Human Behavior Analytics
Homework 3

By: Sameer Kumar Behera **UIN:** 526004296

Question1: Exploring the background of the data

- **What was the motivation behind collecting these data?**

➢ The motivation behind collecting the data is to detect early stage speech disorders in infants along with any neuro-developmental disorders or any other abnormal activities that may occur. Since an infant can only communicate through crying, it is essential to identify what are the factors causing which type of cry. Infant vocalization audio samples can be used as a bio-marker to detect such disorders in infants. Collection of these audio samples aids in creating a dynamic framework for characterizing neuro-functional biomarkers associated with specific disorders in the development of infants.

- **What the predicted outcomes (i.e. classes)?**

➢ The vocalizations of the audio clips were categorized into the following three classes:
   (i) neutral/positive mood vocalizations
   (ii) fussing vocalizations
   (iii) crying vocalizations

- **What features and machine learning algorithms are being used to classify between the classes of interest?**

➢ Machine Learning algorithms used:
   ❖ A convolutional network to extract features from the raw time representation and then a subsequent recurrent network with Gated Recurrent Units (GRUs) which performs the final classification.
   ❖ Implementation of Support Vector Machines (SVM) with linear kernels and Sequential Minimal Optimisation (SMO).

Question2: Exploring the literature

"**Baby Cry Sound Detection: A Comparison of Hand-Crafted Features and Deep Learning Approach**",

18th International Conference on Engineering Applications of Neural Networks, Aug 2017, Athens, Greece. 24, pp.2096 - 179, 2017. Link - https://hal.archives-ouvertes.fr/hal-01588679

- **What acoustic measures did previous work use to model children's vocalizations and crying? Was there any specific motivation behind the use of these features?**

➢ Methods applied in the paper are:
  i) Mel-Frequency Cepstrum Coefficients (MFCCs) as baseline and Support Vector Data Description (SVDD) as the classifier.
  ii) Convolutional Neural Networks (CNN) applied on Mel-Spectrogram.
  iii) Hand Crafted Baby Cry (HCBC) features, a new set of features tailored to baby cry sound recognition.

➢ Motivation for using the above features:
  • In infants, the fundamental frequency (F0) reaches values between 250Hz - 600Hz, which has a higher range than that of adult females and males. Given these spectral properties, MFCCs have been proven to be a good candidate for baby cry detection task.
  • Also, Convolutional Neural Networks, i.e. CNNs have proven to be very successful in speech and music recognition.

- **What outcomes were found to be correlated with children's vocalizations?**

➢ The fundamental frequency F0 is estimated using an autocorrelation method. These features are composed of Voiced Unvoiced Counter (VUVC), Consecutive F0 (CF0) and Harmonic Ratio Accumulation (HRA) which create a 3-D feature vector used by Support Vector Data Description (SVDD) classifier to model the target baby-cry class.
The three features (VUVC, CF0 and HRA) capture correlated, monotonal and harmonic patterns. A baby cry sound has a specific pitch, duration and spectral distribution that requires fine tuning for optimal class differentiation.

- **Where there any specific machine learning frameworks that performed best?**

➢ As per the results, Convolutional Neural Networks, i.e. CNNs performed the best at the expense of high computation complexity. It was also able to automatically extract meaningful patterns from Log Mel-Filtered Spectrograms, achieving the best results.
The proposed method, i.e. HCBC has the same level of performance and is less computational and memory demanding.

Question3: Exploring the data

- Using the Praat Toolbox, extract the mean intensity and mean formant frequencies (F1, F2) from each file. Plot the histograms of the mean intensity and mean formant frequencies (F1, F2) for each class.

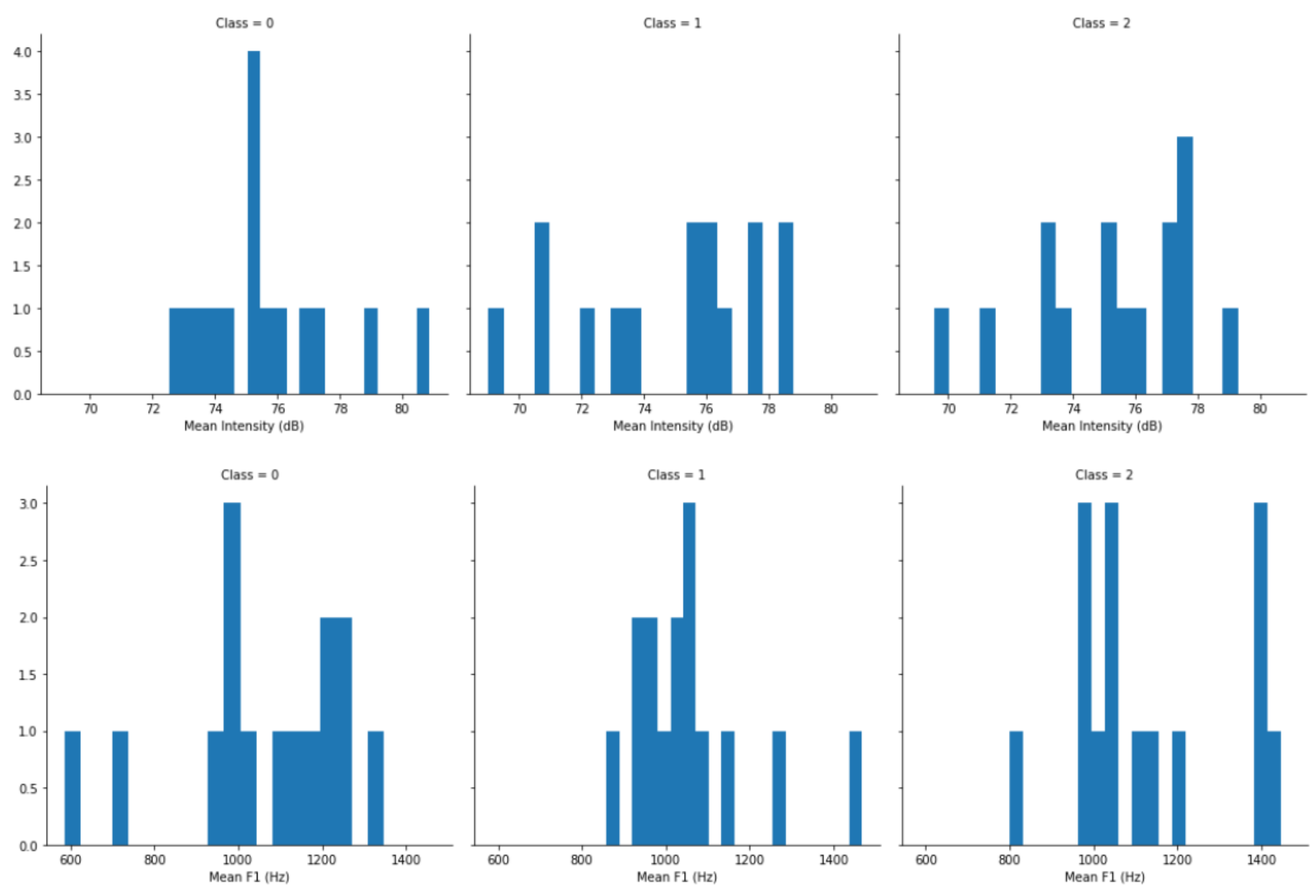
➢ Extracted Mean Intensity and Formant Frequencies for each class are as follows:

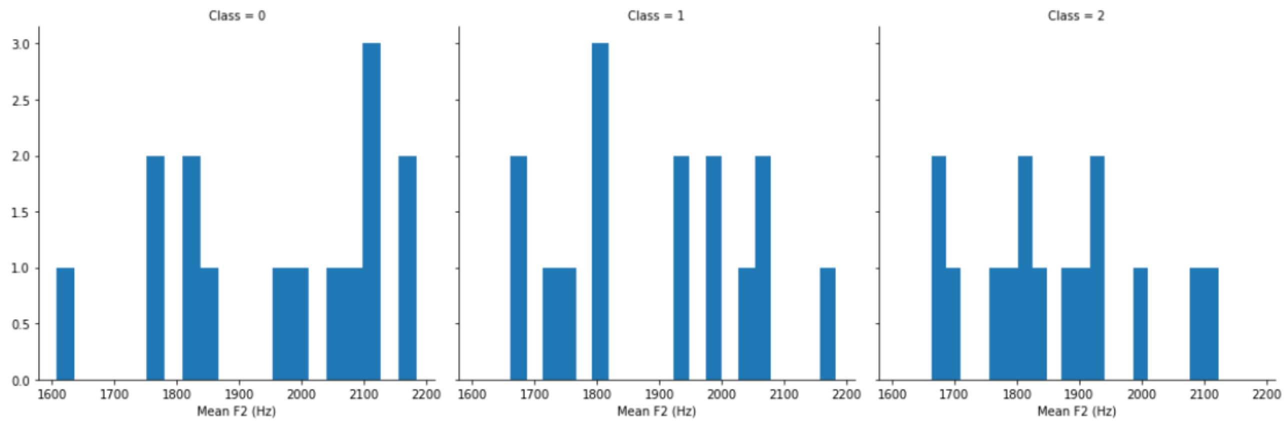
	Input File	Mean Intensity (dB)	Mean F1 (Hz)	Mean F2 (Hz)	Class
0	class0_0001	75.127995	1346.128814	2122.949465	0
1	class0_0002	72.551435	1197.876987	2124.292557	0
2	class0_0003	74.523329	1032.567270	2011.570453	0
3	class0_0004	73.100494	1169.163670	2100.715713	0
4	class0_0005	80.875652	587.327495	1969.573985	0
5	class0_0006	76.228594	1109.207546	2097.243814	0
6	class0_0007	79.175127	971.100241	2046.086783	0
7	class0_0008	73.537049	1122.680708	1826.824272	0
8	class0_0009	75.215367	1003.956666	1860.137496	0
9	class0_0012	77.331906	1261.482471	2179.815191	0
10	class0_0016	75.242341	958.746946	1770.595809	0
11	class0_0017	75.580615	999.165682	1778.273497	0
12	class0_0018	73.851691	1220.563199	2184.974056	0
13	class0_0019	75.160669	1256.406123	1814.304848	0
14	class0_0020	77.022014	737.148119	1607.661538	0

	Input File	Mean Intensity (dB)	Mean F1 (Hz)	Mean F2 (Hz)	Class
0	class1_0010	76.294090	859.129238	1681.352505	1
1	class1_0033	78.793004	965.676981	1717.793129	1
2	class1_0036	76.232112	1044.278403	1799.174521	1
3	class1_0040	73.853653	987.791388	2075.648085	1
4	class1_0051	75.845598	1078.564980	1801.284193	1
5	class1_0054	72.179230	1281.177510	1932.409375	1
6	class1_0064	77.396104	921.694622	1812.336248	1
7	class1_0072	75.418123	1035.643233	1765.968962	1
8	class1_0087	73.319592	1063.592413	1941.517767	1
9	class1_0095	77.656730	1032.923485	1992.220600	1
10	class1_0098	70.915944	1139.131297	2067.208557	1
11	class1_0120	78.320705	938.226453	1662.062376	1
12	class1_0123	69.013036	1070.233604	2183.218012	1
13	class1_0126	70.550783	951.499727	1991.797324	1
14	class1_0137	76.497497	1467.106093	2048.359561	1

	Input File	Mean Intensity (dB)	Mean F1 (Hz)	Mean F2 (Hz)	Class
0	class2_0011	77.762368	992.849930	1783.578967	2
1	class2_0015	79.293635	1214.144875	1841.277744	2
2	class2_0028	77.340683	1407.426485	1919.382967	2
3	class2_0043	77.784575	988.094755	1764.218018	2
4	class2_0053	75.805315	1045.355318	1690.118259	2
5	class2_0065	73.342271	994.551678	1903.226492	2
6	class2_0074	77.065037	1096.560940	1918.425804	2
7	class2_0099	76.219127	1401.141089	1888.044255	2
8	class2_0104	73.455123	1410.993848	2123.757795	2
9	class2_0133	77.447218	1446.876479	2079.052353	2
10	class2_0141	69.562974	1151.898072	2007.528217	2
11	class2_0164	71.378626	800.426544	1664.374589	2
12	class2_0165	75.283311	986.617355	1674.771480	2
13	class2_0167	75.080862	1039.479716	1820.165251	2
14	class2_0183	73.628717	1032.675236	1806.135888	2

- Histograms:





- Use a 2-sample t-test to identify potential significant differences between: 1) neutral/positive mood vocalizations and fussing, 2) neutral/positive mood vocalizations and crying, and 3) fussing and crying.

➢ The Results of the t-tests are as follows:

```
ttest_intensity_class_0_vs_1 = stats.ttest_ind(df0["Mean Intensity (dB)"],df1["Mean Intensity (dB)"])
ttest_intensity_class_1_vs_2 = stats.ttest_ind(df1["Mean Intensity (dB)"],df2["Mean Intensity (dB)"])
ttest_intensity_class_2_vs_0 = stats.ttest_ind(df2["Mean Intensity (dB)"],df0["Mean Intensity (dB)"])
print("Ttest for Mean Intensity Between Class0 and Class1: "+str(ttest_intensity_class_0_vs_1))
print("Ttest for Mean Intensity Between Class1 and Class2: "+str(ttest_intensity_class_1_vs_2))
print("Ttest for Mean Intensity Between Class2 and Class0: "+str(ttest_intensity_class_2_vs_0))
```

```
Ttest for Mean Intensity Between Class0 and Class1: Ttest_indResult(statistic=0.8377423073511991, pvalue=0.4092697076210815)
Ttest for Mean Intensity Between Class1 and Class2: Ttest_indResult(statistic=-0.5223807106171713, pvalue=0.6055141024428679)
Ttest for Mean Intensity Between Class2 and Class0: Ttest_indResult(statistic=-0.3017199242462727, pvalue=0.7650952047736066)
```

```
ttest_f1_class_0_vs_1 = stats.ttest_ind(df0["Mean F1 (Hz)"],df1["Mean F1 (Hz)"])
ttest_f1_class_1_vs_2 = stats.ttest_ind(df1["Mean F1 (Hz)"],df2["Mean F1 (Hz)"])
ttest_f1_class_2_vs_0 = stats.ttest_ind(df2["Mean F1 (Hz)"],df0["Mean F1 (Hz)"])
print("Ttest for Mean F1 Between Class0 and Class1: "+str(ttest_f1_class_0_vs_1))
print("Ttest for Mean F1 Between Class1 and Class2: "+str(ttest_f1_class_1_vs_2))
print("Ttest for Mean F1 Between Class2 and Class0: "+str(ttest_f1_class_2_vs_0))
```

```
Ttest for Mean F1 Between Class0 and Class1: Ttest_indResult(statistic=0.13936267434578162, pvalue=0.8901611531775196)
Ttest for Mean F1 Between Class1 and Class2: Ttest_indResult(statistic=-1.212625808154044, pvalue=0.23540532642409467)
Ttest for Mean F1 Between Class2 and Class0: Ttest_indResult(statistic=0.9425114110646402, pvalue=0.3539925571795449)
```

```
ttest_f2_class_0_vs_1 = stats.ttest_ind(df0["Mean F2 (Hz)"],df1["Mean F2 (Hz)"])
ttest_f2_class_1_vs_2 = stats.ttest_ind(df1["Mean F2 (Hz)"],df2["Mean F2 (Hz)"])
ttest_f2_class_2_vs_0 = stats.ttest_ind(df2["Mean F2 (Hz)"],df0["Mean F2 (Hz)"])
print("Ttest for Mean F2 Between Class0 and Class1: "+str(ttest_f2_class_0_vs_1))
print("Ttest for Mean F2 Between Class1 and Class2: "+str(ttest_f2_class_1_vs_2))
print("Ttest for Mean F2 Between Class2 and Class0: "+str(ttest_f2_class_2_vs_0))
```

```
Ttest for Mean F2 Between Class0 and Class1: Ttest_indResult(statistic=1.1001350362611368, pvalue=0.2806420337447373)
Ttest for Mean F2 Between Class1 and Class2: Ttest_indResult(statistic=0.7137541870579905, pvalue=0.48128688179239654)
Ttest for Mean F2 Between Class2 and Class0: Ttest_indResult(statistic=-1.843353554860907, pvalue=0.0758866795390087)
```

To determine whether the difference between the population means is statistically significant, we need to compare the p-value to the significance level. Usually, a significance level (denoted as α) of 0.05 works well. A significance level of 0.05 indicates a 5% risk of concluding that a difference exists when there is no actual difference. However, as P-value is greater than α, i.e. 0.05, there is not enough evidence to conclude that the difference between the population means is statistically significant.

- Use a 1-way ANOVA to identify potential significant differences between the three classes.

➢ The Results of the ANOVA test are as follows:

```
anova_intensity_class_0_1_2 = stats.f_oneway(df0["Mean Intensity (dB)"],df1["Mean Intensity (dB)"],df2["Mean Intensity (dB)"])
anova_f1_class_0_1_2 = stats.f_oneway(df0["Mean F1 (Hz)"],df1["Mean F1 (Hz)"],df2["Mean F1 (Hz)"])
anova_f2_class_0_1_2 = stats.f_oneway(df0["Mean F2 (Hz)"],df1["Mean F2 (Hz)"],df2["Mean F2 (Hz)"])
print("ANOVA for Mean Intensity Between Class 0, 1 and 2: "+str(anova_intensity_class_0_1_2))
print("ANOVA for Mean F1 Between Class 0, 1 and 2: "+str(anova_f1_class_0_1_2))
print("ANOVA for Mean F2 Between Class 0, 1 and 2: "+str(anova_f2_class_0_1_2))
```

```
ANOVA for Mean Intensity Between Class 0, 1 and 2: F_onewayResult(statistic=0.3640934341007462, pvalue=0.6969977043760991)
ANOVA for Mean F1 Between Class 0, 1 and 2: F_onewayResult(statistic=0.7937741839530502, pvalue=0.45880042324500303)
ANOVA for Mean F2 Between Class 0, 1 and 2: F_onewayResult(statistic=1.7281089217485994, pvalue=0.19000973265533877)
```

The null hypothesis for ANOVA is that the mean (average value of the dependent variable) is the same for all groups. The alternative hypothesis is that the average is not the same for all groups. The ANOVA test procedure produces an F-statistic, which is used to obtain the p-value. Now, if p-value < .05, we reject the null hypothesis. However, as P-value is greater than 0.05, there is not enough evidence to conclude that the difference between the population means is statistically significant.

- **Bonus:** Using a k-means classifier and the three input features (mean intensity, mean F1, mean F2), classify among the three classes.

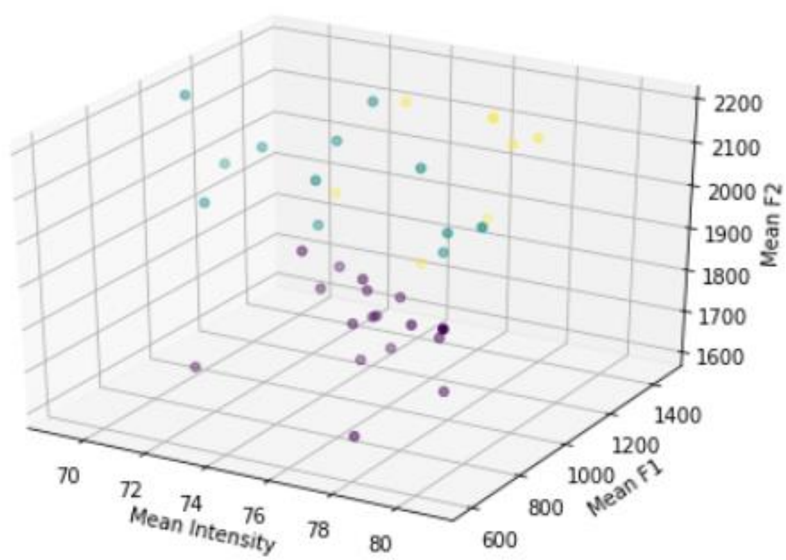
➢ The K-means Classifier was built using Stratified 5-Fold:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from scipy import stats
from sklearn.cluster import KMeans
from sklearn.model_selection import StratifiedKFold
from mpl_toolkits.mplot3d import Axes3D
from sklearn.metrics.cluster import completeness_score
```

```
#Kmeans Classifier Using Stratified 5-Fold
comp_sum = 0
skf = StratifiedKFold(n_splits=5)
skf.get_n_splits(X, y)
for train_index, test_index in skf.split(X, y):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
    kmeans = KMeans(n_clusters=3, n_init=20, max_iter=1000)
    kmeans.fit(X_train,y_train)
    y_predict = kmeans.predict(X_test)
    #Plot 3D Graph of Trained KMeans
    fig = plt.figure()
    ax = Axes3D(fig)
    ax.scatter(X_train[:,0],X_train[:,1],X_train[:,2], c=kmeans.labels_.astype(float))
    ax.set_xlabel('Mean Intensity')
    ax.set_ylabel('Mean F1')
    ax.set_zlabel('Mean F2')
    plt.show()
    #Checking Completeness Score
    print("Completeness Score: ",completeness_score(y_test,y_predict))
    comp_sum += completeness_score(y_test,y_predict)
mean_score = comp_sum/5
print("The Mean Completeness Score of the 5 Training Kmeans Models: ",mean_score)
```

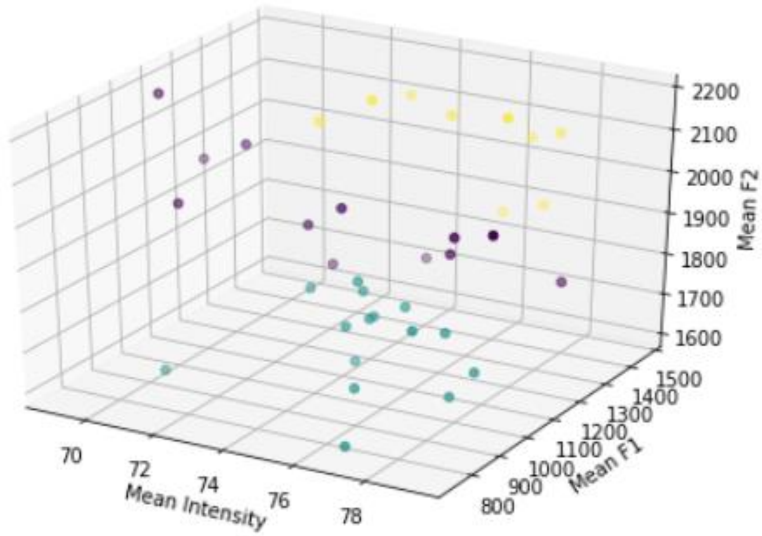
- 3D Visualization of the Training Model for each fold with its Completeness Score:

  - Fold1:



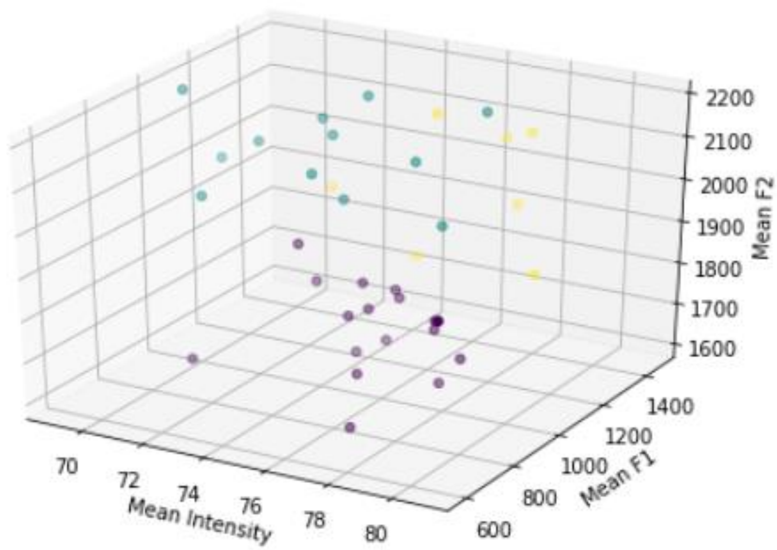

```
Completeness Score:  0.6000000000000001
```

❖ Fold2:



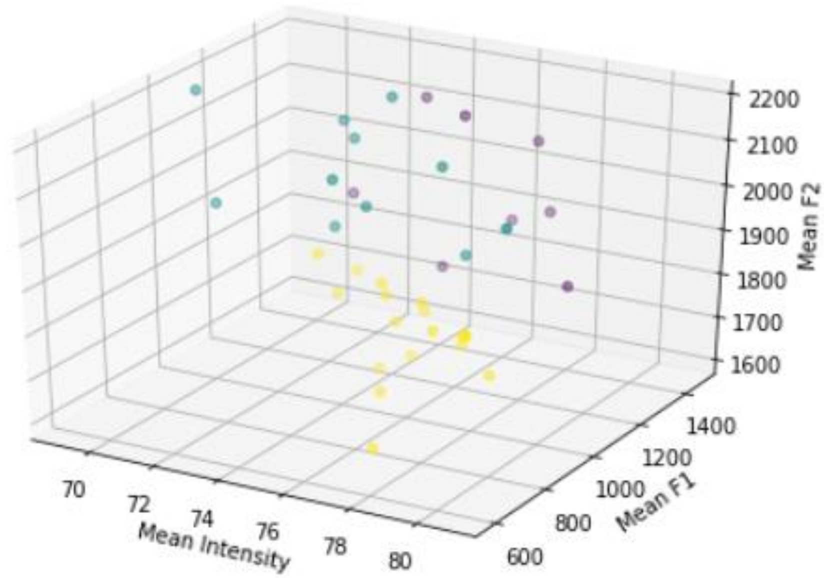

```
Completeness Score:  0.1807489632637144
```

❖ Fold3:



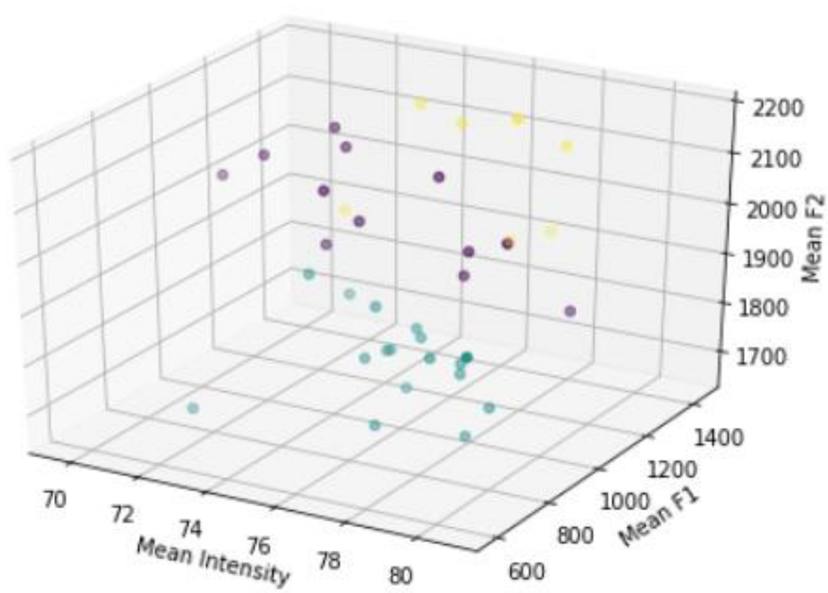

```
Completeness Score:  0.4000000000000001
```

❖ Fold4:




```
Completeness Score:  0.18074896326371442
```

❖ Fold5:




```
Completeness Score:  0.45480350842425626
```

```
The Mean Completeness Score of the 5 Training Kmeans Models:  0.3632602869903371
```