PROJECT REPORT

TEAM MEMBERS: AKHILA AKKUNURI, SRUTHI KANDHALA, SIVA KALYAN GANGALA

PROJECT 1: STROKE ANALYSIS

1. INTRODUCTION

Heart stroke is a fatal condition that occurs when blood supply to a part of the brain is turned off. This results in deprivation of oxygen and glucose to the brain cells which then results in dying. If a stroke is not treated immediately, it can lead to severe brain damage or death.

Heart stroke can be prevented if signs are detected early. A stroke can affect anyone at any age. However, if you have certain risk factors, your odds of having a stroke increase. Some stroke risk factors can be handled, while others cannot. Unhealthy food and alcohol consumption, tobacco use, diabetes, inactive lifestyle, high blood pressure, and family history are all risk factors. Detecting a heart attack and seeking medical care as soon as possible can not only help you live longer, but it can also help you avoid heart disease in the future.

In our project, we would like to detect whether a patient is at risk of heart stroke by checking a few common factors such as gender, age, hypertension, prior heart disease status, marital status, average glucose level, work type, residence type, BMI (Body Mass Index), and smoking status.

Hypertension, diabetes, cholesterol, smoking, and alcohol intake are some of the risk factors for heart stroke that can be controlled. Additionally, there is a link between stroke and stress. Stress causes the heart to work harder, raises blood pressure, and spikes blood sugar and fat levels. These factors can trigger the development of blood clots and travel to the heart or brain, resulting in a heart attack or stroke.

Not all risks are in our hands. Some uncontrollable risk factors are:

- Age: People above the age of 65 are at a higher risk for stroke.
- Gender: It is a known fact that men have more strokes than women, but women have more lethal strokes
- Family history
- Race

2. LITERATURE REVIEW

Heart disease is the most serious and which ultimately can lead to death. It will lead to long-term damage. This condition will attack without caution. We have a lot of medical data, however there is little known about the insights from this data. Because of this, a major part of medical assistance is diagnosing patients in a prompt manner. A hospital's inaccurate diagnosis would lead to loss of life and hospital reputation as well. The significant biological challenge is the accurate assessment of cardiac disease. The main purpose of this research is to use data mining techniques to give an appropriate solution for restorative circumstances. To diagnose cardiac illnesses, data mining classification methods such as decision trees, Association Rule, Bayesian classifiers, Support vector machines, neural networks, and K- closest neighbor categorizations are utilized. Support Vector Machine is one of these specific procedures of data mining.

2.1 RELATED WORK

Govindarajan et al. [1] used a text mining and a machine learning classifier to categorize heart stroke disease. They used multiple machine learning algorithms for training purposes with ANN (Artificial Neural Networks) for their analysis, and the SGD (Stochastic Gradient Descent) algorithm provided the greatest value, which was 95 percent.

Riddhi Kasabe [2] used the decision tree classifier technique to predict heart attack. The model first learns deep features based on the dataset's properties, then trains on the learned features to get the outcome or prediction.

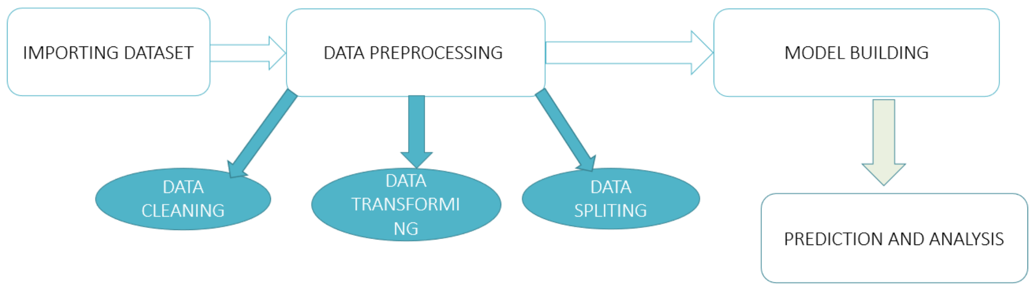
Minhaz Uddin Emon et al. [3] used 10 different classifiers and the results were aggregated by a weighted voting approach to get the highest accuracy.

S. Babu et al., [4] The health-care industry is making to detect the sickness early on. The expense of therapy is prohibitive for most people. As a result, many are hesitant to seek effective therapy early in the disease's progress. The goal of this research is to identify disease earlier for a reasonable price. We can detect sickness early and cure it fully by employing data mining techniques. The healthcare sector accumulates a massive amount of data that is not mined for hidden information. Data mining is a way to solve this issue. The process of examining a large amount of data and refining it into valuable information is known as data mining.

C. Raju et al., [5] here prediction be done after evaluating the applicability of various data mining approaches. For identifying hidden patterns from huge databases, data mining combines statistical analysis, machine learning techniques, and database technology. Clinical and morbid data are often used to diagnose heart disease. The heart disease prediction system will assist medical personnel in forecasting the condition of heart disease based on clinical data from patients. Researchers at this institution use a variety of data mining approaches to aid medical practitioners in improving their accuracy. Some of the strategies utilized here include neural networks, Naive Bayes, Genetic algorithms, Decision Trees, classification by clustering, and Support Vector Machines (SVM).

3. APPROACH

STROKE ANALYSIS METHODOLOGY



The project is implemented in Python in Jupyter notebooks, and the dataset is downloaded from Kaggle.

Dataset - https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset

i. Import Python Libraries:

For this project, we need to import NumPy, pandas and matplotlib. NumPy (numerical Python) is a library that consists of multidimensional array objects and a gathering of functions for manipulating them. NumPy allows you to conduct mathematical and logical functions on arrays.

Pandas is an open-source Python library based on the NumPy library. It is a Python package that lets you manipulate numerical data and time series using a variety of data structures and methods. It is mostly used for data input and analysis.

ii. Import Dataset:

The dataset which we downloaded from Kaggle is imported using the read_csv() method provided by the pandas library. We can then assign it to a DataFrame and perform all the manipulations on it. DataFrame allows us to access each row, column, and value. Our dataset consists of 5110 instances and 12 columns.

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	9046	Male	67.0	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked	1
1	51676	Female	61.0	0	0	Yes	Self-employed	Rural	202.21	NaN	never smoked	1
2	31112	Male	80.0	0	1	Yes	Private	Rural	105.92	32.5	never smoked	1
3	60182	Female	49.0	0	0	Yes	Private	Urban	171.23	34.4	smokes	1
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural	174.12	24.0	never smoked	1
...	...	...	...	...	...	...	...	...	...	...	...	...
5105	18234	Female	80.0	1	0	Yes	Private	Urban	83.75	NaN	never smoked	0
5106	44873	Female	81.0	0	0	Yes	Self-employed	Urban	125.20	40.0	never smoked	0
5107	19723	Female	35.0	0	0	Yes	Self-employed	Rural	82.99	30.6	never smoked	0
5108	37544	Male	51.0	0	0	Yes	Private	Rural	166.29	25.6	formerly smoked	0
5109	44679	Female	44.0	0	0	Yes	Govt_job	Urban	85.28	26.2	Unknown	0

5110 rows × 12 columns

Fig 1: Dataset assigned to a DataFrame

iii. <u>Data Preprocessing:</u>

Data Preprocessing step involved the following tasks:

- Data Cleaning -

The dataset imported in the previous step has null values. The first task in this step is to remove these null values. We used the DataFrame.dropna() method provided by the pandas library. The dataset consists of 161 instances with null values. These instances were removed, and the number of rows was reduced to 4909.

- Data Transformation -

The attributes gender, ever married, work_type, residence_type, and smoking status are categorical in nature. To solve a classification problem, we need to work with numerical data. Therefore, we converted all the categorical values to numeric values. For example, in the gender attribute, we have three distinct values: Male, Female, and other. We have assigned the value of 0 to Female, 1 to Male, and 2 to other.

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	9046	Male	67.0	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked	1
1	51676	Female	61.0	0	0	Yes	Self-employed	Rural	202.21	NaN	never smoked	1
2	31112	Male	80.0	0	1	Yes	Private	Rural	105.92	32.5	never smoked	1
3	60182	Female	49.0	0	0	Yes	Private	Urban	171.23	34.4	smokes	1
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural	174.12	24.0	never smoked	1
...	...	...	...	...	...	...	...	...	...	...	...	...
5105	18234	Female	80.0	1	0	Yes	Private	Urban	83.75	NaN	never smoked	0
5106	44873	Female	81.0	0	0	Yes	Self-employed	Urban	125.20	40.0	never smoked	0
5107	19723	Female	35.0	0	0	Yes	Self-employed	Rural	82.99	30.6	never smoked	0
5108	37544	Male	51.0	0	0	Yes	Private	Rural	166.29	25.6	formerly smoked	0
5109	44679	Female	44.0	0	0	Yes	Govt_job	Urban	85.28	26.2	Unknown	0

5110 rows × 12 columns

Fig 2: Dataset before cleaning

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	9046	Male	67.0	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked	1
1	31112	Male	80.0	0	1	Yes	Private	Rural	105.92	32.5	never smoked	1
2	60182	Female	49.0	0	0	Yes	Private	Urban	171.23	34.4	smokes	1
3	1665	Female	79.0	1	0	Yes	Self-employed	Rural	174.12	24.0	never smoked	1
4	56669	Male	81.0	0	0	Yes	Private	Urban	186.21	29.0	formerly smoked	1
...	...	...	...	...	...	...	...	...	...	...	...	...
4904	14180	Female	13.0	0	0	No	children	Rural	103.08	18.6	Unknown	0
4905	44873	Female	81.0	0	0	Yes	Self-employed	Urban	125.20	40.0	never smoked	0
4906	19723	Female	35.0	0	0	Yes	Self-employed	Rural	82.99	30.6	never smoked	0
4907	37544	Male	51.0	0	0	Yes	Private	Rural	166.29	25.6	formerly smoked	0
4908	44679	Female	44.0	0	0	Yes	Govt_job	Urban	85.28	26.2	Unknown	0

4909 rows × 12 columns

Fig 3: Dataset after cleaning

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	9046	1	67.0	0	1	1	0	0	228.69	36.6	0	1
1	31112	1	80.0	0	1	1	0	1	105.92	32.5	1	1
2	60182	0	49.0	0	0	1	0	0	171.23	34.4	2	1
3	1665	0	79.0	1	0	1	1	1	174.12	24.0	1	1
4	56669	1	81.0	0	0	1	0	0	186.21	29.0	0	1
...	...	...	...	...	...	...	...	...	...	...	...	...
4904	14180	0	13.0	0	0	0	3	1	103.08	18.6	3	0
4905	44873	0	81.0	0	0	1	1	0	125.20	40.0	1	0
4906	19723	0	35.0	0	0	1	1	1	82.99	30.6	1	0
4907	37544	1	51.0	0	0	1	0	1	166.29	25.6	0	0
4908	44679	0	44.0	0	0	1	2	0	85.28	26.2	3	0

4909 rows × 12 columns

Fig 4: After converting categorical data into numerical data

iv. <u>Data Visualization:</u>

After the preprocessing stage, we visualized each attribute using a histogram. This helped us understand the data distribution.

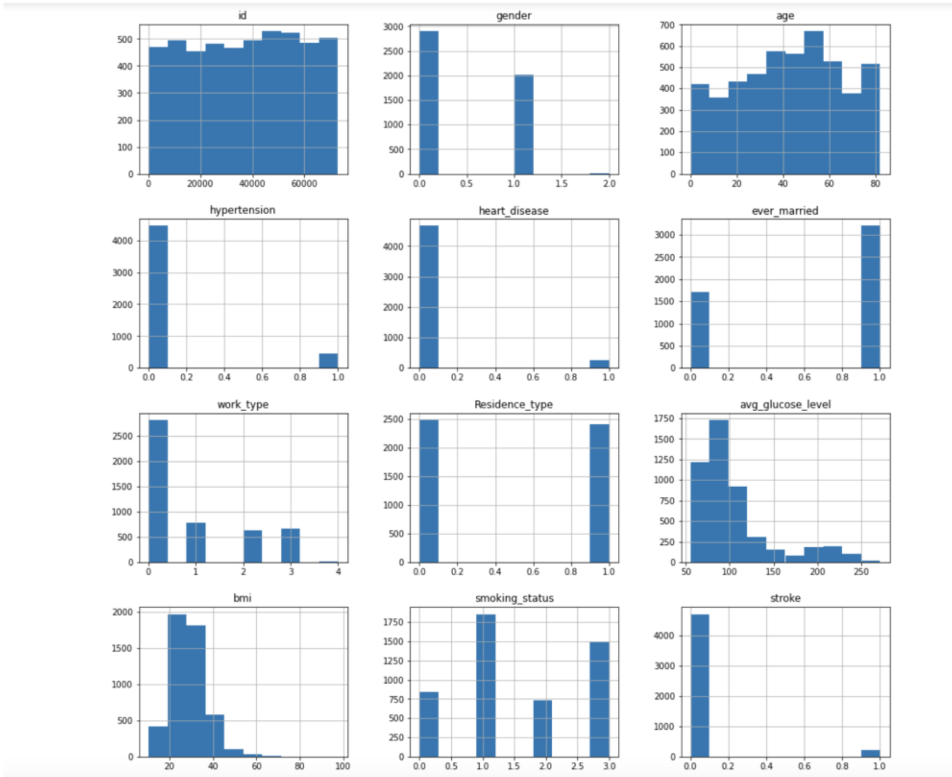


Fig 5: Histogram representing data distribution

v. <u>Log Transformation:</u>

Initially, after the preprocessing stage, we wanted to scale the data values between a range of 0 and 1 using min-max normalization. However, after viewing the data distribution, we decided to use log transformation to map our distribution to as close possible to a normal distribution one.

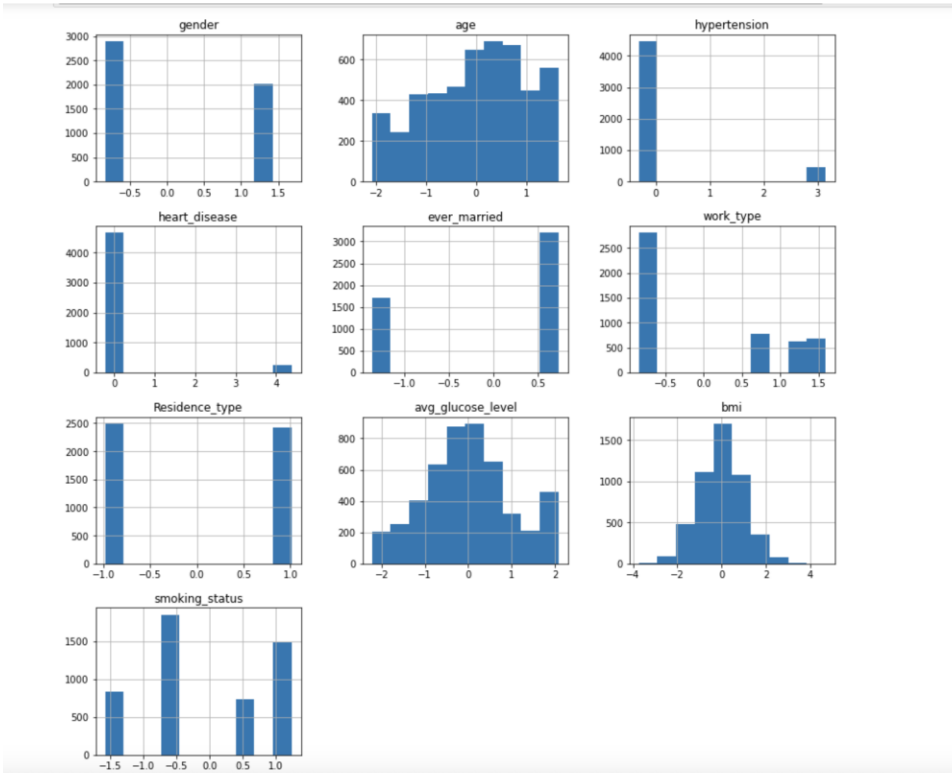


Fig 6: Data Distribution after log transformation

vi. Data Splitting:

Data splitting is done to avoid overfitting and model selection bias. Our dataset is split into two types: training and testing datasets.
To split the dataset, we import train_test_split from the sklearn.model_selection package. In our project we split the dataset into training data (70%) and test data (30%). Additionally, our dataset is imbalanced. Therefore, the splitting is done in such a way that the ratio of the outcomes is the same in both the training and testing datasets. This is achieved by using a parameter called 'stratify' in the train_test_split() method.

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status
3311	37029	1	5.0	0	0	0	3	1	97.64	17.0	3
1310	67942	1	21.0	0	0	0	0	1	65.09	23.5	1
3239	23757	0	60.0	0	0	0	0	0	105.48	28.4	3
728	19419	1	14.0	0	0	0	3	1	91.25	23.8	3
3443	60586	0	68.0	0	0	1	0	1	85.29	27.1	0
...	...	...	...	...	...	...	...	...	...	...	...
3738	35450	0	51.0	0	0	1	0	1	93.67	19.2	1
876	36471	1	65.0	0	0	1	0	0	145.15	28.9	3
1759	24735	0	21.0	0	0	0	0	1	80.84	30.7	3
3086	54395	0	78.0	1	0	1	1	1	152.38	31.8	1
3594	29863	1	44.0	0	0	0	0	0	103.44	28.0	1

3436 rows × 11 columns

Fig 7: Training Dataset - x_train

vii. Modeling:

For the modeling phase, the classification algorithm that we selected is Linear Support Vector Machine (LinearSVM). The dataset that we are using is an imbalanced dataset with the majority class being 0 (no heart stroke) and the minority class being 1 (heart stroke). Many of the classification machine learning algorithms work on the assumption that the data is evenly distributed within the classes. If we fit an imbalanced dataset to the model, the predictions will be biased towards the majority class, that is class 0 in our dataset. A way to go about this issue is by assigning class weights. The entire purpose of assigning class weights is to penalize the misclassification made by the minority class by setting a higher weight and reducing the weight for the majority class. In our project, we assigned a weight of 1.0 to class 0 and 50.0 to class 1.

LinearSVC is like Support Vector Classifier with linear kernel but with more flexibility in the choice of penalties. Therefore, we decided to use LinearSVC. We fit the training dataset to the classifier.

viii. Prediction:

After we were done training the model, we used the testing dataset to make predictions. The metrics selected to calculate the performance of the model are recall, balanced accuracy, and ROC_AUC. As our dataset is an imbalanced dataset, we go for metrics other than accuracy. Classification accuracy is misleading in the case of an imbalanced dataset. In such cases, a performance metric that we can consider is balanced accuracy. Using k-fold cross-validation, with k=5, we calculated balanced accuracy, recall, and ROC_AUC scores. Balanced accuracy is specifically used while dealing with imbalanced data. High recall and ROC_AUC scores mean that the model is performing well. Another metric that we considered is mean squared error. A lower value is ideal.

Accuracy = (True Positive + True Negative) / (True Positive + True Negative + False Positive + False Negative)

Mean Squared Error = The square of the difference between the predicted value and the actual value.

Balanced accuracy = (Sensitivity + Specificity) / 2

Where Sensitivity = True Positive/ (True Positive + False Negative) and Specificity = True Negative/ (True Negative + False Positive)

Recall = True Positive / (True Positive + False Negative)

ROC_AUC = Calculated the area under the ROC curve.

4. RESULTS

```
accu=accuracy_score(y_test,predict1)
print(accu)
```

```
0.5756958587915818
```

```
cm= confusion_matrix(y_test, predict1)
cm
```

```
array([[791, 619],
       [  6,  57]])
```

Fig 8: Accuracy Score and Confusion Matrix

```
Recall:
[0.97619048 0.9047619  0.92857143 0.92857143 0.92682927]
Balanced Accuracy:
[0.7545846  0.72844478 0.73556231 0.73077508 0.74054229]
ROC_AUC:
[0.8654002  0.81557751 0.83690476 0.81818642 0.8654904 ]
```

Fig 9: Cross validation of Recall, Balanced Accuracy, and ROC_AUC

```
mean_squared_error(y_test, predict1)
```

```
0.4243041412084182
```

Fig 10: Mean squared error

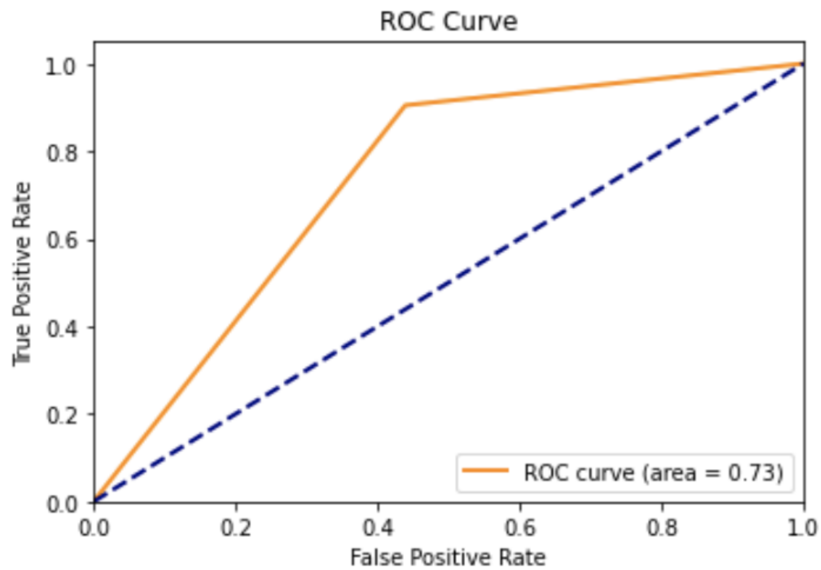


Fig 11: ROC Curve

5. CONCLUSION

Machine learning is a game changer in the medical field. Being able to predict whether a patient is susceptible to a disease based on certain attributes can help in preventing patients from even getting the disease. In this project, we used LinearSVC to do our analysis on patients susceptible to heart stroke. Using k-fold cross validation method where k=5, we achieved a high recall, balanced accuracy, and ROC_AUC score for all values of k. As our dataset is imbalanced, we chose these performance metrics to evaluate our model.

6. REFERENCES

[1] P. Govindarajan, R. K. Soundarapandian, A. H. Gandomi, R. Patan, P. Jayaraman, and R. Manikandan, "Classification of heart stroke disease using machine learning algorithms," Neural Computing and Applications, vol. 32, no. 3, pp. 817–828, Feb. 2020.

[2] Riddhi Kasabe, Heart Disease Prediction using Machine Learning, International Journal of Engineering Research & Technology (IJERT); http://www.ijert.org ISSN: 2278-0181; Vol. 9 Issue 08, August-2020

[3] Minhaz Uddin Emon, Maria Sultana Keya, Tamara Islam Meghla, Md. Mahfujur Rahman, M Shamim Al Mamun, M Shamim Kaiser, "Performance Analysis of Machine Learning Approaches in Stroke Prediction", DOI: 10.1109/ICECA49313.2020.9297525

[4] S. Babu *et al*., "heart disease diagnosis using data mining technique," *2017 International conference of Electronics, Communication and Aerospace Technology (ICECA)*, 2017, pp. 750-753, doi: 10.1109/ICECA.2017.8203643.

[5] C. Raju, E. Philipsy, S. Chacko, L. Padma Suresh and S. Deepa Rajan, "A Survey on Predicting Heart Disease using Data Mining Techniques," *2018 Conference on Emerging Devices and Smart Systems (ICEDSS)*, 2018, pp. 253-255, doi: 10.1109/ICEDSS.2018.8544333.

[6]https://www.webmd.com/heartdisease/stroke#:~:text=Ischemic%20stroke%20is%20similar%20to,flow%20to%20the%20brain's%20cells.

PROJECT 2: HATE SPEECH ANALYSIS

1. INTRODUCTION

Social Media has a significant impact on our lives these days. There are many social media platforms where people are free to express their opinions and feelings in a more open way. One such important platform is Twitter, where people express their thoughts in short phrases. Twitter users can also tag a person or post comments to any event that is trending at that point of time. Many celebrities, political figures and influencers use Twitter as a base to express their thoughts.

As with any media, as the content grows so do the reactions among people which also includes hate/negative speech. The mobility has allowed an increase in hate speech which in turn leads to an increase in hate crime.

With our project we would like to classify the data/tweets into hate, offensive and neutral categories. We have downloaded our dataset from Kaggle, and it has close to 25700 rows of data with seven columns. We take the tweet into account and count the number of classified words (classified into hate, offensive) used and then categorize it. We will consider the quantity and quality of the word semantics to distinguish the neutral, hate and offensive words. Also, we will show the analysis of the most used words in the dataset.

2. LITERATURE REVIEW

Separating hate speech from other forms of abusive language is the main hindrance to automatic hate speech monitoring on social media. Because lexical activities and experience label all communications carrying specified phrases as hate speech, past work utilizing supervised learning has failed to discriminate between the two groups, and they have low accuracy. They collected tweets aiming hate speech buzzwords using a crowd-sourced hate speech glossary. They employed crowdsourcing to classify a sample of these tweets into three groups: those containing hate speech, those containing merely foul language, and those having neither. To discriminate between these multiple categories, we train a multi-class classifier. When we investigate the predictions and mistakes thoroughly, we can reliably distinguish hate speech from other objectionable words.

2.1 RELATED WORK

Davidson, T., Warmsley, D., Macy, M. and Weber, I., 2017. Automated Hate Speech Detection and the Problem of Offensive Language. ArXiv,. [1] Gathered tweets containing hate speech terms using a crowd-sourced hate speech lexicon. They used Crowdsourcing to categorize a sample of these tweets into three categories: hate speech, offensive language alone, and neither. To distinguish between these multiple categories, they train a multi-class classifier.

Schmidt, Anna, and Michael Wiegand. "A survey on hate speech detection using natural language processing." [3] The primary topics that have been investigated to automatically detect these sorts of statements using natural language processing are described in this survey.

On social media hate speech and harsh language have grown commonplace. Automatic hate speech and abuse detection technologies can help to make toxic textual content illegal. The complexities, informality, and ambiguity of English dialects made it difficult to get the materials needed for abusive/hate speech detection study in English. The first publicly available Levantine Hate Speech and Abusive (L-HSAB) [3] Twitter dataset is presented in this research, with the goal of serving as a benchmark dataset for automated identification of online Levantine toxic materials. The annotation requirements are constructed in such a way that a trustworthy dataset annotation is assured. The annotation agreement metrics of Cohen's Kappa (k) and Krippendorff's alpha () demonstrated the consistency, which was further reinforced by a detailed review of the annotations.

The study's focus was on recognizing offensive text. They recommend that the recognition process be redirected from data-driven to human-centric, based on emerging perception perspectives (mesoscopic and microscopic) [4]. Specific person perspectives and opinions could be recognized to attain this. Additional information about persons' redesigned annotation procedures and new multimodal reasoning procedures are all essential for such a transformation. Following this logic, we demonstrated that customized solutions may be generated that are more precise than generic solutions and accurately reflect customer requirements. We also discovered a nearly linear relationship between agreement level and detection quality. Not only in NLP, but in all other activities with a significant amount of uncertainty, conflict, or subjectivity, we feel our methodologies and discoveries can be utilized.

3. APPROACH

The project is implemented in Python in Jupyter notebooks, and the dataset is downloaded from Kaggle. With our project, we would like to classify the data/tweets into hate, offensive and neutral categories. We take the tweet into account and count the number of classified words (classified into hate, offensive) used and then categorize it. For the modeling phase, the classification algorithm that we selected is logistic regression. We have downloaded our dataset from Kaggle, and it has nearly 25700 rows of data with seven columns.

https://www.kaggle.com/datasets/mrmorj/hate-speech-and-offensive-language-dataset

Fig 12: Steps involved in hate text analysis

Steps involved:

i. Import Python libraries:

This project needs pandas, NLTK, seaborn and NumPy libraries.

Pandas is an open-source Python library based on the NumPy library. It is a Python package that lets you manipulate numerical data and time series using a variety of data structures and methods. It is mostly used for data input and analysis.

NLTK (Natural Language Tool Kit) is a suite of libraries to work with natural language processing and computational linguistics for English language in Python.

Seaborn is a Python library built on matplotlib for data analysis and data visualization. Seaborn methods can be comfortably used to work with Pandas dataframe.

NumPy (numerical Python) is a library that consists of multidimensional array objects and a collection of functions for manipulating them. NumPy allows you to conduct mathematical and logical operations on arrays.

ii. <u>Import the data:</u>

The dataset which we downloaded from Kaggle is imported using the **read_csv()** method provided by the pandas library. We can then assign it to a DataFrame which allows us to access every row and column of our dataset and perform all the manipulations on it. Our dataset consists of 24783 instances and 7 columns.

24783 rows × 7 columns

Fig 13: Imported data from the dataset

iii. <u>Preprocessing the data:</u>

The following process has been followed as part of preprocessing the dataset.

Using Regular Expressions pattern search we remove any extra spaces, @mentions, #links, punctuation marks and numbers that were used in the tweet.

Using tokenization from NLTK library, we turn the tweets into tokens (smaller words or phrases). These tokens will be easy to trim and process further.

The tokens are now trimmed of the stopwords using the nltk.corpus library functions. Stop words are the most used words (like ‘a,’ ’and,’ ‘before’ etc.) which can be ignored during linguistic analysis.

The tokens are stemmed using the stemming algorithms provided by the nltk.stem library. Stemming is a process to identify the root/base word for any given word.

```
0        woman complain clean hous amp man alway take t...
1        boy dat cold tyga dwn bad cuffin dat hoe st place
2               dawg ever fuck bitch start cri confus shit
3                                         look like tranni
4           shit hear might true might faker bitch told ya
                                 ...
24778    muthaf lie right tl trash mine bibl scriptur hymn
24779      gone broke wrong heart babi drove redneck crazi
24780    young buck wanna eat dat nigguh like aint fuck...
24781                        youu got wild bitch tellin lie
24782    ruffl ntac eileen dahlia beauti color combin p...
Name: processed_tweets, Length: 24783, dtype: object
```

Fig 14: Data after preprocessing

iv. Visualizations:

We display the most used word in the tweets, along with the most used words under hate and offensive categories. We will use the matplotlib library and wordCloud to display the required output. WordCloud arranges the words and their size in accordance with their frequency of occurrence.

Fig 15: Most used words in the dataset using wordCloud

v. Modelling:

We used the Ngram vectorization and Tf-Idf for feature generation. We convert the textual data to numerical data in these processes and find the accuracies of each method. In Ngram vectorization we generate phrases with 1,2 and 3 words because sometimes a phrase makes us understand the emotion of the sentence better than an individual word. For example, unbelievable is a neutral word but it could be paired with both positive and negative words like unbelievably good or unbelievably sad to express their emotion. Hence, we are generating multiple words which are better for training our model for better accuracy.

```
features=(ngram_vectorizer.get_feature_names())
```

Fig 16: Feature extraction using ngram vectorization

In Tf-Idf we will count the frequency of the words that appear in a document which is called the term frequency and the frequency of the words that appear in all documents which is called the inverse term frequency, using these two metrics we decide the popularity of the word.

```
tfidf = tfidf_vectorizer.fit_transform(dataset['processed_tweets'] ).toarray()
```

Fig 17: Feature extraction using Tf-Idf

We next train the logistic regression model with the ngram features extracted and find the performance metrics. Figures 18 & 19, display the code and accuracy for the features generated using ngram vectorization.

```
print("Accuracy Score:" , accuracy_score(y_test,y_preds))
```

Fig 18: Logistic Regression using ngram features

Fig 19: Evaluation metrics for the logistic regression

We next train the logistic regression model with the tf-idf features extracted and find the performance metrics. Figure 20, displays the code and accuracy for the features generated using Tf-Idf

```
Accuracy Score: 0.8973531310522918
```

Fig 20: Logistic Regression using Tf-Idf features

Since Tf-Idf model provides us with the highest accuracy we will use these features for our sentiment analysis.

Sentiment Analysis: We preprocess the tweets and use these tweets in sentiment analyzer to find their polarities. If the polarity of the given word is greater than 0 then it is a positive word, if the polarity of the word is less than zero then it is a negative word and if the polarity of the word is equal to 0 it is a neutral word. We find the polarities of each word and then find the compound polarity of the tweet and based on the value we categorize them.

Fig 21: Matrix generated by performing sentiment analysis

We append these sentiment analysis metrics to our dataset and use them in the logistic regression model for better accuracy. As you can find in Figure 22, the accuracy of the dataset with sentiment analysis values has increased its accuracy by 0.01% which is less but still worth considering.

Fig 22: Accuracy of the model after adding the polarity values from sentiment analysis

4. RESULTS

The generated confusion matrix of the Tf-Idf features with additional sentiment analysis features is shown below in Figure 23. The recall score for the Neutral class is 0.86, for Offensive it is 0.96 and 0.18 for Hate class. The misclassification on the top right corner is due to the similarity in the words of hate and offensive and shows that the model has performed poorly in classifying hate and offensive words.

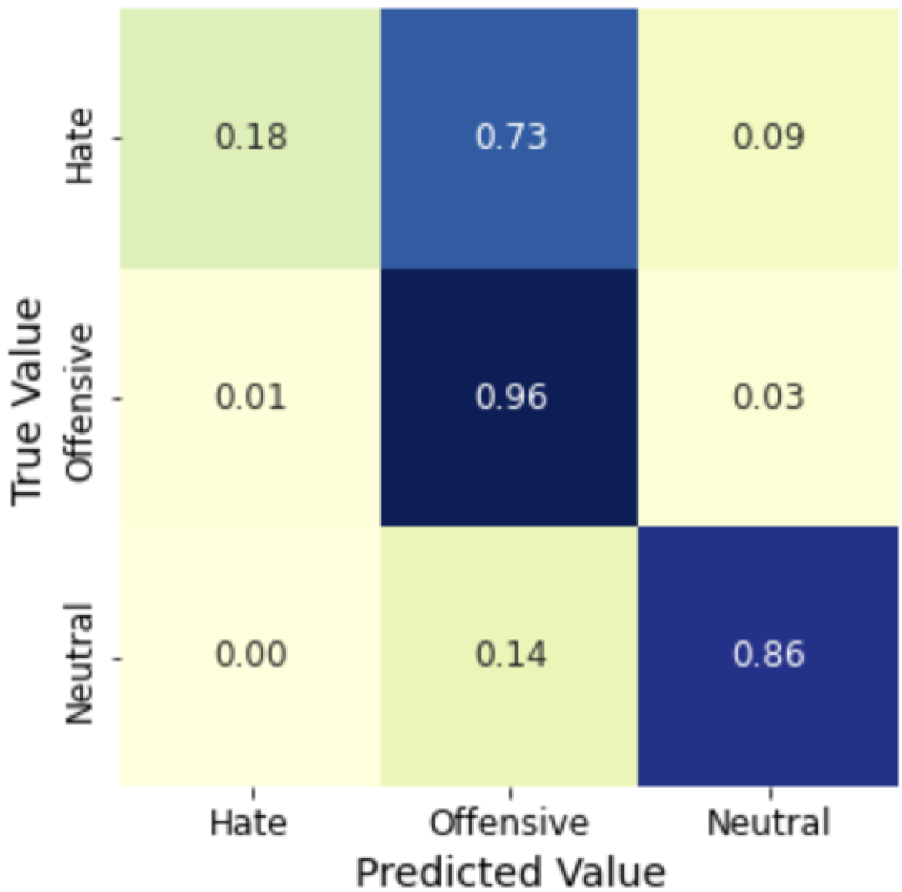


Fig 23: Confusion Matrix for hate text analysis

5. CONCLUSION

Here, by inferring the Confusion Matrix we can observe that the predicted value and true value of the text are coinciding with a percentage of 86 percent precision in detecting neutral words using the given dataset which represents the efficiency of our model. Simultaneously, while considering the predicted value and true value regarding the prediction of the offensive is attaining the maximum precision of 96 percent which is very helpful in detecting the offensive terms with more accuracy.

6. REFERENCES

[1] Davidson, Thomas, Dana Warmsley, Michael W. Macy and Ingmar Weber. "Automated Hate Speech Detection and the Problem of Offensive Language." ICWSM (2017).

[2] Schmidt, Anna, and Michael Wiegand. "A survey on hate speech detection using natural language processing." In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media, April 3, 2017, Valencia, Spain*, pp. 1-10. Association for Computational Linguistics, 2019.

[3] Hala Mulki, Hatem Haddad, Chedi Bechikh Ali, and Halima Alshabani. 2019. L-HSAB: A Levantine Twitter Dataset for Hate Speech and Abusive Language. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 111–118, Florence, Italy. Association for Computational Linguistics.

[4] Kocoń, Jan, Alicja Figas, Marcin Gruza, Daria Puchalska, Tomasz Kajdanowicz, and Przemysław Kazienko. "Offensive, aggressive, and hate speech analysis: From data-centric to human-centered approach." *Information Processing & Management* 58, no. 5 (2021): 102643. https://www.sciencedirect.com/science/article/pii/S0306457321001333

[5]https://towardsdatascience.com/my-first-machine-learning-project-designing-a-hate-speech-detecting-algorithm-56ab32f10833