**ANALYZING DIFFERENT WEATHER CONDITIONS FOR SENSOR APPLICATIONS**

**Abstract**

The main theme of this project is to analyze different weather conditions using sensors. We will implement analysis of weather conditions on the basis of sensor data. We will be using temperature data for analysis of weather conditions. We will use the Temperature Sensor readings for IoT devices dataset for the implementation of our paper. We will use Support Vector Machine (SVM) Classifier for classification of temperature signals. We will train the SVM classifier with the aforementioned dataset and its classification accuracy will be tested using the test data. The data contained in the dataset will be preprocessed before sending for training or testing. The preprocessing stage will include some noise removing techniques namely band pass filter. Later, the classifier’s accuracy will be calculated on the number correct predictions it will make.

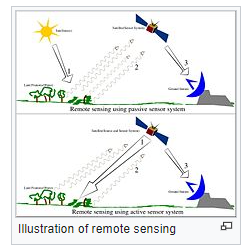
**Keywords:** Sensor data, Internet of Things (IoT), Support Vector Machine (SVM) Classifier, Band pass Filter, Preprocessing Stage.

**Introduction**

Remote sensing is the acquisition of information about an object or phenomenon without making physical contact with the object, in contrast to in situ or on-site observation. The term is applied especially to acquiring information about Earth and other planets. Remote sensing is used in numerous fields, including geophysics, geography, land surveying and most Earth science disciplines (e.g. exploration geophysics, hydrology, ecology, meteorology, oceanography, glaciology, geology); it also has military, intelligence, commercial, economic, planning, and humanitarian applications, among others.

In current usage, the term remote sensing generally refers to the use of satellite- or aircraft-based sensor technologies to detect and classify objects on Earth. It includes the surface and the atmosphere and oceans, based on propagated signals (e.g. electromagnetic radiation). It may be split into "active" remote sensing (when a signal is emitted by a satellite or aircraft to the object and its reflection detected by the sensor) and "passive" remote sensing (when the reflection of sunlight is detected by the sensor).

Remote sensing can be divided into two types of methods: Passive remote sensing and Active remote sensing. Passive sensors gather radiation that is emitted or reflected by the object or surrounding areas. Reflected sunlight is the most common source of radiation measured by passive sensors. Examples of passive remote sensors include film photography, infrared, charge-coupled devices, and radiometers. Active collection, on the other hand, emits energy in order to scan objects and areas whereupon a sensor then detects and measures the radiation that is reflected or backscattered from the target. RADAR and LiDAR are examples of active remote sensing where the time delay between emission and return is measured, establishing the location, speed and direction of an object.

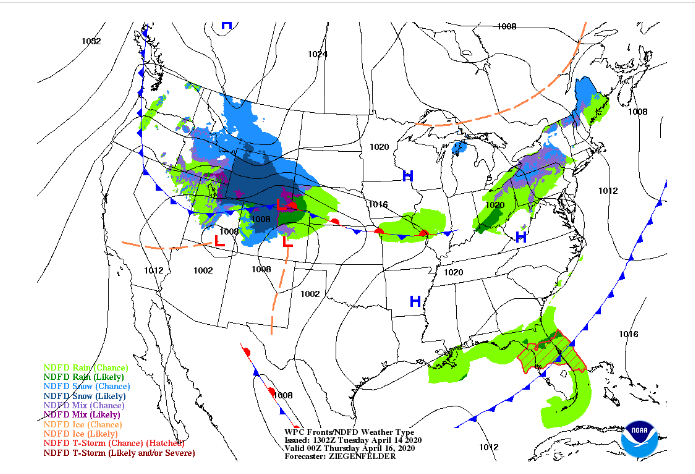


* Conventional radar is mostly associated with aerial traffic control, early warning, and certain large-scale meteorological data. Doppler radar is used by local law enforcements' monitoring of speed limits and in enhanced meteorological collection such as wind speed and direction within weather systems in addition to precipitation location and intensity. Other types of active collection includes plasmas in the ionosphere. Interferometric synthetic aperture radar is used to produce precise digital elevation models of large scale terrain (See RADARSAT, TerraSAR-X, Magellan).
* Laser and radar altimeters on satellites have provided a wide range of data. By measuring the bulges of water caused by gravity, they map features on the seafloor to a resolution of a mile or so. By measuring the height and wavelength of ocean waves, the altimeters measure wind speeds and direction, and surface ocean currents and directions.
* Ultrasound (acoustic) and radar tide gauges measure sea level, tides and wave direction in coastal and offshore tide gauges.
* Light detection and ranging (LIDAR) is well known in examples of weapon ranging, laser illuminated homing of projectiles. LIDAR is used to detect and measure the concentration of various chemicals in the atmosphere, while airborne LIDAR can be used to measure the heights of objects and features on the ground more accurately than with radar technology. Vegetation remote sensing is a principal application of LIDAR.
* Radiometers and photometers are the most common instrument in use, collecting reflected and emitted radiation in a wide range of frequencies. The most common are visible and infrared sensors, followed by microwave, gamma-ray, and rarely, ultraviolet. They may also be used to detect the emission spectra of various chemicals, providing data on chemical concentrations in the atmosphere.

Weather analysis and forecasting involves collecting global meteorological surface and upper-air observations, preparing global surface and upper air pressure, temperature, moisture, and wind analyses at frequent time intervals based upon these observations, solving a closed set of highly nonlinear equations governing.

* Weather forecasting and prediction involves formulating and disseminating information about future weather conditions based upon the collection and analysis of meteorological observations.
* Numerical weather prediction is the prediction of weather phenomena by the numerical solution of the equations governing the motion and changes of condition of the atmosphere.
* Weather analysis and forecasting involves collecting global meteorological surface and upper-air observations, preparing global surface and upper air pressure, temperature, moisture, and wind analyses at frequent time intervals based upon these observations, solving a closed set of highly nonlinear equations governing atmospheric dynamical motions, and applying statistical procedures to the atmospheric simulations to predict a wide variety of weather elements of interest to potential users.
* Now casting is a form of very short-range weather forecasting; that is, the current weather along with forecasts up to about two hours ahead.
* Forecasts in which the lead time for the prediction is more than two weeks, are termed long-range or extended-range climate predictions.

Processes for formulating and disseminating information about future weather conditions based upon the collection and analysis of meteorological observations. Weather forecasts may be classified according to the space and time scale of the predicted phenomena (Fig. 1). Atmospheric fluctuations with a length of less than 100 m (330 ft) and a period of less than 100 s are considered to be turbulent. Prediction of turbulence extends only to establishing its statistical properties, insofar as these are determined by the thermal and dynamic stability of the air and by the aerodynamic roughness of the underlying surface. The study of atmospheric turbulence is called micrometeorology; it is of importance for understanding the diffusion of air pollutants and other aspects of the climate near the ground. Standard meteorological observations are made with sampling techniques that filter out the influence of turbulence. Common terminology distinguishes among three classes of phenomena with a scale that is larger than the turbulent microscale: the mesoscale, synoptic scale, and planetary scale.



The mesoscale includes all moist convection phenomena, ranging from individual cloud cells up to the convective cloud complexes associated with prefrontal squall lines, tropical storms, and the intertropical convergence zone. Also included among mesoscale phenomena are the sea breeze, mountain valley circulations, and the detailed structure of frontal inversions. Because most mesoscale phenomena have time periods less than 12 h, they are little influenced by the rotation of the Earth. The prediction of mesoscale phenomena is an area of active research. Most forecasting methods depend upon empirical rules or the short-range extrapolation of current observations, particularly those provided by radar and geostationary satellites. Forecasts are usually couched in probabilistic terms to reflect the sporadic character of the phenomena. Since many mesoscale phenomena pose serious threats to life and property, it is the practice to issue advisories of potential occurrence significantly in advance. These “watch” advisories encourage the public to attain a degree of readiness appropriate to the potential hazard. Once the phenomenon is considered to be imminent, the advisory is changed to a “warning,” with the expectation that the public will take immediate action to prevent the loss of life.

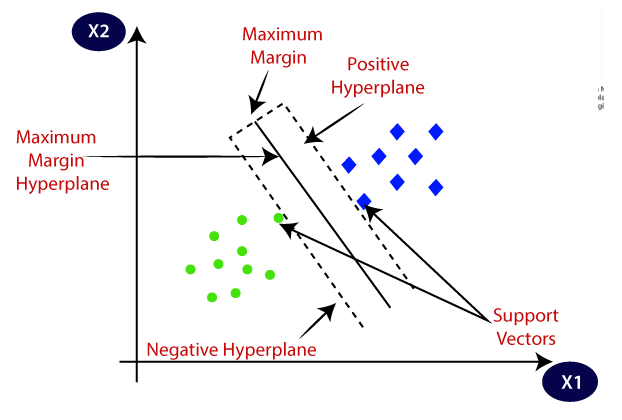
The next-largest scale of weather events is called the synoptic scale, because the network of meteorological stations making simultaneous, or synoptic, observations serves to define the phenomena. The migratory storm systems of the extratropics are synoptic-scale events, as are the undulating wind currents of the upper-air circulation which accompany the storms. The storms are associated with barometric minima, variously called lows, depressions, or cyclones. The sense of the wind rotation about the storm is counterclockwise in the Northern Hemisphere, but clockwise in the Southern Hemisphere. This effect, called geostrophy, is due to the rotation of the Earth and the relatively long period, 3–7 days, of the storm life cycle. Significant progress has been made in the numerical prediction of synoptic-scale phenomena.

**Support Vector Machine or SVM**

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

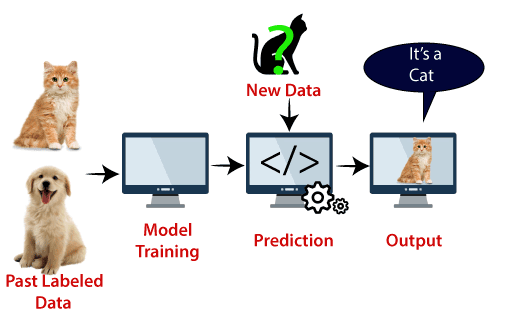
The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:



**Fig: SVM Calculation of Hyper Plane**

Example: SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify it as a cat. Consider the below diagram:



**Fig: Processing of SVM**

**Types of SVM**

SVM can be of two types,

**Linear SVM**

Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.

**Non-linear SVM**

Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

Hyperplane and Support Vectors in the SVM algorithm,

**Hyperplane**

There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.

The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.

We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

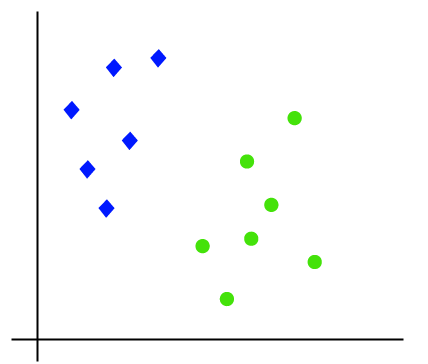
**Support Vectors**

The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

How does SVM works?

**Linear SVM**

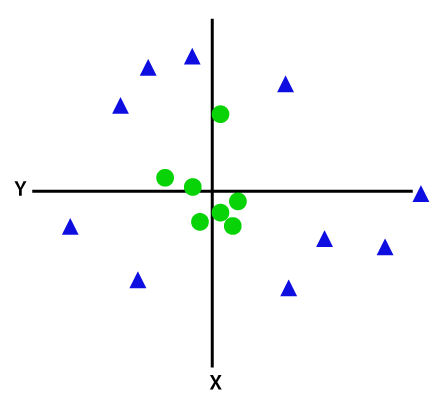
The working of the SVM algorithm can be understood by using an example. Suppose we have a dataset that has two tags (green and blue), and the dataset has two features x1 and x2. We want a classifier that can classify the pair(x1, x2) of coordinates in either green or blue. Consider the below image:



**Linear SVM**

**Non-Linear SVM**

If data is linearly arranged, then we can separate it by using a straight line, but for non-linear data, we cannot draw a single straight line. Consider the below image:



**Fig:** **Non-Linear SVM**

**Literature Review**

**[1] R. B. Alley et al., Advances in weather prediction, Science, 363:342–344, 2019:** Weather forecasting provides numerous societal benefits, from extreme weather warnings to agricultural planning. In recent decades, advances in forecasting have been rapid, arising from improved observations and models, and better integration of these through data assimilation and related techniques. Further improvements are not yet constrained by limits on predictability. Better forecasting, in turn, can contribute to a wide range of environmental forecasting, from forest-fire smoke to bird migrations.

**Summary:** In this paper, we studied about advancements happening in the weather prediction.

**[2] L. Gilbert (ed.), Climatology and Weather Forecasting: An Integrated Approach, Syrawood Publishing House, 2019:** Climatology or climate science is the study of climate, especially weather conditions over a long period of time. The concepts of climatology are used to analyze accumulated data for pattern recognition. Such recognized patterns are used to forecast weather. Recent advances in this field of science have given rise to a number of modern approaches in related areas of research such as paleoclimatology, historical climatology, etc. This book is a compilation of chapters that discuss the most vital concepts and emerging trends in the field of climatology and their application in weather forecasting. It is a complete source of knowledge on the present status of this important field. Easy to understand language and extensive use of examples makes this book an ideal reference text for students and researchers alike.

**Summary:** In this paper, we studied about the climatology and weather forcasting.

**[3] K. A. Teague and N. Gallicchio, The Evolution of Meteorology: A Look into the Past, Present, and Future of Weather Forecasting, Wiley-Blackwell, 2017:** This comprehensive review explores the evolution of the field of meteorology, from its infancy in 3000 bc, through the birth of fresh ideas and the naming of the field as a science, to the technology boom, to today. The Evolution of Meteorology reveals the full story of where meteorology was then to where it is now, where the field is heading, and what needs to be done to get the field to levels never before imagined. Authored by experts of the topic, this book includes information on forecasting technologies, organizations, governmental agencies, and world cooperative projects.

The authors explore the ancient history of the first attempts to understand and predict weather and examine the influence of the very early birth of television, computers, and technologies that are useful to meteorology. This modern-day examination of meteorology is filled with compelling research, statistics, future paths, ideas, and suggestions.

**Summary:** In this paper, we studied about the evolution of meteorology, beginning from the past, present as well as future of weather forecasting.

**Existing Method**

Utilizing the suggested methodology, data gathered from temperature sensors in various contexts is analyzed. The dataset, Temperature Sensor readings for IoT devices, has numerous samples of temperature data gathered from various situations, and this is where the data will come from. The data will be collected and transmitted to the pre-processing stage, where any noise will be eliminated. The subsequent division of the denoised sensor data into training and testing data will involve the majority of the data being divided into training data since better training requires more samples, and the more samples, the better the training. The remaining portion will be viewed as test data and used to evaluate the classifier later.

**Random forest**

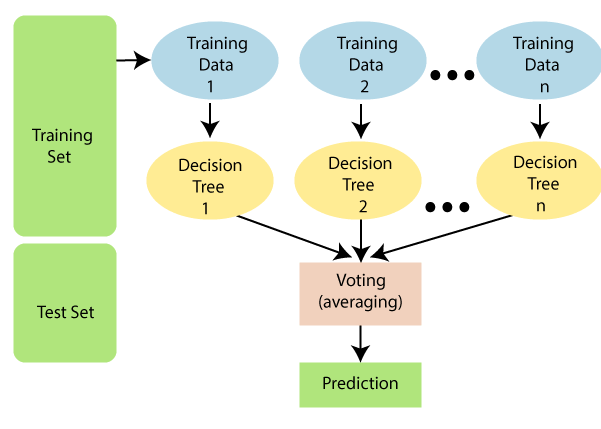
Random forest is a commonly-used machine learning algorithm trademarked by Leo Breiman and Adele Cutler, which combines the output of multiple decision trees to reach a single result. Its ease of use and flexibility have fueled its adoption, as it handles both classification and regression problems.

**Decision trees**

Since the random forest model is made up of multiple decision trees, it would be helpful to start by describing the decision tree algorithm briefly. Decision trees start with a basic question, such as, “Should I surf?” From there, you can ask a series of questions to determine an answer, such as, “Is it a long period swell?” or “Is the wind blowing offshore?”. These questions make up the decision nodes in the tree, acting as a means to split the data. Each question helps an individual to arrive at a final decision, which would be denoted by the leaf node. Observations that fit the criteria will follow the “Yes” branch and those that don’t will follow the alternate path. Decision trees seek to find the best split to subset the data, and they are typically trained through the Classification and Regression Tree (CART) algorithm. Metrics, such as Gini impurity, information gain, or mean square error (MSE), can be used to evaluate the quality of the split.

This decision tree is an example of a classification problem, where the class labels are "surf" and "don't surf."

While decision trees are common supervised learning algorithms, they can be prone to problems, such as bias and overfitting. However, when multiple decision trees form an ensemble in the random forest algorithm, they predict more accurate results, particularly when the individual trees are uncorrelated with each other.



**Fig: Random Forest Algorithm as a Classifier**

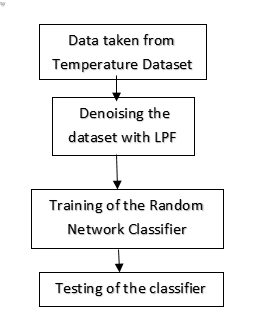
**Assumptions for Random Forest**

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random forest classifier:

* There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
* The predictions from each tree must have very low correlations.

**Uses of the Random Forest algorithm**

* It takes less training time as compared to other algorithms.
* It predicts output with high accuracy, even for the large dataset it runs efficiently.
* It can also maintain accuracy when a large proportion of data is missing.

****

**Fig: Block Diagram of Existing Method**

**How does Random Forest algorithm work?**

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

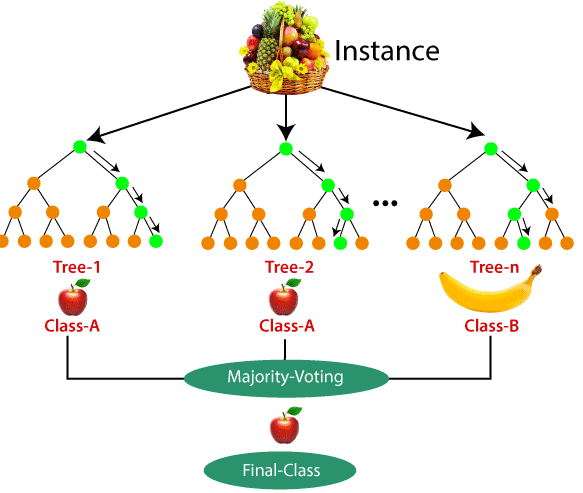
**Step-1:** Select random K data points from the training set.

**Step-2:** Build the decision trees associated with the selected data points (Subsets).

**Step-3:** Choose the number N for decision trees that you want to build.

**Step-4:** Repeat Step 1 & 2.

**Step-5:** For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.



**Fig: Classification using Random Forest Algorithm**

A large number of decision trees are built during the training phase of the random forests or random decision forests ensemble learning approach, which is used for classification, regression, and other tasks. The class that the majority of the trees chose is the output of the random forest for classification problems. The mean or average prediction of each individual tree is returned for regression tasks. The tendency of decision trees to over fit their training set is corrected by random decision forests. Although they frequently outperform decision trees, gradient boosted trees are more accurate than random forests. [Reference needed] But data features can impact how well they work. The classifier will then undergo continual training using these features for better training. After the classifier has been trained, a testing phase will be conducted using test data to evaluate the classifier's accuracy. Then, the accuracy will be determined based on how many accurate forecasts it will make.

**Low-pass filter**

A low-pass filter is a filter that passes signals with a frequency lower than a selected cutoff frequency and attenuates signals with frequencies higher than the cutoff frequency. The exact frequency response of the filter depends on the filter design. The filter is sometimes called a high-cut filter, or treble-cut filter in audio applications. A low-pass filter is the complement of a high-pass filter.

In optics, high-pass and low-pass may have different meanings, depending on whether referring to frequency or wavelength of light, since these variables are inversely related. High-pass frequency filters would act as low-pass wavelength filters, and vice versa. For this reason it is a good practice to refer to wavelength filters as short-pass and long-pass to avoid confusion, which would correspond to high-pass and low-pass frequencies.

Low-pass filters exist in many different forms, including electronic circuits such as a hiss filter used in audio, anti-aliasing filters for conditioning signals prior to analog-to-digital conversion, digital filters for smoothing sets of data, acoustic barriers, blurring of images, and so on. The moving average operation used in fields such as finance is a particular kind of low-pass filter, and can be analyzed with the same signal processing techniques as are used for other low-pass filters. Low-pass filters provide a smoother form of a signal, removing the short-term fluctuations and leaving the longer-term trend.

Filter designers will often use the low-pass form as a prototype filter. That is, a filter with unity bandwidth and impedance. The desired filter is obtained from the prototype by scaling for the desired bandwidth and impedance and transforming into the desired bandform (that is low-pass, high-pass, band-pass or band-stop).

**Implementation**

This project's major focus is on employing sensors to analyse various weather situations. On the basis of sensor data, we will implement analysis of the meteorological conditions. Data on temperatures will be used to analyse the weather. For the implementation of our article, we'll use the dataset for Temperature Sensor Readings for IoT Devices. We will classify temperature signals using a Random Forest classifier. The aforementioned dataset will be used to train the SVM classifier, and test data will be used to gauge its classification accuracy. Before transmitting the dataset for training or testing, the data will be preprocessed. A band pass filter is one of the noise-removal techniques used in the preprocessing stage.

**Disadvantages:**

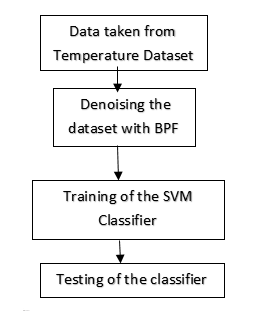
* Choosing a “good” kernel function is not easy.
* Long training time for large datasets.
* Difficult to understand and interpret the final model, variable weights and individual impact.

**Proposed System**

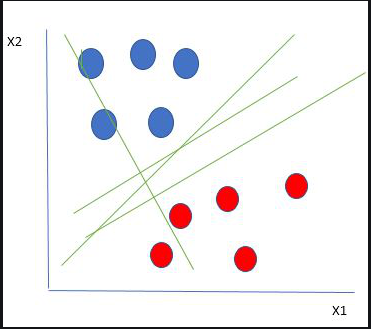
This proposed method is implemented to analyze the data collected from the sensors of temperature from different environments. The data will be taken from Temperature Sensor readings for IoT devices dataset, the dataset contains many samples of temperature data collected from different environments. The data will be taken and sent to pre-processing stage of where the noise will be removed from the data if present. The denoised sensor data will then be divided into training and testing data, majority of the data will be divided into training data because we need more samples for a better training, the more the samples the better the training. The remaining part will be considered as testing data which will be used for testing the classifier later.

**Support Vector Machine (SVM)**

Support Vector Machine (SVM) is a supervised machine learning algorithm used for both classification and regression. Though we say regression problems as well its best suited for classification. The objective of the SVM algorithm is to find a hyperplane in an N-dimensional space that distinctly classifies the data points. The dimension of the hyperplane depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a 2-D plane. It becomes difficult to imagine when the number of features exceeds three.



**Fig: Proposed Methodology**



**Fig: Linearly Separable Data**

The Support Vector Machine Classifier (SVM) is a classification algorithm that uses support vectors for classification. The support vectors are calculated on the basis of closest distance from the Hyper Plane. The Hyper Plane is any random line that will be drawn between classes of input for differentiating them which acts as a boundary between them. Later, the support vectors will be selected from each of the classes which are very close to the Hyper Plane. The classification will be done by analyzing the support vectors and their properties. In this method, the support vectors are sensor data that are from each of the classes namely hot, cold or normal etc. The classifier will then be trained on these support vectors continuously for a better training. Later, the classifier’s training will be followed by a testing phase which use the test data to test the accuracy of the classifier. The accuracy will then be calculated on the basis of number of correct predictions it will make.

From the figure above it’s very clear that there are multiple lines (our hyperplane here is a line because we are considering only two input features x1, x2) that segregate our data points or do a classification between red and blue circles. So how do we choose the best line or in general the best hyperplane that segregates our data points?

**Multiclass Classification Using SVM**

In its most simple type, SVM doesn’t support multiclass classification natively. It supports binary classification and separating data points into two classes. For multiclass classification, the same principle is utilized after breaking down the multiclassification problem into multiple binary classification problems.

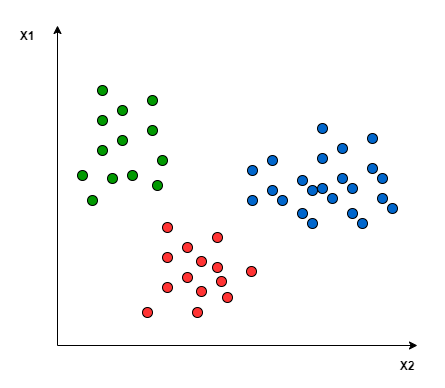
The idea is to map data points to high dimensional space to gain mutual linear separation between every two classes. This is called a One-to-One approach, which breaks down the multiclass problem into multiple binary classification problems. A binary classifier per each pair of classes.

Another approach one can use is One-to-Rest. In that approach, the breakdown is set to a binary classifier per each class.

A single SVM does binary classification and can differentiate between two classes. So that, according to the two breakdown approaches, to classify data points from m classes data set:

In the One-to-Rest approach, the classifier can use \pmb{m} SVMs. Each SVM would predict membership in one of the \pmb{m} classes.

In the One-to-One approach, the classifier can use \pmb{\frac{m (m-1)}{2}} SVMs.

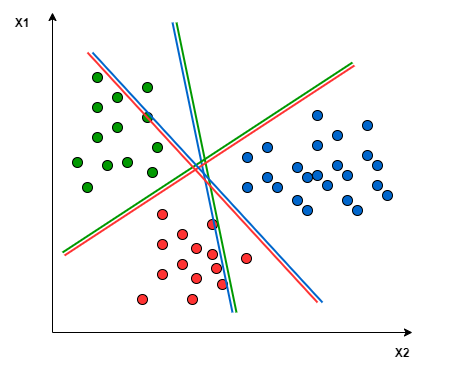


**Fig: Multiple Classes classification using Multi-Class SVM**

Applying the two approaches to this data set results in the followings:

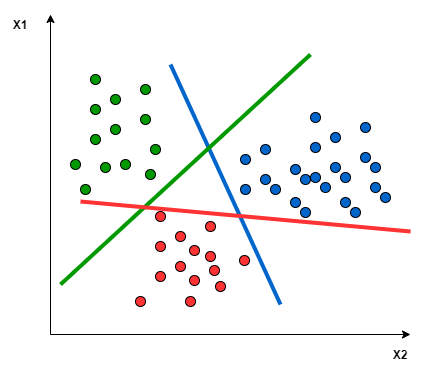
In the One-to-One approach, we need a hyperplane to separate between every two classes, neglecting the points of the third class. This means the separation takes into account only the points of the two classes in the current split. For example, the red-blue line tries to maximize the separation only between blue and red points.

It has nothing to do with green points:



**Fig: Multi-Class SVM calculating Multiple Hyper Planes for Multiple Classes**

In the One-to-Rest approach, we need a hyperplane to separate between a class and all others at once. This means the separation takes all points into account, dividing them into two groups; a group for the class points and a group for all other points. For example, the green line tries to maximize the separation between green points and all other points at once:



**Fig: Multi-Class SVM calculating Multiple Hyper Planes for Multiple Classes**

**ADVANTAGES and APPLICATIONS**

**Advantages:**

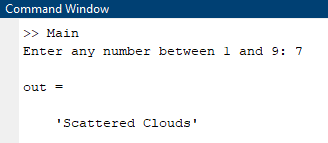
* Gives the better classification results.
* Accuracy is more.

**Applications:**

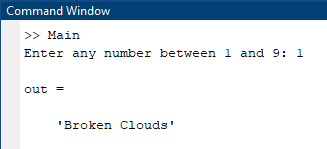
There are numerous applications for the use of Biometric Technology, but the most common ones are as follows:

* Wireless Sensor Networks (WSN)
* Internet of Things (IoT)
* Weather Forecasting
* Weather Monitoring
* Satellite Communication

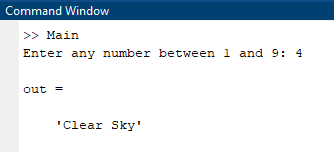
**Results**



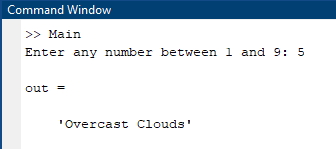
**Fig: Output Prediction1 of Classifier**



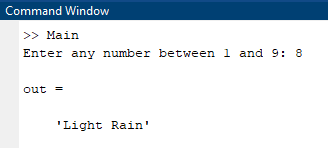
**Fig: Output Prediction2 of Classifier**

****

**Fig: Output Prediction3 of Classifier**

****

**Fig: Output Prediction4 of Classifier**

****

**Fig: Output Prediction5 of Classifier**

**SOFTWARE and HARDWARE REQUIREMENTS**

**Software & Hardware Requirements:**

**Software Requirements:**

MATLAB R2018a or above

**Hardware Requirements:**

**Operating Systems:**

• Windows 10

• Windows 7 Service Pack 1

• Windows Server 2019

• Windows Server 2016

**Processors:**

Minimum: Any Intel or AMD x86-64 processor

Recommended: Any Intel or AMD x86-64 processor with four logical cores and AVX2 instruction set support.

**Disk:**

Minimum: 2.9 GB of HDD space for MATLAB only, 5-8 GB for a typical installation

Recommended: An SSD is recommended a full installation of all MathWorks products may take up to 29 GB of disk space

**RAM:**

Minimum: 4 GB

Recommended: 8 GB

**BIBLIOGRAPHY**

**Introduction To Matlab**

What Is MATLAB?

The name MATLAB stands for Matrix Laboratory. The software is built up around vectors and matrices. This makes the software particularly useful for linear algebra but MATLAB is also a great tool for solving algebraic and differential equations and for numerical integration. MATLAB has powerful graphic tools and can produce nice pictures in both 2D and 3D. It is also a programming language, and is one of the easiest programming languages for writing mathematical programs. These factors make MATLAB an excellent tool for teaching and research.

MATLAB was written originally to provide easy access to matrix software developed by the LINPACK (linear system package) and EISPACK (Eigen system package) projects. It integrates computation, visualization, and programming environment. Furthermore, MATLAB is a modern programming language environment: it has sophisticated data structures, contains built-in editing and debugging tools, and supports object-oriented programming. MATLAB has many advantages compared to conventional computer languages (e.g., C, FORTRAN) for solving technical problems.

MATLAB abilities a family of add-on software program utility software application software program software utility software-unique solutions called toolboxes. Very essential to maximum customers of MATLAB, toolboxes assist you to studies and observe specialized technology. Toolboxes are entire collections of MATLAB abilities (M-files) that increase the MATLAB surroundings to remedy precise schooling of problems. Areas in which toolboxes are to be had embody signal processing, manipulate systems, neural networks, fuzzy correct judgment, wavelets, simulation, and hundreds of others.

It has powerful built-in routines that enable a very wide variety of computations. It also has easy to use graphics commands that make the visualization of results immediately available. Specific applications are collected in packages referred to as toolbox. There are toolboxes for signal processing, symbolic computation, control theory, simulation, optimization, and several other fields of applied science and engineering. MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. The software package has been commercially available since 1984 and is now considered as a standard tool at most universities and industries worldwide.

Brief History of MATLAB:

Cleve Moler, the chairman of the computer science department at the University of New Mexico, started developing MATLAB in the late 1970s. The first MATLAB® was not a programming language; it was a simple interactive matrix calculator. There were no programs, no toolboxes, no graphics and no ODEs or FFTs. He designed it to give his student’s access to LINPACK and EISPACK without them having to learn FORTRAN. It soon spread to other universities and found a strong audience within the applied mathematics community. The mathematical basis for the first version of MATLAB was a series of research papers by J. H. Wilkinson and 18 of his colleagues, published between 1965 and 1970 and later collected in Handbook for Automatic Computation, Volume II, Linear Algebra, edited by Wilkinson and C. Reinsch. These papers present algorithms, implemented in Algol 60, for solving matrix linear equation and Eigen value problems.

In the 1970s and early 1980s, I was teaching Linear Algebra and Numerical Analysis at the University of New Mexico and wanted my students to have easy access to LINPACK and EISPACK without writing FORTRAN programs. By “easy access,” I meant not going through the remote batch processing and the repeated edit-compile-link-load-execute process that was ordinarily required on the campus central mainframe computer. Jack little, an engineer, was exposed to it during a visit Moler made to Stanford University in 1983. Recognizing its commercial potential, he joined with Moler and Steve Bangert. They rewrote MATLAB in C and founded Math Works in 1984 to continue its development. These rewritten libraries were known as JACKPAC. In 2000, MATLAB was rewritten to use a newer set of libraries for matrix manipulation, LAPACK. MATLAB was first adopted by researchers and practitioners in control engineering, Little's specialty, but quickly spread to many other domains. It is now also used in education, in particular the teaching of linear algebra and numerical analysis, and is popular amongst scientists involved in video processing.

EISPACK and LINPACK:

In 1970, a group of researchers at Argonne National Laboratory proposed to the U.S. National Science Foundation (NSF) to “explore the methodology, costs, and resources required to produce, test, and disseminate high-quality mathematical software and to test, certify, disseminate, and support packages of mathematical software in certain problem areas.” The group developed EISPACK (Matrix Eigen system Package) by translating the Algol procedures for Eigen value problems in the handbook into FORTRAN and working extensively on testing and portability. The first version of EISPACK was released in 1971 and the second in 1976.

In 1975, four of us Jack Dongarra, Pete Stewart, Jim Bunch, and myself proposed to the NSF another research project that would investigate methods for the development of mathematical software. A byproduct would be the software itself, dubbed LINPACK, for Linear Equation Package. This project was also centered at Argonne. LINPACK originated in FORTRAN; it did not involve translation from Algol. The package contained 44 subroutines in each of four numeric precisions. In a sense, the LINPACK and EISPACK projects were failures. We had proposed research projects to the NSF to “explore the methodology, costs, and resources required to produce, test, and disseminate high-quality mathematical software.” We never wrote a report or paper addressing those objectives. We only produced software.

So, I studied Niklaus Wirth’s book Algorithms + Data Structures = Programs and learned how to parse programming languages. I wrote the first MATLAB an acronym for Matrix Laboratory in FORTRAN, with matrix as the only data type. The project was a kind of hobby, a new aspect of programming for me to learn and something for my students to use. There was never any formal outside support, and certainly no business plan. This first MATLAB was just an interactive matrix calculator. This snapshot of the start-up screen shows all the reserved words and functions. There are only 71. To add another function, you had to get the source code from me, write a FORTRAN subroutine, add your function name to the parse table, and recompile MATLAB.

Starting MATLAB:

After logging into your account, you can enter MATLAB by double-clicking on the MATLAB shortcut icon (MATLAB 7.0.4) on your Windows desktop. When you start MATLAB, a special window called the MATLAB desktop appears. The desktop is a window that contains other windows. The major tools within or accessible from the desktop are:

The Command Window

The Command History

The Workspace

The Current Directory

The Help Browser

Current Folder: This panel allows you to access the project folders and files.

Command Window: This is the main area where commands can be entered at the command line. It is indicated by the command prompt (>>).

Workspace: The workspace shows all the variables created and/or imported from files.

Command History: This panel shows or return commands that are entered at the command line.

Help Browser:

The critical way to get assist online is to use the MATLAB help browser, opened as a separate window every through clicking at the question mark photograph (?) on the computing tool toolbar, or through manner of typing assist browser on the spark off in the command window. The assist Browser is an internet browser blanketed into the MATLAB computing tool that shows a Hypertext Markup Language (HTML) files. The Help Browser consists of panes, the help navigator pane, used to find out information, and the show pane, used to view the information. Self-explanatory tabs apart from navigator pane are used to performs are searching out.

MATLAB language:

This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create complete large and complex application programs.

MATLAB working environment:

This is the set of tools and facilities that you work with as the MATLAB user or programmer. It includes facilities for managing the variables in your workspace and importing and exporting data. It also includes tools for developing, managing, debugging, and profiling M-files, MATLAB's applications.

MATLAB mathematical function library:

This is a vast collection of computational algorithms ranging from elementary functions like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix eigenvalues, Bessel functions, and fast Fourier transforms.

MATLAB Application Program Interface (API):

This is a library that allows you to write C and FORTRAN programs that interact with MATLAB. It includes facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files.

MATLAB DESKTOP:

MATLAB Desktop is the precept MATLAB utility window. The computing tool includes five sub home windows, the command window, the workspace browser, the modern-day-day list window, the command records window, and one or greater decide domestic windows, which is probably confirmed high-quality on the identical time due to the truth the client suggests a photo. The command window is in which the character types MATLAB instructions and expressions at the spark off (>>) and in which the output of these commands is displayed. MATLAB defines the workspace because the set of variables that the client creates in a bit consultation. The workspace browser suggests those variables and some facts about them. Double clicking on a variable within the workspace browser launches the Array Editor, which may be used to gain statistics and profits instances edit exceptional homes of the variable.

The modern-day-day-day Directory tab above the workspace tab suggests the contents of the cutting-edge list, whose path is shown inside the modern-day list window. For example, in the home windows on foot machine the path is probably as follows: C: MATLAB Work, indicating that listing “artwork” is a subdirectory of the number one list “MATLAB”; WHICH IS INSTALLED IN DRIVE C. Clicking on the arrow within the modern list window suggests a listing of these days used paths. Clicking at the button to the right of the window permits the individual to trade the present day listing. MATLAB uses a seeking out path to find out M-documents and one-of-a-type MATLAB associated documents, which can be put together in directories within the computer document tool. Any report run in MATLAB need to be dwelling in the modern-day-day listing or in a list that is on is looking for course. By default, the documents supplied with MATLAB and math works toolboxes are included inside the searching out direction. The first-rate manner to look which directories are on the searching out route. The satisfactory manner to appearance which directories are speedy the quest route, or to characteristic or regulate a searching for course, is to pick out outset path from the File menu the computing device, and then use the set course talk discipline. It is proper exercise to feature any generally used directories to the hunt route to avoid again and again having the exchange the cutting-edge-day listing.

The Command History Window contains a file of the instructions a person has entered in the command window, together with every contemporary-day and former MATLAB periods. Previously entered MATLAB instructions can be determined on and re-completed from the command statistics window thru proper clicking on a command or series of commands. This movement launches a menu from which to select numerous options similarly to executing the commands. This is useful to select out abilities options in addition to executing the instructions. This is a beneficial feature at the equal time as experimenting with numerous commands in a piece session.

Using the MATLAB Editor to create M-Files:

The MATLAB editorial manager is a literary substance proofreader particular for growing M-facts and a graphical MATLAB debugger. The supervisor can seem in a window through command facts technique for itself, or it is probably a right-clicking inside the PC. M-information this gadget signified through the use of the expansion .M, as in pixel up.M. The MATLAB editorial supervisor window has a few draws down menus for obligations collectively with sparing, seeing, and troubleshooting facts. Since it plays more than one easy test and furthermore affects utilization of shade to separate among exclusive variables of code, this article editorial supervisor is often supported due to reality the system of a need for composing and altering M-talents. To open the manager, type at enact opens the M-document filename. M in a supervisor window, sorted out for enhancing. As stated earlier than, the file should be inside the cutting-edge posting, or in a posting in the seeking out direction.

Features of MATLAB:

Following are the basic features of MATLAB.

It is a high-level language for numerical computation, visualization and application development.

It also provides an interactive environment for iterative exploration, design and problem solving.

It provides vast library of mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, numerical integration and solving ordinary differential equations.

It provides built-in graphics for visualizing data and tools for creating custom plots.

MATLAB's programming interface gives development tools for improving code quality maintainability and maximizing performance.

It provides tools for building applications with custom graphical interfaces.

It provides functions for integrating MATLAB based algorithms with external applications and languages such as C, Java, .NET and Microsoft Excel.

Uses of MATLAB:

MATLAB is widely used as a computational tool in science and engineering encompassing the fields of physics, chemistry, math and all engineering streams. It is used in a range of applications including

Signal Processing and Communications

Video and Video Processing

Control Systems

Test and Measurement

Computational Finance

Computational Biology

Applications of MATLAB:

MATLAB can be used as a tool for simulating various electrical networks but the recent developments in MATLAB make it a very competitive tool for Artificial Intelligence, Robotics, Video processing, Wireless communication, Machine learning, Data analytics and whatnot. Though it’s mostly used by circuit branches and mechanical in the engineering domain to solve a basic set of problems its application is vast. It is a tool that enables computation, programming and graphically visualizing the results. The basic data element of MATLAB as the name suggests is the Matrix or an array. MATLAB toolboxes are professionally built and enable you to turn your imaginations into reality. MATLAB programming is quite similar to C programming and just requires a little brush up of your basic programming skills to start working with.

Below are a few applications of MATLAB –

Statistics and machine learning (ML)

This toolbox in MATLAB can be very handy for the programmers. Statistical methods such as descriptive or inferential can be easily implemented. So is the case with machine learning. Various models can be employed to solve modern-day problems. The algorithms used can also be used for big data applications.

Curve fitting

The curve fitting toolbox helps to analyze the pattern of occurrence of data. After a particular trend which can be a curve or surface is obtained, its future trends can be predicted. Further plotting, calculating integrals, derivatives, interpolation, etc. can be done.

Control systems

Systems nature can be obtained. Factors such as closed-loop, open-loop, its controllability and observability, Bode plot, NY Quist plot, etc. can be obtained. Various controlling techniques such as PD, PI and PID can be visualized. Analysis can be done in the time domain or frequency domain.

Signal Processing

Signals and systems and digital signal processing are taught in various engineering streams. But MATLAB provides the opportunity for proper visualization of this. Various transforms such as Laplace, Z, etc. can be done on any given signal. Theorems can be validated. Analysis can be done in the time domain or frequency domain. There are multiple built-in functions that can be used.

Mapping

Mapping has multiple applications in various domains. For example, in Big Data, the Map Reduce tool is quite important which has multiple applications in the real world. Theft analysis or financial fraud detection, regression models, contingency analysis, predicting techniques in social media, data monitoring, etc. can be done by data mapping.

Deep learning

It’s a subclass of machine learning which can be used for speech recognition, financial fraud detection, and medical video analysis. Tools such as time-series, Artificial neural network (ANN), Fuzzy logic or combination of such tools can be employed.

Financial analysis

An entrepreneur before starting any endeavor needs to do a proper survey and the financial analysis in order to plan the course of action. The tools needed for this are all available in MATLAB. Elements such as profitability, solvency, liquidity, and stability can be identified. Business valuation, capital budgeting, cost of capital, etc. can be evaluated.

Video processing

The most common application that we observe almost every day are bar code scanners, selfie (face beauty, blurring the background, face detection), video enhancement, etc. The digital video processing also plays quite an important role in transmitting data from far off satellites and receiving and decoding it in the same way. Algorithms to support all such applications are available.

Text analysis

Based on the text, sentiment analysis can be done. Google gives millions of search results for any text entered within a few milliseconds. All this is possible because of text analysis. Handwriting comparison in forensics can be done. No limit to the application and just one software which can do this all.

Electric vehicles designing

Used for modeling electric vehicles and analyze their performance with a change in system inputs. Speed torque comparison, designing and simulating of a vehicle, whatnot.

Aerospace

This toolbox in MATLAB is used for analyzing the navigation and to visualize flight simulator.

Audio toolbox

Provides tools for audio processing, speech analysis, and acoustic measurement. It also provides algorithms for audio and speech feature extraction and audio signal transformation.

COMMUNICATION:

Communications System Toolbox™ offers algorithms and gear for the layout, simulation, and analysis of communications systems. These capabilities are furnished as MATLAB ® features, MATLAB System gadgets™, and Simulink ® blocks. The machine toolbox includes algorithms for source coding, channel coding, interleaving, modulation, equalization, synchronization, and channel modeling. Tools are supplied for bit blunders charge evaluation, producing eye and constellation diagrams, and visualizing channel characteristics. The machine toolbox additionally provides adaptive algorithms that allow you to version dynamic communications structures that use OFDM, OFDMA, and MIMO techniques. Algorithms support fixed-point facts arithmetic and C or HDL code era.

Key Features

▪ Algorithms for designing the physical layer of communications systems, which includes supply coding, channel coding, interleaving, modulation, channel fashions, MIMO, equalization, and synchronization

▪ GPU-enabled System objects for computationally intensive algorithms together with Turbo, LDPC, and Viterbi decoders

▪ Interactive visualization equipment, consisting of eye diagrams, constellations, and channel scattering capabilities

▪ Graphical tool for evaluating the simulated bit mistakes rate of a machine with analytical outcomes

▪ Channel models, consisting of AWGN, Multipath Rayleigh Fading, Rician Fading, MIMO Multipath Fading, and

LTE MIMO Multipath Fading

▪ Basic RF impairments, along with nonlinearity, section noise, thermal noise, and section and frequency offsets

▪ Algorithms available as MATLAB features, MATLAB System objects, and Simulink blocks

▪ Support for fixed-point modeling and C and HDL code technology

System Design, Characterization, and Visualization:

The layout and simulation of a communications gadget requires analyzing its reaction to the noise and interference inherent in real-world environments, reading its behavior the usage of graphical and quantitative manner, and determining whether the resulting overall performance meets requirements of acceptability. Communications System Toolbox implements a selection of obligations for communications machine layout and simulation. Many of the functions, System objects™, and blocks inside the device toolbox perform computations associated with a specific thing of a communications gadget, consisting of a demodulator or equalizer. Other talents are designed for visualization or evaluation.

System Characterization

The system toolbox offers several standard methods for quantitatively characterizing system performance:

▪ Bit error rate (BER) computations

▪ Adjacent channel power ratio (ACPR) measurements

▪ Error vector magnitude (EVM) measurements

▪ Modulation error ratio (MER) measurements

Because BER computations are fundamental to the characterization of any communications system, the system toolbox provides the following tools and capabilities for configuring BER test scenarios and accelerating BER simulations:

BER tool— A graphical user interface that enables you to analyze BER performance of communications systems. You can analyze performance via a simulation-based, semi analytic, or theoretical approach.

Error Rate Test Console — A MATLAB object that runs simulations for communications systems to measure error rate performance. It supports user-specified test points and generation of parametric performance plots and surfaces. Accelerated performance can be realized when running on a multi core computing platform.

Multi core and GPU acceleration — A capability provided by Parallel Computing Toolbox™ that enables you to accelerate simulation performance using multi core and GPU hardware within your computer.

Distributed computing and cloud computing support — Capabilities provided by Parallel Computing Toolbox and MATLAB Distributed Computing Server™ that enable you to leverage the computing power of your server farms and the Amazon EC2 Web service. Performance Visualization. The system toolbox provides the following capabilities for visualizing system performance:

Channel visualization tool — For visualizing the characteristics of a fading channel

Eye diagrams and signal constellation scatter plots — for a qualitative, visual understanding of system behavior that enables you to make initial design decisions

Signal trajectory plots — for a continuous picture of the signal’s trajectory between decision points

BER plots — for visualizing quantitative BER performance of a design candidate, parameterized by metrics such as SNR and fixed-point word size

Analog and Digital Modulation

Analog and digital modulation strategies encode the facts circulation into a sign this is appropriate for transmission. Communications System Toolbox presents some of modulation and corresponding demodulation abilities. These talents are available as MATLAB features and gadgets, MATLAB System Modulation sorts provided by the toolbox are:

Source and Channel Coding

Communications System Toolbox affords source and channel coding talents that can help you develop and compare communications architectures fast, enabling you to discover what-if eventualities and avoid the need to create coding competencies from scratch.

Source Coding

Source coding, also referred to as quantization or signal formatting, is a manner of processing facts a good way to lessen redundancy or prepare it for later processing. The system toolbox offers a diffusion of styles of algorithms for imposing source coding and interpreting, inclusive of:

▪ Quantizing

▪ Companding (µ-law and A-law)

▪ Differential pulse code modulation (DPCM)

▪ Huffman coding

▪ Arithmetic coding

Channel Coding

▪ orthogonal area-time block code (OSTBC) (encoder and decoder for MIMO channels)

▪ Turbo encoder and decoder examples

The gadget toolbox offers application functions for developing your personal channel coding. You can create generator polynomials and coefficients and syndrome deciphering tables, in addition to product parity-take a look at and generator matrices.

The system toolbox additionally presents block and convolutional interleaving and deinters leaving functions to reduce facts errors as a result of burst mistakes in a conversation machine:

Block, including General block interleaver, algebraic interleaver, helical scan interleaver, matrix interleaver, and random interleaver.

Convolutional, including General multiplexed interleaver, convolutional interleaver, and helical interleaver

Channel Modeling and RF Impairments

Channel Modeling

Communications System Toolbox provides algorithms and tools for modeling noise, fading, interference, and different distortions which might be commonly found in communications channels. The system toolbox supports the subsequent styles of channels:

▪ Additive white Gaussian noise (AWGN)

▪ Multiple-enter multiple-output (MIMO) fading

▪ Single-enter single-output (SISO), Rayleigh, and Rician fading

▪ Binary symmetric

A MATLAB channel object provides a concise, configurable implementation of channel models, enabling you to

specify parameters such as:

▪ Path delays

▪ Average path gains

▪ Maximum Doppler shifts

▪ K-Factor for Rician fading channels

▪ Doppler spectrum parameters

For MIMO systems, the MATLAB MIMO channel object expands these parameters to also include:

▪ Number of transmit antennas (up to 8)

▪ Number of receive antennas (up to 8)

▪ Transmit correlation matrix

▪ Receive correlation matrix

To combat the effects noise and channel corruption, the system toolbox provides block and convolutional coding and decoding techniques to implement error detection and correction. For simple error detection with no inherent correction, a cyclic redundancy check capability is also available. Channel coding capabilities provided by the system toolbox include:

▪ BCH encoder and decoder

▪ Reed-Solomon encoder and decoder

▪ LDPC encoder and decoder

▪ Convolutional encoder and Viterbi decoder

RF Impairments

To model the effects of a non-ideal RF front end, you can introduce the following impairments into your communications system, enabling you to explore and characterize performance with real-world effects:

▪ Memory less nonlinearity

▪ Phase and frequency offset

▪ Phase noise

▪ Thermal noise

You can include more complex RF impairments and RF circuit models in your design using SimRF™.

Equalization and Synchronization

Communications System Toolbox lets you discover equalization and synchronization strategies. These techniques are usually adaptive in nature and tough to design and symbolize. The machine toolbox affords algorithms and tools that will let you swiftly select the proper approach on your communications machine. Equalization To compare one-of-a-kind techniques to equalization, the device toolbox offers you with adaptive algorithms which include:

▪ LMS

▪ Normalized LMS

▪ Variable step LMS

▪ Signed LMS

▪ MLSE (Viterbi)

▪ RLS

▪ CMA

These adaptive equalizers are available as nonlinear decision feedback equalizer (DFE) implementations and as

Linear (symbol or fractionally spaced) equalizer implementations.

Synchronization

The device toolbox provides algorithms for each service segment synchronization and timing phase synchronization. For timing section synchronization, the machine toolbox presents a MATLAB Timing Phase Synchronizer object that offers the following implementation techniques:

▪ Early-late gate timing method

▪ Gardner’s method

▪ Fourth-order nonlinearity method

Stream Processing in MATLAB and Simulink

Most verbal exchange structures cope with streaming and frame-primarily based statistics using a aggregate of temporal processing and simultaneous multi frequency and multichannel processing. This form of streaming multidimensional processing can be visible in superior communication architectures consisting of OFDM and MIMO. Communications System Toolbox enables the simulation of advanced communications structures via helping move processing and frame-based simulation in MATLAB and Simulink. In MATLAB, circulate processing is enabled by way of System items™, which use MATLAB objects to symbolize time-based and facts-driven algorithms, sources, and sinks. System objects implicitly manipulate many information of flow processing, including information indexing, buffering, and management of set of rules state. You can mix System gadgets with fashionable MATLAB functions and operators. Most System items have a corresponding Simulink block with the identical abilities. Simulink handles circulation processing implicitly with the aid of coping with the float of information thru the blocks that make up a Simulink model. Simulink is an interactive graphical environment for modeling and simulating dynamic systems that uses hierarchical diagrams to symbolize a machine version. It includes a library of widespread-reason, predefined blocks to represent algorithms, resources, sinks, and device hierarchy.

Implementing a Communications System

Fixed-Point Modeling Many communications systems use hardware that requires a fixed-point representation of your design.

Communications System Toolbox supports fixed-point modeling in all relevant blocks and System objects™ with tools that help you configure fixed-point attributes.

Fixed-point support in the system toolbox includes:

▪ Word sizes from 1 to 128 bits

▪ Arbitrary binary-point placement

▪ Overflow handling methods (wrap or saturation)

▪ Rounding methods: ceiling, convergent, floor, nearest, round, simplest, and zero

Fixed-Point Tool in Simulink Fixed Point™ facilitates the conversion of floating-point data types to fixed point. For configuration of fixed-point properties, the tool tracks overflows and maxima and minima.

Code Generation

Once you've got advanced your set of rules or communications device, you can robotically generate C code from it for verification, rapid prototyping, and implementation. Most System gadgets, functions, and blocks in Communications System Toolbox can generate ANSI/ISO C code the use of MATLAB Coder™, Simulink Coder™, or Embedded Coder™. A subset of System gadgets and Simulink blocks also can generate HDL code. To leverage present highbrow belongings, you can choose optimizations for specific processor architectures and integrate legacy C code with the generated code.

You can also generate C code for both floating-point and fixed-point data types.

DSP Proto typing DSPs are used in communication system implementation for verification, rapid prototyping, or final hardware implementation. Using the processor-in-the-loop (PIL) simulation capability found in Embedded Coder, you can verify generated source code and compiled code by running your algorithm’s implementation code on a target processor. FPGA Prototyping

FPGAs are used in communication systems for implementing high-speed signal processing algorithms. Using the FPGA-in-the-loop (FIL) capability found in HDL Verifier™, you can test RTL code in real hardware for any existing HDL code, either manually written or automatically generated HDL code.

**CONCLUSION**

We can finally conclude that, we’ve implemented a Machine Learning (ML) classifier known as Support Vector Machines (SVM) which was trained using the data of weather conditions of a city named Delhi, over the course of a year. The data contained weather conditions from all of the seasons happened in that city. The SVM classifier has been trained over the different parameters such as, temperature, pressure, humidity and wind speed etc. and with labels for the same weather conditions as well, the weather conditions included were, ‘Broken Clouds’, ‘ Scattered Clouds’, ‘Light Rain’, ‘Clear Sky’ and ‘Overcast Clouds’ as well. The classifier was later tested on the same parameters it produced a very close predictions of the same weather conditions with an accuracy close to 89%.

**FUTURE SCOPE**

After the implementation of predicting weather conditions using a Machine Learning (ML) classifier, we get to know that, the classifiers can be extended to Deep Learning (DL) as well. Due to the emergence of deep learning classifiers, the predictions becoming much closer to the actual values. The deep learning classifiers such as, Artificial Neural Networks (ANN), Perceptron including Multi-layered Perceptron and most efficient classifier Convolutional Neural Networks (CNN) can produce a results with an accuracy of over 92%. The classifiers can be trained using the same parameters that we’ve done for the current ML classifier.

**REFERENCES**

**[1]** A. Blum, The Weather Machine: A Journey Inside the Forecast, Ecco, 2019.

**[2]** L. Gilbert (ed.), Climatology and Weather Forecasting: An Integrated Approach, Syrawood Publishing House, 2019.

**[3]** K. A. Teague and N. Gallicchio, The Evolution of Meteorology: A Look into the Past, Present, and Future of Weather Forecasting, Wiley-Blackwell, 2017.