**PROJECT: Classifying Cow’s activities**

**AIM : Classify Cow’s activities into 9 categories based on Data collected from IMU SENSORS**

**Data**

**IMU Data (Accelerometer, Gyroscope, Magnetometer)**

**What is IMU?**Inertial measurement unit, used to describe a collection of measurement tools, when installed in some device, catches movement with the help of accelerometer, gyroscope and magnetometer, in 3d space.

**Variable names:-**

* acc\_x,acc\_y,acc\_z: accelerometer output for all 3 dimensions movement.
* gyr\_x,gyr\_y,gyr\_z: gyroscope outputs, it measures rotation, rotation rate (angular velocity).
* mag\_x,mag\_y,mag\_z: magnetometer outputs, catches magnetic field around the device.
* All three (Acc, Gyr, Mag) gives output in different SI Units i.e The scale for all three are different, so Data must be normalized

***Classes and their Encoded values:-***

* eating = 1
* drinking = 2
* walking = 3
* standing =4
* lying = 5
* ruminating standing = 6
* ruminating lying = 7
* grooming = 8
* idle/other = 9

**Contents :**

1. Dataset Information
2. Exploratory Data Analysis (EDA)
3. Feature Engineering
4. Modeling
5. Conclusion
6. **Dataset Information:**
   * Importing the common libraries such as numpy, pandas, matplotlib, seaborn.
   * Importing and loading the Dataset.
   * Viewing the dataset.
   * Fetching the information about the data, i.e dtype, null values if any.
7. **Exploratory Data Analysis:**
   * Checking the number of rows and columns in the data
   * Getting to know the column/ feature names
   * Creating a table that consists of feature name, dtypes, missing values, number of unique values
   * Getting the statistical insights form the data
   * Understanding the target variable by plotting a bar graph
   * Understanding the target variable by plotting a pie chart
   * Checking for Correlation
   * Splitting the variables into independent and dependent variables
   * Checking for outliers
8. **Feature Enginnering(Data Preprocessing)**
   * Replacing the outliers with median values
   * Normalizing

All three (Acc, Gyr, Mag) gives output in different SI Units i.e The scale for all three are different, so Data must be normalized.

* + Feature selection, selecting kbest features using chi2
  + Creating a new\_df with x\_normalized and y

1. **Modeling**
   * Lets first import few libraries like train\_test\_split, roc\_auc\_score, f1\_score, precison\_score, recall\_score., etc…
   * Splitting the data into train and test
   * Data Balancing using NearMiss :
     1. In order to cope with imbalanced data, there are 2 options :
     2. Undersampling : Trim down the majority samples of the target variable.
     3. Oversampling : Increase the minority samples of the target variable to the majority samples.
     4. we have decided to go with undersampling.
     5. For data balancing, we will use imblearn.
   * pip statement : !pip install imbalanced-learn
   * Fit the x\_train and y\_train to near\_miss
2. Logistic Regression

Observations:

* + Logistic Regression Accuracy : 0.991176097375279
  + Logistic Regression f1 score : 0.99
  + Logistic Regression Precision score : 0.99
  + Logistic Regression recall score : 0.99

1. Decision Tree Classifier

Observations:

* + Decision Tree Classifier Accuracy : 1.00
  + Decision Tree Classifier f1 score : 1.00
  + Decision Tree Classifier Precision score : 1.00
  + Decision Tree Classifier recall score : 1.00

1. Random Forest Classifier

Observations:

* + Random Forest Classifier Accuracy : 1.00
  + Random Forest Classifier f1 score : 1.00
  + Random Forest Classifier Precision score : 1.00
  + Random Forest Classifier recall score : 1.00

1. K Nearest Neighbours:

Observations:

* + KNN Accuracy : 1.00
  + KNN f1 score : 1.00
  + KNN Precision score : 1.00
  + KNN recall score : 1.00

1. Support Vector Classifier

Observations:

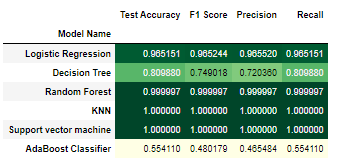
* + Support Vector Classifier Accuracy : 1.00
  + Support vector Classifier f1 score : 1.00
  + Support vector Classifier Precision score 1.00
  + Support vector Classifier recall score : 1.00

1. Ada Boost Classifier

Observations:

* + AdaBoostClassifier Accuracy : 0.55
  + AdaBoostClassifier f1 score : 0.47
  + AdaBoostClassifier Precision score : 0.47
  + AdaBoostClassifier recall score : 0.55

1. Creating a table that consists of all the accuracy scores, mean cross validation score, f1 score, precision score, recall score.



1. **Conclusion:**

We see that almost all the algorithms expect adaboost classifier gives almost 99% accuracy.