

INSURANCE FRAUD DETECTION USING MACHINE LEARNING

A PROJECT REPORT

Submitted by

Mr. Syed Atif	20171CSE0702
Mr. Syed Shoaib	20171CSE0790
Mr. Tejas Bhatnagar	20171CSE0721
Mr. Thanuj Kumar S	20171CSE0726
Mr. Utsav Deep	20171CSE0734

Under the guidance of

Prof. T. Ramesh - Asst Prof CSE

in partial fulfillment for the award of the degree of

**BACHELOR OF TECHNOLOGY
IN**

COMPUTER SCIENCE AND ENGINEERING

At



Department of Computer Science and Engineering

School of Engineering

PRESIDENCY UNIVERSITY

BANGALORE

MAY 2021

**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING
SCHOOL OF ENGINEERING PRESIDENCY UNIVERSITY**

CERTIFICATE

This is to certified that the Project report **“INSURANCE FRAUD
DETECTION USING MACHINE LEARNING”** being submitted by

Mr. Syed Atif	20171CSE0702
Mr. Syed Shoaib	20171CSE0709
Mr. Tejas Bhatnagar	20171CSE0721
Mr. Thanuj Kumar S	20171CSE0726
Mr. Utsav Deep	20171CSE0734

in partial fulfillment of requirement for the award of degree of **Bachelor of
Technology in Computer Science and Engineering** is a bonafide work
carried out under my supervision.

Dr. C. KALAIARASAN

UP-II (In charge)
Associate Dean-Admin
Department of CSE
Presidency University

**Prof. T. Ramesh -
Asst Prof CSE**
Guide Professor
Department of CSE

**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING
SCHOOL OF ENGINEERING PRESIDENCY UNIVERSITY**

DECLARATION

I hereby declare that the work, which is being presented in the project report entitled “**INSURANCE FRAUD DETECTION USING MACHINE LEARNING**” in partial fulfilment for the award of Degree of **Bachelor of Technology in Computer Science and Engineering**, is a record of our own investigations carried under the guidance of **Prof. T. Ramesh - Asst Prof CSE, Professor, Department of Computer Science and Engineering, School of Engineering, Presidency University, Bangalore.**

We have not submitted the matter presented in this report anywhere for the award of any other Degree.

Mr. Syed Atif	20171CSE0702
Mr. Syed Shoaib	20171CSE0709
Mr. Tejas Bhatnagar	20171CSE0721
Mr. Thanuj Kumar S	20171CSE0726
Mr. Utsav Deep	20171CSE0734

ABSTRACT

The insurance industries consist of more than thousand companies in worldwide.

And collect more than one trillion of dollars premiums in each year.

The auto\vehicle insurance fraud is the most prominent type of insurance fraud, which can be done by fake accident claim. In this project, focusing on detecting the auto\vehicle fraud by using, machine learning techniques.

Ideally, an insurance agent would have the capacity to investigate each case and conclude whether it is genuine or not. However, this process is not only time consuming, but costly. Sourcing and funding the skilled labor required to review each of the thousands of claims that are filed a day is simply unfeasible. This is where machine learning comes in to save the day. Where we use two of the well-known machine learning algorithms:

1. XGBoost
2. Support Vector Machine (SVM)

These algorithms were best suited for the dataset we had collected and give the best accuracy of the results we were expecting.

ACKNOWLEDGEMENT

First of all, we are indebted to the GOD ALMIGHTY for giving us an opportunity to excel in our efforts to complete this project on time.

We express our sincere thanks to our respected dean **Dr. Abdul Sharief**, Dean, School of Engineering, Presidency University for getting us permission to undergo the project.

We record our heartfelt gratitude to our beloved professor **Dr. C. Kalaiarasan**, University Project-II In charge, Associate Dean-Admin, Department of Computer Science and Engineering, Presidency University for rendering timely help for the successful completion of this project.

We are greatly indebted to our guide **Prof. T. Ramesh**, Department of Computer Science and Engineering, Presidency University for his/her inspirational guidance, valuable suggestions and providing us a chance to express our technical capabilities in every respect for the completion of the project work.

We thank our friends for the strong support and inspiration they have provided us in bringing out this project.

Mr. Syed Atif
Mr. Syed Shoaib
Mr. Tejas Bhatnagar
Mr. Thanuj Kumar S
Mr. Utsav Deep

List of Tables

Sl. No.	Table Name	Table Caption	Page No.
1.	Table 1.1	Literature review information table	7
2.	Table 1.2	Measure of SVM and XGBoost	25
3.	Table 1.3	Model Evaluation Formulas	33
4.	Table 1.4	Model Evaluation Results	34
5.	Table 1.5	AUC Evaluation	34

List of Figures

Sl. No.	Figure Name	Caption	Page No.
1	Fig 1.1	Block Diagram for Machine Learning Steps	3
2	Fig 1.2	Process of Fraud Detection Used by Companies	9
3	Fig 1.3	System Design for our model	12
4	Fig 1.4	Implementation Steps of Machine Learning	13
5	Fig 1.5	XGBoost Steps	14
6	Fig 1.6	Support Vector Machine	16
7	Fig 1.7	Hyper-Plane for SVM	17
8	Fig 1.8	Clusters of Data Points	22
9	Fig 1.9	DBSCAN	22
10	Fig 2.0	Model Building Steps	23
11	Fig 2.1	Model Building	24
12	Fig 2.2	Model Evaluation	24
13	Fig 2.3	Snapshot of Main.py File	25
14	Fig 2.4	Model Selection Process	27
15	Fig 2.5	Heroku Platform	29
16	Fig 2.6	Heroku Login	30
17	Fig 2.7	Initialize git Repository	30
18	Fig 2.8	Deploying the App on Heroku Platform	31
19	Fig 2.9	Skeleton view of confusion matrix	32
20	Fig 3.0	Confusion Matrix of SVM	32
21	Fig 3.1	Confusion Matrix of XGB	32
22	Fig 3.2	Prediction Results	35
23	Fig 3.2	Working Web App	36

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	ii
	ACKNOWLEDGMENT	iii
1.	INTRODUCTION	1
	1.1 INTRODUCTION TO AUTO INSURANCE FRAUD	1
	1.2 INTRODUCTION MACHINE LEARNING	2
2.	LITERATURE SURVEY	7
3.	EXISTING SYSTEM	8
	3.1 POLICIES IN PLACE	8
	3.2 PRIVATE INVESTIGATORS	9
	3.3 ANALYTICS TOOLS	9
	3.4 THE TECHNOLOGY BEHIND FRAUD DETECTION	10
4.	PROPOSED WORK	11
5.	SYSTEM DESIGN	12
6.	IMPLEMENTATION	13
	6.1 MACHINE LEARNING ALGORITHMS REQUIRED FOR THIS PROJECT	13
	6.1.1 XGBoost	14
	6.1.2 SVM	15
	6.2 DATA COLLECTION	17
	6.3 DATA PREPROCESSING IN MACHINE LEARNING	18
	6.3.1 WHY DO WE NEED DATA PREPROCESSING?	19
	6.4 EDA (EXPLORATORY DATA ANALYSIS)	19
	6.4.1 EXPLORATORY DATA ANALYSIS TOOLS	20
	6.4.2 TYPES OF EXPLORATORY DATA ANALYSIS	20
	6.5 CLUESTRING	21

	6.6 MODEL BUILDING	23
	6.7 MODEL EVALUATION	24
	6.7.1 HOLDOUT	26
	6.7.2 CROSS-VALIDATION	26
	6.7.3 MODEL EVALUATION METRICS	27
	6.8 MODEL SELECTION	27
	6.8.1 CONSIDERATIONS FOR MODEL SELECTION	28
	6.8.2 MODEL SELECTION TECHNIQUES	28
	6.9 MODEL DEPLOYMENT	28
	6.9.1 HEROKU APP CREATION AND DEPLOYMENT	29
7.	TESTING	29
8.	RESULT	32
9.	CONCLUSION	35
10.	REFERENCES	37

1. INTRODUCTION

Insurance fraud is any act committed to defraud an insurance process. It occurs when a claimant attempts to obtain some benefit or advantage they are not entitled to, or when an insurer knowingly denies some benefit that is due. False insurance claims are insurance claims filed with the fraudulent intention towards an insurance provider.

Insurance fraud has existed since the beginning of insurance as a commercial enterprise. Fraudulent claims account for a significant portion of all claims received by insurers, and cost billions of dollars annually. Types of insurance fraud are diverse and occur in all areas of insurance. Insurance crimes also range in severity, from slightly exaggerating claims to deliberately causing accidents or damage. Fraudulent activities affect the lives of innocent people, both directly through accidental or intentional injury or damage, and indirectly by the crimes leading to higher insurance premiums. Insurance fraud poses a significant problem, and governments and other organizations try to deter such activity.

People who commit insurance fraud include:

- organized criminals who steal large sums through fraudulent business activities,
- professionals and technicians who inflate service costs or charge for services not rendered, and
- ordinary people who want to cover their deductible or view filing a claim as an opportunity to make a little money.

1.1 INTRODUCTION TO AUTO INSURANCE FRAUD

Auto insurance fraud ranges from misrepresenting facts on insurance applications and inflating insurance claims to staging accidents and submitting claim forms for injuries or damage that never occurred, to false reports of stolen vehicles.

There are two main categories for auto insurance fraud—hard insurance fraud and soft insurance fraud.

Hard insurance fraud involves creating a situation that allows you to make a claim on an auto insurance policy. A common example for this type of fraud is staging a auto wreck with the goal of benefitting from the resulting claim.

Soft insurance fraud involves exaggerating or flat out fabricating the damages or expenses involved in an auto insurance policy or claim. A common example for this type of fraud is a mechanic making unnecessary repairs to your auto with the goal of charging the insurance company a higher bill.

Soft insurance fraud is the more common form because it's easier to commit and harder to spot. Hard insurance fraud is usually more costly to the insurance provider and comes with higher penalties.

Auto fraud is one of the largest and most well-known problems that insurers face. Fraudulent claims can be highly expensive for each insurer. Therefore, it is important to know which claims are correct and which are not. Ideally, an insurance agent would have the capacity to investigate each case and conclude whether it is genuine or not. However, this process is not only time consuming, but costly. Sourcing and funding the skilled labor required to review each of the thousands of claims that are filed a day is simply unfeasible.

This is where machine learning comes in to save the day. Once the proper data is fed to the system it'll be very easy to find out if the claim is genuine or not.

1.2 INTRODUCTION TO MACHINE LEARNING

Machine Learning is said as a subset of Artificial intelligence that is mainly concerned with the development of algorithms which allow a computer to learn from the data and past experiences on their own. Machine learning is a method of data analysis that automates analytical model building. It is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention.

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

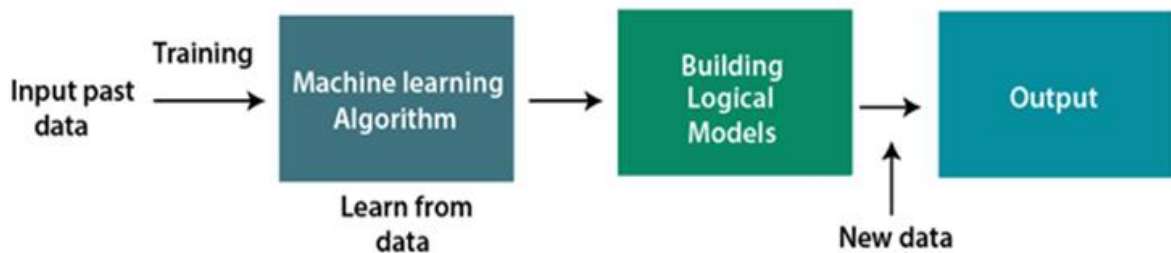


Fig 1.1 Block Diagram for Machine learning Steps

Machine learning algorithms are often categorized as supervised or unsupervised.

- **Supervised machine learning** algorithms can apply what has been learned in the past to new data using labeled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.

Types of Supervised Learning:

- **Classification:** It is a Supervised Learning task where output is having defined labels (discrete value). Example: Gmail classifies mails in more than one classes like social, promotions, updates, forum.
- **Regression:** It is a Supervised Learning task where output is having continuous value.

Example of Supervised Learning Algorithms:

- Linear Regression
 - Nearest Neighbor
 - Guassian Naive Bayes
 - Decision Trees
 - Support Vector Machine (SVM)
 - Random Forest
- **Unsupervised machine learning** algorithms are used when the information used to train is neither classified nor labeled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabeled data. The system doesn't figure out the right output, but it explores the data and can draw inferences from datasets

to describe hidden structures from unlabeled data.

Unsupervised learning is classified into two categories of algorithms:

Clustering: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.

Association: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

Types of Unsupervised Learning: -

- Clustering
- Exclusive (partitioning)
- Agglomerative
- Overlapping
- Probabilistic

Clustering Types: -

- Hierarchical clustering
- K-means clustering
- Principal Component Analysis
- Singular Value Decomposition
- Independent Component Analysis

- **Semi-supervised machine learning** algorithms fall somewhere in between supervised and unsupervised learning, since they use both labeled and unlabeled data for training – typically a small amount of labeled data and a large amount of unlabeled data. The systems that use this method are able to considerably improve learning accuracy. Usually, semi-supervised learning is chosen when the acquired labeled data requires skilled and relevant resources in order to train it / learn from it. Otherwise, acquiring unlabeled data generally doesn't require additional resources.
 - A common example of an application of semi-supervised learning is a text document classifier. This is the type of situation where semi-supervised learning is ideal because it would be nearly impossible to find a large amount of labeled text documents. This is simply because it is not time efficient to have a person read through entire text documents just to assign it a simple classification.

- So, semi-supervised learning allows for the algorithm to learn from a small amount of labeled text documents while still classifying a large amount of unlabeled text documents in the training data

How semi-supervised learning works

- The way that semi-supervised learning manages to train the model with less labeled training data than supervised learning is by using pseudo labeling. This can combine many neural network models and training methods. Here's how it works:
 - Train the model with the small amount of labeled training data just like you would in supervised learning, until it gives you good results.
 - Then use it with the unlabeled training dataset to predict the outputs, which are pseudo labels since they may not be quite accurate.
 - Link the labels from the labeled training data with the pseudo labels created in the previous step.
 - Link the data inputs in the labeled training data with the inputs in the unlabeled data.
 - Then, train the model the same way as you did with the labeled set in the beginning in order to decrease the error and improve the model's accuracy.
- **Reinforcement machine learning** algorithms is a learning method that interacts with its environment by producing actions and discovers errors or rewards. Trial and error search and delayed reward are the most relevant characteristics of reinforcement learning. This method allows machines and software agents to automatically determine the ideal behavior within a specific context in order to maximize its performance. Simple reward feedback is required for the agent to learn which action is best; this is known as the reinforcement signal.

Types of Reinforcement: There are two types of Reinforcement:

1.1 Positive –

Positive Reinforcement is defined as when an event, occurs due to a particular behavior, increases the strength and the frequency of the behavior. In other words, it has a positive effect on behavior.

- Advantages of reinforcement learning are:
- Maximizes Performance

- Sustain Change for a long period of time
- Disadvantages of reinforcement learning:
- Too much Reinforcement can lead to overload of states which can diminish the results

1.2. Negative–

Negative Reinforcement is defined as strengthening of a behavior because a negative condition is stopped or avoided.

- Advantages of reinforcement learning:
- Increases Behavior
- Provide defiance to minimum standard of performance
- Disadvantages of reinforcement learning:
- It Only provides enough to meet up the minimum behavior

2. LITERATURE SURVEY

Authors	Year	Name of the Paper	Problem Address/Objetives
T. Badriyah, Lailul Rahmaniah I. Syarif	2018	Nearest Neighbour and Statistics Method based for Detecting Fraud in Auto Insurance	<ul style="list-style-type: none"> • Nearest Neighbor based Method • Statistics Methods <p>This study includes nearest neighbor method and interquartile method to detect fraud in car insurance data. From the results of the conducted study, the best result is using Nearest Neighbor</p>
K. Supraja and S.J. Saritha	2017	Robust Fuzzy rule based techniques to detect frauds in insurance	<p>The Fuzzy Rule based technique is applied on the training dataset and based on the instances the degree of fraud or Legal is predicted.</p> <p>Concluded that this technique is used for high dimensional large dataset with accuracy</p>
Richard A. Bauder and Taghi M. Khoshgoftaar	2017	Medicare Fraud Detection Using Machine Learning Methods	<ul style="list-style-type: none"> • Supervised learner • Unsupervised learner • Hybrid learner <p>In this Paper we explore different ML methods to detect fraudulent Medicare providers and concludes that supervised learners are better than others</p>
Rekha Pal And Saurabh Pal	2015	Application of Data Mining Techniques in Health Fraud Detection	<p>Classification algorithm</p> <ul style="list-style-type: none"> • ID3(Iterative Dichotomise 3) • J48 • Naive bayes <p>Based on survey they concluded that the decision tree ID3 is the best in the three respective algorithms as it is more accurate.</p>

Table 1.1 Literature review information table

3. EXISTING SYSTEM

When we are talking about the current existing system of insurance fraud detection it usually involves the insurance companies that have a separate team of people, analytics tools, policies and private investigators to do background checks and find out if the claim made by the insured person is fraud or not.

Sometimes, it's obvious. For example, a claim for something the insured probably never owned and can't produce any documentation for, will be scrutinized. One of the front-line defenses though is to analyze claims history. Insurance companies all share information about claims into a central database, and both the insured and insurer generally have access to it. Was there a similar claim shortly before you cancelled your last policy? Big red flag. Do you file a claim every few months? Big red flag. Do you have two policies covering the same thing and you're trying to claim on both? Yet another, big red flag. The insurance companies usually know more about our claims history than we do.

3.1 POLICIES IN PLACE

Insurance companies usually share a list of things that make a claim suspicious. Few of the items that are on the list are:

- Claimant is in financial distress, worse off than they were when the policy started, or claiming shortly after the inception of the policy
- Increases in policy limits shortly before an alleged loss occurs
- Missing police reports, where appropriate
- Valuable items removed from the premises shortly before a fire loss
- Firefighters unable to gain access due to items blocking doors, etc.
- Slip and falls with no witnesses
- Altered documentation
- Delayed reporting of a claim

These things create reasons to look deeper into the claim and investigate the authenticity of the claim.

3.2 PRIVATE INVESTIGATORS

Insurance companies sometimes hire private investigators when they need someone experienced and capable of looking into someone that might seem suspicious to them. PI's are great for gathering information the insured person and gather proof if any. They are cheaper to afford for companies and they don't bad publicity that the company might face if they were to start a police investigation for every single suspicious claim.

3.3 ANALYTICS TOOLS

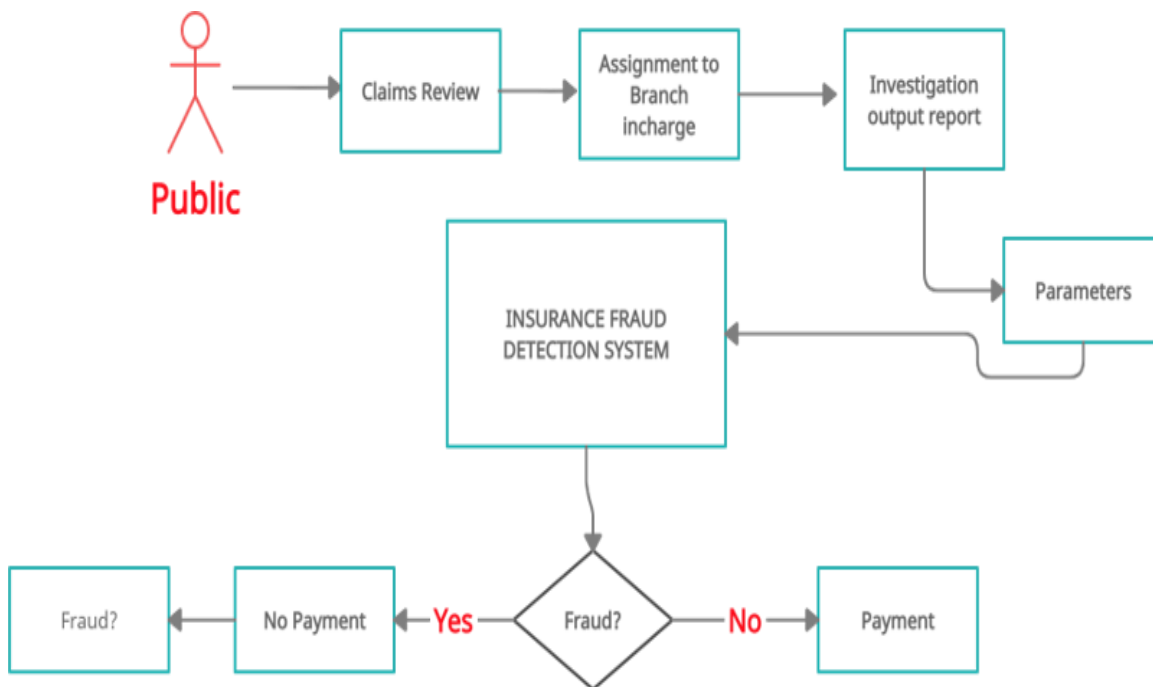


Fig 1.2 Process of Fraud Detection Used by Companies

Many organizations nowadays provide a range of data analytics and pattern recognition tools that help insurance companies detect an insurance fraud at a very early stage. It also helps them recognize patterns between similar types of fraudulent claims and deny the claim at the earliest stage possible so that loss to the company is minimum and they are also able to save time.

The insurance industry has developed sophisticated algorithms to detect fraud, as well as relying on tried-and-true methods like raising questions if multiple claim checks are being sent to the same address.

3.4 THE TECHNOLOGY BEHIND FRAUD DETECTION

In the world of Data Science, there are a great number of other methodologies and algorithms that accurately leverage large amounts of user data. Each of them has proven to effectively perform in some particular scenarios and situations. Machine Learning experts divide them into two main scenarios depending on the available dataset:

Scenario 1: The dataset has a sufficient number of fraud examples.

In this case, classic machine learning or statistics-based techniques are applied to detect fraudulent attacks. This involves training a machine learning model or employing adequate algorithms to estimate transaction legitimacy. We'll go through the most commonly used algorithms below.

Scenario 2: The dataset has no (or just a very little number of) fraud examples.

In the case that none of any previous information on fraudulent transactions was stored, the learning model is built based on examples of legitimate transactions. Before jumping into the commonly used learning models applied to fraud detection, it's to say that the majority has the same purpose of usage and differs only by its mathematical characteristics. Hence available data becomes a decisive factor when choosing appropriate learning models rather than an algorithm itself.

- Random Forest or random decision forests. This algorithm ensembles decision trees and accurately analyzes missing data, noise, outliers and errors. It is fast on train and score and, as a consequence, has become one of the preferable among fraud detection professionals.
- Artificial Neural Networks (ANN). This system simulates the function of the brain to perform tasks by learning from the past, extract rules and predict future activity based on the current situation. It can predict whether the transaction is fraudulent or not by classifying an input into predefined groups.
- Support Vector Machines (SVMs). It's an excellent prediction tool that can resolve a wide range of learning problems, such as handwritten digit recognition, classification of web pages and face detection. This method is capable of detecting fraudulent activity at the time of transaction.

- K-Nearest Neighbors (KNN). Also known as the “lazy learning” algorithm due to its simplicity: instead of making calculations once the data is introduced, it just stores it for further classification. The KNN algorithm rests on feature similarity and its proximity. When the nearest neighbor is fraudulent, the transaction is classified as fraudulent and when the nearest neighbor is legal, it is classified as legal.
- Logistic Regression is a prediction algorithm borrowed by machine learning from the fields of statistics. It's widely used for credit card fraud detection and credit scoring.

Finally, the goal of artificial intelligence in the field of insurance fraud is to make it easier for human agents to find and investigate fraudulent claims and transactions, rather than sifting through tons of claims in an exhausting and time-consuming way.

4. PROPOSED WORK

As part of this project, we propose the following objectives as our primary solutions to the problem of auto fraud insurance fraud:

- 1.** To build a classification methodology using a machine learning technique to determine whether a customer is placing a fraudulent insurance claim by using historical data.
- 2.** Design a front end for our application to make it accessible and easy to use.
- 3.** Design a system that is reliable and gives out accurate results.
- 4.** Deploying our application to a cloud platform like Heroku.

5. SYSTEM DESIGN

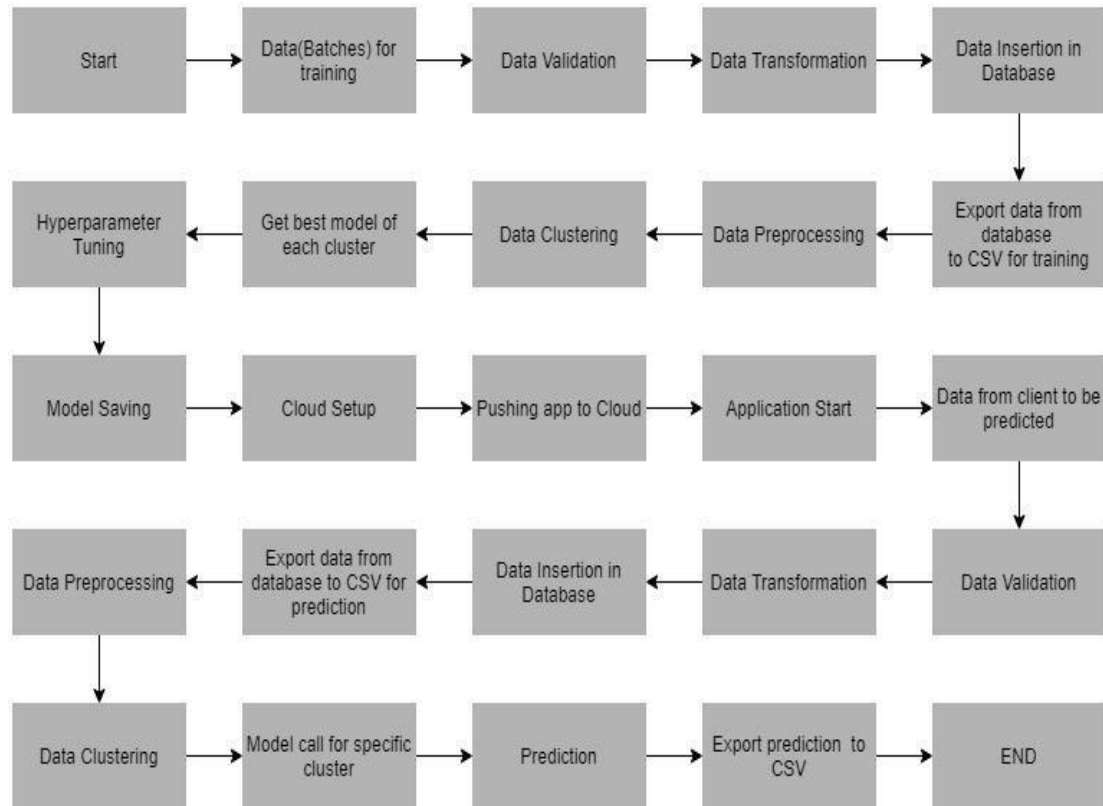


Fig 1.3 System Design for our model

6. IMPLEMENTATION

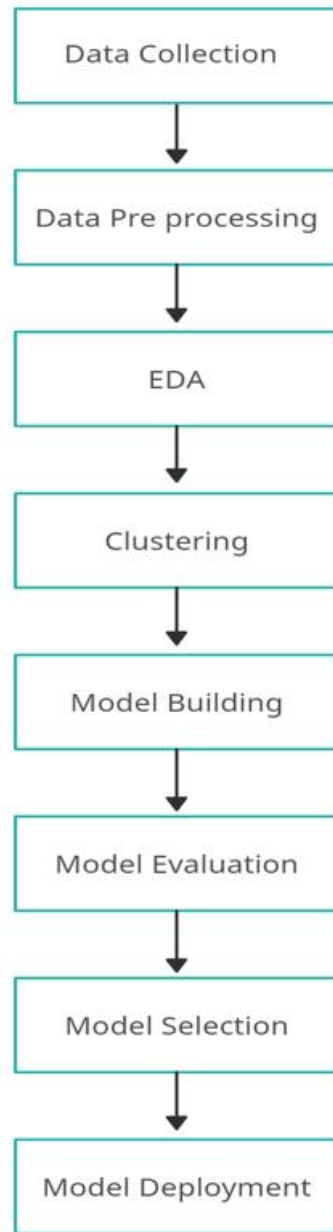


Fig 1.4 Implementation Steps of Machine Learning

6.1 MACHINE LEARNING ALGORITHMS REQUIRED FOR THIS PROJECT

1. XGBoost

2. Support Vector Machine (SVM)

6.1.1 XGBoost

- XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework.
- XGBoost approaches the process of sequential tree building using parallelized implementation.
- XGBoost naturally admits sparse features for inputs by automatically ‘learning’ best missing value depending on training loss and handles different types of patterns in the data more efficiently.
- XGBoost is a popular and efficient open-source implementation of the gradient boosted trees algorithm.
- When using gradient boosting for regression, the weak learners are regression trees, and each regression tree maps an input data point to one of its leafs that contains a continuous score
- The algorithm comes with built-in cross-validation method at each iteration, taking away the need to explicitly program.

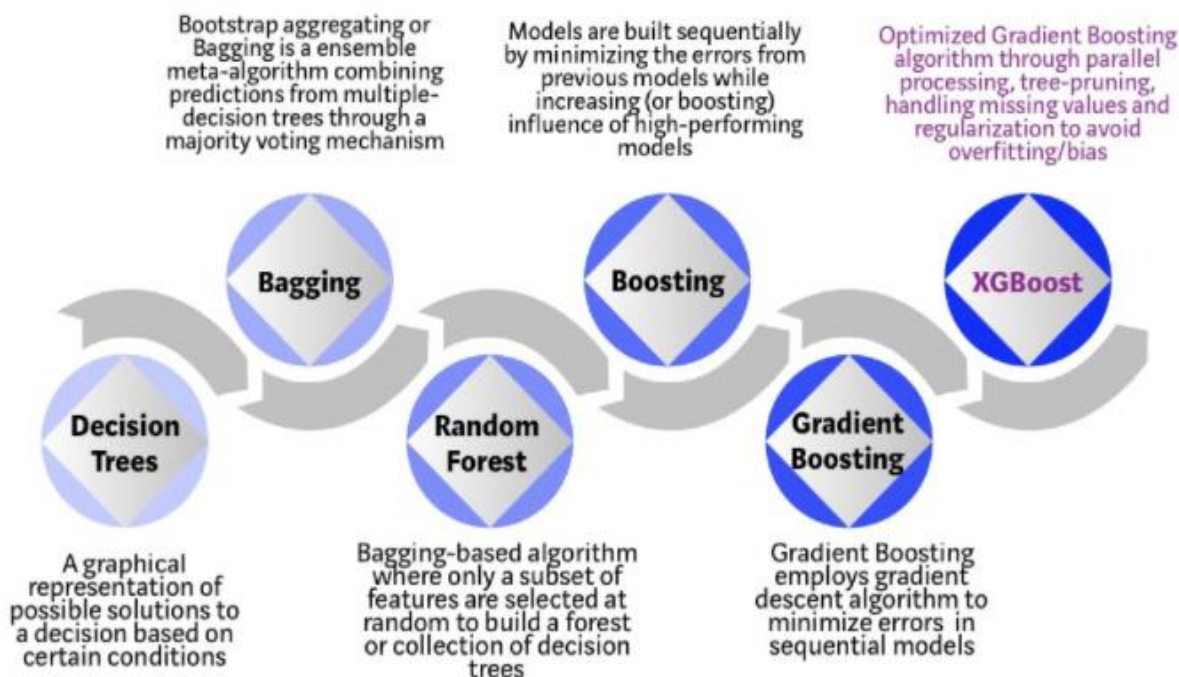


Fig 1.5 XGBoost Steps

Unique Features of XGBoost

XGBoost is a popular implementation of gradient boosting. Let's discuss some features of XGBoost that make it so interesting.

- Regularization: XGBoost has an option to penalize complex models through both L1 and L2 regularization. Regularization helps in preventing overfitting
- Handling sparse data: Missing values or data processing steps like one-hot encoding make data sparse. XGBoost incorporates a sparsity-aware split finding algorithm to handle different types of sparsity patterns in the data
- Weighted quantile sketch: Most existing tree based algorithms can find the split points when the data points are of equal weights (using quantile sketch algorithm). However, they are not equipped to handle weighted data. XGBoost has a distributed weighted quantile sketch algorithm to effectively handle weighted data
- Block structure for parallel learning: For faster computing, XGBoost can make use of multiple cores on the CPU. This is possible because of a block structure in its system design. Data is sorted and stored in in-memory units called blocks. Unlike other algorithms, this enables the data layout to be reused by subsequent iterations, instead of computing it again. This feature also serves useful for steps like split finding and column sub-sampling
- Cache awareness: In XGBoost, non-continuous memory access is required to get the gradient statistics by row index. Hence, XGBoost has been designed to make optimal use of hardware. This is done by allocating internal buffers in each thread, where the gradient statistics can be stored
- Out-of-core computing: This feature optimizes the available disk space and maximizes its usage when handling huge datasets that do not fit into memory.

6.1.2. Support Vector Machine (SVM)

What is Support Vector Machine?

- “Support Vector Machine” (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges.
- However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate.
- Then, we perform classification by finding the hyper-plane that differentiates the two classes very well.

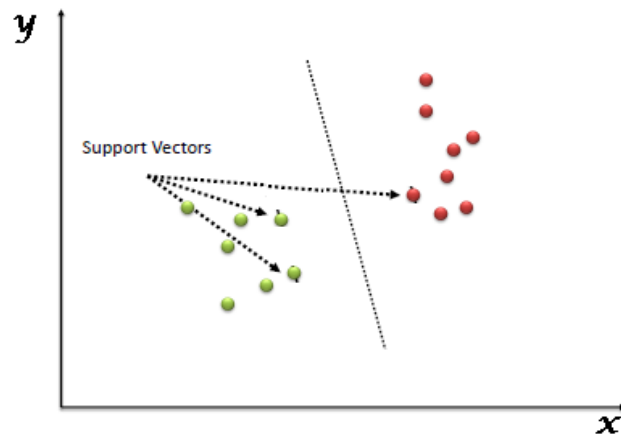
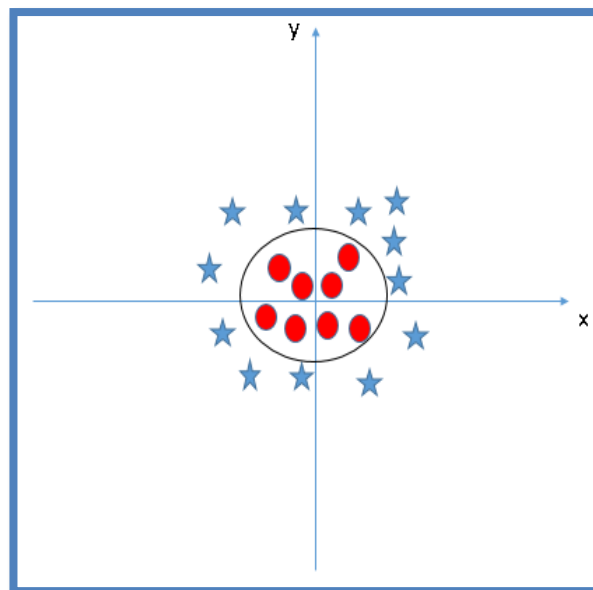


Fig 1.6 Support Vector Machine

- Support Vectors are simply the co-ordinates of individual observation.
- The SVM classifier is a frontier which best segregates the two classes (hyper-plane/ line).
- In the SVM classifier, it is easy to have a linear hyper-plane between these two classes. But another burning question which arises is, should we need to add this feature manually to have a hyper-plane. No, the SVM algorithm has a technique called the kernel trick.
- The SVM kernel is a function that takes low dimensional input space and transforms it to a higher dimensional space i.e., it converts not separable problem to separable problem. It is mostly useful in non-linear separation problem. Simply put, it does some extremely complex data transformations, then finds out the process to separate the data based on the labels or outputs you've defined.
- When we look at the hyper-plane in original input space it looks like a circle:



• **Fig 1.7 Hyper-Plane for SVM**

6.2 DATA COLLECTION

Data collection is defined as the procedure of collecting, measuring and analyzing accurate insights for research using standard validated techniques.

- A researcher can evaluate their hypothesis on the basis of collected data. In most cases, data collection is the primary and most important step for research, irrespective of the field of research.
- The approach of data collection is different for different fields of study, depending on the required information.
- The most critical objective of data collection is ensuring that information-rich and reliable data is collected for statistical analysis so that data-driven decisions can be made for research.

Collecting data for training the ML model is the basic step in the machine learning pipeline. The predictions made by ML systems can only be as good as the data on which they have been trained. Following are some of the problems that can arise in data collection:

- Inaccurate data. The collected data could be unrelated to the problem statement.
- Missing data. Sub-data could be missing. That could take the form of empty values in columns or missing images for some class of prediction.
- Data imbalance. Some classes or categories in the data may have a disproportionately high or low number of corresponding samples. As a result, they risk being under-represented in the model.
- Data bias. Depending on how the data, subjects and labels themselves are chosen, the model could propagate inherent biases on gender, politics, age or region, for example. Data bias is difficult to detect and remove.
- Several techniques can be applied to address those problems:
- Pre-cleaned, freely available datasets. If the problem statement (for example, image classification, object recognition) aligns with a clean, pre-existing, properly formulated dataset, then take advantage of existing, open-source expertise.
- Web crawling and scraping. Automated tools, bots and headless browsers can crawl and scrape websites for data.
- Private data. ML engineers can create their own data. This is helpful when the amount of data required to train the model is small and the problem statement is too specific to generalize over an open-source dataset.
- Custom data. Agencies can create or crowdsource the data for a fee.

6.3 DATA PREPROCESSING IN MACHINE LEARNING

- Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model.
- It is the first and crucial step while creating a machine learning model.
- When creating a machine learning project, it is not always a case that we come across the clean and formatted data.
- Data imputations. Most ML frameworks include methods and APIs for balancing or filling in missing data. Techniques generally include imputing missing values with standard deviation, mean, median and k-nearest neighbors (k-NN) of the data in the given field.
- Oversampling. Bias or imbalance in the dataset can be corrected by generating more observations/samples with methods like repetition, bootstrapping or Synthetic Minority Over-Sampling Technique (SMOTE), and then adding them to the under-represented classes.
- Data integration. Combining multiple datasets to get a large corpus can overcome incompleteness in a single dataset.
- Data normalization. The size of a dataset affects the memory and processing required for iterations during training. Normalization reduces the size by reducing the order and magnitude of data.
- And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So, for this, we use data preprocessing task.

6.3.1 Why do we need Data Pre-processing?

- A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models.
- Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.
- It involves below steps:
 - Getting the dataset
 - Importing libraries
 - Importing datasets
 - Finding Missing Data
 - Encoding Categorical Data
 - Splitting dataset into training and test set
 - Feature scaling

6.4 EDA (Exploratory Data Analysis)

- Exploratory data analysis (EDA) is used by data scientists to analyze and investigate data sets and summarize their main characteristics, often employing data visualization methods.
- It helps determine how best to manipulate data sources to get the answers you need, making it easier for data scientists to discover patterns, spot anomalies, test a hypothesis, or check assumptions.
- EDA is primarily used to see what data can reveal beyond the formal modeling or hypothesis testing task and provides a better understanding of data set variables and the relationships between them.
- It can also help determine if the statistical techniques you are considering for data analysis are appropriate. Originally developed by American mathematician John Tukey in the 1970s, EDA techniques continue to be a widely used method in the data discovery process today.

6.4.1 Exploratory data analysis tools

- Specific statistical functions and techniques you can perform with EDA tools include:
- Clustering and dimension reduction techniques, which help create graphical displays of high-dimensional data containing many variables.
- Univariate visualization of each field in the raw dataset, with summary statistics.
- Bivariate visualizations and summary statistics that allow you to assess the relationship between each variable in the dataset and the target variable you're looking at.
- Multivariate visualizations, for mapping and understanding interactions between different fields in the data.
- K-means Clustering is a clustering method in unsupervised learning where data points are assigned into K groups, i.e., the number of clusters, based on the distance from each group's centroid. The data points closest to a particular centroid will be clustered under the same category. K-means Clustering is commonly used in market segmentation, pattern recognition, and image compression.
- Predictive models, such as linear regression, use statistics and data to predict outcomes.

6.4.2 Types of exploratory data analysis

There are four primary types of EDA:

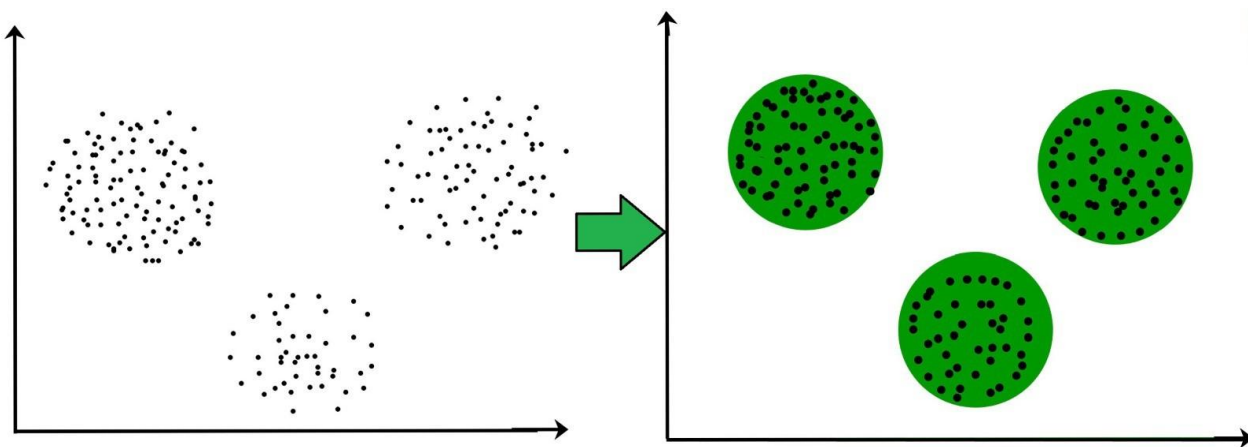
- **Univariate non-graphical.** This is simplest form of data analysis, where the data being analyzed consists of just one variable. Since it's a single variable, it doesn't deal with causes or relationships. The main purpose of univariate analysis is to describe the data and find patterns that exist within it.
- **Univariate graphical.** Non-graphical methods don't provide a full picture of the data. Graphical methods are therefore required. Common types of univariate graphics include:
 - Stem-and-leaf plots, which show all data values and the shape of the distribution.
 - Histograms, a bar plot in which each bar represents the frequency (count) or proportion (count/total count) of cases for a range of values.
 - Box plots, which graphically depict the five-number summary of minimum, first quartile, median, third quartile, and maximum.
- **Multivariate nongraphical:** Multivariate data arises from more than one variable. Multivariate non-graphical EDA techniques generally show the relationship between two or more variables of the data through cross-tabulation or statistics.
- **Multivariate graphical:** Multivariate data uses graphics to display relationships between two or more sets of data. The most used graphic is a grouped bar plot or bar chart with each group representing one level of one of the variables and each bar within a group representing the levels of the other variable.
- Other common types of multivariate graphics include:
 - Scatter plot, which is used to plot data points on a horizontal and a vertical axis to show how much one variable is affected by another.
 - Multivariate chart, which is a graphical representation of the relationships between factors and a response.
 - Run chart, which is a line graph of data plotted over time.
 - Bubble chart, which is a data visualization that displays multiple circles (bubbles) in a two-dimensional plot.
 - Heat map, which is a graphical representation of data where values are depicted by color.

6.5 CLUSTERING

It is basically a type of unsupervised learning method. An unsupervised learning method is a method in which we draw references from datasets consisting of input data without labelled responses. Generally, it is used as a process to find meaningful structure, explanatory underlying processes, generative features, and groupings inherent in a set of examples. Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. It is basically a collection of objects on the basis of similarity and dissimilarity between them.

For ex– The data points in the graph below clustered together can be classified into one single group. We can distinguish the clusters, and we can identify that there are 3 clusters in the below picture.

Fig 1.8 Clusters of Data Points



It is not necessary for clusters to be a spherical. Such as:

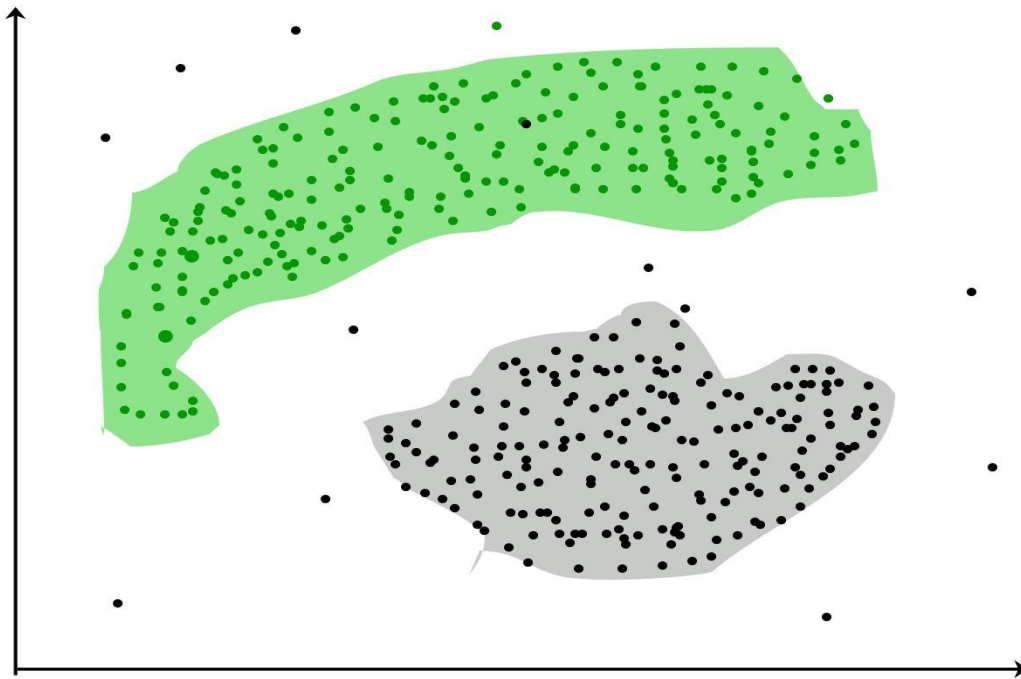


Fig 1.9 DBSCAN

DBSCAN: Density-based Spatial Clustering of Applications with Noise

These data points are clustered by using the basic concept that the data point lies within the given constraint from the cluster center. Various distance methods and techniques are used for calculation of the outliers.

6.6 MODEL BUILDING

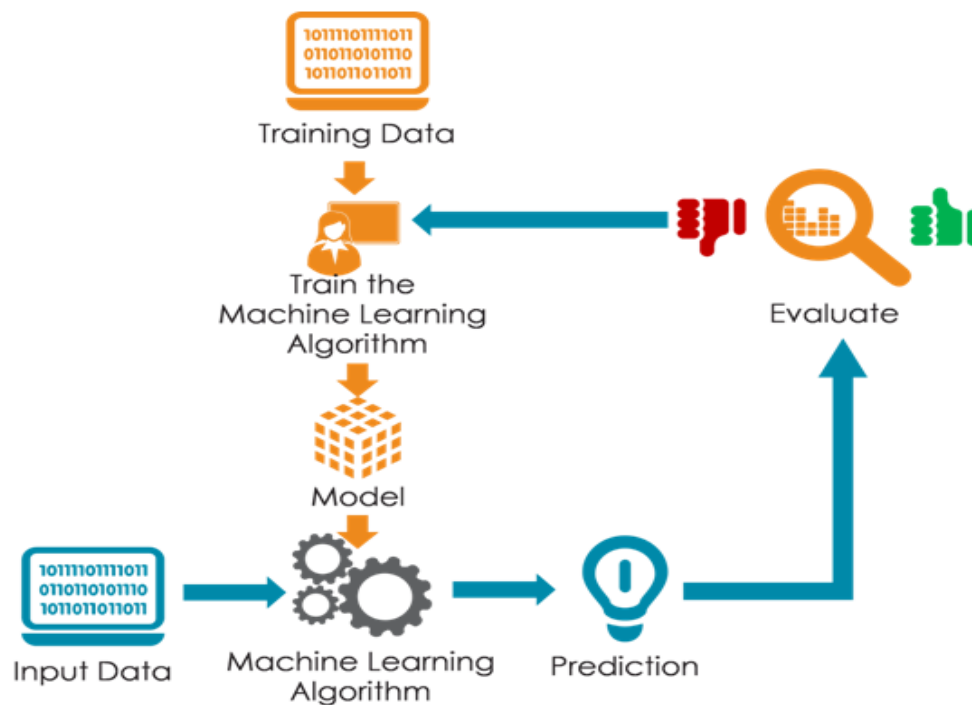


Fig 2.0 Model Building Steps

- Building machine learning models that have the ability to generalize well on future data requires thoughtful consideration of the data at hand and of assumptions about various available training algorithms.
- Ultimate evaluation of a machine learning model's quality requires an appropriate selection and interpretation of assessment criteria.
- Machine learning consists of algorithms that can automate analytical model building.
- Using algorithms that iteratively learn from data, machine learning models facilitate computers to find hidden insights from Big Data without being explicitly programmed where to look.
- This has given rise to a plethora of applications based on Machine learning.

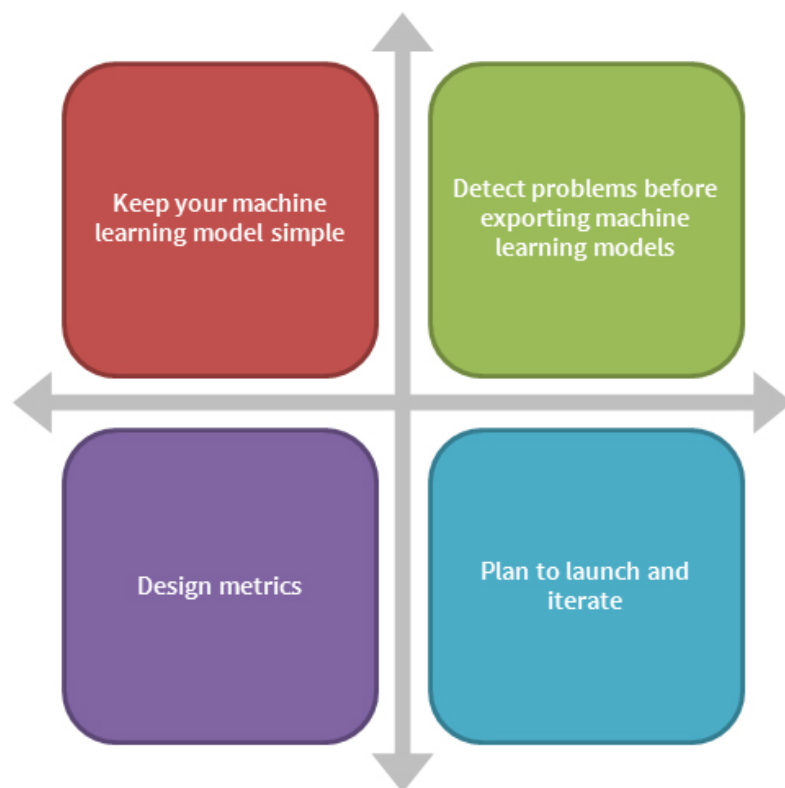
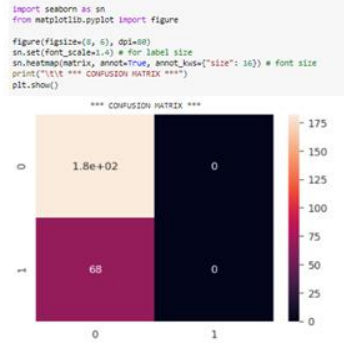


Fig 2.1 Model Building

6.7 MODEL EVALUATION

SVM



```
[466] from sklearn.svm import SVC
sv_classifier=SVC()

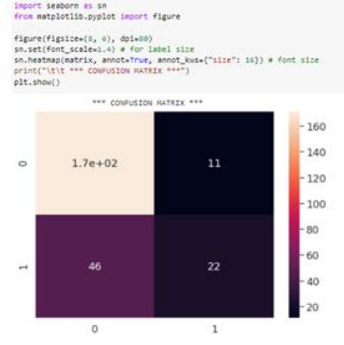
[467] y_pred = sv_classifier.fit(train_x, train_y).predict(test_x)

[468] from sklearn.metrics import accuracy_score

[469] sc=accuracy_score(test_y,y_pred)
print('SVM ACCURACY:',sc)

SVM ACCURACY: 0.728
```

XGB



```
[396] from xgboost import XGBClassifier

[397] xgb=XGBClassifier()

[398] y_pred = xgb.fit(train_x, train_y).predict(test_x)

[409] ac2=accuracy_score(test_y,y_pred)
print('XGB ACCURACY:',ac2)

XGB ACCURACY: 0.772
```

- Model evaluation aims to estimate the generalization accuracy of a model on future (unseen/out-of-sample) data.

Fig 2.2 Model Evaluation

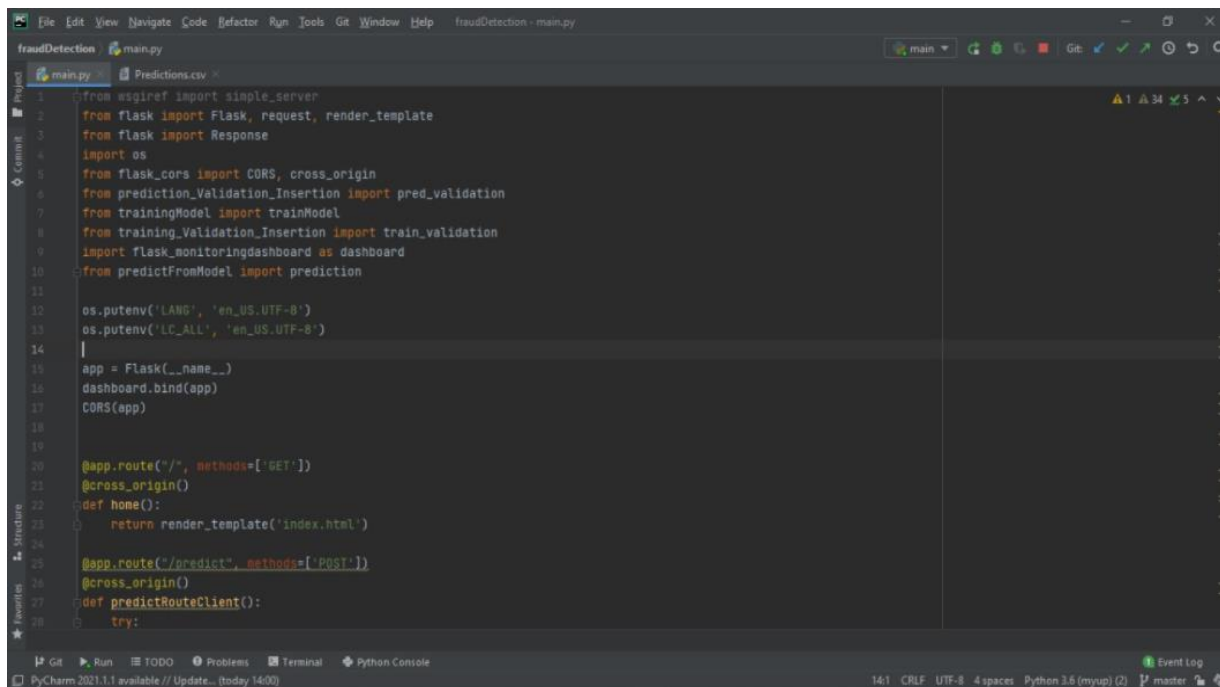


Fig 2.3 Snapshot of Main.py File

Table 1.2 Measure of SVM and XGBoost

- Methods for evaluating a model's performance are divided into 2 categories: namely, holdout and Cross-validation. Both methods use a test set (i.e data not seen by the model) to evaluate model performance.
- It's not recommended to use the data we used to build the model to evaluate it. This is because our model will simply remember the whole training set, and will therefore always predict the correct label for any point in the training set. This is known as overfitting.

6.7.1 Holdout

- The purpose of holdout evaluation is to test a model on different data than it was trained on. This provides an unbiased estimate of learning performance.
- In this method, the dataset is randomly divided into three subsets:
- Training set is a subset of the dataset used to build predictive models.
- Validation set is a subset of the dataset used to assess the performance of the model built in the training phase. It provides a test platform for fine-tuning a model's parameters and selecting the best performing model. Not all modeling algorithms need a validation set.
- Test set, or unseen data, is a subset of the dataset used to assess the likely future performance of a model. If a model fits to the training set much better than it fits the test set, overfitting is probably the cause.
- The holdout approach is useful because of its speed, simplicity, and flexibility. However, this technique is often associated with high variability since differences in the training and test dataset can result in meaningful differences in the estimate of accuracy.

6.7.2 Cross-Validation

- Cross-Validation is a technique that involves partitioning the original observation dataset into a training set, used to train the model, and an independent set used to evaluate the analysis.
- The most common cross-validation technique is k-fold cross-validation, where the original dataset is partitioned into k equal size subsamples, called folds. The k is a user-specified number, usually with 5 or 10 as its preferred value. This is repeated k times, such that each time, one of the k subsets is used as the test set/validation set and the other k-1 subsets are put together to form a training set. The error estimation is averaged over all k trials to get the total effectiveness of our model.
- For instance, when performing five-fold cross-validation, the data is first partitioned into 5 parts of (approximately) equal size. A sequence of models is trained. The first model is trained using the

first fold as the test set, and the remaining folds are used as the training set. This is repeated for each of these 5 splits of the data and the estimation of accuracy is averaged over all 5 trials to get the total effectiveness of our model.

- As can be seen, every data point gets to be in a test set exactly once and gets to be in a training set $k-1$ time. This significantly reduces bias, as we're using most of the data for fitting, and it also significantly reduces variance, as most of the data is also being used in the test set. Interchanging the training and test sets also adds to the effectiveness of this method.

6.7.4 Model Evaluation Metrics

Model evaluation metrics are required to quantify model performance. The choice of evaluation metrics depends on a given machine learning task (such as classification, regression, ranking, clustering, topic modeling, among others). Some metrics, such as precision-recall, are useful for multiple tasks. Supervised learning tasks such as classification and regression constitute a majority of machine learning applications. In this article, we focus on metrics for these two supervised learning models.

6.8 MODEL SELECTION

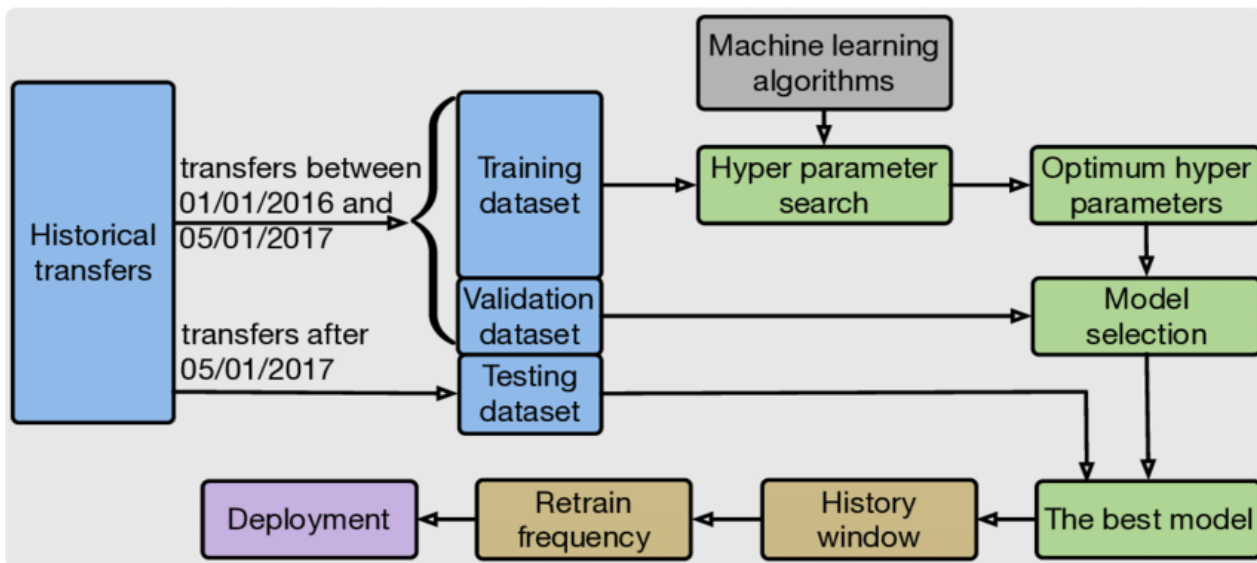


Fig 2.4 Model Selection Process

- Model selection is the process of selecting one final machine learning model from among a collection of candidate machine learning models for a training dataset.
- Model selection is a process that can be applied both across different types of models (e.g., logistic regression, SVM, KNN, etc.) and across models of the same type configured with different model hyperparameters (e.g., different kernels in an SVM).

- For example, we may have a dataset for which we are interested in developing a classification or regression predictive model. We do not know beforehand as to which model will perform best on this problem, as it is unknowable. Therefore, we fit and evaluate a suite of different models on the problem.
- Model selection is the process of choosing one of the models as the final model that addresses the problem.
- Model selection is different from model assessment.
- For example, we evaluate or assess candidate models in order to choose the best one, and this is model selection. Whereas once a model is chosen, it can be evaluated in order to communicate how well it is expected to perform in general; this is model assessment.

6.8.1 Considerations for Model Selection

- Fitting models is relatively straightforward, although selecting among them is the true challenge of applied machine learning.
- Firstly, we need to get over the idea of a “best” model.
- All models have some predictive error, given the statistical noise in the data, the incompleteness of the data sample, and the limitations of each different model type. Therefore, the notion of a perfect or best model is not useful. Instead, we must seek a model that is “good enough.”

6.8.2 Model Selection Techniques

- The best approach to model selection requires “sufficient” data, which may be nearly infinite depending on the complexity of the problem.
- In this ideal situation, we would split the data into training, validation, and test sets, then fit candidate models on the training set, evaluate and select them on the validation set, and report the performance of the final model on the test set.

6.9 MODEL DEPLOYEMENT

- Deployment is the method by which you integrate a machine learning model into an existing production environment to make practical business decisions based on data.
- It is one of the last stages in the machine learning life cycle and can be one of the most cumbersome. Often, an organization’s IT systems are incompatible with traditional model-building

languages, forcing data scientists and programmers to spend valuable time and brainpower rewriting them.

- In order to start using a model for practical decision-making, it needs to be effectively deployed into production. If you cannot reliably get practical insights from your model, then the impact of the model is severely limited.
- Model deployment is one of the most difficult processes of gaining value from machine learning. It requires coordination between data scientists, IT teams, software developers, and business professionals to ensure the model works reliably in the organization's production environment.
- This presents a major challenge because there is often a discrepancy between the programming language in which a machine learning model is written and the languages your production system can understand, and re-coding the model can extend the project timeline by weeks or months.
- In order to get the most value out of machine learning models, it is important to seamlessly deploy them into production so a business can start using them to make practical decisions.

6.9.1 Heroku App Creation and Deployment

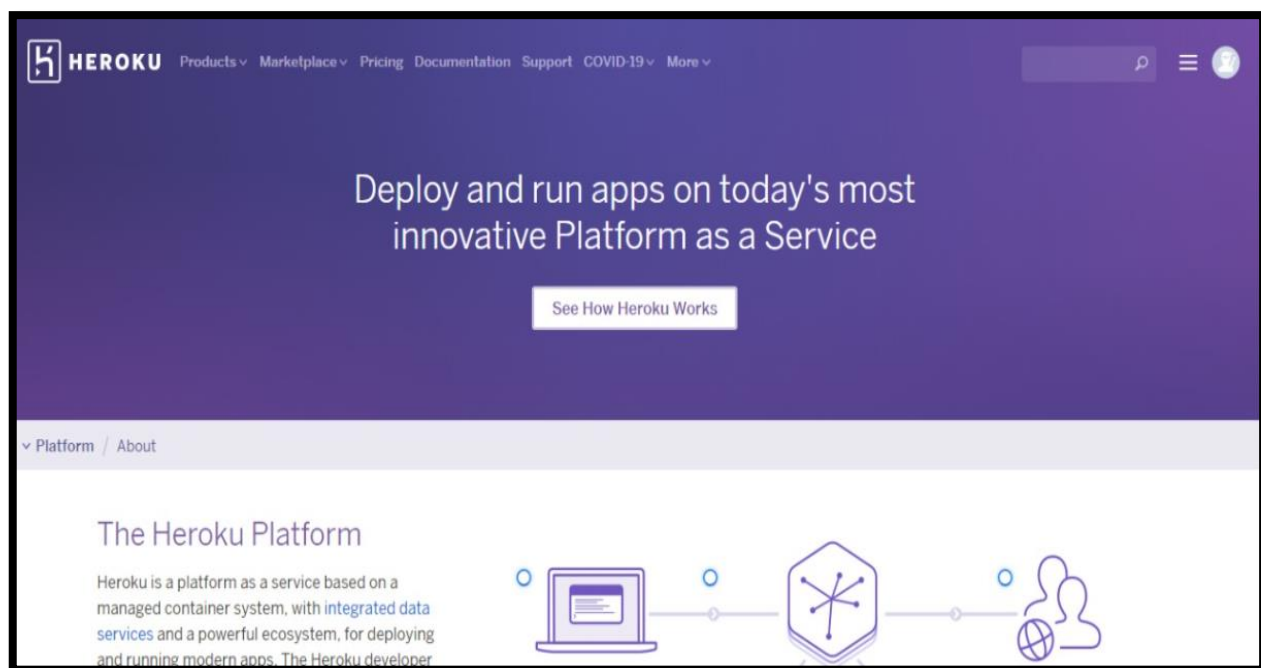


Fig 2.5 Heroku Platform

- a. After installing the Heroku CLI, Open a command prompt window and navigate to your 'reviewScrapper' folder.

b. Type the command 'heroku login' to login to your heroku account as shown below:

```
Thanuj Kumar.S@THANUJ MINGW64 /f/THIS/code/fraudDetection (master)
$ heroku login
heroku: Press any key to open up the browser to login or q to exit:
Opening browser to https://cli-auth.heroku.com/auth/cli/browser/62ee259a-dc3b-4565-9595-b987c44cba14?
hzoxoDaJrS7ikctag
heroku: Waiting for login...
Logging in... done
Logged in as thanujkumars98@gmail.com
```

Fig 2.6 Heroku Login

c. After logging in to Heroku, enter the command 'heroku create' to create a heroku app. It will give you the URL of your Heroku app after successful creation.

d. Before deploying the code to the Heroku cloud, we need to commit the changes to the local git repository.

e. Type the command 'git init' to initialize a local git repository as shown below:

```
Thanuj Kumar.S@THANUJ MINGW64 /f/THIS/code
$ cd fraudDetection

Thanuj Kumar.S@THANUJ MINGW64 /f/THIS/code/fraudDetection
$ dir
DataTransform_Training          Prediction_Logs                 Training_Logs
DataTransformation_Prediction  Prediction_Output_File         Training_Raw_data_validation
DataTypeValidation_Insertion_Prediction Prediction_Raw_Data_Validation Training_Raw_files_validated
DataTypeValidation_Insertion_Training Prediction_Raw_Files_Validated __pycache__
EDA                             Procfile                      application_logging
PredictionArchivedBadData       TrainingArchiveBadData         best_model_finder
Prediction_Batch_files          Training_Batch_Files           data
Prediction_Database             Training_Database              data_ingestion
Prediction_FileFromDB           Training_FileFromDB            data_preprocessing

Thanuj Kumar.S@THANUJ MINGW64 /f/THIS/code/fraudDetection
$ git init
Initialized empty Git repository in F:/THIS/code/fraudDetection/.git/

Thanuj Kumar.S@THANUJ MINGW64 /f/THIS/code/fraudDetection (master)
$ git status
```

Fig 2.7 Initialize git Repository

f. Enter the command 'git status' to see the uncommitted changes

g. Enter the command 'git add .' to add the uncommitted changes to the local repository.

h. Enter the command 'git commit -am "make it better"' to commit the changes to the local repository.

i. Enter the command 'git push heroku master' to push the code to the heroku cloud.

```

Thanuj Kumar-S@THANUJ MINGW64 /f/THIS/code/FraudDetection (master)
$ heroku create
Creating app... done, infinite-citadel-54923
https://infinite-citadel-54923.herokuapp.com/ | https://git.heroku.com/infinite-citadel-54923.git

Thanuj Kumar-S@THANUJ MINGW64 /f/THIS/code/FraudDetection (master)
$ git remote -v
heroku https://git.heroku.com/infinite-citadel-54923.git (fetch)
heroku https://git.heroku.com/infinite-citadel-54923.git (push)

Thanuj Kumar-S@THANUJ MINGW64 /f/THIS/code/FraudDetection (master)
$ git push heroku master
Enumerating objects: 155, done.
Counting objects: 100% (155/155), done.
Delta compression using up to 4 threads
Compressing objects: 100% (144/144), done.
Writing objects: 100% (155/155), 847.92 KiB | 3.61 MiB/s, done.
Total 155 (delta 30), reused 0 (delta 0), pack-reused 0
remote: Compressing source files... done.
remote: Building source:
remote:
remote: ----> Building on the Heroku-20 stack
remote: ----> Determining which buildpack to use for this app
remote: ----> Python app detected
remote: ----> No Python version was specified. Using the buildpack default: python-3.9.5
remote: ----> To use a different version, see: https://devcenter.heroku.com/articles/python-runtimes
remote: ----> Installing python-3.9.5
remote: ----> Installing pip 20.2.4, setuptools 47.1.1 and wheel 0.36.2
remote: ----> Installing SQLite3
remote: ----> Installing requirements with pip
remote: Collecting APScheduler==3.6.3
remote:   Downloading APScheduler-3.6.3-py2.py3-none-any.whl (58 kB)
remote: Collecting attrs==19.3.0
remote:   Downloading attrs-19.3.0-py2.py3-none-any.whl (39 kB)
remote: Collecting certifi==2019.11.28
remote:   Downloading certifi-2019.11.28-py2.py3-none-any.whl (156 kB)
remote: Collecting Click==7.0
remote:   Downloading Click-7.0-py2.py3-none-any.whl (81 kB)
remote: Collecting colorhash==1.0.2
remote:   Downloading colorhash-1.0.2-py2.py3-none-any.whl (6.0 kB)
remote: Collecting configparser==4.0.2
remote:   Downloading configparser-4.0.2-py2.py3-none-any.whl (22 kB)
remote: Collecting cyclical==0.10.0
remote:   Downloading cyclical-0.10.0-py2.py3-none-any.whl (6.5 kB)
remote: Collecting Flask==1.1.1
remote:   Downloading Flask-1.1.1-py2.py3-none-any.whl (94 kB)
remote: Collecting Flask-Cors==3.0.8
remote:   Downloading Flask-Cors-3.0.8-py2.py3-none-any.whl (14 kB)
remote: Collecting Flask-MonitoringDashboard==3.0.6

```

Fig 2.8 Deploying the App on Heroku Platform

j. After deployment, heroku gives you the URL to hit the web API.

k. Once your application is deployed successfully, enter the command 'heroku logs --tail' to see the logs.

7. TESTING

Model Evaluation and Selection

Model Evaluation is a process of evaluating the models based on various performance measuring parameters. Evaluation of the model provides a clear picture of the model's efficiency and helps to select the best model for performing prediction.

In this study, as we are using two algorithms, "SVM" and "XGBoost", so after building the models, these models are undergone a model evaluation phase. Scikit-learn python libraries are used extensively in this study for practical illustration.

We use various performance measuring parameters like the Confusion matrix which further leads for calculations of Accuracy, Precision, Recall, and F1 Score. Also, we are using Area under the curve (AUC).

		PREDICTIVE VALUES	
		POSITIVE (1)	NEGATIVE (0)
ACTUAL VALUES	POSITIVE (1)	TP	FN
	NEGATIVE (0)	FP	TN

Fig 2.9 is the skeleton view of confusion matrix, row wise it's the actual values Column wise it's predicted values, with respect to positive and negative.
 (1,1) True Positive(TP), (1,0) False Negative(FN), (0,1) False Positive(FP), (0,0) True Negative(TN)

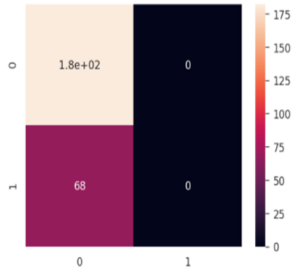


Fig 3.0 Confusion Matrix of SVM

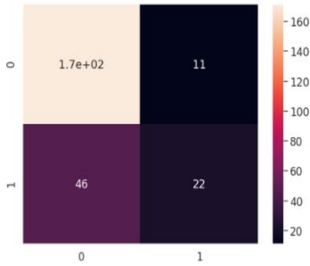


Fig 3.1 Confusion Matrix of XGB

Measure	Formula	SVM	XGBoost
Accuracy	$\frac{TP + TN}{TP + TN + FN + FP}$	0.728	0.772
Precision	$\frac{TP}{TP + FP}$	1.000	0.939
Recall	$\frac{TP}{TP + FN}$	0.000	0.323
F1-score	$\frac{2 * Prec * Rec}{(Prec + Rec)}$	0.842	0.857
Error rate	$\frac{FP + FN}{TP + TN + FN + FP}$	0.272	0.228

Table 1.3 Model Evaluation Formulas

We are using the confusion matrix python library to obtain the confusion matrices of both the

algorithms, (Fig1.2 and Fig 1.3). Based on the confusion matrices, we calculated the other parameters like accuracy, precision, recall, f1 score, and error rate.

The accuracy of XG Boost(77.2%) is higher compared to SVM's(72.8%), though the Precision of SVM is 100% its Recall rate is 0.00% which shows an inconsistency, unlike XG Boost. F1 score is one of the great performance measurements in this XG Boost is shown more rate(85.7%) than SVM's(84.2%). The error rate of the SVM shows up to 27.2% and XG Boost is 22.8% which is less than SVM. In the above measurement, XG Boost seems to be more efficient than SVM.

Although evaluation is not yet finalized based on only these parameters because as we can see in the confusion matrix, data is not distributed uniformly which shows that our dataset is imbalanced therefore these above evaluation parameters are not enough to judge the model.

Therefore we use parameters such as True Positive Rate(TPR), True Negative Rate(TNR), False Positive Rate(FPR), False Negative Rate(FNR).

Measure	Formula	SVM	XGBoost
True Positive Rate	TP/P	1.000	0.939
True Negative Rate	TN/N	0.000	0.323
False Positive Rate	FP/P	0.370	0.252
False Negative Rate	FN/N	0.000	0.161

Table 1.4 Model Evaluation Results

In Ideal scenarios, the True Positive Rate(TPR) and True Negative Rate(TNR) should be high, and False Positive Rate(FPR) and False Negative Rate(FNR) should be low.

In table 1.4, we can see that the TPR of SVM is high(100%) than XG Boost(93.9%) but the TNR of SVM is 0.00% which is less than the XG boost who is bearing 32.3%.

FPR of XG Boost(23.2%) is less(as recommended) than SVM(37.0%). But FNR of SVM(0.00%) is less than XG Boost(16.1%) in a small margin. In the above illustration, XGBoost seems to be efficient and consistent in all cases compared to SVM. We also using Area under Curve(AUC), AUC represents the degree of separability. That means it tells that how much the model is capable of distinguishing the

classes in our case it's fraud and Not fraud. Higher the AUC the better the model is at predicting fraud as fraud and Not fraud as Not fraud. An exceptional model has AUC near to 100%, and an ideal model has above 50%. But when the model is less than 50% or 50%, that means the model is incapable of distinguishing the classes.

Measure	Full form	SVM	XGBoost
AUC	Area under curve	0.500	0.631

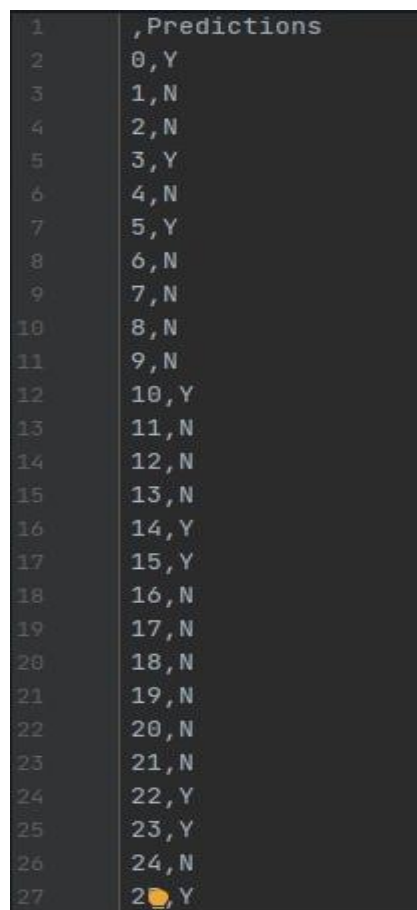
Table 1.5 AUC Evaluation

In above table 1.5, we observe that the AUC rate of XG Boost is 63.1% which is ideal as it is greater than 50%, but whereas SVM is exactly 50%, which shows that in this case, SVM is incapable of distinguishing the classes so SVM is not an ideal model to select.

By the process of evaluating, we yield a significant outcome on selecting the best model among SVM and XG Boost, considering the evaluation results we are selecting XG Boost which is shown more efficient than SVM.

8. RESULT

- Here, we present the experimental comparison between the performance score of two algorithms that are used to build ML models to perform prediction on our dataset. Using evaluation technique, we found that the accuracy of XG Boost (77.2%) is higher compared to SVM's (72.8%) though we can't judge a model just by accuracy as it showed an imbalanced distribution of data in the confusion matrix, we performed various model evaluating parameters which briefly explained in Section 2.6. In addition to it, we also used Area under Curve (AUC), AUC represents the degree of separability between classes. In which, the AUC rate of XG Boost is 63.1% which is ideal as it is greater than



1	,Predictions
2	0,Y
3	1,N
4	2,N
5	3,Y
6	4,N
7	5,Y
8	6,N
9	7,N
10	8,N
11	9,N
12	10,Y
13	11,N
14	12,N
15	13,N
16	14,Y
17	15,Y
18	16,N
19	17,N
20	18,N
21	19,N
22	20,N
23	21,N
24	22,Y
25	23,Y
26	24,N
27	25,Y

Fig 3.2 Prediction Results

50%, but whereas SVM is exactly 50%.

- With the evaluation process result, we select the best model which addresses our problem statement and efficient enough to perform the prediction.

- In the model selection phase of this study, we have chosen XG Boost over SVM based on the performance in the evaluation phase.

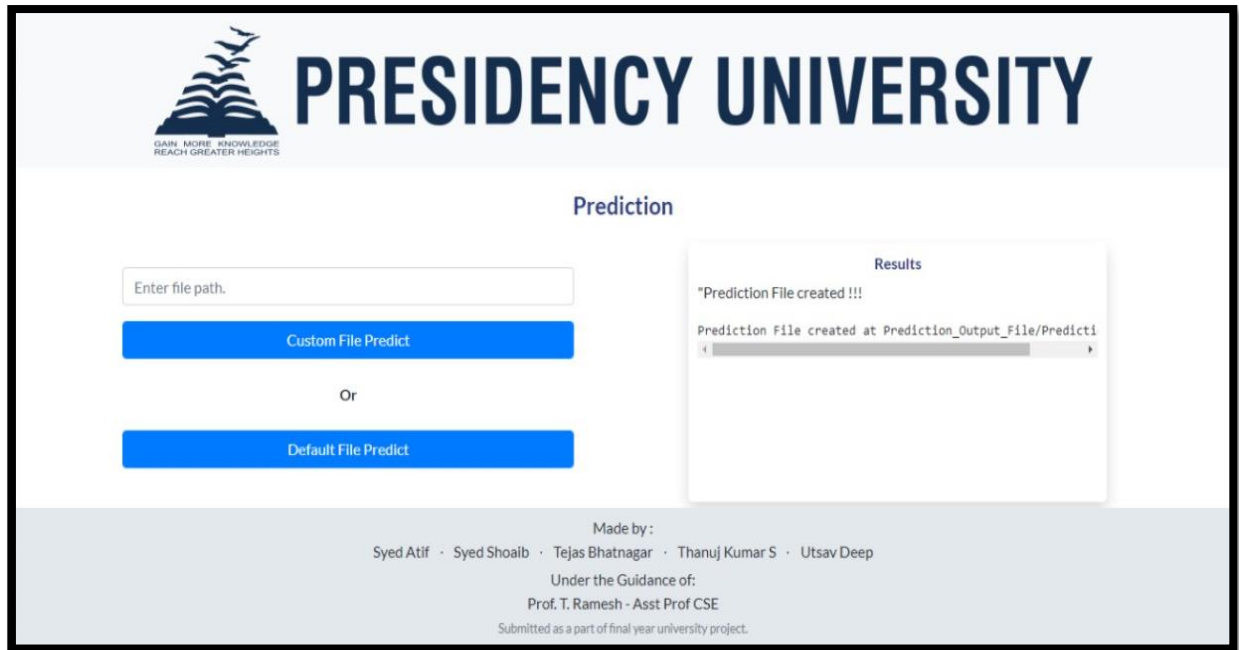


Fig 3.3 Working Web App

- And we have deployed our model as a Web application using the Flask web framework in the Heroku cloud platform.

9. CONCLUSION

- In this study, we imported the relevant dataset which correlated to the problem statement then we used Machine learning techniques like Support Vector Machine (SVM) and XG Boost to build the predictive models.
- These models were undergone a model evaluation process to estimate accuracy, precision, recall, f1 score, error rate, and also AUC rate. By comparing the models on basis of the result of model evaluation, XG Boost outperformed SVM. Then deployment of the selected model is done by wrapping it as a web application by flask web framework, then later application is pushed to the cloud using Heroku cloud platform.
- Now this proposed system is ready to predict whether the claimed insurance is “Fraud “or “Not Fraud” .

10. REFERENCES

1. Ghorbani, A. and Farzai, S. (2018). Fraud Detection in Automobile Insurance using a Data Mining Based Approach.
[Online] Available at:
http://www.aeuso.org/includes/files/articles/Vol8_Iss27_37643771_Fraud_Detection_in_Automobile_Insur.pdf
2. Bahnsen, M. (2016). Feature Engineering Strategies for Credit Card Fraud Detection. [Online] Available at:
https://www.researchgate.net/publication/289585253_Feature_Engineering_Strategies_for_Credit_Card_Fraud_Detection
3. Komal Patil (2018) A Survey on Machine Learning Techniques for Insurance Fraud Prediction [Online] Available at:
https://www.researchgate.net/publication/329448645_A_Survey_on_Machine_Learning_Techniques_for_Insurance_Fraud_Prediction
4. Mali, S. and Salunkhe, U. (2016). Classifier Ensemble Design for Imbalanced Data Classification: A Hybrid Approach
[Online] Available at: <https://cyberleninka.org/article/n/604159>
5. Bodaghi, A. and Teimourpour, B. (2018). Automobile Insurance Fraud Detection Using Social Network Analysis.
[Online] Available at:
https://link.springer.com/chapter/10.1007%2F978-3-319-95810-1_2
6. Richard A Bauder, Raquel Rosa & Taghi Khoshgoftaar (2017). Identifying Medicare Provider Fraud with Unsupervised Machine Learning
[Online] Available at:
https://www.researchgate.net/publication/326557545_Identifying_Medicare_Provider_Fraud_with_Unsupervised_Machine_Learning

7. Santosh Kumar Majhi, (2019). Fuzzy clustering algorithm based on modified whale optimization algorithm for automobile insurance fraud detection
[Online] Available at:
https://link.springer.com/article/10.1007/s12065-019-00260-3#auth-Santosh_Kumar-Majhi

8. Khaled Gubran Al-Hashedi (2019). Financial fraud detection applying data mining techniques: A comprehensive review from 2009 to 2019 [Online] Available at:
<https://www.sciencedirect.com/science/article/abs/pii/S1574013721000423#!>

9. Admel Husejinovic (2020). Credit card fraud detection using naive Bayesian and C4.5 decision tree classifiers
[Online] Available at:
https://www.researchgate.net/publication/338655407_Credit_card_fraud_detection_using_naive_Bayesian_and_C45_decision_tree_classifiers

10. Imane Sadgali, Nawal Sael & Faouzia Benabbou (2019). Fraud detection in credit card transaction using neural networks
[Online] Available at:
<https://dl.acm.org/doi/10.1145/3368756.3369082>

11. A. Maria Nancya, G. Senthil Kumar, S. Veena, N. A. S Vinoth, and Moinak Bandyopadhyay (2020). Fraud detection in credit card transaction using hybrid model
[Online] Available at:
<https://www.scitation.org/doi/10.1063/1.5131111>

12. Dejan Varmedja, Mirjana Karanovic (2019). Credit Card Fraud Detection - Machine Learning methods
[Online] Available at:
<https://www.researchgate.net/publication/338655407>

13. Lakshika Sammani Chandradeva (2020). Monetary Transaction Fraud Detection System Based on Machine Learning Strategies
[Online] Available at:
<https://www.researchgate.net/publication/338655407>

14. I.Sadgalian, Saelaf & Benabboua (2019). Performance of machine learning techniques in the detection of financial frauds
[Online] Available at:
<https://www.sciencedirect.com/science/article/pii/S1877050919300079>