

UNDERWATER SURFACE TARGET PREDICTION THROUGH SONAR

Submitted in partial fulfillment of the requirements
for the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

NALLA SAI KIRAN (Reg. No. 37110486)
MUNAGALA PAVAN V N SAI BHAGAVAN (Reg. no. 37110467)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI
SALAI,**

CHENNAI – 600 119

MARCH – 2021



SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with “A” grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND

ENGINEERING BONAFIDE CERTIFICATE

This is to certify that this project report is the bonafide work of **NALLA SAI KIRAN (Reg. No. 37110486)** and **MUNAGALA PAVAN V N SAI BHAGAVAN (Reg. No.**

37110467) who carried out the project entitled “**UNDERWATER SURFACE TARGET PREDICTION THROUGH SONAR**” under my supervision from August 2020 to March 2021.

Internal guide

Dr.L. SUJI HELEN, M.E., Ph.D.,

Head of the Department

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **MUNAGALA PAVAN V N SAI BHAGAVAN** hereby declare that the Project Report entitled “**UNDERWATER SURFACE TARGET PREDICTION THROUGH SONAR**” is done by me under the guidance of **Dr.L. SUJI HELEN, M.E., Ph.D.**, Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D., Dean**, School of Computing, **Dr. S. Vigneswari, M.E., Ph.D., and Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.L. SUJI HELEN, M.E., Ph.D., Assistant Professor**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Sonar signals recognition is an important task in detecting the presence of some significant objects under the sea. In military, sonar signals are used in lieu of visuals to navigate underwater and/or locate enemy submarines in proximity. In particular, classification algorithm in data mining has been applied in sonar signal recognition for recognizing the type of surfaces from which the sonar waves are bounced. Classification algorithms in traditional data mining approach offer fair accuracy by training a classification model with the full dataset, in batches. It is well known that sonar signals are continuous and they are collected as data streams. Although the earlier classification algorithms are effective in traditional batch training, it may not be practical for incremental classifier learning. Since sonar signal data streams can amount to infinity, the data preprocessing time must be kept to a minimum to fulfill the need for high speed. This paper presents an alternative data mining strategy suitable for the progressive purging of noisy data via fast conflict analysis from the data stream without the need to learn from the whole dataset at one time. Simulation experiments are conducted and superior results are observed in supporting the efficacy of the methodology.

TABLE OF CONTENTS

ABSTRACT	v
LIST OF FIGURES	ix

CHAPTER No.	TITLE	PAGE NO.
1.	INTRODUCTION	1
	1.1 INTRODUCTION	1
	1.2 PURPOSE OF PROJECT	2
	1.3 RESEARCH AND SIGNIFICANCE	2
2.	LITERATURE SURVEY	4
	2.1 THE USE OF THE AREA UNDER THE ROC CURVE IN THE EVALUATION OF MACHINE LEARNING ALGORITHMS	4
	2.2 PERFORMANCE MEASURES OF MACHINE LEARNING	4
	2.3 STATISTICS-BASED NOISE DETECTION METHODS	5
	2.4 CLASSIFICATION-BASED NOISE DETECTION METHODS	5
	2.5 SIMILARITY-BASED NOISE DETECTION METHODS	6
3.	AIM AND SCOPE	7
	3.1 AIM OF PROJECT	7
	3.1.1 OBJECTIVE	7
	3.2 SCOPE	7
	3.2.1 ADVANTAGES	7

4.	SYSTEM DESIGN & METHODOLOGY	8
4.1	EXISTING SYSTEM	8
4.1.1	DISADVANTAGES OF EXISTING SYSTEM	9
4.2	PROPOSED FRAMEWORK	9
4.2.1	ADVANTAGES OF PROPOSED SYSTEM	10
4.3	PROPOSED FRAME WORK METHODS	10
4.4	REQUIREMENT SPECIFICATION	12
4.5	SYSTEM STUDY	12
4.6	SOFTWARE FUNCTIONALITIES	13
4.7	MODULES	16
 5	 SOFTWARE DEVELOPMENT METHODOLOGY	 19
5.1	DESCRIPTION OF DIAGRAM	19
5.2	ACCURACY CALCULATION	22
5.3	PARAMETERS IN THE CONFUSION MATRIX	24
5.4	PYTHON	25
5.5	NUMPY ARRAY	25
5.6	PANDAS	26
5.7	MATPLOTLIB	27
5.7.1	DENSITY PLOT	28
5.7.2	MATPLOTLIB - BOX PLOT	30
5.7.3	MATPLOTLIB HISTOGRAM	32
5.8	SCIKIT-LEARN	34
5.8.1	LOGISTIC REGRESSION	34
5.8.2	K-NEAREST NEIGHBORS' ALGORITHM	36

	5.8.3	GAUSSIAN NAIVE BAYES	38
	5.8.4	SUPPORT VECTOR MACHINE	39
6		RESULTS AND DISCUSSION	41
7		CONCLUSION AND FUTURE WORK	44
		CONCLUSION	44
		FUTURE WORK	44
		REFERENCES	45
		Appendix	
		A. SCREENSHOTS	46
		B. PLAGIARISM REPORT	51
		C. PAPER WORK	52
		D. SOURCE CODE	57

LIST OF FIGURES

FIGURE No.	FIGURE NAME	PAGE No.
4.1	Overview of a proposed system	11
5.1	Process flow diagram	19
5.2	Architecture diagram	20
5.3	Correlation matrix	21
5.4	Confusion matrix for the model predicted values	24
6.1	Box-plot for accuracy output of scaled algorithms	41
6.2	Accuracy of SVM algorithm	41
6.3	Accuracy values of boosting algorithms with cross_val_score	42
6.4	Classification report	42
6.5	Output	43

CHAPTER

1

INTRODUCT ION

1.1 INTRODUCTION

Data mining is extracting information and knowledge from huge amount of data. Data mining is an essential step in discovering knowledge from databases. There are numbers of databases, data marts, data warehouses all over the world. Data Mining is mainly used to extract the hidden information from a large amount of database. Data mining is also called as Knowledge Discovery Database (KDD). The data mining has four main techniques namely Classification, Clustering, Regression, and Association rule. Data mining techniques have the ability to rapidly mine vast amount of data.

Machine learning has drawn the attention of maximum part of today's emerging technology, related from banking to many consumers and product-based industries, by showing the advancements in the predictive analytics. The main aim is to emanate a capable prediction representative method united by the machine learning algorithmic characteristics, which can deduce if the target of the sound wave is a rock, a mine, any other organism or any kind of foreign body. This proposed work is a clear-cut case study which comes up with a machine learning plan for the grading of rocks and minerals, executed on a huge, highly spatial and complex SONAR dataset. The attempts are done on highly spatial SONAR dataset and achieved an accuracy of 83.17% and area under curve (AUC) came out to be 0.92. With random forest algorithm, the results are further optimized by feature selection to get the accuracy of 85.7%. Persuade results are found when the fulfilment of the designed groundwork is set side by side with the standard classifiers like SVM, random forest, etc. Different evaluation metrics like accuracy, sensitivity, etc. are investigated. Machine learning is performing a major role in improving the quality of detection of underwater natural resources and will tend be better paradigm

Data mining is mainly needed in many fields to extract useful information from a large amount of data. The fields like the medical field, business field, and

educational field have a vast amount of data, thus these fields data can be mined through those techniques more useful information. Data mining techniques can be implemented through a machine learning algorithm. Each technique can be extended using certain

machine learning models. In this system, a heart disease data set is used. The main aim of this system is to predict the possibilities of occurring heart disease of the patients in terms of percentage. This is performed through data mining classification techniques.

The classification technique is used for classifying the entire dataset into two categories namely yes and No. Classification technique is applied to the dataset through the machine learning classification algorithm namely Decision tree classification and Naïve Bayes Classification models. These models are used to enhance the accuracy level of the classification technique. This model performs both the classification and prediction methods. These models are performed using python Programming Language.

1.2 PURPOSE OF PROJECT

The core concept of this paper is to predict rock or mine present in the Underwater surface of the water. Here we have used different types machine learning algorithms like, Decision Trees like, Naive Bayes, KNN. This system uses 60 attributes as input and with that input we have to predict rock or mine.

1.3 RESEARCH AND SIGNIFICANCE

The main concern of analysis in the field of machine learning is being to form a scheduled computational machine for the categorizing the forecast of the objects, based on the attainable information. The outcome of proposed framework helps to predict the triggered sound waves reflect back from what surface: Rock or a Mine.

Broadly in physical world or realistic issues, there is no curb over the types of data. Some dire pre-processing like removal of missing values, feature selection, etc. are always required. Machine learning focuses on taking up contemporary techniques to process huge amount of complex data with lower expense. Feature extraction is the important pre-processing step to classification of complex signal such as vision, speech identification or the problem mentioned here of detecting objects in sonar returns. Input dimensionality of these problems becomes a serious drawback to classification. Feature extraction techniques such as PCA, and neural discriminating

analysis (NDA) reduce a high dimensional signal to a lower dimensional feature set, which preserves the most useful and relevant information on the feature space.

A total of six classification algorithms were put under test of sonar recognition. Support vector machine (SVM), DT, K-nearest neighbours' classifier, Naïve bayes. An incremental version of algorithm that is modified from traditional naive Bayesian called updateable naive Bayesian (NBup) is included as well for intellectual curiosity. Basically, all the six algorithms can be used in either batch learning or incremental learning mode. In the incremental learning mode, the data is treated as a data stream where the model is trained and updated section by section with conflict analysis enforced in effect. As the window slides along, the next new data instance ahead of the window is used to test the trained model. The performance statistics are thereby accumulated from the start till the end.

CHAPTER 2

LITERATURE SURVEY

2.1 THE USE OF THE AREA UNDER THE ROC CURVE IN THE EVALUATION OF MACHINELEARNING ALGORITHMS

Predictive accuracy has been used as the main and often only evaluation criterion for the predictive performance of classification learning algorithms. In recent years, the area under the ROC (Receiver Operating Characteristics) curve, or simply AUC, has been proposed as an alternative single-number measure for evaluating learning algorithms. In this paper, we prove that AUC is a better measure than accuracy. More specifically, we present rigorous definitions on consistency and discrimen in comparing two evaluation measures for learning algorithms. We then present empirical evaluations and a formal proof to establish that AUC is indeed statistically consistent and more discriminating than accuracy. Our result is quite significant since we formally prove that, for the first time, AUC is a better measure than accuracy in the evaluation of learning algorithms.

2.2 PERFORMANCE MEASURES OF MACHINE LEARNING

Customer temporal behavioral data was represented as images in order to perform churn prediction by leveraging deep learning architectures prominent in image classification. Supervised learning was performed on labelled data of over 6 million customers using deep convolutional neural networks, which achieved an AUC of 0.743 on the test dataset using no more than 12 temporal features for each customer. Unsupervised learning was conducted using autoencoders to better understand the reasons for customer churn. Images that maximally activate the hidden units of an autoencoder trained with churned customers reveal ample opportunities for action to be taken to prevent churn among strong data, no voice users.

The Wisconsin Breast Cancer Dataset has been heavily cited as a benchmark dataset for classification. Neural Network techniques such as Neural Networks, Probabilistic Neural Networks, and Regression Neural Networks have been shown to perform very well on this dataset. However, despite its obvious practical importance and implications for cancer research, a thorough investigation of all modern classification techniques on this dataset remains to be done. In this paper we examine the efficacy of classifiers such as Random Forests with varying number of trees, Support Vector Machines with different kernels, Naive Bayes model and neural networks on the accuracy of classifying the masses in the dataset as benign/malignant. Results indicate that Support Vector machines with a Radial Basis function kernel give the best accuracy of all the models attempted. This indicates that there are non-linearities present in the dataset and that the Support vector machine does a good job of mapping the data into a higher dimensional space in which the non-linearities fade away and the data becomes linearly separable by large margin classifier like the support vector machine. These methods show that modern machine learning methods could provide for improved accuracy for early prediction of cancerous tumors.

2.3 STATISTICS-BASED NOISE DETECTION METHODS.

Hamed Komari Alaie proposed another strategy for identifying focuses in detached sonars utilizing versatile limit. In this technique, target signal (sound) is handled as expected and recurrence area. For grouping, Bayesian order is utilized and back dissemination is assessed by Maximum Likelihood Estimation calculation. At long last, target was recognized by joining the location focuses in the two areas utilizing Least Mean Square (LMS) versatile channel.

2.4 CLASSIFICATION-BASED NOISE DETECTION METHODS

Dhiraj Neupane Underwater acoustics has been executed for the most part in the field of sound route and going (SONAR) strategies for submarine correspondence, the assessment of sea resources and climate reviewing, target and item acknowledgment, and estimation and investigation of acoustic sources in the submerged environment. With the fast improvement in science and innovation, the

headway in sonar frameworks has expanded, bringing about a decrement in submerged setbacks. The sonar signal handling and programmed target acknowledgment utilizing sonar signs or symbolism is itself a difficult interaction. Then,

exceptionally progressed data driven AI and profound learning-based techniques are being actualized for getting a few sorts of data from submerged sound data. This paper audits the new sonar programmed target acknowledgment, following, or discovery works utilizing profound learning calculations. A careful investigation of the accessible works is done, and the working strategy, results, and other essential insights about the data procurement measure, the dataset utilized, and the data in regard to hyper- boundaries is introduced in this article.

2.5 SIMILARITY-BASED NOISE DETECTION METHODS

This gathering of techniques for the most part requires a reference by which data are contrasted with measure how comparable or divergent they are to the reference. **Zhiyuan Zhang et al** the analysts initial separated data into numerous subsets prior to looking for the subset that would cause the best decrease in uniqueness inside the preparation dataset whenever eliminated. The disparity capacity can be any capacity restoring a low an incentive between comparable components and a high incentive between disparate components, like difference. Nonetheless, the creators commented that it is hard to locate an all-inclusive difference work. **Xiong et al.** proposed the HCleaner strategy applied through a hyper clique-based data cleaner. Each pair of items in a hyper clique design has a significant degree of similitude identified with the strength of the connection between two examples. The HCleaner sift through cases avoided from any hyper clique design as commotion. Another group of specialists **Brighton et al [5]** applied a k-NN calculation, which basically contrasts test data and adjoining data to decide if they are anomalies by reference to their neighbors'. By utilizing their closest neighbors as references, diverse data are treated as erroneously arranged cases and eliminated. The creators considered examples of conduct among data to detail Wilson's altering approach, a bunch of decides that naturally select the data to be cleansed.

CHAPTER 3

AIM AND SCOPE

3.1 AIM OF PROJECT

The primary aim of this proposed system is to collect the dataset provided by the reflected sonar waves on which we are applying various machine learning algorithms to predict whether the obstacle is rock or mine. The dataset is having 60 features that would help us to predict the target.

3.1.1 OBJECTIVE

The proposed model is based on the new technologies which makes the whole process efficient. It is going to be very important and profitable for the naval force. This system will provide more accurate information about the prediction of the target whether it is rock or mine based on the features provided by the reflected waves etc. which will be very helpful for the navy officers in the process of travelling under water and during time of war and thus it will take less time to complete the process

3.2 SCOPE

The scope of this project is to predict the Underwater surface target through sonar mine dataset. In this project, we are using different type of machine learning Algorithms and high-end Visualization by using matplotlib and seaborn. The plots are bar-plots, boxplots. The machine learning models will help in a way that would predict the target with a high accuracy and with-in short time.

3.2.1 ADVANTAGES

Time saving as it is operated online.

Visualization using graphs explains each and every point.

Higher accuracy can be obtained

Here we are using different types of parameter

tuning approaches.

Hyper parameter tuning approach gains more precision.

CHAPTER 4

SYSTEM DESIGN & METHODOLOGY

4.1 EXISTING SYSTEM

The current framework is directed in a Java-based open-source stage called Weka which is a famous programming apparatus for AI tests from University of Waikato. All the previously mentioned calculations are accessible as one or the other norm or module capacities on Weka which have been all around reported in the Weka archive of documentation records (which is accessible for public download at <http://www.cs.waikato.ac.nz/ml/weka/>). Hence, their subtleties are not rehashed here. The equipment utilized is Lenovo Laptop with Intel Pentium Dual-Core T3200 2 GHz processor, 8 Gb RAM, and 64-bits Windows 7. The test dataset utilized is designated "connectionist seat (sonar, mines versus rocks) detail index," truncated as Sonar, which is prominently utilized for testing order calculations. The pioneer explores in utilizing this dataset is by Gorman and Sejnowski where sonar signals are grouped by utilizing various settings of a neural organization.

A similar undertaking is applied here aside from we utilize a data stream mining model called iDSM-CA in learning a summed-up model gradually to recognize sonar flags that are ricocheted off the outside of a metal chamber and those of a coarsely round and hollow stone. A representation of the appropriation of the data focuses that have a place with the two classes (mine or rock) in blue tone and red tone individually and of a vessel distinguishing the submerged articles (mines versus rocks) by sonar signals. The test dataset used is called "connectionist bench (sonar, mines versus rocks) data set," abbreviated as Sonar, which is popularly used for testing classification algorithms. The pioneer experiment in using this dataset is by Gorman and Sejnowski where sonar signals are classified by using different settings of a neural network. The same task is applied here except we use a data stream mining model called iDSM-CA in learning a generalized model incrementally to distinguish between sonar signals that are bounced off the surface of a metal cylinder and those of a coarsely cylindrical rock. A great deal of covers between these two gatherings of data can be found in each

property pair, proposing that the fundamental planning design is profoundly nonlinear. This suggests an extreme grouping issue where high exactness is difficult to accomplish.

4.1.1 DISADVANTAGES OF EXISTING SYSTEM

- The development of machine learning project on Weka is more complex
- Accuracy is very less
- More time consuming

4.2 PROPOSED FRAMEWORK

The principle worry of investigation in the field of AI is being to frame a planned computational machine for the ordering the gauge of the items, in view of the achievable data. The result of proposed system assists with anticipating the set off sound waves reflect from surface Rock or a Mine. Proposed structure techniques: Broadly in actual world or reasonable issues, there is no check over the kinds of data. Some critical pre-handling like evacuation of missing qualities, highlight determination, and so forth are constantly required. AI centers around taking up contemporary procedures to handle immense measure of complex data with lower cost. The theoretical perspective on proposed structure has been addressed in Figure 1. Figure 1 depicts the system of the expectation model made to decide the surface to be a stone or a mine dependent on around 61 factors or highlights, handled by 10 diverse classifier models, which give yields with an adequate exactness and accuracy rate. Pre-handling is utilized for Missing qualities are taken out by supplanting them by mean worth attribution. A total of six classification algorithms were put under test of sonar recognition. Support vector machine (SVM), DT, K-nearest neighbour's classifier, Naïve bayes. An incremental version of algorithm that is modified from traditional naive Bayesian called updateable naive Bayesian (NBup) is included as well for intellectual curiosity. Basically, all the six algorithms can be used in either batch learning or incremental learning mode. In the incremental learning mode, the data is treated as a data stream where the model is trained and updated section by section with conflict analysis enforced in effect. As the window slides along, the next new data instance ahead of the window is used to test the trained model. The performance statistics are thereby accumulated from the start till the end.

What's more, highlight Selection objectives to mean Gini record is utilized to rank the significant highlights. The best 50 highlights positioned by mean Gini file is chosen and taken care of to the forecast model. Furthermore, in the expectation model, Different ML classifiers are investigated and executed to locate the most ideal arrangement. Irregular timberland, being a group model has shown the best with 83.17% of exactness. The outcomes are additionally improved by applying highlight choice method to take care of the forecast model with the best highlights and exactness came to at 85.70% after streamlining. The result of this proposed system assists with anticipating the focused-on surface to be a Rock or a Mine.

4.2.1 ADVANTAGES OF PROPOSED SYSTEM

- Using different types of parameter tuning approaches.
- The main advantage of tuning approach is we can gain more accuracy
- Lesser time consuming

4.3 PROPOSED FRAMEWORK METHODS:

Broadly in physical world or realistic issues, there is no curb over the types of data. Some dire pre-processing like removal of missing values, feature selection, etc. are always required. Machine learning focuses on taking up contemporary techniques to process huge amount of complex data with lower expense. The abstract view of proposed framework has been represented in Figure 4.1. Figure 4.1 describes the framework of the prediction model created to determine the surface to be a rock or a mine based on about 61 factors or features, processed by 10 different classifier models, which give outputs with an acceptable accuracy and precision percentage.0020

A total of six classification algorithms were put under test of sonar recognition. Support vector machine (SVM), DT, K-nearest neighbors classifier, Naïve bayes. An incremental version of algorithm that is modified from traditional naive Bayesian called updateable naive Bayesian (NBup) is included as well for intellectual curiosity. Basically, all the six algorithms can be used in either batch learning or incremental learning mode. In the incremental learning mode, the data is treated as a data stream where the model is trained and updated section by section with conflict

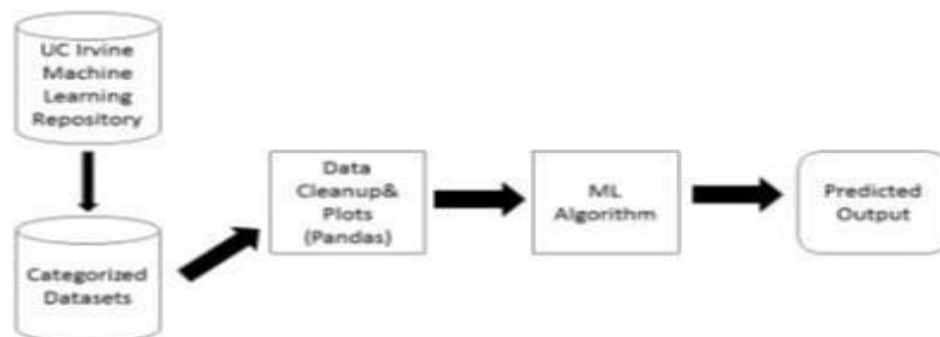
analysis enforced in effect. As the window slides along, the next new data instance ahead of

the window is used to test the trained model. The performance statistics are thereby accumulated from the start till the end.

Preprocessing: Missing values are removed by replacing them by mean value imputation.

Feature Selection: Mean Gini index is used to rank the important features. The top 50 features ranked by mean Gini index is selected and fed to the prediction model.

Prediction Model: Different ML classifiers are explored and implemented to find the best possible solution. Random forest, being an ensemble model has shown the highest performance with 83.17% of accuracy. The results are further optimized by applying feature selection technique to feed the prediction model with the best features and accuracy reached at 85.70% after optimization. The outcome of this proposed framework helps to predict the targeted surface to be a Rock or a Mine.



4.1 OVERVIEW OF THE PROPOSED SYSETM

4.4 REQUIREMENT SPECIFICATION

Hardware Requirements

The basic hardware required to run the program are:

- a. RAM: 4GB and Higher
- b. Processor: Intel i3 and above
- c. Hard Disk: 500GB: Minimum

Software Requirements

- a. OS: Windows or Linux
- b. Python IDE: python 2.7.x and above
- c. PyCharm IDE Required, jupyter notebook
- d. Language: Python Scripting

4.5 SYSTEM STUDY Overview of the system

It has to find Accuracy of the training dataset, Accuracy of the testing dataset, Specification, False Positive rate, precision and recall by comparing algorithm using python code. The following Involvement steps are,

Define a problem
Preparing data
Evaluating algorithms
Improving results
Predicting results

Project Goals

Exploration data analysis of variable identification

- Loading the given dataset
- Import required libraries packages
- Analyze the general properties
- Find duplicate and missing values
- Checking unique and count values

Uni-variate data analysis

- Rename, add data and drop the data
- To specify data type

Exploration data analysis of bi-variate and multi-variate

- Plot diagram of pair plot, heat map, bar chart and Histogram

Method of Outlier detection with feature engineering

- Pre-processing the given dataset
- Splitting the test and training dataset
- Comparing the Decision tree and Logistic regression model and random forest etc.

4.6 SOFTWARE FUNCTIONALITIES

Anaconda is a distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large- scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS. It is developed and maintained by Anaconda, Inc., Package versions in Anaconda are managed by the package management system conda.

This package manager was spun out as a separate open-source package as it ended up being useful on its own and for other things than Python. There is also a small, bootstrap version of Anaconda called **Miniconda**, which includes only conda, Python, the packages they depend on, and a small number of other packages. The next most popular distribution of Python is Anaconda. Anaconda has its own installer tool called **conda** that you could use for installing a third-party package. However, Anaconda comes with many scientific libraries preinstalled, including the Jupyter Notebook, so you don't actually need to do anything other than install Anaconda itself. Anaconda helps in simplified package management and deployment.

Anaconda comes with a wide variety of tools to easily collect data from various sources using various machine learning and AI algorithms. It helps in getting an easily manageable environment setup which can deploy any project with the click of a single button.

Anaconda Navigator

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, macOS and Linux.

The following applications are available by default in Navigator:

- JupyterLab
- Jupyter Notebook
- QtConsole
- Spyder
- Glueviz
- Orange
- Rstudio
- Visual Studio Code

Conda:

Conda is an open source, cross-platform, language-agnostic package manager and environment management system that installs, runs and updates packages and their dependencies. It was created for Python programs, but it can package and distribute software for any language (e.g., R), including multi-languages. The Conda package and environment manager is included in all versions of Anaconda, Miniconda, and Anaconda Repository.

The **Jupyter Notebook** is an open-source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. Jupyter Notebook is maintained by the people at [Project Jupyter](#).

Jupyter Notebooks are a spin-off project from the IPython project, which used to have an IPython Notebook project itself. Jupyter is a free, open-source, interactive web tool known as a computational notebook, which researchers can use to combine software code, computational output, explanatory text and multimedia resources in a single document. The name, Jupyter, comes from the core supported programming languages that it supports: Julia, Python, and R. Jupyter ships with the IPython kernel, which allows you to write your programs in Python, but there are currently over 100 other kernels that you can also use. Jupyter Notebook is built off of IPython, an interactive way of running Python code in the terminal using the REPL model (Read-Eval-Print-Loop).

With a Jupyter Notebook, you can view code, execute it, and display the results directly in your web browser. Live interactions with code. Jupyter Notebook code isn't static; it can be edited and re-run incrementally in real time, with feedback provided directly in the browser. The IPython Kernel runs the computations and communicates with the Jupyter Notebook front-end interface. It also allows Jupyter Notebook to support multiple languages.

Data visualization is the visual representation and presentation of data to enhance understanding. When stakeholders see data presented visually, they are better able to grasp difficult concepts, identify new patterns, or more efficiently take away key messages. Because of the way the human brain processes information, people find using visuals to represent large amounts of complex data is easier than poring over spreadsheets or reports. Data visualization is an essential aspect of effectively communicating data and findings. By using visual representations, state and local data users are able to see large amounts of data in clear, cohesive ways – and draw conclusions. Because it's significantly faster to analyse information in visual form (as opposed to in spreadsheets), state and local program staffs can address problems or answer questions in a timelier manner. Even extensive amounts of complicated data start to make sense when presented visually, and data users can recognize patterns and relationships in the data. Identifying those relationships helps data users dig deeper and

ask better questions about the data.

Once a data user has uncovered new insights from data analysis, she or he needs to communicate those insights to others. Using effective data visualization to tell the

data story is essential to engaging stakeholders and supporting understanding and action. Using data visualization makes it easier to spot outliers and identify a potential data quality issue before it becomes a bigger problem. Visualization or visualisation (see spelling differences) is any technique for creating images, diagrams, or animations to communicate a message. Visualization through visual imagery has been an effective way to communicate both abstract and concrete ideas since the dawn of humanity. We need data visualization because the human brain is not well equipped to devour so much raw, unorganized information and turn it into something usable and understandable. We need graphs and charts to communicate data findings so that we can identify patterns and trends to gain insight and make better decisions faster.

At analytics, we understand the importance of data visualization and what it means to our clients. We provide them with user-friendly and beautiful visualization features and tools to depict their data in a clear and meaningful way. We're here to ensure our clients have everything they need to make quick and informed decisions based on sound data that is easy to interpret. Contact our friendly team of professionals at analytics today to hear how we can better your business.

4.7 MODULES:

Variable Identification Process.

Exploration data analysis of

visualization

Comparing of various machine learning Algorithms

Module description:

Variable Identification Process

Validation techniques in machine learning are used to get the error rate of the Machine Learning (ML) model, which can be considered as close to the true error rate of the dataset. If the data volume is large enough to be representative of the population, you may not need the validation techniques. However, in real-world scenarios, to work with samples of data that may not be a true representative of the

population of given dataset. To finding the missing value, duplicate value and description of data type whether it is float variable or integer. The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyper parameters.

The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration. The validation set is used to evaluate a given model, but this is for frequent evaluation. It as machine learning engineers use this data to fine-tune the model hyper parameters. Data collection, data analysis, and the process of addressing data content, quality, and structure can add up to a time-consuming to-do list. During the process of data identification, it helps to understand your data and its properties; this knowledge will help you choose which algorithm to use to build your model.

A number of different **data cleaning** tasks using Python's Pandas library and specifically, it focusses on probably the biggest data cleaning task, **missing values** and it able to **more quickly clean data**. It wants to **spend less time cleaning data**, and more time exploring and modeling.

Some of these sources are just simple random mistakes. Other times, there can be a deeper reason why data is missing. It's important to understand these different types of missing data from a statistics point of view. The type of missing data will influence how to deal with filling in the missing values and to detect missing values, and do some basic imputation and detailed statistical approach for dealing with missing data.

Variable identification with Uni-variate, Bi-variate and Multi-variate analysis:

- import libraries for access and functional purpose and read the given dataset

- General Properties of Analyzing the given dataset

- Display the given dataset in the form of data frame

- show columns

- shape of the data frame

- To describe the data frame

- Checking data type and information about dataset

- Checking for duplicate data

- Checking Missing values of data frame

Checking unique values of data frame

Checking count values of data frame

Rename and drop the given data frame

To specify the type of values

To create extra columns

Exploration data analysis of visualization

Data visualization is an important skill in applied statistics and machine learning. Statistics does indeed focus on quantitative descriptions and estimations of data. Data visualization provides an important suite of tools for gaining a qualitative understanding. This can be helpful when exploring and getting to know a dataset and can help with identifying patterns, corrupt data, outliers, and much more. With a little domain knowledge, data visualizations can be used to express and demonstrate key relationships in plots and charts that are more visceral and stakeholders than measures of association or significance. Data visualization and exploratory data analysis are whole fields themselves and it will recommend a deeper dive into some the books mentioned at the end.

Sometimes data does not make sense until it can look at in a visual form, such as with charts and plots. Being able to quickly visualize of data samples and others is an important skill both in applied statistics and in applied machine learning. It will discover the many types of plots that you will need to know when visualizing data in Python and how to use them to better understand your own data.

How to chart time series data with line plots and categorical quantities with bar charts. How to summarize data distributions with histograms and box plots.

Comparison of machine learning algorithms

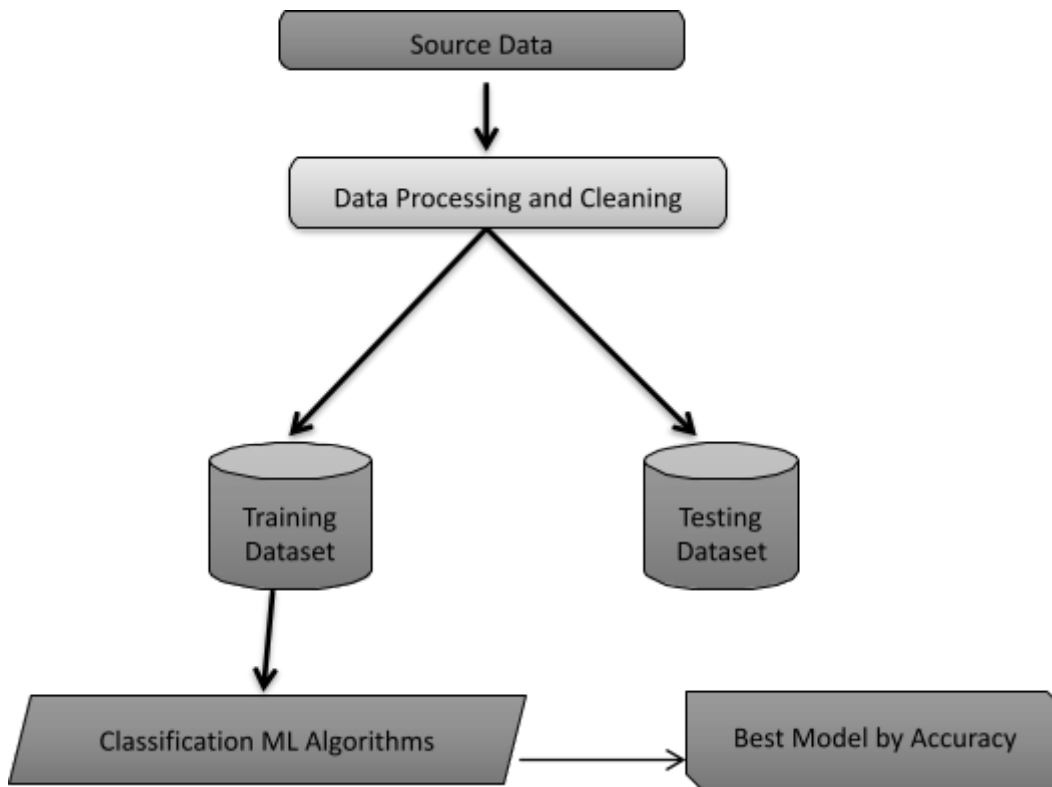
In this module the data is split in train and test data using `train_test_split()` function in `sklearn` package. Then the algorithm is applied on the training data to build the classification model. The training is done by `fit` function and the testing is done by

using predict function provided by the algorithm and the accuracy is compared

CHAPTER 5

SOFTWARE DEVELOPMENT METHODOLOGY

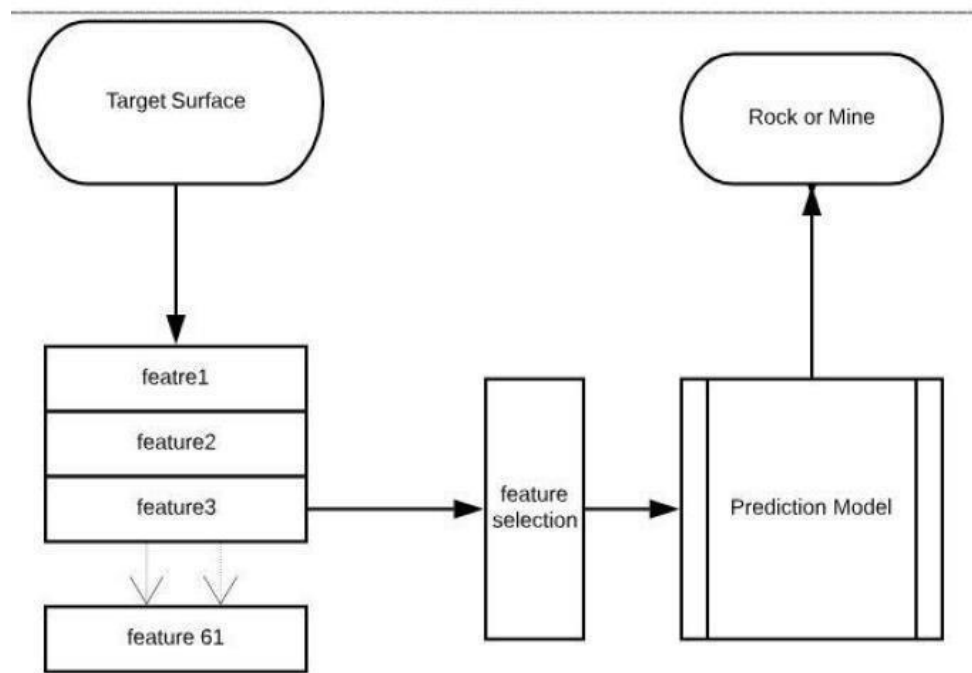
5.1 DESCRIPTION OF DIAGRAM



5.1 Process Flow Diagram

In the Fig 5.2, the knowledge provided to and received from the, “Underwater Surface target prediction using sonar” is identified. Initially the sonar wave signals made to pass through to the water. When they hit with an object, they are reflected which contains 60 features. We will collect those data and cleaning of data is made on it. Then select the features that are necessary and remove the non-important data. Various machine learning models are applied on the data to predict whether the object is rock or a mine. The machine learning models are used by applying hyper parameter tuning so that the accuracy of the prediction will be increased and will take less time for the detection of the object. At last classification report will provide us with the complete report of the data, the predicted values and the

accuracy



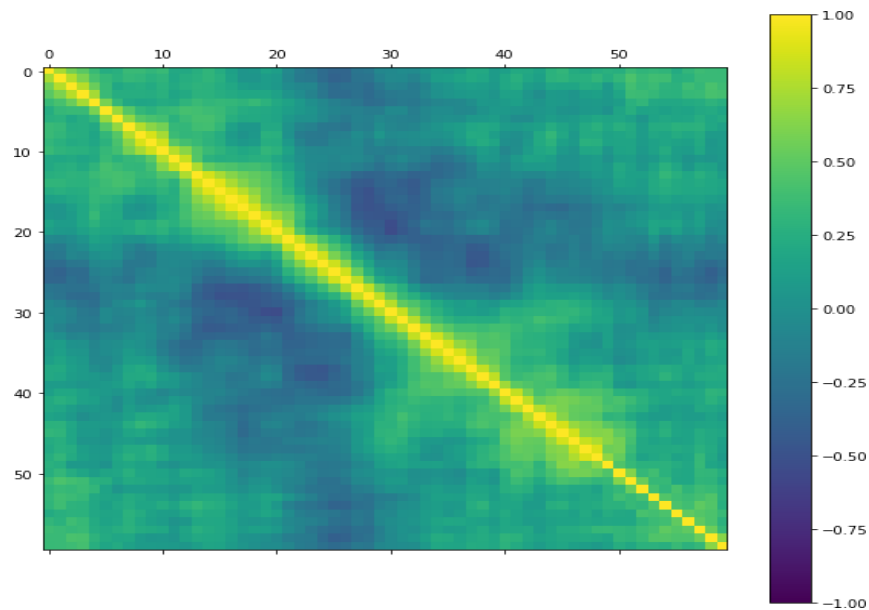
5.2 Architecture Diagram

Comparing Algorithm with prediction in the form of best accuracy result

It is important to compare the performance of multiple different machine learning algorithms consistently and it will discover to create a test harness to compare multiple different machine learning algorithms in Python with scikit-learn. It can use this test harness as a template on your own machine learning problems and add more and different algorithms to compare. Each model will have different performance characteristics. Using resampling methods like cross validation, you can get an estimate for how accurate each model may be on unseen data. It needs to be able to use these estimates to choose one or two best models from the suite of models that you have created. When have a new dataset, it is a good idea to visualize the data using different techniques in order to look at the data from different perspectives. The same idea applies to model selection. You should use a number of different ways of looking at the estimated accuracy of your machine learning algorithms in order to choose the one or two to finalize. A way to do this is to use different visualization methods to show the average accuracy, variance and

other properties of the distribution of model accuracies.

In the next section you will discover exactly how you can do that in Python with scikit-learn. The key to a fair comparison of machine learning algorithms is ensuring that each algorithm is evaluated in the same way on the same data and it can achieve this by forcing each algorithm to be evaluated on a consistent test harness.



5.3 Correlation matrix

A correlation matrix is simply a table which displays the correlation coefficients for different variables. The matrix depicts the correlation between all the possible pairs of values in a table. It is a powerful tool to summarize a large dataset and to identify and visualize patterns in the given data.

A correlation matrix consists of rows and columns that show the variables. Each cell in a table contains the correlation coefficient.

In addition, the correlation matrix is frequently utilized in conjunction with other types of statistical analysis. For instance, it may be helpful in the analysis of multiple linear regression models. Remember that the models contain several independent variables. In multiple linear regression, the correlation matrix determines the correlation coefficients between the independent variables in a model.

Correlation Matrix is basically a covariance matrix. Also known as the auto-covariance matrix, dispersion matrix, variance matrix, or variance-covariance matrix. It is a matrix in which i-j position defines the correlation between the i^{th} and j^{th} parameter of the given data-set.

When the data points follow a roughly straight-line trend, the variables are said to have an approximately linear relationship. In some cases, the data points fall close to a straight line, but more often there is quite a bit of variability of the points around the straight-line trend. A summary measure called the correlation describes the strength of the linear association. Correlation summarizes the strength and direction of the linear (straight-line) association between two quantitative variables. Denoted by r , it takes values between -1 and +1. A positive value for r indicates a positive association, and a negative value for r indicates a negative association. The closer r is to 1 the closer the data points fall to a straight line; thus, the linear association is stronger. The closer r is to 0, making the linear association weaker.

Prediction result by accuracy:

Logistic regression algorithm also uses a linear equation with independent predictors to predict a value. The predicted value can be anywhere between negative infinity to positive infinity. It needs the output of the algorithm to be classified variable data. Higher accuracy predicting result is logistic regression model by comparing the best accuracy.

$$\text{True Positive Rate (TPR)} = \text{TP} / (\text{TP} + \text{FN})$$

$$\text{False Positive rate (FPR)} = \text{FP} / (\text{FP} + \text{TN})$$

Accuracy: The Proportion of the total number of predictions that is correct otherwise overall how often the model predicts correctly defaulters and non-defaulters.

5.2 ACCURACY CALCULATION:

$$\text{Accuracy} = (\text{TP} + \text{TN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN})$$

Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations. One may think that, if we

have high accuracy then our model is best. Yes, accuracy is a great measure but only when you have symmetric datasets where values of false positive and false negatives are almost same.

Precision: The proportion of positive predictions that are actually correct. (When the model predicts default: how often is correct?)

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP})$$

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. The question that this metric answer is of all passengers that labelled as survived, how many actually survived? High precision relates to the low false positive rate. We have got 0.86 precision which is pretty good.

Recall: The proportion of positive observed values correctly predicted. (The proportion of actual defaulters that the model will correctly predict)

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN})$$

Recall (Sensitivity) - Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes.

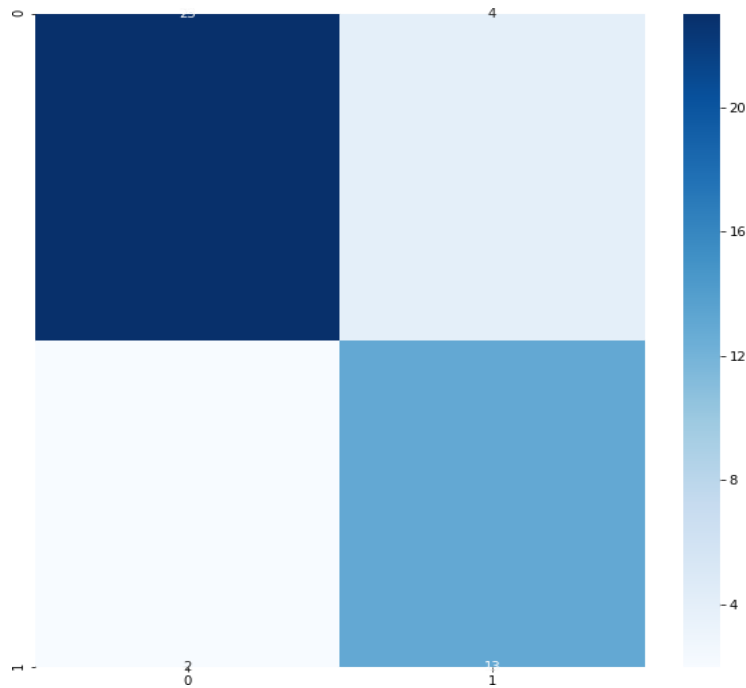
F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it's better to look at both Precision and Recall.

General Formula:

$$\text{F- Measure} = 2\text{TP} / (2\text{TP} + \text{FP} + \text{FN})$$

F1-Score Formula:

$$\text{F1 Score} = 2 * (\text{Recall} * \text{Precision}) / (\text{Recall} + \text{Precision})$$



5.4 Confusion matrix for the model predicted values

5.3 PARAMETERS IN THE CONFUSION MATRIX

False Positives (FP): A person who will pay predicted as defaulter. When actual class is no and predicted class is yes. E.g., if actual class says this passenger did not survive but predicted class tells you that this passenger will survive.

False Negatives (FN): A person who default predicted as payer. When actual class is yes but predicted class is no. E.g., if actual class value indicates that this passenger survived and predicted class tells you that passenger will die.

True Positives (TP): A person who will not pay predicted as defaulter. These are the correctly predicted positive values which means that the value of actual class is yes and the value of predicted class is also yes. E.g., if actual class value indicates that this passenger survived and predicted class tells you the same thing.

True Negatives (TN): A person who default predicted as payer. These are the correctly predicted negative values which means that the value of actual class is no and value of predicted class is also no. E.g., if actual class says this passenger did

not survive and predicted class tells you the same thing.

5.4 PYTHON

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

5.5 NUMPY ARRAY

Array in NumPy is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In NumPy, number of dimensions of the array is called rank of the array. A tuple of integers giving the size of the array along each dimension is known as shape of the array. An array class in NumPy is called as Nd array. Elements in NumPy arrays are accessed by using square brackets and

can be initialized by using nested Python Lists.

4.5. PIP (PACKAGE MANAGER) pip is a de facto standard package-management system used to install and manage software packages written in Python. Many packages can be found in the default source for packages and their dependencies.

pip is a tool for installing packages from the Python Package Index. Virtual env is a tool for creating isolated Python environments containing their own copy of python, pip, and their own place to keep libraries installed from PyPI.

5.6 PANDAS

Pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labelled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real- world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis/manipulation tool available in any language. It is already well on its way toward this goal. Pandas is well suited for many different kinds of data:

- Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet
- Ordered and unordered (not necessarily fixed-frequency) time series data.
- Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels
- Any other form of observational / statistical data sets. The data need not be labelled at all to be placed into a pandas data structure

The two primary data structures of pandas, Series (1-dimensional) and Data frame (2-dimensional), handle the vast majority of typical use cases in finance, statistics, social science, and many areas of engineering. For R users, data frame provides everything that R's data. Frame provides and much more. pandas is built on top of NumPy and is intended to integrate well within a scientific computing environment with many other 3rd party libraries.

5.7 MATPLOTLIB

Matplotlib is one of the most popular Python packages used for data visualization. It is a cross-platform library for making 2D plots from data in arrays. Matplotlib is written in Python and makes use of NumPy, the numerical mathematics extension of Python. It provides an object-oriented API that helps in embedding plots in applications using Python GUI toolkits such as PyQt, WxPython or Tkinter. It can be used in Python and IPython shells, Jupyter notebook and web application servers also.

Matplotlib has a procedural interface named the Pylab, which is designed to resemble MATLAB, a proprietary programming language developed by MathWorks. Matplotlib along with NumPy can be considered as the open source equivalent of MATLAB.

Matplotlib was originally written by John D. Hunter in 2003. The current stable version is 2.2.0 released in January 2018.

jupyter is a loose acronym meaning Julia, Python, and R. These programming languages were the first target languages of the Jupyter application, but nowadays, the notebook technology also supports many other languages.

In 2001, Fernando Pérez started developing IPython. IPython is a command shell for interactive computing in multiple programming languages, originally developed for the Python.

Consider the following features provided by IPython –

- Interactive shells (terminal and Qt-based).
- A browser-based notebook with support for code, text, mathematical expressions, inline plots and other media.
- Support for interactive data visualization and use of GUI toolkits.
- Flexible, embeddable interpreters to load into one's own projects.

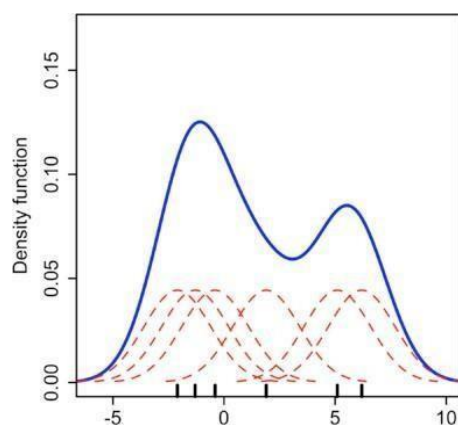
In 2014, Fernando Pérez announced a spin-off project from IPython called Project Jupyter. IPython will continue to exist as a Python shell and a kernel for Jupyter,

while the notebook and other language-agnostic parts of IPython will move under the Jupyter name. Jupyter added support for Julia, R, Haskell and Ruby.

To start the Jupyter notebook, open Anaconda navigator (a desktop graphical user interface included in Anaconda that allows you to launch applications and easily manage Conda packages, environments and channels without the need to use command line commands).

5.7.1 DENSITY PLOTS

First, what is a density plot? A density plot is a smoothed, continuous version of a histogram estimated from the data. The most common form of estimation is known as kernel density estimation. In this method, a continuous curve (the kernel) is drawn at every individual data point and all of these curves are then added together to make a single smooth density estimation. The kernel most often used is a Gaussian (which produces a Gaussian bell curve at each data point). If, like me, you find that description a little confusing, take a look at the following plot:



Kernel Density Estimation (Source)

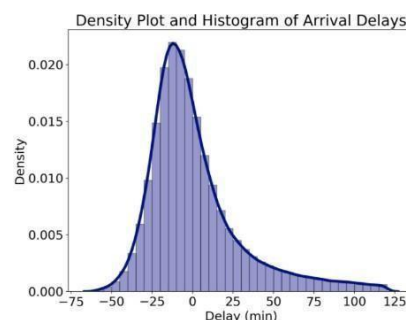
Here, each small black vertical line on the x-axis represents a data point. The individual kernels (Gaussians in this example) are shown drawn in dashed red lines above each point. The solid blue curve is created by summing the individual Gaussians and forms the overall density plot.

The x-axis is the value of the variable just like in a histogram, but what exactly does the y-axis represent? The y-axis in a density plot is the probability density function for the kernel density estimation. However, we need to be careful to specify this is a probability *density* and not a probability. The difference is the probability density is the probability per unit on the x-axis. To convert to an actual probability, we need to find the area under the curve for a specific interval on the x-axis. Somewhat confusingly, because this is a probability density and not a probability, the y-axis can take values greater than one. The only requirement of the density plot is that the total area under the curve integrates to one. I generally tend to think of the y-axis on a density plot as a value only for relative comparisons between different categories.

Density Plots in Seaborn

To make density plots in seaborn, we can use either the `distplot` or `kdeplot` function. I will continue to use the `distplot` function because it lets us make multiple distributions with one function call. For example, we can make a density plot showing all arrival delays on top of the corresponding histogram:

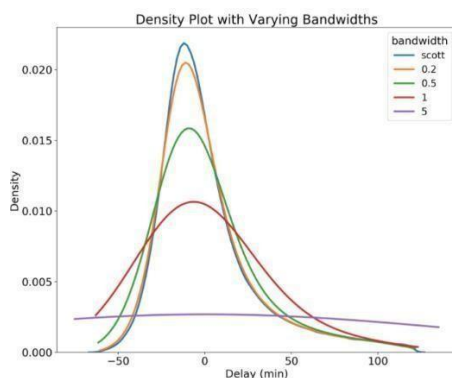
```
# Density Plot and Histogram of all arrival delays
sns.distplot(flights['arr_delay'], hist=True, color='black', kde=True,
             hist_kws={'edgecolor':'black'}, kde_kws={'linewidth':4},
             kde_kws={'color':'darkblue'})
```



Density Plot and Histogram using seaborn

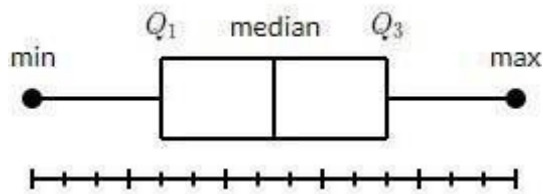
The curve shows the density plot which is essentially a smooth version of the histogram. The y-axis is in terms of density, and the histogram is normalized by default so that it has the same y-scale as the density plot.

Analogous to the bandwidth of a histogram, a density plot has a parameter called the bandwidth that changes the individual kernels and significantly affects the final result of the plot. The plotting library will choose a reasonable value of the bandwidth for us (by default using the 'Scott' estimate), and unlike the bandwidth of a histogram, I usually use the default bandwidth. However, we can look at using different bandwidths to see if there is a better choice. In the plot, 'Scott' is the default, which looks like the best option.



5.7.2 MATPLOTLIB - BOX PLOT

A box plot which is also known as a whisker plot displays a summary of a set of data containing the minimum, first quartile, median, third quartile, and maximum. In a box plot, we draw a box from the first quartile to the third quartile. A vertical line goes through the box at the median. The whiskers go from each quartile to the minimum or maximum.



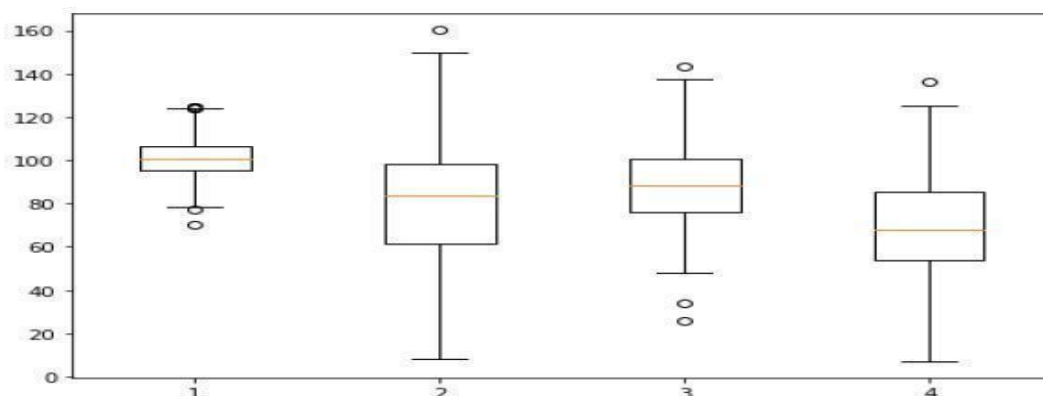
Let us create the data for the boxplots. We use the `numpy.random.normal()` function to create the fake data. It takes three arguments, mean and standard deviation of the normal distribution, and the number of values desired.

```
np.random.seed(10)
collectn_1 = np.random.normal(100, 10, 200)
collectn_2 = np.random.normal(80, 30, 200)
collectn_3 = np.random.normal(90, 20, 200)
collectn_4 = np.random.normal(70, 25, 200)
```

The list of arrays that we created above is the only required input for creating the boxplot. Using the `data_to_plot` line of code, we can create the boxplot with the following code –

```
fig = plt.figure()
# Create an axes instance
ax = fig.add_axes([0,0,1,1])
# Create the boxplot
bp = ax.boxplot(data_to_plot)
plt.show()
```

The above line of code will generate the following output –



5.7.3 MATPLOTLIB HISTOGRAM

A histogram is an accurate representation of the distribution of numerical data. It is an estimate of the probability distribution of a continuous variable. It is a kind of bar graph.

To construct a histogram, follow these steps –

- Bin the range of values.
- Divide the entire range of values into a series of intervals.
- Count how many values fall into each interval.

The bins are usually specified as consecutive, non-overlapping intervals of a variable.

The `matplotlib.pyplot.hist()` function plots a histogram. It computes and draws the histogram of `x`.

Parameters

The following table lists down the parameters for a histogram –

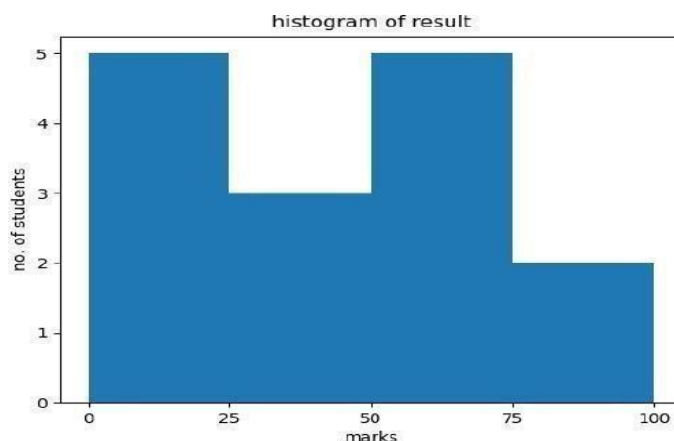
X	array or sequence of arrays
Bins	integer or sequence or 'auto', optional
optional parameters	
Range	The lower and upper range of the bins.
Density	If True, the first element of the return tuple will be the counts normalized to form a probability density
cumulative	If True, then a histogram is computed where each bin gives the counts in that bin plus all bins for smaller values.

Histtype	The type of histogram to draw. Default is 'bar' • 'bar' is a traditional bar-type histogram. If multiple data are given the bars are arranged side by side. • 'barstacked' is a bar-type histogram where multiple data are stacked on top of each other. • 'step' generates a lineplot that is by default unfilled. • 'stepfilled' generates a lineplot that is by default filled.
----------	--

Following example plots a histogram of marks obtained by students in a class. Four bins, 0-25, 26-50, 51-75, and 76-100 are defined. The Histogram shows number of students falling in this range.

```
from matplotlib import pyplot as
plt import numpy as np
fig,ax = plt.subplots(1,1)
a = np.array([22,87,5,43,56,73,55,54,11,20,51,5,79,31,27])
ax.hist(a, bins = [0,25,50,75,100])
ax.set_title("histogram of result")
ax.set_xticks([0,25,50,75,100])
ax.set_xlabel('marks')
ax.set_ylabel('no. of students')
plt.show()
```

The plot appears as shown below –



5.8 SCIKIT-LEARN

Scikit-learn (formerly scikit.learn and also known as sklearn) is a free software machine learning library for the Python programming language.^[3] It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k -means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

The scikit-learn project started as scikits.learn, a Google Summer of Code project by David Cournapeau. Its name stems from the notion that it is a "SciKit" (SciPy Toolkit), a separately-developed and distributed third-party extension to SciPy.^[4] The original codebase was later rewritten by other developers. In 2010 Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort and Vincent Michel, all from the French Institute for Research in Computer Science and Automation in Rocquencourt, France, took leadership of the project and made the first public release on February the 1st 2010.^[5] Of the various scikits, scikit-learn as well as scikit-image were described as "well-maintained and popular" in November 2012.^[6] Scikit-learn is one of the most popular machine learning libraries on GitHub.

5.8.1 LOGISTIC REGRESSION

Logistic Regression, also known as *Logit Regression* or *Logit Model*, is a mathematical model used in statistics to estimate (guess) the probability of an event occurring having been given some previous data. Logistic Regression works with binary data, where either the event happens (1) or the event does not happen (0). So, given some feature x it tries to find out whether some event y happens or not. So, y can either be 0 or 1. In the case where the event happens, y is given the value

1. If the event does not happen, then y is given the value of 0. For example, if y represents whether a sports team wins a match, then y will be 1 if they win the match or y will be 0 if they do not. This is known as *Binomial Logistic Regression*. There is also another form of Logistic Regression which uses multiple values for the

variable y . This form of Logistic Regression is known as *Multinomial Logistic Regression*.

Logistic Regression uses the logistic function to find a model that fits with the data points. The function gives an 'S' shaped curve to model the data. The curve is restricted between 0 and 1, so it is easy to apply when y is binary. Logistic Regression can then model events better than linear regression, as it shows the probability for y being 1 for a given x value. Logistic Regression is used in statistics and machine learning to predict values of an input from previous test data.

Logistic regression is an alternative method to use other than the simpler Linear Regression. Linear regression tries to predict the data by finding a linear – straight line – equation to model or predict future data points. Logistic regression does not look at the relationship between the two variables as a straight line. Instead, Logistic regression uses the natural logarithm function to find the relationship between the variables and uses test data to find the coefficients. The function can then predict the future results using these coefficients in the logistic equation.

Logistic regression uses the concept of odds ratios to calculate the probability. This is defined as the ratio of the odds of an event happening to its not happening. For example, the probability of a sports team to win a certain match might be 0.75. The probability for that team to lose would be $1 - 0.75 = 0.25$. The odds for that team winning would be $0.75/0.25 = 3$. This can be said as the odds of the team winning are 3 to 1.^[1]

The odds can be defined as:

$$Odds = \frac{P(y = 1|x)}{1 - P(y = 1|x)}$$

The natural logarithm of the odds ratio is then taken in order to create the logistic equation. The new equation is known as the logit:

$$Logit(P(x)) = \ln\left(\frac{P(y = 1|x)}{1 - P(y = 1|x)}\right)$$

In Logistic regression the Logit of the probability is said to be linear with respect to x , so the logit becomes:

$$\text{Logit}(P(x)) = a + bx$$

Using the two equations together then gives the following:

$$\frac{P(y = 1|x)}{1 - P(y = 1|x)} = e^{a+bx}$$

This then leads to the probability:

$$P(y = 1|x) = \frac{e^{a+bx}}{1 + e^{a+bx}} = \frac{1}{1 + e^{-(a+bx)}}$$

his final equation is the logistic curve for Logistic regression. It models the non-linear relationship between x and y with an 'S'-like curve for the probabilities that $y = 1$ - that event the y occurs. In this example a and b represent the gradients for the logistic function just like in linear regression. The logit equation can then be expanded to handle multiple gradients. This gives more freedom with how the logistic curve matches the data. The multiplication of two vectors can then be used to model more gradient values and give the following equation:

$$\text{Logit}(P(x)) = w_0x^0 + w_1x^1 + w_2x^2 + \dots + w_nx^n = w^T x$$

In this equation $w = [w_0, w_1, w_2, \dots, w_n]$ and represents the n gradients for the equation. The powers of x are given by the vector $x = [1, x, x^2, \dots, x^n]$. These two vectors give the new logit equation with multiple gradients. The logistic equation then can then be changed to show this:

$$P(y = 1|x) = \frac{1}{1 + e^{-(w^T x)}}$$

This is then a more general logistic equation allowing for more gradient values.

5.8.2 K-NEAREST NEIGHBORS ALGORITHM

In statistics, the k -nearest neighbors algorithm (k -NN) is a non-parametric classification method first developed by Evelyn Fix and Joseph Hodges

in

1951,^[1] and later expanded by Thomas Cover.^[2] It is used for classification and regression. In both cases, the input consists of the k closest training examples in data set. The output depends on whether k -NN is used for classification or regression:

- In *k-NN classification*, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If $k = 1$, then the object is simply assigned to the class of that single nearest neighbor.
- In *k-NN regression*, the output is the property value for the object. This value is the average of the values of k nearest neighbors.

k -NN is a type of classification where the function is only approximated locally and all computation is deferred until function evaluation. Since this algorithm relies on distance for classification, if the features represent different physical units or come in vastly different scales then normalizing the training data can improve its accuracy dramatically.

Both for classification and regression, a useful technique can be to assign weights to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of $1/d$, where d is the distance to the neighbor.

The neighbors are taken from a set of objects for which the class (for k -NN classification) or the object property value (for k -NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required.

A peculiarity of the k -NN algorithm is that it is sensitive to the local structure of the data.

5.8.3 GAUSSIAN NAIVE BAYES

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. There is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness, and diameter features.

For some types of probability models, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without accepting Bayesian probability or using any Bayesian methods.

Despite their naive design and apparently oversimplified assumptions, naive Bayes classifiers have worked quite well in many complex real-world situations. In 2004, an analysis of the Bayesian classification problem showed that there are sound theoretical reasons for the apparently implausible efficacy of naive Bayes classifiers. Still, a comprehensive comparison with other classification algorithms in 2006 showed that Bayes classification is outperformed by other approaches, such as boosted trees or random forests.

An advantage of naive Bayes is that it only requires a small number of training data to estimate the parameters necessary for classification.

When dealing with continuous data, a typical assumption is that the continuous values associated with each class are distributed according to a normal (or Gaussian) distribution. For example, suppose the training data contains a continuous attribute,

x. We first segment the data by the class, and then compute the mean and variance

of x in each class.

The equation for normal distribution is:

$$p(x = v \mid C_k) = \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{(v-\mu_k)^2}{2\sigma_k^2}}$$

Another common technique for handling continuous values is to use binning to discretize the feature values, to obtain a new set of Bernoulli-distributed features; some literature in fact suggests that this is necessary to apply naive Bayes, but it is not, and the discretization may throw away discriminative information.

Sometimes the distribution of class-conditional marginal densities is far from normal. In these cases, kernel density estimation can be used for a more realistic estimate of the marginal densities of each class. This method, which was introduced by John and Langley,^[9] can boost the accuracy of the classifier considerably.

5.8.4 SUPPORT VECTOR MACHINE

In machine learning, support-vector machines (SVMs, also support-vector networks) are supervised learning models with associated learning algorithms that analyse data for classification and regression analysis. Developed at AT&T Bell Laboratories by Vladimir Vapnik with colleagues (Boser et al., 1992, Guyon et al., 1993, Vapnik et al., 1997), SVMs are one of the most robust prediction methods, being based on statistical learning frameworks or VC theory proposed by Vapnik and Chervonenkis (1974) and Vapnik (1982, 1995). Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non- probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting). An SVM maps training examples to points in space so as to maximise the width of the gap between the two categories. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

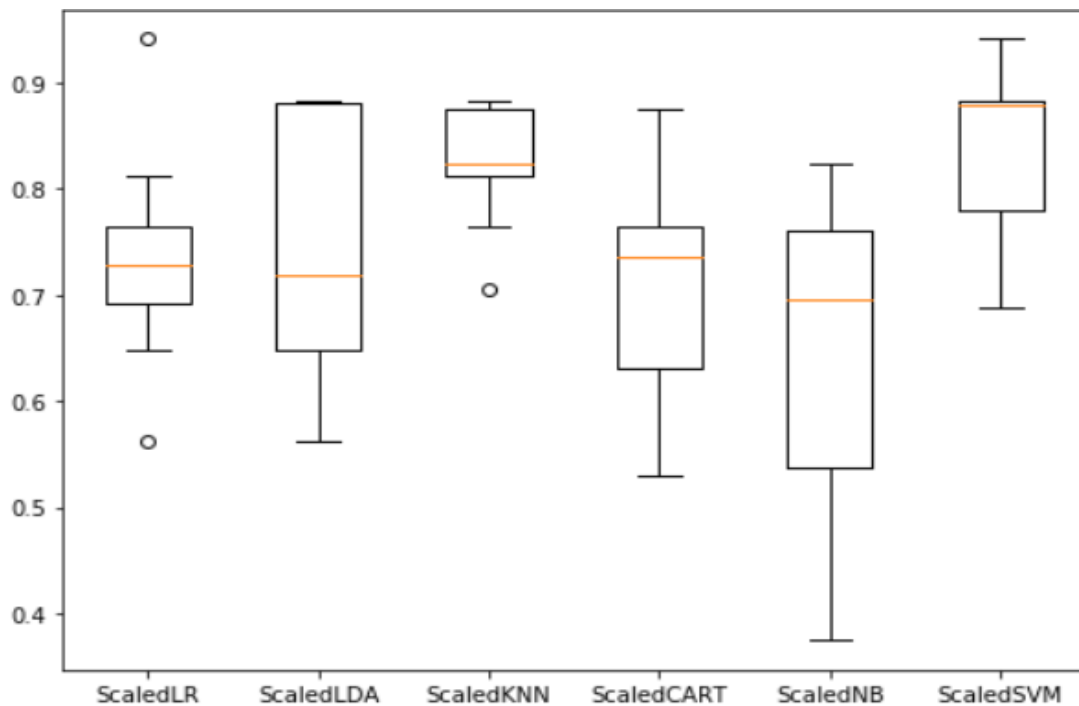
When data are unlabeled, supervised learning is not possible, and an unsupervised learning approach is required, which attempts to find natural clustering of the data to

groups, and then map new data to these formed groups. The support-vector clustering algorithm, created by Hava Siegelmann and Vladimir Vapnik, applies the statistics of support vectors, developed in the support vector machines algorithm, to categorize unlabeled data, and is one of the most widely used clustering algorithms in industrial applications.

$$\sum_i \alpha_i k(x_i, x) = \text{constant}.$$

CHAPTER 6

RESULTS AND DISCUSSION



6.1 Box-plot for Accuracy output of scaled algorithms

We have compared 6 machine learning algorithms after they are scaled using standard scaler method. The accuracy provided by the naive bayes algorithm is very less among the compared algorithms and Support vector machine algorithm achieved the highest accuracy.

```
# SVM algorithm tuning
scaler = StandardScaler().fit(X_train)
rescaledX = scaler.transform(X_train)
c_values = [0.1, 0.3, 0.5, 0.7, 0.9, 1.0, 1.3, 1.5, 1.7, 2.0]
kernel_values = ['linear', 'poly', 'rbf', 'sigmoid']
param_grid = dict(C=c_values, kernel=kernel_values)
model = SVC()
kfold = KFold(n_splits=num_folds, random_state=seed)
grid = GridSearchCV(estimator=model, param_grid=param_grid, scoring=scoring, cv=kfold)
grid_result = grid.fit(rescaledX, Y_train)
```

Best: 0.867470 using {'C': 1.5, 'kernel': 'rbf'}

6.2 Accuracy of SVM algorithm

```

: # ensembles
ensembles = []
# Boosting methods
ensembles.append(('AB', AdaBoostClassifier()))
ensembles.append(('GBM', GradientBoostingClassifier()))
# Bagging methods
ensembles.append(('RF', RandomForestClassifier()))
ensembles.append(('ET', ExtraTreesClassifier()))

```

```

[0.70588235 0.82352941 0.88235294 0.82352941 0.82352941 0.70588235
 0.8125      0.8125      0.8125      0.9375      ]
AB: 0.813971 (0.066017)
[0.76470588 1.          1.          0.82352941 0.94117647 0.88235294
 0.75        0.8125      0.6875      0.875        ]
GBM: 0.853676 (0.100318)
[0.64705882 0.82352941 0.82352941 0.64705882 0.70588235 0.64705882
 0.75        0.8125      0.75        0.8125        ]
RF: 0.741912 (0.071909)
[0.82352941 0.94117647 0.82352941 0.76470588 0.94117647 0.76470588
 0.875       1.          0.6875      0.875        ]
ET: 0.849632 (0.090958)

```

6.3 Accuracy values of boosting algorithms with cross_val_score method

Support Vector Machine algorithm is providing the highest accuracy compared to other scaled algorithms. The accuracy achieved is 86.7% which is very huge compared to the existing system.

```

# estimate accuracy on validation set
rescaledValidationX = scaler.transform(X_validation)
predictions = model.predict(rescaledValidationX)
print(accuracy_score(Y_validation, predictions))
cm=confusion_matrix(Y_validation, predictions)
print(cm)
print(classification_report(Y_validation, predictions))

```

```

0.8571428571428571
[[23  4]
 [ 2 13]]

```

	precision	recall	f1-score	support
M	0.92	0.85	0.88	27
R	0.76	0.87	0.81	15
accuracy			0.86	42
macro avg	0.84	0.86	0.85	42
weighted avg	0.86	0.86	0.86	42

The accuracy on the validation set was 85.7%. Very close to our original estimates.

6.4 Classification Report

Accuracy score provides us with the over-all accuracy of the prediction values and the original values. Confusion matrix is a kind of matrix will give us the detailed true-negative, true-positive, false-negative, false-positive values of the machine learning model predicted values compared to dataset values. Classification report come up with the precision, recall, f1-score, support values. It also produce accuracy, macro average and the weighted average of the above values.

```
model.predict([[0.02,0.0371,0.0428,0.0207,0.0954,0.0986,0.1539,0.1601,0.3109,0.2111,0.1609,0.1582,0.2238,0.0645,
0.066,0.2273,0.31,0.2999,0.5078,0.4797,0.5783,0.5071,0.4328,0.555,0.6711,0.6415,0.7104,0.808,0.6791,
0.3857,0.1307,0.2604,0.5121,0.7547,0.8537,0.8507,0.6692,0.6097,0.4943,0.2744,0.051,0.2834,0.2825,0.4256,
0.2641,0.1386,0.1051,0.1343,0.0383,0.0324,0.0232,0.0027,0.0065,0.0159,0.0072,0.0167,0.018,0.0084,0.009,0.0032
]])

array(['M'], dtype=object)
```

OUTPUT

We have passed the 60 features into the predict method which is one of the function in machine learning and the model predicted the output, the way it is trained and it is the output.

CHAPTER 7

CONCLUSION AND FUTURE WORK

CONCLUSION

An adequate prediction miniature, united with the machine learning classifying features, is proposed which can conclude if the target of the sound wave is either a rock or a mine or any other organism or any kind of other body. Research is carried out for predicting the best possible result for the target to be a rock or a mine, which is found to be best through the random forest model, which is an ensemble tree- based classifier in machine learning with the highest accuracy rate of 83.17% and giving the best ROC-AUC rate 0.93, with least error for better elaboration of this prediction model.

FUTURE WORK

In the future, the designed system with the used machine learning classification algorithm can be used to predict Rock or mine. Further user interface might be added for this proposed work for easy usability of the code and that can be easily understandable. The work can be extended or improved for the automation of the real time model like deep learning with open cv.

REFERENCES

- [1] Hamed Komari Alaie, Hassan Farsi, "Passive Sonar Target Detection Using Statistical Classifier and Adaptive Threshold" Department of Electrical and Computer Engineering, University of Birjand, Appl. Sci. 2018, 8, 61; doi:10.3390/app8010061
- [2] Dhiraj Neupane and Jongwon Seok, "A Review on Deep Learning-Based Approaches for Automatic Sonar Target Recognition" Department of Data and Communication Engineering, Changwon National University, Changwon-si, Gyeongsangnam-do 51140, Electronics 2020, 9, 1972; doi:10.3390/electronics9111972 www.mdpi.com/journal/electronics
- [3] Zhiyuan Zhang, Xia Feng, "New Methods for Deviation-based Outlier Detection in Large Database", School of Computer Science & Technology, Civil Aviation University of China, Tianjin, 300300 zy-zhang@cauc.edu.cn, xfeng@cauc.edu.cn.
- [4] H. Xiong, G. Pandey, M. Steinbach, and V. Kumar, "Enhancing data analysis with noise removal," IEEE Transactions on Knowledge and Data Engineering, vol. 18, no. 3, pp. 304–319, 2006.
- [5] H. Brighton and C. Mellish, "Advances in instance selection for instance-based learning algorithms," Data Mining and Knowledge Discovery, vol. 6, no. 2, pp. 153–172, 2002.
- [6] N. Hooda et al. "B 2 FSE framework for high dimensional imbalanced data: A case study for drug toxicity prediction", Neurocomputing, (2018)
- [7] Corinna Cortes; Vladimir N., Vapnik. "Support-vector networks". Machine Learning. 20 (3): 273–297; doi: 10.1007/BF00994018.
- [8] Kégl, Balázs. "The return of AdaBoost.MH: multiclass Hamming trees". arXiv:1312.6086. (20 December 2013).
- [9] Pearl, Judea. Causality: Models, Reasoning, and Inference. Cambridge University Press. ISBN 0- 521-77362-8. OCLC 4229125.

(2010).

[10]. Bradley, Andrew P. "The use of the area under the ROC curve in the evaluation of machine learning algorithms." *Pattern recognition* 30.7: 1145- 1159.

APPENDIX

A. SCREENSHOTS

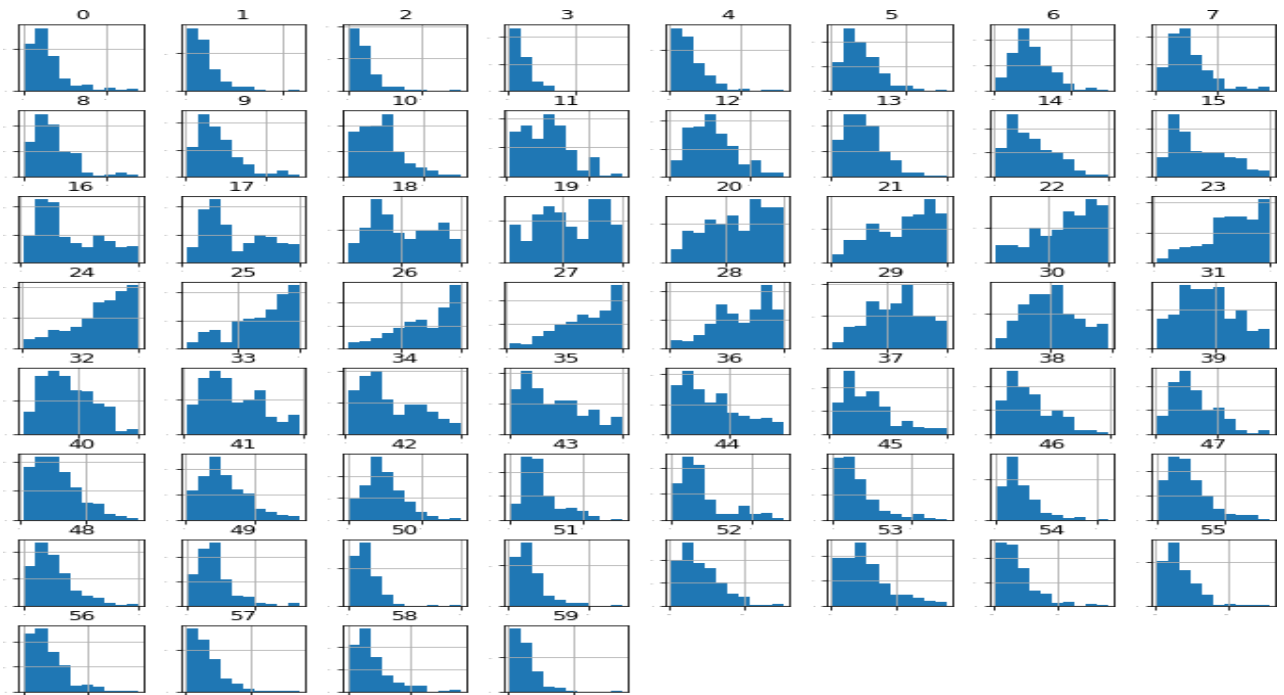
```
In [3]: dataset.head()
```

```
Out[3]:
```

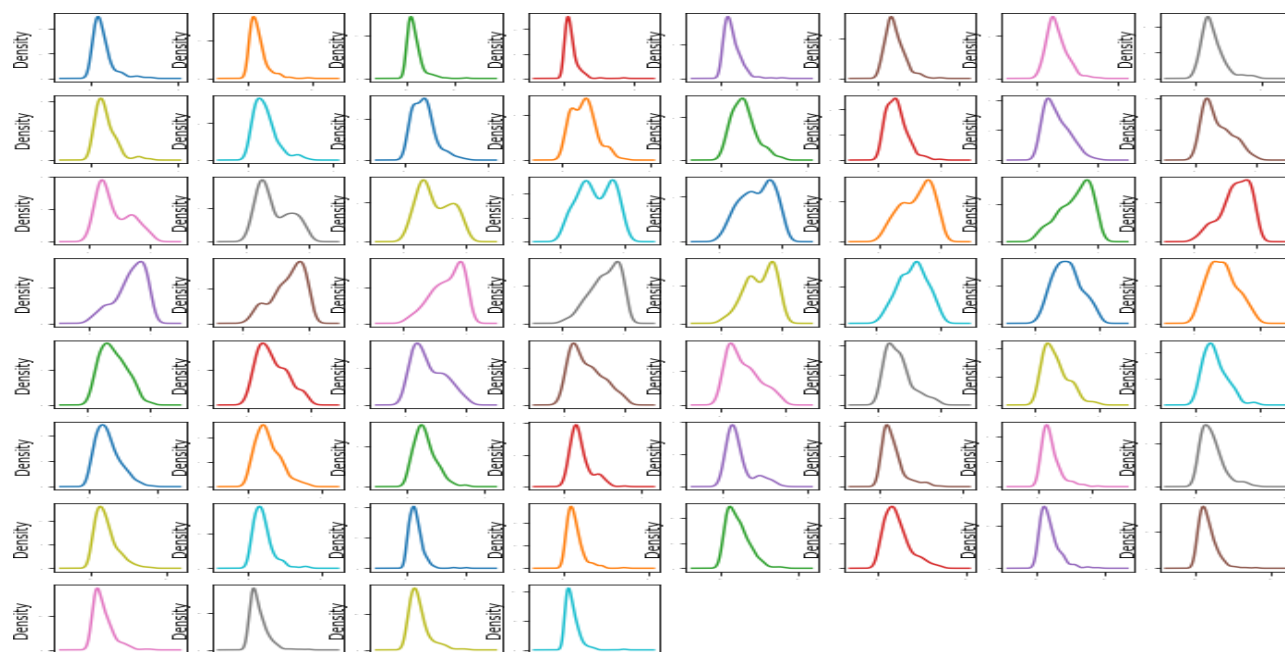
	0	1	2	3	4	5	6	7	8	9	...	51	52	53	54	55	56	57	58	59	60
0	0.0200	0.0371	0.0428	0.0207	0.0954	0.0986	0.1539	0.1601	0.3109	0.2111	...	0.0027	0.0065	0.0159	0.0072	0.0167	0.0180	0.0084	0.0090	0.0032	R
1	0.0453	0.0523	0.0843	0.0689	0.1183	0.2583	0.2156	0.3481	0.3337	0.2872	...	0.0084	0.0089	0.0048	0.0094	0.0191	0.0140	0.0049	0.0052	0.0044	R
2	0.0262	0.0582	0.1099	0.1083	0.0974	0.2280	0.2431	0.3771	0.5598	0.6194	...	0.0232	0.0166	0.0095	0.0180	0.0244	0.0316	0.0164	0.0095	0.0078	R
3	0.0100	0.0171	0.0623	0.0205	0.0205	0.0368	0.1098	0.1276	0.0598	0.1264	...	0.0121	0.0036	0.0150	0.0085	0.0073	0.0050	0.0044	0.0040	0.0117	R
4	0.0762	0.0666	0.0481	0.0394	0.0590	0.0649	0.1209	0.2467	0.3564	0.4459	...	0.0031	0.0054	0.0105	0.0110	0.0015	0.0072	0.0048	0.0107	0.0094	R

5 rows × 61 columns

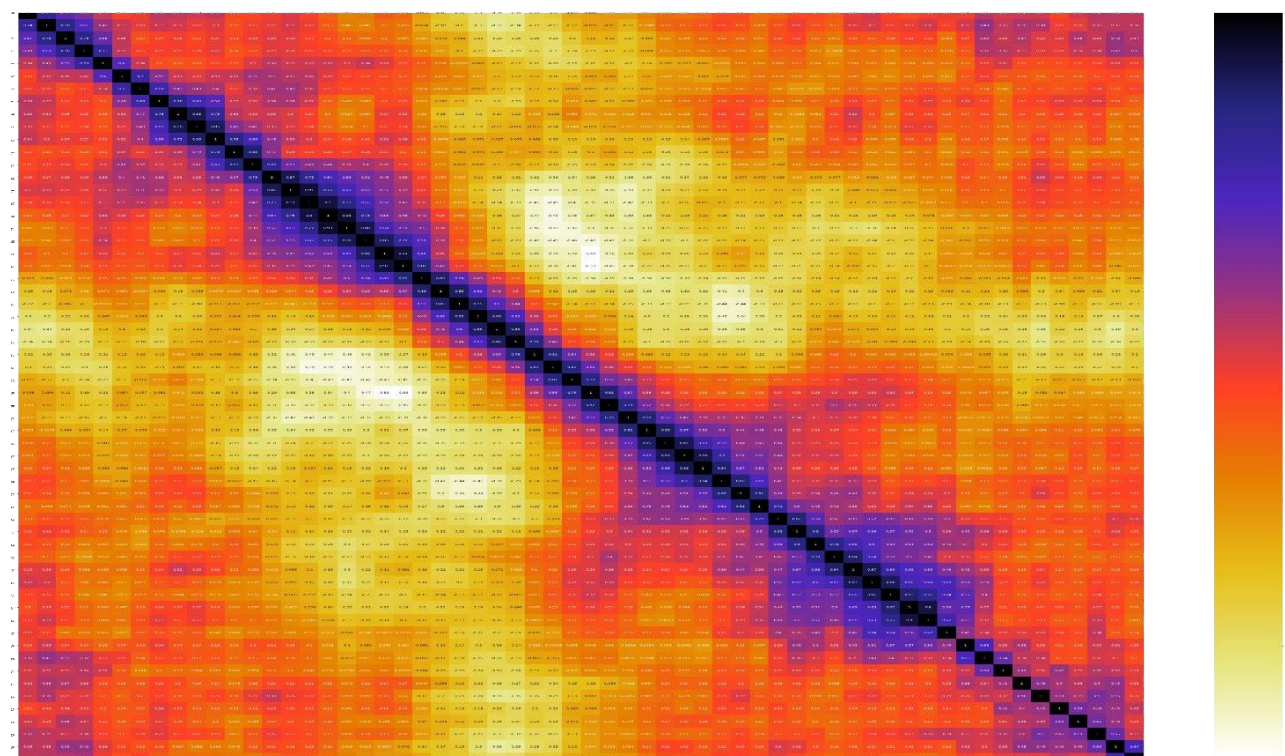
1. Sonar dataset



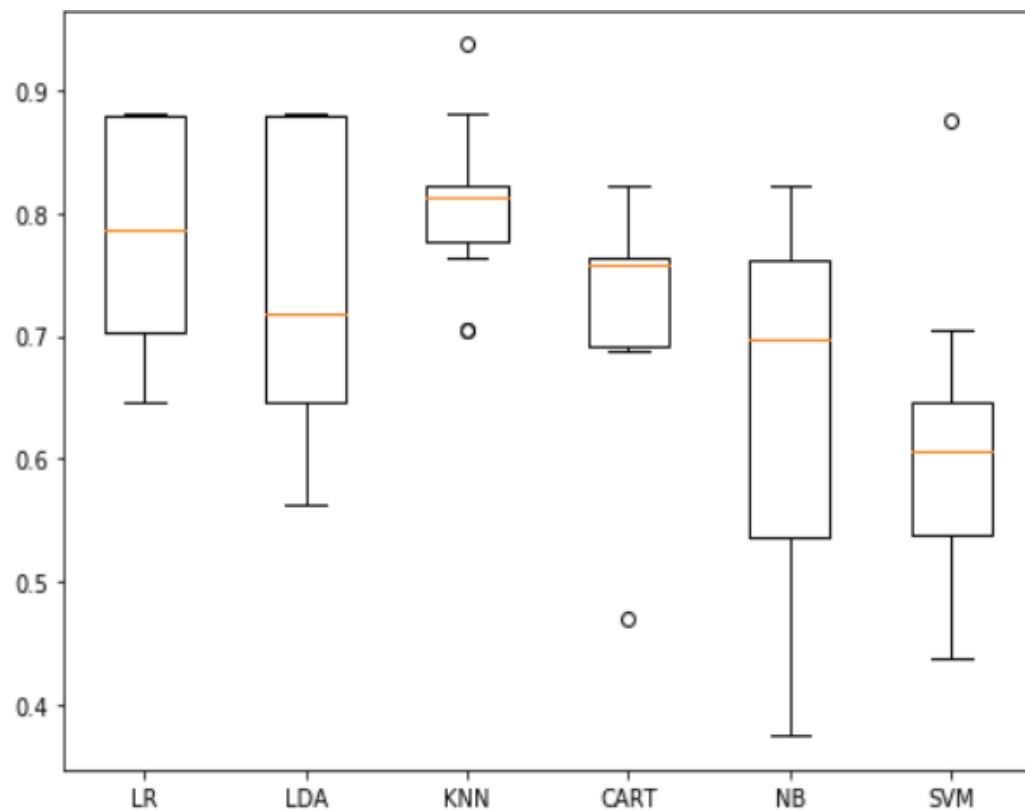
2. Histogram representation of dataset



3. Density plots for sonar dataset



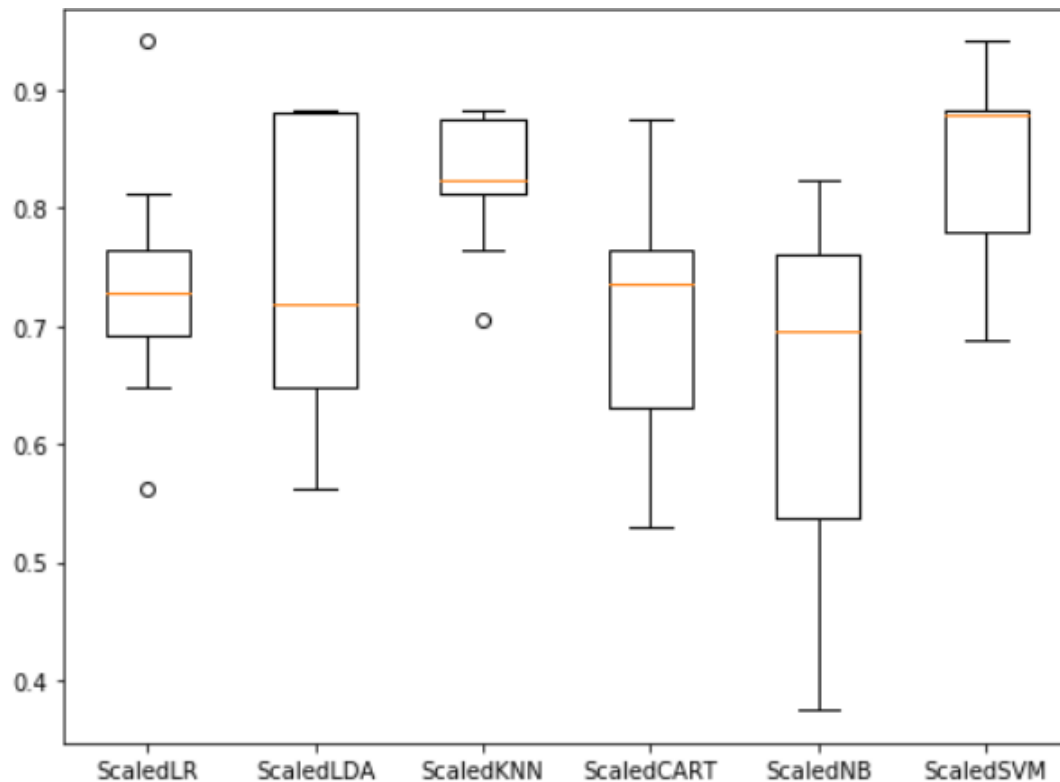
4. Correlation matrix for dataset



5. Boxplot for machine learning algorithm comparison initially

LR: 0.782721 (0.093796)	ScaledLR: 0.734191 (0.095885)
LDA: 0.746324 (0.117854)	ScaledLDA: 0.746324 (0.117854)
KNN: 0.808088 (0.067507)	ScaledKNN: 0.825735 (0.054511)
CART: 0.723162 (0.095178)	ScaledCART: 0.711397 (0.098859)
NB: 0.648897 (0.141868)	ScaledNB: 0.648897 (0.141868)
SVM: 0.608824 (0.118656)	ScaledSVM: 0.836397 (0.088697)

6. Accuracy comparison before and after scaling the algorithms



7. Scaled Algorithm accuracy comparison

```
print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
means = grid_result.cv_results_['mean_test_score']
stds = grid_result.cv_results_['std_test_score']
params = grid_result.cv_results_['params']
ranks = grid_result.cv_results_['rank_test_score']
for mean, stdev, param, rank in zip(means, stds, params, ranks):
    print("#%d %f (%f) with: %r" % (rank, mean, stdev, param))
```

Best: 0.849398 using {'n_neighbors': 1}

8. Accuracy using K-Nearest Neighbour Algorithm

```
print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
means = grid_result.cv_results_['mean_test_score']
stds = grid_result.cv_results_['std_test_score']
params = grid_result.cv_results_['params']
ranks = grid_result.cv_results_['rank_test_score']
for mean, stdev, param, rank in zip(means, stds, params, ranks):
    print("#%d %f (%f) with: %r" % (rank, mean, stdev, param))
```

Best: 0.867470 using {'C': 1.5, 'kernel': 'rbf'}

9 Accuracy using Support Vector Machine (SVM)

0.8571428571428571

```
[[23  4]
 [ 2 13]]
```

	precision	recall	f1-score	support
M	0.92	0.85	0.88	27
R	0.76	0.87	0.81	15
accuracy			0.86	42
macro avg	0.84	0.86	0.85	42
weighted avg	0.86	0.86	0.86	42

10. Classification Report of the data

```
model.predict([[0.02,0.0371,0.0428,0.0207,0.0954,0.0986,0.1539,0.1601,0.3109,0.2111,0.1609,0.1582,0.2238,0.0645,
0.066,0.2273,0.31,0.2999,0.5078,0.4797,0.5783,0.5071,0.4328,0.555,0.6711,0.6415,0.7104,0.808,0.6791,
0.3857,0.1307,0.2604,0.5121,0.7547,0.8537,0.8507,0.6692,0.6097,0.4943,0.2744,0.051,0.2834,0.2825,0.4256,
0.2641,0.1386,0.1051,0.1343,0.0383,0.0324,0.0232,0.0027,0.0065,0.0159,0.0072,0.0167,0.018,0.0084,0.009,0.0032
]])
```

```
array(['M'], dtype=object)
```

11. OUTPUT

B. PLAGIARISM REPORT

paper (3).docx

ORIGINALITY REPORT

8%

SIMILARITY INDEX

6%

INTERNET SOURCES

3%

PUBLICATIONS

2%

STUDENT PAPERS

PRIMARY SOURCES

1

mafiadoc.com

Internet Source

4%

2

Submitted to Humble Independent School District

Student Paper

2%

3

www.mdpi.com

Internet Source

1%

4

"Microservices in Big Data Analytics", Springer Science and Business Media LLC, 2020

Publication

1%

Exclude quotes Off

Exclude bibliography Off

Exclude matches Off

C. PAPER WORK

UNDERWATER SURFACE TARGET PREDICTION THROUGH SONAR

NALLA SAIKIRAN [1], MUNAGALA PAVAN V N SAI BHAGAVAN [2], SUJIHELEN L [3]

[1][2] UG Student, Dept. of CSE, Sathyabama Institute of Science and Technology, Chennai, India

[3] Associate professor, Dept. of CSE, Sathyabama Institute of Science and Technology, Chennai, India

ABSTRACT: Sonar signals acknowledgment is the significant assignment in recognizing the existence of few critical articles beneath the ocean. In naval infantry, sonar wave signals are utilized in lieu of visuals to visit submerged as well as find foe submarines in closeness. Specifically, grouping calculation in data-mining had been applied in the sonar wave acknowledgment for perceiving the sort of surfaces from which sonar signals are bobbed. Grouping calculations in customary data mining and data science approach offer reasonable precision via preparing an order prototype with complete dataset, in clumps. It is notable that the sonar waves are constant, and they are gathered as stream of data. Albeit the prior order calculations are viable in

conventional groups preparing, it might not be workable for measured classification learning. As sonar waves streams of data can add up to boundlessness, the whole dataset pre-processing time should be set to base to satisfy the need for fast. This paper provides the elective data mining technique reasonable for reformist cleansing of boisterous data by means of quick clash examination from stream of data without need of gaining from the entire stream of data at a single time. Reenactment tests led and prevalent outcomes are seen in carrying the viability of approach.

Keywords: SONAR signals, Underwater surface, prediction, data visualization, rocks, mines.

I INTRODUCTION

Sonar, an aggregate term for various gadgets that utilization sound frequencies as data carriage, is a method that grants boats and different things to find and perceive objects inside water through an arrangement of sound signals, echoes [2]. It can achieve discovery, area, recognizable proof, and following of focuses in the marine climate and perform submerged correspondence, route, estimation, and different capacities [3]. In the previous few years, the utilization of sonar hardware is blasting forward. In light of the outrageous blurring of radio waves and visual signs inside the water region, acoustic waves are for the most part viewed as amazing techniques to detect submerged articles and targets [4]. Submerged acoustics, the common area for the investigation of the multitude of cycles related

with the age, event, transmission, and gathering of sound waves in the water region and its obstruction with limits, has generally been executed to submarine Distant Distance detecting, which can be depicted as a sonar signal method, is made available when an objective of interest can't be straightforwardly checked, and the data about is to get acquired optionally [1]. Picking the correct characterization machine model for the sonar waves acknowledgment is a significant part in identifying the existence of things of interest beneath the ocean. As it was called attention to in [2], submerged sensor networks uphold an assortment of utilizations, for example, sea testing organizations, climate checking, seaward

investigations, calamity anticipation, helped route, and mine surveillance. Submerged sensor networks are not difficult to send and wipe out the need of links, and they don't meddle with delivery action. Nonetheless,

sonar flags that engender submerged particularly in long stretch are inclined to commotion and obstructions. Specifically, characterization methods in data mining have been utilized broadly in sonar wave acknowledgment to recognize the outside of the objective item from which the sonar waves are repeated [3–5]. Order calculations in conventional data mining approach might have the option to accomplish generous exactness by inciting a characterization model utilizing the entire dataset. The acceptance anyway is generally made and rehashed in clumps, which infers certain decrease in exactness among the machine learning models refreshes which might be normal [6]. Also, the time to update may turn out to be progressively large as the entire majority of data gets bigger when new data gathers. Much the same as any stream of data, it is being realized that sonar waves are unremitting, and they were detected in persistent way.

Beneficial outcomes are accomplished, when we think about the exhibition of the classifier algorithms in the system like standard classification algorithms like Support Vector Machine, irregular backwoods, neural organizations, Adabag and so on, utilizing different assessing measurements like exactness, territory under bend, affectability, specifically and so on signals can be utilized to build forecasts for the submerged areas, mineshafts and shakes [3]. Scientists are utilizing the aftereffects of AI for constructing the forecast models in various spaces [4]. In this test, after the pre-preparing of the data, distinctive AI classification algorithms are prepared to validate the accomplishment of arrangement. The lead for the best classifiers included correlation with few norm classification algorithms like C4.5, Adabag, Random Forest, SVM and so forth

II RELATED WORK

2.1 STATISTICS-BASED NOISE DETECTION METHODS.

Hamed Komari Alaie et al [1] proposed another strategy for identifying

focuses in detached sonar signals utilizing variable limit. Here, the target data is handled as expected and recurrence area. For arranging, Bayesian method is utilized and back dissemination is assessed by estimating the parameters of a probability distribution by maximizing a likelihood function. Then object was found by combining the area focuses in the 2 areas using Least Mean Square (LMS) versatile channel.

2.2 CLASSIFICATION-BASED NOISE DETECTION METHODS

Dhiraj Neupane et al [2] Underwater acoustics has been executed for the most part in the form of the sound route and going strategies for underwater craft resemblance, assessment of sea resources, climate reviewing, target and item acknowledgment, and estimation and look over of aural references in the submerged environment. Accompanied by the fast improvement in innovation and science, the headway in the sonar frameworks has enlarged, bringing about a down growth in submerged setbacks. The sonar wave handling and programmed intent acknowledgment utilizing sonar signs or symbolism is itself a difficult interaction. Then, exceptionally progressed data driven AI and profound learning-based techniques are being actualized for getting a few sorts of data from submerged sound dataset. This paper audits the new sonar programmed object acknowledgment, following, or discovery will works utilizing profound learning calculations. A careful investigation of the accessible services is completed, and the working strategy, results, and other essential insights about the dataset procurement measure, the data made use, and data in regard to hyper-boundaries is introduced in this article.

2.3 SIMILARITY-BASED NOISE DETECTION PROCEDURE

This gathering of techniques for the most part requires a backing by which

the data is contrasted with quantify how comparable or divergent they are to backing. *Zhiyuan Zhang et al*[3] the analysts initial separated data into numerous subsets earlier to looking for subset

that would have caused best decrease in uniqueness inside the preparation dataset whenever deleted. The disparity capacity could be any capacity restoring a low an incentive between comparable components and a high incentive in the middle of disparate components, like difference. Nonetheless, the creators commented that it is hard to locate an all-inclusive difference work. Each pair of items in a hyperclique design has a significant degree of similitude identified with the power of connection between the two examples. The HCleaner sift through cases avoided from any hyperclique design as commotion. Another group of specialists *Brighton et al [5]* applied k-NN calculation, which basically contrasts test data and adjoining data to decide if they are anomalies by making use of their neighbors. The creators considered examples of conduct among data to detail Wilson's altering approach, a bunch of decides that naturally selects the whole data to be cleansed.

III PROPOSED SYSTEM

The principle worry of investigation in the area of AI is making to frame a planned machine for ordering the gauge of items, in view of the achievable data. The result of proposed system assists with anticipating the set off sound signals reflected from the surface Rock or Mine. Proposed structure techniques: In general, in the real world or reasonable matter, then there is no check around the kinds of dataset. Some critical pre-handling like evacuation of not found qualities, highlight determination. AI centres around taking up contemporary procedures to handle immense measure of complicated data with cheap cost. The theoretical perspective on proposed structure have been addressed in Figure 1.

Figure 1 depicts the system of the expectation model made to decide the surface to be a stone or mine dependent on around sixty-one factors or highlights, handled by 10 diverse classification models, which provide yields with an adequate exactness and accuracy rate. Pre-handling is made use for Missing qualities are brought out by displacing them by means of mean worth attribution.

The best 50 highlights positioned by mean Gini file is chosen and taken care of to the forecast model. Furthermore, in the expectation model, Various machine learning classifiers are investigated and executed to locate the most ideal arrangement. Irregular timberland, being a group model has shown the best with 83.17% of exactness. The result of this proposed system assists with anticipating the focused-on area to be Rock or a Mine.

FIG 1 OVERVIEW OF THE PROPOSED SYSETM

IV RESULT AND DISCUSSION

With quick growth in innovation and science, sonar programmed target acknowledgment had been created in brief timeframe length. Nonetheless, these strategies have numerous weaknesses that should be survived. Because of the idea of submerged land, data obtaining or handling strategies are more confined than that ashore, hence there are a few difficulties in utilizing more forceful and open procedures. The aftereffects of 10-overlay cross approval strategy are

introduced graphically and talked about much in detail.

The data is being made into 2 parts. Learning algorithm have applied on training data and based on the learning, assumptions are being made on testing data.

The testing data is 30% of the whole dataset as shown in Fig 2.

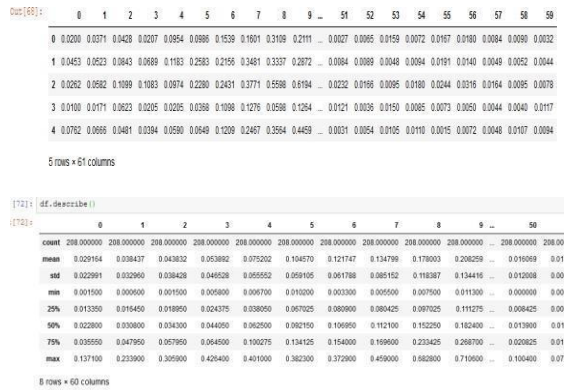


FIG 2 DATA SET

4.3 DATA CORRELATION REPRESENTATION

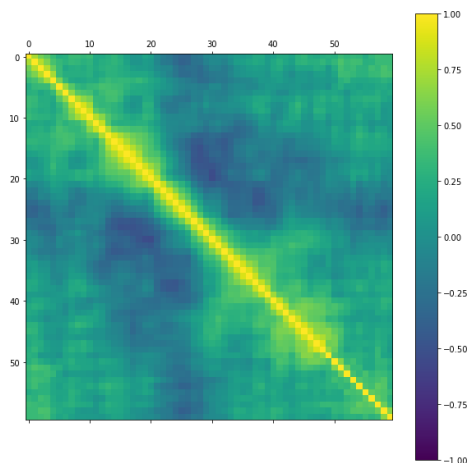


Fig 2 DATA CORRELATION REPRESENTATION

4.4 DIFFERENT DIMENSIONS OF FREQUENCY IN VERTICAL AXIS AND HORIZONTAL AXIS



Fig 3 DIFFERENT DIMENSIONS OF FREQUENCY USING HISTOGRAMS

4.5 DATA DISTRIBUTION PROCESS IN DENSITY PLOTS REPRESENTATION

Fig 4 DATA DISTRIBUTION PROCESS IN DENSITY PLOTS REPRESENTATION

V CONCLUSION

Precise acknowledgment of sonar wave is familiar to be a difficult grievance however it is having a huge commitment in military usage. One central point in falling apart the exactness is commotion in the surrounding submerged climate. Commotion creates turmoil in the

development of arrangement models. A
satisfactory forecast smaller than
expected, joined with the AI
characterizing highlights, is

mentioned which could close if objective of sound signal is either a stone or a mineshaft or some other life form or any other water object.

REFERENCES:

- [1] Hamed Komari Alaie, Hassan Farsi, "Passive Sonar Target Detection Using Statistical Classifier and Adaptive Threshold" Department of Electrical and Computer Engineering, University of Birjand, Appl. Sci. 2018, 8, 61; doi:10.3390/app8010061
- [2] Dhiraj Neupane and Jongwon Seok, "A Review on Deep Learning-Based Approaches for Automatic Sonar Target Recognition" Department of Data and Communication Engineering, Changwon National University, Changwon-si, Gyeongsangnam-do 51140, Electronics 2020, 9, 1972; doi:10.3390/electronics9111972
www.mdpi.com/journal/electronics
- [3] Zhiyuan Zhang, Xia Feng, "New Methods for Deviation-based Outlier Detection in Large Database", School of Computer Science & Technology, Civil Aviation University of China, Tianjin, 300300 zy-zhang@cauc.edu.cn, xfeng@cauc.edu.cn.
- [4] H. Xiong, G. Pandey, M. Steinbach, and V. Kumar, "Enhancing data analysis with noise removal," IEEE Transactions on Knowledge and Data Engineering, vol. 18, no. 3, pp. 304–319, 2006.
- [5] H. Brighton and C. Mellish, "Advances in instance selection for instance-based learning algorithms," Data Mining and Knowledge Discovery, vol. 6, no. 2, pp. 153–172, 2002.
- [6] N. Hooda et al. "B 2 FSE framework for high dimensional imbalanced data: A case study for drug toxicity prediction", Neurocomputing, (2018)
- [7] Corinna, Cortes; Vladimir N., Vapnik. "Support-vector networks". Machine Learning. 20 (3): 273–297; doi: 10.1007/BF00994018.
- [8] Kégl, Balázs. "The return of AdaBoost.MH: multiclass Hamming trees". arXiv:1312.6086. (20 December 2013).

[9] Pearl, Judea. Causality: Models, Reasoning, and Inference. Cambridge University Press. ISBN 0- 521-77362-8. OCLC 4229125. (2010).

[10] Huang, Jin. Performance measures of machine learning. University of Western Ontario, (2006).

[11]. Bradley, Andrew P. "The use of the area under the ROC curve in the evaluation of machine learning algorithms." Pattern recognition 30.7: 1145-1159.

D. SOURCE CODE

Sonar data set Machine Learning project

```
import numpy as np

from matplotlib import
pyplot import
matplotlib.pyplot as plt

import seaborn as sn

from pandas import read_csv, set_option

from pandas.plotting import scatter_matrix

from sklearn.preprocessing import StandardScaler

from sklearn.model_selection import train_test_split, KFold, cross_val_score, GridS
earchCV

from sklearn.linear_model import
LogisticRegression from sklearn.tree import
DecisionTreeClassifier from sklearn.neighbors
import KNeighborsClassifier

from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

from sklearn.naive_bayes import GaussianNB

from sklearn.svm import SVC

from sklearn.pipeline import Pipeline

from sklearn.ensemble import AdaBoostClassifier, GradientBoostingClassifier, Ran
domForestClassifier, ExtraTreesClassifier

from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
```

```
import warnings
```

```
warnings.filterwarnings('ignore')
```

```
dataset = read_csv('sonar.all-data.csv', header=None)
```

```
dataset.head()
```

Exploratory Data analysis (EDA)

```
dataset.shape
```

```
dataset.dtypes
```

```
dataset.head()
```

```
dataset.describe()
```

```
dataset.groupby(60).size()
```

```
dataset.hist(sharex=False, sharey=False, xlabelsize=1, ylabelsize=1, figsize=(12,12)
)
```

```
pyplot.show()
```

```
dataset.plot(kind='density', subplots=True, layout=(8,8), sharex=False,
legend=False, fontsize=1, figsize=(12,12))
```

```
pyplot.show()
```

```
fig = pyplot.figure()
```

```
ax = fig.add_subplot(111)
```

```
cax = ax.matshow(dataset.corr(), vmin=-1, vmax=1,
```

```
interpolation='none') fig.colorbar(cax)
```

```
fig.set_size_inches(10,10)
```

```
pyplot.show()
```

```
x = dataset.corr()
```

```
plt.figure(figsize = (60,60))
```

```
sn.heatmap(x,annot = True,cmap =
```

```
plt.cm.CMRmap_r) plt.show()
```

Bulding Models

```
array = dataset.values
```

```
X = array[:,0:-1].astype(float)
```

```
Y = array[:, -1] validation_size
```

```
= 0.2
```

```
seed = 7
```

```
X_train, X_validation, Y_train, Y_validation = train_test_split(X, Y, test_size=validation_size, random_state=seed)
```

```
num_folds = 10
```

```
seed = 7
```

```
scoring =
```

```
'accuracy' models
```

```
= []
```

```
models.append(('LR', LogisticRegression()))
```

```
models.append(('LDA', LinearDiscriminantAnalysis()))
```

```
models.append(('KNN', KNeighborsClassifier()))
```

```
models.append(('CART', DecisionTreeClassifier()))
```

```
models.append(('NB', GaussianNB()))
```

```
models.append(('SVM', SVC()))
```

```
results = []
```

```
names = []
```

```
for name, model in models:
```

```
    kfold = KFold(n_splits=num_folds, random_state=seed)
```

```
    cv_results = cross_val_score(model, X_train, Y_train, cv=kfold, scoring=scoring)
```

```
    results.append(cv_results)
```

```
    names.append(name)
```

```
msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
```

```

print(msg)

fig = pyplot.figure()

fig.suptitle('Algorithm
Comparison') ax =

fig.add_subplot(111)

pyplot.boxplot(results)

ax.set_xticklabels(names)

fig.set_size_inches(8,6)

pyplot.show()

pipelines = []

pipelines.append(('ScaledLR', Pipeline([('Scaler', StandardScaler()), ('LR', LogisticRe
gression())])))

pipelines.append(('ScaledLDA', Pipeline([('Scaler', StandardScaler()), ('LDA', LinearD
iscriminantAnalysis())])))

pipelines.append(('ScaledKNN', Pipeline([('Scaler', StandardScaler()), ('KNN', KNeig
hbsClassifier())])))

pipelines.append(('ScaledCART', Pipeline([('Scaler', StandardScaler()), ('CART', Dec
isionTreeClassifier())])))

pipelines.append(('ScaledNB', Pipeline([('Scaler', StandardScaler()), ('NB', Gaussian
NB())])))

pipelines.append(('ScaledSVM', Pipeline([('Scaler', StandardScaler()), ('SVM', SVC()
)])))

results = []

names = []

for name, model in pipelines:

```

```
kfold = KFold(n_splits=num_folds, random_state=seed)
```

```
cv_results = cross_val_score(model, X_train, Y_train, cv=kfold, scoring=scoring)
```

```

print(cv_results)

results.append(cv_results)

names.append(name)

msg = "%s: %f (%f)" % (name, cv_results.mean(),
cv_results.std()) print(msg)

fig = pyplot.figure()

fig.suptitle('Scaled Algorithm
Comparison') ax = fig.add_subplot(111)

pyplot.boxplot(results)

ax.set_xticklabels(names)

fig.set_size_inches(8,6)

pyplot.show()

```

Algorithm Tuning: KNN and SVM show as the most promising options

KNN algorithm tuning

```

scaler = StandardScaler().fit(X_train)

rescaledX = scaler.transform(X_train)

neighbors = [1,3,5,7,9,11,13,15,17,19,21]

param_grid = dict(n_neighbors=neighbors)

model = KNeighborsClassifier()

kfold = KFold(n_splits=num_folds, random_state=seed)

grid = GridSearchCV(estimator=model, param_grid=param_grid, scoring=scoring, cv
=kfold)

grid_result = grid.fit(rescaledX, Y_train)

```

```
print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))  
  
means = grid_result.cv_results_['mean_test_score']
```

```

stds =

grid_result.cv_results_['std_test_score']

params = grid_result.cv_results_['params']

ranks = grid_result.cv_results_['rank_test_score']

for mean, stdev, param, rank in zip(means, stds, params, ranks):

    print("#%d %f (%f) with: %r" % (rank, mean, stdev, param))

```

Parameters of SVM are C and kernel. Try a number of kernels with various values of C with less bias and more bias (less than and greater than 1.0 respectively

SVM algorithm tuning

```

scaler = StandardScaler().fit(X_train)

rescaledX = scaler.transform(X_train)

c_values = [0.1, 0.3, 0.5, 0.7, 0.9, 1.0, 1.3, 1.5, 1.7, 2.0]

kernel_values = ['linear', 'poly', 'rbf', 'sigmoid']

param_grid = dict(C=c_values, kernel=kernel_values)

model = SVC()

kfold = KFold(n_splits=num_folds, random_state=seed)

grid = GridSearchCV(estimator=model, param_grid=param_grid, scoring=scoring, cv=kfold)

grid_result = grid.fit(rescaledX, Y_train)

print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))

means = grid_result.cv_results_['mean_test_score']

stds =

grid_result.cv_results_['std_test_score']

params = grid_result.cv_results_['params']

```

```
ranks = grid_result.cv_results_['rank_test_score']  
  
for mean, stdev, param, rank in zip(means, stds, params, ranks):  
    print("#%d %f (%f) with: %r" % (rank, mean, stdev, param))
```

Right now SVM is proving the best with accuracy of 86.7% over KNN's best of 84.9%. (But what about variance? KNN seemed to indicate a tighter variance during spot checking).

```
# ensembles
```

```
ensembles = []
```

```
# Boosting methods
```

```
ensembles.append(('AB', AdaBoostClassifier()))
```

```
ensembles.append(('GBM', GradientBoostingClassifier()))
```

```
# Bagging methods
```

```
ensembles.append(('RF', RandomForestClassifier()))
```

```
ensembles.append(('ET', ExtraTreesClassifier()))
```

```
results = []
```

```
names = []
```

```
for name, model in ensembles:
```

```
    kfold = KFold(n_splits=num_folds, random_state=seed)
```

```
    cv_results = cross_val_score(model, X_train, Y_train, cv=kfold, scoring=scoring)
```

```
    print(cv_results)
```

```
    results.append(cv_results)
```

```
    names.append(name)
```

```
    msg = "%s: %f (%f)" % (name, cv_results.mean(),
```

```
    cv_results.std()) print(msg)
```

```
# compare ensemble algorithms
```

```
fig = pyplot.figure()
```

```
fig.suptitle('Ensemble Algorithm
```

```
Comparison') ax = fig.add_subplot(111)
```

```
pyplot.boxplot(results)
```

```
ax.set_xticklabels(names)
```

```
fig.set_size_inches(8,6)
```

```
pyplot.show()
```

GBM might be worthy of further study, but for now SVM shows a lot of promise as a low complexity and stable model for this problem.

Finalize Model with best parameters found during tuning step.

```
# prepare model
```

```
scaler = StandardScaler().fit(X_train)
```

```
rescaledX = scaler.transform(X_train)
```

```
model = SVC(C=1.5) # rbf is default kernel
```

```
model.fit (rescaledX, Y_train)
```

```
# estimate accuracy on validation set
```

```
rescaledValidationX = scaler. transform(X_validation)
```

```
predictions = model.predict(rescaledValidationX)
```

```
print(accuracy_score(Y_validation, predictions))
```

```
cm=confusion_matrix(Y_validation, predictions)
```

```
print(cm)
```

```
print(classification_report(Y_validation, predictions))
```

Y_validation