Task 4

Kosovan Ivan

In [13]:
```
import pandas as pd
import numpy as np
from itertools import repeat
```

In [3]:
```
features = pd.read_csv("features.txt", sep=' ', names=['id', 'cols'])
subject_train = np.loadtxt("subject_train.txt")
subject_test = np.loadtxt("subject_test.txt")
x_train = np.loadtxt("X_train.txt")
y_train = np.loadtxt("Y_train.txt")
x_test = np.loadtxt("X_test.txt")
y_test = np.loadtxt("Y_test.txt")
```

1. Merges the training and the test sets to create one data set.

In [4]:
```
x_train = np.append(x_train, subject_train.reshape(-1,1), axis=1)
x_train = np.append(x_train, y_train.reshape(-1,1), axis=1)

x_test = np.append(x_test, subject_test.reshape(-1,1), axis=1)
x_test = np.append(x_test, y_test.reshape(-1,1), axis=1)

X = pd.DataFrame(np.append(x_train, x_test, axis=0), columns=features.cols.values.
X
```

Out[4]:

	tBodyAcc-mean()-X	tBodyAcc-mean()-Y	tBodyAcc-mean()-Z	tBodyAcc-std()-X	tBodyAcc-std()-Y	tBodyAcc-std()-Z	tBodyAcc-mad()-X	tBodyAcc-mad()-
0	0.288585	-0.020294	-0.132905	-0.995279	-0.983111	-0.913526	-0.995112	-0.98318
1	0.278419	-0.016411	-0.123520	-0.998245	-0.975300	-0.960322	-0.998807	-0.97491
2	0.279653	-0.019467	-0.113462	-0.995380	-0.967187	-0.978944	-0.996520	-0.96366
3	0.279174	-0.026201	-0.123283	-0.996091	-0.983403	-0.990675	-0.997099	-0.98275
4	0.276629	-0.016570	-0.115362	-0.998139	-0.980817	-0.990482	-0.998321	-0.97967
...	...	...	...	...	...	...	...	
10294	0.310155	-0.053391	-0.099109	-0.287866	-0.140589	-0.215088	-0.356083	-0.14877
10295	0.363385	-0.039214	-0.105915	-0.305388	0.028148	-0.196373	-0.373540	-0.03003
10296	0.349966	0.030077	-0.115788	-0.329638	-0.042143	-0.250181	-0.388017	-0.13325
10297	0.237594	0.018467	-0.096499	-0.323114	-0.229775	-0.207574	-0.392380	-0.27961
10298	0.153627	-0.018437	-0.137018	-0.330046	-0.195253	-0.164339	-0.430974	-0.21829

10299 rows × 563 columns

2. Extracts only the measurements on the mean and standard deviation for each measurement.

In [8]:
```
X_mean_std = X[ features[features.cols.str.contains('mean') | features.cols.str.co
X_mean_std.head()
```

Out[8]:

	tBodyAcc-mean()-X	tBodyAcc-mean()-Y	tBodyAcc-mean()-Z	tBodyAcc-std()-X	tBodyAcc-std()-Y	tBodyAcc-std()-Z	tGravityAcc-mean()-X	tGravityAcc mean()-Y
0	0.288585	-0.020294	-0.132905	-0.995279	-0.983111	-0.913526	0.963396	-0.14084
1	0.278419	-0.016411	-0.123520	-0.998245	-0.975300	-0.960322	0.966561	-0.14155
2	0.279653	-0.019467	-0.113462	-0.995380	-0.967187	-0.978944	0.966878	-0.14201
3	0.279174	-0.026201	-0.123283	-0.996091	-0.983403	-0.990675	0.967615	-0.14397
4	0.276629	-0.016570	-0.115362	-0.998139	-0.980817	-0.990482	0.968224	-0.14875

5 rows × 79 columns

3. Uses descriptive activity names to name the activities in the data set

In [10]:
```
activities = pd.read_csv("activity_labels.txt",sep=' ', names=['activity', 'activi
activities
```

Out[10]:

	activity	activity_type
0	1	WALKING
1	2	WALKING_UPSTAIRS
2	3	WALKING_DOWNSTAIRS
3	4	SITTING
4	5	STANDING
5	6	LAYING

In [12]:
```
X = X.merge(activities, how='inner', on='activity')
X
```

ityMean)	angle(tBodyGyroMean,gravityMean)	angle(tBodyGyroJerkMean,gravityMean)	angle(X,gravityMean)	an
0.030400	-0.464761	-0.018446	-0.841247	
-0.007435	-0.732626	0.703511	-0.844788	
0.177899	0.100699	0.808529	-0.848933	
-0.012892	0.640011	-0.485366	-0.848649	
0.122542	0.693578	-0.615971	-0.847865	
...	...	...	...	
0.346295	0.884904	-0.698885	-0.651732	
-0.372889	-0.657421	0.322549	-0.655181	
0.088574	0.696663	0.363139	-0.655357	
-0.819188	0.929294	-0.008398	-0.659719	
-0.287951	0.876030	-0.024965	-0.660080	

5. From the data set in step 4, creates a second, independent tidy data set with the average of each variable for each activity and each subject.

In [14]:
```
X.groupby(['activity','subject']).agg(dict(zip(features[features.cols.str.contains
```

Out[14]:

		tBodyAcc-mean()-X	tBodyAcc-mean()-Y	tBodyAcc-mean()-Z	tGravityAcc-mean()-X	tGravityAcc-mean()-Y	tGravityAcc-mean()-Z	tBod
activity	**subject**							
1.0	**1.0**	0.277331	-0.017384	-0.111148	0.935223	-0.282165	-0.068103	
	2.0	0.276427	-0.018595	-0.105500	0.913017	-0.346607	0.084727	
	3.0	0.275567	-0.017177	-0.112675	0.936507	-0.261986	-0.138108	
	4.0	0.278582	-0.014840	-0.111403	0.964000	-0.085854	0.127764	
	5.0	0.277842	-0.017285	-0.107742	0.972625	-0.100440	0.002476	
...	**...**	...	...	...	...	...	...	
6.0	**26.0**	0.271646	-0.019190	-0.105003	-0.621310	0.803015	0.614523	
	27.0	0.274102	-0.017987	-0.107700	-0.530435	0.567879	0.845392	
	28.0	0.275913	-0.016754	-0.108345	-0.490335	0.144073	0.015757	
	29.0	0.287295	-0.017197	-0.109462	-0.346790	0.807535	0.590452	
	30.0	0.281034	-0.019449	-0.103658	-0.344738	0.732661	0.681459	

180 rows × 46 columns