featurization_activity

July 18, 2019

```
In [0]: %matplotlib inline
        import warnings
        warnings.filterwarnings("ignore")



        import pandas as pd
        import seaborn as sns
        import matplotlib.pyplot as plt
        import numpy as np
        from sklearn.model_selection import train_test_split

        import math as m
```

```
In [0]: from google.colab import drive
        drive.mount('/content/gdrive')
```

```
Drive already mounted at /content/gdrive; to attempt to forcibly remount, call drive.mount("/c
```

```
In [0]: dataset_url = '/content/gdrive/My Drive/case_studies/UT_Data_Complex/smartphoneatpocke
        data = pd.read_csv(dataset_url)
        data.head
```

```
Out[0]: <bound method NDFrame.head of              1.39e+12   -3.1327  -10.188    -1.0488  ...
        0        1.390000e+12 -5.924900 -10.9780   1.007900  ...  21.12  42.60  15.78  11111
        1        1.390000e+12 -6.960000 -12.1360   0.286030  ...  20.82  42.60  16.74  11111
        2        1.390000e+12 -3.963500 -15.5680  -3.377800  ...  20.64  42.36  17.34  11111
        3        1.390000e+12 -0.054481 -15.6770  -4.440200  ...  20.40  42.06  18.36  11111
        4        1.390000e+12  0.354130 -13.0480  -2.574200  ...  20.34  42.06  19.02  11111
        5        1.390000e+12 -0.217930  -9.9156  -0.449470  ...  20.16  41.82  19.68  11111
        6        1.390000e+12 -0.653780  -7.7091   0.708260  ...  19.44  41.28  21.30  11111
        7        1.390000e+12 -1.076000  -7.3822  -0.258790  ...  19.26  41.04  22.14  11111
        8        1.390000e+12 -1.389300  -6.2245  -1.062400  ...  18.96  40.80  22.98  11111
        9        1.390000e+12 -1.838700  -2.3563  -1.443800  ...  18.54  39.96  24.78  11111
        10       1.390000e+12 -1.729800  -1.6889  -1.130500  ...  18.42  39.60  25.80  11111
        11       1.390000e+12 -1.280300  -1.7843  -0.068102  ...  18.24  39.24  26.40  11111
        12       1.390000e+12 -1.021500  -2.8603   0.967040  ...  18.12  39.06  27.18  11111
```

13	1.390000e+12	-0.939800	-3.7728	1.089600	...	18.24	38.52	28.26	11111
14	1.390000e+12	-1.089600	-2.9829	0.667400	...	18.12	38.58	28.62	11111
15	1.390000e+12	-1.702500	-2.4108	-0.803600	...	18.12	38.64	29.04	11111
16	1.390000e+12	-1.729800	-8.6081	-8.526300	...	17.94	38.82	29.52	11111
17	1.390000e+12	-1.443800	-13.6070	-12.163000	...	18.00	39.18	29.64	11111
18	1.390000e+12	0.749120	-19.4500	-14.519000	...	18.12	39.60	29.28	11111
19	1.390000e+12	-0.612920	-16.5080	-11.931000	...	18.24	39.78	29.16	11111
20	1.390000e+12	-1.866000	-9.3163	-5.775000	...	18.36	40.02	28.98	11111
21	1.390000e+12	-0.449470	-9.5206	-3.214400	...	18.66	40.80	28.44	11111
22	1.390000e+12	-0.885320	-15.6770	-1.688900	...	18.84	41.16	28.08	11111
23	1.390000e+12	-4.372100	-16.7530	-2.642300	...	19.02	41.40	27.84	11111
24	1.390000e+12	-2.860300	-9.6023	-0.898940	...	18.96	41.88	27.36	11111
25	1.390000e+12	-1.934100	-3.4051	4.467500	...	18.72	42.18	27.18	11111
26	1.390000e+12	-4.644500	-1.6344	0.735500	...	18.60	42.42	27.00	11111
27	1.390000e+12	-1.443800	-5.9112	-1.021500	...	18.66	42.54	26.82	11111
28	1.390000e+12	1.866000	-14.0700	1.838700	...	18.42	42.72	26.70	11111
29	1.390000e+12	1.634400	-11.8090	0.217930	...	18.30	42.60	26.88	11111
...	...	...	...	...	...	...	...	...	...
1169969	1.430000e+12	5.788600	-2.0567	-7.899800	...	-21.00	-1.50	46.44	11123
1169970	1.430000e+12	5.802300	-2.0839	-7.899800	...	-21.00	-1.62	46.44	11123
1169971	1.430000e+12	5.870400	-2.0975	-7.709100	...	-21.24	-1.56	46.56	11123
1169972	1.430000e+12	5.870400	-2.1384	-7.763600	...	-21.24	-1.32	46.62	11123
1169973	1.430000e+12	5.897600	-2.1520	-7.736400	...	-21.24	-1.32	46.56	11123
1169974	1.430000e+12	5.938500	-2.0975	-7.804500	...	-21.36	-1.44	46.50	11123
1169975	1.430000e+12	5.965700	-2.0567	-7.709100	...	-21.48	-1.50	46.62	11123
1169976	1.430000e+12	5.993000	-2.0839	-7.695500	...	-21.42	-1.50	46.68	11123
1169977	1.430000e+12	5.924900	-2.0294	-7.831700	...	-21.42	-1.26	46.80	11123
1169978	1.430000e+12	5.897600	-2.0975	-7.722700	...	-21.42	-1.14	46.80	11123
1169979	1.430000e+12	5.938500	-2.0567	-7.709100	...	-21.54	-1.08	46.86	11123
1169980	1.430000e+12	5.856700	-2.1112	-7.858900	...	-21.48	-1.02	46.80	11123
1169981	1.430000e+12	5.870400	-2.0567	-7.818100	...	-21.24	-1.14	46.86	11123
1169982	1.430000e+12	5.802300	-2.0839	-7.845300	...	-21.42	-1.32	46.80	11123
1169983	1.430000e+12	5.802300	-2.0975	-7.858900	...	-21.30	-1.44	46.86	11123
1169984	1.430000e+12	5.843100	-2.1384	-7.750000	...	-21.30	-1.56	46.86	11123
1169985	1.430000e+12	5.829500	-2.1384	-7.777200	...	-21.48	-1.44	46.80	11123
1169986	1.430000e+12	5.870400	-2.1520	-7.831700	...	-21.36	-1.50	46.74	11123
1169987	1.430000e+12	5.802300	-2.1520	-7.886200	...	-21.42	-1.38	46.80	11123
1169988	1.430000e+12	5.802300	-2.1520	-7.763600	...	-21.36	-1.26	46.92	11123
1169989	1.430000e+12	5.897600	-2.1248	-7.804500	...	-21.48	-1.20	46.86	11123
1169990	1.430000e+12	5.815900	-2.0839	-7.709100	...	-21.60	-1.02	46.80	11123
1169991	1.430000e+12	5.870400	-2.0431	-7.709100	...	-21.54	-1.02	46.86	11123
1169992	1.430000e+12	5.870400	-2.0431	-7.763600	...	-21.48	-0.90	46.68	11123
1169993	1.430000e+12	5.843100	-2.0975	-7.777200	...	-21.48	-0.90	46.74	11123
1169994	1.430000e+12	5.870400	-2.0839	-7.777200	...	-21.48	-0.90	46.80	11123
1169995	1.430000e+12	5.870400	-2.1929	-7.858900	...	-21.54	-1.02	46.98	11123
1169996	1.430000e+12	5.870400	-2.1384	-7.763600	...	-21.60	-1.14	46.68	11123
1169997	1.430000e+12	5.843100	-2.1248	-7.736400	...	-21.66	-1.14	46.62	11123
1169998	1.430000e+12	5.802300	-2.0567	-7.831700	...	-21.72	-1.14	46.50	11123

```
        [1169999 rows x 14 columns]>
```

```
In [0]: my_columns = ["timestamp","acc_X","acc_Y","acc_Z","linearacc_X","linearacc_Y","linearac
        data.columns = my_columns
        print (data)
```

```
            timestamp     acc_X    acc_Y  ...  mag_Y  mag_Z  labels
0        1.390000e+12 -5.924900 -10.9780  ...  42.60  15.78   11111
1        1.390000e+12 -6.960000 -12.1360  ...  42.60  16.74   11111
2        1.390000e+12 -3.963500 -15.5680  ...  42.36  17.34   11111
3        1.390000e+12 -0.054481 -15.6770  ...  42.06  18.36   11111
4        1.390000e+12  0.354130 -13.0480  ...  42.06  19.02   11111
5        1.390000e+12 -0.217930  -9.9156  ...  41.82  19.68   11111
6        1.390000e+12 -0.653780  -7.7091  ...  41.28  21.30   11111
7        1.390000e+12 -1.076000  -7.3822  ...  41.04  22.14   11111
8        1.390000e+12 -1.389300  -6.2245  ...  40.80  22.98   11111
9        1.390000e+12 -1.838700  -2.3563  ...  39.96  24.78   11111
10       1.390000e+12 -1.729800  -1.6889  ...  39.60  25.80   11111
11       1.390000e+12 -1.280300  -1.7843  ...  39.24  26.40   11111
12       1.390000e+12 -1.021500  -2.8603  ...  39.06  27.18   11111
13       1.390000e+12 -0.939800  -3.7728  ...  38.52  28.26   11111
14       1.390000e+12 -1.089600  -2.9829  ...  38.58  28.62   11111
15       1.390000e+12 -1.702500  -2.4108  ...  38.64  29.04   11111
16       1.390000e+12 -1.729800  -8.6081  ...  38.82  29.52   11111
17       1.390000e+12 -1.443800 -13.6070  ...  39.18  29.64   11111
18       1.390000e+12  0.749120 -19.4500  ...  39.60  29.28   11111
19       1.390000e+12 -0.612920 -16.5080  ...  39.78  29.16   11111
20       1.390000e+12 -1.866000  -9.3163  ...  40.02  28.98   11111
21       1.390000e+12 -0.449470  -9.5206  ...  40.80  28.44   11111
22       1.390000e+12 -0.885320 -15.6770  ...  41.16  28.08   11111
23       1.390000e+12 -4.372100 -16.7530  ...  41.40  27.84   11111
24       1.390000e+12 -2.860300  -9.6023  ...  41.88  27.36   11111
25       1.390000e+12 -1.934100  -3.4051  ...  42.18  27.18   11111
26       1.390000e+12 -4.644500  -1.6344  ...  42.42  27.00   11111
27       1.390000e+12 -1.443800  -5.9112  ...  42.54  26.82   11111
28       1.390000e+12  1.866000 -14.0700  ...  42.72  26.70   11111
29       1.390000e+12  1.634400 -11.8090  ...  42.60  26.88   11111
...               ...       ...      ...  ...    ...    ...     ...
1169969  1.430000e+12  5.788600  -2.0567  ...  -1.50  46.44   11123
1169970  1.430000e+12  5.802300  -2.0839  ...  -1.62  46.44   11123
1169971  1.430000e+12  5.870400  -2.0975  ...  -1.56  46.56   11123
1169972  1.430000e+12  5.870400  -2.1384  ...  -1.32  46.62   11123
1169973  1.430000e+12  5.897600  -2.1520  ...  -1.32  46.56   11123
1169974  1.430000e+12  5.938500  -2.0975  ...  -1.44  46.50   11123
1169975  1.430000e+12  5.965700  -2.0567  ...  -1.50  46.62   11123
1169976  1.430000e+12  5.993000  -2.0839  ...  -1.50  46.68   11123
1169977  1.430000e+12  5.924900  -2.0294  ...  -1.26  46.80   11123
```

```
1169978  1.430000e+12  5.897600  -2.0975  ...  -1.14  46.80   11123
1169979  1.430000e+12  5.938500  -2.0567  ...  -1.08  46.86   11123
1169980  1.430000e+12  5.856700  -2.1112  ...  -1.02  46.80   11123
1169981  1.430000e+12  5.870400  -2.0567  ...  -1.14  46.86   11123
1169982  1.430000e+12  5.802300  -2.0839  ...  -1.32  46.80   11123
1169983  1.430000e+12  5.802300  -2.0975  ...  -1.44  46.86   11123
1169984  1.430000e+12  5.843100  -2.1384  ...  -1.56  46.86   11123
1169985  1.430000e+12  5.829500  -2.1384  ...  -1.44  46.80   11123
1169986  1.430000e+12  5.870400  -2.1520  ...  -1.50  46.74   11123
1169987  1.430000e+12  5.802300  -2.1520  ...  -1.38  46.80   11123
1169988  1.430000e+12  5.802300  -2.1520  ...  -1.26  46.92   11123
1169989  1.430000e+12  5.897600  -2.1248  ...  -1.20  46.86   11123
1169990  1.430000e+12  5.815900  -2.0839  ...  -1.02  46.80   11123
1169991  1.430000e+12  5.870400  -2.0431  ...  -1.02  46.86   11123
1169992  1.430000e+12  5.870400  -2.0431  ...  -0.90  46.68   11123
1169993  1.430000e+12  5.843100  -2.0975  ...  -0.90  46.74   11123
1169994  1.430000e+12  5.870400  -2.0839  ...  -0.90  46.80   11123
1169995  1.430000e+12  5.870400  -2.1929  ...  -1.02  46.98   11123
1169996  1.430000e+12  5.870400  -2.1384  ...  -1.14  46.68   11123
1169997  1.430000e+12  5.843100  -2.1248  ...  -1.14  46.62   11123
1169998  1.430000e+12  5.802300  -2.0567  ...  -1.14  46.50   11123

[1169999 rows x 14 columns]
```

```
In [0]: print (data.columns)
```

```
Index(['timestamp', 'acc_X', 'acc_Y', 'acc_Z', 'linearacc_X', 'linearacc_Y',
       'linearacc_Z', 'gyro_X', 'gyro_Y', 'gyro_Z', 'mag_X', 'mag_Y', 'mag_Z',
       'labels'],
      dtype='object')
```

```
In [0]: data["labels"].value_counts()
```

```
Out[0]: 11122    90000
        11123    90000
        11120    90000
        11121    90000
        11114    90000
        11115    90000
        11112    90000
        11113    90000
        11118    90000
        11119    90000
        11116    90000
        11117    90000
        11111    89999
        Name: labels, dtype: int64
```

```
In [0]: data = data.drop(["timestamp"],axis = 1)
        #X = data.iloc[:,1:14]
        #y = data.iloc[1:,13:14]
        #print(X.columns)
        print(data.columns)
        y_activity = data["labels"].map({11111: 'walk',
            11112: 'stand',
            11113: 'jog',
            11114: 'sit',
            11115: 'bike',
            11116: 'upstairs',
            11117: 'downstairs',
            11118: 'type',
            11119: 'write',
            11120: 'coffee',
            11121: 'talk',
            11122: 'smoke',
            11123: 'eat',})
        data['Activity_Name'] = y_activity
        #print(data)

Index(['acc_X', 'acc_Y', 'acc_Z', 'linearacc_X', 'linearacc_Y', 'linearacc_Z',
       'gyro_X', 'gyro_Y', 'gyro_Z', 'mag_X', 'mag_Y', 'mag_Z', 'labels'],
      dtype='object')


In [0]: df = data

In [0]: df.head()

Out[0]:       acc_X   acc_Y    acc_Z  linearacc_X  ...  mag_Y  mag_Z  labels  Activity_Name
        0 -5.924900 -10.978  1.00790      -5.3538  ...  42.60  15.78   11111           walk
        1 -6.960000 -12.136  0.28603      -5.4353  ...  42.60  16.74   11111           walk
        2 -3.963500 -15.568 -3.37780      -2.3250  ...  42.36  17.34   11111           walk
        3 -0.054481 -15.677 -4.44020       1.2882  ...  42.06  18.36   11111           walk
        4  0.354130 -13.048 -2.57420       1.5373  ...  42.06  19.02   11111           walk

        [5 rows x 14 columns]

In [0]: df = df.reset_index()

In [0]: #end indices of run of values in 'Activity_Name' column
        df.loc[df.groupby(['Activity_Name'])['index'].idxmax()]['labels']

Out[0]: 449998     11115
        899998     11120
        629998     11117
        1169998    11123
        269998     11113
```

```
        359998     11114
        1079998    11122
        179998     11112
        989998     11121
        719998     11118
        539998     11116
        89998      11111
        809998     11119
        Name: labels, dtype: int64
```

```
In [0]: print(df[0:89999]['Activity_Name'].value_counts())
        print(df[89999:179999]['Activity_Name'].value_counts())
        print(df[179999:269999]['Activity_Name'].value_counts())
        print(df[269999:359999]['Activity_Name'].value_counts())
        print(df[359999:449999]['Activity_Name'].value_counts())
        print(df[449999:539999]['Activity_Name'].value_counts())
        print(df[539999:629999]['Activity_Name'].value_counts())
        print(df[629999:719999]['Activity_Name'].value_counts())
        print(df[719999:809999]['Activity_Name'].value_counts())
        print(df[809999:899999]['Activity_Name'].value_counts())
        print(df[899999:989999]['Activity_Name'].value_counts())
        print(df[989999:1079999]['Activity_Name'].value_counts())
        print(df[1079999:1700000]['Activity_Name'].value_counts())
```

```
walk    89999
Name: Activity_Name, dtype: int64
stand    90000
Name: Activity_Name, dtype: int64
jog    90000
Name: Activity_Name, dtype: int64
sit    90000
Name: Activity_Name, dtype: int64
bike    90000
Name: Activity_Name, dtype: int64
upstairs    90000
Name: Activity_Name, dtype: int64
downstairs    90000
Name: Activity_Name, dtype: int64
type    90000
Name: Activity_Name, dtype: int64
write    90000
Name: Activity_Name, dtype: int64
coffee    90000
Name: Activity_Name, dtype: int64
talk    90000
Name: Activity_Name, dtype: int64
smoke    90000
Name: Activity_Name, dtype: int64
```

```
eat    90000
Name: Activity_Name, dtype: int64
```

- so we can see from the above table there is no overlap between activities

```
In [0]: df.head()
```

```
Out[0]:    index     acc_X   acc_Y    acc_Z  ...  mag_Y  mag_Z  labels  Activity_Name
        0      0 -5.924900 -10.978  1.00790  ...  42.60  15.78   11111           walk
        1      1 -6.960000 -12.136  0.28603  ...  42.60  16.74   11111           walk
        2      2 -3.963500 -15.568 -3.37780  ...  42.36  17.34   11111           walk
        3      3 -0.054481 -15.677 -4.44020  ...  42.06  18.36   11111           walk
        4      4  0.354130 -13.048 -2.57420  ...  42.06  19.02   11111           walk

        [5 rows x 15 columns]
```

- 10 healthy participants aged 25-35 - Seven activities were performed by all ten participants which are walking, jogging, biking, walking upstairs, walking downstairs, sitting and standing.
- These activities were performed for 3 min by each participant
- Seven out of these ten participants performed eating, typing, writing, drinking coffee and giving a talk.
- These activities were performed for 5–6 min. Smoking was performed by six out of these ten participants, where each of them smoked one cigarette. Only six participants were smokers among the ten participants.
- We used 30 min of data for each activity with an equal amount of data from each participant. This resulted in a dataset of 390 (13 Œ 30) min.

```
11111: 'walk',
11112: 'stand',
11113: 'jog',
11114: 'sit',
11115: 'bike',
11116: 'upstairs',
11117: 'downstairs',
11118: 'type',
11119: 'write',
11120: 'coffee',
11121: 'talk',
11122: 'smoke',
11123: 'eat'
```

0.1 Featurization

- The TYPE_ACCELERATION measures the acceleration force in 'meters per second' that is applied to a device on all three physical axes (x, y, and z), including the force of gravity.
- The TYPE_LINEAR_ACCELERATION measures the acceleration force in 'meters per second' that is applied to a device on all three physical axes (x, y, and z), excluding the force of gravity.

```
In [0]: def mag(a, b, c) :
          return m.sqrt(a**2 + b**2 + c**2)

In [0]: mag(-5.924900, -10.978, 1.00790)

Out[0]: 12.515461893993365
```

0.2 Finding Start Indices of an activity

```
In [0]: df.loc[df.groupby(['Activity_Name'])['index'].idxmax()]['labels']

Out[0]: 449998     11115
        899998     11120
        629998     11117
        1169998    11123
        269998     11113
        359998     11114
        1079998    11122
        179998     11112
        989998     11121
        719998     11118
        539998     11116
        89998      11111
        809998     11119
        Name: labels, dtype: int64
```

0.3 Calculating linear velocity

```
In [0]: def calVectX(sind) :
          t = 0.02
          comp = []
          for k in sind :

            comp.append(0)
            if k == 0 :

              for i in range(k, k+89999-1):
                comp.append(comp[i] + laccx[i]*t)

            else :

              for i in range(k, k+89999):
                comp.append(comp[i] + laccx[i]*t)

          return comp


        def calVectY(sind) :
```

```
    t = 0.02
    comp = []
    for k in sind :

      comp.append(0)
      if k == 0 :

        for i in range(k, k+89999-1):
          comp.append(comp[i] + laccy[i]*t)

      else :

        for i in range(k, k+89999):
          comp.append(comp[i] + laccy[i]*t)

    return comp



  def calVectZ(sind) :

    t = 0.02
    comp = []
    for k in sind :

      comp.append(0)
      if k == 0 :

        for i in range(k, k+89999-1):
          comp.append(comp[i] + laccz[i]*t)

      else :

        for i in range(k, k+89999):
          comp.append(comp[i] + laccz[i]*t)

    return comp
```

In [0]:
```
laccx = df['linearacc_X'].values.tolist()
laccy = df['linearacc_Y'].values.tolist()
laccz = df['linearacc_Z'].values.tolist()
```

In [0]:
```
sind = [0, 89999, 179999, 269999, 359999, 449999, 539999, 629999, 719999, 809999, 8999
```

In [0]:
```
xcom = calVectX(sind)

xcom[89999:179998] = df['linearacc_X'][89999:179998]*0.02
xcom[269999:359998] = df['linearacc_X'][269999:359998]*0.02
```

```
        xcom[629999:719998] = df['linearacc_X'][629999:719998]*0.02
        xcom[719999:809998] = df['linearacc_X'][719999:809998]*0.02
        xcom[809999:899998] = df['linearacc_X'][809999:899998]*0.02
        xcom[899999:989998] = df['linearacc_X'][899999:989998]*0.02
        xcom[1079999:1169998] = df['linearacc_X'][1079999:1169998]*0.02
```

```
In [0]: ycom = calVectY(sind)
```

```
        ycom[89999:179998] = df['linearacc_Y'][89999:179998]*0.02
        ycom[269999:359998] = df['linearacc_Y'][269999:359998]*0.02
        ycom[629999:719998] = df['linearacc_Y'][629999:719998]*0.02
        ycom[719999:809998] = df['linearacc_Y'][719999:809998]*0.02
        ycom[809999:899998] = df['linearacc_Y'][809999:899998]*0.02
        ycom[899999:989998] = df['linearacc_Y'][899999:989998]*0.02
        ycom[1079999:1169998] = df['linearacc_Y'][1079999:1169998]*0.02
```

```
In [0]: zcom = calVectZ(sind)
```

```
        zcom[89999:179998] = df['linearacc_Z'][89999:179998]*0.02
        zcom[269999:359998] = df['linearacc_Z'][269999:359998]*0.02
        zcom[629999:719998] = df['linearacc_Z'][629999:719998]*0.02
        zcom[719999:809998] = df['linearacc_Z'][719999:809998]*0.02
        zcom[809999:899998] = df['linearacc_Z'][809999:899998]*0.02
        zcom[899999:989998] = df['linearacc_Z'][899999:989998]*0.02
        zcom[1079999:1169998] = df['linearacc_Z'][1079999:1169998]*0.02
```

```
In [0]: df['velx'] = xcom
        df['vely'] = ycom
        df['velz'] = zcom
```

```
In [0]: df.head()
```

```
Out[0]:    index     acc_X   acc_Y    acc_Z  ...  Activity_Name      velx      vely      velz
        0      0 -5.924900 -10.978  1.00790  ...           walk  0.000000  0.000000  0.000000
        1      1 -6.960000 -12.136  0.28603  ...           walk -0.107076 -0.024436  0.036420
        2      2 -3.963500 -15.568 -3.37780  ...           walk -0.215782 -0.073720  0.053232
        3      3 -0.054481 -15.677 -4.44020  ...           walk -0.262282 -0.192286  0.000644
        4      4  0.354130 -13.048 -2.57420  ...           walk -0.236518 -0.313222 -0.062654

        [5 rows x 18 columns]
```

0.4 Calculating distance travelled

```
In [0]: def calVectDistX(sind) :
          t = 0.02
          comp = []
          for k in sind :
```

```
    comp.append(0)
    if k == 0 :

      for i in range(k, k+89999-1):
        comp.append(comp[i]*t + 0.5*laccx[i]*(t**2))

    else :

      for i in range(k, k+89999):
        comp.append(comp[i]*t + 0.5*laccx[i]*(t**2))

  return comp

def calVectDistY(sind) :
  t = 0.02
  comp = []
  for k in sind :

    comp.append(0)
    if k == 0 :

      for i in range(k, k+89999-1):
        comp.append(comp[i]*t + 0.5*laccy[i]*(t**2))

    else :

      for i in range(k, k+89999):
        comp.append(comp[i]*t + 0.5*laccy[i]*(t**2))

  return comp


def calVectDistZ(sind) :
  t = 0.02
  comp = []
  for k in sind :

    comp.append(0)
    if k == 0 :

      for i in range(k, k+89999-1):
        comp.append(comp[i]*t + 0.5*laccz[i]*(t**2))

    else :

      for i in range(k, k+89999):
        comp.append(comp[i]*t + 0.5*laccz[i]*(t**2))
```

```
        return comp
```

In [0]:
```
xcom = calVectDistX(sind)

xcom[89999:179998] = 0.5*df['linearacc_X'][89999:179998]*(0.02**2)
xcom[269999:359998] = 0.5*df['linearacc_X'][269999:359998]*(0.02**2)
xcom[629999:719998] = 0.5*df['linearacc_X'][629999:719998]*(0.02**2)
xcom[719999:809998] = 0.5*df['linearacc_X'][719999:809998]*(0.02**2)
xcom[809999:899998] = 0.5*df['linearacc_X'][809999:899998]*(0.02**2)
xcom[899999:989998] = 0.5*df['linearacc_X'][899999:989998]*(0.02**2)
xcom[1079999:1169998] = 0.5*df['linearacc_X'][1079999:1169998]*(0.02**2)
```

In [0]:
```
ycom = calVectDistY(sind)

ycom[89999:179998] = 0.5*df['linearacc_Y'][89999:179998]*(0.02**2)
ycom[269999:359998] = 0.5*df['linearacc_Y'][269999:359998]*(0.02**2)
ycom[629999:719998] = 0.5*df['linearacc_Y'][629999:719998]*(0.02**2)
ycom[719999:809998] = 0.5*df['linearacc_Y'][719999:809998]*(0.02**2)
ycom[809999:899998] = 0.5*df['linearacc_Y'][809999:899998]*(0.02**2)
ycom[899999:989998] = 0.5*df['linearacc_Y'][899999:989998]*(0.02**2)
ycom[1079999:1169998] = 0.5*df['linearacc_Y'][1079999:1169998]*(0.02**2)
```

In [0]:
```
zcom = calVectDistZ(sind)


zcom[89999:179998] = 0.5*df['linearacc_Z'][89999:179998]*(0.02**2)
zcom[269999:359998] = 0.5*df['linearacc_Z'][269999:359998]*(0.02**2)
zcom[629999:719998] = 0.5*df['linearacc_Z'][629999:719998]*(0.02**2)
zcom[719999:809998] = 0.5*df['linearacc_Z'][719999:809998]*(0.02**2)
zcom[809999:899998] = 0.5*df['linearacc_Z'][809999:899998]*(0.02**2)
zcom[899999:989998] = 0.5*df['linearacc_Z'][899999:989998]*(0.02**2)
zcom[1079999:1169998] = 0.5*df['linearacc_Z'][1079999:1169998]*(0.02**2)
```

In [0]:
```
df['distx'] = xcom
df['disty'] = ycom
df['distz'] = zcom
```

In [0]:
```
df.head()
```

Out[0]:

	index	acc_X	acc_Y	acc_Z	...	velz	distx	disty	distz
0	0	-5.924900	-10.978	1.00790	...	0.000000	0.000000	0.000000	0.000000
1	1	-6.960000	-12.136	0.28603	...	0.036420	-0.001071	-0.000244	0.000364
2	2	-3.963500	-15.568	-3.37780	...	0.053232	-0.001108	-0.000498	0.000175
3	3	-0.054481	-15.677	-4.44020	...	0.000644	-0.000487	-0.001196	-0.000522
4	4	0.354130	-13.048	-2.57420	...	-0.062654	0.000248	-0.001233	-0.000643

[5 rows x 21 columns]

0.5 Calculating Magnitude of Velocity, Distance, accelerometer, linearacc, gyro, magnetometer

```
In [0]: df['mag_vel'] = df.apply(lambda row : mag(row['velx'], row['vely'], row['velz']), axis
        df['mag_dist'] = df.apply(lambda row : mag(row['distx'], row['disty'], row['distz']), a
        df['mag_acc'] = df.apply(lambda row : mag(row['acc_Y'], row['acc_X'], row['acc_Z']), ax
        df['mag_lacc'] = df.apply(lambda row : mag(row['linearacc_X'], row['linearacc_Y'], row
        df['mag_gyro'] = df.apply(lambda row : mag(row['gyro_X'], row['gyro_Y'], row['gyro_Z'])
        df['mag_magnet'] = df.apply(lambda row : mag(row['mag_X'], row['mag_Y'], row['mag_Z'])
```

```
In [0]: df.head()
```

```
Out[0]:    index     acc_X   acc_Y    acc_Z  ...    mag_acc  mag_lacc  mag_gyro  mag_magnet
        0      0 -5.924900 -10.978  1.00790  ...  12.515462  5.785500  3.732066   50.098132
        1      1 -6.960000 -12.136  0.28603  ...  13.993066  6.026725  3.036441   50.283795
        2      2 -3.963500 -15.568 -3.37780  ...  16.415891  6.889420  3.109439   50.210107
        3      3 -0.054481 -15.677 -4.44020  ...  16.293762  6.945491  4.217099   50.222437
        4      4  0.354130 -13.048 -2.57420  ...  13.304218  3.935439  2.937097   50.443231

        [5 rows x 27 columns]
```

```
In [0]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1169999 entries, 0 to 1169998
Data columns (total 27 columns):
index            1169999 non-null int64
acc_X            1169999 non-null float64
acc_Y            1169999 non-null float64
acc_Z            1169999 non-null float64
linearacc_X      1169999 non-null float64
linearacc_Y      1169999 non-null float64
linearacc_Z      1169999 non-null float64
gyro_X           1169999 non-null float64
gyro_Y           1169999 non-null float64
gyro_Z           1169999 non-null float64
mag_X            1169999 non-null float64
mag_Y            1169999 non-null float64
mag_Z            1169999 non-null float64
labels           1169999 non-null int64
Activity_Name    1169999 non-null object
velx             1169999 non-null float64
vely             1169999 non-null float64
velz             1169999 non-null float64
distx            1169999 non-null float64
disty            1169999 non-null float64
distz            1169999 non-null float64
mag_vel          1169999 non-null float64
mag_dist         1169999 non-null float64
```

```
mag_acc          1169999 non-null float64
mag_lacc         1169999 non-null float64
mag_gyro         1169999 non-null float64
mag_magnet       1169999 non-null float64
dtypes: float64(24), int64(2), object(1)
memory usage: 241.0+ MB
```

0.6 Calculating Jerk

```
In [0]: accx = df['acc_X'].values.tolist()
        accy = df['acc_Y'].values.tolist()
        accz = df['acc_Z'].values.tolist()
```

```
In [0]: accx.append(0)
        accy.append(0)
        accz.append(0)
```

```
In [0]: sind = [0, 89999, 179999, 269999, 359999, 449999, 539999, 629999, 719999, 809999, 8999
```

```
In [0]: def calJerkX(sind) :
          t = 0.02
          comp = []
          for k in sind :

            if k == 0 :

              for i in range(k, k+89999):
                comp.append((accx[i+1] - accx[i])/t)
              comp[k+89999-1] = (-accx[k+89999-1])/0.02
              len(comp)


            else :

              for i in range(k, k+90000):
                comp.append((accx[i+1] - accx[i])/t)
              comp[k+90000-1] = (-accx[k+90000-1])/0.02



          return comp



        def calJerkY(sind) :
          t = 0.02
          comp = []
          for k in sind :
```

```
        if k == 0 :

          for i in range(k, k+89999):
            comp.append((accx[i+1] - accy[i])/t)
          comp[k+89999-1] = (-accy[k+89999-1])/0.02
          len(comp)


        else :

          for i in range(k, k+90000):
            comp.append((accx[i+1] - accy[i])/t)
          comp[k+90000-1] = (-accy[k+90000-1])/0.02



      return comp



    def calJerkZ(sind) :
      t = 0.02
      comp = []
      for k in sind :

        if k == 0 :

          for i in range(k, k+89999):
            comp.append((accx[i+1] - accz[i])/t)
          comp[k+89999-1] = (-accz[k+89999-1])/0.02
          len(comp)


        else :

          for i in range(k, k+90000):
            comp.append((accx[i+1] - accz[i])/t)
          comp[k+90000-1] = (-accz[k+90000-1])/0.02



      return comp
```

In [0]:
```
df['jerkx'] = calJerkX(sind)
df['jerky'] = calJerkY(sind)
df['jerkz'] = calJerkZ(sind)
```

In [0]:
```
df.head()
```

```
Out[0]:    index     acc_X   acc_Y  ...       jerkx      jerky      jerkz
        0      0 -5.924900 -10.978  ...   -51.75500  200.90000 -398.39500
        1      1 -6.960000 -12.136  ...   149.82500  408.62500 -212.47650
        2      2 -3.963500 -15.568  ...   195.45095  775.67595  166.16595
        3      3 -0.054481 -15.677  ...    20.43055  801.55650  239.71650
        4      4  0.354130 -13.048  ...   -28.60300  641.50350  117.81350

        [5 rows x 30 columns]

In [0]: df[269999:359999].head()

Out[0]:          index   acc_X   acc_Y   acc_Z  ...  mag_magnet  jerkx    jerky    jerkz
        269999  269999  2.4380 -1.5800 -9.5342  ...   56.781370 -5.445  195.455  593.165
        270000  270000  2.3291 -1.4438 -9.6704  ...   56.578377  2.040  190.685  602.015
        270001  270001  2.3699 -1.3893 -9.5206  ...   56.610119 -0.680  187.280  593.845
        270002  270002  2.3563 -1.3757 -9.5615  ...   56.538723 -5.445  181.155  590.445
        270003  270003  2.2474 -1.4165 -9.5206  ...   56.478137 -2.725  180.470  585.675

        [5 rows x 30 columns]

In [0]: df['mag_jerk'] = df.apply(lambda row : mag(row['jerkx'], row['jerky'], row['jerkz']), a

In [0]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1169999 entries, 0 to 1169998
Data columns (total 31 columns):
index            1169999 non-null int64
acc_X            1169999 non-null float64
acc_Y            1169999 non-null float64
acc_Z            1169999 non-null float64
linearacc_X      1169999 non-null float64
linearacc_Y      1169999 non-null float64
linearacc_Z      1169999 non-null float64
gyro_X           1169999 non-null float64
gyro_Y           1169999 non-null float64
gyro_Z           1169999 non-null float64
mag_X            1169999 non-null float64
mag_Y            1169999 non-null float64
mag_Z            1169999 non-null float64
labels           1169999 non-null int64
Activity_Name    1169999 non-null object
velx             1169999 non-null float64
vely             1169999 non-null float64
velz             1169999 non-null float64
distx            1169999 non-null float64
disty            1169999 non-null float64
distz            1169999 non-null float64
mag_vel          1169999 non-null float64
```

```
mag_dist          1169999 non-null float64
mag_acc           1169999 non-null float64
mag_lacc          1169999 non-null float64
mag_gyro          1169999 non-null float64
mag_magnet        1169999 non-null float64
jerkx             1169999 non-null float64
jerky             1169999 non-null float64
jerkz             1169999 non-null float64
mag_jerk          1169999 non-null float64
dtypes: float64(28), int64(2), object(1)
memory usage: 276.7+ MB
```

0.7 Calculating Gyro Jerk

```
In [0]: accx = df['gyro_X'].values.tolist()
        accy = df['gyro_Y'].values.tolist()
        accz = df['gyro_Z'].values.tolist()
```

```
In [0]: accx.append(0)
        accy.append(0)
        accz.append(0)
```

```
In [0]: def calGJerkX(sind) :
          t = 0.02
          comp = []
          for k in sind :

            if k == 0 :

              for i in range(k, k+89999):
                comp.append((accx[i+1] - accx[i])/t)
              comp[k+89999-1] = (-accx[k+89999-1])/0.02
              len(comp)


            else :

              for i in range(k, k+90000):
                comp.append((accx[i+1] - accx[i])/t)
              comp[k+90000-1] = (-accx[k+90000-1])/0.02



          return comp
```

```
def calGJerkY(sind) :
  t = 0.02
  comp = []
  for k in sind :

    if k == 0 :

      for i in range(k, k+89999):
        comp.append((accx[i+1] - accy[i])/t)
      comp[k+89999-1] = (-accy[k+89999-1])/0.02
      len(comp)


    else :

      for i in range(k, k+90000):
        comp.append((accx[i+1] - accy[i])/t)
      comp[k+90000-1] = (-accy[k+90000-1])/0.02



  return comp



def calGJerkZ(sind) :
  t = 0.02
  comp = []
  for k in sind :

    if k == 0 :

      for i in range(k, k+89999):
        comp.append((accx[i+1] - accz[i])/t)
      comp[k+89999-1] = (-accz[k+89999-1])/0.02
      len(comp)


    else :

      for i in range(k, k+90000):
        comp.append((accx[i+1] - accz[i])/t)
      comp[k+90000-1] = (-accz[k+90000-1])/0.02



  return comp
```

```
In [0]: df['gjerkx'] = calGJerkX(sind)
        df['gjerky'] = calGJerkY(sind)
        df['gjerkz'] = calGJerkZ(sind)

In [0]: df['mag_gjerk'] = df.apply(lambda row : mag(row['gjerkx'], row['gjerky'], row['gjerkz'])

In [0]: df.head()

Out[0]:    index     acc_X   acc_Y    acc_Z  ...  gjerkx    gjerky    gjerkz   mag_gjerk
        0      0 -5.924900 -10.978  1.00790  ...  33.855 -115.4855 -119.7770  169.792790
        1      1 -6.960000 -12.136  0.28603  ...  49.070  -81.6415  -68.4160  117.277229
        2      2 -3.963500 -15.568 -3.37780  ... -38.225  -23.1500  -98.4860  108.150660
        3      3 -0.054481 -15.677 -4.44020  ...  -0.155   16.9050  -92.4715   94.004156
        4      4  0.354130 -13.048 -2.57420  ...  36.500  -77.1685  -53.4515  100.718867

        [5 rows x 35 columns]

In [0]: new_df = df.drop(['index'], axis=1)

In [0]: new_df.head()

Out[0]:       acc_X   acc_Y    acc_Z  ...    gjerky    gjerkz   mag_gjerk
        0 -5.924900 -10.978  1.00790  ... -115.4855 -119.7770  169.792790
        1 -6.960000 -12.136  0.28603  ...  -81.6415  -68.4160  117.277229
        2 -3.963500 -15.568 -3.37780  ...  -23.1500  -98.4860  108.150660
        3 -0.054481 -15.677 -4.44020  ...   16.9050  -92.4715   94.004156
        4  0.354130 -13.048 -2.57420  ...  -77.1685  -53.4515  100.718867

        [5 rows x 34 columns]

In [0]: new_df.to_csv('/content/gdrive/My Drive/case_studies/UT_Data_Complex/new_df.csv')
```