Mounting Drive and Configuring environment

In [1]:

```
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Mounted at /content/gdrive
```

In [2]:

```
import os
os.environ['KAGGLE_CONFIG_DIR'] = "/content"
%cd /content
```

```
/content
```

In [2]:

```
from google.colab import files
files.upload()
```

Choose File No file selected

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving kaggle.json to kaggle.json
```

Out[2]:

```
{'kaggle.json': b'{"username":"sankalpchawla","key":"1d2fc96a3f385db0c3807663a6ffb2c5"}'}
```

In [3]:

```
!pip install -q kaggle
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
!ls ~/.kaggle
!chmod 600 /root/.kaggle/kaggle.json
```

```
kaggle.json
```

Unzipping data

In [4]:

```
!unzip -q "/content/gdrive/My Drive/train.csv.zip" -d "/content/"
!unzip -q "/content/gdrive/My Drive/test.csv.zip" -d "/content/"
```

Importing required libraries

In [5]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from sklearn.manifold import TSNE
from scipy import stats
import imblearn
import warnings
```

```
import math
import pickle
import gc
import joblib
from sklearn import metrics
from sklearn.externals import joblib
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import log_loss
from prettytable import PrettyTable
import lightgbm as lgb
from imblearn.over_sampling import SMOTE
from sklearn.model_selection import train_test_split
warnings.filterwarnings('ignore')
from sklearn.preprocessing import MinMaxScaler
```

```
/usr/local/lib/python3.6/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning:
pandas.util.testing is deprecated. Use the functions in the public API at pandas.testing instead.
  import pandas.util.testing as tm
/usr/local/lib/python3.6/dist-packages/sklearn/externals/six.py:31: FutureWarning: The module is d
eprecated in version 0.21 and will be removed in version 0.23 since we've dropped support for
Python 2.7. Please rely on the official version of six (https://pypi.org/project/six/).
  "(https://pypi.org/project/six/).", FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/utils/deprecation.py:144: FutureWarning: The
sklearn.neighbors.base module is  deprecated in version 0.22 and will be removed in version 0.24.
The corresponding classes / functions should instead be imported from sklearn.neighbors. Anything
that cannot be imported from sklearn.neighbors is now part of the private API.
  warnings.warn(message, FutureWarning)
/usr/local/lib/python3.6/dist-packages/sklearn/externals/joblib/__init__.py:15: FutureWarning: skl
earn.externals.joblib is deprecated in 0.21 and will be removed in 0.23. Please import this
functionality directly from joblib, which can be installed with: pip install joblib. If this
warning is raised when loading pickled models, you may need to re-serialize those models with scik
it-learn 0.21+.
  warnings.warn(msg, category=FutureWarning)
```

Reading the data

In [6]:

```
df_train = pd.read_csv('/content/train.csv')
df_test = pd.read_csv('/content/test.csv')
```

Final function 1

In [10]:

```
def final_fun_1(df_te):

  test = df_te.copy()

  #Encoding categorical data
  print('Encoding categorical data...')
  test['experiment'] = test['experiment'].map({'CA': 0, 'DA': 1, 'SS': 3, 'LOFT': 4})
  test["experiment"] = test["experiment"].astype('int8')

  test_id = test['id']
  test = test.drop('id',axis=1)

  print('Normalizing the data...')
  #Normalizing the data
  feat = ['eeg_fp1', 'eeg_f7', 'eeg_f8',
       'eeg_t4', 'eeg_t6', 'eeg_t5', 'eeg_t3', 'eeg_fp2', 'eeg_o1', 'eeg_p3',
       'eeg_pz', 'eeg_f3', 'eeg_fz', 'eeg_f4', 'eeg_c4', 'eeg_p4', 'eeg_poz',
       'eeg_c3', 'eeg_cz', 'eeg_o2', 'ecg', 'r', 'gsr']

  sc = MinMaxScaler()

  for i in feat:
    test[i] = sc.fit_transform(np.array(test[i]).reshape(-1,1))
```

```
  print('Loading the model...')

  model_lgb = joblib.load('model_final.pkl')

  print('Predicting for test data...')
  #Predicting for test data
  predicted_lgb = model_lgb.predict(test, num_iteration= model_lgb.best_iteration)
  submission = pd.DataFrame(np.concatenate((np.arange(len(test))[:, np.newaxis], predicted_lgb),
axis=1), columns=['id', 'A', 'B', 'C', 'D'])
  submission['id'] = test_id.astype(int)
  submission.to_csv("Submission.csv", index=False)
  print('Predictions stored')
  print(submission)

  del test
  gc.collect()

  return predicted_lgb
```

In [11]:

```
predicted = final_fun_1(df_test)
```

```
Encoding categorical data...
Normalizing the data...
Loading the model...
Predicting for test data...
Predictions stored
                id         A         B         C         D
0                0  0.968136  0.001681  0.025381  0.004801
1                1  0.983337  0.000670  0.013927  0.002066
2                2  0.968136  0.001681  0.025381  0.004801
3                3  0.983337  0.000670  0.013927  0.002066
4                4  0.968136  0.001681  0.025381  0.004801
...            ...       ...       ...       ...       ...
17965138  17965138  0.951172  0.001225  0.036212  0.011391
17965139  17965139  0.939180  0.001217  0.050339  0.009264
17965140  17965140  0.951172  0.001225  0.036212  0.011391
17965141  17965141  0.939180  0.001217  0.050339  0.009264
17965142  17965142  0.951172  0.001225  0.036212  0.011391

[17965143 rows x 5 columns]
```

In [19]:

```
!kaggle competitions submit -c reducing-commercial-aviation-fatalities -f Submission.csv -m "Final
submission"
```

```
Warning: Your Kaggle API key is readable by other users on this system! To fix this, you can run '
chmod 600 /content/kaggle.json'
Warning: Looks like you're using an outdated API Version, please consider updating (server 1.5.6 /
client 1.5.4)
100% 1.51G/1.51G [00:15<00:00, 107MB/s]
Successfully submitted to Reducing Commercial Aviation Fatalities
```

Predictions

In [12]:

```
def final_fun_2(X):
  predicted_lgb = model_final.predict(X, num_iteration= model_final.best_iteration)
  mul_loss = round(log_loss(y_test.to_numpy(),predicted_lgb),8)
  print('Multi class Log loss = ',round(log_loss(y_test.to_numpy(),predicted_lgb),8))
  return mul_loss
```

In [13]:

```
model_final = joblib.load('model_final.pkl')
df_train['experiment'] = df_train['experiment'].map({'CA': 0, 'DA': 1, 'SS': 3, 'LOFT': 4})
df_train["experiment"] = df_train["experiment"].astype('int8')
```

```
df_train['event'] = df_train['event'].map({'A': 0, 'B': 1, 'C': 2, 'D': 3})
df_train["event"] = df_train["event"].astype('int8')
y = df_train['event']

df_test['experiment'] = df_test['experiment'].map({'CA': 0, 'DA': 1, 'SS': 3, 'LOFT': 4})
df_test["experiment"] = df_test["experiment"].astype('int8')


feat = ['eeg_fp1', 'eeg_f7', 'eeg_f8',
      'eeg_t4', 'eeg_t6', 'eeg_t5', 'eeg_t3', 'eeg_fp2', 'eeg_o1', 'eeg_p3',
      'eeg_pz', 'eeg_f3', 'eeg_fz', 'eeg_f4', 'eeg_c4', 'eeg_p4', 'eeg_poz',
      'eeg_c3', 'eeg_cz', 'eeg_o2', 'ecg', 'r', 'gsr']

sc = MinMaxScaler()
sc2 = MinMaxScaler()
for i in feat:
  df_train[i] = sc.fit_transform(np.array(df_train[i]).reshape(-1,1))

for i in feat:
  df_test[i] = sc2.fit_transform(np.array(df_test[i]).reshape(-1,1))

X_train, X_test, y_train, y_test = train_test_split(df_train, y,test_size=0.25, shuffle=False)
```

In [14]:

```
mul_loss = final_fun_2(X_test)
```

```
Multi class Log loss =  0.32618856
```

Final score obtained