%20x_i%2C%20x_j%20%5Crangle)%20#0)

where $M$ is the number of support vectors, which are the training examples with non-zero Lagrange multipliers.

The dual formulation of the problem has the advantage that it only depends on the inner products of the feature vectors, which can be computed using a kernel function, allowing for non-linear separation of the two classes.

1. **Give the definition of a positive definite kernel, and give an example of kernel and explain its key property used in SVM to deal with data non linearly separable.**

A positive kernel is a function K that take two vectors in R^d and returns a real value, with the properties of being symmetric (K(x,y) = K(y,x)) and the matrix defined by K(xi, xj) for the vector of points xi being symmetric semi definite.

One common kernel is the RBF kernel ([](https://www.codecogs.com/eqnedit.php?latex=k(x%2Cx')%20%3D%20%5Cexp%5Cleft(-%5Cfrac%7B%7Cx-x'%7C%5E2%7D%7B2%5Csigma%5E2%7D%5Cright)#0)) that implicitly projects the data into a higher dimension with a nonlinear function and then takes the dot product of the vectors. In this higher dimension the data becomes linearly separable.

1. **Give the decision function computed by a SVM based on a kernel.**

Alpha is the lagrange multiplier, y is the output, xi is the ith support vector, b is the bias and x is the input.

**3. Decision Trees and Ensemble methods**

1. **Describe the construction algorithm for a decision tree.**
   1. We start with the entire dataset in the root node.
   2. We choose a feature to split this dataset in order to minimize the impurity according to some metric like gini index or information gain.
   3. For each split recursively split until stopping criterion is met (can be a minimum number of samples the leaf must have or maximum depth of the tree).
2. **When do we find a null training error when building a decision tree?**

We find null training error when all the leaves are pure. A leaf is pure when all of its training samples have the same class. This happens when the tree is allowed to grow indefinitely.

1. **Explain in this framework the notion of overfitting.**

The decision tree overfits when it grows too deep, learns the noise of the data and cannot generalize to other datasets. In this case the bias of the tree is low but the variance is high.

1. **Which hyperparameter do you advise to tune in order to control overfitting?**

We can tune the maximum depth of the tree, the minimum number of samples a leaf must have or the minimum number of samples is required to make a split. All these methods are a trade off between how well the model can fit the data and still generalize.

1. **What is the bias-variance decomposition? Explain the 3 terms. How is it useful for analyzing bagging?**

It’s a way of decomposing the expected error in three terms: bias, variance and irreducible error or noise (expected error = noise + bias² + variance).

Bias is the difference between the minimal error and the average model error.

Variance is how much the results vary between one training set to another.

Noise is the error due to the inherent noise of the data.

Bagging is a way of combining multiple models with high variance and low bias in order to reduce the variance.

1. **Give the pseudocode of the random forest. Specify and justify a halting condition for growing each tree and the number of trees.**

RandomForest(X,numTrees, numFeatures):

forrest = empty\_list()

for numTrees:

sample = new\_sample(X, numFeatures)

forrest.add(DecisionTree(sample))

return forrest

DecisionTree(X):

if stopping\_condition:

return leaf(X).

else:

leftX, rightXt = find\_best\_split(X)

left\_node = DecisionTree(leftX)

right\_node = DecitionTree(rightX)

return node(left\_node, right\_node)

The halting condition of a tree can be its depth or the minimum number of samples per leaf. These are regularization hyperparameters and can be chosen with cross validation. The number of trees can be chosen with a validation set but it’s usually a trade off between accuracy and computational cost since a large number of trees is robust to overfitting.

1. **Give the definition of importance feature.**

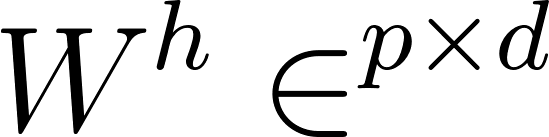
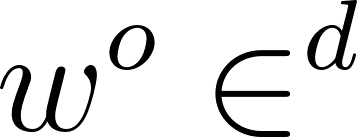
Importance feature is finding the features that are more relevant in the prediction of a model, how much each feature contributed to the prediction. In decision trees, the importance feature can be calculated as the sum of the impurity that one feature reduced in the tree. In bagging, it can be calculated as the average of the importance in each tree.

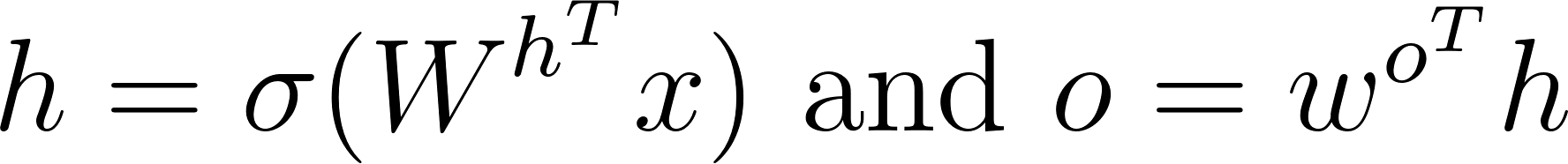
1. **How will the bias/variance improve with regard to a single tree? When and how does it improve over bagging?**

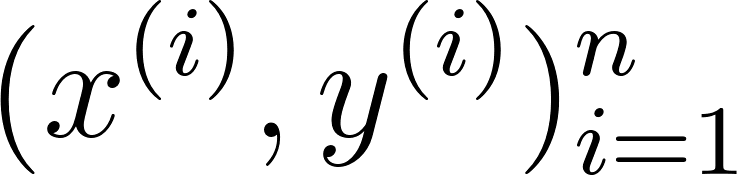
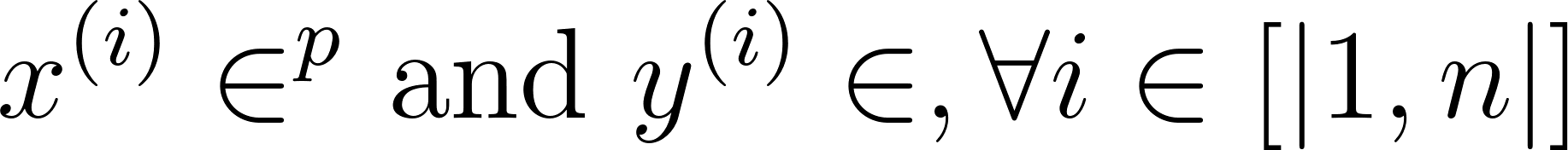
In a single tree, the larger it overfits more, the less bias it has and more variance. In bagging, each tree has low bias and high variance, by training them in different parts of the dataset and averaging then it’s possible to reduce the variance of the whole model.

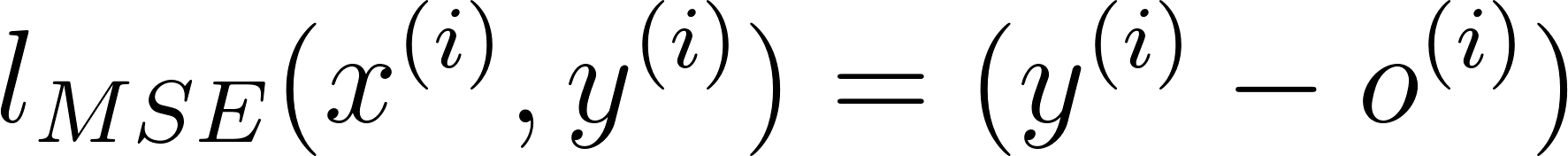
**4. Introduction to Deep Learning prova 2022**

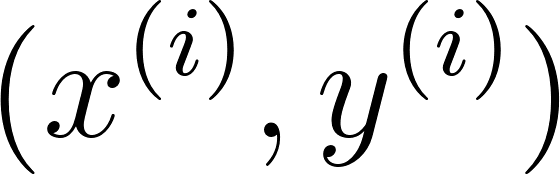
1. **Recall the update rule used in the gradient descent method when optimizing a loss function which depends on parameters , using a learning rate .**

**We now work with a one-hidden-layer neural network, taking as inputs data points , and outputting a scalar value . The model is comprised of parameters where** [****](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=W%5Eh%20%5Cin%20%5CR%5E%7Bp%20%5Ctimes%20d%7D#0) **and** [****](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=w%5Eo%20%5Cin%20%5CR%5Ed#0) **(we assume no bias for simplicity). The output is obtained as follows:**

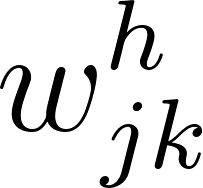
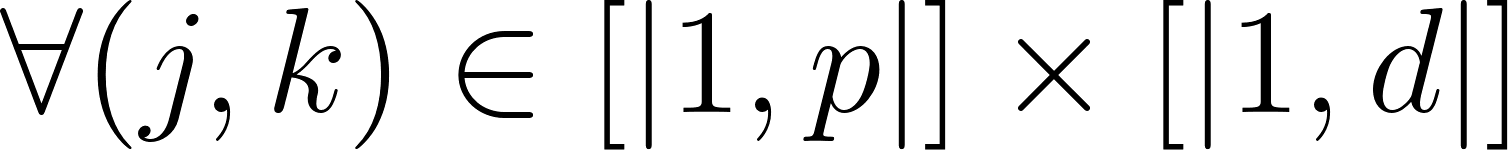
[****](https://www.codecogs.com/eqnedit.php?latex=h%20%3D%20%5Csigma(W%5E%7Bh%5ET%7D%20x)%24%20and%20%24o%20%3D%20w%5E%7Bo%5ET%7Dh#0)

**where** [****](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=h%20%5Cin%20%5CR%5Ed#0) **and** [****](https://www.codecogs.com/eqnedit.php?latex=%5Csigma#0) **is the sigmoid activation function. We want to train this model on a regression task, using a dataset** [****](https://www.codecogs.com/eqnedit.php?latex=(x%5E%7B(i)%7D%2C%20y%5E%7B(i)%7D)%5En_%7Bi%3D1%7D#0) **with** [****](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=x%5E%7B(i)%7D%20%5Cin%20%5CR%5Ep%24%20and%20%24y%5E%7B(i)%7D%20%5Cin%20%5CR%2C%20%5Cforall%20i%20%5Cin%20%5B%7C1%2Cn%7C%5D#0)**. Hence, we use the MSE loss function; when computed on one training pair, it is expressed as:**

[****](https://latex-staging.easygenerator.com/eqneditor/editor.php?latex=l_%7BMSE%7D(x%5E%7B(i)%7D%2C%20y%5E%7B(i)%7D)%20%3D%20(y%5E%7B(i)%7D%20-%20o%5E%7B(i)%7D)%C2%B2#0)**²**

1. **Using backpropagation, compute the gradient updates corresponding to one training pair** [****](https://www.codecogs.com/eqnedit.php?latex=(x%5E%7B(i)%7D%2C%20y%5E%7B(i)%7D)#0) **for:**

**- the components** [****](https://www.codecogs.com/eqnedit.php?latex=w%5Eo_k%24%20of%20%24w%5Eo%24%2C%20%24%5Cforall%20k%20%5Cin%20%5B%7C1%2Cd%7C%5D#0)**,**

**- the components** [****](https://www.codecogs.com/eqnedit.php?latex=w%5Eh_%7Bjk%7D#0) **of** [****](https://www.codecogs.com/eqnedit.php?latex=W%5Eh#0)**,** [****](https://www.codecogs.com/eqnedit.php?latex=%5Cforall(j%2Ck)%20%5Cin%20%5B%7C1%2Cp%7C%5D%20%5Ctimes%20%20%5B%7C1%2Cd%7C%5D#0)**.**

1. **With gradient updates computed via backpropagation, optimization of deep neural network remain difficult. Give and explain the idea behind 3 innovations made to ease training of deep neural models.**

Relu which solved the problem of the vanishing gradient. Dropout, a regularization technique to prevent overfitting. Batch normalization, that normalizes the input to each layer of the network and improves the training speed.

**5. Introduction to Deep Learning prova 2017**

**Let be the function computed by a one-hidden layer perceptron for a classification problem with C classes**

1. **Propose an architecture for (describe each layer) and define the required functions.**

* The architecture is one input layer that takes a d dim vector as input.
* One hidden layer that computes the hidden features h of p dimension with a weight matrix W of dxp dim and an activation function g. h = g(W1\*x+ b1).
* The output layer computes the output by multiplying the hidden features by a pxc matrix that outputs a dxc matrix and then we use a softmax function to calculate the probabilities of each class. o = softmax(W2\*h+b2)
* The final result is the argmax of the output.

1. **Define the optimization problem for the architecture you defined for multiclass classification.**

The optimization problem is the minimize the loss function of model, which is the average of loss for each datapoint (y^T \* log(f(x))). y is the true label.

1. **Without giving all the details, express which gradients need to be computed for each kind of parameters in the network.**

**6. Towards Large Scale**

**Among the approaches that you have studied during the lectures or the practical session for supervised classification, which one would you use when you have a very large training set (big data)? Justify briefly your choice.**