Enhancing Activity Recognition and Sensor Placement Optimization:

Imputed Data for PAMAP2 dataset

Code link

PROBLEM STATEMENT AND IMPORTANCE

Problem statement

- Develop an effective activity monitoring system using wearable sensors
- identify the best sensor placement for accurate activity recognition
- Impute missing values in data
- Build a classifier to detect activity using optimal sensor placement

Importance/ Motivation

Wearable sensors are becoming increasingly popular for activity monitoring in various domains such as healthcare, sports, and fitness

Data Set Description

- Dataset : *PAMAP2 [Physical Activity Monitoring]* Instances: 3.85 M

- The PAMAP2 dataset provides comprehensive data on **human physical activity**, recording the actions of **nine individuals** performing **18 different activities**, equipped with three **inertial measurement units (IMUs) and a heart rate monito**.

- This dataset is a valuable resource for researchers and professionals in various fields, offering insights into activity identification, motion assessment, and health tracking.

- The data files contain 54 columns: each line consists of a timestamp, an activity label (the ground truth) and 52 attributes of raw sensory data, which also contains null values.

Preprocessing

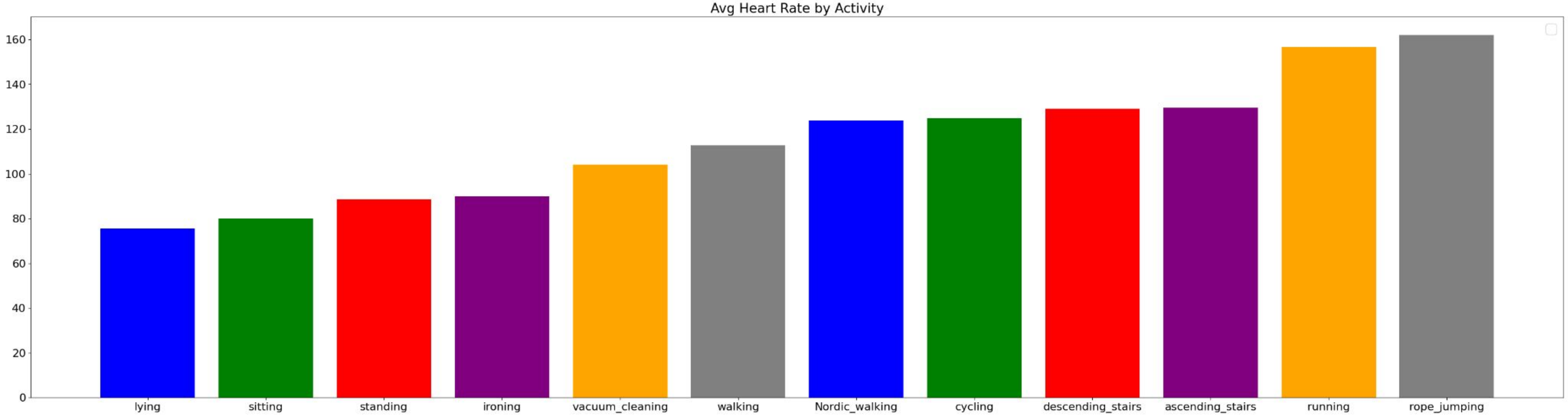
Hypothesis Testing

H_0: *Heart rate for cumbersome activities is same as heart rate for other activities*

- Took a random sample from data of size 300
- Tested using Z-test

VALIDATION TESTS PERFORMED

- **Mean calculation:** mean for activities like rope jumping and running was much higher
- **Plotting mean heart rate with activities**



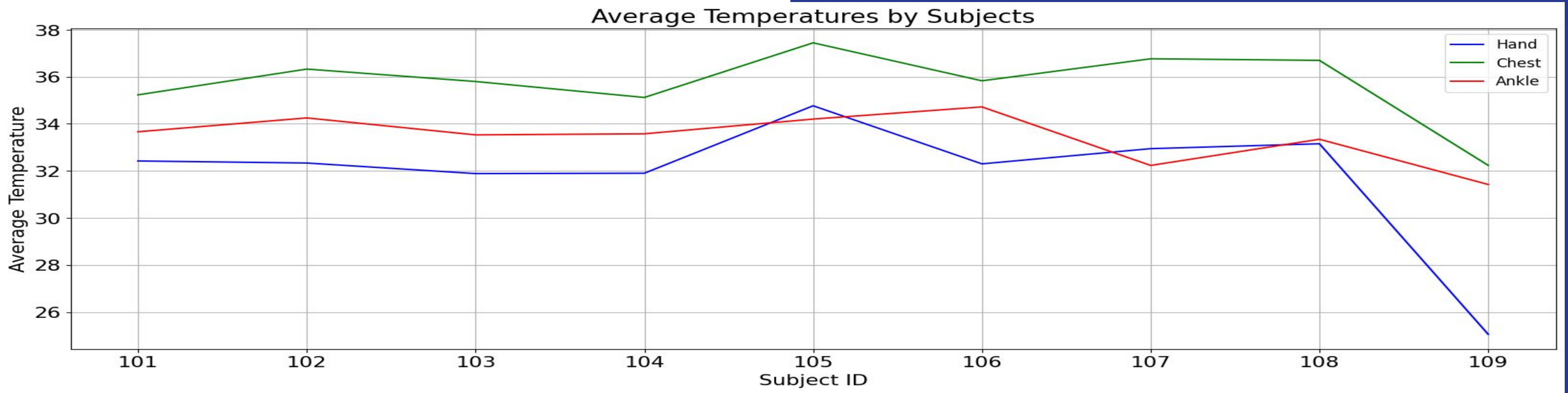
Result : Hypothesis rejected with level of significance 5%

H_0: *Mean temperature is greater in chest than ankle and hand for all subjects*

- Took a random sample from data of size 300
- Tested using Z-test

VALIDATION TESTS PERFORMED

- **Mean calculation:** mean for chest higher for all participants
- **Plotting Temperature with participants for all sensors**



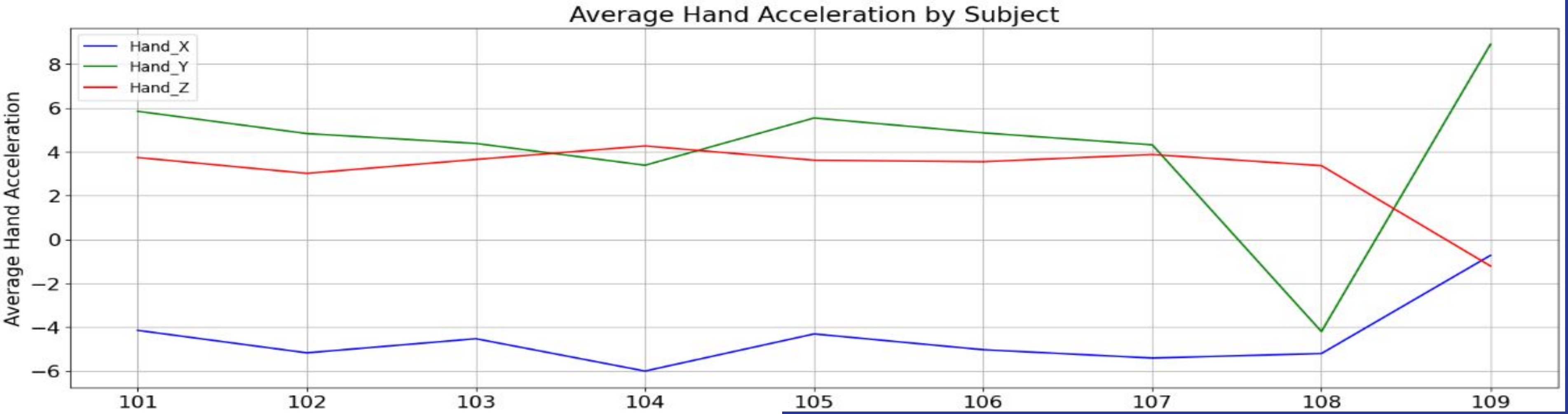
Result : Hypothesis accepted with level of significance 5%

H_0: *Mean hand acceleration is least in x direction*

- Took a random sample from data of size 300
- Tested using Z-test

VALIDATION TESTS PERFORMED

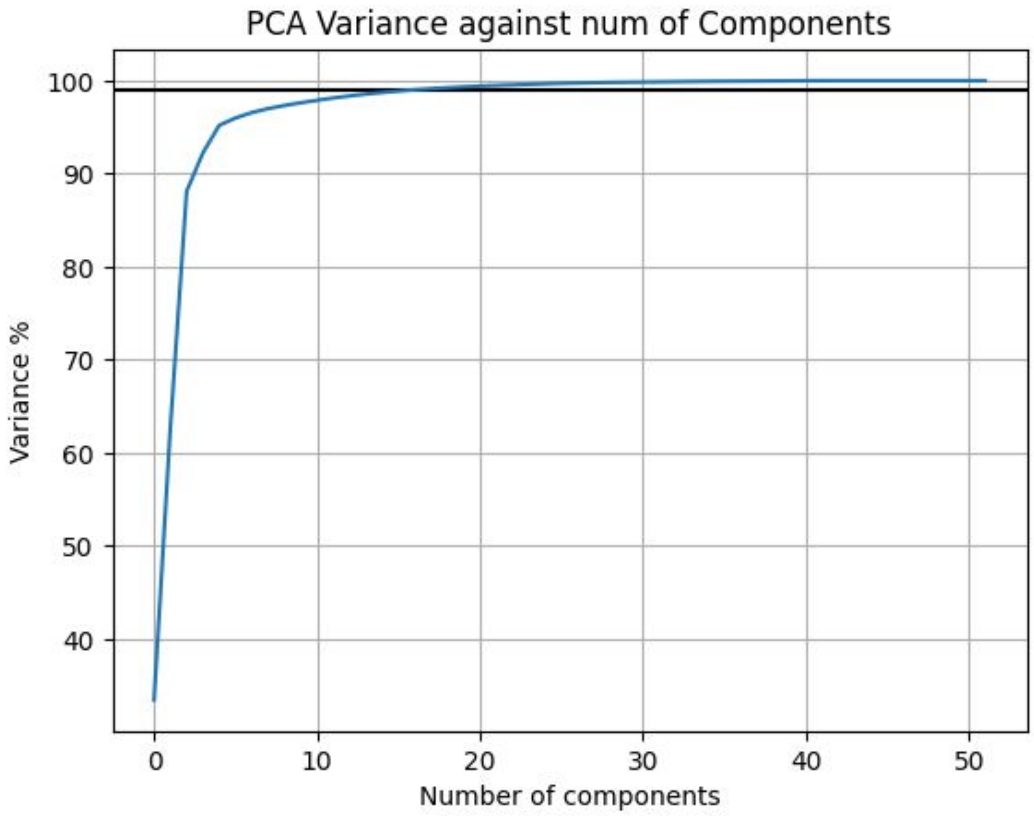
- **Checked using python script**
- **Plotting x,y,z hand acceleration with subjects**



Result: Hypothesis accepted with level of significance 5%

ACTIVITY RECOGNITION

Dimensionality reduction using PCA



Usually 90-98% of the variance will explain our data really well.

We see that around 15-17 number of components have 99% variance

So, we take only 17 components

Key Findings

Final Observation

We can see the data isn't balanced:

Subject 109 as less sampels then all others subjects.

rope_jumping activity as less samples then other activities

MCAR (Missing Completely at Random):

The code aims to assess whether the missing data in a DataFrame is distributed randomly and unrelated to both observed and unobserved data.

The chi-square test is a statistical test used to assess the independence of two categorical variables. In this case, it's applied to evaluate the independence between the presence or absence of missing data (binary variable) and the observed data in each column. The null hypothesis (H0) of the chi-square test is that the missingness is independent of the observed data, i.e., the data is MCAR. The alternative hypothesis (H1) is that the missingness is not independent, indicating a pattern other than MCAR.

DATA IS CONSISTENT WITH MCAR

Sampling Data

Huge dataset with around 3M instances
Any decent model would take a lot of time
So, we have to do random sampling for sampling the data points

We use stratified sampling i.e. sample random data points for each subject and do activity recognition for them.

We run the softmax regression, gaussian naive bayes, KNN classifier, Adaboost Classifier, Random Forest and Multilayer perceptron

Gaussian Naive Bayes

- Outputs a probability distribution of a point being in a particular class
- Choosing the class with max probability does multiclass classification
- Scalable because it can handle large datasets efficiently
- Since our heart rate was Independent of other features, it makes sense to use a model that uses the information of independence of these features

Softmax Regression

- Outputs a probability distribution of a point being in a particular class
- Choosing the class with max probability does multiclass classification
- Introduces Non linearity into the classification
- Similar to logistic regression with multiple classes

Random Forest Classifier

- Random Forest provides robust and accurate classification results.
- Effective on datasets with diverse, complex features.
- Identifies and ranks influential features for better insights.
- Ensemble approach mitigates overfitting by combining multiple trees.

KNN Classifier

- KNNs have been used in other papers that talk about human activity recognition
- It makes no assumption about the data or any properties of the data as it is an unsupervised model.
- Since we know the number of classes, it makes sense to use this

Adaboost Classifier

- Less Prone to overfitting than other machine learning algorithms
- Adaptive in nature as it focuses on misclassified points
- Is known for the tasks of enhancing the accuracy of weak classifiers

MLP

- MLP provides efficient estimates with low variance.
- Estimates converge to true values with increasing sample size.
- MLP remains consistent under parameter estimate transformations.
- MLP achieves the smallest asymptotic variance in large samples.

Final Results

	Model	Validation-Accuracy	Accuracy	Precision	Recall	F1-score	Training Time
0	SoftmaxRegression	97.2249+-0.3922	90.5696	92.11	91.0688	91.0772	426.7078490257263s
1	KNeighborsClassifier	97.1483+-0.4275	91.1298	93.0108	90.2071	91.0186	399.1330380439758s
2	GaussianNB	87.6459+-2.7594	88.422	89.0196	88.255	87.984	113.7996406555175s
3	RandomForestClassifier	98.6124+-0.3688	97.8525	96.8948	96.4397	96.563	311.94351959228516s
4	AdaBoostClassifier	86.0853+-0.0	79.4585	79.4858	74.556	73.4033	122.0897598266016s
5	MLPClassifier	97.2918+-0.4918	92.7171	93.8333	92.9239	92.9769	44.42853665351868s

So, we see that **RandomForest Classifier** performs best

OPTIMAL SENSOR PLACEMENTS

Optimal Sensor Placement

- Identified **RandomForest Classifier** as the most effective model.
- Sensors placed on the wrist, chest, and ankle.
- Currently using RandomForest Classifier for classification.
- Exploring optimal sensor placement by classifying activities with single-sensor features.

Optimal Sensor Placement

```
# Hand DataFrame
hand_columns = [
    'hand_temperature',
    'hand_3D_acceleration_16_x', 'hand_3D_acceleration_16_y', 'hand_3D_acceleration_16_z',
    'hand_3D_acceleration_6_x', 'hand_3D_acceleration_6_y', 'hand_3D_acceleration_6_z',
    'hand_3D_gyroscope_x', 'hand_3D_gyroscope_y', 'hand_3D_gyroscope_z',
    'hand_3D_magnetometer_x', 'hand_3D_magnetometer_y', 'hand_3D_magnetometer_z',
    'hand_4D_orientation_x', 'hand_4D_orientation_y', 'hand_4D_orientation_z', 'hand_4D_orientation_w'
]
hand_df = pd.DataFrame(train_df[hand_columns])

# Chest DataFrame
chest_columns = [
    'chest_temperature',
    'chest_3D_acceleration_16_x', 'chest_3D_acceleration_16_y', 'chest_3D_acceleration_16_z',
    'chest_3D_acceleration_6_x', 'chest_3D_acceleration_6_y', 'chest_3D_acceleration_6_z',
    'chest_3D_gyroscope_x', 'chest_3D_gyroscope_y', 'chest_3D_gyroscope_z',
    'chest_3D_magnetometer_x', 'chest_3D_magnetometer_y', 'chest_3D_magnetometer_z',
    'chest_4D_orientation_x', 'chest_4D_orientation_y', 'chest_4D_orientation_z', 'chest_4D_orientation_w'
]
chest_df = pd.DataFrame(train_df[chest_columns])

# Ankle DataFrame
ankle_columns = [
    'ankle_temperature',
    'ankle_3D_acceleration_16_x', 'ankle_3D_acceleration_16_y', 'ankle_3D_acceleration_16_z',
    'ankle_3D_acceleration_6_x', 'ankle_3D_acceleration_6_y', 'ankle_3D_acceleration_6_z',
    'ankle_3D_gyroscope_x', 'ankle_3D_gyroscope_y', 'ankle_3D_gyroscope_z',
    'ankle_3D_magnetometer_x', 'ankle_3D_magnetometer_y', 'ankle_3D_magnetometer_z',
    'ankle_4D_orientation_x', 'ankle_4D_orientation_y', 'ankle_4D_orientation_z', 'ankle_4D_orientation_w'
]
ankle_df = pd.DataFrame(train_df[ankle_columns])
```

Feature Distribution for each sensor data

Optimal Sensor Placement

- The average accuracy for single sensors was notably lower than the accuracy obtained using all sensors together.
- So, we conclude that for best results we need all the three sensors
- Different activities have different involved body parts.
- For some activities the recognition rate was lesser like ascending stairs and descending stairs

Novelty

- Performed the MCAR test, heart rate is independent of the other attributes which had 90% missing values
- Polynomial interpolation provided a more contextually relevant non-linear imputation compared to the global linear nature of mean imputation.
- Mean imputation may introduce bias, especially in the presence of non-linearities.
- Polynomial interpolation adapts to the local variations in the data, offering a dynamic imputation strategy

Questions?