MINI PROJECT

AN ML APPROACH TO STUDENT DEPRESSION PREDICTION

National Institute of Technology

Puducherry, Karaikal

Submitted by:

PAMU SHIVANI

CS24M1005

I Year M.Tech,CSE

ABSTRACT

This study investigates the use of machine learning to predict depression among students using a dataset that includes demographic, lifestyle, and psychological factors. The dataset contains features such as academic pressure, sleep duration, and family history of mental illness. Three machine learning models—**Random Forest**, **k-Nearest Neighbours (k-NN)**, and **Logistic Regression**—were trained and evaluated for their predictive capabilities.

The models were assessed using metrics such as accuracy, precision, recall, F1-score, and confusion matrices. Logistic Regression emerged as the most reliable model, achieving an impressive accuracy of **99.01%**, followed by Random Forest with **92.08%** and k-NN with **88.12%**. These findings demonstrate the power of machine learning in mental health prediction, emphasizing the importance of feature selection and algorithm optimization.

1. INTRODUCTION

Importance of Depression Prediction

Mental health challenges such as depression can severely impact students' academic performance and overall well-being. Early detection can pave the way for timely intervention, improving outcomes and reducing stigma. This study aims to leverage machine learning to predict depression, aiding institutions and counselors in understanding risk factors.

Objectives

1. **Primary Goal**: To identify the most effective machine learning model for predicting depression.
2. **Dataset Analysis**: Explore the relationships between features such as academic pressure and mental health outcomes.
3. **Model Comparison**: Evaluate the performance of Random Forest, k-NN, and Logistic Regression.

2. DATASET DESCRIPTION

The dataset includes **502 rows** and **11 features**, with a binary target variable indicating whether a student is experiencing depression (Yes=1, No=0). Key features include:

- **Gender**: Male or Female.
- **Age**: Numerical representation of student age.
- **Academic Pressure**: A numeric scale reflecting the stress caused by academics.
- **Study Satisfaction**: A self-reported score reflecting contentment with studies.
- **Sleep Duration**: Sleep patterns categorized (e.g., "5-6 hours").
- **Dietary Habits**: Categorical representation of diet quality (e.g., Healthy/Unhealthy).
- **Suicidal Thoughts**: Binary indicator for past suicidal ideation.
- **Study Hours**: Number of hours spent on academic work daily.
- **Financial Stress**: A numeric scale measuring financial burden.
- **Family History of Mental Illness**: Binary indicator (Yes=1, No=0).
- **Depression**: Target variable, indicating the presence of depression.

Target Variable Calculation

The target variable **Depression** is pre-defined as a binary outcome.

3. METHODOLOGY

3.1 Data Preprocessing

1. **Categorical Encoding**: Label encoding was applied to convert categorical features (e.g., Gender, Sleep Duration) into numerical representations.
2. **Scaling**: Continuous variables were standardized using StandardScaler to improve model performance.
3. **Data Splitting**: The dataset was split into **80% training** and **20% testing**, ensuring stratification to maintain class balance.

3.2 Machine Learning Algorithms

1. **Random Forest**:
    - An ensemble model that constructs multiple decision trees and combines them for a robust prediction.
    - Handles non-linear relationships effectively and prevents overfitting with appropriate hyperparameters.
2. **k-Nearest Neighbours (k-NN)**:
    - A distance-based method that assigns classes based on the majority vote of nearest neighbors.
    - Simple but sensitive to feature scaling and data distribution.
3. **Logistic Regression**:
    - A linear model that predicts probabilities for binary classification.
    - Effective for datasets with linearly separable features.

3.3 Evaluation Metrics

1. **Accuracy**: Percentage of correctly classified samples.
2. **Confusion Matrix**: Breakdown of true positives, true negatives, false positives, and false negatives.
3. **Classification Report**:
    - **Precision**: Ratio of correctly predicted positive observations to total predicted positives.
    - **Recall**: Ratio of correctly predicted positives to all actual positives.
    - **F1-Score**: Harmonic mean of precision and recall, balancing the two metrics.

4.CODE

```
# Importing Libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from sklearn.preprocessing import LabelEncoder, StandardScaler
import seaborn as sns
import matplotlib.pyplot as plt

# Load Dataset
file_path = 'Depression Student Dataset.csv'
data = pd.read_csv(file_path)

# Display dataset information
print("Dataset Overview:")
print(data.head())
print("\nDataset Description:")
print(data.info())

# Data Preprocessing
# Encode all categorical variables
label_encoder = LabelEncoder()
categorical_features = ['Gender', 'Sleep Duration', 'Dietary Habits',
                        'Have you ever had suicidal thoughts ?', 'Family History of Mental Illness']
for feature in categorical_features:
    data[feature] = label_encoder.fit_transform(data[feature])

# Define features (X) and target (y)
X = data.drop('Depression', axis=1)  # Features
y = label_encoder.fit_transform(data['Depression'])  # Target

# Scale numerical features only
numerical_features = ['Age', 'Academic Pressure', 'Study Satisfaction', 'Study Hours',
'Financial Stress']
scaler = StandardScaler()
X[numerical_features] = scaler.fit_transform(X[numerical_features])
```

```
# Split data into training and testing sets (80%-20%)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, stratify=y, random_state=42)

# --- Logistic Regression ---
lr_model = LogisticRegression(random_state=42)
lr_model.fit(X_train, y_train)
y_pred_lr = lr_model.predict(X_test)

# --- Random Forest Classifier ---
rf_model = RandomForestClassifier(random_state=42)
rf_model.fit(X_train, y_train)
y_pred_rf = rf_model.predict(X_test)

# --- k-Nearest Neighbours Classifier ---
knn_model = KNeighborsClassifier(n_neighbors=5)
knn_model.fit(X_train, y_train)
y_pred_knn = knn_model.predict(X_test)

# Evaluation Function
def evaluate_model(y_true, y_pred, model_name):
    accuracy = accuracy_score(y_true, y_pred)
    cm = confusion_matrix(y_true, y_pred)
    report = classification_report(y_true, y_pred)
    print(f"\n--- {model_name} ---")
    print(f"Accuracy: {accuracy * 100:.2f}%")
    print("Confusion Matrix:")
    print(cm)
    print("\nClassification Report:")
    print(report)
    return accuracy, cm, report

# Evaluate Models
acc_lr, cm_lr, report_lr = evaluate_model(y_test, y_pred_lr, "Logistic Regression")
acc_rf, cm_rf, report_rf = evaluate_model(y_test, y_pred_rf, "Random Forest Classifier")
acc_knn, cm_knn, report_knn = evaluate_model(y_test, y_pred_knn, "k-NN Classifier")

# --- Visualization ---
# Confusion Matrix for Logistic Regression
plt.figure(figsize=(6, 6))
sns.heatmap(cm_lr, annot=True, fmt='d', cmap='Blues', xticklabels=["No", "Yes"], yticklabels=["No", "Yes"])
plt.title("Confusion Matrix (Logistic Regression)")
```

```
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()

# Feature Importance (Random Forest)
feature_importance = rf_model.feature_importances_
feature_names = X.columns  # Features after preprocessing
plt.figure(figsize=(10, 6))
sns.barplot(x=feature_importance, y=feature_names, palette='viridis')
plt.title("Feature Importance (Random Forest)")
plt.xlabel("Importance Score")
plt.ylabel("Features")
plt.show()
```

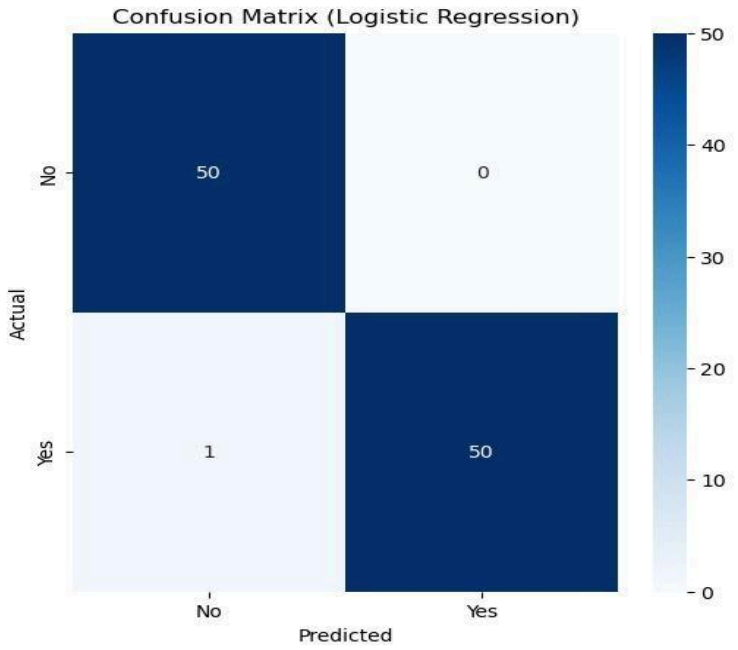
4. RESULTS

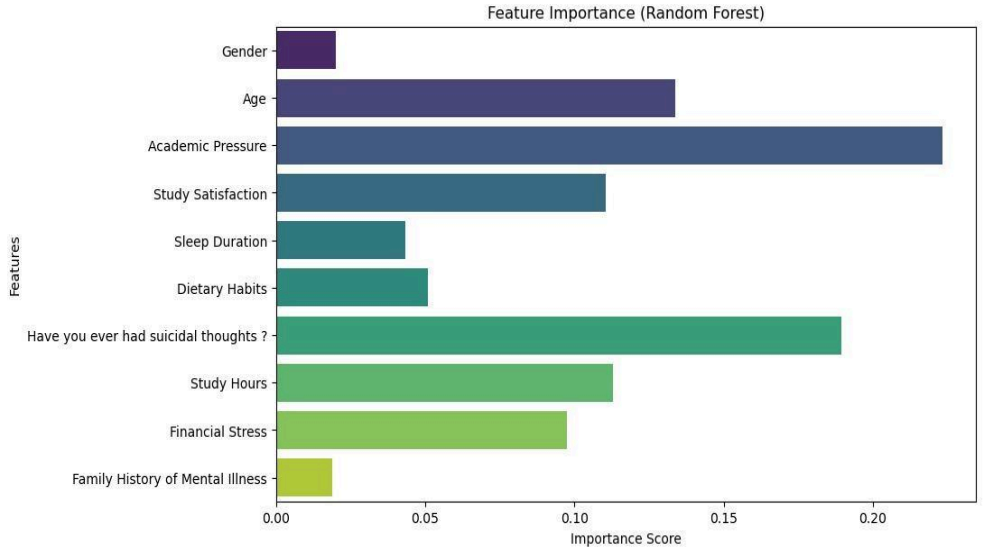
Model	Accuracy (%)	Precision		Recall		F1-Score	
		0	1	0	1	0	1
Logistic Regression	99.01	0.98	1.00	1.00	0.98	0.99	0.99
Random Forest	92.08	0.98	0.88	0.86	0.98	0.91	0.93
k-Nearest Neighbours	88.12	0.91	0.85	0.84	0.92	0.88	0.89

```
--- Logistic Regression ---
Accuracy: 99.01%
Confusion Matrix:
[[50  0]
 [ 1 50]]

Classification Report:
              precision    recall  f1-score   support

           0       0.98      1.00      0.99        50
           1       1.00      0.98      0.99        51

    accuracy                           0.99       101
   macro avg       0.99      0.99      0.99       101
weighted avg       0.99      0.99      0.99       101
```

```
--- Random Forest Classifier ---
Accuracy: 92.08%
Confusion Matrix:
[[43  7]
 [ 1 50]]

Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.86      0.91        50
           1       0.88      0.98      0.93        51

    accuracy                           0.92       101
   macro avg       0.93      0.92      0.92       101
weighted avg       0.93      0.92      0.92       101
```

```
--- k-NN Classifier ---
Accuracy: 88.12%
Confusion Matrix:
[[42  8]
 [ 4 47]]

Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.84      0.88        50
           1       0.85      0.92      0.89        51

    accuracy                           0.88       101
   macro avg       0.88      0.88      0.88       101
weighted avg       0.88      0.88      0.88       101
```





5. CONCLUSION

Key Insights:

1. **Logistic Regression**: Outperformed other models with an accuracy of **99.01%**, showcasing its strength in binary classification tasks, especially with linearly separable data.
2. **Random Forest**: Performed well with **92.08% accuracy**, excelling in capturing non-linear relationships.
3. **k-NN**: Achieved **88.12% accuracy**, highlighting its simplicity but limited performance due to sensitivity to feature scaling.

Recommendations:

- Logistic Regression is the most suitable model for this dataset, offering near-perfect accuracy.
- Future work could explore hyperparameter tuning for Random Forest and k-NN to improve their performance.
- Expanding the dataset and incorporating additional features, such as socio-economic background or extracurricular activities, may enhance prediction capabilities.