ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

Eduonix Learning Solutions

Better Learning Everyday

Contents

What is Artificial Intelligence and why is there is so much hype around it?

Introduction

Every decade had its fair share of technological advancements which was enough to generate hype around it and generate waves of revolution in the industry. Cloud computing, cryptocurrency, and data science all had their fair shares of hype in recent years due to the rapid advancements. Artificial Intelligence is touted to be the next big revolution in the industry due to its vast applications and wider adoption by big players in the industry. Enterprises are cashing in big money for this technology as they are confident enough that it will stimulate economic growth, intensify global health, enhance cyber-security and enable better options in terms of education.

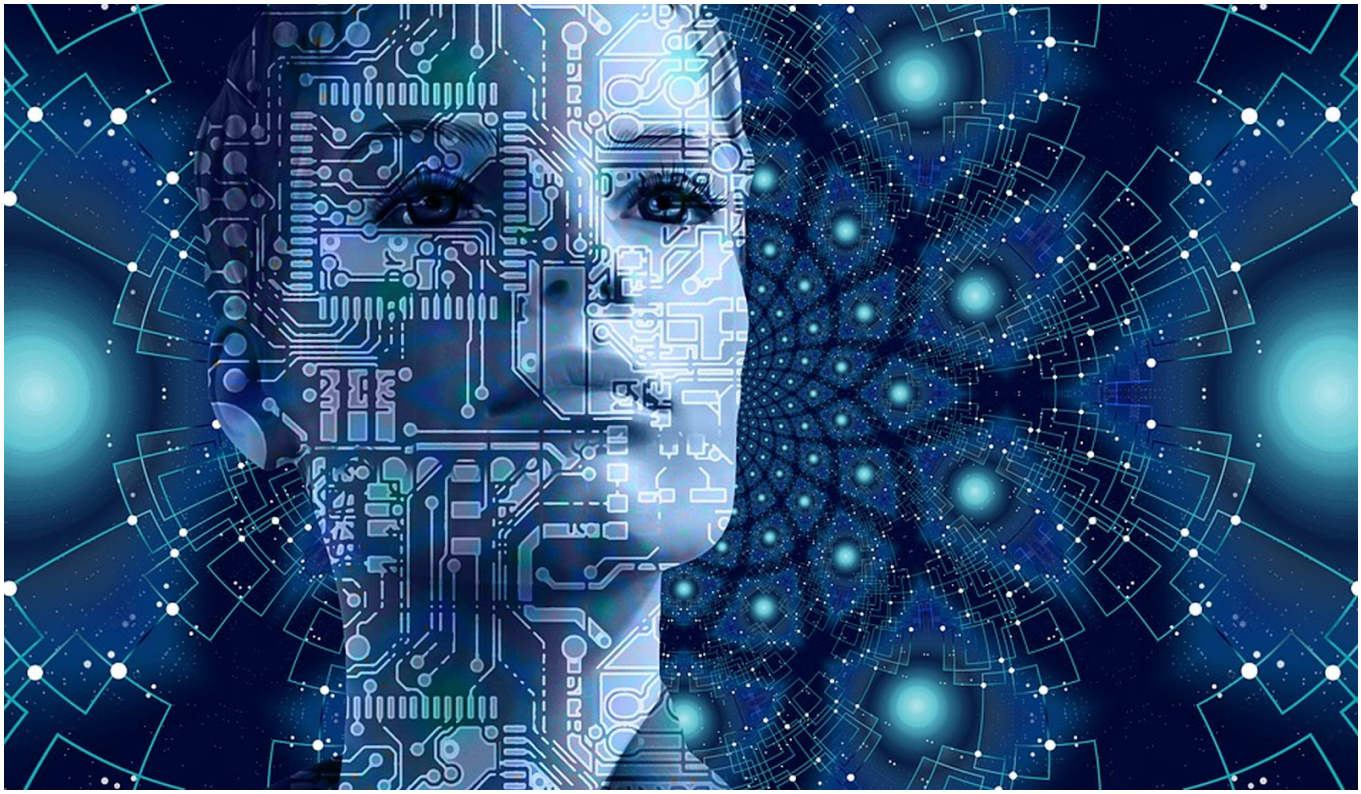
AI is more than just a fad. It delivers a whole new direction of polarising your business considering the current requirements and by automating the existing business resources. AI is a worthy extension to computer science which already included algorithms that defined how machines function and perform their assigned tasks on a daily basis. The ever-growing amount of data and its continuous exploitation has led companies to adopt AI in their systems and make the most of it to understand the current business trends and predict what business would demand in the near future.

History

The idea of simulation was earlier suggested by Alan Turing, who came up with a theory that, any machine can perform the simulation of any conceivable act or mathematical deduction by continuously shuffling the symbols 0 and 1. This was termed as the Church-Turing thesis. At that time, researches were of the opinion of creating an electronic brain which supported this thesis. There were numerous inventions in neurobiology, information technology and cybernetics using this thesis.

Originally, the research in the field of AI began in 1956 in a workshop. Leaders from MIT and IBM became the founders and the driving force behind this research. They were working on some programs related to AI which were both astonishing and ground-breaking during that period. Computers were programmed to learn different algorithms. They were taught problems in algebra and were showing minor signs of working out basic logical theorems in mathematics. The ones who were investing in AI during the 60s were optimistic of its application in the future. They had the opinion that in twenty years down the lane, machines will be able to perform the tasks that a man can perform now. However, criticism from many government bodies made them realize the lack of availability of resources for artificial intelligence at that time.

Groundwork



The effectiveness of artificial intelligence depends on its overall learning and adoption to the algorithm that is written in the system. Unlike machine learning which revolves around supervised and unsupervised learning, artificial intelligence is guided by reinforcement learning. This has a pre-defined set of rules and guidelines that a machine has to learn and follow in the very beginning to understand the involved entities and their characteristics. Slowly and gradually, this learning process continues until the machine is capable enough to perform it without any external guidance. This cycle repeats and at times, machines might learn some new techniques or algorithms that were not fed or programmed in their memory due to continuous machine learning taking place.

Enterprises have led the basic foundation for AI by understanding its capability and effectiveness for their business solutions. Some term it as a "magic wand which can eventually turn things around" and some envision it as "a journey that will take time to come around as it will learn from its surroundings". Everyone has their version of a theory regarding AI which makes them think that it will be a game-changer. Just recently, a study revealed that 37% of the enterprises around the globe have successfully adopted artificial intelligence in their businesses. This proves the fact that AI is the current trendsetter and more companies will be looking to deploy it in their systems.

Application of machine learning in AI

Artificial intelligence is basically feeding some knowledge or intelligence in a computer which it can understand and perform. Machine learning algorithms have a plethora of classification and regression algorithms. Decision trees can be used to solve both, simple and complex datasets. Support vector machines can be utilized to predict the cab fares for any area within the domain.Naive Bayes is used for determining complex financial problems and other related issues.Linear regression can help in predicting future stock prices or house prices which can prove helpful for many industrialists.

AI, on the other hand, is deployed to handle such predefined algorithms or tasks in the system which hardly needs any monitoring as AI will get the job done without any external guidance.

The AI-inspired robot-Sophia

Hanson Robotics started working on a human-like robot named Sophia when they saw the boom in the AI industry. It was made public in March 2016. Sophia can imitate 50 human expressions that we encounter in our daily lives. It is the only robot which has valid citizenship! It is a classic example of artificial intelligence when we consider imparting human behavior and emotions in machines which can solve our common day-to-day problems using machine learning and artificial intelligence at its disposal.

Sophia has been designed keeping in mind the previous iterations of humanoids and their features. Its eyes have built-in cameras and pre-programmed algorithms which help it to observe the surroundings.

Its brain has been programmed to be similar to a chat-bot but more inclined towards the human behavior which enables it to listen to questions and answer them. All of its data are collected in a cloud which allows researchers to analyze its responses using blockchain technology.

How AI affects our daily lives

Smart cars

Smart cars have been making quite the appearance in the market where companies like Tesla are looking to make semi-automatic cars in abundance. These cars are programmed with the data that is being collected and monitored by Tesla. All these cars are connected with each other and know where their current location is. This data that is collected by Tesla enables more reliable and fast learning by these cars and the chances of accidents are quite low.

Smartphones

As almost everyone around us owns a smartphone, it's not a surprise that AI has forayed into this market years ago. Manufacturers like Google and Samsung have started integrating their cameras with AI which enables better portrait mode and improved exposure for night pictures. Furthermore, they are trying to integrate AI to increase the battery life and improve the Android Operating System's functionalities which can promote flexibility and user-friendly features.

Banking and Finance

Banking had adopted AI on a large scale for fraud detection, spam filtering, customer service and many more which can benefit its portfolio and enable a healthy cash flow.

Social media
Social media has gained a lot of flak as well as praise for its continuous integration with AI. Facebook and Instagram are the prime examples where all the recommendations, notifications and suggestion are curated by AI to provide the user with a good experience.

Surveillance
Security cameras have data collected in the form of images and videos which can help to understand the pattern of customer behavior or any suspicious activity being carried out. AI can help to explore this data and gain some useful insights for better understanding of the situation.

Smart homes
Many smart homes and smart cities have been developed with the help of AI. These homes function with the help of sensors that collect data and enable AI to learn their daily patterns. It can include adjusting the temperature of the room, playing a person's favorite songs, ordering vegetables and groceries etc.

What is Machine Learning?

Introduction

The current generation is witnessing a paradigm shift where machines are getting smarter and sharper than humans. A chess-playing computer which goes by the name, Deep Blue was designed by IBM to interact and compete with real-time chess champions. It once defeated world champion, Garry Kasparov, in a six-game match which saw Garry losing all the six matches to Deep Blue. The inputs that are taken from this computer while playing with a chess player formulates into an algorithm which Deep Blue defines to compete with these players. This is just one of the applications that we have encountered where machines tend to over-smart humans and are capable to make some bold decisions if the circumstances demand.

Machine Learning has been practiced since the 1970s. This term was coined in 1959. "A computer program is said to learn from experience with respect to some class of tasks and performance measure. Its performance improves with more exposure to the number of tasks". Alan Turing had a fascination towards machine learning. He published a paper named "Computer machinery and intelligence" which discussed the growing inclination towards the machine and whether machines can think like humans or not.

Groundwork

Machine Learning as a concept was derived from statistics as its base. Statistics is the subdivision of mathematics that deals with data collection, categorization, interpretation, and presentation. It basically can handle all phases of data including the planning of data collection in terms of the design of surveys and experiments.

At times, when companies need a profitable business model which can deliver in terms of flexibility and reliability, they are bound to apply machine learning techniques to deliver immense value to the business. While hosting these ML and AI services, the company has to look into some glaring factors which could shape the foundation for these technologies.

Data Infrastructure :
Setting up the right infrastructure means inspecting what is the right amount of data that is available and is necessary to identify trends and fulfill customer requirements. At times, experienced analysts can help the organization to understand what is the right amount of data that will suffice the company's touch points.

Customer comes first :
The customer plays an important role when a company needs to build a strong portfolio and get its services noticed. This, in turn, requires wider visibility and better awareness of the customer and their diverse experiences. This is achievable through mediums like email, ads and computer vision. Getting the right mindset of the customer is important as it helps in better analysis of his choices.

Suitable Framework :
Machine Learning and Artificial Intelligence have the horsepower to capture the existing customer's choices and create an algorithm that takes care of the customer's future demands. AI has the capability of combining all the purchase indicators of the customer and build a generalized framework which will give the insights into the customer's journey. This will lead to greater insights and a path to efficient customer understanding.

Statistics as the base

Statistics deals with data collection, preparation, examination and presentation. Data such as statistical population or statistical model analysis are taken care of with the help of stats. Statisticians were the ones who used to come up with interesting insights and findings using different statistical techniques in data before machine learning became mainstream. They used to undergo the task of assembling the measurements of the system under the radar, manipulating it and in the end, capturing additional measurements of the same system using the same technique to determine if the manipulation has changed the values of the readings or not.

Let's have a look into some important terms in stats which are fundamental in machine learning.

Mean

Mean is the arithmetic average of data values. It comes under the central tendency of stats. Mean takes into consideration all the values for computation.

Median

Median is the central observation when all observations are arranged in order of magnitude. It is an important measure of central tendency. Median is the observation that divides the series into 2 equal halves.

Skewness

Skewness illustrates the level of degree a set of data varies when compared to standard distribution in a set of statistical data. Its resultant curve depends on the data, whether the data is having a positive or negative skew towards the data average.

Kurtosis
Kurtosis measures the data for light-tailed which means less outlier prone or heavy-tailed when compared to the normal distribution.

Variance
Variance is the measurement of the span of numbers in a dataset. It is also the square of standard deviation.

Getting started

Machine learning has taken the market by storm and there is a varied interest in individuals from different backgrounds that want to get into this field. Nowadays, IoT devices, sensors, and CCTV cameras are all used to collect data and provide some useful insights to learn new findings and explore new areas of improvement for the company. There are some basic fundamental steps that are needed to learn in order to get a hang of the machine learning algorithms.

Supervised Learning
This learning includes algorithms that are designed based on a target variable which is to be predicted or calculated with the help of a given set of independent values also known as predictors. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values.

Unsupervised Learning
This learning includes algorithms that are used for clustering as it has no target variable of the coefficient to predict in the picture. K-means and apriori are prime examples of this learning. Unsupervised learning studies how systems can infer a function to describe a hidden structure from the unlabelled data.

Reinforcement Learning

Reinforcement machine learning algorithms is a learning technique that interacts with its environment by producing actions and discovers errors or rewards. Trial and error search and delayed reward are the most relevant characteristics of reinforcement learning. This enables machines to automatically determine the ideal behavior within a specific context in order to maximize its performance.

Exploring Python as a tool for machine learning

Introduction

Python is a general-purpose, multi-platform and a high-level programming language used to take care of the coding prowess for small and big applications. It has support for both, object-oriented programming and structured programming. It was designed to be highly reliable and easily readable.

Python was created by Guido van Rossum in the 80s when programming languages were gaining momentum in the IT industry. Python has the capability to be interpreted at runtime by its interpreter. It has its own command prompt for users to directly interact with the interpreter. And plus, it is a great language for anyone who is looking to start their career or looking to explore coding knowledge in programming.

Python, as an open-source tool, encapsulated by many scripts and libraries which have been created and nailed over the years since its invent in the market.

Features

Dynamically ridden :
Python is a dynamic language which is easy to execute and takes care of the bigger problems involved in the application by solving it in small bits and pieces to come up with a collective solution.

Adaptable :
Python was designed with the goal of harnessing the various algorithms to its maximum potential with a minimum piece of code. At times, its code can be directly deployed from the cloud for faster execution.

Endless support :
Over the years, the python community has gained a massive number of developers and experts who are constantly pushing new updates and upgrading its database of libraries which help in smooth and faster execution of code for newbies.

Open Source :
Free of cost and easily available online, people are entitled to make changes to its source code and can experiment with its libraries. It has been improved and is consistent over the years due to its loyal community and massive updates being pushed almost every day.

Python libraries
Python has tons of libraries which serves the various coding purposes of all the end users like developers and deep problem-solvers in an organization. Let's study some of them which are being used in the field of machine learning and deep learning.

NumPy :
The most common and fundamental package built around machine learning in python is the Numerical Python package also known as NumPy. It complicates Python by delivering a number of useful operations for addressing arrays and matrices in python. It provides a vector for mathematical operations which help in speedy execution.

Pandas :
Pandas was specifically designed to take care of data wrangling. It also enables easy manipulation and visualization of the given data. It can be used for manipulating one-dimensional and two-dimensional data frames which have complex data to deal with.

Scikit-learn :
Scikit learn takes care of implementing both supervised and unsupervised machine learning algorithms in the pipeline. It depends on basic libraries like Numpy and Pandas and provides enough functions to perform regression and clustering.

Tensorflow :
Tensorflow has been loitering in the industry due to its deep learning capabilities. It has a dedicated function assigned to the GPU of the system for it to utilize it without the intrusion of any third-party code. It has multi-layered nodes that allow the user to quickly train, deploy the deep learning algorithms in the system.

Matplotlib :
There are some libraries that are built in python for users to communicate their story and various findings in layman terms to the general audience. Matplotlib takes care of the visualization part that comes after implementing a machine learning or deep learning model in the system. It represents data in the form of graphs and charts and clearly makes a statement by reducing the stress to understand the given data in deep.

Why Python is preferred by more companies to implement machine learning models?

Over the years, many programming languages have adopted the market trend and have enabled the customization and implementation of various market-related applications which exist in fields like finance, marketing, banking, core IT technology, transportation, military, agriculture, and politics.

Many diverse programming languages like C, C++, C#, Java, Perl, Ruby were adopted and implemented as per market requirements which produced some expected results as well as increased challenges for the business strategists to look for better alternatives in their core domain. Many online repositories like Github started inviting developers to share their code with everyone to ease out the stress caused by difficult lines of code along with increasing productivity and effectiveness of each programming language.

According to a poll conducted by Github in 2017, the second most pulled or used language in their repository to post projects and other related stuff was Python. It was pulled by 1 million people across the globe.

Let us examine some real-world market application of Python that has been widely adopted by its followers and enterprises.

Operating Systems(OS)
Linux OS has python as its backbone for integral deployments and implementations. Ubuntu's Ubiquity, Fedora's, Red Hat Enterprise owned Anaconda are all written in Python language. Many more elements of Linux use Python as its base language. Enterprises are deploying Linux as their primary language after realizing its true potential and future-scope.

Image Processing
Python has been used to create 2D imaging software like Inkscape and Scribus. Furthermore, 3D imaging was made famous by Python with applications like 3DS Max, Houdini, Blender, and Lightwave.

Web Development
Web-related frameworks like Django and Flask are available in Python which is used extensively for web development. They enable to create the back-end code for any server which enables the smooth running of servers.

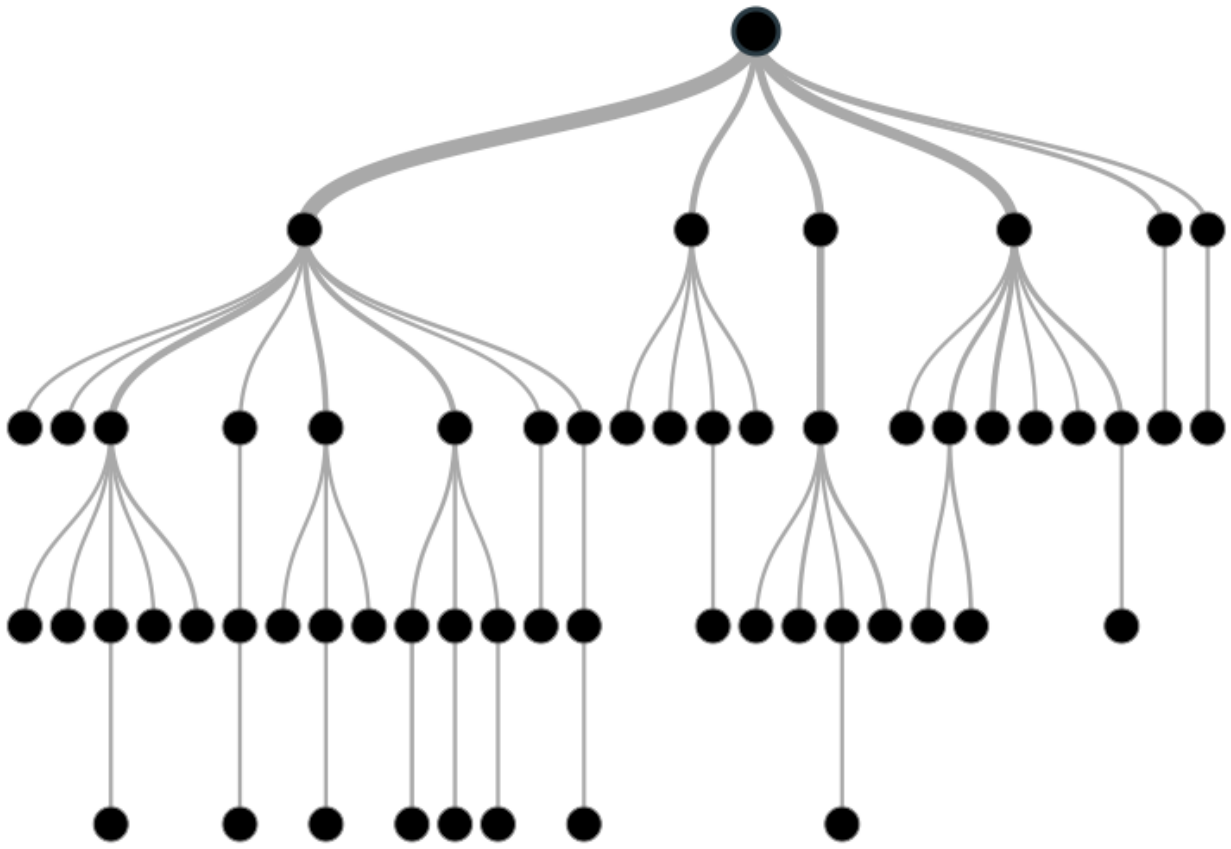
Data Science
Data science includes feature engineering, machine learning, data visualization which help us gain insights from the data. Python in data science can be used in recommendation systems, computer vision, spam filtering, fraud detection, predictive analysis. All of these can be easily achieved with the help of the various libraries that are available in Python.

Classification models in ML

Decision Trees

The most favored and preferred algorithm in classification is the decision tree machine learning algorithm. It comes under supervised learning and is used mostly for classification problems. It can be implemented for both, categorical and continuous input and output variables. There is always a predefined target variable assigned to this problem.

The basic functioning of decision trees goes this way- there are a set of points that are plotted on a plane. These points can't be separated easily by a line due to their heterogeneous properties. Hence, decision trees divide these points into different clusters or leaves based on some predefined conditions and take care of them individually.



A decision tree

Decision Tree encapsulates some mandatory concepts that one needs to know to have a better understanding.

Root Node
It represents the entire population of a given sample and further gets divided into two or more homogeneous sets.

Splitting
It enables the division of a node in the tree into two or more sub-nodes.

Decision Node
This is like sub-node splitting into further sub-nodes in the tree.

Leaf/Terminal Node
These are nodes with zero sub-nodes, that is, these nodes can't be split further.

Pruning
When the size of the decision trees is reduced by removing nodes, the process is called pruning.

Branch/Subtree
A subsection of a decision tree is called as a branch or a sub-tree.

Parent and Child Node
A node which is divided further into small sub-nodes is called a parent node, whereas sub-nodes are the children of this parent node.

Some of the important terms that affect the decision tree algorithm need to be dealt with in order to optimize it.

Gini Impurity
Gini impurity is a measure of the probability for an incorrect classification of a random variable present in the dataset. Its minimum value can be 0 when all cases in the node fail to exist in a single category.
Gini impurity can be classified by the formula.

$G(K) = \Sigma P(i) * (1-p(i))$

Where p(i) is the probability of classification i.

Entropy
Entropy is used to determine the homogeneity of a given sample. If the sample is totally homogeneous then its entropy is 0 and if the sample is equally divided, then its entropy value is 1. In simple terms, it is a measure of impurity. Mathematically, it can be calculated with the help of probability of the items as:

$H= -\Sigma p(x)*\log[p(x)]$

It is the negative summation of probability times the log of the probability of item x.

Information Gain
Information gain is the main ingredient that is conducive in the making of a decision tree. Information gain is used to determine the attribute that will return the highest information gain which can provide the maximum accuracy for the decision trees.

Therefore, IG is equal to entropy(parent) - (average weights) * entropy(children)

Code Snippet

```
# Importing the important libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import cross_val_predict

# Retrieving the dataset
train = pd.read_csv("../input/train.csv", index_col="PassengerId")
test = pd.read_csv("../input/test.csv", index_col = "PassengerId")

# Check for missing values
for i in train.columns:
print("{0} has {1:.2f}% missing data".format(i, (len(train[train[i].isnull()]) /
train.shape[0]) *100))

# Some basic visualization
sns.distplot(train["Age"])
sns.barplot(train["SibSp"],train["Survived"])

# Fitting the Decision Tree Classification model to the data
from sklearn.tree import DecisionTreeClassifier
model =
DecisionTreeClassifier(max_depth=71,max_leaf_nodes=6,min_impurity_d
ecrease=0.002818)
model.fit(x_train, y_train)

# Check for accuracy
y_predict = cross_val_predict(model,x_train,y_train)
accuracy = accuracy_score(y_predict,y_train,)
print("accuracy = {0:.2f}".format(accuracy))
```

Random Forest

Being part of the supervised learning family, random forest reprises a collection of trees to build a forest. This forest includes algorithms that are collectively applied and implemented to create a random machine learning model. Random forest models multiple decision trees and blends them mutually to obtain a more accurate and stable prediction.

Random forests are equivalent to ensemble learning technique for classification and regression techniques. Random forest avoids the problem of overfitting by taking care of the fact that there are enough trees in the model. Another advantage is that the classifier of random forests can easily manage missing values. It can also be modeled for categorical values.

Working

This involves 2 stages where the first one is creating a random forest and the other one is performing predictions and obtaining useful observations from the random forest classifier created in the first stage.

The random forest can be used for both, classification as well as regression problems. At times, a random forest will consist of the same hyperparameters which there are in a decision tree. It has the capability of merging additional randomness to the model. It also scans through all the features to find a particular feature among the random set of features. This leads to the creation of a better and diverse model which will produce maximum accuracy.

Real-World Applications

Stock market

Investing in the stock market is all about tracking the top-performing stocks throughout the quarter and investing in the ones which guarantee higher returns. The random forest can help to track the trend of each stock and decide which one is worth investing for.

Finance companies

By keeping a track of the customers over the years, it can predict whether a customer is loyal to the company or not by tracking its debt repayment patterns.

Code Snippet

```
# Loading basic libraries
import numpy as np
import pandas as pd
Import seaborn as sns
from sklearn.ensemble import
RandomForestClassifier,GradientBoostingClassifier
```

```
# Importing the dataset
titanic_df = pd.read_csv("../input/train.csv")
test_df    = pd.read_csv("../input/test.csv")

# Dropping columns which are not required for our analysis
titanic_df = titanic_df.drop(['PassengerId','Name','Ticket'], axis=1)
test_df    = test_df.drop(['Name','Ticket'], axis=1)

# Some visualizations
sns.factorplot('Embarked','Survived', data=titanic_df,size=4,aspect=3)
fig, (axis1,axis2,axis3) = plt.subplots(1,3,figsize=(15,5))
sns.countplot(x='Embarked', data=titanic_df, ax=axis1)
sns.countplot(x='Survived', hue="Embarked", data=titanic_df, order=[1,0],
ax=axis2)

# Defining the training and test set
X_train = titanic_df.drop("Survived",axis=1)
Y_train = titanic_df["Survived"]
X_test  = test_df.drop("PassengerId",axis=1).copy()

# Fitting the random forest algorithm to the dataset
random_forest =
RandomForestClassifier(n_estimators=100,oob_score=True,max_features=
5)
random_forest.fit(X_train, Y_train)
Y_pred = random_forest.predict(X_test)
random_forest.score(X_train, Y_train)
```

Naive Bayes

Naive Bayes can be considered one of the simplest yet powerful algorithms in classification modeling. It is also being used for natural language processing problems (NLP) along with machine learning use cases. It comes from an algorithmic family of probability theory and takes advantage of probability calculations to predict a suitable label to a text which is specifically used for tagging categories in newspaper articles. The only technique to find the probabilities is by utilizing Bayes theorem which showcases the probability of a feature which is related to some pre-defined conditions of a feature present in the dataset.

The Bayes classifier can be shown by the formula.
$P(c/x) = P(x/c)*P(x) / P(x)$
where, P(c|x) is called the probability of target predictor where x is the feature, P(c) is known prior probability of class, P(x|c) is the likelihood, which is the probability of predictor given class and P(x) is the prior probability of predictor.

We can do many useful tweaks and improvements to our Naive Bayes model. Let us shed some light and understand if it can help to improve text mining to any extent.

Removing stopwords

Some general letters or words like and, or, a, else, and some common fillers are discarded from the text in order to get a more deep insight into the text.

Lemmatizing

This is used to group the same words like department, departments, departmental and so on to form a common group of words to reduce their count.

Removing punctuation

Punctuation marks like !, @, #, &, ? are taken care of in order to reduce the use of symbols and gain more control of the text.

Designing TF-IDF

TF-IDF stands for term frequency-inverse document frequency which is like a data frame used to store the processed text in a single variable to find insights and gain some knowledge out of it.

Code Snippet

```
# Importing basic libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

# load the training and test data sets
url1 = "../input/train.csv"
train = pd.read_csv(url1);
# check the data set
print("\n"); print(train.drop(["Name"], axis=1).head(3).to_string()); print("\n");
url2 = "../input/test.csv"
test = pd.read_csv(url2);
df_train = simple_transform(train)
df_test = simple_transform(test)

# Some basic EDA
train.Age = train['Age'].apply(age_class)
fig = sns.factorplot(x="Age", y="Survived",data=train,
                size=4,aspect=3, kind="bar", palette="muted")
plt.title("The effect of the feature 'Age' on survival")
plt.ylabel("Fraction Survived", FontSize=12)
plt.show()
```

```
# Train the dataset with Naive Bayes
from sklearn.naive_bayes import MultinomialNB
mnb = MultinomialNB(alpha=1)
mnb.fit(train_mat,target_feature)
print("\n"); print("The fraction of correctly predicted outcome from the test
data set");
print(mnb.score(train_mat, target_feature)); print("\n");

# Data visualization to display our results
fig = sns.factorplot(x="Salutation", y="Survived", data=train,
                size=3, aspect=6, kind="bar", palette="muted")
plt.title("The effect of the feature 'Salutation' on survival")
plt.ylabel("Fraction Survived", FontSize=12)
plt.show()
```

K-Nearest Neighbors

K nearest neighbors algorithm is a part of classification algorithms that classify a new case based on a similarity measure with the help of distance measures. It first traverses through all the available cases and tries to figure out new cases based on the features of similar behavior or characteristic. Its applications include in pattern recognition, data mining and spam detection. It is also known as the lazy learning algorithm.

KNN includes k which is the count of nearest neighbors. This value has to be decided depending on the type of output needed. It can be either 5 or 2 depending on the number of cases required which depends on the number of data points given. We need to calculate the distance between these data points for which we require some measures which are as follows.

Euclidean Distance

Euclidean distance is the square root of the sum of the squared differences between a new point (x) and an existing point (y).

$ED= \sqrt{\Sigma(x^2-y^2)}$

Manhattan Distance

Manhattan distance is the mode of the distance between vectors using the sum of their absolute difference.

$MD= \Sigma|x-y|$

Hamming Distance

If the value (x) and the value (y) are same, the distance D will be equivalent to zero.

$HD= \Sigma|x-y|$

Where $x=y$ when $D=0$ and $x \neq y$ when $D=1$

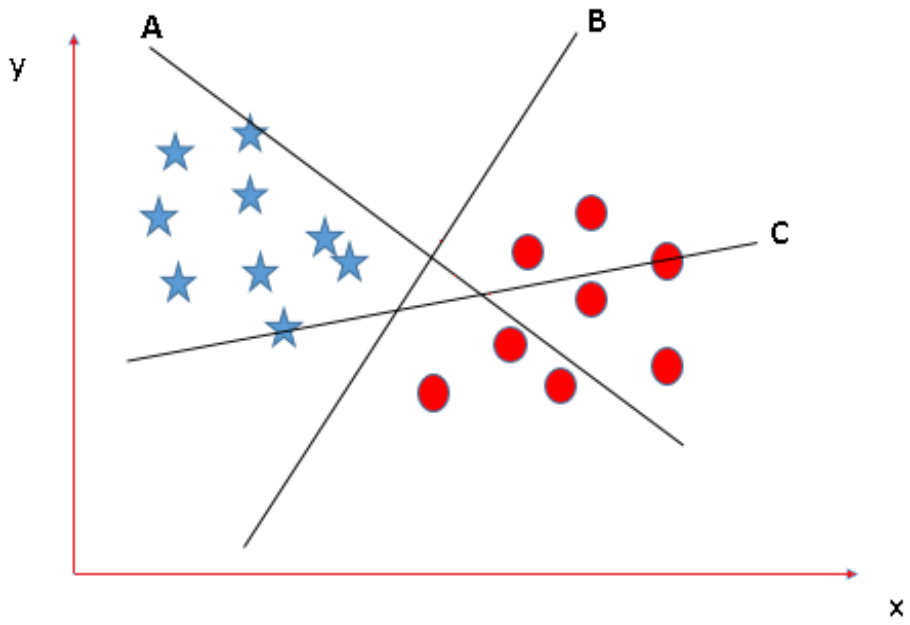
Determining the correct value for k is done by going through the data. If a large value for k is taken into consideration, then it can reduce the overall disturbance at times but it also might not affect the algorithm to any extent. Another way of determining a suitable k value is by applying Cross-validation on an independent dataset. Technically, the optimal k value for the majority of datasets is between 3 and 10. Taking these values for k can surely bring out the best from the dataset.

Code Snippet

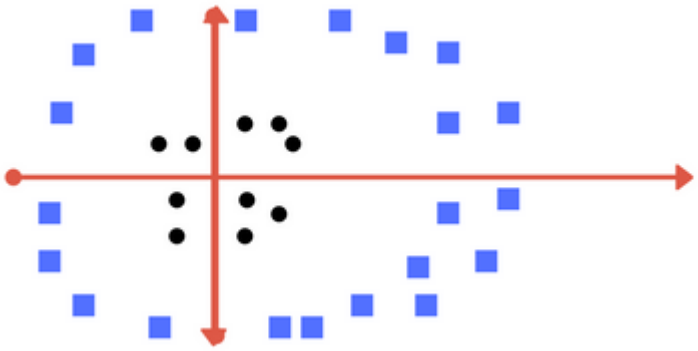
```
import numpy as np
import pandas as pd
dataset=pd.read_csv('../input/train.csv')
testset=pd.read_csv('../input/test.csv')
label=dataset.iloc[0:890,1]
data=dataset.iloc[0:890,[2,4,5]]
testdat=testset.iloc[0:418,[1,3,4]]
x=[data,testdat]
data=(data.fillna(0)) #filling NA values
testdat=testdat.fillna(0)
from sklearn.model_selection import train_test_split
train_data,test_data,train_labels,test_labels=train_test_split(data,label,rando
m_state=7,train_size=0.7)
from sklearn.neighbors import KNeighborsClassifier
clf=KNeighborsClassifier()
clf.fit(data,label)
predictions=clf.predict(test_data)
from sklearn.metrics import accuracy_score
print(accuracy_score(test_labels,predictions))
result=clf.predict(testdat)
print(result)
```

Support Vector Machines

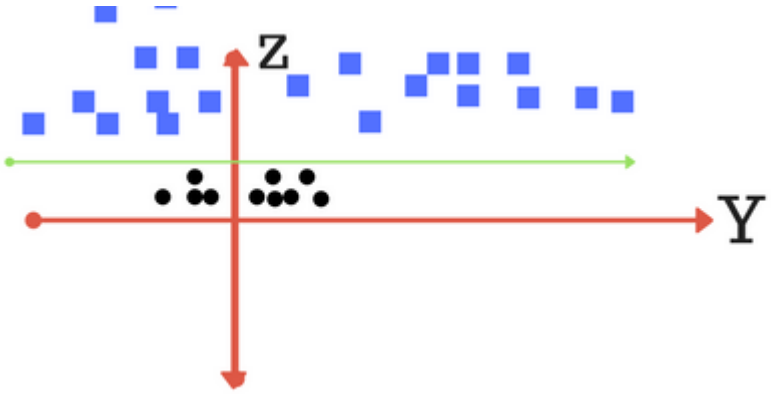
Support Vector Machines, in short, SVMs are used to regroup the available data points in a plane with the help of a hyperplane in a supervised learning environment. Let us assume that in our dataset there are some random features which need to be classified into two different classes for simplicity. These features are plotted in a two-dimensional plane with x and y-axis. Now the challenge is, how will SVM segregate these points and where will the hyperplane in the plane exist?



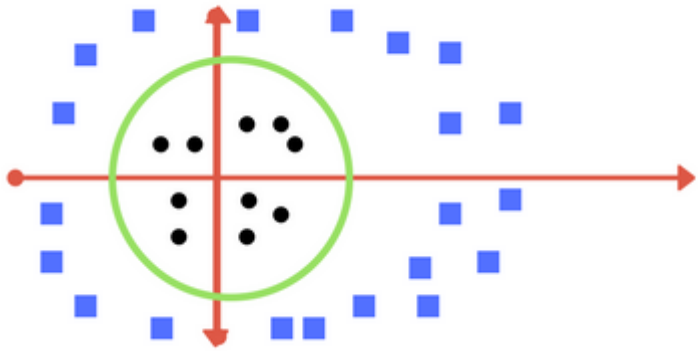
Let's get deep into SVM. At times, we need to take care of data points that are not close to each other. As you can see in the figure below, the blue data points are away from the black ones. In the case, there is clearly no hyperplane that can separate these two classes in the X-Y plane.



In this case, we will introduce one more axis known as the Z-axis. We will assume the value of points on z plane, w=x^2 + y^2. After plotting the z-axis, it is evident that there is a division between these classes.



If we try to change this plane to its original form, it will map to a circular boundary as shown in the figure below. These transformations are known as kernels



There are many functions in SVM which can be used to tune the parameters. Let's have a look.

Kernel
Linear algebra plays a crucial role in transforming the problem which enables to learn and understand the hyperplane in linear SVM. The kernel has an important role in this part. For a kernel, the prediction equation for a new input using the dot product between the input(a) and each support vector(b) is calculated using the following equation.

f(x) = B(0) + sum(ai * (a,b))

Coefficients B(0) and ai for each input from the above equation need to be estimated from the training data by the algorithm. Polynomial and exponential kernels estimate the separation line in the higher dimensions. This is known as kernel trick.

Regularization
Regularization parameter guides the SVM optimization in order to avoid the measure of misclassification for each training sample. It is also referred to as the C parameter in python's sk-learn library. For large values of C, the algorithm meant for optimization will select a smaller-margin hyperplane if it performs a better job of getting all the training points assigned precisely. On the other hand, small C value will lead the optimizer to look for a different hyperplane even if it misclassifies more points.

Code Snippet

```
import pandas as pd
from pandas import Series,DataFrame
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style('whitegrid')
from sklearn.svm import SVC, LinearSVC
titanic_df = pd.read_csv("../input/train.csv")
test_df    = pd.read_csv("../input/test.csv")
titanic_df = titanic_df.drop(['PassengerId','Name','Ticket'], axis=1)
test_df    = test_df.drop(['Name','Ticket'], axis=1)
titanic_df['Fare'].plot(kind='hist', figsize=(15,3),bins=100, xlim=(0,50))
avgerage_fare.index.names = std_fare.index.names = ["Survived"]
avgerage_fare.plot(yerr=std_fare,kind='bar',legend=False)
svc = SVC()
svc.fit(X_train, Y_train)
Y_pred = svc.predict(X_test)
svc.score(X_train, Y_train)
```

Logistic Regression

One of the few common algorithms that are used to predict output in classification problems is the logistic regression. It is incremental in describing the structure of data and helps in determining the correlation between a dependent binary variable and one or more nominal independent variables. It is generally used when the results are expected in binary form i.e.0 and 1.

Logistic regression calculates the probability of an outcome that can only have two values either yes/no or 0/1. This calculation depends on the application of one or more predictors which can be numerical and categorical. As a linear regression is not relevant for predicting the value of a binary variable, logistic regression plays an important role in this scenario.

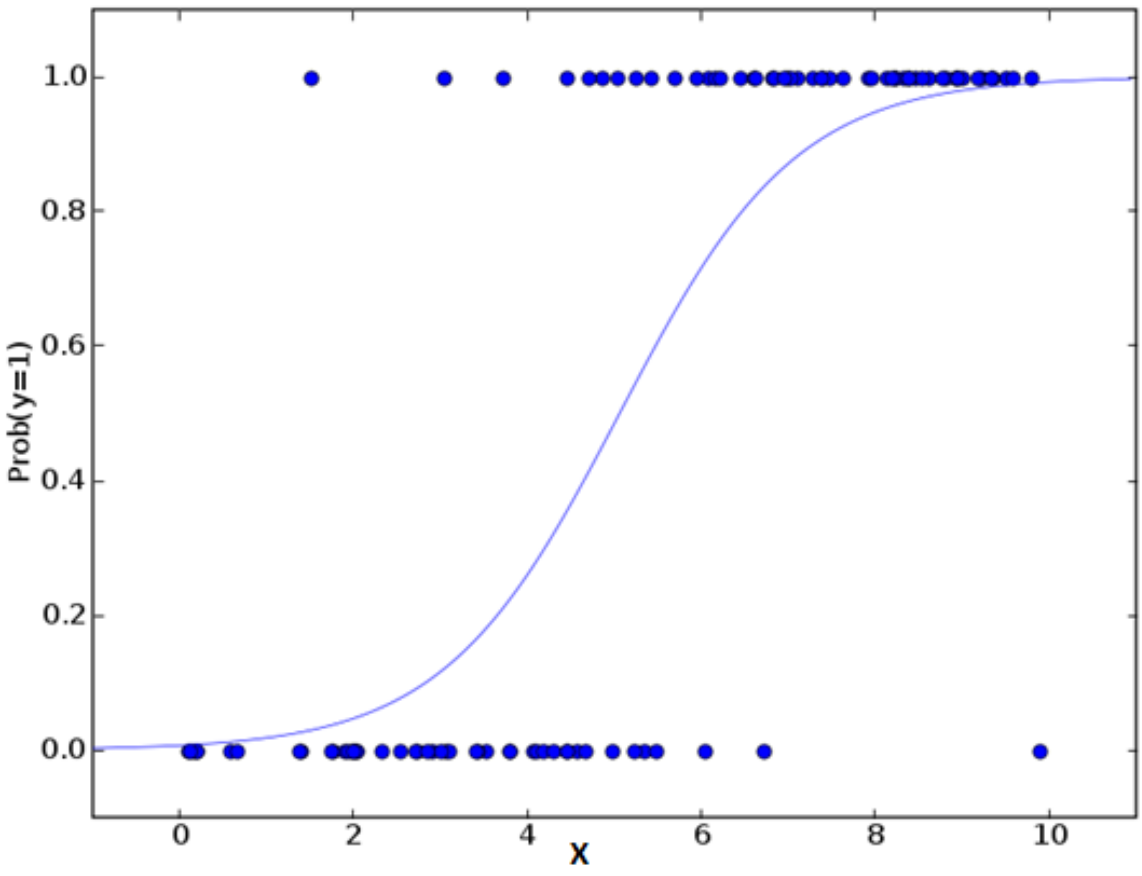
Regression can be distinguished by probabilities of the following events-

Odds = p/(1-p) = probability of event occurring/probability of event not occurring

Ln (odds) = ln (p/(1-p))

Logit (p) = ln (p/(1-p))

In this, (p/1-p) is the odds ratio. If the log of the odd-oriented ratio is positive, the probability of success rate will always be higher than 50%. The range of probability ranges between 0 and 1.



Following parameters are necessary to check the effectiveness of our logistic regression model.

Akaike Information Criteria
The model with the minimum AIC is always preferred for effective result. It is a measure of fitness which can penalize any model for the frequency of its model coefficients.

Null Deviance
Null Deviance represents the outcome predicted by a model with the help of the intercept. It all depends, if the null deviance is less, then the model will be better.

Residual Deviance
Residual deviance describes the response predicted by a model on the addition of independent variables. Same goes for residual deviance, lower the value, better the results.

Confusion Matrix
The confusion matrix is the tabular representation of the comparison between actual and predicted values. It helps in deciding the performance of a machine learning model, either a classification model or a regression one and tries to avoid overfitting.

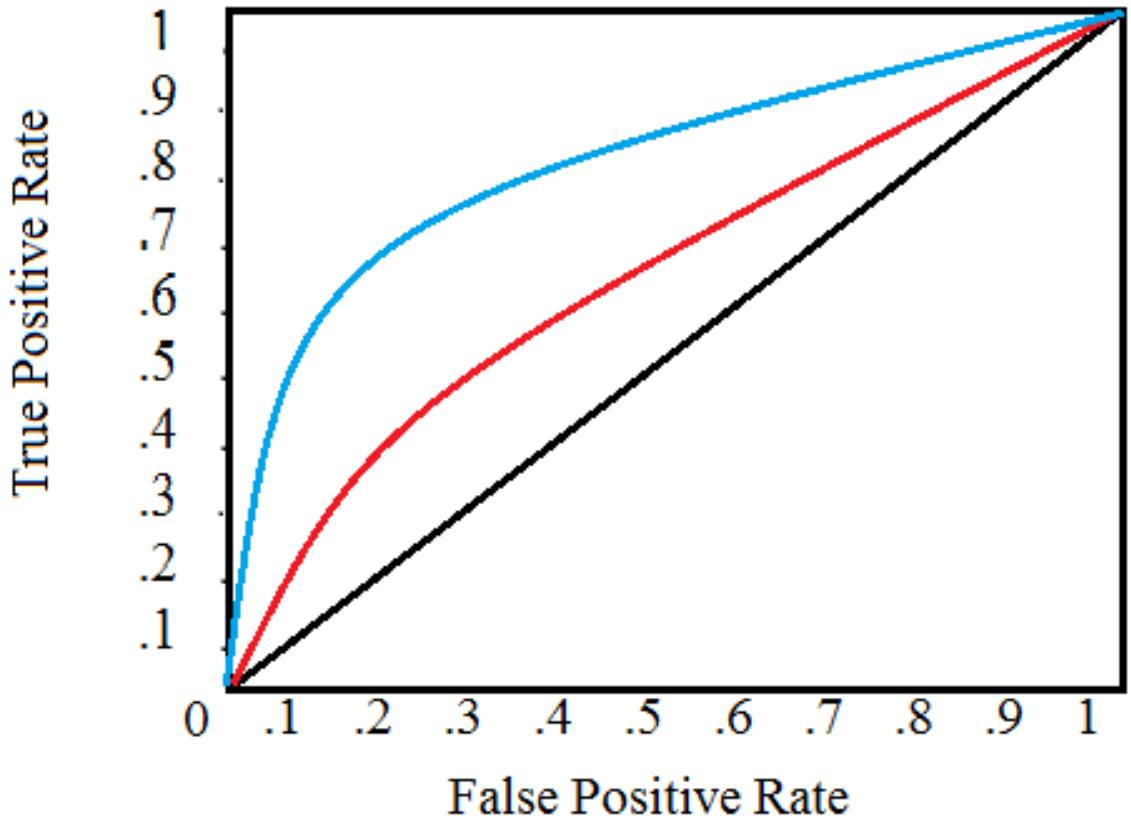
	Predicted Values	
Actual Values	True Positive	False Positive
	False Positive	True Negative

The accuracy of a model can be calculated by

[True Positive(TP) + True Negative(TN)] / [True Positive(TP) + True Negative(TN) + False Positive(FP) + False Negative(FN)]

ROC curve

Receiver operating characteristic is meant to show the trade-off value between sensitivity and specificity for a sequence of tests in a graphical representation. This curse is achieved by comparing the true positive rates with the false positive rates or the true negative rate with the false negative rate.



Code Snippet

```
import numpy as np
import pandas as pd
from sklearn import preprocessing
import matplotlib.pyplot as plt
```

```
plt.rc("font", size=14)
import seaborn as sns
train_df = pd.read_csv("../input/train.csv")
test_df = pd.read_csv("../input/test.csv")
ax = train_df["Age"].hist(bins=15, density=True, stacked=True, color='teal',
alpha=0.6)
train_df["Age"].plot(kind='density', color='teal')
ax.set(xlabel='Age')
plt.xlim(-10,85)
plt.show()
plt.figure(figsize=(20,8))
avg_survival_byage = final_train[["Age", "Survived"]].groupby(['Age'],
as_index=False).mean()
g = sns.barplot(x='Age', y='Survived', data=avg_survival_byage,
color="LightSeaGreen")
plt.show()
from sklearn.linear_model import LogisticRegression
from sklearn.feature_selection import RFE
cols=
["Age","Fare","TravelAlone","Pclass_1","Pclass_2","Embarked_C","Embarked_S
","Sex_male","IsMinor"]
X = final_train[cols]
y = final_train['Survived']
model = LogisticRegression()
rfe = RFE(model, 8)
rfe = rfe.fit(X, y)
print('Selected features: %s' % list(X.columns[rfe.support_]))
from sklearn.model_selection import cross_validate
scoring = {'accuracy': 'accuracy', 'log_loss': 'neg_log_loss', 'auc': 'roc_auc'}
modelCV = LogisticRegression()
results = cross_validate(modelCV, X, y, cv=10, scoring=list(scoring.values()),
                         return_train_score=False)
```

Regression models in ML

Linear Regression

Linear regression is a part of the predictive analysis and is the common algorithm used for prediction of data values in numeric form. Both the input and output values are supposed to be numeric. There is one coefficient represented by Beta (β) in the linear equation of linear regression. Another coefficient is added, providing the line an additional degree of freedom which is known as the intercept.

A simple regression problem would be classified by the following equation.

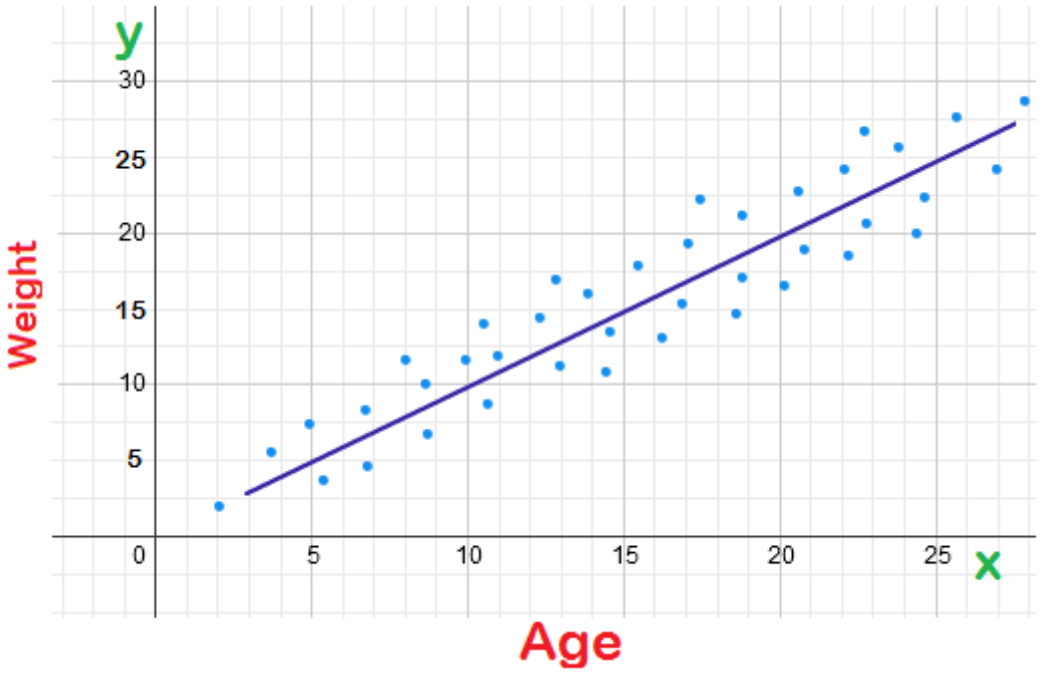
$Y = \beta 0 + \beta 1^{*}x$

The overall notion of regression is to examine two important criteria, does the given group of input variables do a capable job of predicting the dependent variable? And which variables, in particular, are the real predictors of the dependent variable, and what is their impact on the outcome variable?

Reliability of our linear regression model can be tested with the help of the following functions.

The Line of Best Fit

The line of best fit is a validation line which explains the direct correlation between the observed values and the calculated ones. It helps in understanding the difference in distance of the predicted values to the real ones. This distance is known as residual. These residuals are symbolized by the vertical lines showing the comparison between the predicted and actual values.



The graph gives us a simple understanding of how this works. The blue line which is separating the points is our line of best fit., gives a clear indication of the exact difference between the original values and the predicted ones.

For calculating the distance between the line and the points, we need the following formula

Residual = ∑[h(x)-y]^2
where h(x) is the predicted value and y is the actual value

Gradient Descent

Gradient Descent was introduced in linear regression in order to find the minimalistic cost of a function in the equation. Depending on the convex function, the gradient descent function gradually makes some small changes to its parameters in order to minimize a given function to a possible local minimum if necessary.

There are two important terms in gradient descent, learning rate, and the cost function. The size of each step that we take in our model is measured by the learning rate. The curve plotted from our model has its own slope which updates us on the parameters and how to make our model more accurate. This is known as the cost function.

Code Snippet

```
import pandas as pd
from pandas import Series,DataFrame
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style('whitegrid')
from sklearn.linear_model import LogisticRegression
titanic_df = pd.read_csv("../input/train.csv")
test_df    = pd.read_csv("../input/test.csv")
titanic_df = titanic_df.drop(['PassengerId','Name','Ticket'], axis=1)
test_df    = test_df.drop(['Name','Ticket'], axis=1)
titanic_df['Fare'].plot(kind='hist', figsize=(15,3),bins=100, xlim=(0,50))
avgerage_fare.index.names = std_fare.index.names = ["Survived"]
avgerage_fare.plot(yerr=std_fare,kind='bar',legend=False)
logreg = LogisticRegression()
logreg.fit(X_train, Y_train)
Y_pred = logreg.predict(X_test)
logreg.score(X_train, Y_train)
```

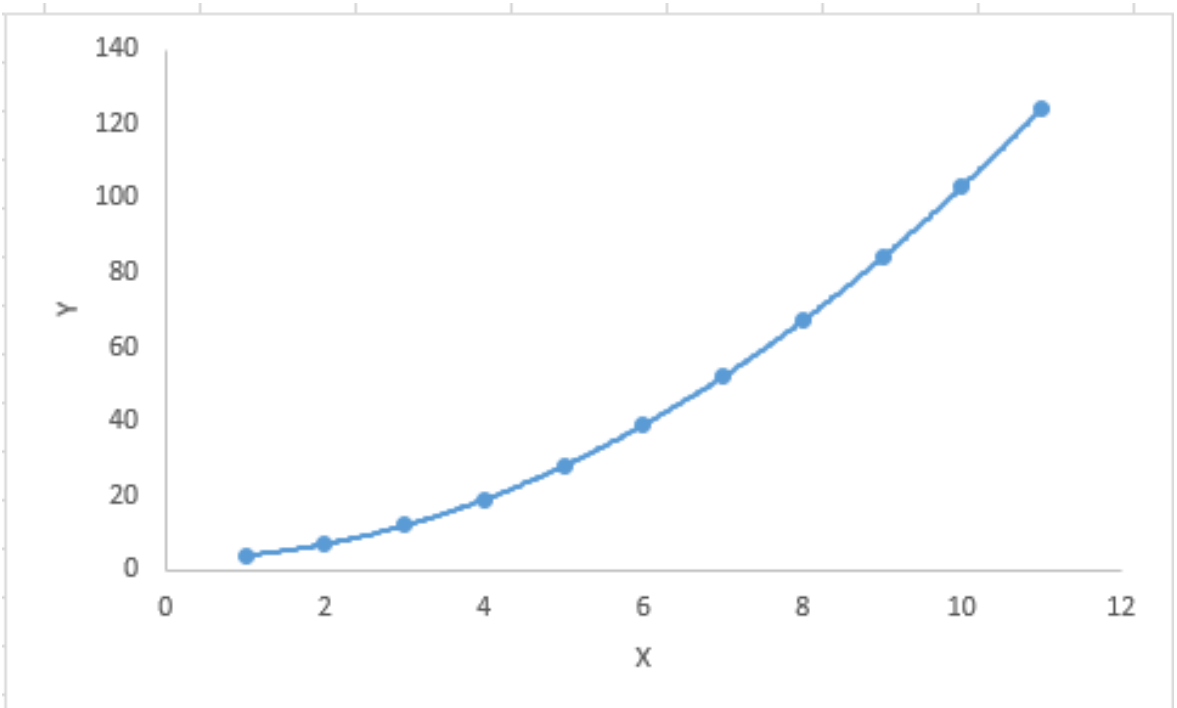
Polynomial Regression

Just like linear regression, a polynomial regression model is used to find the relation between a dependent and an independent variable. The only difference is at times, in linear regression, the line of best fit may end up under-fitting the data points in the picture as it couldn't capture the necessary trends and patterns in data. Hence, polynomial regression plays a crucial part in improving this model and finding a more subtle relationship between the variables.

A regression polynomial is represented by the following equation.

$Y = a + b * x \wedge 2$

The line of best fit is actually a straight line in linear regression, whereas in polynomial regression, it is a curved line. If we consider the case of under-fitting, then this can be resourceful. But, at times, when we tend to increase the accuracy of our model, we might end up over-fitting it which is also not recommended for real-world implementations. Hence, we need to observe the nature of the curve and make changes in our model accordingly.



Focusing on the curve at the ends and checking whether those trends make any sense is vital. Sometimes, higher polynomials will end up producing mysterious results on extrapolation.

Code Snippet

```
import pandas as pd
import numpy as np
prices = nyc_pumpkins.values[:, :1]
sizes = nyc_pumpkins.values[:, 1:2]
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
poly = PolynomialFeatures(degree=2)
prices_poly = poly.fit_transform(prices)
clf = LinearRegression()
clf.fit(prices_poly, sizes)
predicted_sizes = np.round(clf.predict(prices_poly))
poly = PolynomialFeatures(degree=2)
prices_poly = poly.fit_transform(prices)
pd.Series((sizes -
predicted_sizes).flatten()).value_counts().sort_index().plot.bar(
  title='$y - \hat{y}$'
)
pd.Series(
  np.abs((sizes - predicted_sizes).flatten()) <= 1
).value_counts().plot.bar(title='Accuracy Within 1 Class')
```

Lasso Regression

Least Absolute Shrinkage and Selection Operator is similar to ridge regression in terms of working which changes the size of regression coefficients. It can also reduce the variability and advances the accuracy of linear regression models. Lasso regression handles the real absolute values in the penalty function rather than squares like in ridge. This enables it to penalize values which cause some of the parameter estimates to be exactly zero. If the penalty is high, then the estimates get shrunk to zero. This results in variable selection, out of the given n variables.

Some of the assumptions that are made in this algorithm are :

- The assumptions of this algorithm are the same as of the least square regression except for one thing that normality is not to be assumed here.
- It shrinks the coefficients to zero which makes to easy for feature selection.
- This is a regularization technique and adopts L1 regularization.
- If a group of predictors is highly correlated, then it picks any one of them and shrinks the others to zero.

Code Snippet

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import Lasso
train = pd.read_csv("../input/train.csv")
test = pd.read_csv("../input/test.csv")
x = all_data.loc[np.logical_not(all_data["LotFrontage"].isnull()), "LotArea"]
y = all_data.loc[np.logical_not(all_data["LotFrontage"].isnull()), "LotFrontage"]
# plt.scatter(x, y)
t = (x <= 25000) & (y <= 150)
p = np.polyfit(x[t], y[t], 1)
all_data.loc[all_data['LotFrontage'].isnull(), 'LotFrontage'] = np.polyval(p,
all_data.loc[all_data['LotFrontage'].isnull(), 'LotArea'])
model_lasso = Lasso(alpha=5e-4, max_iter=50000).fit(X_train, y)
coef = pd.Series(model_lasso.coef_, index = X_train.columns).sort_values()
```

```
imp_coef = pd.concat([coef.head(10), coef.tail(10)])
imp_coef.plot(kind = "barh")
plt.title("Coefficients in the Model")
```

Clustering models in ML

K-means clustering

K-means clustering comes under unsupervised learning algorithm. It is utilized for unlabeled data, that is, the data points that do not belong to any group or category. The modus operandi of k-means is to determine the number of groups present in the data, and these groups are represented by k. This algorithm works in a way which assigns each data point to one of the k groups iteratively based on the features that are already present in the dataset. These data points are clustered together based on similarity.

Clustering can be used for exploratory data analysis to get a rough estimation of the structure of data. It is usually assigned to find the subgroups in the data where the data points in this subgroup are similar to each other in characteristics while points in other groups are different. K-means has the job of polarising homogenous subgroups within the data wherein all the data points in all the groups that exist within the data are similar.

These are the basic steps used to cluster any data point for any data :

- At first, define the number of k clusters.
- Initialize the centroids by randomly selecting k data points for clustering.
- Perform this iteration until there is no change to the centroids.
- Calculate the sum of the squared distance between data points and all centroids.
- Next, assign each data point to the closest centroid or cluster.
- Lastly, estimate the centroid for the clusters by taking the average of all the data points that are assigned to each cluster.

Business use cases

Managing sensors :

Sensors collect data of different types like audios, images etc. in its database. K-means is used to separate that data and create different categories.

Managing inventory :

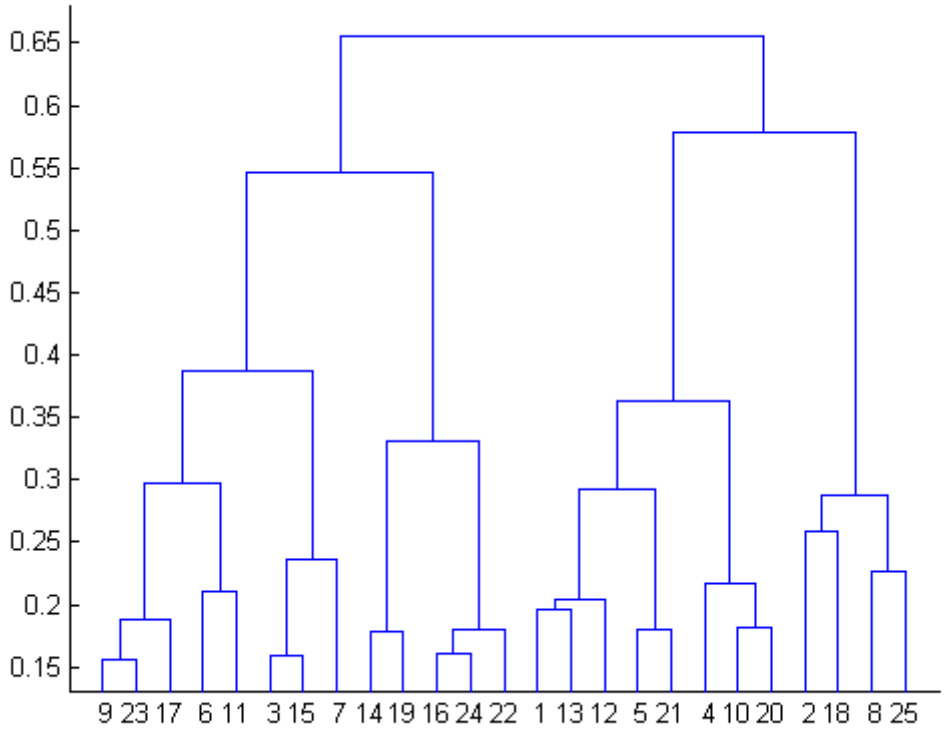
An inventory consists of diverse data like sales activity, manufacturing data, import-export data etc, which is grouped into categories by k-means.

Code Snippet

```
import numpy as np # linear algebra
import pandas as pd
df = pd.read_csv("../input/Wine.csv")
df.info()
corrmat = df.corr(method='spearman')
f, ax = plt.subplots(figsize=(8, 8))
sns.heatmap(corrmat, vmax=.8, square=True)
plt.show()
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_samples, silhouette_score
matrix = df.as_matrix()
for n_clusters in range(2,30):
  kmeans = KMeans(init='k-means++', n_clusters = n_clusters, n_init=30)
  kmeans.fit(matrix)
  clusters = kmeans.predict(matrix)
  silhouette_avg = silhouette_score(matrix, clusters)
  y.append(silhouette_avg)
  print("For n_clusters =", n_clusters, "The average silhouette_score is :",
silhouette_avg)
plt.figure(figsize=(12,8))
plt.plot(range(2,30),y)
plt.xlabel('No of Clusters')
plt.ylabel('Silhouette_avg')
plt.title('Silhoutte Score for different clusters')
```

Hierarchical clustering

Hierarchy is the level of clusters that are involved in the dataset. Hierarchical clustering involves groups of clusters which are merged into the nearest similar cluster until they form a single cluster for the entire dataset.
Its working can be shown in the form of a dendrogram below.



Hierarchical clustering

Hierarchy is the level of clusters that are involved in the dataset. Hierarchical clustering involves groups of clusters which are merged into the nearest similar cluster until they form a single cluster for the entire dataset.
Its working can be shown in the form of a dendrogram below.

Hierarchical clustering can be of two types :

Divisive

The divisive approach which is also known as the top-down clustering includes data points assigned to a single cluster which is later segregated to two or more similar looking clusters. Each cluster is examined recursively until the algorithm settles for one in the end. It is true that the divisive approach has more rewards compared to agglomerative approach due to its capability of producing more accurate hierarchies for diverse datasets.

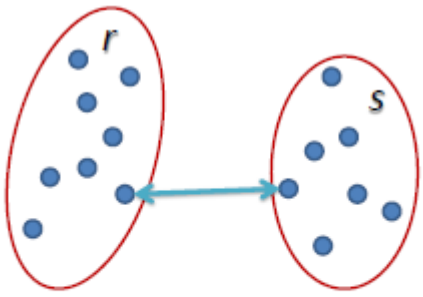
Agglomerative

Agglomerative approach defines a bottom-up approach. Then it assigns the clusters to data points and merges them with each other just like divisive to gain a single cluster for the dataset.

For performing any clustering operation, it is mandatory to introduce a proximity matrix which contains the distance between each point with the help of a distance function. This matrix needs to be updated for it to display the distance between each cluster. The following three methods are used to check the distance between each cluster.

Single Linkage

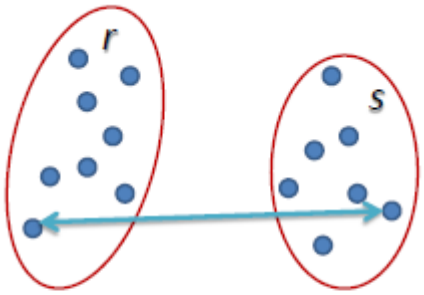
Single linkage technique includes the concept of defining the distance between two clusters as the shortest distance between two points in each cluster. In this case, the distance between clusters r and s is equal to the length of the arrow measured between their closest points.



$$L(r,s) = \min(D(x_{ri}, x_{sj}))$$

Complete Linkage

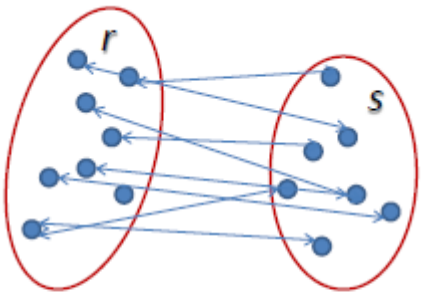
The longest distance between points of two nearby clusters which can also be termed as the longest distance for each cluster is defined in the complete linkage. In our case, clusters r and s are equivalent to the length of the arrow as shown in the image.



$$L(r,s) = \max(D(x_{ri}, x_{sj}))$$

Average Linkage

This defines the average distance between data points of two nearest clusters. The clusters r and s are equal to the average length.



$$L(r,s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} D(x_{ri}, x_{sj})$$

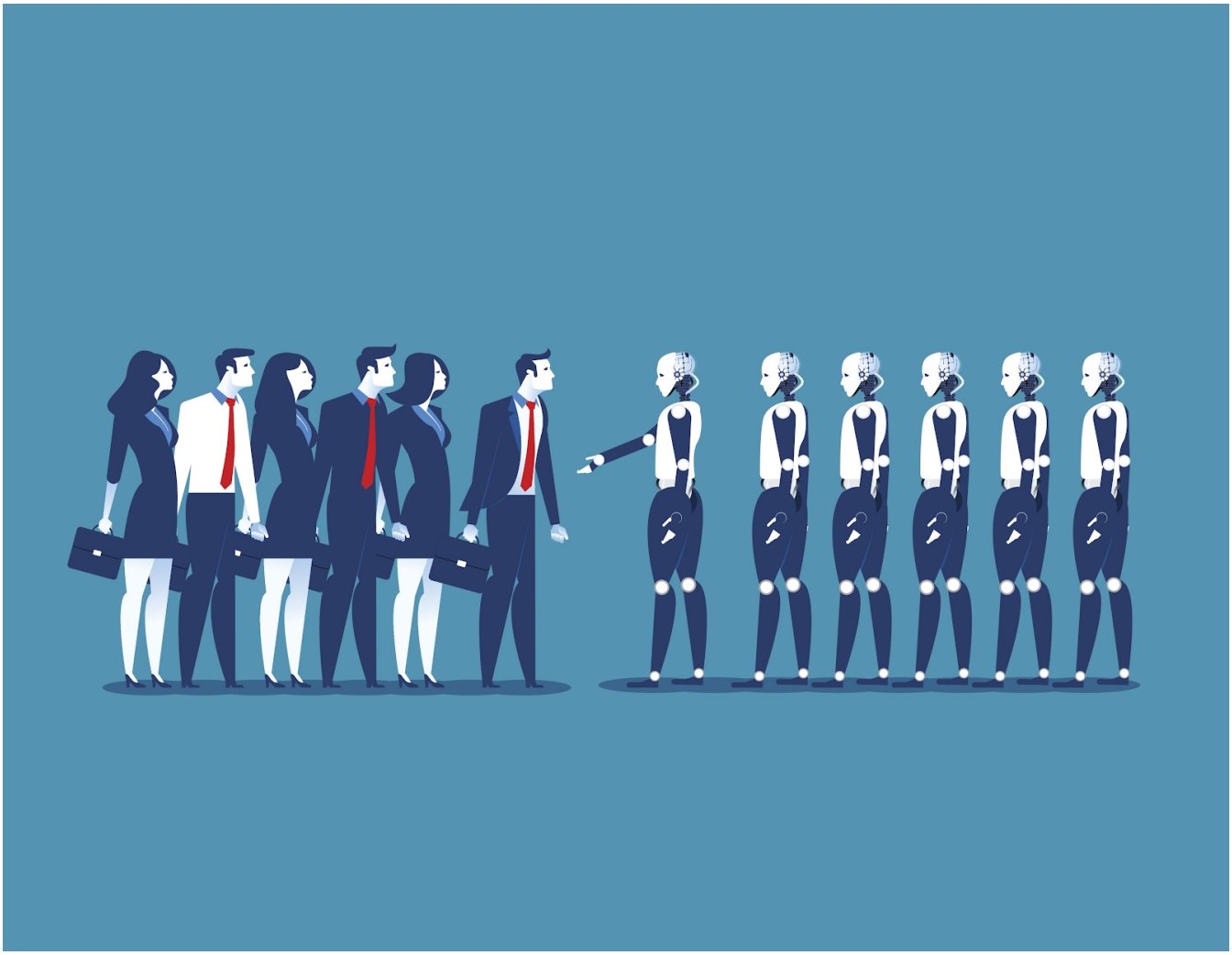
Code Snippet

```
import numpy as np # linear algebra
import pandas as pd
df = pd.read_csv("../input/Wine.csv")
df.info()
corrmat = df.corr(method='spearman')
f, ax = plt.subplots(figsize=(8, 8))
sns.heatmap(corrmat, vmax=.8, square=True)
plt.show()
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_set, y_pred)
print("success ratio : ",success_ratio(cm=cm), "%")
from sklearn.cluster import AgglomerativeClustering
hc = AgglomerativeClustering(n_clusters = 3, affinity = 'euclidean', linkage =
'ward')
y_hc = hc.fit_predict(X)
```

Rise and emergence of Artificial Intelligence in the modern era

Why AI?

Artificial intelligence was just another hyped technical jargon some 10 years ago. We could only find its applications in movies where AI-powered robots and humanoids were created using computer graphics and special effects. It was a distinct dream at that time. Fast forward 10 years later, it has become quite a possible phenomenon to establish and nurture a machine which can not only read and understand human emotions but can also perform tasks that are difficult for humans. They have managed to outsmart us in every aspect and have attained a certain degree of sophistication and implication where they can be utilized for various automation processes and machine-intensive tasks.



Moore earlier predicted that the use of machines will grow exponentially from 1960 to 2020 which will see more number of machines being utilized for various industry-ridden purposes. His prediction gave everyone the perspective of the direction where the current computer science market was headed and made the industry aware of this revolution. Deepmind, a deep-Ai powered chess-playing machine was designed to understand the patterns of chess and outsmart humans. It was able to defeat the world champion chess player after applying its carefully-curated algorithm learned through machine learning which was also the most broadcasted chess match on television at the time. AI can definitely be utilized for the upbringing of any business with the help of its modern techniques.

What is reinforcement learning?

Reinforcement learning is an important foundation for artificial intelligence. It is dynamic programming mechanism which trains the agent(machine) using the constants of success and failure. It needs to have an uninterrupted interaction with its current surroundings by observing and gaining knowledge from it. This process is carried out for quite some time until the host understands the difference between right and wrong. This is carried out considering the fact that external human intervention is minimal so that the agents can learn and interpret the current scenario on their own.

Machine learning has an integral part to play in reinforcement learning as the agents learn from their mistakes and ensure that it won't repeat again. But unlike machine learning, where the machine is expected to produce a certain output, this learning trains agent to come up with their own observations and inferences. There are certain algorithms of control learning that are used to execute this process like criterion for optimality, brute force, and direct policy search.

Boston Dynamics' Spotmini

A good example of reinforcement learning is the Boston Dynamics' newly invented agent Spotmini. It was designed and manufactured on the principles of self-learning and self-evaluation which made it easy to overcome external circumstances while closing a door.

The Spotmini robot trying to open the door

There is a small video that was released by Boston Dynamics showcasing what it is capable of. As scary it may look, humanoids surely are learning to live and melt in their surroundings even if any external intervention is certain. In this video, the tester is trying to block the agent's way to open the door. But clearly, it outshines the human intentions and opens the door with great ease due to reinforcement learning.

Plan Of Attack

Bellman Equation

Dynamic programming was first composed in the year 1956 when Richard Ernest Bellman came up with an equation which constituted factors like the current state(s), action(a), reward(r) and discount rate(y). He came up with an equation which was named as the Bellman equation to represent the value of the current state of an agent or machine.

$V(s) = \max(R(s,a)) + \gamma V(s')$
Where s' is the s prime which is our next state

The Bellman equation was created to understand the working of an agent in a dynamic environment which handled the analysis of the current state and resultant state. This equation illustrates that the value of being in a state is the maximum of all the possible actions taken in that state which comes from rewards taken in that action in that particular state along with a discount factor of the values of the next state.

Markov Decision Process

The Markov Decision Process is a framework which provides a platform for modeling decision making for situations where results are partly random and partly controlled. This process involves steps in which the decision maker (which is making changes externally) or the random state may choose a particular set of actions for each level. This helps it to move to the next state and to deliver a reward state to the decision maker.

The Markov Decision Process is an extension of the Bellman equation. As the bellman equation provides values for the next state by utilizing the action which rewards the current state, MDP gives a random probability to take a decision-ridden step to move to the next state.

Its equation can be stated as :

$V(s) = \max(R(s,a)) + \gamma \sum P(s,a,s')V(s')))$
Where P(s,a,s') represents probability of a state regardless of what actions we take.

Q Learning Intuition
Q-learning comes under reinforcement learning where it can be considered as an extension to the Markov Decision process. Its intention is to acquire a policy, which guides the agent involved to perform actions under a certain set of rules and regulations. It has the capability of tackling problems which can have stochastic or random transitions without the need for external enactments.

It includes a function q which takes a state and an action as an input. For the output, it delivers the reward for that action in that particular state. As we keep on exploring this environment more and more, Q delivers us better and more accurate value for action "a" at the current state "s". The function Q keeps on updating as we go further.

The basic functioning of Q-learning can be represented with the following equation.

$Q(s2,a2) = Q(s1,a1) + \alpha * (r + \gamma * \max Q(s3+1,a) - Q(s1,a1))$

Where Q(s1,a1) is the old value for Q, α is the learning rate, r is the reward, γ is the discount rate, maxQ() is the estimate of optimate future value.

Artificial Neural Networks

What is Deep Learning?

Deep Learning is a major machine learning field which is currently in its evolving stage. It involves the deep study of algorithms that are inspired by the structure of the human brain where all the neurons are communicating different functions with each other through axons. Similarly, there are layers of nodes present in the artificial neural networks which are connected to each other and coordinate all the activities in the network.

It uses a network of multiple levels of nonlinear processing. It can learn in a supervised, unsupervised and a semi-supervised environment. Deep learning is used in large deep neural networks to understand their functioning and how we can gain results from it. Andrew Ng from coursera describes deep learning as the algorithm which can use brain simulations to make algorithms learn faster and better. This can help in incremental advancements in AI and machine learning and will help the industry in realizing its untapped potential in the market.

Let's dive into some applications of deep learning.

Image Recognition

This involves learning and identifying the images through deep learning. It is already used in areas like retail, social media, gaming etc. This will not only identify the people in the image but can also the objects that are present in the image along with their characteristics.

Earthquake Prediction
Deep learning has been exploited to understand the viscoelastic computations that are involved behind the physics of an earthquake. The prediction done by deep learning is touted to be near accurate and more advancements are being made to monitor and understand its patterns for better and quicker predictions.

Text Generation
Deep learning has the capability of completing texts and sentences. A text corpus is learned and evaluated by the algorithm which enables it to understand and predict the punctuation, spelling, and the style of the text present in the corpus.

Activation function
Activation functions are considered to be an integral function in artificial neural networks. They have the task of deciding whether a particular neuron should be activated or not. It examines the fact that whether a neuron is receiving the information that is related to the given function or not. Activation function can be represented by the following equation.

Y = Activation(∑(weight * input) + bias)

There are some important functions that are involved in the activation function.

Binary step function :
The binary function is extremely simple. It can be used while creating a binary classifier. When we simply need to say yes or no for a single class, step function would be the best choice, as it would either activate the neuron or leave it to zero.

f(x) = 1, x>0

As we are bound to classify the given data into multiple classes rather than just a single class, the step function is more useful In theoretical purpose then practical purpose.

Linear function :
The linear function takes care of the loopholes in the binary step function. In the step function as the gradient was 0 it was impossible to update the gradient during backpropagation, instead, we can use a linear function which can be defined as.

f(a) = ax

The input x in binary step function is transformed to ax. Hence we can activate multiple neurons at once. When we have multiple cases, we can choose the neuron which has the maximum value.

Softmax
The softmax function is basically a sigmoid function which can handle classification problems. It can take care of two different classes. Let's see if it can handle more than two classes. Just sorting yes or no for a particular class is not enough. The softmax function is useful for calculating the outputs for each available class 0 and 1 and later divide it by the sum of the outputs. Its general equation can be defined as.

$\sigma(z) = e^z / \sum e^z$ for j=1,...,k

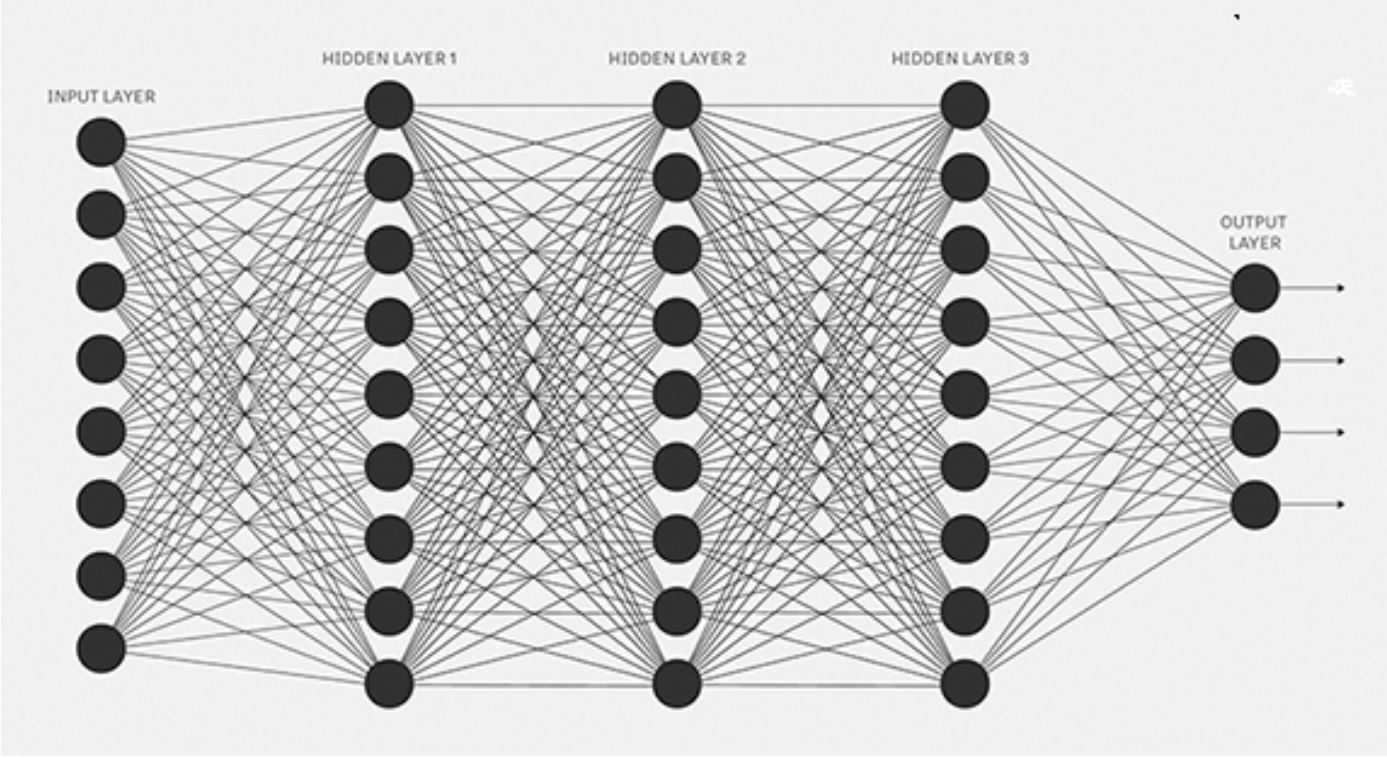
The softmax function is used in the output layer of the classifier in which we predict the probabilities to determine the class of the given input.

Convolutional Neural Networks

What is CNN?

CNN was designed specifically to tackle the large processing power required for image processing. This neural network requires less processing power compared to the current image classification algorithms being implemented. It has industrial applications in video and image recognition, recommendation systems, image and text classification and natural language processing(NLP).

It is a subdivision of deep neural networks which are commonly recommended for analyzing visual related imagery and applications. They are made up of a variety of multilayer perceptrons which handles the processing power. Inspired by the biological evolution, they are connected with each other just as the neurons are connected in the brain of an animal.



A convolutional neural network comprises of an input and an output layer along with multiple hidden layers which have a network to remain connected with other. These layers are also known as convolutional layers. Now, let us see some of the important concepts that are useful in CNN.

ReLU layer

The ReLU function is known as the Rectified Linear Unit. For many applications, it is the widely used activation functions. It is represented as.

$f(x) = \max(0,x)$

Building a set of networks involves the use of this function. It is a non-linear function where it can easily backpropagate the errors and contain multiple layers of neurons which are activated by the ReLU function.

The primary reason why ReLu is favored over other activation functions is that it does not activate all the neurons present in the layers at once. The Relu function takes care of the input if it is negative. All the negative inputs are converted to zero and the neurons stay inactive. As only a few selective neurons are getting activated, the workload on the network is reduced which makes it easy and effective for implementation.

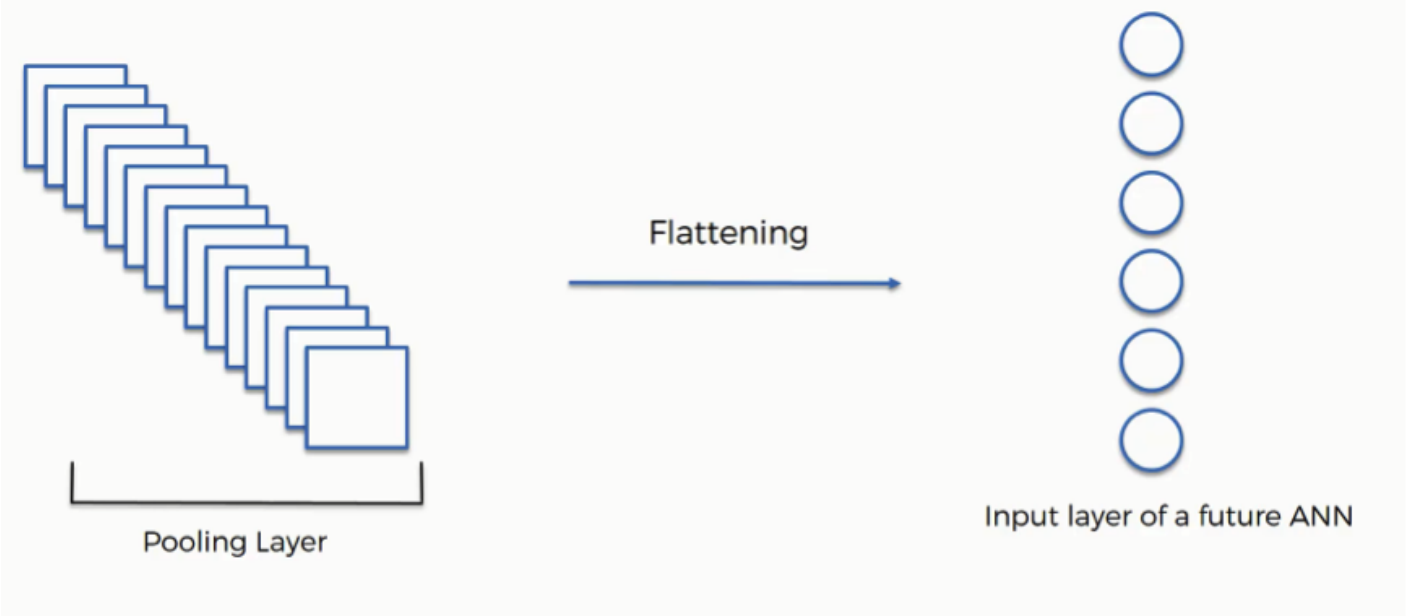
Pooling

After undergoing and applying the ReLU layers, deep learning uses a pooling layer. It is used to reduce the total number of hidden layer to a single layer which has the maximum pooling value. This enables the reduction in dimensionality of the hidden layers and delivers a clean network which puts less stress in processing speed.

Max pooling is done by applying a max filter to the non-overlapping subregions in the initial input layers. It can be considered as the algorithm which reduces a multi-dimensional network into a single network without the loss of any data from the neurons and accelerating the process of deep learning in convolutional neural networks.

Flattening

Flattening comes right after pooling is performed and is pretty simple. After pooling all the features into one single maximum feature, it is converted or flattened into a column in order to insert this data into an artificial neural network.



As ANN requires this feature in the form of an input layer, the pooling layer has to undergo flattening.

Cross-Entropy

The cross-entropy technique is always compared with the mean squared error. But for practical use, cross-entropy is better. During the backpropagation process, the output that we get in very minimal compared to the desired one. At times, the gradient value is also quite low making it difficult for the neural networks to optimize itself. Hence, the cross-entropy function enables the network to correct these small errors and eliminate them.
Cross-entropy can be illustrated by the following equation.

$$H(p,q) = - \sum p(x) * \log q(x)$$

Applications and future potential of AI

Self-Driving Cars

The testing and manufacturing of self-driving cars have already seen tremendous growth in the last 5 years. This technology can enable driver-less cars and better monitoring of street ethics. Many manufacturers like Tesla, Volvo, Audi are enabling better driving experience by collecting data and using them to counter traffic and severe mishaps on the roads.

Smarter Cities

Though this concept was prevalent in the market for a long time, it has a long way to go until it can become mainstream. Smarter cities are bring envisioned to provide better facilities like on-the-go shopping, better communication, enhanced security measures powered through AI etc.

Outgrow businesses

Business will be highly reliant on AI as it can accelerate the current business stint and outgrow it into a subtle and more balanced entity which will work on ethics set by the AI. This will enable smarter ideas and improved customer relationship.

Conclusion/End-Notes

Machine learning was vastly adopted by statistics and most of its key concepts all are available to explore in real-time scenarios. It has seen many companies use it to extract meaningful and worthy inferences from their pool of data and utilize the same for their businesses. However, artificial intelligence is still at the brink of evolution and a lot has to be unearthed and discovered in this ever-growing and evolving genre. Massive enterprises and biggies like Google and Microsoft are investing billions of dollars just to have a deep look into deep learning and artificial intelligence. If machine learning is taking care of the present, then artificial intelligence is the key to the future.

Sign up for more!

Sign up on Eduonix for free to get many other cool e-books like this one. Also, you can get support for this and other e-books.

Click here to sign up

Course for Free!

To grasp the basics of Machine Learning opt for our course "Machine Learning for Absolute Beginners".

*Coupon code : **100OFF***

Machine Learning E-Degree!

For a comprehensive understanding of all major concepts take our E - degree comprised of 6 detailed courses on Artificial Intelligence and Machine Learning like Foundational Mathematics, Machine Learning using Python, Deep Learning and more at a 50% OFF

Click here to to get the discount

*Coupon code : **AIML50OFF***

eduonix