In [85]:
```
import pandas as pd
```

Qn1(a)

In [86]:
```
bodyfat2 = pd.read_csv("bodyfat2.csv") # reading in the bodyfat2.csv file
```

In [87]:
```
bodyfat2.head() # printing out the top 5 datasets to see
```

Out[87]:

	density	bodyfat	age	weight	height	neck	chest	abdomen	hip	thigh	knee	ankle	bic
0	1.0708	12.3	23	154.25	67.75	36.2	93.1	85.2	94.5	59.0	37.3	21.9	3
1	1.0853	6.1	22	173.25	72.25	38.5	93.6	83.0	98.7	58.7	37.3	23.4	3
2	1.0414	25.3	22	154.00	66.25	34.0	95.8	87.9	99.2	59.6	38.9	24.0	2
3	1.0751	10.4	26	184.75	72.25	37.4	101.8	86.4	101.2	60.1	37.3	22.8	3
4	1.0340	28.7	24	184.25	71.25	34.4	97.3	100.0	101.9	63.2	42.2	24.0	3

In [88]:
```
columns = list(bodyfat2.columns) # getting the column names for slicing purpos
es later, see next cell
```

In [89]:
```
required_features = columns[5:] # from neck all the way to wrist
```

In [90]:
```
# getting the mean, median, and sum from the required features
meanFromRequiredFeatures = bodyfat2[required_features].mean(axis=1)
medianFromRequiredFeatures = bodyfat2[required_features].median(axis=1)
sumFromRequiredFeatures = bodyfat2[required_features].sum(axis=1)
```

In [91]:
```
# getting the top 3 and bottom 3 datasets
topThreeMeans = meanFromRequiredFeatures.head(3)
bottomThreeMeans = meanFromRequiredFeatures.tail(3)
topThreeMedians = medianFromRequiredFeatures.head(3)
bottomThreeMedians = medianFromRequiredFeatures.tail(3)
topThreeSums = sumFromRequiredFeatures.head(3)
bottomThreeSums = sumFromRequiredFeatures.tail(3)

meanColumn = topThreeMeans.append(bottomThreeMeans)
medianColumn = topThreeMedians.append(bottomThreeMedians)
sumColumn = topThreeSums.append(bottomThreeSums)

# creating a dictionary using the above values so that a dataframe can be form
ed at the next cell
stats = {"Mean": meanColumn, "Median": medianColumn, "Sum": sumColumn}
```

In [134]:
```
# creating a dataframe for the top 3 and bottom 3 datasets
# the row names represent the actual indexes of the datasets in bodyfat2.csv
description_df = pd.DataFrame(stats)
description_df
print("Qn 1(a):")
"\n"
description_df
```

```
Qn 1(a):
```

Out[134]:

	Mean	Median	Sum
0	50.37	36.75	503.7
1	51.08	37.90	510.8
2	51.00	36.45	510.0
47	48.22	36.05	482.2
48	47.79	34.30	477.9
49	45.13	34.20	451.3

Qn 1(b)

In [136]:
```
# getting the mean, median, and sum for each feature
stats_df = pd.DataFrame({'Mean': bodyfat2.iloc[:,:].mean(), 'Median': bodyfat2
.iloc[:,:].median(), 'Sum': bodyfat2.iloc[:,:].sum()})
print("Qn 1b: ")
"\n"
stats_df
```

```
Qn 1b:
```

Out[136]:

	Mean	Median	Sum
density	1.05838	1.0616	52.919
bodyfat	17.88200	16.2500	894.100
age	33.66000	32.0000	1683.000
weight	183.25800	182.8750	9162.900
height	70.13000	70.0000	3506.500
neck	38.01600	38.1000	1900.800
chest	101.12800	101.1000	5056.400
abdomen	92.08000	89.1500	4604.000
hip	102.05800	100.4500	5102.900
thigh	61.50600	61.6000	3075.300
knee	38.82800	38.7000	1941.400
ankle	23.58600	23.1000	1179.300
biceps	32.79800	32.4500	1639.900
forearm	28.85200	29.0000	1442.600
wrist	18.11800	18.2000	905.900

Qn 2

In [94]:
```
# Getting the names of the required columns
columns = list(bodyfat2.columns)
columns[:2] + columns[5:]
```

Out[94]:
```
['density',
 'bodyfat',
 'neck',
 'chest',
 'abdomen',
 'hip',
 'thigh',
 'knee',
 'ankle',
 'biceps',
 'forearm',
 'wrist']
```

In [95]:
```
# maxValues of various features, from "density" ... "wrist"
maxValues = list(bodyfat2[columns[:2] + columns[5:]].max())
maxValues
```

Out[95]:
```
[1.0911, 40.1, 51.2, 136.2, 148.1, 147.7, 87.3, 49.1, 33.9, 45.0, 32.8, 21.4]
```

In [96]:
```
# Individual IDs for maxValues
maxValuesIDs = list(bodyfat2[columns[:2] + columns[5:]].idxmax())
maxValuesIDs
```

Out[96]:
```
[25, 35, 38, 38, 38, 38, 38, 38, 30, 38, 21, 38]
```

In [97]:
```
# minValues of various features, from "density" ... "wrist"
minValues = list(bodyfat2[columns[:2] + columns[5:]].min())
minValues
```

Out[97]:
```
[1.0101, 3.7, 31.5, 83.4, 70.4, 85.3, 50.0, 34.4, 20.6, 26.1, 23.1, 16.1]
```

In [98]:
```
# Individual IDs for minValues
minValuesIDs = list(bodyfat2[columns[:2] + columns[5:]].idxmin())
minValuesIDs
```

Out[98]:
```
[35, 25, 44, 49, 49, 26, 44, 49, 48, 44, 44, 44]
```

In [137]:
```
# getting the required dataframe, maxAndMinFat_df
list_of_tuples = list(zip(maxValues, maxValuesIDs, minValues, minValuesIDs))
maxAndMinFat_df = pd.DataFrame(list_of_tuples, columns = ["Max Value", "Indivi
dual ID", "Min Value", "Individual ID"])
maxAndMinFat_df.index = [columns[:2] + columns[5:]]
print("Qn2: ")
"\n"
maxAndMinFat_df
```

```
Qn2:
```

Out[137]:

	Max Value	Individual ID	Min Value	Individual ID
density	1.0911	25	1.0101	35
bodyfat	40.1000	35	3.7000	25
neck	51.2000	38	31.5000	44
chest	136.2000	38	83.4000	49
abdomen	148.1000	38	70.4000	49
hip	147.7000	38	85.3000	26
thigh	87.3000	38	50.0000	44
knee	49.1000	38	34.4000	49
ankle	33.9000	30	20.6000	48
biceps	45.0000	38	26.1000	44
forearm	32.8000	21	23.1000	44
wrist	21.4000	38	16.1000	44

Qn 3

In [100]:
```
# To merely observe the respective means and medians of the features
description = bodyfat2.describe()
description
```

Out[100]:

	density	bodyfat	age	weight	height	neck	chest	abdomen
count	50.000000	50.000000	50.000000	50.000000	50.00000	50.000000	50.000000	50.000000
mean	1.058380	17.882000	33.660000	183.258000	70.13000	38.016000	101.128000	92.080000
std	0.022115	9.842032	8.402162	40.257781	2.66508	3.286917	10.614315	14.508899
min	1.010100	3.700000	22.000000	125.250000	64.75000	31.500000	83.400000	70.400000
25%	1.044550	10.500000	27.000000	155.500000	68.00000	36.200000	93.500000	82.625000
50%	1.061600	16.250000	32.000000	182.875000	70.00000	38.100000	101.100000	89.150000
75%	1.074875	23.875000	40.750000	202.812500	72.06250	39.400000	106.150000	99.700000
max	1.091100	40.100000	50.000000	363.150000	76.00000	51.200000	136.200000	148.100000

In [101]:
```
# extract the mean row from description
means = description.loc["mean"]
means
```

Out[101]:
```
density      1.05838
bodyfat     17.88200
age         33.66000
weight     183.25800
height      70.13000
neck        38.01600
chest      101.12800
abdomen     92.08000
hip        102.05800
thigh       61.50600
knee        38.82800
ankle       23.58600
biceps      32.79800
forearm     28.85200
wrist       18.11800
Name: mean, dtype: float64
```

In [102]:
```
medians = description.loc["50%"]
medians
```

Out[102]:
```
density      1.0616
bodyfat     16.2500
age         32.0000
weight     182.8750
height      70.0000
neck        38.1000
chest      101.1000
abdomen     89.1500
hip        100.4500
thigh       61.6000
knee        38.7000
ankle       23.1000
biceps      32.4500
forearm     29.0000
wrist       18.2000
Name: 50%, dtype: float64
```

In [138]:
```
# find the number of entries in each feature that fall within 10% of standard
 deviation from its respective mean and median

columns = bodyfat2.columns #getting the column names

# j and k are used to create a 2D dataframe at the end
j= []
for column in columns:
    k = []
    k.append(bodyfat2[column][(bodyfat2[column]<=(means[column] + 0.1*descript
ion.loc['std',column])) &
                              (bodyfat2[column]>=(means[column] - 0.1*description
.loc['std',column]))].count())

    k.append(bodyfat2[column][(bodyfat2[column]<=(medians[column] + 0.1*descri
ption.loc['std',column])) &
                              (bodyfat2[column]>=(medians[column] - 0.1*descripti
on.loc['std',column]))].count())
    j.append(k)

noOfEntriesWithin10percentSD_df = pd.DataFrame(j, columns=['Mean','Medians'],i
ndex=[means.index])
print("Qn3: ")
"\n"
noOfEntriesWithin10percentSD_df
```

```
Qn3:
```

Out[138]:

	Mean	Medians
density	1	3
bodyfat	1	3
age	3	4
weight	7	8
height	3	5
neck	4	6
chest	7	7
abdomen	4	6
hip	4	6
thigh	2	2
knee	6	5
ankle	5	7
biceps	5	6
forearm	4	3
wrist	5	7

Qn 4

In [104]:
```
bodyfat3 = pd.read_csv("bodyfat3.csv") # reading in bodyfat3.csv
```

In [105]:
```
bodyfat3.head() # printing out the first 5 datasets to see
```

Out[105]:

	density	bodyfat	age	weight	height	neck	chest	abdomen	hip	thigh	knee	ankle	bic
0	1.0708	12.3	23	154.25	67.75	36.2	93.1	85.2	94.5	59.0	37.3	21.9	3
1	1.0853	6.1	22	173.25	72.25	38.5	93.6	83.0	98.7	58.7	37.3	23.4	N
2	1.0414	NaN	22	154.00	66.25	34.0	95.8	87.9	99.2	59.6	38.9	24.0	N
3	1.0751	10.4	26	NaN	72.25	37.4	101.8	86.4	101.2	60.1	37.3	22.8	N
4	1.0340	28.7	24	184.25	71.25	34.4	97.3	100.0	101.9	NaN	42.2	24.0	N

In [139]:
```
# find the number of missing values in each feature
numOfMissingValues = bodyfat3.isnull().sum()

# Converting it into a dataframe
noOfMissingValues_df = pd.DataFrame(numOfMissingValues).transpose().rename(ind
ex={0:"missing values"})
print("Qn 4: ")
"\n"
noOfMissingValues_df
```

```
Qn 4:
```

Out[139]:

	density	bodyfat	age	weight	height	neck	chest	abdomen	hip	thigh	knee	ankle
missing values	0	4	0	7	2	3	1	0	6	3	1	2

Qn 5(a), part 1 - replacing missing values with mean

In [107]:
```
# reading in bodyfat3.csv
bodyfat3b = pd.read_csv("bodyfat3b.csv")
```

In [108]:
```
# replacing the missing values with the mean of that particular feature
columns = list(bodyfat3b.columns)
for column in columns:
    bodyfat3b[column].fillna(value=bodyfat3b[column].mean(), inplace = True)
```

In [140]:
```
# printing the first 5 datasets to check if it works
print("Qn 5(a), part 1: ")
"\n"
bodyfat3b.head()
```

```
Qn 5(a), part 1:
```

Out[140]:

	density	bodyfat	age	weight	height	neck	chest	abdomen	hip	thigh	knee
0	1.0708	12.300000	23	154.250000	67.75	36.2	93.1	85.2	94.5	59.000000	37.3
1	1.0853	6.100000	22	173.250000	72.25	38.5	93.6	83.0	98.7	58.700000	37.3
2	1.0414	17.345652	22	154.000000	66.25	34.0	95.8	87.9	99.2	59.600000	38.9
3	1.0751	10.400000	26	184.259302	72.25	37.4	101.8	86.4	101.2	60.100000	37.3
4	1.0340	28.700000	24	184.250000	71.25	34.4	97.3	100.0	101.9	61.312766	42.2

Qn 5(a), part 2 - compute the difference in mean value for each feature

In [110]:
```
# getting the mean row from bodyfat3b using the .describe() method
description_3b = bodyfat3b.describe()
mean_3b = description_3b.loc["mean"]
mean_3b
```

Out[110]:
```
density      1.058380
bodyfat     17.345652
age         33.660000
weight     184.259302
height      70.166667
neck        37.876596
chest      101.110204
abdomen     92.080000
hip        102.468182
thigh       61.312766
knee        38.753061
ankle       23.620833
biceps      32.956522
forearm     28.852000
wrist       18.131250
Name: mean, dtype: float64
```

In [111]:
```
# getting the mean row from the bodyfat2 using the .descirbe() method
description_2 = bodyfat2.describe()
mean_2 = description_2.loc["mean"]
mean_2
```

Out[111]:
```
density      1.05838
bodyfat     17.88200
age         33.66000
weight     183.25800
height      70.13000
neck        38.01600
chest      101.12800
abdomen     92.08000
hip        102.05800
thigh       61.50600
knee        38.82800
ankle       23.58600
biceps      32.79800
forearm     28.85200
wrist       18.11800
Name: mean, dtype: float64
```

In [112]:
```
# getting the difference between the two means
mean_differnces = abs(mean_3b - mean_2)
mean_differnces
```

Out[112]:
```
density    0.000000
bodyfat    0.536348
age        0.000000
weight     1.001302
height     0.036667
neck       0.139404
chest      0.017796
abdomen    0.000000
hip        0.410182
thigh      0.193234
knee       0.074939
ankle      0.034833
biceps     0.158522
forearm    0.000000
wrist      0.013250
Name: mean, dtype: float64
```

In [141]:
```
# putting the means from bodyfat2, bodyfat3b, and their differences into a sin
gle dataframe
diff = pd.DataFrame(list(zip(mean_2, mean_3b, mean_differnces)), columns = ["b
odyfat2_mean", "bodyfat3b_mean", "diff_mean"], index=[mean_3b.index])
print("Qn 5(a), part 2: ")
"\n"
diff
```

```
Qn 5(a), part 2:
```

Out[141]:

	bodyfat2_mean	bodyfat3b_mean	diff_mean
density	1.05838	1.058380	0.000000
bodyfat	17.88200	17.345652	0.536348
age	33.66000	33.660000	0.000000
weight	183.25800	184.259302	1.001302
height	70.13000	70.166667	0.036667
neck	38.01600	37.876596	0.139404
chest	101.12800	101.110204	0.017796
abdomen	92.08000	92.080000	0.000000
hip	102.05800	102.468182	0.410182
thigh	61.50600	61.312766	0.193234
knee	38.82800	38.753061	0.074939
ankle	23.58600	23.620833	0.034833
biceps	32.79800	32.956522	0.158522
forearm	28.85200	28.852000	0.000000
wrist	18.11800	18.131250	0.013250

Qn 5(b), part 1 - replacing missing values with median

In [114]:
```
# reading in bodyfat3c.csv
bodyfat3c = pd.read_csv("bodyfat3c.csv")
```

In [115]:
```
# replacing the missing values with the median of that particular feature
columns = list(bodyfat3c.columns)
for column in columns:
    bodyfat3c[column].fillna(value=bodyfat3c[column].median(), inplace = True)
```

In [142]:
```
# printing the first 20 datasets to check if it works
print("Qn 5(b), part 1: ")
"\n"
bodyfat3c.head()
```

```
Qn 5(b), part 1:
```

Out[142]:

	density	bodyfat	age	weight	height	neck	chest	abdomen	hip	thigh	knee	ankle	bic
0	1.0708	12.3	23	154.25	67.75	36.2	93.1	85.2	94.5	59.0	37.3	21.9	3
1	1.0853	6.1	22	173.25	72.25	38.5	93.6	83.0	98.7	58.7	37.3	23.4	3
2	1.0414	15.4	22	154.00	66.25	34.0	95.8	87.9	99.2	59.6	38.9	24.0	3
3	1.0751	10.4	26	182.00	72.25	37.4	101.8	86.4	101.2	60.1	37.3	22.8	3
4	1.0340	28.7	24	184.25	71.25	34.4	97.3	100.0	101.9	60.1	42.2	24.0	3

Qn 5(b), part 2 - compute the difference in median value for each feature

In [117]:
```
# getting the median row from bodyfat3c using the .describe() method
description_3c = bodyfat3c.describe()
median_3c = description_3c.loc["50%"]
median_3c
```

Out[117]:
```
density      1.0616
bodyfat     15.4000
age         32.0000
weight     182.0000
height      70.0000
neck        38.0000
chest      100.9000
abdomen     89.1500
hip        101.5500
thigh       60.1000
knee        38.7000
ankle       23.1000
biceps      32.5000
forearm     29.0000
wrist       18.2000
Name: 50%, dtype: float64
```

In [118]:
```
# getting the median row from the bodyfat2 using the .descirbe() method
description_2 = bodyfat2.describe()
median_2 = description_2.loc["50%"]
median_2
```

Out[118]:
```
density      1.0616
bodyfat     16.2500
age         32.0000
weight     182.8750
height      70.0000
neck        38.1000
chest      101.1000
abdomen     89.1500
hip        100.4500
thigh       61.6000
knee        38.7000
ankle       23.1000
biceps      32.4500
forearm     29.0000
wrist       18.2000
Name: 50%, dtype: float64
```

In [119]:
```
# getting the difference between the two medians
median_differnces = abs(median_3c - median_2)
median_differnces
```

Out[119]:
```
density    0.000
bodyfat    0.850
age        0.000
weight     0.875
height     0.000
neck       0.100
chest      0.200
abdomen    0.000
hip        1.100
thigh      1.500
knee       0.000
ankle      0.000
biceps     0.050
forearm    0.000
wrist      0.000
Name: 50%, dtype: float64
```

In [143]:
```
# putting the medians from bodyfat2, bodyfat3c, and their differences into a s
ingle dataframe
diff = pd.DataFrame(list(zip(median_2, median_3c, median_differnces)), columns
= ["bodyfat2_median", "bodyfat3b_median", "diff_median"], index=[mean_differnc
es.index])
print("Qn 5b, part 2: ")
"\n"
diff
```

```
Qn 5b, part 2:
```

Out[143]:

	bodyfat2_median	bodyfat3b_median	diff_median
density	1.0616	1.0616	0.000
bodyfat	16.2500	15.4000	0.850
age	32.0000	32.0000	0.000
weight	182.8750	182.0000	0.875
height	70.0000	70.0000	0.000
neck	38.1000	38.0000	0.100
chest	101.1000	100.9000	0.200
abdomen	89.1500	89.1500	0.000
hip	100.4500	101.5500	1.100
thigh	61.6000	60.1000	1.500
knee	38.7000	38.7000	0.000
ankle	23.1000	23.1000	0.000
biceps	32.4500	32.5000	0.050
forearm	29.0000	29.0000	0.000
wrist	18.2000	18.2000	0.000

Qn 6(i)

In [121]:
```
bodyfat2.describe() # see the mean and standard deviation
```

Out[121]:

	density	bodyfat	age	weight	height	neck	chest	abdomen
count	50.000000	50.000000	50.000000	50.000000	50.00000	50.000000	50.000000	50.000000
mean	1.058380	17.882000	33.660000	183.258000	70.13000	38.016000	101.128000	92.080000
std	0.022115	9.842032	8.402162	40.257781	2.66508	3.286917	10.614315	14.508899
min	1.010100	3.700000	22.000000	125.250000	64.75000	31.500000	83.400000	70.400000
25%	1.044550	10.500000	27.000000	155.500000	68.00000	36.200000	93.500000	82.625000
50%	1.061600	16.250000	32.000000	182.875000	70.00000	38.100000	101.100000	89.150000
75%	1.074875	23.875000	40.750000	202.812500	72.06250	39.400000	106.150000	99.700000
max	1.091100	40.100000	50.000000	363.150000	76.00000	51.200000	136.200000	148.100000

In [122]:
```
columns = list(bodyfat2.columns) # create a list of column names
```

In [123]:
```
# getting the new dataframe with normalized values

normalized_df = bodyfat2.copy() # have a new copy for bodyfat2 for editing pur
poses
for column in columns:
    feature_SD = normalized_df[column].std()
    feature_mean = normalized_df[column].mean()
    normalized_df[column] = (normalized_df[column] - feature_mean) / feature_S
D

normalized_df.head() # printing out the first 5 datasets for checking
```

Out[123]:

	density	bodyfat	age	weight	height	neck	chest	abdomen	h
0	0.561613	-0.567159	-1.268721	-0.720556	-0.893031	-0.552493	-0.756337	-0.474192	-0.7222
1	1.217281	-1.197111	-1.387738	-0.248598	0.795473	0.147250	-0.709231	-0.625823	-0.3208
2	-0.767810	0.753706	-1.387738	-0.726766	-1.455866	-1.221814	-0.501964	-0.288099	-0.2731
3	0.756053	-0.760209	-0.911670	0.037061	0.795473	-0.187410	0.063311	-0.391484	-0.0819
4	-1.102426	1.099163	-1.149704	0.024641	0.420250	-1.100119	-0.360645	0.545872	-0.0150

In [124]:
```
# printing the top 3 and bottom 3 rows from normalized_df
topThree = normalized_df.head(3)
bottomThree = normalized_df.tail(3)
```

In [144]:
```
# getting the dataset that represents the top 3 and bottom 3 rows
topThreeAndBottomThree_normalized_df = pd.concat([topThree, bottomThree])
print("Qn 6(i): ")
"\n"
topThreeAndBottomThree_normalized_df
```

Qn 6(i):

Out[144]:

	density	bodyfat	age	weight	height	neck	chest	abdomen	
0	0.561613	-0.567159	-1.268721	-0.720556	-0.893031	-0.552493	-0.756337	-0.474192	-0.722
1	1.217281	-1.197111	-1.387738	-0.248598	0.795473	0.147250	-0.709231	-0.625823	-0.320
2	-0.767810	0.753706	-1.387738	-0.726766	-1.455866	-1.221814	-0.501964	-0.288099	-0.273
47	1.271543	-1.247913	0.635551	-0.863386	0.420250	-1.039272	-1.067238	-0.867054	-0.894
48	0.425958	-0.435073	1.349653	-1.180095	-0.611614	-1.586898	-0.831707	-0.598254	-1.114
49	1.443374	-1.410481	1.587687	-1.385024	-1.268254	-1.221814	-1.670197	-1.494255	-1.419

Qn 6(ii)

In [145]:
```
noOfFeaturesGreaterThanItsMean = {} # creating an empty dictionary to store ke
y-value pairs
columns = list(normalized_df.columns)
for column in columns:
    count = normalized_df[column].loc[normalized_df[column] > 0].count()
    noOfFeaturesGreaterThanItsMean[column] = [count] # key-value pair represen
tation

noOfFeaturesGreaterThanItsMean_df = pd.DataFrame(noOfFeaturesGreaterThanItsMea
n, index = ["No. of features more than its mean"] )
print("Qn 6(ii): ")
"\n"
noOfFeaturesGreaterThanItsMean_df
```

Qn 6(ii):

Out[145]:

	density	bodyfat	age	weight	height	neck	chest	abdomen	hip	thigh	knee	ankle
No. of features more than its mean	27	23	21	25	23	26	25	21	22	25	23	21

In [ ]: