**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](#Dataset_info)
2. [Exploratory Data Analysis (EDA)](#EDA)
3. [Feature Engineering](#FeatureEnginnering)
4. [Modeling](#Modeling)
5. [Conclusion](#conclusion)
6. **Dataset Information:**
   * Importing the common libraries such as numpy, pandas, matplotlib, seaborn.
   * Importing and loading the Dataset.
   * Concatenating all the loaded dataframes into one dataframe.
   * 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…
   * As the dataset is huge,
     1. Lets take sample data from the population data
     2. Build a different model on each sample set
     3. Check which model gives good results
   * Data Balancing using Smote :
     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 Oversampling.
     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. Sample1 – df1 - Logistic Regression

Sample of 40,000

Observations:

* + Logistic Regression Accuracy : 0.31875
  + Logistic Regression F1-score (average = weighted): 0.33
  + Logistic Regression Precision (average = weighted): 0.44
  + Logistic Regression Recall (average = weighted): 0.32

1. F1 score

average = 'weighted':

Calculate metrics for each label, and find their average weighted by support (the number of true instances for each label). This alters ‘macro’ to account for label imbalance; it can result in an F-score that is not between precision and recall.

2. precision

average = 'weighted'

Calculate metrics for each label, and find their average weighted by support (the number of true instances for each label). This alters ‘macro’ to account for label imbalance; it can result in an F-score that is not between precision and recall.

3. recall

average = 'weighted'

Calculate metrics for each label, and find their average weighted by support (the number of true instances for each label). This alters ‘macro’ to account for label imbalance; it can result in an F-score that is not between precision and recall. Weighted recall is equal to accuracy.

1. Sample2 – df2 - Decision Tree Classifier

Sample of 40,000

Observations:

* + Decision Tree Classifier Accuracy : 0.94
  + Decision Tree Classifier f1 score (average = weighted): 0.89
  + Decision Tree Classifier Precision score (average = weighted): 0.90
  + Decision Tree Classifier recall score (average = weighted): 0.88

1. Sample3 – df3 - Random Forest Classifier

Sample of 40,000

Observations:

* + Random Forest Classifier Accuracy :0.92
  + Random Forest Classifier f1 score (average = weighted): 0.92
  + Random Forest Classifier Precision score (average = weighted): 0.92
  + Random Forest Classifier recall score (average = weighted): 0.92

1. Sample4 – df4 - K Nearest Neighbours:

Sample of 40,000

Observations:

* + KNN Accuracy : 0.76
  + KNN f1 score(average = weighted) : 0.78
  + KNN Precision score(average = weighted) : 0.79
  + KNN recall score(average = weighted) : 0.77

1. Sample5 – df5 - Support Vector Classifier

Sample of 40,000

Observations:

* + Support Vector Classifier Accuracy : 0.67
  + Support vector Classifier f1 score(average = weighted) : 0.68
  + Support vector Classifier Precision score(average = weighted): 0.71
  + Support vector Classifier recall score(average = weighted) : 0.67

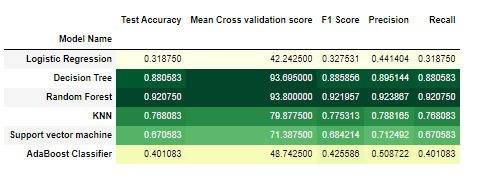
1. Sample6 – df6 - Ada Boost Classifier

Sample of 40,000

Observations:

* + AdaBoostClassifier Accuracy : 0.40
  + AdaBoostClassifier f1 score(average = weighted) : 0.43
  + AdaBoostClassifier Precision score(average = weighted) : 0.51
  + AdaBoostClassifier recall score(average = weighted) : 0.40

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 Random Forest gives the highest accuracy.