In [103]:

```
import matplotlib.pyplot as plt
import numpy as np
import random as rn
import pandas as pd
```

In [106]:

```
df=pd.read_csv("D:\python jupyter\data files\Breast_cancer_data.csv")
df.head()
```

Out[106]:

	mean_radius	mean_texture	mean_perimeter	mean_area	mean_smoothness	diagnosis
0	17.99	10.38	122.80	1001.0	0.11840	0
1	20.57	17.77	132.90	1326.0	0.08474	0
2	19.69	21.25	130.00	1203.0	0.10960	0
3	11.42	20.38	77.58	386.1	0.14250	0
4	20.29	14.34	135.10	1297.0	0.10030	0

In [113]:

```
df.describe()
```

Out[113]:

	mean_radius	mean_texture	mean_perimeter	mean_area	mean_smoothness	diagnosis
count	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000
mean	14.127292	19.289649	91.969033	654.889104	0.096360	0.627417
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.483918
min	6.981000	9.710000	43.790000	143.500000	0.052630	0.000000
25%	11.700000	16.170000	75.170000	420.300000	0.086370	0.000000
50%	13.370000	18.840000	86.240000	551.100000	0.095870	1.000000
75%	15.780000	21.800000	104.100000	782.700000	0.105300	1.000000
max	28.110000	39.280000	188.500000	2501.000000	0.163400	1.000000

Gives 95 percentile value from mean_smoothness

In [118]:

```
max_th=df["mean_smoothness"].quantile(0.95)
max_th
```

Out[118]:

```
0.11878000000000001
```

In [117]:

```
df[df["mean_smoothness"]>max_th]
```

Out[117]:

	mean_radius	mean_texture	mean_perimeter	mean_area	mean_smoothness	diagnosis
3	11.420	20.38	77.58	386.1	0.1425	0
5	12.450	15.70	82.57	477.1	0.1278	0
7	13.710	20.83	90.20	577.9	0.1189	0
8	13.000	21.82	87.50	519.8	0.1273	0
32	17.020	23.98	112.80	899.3	0.1197	0
41	10.950	21.35	71.90	371.1	0.1227	0
61	8.598	20.98	54.66	221.8	0.1243	1
76	13.530	10.94	87.91	559.2	0.1291	1
78	20.180	23.97	143.70	1245.0	0.1286	0
83	19.100	26.29	129.10	1132.0	0.1215	0
105	13.110	15.56	87.21	530.2	0.1398	0
108	22.270	19.67	152.80	1509.0	0.1326	0
122	24.250	20.20	166.20	1761.0	0.1447	0
172	15.460	11.89	102.50	736.9	0.1257	0
196	13.770	22.29	90.63	588.9	0.1200	0
203	13.810	23.75	91.56	597.8	0.1323	0
257	15.320	17.27	103.20	713.3	0.1335	0
275	11.890	17.36	76.20	435.6	0.1225	1
351	15.750	19.22	107.10	758.6	0.1243	0
379	11.080	18.83	73.30	361.6	0.1216	0
380	11.270	12.96	73.16	386.3	0.1237	1
400	17.910	21.02	124.40	994.0	0.1230	0
504	9.268	12.87	61.49	248.7	0.1634	1
505	9.676	13.14	64.12	272.5	0.1255	1
507	11.060	17.12	71.25	366.5	0.1194	1
518	12.880	18.22	84.45	493.1	0.1218	1
520	9.295	13.90	59.96	257.8	0.1371	1
528	13.940	13.17	90.31	594.2	0.1248	1
537	11.690	24.44	76.37	406.4	0.1236	1

Gives the 5 percentile value of mean_smoothness

In [120]:

```
min_th=df["mean_smoothness"].quantile(0.05)
min_th
```

Out[120]:

```
0.07504200000000001
```

In [121]:

```
df.loc[df["mean_smoothness"]<min_th]
```

Out[121]:

	mean_radius	mean_texture	mean_perimeter	mean_area	mean_smoothness	diagnosis
92	13.27	14.76	84.74	551.7	0.07355	1
124	13.37	16.39	86.10	553.5	0.07115	1
157	16.84	19.46	108.40	880.2	0.07445	1
178	13.01	22.22	82.01	526.4	0.06251	1
192	9.72	18.22	60.73	288.1	0.06950	1
197	18.08	21.84	117.40	1024.0	0.07371	0
231	11.32	27.08	71.76	395.7	0.06883	1
246	13.20	17.43	84.13	541.6	0.07215	1
270	14.29	16.82	90.30	632.6	0.06429	1
287	12.89	13.12	81.89	515.9	0.06955	1
298	14.26	18.17	91.22	633.1	0.06576	1
305	11.60	24.49	74.23	417.2	0.07474	1
307	9.00	14.40	56.36	246.3	0.07005	1
308	13.50	12.71	85.69	566.2	0.07376	1
354	11.14	14.07	71.24	384.6	0.07274	1
360	12.54	18.07	79.42	491.9	0.07436	1
382	12.05	22.72	78.75	447.8	0.06935	1
387	13.88	16.16	88.37	596.6	0.07026	1
402	12.96	18.29	84.18	525.2	0.07351	1
450	11.87	21.54	76.83	432.0	0.06613	1
462	14.40	26.99	92.25	646.1	0.06995	1
464	13.17	18.22	84.28	537.3	0.07466	1
477	13.90	16.62	88.97	599.4	0.06828	1
489	16.69	20.20	107.10	857.6	0.07497	0
493	12.46	12.83	78.83	477.3	0.07372	1
494	13.16	20.54	84.06	538.7	0.07335	1
550	10.86	21.48	68.51	360.5	0.07431	1
561	11.20	29.37	70.67	386.0	0.07449	1
568	7.76	24.54	47.92	181.0	0.05263	1

In [142]:

```
df[df["mean_smoothness"]<min_th].sort_values(by="mean_smoothness",axis=0,ascending=False)
```

Out[142]:

	mean_radius	mean_texture	mean_perimeter	mean_area	mean_smoothness	diagnosis
489	16.69	20.20	107.10	857.6	0.07497	0
305	11.60	24.49	74.23	417.2	0.07474	1
464	13.17	18.22	84.28	537.3	0.07466	1
561	11.20	29.37	70.67	386.0	0.07449	1
157	16.84	19.46	108.40	880.2	0.07445	1
360	12.54	18.07	79.42	491.9	0.07436	1
550	10.86	21.48	68.51	360.5	0.07431	1
308	13.50	12.71	85.69	566.2	0.07376	1
493	12.46	12.83	78.83	477.3	0.07372	1
197	18.08	21.84	117.40	1024.0	0.07371	0
92	13.27	14.76	84.74	551.7	0.07355	1
402	12.96	18.29	84.18	525.2	0.07351	1
494	13.16	20.54	84.06	538.7	0.07335	1
354	11.14	14.07	71.24	384.6	0.07274	1
246	13.20	17.43	84.13	541.6	0.07215	1
124	13.37	16.39	86.10	553.5	0.07115	1
387	13.88	16.16	88.37	596.6	0.07026	1
307	9.00	14.40	56.36	246.3	0.07005	1
462	14.40	26.99	92.25	646.1	0.06995	1
287	12.89	13.12	81.89	515.9	0.06955	1
192	9.72	18.22	60.73	288.1	0.06950	1
382	12.05	22.72	78.75	447.8	0.06935	1
231	11.32	27.08	71.76	395.7	0.06883	1
477	13.90	16.62	88.97	599.4	0.06828	1
450	11.87	21.54	76.83	432.0	0.06613	1
298	14.26	18.17	91.22	633.1	0.06576	1
270	14.29	16.82	90.30	632.6	0.06429	1
178	13.01	22.22	82.01	526.4	0.06251	1
568	7.76	24.54	47.92	181.0	0.05263	1

df1 is dataset with outliers removed using percentile method

In [145]:

```
df1=df[(df["mean_smoothness"]>min_th) & (df["mean_smoothness"]<max_th)]
df1.head()
```

Out[145]:

	mean_radius	mean_texture	mean_perimeter	mean_area	mean_smoothness	diagnosis
0	17.99	10.38	122.8	1001.0	0.11840	0
1	20.57	17.77	132.9	1326.0	0.08474	0
2	19.69	21.25	130.0	1203.0	0.10960	0
4	20.29	14.34	135.1	1297.0	0.10030	0
6	18.25	19.98	119.6	1040.0	0.09463	0

In [146]:

```
df1.shape
```

Out[146]:

```
(511, 6)
```

In [166]:

```
df["height"].quantile(1) #gives the max value
```

Out[166]:

```
78.99874235
```

In [169]:

```
df.height.describe()
```

Out[169]:

```
count    10000.000000
mean        66.367560
std          3.847528
min         54.263133
25%         63.505620
50%         66.318070
75%         69.174262
max         78.998742
Name: height, dtype: float64
```

In [156]:

```
df=pd.read_csv("D:\python jupyter\data files\heights.csv")
df.head()
```

Out[156]:

	gender	height
0	Male	73.847017
1	Male	68.781904
2	Male	74.110105
3	Male	71.730978
4	Male	69.881796

In [159]:

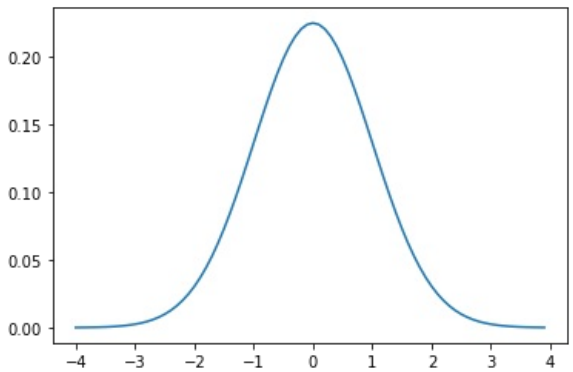
```
import math
x=np.arange(-4,4,0.1)
y=1/(2**(.5)*math.pi)*np.exp(-(x**2)/2)
```

In [160]:

```
plt.plot(x,y)
```

Out[160]:

```
[<matplotlib.lines.Line2D at 0x272133cfb80>]
```



In [161]:

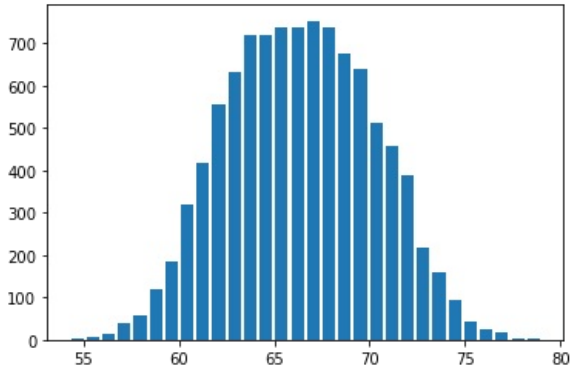
```
4.5/3.3
```

Out[161]:

```
1.3636363636363638
```

In [173]:

```
plt.hist(df.height,bins=30,rwidth=0.8)
plt.show()
```



In [177]:

```
df.height.max()
```

Out[177]:

```
78.99874235
```

In [178]:

```
df.height.quantile(1)
```

Out[178]:

```
78.99874235
```

Using standard deviation method for outlier detection

In [203]:

```
stdv=df.height.std()
mn=df.height.mean()
```

using 3 standard deviation from the mean for upper and lower threshold values

In [204]:

```
lowerth=mn-3*stdv
upperth=mn+3*stdv
```

In [205]:

```
lowerth
```

Out[205]:

```
54.824975392479274
```

In [206]:

```
upperth
```

Out[206]:

```
77.91014411725271
```

In [207]:

```
df1=df[(df.height>lowerth)&(df.height<upperth)]
df1.shape
```

Out[207]:

```
(9993, 2)
```

In [208]:

```
df1.height.max()
```

Out[208]:

```
77.54718634
```

In [209]:

```
df.height.max()
```

Out[209]:

```
78.99874235
```

In [210]:

```
df1.height.min()
```

Out[210]:

```
54.87372753
```

In [211]:

```
df.height.min()
```

Out[211]:

```
54.26313333
```

In [212]:

```
df[(df.height<lowerth)|(df.height>upperth)]
```

Out[212]:

	gender	height
994	Male	78.095867
1317	Male	78.462053
2014	Male	78.998742
3285	Male	78.528210
3757	Male	78.621374
6624	Female	54.616858
9285	Female	54.263133

In [213]:

```
df["z-score"]=(df.height-df.height.mean())/(df.height.std())  #calculation of z-score
df.head()
```

Out[213]:

	gender	height	z-score
0	Male	73.847017	1.943964
1	Male	68.781904	0.627505
2	Male	74.110105	2.012343
3	Male	71.730978	1.393991
4	Male	69.881796	0.913375

Using z-score method, quite similar to standard deviation method

In [218]:

```
df_new=df[(df["z-score"]<3)&(df["z-score"]>-3)]
df_new.head()
```

Out[218]:

	gender	height	z-score
0	Male	73.847017	1.943964
1	Male	68.781904	0.627505
2	Male	74.110105	2.012343
3	Male	71.730978	1.393991
4	Male	69.881796	0.913375

In [219]:

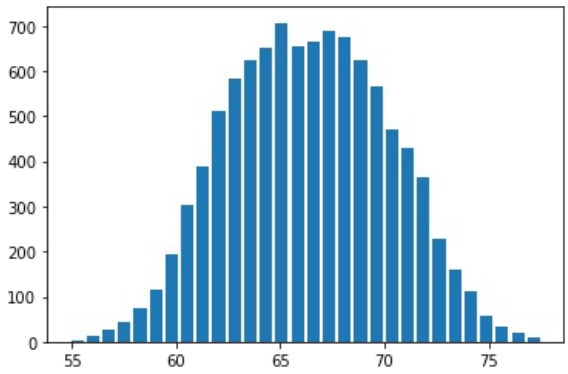
```
df[(df["z-score"]>3)|(df["z-score"]<-3)]
```

Out[219]:

	gender	height	z-score
994	Male	78.095867	3.048271
1317	Male	78.462053	3.143445
2014	Male	78.998742	3.282934
3285	Male	78.528210	3.160640
3757	Male	78.621374	3.184854
6624	Female	54.616858	-3.054091
9285	Female	54.263133	-3.146027

In [221]:

```
plt.hist(df_new.height,bins=30,rwidth=0.8)
plt.show()
```



Taking a new dataframe performing the same operations for practice

In [224]:

```
df=pd.read_csv("D:\python jupyter\data files\\banglore_house_prices.csv")
df.head()
```

Out[224]:

	location	size	total_sqft	bath	price	bhk	price_per_sqft
0	Electronic City Phase II	2 BHK	1056.0	2.0	39.07	2	3699
1	Chikka Tirupathi	4 Bedroom	2600.0	5.0	120.00	4	4615
2	Uttarahalli	3 BHK	1440.0	2.0	62.00	3	4305
3	Lingadheeranahalli	3 BHK	1521.0	3.0	95.00	3	6245
4	Kothanur	2 BHK	1200.0	2.0	51.00	2	4250

In [231]:

```
lowert=df["price_per_sqft"].quantile(0.001)
lowert
```

Out[231]:

```
1366.184
```

In [232]:

```
uppert=df["price_per_sqft"].quantile(0.999)
uppert
```

Out[232]:

```
50959.36200000099
```

In [233]:

```
df.describe()
```

Out[233]:

	total_sqft	bath	price	bhk	price_per_sqft
count	13200.000000	13200.000000	13200.000000	13200.000000	1.320000e+04
mean	1555.302783	2.691136	112.276178	2.800833	7.920337e+03
std	1237.323445	1.338915	149.175995	1.292843	1.067272e+05
min	1.000000	1.000000	8.000000	1.000000	2.670000e+02
25%	1100.000000	2.000000	50.000000	2.000000	4.267000e+03
50%	1275.000000	2.000000	71.850000	3.000000	5.438000e+03
75%	1672.000000	3.000000	120.000000	3.000000	7.317000e+03
max	52272.000000	40.000000	3600.000000	43.000000	1.200000e+07

In [234]:

```
df_new=df[(df["price_per_sqft"]>lowert)&(df["price_per_sqft"]<uppert)]
df_new.head()
```

Out[234]:

	location	size	total_sqft	bath	price	bhk	price_per_sqft
0	Electronic City Phase II	2 BHK	1056.0	2.0	39.07	2	3699
1	Chikka Tirupathi	4 Bedroom	2600.0	5.0	120.00	4	4615
2	Uttarahalli	3 BHK	1440.0	2.0	62.00	3	4305
3	Lingadheeranahalli	3 BHK	1521.0	3.0	95.00	3	6245
4	Kothanur	2 BHK	1200.0	2.0	51.00	2	4250

In [235]:

```
df.shape
```

Out[235]:

```
(13200, 7)
```

In [236]:

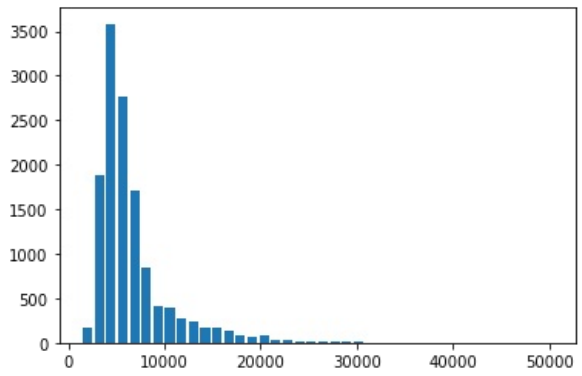
```
df_new.shape
```

Out[236]:

```
(13172, 7)
```

In [246]:

```
plt.hist(df_new.price_per_sqft,bins=40,rwidth=0.8)
plt.show()
```



In [244]:

```
df["price_per_sqft"].mean()
```

Out[244]:

```
7920.336742424242
```

In [247]:

```
df["price_per_sqft"].median()
```

Out[247]:

```
5438.0
```

In [255]:

```
stdev=df_new["price_per_sqft"].std()
meanval=df_new["price_per_sqft"].mean()
```

In [256]:

```
lowerthreshold=meanval-4*stdev
upperthreshold=meanval+4*stdev
```

In [267]:

```
df_mynew=df_new[(df_new["price_per_sqft"]>lowerthreshold)&(df_new["price_per_sqft"]<upperthreshold)]
```

In [268]:

```
df_mynew.head()
```

Out[268]:

	location	size	total_sqft	bath	price	bhk	price_per_sqft
0	Electronic City Phase II	2 BHK	1056.0	2.0	39.07	2	3699
1	Chikka Tirupathi	4 Bedroom	2600.0	5.0	120.00	4	4615
2	Uttarahalli	3 BHK	1440.0	2.0	62.00	3	4305
3	Lingadheeranahalli	3 BHK	1521.0	3.0	95.00	3	6245
4	Kothanur	2 BHK	1200.0	2.0	51.00	2	4250

In [269]:

```
df_mynew.shape
```

Out[269]:

```
(13047, 7)
```

In [270]:

```
df_new.shape
```

Out[270]:

```
(13172, 7)
```

In [271]:

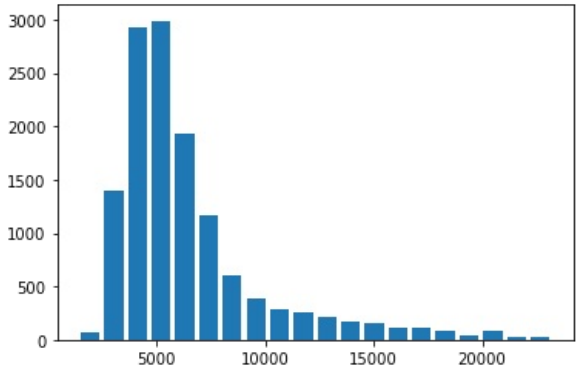
```
df_new[(df_new["price_per_sqft"]<lowerthreshold)|(df_new["price_per_sqft"]>upperthreshold)].shape
```

Out[271]:

```
(125, 7)
```

In [272]:

```
plt.hist(df_mynew["price_per_sqft"],bins=20,rwidth=0.8)
plt.show()
```



In [274]:

```
mymean=df["price_per_sqft"].mean()
mystd=df["price_per_sqft"].std()
```

Out[274]:

```
106727.16032810867
```

In [276]:

```
df_mynew["zscore"]=(df_mynew["price_per_sqft"]-mymean)/(mystd)
df_mynew.head()
```

```
<ipython-input-276-439251d624cb>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexi
ng.html#returning-a-view-versus-a-copy
  df_mynew["zscore"]=(df_mynew["price_per_sqft"]-mymean)/(mystd)
```

Out[276]:

	location	size	total_sqft	bath	price	bhk	price_per_sqft	zscore
0	Electronic City Phase II	2 BHK	1056.0	2.0	39.07	2	3699	-0.039553
1	Chikka Tirupathi	4 Bedroom	2600.0	5.0	120.00	4	4615	-0.030970
2	Uttarahalli	3 BHK	1440.0	2.0	62.00	3	4305	-0.033875
3	Lingadheeranahalli	3 BHK	1521.0	3.0	95.00	3	6245	-0.015697
4	Kothanur	2 BHK	1200.0	2.0	51.00	2	4250	-0.034390

In [277]:

```
df_new["zscore"]=(df_new["price_per_sqft"]-meanval)/(stdev)
df_new.head()
```

```
<ipython-input-277-c1cb86943c43>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexi
ng.html#returning-a-view-versus-a-copy
  df_new["zscore"]=(df_new["price_per_sqft"]-meanval)/(stdev)
```

Out[277]:

	location	size	total_sqft	bath	price	bhk	price_per_sqft	zscore
0	Electronic City Phase II	2 BHK	1056.0	2.0	39.07	2	3699	-0.715923
1	Chikka Tirupathi	4 Bedroom	2600.0	5.0	120.00	4	4615	-0.494722
2	Uttarahalli	3 BHK	1440.0	2.0	62.00	3	4305	-0.569583
3	Lingadheeranahalli	3 BHK	1521.0	3.0	95.00	3	6245	-0.101099
4	Kothanur	2 BHK	1200.0	2.0	51.00	2	4250	-0.582864

In [279]:

```
df_znew=df_new[(df_new["zscore"]>-4)&(df_new["zscore"]<4)]
df_znew.head()
```

Out[279]:

	location	size	total_sqft	bath	price	bhk	price_per_sqft	zscore
0	Electronic City Phase II	2 BHK	1056.0	2.0	39.07	2	3699	-0.715923
1	Chikka Tirupathi	4 Bedroom	2600.0	5.0	120.00	4	4615	-0.494722
2	Uttarahalli	3 BHK	1440.0	2.0	62.00	3	4305	-0.569583
3	Lingadheeranahalli	3 BHK	1521.0	3.0	95.00	3	6245	-0.101099
4	Kothanur	2 BHK	1200.0	2.0	51.00	2	4250	-0.582864

In [281]:

```
df_znew.shape
```

Out[281]:

```
(13047, 8)
```

In [282]:

```
df_new.shape
```

Out[282]:

```
(13172, 8)
```

In [283]:

```
df_new[(df_new["zscore"]<-4)|(df_new["zscore"]>4)]
```

Out[283]:

	location	size	total_sqft	bath	price	bhk	price_per_sqft	zscore
9	other	6 Bedroom	1020.0	6.0	370.0	6	36274	7.150495
45	HSR Layout	8 Bedroom	600.0	9.0	200.0	8	33333	6.440283
190	Bellandur	4 Bedroom	1200.0	5.0	325.0	4	27083	4.930994
733	Cunningham Road	4 BHK	5270.0	4.0	1250.0	4	23719	4.118633
760	other	9 Bedroom	600.0	9.0	190.0	9	31666	6.037725
...	...	...	...	...	...	...	...	...
13081	other	6 Bedroom	8000.0	6.0	2800.0	6	35000	6.842841
13094	other	4 Bedroom	1200.0	5.0	325.0	4	27083	4.930994
13127	other	4 Bedroom	1200.0	5.0	325.0	4	27083	4.930994
13185	Hulimavu	1 BHK	500.0	1.0	220.0	1	44000	9.016218
13186	other	4 Bedroom	1200.0	5.0	325.0	4	27083	4.930994

125 rows × 8 columns

In [284]:

```
df_h=pd.read_csv("D:\python jupyter\data files\heights.csv")
df_h.head()
```

Out[284]:

	gender	height
0	Male	73.847017
1	Male	68.781904
2	Male	74.110105
3	Male	71.730978
4	Male	69.881796

In [285]:

```
df_h.describe()
```

Out[285]:

	height
count	10000.000000
mean	66.367560
std	3.847528
min	54.263133
25%	63.505620
50%	66.318070
75%	69.174262
max	78.998742

using IQR method for outlier detection, not present in above example

In [286]:

```
Q1=df_h.height.quantile(0.25)
Q2=df_h.height.quantile(0.5)
Q3=df_h.height.quantile(0.75)
```

In [287]:

```
IQR=Q3-Q1
IQR
```

Out[287]:

```
5.668641247499998
```

In [288]:

```
upperl=Q3+1.5*IQR
lowerl=Q1-1.5*IQR
lowerl,upperl
```

Out[288]:

```
(55.00265860875, 77.67722359874999)
```

In [289]:

```
df_hnew=df_h[(df_h["height"]>lowerl)&(df_h["height"]<upperl)]
df_hnew.shape
```

Out[289]:

```
(9992, 2)
```

In [290]:

```
df_h.shape
```

Out[290]:

```
(10000, 2)
```

In [ ]: