VIBRATION DETECTION FOR CHOPPERS

# Problem Statement

Choppers are source of vibration due to its Main Rotors and Tails Rotors and to make them airworthy in view of aerodynamics system need to have less vibration. Vibration may cause due to various reason and that data will be collected by Flight Monitoring System.

# Architecture

# 

# Data Description

The dataset consists of data collected from Flight Monitoring System in everyday usage. The main source of data is from Instrument System, Control System and from Avionics System. The datasets positive class consists of less vibration which is again set by pilot. The negative class consists of high vibration. The data consists of a subset of all available data, selected by domain experts.

The attribute names of the data have been anonymized for proprietary reasons.

Apart from training files, we also require a "schema" file from the client, which contains all the relevant information about the training files such as Name of the files, Length of Date value in File Name, Length of Time value in File Name, Number of Columns, Name of the Columns, and their datatype.

Data collection and preprocessing & EDP

In this step, we perform different sets of validation on the given set of training files.

1. **Name Validation**- We validate the name of the files based on the given name in the schema file. We have created a regex pattern as per the name given in the schema file to use for validation. After validating the pattern in the name, we check for the length of date in the file name as well as the length of time in the file name. If all the values are as per requirement, we move such files to "Good\_Data\_Folder" else we move such files to "Bad\_Data\_Folder."

2. **Number of Columns** - We validate the number of columns present in the files, and if it doesn't match with the value given in the schema file, then the file is moved to "Bad\_Data\_Folder."

3. **Name of Columns** - The name of the columns is validated and should be the same as given in the schema file. If not, then the file is moved to "Bad\_Data\_Folder".

4. **The datatype of columns** - The datatype of columns is given in the schema file. This is validated when we insert the files into Database. If the datatype is wrong, then the file is moved to "Bad\_Data\_Folder".

5. **Null values in columns** - If any of the columns in a file have all the values as NULL or missing, we discard such a file and move it to "Bad\_Data\_Folder".

Data Insertion in database

1. **Database Creation and connection** - Create a database with the given name passed. If the database is already created, open the connection to the database.

2. **Table creation in the database** - Table with name - "Good\_Data", is created in the database for inserting the files in the "Good\_Data\_Folder" based on given column names and datatype in the schema file. If the table is already present, then the new table is not created and new files are inserted in the already present table as we want training to be done on new as well as old training files.

3. **Insertion of files in the table** - All the files in the "Good\_Data\_Folder" are inserted in the above-created table. If any file has invalid data type in any of the columns, the file is not loaded in the table and is moved to "Bad\_Data\_Folder".

# Model Training

1. Data Export from Db - The data in a stored database is exported as a CSV file to be used for model training.

2. Data Preprocessing

a. Replace the invalid “na” values with numpy “nan” so we can use imputer on such values.

b. Check for null values in the columns. If present, impute the null with mean values.

3. Model Selection - We are using two algorithms, "SVM" and "KNN". The data is trained on both the algorithms with the best parameters derived from GridSearch. We calculate the AUC scores for both models and select the model with the best score.

# Deployment

We will be deploying the model to the GCP Platform.