# ml_group28

# Group 28 Final Report

## Report

**Introduction**
Wearable technology has become a key component in monitoring personal health, with devices like Fitbit, Garmin, and Apple Watch leading the way. We started by looking into Using Wearable Technology to Predict Health Outcomes: A Literature Review, an article by Burnham et. al, 2018, which explores how such devices predict health using tracking data like heart rate, steps, and sleep. Studies included in their review showed that wearable data can predict whether individuals might develop certain health issues, suggesting a promising future for health analytics [1].

This project leverages Fitbit's dataset, which tracks daily activity (steps, calories burned, heart rate, and sleep). This data, collected at daily, hourly, and minute intervals, will allow us to explore how personalized insights can be generated through machine learning. The dataset can be accessed via Kaggle.

**Problem Definition**
Many individuals struggle to maintain consistent fitness routines, often finding it difficult to interpret fitness metrics from their devices. Therefore, many lack the motivation to adhere to their goals for a longer period. According to Tiga Healthcare, wearable metrics "can offer insights, predict trends, or even warn the user of potential health concerns" [4]. We plan to analyze this data to help users make better health choices. Another motivation for this project is to be able to support long-term outcomes like weight management, mental well-being, and also improved cardiovascular health. If we can use models to predict potential health outcomes, we can provide customized interventions too.

**Methods**
Data Preprocessing Method
For the model we have created, as well as the data we have selected, we have realized that picking a robust normalization technique would be best for this scenario. This is due to the fact that within the data we have collected and are analyzing, we have a multitude of outliers, such as total steps being 0, heart rate being 0 (due to not wearing the Fitbit), etc. When figuring out data preprocessing, we had three options in front of us: min-max, z-score normalization, and robust normalization technique. We chose the robust normalization technique as it uses the interquartile

range for scaling and is much less sensitive to outliers in comparison to other normalization techniques such as min-max scaling and z-score normalization.

The formula for the robust scaler method we utilized is:

$$\boxed{X_{new} = \frac{X - X_{median}}{IQR}}$$

This scaler is able to remove the median value and utilize the 25th and 75th percentile data points to create the best normalization for our data.

Random Forest Regression Model (Supervised Learning)
We chose to implement a Random Forest Regression model to predict the calories that would be burned given user activity. We thought this would be effective because a regression model can make smart predictions without too much preprocessing and works well with structured data. Random Forest is ideal because we can experiment with combining different decision trees and then average out those predictions, which is good for preventing overfitting. This is something we discussed in our proposal, and it seems to be working well during implementation. For something like this, which could hold invalid data points like major outliers or miscalculated values, Random Forest is useful to reduce that noise.

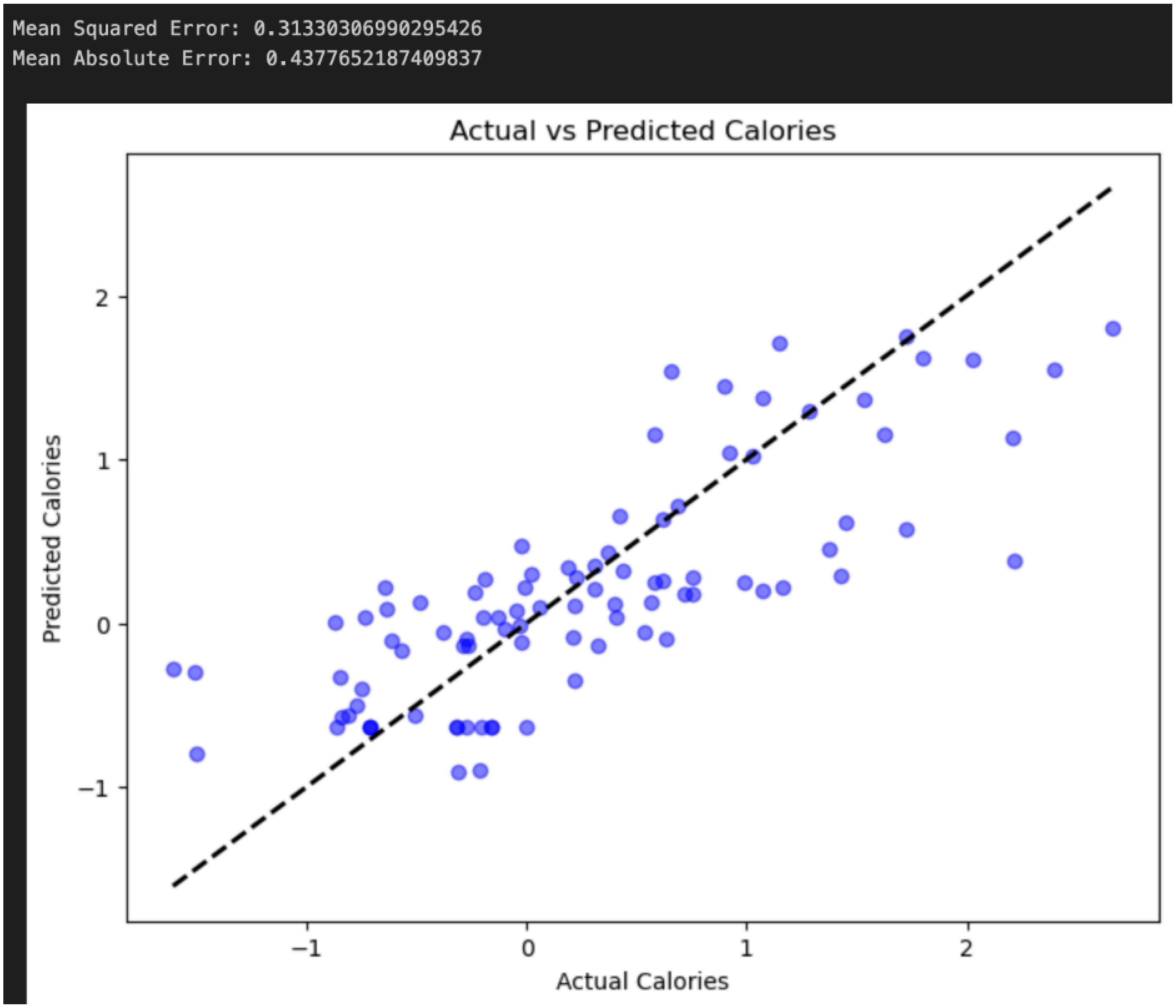
Random Forest Classification Model (Supervised Learning)
We chose to implement a Random Forest Classifier model to determine whether a workout is active or sedentary based on the user’s activity. We believed this would be effective because the model would be able to rule out any workout that is not significant to a user’s health. This classification model is also robust and would handle missing accuracy well. Additionally, it is resistant to outliers making it effective for future and scalable data.

Support Vector Regression Model (Supervised Learning)
We chose to implement a Support Vector Regression (SVR) model to predict the calories burned based on user activity. SVR is well-suited for this task because it can model complex, non-linear relationships between input features and the target variable through the use of kernel functions, such as the Radial Basis Function (RBF). This makes it a flexible choice for structured data with potentially non-linear patterns. By tuning the hyperparameters, such as the regularization parameter and kernel type, we can balance the trade-off between model complexity and

accuracy. This approach aligns with our goal of making accurate predictions while handling potential outliers and noise in the dataset effectively.

**Results/Discussion**

Visualizations

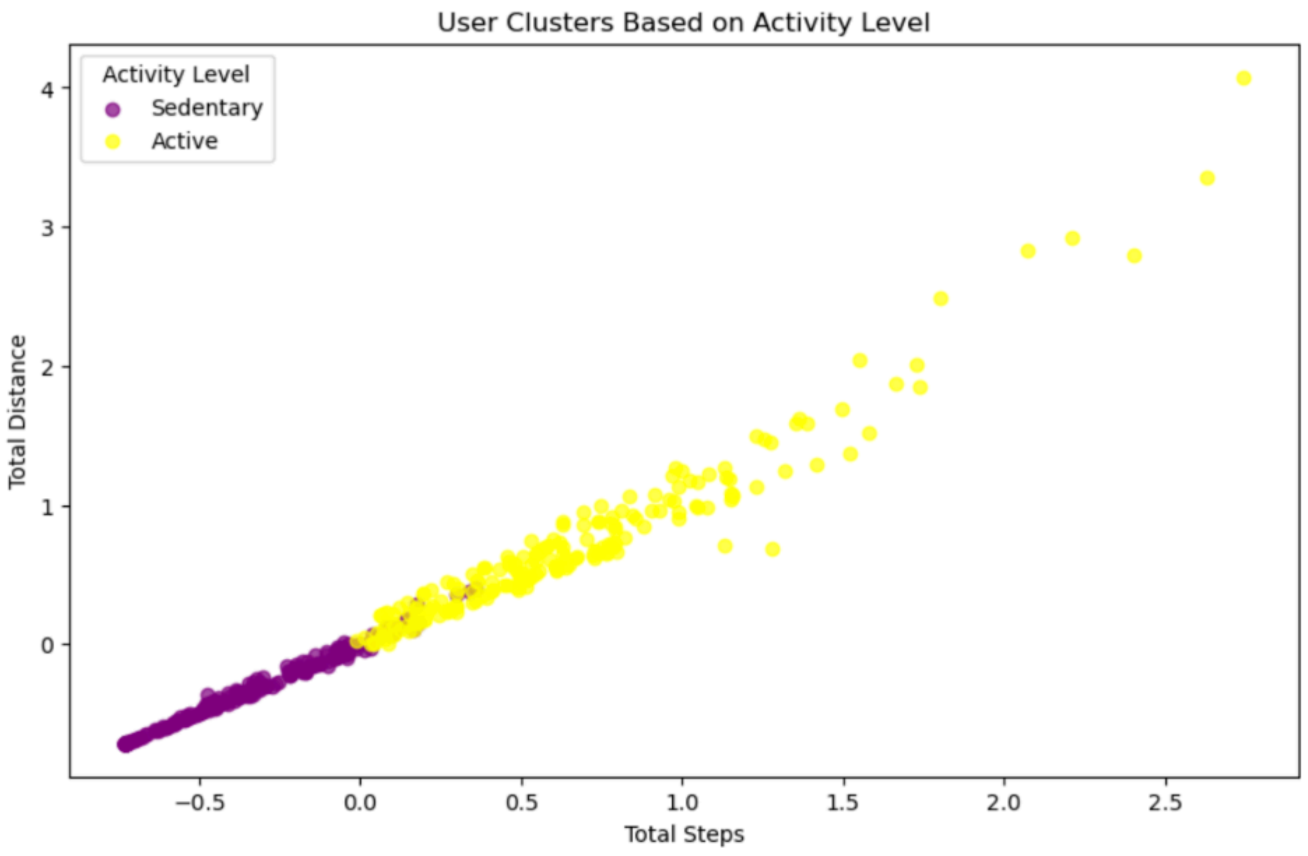


```
Mean Squared Error: 0.31330306990295426
Mean Absolute Error: 0.4377652187409837
```

This scatter plot visualizes the relationship between predicted and actual calorie values for a regression model applied to the test set. The plot spans from approximately -1 to 2 along both the x-axis (Actual Calories) and y-axis (Predicted Calories), reflecting that the data was normalized or scaled as part of the preprocessing steps. The diagonal black dashed line in the center represents the line of perfect predictions, where each point on the line would indicate when the predicted calorie value exactly matches the actual calorie value. Points that closely align with this line suggest strong predictive accuracy, whereas points that are more scattered from the line indicate larger prediction errors.

The majority of data points cluster around this ideal line, indicating that the model is reasonably effective in capturing the trend between actual and predicted calorie values. However, there is some variance, with some points deviating significantly from the line, suggesting areas where the model's predictions differ from the true values. This distribution reflects that while there is a correlation between the predicted and actual values, the model's accuracy is not perfect, and some predictions may have a margin of error.
The mean squared error (MSE) and mean absolute error (MAE) are included above the plot to provide quantitative measures of the model's performance. The MSE value of 0.3133 represents the average squared difference between the predicted and actual values, while the MAE of 0.4378 represents the average absolute difference. These metrics, in combination with the visual scatter, indicate that while the model performs reasonably well, there is room for further refinement.

For future steps, we could look to reverse the normalization or scaling to present the calorie predictions in their original units. This could provide a more interpretable comparison of the predicted vs. actual calorie values, giving clearer insights into the practical accuracy of the model in real-world terms.

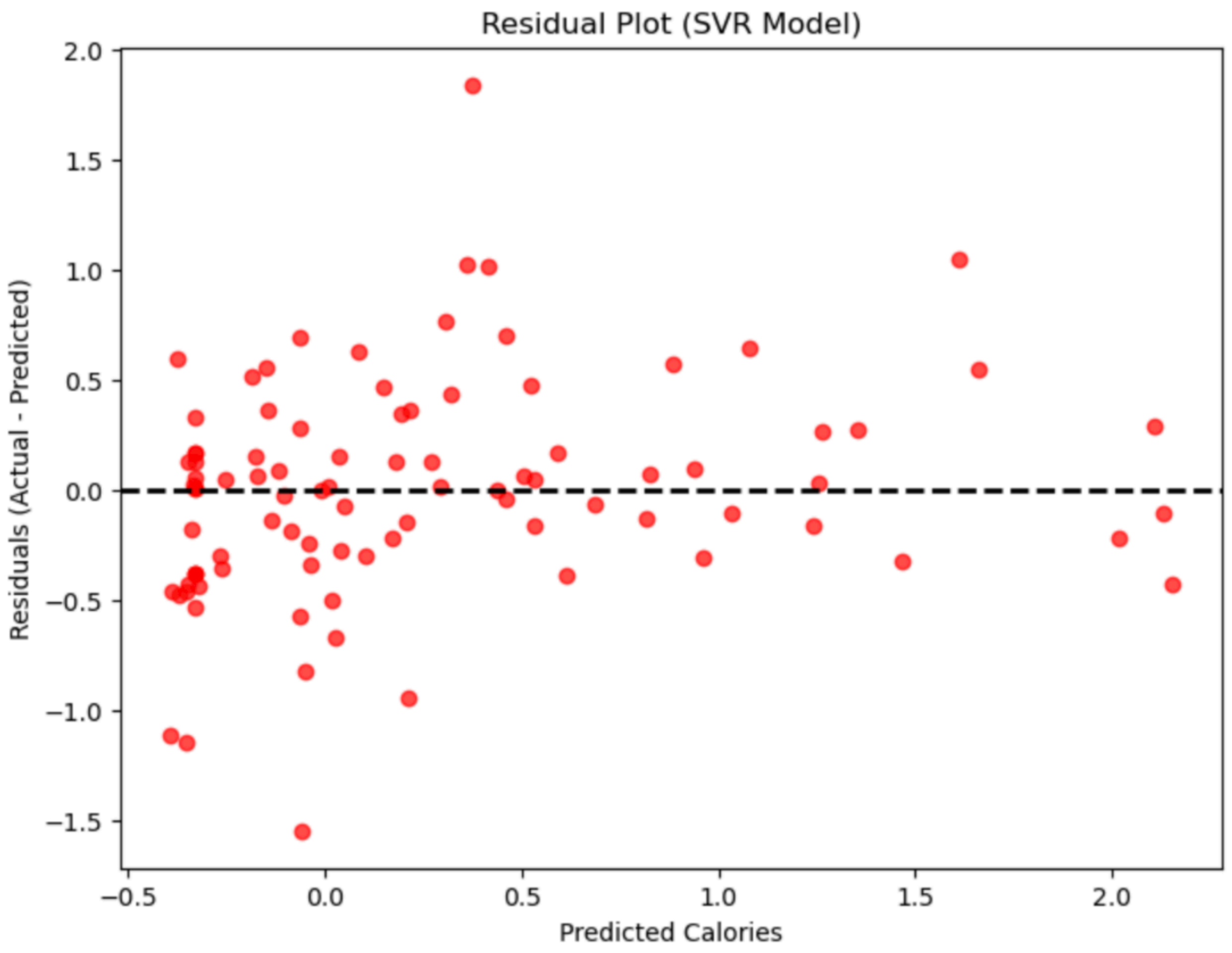


This scatter plot shows the clustering of user activity levels into two distinct groups, labeled "Sedentary" (represented by purple points) and "Active" (represented by yellow points). These clusters were produced using the KMeans clustering algorithm, an unsupervised machine-learning technique that groups data points based on similarity across specified features.

In this case, the clustering is based on two key activity metrics: total steps (x-axis) and total distance (y-axis). These metrics were likely normalized, as indicated by the values ranging around 0 to 2, which helps enhance the clustering process by ensuring features are on a similar scale.

The pattern of points following a clear diagonal trend signifies a positive correlation between total steps and total distance, which intuitively makes sense; as users take more steps, they tend to cover greater distances. This correlation reflects that step count can serve as a proxy for distance, and both metrics together are effective in distinguishing between levels of user activity. Users identified as “Sedentary” generally occupy the lower end of both total steps and total distance, indicating limited physical activity. In contrast, the “Active” group appears in higher ranges along both axes, signifying users who have higher daily movement and likely engage in more physical activity.

By visualizing the data in this way, the plot provides an understanding of how user activity varies within the dataset, allowing for quick identification of behavioral patterns in physical activity. This information could be particularly useful for applications aimed at health monitoring, fitness tracking, or personalized recommendations. For instance, insights derived from this clustering could inform targeted health recommendations, such as encouraging users in the “Sedentary” category to gradually increase their step count or distance to reach the “Active” level.



This residual plot for the Support Vector Regression (SVR) model shows that the residuals are relatively random and symmetric around the horizontal line (y = 0), showing there are no major biases such as constant overprediction or underprediction. The majority of residuals are close to zero, suggesting the model overall performs reasonably well. However, the plot highlights that the model struggles slightly with lower to mid-range calorie predictions (from 0.3 to 0.5 predicted calories), as shown by an increasing spread of residuals in this range. This widening of the residuals suggests where the variance of the errors is not constant across the range of predicted values, indicating the model may perform less consistently for lower or mid-range calorie predictions but more consistently for higher calorie predictions. Outliers—points with large residuals—suggest instances where the model's predictions deviate significantly from actual values, likely due to insufficient feature representation. To improve performance, steps such as balancing the dataset to better represent high-calorie values, feature engineering (like adding interaction terms), and hyperparameter tuning of the SVR model could be explored. Additionally, analyzing outliers and comparing the SVR model with alternative regression techniques could provide better handling of complex relationships in the data. Despite these limitations, the model demonstrates balanced errors and accuracy across most of the predicted calorie range. <br /

> Quantitative Metrics
> Our quantitative metrics include: Accuracy Score: To evaluate overall prediction correctness

> for health outcomes. F1 Score: To assess classification tasks like identifying user behavior patterns. Mean Absolute Error (MAE): To measure the accuracy of continuous predictions (e.g., estimating sleep duration). Area Under the ROC Curve (AUC): To evaluate binary health outcomes and risk prediction

Calculated Values:

- RandomForestRegressor:
  - Mean Absolute Error (MAE): 0.44
  - Mean Squared Error (MSE): 0.31
- RandomForestClassifier:
  - Accuracy Score: 0.8586956521739131
  - F1 Score: 0.7719298245614035
  - Area Under the ROC Curve (AUC): 0.9375991538868323
- Support Vector Regression:
  - Mean Squared Error (MSE): 0.247
  - Mean Absolute Error (MAE): 0.363
  - R-squared (R2): 0.692

**Analysis of Models Used**
We used a Random Forest Regressor model in order to predict calories burnt and used a Random Forest Classifier model in order to classify user activity levels.

RandomForestRegressor Model:
This model builds decision trees and then merges them. With high-dimensional datasets, the model's strength is present. It was chosen due to its effectiveness in handling outliers, reduction of noise, and feature importance. This is true since the model offers input into how certain features impact calories used, including active minutes and steps. The noise is also reduced since the Random Forest Regression model averages more than one decision tree. Since MAE is used for regression, we are able to compute the MSE and MAE for the RandomForestRegressor. The MAE allows us to get an idea of how far the predictions are from the results. The calculated values were the following:
Mean Absolute Error (MAE): 0.44
Mean Squared Error (MSE): 0.31
Since there is a lower MAE, this means that the predictions are closer to the actual values. The MSE value also suggests that the predictions of the model deviate from the true value by a small amount. Overall, this regression model helps to provide precise insights about predictions related to caloric burn. This can be used for personalized health advice.

RandomForestClassifier Model:
This model constructs the output of more than one decision tree to get a single output. The

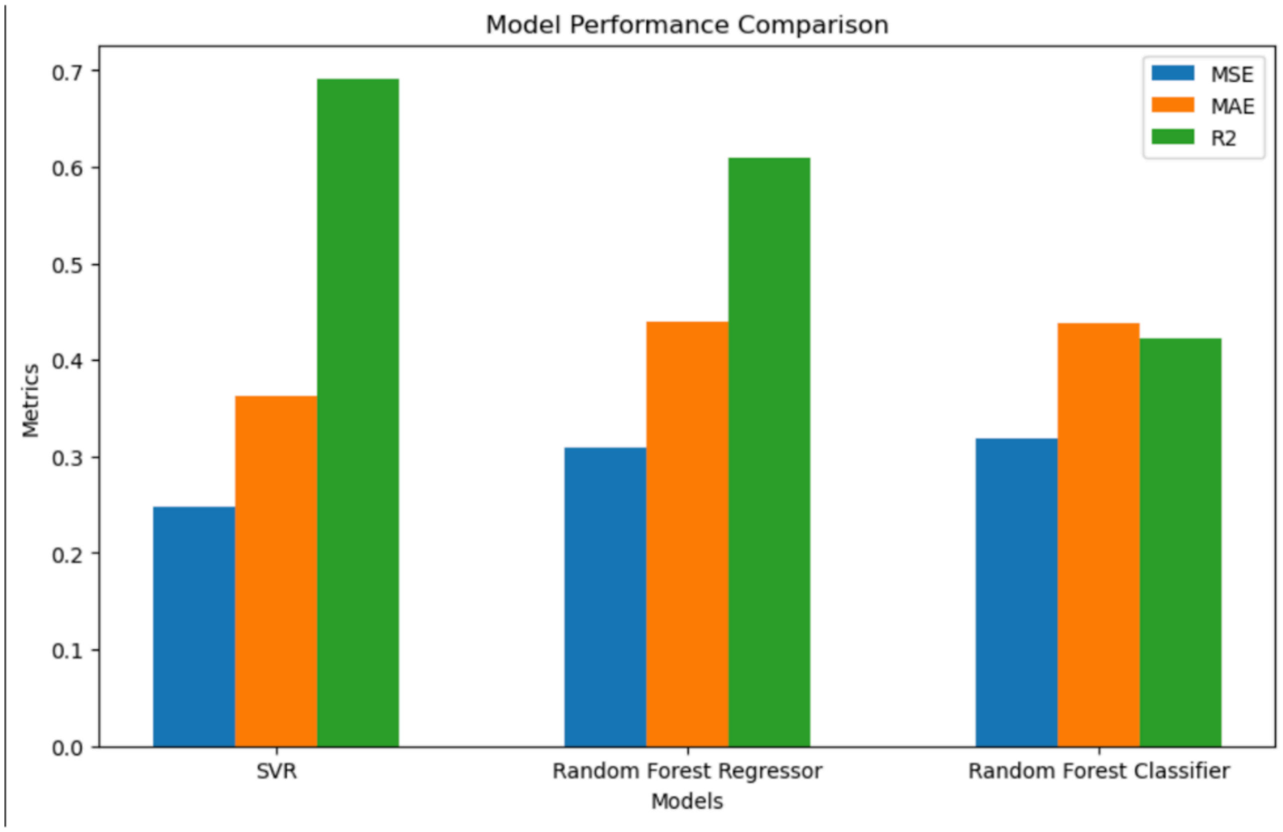
Random Forest Classification model is less prone to overfitting, can handle imbalance data, and can handle a large amount of features. This is really important to be able to tell which behaviors can classify behavior as active or more sedentary. In order to get a better understanding of how well our model can classify our data, we also analyzed classification-related metrics. This model was used to categorize user activity into segments in order to be able to support health interventions. For the RandomForestClassifier, we were able to find the following metrics to represent our results:
Accuracy Score: 0.8586956521739131
F1 Score: 0.7719298245614035
Area Under the ROC Curve (AUC): 0.9375991538868323
Overall, the accuracy rate is generally positive, F1 score indicates precision and recall balance, and the AUC being close to one can indicate that the model has a high true positive rate. These all help to underscore the high efficacy of the model in its ability to classify the activities in the context of health analytics. It is important to make sure that the results are not biased by any class imbalance.

Support Vector Regression Model:
The Support Vector Regression model helps to be able to make predictions from wearable device data. We wanted to use it to analyze caloric burn. We were able to calculate three metrics which are the Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared values. Since SVR is used for regression tasks over classification tasks, there are certain metrics that were not applicable in this context.

For the Support Vector Regression Model, we found the following results:
Mean Squared Error (MSE): 0.24743318474838744 Mean Absolute Error (MAE): 0.3632608230082016 R-squared (R2): 0.6917182540597702

The model's predictions are close to the true values, and this is supported by the low MSE value. The low MAE value also supports the fact that the predictions are around 0.3633 away from the true values. The R-squared value indicates variability in data too, but there is some room for improvement if needed in terms of factors that impact burning calories. Overall, these metrics and calculated results indicate caloric burn is predicted well with the SVR model.

**Comparison of Models**



The performance comparison between the SVR, Random Forest Regressor, and Random Forest Classifier models highlights clear differences in their effectiveness for calorie prediction through a bar graph. The Random Forest Regressor can be seen to be the best-performing model, achieving the lowest Mean Squared Error (MSE) and Mean Absolute Error (MAE), while also attaining the highest R-squared ($R^2$) score. This highlights the model's ability to minimize prediction errors and explain the variance in calorie data. The SVR model closely followed, with slightly higher MSE and MAE values, and a marginally lower $R^2$ score, indicating that it also performs well but may not capture complex relationships in the data as effectively as the Random Forest Regressor. In contrast, the Random Forest Classifier showed the weakest performance, with the highest MSE and MAE and the lowest $R^2$ score, as it is inherently designed for categorical predictions rather than continuous regression tasks.
For regression tasks like calorie prediction, the Random Forest Regressor is the most reliable model due to its ability to handle non-linear relationships and its strong performance across all metrics. The SVR model is also a strong choice and may be preferable if computational resources are less or if a simpler model is desired. The Random Forest Classifier should be reserved for tasks involving categorical outcomes, as it is not optimal for continuous predictions. Overall, the Random Forest Regressor came to be the most suitable model for this task, while SVR provides a solid alternative.

### Next Steps

Random Forest Regressor: For this model, we can improve it with hyperparameter tuning, such as changing the number of n_estimators or the max_depth. This would help to try to improve the

accuracy of the model since fine-tuning will help to reduce variance and bias. Additionally, we can clean the data to make it better for training, as normalization or standardization would allow for uniform data across all fields.
Random Forest Classifier: For this model, we can improve it by increasing the n_estimator. This can reduce variance however, it would result in more computational time. Additionally, we can also tune some other hyperparameters like min_samples_split to optimize the model performance.
Support Vector Regressor: This model was pretty effective as is considering the good MSE score. The main difference or improvement we could make is strongly tuning the hyperparameters for retraining, as that would definitely improve the model. Even then, though, to reach the current scores our model is has taken a good amount of fine-tuning, so the extent to which this current model could be improved is likely capped. Another change we can introduce is different kernel functions, which can definitely change results in a large way. The best way to further improve our SVR model though would be with more data inputs, which we'll discuss next.

As for the next steps overall with the project, we are looking to access more data fields, such as heart rate and weight logged to make more impactful predictions. This will allow us to implement newer models on slightly different data. This is a change that could greatly support all of our current models by providing more fields/inputs to make predictions with. Furthermore with the data we have, there are other predictions/classifications we can do as well, such as via KMeans Clustering, and so there is lots of room to continue experimenting. We believe had there been more checkpoints or more time for the overall project, it would've been a great opportunity to add more to our dataset by introducing these new metrics. With them, we could have entirely new predictions made or could perform the same functions and test new models versus ones previously developed.

Initially, we thought of doing Reinforcement Learning for this project, but it seems more feasible and applicable to this dataset, and our goals to implement the Supervised and Unsupervised methods we thought of. They align very well without a dataset and as we expand our data, we can further use more methods that would work. Overall, we believe our two models thus far do a good job of predicting results and we hope to improve them for the next checkpoint.

## References

[1] J. P. Burnham, C. Lu, L. H. Yaeger, T. C. Bailey, and M. H. Kollef, "Using wearable technology to predict health outcomes: a literature review," Journal of the American Medical Informatics Association, vol. 25, no. 9, pp. 1221–1227, Jun. 2018, doi: https://doi.org/10.1093/jamia/ocy082.
[2] O. H. Zahrt et al., "Effects of Wearable Fitness Trackers and Activity Adequacy Mindsets on Affect, Behavior, and Health: Longitudinal Randomized Controlled Trial," Journal of Medical Internet Research, vol. 25, no. 1, p. e40529, Jan. 2023, doi: https://doi.org/10.2196/40529.
[3] Parthiv Upreti, "Data Analyst Project For Beginner : Analysis of Fitness Tracker," Geekster Article, Jul. 08, 2024. https://www.geekster.in/articles/data-analyst-project-for-beginner-analysis-of-fitness-tracker/ (accessed Oct. 04, 2024).

[4] “The Rise of Wearable Health Devices and Their Impact on Patient Monitoring,” Sep. 08, 2023. https://www.tigahealth.com/the-rise-of-wearable-health-devices-and-their-impact-on-patient-monitoring/

Youtube Video: https://www.youtube.com/watch?v=afq0nE1yMqs

GitHub Repository: https://github.gatech.edu/ncharagulla3/ml_group28.git
Dataset: https://www.kaggle.com/datasets/arashnic/fitbit
**Gantt Chart**
https://docs.google.com/spreadsheets/d/16klzN2BFJ347EOxknwqKddX_WBE0mQqN/edit?usp=sharing&ouid=111773263312731429764&rtpof=true&sd=true
**Contribution Table**
https://docs.google.com/document/d/1FUEkT8NKHrq8ExcbhZQXgqll2Q9LJ_ElxtFRBXCAkG0/edit?usp=sharing

**Note** We tried to embed the Gantt Chart and Contribution Table into the README, but there were tons of formatting errors we ran into that we weren’t able to fix, so we opted to just use an external link for simplicity.

### File Directories

- /analysis – Model analysis and scoring files to output metrics like accuracy and F1 Score.
    - /analysis/classificationMetric.py - Metric calculation on Classifier Model
    - /analysis/regressionMetric.py - Metric calculator on Regressor Model
    - /analysis/svrMetric.py - Metric calculator on SVR Model
- /data - data files
    - /data/dailyActivity.csv - compiled and cleaned dataset from Kaggle
    - /data/data_processing.py - file containing preprocessing methods to clean and output to dailyActivity.csv
- /midterm_images - images used in README.md report
    - /midterm_images/graph - scatterplot of regression model
    - /midterm_images/graph2 - clustering visualization
    - /midterm_images/graph3 - svr visualization
    - /midterm_images/graph4 - model comparison
    - /midterm_images/pix1 - image of robust scaler method
- /model_training - files to train/create models
    - /model_training/forest_train.py - training Random Forest Regressor Model
    - /model_training/random_forest_activity_classifier.py training Random Forest Classifier Model
    - /model_training/svr.py training SVR model
- /trained_forest_models - pickled trained models

- /trained_forest_models/forest_regression_v1.pkl - trained Regressor model, v1
- /trained_forest_models/forest_regression_v2.pkl - trained Regressor model, v2
- /trained_forest_models/random_forest_classifier.pkl - trained Classifier model
- /trained_forest_models/svr.pkl - trained SVR model

- /visualizations - visualizations and graphs of models
    - /visualizations/cluster_vis.py - Clustering scatterplot
    - /visualizations/regress_vis.py - Regressor model scatterplot
    - /visualizations/model_visuals.ipynb - SVR model visualization code