Applications of Machine learning

Machine learning is a buzzword for today's technology, and it is growing very rapidly day by day. We are using machine learning in our daily life even without knowing it such as Google Maps, Google assistant, Alexa, etc. Below are some most trending real-world applications of Machine Learning:



### 1. Image Recognition:

Image recognition is one of the most common applications of machine learning. It is used to identify objects, persons, places, digital images, etc. The popular use case of image recognition and face detection is, **Automatic friend tagging suggestion**:

Facebook provides us a feature of auto friend tagging suggestion. Whenever we upload a photo with our Facebook friends, then we automatically get a tagging suggestion with name, and the technology behind this is machine learning's **face detection** and **recognition algorithm**

It is based on the Facebook project named "**Deep Face**," which is responsible for face recognition and person identification in the picture.

### 2. Speech Recognition

While using Google, we get an option of "**Search by voice**," it comes under speech recognition, and it's a popular application of machine learning.

Speech recognition is a process of converting voice instructions into text, and it is also known as "**Speech to text**", or "**Computer speech recognition**." At present, machine learning algorithms are widely used by various applications of speech recognition. **Google assistant**, **Siri**, **Cortana**, and **Alexa** are using speech recognition technology to follow the voice instructions.

### 3. Traffic prediction:

If we want to visit a new place, we take help of Google Maps, which shows us the correct path with the shortest route and predicts the traffic conditions.

It predicts the traffic conditions such as whether traffic is cleared, slow-moving, or heavily congested with the help of two ways:

* **Real Time location** of the vehicle form Google Map app and sensors
* **Average time has taken** on past days at the same time.

Everyone who is using Google Map is helping this app to make it better. It takes information from the user and sends back to its database to improve the performance.

### 4. Product recommendations:

Machine learning is widely used by various e-commerce and entertainment companies such as **Amazon**, **Netflix**, etc., for product recommendation to the user. Whenever we search for some product on Amazon, then we started getting an advertisement for the same product while internet surfing on the same browser and this is because of machine learning.

Google understands the user interest using various machine learning algorithms and suggests the product as per customer interest.

As similar, when we use Netflix, we find some recommendations for entertainment series, movies, etc., and this is also done with the help of machine learning.

### 5. Self-driving cars:

One of the most exciting applications of machine learning is self-driving cars. Machine learning plays a significant role in self-driving cars. Tesla, the most popular car manufacturing company is working on self-driving car. It is using unsupervised learning method to train the car models to detect people and objects while driving.

### 6. Email Spam and Malware Filtering:

Whenever we receive a new email, it is filtered automatically as important, normal, and spam. We always receive an important mail in our inbox with the important symbol and spam emails in our spam box, and the technology behind this is Machine learning. Below are some spam filters used by Gmail:

* Content Filter
* Header filter
* General blacklists filter
* Rules-based filters
* Permission filters

Some machine learning algorithms such as **Multi-Layer Perceptron**, **Decision tree**, and **Naïve Bayes classifier** are used for email spam filtering and malware detection.

### 7. Virtual Personal Assistant:

We have various virtual personal assistants such as **Google assistant**, **Alexa**, **Cortana**, **Siri**. As the name suggests, they help us in finding the information using our voice instruction. These assistants can help us in various ways just by our voice instructions such as Play music, call someone, Open an email, Scheduling an appointment, etc.

These virtual assistants use machine learning algorithms as an important part.

These assistant record our voice instructions, send it over the server on a cloud, and decode it using ML algorithms and act accordingly.

### 8. Online Fraud Detection:

Machine learning is making our online transaction safe and secure by detecting fraud transaction. Whenever we perform some online transaction, there may be various ways that a fraudulent transaction can take place such as **fake accounts**, **fake ids**, and **steal money** in the middle of a transaction. So to detect this, **Feed Forward Neural network** helps us by checking whether it is a genuine transaction or a fraud transaction.

For each genuine transaction, the output is converted into some hash values, and these values become the input for the next round. For each genuine transaction, there is a specific pattern which gets change for the fraud transaction hence, it detects it and makes our online transactions more secure.

### 9. Stock Market trading:

Machine learning is widely used in stock market trading. In the stock market, there is always a risk of up and downs in shares, so for this machine learning's **long short term memory neural network** is used for the prediction of stock market trends.

### 10. Medical Diagnosis:

In medical science, machine learning is used for diseases diagnoses. With this, medical technology is growing very fast and able to build 3D models that can predict the exact position of lesions in the brain.

It helps in finding brain tumors and other brain-related diseases easily.

### 11. Automatic Language Translation:

Nowadays, if we visit a new place and we are not aware of the language then it is not a problem at all, as for this also machine learning helps us by converting the text into our known languages. Google's GNMT (Google Neural Machine Translation) provide this feature, which is a Neural Machine Learning that translates the text into our familiar language, and it called as automatic translation.

The technology behind the automatic translation is a sequence to sequence learning algorithm, which is used with image recognition and translates the text from one language to another language.

# Regression Analysis in Machine learning

Regression analysis is a statistical method to model the relationship between a dependent (target) and independent (predictor) variables with one or more independent variables. More specifically, Regression analysis helps us to understand how the value of the dependent variable is changing corresponding to an independent variable when other independent variables are held fixed. It predicts continuous/real values such as **temperature, age, salary, price,** etc.

We can understand the concept of regression analysis using the below example:

**Example:** Suppose there is a marketing company A, who does various advertisement every year and get sales on that. The below list shows the advertisement made by the company in the last 5 years and the corresponding sales:



Now, the company wants to do the advertisement of $200 in the year 2019 **and wants to know the prediction about the sales for this year**. So to solve such type of prediction problems in machine learning, we need regression analysis.

Regression is a [supervised learning technique](https://www.javatpoint.com/supervised-machine-learning) which helps in finding the correlation between variables and enables us to predict the continuous output variable based on the one or more predictor variables. It is mainly used for **prediction, forecasting, time series modeling, and determining the causal-effect relationship between variables**.

In Regression, we plot a graph between the variables which best fits the given datapoints, using this plot, the machine learning model can make predictions about the data. In simple words, **"Regression shows a line or curve that passes through all the datapoints on target-predictor graph in such a way that the vertical distance between the datapoints and the regression line is minimum."** The distance between datapoints and line tells whether a model has captured a strong relationship or not.

Some examples of regression can be as:

* Prediction of rain using temperature and other factors
* Determining Market trends
* Prediction of road accidents due to rash driving.

## Terminologies Related to the Regression Analysis:

* **Dependent Variable:** The main factor in Regression analysis which we want to predict or understand is called the dependent variable. It is also called **target variable**.
* **Independent Variable:** The factors which affect the dependent variables or which are used to predict the values of the dependent variables are called independent variable, also called as a **predictor**.
* **Outliers:** Outlier is an observation which contains either very low value or very high value in comparison to other observed values. An outlier may hamper the result, so it should be avoided.
* **Multicollinearity:** If the independent variables are highly correlated with each other than other variables, then such condition is called Multicollinearity. It should not be present in the dataset, because it creates problem while ranking the most affecting variable.
* **Underfitting and Overfitting:** If our algorithm works well with the training dataset but not well with test dataset, then such problem is called **Overfitting**. And if our algorithm does not perform well even with training dataset, then such problem is called **underfitting**.

## Why do we use Regression Analysis?

As mentioned above, Regression analysis helps in the prediction of a continuous variable. There are various scenarios in the real world where we need some future predictions such as weather condition, sales prediction, marketing trends, etc., for such case we need some technology which can make predictions more accurately. So for such case we need Regression analysis which is a statistical method and used in machine learning and data science. Below are some other reasons for using Regression analysis:

* Regression estimates the relationship between the target and the independent variable.
* It is used to find the trends in data.
* It helps to predict real/continuous values.
* By performing the regression, we can confidently determine the **most important factor, the least important factor, and how each factor is affecting the other factors**.

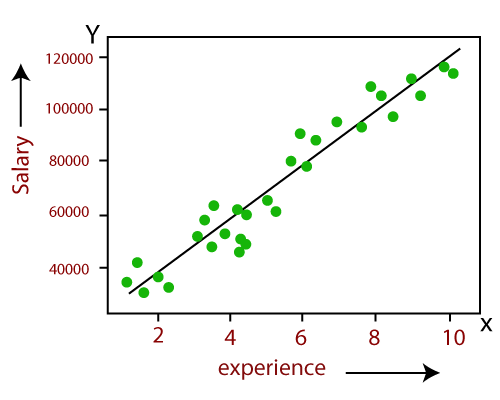
## Types of Regression

There are various types of regressions which are used in data science and machine learning. Each type has its own importance on different scenarios, but at the core, all the regression methods analyze the effect of the independent variable on dependent variables. Here we are discussing some important types of regression which are given below:

* **Linear Regression**
* **Logistic Regression**
* **Polynomial Regression**
* **Support Vector Regression**
* **Decision Tree Regression**
* **Random Forest Regression**
* **Ridge Regression**
* **Lasso Regression:**

### Linear Regression:

* Linear regression is a statistical regression method which is used for predictive analysis.
* It is one of the very simple and easy algorithms which works on regression and shows the relationship between the continuous variables.
* It is used for solving the regression problem in machine learning.
* Linear regression shows the linear relationship between the independent variable (X-axis) and the dependent variable (Y-axis), hence called linear regression.
* If there is only one input variable (x), then such linear regression is called **simple linear regression**. And if there is more than one input variable, then such linear regression is called **multiple linear regression**.
* The relationship between variables in the linear regression model can be explained using the below image. Here we are predicting the salary of an employee on the basis of **the year of experience**.



Below is the mathematical equation for Linear regression:

Y= aX+b

**Here, Y = dependent variables (target variables),**  
**X= Independent variables (predictor variables),**  
**a and b are the linear coefficients**

Some popular applications of linear regression are:

* **Analyzing trends and sales estimates**
* **Salary forecasting**
* **Real estate prediction**
* **Arriving at ETAs in traffic.**

# Linear Regression in Machine Learning

Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as **sales, salary, age, product price,** etc.

Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

The linear regression model provides a sloped straight line representing the relationship between the variables. Consider the below image:



Mathematically, we can represent a linear regression as:

y= a0+a1x+ ε

**Here,**

Y= Dependent Variable (Target Variable)  
X= Independent Variable (predictor Variable)  
a0= intercept of the line (Gives an additional degree of freedom)  
a1 = Linear regression coefficient (scale factor to each input value).  
ε = random error

The values for x and y variables are training datasets for Linear Regression model representation.

## Types of Linear Regression

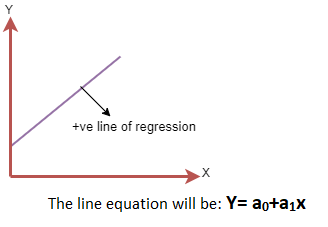
Linear regression can be further divided into two types of the algorithm:

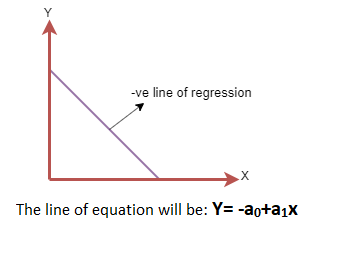
* **Simple Linear Regression:**  
  If a single independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Simple Linear Regression.
* **Multiple Linear regression:**  
  If more than one independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Multiple Linear Regression.

## Linear Regression Line

A linear line showing the relationship between the dependent and independent variables is called a **regression line**. A regression line can show two types of relationship:

* **Positive Linear Relationship:**  
  If the dependent variable increases on the Y-axis and independent variable increases on X-axis, then such a relationship is termed as a Positive linear relationship.



* **Negative Linear Relationship:**  
  If the dependent variable decreases on the Y-axis and independent variable increases on the X-axis, then such a relationship is called a negative linear relationship.
* 

## Finding the best fit line:

When working with linear regression, our main goal is to find the best fit line that means the error between predicted values and actual values should be minimized. The best fit line will have the least error.

The different values for weights or the coefficient of lines (a0, a1) gives a different line of regression, so we need to calculate the best values for a0 and a1 to find the best fit line, so to calculate this we use cost function.

### Cost function-

* The different values for weights or coefficient of lines (a0, a1) gives the different line of regression, and the cost function is used to estimate the values of the coefficient for the best fit line.
* Cost function optimizes the regression coefficients or weights. It measures how a linear regression model is performing.
* We can use the cost function to find the accuracy of the **mapping function**, which maps the input variable to the output variable. This mapping function is also known as **Hypothesis function**.

For Linear Regression, we use the **Mean Squared Error (MSE)** cost function, which is the average of squared error occurred between the predicted values and actual values. It can be written as:

For the above linear equation, MSE can be calculated as:

**C:\Users\Dileep\Desktop\linear-regression-in-machine-learning4.png**

**Where,**

N=Total number of observation  
Yi = Actual value  
(a1xi+a0)= Predicted value.

**Residuals:** The distance between the actual value and predicted values is called residual. If the observed points are far from the regression line, then the residual will be high, and so cost function will high. If the scatter points are close to the regression line, then the residual will be small and hence the cost function.

### Gradient Descent:

* Gradient descent is used to minimize the MSE by calculating the gradient of the cost function.
* A regression model uses gradient descent to update the coefficients of the line by reducing the cost function.
* It is done by a random selection of values of coefficient and then iteratively update the values to reach the minimum cost function.

## Model Performance:

The Goodness of fit determines how the line of regression fits the set of observations. The process of finding the best model out of various models is called **optimization**. It can be achieved by below method:

**1. R-squared method:**

* R-squared is a statistical method that determines the goodness of fit.
* It measures the strength of the relationship between the dependent and independent variables on a scale of 0-100%.
* The high value of R-square determines the less difference between the predicted values and actual values and hence represents a good model.
* It is also called a **coefficient of determination,** or **coefficient of multiple determination** for multiple regression.
* It can be calculated from the below formula:

## Assumptions of Linear Regression

Below are some important assumptions of Linear Regression. These are some formal checks while building a Linear Regression model, which ensures to get the best possible result from the given dataset.

* **Linear relationship between the features and target:**  
  Linear regression assumes the linear relationship between the dependent and independent variables.
* **Small or no multicollinearity between the features:**  
  Multicollinearity means high-correlation between the independent variables. Due to multicollinearity, it may difficult to find the true relationship between the predictors and target variables. Or we can say, it is difficult to determine which predictor variable is affecting the target variable and which is not. So, the model assumes either little or no multicollinearity between the features or independent variables.
* **Homoscedasticity Assumption:**  
  Homoscedasticity is a situation when the error term is the same for all the values of independent variables. With homoscedasticity, there should be no clear pattern distribution of data in the scatter plot.
* **Normal distribution of error terms:**  
  Linear regression assumes that the error term should follow the normal distribution pattern. If error terms are not normally distributed, then confidence intervals will become either too wide or too narrow, which may cause difficulties in finding coefficients.  
  It can be checked using the **q-q plot**. If the plot shows a straight line without any deviation, which means the error is normally distributed.
* **No autocorrelations:**  
  The linear regression model assumes no autocorrelation in error terms. If there will be any correlation in the error term, then it will drastically reduce the accuracy of the model. Autocorrelation usually occurs if there is a dependency between residual errors.

# Simple Linear Regression in Machine Learning

Simple Linear Regression is a type of Regression algorithms that models the relationship between a dependent variable and a single independent variable. The relationship shown by a Simple Linear Regression model is linear or a sloped straight line, hence it is called Simple Linear Regression.

The key point in Simple Linear Regression is that the **dependent variable must be a continuous/real value**. However, the independent variable can be measured on continuous or categorical values.

Simple Linear regression algorithm has mainly two objectives:

* **Model the relationship between the two variables.** Such as the relationship between Income and expenditure, experience and Salary, etc.
* **Forecasting new observations.** Such as Weather forecasting according to temperature, Revenue of a company according to the investments in a year, etc.

## Simple Linear Regression Model:

The Simple Linear Regression model can be represented using the below equation:

y= a0+a1x+ ε

Where,

**a0= It is the intercept of the Regression line (can be obtained putting x=0)**  
**a1= It is the slope of the regression line, which tells whether the line is increasing or decreasing.**  
**ε = The error term. (For a good model it will be negligible)**

Multiple Linear Regression

In the previous topic, we have learned about Simple Linear Regression, where a single Independent/Predictor(X) variable is used to model the response variable (Y). But there may be various cases in which the response variable is affected by more than one predictor variable; for such cases, the Multiple Linear Regression algorithm is used.

Moreover, Multiple Linear Regression is an extension of Simple Linear regression as it takes more than one predictor variable to predict the response variable. We can define it as:

*Multiple Linear Regression is one of the important regression algorithms which models the linear relationship between a single dependent continuous variable and more than one independent variable.*

Prediction of CO2 emission based on engine size and number of cylinders in a car.

**Some key points about MLR:**

* For MLR, the dependent or target variable(Y) must be the continuous/real, but the predictor or independent variable may be of continuous or categorical form.
* Each feature variable must model the linear relationship with the dependent variable.
* MLR tries to fit a regression line through a multidimensional space of data-points.

### MLR equation:

In Multiple Linear Regression, the target variable(Y) is a linear combination of multiple predictor variables x1, x2, x3, ...,xn. Since it is an enhancement of Simple Linear Regression, so the same is applied for the multiple linear regression equation, the equation becomes:

1. Y= b<sub>0</sub>+b<sub>1</sub>x<sub>1</sub>+ b<sub>2</sub>x<sub>2</sub>+ b<sub>3</sub>x<sub>3</sub>+...... bnxn       ............... (a)

Where,

**Y= Output/Response variable**

**b0, b1, b2, b3 , bn....= Coefficients of the model.**

**x1, x2, x3, x4,...= Various Independent/feature variable**

### Assumptions for Multiple Linear Regression:

* A **linear relationship** should exist between the Target and predictor variables.
* The regression residuals must be **normally distributed**.
* MLR assumes little or **no multicollinearity** (correlation between the independent variable) in data.

# ML Polynomial Regression

* Polynomial Regression is a regression algorithm that models the relationship between a dependent(y) and independent variable(x) as nth degree polynomial. The Polynomial Regression equation is given below:

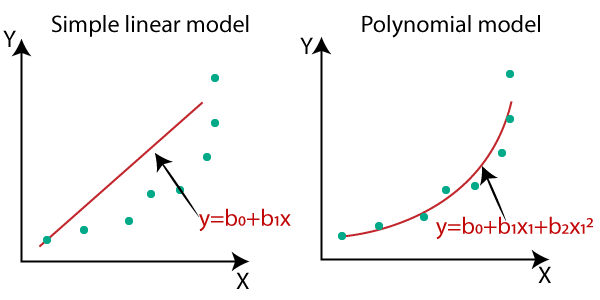
y= b0+b1x1+ b2x12+ b2x13+...... bnx1n

* It is also called the special case of Multiple Linear Regression in ML. Because we add some polynomial terms to the Multiple Linear regression equation to convert it into Polynomial Regression.
* It is a linear model with some modification in order to increase the accuracy.
* The dataset used in Polynomial regression for training is of non-linear nature.
* It makes use of a linear regression model to fit the complicated and non-linear functions and datasets.
* **Hence, "In Polynomial regression, the original features are converted into Polynomial features of required degree (2,3,..,n) and then modeled using a linear model."**

## Need for Polynomial Regression:

The need of Polynomial Regression in ML can be understood in the below points:

* If we apply a linear model on a **linear dataset**, then it provides us a good result as we have seen in Simple Linear Regression, but if we apply the same model without any modification on a **non-linear dataset**, then it will produce a drastic output. Due to which loss function will increase, the error rate will be high, and accuracy will be decreased.
* So for such cases, **where data points are arranged in a non-linear fashion, we need the Polynomial Regression model**. We can understand it in a better way using the below comparison diagram of the linear dataset and non-linear dataset.



#### Note: A Polynomial Regression algorithm is also called Polynomial Linear Regression because it does not depend on the variables, instead, it depends on the coefficients, which are arranged in a linear fashion.

## Equation of the Polynomial Regression Model:

**Simple Linear Regression equation:         y = b0+b1x         .........(a)**

**Multiple Linear Regression equation:         y= b0+b1x+ b2x2+ b3x3+....+ bnxn**

**Polynomial Regression equation:         y= b0+b1x + b2x2+ b3x3+....+ bnxn  ..........(c)**

When we compare the above three equations, we can clearly see that all three equations are Polynomial equations but differ by the degree of variables. The Simple and Multiple Linear equations are also Polynomial equations with a single degree, and the Polynomial regression equation is Linear equation with the nth degree. So if we add a degree to our linear equations, then it will be converted into Polynomial Linear equations.

## Gradient Descent

Gradient descent is an optimization algorithm used to find the values of parameters (coefficients) of a function (f) that minimizes a cost function (cost).

Gradient descent is best used when the parameters cannot be calculated analytically (e.g. using linear algebra) and must be searched for by an optimization algorithm.

### Gradient Descent Procedure

The procedure starts off with initial values for the coefficient or coefficients for the function. These could be 0.0 or a small random value.

coefficient = 0.0

The cost of the coefficients is evaluated by plugging them into the function and calculating the cost.

cost = f(coefficient)

or

cost = evaluate(f(coefficient))

The derivative of the cost is calculated. The derivative is a concept from calculus and refers to the slope of the function at a given point. We need to know the slope so that we know the direction (sign) to move the coefficient values in order to get a lower cost on the next iteration.

delta = derivative(cost)

Now that we know from the derivative which direction is downhill, we can now update the coefficient values. A [learning rate parameter](https://machinelearningmastery.com/learning-rate-for-deep-learning-neural-networks/) (alpha) must be specified that controls how much the coefficients can change on each update.

coefficient = coefficient – (alpha \* delta)

This process is repeated until the cost of the coefficients (cost) is 0.0 or close enough to zero to be good enough.

You can see how simple gradient descent is. It does require you to know the gradient of your cost function or the function you are optimizing, but besides that, it’s very straightforward. Next we will see how we can use this in machine learning algorithms.

## Batch Gradient Descent for Machine Learning

The goal of all supervised machine learning algorithms is to best estimate a target function (f) that maps input data (X) onto output variables (Y). This describes all classification and regression problems.

Some machine learning algorithms have coefficients that characterize the algorithms estimate for the target function (f). Different algorithms have different representations and different coefficients, but many of them require a process of optimization to find the set of coefficients that result in the best estimate of the target function.

Common examples of algorithms with coefficients that can be optimized using gradient descent are Linear Regression and Logistic Regression.

The evaluation of how close a fit a machine learning model estimates the target function can be calculated a number of different ways, often specific to the machine learning algorithm. The cost function involves evaluating the coefficients in the machine learning model by calculating a prediction for the model for each training instance in the dataset and comparing the predictions to the actual output values and calculating a sum or average error (such as the Sum of Squared Residuals or SSR in the case of linear regression).

From the cost function a derivative can be calculated for each coefficient so that it can be updated using exactly the update equation described above.

The cost is calculated for a machine learning algorithm over the entire training dataset for each iteration of the gradient descent algorithm. One iteration of the algorithm is called one batch and this form of gradient descent is referred to as batch gradient descent.

Batch gradient descent is the most common form of gradient descent described in machine learning.

## Stochastic Gradient Descent for Machine Learning

Gradient descent can be slow to run on very large datasets.

Because one iteration of the gradient descent algorithm requires a prediction for each instance in the training dataset, it can take a long time when you have many millions of instances.

In situations when you have large amounts of data, you can use a variation of gradient descent called stochastic gradient descent.

In this variation, the gradient descent procedure described above is run but the update to the coefficients is performed for each training instance, rather than at the end of the batch of instances.

The first step of the procedure requires that the order of the training dataset is randomized. This is to mix up the order that updates are made to the coefficients. Because the coefficients are updated after every training instance, the updates will be noisy jumping all over the place, and so will the corresponding cost function. By mixing up the order for the updates to the coefficients, it harnesses this random walk and avoids it getting distracted or stuck.

The update procedure for the coefficients is the same as that above, except the cost is not summed over all training patterns, but instead calculated for one training pattern.

The learning can be much faster with stochastic gradient descent for very large training datasets and often you only need a small number of passes through the dataset to reach a good or good enough set of coefficients, e.g. 1-to-10 passes through the dataset.

# kernel function

**So why Kernels?**

Consider the Fig. 3 below

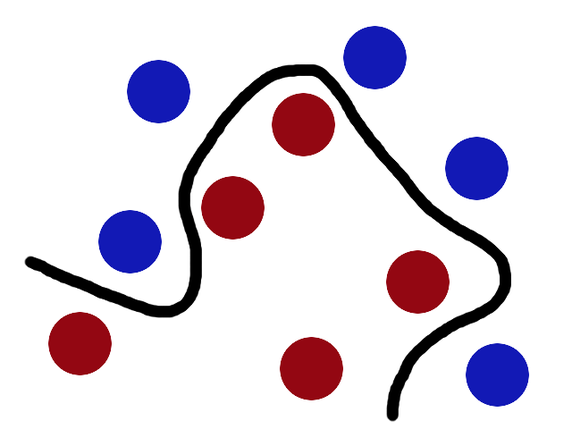


Fig. 3

Can you try to solve the above problem linearly like we did with Fig. 2?

NO!

The red and blue balls cannot be separated by a straight line as they are randomly distributed and this, in reality, is how most real life problem data are -**randomly distributed.**

In machine learning, a “kernel” is usually used to refer to the kernel trick, a method of using a linear classifier to solve a non-linear problem. It entails transforming linearly inseparable data like (Fig. 3) to linearly separable ones (Fig. 2). The kernel function is what is applied on each data instance to map the original non-linear observations into a higher-dimensional space in which they become separable.

Using the dog breed prediction example again, kernels offer a better alternative. Instead of defining a slew of features, you define a single kernel function to compute **similarity** between breeds of dog. You provide this kernel, together with the data and labels to the learning algorithm, and out comes a classifier.

# How does it work?

To better understand how Kernels work, let us use Lili Jiang’s mathematical [illustration](https://www.quora.com/What-are-Kernels-in-Machine-Learning-and-SVM/answer/Lili-Jiang?srid=oOgT)

***Mathematical definition****: K(x, y) = <f(x), f(y)>. Here K is the kernel function, x, y are n dimensional inputs. f is a map from n-dimension to m-dimension space. < x,y> denotes the dot product. usually m is much larger than n.*

***Intuition****: normally calculating <f(x), f(y)> requires us to calculate f(x), f(y) first, and then do the dot product. These two computation steps can be quite expensive as they involve manipulations in m dimensional space, where m can be a large number. But after all the trouble of going to the high dimensional space, the result of the dot product is really a scalar: we come back to one-dimensional space again! Now, the question we have is: do we really need to go through all the trouble to get this one number? do we really have to go to the m-dimensional space? The answer is no, if you find a clever kernel.*

***Simple Example:****x = (x1, x2, x3); y = (y1, y2, y3). Then for the function f(x) = (x1x1, x1x2, x1x3, x2x1, x2x2, x2x3, x3x1, x3x2, x3x3), the kernel is K(x, y ) = (<x, y>)².*

*Let’s plug in some numbers to make this more intuitive: suppose x = (1, 2, 3); y = (4, 5, 6). Then:  
f(x) = (1, 2, 3, 2, 4, 6, 3, 6, 9)  
f(y) = (16, 20, 24, 20, 25, 30, 24, 30, 36)  
<f(x), f(y)> = 16 + 40 + 72 + 40 + 100+ 180 + 72 + 180 + 324 = 1024*

*A lot of algebra, mainly because f is a mapping from 3-dimensional to 9 dimensional space.*

*Now let us use the kernel instead:  
K(x, y) = (4 + 10 + 18 ) ^2 = 32² = 1024  
Same result, but this calculation is so much easier.*

In machine learning, a “kernel” is usually used to refer to the kernel trick, a method of using a linear classifier to solve a non-linear problem. It entails transforming linearly inseparable data like (Fig. 3) to linearly separable ones (Fig. 2). The kernel function is what is applied on each data instance to map the original non-linear observations into a higher-dimensional space in which they become separable.

Using the dog breed prediction example again, kernels offer a better alternative. Instead of defining a slew of features, you define a single kernel function to compute **similarity** between breeds of dog. You provide this kernel, together with the data and labels to the learning algorithm, and out comes a classifier.

# How does it work?

To better understand how Kernels work, let us use Lili Jiang’s mathematical [illustration](https://www.quora.com/What-are-Kernels-in-Machine-Learning-and-SVM/answer/Lili-Jiang?srid=oOgT)

***Mathematical definition****: K(x, y) = <f(x), f(y)>. Here K is the kernel function, x, y are n dimensional inputs. f is a map from n-dimension to m-dimension space. < x,y> denotes the dot product. usually m is much larger than n.*

***Intuition****: normally calculating <f(x), f(y)> requires us to calculate f(x), f(y) first, and then do the dot product. These two computation steps can be quite expensive as they involve manipulations in m dimensional space, where m can be a large number. But after all the trouble of going to the high dimensional space, the result of the dot product is really a scalar: we come back to one-dimensional space again! Now, the question we have is: do we really need to go through all the trouble to get this one number? do we really have to go to the m-dimensional space? The answer is no, if you find a clever kernel.*

***Simple Example:****x = (x1, x2, x3); y = (y1, y2, y3). Then for the function f(x) = (x1x1, x1x2, x1x3, x2x1, x2x2, x2x3, x3x1, x3x2, x3x3), the kernel is K(x, y ) = (<x, y>)².*

*Let’s plug in some numbers to make this more intuitive: suppose x = (1, 2, 3); y = (4, 5, 6). Then:  
f(x) = (1, 2, 3, 2, 4, 6, 3, 6, 9)  
f(y) = (16, 20, 24, 20, 25, 30, 24, 30, 36)  
<f(x), f(y)> = 16 + 40 + 72 + 40 + 100+ 180 + 72 + 180 + 324 = 1024*

*A lot of algebra, mainly because f is a mapping from 3-dimensional to 9 dimensional space.*

*Now let us use the kernel instead:  
K(x, y) = (4 + 10 + 18 ) ^2 = 32² = 1024  
Same result, but this calculation is so much easier.*

In machine learning, a “kernel” is usually used to refer to the kernel trick, a method of using a linear classifier to solve a non-linear problem. It entails transforming linearly inseparable data like (Fig. 3) to linearly separable ones (Fig. 2). The kernel function is what is applied on each data instance to map the original non-linear observations into a higher-dimensional space in which they become separable.

Using the dog breed prediction example again, kernels offer a better alternative. Instead of defining a slew of features, you define a single kernel function to compute **similarity** between breeds of dog. You provide this kernel, together with the data and labels to the learning algorithm, and out comes a classifier.

# How does it work?

To better understand how Kernels work, let us use Lili Jiang’s mathematical [illustration](https://www.quora.com/What-are-Kernels-in-Machine-Learning-and-SVM/answer/Lili-Jiang?srid=oOgT)

***Mathematical definition****: K(x, y) = <f(x), f(y)>. Here K is the kernel function, x, y are n dimensional inputs. f is a map from n-dimension to m-dimension space. < x,y> denotes the dot product. usually m is much larger than n.*

***Intuition****: normally calculating <f(x), f(y)> requires us to calculate f(x), f(y) first, and then do the dot product. These two computation steps can be quite expensive as they involve manipulations in m dimensional space, where m can be a large number. But after all the trouble of going to the high dimensional space, the result of the dot product is really a scalar: we come back to one-dimensional space again! Now, the question we have is: do we really need to go through all the trouble to get this one number? do we really have to go to the m-dimensional space? The answer is no, if you find a clever kernel.*

***Simple Example:****x = (x1, x2, x3); y = (y1, y2, y3). Then for the function f(x) = (x1x1, x1x2, x1x3, x2x1, x2x2, x2x3, x3x1, x3x2, x3x3), the kernel is K(x, y ) = (<x, y>)².*

*Let’s plug in some numbers to make this more intuitive: suppose x = (1, 2, 3); y = (4, 5, 6). Then:  
f(x) = (1, 2, 3, 2, 4, 6, 3, 6, 9)  
f(y) = (16, 20, 24, 20, 25, 30, 24, 30, 36)  
<f(x), f(y)> = 16 + 40 + 72 + 40 + 100+ 180 + 72 + 180 + 324 = 1024*

*A lot of algebra, mainly because f is a mapping from 3-dimensional to 9 dimensional space.*

*Now let us use the kernel instead:  
K(x, y) = (4 + 10 + 18 ) ^2 = 32² = 1024  
Same result, but this calculation is so much easier.*

### Locally Weighted Linear Regression:

<https://www.geeksforgeeks.org/ml-locally-weighted-linear-regression/> 