Lab 7 - ML Programming

January 1, 2022

[1]:
```
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
```

1 Exercise 0

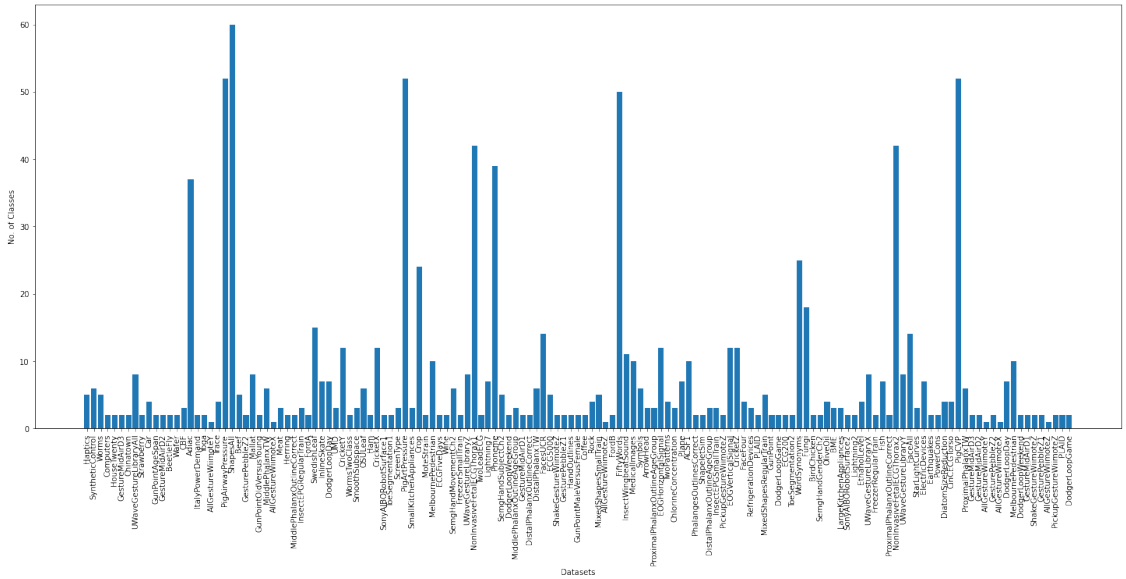
1.1 Dataset Preprocessing

[2]:
```
varying_length = ["AllGestureWiimoteX", "AllGestureWiimoteY",␣
↪"AllGestureWiimoteZ", "GestureMidAirD1", "GestureMidAirD2",␣
↪"GestureMidAirD3","GesturePebbleZ1","GesturePebbleZ2","PickupGestureWiimoteZ",␣
↪"PLAID", "ShakeGestureWiimoteZ"]
```

[3]:
```
dataset_name = []
no_of_classes = []
no_of_samples = []
length_of_samples = []
dataset=dict()
for _, dirs, _ in os.walk("./UCRArchive_2018/"):
    for directory in dirs:
        if directory == "Missing_value_and_variable_length_datasets_adjusted":
            continue
        for _, _, files in os.walk("./UCRArchive_2018/"+directory):
            file_list = []
            df_list = []
            for file in files:
                if file.endswith(".tsv"):
                    file_list.append(file)
            dataset_name.append(directory)
            for f in file_list:
                temp_df = pd.read_csv("./UCRArchive_2018/"+ directory + "/" +␣
↪f, header=None, sep='\t')
                df_list.append(temp_df)
            df = pd.concat(df_list, ignore_index=True)
            if directory in varying_length:
                df_length = df.count(axis = 'columns')
```

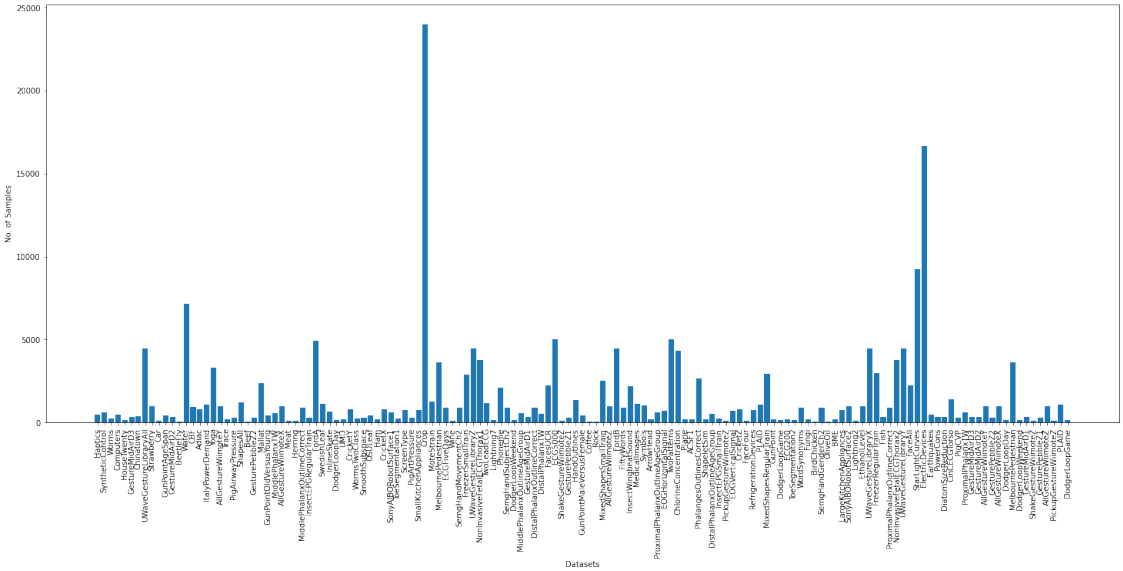
```
                df_length = len(df.columns)-df_length
                df = df.apply(lambda x: pd.Series(np.pad(x.values,(df_length[x.
↪name], 0), 'constant', constant_values=0)), axis=1)
                df = df.apply(lambda x: pd.Series(x.dropna().values), axis=1).
↪fillna('')
            no_of_samples.append(len(df))
            length_of_samples.append(len(df.columns))
            no_of_classes.append(len(df[0].unique()))
            grouped_df = df.groupby(df.columns[0])
            stratified_data = [np.split(g, [int(0.70 * len(g)), int((1 - 0.15)␣
↪* len(g))]) for i, g in grouped_df]
            train = pd.concat([d[0] for d in stratified_data])
            val = pd.concat([d[1] for d in stratified_data])
            test = pd.concat([d[2] for d in stratified_data])
            dataset[directory] = [train,val,test]
```

[4]:
```
fig = plt.figure(figsize = (25, 10))
x_pos = np.arange(len(dataset_name))
plt.bar(x_pos, no_of_classes)
plt.xticks(x_pos, dataset_name, rotation=90)
plt.xlabel("Datasets")
plt.ylabel("No. of Classes")
plt.show()
```

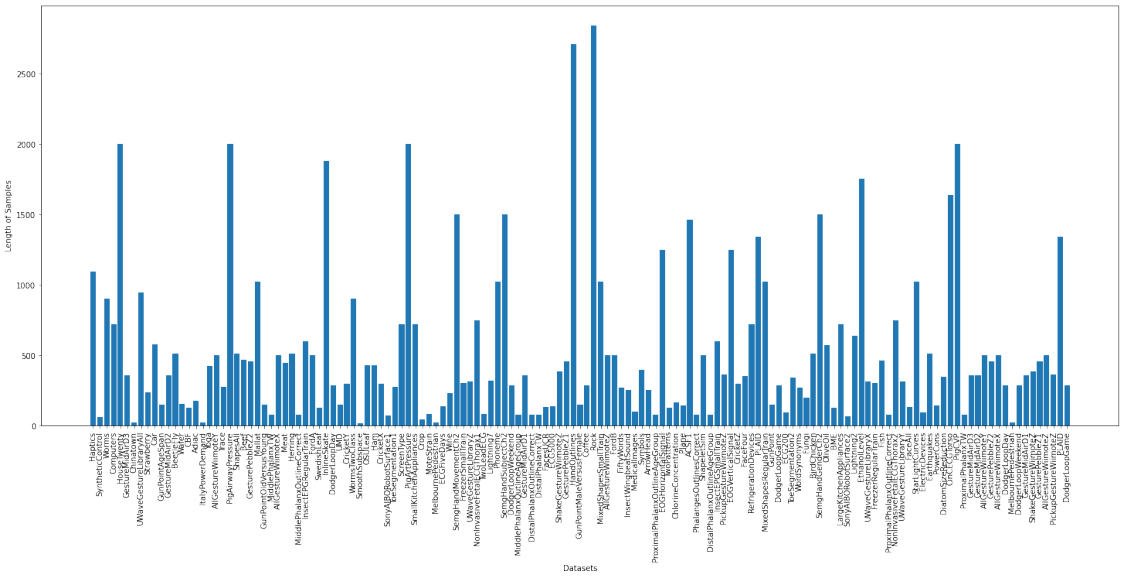


[5]:
```
fig = plt.figure(figsize = (25, 10))
plt.bar(x_pos, no_of_samples)
plt.xticks(x_pos, dataset_name, rotation=90)
```

```
plt.xlabel("Datasets")
plt.ylabel("No. of Samples")
plt.show()
```



[6]:
```
fig = plt.figure(figsize = (25, 10))
plt.bar(x_pos, length_of_samples)
plt.xticks(x_pos, dataset_name, rotation=90)
plt.xlabel("Datasets")
plt.ylabel("Length of Samples")
plt.show()
```



2 Exercise 1

2.1 Dataset Imputation with KNN

[1]:
```
missing_values = ['DodgerLoopDay', 'DodgerLoopGame', 'DodgerLoopWeekend',␣
↪'MelbournePedestrain']
```

[ ]:
```
column_sum = dataset[missing_values[0]][0].isnull().sum()
```

[ ]:
```
columns_with_NAN = column_sum[column_sum != 0].to_dict()
```

[ ]:
```
def find_average(k, column, value):
    distances_with_indices = []
    for i, v in column:
        dist = abs(value - v)
        distances_with_indices.append((dist, i))
    distances_with_indices.sort(key=lambda x:x[0])
    distances_with_indices=distances_with_indices[:k]
    sum_val = 0
    for _, ind in distances_with_indices:
        sum_val += column[ind]
    return sum_val/k
```

[ ]:
```
ridiculous
```