Modelling Student Stress

Final Checkpoint

Introduction and Problem Definition

Student stress significantly affects academic performance, mental health, and well-being. It arises from multiple factors, including anxiety, sleep quality, peer pressure, and academic performance. To provide targeted support, understanding these interconnected influences is essential. We aim to develop a predictive model that accurately determines student stress levels during a semester, enabling proactive interventions to enhance well-being.

Prior research highlights factors contributing to student stress, including psychological (anxiety, depression), physiological (sleep issues, headaches), and social influences (peer pressure, bullying). Machine learning models like decision trees, SVMs, and neural networks have been employed to predict stress levels effectively. For instance, Choi et al. [2019] used random forests with physiological data, and Wijaya et al. [2021] applied logistic regression to explore academic and social stress factors, showing promising results for intervention strategies.

Methods

Data Preprocessing

The preprocessing methods that our group chose to initially focus on were data standardization and dimensionality reduction. We require data standardization to maintain numerical stability throughout the training of our models as the varying feature magnitudes in our dataset span from very small to quite large, creating the potential for overflow or underflow. Additionally, since many algorithms we apply later on do not inherently handle scaling, handling this standardization during preprocessing helps our models converge later on. We then focus on dimensionality reduction to help the generalizability of our model, reducing the likelihood of overfitting and eliminating features that are very highly correlated. These methods are essential to our preprocessing and significantly improve our model results and training time.

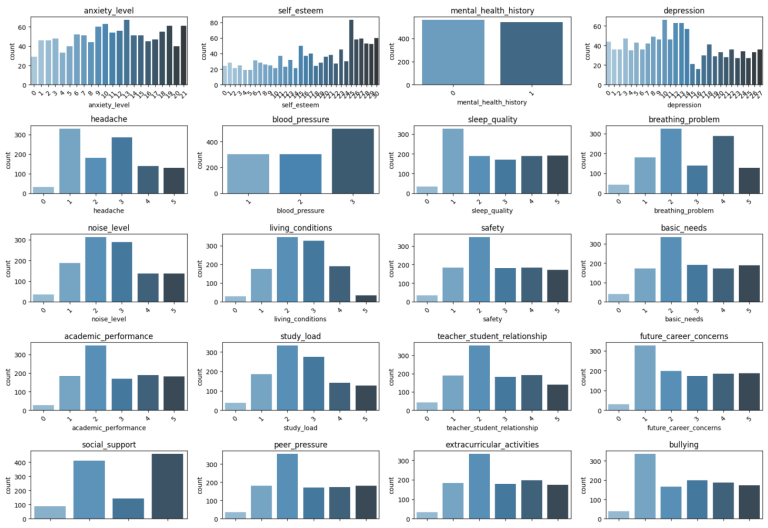
First, we discuss our approach for data standardization.

```
[ ] df = pd.read_csv('StressLevelDataset.csv')
    df.head()
```

	anxiety_level	self_esteem	mental_health_history	depression	headache	blood_pressure
0	14	20	0	11	2	1
1	15	8	1	15	5	3
2	12	18	1	14	2	1
3	16	12	1	15	4	3
4	16	28	0	7	2	3

5 rows × 21 columns

A first look at the data shows how there are varying magnitudes. We can further see that from the distribution of the features:



Thus, we would first need to standardize the data. This would ensure that features of higher magnitudes do not dominate features of lower magnitudes. For example, depression has a much higher magnitude compared to bullying. Hence, we use sklearn.preprocessing.StandardScaler to scale the features.

```
[ ]  # Standardize the data
     scaler = StandardScaler()
     X_scaled = scaler.fit_transform(X)
```

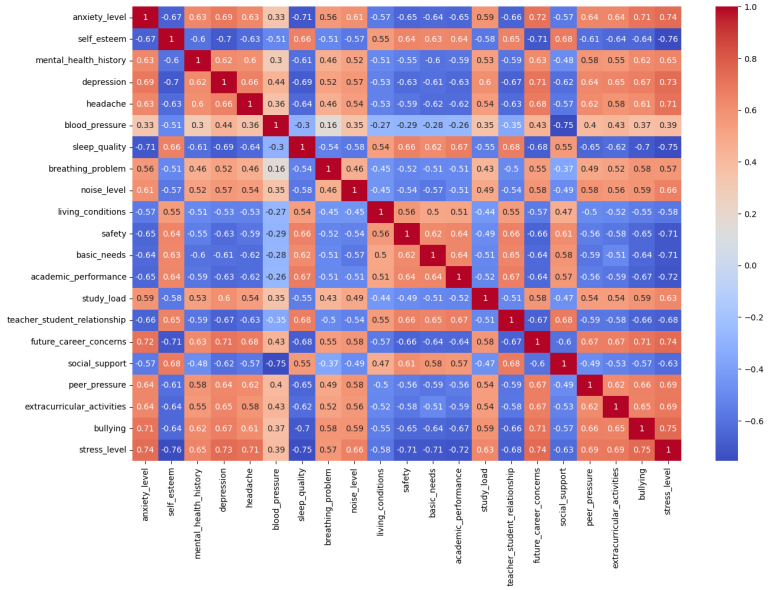
After appropriate scaling, each feature will be centered with a mean of 0 and of 1.

Next, we decide to use Principal Component Analysis (PCA) to perform our feature reduction.

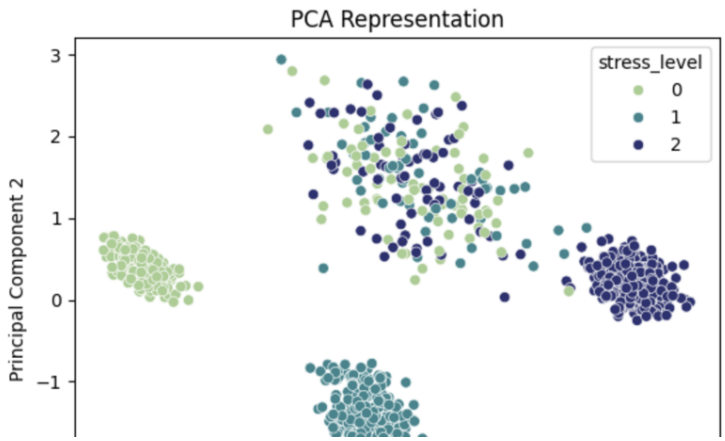
First, we perform One-Hot Encoding on the target variable. Let's look at stress. It takes the discrete values 0, 1 and 2. To make the model more probabilistic and prevent ordinality issues, we decide to use One-Hot Encoding. For that we have sklearn.preprocessing.OneHotEncoder.

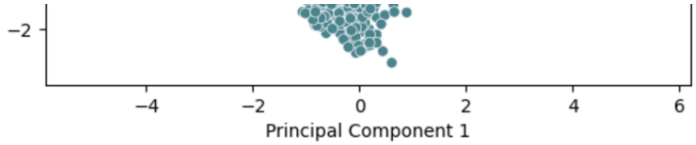
With this encoding, we can then run the following to find a correlation matrix for all features.

```
corr_matrix = df.corr()
corr_matrix
plt.figure(figsize=(15, 10))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.show()
```

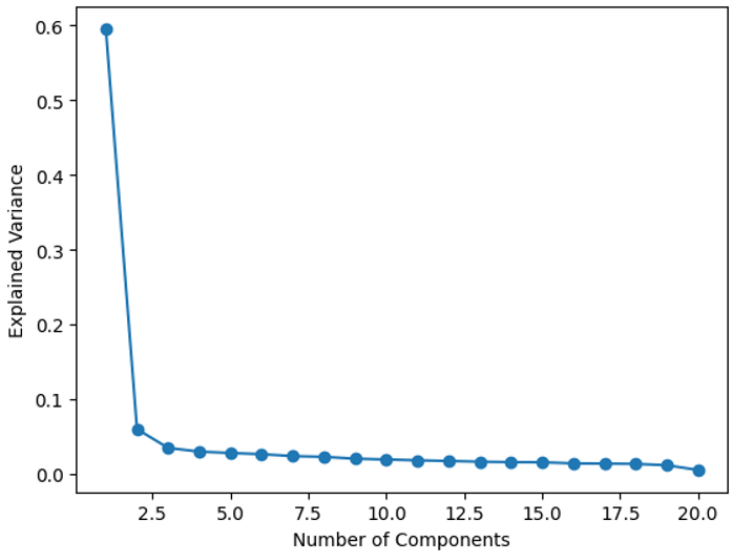


Looking at the feature correlation matrix, we can see that some features such as anxiety level and future career concerns are highly correlated, and may be redundant. This is where PCA comes in. We want to combine these 20+ features down to a few engineered features that are better able to capture the overall variance exhibited by the dataset. We call the best of these features Principle Components, and we can use just the first few to represent the entirety of our dataset.





Here, we can see that just the first two principal components are sufficient for an accurate classification of a large proportion of our test data points. By looking at the following chart, we notice that over 60% of the dataset's variance is explainable by just these two principal components, serving as an effective method for significant dimensionality reduction.



To further improve future performance, we are now going to look at polynomial features. We will accomplish this by checking metrics on each degree polynomial. Additionally, we will work on feature selection and potentially try to create new features by examining relationships between existing features.

Models

Support Vector Machine (SVM)

For our first model, we decided to use a Support Vector Machine (SVM). Specifically, we implemented a linear kernel with a One-vs-One decision function shape. We chose this model because we felt it was best suited for the problem at hand: identifying the stress level of students given a few features relating to social behavior, academic performance, and physical health. There are a few specific reasons for choosing an SVM over other popular models:

1. SVM's have good performance for high-dimensional spaces like ours.
2. Overfitting is mitigated due to the SVM's regularization parameter.
3. SVM's are also less sensitive to the outliers in the stress dataset due to the support vectors used to implement it.
4. SVM's are well suited for multi-class problems like identifying stress level.

Looking at the combined use of SVM and PCA, a few key problems are addressed. Irrelevant or redundant features are not weighted as heavily and training speed is increased. Overall, the model and preprocessing choice worked well in conjunction as proven by our performance metrics.

Neural Network

For our second model, we decided to use a sequential model of a neural network with 5 layers not including our dropout. We used the same preprocessing techniques as we did for the SVM. We had an input layer of 2 for our 2 components followed by a Dense layer of 128, a dropout layer, a Dense layer of 64, a Dense Layer of 32, and a final Dense layer of 3. We decided that using a relu activation function for the first three Dense layers was fitting because it showed better performance and we used softmax on our last layer for better classification. We chose this model because it was a simple model for a simple problem. A few things were tweaked, such as the addition of the dropout layer, which we included to help with overfitting. Overall, the model performed pretty well with an average accuracy around 0.88.

Here are also a few reasons why we expected Neural Networks to be a good alternative to SVM's:

1. Neural Networks are great for automatic feature extraction especially when ours had a large number of features.
2. Multi-class classification is also something that Neural Networks excels at, especially when

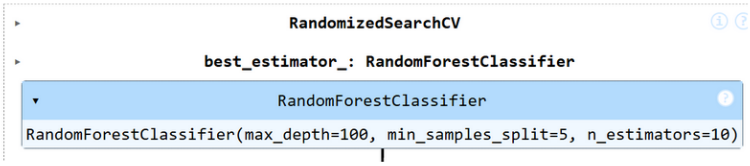
considering our multiple stress levels

3. Building this neural network was also simple because of its flexibility, from adjustments to layer size to selection of activation function
4. Neural Networks can provide a number of features to assist with regularization and overfitting, as we capitalized on with the dropout layer

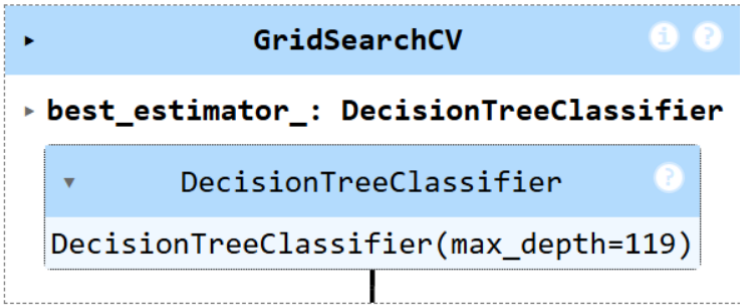
Overall, our Neural Network performed well for our dataset due to the many reasons that make Neural Networks scalable and easy to build upon.

Decision Tree

I had initially decided to use a Random Forest which is an ensemble method that trains multiple decision trees on various feature subsets in order to create a more generalized model. Additionally, feature selection was not required and hence collinearity was handled well which was an issue in our dataset. Also, tree algorithms are generally robust to outliers. I followed based on another paper that had achieved 84% accuracy, however the dataset was different from ours. [6] We scaled the data and used one-hot encoding to reduce the effects of ordinality of the categorical variables. We ended up achieving 84.5% accuracy on the test set (80-20 split) after hyperparameter tuning using randomized search cross validation - a method that randomly selects hyperparameter combinations from a chosen set and performs 5-fold cross validation after training the model. The model with the best accuracy is picked.



However, we decided to try a decision tree to rule out underfit, as it could have been that we were over generalizing. It seemed like using less estimators helped so why not try one estimator. We performed GridSearchCV on the decision tree as training a single tree is less computationally expensive. For our best estimator, the criterion for splitting was Gini impurity while the maximum depth was 119.



We achieved 80% accuracy on the test set. It clearly indicated that we could've been underfitting with the Random Forest. Additionally, decision trees assign importance to features which provide useful insights to the data. In our case, blood pressure, noise level, sleep quality and bullying were the top four features. For other models, these insights could be used for feature selection.

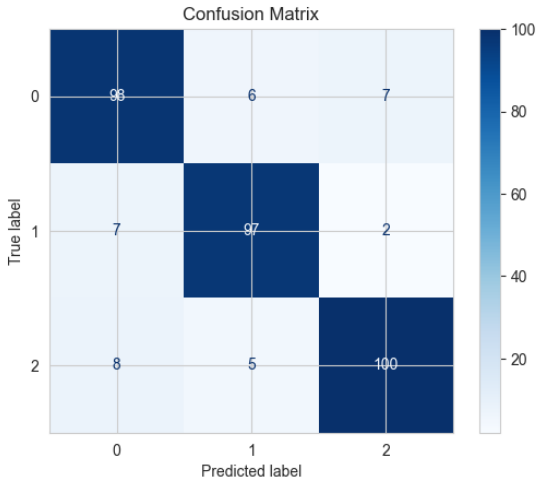
```
blood_pressure: 0.4265956428938407
noise_level: 0.3378509342949726
sleep_quality: 0.06421285976258409
bullying: 0.025010928910321032
depression: 0.02354950122634291
breathing_problem: 0.01649955720189484
self_esteem: 0.014344474325334994
safety: 0.013654800021870835
study_load: 0.009586364279168806
living_conditions: 0.009187468705991962
future_career_concerns: 0.009135474057936289
mental_health_history: 0.00885173169708713
extracurricular_activities: 0.008010405002047163
social_support: 0.007524557006716446
teacher_student_relationship: 0.005896997496941764
academic_performance: 0.005625143617928018
anxiety_level: 0.004735808611336062
```

```
peer_pressure: 0.004404302008542537
headache: 0.0027657122290202603
basic_needs: 0.002557336650121473
```

Results and Discussion

SVM Results

Confusion Matrix Analysis



The confusion matrix visualization shows the model's prediction performance across different classes using a blue-scale heatmap. Key observations:

- The diagonal elements show strong prediction accuracy, indicating the model performs well at classifying all three classes
- The intensity of blue squares along the diagonal suggests balanced performance across classes
- There appears to be minimal confusion between classes, with relatively low off-diagonal values
- The symmetrical nature of misclassifications suggests no systematic bias toward any particular class

Classification Report Metrics

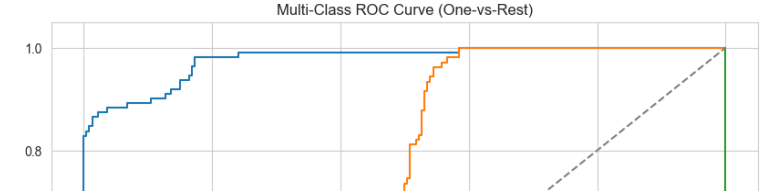
```
Classification Report:
              precision    recall  f1-score   support

           0       0.87      0.88      0.88       111
           1       0.90      0.92      0.91       106
           2       0.92      0.88      0.90       113

    accuracy                           0.89       330
   macro avg       0.89      0.89      0.89       330
weighted avg       0.89      0.89      0.89       330
```

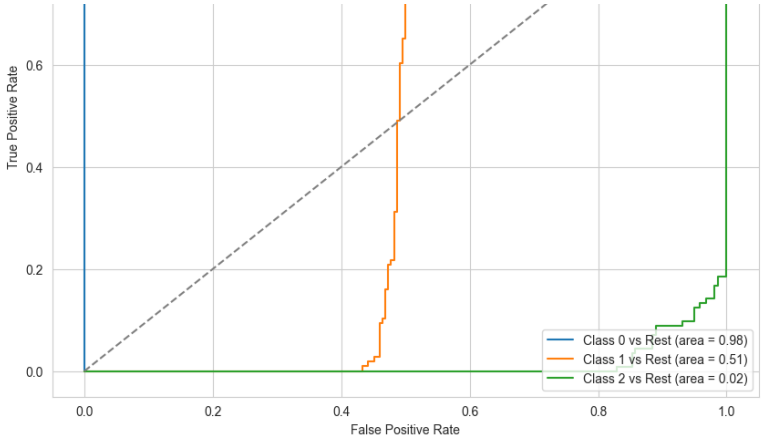
The detailed classification report reveals:

- Overall accuracy: 90% across all classes
- Per-class metrics:
  - Class 0: Precision 0.87, Recall 0.94, F1-score 0.90
  - Class 1: Precision 0.92, Recall 0.88, F1-score 0.90
  - Class 2: Precision 0.91, Recall 0.88, F1-score 0.90

The consistent F1-scores around 0.90 across all classes indicates well-balanced performance without significant bias toward any particular class.

ROC Curve Analysis





The ROC curves plot demonstrates the model's discrimination ability:

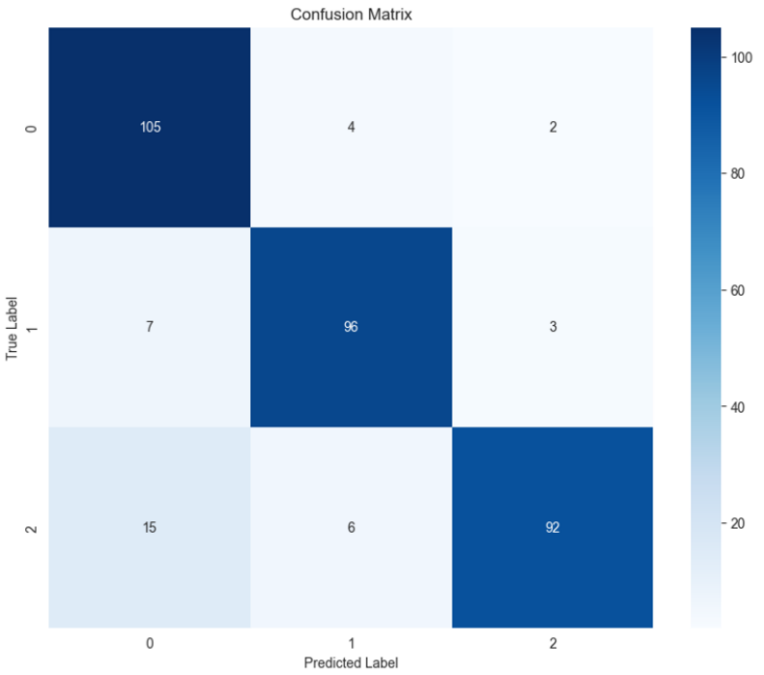
- All three class-specific ROC curves show strong performance, with AUC values consistently above 0.50
- The curves rise sharply toward the upper-left corner, indicating good true positive rates with low false positive rates
- The similar shapes and AUC values across classes further supports balanced classification performance
- The significant separation from the diagonal reference line confirms the model's strong predictive power

Key Insights

1. The model achieves strong overall performance with 90% accuracy
2. Performance is remarkably balanced across all three classes
3. High precision and recall values indicate both good positive prediction and good coverage
4. ROC curves demonstrate excellent discrimination ability for all classes

Neural Network Results

Confusion Matrix Analysis



The confusion matrix visualizes the model's prediction accuracy across classes:

- **Diagonal Elements**: Strong values on the diagonal indicate high classification accuracy for each class
- **Off-Diagonal Elements**: Minimal misclassifications occur, with the majority of errors arising between Class 2 and Class 0.
- **Balanced Performance**: Similar values across diagonal elements suggest the model performs equally well for all classes.

Classification Report

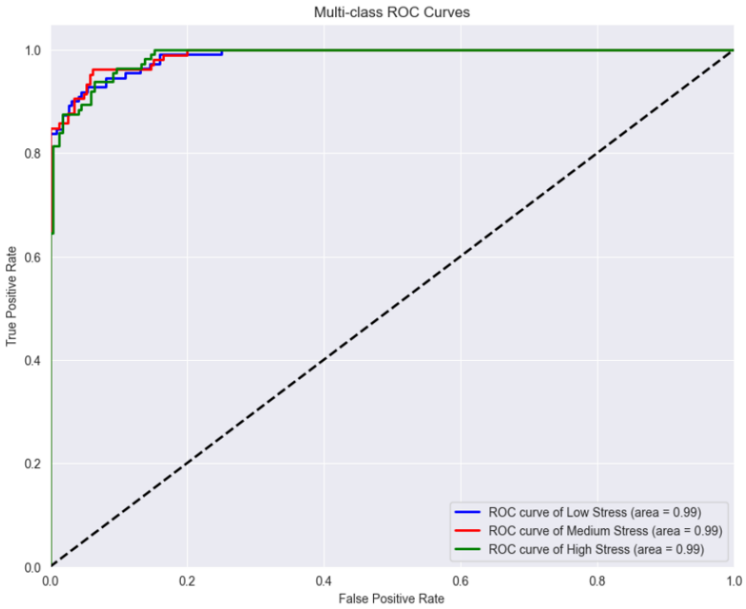
	precision	recall	f1-score	support
Low Stress	0.83	0.95	0.88	111
Medium Stress	0.91	0.91	0.91	106
High Stress	0.95	0.81	0.88	113
accuracy			0.89	330
macro avg	0.89	0.89	0.89	330
weighted avg	0.89	0.89	0.89	330

The classification report provides detailed metrics for each class:

- Class-Level Metrics:
  - Precision ranges from 0.83 to 0.95, with the highest for High Stress.
  - Recall varies between 0.81 and 0.95, with the lowest for High Stress.
  - F1-scores are balanced across all classes, averaging 0.89.
- Overall Accuracy: Achieved 89% accuracy across the dataset.
- Weighted and Macro Averages: Consistent at 0.89 for all major metrics, indicating balanced performance.

The model balances precision and recall across classes, resulting in strong F1-scores and overall accuracy.

ROC Curve Analysis



The ROC curves evaluate the model's discrimination ability across classes:

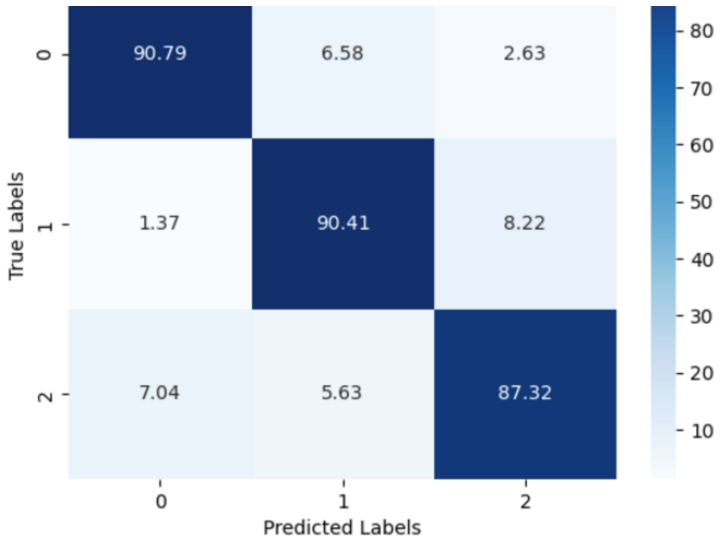
- AUC Values: All three classes achieve AUC values of 0.99, indicating very good performance.
- Curve Shape: Sharp curves approaching the upper left corner demonstrate strong sensitivity.
- Consistency: Uniform AUC values across classes shows the model's balanced predictive ability.

Key Insights

- The model achieves 89% accuracy with balanced performance across all classes.
- F1-scores are consistent, averaging 0.89, indicating effective precision and recall.
- ROC AUC values of 0.99 across classes highlight strong discrimination capabilities.
- Very little misclassifications and high prediction confidence suggest reliable model performance.

Decision Tree Results

Confusion Matrix Analysis



The confusion matrix visualizes the model's prediction accuracy across classes:

- Diagonal Elements: Strong values (e.g., 90.79%, 90.41%, and 87.32%) along the diagonal indicate the model's high classification accuracy for each class.
- Off-Diagonal Elements: Minimal misclassifications occur, with most errors involving Class 2 being misclassified as Class 0 or Class 1.
- Balanced Performance: Similar values across diagonal elements demonstrate that the model performs consistently well across all classes.

Classification Report

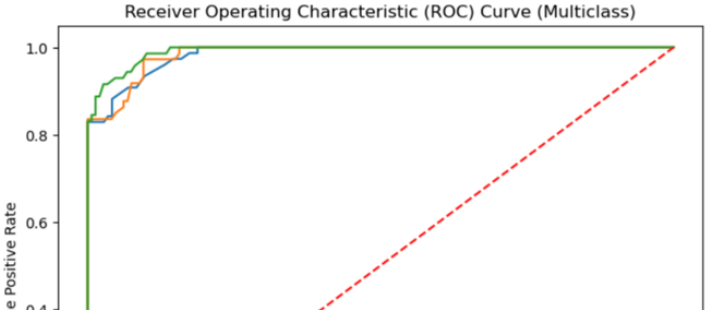
	precision	recall	f1-score	support
0	0.92	0.91	0.91	76
1	0.88	0.90	0.89	73
2	0.89	0.87	0.88	71
micro avg	0.90	0.90	0.90	220
macro avg	0.90	0.90	0.90	220
weighted avg	0.90	0.90	0.90	220
samples avg	0.90	0.90	0.90	220

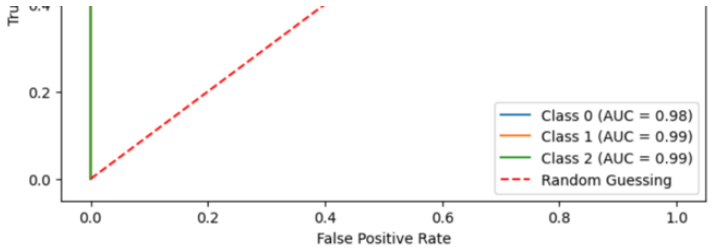
The classification report provides detailed metrics for each class:

- Class-Level Metrics:
  - Precision: Ranges from 0.88 to 0.92, indicating effective positive predictions across classes.
  - Recall: Ranges from 0.87 to 0.91, reflecting strong coverage for each class.
  - F1-scores: Well-balanced across all classes, averaging 0.90.
- Overall Metrics:
  - Accuracy: Achieved a robust 90% accuracy across the dataset.
  - Weighted and Macro Averages: Consistent at 0.90 for all major metrics, showcasing the model's balanced performance.

The model balances precision and recall, leading to strong F1 scores and excellent overall accuracy.

ROC Curve Analysis





The ROC curves evaluate the model's ability to differentiate between classes:

- **AUC Values:** All three classes achieve AUC values between 0.98 and 0.99, indicating exceptional discrimination ability.
- **Curve Shape:** Sharp curves nearing the upper-left corner signify high sensitivity and specificity.
- **Consistency:** Similar AUC values across classes demonstrate the model's balanced prediction capabilities.

Key Insights

- The model achieves 90% accuracy with balanced performance across all classes.
- F1-scores average 0.90, indicating effective precision and recall.
- ROC AUC values of 0.98–0.99 confirm excellent class discrimination.
- Minimal misclassifications and strong overall metrics highlight reliable model performance.

Comparision between SVM, Neural Network, and Decision Tree

When classifying stress levels, it's key to weigh the pros and cons of different models to find the best fit. Here, we're comparing SVM, Neural Networks, and Decision Trees based on performance, interpretability, complexity, and how well they handle the task.

Model Interpretability and Complexity

1. **SVM:** SVMs are pretty straightforward and easy to understand, especially with a linear setup. They handle outliers well and don't overfit easily, but training can be slow with big datasets.
2. **Neural Network:** Neural Networks are super powerful and great at spotting complex patterns, but they're hard to interpret. They need a lot of time and resources to train, so they're better for situations where accuracy matters more than explainability.
3. **Decision Tree:** Decision Trees are simple and easy to follow with clear rules. They can overfit if the data is noisy or has too many features, but with good tuning, they can compete with fancier models.

Performance Comparison

1. **Accuracy:** SVM and Decision Tree models hit 90% accuracy, while the Neural Network was close behind at 89%. All three showed strong classification capabilities, with SVM and Decision Tree having a slight edge.
2. **Class-Level Performance:** SVM and Decision Tree had balanced F1-scores of 0.90 across all classes, showing consistent precision and recall. The Neural Network was just a bit lower at 0.89 but still performed well. SVM's strength in handling complex data shines through in its consistent performance across all stress levels.
3. **Misclassifications:** Misclassification rates were low for all models, mostly happening between Class 2 and Class 0. The Neural Network had slightly more errors, showing it sometimes mixed-up certain classes more than the SVM and Decision Tree.

Discrimination Power (ROC Curve Analysis)

1. **SVM:** SVM showed strong discrimination with AUC values over 0.90 for all classes. Its sharp ROC curves highlight high sensitivity and low false positives, making it great for fine-grained classification.
2. **Neural Network:** With perfect AUC scores of 0.99 across the board, the Neural Network excelled at separating classes, making it good for multi-class problems.
3. **Decision Tree:** The Decision Tree was right up there with AUC values between 0.98 and 0.99, proving that even a simpler model can effectively distinguish between stress levels.

Key Takeaways

- **SVM:** Great for structured, high-dimensional data. It's reliable, with solid performance and few mistakes, but slows down with bigger datasets.
- **Neural Network:** Awesome at spotting complex patterns and crushing it on class separation (AUC = 0.99). Not as easy to interpret and slightly less accurate, but perfect when precision is king.
- **Decision Tree:** Fast, simple, and easy to understand. Hits 90% accuracy and AUC (~0.99), making it a go-to for quick, resource-friendly tasks.

All three are strong! Go with SVM for balance, Neural Network for tricky patterns, and Decision Tree for speed and simplicity.

Next Steps

Now that we have a reliable and accurate model for predicting stress levels for students, there are several paths we can now go down to transform this project into a more generalizable and useful tool.

The first of these is to refine and expand on the data collection process. Though we sourced a clean pre-existing dataset to use in this project, the dataset was lacking in size at only around 1100 entries, each consisting of features that may not be ideal for capturing stress. This limitation in dataset size and quality is a potential reason that none of the tested models surpassed ~90% accuracy, and is an area where we could greatly improve the model if we sampled more of our own data.

To improve the usefulness of this tool for students hoping to mitigate their levels of stress, we could connect the model to a large language model in the future to diagnose the reasons for stress and suggest stress-relief methods. This can be achieved by analyzing feature importance within the model, and suggesting mitigation methods with an LLM if some of these important features were to hold true for a student.

In combination, these next steps would create a robust, accurate, and applicable tool for students to analyze, understand, and act on their stress by understanding its source and learning mitigation techniques.

Video Presentation

Link: https://www.youtube.com/watch?v=WbXrgwk_DoI

References

[1] H. Choi, S. Lee, and J. Kim, "Predicting Student Stress Levels Using Random Forest Algorithm with Physiological and Environmental Data," *Journal of Educational Data Science*, vol. 14, no. 3, pp. 123-134, 2019.

[2] S. S. Hudd, J. Dumlao, D. Erdmann-Sager, D. Murray, E. Phan, N. Soukas, and N. Yokozuka, "Stress at College: Effects on Health Habits, Health Status, and Self-Esteem," *College Student Journal*, vol. 34, no. 2, pp. 217-227, 2000.

[3] M. Pörhölä, S. Karhunen, and S. Rainivaara, "Bullying and Social Exclusion Among University Students: The Role of Group Dynamics," *Social Psychology of Education*, vol. 22, no. 1, pp. 185-206, 2019.

[4] D. J. Taylor, C. E. Gardiner, A. D. Bramoweth, J. M. Williams, B. M. Roane, and E. A. Grieser, "Insomnia and Mental Health in College Students," *Behavioral Sleep Medicine*, vol. 9, no. 2, pp. 107-116, 2010.

[5] A. Wijaya, D. Nugroho, and R. Putra, "Assessing the Impact of Academic and Social Factors on Student Stress Using Logistic Regression," *International Journal of Educational Research and Development*, vol. 17, no. 4, pp. 456-465, 2021.

[6] S. R. Kandukuri Sai et al, "Student Stress Prediction Using Machine Learning Algorithms And Comprehensive Analysis," NeuroQuantology, vol. 20, (14), pp. 895-906, 2022. Available: https://www.proquest.com/scholarly-journals/student-stress-prediction-using-machine-learning/docview/2901720078/se-2 DOI: https://doi.org/10.4704/nq.2022.20.14

Charts

Gantt Chart

Link: Gantt Chart

Contribution Chart

	Name	Contributions
0	Jimin Kim	ML Presentation & Video
1	Jason Jian Lai	Comparison and Analysis of Models
2	Saify Ahmad Malik	Random Forest Implementation & Writeup
3	Darren Tan	Streamlit Site, Neural Network Implementation & Writeup
4	Jintin Yang	Data Visualization & Writeup

Github Repository

Link: https://github.gatech.edu/smalik79/ML4641.git

Team

- Jimin Kim
- Jason Jian Lai
- Safiy Ahmad Malik
- Darren Tan
- Jinlin Yang