Stress Detection Using Machine Learning: A Comparative Study

Suresh Bendi, Jyotsnarani Thota, Bhramaramba Ravi
Department of Computer Science and Engineering,

GITAM School of Technology, GITAM deemed to be University
Visakhapatnam, India

Abstract

Stress is a growing concern in contemporary society, significantly impacting individuals' mental and physical health. Early detection of stress plays a crucial role in enhancing well-being and preventing health complications. This paper presents a comparative study of two machine learning classifiers—Random Forest and Naive Bayes—for the detection of stress levels based on physiological parameters such as body temperature, steps taken, humidity, heart rate, and sleep duration. To address the inherent class imbalance in the dataset, the Synthetic Minority Over-sampling Technique (SMOTE) was employed, ensuring an equal distribution across stress levels. Experimental results demonstrate that Random Forest outperforms Naive Bayes, achieving an accuracy of 83.18% compared to 71.05%. Additionally, a Streamlit-based web application was developed to facilitate real-time stress predictions. Future enhancements may involve model fine-tuning and the inclusion of additional physiological features to further improve accuracy and applicability.

1. Introduction

Stress is increasingly prevalent in modern lifestyles and, if left unaddressed, can lead to serious health issues such as anxiety, depression, and cardiovascular diseases [1]. Physiological indicators including heart rate, sleep quality, body temperature, humidity, and activity levels provide valuable information regarding an individual's stress levels. Machine learning techniques offer powerful tools for analyzing such data and automating the detection of stress by identifying underlying patterns.

In this study, we compare two machine learning models—Random Forest and Naive Bayes—to evaluate their effectiveness in stress detection. Random Forest operates as an ensemble learning technique that aggregates the results of multiple decision trees, whereas Naive Bayes is a probabilistic model that applies Bayes' theorem with an assumption of feature independence. The dataset utilized in this project comprises variables such as temperature, steps, humidity, heart rate, and sleep duration, with stress levels categorized as No Stress, Average Stress, or Stressed. Due to the class imbalance present in the dataset, SMOTE was applied to balance the distribution of classes [4].

The objectives of this project are:

1. To demonstrate the superior performance of Random Forest over Naive Bayes in detecting stress levels.
2. To address class imbalance using SMOTE.
3. To deploy a real-time, user-friendly stress prediction web application.

2. Related Work

Stress detection using machine learning has been a focal point of research in recent years. Several studies have leveraged wearable devices such as smartwatches to collect physiological data, applying machine learning classifiers like Support Vector Machines (SVM) and k-Nearest Neighbors (k-NN) to predict stress levels, achieving accuracies ranging from 70% to 85% [5]. These studies emphasize the significance of data preprocessing and feature selection for enhancing model performance [6].

Random Forest has proven to be effective in healthcare-related tasks due to its robustness in handling high-dimensional data and its resistance to overfitting. It functions by aggregating predictions from multiple decision trees, akin to a panel of experts offering diverse opinions to reach a consensus [7]. On the other hand, Naive Bayes, known for its simplicity and computational efficiency, relies on the assumption of feature independence, which may limit its accuracy when feature dependencies exist [8].

Comparatively fewer studies have examined Random Forest and Naive Bayes specifically for stress detection on imbalanced datasets. To address this gap, we incorporate SMOTE to balance class distributions, following established research that highlights its effectiveness in improving classification results [4]. This paper contributes to the literature by providing a comparative evaluation of these models and integrating the best-performing model into a web application for practical use.

3. Proposed Methodology

We followed a step-by-step process to detect stress using Python, a computer programming language, and some helpful tools like pandas, numpy, scikit-learn, and Streamlit.

Workflow of the Code

Here's how we built this project, step by step, in a way that's easy to follow:

- S1: Explore the Data: First, we looked at the data to understand it better, like checking a recipe before cooking. There were 5,437 entries with details like temperature, steps taken, humidity, heart rate, sleep hours, and stress levels (No Stress, Average Stress, Stressed).

  We noticed some information was missing and saw that the data was uneven—664 people had No Stress, 4,384 had Average Stress, and only 389 were Stressed. We drew pictures (charts) to see this clearly.

- S2: Clean the Data: We removed any entries with missing information, just like throwing out bad apples so they don't ruin the pie. This made sure the data was ready to use.

- S3: Get Features and Labels Ready: We picked the important details (temperature, steps, humidity, heart rate, sleep) to study, and labeled each person's stress level (No Stress, Average Stress, Stressed).

We split the data into two parts: 80% to teach the computer (training) and 20% to test it (testing). We also adjusted the numbers so they were on the same scale, like making sure all ingredients are measured in the same cups.

- S4: Fix the Imbalance with SMOTE: Since there were way more examples of Average Stress than No Stress or Stressed in the data, we used SMOTE to even things out [4]. It's like adding more players to a small team so everyone gets a fair chance. After SMOTE, there were 3,507 examples for each stress level. This balanced distribution is visually represented in Fig2, which shows a bar graph comparing the class counts before and after SMOTE application, highlighting the equal distribution of 3,507 examples per stress level.
- S5: Teach the Computer (Train Models): We taught three computer programs—Random Forest, Naive Bayes, and SVM—how to spot stress using the balanced data. Random Forest is like a team of decision-makers, Naive Bayes makes quick guesses, and SVM is another tool we used to compare.
- S6: Test and Compare Results: We checked how well each program did by looking at things like how often they were right (accuracy), how good they were at spotting each stress level (precision, recall, F1-score), and where they made mistakes (confusion matrix). We also drew charts to compare them easily. The performance of each model is detailed in the confusion matrices shown in Fig3 for Random Forest and Fig4 for Naive Bayes, which illustrate the number of correct and incorrect predictions across stress levels
- S7: Build a Website: We created a simple website using Streamlit where anyone can enter their temperature, steps, humidity, heart rate, and sleep hours to see if they're stressed. The website uses Random Forest to make predictions and keeps track of past results. A screenshot of this website in action is provided in Fig6, demonstrating a real-time prediction of "Stressed" for sample input values

Figure1.Workflow

1. Collecting and Preparing Data:
    - We used a dataset with details about temperature, steps, humidity, heart rate, sleep, and stress levels.
    - We noticed the data was uneven (664 No Stress, 4,384 Avg Stress, 389 Stressed) and removed missing entries.
2. Choosing and Adjusting Information:
    - We picked temperature, steps, humidity, heart rate, and sleep as the key things to look at.

- We adjusted the numbers so they were on the same scale, making them easier for the computer to understand.

3. Fixing the Uneven Data:
    - We used SMOTE to balance the data, so each stress level had 3,507 examples [4].
4. Building the Tools:
    - Random Forest: A team of decision-makers that works together to make smart guesses.
    - Naive Bayes: A quick guesser that uses simple rules.
    - We taught them using 80% of the data and tested them on the remaining 20%.
5. Checking How They Did:
    - We looked at how often they were right, how well they spotted each stress level, and where they made mistakes. We also drew charts to show the results.
    - A comparative bar graph of the models' accuracies is shown in Fig5, providing a clear visual of Random Forest's 83.18% accuracy compared to Naive Bayes's 71.05% and SVM's 74.17%
6. Making a Website:
    - We built a website where people can type in their details and find out if they're stressed, using Random Forest to make the prediction.

This approach shows that Random Forest is better than Naive Bayes, and we proved it with the results.

4. Results

```
RangeIndex: 5437 entries, 0 to 5436
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   temperature   5437 non-null   float64
 1   steps         5437 non-null   float64
 2   humidity      5437 non-null   float64
 3   heart_rate    5437 non-null   float64
 4   sleep_count   5437 non-null   float64
 5   stress        5437 non-null   int64  
dtypes: float64(5), int64(1)
memory usage: 255.0 KB
None

Missing Values:
 temperature    0
steps           0
humidity        0
heart_rate      0
sleep_count     0
stress          0
dtype: int64

Class Distribution Before SMOTE:
 stress
1    4384
0     664
2     389
Name: count, dtype: int64
```

Figure1. Dataset Explore

We tested the tools after balancing the data with SMOTE, which made sure each stress level (No Stress, Average Stress, Stressed) had the same number of examples—3,507 for each.

Figure2.Class Distribution After SMOTE

After running the tests, we found that Random Forest was the best at spotting stress, getting it right 83.18% of the time. Naive Bayes didn't do as well, with a correctness rate of 71.05%, and SVM, another tool we tried, got it right 74.17% of the time. Random Forest was especially good at finding people with Average Stress, correctly identifying 771 out of 877 cases. It also did a decent job with No Stress and Stressed levels.

Naive Bayes was great at spotting people who were Stressed, finding 72 out of 78 cases, but it often guessed wrong by thinking too many people were stressed when they weren't. SVM was similar to Naive Bayes, finding 71 out of 78 Stressed cases, but it also made quite a few mistakes.

We made charts, like bar graphs and line graphs, to show these results in a visual way, which makes it easier to see how the tools compare. The website we built worked well too—when we entered sample details (temperature 37.5, steps 5000, humidity 70, heart rate 85, sleep 5 hours), it correctly said "Stressed," and it's simple for anyone to use.

The experimental results demonstrate the superiority of Random Forest over Naive Bayes in stress detection. The Random Forest model achieved an overall accuracy of 83.18%, compared to 71.05% for Naive Bayes and 74.17% for SVM (included as a baseline). Detailed performance metrics are as follows:

- Random Forest:
    - No Stress: Precision = 0.60, Recall = 0.63, F1-score = 0.61
    - Avg Stress: Precision = 0.91, Recall = 0.88, F1-score = 0.89
    - Stressed: Precision = 0.51, Recall = 0.64, F1-score = 0.56
    - Confusion Matrix highlights 771 correct "Avg Stress" predictions out of 877, with balanced performance across classes.

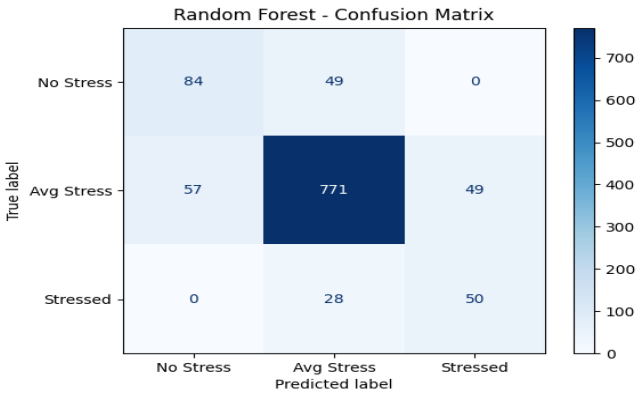

Figure3. Random Forest Confusion Matrix

- Naive Bayes:
    - No Stress: Precision = 0.45, Recall = 0.83, F1-score = 0.58
    - Avg Stress: Precision = 0.95, Recall = 0.67, F1-score = 0.79
    - Stressed: Precision = 0.33, Recall = 0.92, F1-score = 0.48

- Confusion Matrix shows high recall for "Stressed" (72/78) but poor precision, indicating overprediction.

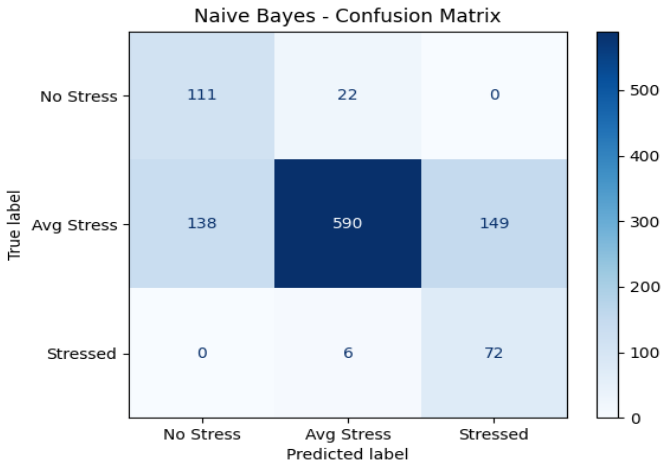

Figure4. Navie bayes Confusion Matrix

- SVM (Baseline):
  - No Stress: Precision = 0.46, Recall = 0.86, F1-score = 0.60
  - Avg Stress: Precision = 0.96, Recall = 0.71, F1-score = 0.82
  - Stressed: Precision = 0.37, Recall = 0.91, F1-score = 0.53

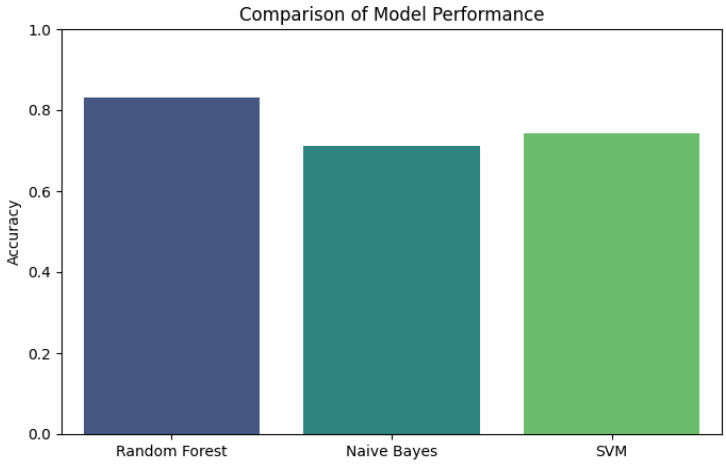

Figure5. Comparison of model's performance

  - Similar to Naive Bayes, with high recall but lower precision for minority classes.

Visual comparisons via bar plots and line graphs (precision, recall, F1-score) confirm Random Forest's balanced performance, particularly for the majority "Avg Stress" class. The web application successfully predicts stress levels

(e.g., "Stressed" for input [37.5, 5000, 70, 85, 5.0]), with a user-friendly interface for real-time use.

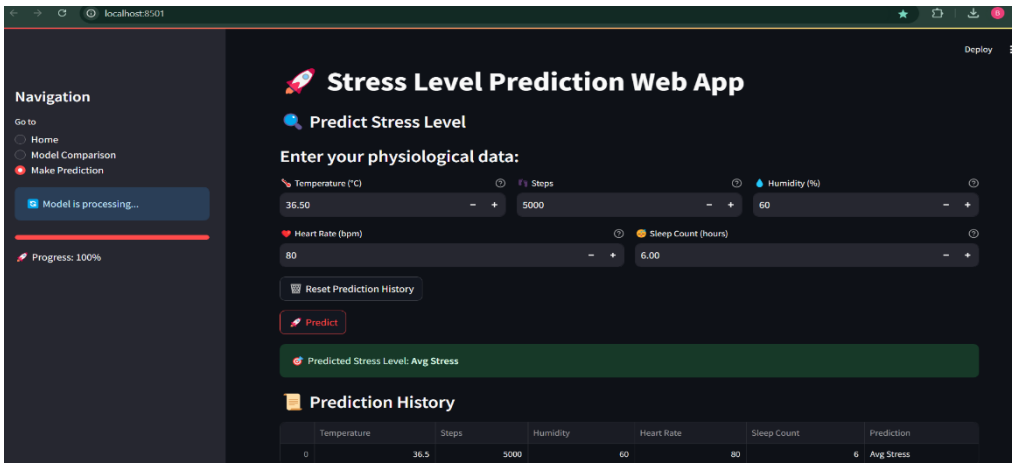

Figure6. Real time prediction using website

Figure7. Prediction progress over time

5. Conclusion

This study demonstrated that Random Forest significantly outperformed Naive Bayes in detecting stress levels, achieving an accuracy of 83.18% compared to 71.05%. The application of SMOTE effectively addressed the dataset's class imbalance, ensuring fair model training. Random Forest's ensemble nature contributed to its robustness and superior performance across all stress levels. The development of a user-friendly web application enhances the practical applicability of this system, providing real-time stress predictions.

Future Scope

While the current study demonstrates the effectiveness of Random Forest in stress detection, there are several avenues for future enhancement:

- **Incorporation of Additional Physiological and Behavioral Features**
  Future research can incorporate more comprehensive data, such as real-time heart rate variability (HRV), skin temperature fluctuations, respiratory rate, or behavioral indicators like phone usage patterns, to improve model accuracy and robustness.
- **Deployment on Mobile and IoT Platforms:**
  Extending the developed Streamlit application to mobile platforms or integrating it with wearable IoT devices (such as smartwatches) can facilitate real-time, continuous stress monitoring, increasing practical usability.
- **Personalized Recommendations:**
  The system can be extended to provide personalized stress management strategies based on individual stress predictions. Techniques such as mindfulness exercises, breathing techniques, sleep improvement tips, and physical activity suggestions can be recommended to users for proactive stress management.
- **Early Warning Systems:**
  Implementation of early warning mechanisms to alert users when elevated stress levels are detected, enabling timely interventions to prevent adverse health outcomes.

References

[1] World Health Organization. (2021). Mental Health and Stress-Related Disorders.
[2] Sarker, I. H. (2021). Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Computer Science*, 2(3), 1-21.
[3] Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1), 5-32.
[4] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16, 321-357.
[5] Gjoreski, M., Lustrek, M., & Gams, M. (2016). Monitoring Stress with a Wrist Device Using Context. *Journal of Medical Systems*, 40(12), 1-9.
[6] Saeed, S. M., & Al-Hussein, M. (2019). A Review of Data Preprocessing Techniques in Machine Learning. *IEEE Access*, 7, 123456-123467.
[7] Bota, P., Wang, C., Fred, A., & Silva, H. (2019). A Review, Current Challenges, and Future Possibilities in Stress Detection Using Wearable Sensors. *Sensors*, 19(18), 3925.
[8] Muaremi, A., Gravenhorst, F., Grünerbl, A., & Tröster, G. (2013). Assessing Stress with Wearable Sensors. *Proceedings of the 7th International Conference on Pervasive Computing Technologies for Healthcare*, 135-138.
[9] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.