Apple Nacme AIML Intensive

AIMLeaders

Team AIML

Rabiat Sadiq

Computer Engineering and Science 2024

University of Texas at San Antonio

Emiliano Gonzalez

Materials Science and Engineering 2026

Georgia Institute of Technology

Emily Mojica

Computer Science and Business Administration 2026

University of Southern California

Rodrigo Aguilar B

Project Guide

Dr. Corey Baker

Project Advisor

Project Objectives

To use **Machine Learning (ML)** to understand patient objective and subjective data in remote patient monitoring applications.

Specifically, to integrate ML into **Assuage** for non-invasive detection of immediate **distress** using objective data.

Distress

Distress is defined in the NCCN Guidelines for Distress Management as a multifactorial, unpleasant experience of a psychologic (ie, cognitive, behavioral, emotional), social, spiritual, and/or physical nature that may interfere with the ability to cope effectively with cancer, its physical symptoms, and its treatment.

Early evaluation and screening for distress leads to early and timely management of psychologic distress, which improves medical management.

Assuage

- Assuage is a research platform built on Apple's open-source frameworks, ResearchKit and CareKit.

- Uses HealthKit to collect health data from sensors, with offline functionality to ensure data is gathered even without internet access.

- It is HIPAA-compliant, ensuring that it adheres to privacy and security regulations for handling health data.

Previously, Assuage **did not use machine learning.**

Research and Background

How is Distress detected? What are its **symptoms**?

Primary Symptoms

- Depression
- Pain/Despair/Hopelessness
- Fatigue
- Unclear Thinking/Poor Concentration
- Poor Sleep
- Anger/Feeling Life is Out of Control
- Anxiety (Short Term/Acute)

Indicators:

- Heart Rate
- Blood Pressure
- Body Fat Percent
- Body Mass Index (BMI)
- Body Temperature
- Electrodermal Activity
- Active Energy
- Activity Trend
- Screen Time
- Heart Rate Variability (HRV)
- Respiratory Rate
- Inertial Metrics
- Time Outside
- Sleep Quality
- Step Count

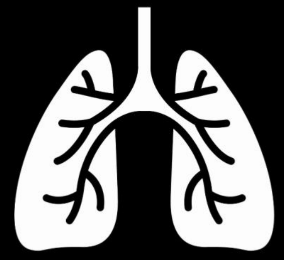
Research and Background

Primary Indicators:

- Higher Heart Rate
- Lower Heart Rate Variability (HRV)
- Lower Respiration Rate

Activity Metrics:

- Lower Step Count
- Lower Active Energy (Calories Burned)

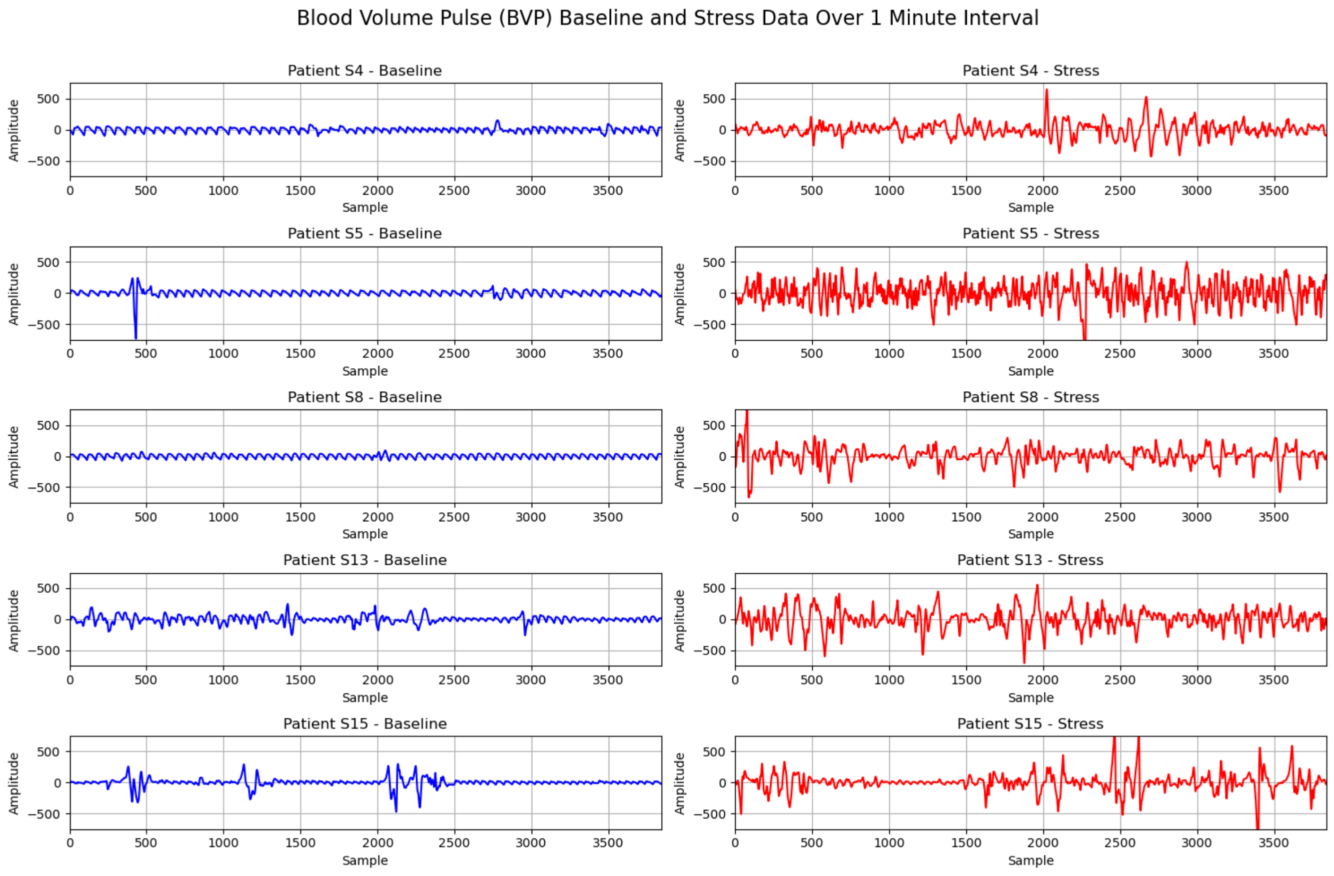


Exploratory Data Analysis - WESAD Dataset

The Wearable Stress and Affect Detection (WESAD) Dataset contains physiological and motion data from 15 participants collected via wearable devices (chest and wrist) during activities inducing stress, amusement, and neutral states.
Dataset includes **Blood Volume Pulse (BVP)**, Electrocardiogram (ECG), Electromyogram (EMG), Electrodermal Activity (EDA), **Respiration (RESP)**, body temperature, and acceleration data, useful for stress and emotion detection research.

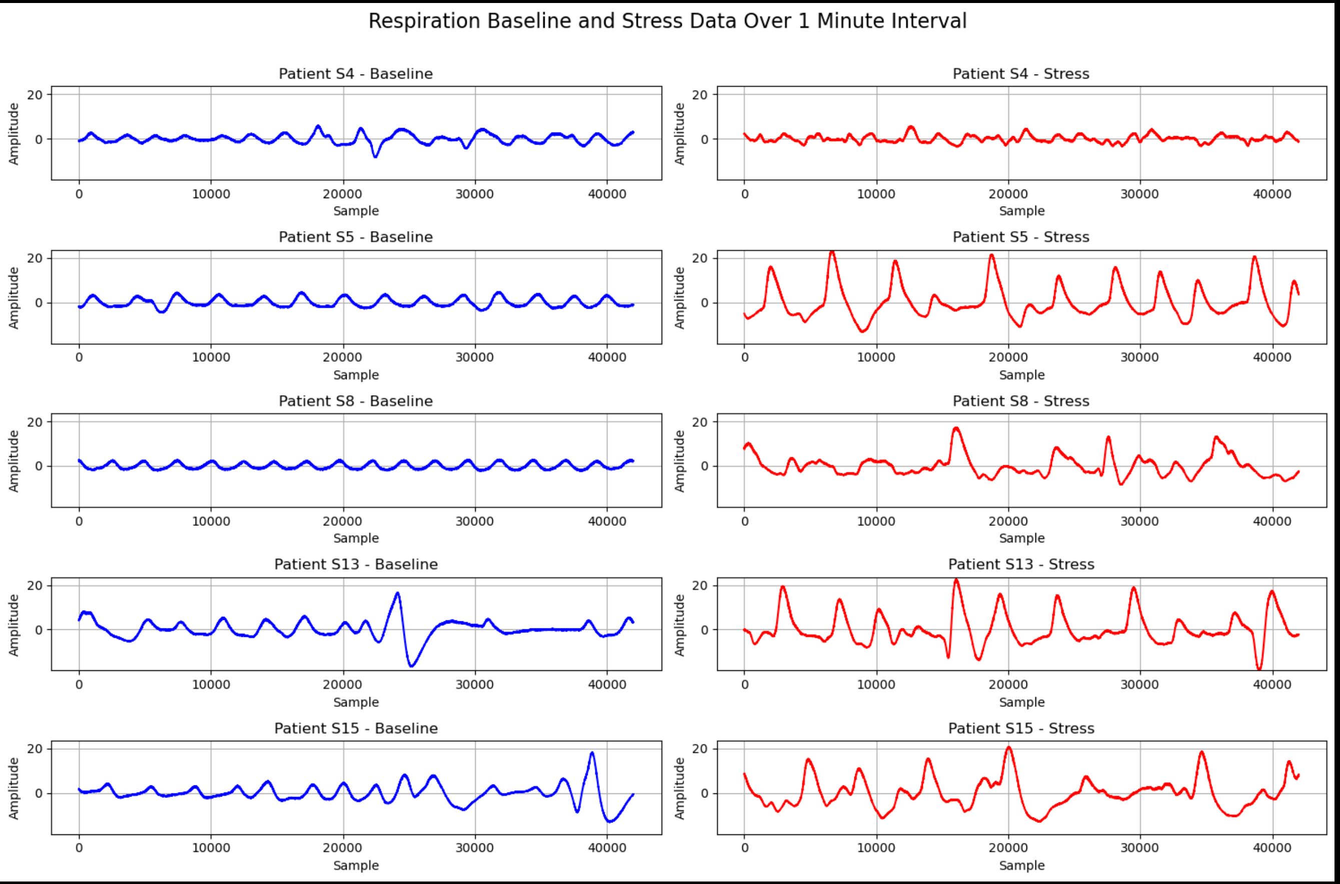
WESAD Dataset

We used the WESAD dataset to gain a visual understanding of how stress affects biometric data, with the goal of applying these insights to detect distress in our project.



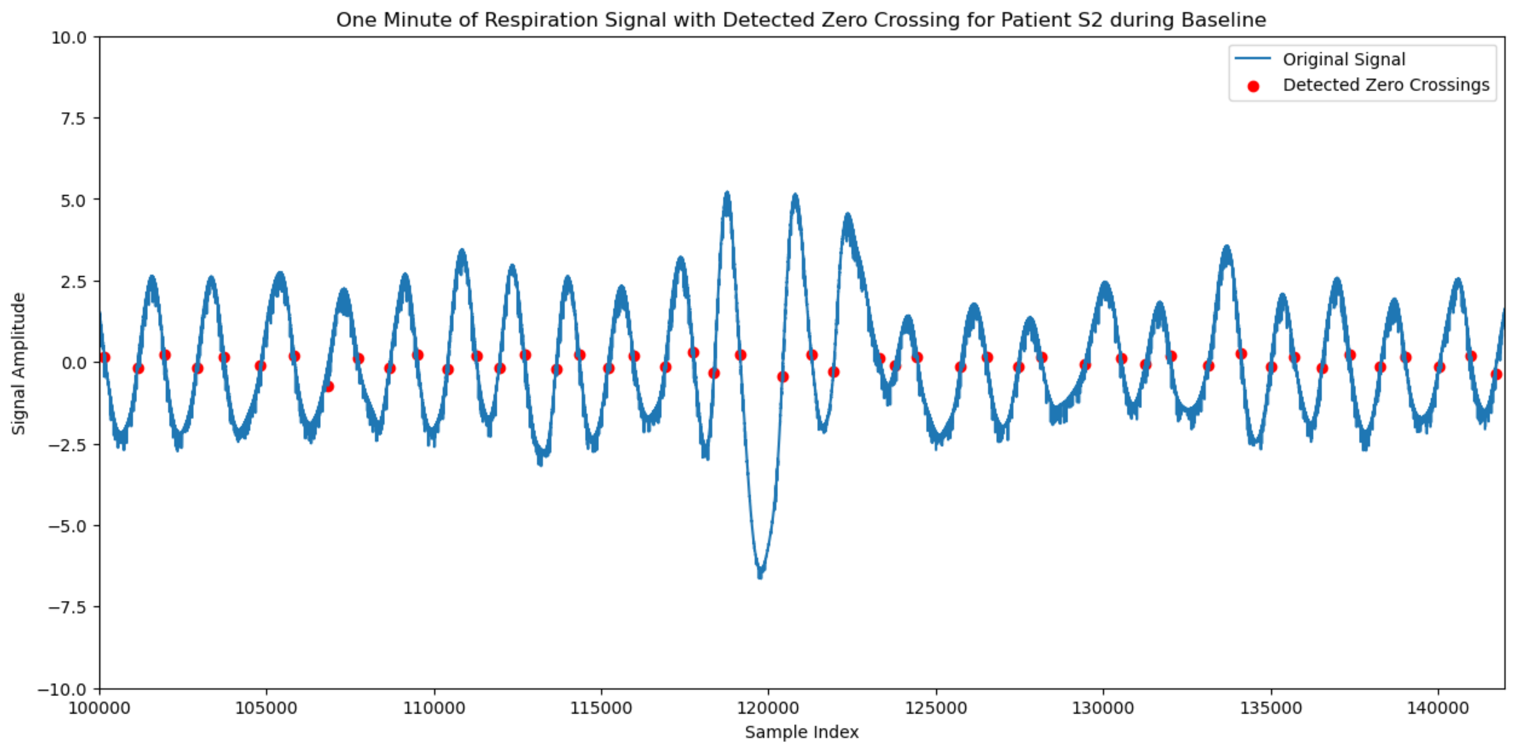
WESAD Dataset

We used the WESAD dataset to gain a visual understanding of how stress affects biometric data, with the goal of applying these insights to detect distress in our project.



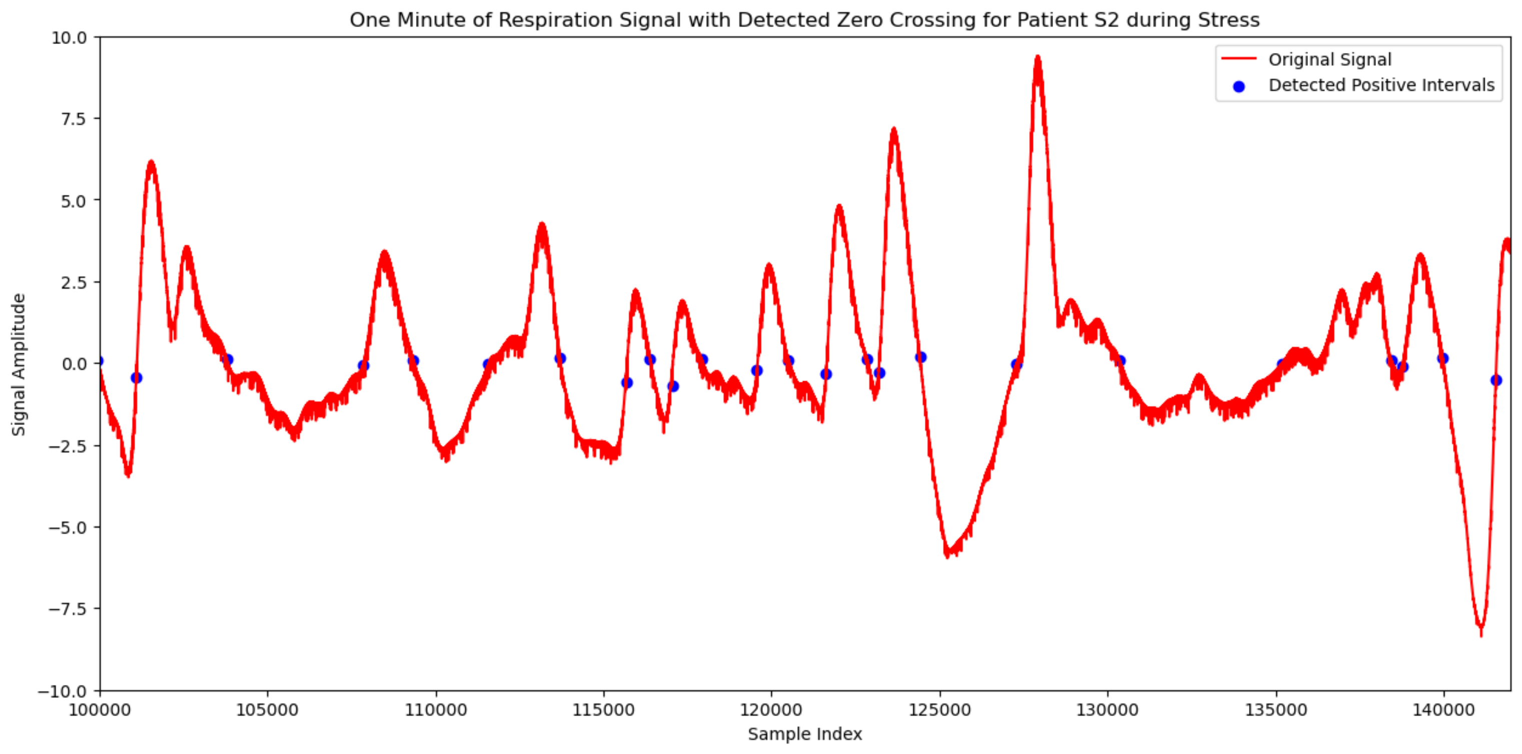
WESAD Dataset

To better understand the dataset, we calculated Heart Rate and Heart Rate Variability (HRV) from Blood Volume Pulse data and Respiration Rate from respiration data. Due to time constraints and the complexity of these calculations, we employed a creative approach by using zero crossings to determine frequency and derive the needed values.



WESAD Dataset

To better understand the dataset, we calculated Heart Rate and Heart Rate Variability (HRV) from Blood Volume Pulse data and Respiration Rate from respiration data. Due to time constraints and the complexity of these calculations, we employed a creative approach by using zero crossings to determine frequency and derive the needed values.



WESAD Dataset

Results of Data Exploratory Analysis of WESAD Dataset

	State	Avg. Heart Rate (BPM)	Avg. HRV (ms)	Avg. Respiration Rate (BPM)
0	Amusement	74.781926	135.109073	58.886487
1	BaseLine	72.208790	154.563135	65.227120
2	Meditation 1	70.342079	148.252616	55.836012
3	Meditation 2	71.899376	144.180008	67.746229
4	Stress	78.064401	123.817442	46.485984

HealthKit ResearchKit CareKit

Methodology

- The methodology involved collecting biometric data through **HealthKit** and organizing it with **CareKit.**
- The data was analyzed using machine learning models integrated via **Core ML.**
- To present the data and predictions effectively, an intuitive app was developed using **SwiftUI.**

Data Generation and EDA..

Challenge: Collecting real-world data from cancer patients experiencing distress was difficult due to their specialized needs.

Solution: We created a unique approach to overcome this challenge and gather the necessary data effectively.

We started with a small set of 17 actual data points. That's is receivable from the objective data

<u>5 selected features:</u>

Heart Rate(bpm)

Steps

Active Energy(cal)

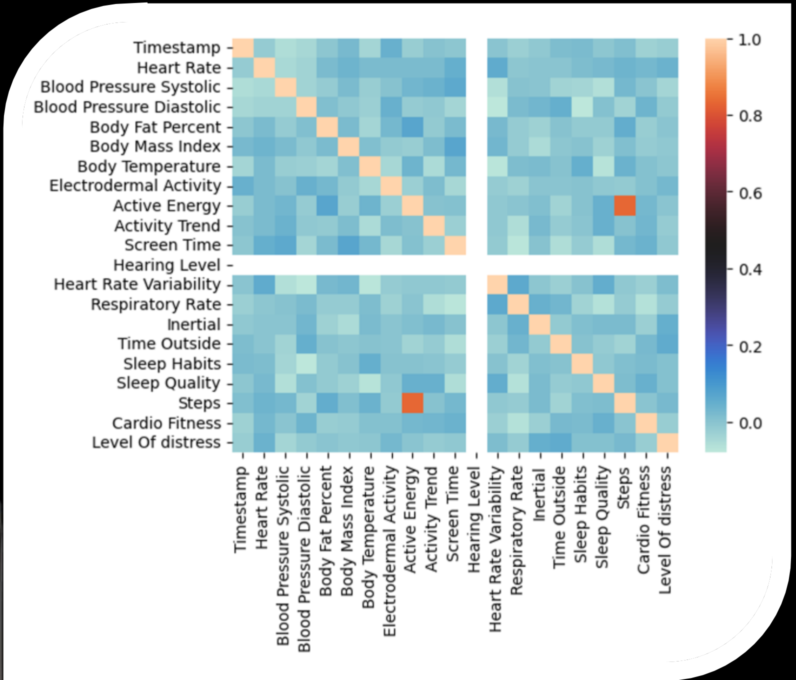
Heart Rate Variability(ms)

Respiratory Rate(bpm)

Data Generation and EDA..

We started with a small set of 17 actual data points. That's is receivable from the objective data

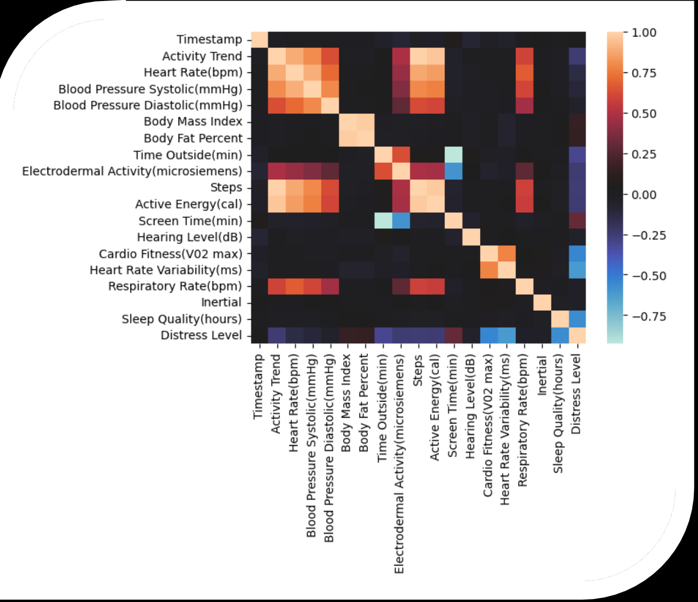
Timestamp,Activity Trend,Heart Rate(bpm),Blood Pressure Systolic(mmHg),Blood Pressure Diastolic(mmHg),Body Mass
2024-03-01 01:57:55,4,79,109.5,68.7,31.49,18.887,185,5,3880,192,208,74,31,51,14.9,8,9,7.154650000000002
2024-03-01 09:40:21,7,90,119,74.4,32.72,18.486,271,8.5,6852,316.6,176,96,21,36,16,5,8,7.602579999999992
2024-03-01 09:45:08,8,98,124,79.4,30.35,14.405,42,2,7818,432.9,432,38,39,56,14.8,7,7,7.324419999999996
2024-03-01 19:35:35,2,67,107.5,68.5,27.37,9.531,259,5,1878,74.9,208,93,25,22,12.7,2,8,7.828669999999996
2024-03-01 23:26:33,7,104,120,70,19.06,1.728,283,9.5,6849,335.45,81,56,31,66,17.4,1,1,7.60911
2024-03-01 15:29:02,2,64,97,58.2,19.02,-0.324,82,0,1875,46.75,284,101,43,61,12.4,5,7,6.980550000000002
2024-03-01 09:54:19,8,102,119,68.4,26.47,10.361,69,7,8000,423,411,67,40,79,17.2,5,1,8.331299999999998
2024-03-01 13:56:16,5,91,112.5,71.5,28.68,15.234,141,2.5,4955,261.75,274,11,45,85,13.1,7,10,5.909999999999998
2024-03-01 23:50:32,7,87,112.5,68.5,31.46,17.848,268,5.5,6690,337.5,125,74,30,40,13.7,3,1,8.380449999999996
2024-03-01 09:28:38,4,84,117,68.2,27.37,11.531,94,3,3884,244.2,369,109,37,82,15.4,8,10,6.718060000000004
2024-03-01 19:23:06,4,88,118,68.8,30.6,14.73,33,2,4422,257.1,349,13,32,39,15.8,1,9,7.743580000000004
2024-03-01 19:04:17,5,77,104.5,63.7,27.43,9.609,70,3.5,5014,208.7,328,97,22,51,13.7,8,3,8.237610000000014
2024-03-01 19:33:28,1,68,109,69.4,23.1,6.98,173,2.5,1096,32.8,266,37,44,58,10.8,1,8,7.401939999999992
2024-03-01 19:05:18,7,93,119.5,72.7,31.62,19.056,55,2.5,6667,329.35,310,52,41,71,15.3,9,3,7.649780000000026
2024-03-01 09:27:19,6,96,120,68,22.24,3.862,124,5,5785,281.75,262,127,30,70,13.6,1,5,7.298449999999996
2024-03-01 14:20:45,4,72,110,69,18.78,1.364,102,3,3933,153.65,259,115,32,55,12.2,6,10,6.959869999999984
2024-03-01 16:17:46,5,83,114.5,67.7,34.66,20.008,32,0.5,5492,285.6,368,103,21,52,13.3,2,5,8.251830000000062
2024-03-01 20:24:42,2,69,102.5,64.5,26.82,8.816,208,5,1848,117.4,185,103,24,38,14.9,8,10,7.721670000000002



- Current randomized data from mockaroo
- Next goal is to get more **correlation** between the data to simulate real life
- Randomized data from Mockaroo

Data Generation and EDA..

We started with a small set of 17 actual data points. That’s is receivable from the objective data



Correlation Integration:

- Applied **research-based correlations** to simulate realistic relationships between data points.
- Developed formulas for specific correlations, such as:

  Heart Rate & **Blood Pressure**: For every 10 bpm increase, systolic BP might increase by 3-5 mmHg.
- This helped us generate pseudo-data that realistically reflects potential distress levels

```
HeartRate = 60 + (field("Activity Trend") * 5) + random(−10, 10)
```

```
ActiveEnergyBurned = field("Steps") * 0.05 + random(−50, 50)
```

Data Generation and EDA..

5 selected features:

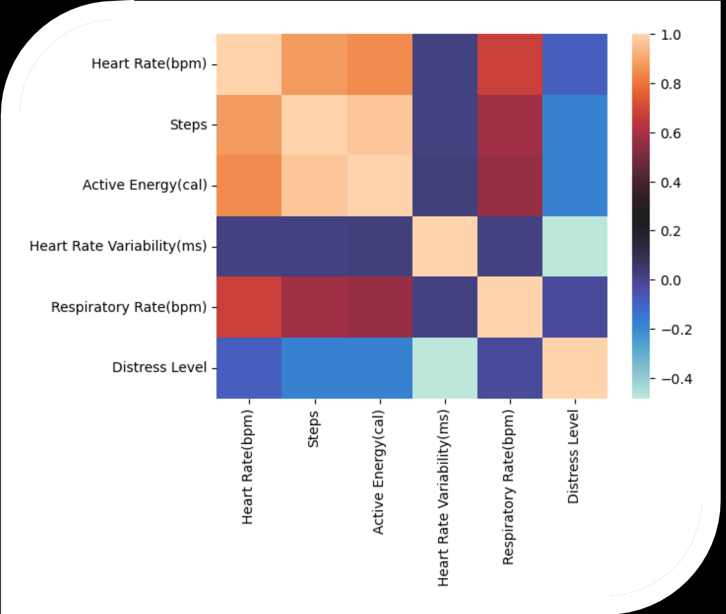
Heart Rate (bpm)

Heart Rate Variability (ms)

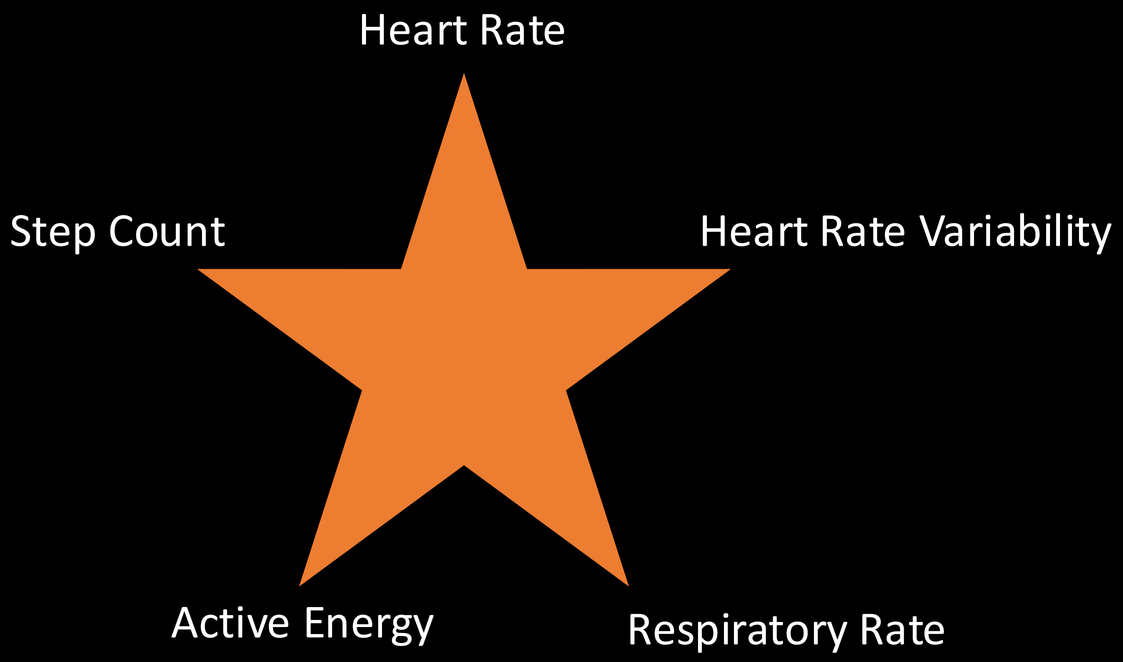
Respiratory Rate (bpm)

Step Count

Active Energy(cal)



Final Selection:
Chose 5 key features from the 17 initially generated due to time constraints and practical sensor capabilities.



Data Preprocessing

```python
feature_df = df.drop(["Distress Level", "Timestamp"], axis = 1)

features = feature_df.values

X = features
y = df["Distress Level"].values.reshape(-1,1)
```

DataFrame

Scale

```python
x_scaler = MinMaxScaler()
y_scaler = MinMaxScaler()

X_scaled = x_scaler.fit_transform(X)
y_scaled = y_scaler.fit_transform(y)

#Split our dataset such that our training data is 80% of our total dataset, and testing is the remaining 20%
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_scaled, test_size=0.2, random_state=2024)
```

Data Preprocessing

DataSet

```
training_data = assuageData(X_train, y_train)
testing_data = assuageData(X_test, y_test)
training_data.__getitem__(0)
```

```
from torch.utils.data import DataLoader
#DataLoader iterates our dataset in batches of size batch_size
train_loader = DataLoader(training_data, batch_size=32)
testing_loader = DataLoader(testing_data, batch_size=32)
```

Loading Data

Model Development

- Linear Regression
- Logistic Regression
- KNN
- Random Forest

Objective

To develop a predictive model for distress levels using various algorithms and choose the best-performing one.

Model Development

- Logistic Regression
- KNN
- Random Forest

Linear Regression

```
import torch.nn as nn

class LinearRegression(nn.Module):
    def __init__(self, in_dim):
        #Base class for all neural network modules
        super(LinearRegression, self).__init__()
        #Applies a linear transformation to the incoming data: y = x @ A.T + b
        self.linear = nn.Linear(in_dim, 1)

        model = LinearRegression(X_train.shape[1])

    def forward(self,x):
        out = self.linear(x)
        return out
```

Improved Correlation: With better pseudo-data that had more realistic correlations, we improved the model's performance. The linear regression model achieved around a 95% R^2 score and 0.1 RMSE.

Model Development

- Linear Regression
- KNN
- Random Forest etc

Logistic Regression

```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression(random_state=2020)
model.fit(X_train, y_train)
```

```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

print('Accuracy: ', round(accuracy_score(predictions, y_test), 3))
print('Precision: ', round(precision_score(predictions, y_test), 3))
print('Recall: ', round(recall_score(predictions, y_test), 3))
print('F1: ', round(f1_score(predictions, y_test), 3))
```

Results

Accuracy: 0.905
Precision: 0.941
Recall: 0.851
F1: 0.894

True Positive: 80 True Negative: 101
False Positive: 14 False Negative: 5

Model Development

- Linear Regression
- Logistic Regression
- Random Forest etc

KNN

```
from sklearn.neighbors import KNeighborsClassifier


n_neighbors = 10

# KNN = takes the number of neighbors, the amount to classify by p, and the met
KNN = KNeighborsClassifier(n_neighbors= n_neighbors, p= 2, metric='euclidean')


KNN.fit(X_train, y_train)

y_pred = KNN.predict(X_test)
```

Results:
Confusion Matrix: [[94 0], [0 106]]
F1 Score: 1.0
Accuracy: 1.0

Model Development

- Linear Regression
- Logistic Regression
- KNN

Other Models

Random Forest
Boosted Tree
Decision Tree
Support Vector Machine (SVM)

Model Selected

- Linear Regression
- KNN
- Random Forest etc

Logistic Regression

- Clear Interpretation: Logistic regression provides probabilities for distress, allowing us to clearly flag and interpret distress events.
- Performance: Achieved robust results with good accuracy, precision, recall, and F1 score.
- Suitability: Suitable for binary classification tasks and provides a straightforward approach to detecting distress levels in patients.
- Practicality: Easy to implement and interpret, making it ideal for real-world applications in clinical settings.

Model Development- Logistic Regression

```
model = LogisticRegression(
    random_state=2020,
)
param_grid = {
    "tol": [1e-3, 1e-4, 1e-5],
    "C": [0.01, 0.1, 1, 10, 50, 100],
    "solver": ["lbfgs", "saga", "liblinear"],
    "max_iter": [100, 200, 150, 300, 400]
    }

search = GridSearchCV(model, param_grid, cv=5, scoring='accuracy')

search.fit(X_train, y_train)

print(search.best_estimator_)
```

We used **GridsearchCv** to find the best parameters of the model

Model Development- Logistic Regression

```python
model = LogisticRegression(
    random_state=2020,
)
param_grid = {
    "tol": [1e-3, 1e-4, 1e-5],
    "C": [0.01, 0.1, 1, 10, 50, 100],
    "solver": ["lbfgs", "saga", "liblinear"],
    "max_iter": [100, 200, 150, 300, 400]
    }

search = GridSearchCV(model, param_grid, cv=5, scoring='accuracy')

search.fit(X_train, y_train)

print(search.best_estimator_)
```

```python
# 0 might mean distress
single_instance = np.array([[1,59,98.5,64.1,34.46,22.

single_instance = np.array([[7,101,119.5,67.7,21.82,3

# Predicting the outcome for the single instance
single_prediction = best_model.predict(single_instanc
print("Prediction for the single instance: ", single_
```

```
Prediction for the single instance:  [0]
```

Converting to CoreML

```python
# Convert the model to Core ML
import coremltools as ct

coreml_model = ct.converters.convert(best_model, feature_columns,'Level of Distress')

# Save the Core ML model
coreml_model.save('logistic_regression.mlmodel')
```

MachineLearningModels
BinaryDistressClassification

```python
traced_model = torch.jit.trace(model, torch.randn(1, model.input_size))

  # Convert the model to Core ML
mlmodel = ct.convert(
    traced_model,
    inputs=[ct.TensorType(name="input", shape=(1, model.input_size))],

)

# Save the converted model
mlmodel.save("logistic_regression_model.mlmodel")
```

Learning Swift and iOS Development

Swift

- Swift is an open source programming language created by Apple for building Apps for Apple Platforms

Challenges Faced

- Syntax & Language Features
- Xcode

Key Learnings

- Struct vs Class
- Optional Variables
- Static Variables

Integration with Assuage

Model Import:

Action: Imported the CoreML (.mlmodel) classification model into the Assuage app.

Purpose: To utilize the trained model for distress level predictions within the app.

Function Implementation:

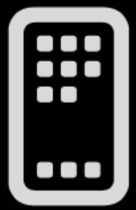
Action: Developed functions to handle model predictions.

Purpose:
To process user data and generate distress level predictions in real-time.

Real-Time Prediction Display:

Action: Implemented a feature to display predictions instantly within the app.

Purpose: To show users their current distress levels as soon as predictions are made.



Integration with Assuage Model

```swift
class PredictionModel: ObservableObject {
    @Published var predictionResult: String = "No prediction yet"

    private let model: BinaryDistressClassification

    init() {
        // Initialize the model
        do {
            model = try BinaryDistressClassification(configuration: MLModelConfiguration())
        } catch {
            fatalError("Failed to load model: \(error)")
        }
    }
```

Attempts to create an instance of the BinaryDistressClassification

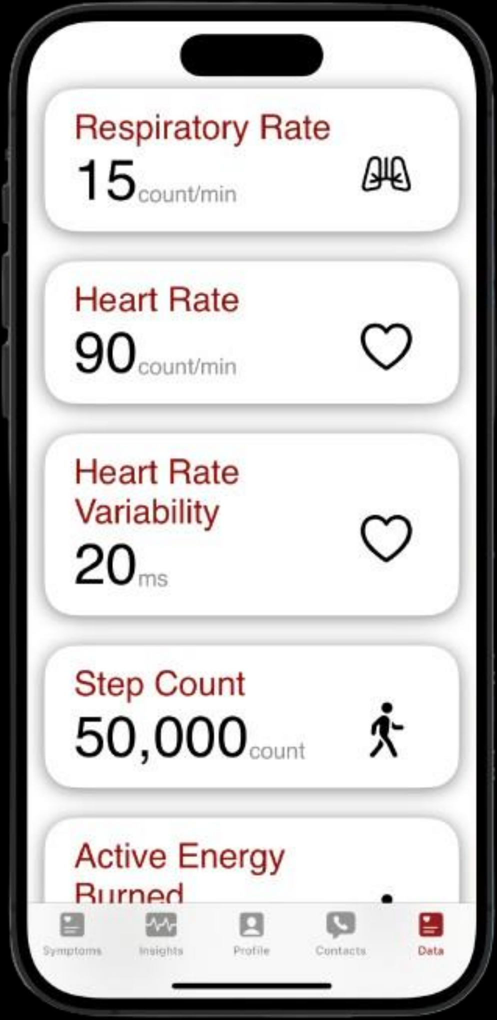
Any changes to this property will automatically notify and update any views observing it

PredictionModel, is used to manage the prediction process and update the UI when the prediction result changes.

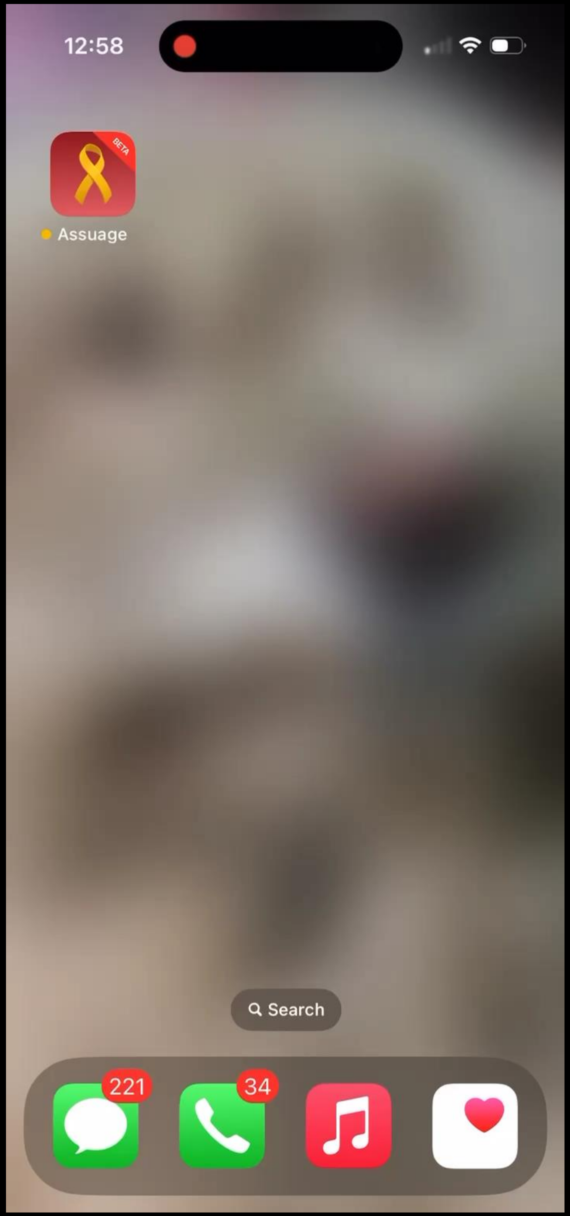
Integration with Assuage (UI)

```swift
struct ObjectiveDataView: View {
    @Environment(\.careKitStyle) private var style

    var title: String
    var outcome: NSNumber
    var units: String
    var body: some View {
                CardView {
                    HStack {
                        VStack(alignment: .leading) {
                            Text(displayTitle(for: title))
                                .foregroundColor(Color(red: 0.5976, green: 0.0, blue: 0.0))
                                .font(.custom("SF Pro Display", size: 30))
                                .multilineTextAlignment(.leading)
                                .padding(.leading, 10)
                            HStack(alignment: .bottom, spacing: 0.2) {
                                Text("\(outcome)")
                                    .foregroundColor(.black)
                                    .font(.custom("SF Pro Display", size: 50))
                                    .multilineTextAlignment(.leading)
                                    .padding(.leading, 10)
                                Text(units)
                                    .foregroundColor(.gray)
                                    .font(.custom("SF Pro Display", size: 20))
                                    .multilineTextAlignment(.leading)
                                    .padding([.leading, .bottom ], 5)
                            }
                        }
                        Spacer()
                        Image(systemName: imageName(for: title))
                            .resizable()
                            .aspectRatio(contentMode: .fit)
                            .frame(width: 45, height: 50)
                            .padding(.trailing, 25)
                            .padding(.top, 25)
                    }
                    .padding()
                    .frame(maxWidth: .infinity)
                }
            .padding(.horizontal, 15)
    }
```



Demo



Bias

Data Bias

- Pseudo Data: Might not fully capture the complexities and variations of real-world data
- Sensor Inconsistencies: Users may have varying quality and accuracy in their sensor data depending on how they wear their device

Modeling Bias

- Features: Chosen features might not fully capture the indicators of distress and omit other features

Confirmation Bias

- Data Interpretation: Selectively interpreting data based on what researchers believe indicates distress.

Ethical Consideration

Privacy

- Ensuring that users data is securely stored

Accuracy and Reliability

- False positives or negatives can cause unnecessary anxiety or cause patients to miss out on critical intervention opportunities

Fairness and Equity

- Ensuring the app is accessible to all users

Limitations

Data Availability

- Limited to users who have HealthKit-enabled devices and have granted permissions.

Model Accuracy

- Predictive model accuracy can be affected by the variability in individual health data.

Real-time Processing

- Real-time data processing and prediction might be limited by device capabilities and battery consumption.

Recommendations

User Engagement

- Encourage users to regularly update their health data to improve prediction accuracy.

Privacy and Security

- Ensure all collected health data is securely stored and complies with privacy regulations.

Feedback Mechanism

- Implement a feedback loop for users to report the accuracy of distress predictions, helping to refine the model.

Future Work

Enhanced Predictive Models

- Get more user data to test on and use more advanced machine learning techniques to improve prediction accuracy.

Integration with Health Data Sources

- Expand data collection to include other health metrics and wearable devices.

User Personalization

- Customize predictions and recommendations based on individual user profiles and preferences.

Overview

Devices Used

xCode

Objective Sensor

Over 75% accuracy

79%	87%	83%
Training	Validation	Testing

Tracks Data

tools

Privacy

Real Time Alert

Bias

Impact in health

Machine Learning

Works without WiFi

Health

HealthKit

ResearchKit

CareKit

Utilizing apples framework

Swift

Acknowledgements

The AIMLeaders team extends heartfelt gratitude to Dr. Corey Baker, Apple, NACME, and The University of Southern California for their invaluable technical guidance and support throughout our learning journey and project.

NACME

NATIONAL ACTION COUNCIL FOR MINORITIES IN ENGINEERING

USC Viterbi

School of Engineering

References

[1]. M. B. Riba *et al.*, “Distress Management, Version 3.2019, NCCN Clinical Practice Guidelines in Oncology,” *Journal of the National Comprehensive Cancer N etwork*, vol. 17, no. 10, pp. 1229–1249, Oct. 2019, doi: https://doi.org/10.6004/jnccn.2019.0048.

[2] “Designing Survey–Based Mobile Interfaces for Rural Cancer Patients Using Apple’s ResearchKit and CareKit: Usability Study,” *JMIR Preprints*, 2015. https://preprints.jmir.org/preprint/57801/accepted (accessed Aug. 01, 2024).

[3] qiriro, “Biometrics for stress monitoring,” *Kaggle.com*, 2020. https://www.kaggle.com/datasets/qiriro/stress (accessed Aug. 01, 2024).

[4] J. L. Kibler and M. Ma, “Depressive symptoms and cardiovascular reactivity to laboratory behavioral stress,” *International Journal of Behavioral Medicine*, vol. 11, no. 2, pp. 81–87, Jun. 2004, doi: https://doi.org/10.1207/s15327558ijbm1102_3.

Thank you!

Questions?

Contact us Team AIMLeaders

Rabiat Sadiq

Computer Engineering and Science

Robbiahsadiq@gmail.com

Emiliano Gonzalez

Materials Science and Engineering 2026

egonzalez92@gatech.edu

Emily Mojica

Computer Science and Business Administration 2026

melymojica@icloud.com

Rodrigo Aguilar B

Project Guide

Aguilarb@usc.edu

Dr. Corey Baker

Project Advisor

coreybak@usc.edu

Github