




# Final Project Presentation



Team 1  
Amanda, Andrew, Avi, and Ryan



# Agenda

---

1. Project Summary
2. Phase 1 problem & solutions
3. Phase 2 problem & solutions
4. Phase 3 problem & solutions
5. What We Learned

# Project Summary

---

- Main goal is to understand how multivariate time series data works using supervised and unsupervised algorithms
- **Time Series Data:** Refers to data, period to period, in an ordered sequence of events.
- **Multivariate Data** refers to how data with multiple variables change overtime with interdependencies to each other.
- By utilizing the Summer Time Algorithm, which makes the data into fixed lengths, we can understand how to detect and classify patterns using the NATOPS data

# NATOPS and the Summertime Algorithm

---

- **NATOPS** refers to the dataset we had to organize, it's comprised of 6 gestures with 20 subjects and over 400 samples.
- **The Summer Time Algorithm** summarizes the data using Gaussian Mixture Models, Bayesian techniques, K-Clustering and Neural Network Classifiers
- Which ones we use will be described in other slides
- Some challenges that the algorithm pose are data quality sensitivity, variable length sequences and a high computational cost

# Phase 1: Objective

---

## **Objective for phase 1:**

Prepare convert the raw .arff files from the previously mentioned NATOPS dataset into an organized CSV with the following:

- **All 24 Dimensions (features)**
- **Sample IDs (sid)**
- **Gesture class labels (class)**
- **Training/test split indicators (new column, is\_test)**

# Phase 1: How did we do it?

---

1. **Loaded** all 24 .arff files for both the **TRAIN** and **TEST** sets using a for loop.  
Each file corresponds to one sensor dimension (e.g., hand, wrist, elbow — across 3D coordinates).
2. **Dropped and separated the class labels:**
  - The classAttribute column was **dropped** during the initial load of each dimension file to avoid redundancy.
  - **Class labels were then loaded separately** from NATOPS\_TRAIN.arff and NATOPS\_TEST.arff.
  - A decoding function was used to convert labels from bytes to float values (if needed).

# Phase 1: How did we do it?

---

## 3. Merged all dimensions horizontally:

- Created a single wide table per sample by concatenating the 24 dimension-specific DataFrames.
- Ensured all 51 time steps across dimensions were correctly aligned.

## 4. Reshaped data into long format:

- Transformed each sample into 51 rows (1 per time step), each with 24 features.
- Added sid (sample ID), class, timestep, and is\_test (0 = train, 1 = test).

# Phase 1: How did we do it?

## Final output:

- Dropped the timestep column.
- Saved the processed data as natops\_processed.csv.
- Printed the first 5 rows to verify structure and values.

natops\_processed.csv

✓ CSV saved as 'natops\_processed.csv' with proper class labels, 'sid', and 'is\_test' flags.

	fea1	fea2	fea3	fea4	fea5	fea6	fea7	fea8	fea9	fea10	fea11	...	fea17	fea18	fea19	fea20	fea21	fea22	fea23	fea24	sid	class	is_test
0	-0.372758	-1.821679	-0.846321	0.465208	-2.015072	-0.839242	-0.564097	-0.796225	-0.149604	0.599967	-0.823936	...	-1.534954	-0.673190	-0.536343	-1.626957	-0.594337	0.619205	-1.771773	-0.810086	1	4.0	0
1	-0.367844	-1.841987	-0.846325	0.467033	-2.007557	-0.838151	-0.564499	-0.797622	-0.150012	0.597535	-0.822723	...	-1.532795	-0.671919	-0.533816	-1.642514	-0.605328	0.617045	-1.796660	-0.818863	1	4.0	0
2	-0.378445	-1.821358	-0.839571	0.471135	-2.010042	-0.832021	-0.563753	-0.795704	-0.151608	0.597007	-0.820954	...	-1.532478	-0.671555	-0.526319	-1.697145	-0.624302	0.624789	-1.738568	-0.788060	1	4.0	0
3	-0.386751	-1.845643	-0.848031	0.506153	-2.032552	-0.841696	-0.565008	-0.790238	-0.152350	0.599099	-0.822648	...	-1.535441	-0.672198	-0.554538	-1.644413	-0.602884	0.634100	-1.749744	-0.816695	1	4.0	0
4	-0.417101	-1.941721	-0.885500	0.611207	-1.953282	-0.902529	-0.573550	-0.799730	-0.169575	0.606181	-0.826938	...	-1.482552	-0.659393	-0.576196	-1.763092	-0.694843	0.680086	-1.664565	-0.857897	1	4.0	0

[5 rows x 27 columns]



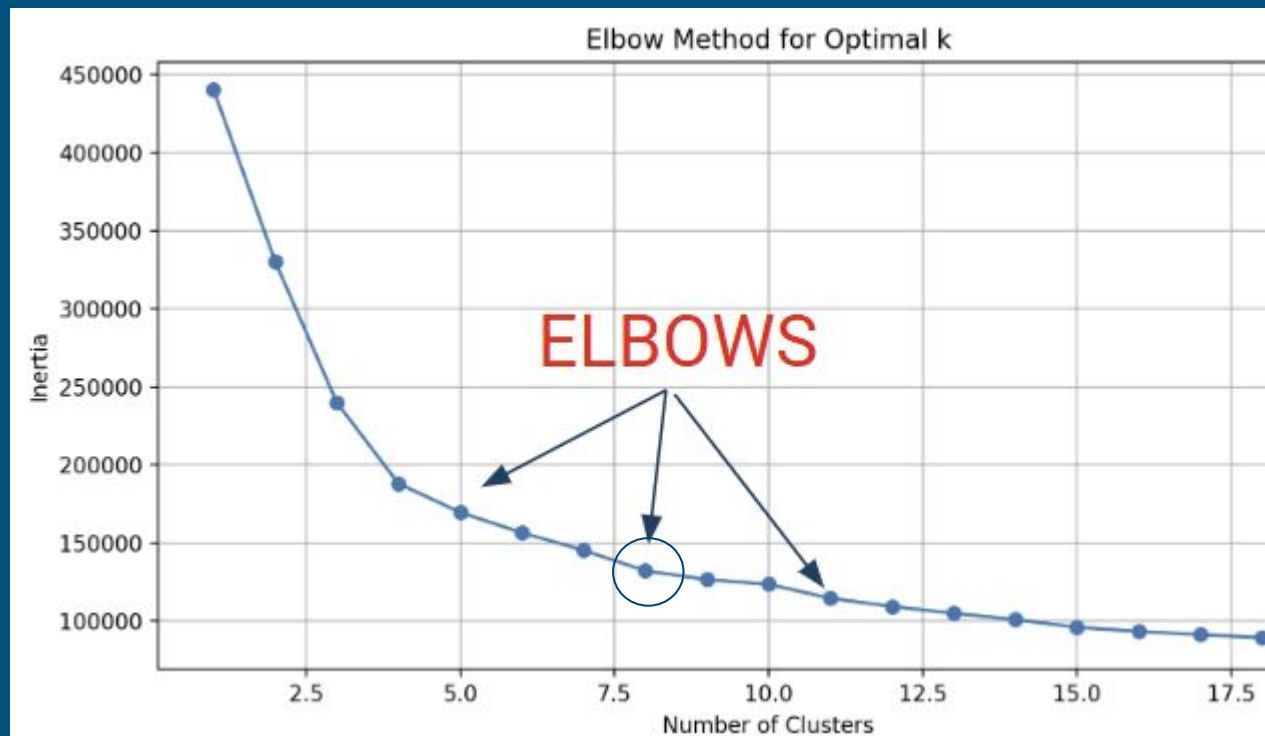
# Phase 2: Find the Clusters

---

- We determined that the data to cluster is each time step based on its features
- Since we are using an unsupervised method to cluster the data, we normalized the atomic units using `StandardScaler()` to find the mean and std over both training and testing data.
- We used the elbow method to find the optimal number of clusters for the Kmeans function
- Kmeans determines the centroids of clusters based on the Euclidean distance of each timestep

# Elbow Method

- Kmeans inertia is determined by the distance of each data point to its cluster's centroid



- The elbow method is the graphed clusters and their inertia. The elbow is the point where the inertia plateaus on the graph and is the optimal number of clusters to separate the data. We initially decided 8 clusters.

# Cluster Ratios

---

- We formatted the each time series to a single row that showed the ratios of number of timesteps in that series belonged to which cluster.
- We also included the sid, is\_test, and class values for the time series
- This data was then exported to CSV file name “natops\_cluster\_ratios.csv”

sid	is_test	class	cluster_0_ratio	...	cluster_7_ratio	cluster_8_ratio	cluster_9_ratio	cluster_10_ratio
1	0	4.0	0.372549	...	0.000000	0.000000	0.058824	0.196078
2	0	3.0	0.529412	...	0.000000	0.000000	0.176471	0.000000
3	0	3.0	0.627451	...	0.000000	0.000000	0.078431	0.000000
4	0	4.0	0.313725	...	0.000000	0.176471	0.058824	0.176471
5	0	3.0	0.450980	...	0.039216	0.000000	0.176471	0.000000

# Phase 3

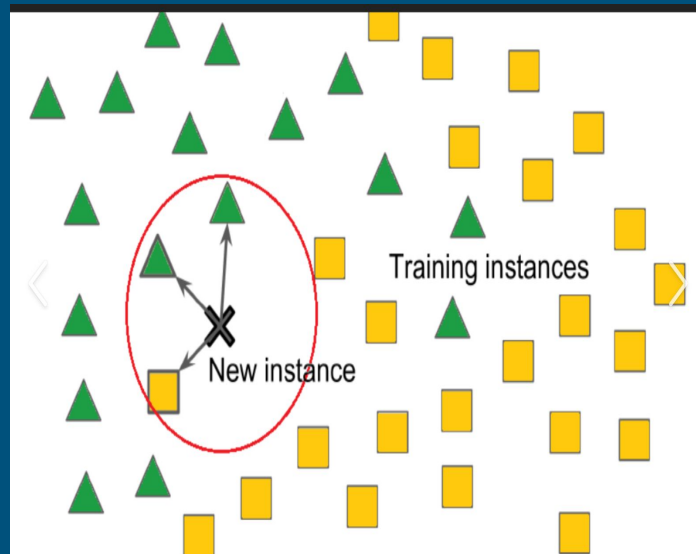
**GOAL:** "Apply supervised learning to evaluate whether the clusters generated through unsupervised learning could effectively categorize the class labels (I have a command; All clear; Not clear; Spread wings; Fold wings; Lock wings).

**Classification Model: KNN(K nearest Neighbors):**

- Measure distances between a test point and all training points.
- Pick the k nearest neighbors based on those distances

(we chose 5).

- Assign the most common class among the neighbors to the test point.

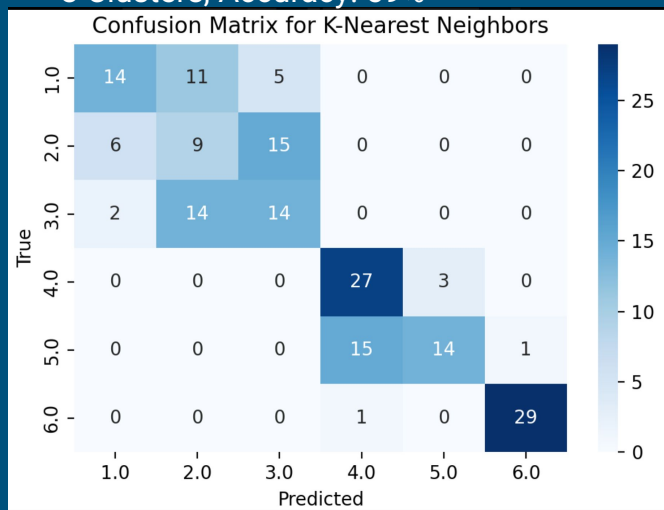


# Evaluation of Our KNN

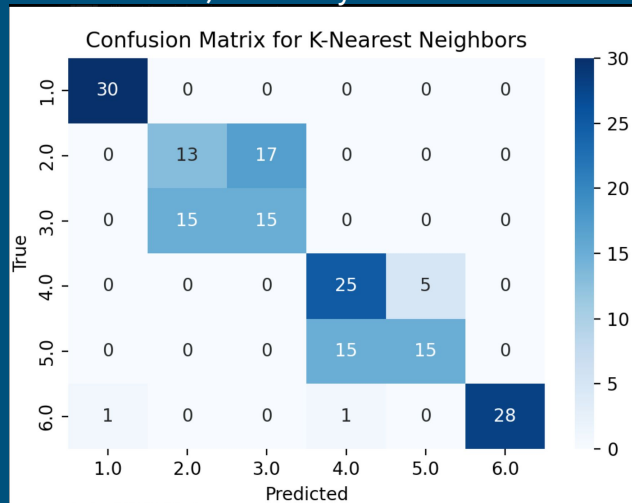
## Evaluation metrics:

- **Confusion Matrix:** A table that shows how well a classification model performs by comparing predicted vs. actual class labels.
- **Accuracy:** The percentage of correct predictions out of all predictions made.
- **6 classes for NATOPS:** I have a command(1.0); All clear(2.0); Not clear(3.0); Spread wings(4.0); Fold wings(5.0); Lock wings(6.0)

8 Clusters; Accuracy: 59%



11 Clusters; Accuracy: 70%



# Big Takeaways

---

- The number of clusters you pick has an impact on a models predictive results.
- The Elbow method is an important mechanism for choosing appropriate number of clusters
- The number of neighbors you pick for a KNN model has an impact on its results.
- Data not preprocessed accurately will negatively affect a models performance