MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION, MUMBAI

ACADEMIC YEAR 2020-21

GOVERNMENT POLYTECHNIC, JALGAON

DEPARTMENT OF ELECTRONICS & TELICOMMUNICATION

CourseTitle: Emerging Trends in Electronics (22636)

Titel : Build Temperature Prediction System Using Machine Learning.

GROUP MEMBER NAME

NIKHIL DNYANESHWAR MAHAJAN (35)

REHAN SHAIKH (39)

GAURAV GHANSHAM ATKALE (61)

VIJAY GANESH PATIL (65)

PROJECT GUID NAME

K. P. Akole (Sir)

GOVERNMENT POLYTECHNIC, JALGAON

Certificate

Certified that this Report submitted by

Shri/Kum. _____

Roll No. _____ Exam Seat No. _____

Isyudent of _____ Sem/Year of course _____

is a Part of projest Work/ Seminar/ Practical/ Drawing/ Workshop/ Term Work/ Industrial visits as prescribed by the Board of Technical Examinations Mumbai for the Subject _____

_____

And that I have instructed/ guided him for the said work for time to time and I found him to be satisfactorily progressive.

And that following students were associated with him for this work. However his contribution was propartionate.

_____ _____ _____

_____ _____ _____

And the Said work has been assessed by me and I am saatisfied that the same is up to the standard ennisaged for the level of the course.

And that the said work may be presented to the External Examinar.

Signature
Name of the Teacher/Guide
Date : / /20

Signature
Head of the Department

Maharashtra State Board Of Technical Education
Certificate of Completion

Of Micro-Project Assessment at the end of the Diploma Programme
(By respective Head of the Institute & Head of the Department)

This is to certify that Student Of **6th** semester has successfully completed **Emerging Trends in Electronics (**22636**)** Micro-Project as in the enclosed 'Portfolio' during his tenure of completing the Diploma Program in **Electronic & Telecommunication** From **Government Polytechnic Jalgaon** Institute Code **0018**. As per the academic year **2020- 2021** as prescribed in the curriculum.

Place: Jalgaon

Sr. No.	Roll No	Enrollment No
1	61	1900180410
2	65	1900180414
3	39	1900180408
4	35	1900180400

Date: _____/____/_____

Signature

Course Teacher

Seal of The Institute

Signature

Head of the Department

TITLE

Build Temperature Prediction System Using Machine Learning

Represented By

Roll No.	Name	Enrollment
61	Gaurav Atkale	1900180410
65	Vijay patil	1900180414
39	Rehan Shaikh	1900180408
35	Nikhil Mahajan	1900180400

Under the guidance of:

K. P. Akole (Sir)

1. RATIONALE

Every technological area is developing at an exponential rate. New applications are coming up and it is mandatory for all technologists to be well versed in these developments to survive and provide satisfactory and quality services to the society and industry. This course aims to prepare the diploma graduates to be conversant with such emerging trends. The main areas in which such developments are encompass Smart systems, Digital Factory and Communication. The course gives an introduction of these areas and helps the students to apply emerging trends.

2. COMPETENCY

Aim of this course is to help the student to attain the following industry identified competency through various teaching learning experiences:

- **Use the trending practices in Electronics fields.**

3. COURSE OUTCOMES (COs)

The theory, practical experiences and relevant soft skills associated with this course are to be taught and implemented, so that the student demonstrates the following industry oriented COs associated with the above mentioned competency:

a) Suggest the relevant computing systems/processor for specific type of application.
b) Suggest the relevant components for the emerging application/s.
c) Suggest different telecom network for given application.
d) Suggest the relevant IOT technologies for Digital Factory.
e) Suggest the different electronic systems for smart world.

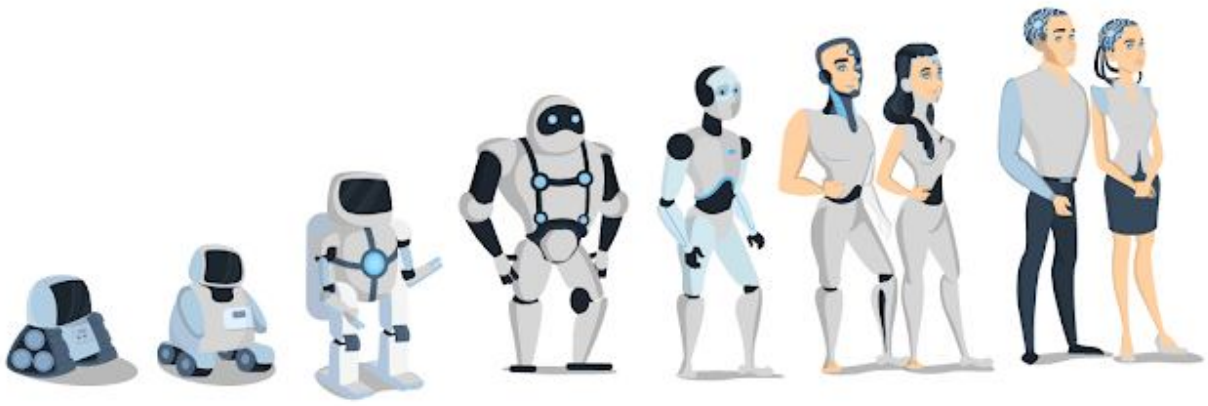
INDEX

1. Machine Learning

1.1. Introduction Machine Learning

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. **Machine learning focuses on the development of computer programs** that can access data and use it learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. **The primary aim is to allow the computers learn automatically** without human intervention or assistance and adjust actions accordingly. **What is Machine Learning - Evolution of Machines.**

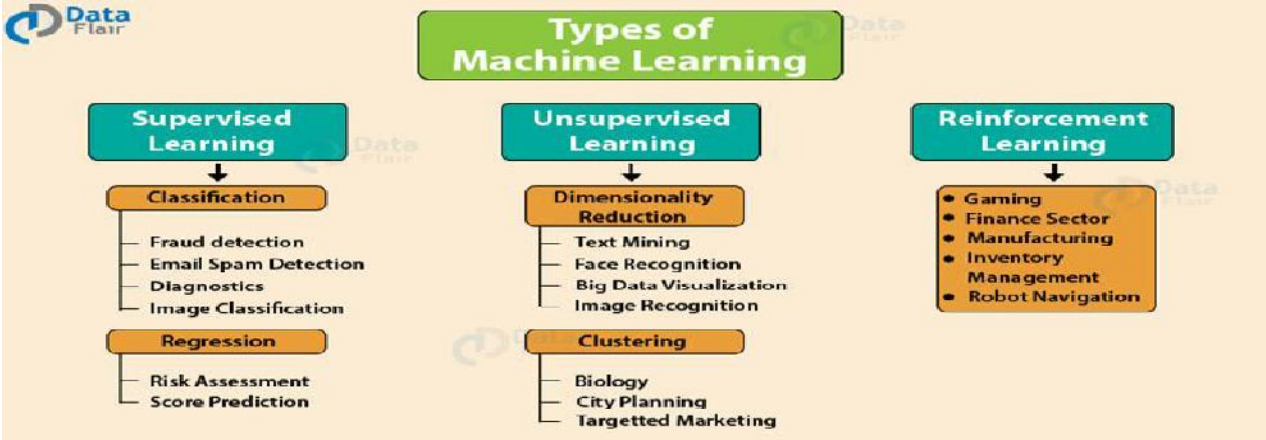


As you know, we are living in the world of humans and machines. The Humans have been evolving and learning from their past experience since millions of years. On the other hand, the era of machines and robots have just begun. You can consider it in a way that currently we are living in the primitive age of machines, while the future of machine is enormous and is beyond our scope of imagination.

In today's world, these machines or the robots have to be programmed before they start following your instructions. But what if the machine started learning on their own from their experience, work like us, feel like us, do things more accurately than us? These things sound fascinating, Right? Well, just remember this is just the beginning of the new era.

1.2. Types of Machine Learning

There are also some types of machine learning algorithms that are used in very specific use-cases, but three main methods are used today.



1.2.1. Supervised Learning

Supervised learning is one of the most basic types of machine learning. In this type, the machine learning algorithm is trained on labeled data. Even though the data needs to be labeled accurately for this method to work, supervised learning is extremely powerful when used in the right circumstances.

In supervised learning, the ML algorithm is given a small training dataset to work with. This training dataset is a smaller part of the bigger dataset and serves to give the algorithm a basic idea of the problem, solution, and data points to be dealt with. The training dataset is also very similar to the final dataset in its characteristics and provides the algorithm with the labeled parameters required for the problem.

The algorithm then finds relationships between the parameters given, essentially establishing a cause and effect relationship between the variables in the dataset. At the end of the training, the **algorithm** has an idea of how the data works and the relationship between the input and the output.

This solution is then deployed for use with the final dataset, which it learns from in the same way as the training dataset. This means that supervised machine learning algorithms will continue to improve even after being deployed, discovering new patterns and relationships as it trains itself on new data.

1.2.2. Unsupervised Learning

Unsupervised machine learning holds the advantage of being able to work with unlabeled data. This means that human labor is not required to make the dataset machine-readable, allowing much larger datasets to be worked on by the program.

In supervised learning, the labels allow the algorithm to find the exact nature of the relationship between any two data points. However, unsupervised learning does not have labels to work off of, resulting in the creation of hidden structures. Relationships between data points are perceived by the algorithm in an abstract manner, with no input required from human beings.

The creation of these hidden structures is what makes unsupervised learning algorithms versatile. Instead of a defined and set problem statement, unsupervised learning algorithms can adapt to the data by dynamically changing hidden structures. This offers more post-deployment development than supervised learning algorithms.

1.2.3. Reinforcement Learning

Reinforcement learning directly takes inspiration from how human beings learn from data in their lives. It features an algorithm that improves upon itself and learns from new situations using a trial-and-error method. Favorable outputs are encouraged or 'reinforced', and non-favorable outputs are discouraged or 'punished'.

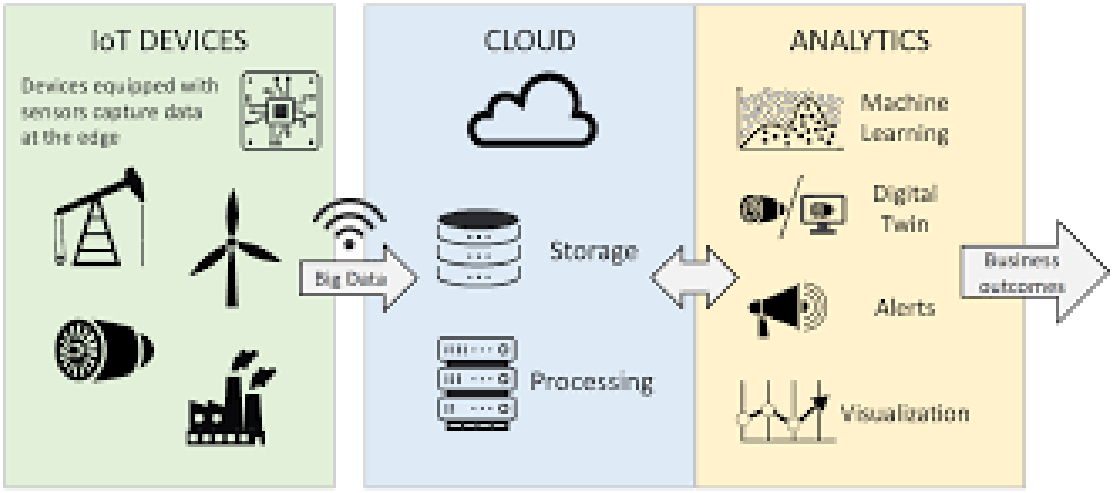
Based on the psychological concept of conditioning, reinforcement learning works by putting the algorithm in a work environment with an interpreter and a reward system. In every iteration of the algorithm, the output result is given to the interpreter, which decides whether the outcome is favorable or not.

In case of the program finding the correct solution, the interpreter reinforces the solution by providing a reward to the algorithm. If the outcome is not favorable, the algorithm is forced to reiterate until it finds a better result. In most cases, the reward system is directly tied to the effectiveness of the result.

In typical reinforcement learning use-cases, such as finding the shortest route between two points on a map, the solution is not an absolute value. Instead, it takes on a score of effectiveness, expressed in a percentage value. The higher this percentage value is, the more reward is given to the algorithm. Thus, the program is trained to give the best possible solution for the best possible reward.

---

2. Relation Between Machine Learning And IOT



2.2. Machine Learning

Machine learning is a field of computer Science that gives computer system the ability to "learn" with data,without being explicitly programmed.

The philosophy behind machine learning is to automate the creation of analytical models in order to enable algorithms to learn continuously with the help of available data.

Machine Learning can be supervised(when desired outcome known) , Unsupervised (when data is unknown) and Reinforcement Learning (learning is the result of interaction between a model and the environment).

2.3. The Internet of Things (IOT)

The Internet of Things (IOT), refers to the connection of Objects over the Internet. This connection is establish through IOT sensors with some local processing to enable smart capabilities. This means that the Internet has already transformed from being a network where humans share data and connect, to Things doing the same, with, or without the need for human intervention and monitoring.

2.4. Main Uses of Machine Learning in IOT :-

- **Clustering of data:-** Binary Classification (positive or negative), Logistic Regression (discrete outcome), K-Means for clustering the data are some popular Algorithms of Clustering in Machine Learning. An application could be analysis of accelerometer, gyroscope and other Inertial Measurement Unit (IMU) data to identify behaviors such as walking, standing, running or sitting in IOT devices.
- **Anomaly Detection:-**A third use-case which is strongly applicable for IOT is Anomaly Detection, i.e. identifying outliers in a data-set. For instance, if your temperature sensor itself goes bad, and starts giving an output of 10 degrees Celsius out in the sun in a desert in summer, you would definitely want to detect such an anomaly.
- **Prediction of data trends:-** The state of the art Machine Learning algorithms used for prediction in the case of IOT sensors i.e. time dependent data include Recurrent Neural Networks (RNNs) and Long Short Term Memory (LSTM) Networks. In general, IOT devices used to store the Data in cloud and Machine Learning applied on those data to predict the accurate outcome.

3. Why companies making investment into AI and ML

Reasons behind

1. Number of devices are connected to the internet they emit the lot of data on regular basis. (Ex Laptop, smart phones etc.)
2. The cost of storing data is gone down significantly.
3. The computational costs continues fall .
4. For not losing comparatives Advantage.

These are four forces which derive Machine Learning as technology

4. Algorithm for Machine learning

Here is the list of commonly used machine learning algorithms. These algorithms can be applied to almost any data problem:

1. Linear Regression
2. Logistic Regression
3. Decision Tree
4. Random Forest
5. Clustering
6. KNN
7. K-Means

4.1. Linear Regression

It is used to estimate real values (cost of houses, number of calls, total sales etc.) based on continuous variable(s). Here, we establish relationship between independent and dependent variables by fitting a best line. This best fit line is known as regression line and represented by a linear equation Y= a *X + b.

The best way to understand linear regression is to relive this experience of childhood. Let us say, you ask a child in fifth grade to arrange people in his class by increasing order of weight, without asking them their weights! What do you think the child will do? He / she would likely look (visually analyze) at the height and build of people and arrange them using a combination of these visible parameters. This is linear regression in real life! The child has actually figured out that height and build would be correlated to the weight by a relationship, which looks like the equation above.

In this equation:

Y – Dependent Variable

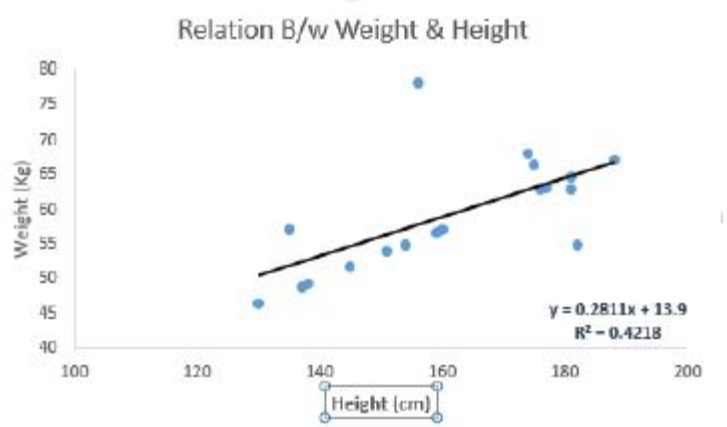
a – Slope

X – Independent variable

b – Intercept

These coefficients a and b are derived based on minimizing the sum of squared difference of distance between data points and regression line.

Look at the below example. Here we have identified the best fit line having linearequation **y=0.2811x+13.9**. Now using this equation, we can find the weight,
knowing the height of a person.



Linear Regression is mainly of two types: Simple Linear Regression and Multiple Linear Regression. Simple Linear Regression is characterized by one independent variable. And, Multiple Linear Regression(as the name suggests) is characterized by multiple (more than 1) independent variables. While finding the best fit line, you can fit a polynomial or curvilinear regression. And these are known as polynomial or curvilinear regression.

4.2. Logistic Regression

Don't get confused by its name! It is a classification not a regression algorithm. It is used to estimate discrete values (Binary values like 0/1, yes/no, true/false) based on given set of independent variable(s). In simple words, it predicts the probability of occurrence of an event by fitting data to a logit function. Hence, it is also known as **logit regression**. Since, it predicts the probability, its output values lies between 0 and 1 (as expected).

Again, let us try and understand this through a simple example.

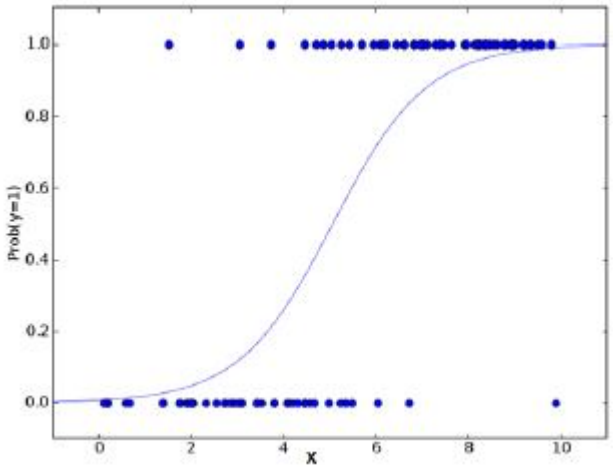
Let's say your friend gives you a puzzle to solve. There are only 2 outcome scenarios – either you solve it or you don't. Now imagine, that you are being given wide range of puzzles / quizzes in an attempt to understand which subjects you are good at. The outcome to this study would be something like this – if you are given a trignometry based tenth grade problem, you are 70% likely to solve it. On the other hand, if it is grade fifth history question, the probability of getting an answer is only 30%. This is what Logistic Regression provides you.

Coming to the math, the log odds of the outcome is modeled as a linear combination of the predictor variables.

odds= p/ (1-p) = probability of event occurrence / probability of not event occurrence
ln(odds) = ln(p/(1-p))
logit(p) = ln(p/(1-p)) = b0+b1X1+b2X2+b3X3....+bkXk

Above, p is the probability of presence of the characteristic of interest. It chooses parameters that maximize the likelihood of observing the sample values rather than that minim
ize the sum of squared errors (like in ordinary regression).

Now, you may ask, why take a log? For the sake of simplicity, let's just say that this is one of the best mathematical way to replicate a step function. I can go in more details, but that will beat the purpose of this article.



4.3. Decision Tree

This is one of my favorite algorithm and I use it quite frequently. It is a type of supervised learning algorithm that is mostly used for classification problems. Surprisingly, it works for both categorical and continuous dependent variables. In this algorithm, we split the population into two or more homogeneous sets. This is done based on most significant attributes/ independent variables to make as distinct groups as possible. For more details, you

In the image above, you can see that population is classified into four different groups based on multiple attributes to identify 'if they will play or not'. To split the population into different heterogeneous groups, it uses various techniques like Gini, Information Gain, Chi-square, entropy.

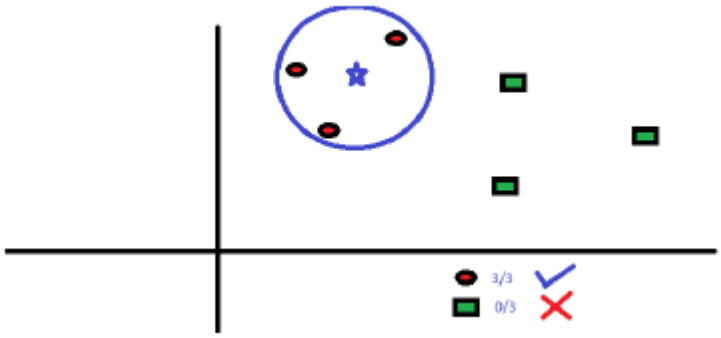
The best way to understand how decision tree works, is to play Jezzball – a classic game from Microsoft (image below). Essentially, you have a room with moving walls and you need to create walls such that maximum area gets cleared off with out the balls.

4.4. kNN (k- Nearest Neighbors)

It can be used for both classification and regression problems. However, it is more widely used in classification problems in the industry. K nearest neighbors is a simple algorithm that stores all available cases and classifies new cases by a majority vote of its k neighbors. The case being assigned to the class is most common amongst its K nearest neighbors measured by a distance function.

These distance functions can be Euclidean, Manhattan, Minkowski and Hamming distance. First three functions are used for continuous function and fourth one (Hamming) for categorical variables. If K = 1, then the case is simply assigned to the class of its nearest neighbor. At times, choosing K turns out to be a challenge while performing kNN modeling.

More: Introduction to k-nearest neighbors : Simplified.



KNN can easily be mapped to our real lives. If you want to learn about a person, of whom you have no information, you might like to find out about his close friends and the circles he moves in and gain access to his/her information!

Things to consider before selecting kNN:

- KNN is computationally expensive
- Variables should be normalized else higher range variables can bias it
- Works on pre-processing stage more before going for kNN like an outlier, noise removal

4.5. K-Means

It is a type of unsupervised algorithm which solves the clustering problem. Its procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters). Data points inside a cluster are homogeneous and heterogeneous to peer groups.

Remember figuring out shapes from ink blots? k means is somewhat similar this activity. You look at the shape and spread to decipher how many different clusters / population are present!

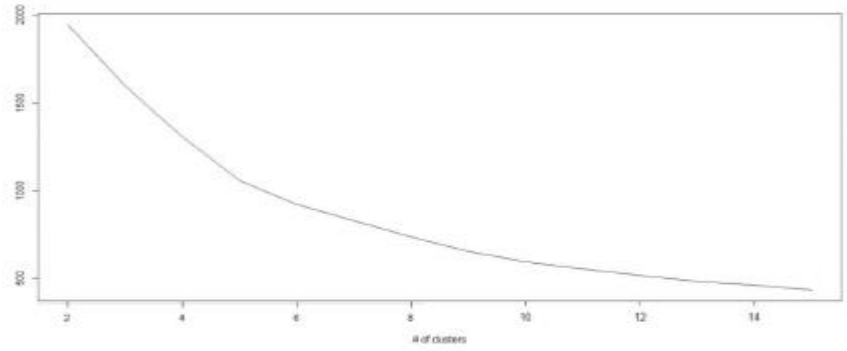
How K-means forms cluster:

1. K-means picks k number of points for each cluster known as centroids.
2. Each data point forms a cluster with the closest centroids i.e. k clusters.
3. Finds the centroid of each cluster based on existing cluster members. Here we have new centroids.
4. As we have new centroids, repeat step 2 and 3. Find the closest distance for each data point from new centroids and get associated with new k-clusters. Repeat this process until convergence occurs i.e. centroids does not change.

How to determine value of K:

In K-means, we have clusters and each cluster has its own centroid. Sum of square of difference between centroid and the data points within a cluster constitutes within sum of square value for that cluster. Also, when the sum of square values for all the clusters are added, it becomes total within sum of square value for the cluster solution.

We know that as the number of cluster increases, this value keeps on decreasing but if you plot the result you may see that the sum of squared distance decreases sharply up to some value of k, and then much more slowly after that. Here, we can find the optimum number of cluster.



4.6. Random Forest

Random Forest is a trademark term for an ensemble of decision trees. In Random Forest, we've collection of decision trees (so known as "Forest"). To classify a new object based on attributes, each tree gives a classification and we say the tree "votes" for that class. The forest chooses the classification having the most votes (over all the trees in the forest).

Each tree is planted & grown as follows:

1. If the number of cases in the training set is N, then sample of N cases is taken at random but with replacement. This sample will be the training set for growing the tree.
2. If there are M input variables, a number m<<M is specified such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node. The value of m is held constant during the forest growing.
3. Each tree is grown to the largest extent possible. There is no pruning.

For more details on this algorithm, comparing with decision tree and tuning model parameters, I would suggest you to read these articles

5. Predictive Modeling

5.1. Definition

- Predictive modeling is the process of using known results to create, process, and validate a model that can be used to forecast future outcomes. It is a tool used in predictive analytics, a data mining technique that attempts to answer the question "what might possibly happen in the future?"
- Two of the most widely used predictive modeling techniques are regression and neural networks.
- Companies can use predictive modeling to forecast events, customer behavior, as well as financial, economic, and market risks.

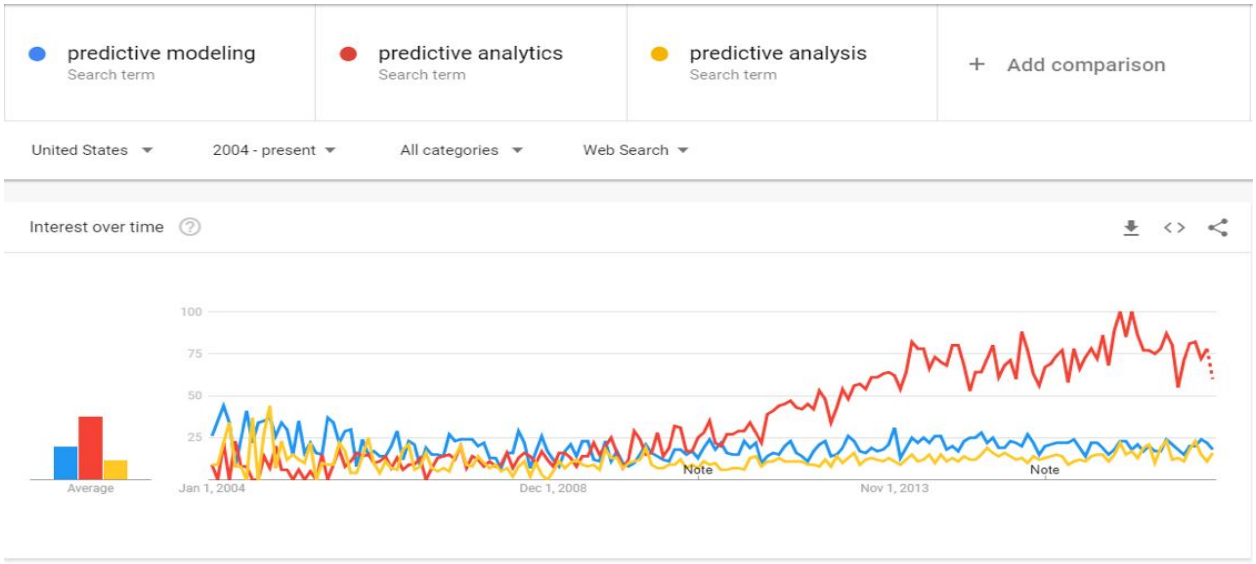
5.2. Making Use Of Past Data predict Future

Predictive modeling is a process that uses data and statistics to predict outcomes with data models. These models can be used to predict anything from sports outcomes and TV ratings to technological advances and corporate earnings.

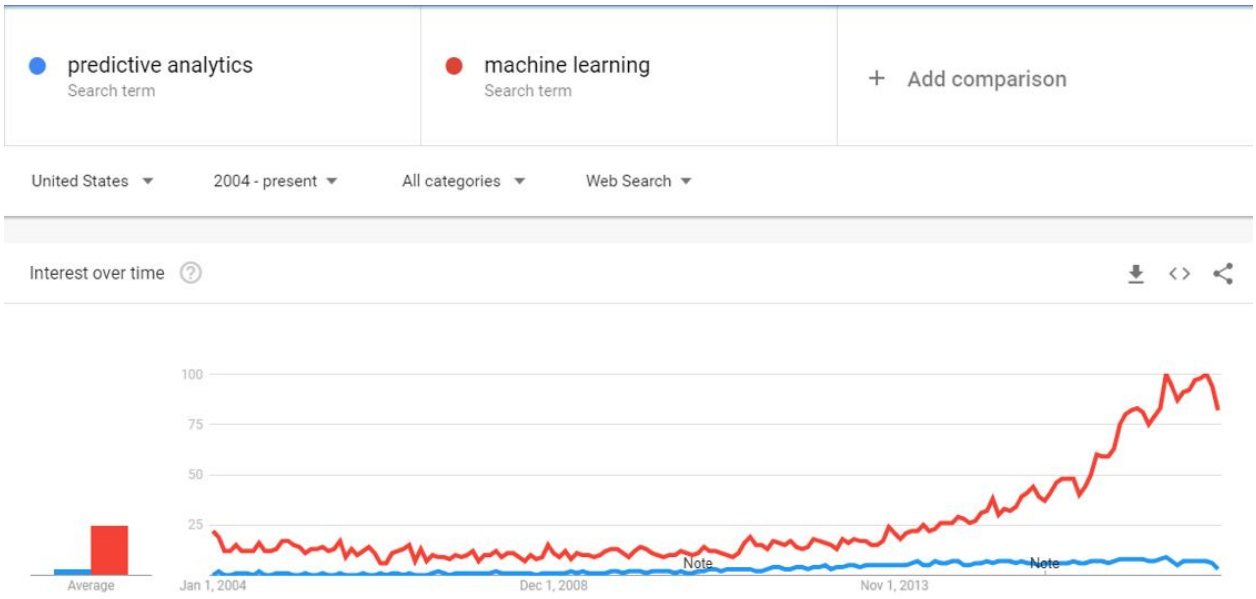
Predictive modeling is also often referred to as:

- Predictive analytics
- Predictive analysis
- Machine learning

These synonyms are often used interchangeably. However, predictive analytics most often refers to commercial applications of predictive modeling, while predictive modeling is used more generally or academically. Of the terms, predictive modeling is used more frequently, which is illustrated in the Google Trends chart below. Machine learning is also distinct from predictive modeling and is defined as the use of statistical techniques to allow a computer to construct predictive models. In practice, machine learning and predictive modeling are often used interchangeably. However, machine learning is a branch of artificial intelligence, which refers to intelligence displayed by machines.



We will primarily use the term “predictive modeling” in this article, but the terms predictive modeling, predictive analytics, predictive analysis, and machine learning can be used interchangeably.



Since 2004, searches for machine learning have been more popular than predictive analytics, and machine learning has steadily risen in search popularity in recent years.

---

6. The Art of Building Effective Predictive Models: 7 Steps to a Better Model

Want to build an accurate model? Rely on technical knowledge. Want a model that’s effective and usable? You’ll need a lot more than just technical chops.

We've seen it over and over again: predictive models that don't work. Maybe they're making false predictions. Or they're accurately predicting something quite irrelevant. Or, sometimes, models can be very accurate – and so very complex that people don't use them.

Models that don't work for users are not useful. And nobody wants to waste time and money developing a non-usable model. So, what can you do to ensure your model a) works correctly and b) meets your company or client's needs?

7 Steps to More Effective Predictive Modeling

Building a predictive model is an art. It's also a process. In our experience, it's often the basics that get overlooked. So, here are our (common-sense) steps to building a better model:

1. Be sure everyone understands the business question.

It's tempting to jump right into a modeling project, especially if you think you know what's needed. But you may not know that as well as

you think. Your client may not know what they really need or want. So, ask. And ask. And ask again. Reframe and restate. Ask in multiple ways. If you don't understand the problem, you won't be able to build the model.

This helps your client, too. They may not realize how vague their goals are! Make sure both parties understand why the model is needed and how it will be used.

Once you understand that, you can build a clear mental predictive framework to answer the business question. This requires translating the business problem to an analytical problem, which means doing a thorough analysis of the data.

2. Do an exploratory data analysis first.

Now that you have a clear idea of the business need, get to know your data. This step will save you time in the long run, so don't gloss over it in your enthusiasm to start building.

Use business understanding to guide your analysis.

It's not enough to simply look at trends; you should understand the relationships between variables. Decompose the problem's components and see what's really happening. For example, suppose you're exploring the relationship between revenue, traffic, price, and incidence in a retail outlet. Look at the different segments – i.e. high-traffic vs. low-traffic stores, stores in high-price areas vs. stores in low-priced areas. There will be different patterns there. Figure out where the drivers fit into your data.

Develop a hypothesis before starting modeling.

Before you model, create a simple chart of your data and study it. You'll learn a lot! You may get a good idea of the answer to the business problem based on the overall trend. At the very least, charting will give you a feel for the relationships between variables. And it will help you spot problems with the data before you get into the actual modeling.

Charting data can also inform your choices of modeling techniques. For example, if you see a non-linear relationship between variables, you won't use a linear regression model.

Understand the interactions between the variables.

If you don't understand how the data is behaving, you won't be able to build a good model. It's as simple as that.

Look for the not-so-obvious connections. Dig deeper to understand the relationships. For example, we generally accept there's a negative relationship between sale quantities and price, but that's not always the case. For some items, a closer examination shows that this accepted theory doesn't tell the full story at all price points.

3. Simpler is usually better.

Resist the urge to plan a complicated, highly technical model just because you have the skills to do it. Consider your user groups: those newer to analytics probably need a simplistic model; experienced analytics consumers might be comfortable with more detail.

More complex models are not always more effective. Simpler models are generally easier to understand, which makes them the most effective for business users. People can see the relationship and quickly assimilate it.

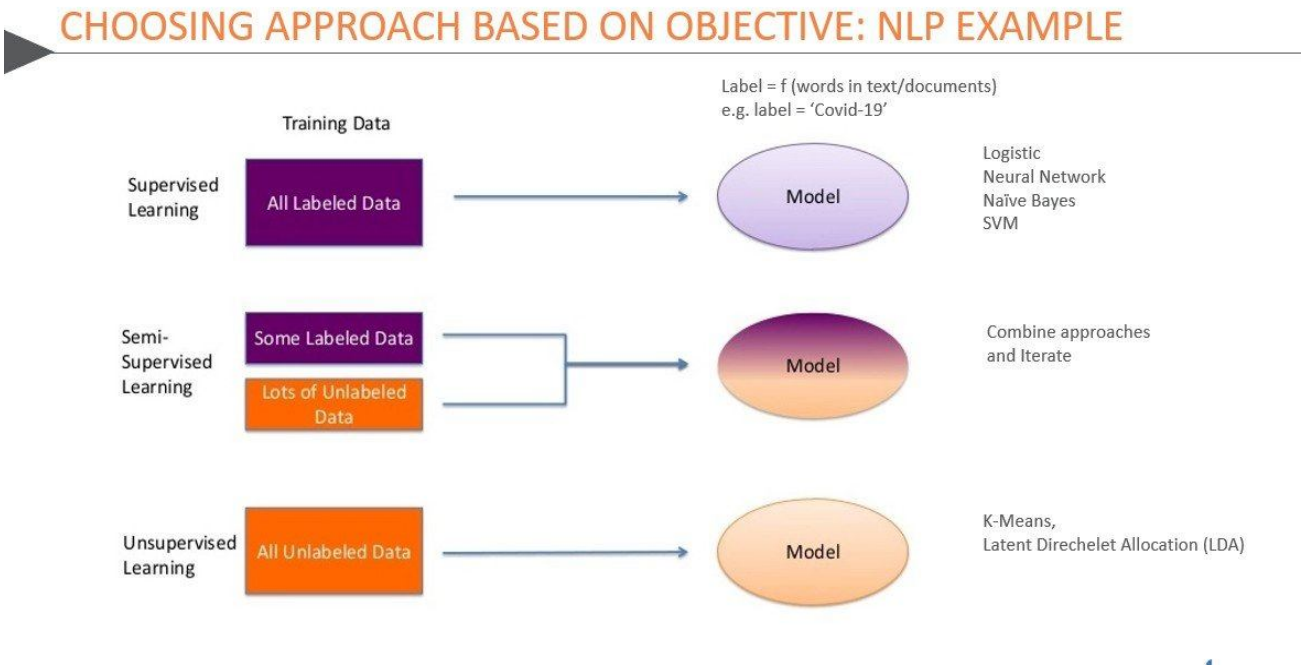
4. Use an iterative model development process.

No one has ever built the perfect model in one shot. Refinements are part of the process. So, revisit your model frequently. If there are problems, be proactive: find another technique, look at another variable. Look at your test data and predictions – are they all wrong? Or are certain data types having issues? Did your model work for some time periods and not others? It's normal to have problems during the build process. Look at them as a chance to improve your work.

5. Understand which techniques are better for which problems.

As humans, we like to use what we know. And we really like to use techniques we're good at. This can cause a problem in predictive modeling since different data types respond better to different modeling techniques.

So, base your approach on the objective and the data you have. For example, supervised learning, semi-supervised learning, and unsupervised learning work with different data types. It's important to make the right choice for your data, as shown in the table below.



And, as you know, there are many options for building a neural network! Again, it's vital to understand your data and the ultimate use of the model to make the best choice.

6. Prepare for the trade-off between interpretability and accuracy.

There will always be a trade-off in a predictive model: interpretability vs. accuracy. The more detailed and technical the model, the higher its accuracy. Unfortunately, there's another correlation between detail and technicality: the higher its levels, the less usable the model. To create a useful model, simplification may be needed.

The deciding factors are the user audience, the client, whether the model is for a product or service, and its scalability. Will simplicity and interpretability be more helpful than higher accuracy?

Also, consider the scope of the business question. If you're looking for the big picture, you may need multiple models to understand the full situation. Will higher levels of detail be helpful in such a model?

7. Build a narrative.

Data points tell a story, but it's not always a complete story. You may have to connect the dots for your audience, especially if you're using multiple models.

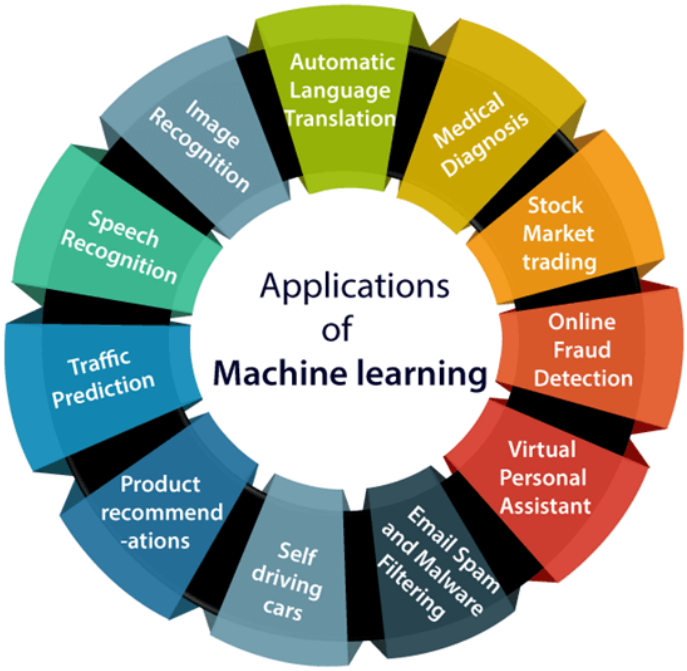
Throughout your analysis, keep the problem (and thus the story) clear in your mind. Focus on the main theme. Identify the main messages and sub-messages you want to share. Then you'll build trust with your audience and create an effective predictive model.

Listen to the entire webcast here and hear our experts share their experiences in building predictive models over the years. Take along the learnings and become a modeling ninja.

---

7. 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.

8. IoT Based Humidity And Temperature Monitoring Using Arduino Uno (Simulation)

8.1. Introduction

Using Internet of Things (IOT), we can control any electronic equipment in homes and industries. Moreover, you can read a data from any sensor and analyse it graphically from anywhere in the world. Here, we can read temperature and humidity data from DHT11 sensor and upload it to a Thing-Speak cloud using Arduino Uno and ESP8266-01 module. Arduino Uno is MCU, it fetch a data of humidity and temperature from DHT11 sensor and Process it and give it to a ESP8266 Module.ESP8266 is a Wi-Fi module, it is one of the leading platform for Internet of Things. It can transfer a data to IOT cloud.

8.2. Hardware Requirements

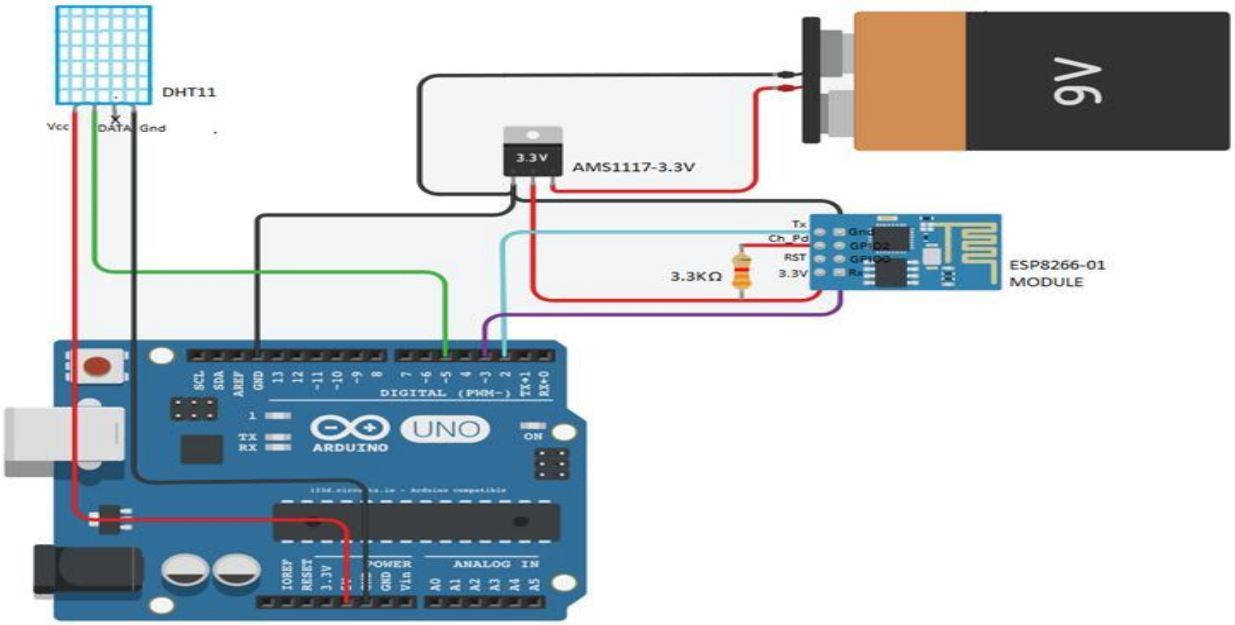
1. Arduino Uno
2. ESP8266-01
3. DHT11
4. AMS1117-3.3V
5. 9V battery

8.3. Software Requirements

1. Tinkercad

Tinkercad is a free, online 3D modeling program that runs in a web browser, known for its simplicity and ease of use. Since it became available in 2011 it has become a popular platform for creating models for 3D printing as well as an entry-level introduction to constructive solid geometry in schools.

8.4. Circuit and it's Working

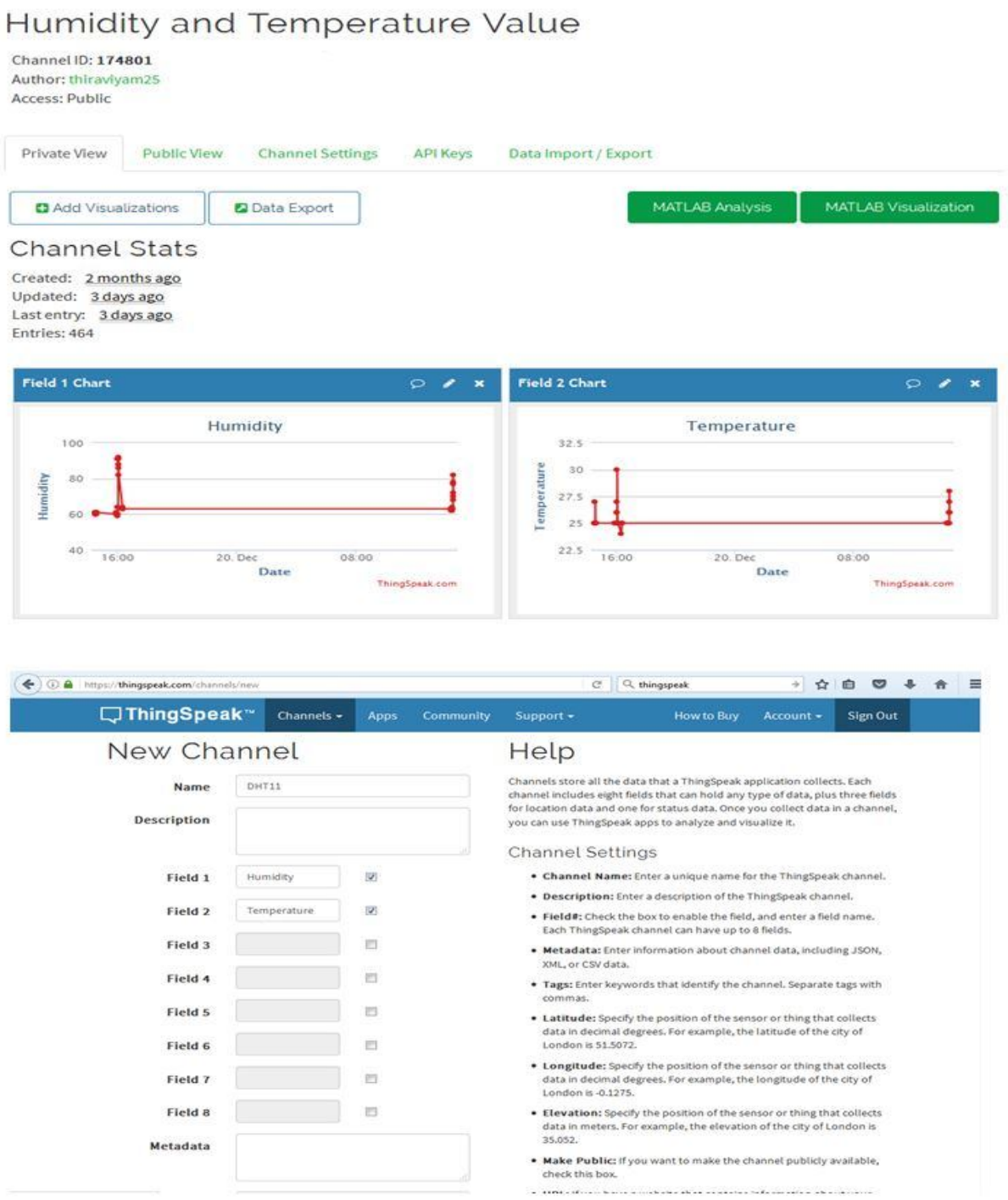


First make the connection as shown in fig: 1.1.The 2nd pin is of DHT11 is a data pin, it can send a temperature and humidity value to the 5th pin of Arduino Uno.1st and 4th pin of DHT11 is a **Vcc** and **Gnd** and 3rd pin is no connection. The Arduino Uno process a temperature and humidity value and send it to a ESP8266 WiFi module. The Tx and Rx pin of ESP8266 is connected to the 2nd (Rx) and 3rd (Tx) of Arduino Uno. Make sure that input voltage of ESP8266 must be 3.3V, not a 5V (otherwise it would damage a device).For that, we are using AMS1117 Voltage regulator circuit. It can regulate a voltage from 9V to 3.3V and will give it to Vcc pin of ESP8266.The Ch_Pd is a chip enable pin of ESP8266 and should be pullup to 3.3V through 3.3KΩ resistor. For reset the module pull down the RST pin of ESP8266 to Gnd.ESP8266 have 2 GPIO pins GPIO 0 and GPIO 2.

8.5. Construction and Testing

Thing-Speak is an open source platform to store and retrieve a data for Internet of Things application. To use this, you need to register in Thing-Speak cloud and then login to your account. After create a new channel with temperature in one field and humidity in another field as shown in Fig: 1.2. Once you created a new channel, it will generate a two API keys, they are READ API keys and WRITE API keys. First, copy the WRITE API keys from ThingsSpeak and paste it into the line (String apiKey = "OX9T8Y9OL9HD0UBP";) of the program. Next, replace the Host_Name and Password with your WiFi name and WiFi password in the two lines given below in the program. (String Host_Name = "Pantech" and String Password = "pantech123")

The Arduino program Uses DHT library, if it is not presented in your arduino IDE, select SketchàInclude libraryà Manage librariesà Install DHT Sensor library. Then compile the program and upload to a Arduino Uno through Arduino IDE. Ensure that WiFi modem and internet connection in your Smartphone or PC are working properly. After uploaded a program, the Temperature and Humidity data is uploaded on Thing-Speak platform. You can see it graphically in the private view window of your channel as shown in Fig: 1.3. And you can able to see the uploaded data from serial port of Arduino IDE.



(From above information we build a simulation of IoT base temperature monitoring system and connect this system to Thing-Speak site and collect a data of temperature and humidity for next step of our project.

The next step is using a data build a machine lerning model in Jupiter not book)

Build a temprature prediction system using Machine Learning

```
## Represented by
       Name                        Roll No                Enrollment No
1) Nikhil Mahajan                    35                     1900180400
2) Rehan Shaikh                      39                     1900180408
3) Gaurav Atkale                     61                     1900180410
4) Vijay Patil                       65                     1900180414



Guided by : K.P. Akole (Sir)
```

Problem Statment

From the given DATA Predict the Temprature

```
# Variable Declaration

Temp = Temprature
Hum  = Humidity
Time = Time
```

In [1]:
```
# import necessary librarys
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
# reading a data using a pandas

df = pd.read_csv ("ETI microproject Data.csv")
```

In [3]:
```
df.head(5)
```

Out[3]:

	Temp	Hum
0	22.0	20.0
1	22.0	20.0
2	22.0	26.0
3	22.0	26.0
4	22.0	20.0

In [4]:
```
# check a dimention of data set
df.shape
```

Out[4]:
```
(336, 2)
```

In [5]:
```
df.columns
```

Out[5]:
```
Index(['Temp', 'Hum'], dtype='object')
```

In [6]:
```
# variabe identification
df.dtypes
```

Out[6]:
```
Temp    float64
Hum     float64
dtype: object
```

Univariate Analysis

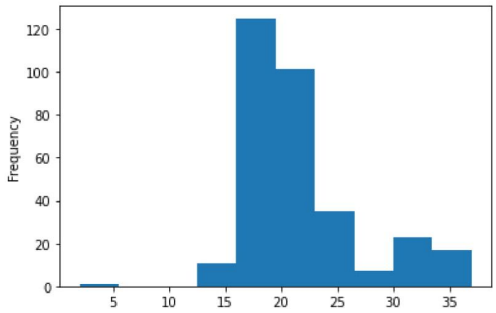
In [7]:
```
# for a discribing continuas variable
df.describe()
```

Out[7]:

	Temp	Hum
count	320.000000	320.000000
mean	21.321875	21.475000
std	5.117383	3.800594
min	2.000000	12.000000
25%	18.000000	20.000000
50%	20.000000	21.000000
75%	23.000000	23.000000
max	37.000000	37.000000

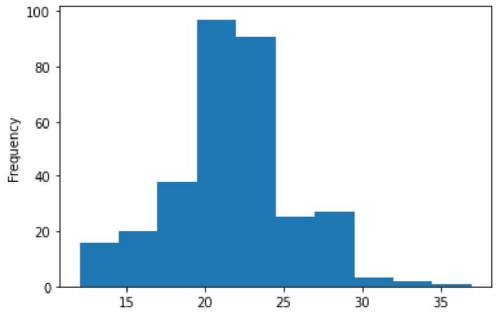
In [8]:
```
# ploting histogram of the temprature variable
df['Temp'].plot.hist()
```

Out[8]: <AxesSubplot:ylabel='Frequency'>



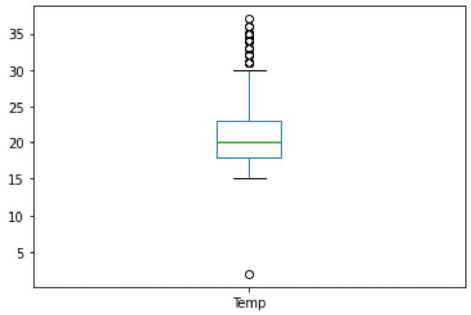
In [9]:
```
# ploting a histogram of the humadity variable
df['Hum'].plot.hist()
```

Out[9]: <AxesSubplot:ylabel='Frequency'>



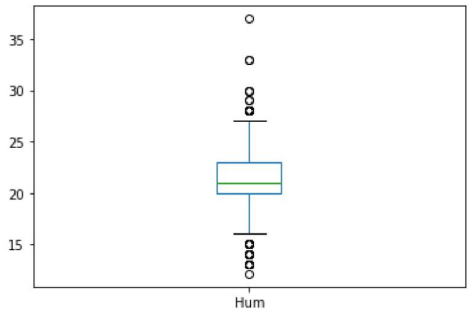
In [10]:
```
# ploting a box for the temprature
df['Temp'].plot.box()
```

Out[10]: <AxesSubplot:>



In [11]:
```
# ploting a box for the humadity
df['Hum'].plot.box()
```

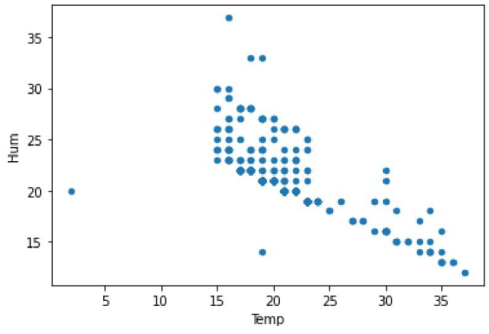
Out[11]: <AxesSubplot:>



Bivariate Analysis

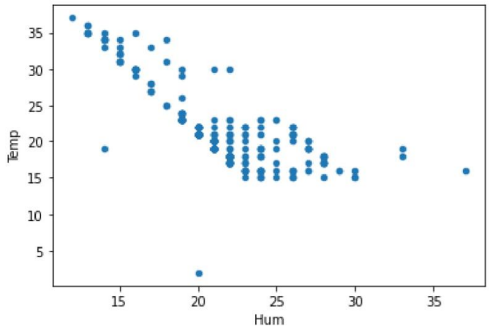
In [12]:
```
# bivariate anyalysis for continuas continuas variable
df.plot.scatter('Temp', 'Hum')
```

Out[12]: <AxesSubplot:xlabel='Temp', ylabel='Hum'>



In [13]:
```
# bivariate anyalysis for continuas continuas variable
df.plot.scatter('Hum', 'Temp')
```

Out[13]:
```
<AxesSubplot:xlabel='Hum', ylabel='Temp'>
```



In [14]:
```
df.corr()
```

Out[14]:

	Temp	Hum
Temp	1.000000	-0.753825
Hum	-0.753825	1.000000

In [15]:
```
df['Temp'].corr(df['Hum'])
```

Out[15]:
```
-0.7538246593306245
```

Missing value treatment

In [16]:
```
# check a missing values
df.describe()
```

Out[16]:

	Temp	Hum
count	320.000000	320.000000
mean	21.321875	21.475000
std	5.117383	3.800594
min	2.000000	12.000000
25%	18.000000	20.000000
50%	20.000000	21.000000
75%	23.000000	23.000000
max	37.000000	37.000000

In [17]:
```
df.isnull()
```

Out[17]:

	Temp	Hum
0	False	False
1	False	False
2	False	False
3	False	False
4	False	False
...	...	...
331	False	False
332	False	False
333	False	False
334	False	False
335	False	False

336 rows × 2 columns

In [18]:
```
df.isnull().sum()
```

Out[18]:
```
Temp    16
Hum     16
dtype: int64
```

In [19]:
```
# fill a missing values with mean of columen
```

In [20]:
```
df['Temp'].fillna(value = (df['Temp'].mean()), inplace = True)
```

In [21]:
```
df['Hum'].fillna(value = (df['Hum'].mean()), inplace = True)
```

In [22]:
```
df.describe()
```

Out[22]:

	Temp	Hum
count	336.000000	336.000000
mean	21.321875	21.475000
std	4.993682	3.708723
min	2.000000	12.000000
25%	18.000000	20.000000
50%	21.000000	21.000000
75%	22.000000	23.000000
max	37.000000	37.000000

In [23]:
```
df.head(5)
```

Out[23]:

	Temp	Hum
0	22.0	20.0
1	22.0	20.0
2	22.0	26.0
3	22.0	26.0
4	22.0	20.0

In [24]:
```
# Split a Data set for Testing and Trening purpose

X = df.drop('Temp', axis = 1)
```

In [25]:
```
y = df.drop('Hum', axis = 1)
```

In [26]:
```
print("shape of X = ", X.shape)
print("shape of y = ", y.shape)
```

```
shape of X =  (336, 1)
shape of y =  (336, 1)
```

In [27]:
```
from sklearn.model_selection import train_test_split
```

In [28]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
```

In [29]:
```
print("shape of X_train = ", X_train.shape )
print("shape of X_test = ", X_test.shape )
print("shape of y_train = ", y_train.shape )
print("shape of y_test = ", y_test.shape )
```

```
shape of X_train =  (268, 1)
shape of X_test =  (68, 1)
shape of y_train =  (268, 1)
shape of y_test =  (68, 1)
```

In [30]:
```
X_train
```

Out[30]:

	Hum
268	21.000
292	26.000
328	22.000
252	20.000
227	21.475
...	...
53	21.000
245	19.000
247	19.000
94	22.000
98	26.000

268 rows × 1 columns

Linear Regression

In [31]:
```
# select a model and train it

from sklearn.linear_model import LinearRegression
```

In [32]:
```
lr = LinearRegression()
```

In [33]:
```
lr.fit(X_train, y_train)
```

Out[33]:
```
LinearRegression()
```

In [34]:
```
# equation of line
# y = mx + c
```

In [35]:
```
# value of m(slope)
lr.coef_
```

Out[35]:
```
array([[-1.00396644]])
```

In [36]:
```
# value of c (intercept)
lr.intercept_
```

Out[36]:
```
array([42.87181601])
```

In [37]:
```
# put value of Humedity to predict a tempreture
lr.predict([[33.0]])
```

Out[37]:
```
array([[9.74092338]])
```

In [38]:
```
y_pred = lr.predict(X_test)
y_pred
```

Out[38]:
```
array([[22.79248714],
       [22.79248714],
       [22.79248714],
       [26.80835292],
       [22.79248714],
       [28.8162858 ],
       [29.82025225],
       [18.77662137],
       [22.79248714],
       [15.76472204],
       [19.78058781],
       [16.76868848],
       [22.79248714],
       [13.75678915],
       [22.79248714],
       [20.78455426],
       [25.80438647],
       [19.78058781],
       [26.80835292],
       [23.79645359],
       [18.77662137],
       [15.76472204],
       [21.7885207 ],
       [17.77265493],
       [27.81231936],
       [21.7885207 ],
       [16.76868848],
       [21.7885207 ],
       [23.79645359],
       [21.7885207 ],
       [21.7885207 ],
       [19.78058781],
       [23.79645359],
       [20.78455426],
       [16.76868848],
       [15.76472204],
       [21.7885207 ],
       [20.78455426],
       [19.78058781],
       [22.79248714],
       [14.7607556 ],
       [20.78455426],
       [20.78455426],
       [22.79248714],
       [21.7885207 ],
       [22.79248714],
       [20.78455426],
       [22.79248714],
       [23.79645359],
       [20.78455426],
       [23.79645359],
       [25.80438647],
       [23.79645359],
       [20.78455426],
       [19.78058781],
       [21.7885207 ],
       [12.75282271],
       [21.7885207 ],
       [23.79645359],
       [29.82025225],
       [12.75282271],
       [20.78455426],
       [28.8162858 ],
       [22.79248714],
       [20.78455426],
       [20.78455426],
       [23.79645359],
       [21.7885207 ]])
```

In [39]:
```
# cheack accurasy of model

lr.score(X_test, y_test)
```

Out[39]:
```
0.6858341701932492
```

Support Vector Regression (SVR)

In [40]:
```
# import SVR from SVM
from sklearn.svm import SVR
```

In [41]:
```
svr_rbf = SVR(kernel = 'rbf')
```

In [42]:
```
svr_rbf.fit(X_train, y_train)
svr_rbf.score(X_test, y_test)
```

```
/srv/conda/envs/notebook/lib/python3.6/site-packages/sklearn/utils/validation.py:63: DataConversionW
arning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(*args, **kwargs)
```

Out[42]:
```
0.8663860285670795
```

In [43]:
```
# using linear kernel
svr_linear = SVR(kernel = 'linear')
svr_linear.fit(X_train, y_train)
svr_linear.score(X_test, y_test)
```

```
/srv/conda/envs/notebook/lib/python3.6/site-packages/sklearn/utils/validation.py:63: DataConversionW
arning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(*args, **kwargs)
```

Out[43]:
```
0.6577265749745045
```

In [44]:
```
# using poly kernel
svr_poly = SVR(kernel = 'poly', degree = 2)
svr_poly.fit(X_train, y_train)
svr_poly.score(X_test, y_test)
```

```
/srv/conda/envs/notebook/lib/python3.6/site-packages/sklearn/utils/validation.py:63: DataConversionW
arning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(*args, **kwargs)
```

Out[44]:
```
0.5483617032968008
```

Random Forest

In [45]:
```
# import random forest from ensemble
from sklearn.ensemble import RandomForestRegressor
```

In [46]:
```
regressor = RandomForestRegressor(n_estimators = 100, criterion = 'mse')
regressor.fit (X_train, y_train)
regressor.score(X_test, y_test)
```

```
/srv/conda/envs/notebook/lib/python3.6/site-packages/ipykernel_launcher.py:2: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
```

Out[46]:
```
0.873051430708889
```

In [47]:
```
regressor = RandomForestRegressor(n_estimators = 500, criterion = 'mse')
regressor.fit (X_train, y_train)
regressor.score(X_test, y_test)
```

```
/srv/conda/envs/notebook/lib/python3.6/site-packages/ipykernel_launcher.py:2: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
```

Out[47]: 0.8708404215126806

In [48]:
```
regressor = RandomForestRegressor(n_estimators = 1000, criterion = 'mse')
regressor.fit (X_train, y_train)
regressor.score(X_test, y_test)
```

```
/srv/conda/envs/notebook/lib/python3.6/site-packages/ipykernel_launcher.py:2: DataConversionWarning:
A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_sample
s,), for example using ravel().
```

Out[48]: 0.8728332679328272

In [ ]: