International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Suicide Rate Prediction Using Machine Learning

Kanishka Khatavkar[1], Vaishnavi Bhosale[2], Dineshkumar P[3], Vaishnavi Phalke[4], Akshada Ghorpade[5]

[1,2,4,5]Department of Computer Science, & Engineering Yashoda Technical Campus Satara,Maharashtra-415015.
[3]Department of E & T Engineering, Yashoda Technical Campus, Satara, Maharashtra-415015

Abstract
Suicide is a growing public health concern with accurate prediction and prevention being crucial for saving lives. This project utilized advanced machine learning algorithms to predict suicide rates using a dataset of demographic, economic, mental-health and social factors. Our approach employs a combination of regression models, decision trees, Random Forest, multilayered perception to identify key predictors and patterns in suicide rates. Results show that our machine learning model achieves a high accuracy rate in predicting suicide rates, outperforming traditional statistical methods. Feature importance analysis reveals that economic factors, such as unemployment rates and income levels, are significant predictors of suicide rates. These findings have important implications for policy makers and mental health professionals, enabling targeted interventions and resource allocation to high-risk populations. Our study demonstrates the potential of machine learning in improving suicide prevention of machine learning in improving suicide prevention efforts and highlights the need for further research in this critical area.

Keywords: Suicide Rate Prediction, Machine Learning, Supervised Learning, Mental health services, public health interventions.

INTRODUCTION

Suicide is defined as death caused by self-directed injurious behavior with intent to die as a result of the behavior. A suicide attempt is a non-fatal, self-directed, potentially injurious behavior with intent to die as a result of the behavior. A suicide attempt might not result in injury. Suicidal ideation refers to thinking about, considering, or planning suicide.

Suicide rate prediction refers to the use of statistical models, machine learning algorithms, and data analysis techniques to forecast the number of suicides or suicide attempts in a populations over a specific period. The main objectives of this project is to leverage machine learning techniques to predict suicide rates based on various socio-economic, demographic, and mental health-related factors. It's Identifying high-risk individuals or groups enables targeted interventions, potentially preventing suicides. Predicting suicide rates helps optimize resource distribution, ensuring effecting support for those in need.

Use machine learning to create a model that predicts suicide rates based on factors like age, gender, income, and unployment . Attempt to predict the suicide rate using various regression algorithms such as Linear regression, decision trees, Random Forest, multilayered perception.

Linear regression is a machine learning technique used to predict suicide rates basecd on various factors. It works by creating a linear equation that best predicts the suicide rate based on the suicide rate based on

various factors. It works by creating a linear equation that best predicts the suicide rate based on the input features. The linear equation is used to non-linear regression techniques, such as decision trees, random forests, or neural networks, to capture complex relationships. Decision trees can be combined with other techniques, such as Random Forests, Gradient Boosting. By using decision trees and machine learning techniques, researchers can identify key factors contributing to suicide rates and develop targeted interventions to prevent suicide.

Random Forest are a machine learning technique used to predict suicide rates by combining multiple decision trees. Gather information on various factors such as demographic variables, mental health history, social support, and economic conditions. By using random forests and machine learning techniques, researchers can identify key factors contributing to suicide rates and develop targeted interventions to prevent suicide.

KNN is a popular supervised learning algorithm used for classification and regression tasks. It works by calculating the distance between the input sample and each training sample, selecting the K nearest neighbors, and then voting to determine the class label or regression value. There are two types of KNN: classification KNN for classification tasks and regression KNN for regression tasks. Various distance metrics can be used, including Euclidean, Manhattan, Minkowsk, and Hamming distance. KNN has several advantages, including being easy to implement, non-parametric and robust to noise. However, it can be computationally expensive, sensitive to the choice of K, and not suitable for high-dimensional data. Despite these limitations, KNN has many applications, including image classification, speech recognition, text classification , and recommendation systems. The algorithm has two key hyperparameters: K (the number of neighbors) and the distance metric. Additionally, weighting functions can be used to assign different importance to each neighbor. Overall, KNN is a versatile and widely used algorithm in machine learning.

METHODOLOGY

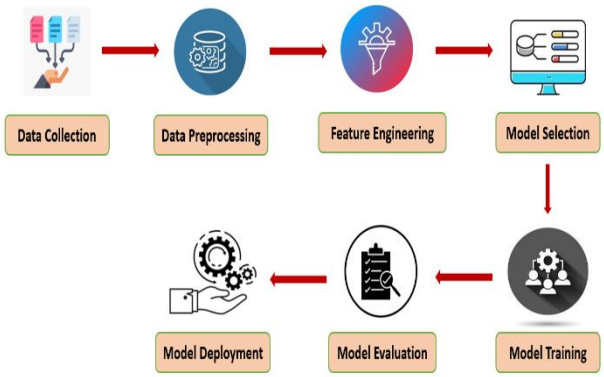


Fig. 1. Machine Learning Operation step-by-step procedure

A. Data Collection

The goal is to collect data that provides both historical suicide rates and various socio-economic andd demographic factors that could correlate with or contribute to suicide rates in India.

The primary source for historical suicide rates in India, containing detailed records categorized by demographic factors (age, gender, marital status) and socio-economic conditions. Census data or databases like those from the World bank or the Ministry of Statistic provide information on economic indicators

(e.g., unemployment rates, GDP per capita, literacy rates, population destiny).

B. Data Preprocessing

Handling Missing Values:

Identify missing values and decide on an imputation strategy. For example:

Mean or Median Imputation for numerical features.

Mode Imputation or predictive imputation (e.g., k- Nearest Neighbors) for categorical variables. If certain features or rows have excessive missing data, consider excluding them to maintain data quality.

Outlier Detection and Treatment: Use statistical methods like Z-scores or IQR(Interquartile Range) to detect outliers in numerical data.Consider techniques likes Winsorization or log transformations to treat outliers,as they can distort model training if not addressed.

Data Normalization or Standardization:

For features with varying scales, apply normalization (scaling between 0 and 1) or standardization(mean od 0 and standard deviation of 1) to ensure that all features contribute equally to the model.

Encoding Categorical Variables: Use coding techniques such as one-hot coding for categorical features, such as income levels, use ordinal encoding to maintain the rank order.

Data Aggregation: Aggregate data at a desired level, such as state or district level, and by time period (e.g., annual or monthly averages) to simplify analysis and focus on relevant trends.

C. Feature Engineering

Based on existing features, create new variables that could improve model interpretability. for example:

Population Density per Region: Calculated as population per area, which could indicate stress due to crowded living conditions.

Dependency Ratio: Ratio of dependent population to the working age population, potentially impacting economic stress.

Temporal Features: Create time-based features such as year-on-year change in economic indicators (e.g., change in unemployment rates) or seasonal indicators if monthly or quarterly data is available.

Features Transformation: Consider transforming certain features, such as taking the log of highly skewed features, to reduce skewness and improve model performance.

Correlation Analysis: Calculate correlation between features and the target variables (suicide rates) to identify key predictors.

Use domain knowledge or tools like heatmaps to visualize and select features with high predictive power.

Dimensionality Reduction: If you have high number of features, consider dimensionality reduction techniques like Principal Components Analysis (PCA) or feature selection algorithms to reduce model complexity.

D. Model Selection

Experiment with a range of models that suit the prediction task, including:

Linear Regression for simpler, interpretable models.

Decision Trees and Random Forests to handle non-linear relationships and capture feature interactions.

Gradient Boosting machines for high performance prediction.

If accessible, national health surveys (like the National Family Health Survey in India) or datasets on mental health conditions, substance abuse prevalence, and accessed to healthcare can provide additional insights into behavioral health aspects.

Discuss how data is collected from various sources and merged based on common field like year and geographic region. Also, mention any tools or programming libraries used for data extraction (e.g., web

scraping, APIs).

E. Model Training

Train each selection model on the training set. Use the validation set to fine-tune hyperparameters and evaluate model adjustments.

Apply regularization techniques (L1, L2) if the model is prone to overfitting, especially with high-dimensional data.

Ensemble Methods: Combine the predictions of multiple models using ensemble techniques like bagging, boosting, or stacking to improve overall predictive accuracy.

F. Model Evaluation

Evaluation Metrics: Choose metrics based on the prediction goal:

Regression Metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), or Root Mean Squared Error (RMSE) if predicting suicide rate directly.

Classification metrics like accuracy, precision, recall, F1-score, or AUC-ROC if the problem is framed as classification task (e.g., high-risk vs. low-risk regions).

Cross Validation: Evaluate models using k-fold cross-validation to ensure that is consistent across various splits of the data.

Model Interpretation: Identify the most influential feature in the best-performing models using features importance scores (e.g., SHAP values or feature importance in tree-bases models). Discuss the relationship between these features and suicide rates.

G. Model Deployment

Deployment Strategy: Outline the intended deployment environment (e.g., clous-based services like AWS or GCP) or local deployment if the model will be used within a specific organization.

API Creation: Create restful APIs or micro services that allow other applications or stakeholders to interact with model. This enables real-time or batch predictions for new data.

Model Monitoring: Establish Monitoring systems to track the model's performance post-deployment. This could include tracking prediction errors, re-evaluating the model periodically with new data and recalibrating as necessary.

Retraining the Maintenance: Describe the plan for retraining the model periodically, particularly as new data becomes available, to ensure accuracy remains high over time.

IMPLEMENTATION

There are 7 columns in our dataset. Some of the columns are numerical types which include State, Year, Type code, Type, Gender, Age group, etc. Total are categorical. Using various Machine Learning Algorithms, we want to investigate the potential triggers that could increase the risk if suicide in the societies. The Kaggle dataset we collected includes data from over 100 countries from 2012 to 2024. To make the study more insightful, we agreed to reduce the number of nations. We selected 40 countries from around the world that we believe are good sample of various regions.

The main components of our dataset now that we have it is selecting the features that have the most differentiating effect. This is essentially Feature engineering for training accurate prediction. Collinearity refers to the fact that certain features are extremely similar to nan output class. We will use exploratory data analysis to understand the provided features before selecting the main features.

A. Exploratory analysis for feature selection

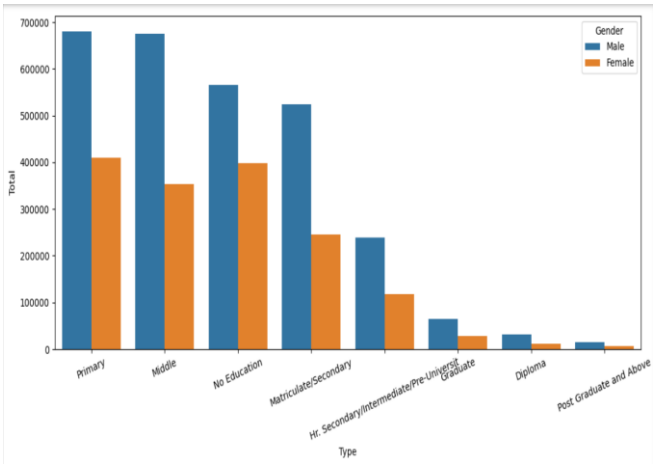


Fig.2. Count of Suicide by Type

The figure underscores the importance of understanding suicide types to address root causes, prioritize resource allocation, and design preventive measures that focus on high-risk areas.

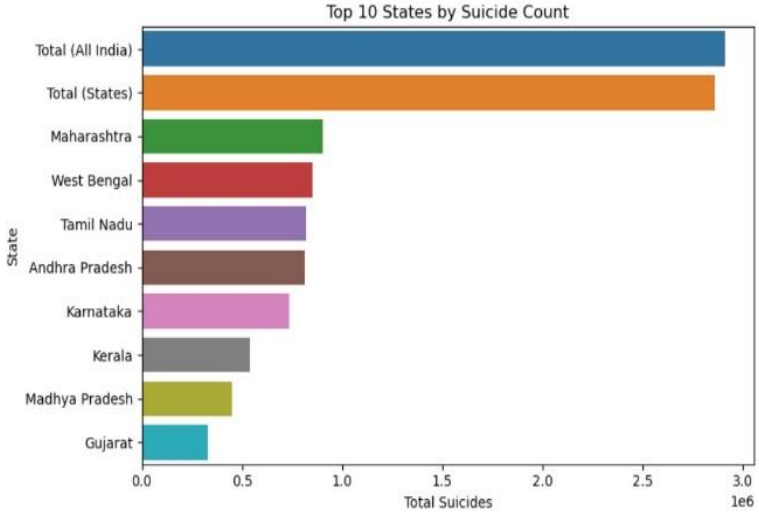


Fig 3. Count of Suicide by States

This figure shows the top 10 states in India by suicide counts, along with the total figures for all of Indian.

1. Maharashtra leads among individual states with the highest number of suicides, followed by West Bengal and Tamil Nadu.
2. The top 3 states (Maharashtra, West Bengal, and Tamil Nadu) appear to have significantly higher counts compare to the lower rank states.
3. Gujarat has the lowest count among the top 10 states shown .
4. There seems to be a considerable gap between the total national figure and the sum of state figure

This data suggest a need for focused mental health interventions, particularly in starts with higher counts Urban pressures may be a contributing factor given the high numbers in status with major metropolitan areas the significant variation between states indicates a need for state specific approaches to suicide prevention. The data points to the importance of understanding regional factors that might contribute to these differences .

It's crucial to note that absolute numbers don't tell the complete story – per capita rates would provide better insight into the relative severity across states with different population size. Also ,this type of sensitive data should be use constructively to inform prevention strategies and support system.

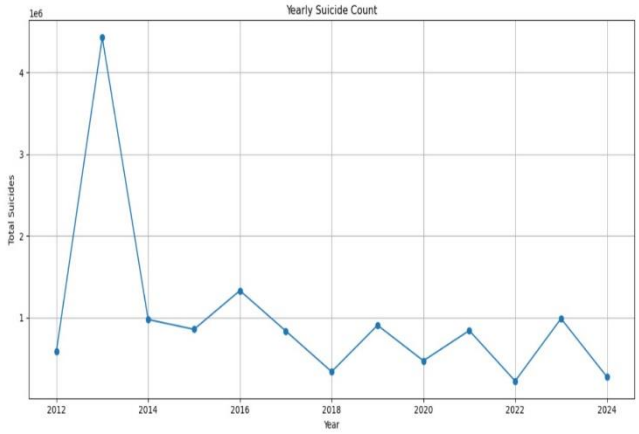


Fig.4 Count of suicide by Year

B. Linear Regression Model

Imports & Setup

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import OneHotEncoder
from sklearn.metrics import (
    mean_squared_error,
    mean_absolute_error,
    r2_score,
    accuracy_score,
    precision_score,
    recall_score,
    f1_score
)
```

These imports bring in necessary libraries for data manipulation , machine learning and metrics calculations .

Data Preprocessing

```
categorical_features = ['Type', 'Age_group']
encoder = OneHotEncoder(sparse_output=False, handle_unknown='ignore')
encoder.fit(data_new[categorical_features])
encoded_features = encoder.transform(data_new[categorical_features])
```

Identifies categorical columns (‘Type’ and ‘Age_group’) Creates a OneHotEncoder to convert categorical variables into numeric format. Fits and transforms the categorical data.

Feature Engineering

```
categorical_features = ['Type', 'Age_group']
encoder = OneHotEncoder(sparse_output=False, handle_unknown='ignore')
encoder.fit(data_new[categorical_features])
encoded_features = encoder.transform(data_new[categorical_features])
```

Creates a DataFrame from encoded categorical features. Separates numerical features(dropping categorical columns and target ‘Total’).Combines numerical and encoded categorical features. Sets the target variable ‘y’ as the ‘Total’
column.

Train-Test Model

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=10)
```

Splits the data into training(80%) and testing (20%)

Model Training

```
lr = LinearRegression()
lr.fit(x_train, y_train)
```

```
y_pred = lr.predict(x_test)
```

Creates a Linear Regression model. Trains it on the training data. Makes prediction on the test data.

Binary Classification Conversion

```
median = np.median(y_true)
y_true_binary = (y_true > median).astype(int)
y_pred_binary = (y_pred > median).astype(int)
```

Calculate the median of actual values. Converts both actual and predicted values to binary (0/1) based on whether there are above the median. This effectively turns the regression problem into a classification problem.

Classification Metrics Calculation

```
classification_metrics = {
    'accuracy': accuracy_score(y_true_binary, y_pred_binary),
    'precision': precision_score(y_true_binary, y_pred_binary),
    'recall': recall_score(y_true_binary, y_pred_binary),
    'f1': f1_score(y_true_binary, y_pred_binary)
}
```

The final results show the after converting the regression predictions to binary classifications, the model achieves:

- 75% Accuracy
- 67% Precision
- 83% Recall
- 74% F1Score

C. Linear Regression Model

Import Libraries

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, r2_score
```

These imports provide necessary tools for model building, preprocessing and evaluation.

Train-Test Split

```
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
```

Splits data into 80% training and 20% testing sets

Model Training with Timing

```
from time import time
start_time = time()
rf_model.fit(X_train, y_train)
training_time = time() - start_time

print(f"\nTraining time: {training_time:.2f} seconds")
```

Trains the model and measures training time

Calculate Performance Metrics

```
metrics = {
    'R2 Score': r2_score(y_test, y_pred),
    'Accuracy': accuracy_score(y_test_binary, y_pred_binary),
    'Precision': precision_score(y_test_binary, y_pred_binary),
    'Recall': recall_score(y_test_binary, y_pred_binary),
    'F1 Score': f1_score(y_test_binary, y_pred_binary)
}
```

The final result show that after converting the regression predictions to binary classifications, the model achieves;

- 82% R2Score
- 74% Accuracy
- 63% Precision
- 97% Recall
- 76% F1Score

D. Decision Tree Model

Import Libraries

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, r2_score
```

Imports all necessary libraries and identifies categorical columns. Coverts categorical variable into numeric format using one-hot encoding.

Train-Test Split

```
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
```

Splits data into 80% training and 20% testing sets.

Key hyperparameters are

- max_depth=10 prevents overfitting
- min samples split=5 ensures enough samples spliting
- min samples leaf= 2 ensures minimum samples in leaf nodes

Model Training with Timing

```
from time import time
start_time = time()
dt_model.fit(X_train, y_train)
training_time = time() - start_time

print(f"\nTraining time: {training_time:.2f} seconds")
```

Trains model and measures execution time and generates predictions on test data.

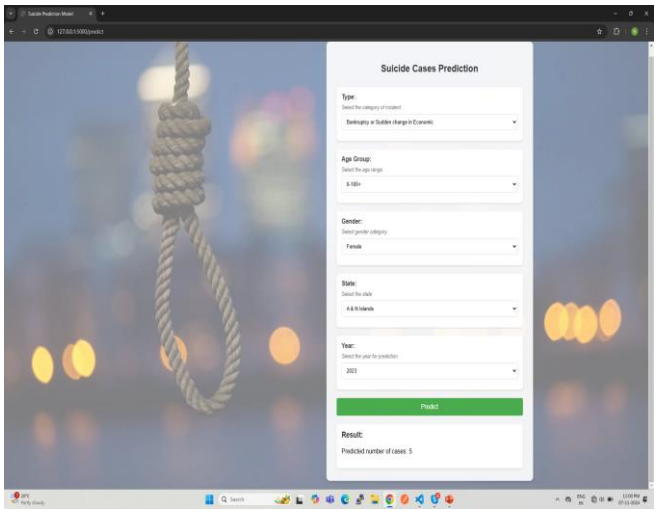
Calculate Performance Metrics

```
metrics = {
    'R2 Score': r2_score(y_test, y_pred),
    'Accuracy': accuracy_score(y_test_binary, y_pred_binary),
    'Precision': precision_score(y_test_binary, y_pred_binary),
    'Recall': recall_score(y_test_binary, y_pred_binary),
    'F1 Score': f1_score(y_test_binary, y_pred_binary)
}
```

The final results show that after converting the regression
Predictions to binary classifications, the model achieves:

- 83% R2Source
- 42% Accuracy
- 42% Precision
- 100% Recall
- 59% F1Source

IV. RESULT AND DISCUSSION



CONCLUSION & FUTURE SCOPE

- Linear regression can provide an initial understanding of how certain factors relate to suicide rates, but it is not ideal for precise prediction in complex, multi-dimensional problems like this
- Decision trees offer more flexibility than linear regression and provide insights into how different features contribute to predictions,but regularization methods like pruning are necessary to avoid owerfitting.
- Random forest is the most reliable and accurate model as of now for predicting suicide rates in this project .it performes well across different datasets and handles complex relationship between features efficiently .
- The final take away from this project is the working of different machine learning models on a dataset and understanding their parameters. Creating this notebook helped me to learn a lot about the parameters of the models, how to tune them and how they affect the model perforamance.The final conclusion suicide dataset are that the irrespective of age group and generation,male population are more prone to commit suicide than female .The future scope of suicide rate prediction using machine learing is vast, and as technology ,data.and ethical standards evolve,these algorithms will play a crucial role in suicide prevention .with a focus on improving the precision ,adaptability,and ethical consideration of these models,there is great potenstial to save lives,provide personalized care,and guide mental health professionals in more effective and timely interventions.

REFERENCES

1. Gazi Md.Omar Fraque, Mohammad Arman Jawad. “Sucide Rate Prediction Using Machine Learning Approch”. MAT journals February 19, 2024.
2. Shreya Gopal Sundari. Sucide Rate Prediction With Machine Learning. 2016
3. Hassane Djamous Hamid, Dr.k.Santhi Sree. “Detecting Sucidal Tendency Using Machine Learning”. IJSRT 6 June 2021
4. Vishnu Kumar, Kristin k. Sznajder and Sounder Kumara. Machine Learning Based Sucide Rate Prediction And Developed Of Sucide Vulnerability Index For Us Counties. Npj mental health research, 2022.

5. Alessandro Pigoni, Giuseppe Delvcchio, Nunzio Turpunici, Bomenico Madonna, Pietro Pietriani, Lukca Cecchetti and Paolo Brambilla. Machine Learning And Prediction Of Suicide In Psychiatric Population: Symetric Review. springer nature, 09 March 2024
6. Jeonjyoon Lee, Tae-Young pak. Machine Learning Prediction Suicidal Ideation, Planning, and Attempt Among Korean Adults: A Population Based Study. Elsevier 13 September 2024
7. Iqbal H.Sarker, Machine Learning: Algorithm real World Application Research Direction. Springer nature 2021
8. Imran Amim, Sobia Sayed. "Prediction Of Suicide Causes In India Using Learning" Springer nature, 2017
9. Vishnu Kumar, Kristin K.Sznajder And Soundar Kumara. "Machine Learning Based Prediction And Development Of Suicide Vulnerability Index for Us Contities", npi Mental health research, 11 Nov 2022
10. Alessandro Pigoni, Giuseppe Delvecchio, Nunzio Turtuci Domenico Madonna, Pietro Pietrini, Luca Cecchetti And Paolo Brambilla. Machine Learning And The Prediction Of Suicide In Psychiatric Populations: A Systimatic Review. 2024
11. Ayushi Chahal, Preeti Gulia. Machine Learning And Deep Learning". IJITEE, Octomber 2019
12. Jia -qi Sun and Seok-Pil Lee.Query by Singing /Humming System based on deep learnig.Research India Publications, International Journal of Computer Applications, 2017.
13. Q Liang H A Tvete and H W Brinks, Herbrt .B L Putro , R R J Putra , W Setiawan , E W S Rochman , B D Satato.Machine Learning From Theory To Algorithm. 2018
14. Santosh Kumar Behera,Mitali M Nayak.Ntural language Processing :A Review Paper.