TRUST MANAGEMENT USING MACHINE LEARNING

**Gaurav Singh, Himanshu Souda, Ramanpreet Kaur**

**Jb Knowledge ParkFaridadbad, Haryana, India**[**gaurav31singh@gmail.com**](mailto:gaurav31singh@gmail.com)**,** [**ramanpreet.official9911@gmail.com**](mailto:ramanpreet.official9911@gmail.com)**,** [**himanshu.scrush01@gmail.com**](mailto:himanshu.scrush01@gmail.com)

**Abstract**— Trust management is otherwise called trust worthiness, dependable client commitment management. It is utilized for building up the methodologies in the set up trust. In this day and age it is difficult to have a confidence on somebody, and in IT Sector additionally, there is significant trust issue since information and its administrations are facilitated by a third-gathering association which makes a sort of dark divider between the genuine client and the principle information source, now and then it is difficult to see that what is really occurring behind the dividers. So we're making a trust management base model which will tackle trust issues Trust is a significant fixing working with dependable interactions1 among self-sufficient members without worldwide coordination in different huge scope frameworks including online business, disseminated and distributed frameworks, multi-specialist frameworks, and dynamic community-oriented frameworks. Because of the enormous scope and transparency of these frameworks, a specialist is frequently needed to interface with different specialists with which it has not many or no common past cooperation. To survey the danger of such cooperation and to decide if an obscure specialist is trustworthy, a productive trust component is vital

1. **INTRODUCTION**

There are several factors that make contributions to the success or failure of a business, and customer pleasure is normally listed as one of the most crucial ones. Customer pleasure and an great recognition are a matter of life and lack of existence for masses businesses. It is crucial to take customer reviews into attention significantly and address their issues concisely. However, on-line reviews additionally may be abused with the useful resource of the use of adversaries for one in every of a type reasons. It is therefore crucial to distinguish amongst real and dubious reviews. The automatic detection of fake on-line reviews is an inherent instance of a smooth binary magnificence problem. As a result, conventional tool learning-based totally completely magnificence or smooth statistical-based totally completely strategies can be applicable to this problem. Given the natural language-based totally completely nature of reviews and the hidden capabilities that might be latent for particular modeling, the accuracy may also vary.

**2. Problem Statement**

In this day and age, it is extremely difficult to decide whether the news we come across is real or not. There are very few options to check the authenticity and all of them are sophisticated and not accessible to the average person. There is an acute need for a web-based fact-checking platform that harnesses the power of Machine Learning to provide us with that opportunity. Fake News Detector 8

**3. Motivation**

Social media facilitates the creation and sharing of information that uses computer-mediated technologies. This media changed the way groups of people interact and communicate. It allows low cost, simple access and fast dissemination of information to them. The majority of people search and consume news from social media rather than traditional news organizations these days. On one side, where social media have become a powerful source of information and bringing people together, on the other side it also 1 put a negative impact on society. Look at some examples herewith; Facebook Inc’s popular messaging service, WhatsApp became a political battle-platform in Brazil’s election. False rumours, manipulated photos, de-contextualized videos, and audio jokes were used for campaigning. These kinds of stuff went viral on the digital platform without monitoring their origin or reach. A nationwide block on major social media and messaging sites including Facebook and Instagram was done in Sri Lanka after multiple terrorist attacks in the year 2019. The government claimed that “false news reports” were circulating online. This is evident in the challenges the world's most powerful tech companies face in reducing the spread of misinformation. Such examples show that Social Media enables the widespread use of “fake news” as well. The news Fake News Detector 9 disseminated on social media platforms may be of low quality carrying misleading information intentionally. This sacrifices the credibility of the information. Millions of news articles are being circulated every day on the Internet – how one can trust which is real and which is fake? Thus incredible or fake news is one of the biggest challenges in our digitally connected world. Fake news detection on social media has recently become an emerging research domain. The domain focuses on dealing with the sensitive issue of preventing the spread of fake news on social media. Fake news identification on social media faces several challenges. Firstly, it is difficult to collect fake news data. Furthermore, it is difficult to label fake news manually. Since they are intentionally written to mislead readers, it is difficult to detect them simply based on news content. Furthermore, Facebook, Whatsapp, and Twitter are closed messaging apps. The misinformation disseminated by trusted news outlets or their friends and family is therefore difficult to be considered as fake. It is not easy to verify the credibility of newly emerging and time-bound news as they are not sufficient to train the application dataset. Significant approaches to differentiate credible users, extract useful news features and develop authentic information dissemination systems are some useful domains of research and need further investigations. If we can’t control the spread of fake news, the trust in the system will collapse. There will be widespread Fake News Detector 10 distrust among people. There will be nothing left that can be objectively used. It means the destruction of political and social coherence. We wanted to build some sort of web-based system that can fight this nightmare scenario. And we made some significant progress towards that goal. Fake News Detector

**4. Machine Learning with Python**

Machine Learning is the ability of the computer to learn without being explicitly programmed. In layman’s terms, it can be described as automating the learning process of computers based on their experiences without any human assistance. Machine learning is actively used in our daily life and perhaps in more places than one would expect.

Machine Learning (ML) is that field of computer science with the help of which computer systems can provide sense to data in much the same way as human beings do.

In simple words, ML is a type of artificial intelligence that extract patterns out of raw data by using an algorithm or method. The main focus of ML is to allow computer systems learn from experience without being explicitly programmed or human intervention.

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems. On the other side, AI is still in its initial stage and haven’t surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, “to make decisions, based on data, with efficiency and scale”.

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programing logic, in the problems that cannot be programmed inherently. The fact is that we can’t do without human intelligence, but other aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

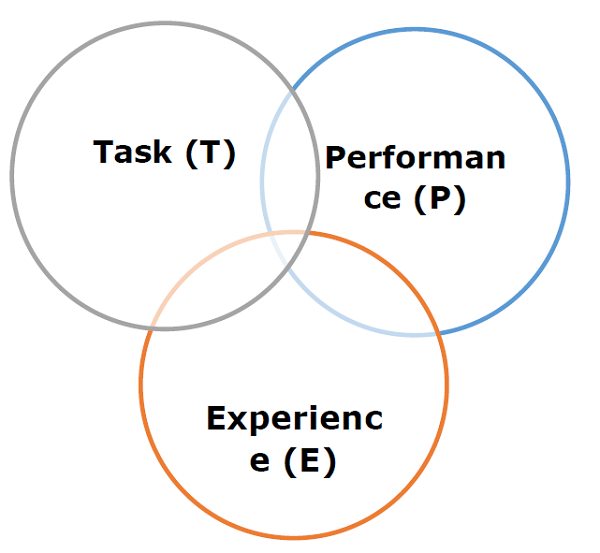
A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.”

The above definition is basically focusing on three parameters, also the main components of any learning algorithm, namely Task(T), Performance(P) and experience (E). In this context, we can simplify this definition as –

ML is a field of AI consisting of learning algorithms that −

* Improve their performance (P)
* At executing some task (T)
* Over time with experience (E)

Based on the above, the following diagram represents a Machine Learning Model −



Let us discuss them more in detail now −

Task(T)

From the perspective of problem, we may define the task T as the real-world problem to be solved. The problem can be anything like finding best house price in a specific location or to find best marketing strategy etc. On the other hand, if we talk about machine learning, the definition of task is different because it is difficult to solve ML based tasks by conventional programming approach.

A task T is said to be a ML based task when it is based on the process and the system must follow for operating on data points. The examples of ML based tasks are Classification, Regression, Structured annotation, Clustering, Transcription etc.

Experience (E)

As name suggests, it is the knowledge gained from data points provided to the algorithm or model. Once provided with the dataset, the model will run iteratively and will learn some inherent pattern. The learning thus acquired is called experience(E). Making an analogy with human learning, we can think of this situation as in which a human being is learning or gaining some experience from various attributes like situation, relationships etc. Supervised, unsupervised and reinforcement learning are some ways to learn or gain experience. The experience gained by out ML model or algorithm will be used to solve the task T.

Performance (P)

An ML algorithm is supposed to perform task and gain experience with the passage of time. The measure which tells whether ML algorithm is performing as per expectation or not is its performance (P). P is basically a quantitative metric that tells how a model is performing the task, T, using its experience, E. There are many metrics that help to understand the ML performance, such as accuracy score, F1 score, confusion matrix, precision, recall, sensitivity etc.

**5. Background Study**

From an NLP perspective, researchers have studied numerous aspects of the credibility of online information. For example, [1] applied the time-sensitive supervised approach by relying on tweet content to address the credibility of a tweet in different situations. [2] used LSTM in a similar problem of early rumour detection. In another work, [3] aimed at detecting the stance of tweets and determining the veracity of the given rumor with convolution neural networks. A submission [4] to the Samvel 2016 Twitter Stance Detection task focuses on creating a bag-of-words auto encoder and training it over the tokenized tweets. Another team, [5], combined multiple models in an ensemble providing a 50/50 weighted average between a deep convolutional neural network and a gradient-boosted decision tree. Though this work seems to be similar to our work, the difference lies in the construction of an ensemble of classifiers. In a similar attempt, a team [6] concatenated various features vectors and passed them through an NLP model. Passive Aggressive algorithm is a margin-based online learning algorithm for binary classification. It is also an algorithm of a soft margin-based method and robust to noise. It can be used in fake news detection [16] Term Frequency-Inverse Document Frequency is also a method used to represent text in Fake News Detector 12 a format that can be easily processed by machine learning algorithms. It is a numerical statistic that shows how important a word is to news in a news dataset. The importance of a word is proportional to the number of times the word appears in the news (fake and real) but inversely proportional to the number of times the word appears in the news dataset (fake or real)

**6. Feasibility Study**

Passive-aggressive classifier, logistic regression, LSTM can be used in fake news detection. Bi-directional LSTM was used in [7] to detect fake news. It had reasonably good accuracy but if the news was a bit more sophisticated, it would be difficult to achieve good accuracy. Because this model picks up the sensational/clickbait words as part of fake news. For example, if a news title says, ‘Donald Trump is the greatest president ever, the model will pick it up as fake news with reasonable accuracy. If the title is more nuanced and written in a sophisticated way, it’d be difficult to do so. We believe that our LSTM model is not enough by itself to detect fake news. That’s why we included passive aggressive classifier with it and when we compared passive news with reputable news sources, but the scope of the work is so vast that we couldn’t do it with the resources available to us. Our model can act as a first step in detecting fake news. But more work is needed to call the model reliable enough.

7. **RELATED WORK**

Considering the horrible consequences of fake critiques a number of techniques had been proposed. Most gift techniques use this as a trouble of binary classification [3-10] using super kinds of classifications, together with the Support Vector Machine (SVM), Artificial Neural Networks (ANN). While the proper preference of classifiers surely improves performance, studies show [3] that proper preference of features has a much greater vital influence. A type of features had been proposed and evaluated in view of the impact of features on the accuracy of the classifier. Part of the speech (POS) tags, Linguistic inquiry and word count, which in aggregate with unigrams and bigrams are used to similarly beautify performance. Jindal et al [17] first addressed the hassle that is then observed thru manner of manner of a number of one of a kind techniques. These techniques can be divided into major types (texts or metadata assessment) thru manner of manner of analyzing techniques (supervised, unsupervised and semi-supervised) or characteristic types. The techniques supervised like Liu et al.

[18] used the Bayesian approach and laid out a clustering trouble with opinion direct mail sensing. Mukherjee et al. [4], Chengai et al. [4] and Luyang et al. [9], no matter the reality that each one used the Support Vector Machine (SVM) as a classification, are one of a kind literature quantities that also have taken supervised analyzing. Mukherjee et al [18] advise an unmonitored technique based totally completely on the Bayesian framework. Other techniques are presented in [22] and [23], respectively, together with the relationship- based totally completely model (GSRank) and the sectioning of surprising guidelines and guidelines.

Moving to the features used, content material cloth Based techniques use the linguistic developments of the assessment to distinguish faux critiques. Part of the language tagging (POS) is one of the most widely used language features. The tagging of POS classifies terms used at the concept of word definition and context into noun, pronoun, verb, adverb, adjective etc. The authors of [Ref24] observed POS tags to collection patterns to come across gender from the blog writer. For a faux assessment detection substantially applied in [Ref3], POS unigram, POS bigrams, POS collection patterns, etc.

Many quantities of literatures [3, 5, 10, 14] genuinely monitor the general overall performance of the POS tags to discover linguistic functions. Another method for gaining knowledge of emotional, tremendous and cognitive components discovered in writing text samples is a geared up and operative method. Language inquiries and Word Counting (LIWC) The Otto et al. [10] cited that the LIWC has slightly superior on POS with inside the detection of deceptive evaluations spam. The LIWC with smooth unigrams and bigrams is also studied with inside the [3, 9, 15, 25]. Li et al. [25] analyzed quite a number disappointing symptoms and symptoms for detecting fake evaluations. They moreover decided that the aggregate of severalabilities which incorporates LIWC or POS allows for added accuracy in detection. In [7], every other language feature referred to as Deep Syntax abilties appears to be similar to POS and LIWC as it improves the sort reminder but reduces accuracy. Other word- based completely abilties which incorporates word unigrams and bigrams are lots much less effective for the reason that they are lots much less effective than POS tags and LIWC in extracting useful linguistic abilties. Because of the capability of the fake assessment writer to learn, the fake evaluations can be written very much like real evaluations. In the ones cases, linguistic abilties can fail high-quality to discover them; therefore a writer may moreover considerably discover fake evaluations with the resource of the use of more behavioral information which incorporates the rating pattern, assessment writing frequencies, etc. Lim et al. [21] provides the pattern of the product rankings and the assessment of spammers. The plunge withinside the examination pattern to understand a view spammer (some evaluations all through a brief time interval) is furnished in [10]. Fake evaluations which incorporates those furnished in [19, 26] or the text revision [22] are high-quality possible with behavioral information. More information on and among reviewers, who can be used to discover spammer businesses furnished in [18, 20, 21, 22], is available with inside the behavioral abilities. In addition, the behaviors, which incorporates word counts, POS tagging etc, have a look at the behavior of reviewer from different angles and do now now not include time ingesting linguistic analyses.

8.. **METHODOLOGY**

 Data collection

 Data Preprocessing

 Feature Engineering

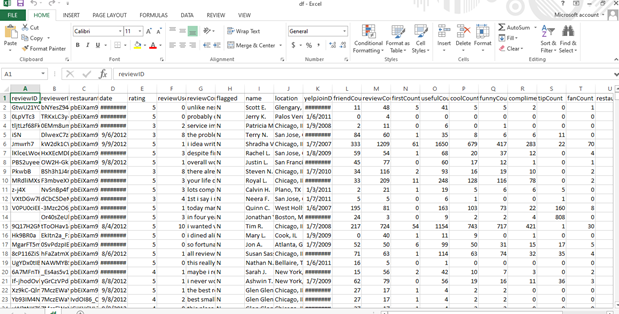
 Algorithms Used

 Visualization Of Output

**Data Collection**

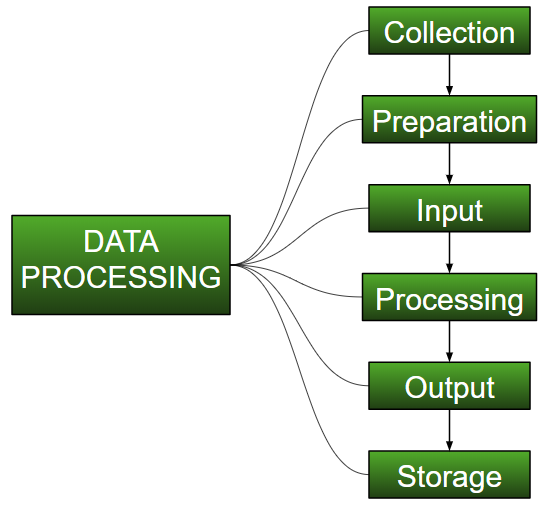
We have downloaded the dataset from "https://www.yelp.com".

It is a dataset of a restaurant where the clients has given the input of the restaurant by their ID, presently we will additionally execute our strategies on it.



**Data Processing**

Data is gathered and converted into usable data. Generally performed by a data scientist or group of data scientists, it is significant for data handling to be done effectively as not to contrarily influence the finished result or data yield.

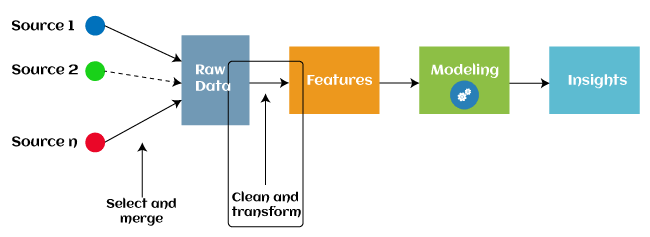


Data preprocessing begins with data in its crude structure and converts it into a more comprehensible arrangement (charts, records, and so on), giving it the structure and setting important to be deciphered by PCs and used by representatives all through an association.

We perform some operations for Converting to lowercase and for Spell error.

**Feature Engineering**

Utilizing area information to remove highlights from raw data, these features can be utilized to improve the exhibition of machine learning calculations.



We perform include designing for three areas, i.e

 Review centric

 Reviewer centric

 Network Centric

**Model-centric approach**

**The model-centric approach** means developing experimental research to improve the ml model performance. This involves selecting the best model architecture and training process from a wide range of possibilities.

* In this approach you keep the data the same, and improve the code or model architecture.
* Working on code is the central objective of this approach

**Data-centric approach**

In an age where data is at the core of every decision-making process, a data-centric company can better align its strategy with the interests of its stakeholders by using information generated from its operations. This way the result can be more accurate, organized, and transparent which can help an organization run more smoothly.

* This approach involves systematically altering/improving datasets in order to increase the accuracy of your ML applications.
* Working on data is the central objective of this approach.

Algorithms Used

 Random Forest Algorithm

 Naive Bayes Algorithm

**Random Forest Algorithm**

Random Forest Algorithm is an adaptable, simple-to-utilize machine learning calculation that produces, even without hyper-boundary tuning, an extraordinary outcome more often than not. It is likewise perhaps the most utilized calculations, as a result of its straightforwardness and variety (it very well may be utilized for both classification and regression undertakings).



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

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

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

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

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

**Step-4:** Repeat Step 1 & 2.

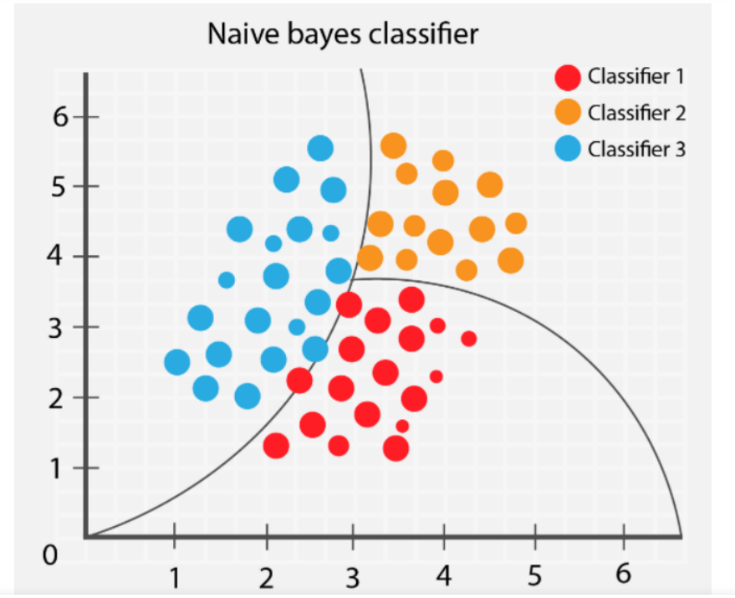
**Step-5:** For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

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

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

**Naive Bayes Algorithm**

Naive Bayes classifiers are an assortment of classification algorithms dependent on the Bayes Theorem. It's anything but a solitary calculation however a group of calculations where every one of them shares a typical standard. For example, each pair of highlights being grouped is free of one another.\



We had utilized Random Forest and Naive Bayes algorithms to discover the precision of program.

Ensemble modeling is a cycle where numerous assorted models are made to foresee a result, either by utilizing various modeling calculations or utilizing diverse preparing informational indexes. The ensemble model at that point totals the forecast of each base model and results in one last expectation for the concealed information. The inspiration for utilizing ensemble models is to diminish the speculation mistake of the expectation.

Also we have used

 Precision Score

In an imbalanced classification problem with two classes, precision is calculated as the number of true positives divided by the total number of true positives and false positives.

* Precision = TruePositives / (TruePositives + FalsePositives)

The result is a value between 0.0 for no precision and 1.0 for full or perfect precision.

Let’s make this calculation concrete with some examples.

Consider a dataset with a 1:100 minority to majority ratio, with 100 minority examples and 10,000 majority class examples.

A model makes predictions and predicts 120 examples as belonging to the minority class, 90 of which are correct, and 30 of which are incorrect.

The precision for this model is calculated as:

* Precision = TruePositives / (TruePositives + FalsePositives)
* Precision = 90 / (90 + 30)
* Precision = 90 / 120
* Precision = 0.75

The result is a precision of 0.75, which is a reasonable value but not outstanding.

You can see that precision is simply the ratio of correct positive predictions out of all positive predictions made, or the accuracy of minority class predictions.

Consider the same dataset, where a model predicts 50 examples belonging to the minority class, 45 of which are true positives and five of which are false positives. We can calculate the precision for this model as follows:

* Precision = TruePositives / (TruePositives + FalsePositives)
* Precision = 45 / (45 + 5)
* Precision = 45 / 50
* Precision = 0.90

In this case, although the model predicted far fewer examples as belonging to the minority class, the ratio of correct positive examples is much better.

This highlights that although precision is useful, it does not tell the whole story. It does not comment on how many real positive class examples were predicted as belonging to the negative class, so-called false negatives.

 Accuracy Score

Accuracy is a metric used in classification problems used to tell the percentage of accurate predictions. We calculate it by dividing the number of correct predictions by the total number of predictions.

Accuracy

This formula provides an easy-to-understand definition that assumes a [**binary classification**](https://en.wikipedia.org/wiki/Binary_classification) problem. (We discuss [**multiclass**](https://en.wikipedia.org/wiki/Multiclass_classification#General_strategies) and [**multilabel**](https://en.wikipedia.org/wiki/Multi-label_classification) problems in the second part of this article.)

In the binary classification case, we can express accuracy in [**True/False Positive/Negative**](https://en.wikipedia.org/wiki/Confusion_matrix#Table_of_confusion) values.

Accuracy model

Where

* TP : True Positives
* FP : False Positives
* TN : True Negatives
* FN : False Negatives
* The default form of accuracy gives an overall metric about model performance on the whole dataset.
* However, overall accuracy can be misleading when the class distribution is imbalanced, and it is critical to predict the minority class correctly.
* For example, in cancer prediction, we cannot miss malignant cases. Neither should we diagnose benign ones as malignant. Doing so would put healthy people through serious treatment and decrease trust in the whole diagnostic process.

 Recall Score

Recall is not limited to binary classification problems.

In an imbalanced classification problem with more than two classes, recall is calculated as the sum of true positives across all classes divided by the sum of true positives and false negatives across all classes.

* Recall = Sum c in C TruePositives\_c / Sum c in C (TruePositives\_c + FalseNegatives\_c)

As in the previous section, consider a dataset with a 1:1:100 minority to majority class ratio, that is a 1:1 ratio for each positive class and a 1:100 ratio for the minority classes to the majority class, and we have 100 examples in each minority class, and 10,000 examples in the majority class.

A model predicts 77 examples correctly and 23 incorrectly for class 1, and 95 correctly and five incorrectly for class 2. We can calculate recall for this model as follows:

* Recall = (TruePositives\_1 + TruePositives\_2) / ((TruePositives\_1 + TruePositives\_2) + (FalseNegatives\_1 + FalseNegatives\_2))
* Recall = (77 + 95) / ((77 + 95) + (23 + 5))
* Recall = 172 / (172 + 28)
* Recall = 172 / 200
* Recall = 0.86

 F1 score

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. In our case, F1 score is 0.701.

F1 Score = 2\*(Recall \* Precision) / (Recall + Precision)

So, whenever you build a model, this article should help you to figure out what these parameters mean and how good your model has performed.

 Visualization of Output

Visualization has been used a lot in domains like autonomous driving, urban planning, medical imaging to increase user trust in the models.

Visual analytics systems are being developed to understand more about harder kinds of networks like GANs, which have only been around for a couple of years but have produced remarkable results for data generation.

Examples include DGMTracker and GANViz, which focus on understanding the training dynamics of GANs to help model developers better train these complex models.

**Research & development**

Combining visualization with research has led to the creation of tools and frameworks for model interpretability and democratization. Another consequence of this rapidly developing area is that new work is immediately publicized and open-sourced without waiting for it to be “officially” published at some conference.

For example, the most popular libraries for implementing neural networks are open source and have consistent contributions for improving all areas of the codebase.

So far, we’ve talked about all the theoretical aspects of doing visualization, now let’s take a look at the most important one.

**9. Libraries Used:**

* **PANDA**

Pandas is an open-source library that is made mainly for working with relational or labeled data both easily and intuitively. It provides various data structures and operations for manipulating numerical data and time series. This library is built on top of the NumPy library.

* **STOPWORDS**

The process of converting data to something a computer can understand is referred to as pre-processing. One of the major forms of pre-processing is to filter out useless data. In natural language processing, useless words (data), are referred to as stop words.

* **RegexpTokenizer**

Class RegexpTokenizer(TokenizerI): r""" A tokenizer that splits a string using a regular expression, which matches either the tokens or the separators between tokens.

* **DATETIME**

The [datetime](https://docs.python.org/3/library/datetime.html#module-datetime) module supplies classes for manipulating dates and times.

While date and time arithmetic is supported, the focus of the implementation is on efficient attribute extraction for output formatting and manipulation.

* **NUMPY**

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. This tutorial explains the basics of NumPy such as its architecture and environment. It also discusses the various array functions, types of indexing, etc. An introduction to Matplotlib is also provided. All this is explained with the help of examples for better understanding.

TFIDF works by proportionally increasing the number of times a word appears in the document but is counterbalanced by the number of documents in which it is present.

* **Random Forest Classifiers**

Random forest classifiers are popular machine learning algorithms that are used for classification. In this post, you will learn about the concepts of random forest classifiers and how to train a Random Forest Classifier using the [Python](https://vitalflux.com/category/python)Sklearn library. This code will be helpful if you are a beginner [data scientist](https://vitalflux.com/category/data-science) or just want to quickly get a code sample to get started with training a machine learning model using the Random Forest algorithm

* **Train\_Test\_Split**

Quick utility that wraps input validation and next(ShuffleSplit().split(X, y)) and application to input data into a single call for splitting (and optionally subsampling) data in a oneliner.

* **GaussianNB**

A Gaussian Naive Bayes algorithm is a special type of NB algorithm. It's specifically used when the features have continuous values. It's also assumed that all the features are following a gaussian distribution i.e, normal distribution.

* **Sklearn.metrics**

Classification metrics. The sklearn. metrics module implements several loss, score, and utility functions to measure classification performance. Some metrics might require probability estimates of the positive class, confidence values, or binary decisions values.

* **Confusion\_Matrix**

A confusion matrix is a table that is used to define the performance of a classification algorithm. A confusion matrix visualizes and summarizes the performance of a classification algorithm. A confusion matrix is shown in Table 5.1, where benign tissue is called healthy and malignant tissue is considered cancerous.

* **OrderedDict**

An OrderedDict is a dictionary subclass that remembers the order that keys were first inserted. The only difference between dict() and OrderedDict() is that: OrderedDict preserves the order in which the keys are inserted.

* **plt**

Matplotlib is a library in Python and it is numerical – mathematical extension for NumPy library. Pyplot is a state-based interface to a Matplotlib module which provides a MATLAB-like interface. There are various plots which can be used in Pyplot are Line Plot, Contour, Histogram, Scatter, 3D Plot, etc.

* **Tqdm(Progress Meters Or Progress Bars.**)

tqdm is a library in Python which is used for creating Progress Meters or Progress Bars. tqdm got its name from the Arabic name taqaddum which means 'progress'

**10. The ML Model**

import pandas as pd

from nltk.corpus import stopwords

from nltk.tokenize import RegexpTokenizer

from datetime import datetime

import numpy as np

from sklearn.ensemble import RandomForestClassifier

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.model\_selection import train\_test\_split

from sklearn.naive\_bayes import GaussianNB

from sklearn.metrics import accuracy\_score, recall\_score, precision\_score, f1\_score, pairwise\_distances

from sklearn.metrics import confusion\_matrix

from collections import OrderedDict

import matplotlib.pyplot as plt

from tqdm import tqdm

pd.options.mode.chained\_assignment = None

df = pd.read\_csv("df.csv")

def data\_cleaning(df):

print("Cleaning Data")

# Removing \n from date field

for i in range(len(df['date'])):

if df['date'][i][0] == '\n':

df['date'][i] = df['date'][i][1:]

# Making yelpJoinDate Format Uniform

df['yelpJoinDate'] = df['yelpJoinDate'].apply(

lambda x: datetime.strftime(datetime.strptime(x, '%B %Y'), '01/%m/%Y'))

# Removing emtpy cells

if len(np.where(pd.isnull(df))) > 2:

# TODO

pass

# Pre-processing Text Reviews

# Remove Stop Words

stop = stopwords.words('english')

df['reviewContent'] = df['reviewContent'].apply(

lambda x: ' '.join(word for word in x.split() if word not in stop))

# Remove Punctuations

tokenizer = RegexpTokenizer(r'\w+')

df['reviewContent'] = df['reviewContent'].apply(

lambda x: ' '.join(word for word in tokenizer.tokenize(x)))

# Lowercase Words

df['reviewContent'] = df['reviewContent'].apply(

lambda x: x.lower())

print("Data Cleaning Complete")

return df

def feature\_engineering(df):

print("Feature Engineering: Creating New Features")

# Maximum Number of Reviews per day per reviewer

mnr\_df1 = df[['reviewerID', 'date']].copy()

mnr\_df2 = mnr\_df1.groupby(by=['date', 'reviewerID']).size().reset\_index(name='mnr')

mnr\_df2['mnr'] = mnr\_df2['mnr'] / mnr\_df2['mnr'].max()

df = df.merge(mnr\_df2, on=['reviewerID', 'date'], how='inner')

# Review Length

df['rl'] = df['reviewContent'].apply(

lambda x: len(x.split()))

# Review Deviation

df['rd'] = abs(df['rating'] - df['restaurantRating']) / 4

# Maximum cosine similarity

review\_data = df

res = OrderedDict()

# Iterate over data and create groups of reviewers

for row in review\_data.iterrows():

if row[1].reviewerID in res:

res[row[1].reviewerID].append(row[1].reviewContent)

else:

res[row[1].reviewerID] = [row[1].reviewContent]

individual\_reviewer = [{'reviewerID': k, 'reviewContent': v} for k, v in res.items()]

df2 = dict()

df2['reviewerID'] = pd.Series([])

df2['Maximum Content Similarity'] = pd.Series([])

vector = TfidfVectorizer(min\_df=0)

count = -1

for reviewer\_data in individual\_reviewer:

count = count + 1

# Handle Null/single review gracefully -24-Apr-2019

try:

tfidf = vector.fit\_transform(reviewer\_data['reviewContent'])

except:

pass

cosine = 1 - pairwise\_distances(tfidf, metric='cosine')

np.fill\_diagonal(cosine, -np.inf)

max = cosine.max()

# To handle reviewier with just 1 review

if max == -np.inf:

max = 0

df2['reviewerID'][count] = reviewer\_data['reviewerID']

df2['Maximum Content Similarity'][count] = max

df3 = pd.DataFrame(df2, columns=['reviewerID', 'Maximum Content Similarity'])

# left outer join on original datamatrix and cosine dataframe -24-Apr-2019

df = pd.merge(review\_data, df3, on="reviewerID", how="left")

df.drop(index=np.where(pd.isnull(df))[0], axis=0, inplace=True)

print("Feature Engineering Complete")

return df

def under\_sampling(df):

print("Under-Sampling Data")

# Count of Reviews

# print("Authentic", len(df[(df['flagged'] == 'N')]))

# print("Fake", len(df[(df['flagged'] == 'Y')]))

sample\_size = len(df[(df['flagged'] == 'Y')])

authentic\_reviews\_df = df[df['flagged'] == 'N']

fake\_reviews\_df = df[df['flagged'] == 'Y']

authentic\_reviews\_us\_df = authentic\_reviews\_df.sample(sample\_size)

under\_sampled\_df = pd.concat([authentic\_reviews\_us\_df, fake\_reviews\_df], axis=0)

# print("Under-Sampled Fake", len(under\_sampled\_df[(under\_sampled\_df['flagged'] == 'Y')]))

# print("Under-Sampled Authentic", len(under\_sampled\_df[(under\_sampled\_df['flagged'] == 'N')]))

# Graph of Data Distribution

# fig, ax = plt.subplots(figsize=(6, 4))

# sns.countplot(x='flagged', data=under\_sampled\_df)

# plt.title("Count of Reviews")

# plt.show()

print("Under-Sampling Complete")

return under\_sampled\_df

def semi\_supervised\_learning(df, model, algorithm, threshold=0.8, iterations=40):

df = df.copy()

print("Training "+algorithm+" Model")

labels = df['flagged']

df.drop(['reviewID', 'reviewerID', 'restaurantID', 'date', 'name', 'location', 'yelpJoinDate', 'flagged',

'reviewContent', 'restaurantRating'], axis=1, inplace=True)

train\_data, test\_data, train\_label, test\_label = train\_test\_split(df, labels, test\_size=0.25, random\_state=42)

test\_data\_copy = test\_data.copy()

test\_label\_copy = test\_label.copy()

all\_labeled = False

current\_iteration = 0

# param\_grid = {

# 'n\_estimators': [10, 500],

# 'max\_features': ['auto', 'sqrt', 'log2'],

# 'max\_depth': [4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14],

# 'criterion': ['gini', 'entropy']

# }

# grid\_clf\_acc = GridSearchCV(estimator=model, param\_grid=param\_grid, cv=5)

#

# grid\_clf\_acc.fit(train\_data, train\_label)

pbar = tqdm(total=iterations)

while not all\_labeled and (current\_iteration < iterations):

# print("Before train data length : ", len(train\_data))

# print("Before test data length : ", len(test\_data))

current\_iteration += 1

model.fit(train\_data, train\_label)

probabilities = model.predict\_proba(test\_data)

pseudo\_labels = model.predict(test\_data)

indices = np.argwhere(probabilities > threshold)

# print("rows above threshold : ", len(indices))

for item in indices:

train\_data.loc[test\_data.index[item[0]]] = test\_data.iloc[item[0]]

train\_label.loc[test\_data.index[item[0]]] = pseudo\_labels[item[0]]

test\_data.drop(test\_data.index[indices[:, 0]], inplace=True)

test\_label.drop(test\_label.index[indices[:, 0]], inplace=True)

# print("After train data length : ", len(train\_data))

# print("After test data length : ", len(test\_data))

print("--" \* 20)

if len(test\_data) == 0:

print("Exiting loop")

all\_labeled = True

pbar.update(1)

pbar.close()

predicted\_labels = model.predict(test\_data\_copy)

# print('Best Params : ', grid\_clf\_acc.best\_params\_)

print(algorithm + ' Model Results')

print('--' \* 20)

print('Accuracy Score : ' + str(accuracy\_score(test\_label\_copy, predicted\_labels)))

print('Precision Score : ' + str(precision\_score(test\_label\_copy, predicted\_labels, pos\_label="Y")))

print('Recall Score : ' + str(recall\_score(test\_label\_copy, predicted\_labels, pos\_label="Y")))

print('F1 Score : ' + str(f1\_score(test\_label\_copy, predicted\_labels, pos\_label="Y")))

print('Confusion Matrix : \n' + str(confusion\_matrix(test\_label\_copy, predicted\_labels)))

plot\_confusion\_matrix(test\_label\_copy, predicted\_labels, classes=['N', 'Y'],

title=algorithm + ' Confusion Matrix').show()

def plot\_confusion\_matrix(y\_true, y\_pred, classes, title=None, cmap=plt.cm.Blues):

# Compute confusion matrix

cm = confusion\_matrix(y\_true, y\_pred)

# Only use the labels that appear in the data

fig, ax = plt.subplots()

im = ax.imshow(cm, interpolation='nearest', cmap=cmap)

ax.figure.colorbar(im, ax=ax)

# We want to show all ticks...

ax.set(xticks=np.arange(cm.shape[1]),

yticks=np.arange(cm.shape[0]),

xticklabels=classes,

yticklabels=classes,

title=title,

ylabel='True label',

xlabel='Predicted label')

# Rotate the tick labels and set their alignment.

plt.setp(ax.get\_xticklabels(), rotation=45, ha="right",

rotation\_mode="anchor")

# Loop over data dimensions and create text annotations.

fmt = 'd'

thresh = cm.max() / 2.

for i in range(cm.shape[0]):

for j in range(cm.shape[1]):

ax.text(j, i, format(cm[i, j], fmt),

ha="center", va="center",

color="white" if cm[i, j] > thresh else "black")

fig.tight\_layout()

return plt

def main():

under\_sampled\_df = under\_sampling(df)

rf = RandomForestClassifier(random\_state=42, criterion='entropy', max\_depth=14, max\_features='auto',

n\_estimators=500)

nb = GaussianNB()

'''

from sklearn import svm

clf = svm.SVC(kernel='linear')

'''

semi\_supervised\_learning(under\_sampled\_df, model=rf, threshold=0.7, iterations=15, algorithm='Random Forest')

semi\_supervised\_learning(under\_sampled\_df, model=nb, threshold=0.7, iterations=15, algorithm='Naive Bayes')

if \_\_name\_\_ == '\_\_main\_\_':

main()

from sklearn.ensemble import VotingClassifier

from sklearn.metrics import log\_loss

# Splitting between train data into training and validation dataset

#X\_train, X\_test, y\_train, y\_test = train\_test\_split(train, target, test\_size=0.20)

# initializing all the model objects with default parameters

model\_1 =RandomForestClassifier()

model\_2 = GaussianNB()

# Making the final model using voting classifier

final\_model = VotingClassifier(

estimators=[('rf', model\_1), ('nb', model\_2)], voting='hard')

# training all the model on the train dataset

final\_model.fit(X\_train, y\_train)

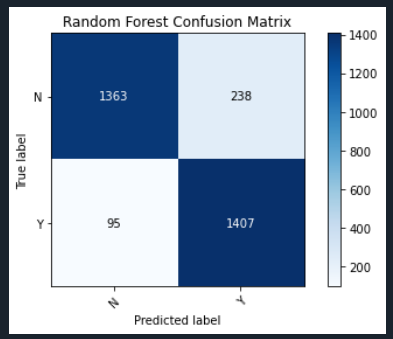
# predicting the output on the test dataset

pred\_final = final\_model.predict(X\_test)

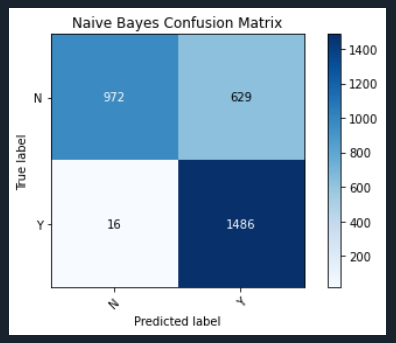
# printing log loss between actual and predicted value

print(log\_loss(y\_test, pred\_final))

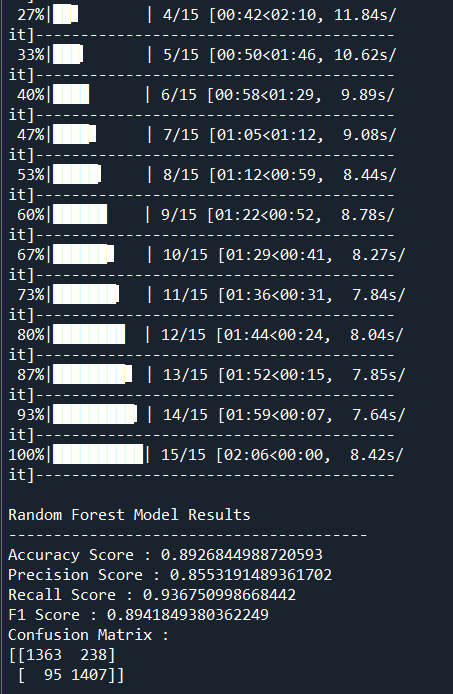
11. **EXPERIMENTAL RESULT**



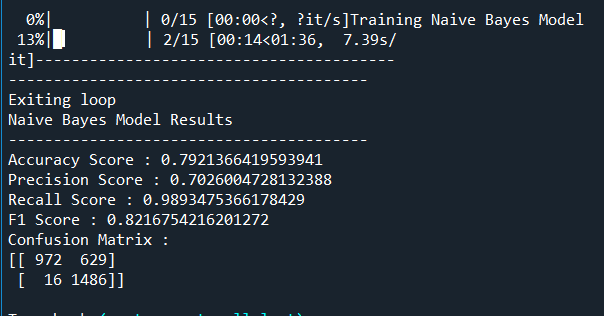
(Random Forest Confusion Matrix)



(Naive Bayes Confusion Matrix)



(Random Forest Algorithm Results)



(Naive Bayes Algorithm Results)

12. **CONCLUSION**

In this paper, we implemented three different Human Pose Estimation models to find out the most relevant among them. Our study was focused on main elements like “missing or false parts detection”, “occlusions”, “overlapping part” etc. and after looking to the results we came up with the conclusion that for a device like mobile phone or any lightweight device Blaze Pose estimation is the best techniques . For a Desktop CPU, PoseNet has given the best results.

The fake news by reviewing it in two stages: characterization and disclosure. In the first stage, the basic concepts and principles of fake news are highlighted in machine learning with dataset. During the discovery stage, the current methods are reviewed for detection of fake news using different supervised learning algorithms. As for [20] the displayed fake news detection approaches that is based on text analysis in the paper utilizes models based on speech characteristics and predictive models that do not fit with the other current models. From [21] they utilized Naive Bayes classifier to detect fake news from different sources, with results of accuracy of 74%. [22] Used combined ML algorithms, but they depend on unreliable probability threshold with 85-91% accuracy. [23] uses the Naive Bayes to detect fake news from different social media websites, but the results were not accurate for the untruthful sources. [24] They got their data from Kaggle with average accuracy of 74.5%. [27] Used Naive Bayes algorithms to detect Twitter spam senders, with accuracy rated from 70% to 71.2%. [28] They tried different approaches with accuracy of 76%. [29] Three common methods are utilized through their researches: Naïve Bayes, Neural Network and Support Vector Machine (SVM). Naïve Bayes has an accuracy of 96.08% for detecting fake messages. The neural network and the machine vector (SVM) reached an accuracy of 99.9 0%. [30] They used the combination of KNN and random forests that gave the final results improved by up to 8% using a mixed false message detection model. [31] They worked on 2012 Dutch elections fake news on Twitter, they examine the execution of 8 supervised machine learning classifiers in the Twitter dataset. And they assume that the decision tree algorithm works best for the data set used with a F score of 88%. [32] Presented a counterfeit detection model using N-gram analysis achieved the highest accuracy in use contains a unigram and a linear SVM workbook. The highest accuracy is 92%. In the aforementioned research summary and system analysis, we concluded that most of the research papers used naïve bays algorithm, and the prediction precision was between 70-76%, they mostly use qualitative analysis depending on sentiment analysis, titles, word frequency repetition [40][41][42]. In our approach we propose to add to these methodologies, another aspect, which is POS textual analysis, it is a quantitative approach, it depends on adding numeric statistical values as features, we thought that increasing these features and using random forest will give further improvements to precession results. The features we propose to add in our dataset are total words (tokens), Total unique words (types), Type/Token Ratio (TTR), Number of sentences, Average sentence length (ASL), Number of characters, Average word length (AWL), nouns, prepositions, adjectives etc.

13. **REFERENCES**

[1] F. Y. Wang, “The emergence of intelligent enterprises: From CPS to CPSS,” IEEE Intell. Syst., vol. 25, no. 4, pp. 85–88, Jul.-Aug.2010.

[2]. L. Atzori, A. Iera, G. Morabito, and M. Nitti, “The social internet of things (SIoT) – When social networks meet the internet of things: Concept, architecture and network characterization,” Comput. Netw., vol. 56, no. 16, pp. 3594–3608, 2012.

[3]. Overview of trust provisioning for information and communication technology infrastructures and services, ITU-T, Recommedation Y.3052, 2017.

[4]. Overview of trust provisioning for information and communication technology infrastructures and services, ITU-T, Recommedation Y.3052, 2017.

[5]. S. P. Marsh, “Formalising trust as a computational concept,” Ph.D.dissertation, Dept. Comput. Sci. Math., Stirling Univ., Scotland, UK, 1994.

[6]. Hwang, K., Li, D.: Trusted Cloud Computing with Secure Resources and Data Coloring. IEEE Internet Computing 14(5), 14–22 (2010)

[7] M. L. Jensen, J. M. Averbeck, Z. Zhang, and K. B. Wright, ``Credibility of anonymous online product reviews: A language expectancy perspective, J. Manage. Inf. Syst., vol. 30, no. 1, pp. 293\_324, Jul. 2013.

[8] A. U. Akram, H. U. Khan, S. Iqbal, T. Iqbal, E. U. Munir, and M. Sha\_,``Finding rotten eggs: A review spam detection model using diverse feature sets,'' KSII Trans. Internet Inf. Syst., vol. 12, no. 10, pp. 5120\_5142, Oct. 2018

[9] M. Luca and G. Zervas, ``Fake it till you make it: Reputation, competition,and yelp review fraud,'' Manage. Sci., vol. 62, no. 12, pp. 3412\_3427,Dec. 2016.

[10] M. Luca and G. Zervas, ``Fake it till you make it: Reputation, competition, and yelp review fraud,'' Manage. Sci., vol. 62, no. 12, pp. 3412\_3427, Dec. 2016.

[11] J. Wang, H. Kan, F. Meng, Q. Mu, G. Shi, and X. Xiao, ``Fake reviewdetection based on multiple feature fusion and rolling collaborative training,''IEEE Access, vol. 8, pp. 182625\_182639, 2020.

[12] M. Ott, Y. Choi, C. Cardie, and J. T. Hancock, ``Finding deceptive opinionspam by any stretch of the imagination,'' in Proc. 49th Annu. Meeting Assoc. Comput. Linguistics, Hum. Lang. Technol. (HLT). Portland, OR,USA: Association for Computational Linguistics, Jun. 2011

[13] M. R. Martinez-Torres and S. L. Toral, ``A machine learning approach for the identification of the deceptive reviews in the hospitality sector using unique attributes and sentiment orientation,'' Tourism Manage., vol. 75, pp. 393\_403, Dec. 2019.

[14] G. Cui, H.-K. Lui, and X. Guo, ``The effect of online consumer reviews on new product sales,'' Int. J. Electron. Commerce, vol. 17, no. 1, pp. 39\_58,Oct. 2012.

[15] D. Zhang, L. Zhou, J. L. Kehoe, and I. Y. Kilic, ``What online reviewer behaviors really matter? Effects of verbal and nonverbal behaviors on detection of fake online reviews,'' J. Manage. Inf. Syst., vol. 33, no. 2,pp. 456\_481, Apr. 2016.

[16] T. C. Alberto, J. V. Lochter, and T. A. Almeida, ``Post or block? Advancesin automatically \_ltering undesired comments,'' J. Intell. Robot. Syst.,vol. 80, no. S1, pp. 245\_259, Dec. 2015.

[17] G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, and G. Bing,``Learning from class-imbalanced data: Review of methods and applications,''Expert Syst. Appl., vol. 73, pp. 220\_239, May 2017.

[18] R. Barbado, O. Araque, and C. A. Iglesias, ``A framework for fake review detection in online consumer electronics retailers,'' Inf. Process. Manage.,vol. 56, no. 4, pp. 1234\_1244, Jul. 2019.

[19] L. Chen,W. Li, H. Chen, and S. Geng, ``Detection of fake reviews: Analysis of Sellers' manipulation behavior,'' Sustainability, vol. 11, no. 17, p. 4802,Sep. 2019.

[20] D. Liang, C.-F. Tsai, and H.-T. Wu, ``The effect of feature selection on nancial distress prediction,'' Knowl.-Based Syst., vol. 73, pp. 289\_297, Jan. 2015.

[21] Y. Li and Z. Yang, ``Application of EOS-ELM with binary jaya-based feature selection to real-time transient stability assessment using PMU data,'' IEEE Access, vol. 5, pp. 23092\_23101, 2017.

[22] H. Ullah, M. Uzair, A. Mahmood, M. Ullah, S. D. Khan, and F. A. Cheikh,``Internal emotion classi\_cation using EEG signal with sparse discriminative ensemble,'' IEEE Access, vol. 7, pp. 40144\_40153, 2019.

[23] M. Ott, C. Cardie, and J. T. Hancock, ``Negative deceptive opinion spam,'' in Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technologie (HLT-NAACL), Atlanta, GA, USA, Jun. 2013, pp. 497\_501.

[24] G. Wang, S. Xie, B. Liu, and P. S. Yu, ``Identify online store review spammers via social review graph,'' ACM Trans. Intell. Syst. Technol., vol. 3, no. 4, pp. 1\_21, Sep. 2012.

[25] D. H. Fusilier, M. Montes-y-Gómez, P. Rosso, and R. G. Cabrera, ``Detecting positive and negative deceptive opinions using PU-learning,'' Inf. Process. Manage., vol. 51, no. 4, pp. 433\_443, Jul. 2015.

[26] W. Zhang, Y. Du, T. Yoshida, and Q. Wang, ``DRI-RCNN: An approach to deceptive review identification using recurrent convolutional neural network,'' Inf. Process. Manage. vol. 54, no. 4, pp. 576\_592, Jul. 2018.

[27] N. Ruan, R. Deng, and C. Su, ``GADM: Manual fake review detection for O2O commercial platforms,'' Comput. Secur., vol. 88, Jan. 2020,