**Machine Learning Models / Algorithms**

**Week 4 - Logistic Regression:**

Logistic Regression is a variation of Linear Regression, used when the observed dependent variable, **y**, is categorical. It produces a formula that predicts the probability of the class label as a function of the independent variables.

A math equation with numbers and symbols

Description automatically generatedLogistic regression fits a special s-shaped curve by taking the linear regression function and transforming the numeric estimate into a probability with the following function, which is called the sigmoid function 𝜎:

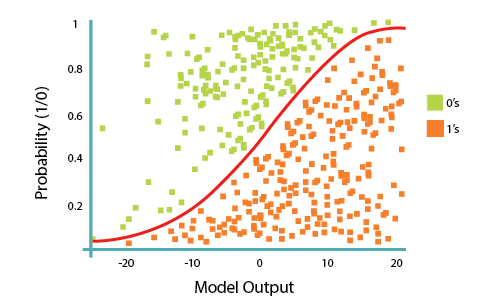
In this equation, θTX is the regression result (the sum of the variables weighted by the coefficients), exp is the exponential function and σ(θTX) is the sigmoid or [logistic function](http://en.wikipedia.org/wiki/Logistic_function?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkML0101ENSkillsNetwork20718538-2021-01-01), also called logistic curve. It is a common "S" shape (sigmoid curve).

Logistic regression is a data analysis technique that uses mathematics to find the relationships between two data factors. It then uses this relationship to predict the value of one of those factors based on the other. The prediction usually has a finite number of outcomes, like yes or no.

**Logistic Regression which is used for classification. Logistic regression is a statistical and machine learning technique for classifying records of a dataset based on the values of the input fields.**

Logistic regression is analogous to linear regression but tries to predict a categorical or discrete target field instead of a numeric one.

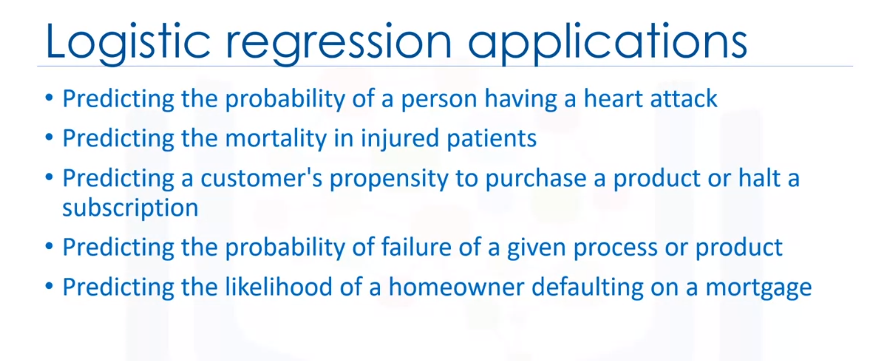
So, briefly, Logistic Regression passes the input through the logistic/sigmoid but then treats the result as a probability:



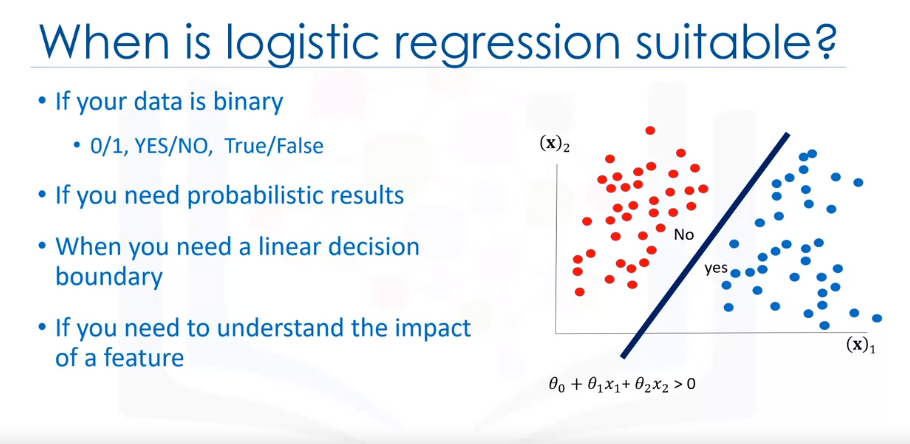
The objective of the **Logistic Regression** algorithm is to find the best parameters θ, for h\_θ(x) = σ(θTX)in such a way that the model best predicts the class of each case.

Please note that logistic regression can be used for both binary classification and multi-class classification.

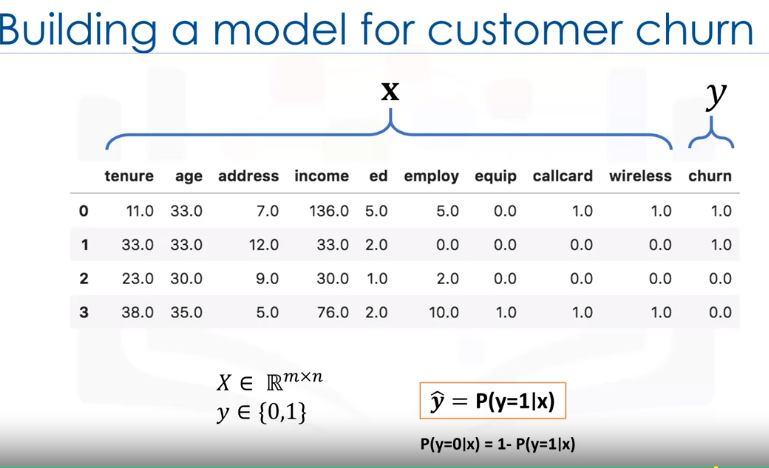
**Logistic regression Applications/Example:**



Logistic regression returns a probability score between zero and one for a given sample of data. In fact, logistic regression predicts the probability of that sample and we map the cases to a discrete class based on that probability.



you need to understand the impact of a feature. You can select the best features based on the statistical significance of the logistic regression model coefficients or parameters.

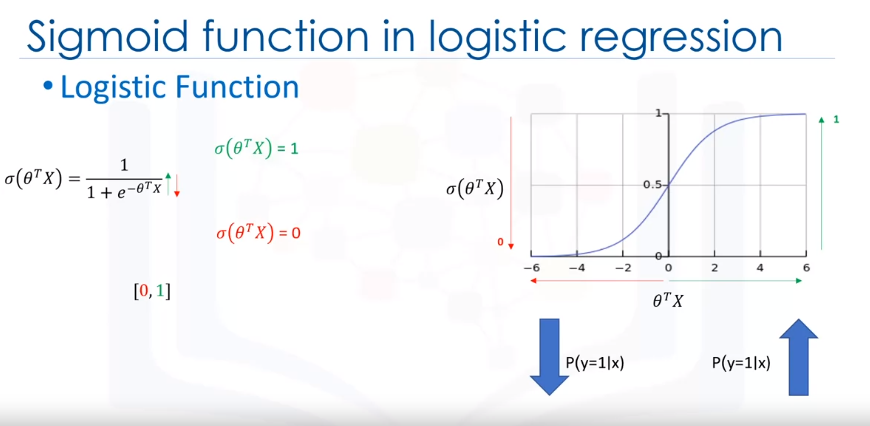
It allows us to understand the impact an independent variable has on the dependent variable while controlling other independent variables.

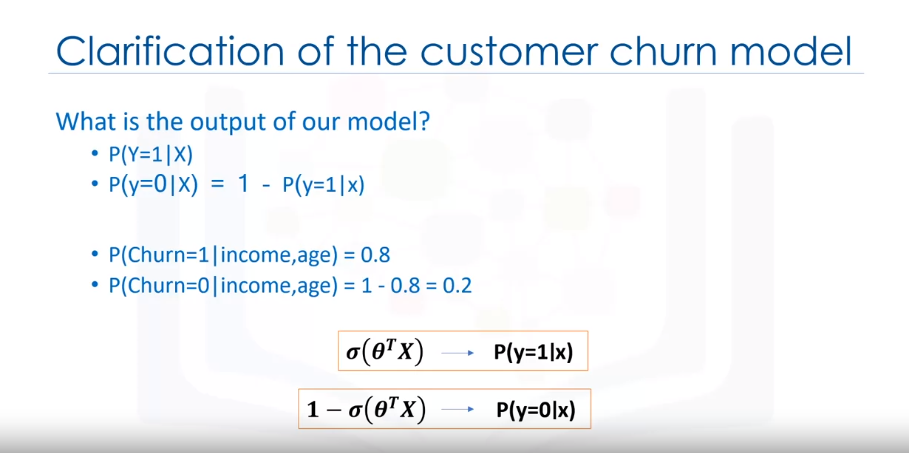
The goal of logistic regression is to build a model to predict the class of a dependent variable and also the probability of each sample belonging to a class. Ideally, we want to build a model, y hat, that can estimate that the class of a dependent variable is one given its feature is x.

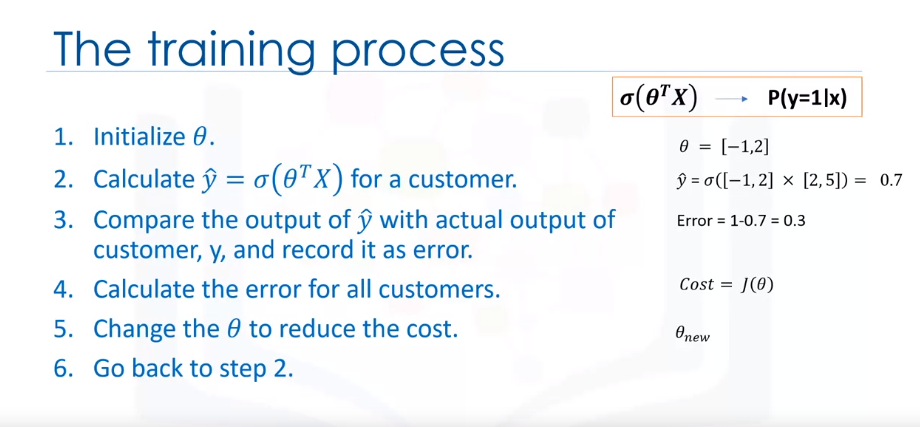
The sigmoid function, also called the logistic function, resembles the step function and is used by the following expression in the logistic regression.

The sigmoid function looks a bit complicated at first, but don't worry about remembering this equation, it'll make sense to you after working with it.

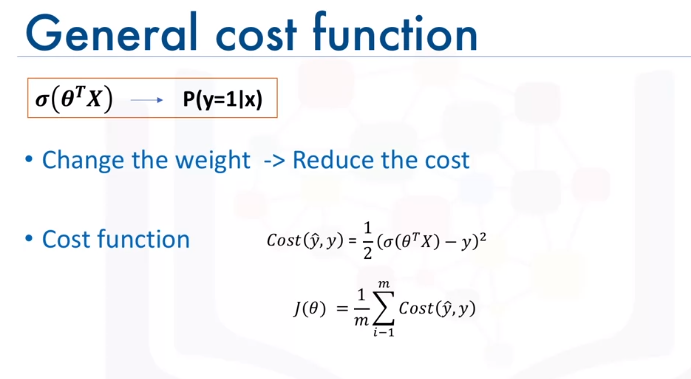
Notice that in the sigmoid equation, when Theta transpose x gets very big, the e power minus Theta transpose x in the denominator of the fraction becomes almost 0, and the value of the sigmoid function gets closer to 1. If Theta transpose x is very small, the sigmoid function gets closer to 0. Depicting on the in sigmoid plot, when Theta transpose x gets bigger, the value of the sigmoid function gets closer to 1, and also, if the Theta transpose x is very small, the sigmoid function gets closer to 0. So, the sigmoid functions output is always between 0 and 1, which makes it proper to interpret the results as probabilities. It is obvious that when the outcome of the sigmoid function gets closer to 1, the probability of y equals 1 given x goes up. In contrast, when the sigmoid value is closer to 0, the probability of y equals 1 given x is very small.

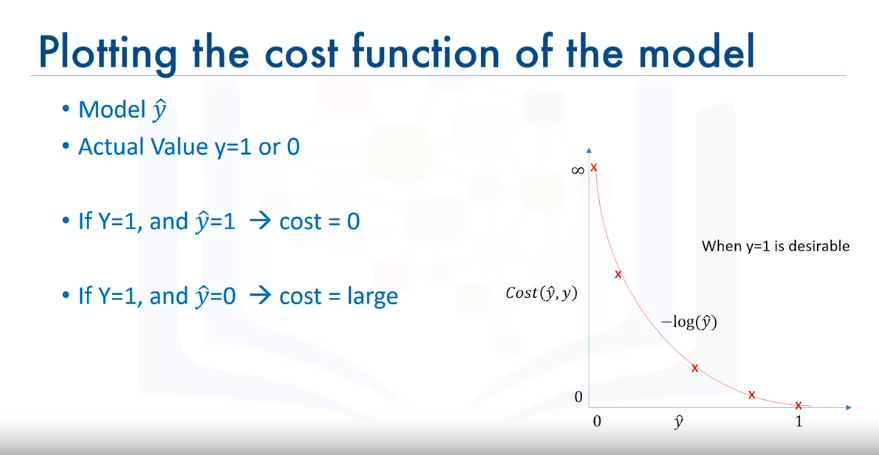




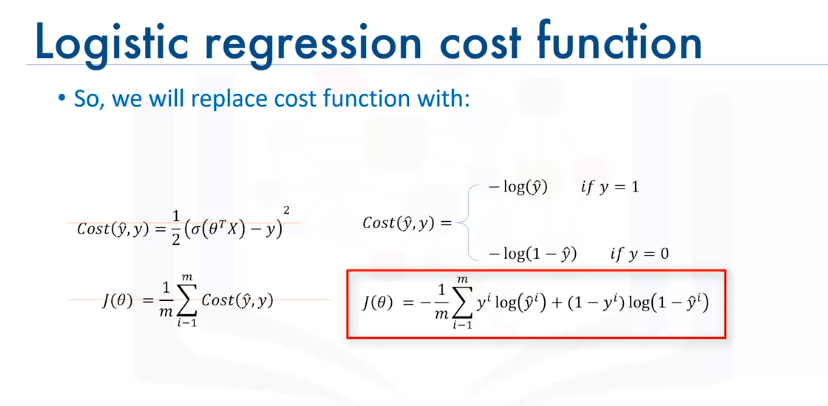


**Cost Function of Logistic Regression:**



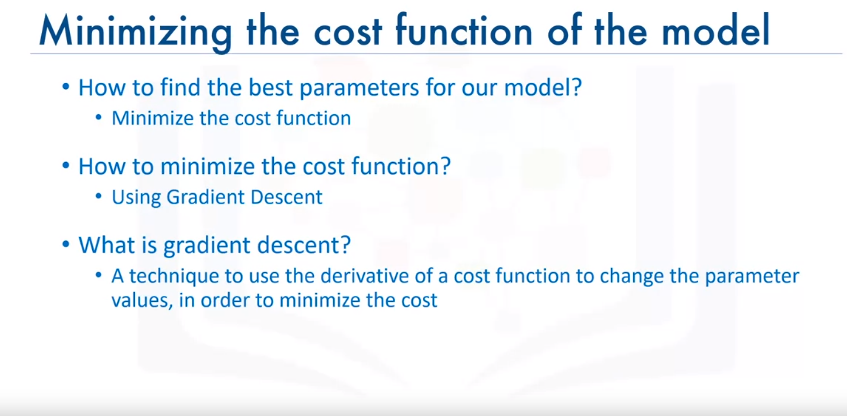


Simply put, the lower the cost, the better the model or the higher the cost, the worse the model. So, the ultimate goal is to minimize the cost of the function J(theta).



So, this is the logistic regression cost function. As you can see for yourself it penalizes situations in which the class is zero and the model output is one, and vice versa.

Remember, however, that y hat does not return a class as output but it's a value of zero or one which should be assumed as a probability.

Now, we can easily use this function to find the parameters of our model in such a way as to minimize the cost.

**Gradient Descent:**

Generally, gradient descent is an iterative approach to finding the minimum of a function. Specifically in our case gradient descent is a technique to use the derivative of a cost function to change the parameter values to minimize the cost or error.

**The main objective of gradient descent is to change the parameter values so as to minimize the cost.**

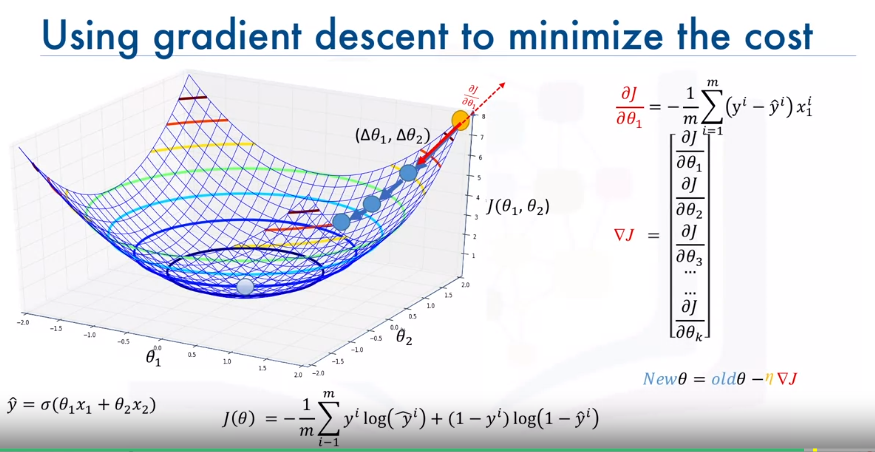
The gradient is the slope of the surface at every point and the direction of the gradient is the direction of the greatest uphill.

If you select a random point on this surface, for example the yellow point, and take the partial derivative of J of theta with respect to each parameter at that point, it gives you the slope of the move for each parameter at that point.

Now, if we move in the opposite direction of that slope, it guarantees that we go down in the error curve. For example, if we calculate the derivative of J with respect to theta one, we find out that it is a positive number. This indicates that function is increasing as theta one increases.

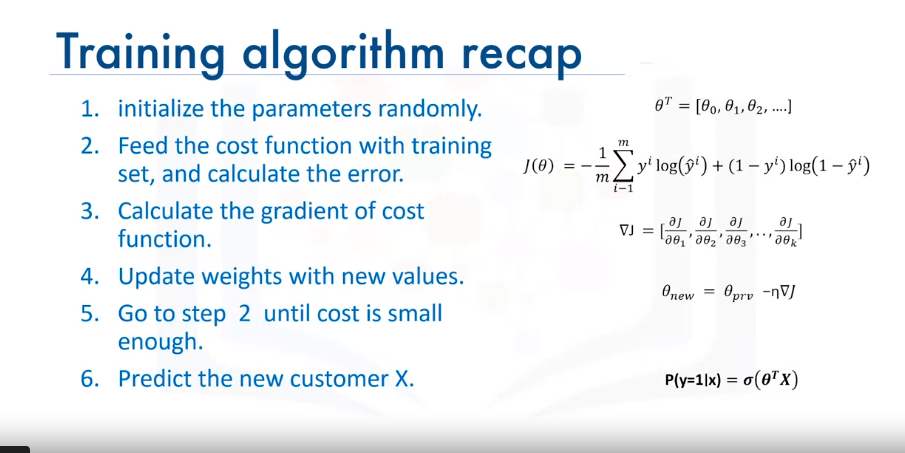
So, to decrease J, we should move in the opposite direction. This means to move in the direction of the negative derivative for theta one, i.e. slope. We have to calculate it for other parameters as well at each step.

The gradient value also indicates how big of a step to take. If the slope is large we should take a large step because we are far from the minimum. If the slope is small we should take a smaller step.

Gradient descent takes increasingly smaller steps towards the minimum with each iteration. The partial derivative of the cost function J is calculated using this expression.

**Leaning Rate (ɳ - Mu) :** Learning rate, gives us additional control on how fast we move on the surface. In sum, we can simply say, gradient descent is like taking steps in the current direction of the slope, and the learning rate is like the length of the step you take.

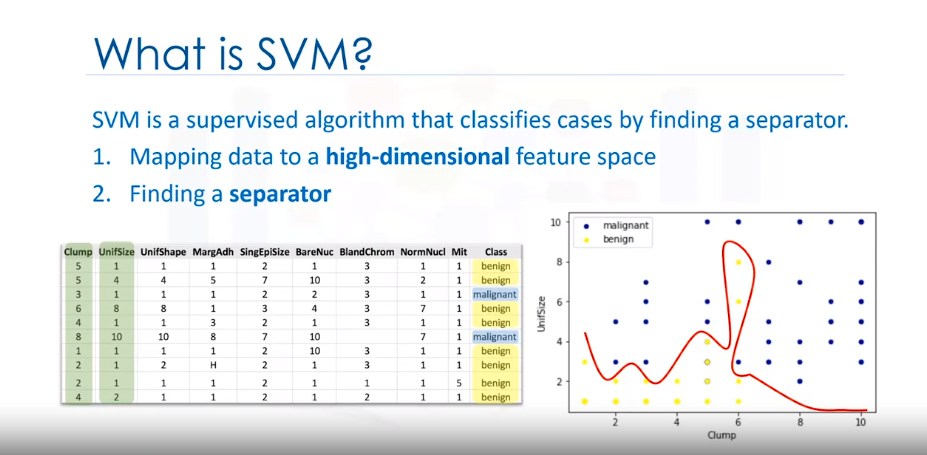
**Summary and Algorithm Recap:**



**Support Vector Machine**

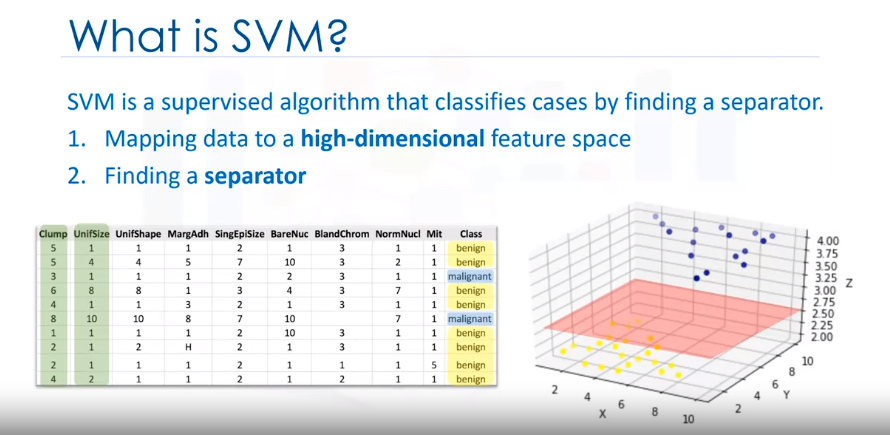
SVM is a supervised algorithm or machine learning model that classifies cases by finding a separator.

SVM works by first mapping data to a high dimensional feature space so that data points can be categorized, even when the data are not linearly separable. Then, a separator is estimated for the data. The data should be transformed in such a way that a separator could be drawn as a hyperplane.



For example, consider the following figure, which shows the distribution of a small set of cells only based on their unit size and clump thickness. As you can see, the data points fall into two different categories. It represents a linearly non separable data set. The two categories can be separated with a curve but not a line.

SVM works by mapping data to a high-dimensional feature space so that data points can be categorized, even when the data are not otherwise linearly separable. A separator between the categories is found, then the data is transformed in such a way that the separator could be drawn as a hyperplane. Following this, characteristics of new data can be used to predict the group to which a new record should belong.



That is, it represents a linearly non separable data set, which is the case for most real world data sets. We can transfer this data to a higher-dimensional space, for example, mapping it to a three-dimensional space. After the transformation, the boundary between the two categories can be defined by a hyperplane. As we are now in three-dimensional space, the separator is shown as a plane. This plane can be used to classify new or unknown cases. Therefore, the SVM algorithm outputs an optimal hyperplane that categorizes new examples.

The SVM algorithm offers a choice of kernel functions for performing its processing. Basically, mapping data into a higher dimensional space is called kernelling. The mathematical function used for the transformation is known as the **kernel function**, and can be of different types, such as:

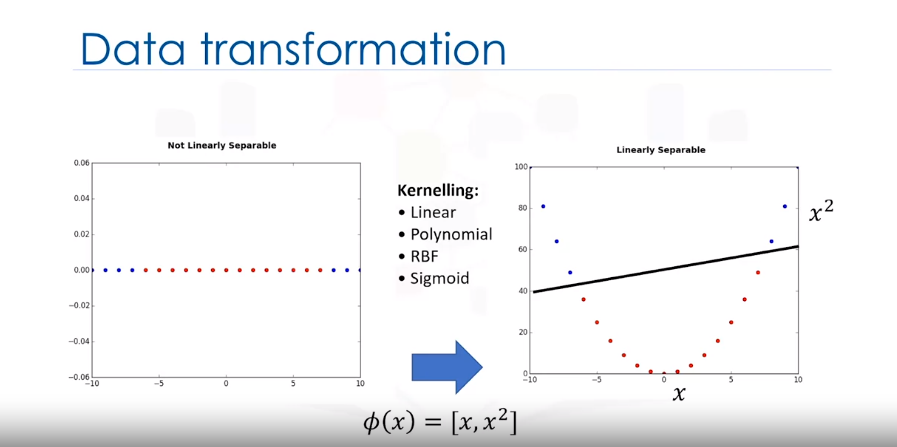
1.Linear

2.Polynomial

3.Radial basis function (RBF)

4.Sigmoid

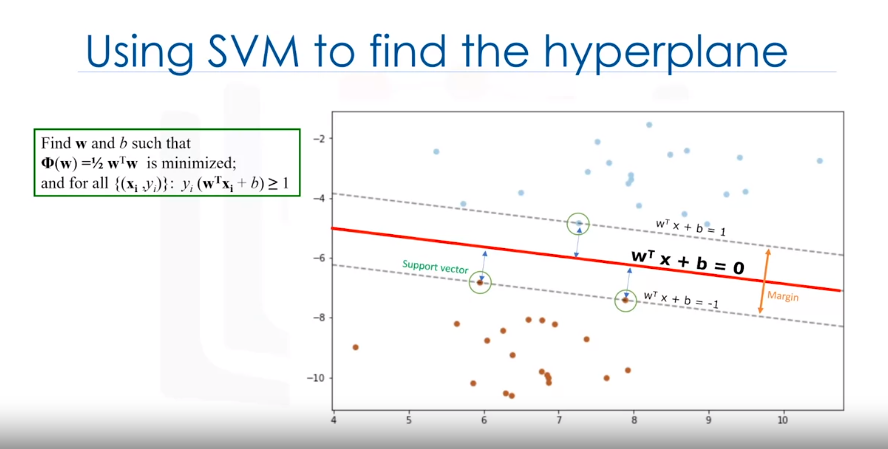
In order to make the data points linearly separable, we can transform the data (independent variable) using the kernelling which can be done using different types of techniques such linear, polynomial, Radial Basis Function, and sigmoid.



One reasonable choice as the best hyperplane is the one that represents the largest separation or margin between the two classes. So the goal is to choose a hyperplane with as big a margin as possible. Examples closest to the hyperplane are support vectors. It is intuitive that only support vectors matter for achieving our goal.

Each of these functions has its characteristics, its pros and cons, and its equation, but as there's no easy way of knowing which function performs best with any given dataset. We usually choose different functions in turn and compare the results. Let's just use the default, RBF (Radial Basis Function) for this lab.

**Finding the hyperplane and best values:**

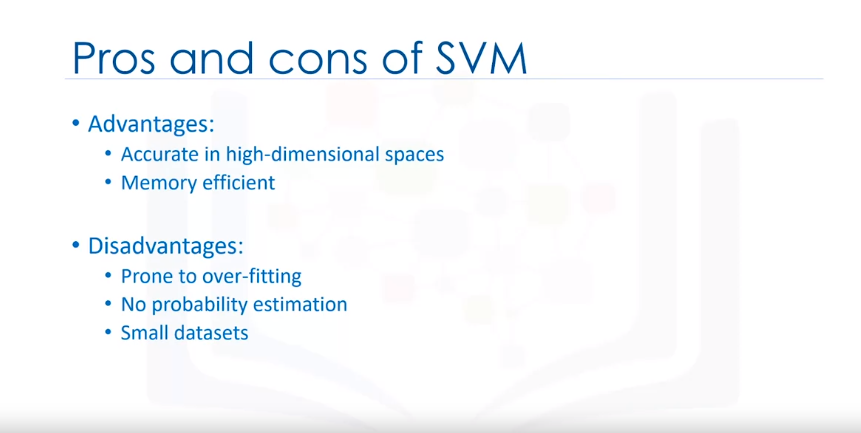


the hyperplane is learned from training data using an optimization procedure that maximizes the margin. And like many other problems, this optimization problem can also be solved by gradient descent.

Therefore, the output of the algorithm is the values w and b for the line. You can make classifications using this estimated line. It is enough to plug in input values into the line equation.

Then, you can calculate whether an unknown point is above or below the line. If the equation returns a value greater than 0, then the point belongs to the first class which is above the line, and vice-versa.

**Advantages and Disadvantages:**



* The disadvantages of Support Vector Machines include the fact that the algorithm is prone for over-fitting if the number of features is much greater than the number of samples.
* SVMs do not directly provide probability estimates, which are desirable in most classification problems.
* And finally, SVMs are not very efficient computationally if your dataset is very big, such as when you have more than 1,000 rows.

**In which situation to use SVM?**

**Multiclass Prediction:**

In statistics, multinomial logistic regression is a classification method that generalizes logistic regression to multiclass problems, i.e. with more than two possible discrete outcomes.

In supervised learning, a [**classification**](https://developers.google.com/machine-learning/glossary#classification_model) problem in which the dataset contains *more than two* [**classes**](https://developers.google.com/machine-learning/glossary#class) of labels. For example, the labels in the Iris dataset must be one of the following three classes:

* Iris setosa
* Iris virginica
* Iris versicolor

A model trained on the Iris dataset that predicts Iris type on new examples is performing multi-class classification.

In contrast, classification problems that distinguish between exactly two classes are [**binary classification models**](https://developers.google.com/machine-learning/glossary#binary_classification). For example, an email model that predicts either *spam* or *not spam* is a binary classification model.

In clustering problems, multi-class classification refers to more than two clusters.

**class**

#fundamentals

A category that a [**label**](https://developers.google.com/machine-learning/glossary#label) can belong to. For example:

* In a [**binary classification**](https://developers.google.com/machine-learning/glossary#binary_classification) model that detects spam, the two classes might be *spam* and *not spam*.
* In a [**multi-class classification**](https://developers.google.com/machine-learning/glossary#multi-class) model that identifies dog breeds, the classes might be *poodle*, *beagle*, *pug*, and so on.

A [**classification model**](https://developers.google.com/machine-learning/glossary#classification_model) predicts a class. In contrast, a [**regression model**](https://developers.google.com/machine-learning/glossary#regression_model) predicts a number rather than a class.

The softmax function is used in multiclass classification methods such as neural networks, multinomial logistic regression, multiclass LDA, and Naive Bayes classifiers. The softmax function is used to output action probabilities in case of reinforcement learning.

The concept of Multi-class classification for linear classifiers is not as straightforward. We can convert logistic regression to Multi-class classification using multinomial logistic regression or SoftMax regression; this is a generalization of logistic regression. SoftMax regression will not work for Support Vector Machines (SVM); One vs. All (One-vs-Rest) and One vs One are two other multi-class classification techniques that can convert most two-class classifiers to a multi-class classifier.

**Softmax classifiers give you probabilities for each class label while hinge loss gives you the margin.**

Source: <https://pyimagesearch.com/2016/09/12/softmax-classifiers-explained/>

<https://developers.google.com/machine-learning/crash-course/multi-class-neural-networks/one-vs-all>

<https://developers.google.com/machine-learning/glossary#class>

The np. argmax function simply returns the index of the maximum value in the array. Having said that, there are some more complicated ways of using the function. For example, you can use the function along particular axes and retrieve the index of the maximum value for a particular array axis

The softmax function is a function that turns a vector of K real values into a vector of K real values that sum to 1. The output of the function is always between 0 and 1, which can be used as a probability score.

**One vs. all** provides a way to leverage binary classification. Given a classification problem with N possible solutions, a one-vs.-all solution consists of N separate binary classifiers—one binary classifier for each possible outcome. During training, the model runs through a sequence of binary classifiers, training each to answer a separate classification question. For example, given a picture of a dog, five different recognizers might be trained, four seeing the image as a negative example (not an apple, not a bear, etc.) and one seeing the image as a positive example (a dog). That is:

1. Is this image an apple? No.
2. Is this image a bear? No.
3. Is this image candy? No.
4. Is this image a dog? Yes.
5. Is this image an egg? No.

This approach is fairly reasonable when the total number of classes is small, but becomes increasingly inefficient as the number of classes rises.

We can create a significantly more efficient one-vs.-all model with a deep neural network in which each output node represents a different class. The following figure suggests this approach:

# **One-vs-One classification**

In One-vs-One classification, we split up the data into each class; we then train a two-class classifier on each pair of classes. For example, if we have class 0,1, and 2, we would train one classifier on the samples that are class 0 and class 1, a second classifier on samples that are of class 0 and class 2, and a final classifier on samples of class 1 and class 2. Fig 7 is an example of class 0 vs class 1, where we drop training samples  of class 2.  Using the same convention as above where the color of the training samples are based on their class. The separating plane of the classifier is in black.  The color represents the output of the classifier for that particular point in space.

A colorful squares with black text

Description automatically generated with medium confidence

**For *K* classes, we have to train *K*(*K*−1)/2 classifiers. So if *K*=3, we have (3×2)/2=3(3×2)/2=3classes.**

To perform Classification on a sample, we perform a majority vote where we select the class with the most predictions.  This is shown in Fig  9 where the black point represents a new sample and the output of each classifier is shown in the table. In this case, the final output is one as selected by two of the three classifiers. There is also an ambiguous region but it’s smaller, we can also use similar schemes as in One vs all like the fusion rule or using the probability. Check out the labs for more.

A diagram of mathematical equations

Description automatically generated with medium confidence

**In Multi-class classification, we classify data into multiple class labels . Unlike classification trees and k-nearest neighbour, the concept of Multi-class classification for linear classifiers is not as straightforward. We can convert logistic regression to Multi-class classification using multinomial logistic regression or softmax regression; this is a generalization of logistic regression, this will not work for support vector machines. One vs. All (One-vs-Rest) and One vs. One are two other multi-class classification techniques can covert any two-class classifier to a multi-class classifier.**