Probability and Statstics

Dr. Faisal Bukhari
Associate Professor
Department of Data Science
Faculty of Computing and Information Technology
University of the Punjab

Textbooks

- **Probability & Statistics for Engineers & Scientists**, Ninth Edition, Ronald E. Walpole, Raymond H. Myer

- **Elementary Statistics: Picturing the World,** 6th Edition, Ron Larson and Betsy Farber

- **Elementary Statistics,** 13th Edition, Mario F. Triola

Reference books

- **Probability Demystified**, Allan G. Bluman
- **Schaum's Outline of Probability and Statistics**
- **MATLAB Primer**, Seventh Edition
- **MATLAB Demystified** by McMahon, David

References

Readings for these lecture notes:

- Probability & Statistics for Engineers & Scientists, Ninth edition, Ronald E. Walpole, Raymond H. Myer
- Probability Demystified, Allan G. Bluman
- Schaum's Outline of Probability, Seymour Lipschutz, Marc Lipson

These notes contain material from the above resources.

Textbooks

- **Probability & Statistics for Engineers & Scientists**, Ninth Edition, Ronald E. Walpole, Raymond H. Myer

- **Elementary Statistics: Picturing the World,** 6th Edition, Ron Larson and Betsy Farber

- **Elementary Statistics,** 13th Edition, Mario F. Triola

Reference books

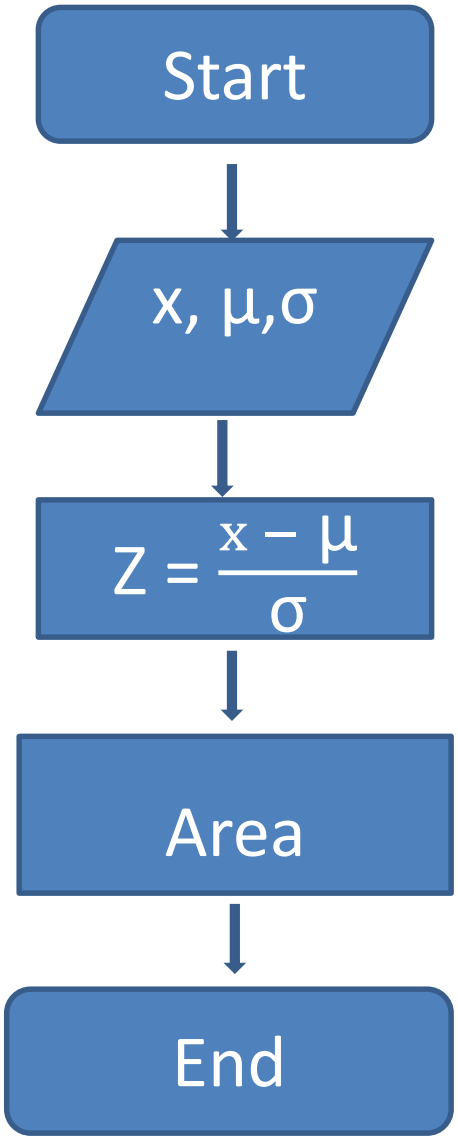
- **Probability and Statistical Inference, Ninth Edition,** Robert V. Hogg, Elliot A. Tanis, Dale L. Zimmerman
- **Probability Demystified**, Allan G. Bluman
- **Practical Statistics for Data Scientists: 50 Essential Concepts,** Peter Bruce and Andrew Bruce
- **Schaum's Outline of Probability,** Second Edition, Seymour Lipschutz, Marc Lipson
- **Python for Probability, Statistics, and Machine Learning,** José Unpingco

References

Readings for these lecture notes:

- **Schaum's Outline of Probability, Second Edition (Schaum's Outlines)** by Seymour Lipschutz, Marc Lipson
- **Probability & Statistics for Engineers & Scientists**, Ninth Edition, Ronald E. Walpole, Raymond H. Myer
- **Anoka Hennepin Probability and Statistics by** by Mr. Michael Engelhaupt
- These notes contain material from the above resources.

Flowchart for Computing Probability



Flowchart for Computing x (Inverse Problem)

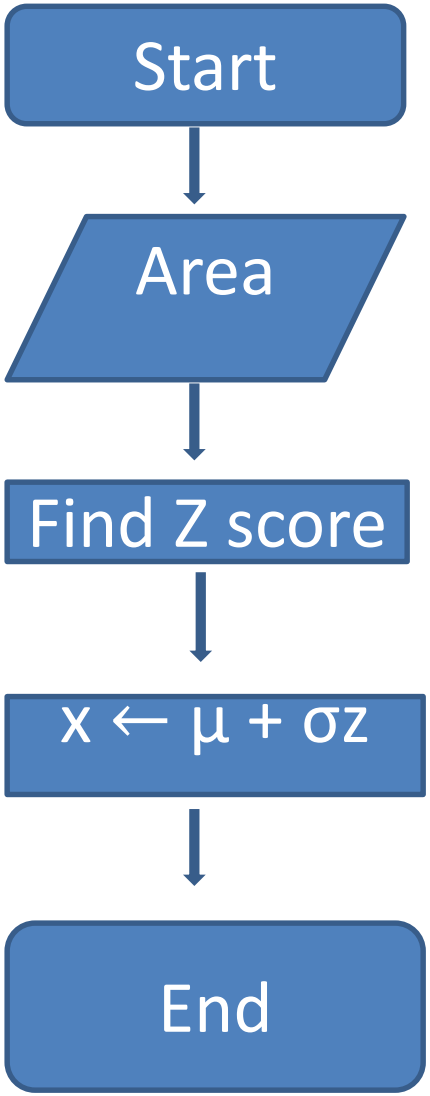


Table A.3 Areas under the Normal Curve

z	.00	.01	.02	.03	.04	.05	.06	.07	.08	.09
−3.4	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0002
−3.3	0.0005	0.0005	0.0005	0.0004	0.0004	0.0004	0.0004	0.0004	0.0004	0.0003
−3.2	0.0007	0.0007	0.0006	0.0006	0.0006	0.0006	0.0006	0.0005	0.0005	0.0005
−3.1	0.0010	0.0009	0.0009	0.0009	0.0008	0.0008	0.0008	0.0008	0.0007	0.0007
−3.0	0.0013	0.0013	0.0013	0.0012	0.0012	0.0011	0.0011	0.0011	0.0010	0.0010
−2.9	0.0019	0.0018	0.0018	0.0017	0.0016	0.0016	0.0015	0.0015	0.0014	0.0014
−2.8	0.0026	0.0025	0.0024	0.0023	0.0023	0.0022	0.0021	0.0021	0.0020	0.0019
−2.7	0.0035	0.0034	0.0033	0.0032	0.0031	0.0030	0.0029	0.0028	0.0027	0.0026
−2.6	0.0047	0.0045	0.0044	0.0043	0.0041	0.0040	0.0039	0.0038	0.0037	0.0036
−2.5	0.0062	0.0060	0.0059	0.0057	0.0055	0.0054	0.0052	0.0051	0.0049	0.0048
−2.4	0.0082	0.0080	0.0078	0.0075	0.0073	0.0071	0.0069	0.0068	0.0066	0.0064
−2.3	0.0107	0.0104	0.0102	0.0099	0.0096	0.0094	0.0091	0.0089	0.0087	0.0084
−2.2	0.0139	0.0136	0.0132	0.0129	0.0125	0.0122	0.0119	0.0116	0.0113	0.0110
−2.1	0.0179	0.0174	0.0170	0.0166	0.0162	0.0158	0.0154	0.0150	0.0146	0.0143
−2.0	0.0228	0.0222	0.0217	0.0212	0.0207	0.0202	0.0197	0.0192	0.0188	0.0183
−1.9	0.0287	0.0281	0.0274	0.0268	0.0262	0.0256	0.0250	0.0244	0.0239	0.0233
−1.8	0.0359	0.0351	0.0344	0.0336	0.0329	0.0322	0.0314	0.0307	0.0301	0.0294
−1.7	0.0446	0.0436	0.0427	0.0418	0.0409	0.0401	0.0392	0.0384	0.0375	0.0367
−1.6	0.0548	0.0537	0.0526	0.0516	0.0505	0.0495	0.0485	0.0475	0.0465	0.0455
−1.5	0.0668	0.0655	0.0643	0.0630	0.0618	0.0606	0.0594	0.0582	0.0571	0.0559
−1.4	0.0808	0.0793	0.0778	0.0764	0.0749	0.0735	0.0721	0.0708	0.0694	0.0681
−1.3	0.0968	0.0951	0.0934	0.0918	0.0901	0.0885	0.0869	0.0853	0.0838	0.0823
−1.2	0.1151	0.1131	0.1112	0.1093	0.1075	0.1056	0.1038	0.1020	0.1003	0.0985
−1.1	0.1357	0.1335	0.1314	0.1292	0.1271	0.1251	0.1230	0.1210	0.1190	0.1170
−1.0	0.1587	0.1562	0.1539	0.1515	0.1492	0.1469	0.1446	0.1423	0.1401	0.1379
−0.9	0.1841	0.1814	0.1788	0.1762	0.1736	0.1711	0.1685	0.1660	0.1635	0.1611
−0.8	0.2119	0.2090	0.2061	0.2033	0.2005	0.1977	0.1949	0.1922	0.1894	0.1867
−0.7	0.2420	0.2389	0.2358	0.2327	0.2296	0.2266	0.2236	0.2206	0.2177	0.2148
−0.6	0.2743	0.2709	0.2676	0.2643	0.2611	0.2578	0.2546	0.2514	0.2483	0.2451
−0.5	0.3085	0.3050	0.3015	0.2981	0.2946	0.2912	0.2877	0.2843	0.2810	0.2776
−0.4	0.3446	0.3409	0.3372	0.3336	0.3300	0.3264	0.3228	0.3192	0.3156	0.3121
−0.3	0.3821	0.3783	0.3745	0.3707	0.3669	0.3632	0.3594	0.3557	0.3520	0.3483
−0.2	0.4207	0.4168	0.4129	0.4090	0.4052	0.4013	0.3974	0.3936	0.3897	0.3859
−0.1	0.4602	0.4562	0.4522	0.4483	0.4443	0.4404	0.4364	0.4325	0.4286	0.4247
−0.0	0.5000	0.4960	0.4920	0.4880	0.4840	0.4801	0.4761	0.4721	0.4681	0.4641

Area under the Normal Curve [2]

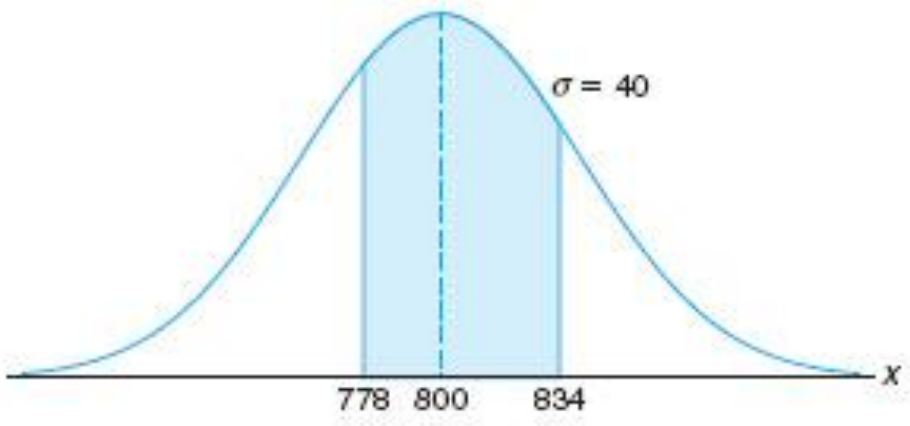
Table A.3 (continued) Areas under the Normal Curve

z	.00	.01	.02	.03	.04	.05	.06	.07	.08	.09
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.7	0.7580	0.7611	0.7642	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7852
0.8	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.9	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389
1.0	0.8413	0.8438	0.8461	0.8485	0.8508	0.8531	0.8554	0.8577	0.8599	0.8621
1.1	0.8643	0.8665	0.8686	0.8708	0.8729	0.8749	0.8770	0.8790	0.8810	0.8830
1.2	0.8849	0.8869	0.8888	0.8907	0.8925	0.8944	0.8962	0.8980	0.8997	0.9015
1.3	0.9032	0.9049	0.9066	0.9082	0.9099	0.9115	0.9131	0.9147	0.9162	0.9177
1.4	0.9192	0.9207	0.9222	0.9236	0.9251	0.9265	0.9279	0.9292	0.9306	0.9319
1.5	0.9332	0.9345	0.9357	0.9370	0.9382	0.9394	0.9406	0.9418	0.9429	0.9441
1.6	0.9452	0.9463	0.9474	0.9484	0.9495	0.9505	0.9515	0.9525	0.9535	0.9545
1.7	0.9554	0.9564	0.9573	0.9582	0.9591	0.9599	0.9608	0.9616	0.9625	0.9633
1.8	0.9641	0.9649	0.9656	0.9664	0.9671	0.9678	0.9686	0.9693	0.9699	0.9706
1.9	0.9713	0.9719	0.9726	0.9732	0.9738	0.9744	0.9750	0.9756	0.9761	0.9767
2.0	0.9772	0.9778	0.9783	0.9788	0.9793	0.9798	0.9803	0.9808	0.9812	0.9817
2.1	0.9821	0.9826	0.9830	0.9834	0.9838	0.9842	0.9846	0.9850	0.9854	0.9857
2.2	0.9861	0.9864	0.9868	0.9871	0.9875	0.9878	0.9881	0.9884	0.9887	0.9890
2.3	0.9893	0.9896	0.9898	0.9901	0.9904	0.9906	0.9909	0.9911	0.9913	0.9916
2.4	0.9918	0.9920	0.9922	0.9925	0.9927	0.9929	0.9931	0.9932	0.9934	0.9936
2.5	0.9938	0.9940	0.9941	0.9943	0.9945	0.9946	0.9948	0.9949	0.9951	0.9952
2.6	0.9953	0.9955	0.9956	0.9957	0.9959	0.9960	0.9961	0.9962	0.9963	0.9964
2.7	0.9965	0.9966	0.9967	0.9968	0.9969	0.9970	0.9971	0.9972	0.9973	0.9974
2.8	0.9974	0.9975	0.9976	0.9977	0.9977	0.9978	0.9979	0.9979	0.9980	0.9981
2.9	0.9981	0.9982	0.9982	0.9983	0.9984	0.9984	0.9985	0.9985	0.9986	0.9986
3.0	0.9987	0.9987	0.9987	0.9988	0.9988	0.9989	0.9989	0.9989	0.9990	0.9990
3.1	0.9990	0.9991	0.9991	0.9991	0.9992	0.9992	0.9992	0.9992	0.9993	0.9993
3.2	0.9993	0.9993	0.9994	0.9994	0.9994	0.9994	0.9994	0.9995	0.9995	0.9995
3.3	0.9995	0.9995	0.9995	0.9996	0.9996	0.9996	0.9996	0.9996	0.9996	0.9997
3.4	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9998

Example 6.8: An electrical firm manufactures light bulbs that have a life, before burn-out, that is normally distributed with **mean equal to 800 hours** and a **standard deviation of 40 hours**. Find the probability that a bulb burns between 778 and 834 hours. Also implement it in Python.

Solution:

μ = 800 and σ = 40



$$Z_1 = \frac{778 - 800}{40} = \mathbf{-0.55}$$

$$Z_2 = \frac{834 - 800}{40} = \mathbf{0.85}$$

P(778 ≤ X ≤ 834) = P(-0.55 ≤ Z ≤ 0.85)

= P(Z ≤ 0.85) - P(Z ≤ -0.55)

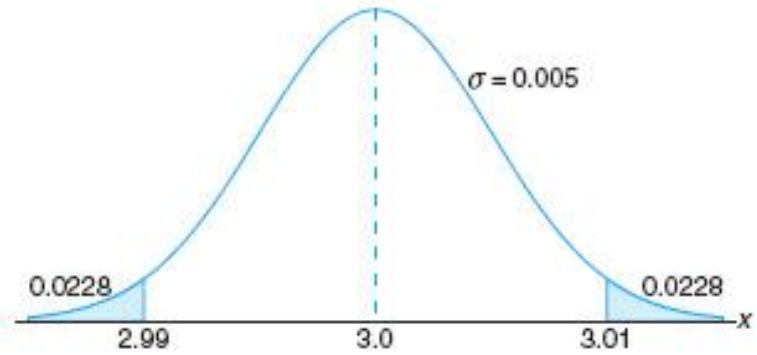
= 0.8023 – 0.2912

= 0.5111

```
from scipy.stats import norm

# Given data
mean = 800
std_dev = 40
lower_bound = 778
upper_bound = 834

# Calculate z-scores
z_lower = (lower_bound - mean) / std_dev
z_upper = (upper_bound - mean) / std_dev

# Find the probability using the cumulative distribution function (CDF)
probability = norm.cdf(z_upper) - norm.cdf(z_lower)

# Print the result
print(f"The probability that a bulb burns between {lower_bound} and {upper_bound} hours is: {probability:.4f}")
```

Example 6.9: In an industrial process, the diameter of a ball bearing is an important measurement. The buyer sets specifications for the diameter to be **2.99** to **3.01**. The implication is that no part falling outside these specifications will be accepted. It is known that in the process the diameter of a ball bearing has a normal distribution with mean $\mu = 3.0$ and standard deviation $\sigma = 0.005$. On average, how many manufactured ball bearings will be scrapped? Also implement it in Python.

Solution:



$\mu = 3.00$ and $\sigma = 0.005$

$$Z_1 = \frac{2.99 - 3.00}{0.005} = -2.0$$

$$Z_2 = \frac{3.01 - 3.0}{0.005} = +2.0$$

$$P(2.99 \le X \le 3.01) = P(-2 \le Z \le 2)$$
$$= P(Z \le 2) - P(Z \le -2)$$
$$= 0.9772 - 0.0228$$
$$= 0.9544$$

Required probability = $1 - P(2.99 \le X \le 3.01)$ = **0.0456.**

OR

From Table A.3, $P(Z < -2.0) = 0.0228$. Due to symmetry of the normal distribution, we find that $P(Z < -2.0) + P(Z > 2.0) = 2(0.0228) =$ **0.0456.**

```
from scipy.stats import norm

# Given data
mean_diameter = 3.0
std_dev_diameter = 0.005
lower_specification = 2.99
upper_specification = 3.01

# Calculate the z-scores for the lower and upper specifications
z_lower = (lower_specification - mean_diameter) / std_dev_diameter
z_upper = (upper_specification - mean_diameter) / std_dev_diameter

# Calculate the probability of scrapping for a single ball bearing
prob_scrapped_single = norm.cdf(z_lower) + (1 - norm.cdf(z_upper))
```

```
# Assuming a total production of 1 for simplicity, calculate the average number of scrapped ball bearings
average_scrapped = 1 * prob_scrapped_single

# Print the result
print(f"On average, {average_scrapped:.2f} manufactured ball bearings will be scrapped.")
```

Table A.3 Areas under the Normal Curve

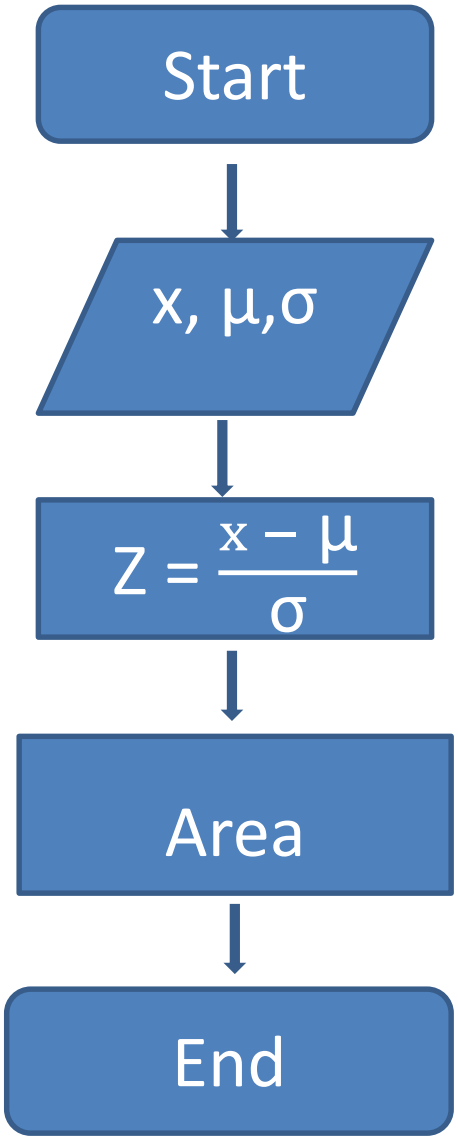
z	.00	.01	.02	.03	.04	.05	.06	.07	.08	.09
−3.4	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0003	0.0002
−3.3	0.0005	0.0005	0.0005	0.0004	0.0004	0.0004	0.0004	0.0004	0.0004	0.0003
−3.2	0.0007	0.0007	0.0006	0.0006	0.0006	0.0006	0.0006	0.0005	0.0005	0.0005
−3.1	0.0010	0.0009	0.0009	0.0009	0.0008	0.0008	0.0008	0.0008	0.0007	0.0007
−3.0	0.0013	0.0013	0.0013	0.0012	0.0012	0.0011	0.0011	0.0011	0.0010	0.0010
−2.9	0.0019	0.0018	0.0018	0.0017	0.0016	0.0016	0.0015	0.0015	0.0014	0.0014
−2.8	0.0026	0.0025	0.0024	0.0023	0.0023	0.0022	0.0021	0.0021	0.0020	0.0019
−2.7	0.0035	0.0034	0.0033	0.0032	0.0031	0.0030	0.0029	0.0028	0.0027	0.0026
−2.6	0.0047	0.0045	0.0044	0.0043	0.0041	0.0040	0.0039	0.0038	0.0037	0.0036
−2.5	0.0062	0.0060	0.0059	0.0057	0.0055	0.0054	0.0052	0.0051	0.0049	0.0048
−2.4	0.0082	0.0080	0.0078	0.0075	0.0073	0.0071	0.0069	0.0068	0.0066	0.0064
−2.3	0.0107	0.0104	0.0102	0.0099	0.0096	0.0094	0.0091	0.0089	0.0087	0.0084
−2.2	0.0139	0.0136	0.0132	0.0129	0.0125	0.0122	0.0119	0.0116	0.0113	0.0110
−2.1	0.0179	0.0174	0.0170	0.0166	0.0162	0.0158	0.0154	0.0150	0.0146	0.0143
−2.0	0.0228	0.0222	0.0217	0.0212	0.0207	0.0202	0.0197	0.0192	0.0188	0.0183
−1.9	0.0287	0.0281	0.0274	0.0268	0.0262	0.0256	0.0250	0.0244	0.0239	0.0233
−1.8	0.0359	0.0351	0.0344	0.0336	0.0329	0.0322	0.0314	0.0307	0.0301	0.0294
−1.7	0.0446	0.0436	0.0427	0.0418	0.0409	0.0401	0.0392	0.0384	0.0375	0.0367
−1.6	0.0548	0.0537	0.0526	0.0516	0.0505	0.0495	0.0485	0.0475	0.0465	0.0455
−1.5	0.0668	0.0655	0.0643	0.0630	0.0618	0.0606	0.0594	0.0582	0.0571	0.0559
−1.4	0.0808	0.0793	0.0778	0.0764	0.0749	0.0735	0.0721	0.0708	0.0694	0.0681
−1.3	0.0968	0.0951	0.0934	0.0918	0.0901	0.0885	0.0869	0.0853	0.0838	0.0823
−1.2	0.1151	0.1131	0.1112	0.1093	0.1075	0.1056	0.1038	0.1020	0.1003	0.0985
−1.1	0.1357	0.1335	0.1314	0.1292	0.1271	0.1251	0.1230	0.1210	0.1190	0.1170
−1.0	0.1587	0.1562	0.1539	0.1515	0.1492	0.1469	0.1446	0.1423	0.1401	0.1379
−0.9	0.1841	0.1814	0.1788	0.1762	0.1736	0.1711	0.1685	0.1660	0.1635	0.1611
−0.8	0.2119	0.2090	0.2061	0.2033	0.2005	0.1977	0.1949	0.1922	0.1894	0.1867
−0.7	0.2420	0.2389	0.2358	0.2327	0.2296	0.2266	0.2236	0.2206	0.2177	0.2148
−0.6	0.2743	0.2709	0.2676	0.2643	0.2611	0.2578	0.2546	0.2514	0.2483	0.2451
−0.5	0.3085	0.3050	0.3015	0.2981	0.2946	0.2912	0.2877	0.2843	0.2810	0.2776
−0.4	0.3446	0.3409	0.3372	0.3336	0.3300	0.3264	0.3228	0.3192	0.3156	0.3121
−0.3	0.3821	0.3783	0.3745	0.3707	0.3669	0.3632	0.3594	0.3557	0.3520	0.3483
−0.2	0.4207	0.4168	0.4129	0.4090	0.4052	0.4013	0.3974	0.3936	0.3897	0.3859
−0.1	0.4602	0.4562	0.4522	0.4483	0.4443	0.4404	0.4364	0.4325	0.4286	0.4247
−0.0	0.5000	0.4960	0.4920	0.4880	0.4840	0.4801	0.4761	0.4721	0.4681	0.4641

Table A.3 (continued) Areas under the Normal Curve

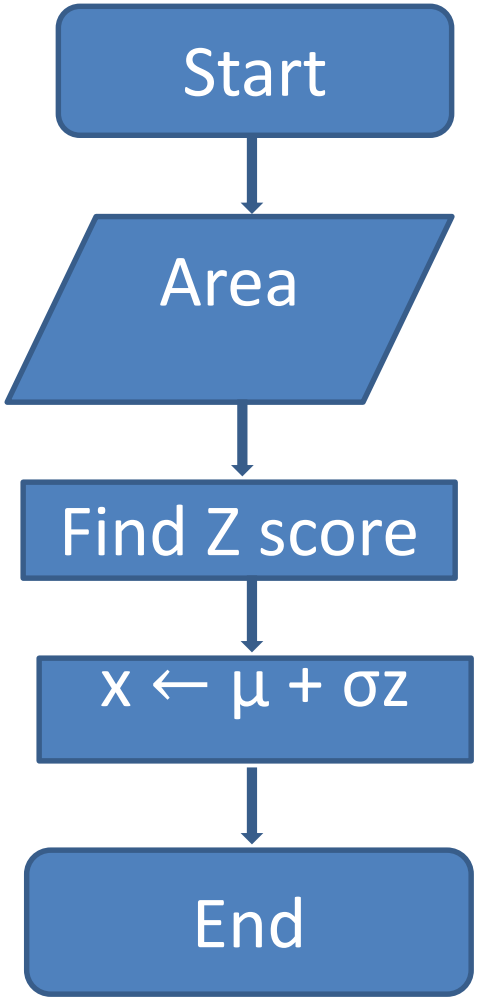
z	.00	.01	.02	.03	.04	.05	.06	.07	.08	.09
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.7	0.7580	0.7611	0.7642	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7852
0.8	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.9	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389
1.0	0.8413	0.8438	0.8461	0.8485	0.8508	0.8531	0.8554	0.8577	0.8599	0.8621
1.1	0.8643	0.8665	0.8686	0.8708	0.8729	0.8749	0.8770	0.8790	0.8810	0.8830
1.2	0.8849	0.8869	0.8888	0.8907	0.8925	0.8944	0.8962	0.8980	0.8997	0.9015
1.3	0.9032	0.9049	0.9066	0.9082	0.9099	0.9115	0.9131	0.9147	0.9162	0.9177
1.4	0.9192	0.9207	0.9222	0.9236	0.9251	0.9265	0.9279	0.9292	0.9306	0.9319
1.5	0.9332	0.9345	0.9357	0.9370	0.9382	0.9394	0.9406	0.9418	0.9429	0.9441
1.6	0.9452	0.9463	0.9474	0.9484	0.9495	0.9505	0.9515	0.9525	0.9535	0.9545
1.7	0.9554	0.9564	0.9573	0.9582	0.9591	0.9599	0.9608	0.9616	0.9625	0.9633
1.8	0.9641	0.9649	0.9656	0.9664	0.9671	0.9678	0.9686	0.9693	0.9699	0.9706
1.9	0.9713	0.9719	0.9726	0.9732	0.9738	0.9744	0.9750	0.9756	0.9761	0.9767
2.0	0.9772	0.9778	0.9783	0.9788	0.9793	0.9798	0.9803	0.9808	0.9812	0.9817
2.1	0.9821	0.9826	0.9830	0.9834	0.9838	0.9842	0.9846	0.9850	0.9854	0.9857
2.2	0.9861	0.9864	0.9868	0.9871	0.9875	0.9878	0.9881	0.9884	0.9887	0.9890
2.3	0.9893	0.9896	0.9898	0.9901	0.9904	0.9906	0.9909	0.9911	0.9913	0.9916
2.4	0.9918	0.9920	0.9922	0.9925	0.9927	0.9929	0.9931	0.9932	0.9934	0.9936
2.5	0.9938	0.9940	0.9941	0.9943	0.9945	0.9946	0.9948	0.9949	0.9951	0.9952
2.6	0.9953	0.9955	0.9956	0.9957	0.9959	0.9960	0.9961	0.9962	0.9963	0.9964
2.7	0.9965	0.9966	0.9967	0.9968	0.9969	0.9970	0.9971	0.9972	0.9973	0.9974
2.8	0.9974	0.9975	0.9976	0.9977	0.9977	0.9978	0.9979	0.9979	0.9980	0.9981
2.9	0.9981	0.9982	0.9982	0.9983	0.9984	0.9984	0.9985	0.9985	0.9986	0.9986
3.0	0.9987	0.9987	0.9987	0.9988	0.9988	0.9989	0.9989	0.9989	0.9990	0.9990
3.1	0.9990	0.9991	0.9991	0.9991	0.9992	0.9992	0.9992	0.9992	0.9993	0.9993
3.2	0.9993	0.9993	0.9994	0.9994	0.9994	0.9994	0.9994	0.9995	0.9995	0.9995
3.3	0.9995	0.9995	0.9995	0.9996	0.9996	0.9996	0.9996	0.9996	0.9996	0.9997
3.4	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9998

Example 6.10: Gauges are used to reject all components for which a certain dimension is not within the specification $1.50 \mp d$. It is known that this measurement is normally distributed with mean **1.50** and standard deviation **0.2**. Determine the value **d** such that the specifications "cover" **95%** of the measurements. Also implement it in Python.

Flowchart for Computing Probability



Flowchart for Computing x (Inverse Problem)



Solution: μ = 1.5 and **σ** = 0.2

From Table A.3 we know that

P(Z < −1.96) = 0.0250 $\because \frac{0.05}{2} = 0.0250$

P(Z > 1.96) = 0.0250 (Due to symmetry of normal curve)

P(Z < 1.96) = 0.9750 $\because$ 1 - 0.0250 = **0.9750**

$\because$ **P(Z > 1.96) = 0.0250**

Or P(Z < 1.96) = 0.9750

$$z = \frac{x - \mu}{\sigma} \Longrightarrow x = \mu + \sigma z$$

x = μ + σz

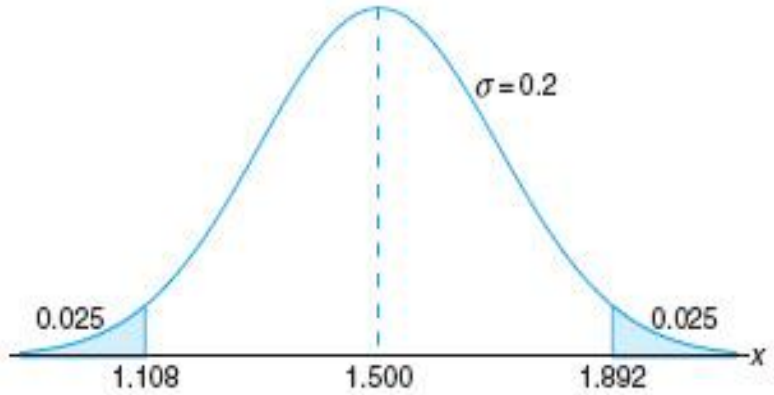
Let x = 1.50 + d

⇒ 1.50 + d **= 1.50 + 0.2(1.96)**

⇒ d **= 0.392**

$x_1 = \mu - d =$ 1.50 **- 0.392 =** 1.108

$x_2 = \mu + d =$ 1.50 **+ 0.392 =** 1.892



OR

$P(Z < -1.96) = 0.0250$

$x = \mu + \sigma z$

Let $x =$ $1.50 - d$

$\Rightarrow 1.50 - d$ **$= 1.50 + 0.2(-1.96)$**

$\Rightarrow d$ **$= 0.392$**

$x_1 = \mu - d$

$x_1 =$ 1.50 **$- 0.392 =$** 1.108

$x_2 = \mu + d$

$x_2 =$ 1.50 **$+ 0.392 =$** 1.892

```
import scipy.stats as stats

# Given data
mean = 1.50
std_deviation = 0.2
coverage = 0.95

# Find the z-score corresponding to the desired coverage
# Since it's two-tailed, we want the middle 95%, so the remaining is
# (100 - 95) / 2 = 2.5% on each tail
area_in_tail = (1 - coverage) / 2

# Find the z-score for the area in the tail
z_score = stats.norm.ppf(1 - area_in_tail)

# Calculate d
d = z_score * std_deviation

# Print the result
print(f"The value of d such that 95% of the measurements are within
1.50 +- d is {d:.3f}")
```

Example 6.11: A certain machine makes electrical resistors having a mean resistance of **40** ohms and a standard deviation of **2** ohms. Assuming that the resistance follows a normal distribution and can be measured to any degree of accuracy, what percentage of resistors will have a resistance exceeding **43** ohms? Also implement it in Python.

Solution:

$P(X > 43) = ?$

$\mu = 40$ and $\sigma = 2$

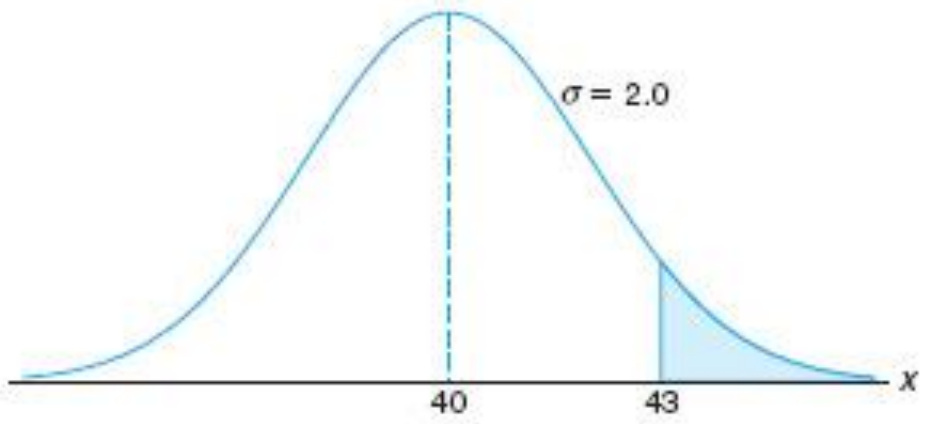
$$z = \frac{43 - 40}{2} = 1.5$$

$P(X > 43) = P(Z > 1.5)$

$= 1 - P(Z < 1.5)$

$= 1 - 0.9332$

$= 0.0668$ Or **6.68 %**



```
import scipy.stats as stats

# Given data
mean_resistance = 40
std_dev_resistance = 2
resistance_threshold = 43

# Calculate the z-score
z_score = (resistance_threshold - mean_resistance) /
std_dev_resistance

# Calculate the percentage of resistors exceeding 43 ohms
percentage_exceeding = (1 - stats.norm.cdf(z_score)) * 100

# Print the result
print(f"The percentage of resistors with resistance exceeding 43
ohms is: {percentage_exceeding:.2f}%")
```

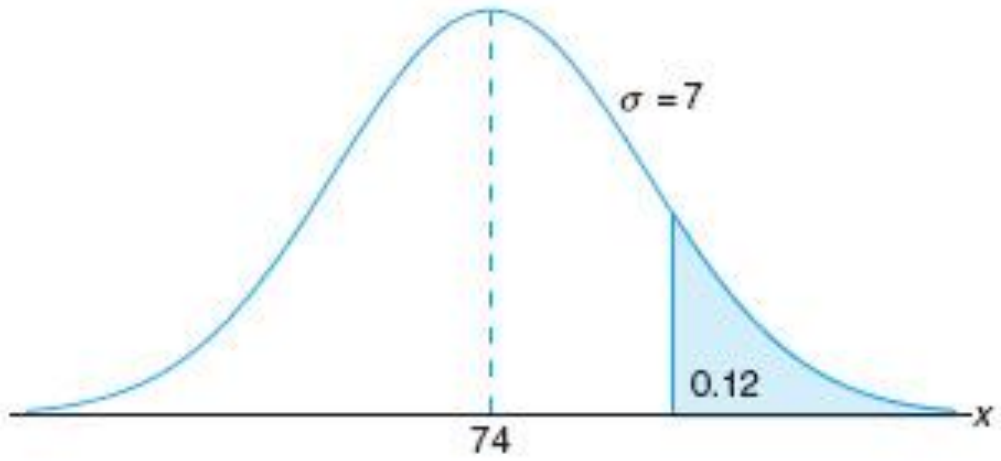
Example 6.13: The average grade for an exam is **74**, and the standard deviation is **7**. If **12%** of the class is given **As**, and the grades are curved to follow a normal distribution, what is the **lowest possible A** and the **highest possible A^{-1}**? Also implement it in Python.

Solution:

$\mu = 74$ and $\sigma = 7$

$P(Z < z) = 0.88$

$P(Z < 1.18) = 0.88$



$x = \sigma z + \mu$

$x = (7)(1.18) + 74 = 82.26.$

Therefore, the lowest ***A* is 83** and the highest **A^{-1} is 82**.

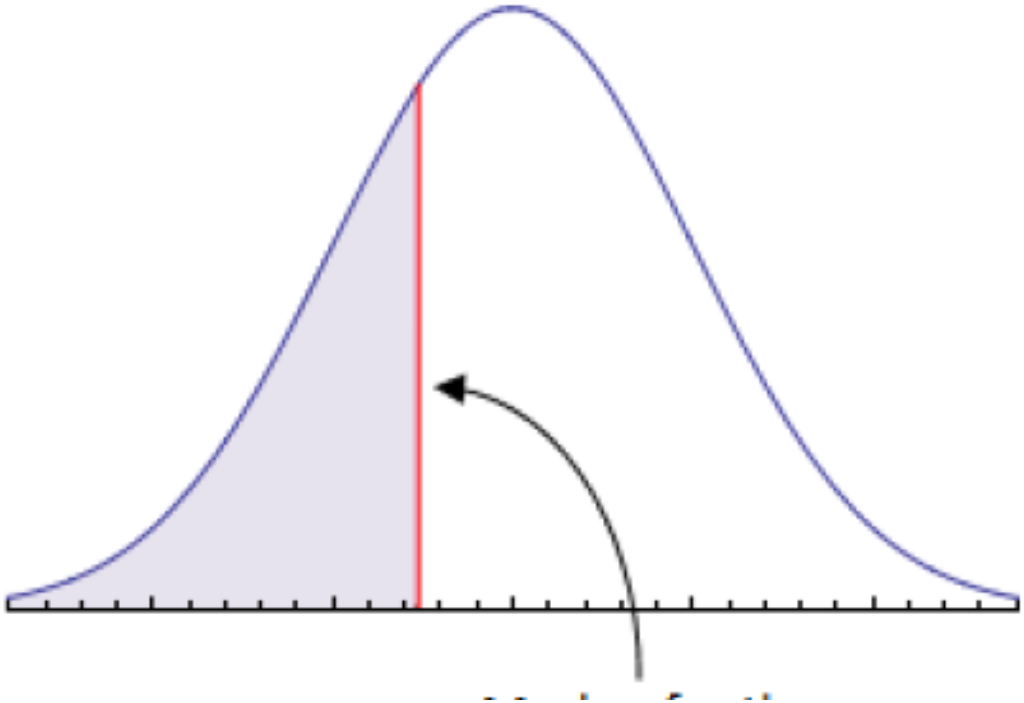
```
import scipy.stats as stats
import math
# Given data
mean_grade = 74
std_deviation = 7

# Find the z-score corresponding to the 88th percentile for A
percentile_A = 88
z_score_A = stats.norm.ppf(percentile_A / 100)

# Calculate the actual grade cutoff for A
grade_cutoff_A = math.floor(mean_grade + z_score_A * std_deviation)

# Calculate the actual grade cutoff for A-
grade_cutoff_A_minus = math.ceil(mean_grade + z_score_A *
std_deviation)
# Print the results
print(f"The lowest possible A is {grade_cutoff_A:.2f}")
print(f"The highest possible A- is {grade_cutoff_A_minus:.2f}")
```

Percentiles

A **percentile** is a marker on a normal curve such that the **marker is greater than or equal** to that percentage of results.

For example, suppose you are at the **30th percentile** for **how fast you type**. This means that you can type faster than **30% of all people**. The percentile can also be thought of as the **percent of area** to the **left of its marker**.

The shaded area to the left of the marker represents 30% of the normal curve.



Marker for the 30th percentile

Example Suppose the **mean length** of the hair of 10th grade girls is **10 inches** with a **standard deviation of 4** inches. At **what percentile** for hair length is a 10th grade girl if her hair is **17 inches long**? Also implement it in Python.

Solution

$\mu = 10$ and $\sigma = 4$

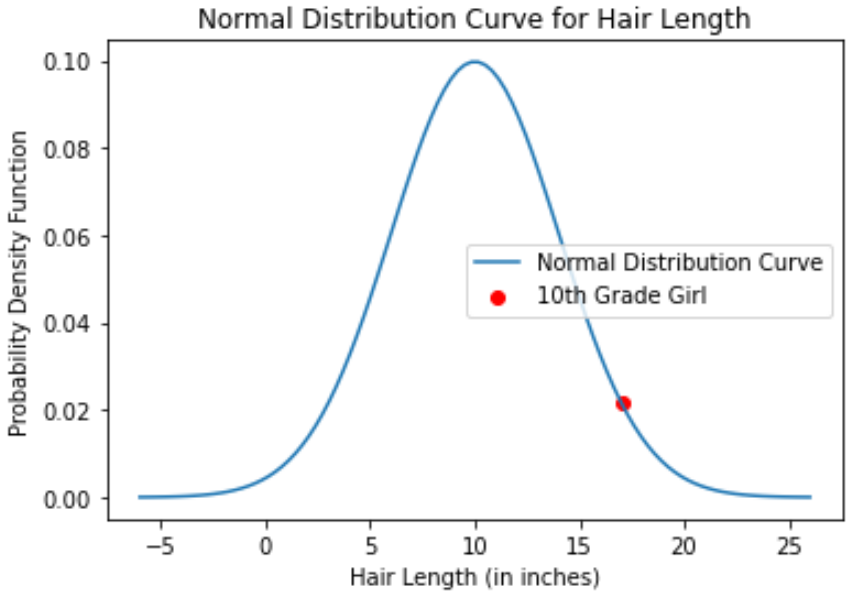
$x = 17$

$P(X < 17) = ?$

$$z = \frac{17 - 10}{4} = 1.75$$

$P(X < 17) = P(Z < 1.75)$

$= 0.9599.$

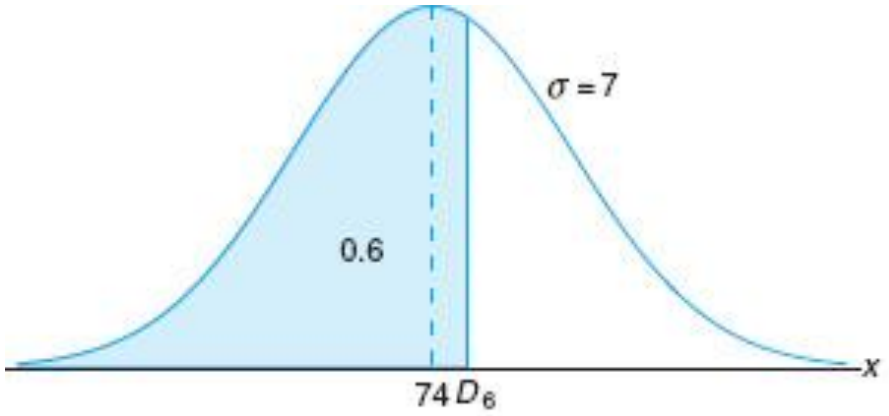


This tells us the girl is at about the **96th percentile** for hair length. In other words, this girl's hair is longer than 96% of all 10^{th} grade girls

```
import scipy.stats as stats

# Given data
mean_hair_length = 10
std_dev_hair_length = 4
girl_hair_length = 17

# Calculate the z-score
z_score = (girl_hair_length - mean_hair_length) / std_dev_hair_length

# Calculate the percentile using the CDF
percentile = stats.norm.cdf(z_score) * 100

# Print the result
print(f"A 10th-grade girl with a hair length of 17 inches is at the
{percentile:.2f}th percentile.")
```

Example 6.14: Refer to Example 6.13 and find the sixth decile.

Solution: The sixth decile, written D_6, is the x value that leaves **60%** of the area to the left, as shown in Figure 6.21.



$P(Z < z) = 0.6$

$P(Z < 0.25) = 0.6$

$z = 0.25.$

$D_6 = \sigma z + \mu$

$D_6 = (7)(0.25)+74 = 75.75.$

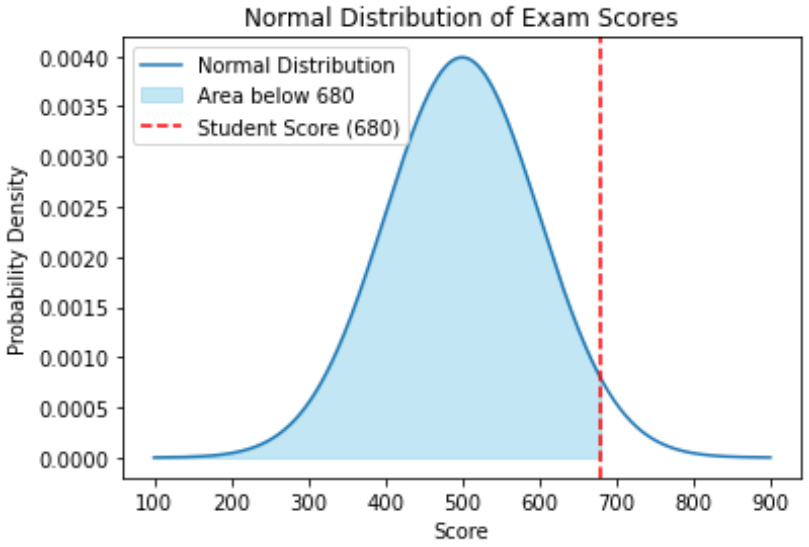
Hence, $D_6 = 75.75.$

That is, **60% of the grades are 75 or less**

Example: The scores on a national achievement exam are normally distributed with a **mean** of **500** and a **standard deviation** of **100**. If a student is selected at random, find the probability that the student scored **below 680**. Also implement it in Python.

Solution:

$\mu = 500$ and $\sigma = 100$



Find the z value for 680:

$$z = \frac{680 - 500}{100} = 1.8$$

$P(X < 680) = P(Z < 1.8) = .9641$

```
import scipy.stats as stats

# Given data
mean_score = 500
std_dev_score = 100
student_score = 680

# Calculate the probability using the CDF
probability_below_680 = stats.norm.cdf(student_score, mean_score,
std_dev_score)

# Print the result
print(f"The probability that a randomly selected student scored below
680 is: {probability_below_680:.4f}")
```

Normal Distribution Applications in Machine Learning

Application	Use of Normal Distribution
Linear Regression	Assumption of normally distributed errors for inference and prediction
Gaussian Naive Bayes	Assumes normally distributed features in classification problems
Principal Component Analysis (PCA)	Data assumed to follow multivariate normal distribution for dimensionality reduction
Gaussian Mixture Models	Clustering based on mixture of multiple Gaussian distributions
Kalman Filters	Used for tracking and forecasting in time-series models with Gaussian noise
Neural Network Initialization	Weights initialized from a normal distribution for training stability
Logistic Regression	Approximation of residuals using normal distribution in large datasets
Anomaly Detection	Modeling normal behavior using normal distribution to detect anomalies