Data Science in Inventory Management: Real case in managing a warehouse

Think about the problem

Alright, let's think about **2 shopping cases** that you have experienced when you go shopping. Someday you go to the **shoes shop** and **decided to buy a new pair of shoes**, and unfortunately, you got **problem**:

1. You picked up a good model, but they don't have any left that fit your foot size

2. They offered another model which fit your foot size, but you don't like it

As a shop manager, they are not happy too. They lost a customer — it's you. And **if this is the first time you visited the store, will you come back? Of course... "maybe"**. But **if this is the second time you got the same problem with this store, will you come back?** It's absolutely a shop manager's nightmare.

Find a way

So how to **avoid this problem if you are playing a role as shop manager?**

Of course, we cannot because we cannot control how many customer has the same style and the same size will visit your store at the same day. It depends on our luck!

Then we close the case, give up on this problem and try to explain to our big boss like "okay I'm sorry, it's my bad luck"?

Absolutely no! We have enough **statistics tool to solve the problem**. But remember, **we cannot make sure every customer will have their shoes BUT we can increase the chance that customer will find their favorite shoes at our store using Data Science** (so don't mess with our luck).

Check the historical data

Before making plan for future, we need to examine the history in the past — remember this! So first step, we need to meet the **Sales Department to get the sales history of few years in the past**. And this is the data we got from Sales Department (https://s3.us-west-2.amazonaws.com/public.gamelab.fun/dataset/Al-Bundy_raw-data.csv)

InvoiceNo	Date	Country	ProductID	Shop	Gender	Size (US)	Size (Europe)	Size (UK)	UnitPrice	Discount	Year	Month	SalePrice
52389	1/1/2014	United Kingdom	2152	UK2	Male	11	44	10.5	159	0%	2014	1	159
52390	1/1/2014	United States	2230	US15	Male	11.5	44-45	11	199	20%	2014	1	159.2
52391	1/1/2014	Canada	2160	CAN7	Male	9.5	42-43	9	149	20%	2014	1	119.2
52392	1/1/2014	United States	2234	US6	Female	9.5	40	7.5	159	0%	2014	1	159
52393	1/1/2014	United Kingdom	2222	UK4	Female	9	39-40	7	159	0%	2014	1	159
52394	1/1/2014	United States	2173	US15	Male	10.5	43-44	10	159	0%	2014	1	159
52395	1/2/2014	Germany	2200	GER2	Female	9	39-40	7	179	0%	2014	1	179
52396	1/2/2014	Canada	2238	CAN5	Male	10	43	9.5	169	0%	2014	1	169
52397	1/2/2014	United States	2191	US13	Male	10.5	43-44	10	139	0%	2014	1	139
52398	1/2/2014	United Kingdom	2237	UK1	Female	9	39-40	7	149	0%	2014	1	149
52399	1/2/2014	United States	2197	US1	Male	10	43	9.5	129	0%	2014	1	129
52399	1/2/2014	United States	2213	US11	Female	9.5	40	7.5	169	10%	2014	1	152.1
52399	1/2/2014	United States	2206	US2	Female	9.5	40	7.5	139	0%	2014	1	139
52400	1/2/2014	United States	2152	US15	Male	8	41	7.5	139	0%	2014	1	139
52401	1/3/2014	Germany	2235	GER1	Male	10.5	43-44	10	169	50%	2014	1	84.5
52401	1/3/2014	Germany	2197	GER1	Female	8.5	39	6.5	179	20%	2014	1	143.2
52402	1/3/2014	Canada	2240	CAN6	Male	9.5	42-43	9	199	30%	2014	1	139.3
52403	1/3/2014	United States	2221	US7	Male	11	44	10.5	149	50%	2014	1	74.5
52404	1/3/2014	United States	2234	US6	Female	9.5	40	7.5	159	0%	2014	1	159
52404	1/3/2014	United States	2197	US1	Male	10	43	9.5	129	0%	2014	1	129
52404	1/3/2014	United States	2213	US11	Female	9.5	40	7.5	169	10%	2014	1	152.1
52405	1/3/2014	United States	2213	US7	Female	8	38-39	6	139	0%	2014	1	139
52406	1/3/2014	United States	2147	US15	Male	9.5	42-43	9	139	0%	2014	1	139
52407	1/4/2014	United States	2224	US13	Male	9	42	8.5	149	10%	2014	1	134.1
52408	1/4/2014	Germany	2206	GER2	Male	8.5	41-42	8	149	20%	2014	1	119.2
52409	1/4/2014	Germany	2157	GER2	Male	12	45	11.5	149	20%	2014	1	119.2
52410	1/4/2014	Canada	2169	CAN3	Female	8	38-39	6	129	0%	2014	1	129
52411	1/4/2014	United States	2191	US13	Male	10.5	43-44	10	139	0%	2014	1	139
52411	1/4/2014	United States	2234	US6	Female	9.5	40	7.5	159	0%	2014	1	159
52411	1/4/2014	United States	2197	US1	Male	10	43	9.5	129	0%	2014	1	129

Examine and Cleaning up the data

Let's examine our fields in the data set:

InvoiceNo, ProductID, Year, Month: they contain only number, but they're not a numerical field, they're categorical field

Date, Country: categorical field. We already have Date field which have the meaning of both Year, and Month fields, so we should keep Date and exclude Year, and Month out of our data set

Shop: absolutely a categorical field grouped by Country ID (US, UK, CAN, GER)

Size (US), Size (Europe), Size (UK): categorical field. But let's think twice! 3 fields has the same meaning because we can convert Size (US) to Size (Europe) and Size (UK) and of course, vice versa using this conversion chart. So in this case, we should keep only 1 field as a representative. I will pick Size (US) because it looks simpler than Europe one and has less floating number than UK. Please remember that the size of Male and Female are different, so do not using Size field without Gender field or you will make a serious mistake!
Gender: another basic categorical field

UnitPrice, Discount, SalePrice: numerical field. And before doing any calculation, I suggest we should convert the Discount field which contain percentage values into float for easier calculation. If you are a clever one, you can see the hidden equation between these field which is

SalePrice = UnitPrice * (1 — Discount)

...so we should keep only SalePrice for our data set
After doing this step, we will have a much more simpler data set:

InvoiceNo	Date	Country	ProductID	Shop	Gender	Size (US)	Size (Europe)	Size (UK)	UnitPrice	Discount	Year	Month	SalePrice
52389	1/1/2014	United Kingdom	2152	UK2	Male	11	44	10.5	159	0%	2014	1	159
52390	1/1/2014	United States	2230	US15	Male	11.5	44-45	11	199	20%	2014	1	159.2
52391	1/1/2014	Canada	2160	CAN7	Male	9.5	42-43	9	149	20%	2014	1	119.2
52392	1/1/2014	United States	2234	US6	Female	9.5	40	7.5	159	0%	2014	1	159
52393	1/1/2014	United Kingdom	2222	UK4	Female	9	39-40	7	159	0%	2014	1	159
52394	1/1/2014	United States	2173	US15	Male	10.5	43-44	10	159	0%	2014	1	159
52395	1/2/2014	Germany	2200	GER2	Female	9	39-40	7	179	0%	2014	1	179
52396	1/2/2014	Canada	2238	CAN5	Male	10	43	9.5	169	0%	2014	1	169
52397	1/2/2014	United States	2191	US13	Male	10.5	43-44	10	139	0%	2014	1	139
52398	1/2/2014	United Kingdom	2237	UK1	Female	9	39-40	7	149	0%	2014	1	149
52399	1/2/2014	United States	2197	US1	Male	10	43	9.5	129	0%	2014	1	129
52399	1/2/2014	United States	2213	US11	Female	9.5	40	7.5	169	10%	2014	1	152.1

Date	Country	Shop	Gender	Size (US)	SalePrice
1/1/2014	United Kingdom	UK2	Male	11	159
1/1/2014	United States	US15	Male	11.5	159.2
1/1/2014	Canada	CAN7	Male	9.5	119.2
1/1/2014	United States	US6	Female	9.5	159
1/1/2014	United Kingdom	UK4	Female	9	159
1/1/2014	United States	US15	Male	10.5	159
1/2/2014	Germany	GER2	Female	9	179
1/2/2014	Canada	CAN5	Male	10	169
1/2/2014	United States	US13	Male	10.5	139
1/2/2014	United Kingdom	UK1	Female	9	149
1/2/2014	United States	US1	Male	10	129
1/2/2014	United States	US11	Female	9.5	152.1

Analyze the data

The data set is ready to be analyzed! Now we have to **count how many items sold in previous period (could be year, or month…)**. In data analysis, counting the appearance of single object is called “frequency”. To **count the frequency, in this case, we will segment the data set by:**

- **Country**
- **Size**
- **Gender** (as I mentioned before, Size could not be used without Gender, so we have to include the Gender too)

Our crosstable will have 2 dimensions only but we already have 3 dimensions here so we have to split the table into 2 different one by Gender. This is the result:

Men sizes					
	Country				
Size (US)	Canada	United States	United Kingdom	Germany	Total
6	15	54	6	30	105
6.5	15	45	12	18	90
7	24	39	21	30	114
7.5	45	66	12	48	171
8	51	141	45	117	354
8.5	192	225	87	174	678
9	324	492	183	348	1347
9.5	375	741	225	549	1890
10	237	543	156	411	1347
10.5	243	462	150	453	1308
11	114	213	69	156	552
11.5	75	156	39	129	399
12	51	87	24	78	240
13	12	39	3	33	87
14	21	60	15	30	126
15	27	24	12	48	111
16	0	0	0	0	0
Total	1821	3387	1059	2652	8919

Women sizes					
	Country				
Size (US)	Canada	United States	United Kingdom	Germany	Total
4	0	0	0	0	0
4.5	6	21	15	9	51
5	6	9	9	12	36
5.5	6	42	6	9	63
6	21	33	12	15	81
6.5	51	93	24	84	252
7	93	147	27	156	423
7.5	153	318	87	222	780
8	192	618	168	324	1302
8.5	171	399	129	339	1038
9	213	384	93	264	954
9.5	84	189	57	126	456
10	48	75	21	87	231
10.5	36	87	18	57	198
11.5	12	30	3	15	60
Total	1092	2445	669	1719	5925

Define the problem

1. **The problem**: Based on our observation, **what is the number of shoes (model, size) that are likely to be sold?**

2. **The condition**: We will find the answer of our problem with 95% confidence interval

3. **The plan**:

- Use the last 12 months data set
- Use men shoes data set
- Use US data set

This step, you will think about **why do we use men shoes and US data set only**.

Let's think: if a man visited the store, would a woman come? We cannot answer, because [redacted].

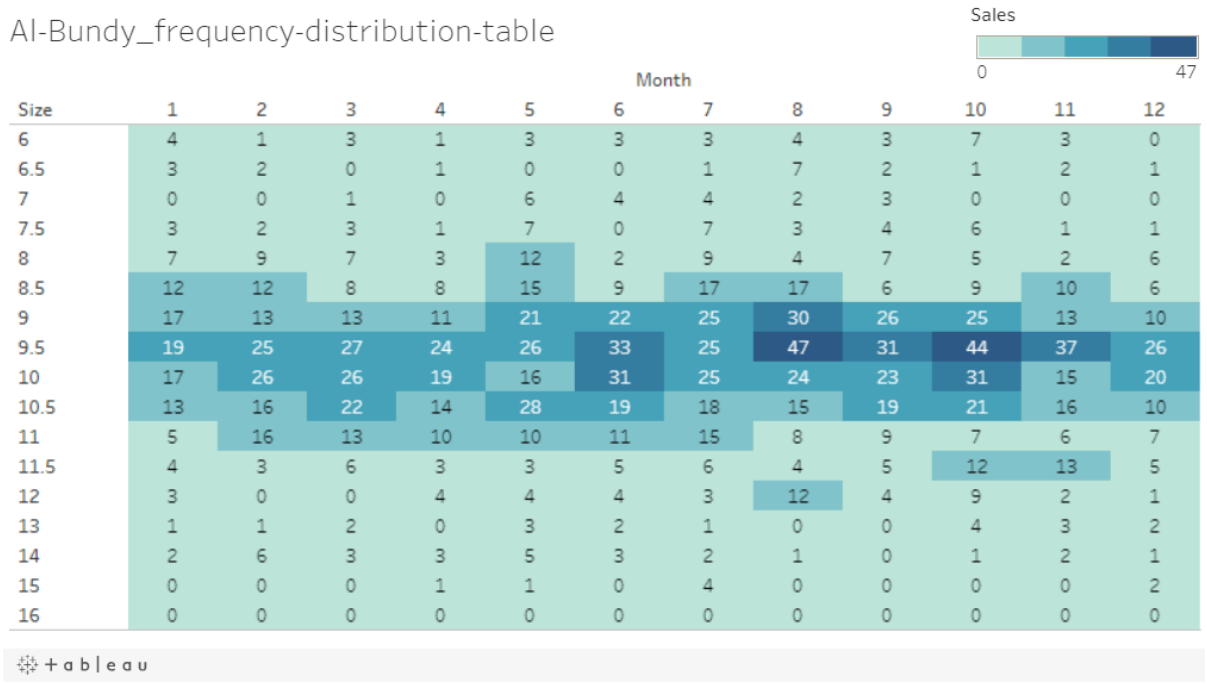
In another hand, for instance, if a kid visited the store, there is a high chance that his parents

come too, because that kid cannot go shopping alone and using money by himself. But for our case, if a man visited the store, he could come alone, or with his bros, or with his girlfriend, or even a group of his friends. There is no connection between the chance of a male visitor and a female visitor. That's how we call it identical.

Of course, the Country is the same with Gender, they are both identical.

Then **why 12 months data set?** Because it's enough the represent a full cycle of a year. [redacted], people likely to change their shoes when winter comes (to buy warmer shoes)...

	Men shoes, United States, 2016											
Size (US) - Month	1	2	3	4	5	6	7	8	9	10	11	12
6	4	1	3	1	3	3	3	4	3	7	3	0
6.5	3	2	0	1	0	0	1	7	2	1	2	1
7	0	0	1	0	6	4	4	2	3	0	0	0
7.5	3	2	3	1	7	0	7	3	4	6	1	1
8	7	9	7	3	12	2	9	4	7	5	2	6
8.5	12	12	8	8	15	9	17	17	6	9	10	6
9	17	13	13	11	21	22	25	30	26	25	13	10
9.5	19	25	27	24	26	33	25	47	31	44	37	26
10	17	26	26	19	16	31	25	24	23	31	15	20
10.5	13	16	22	14	28	19	18	15	19	21	16	10
11	5	16	13	10	10	11	15	8	9	7	6	7
11.5	4	3	6	3	3	5	6	4	5	12	13	5
12	3	0	0	4	4	4	3	12	4	9	2	1
13	1	1	2	0	3	2	1	0	0	4	3	2
14	2	6	3	3	5	3	2	1	0	1	2	1
15	0	0	0	1	1	0	4	0	0	0	0	2
16	0	0	0	0	0	0	0	0	0	0	0	0
Total	110	132	134	103	160	148	165	178	142	182	125	98



Size	1	2	3	4	5	6	7	8	9	10	11	12
6	4	1	3	1	3	3	3	4	3	7	3	0
6.5	3	2	0	1	0	0	1	7	2	1	2	1
7	0	0	1	0	6	4	4	2	3	0	0	0
7.5	3	2	3	1	7	0	7	3	4	6	1	1
8	7	9	7	3	12	2	9	4	7	5	2	6
8.5	12	12	8	8	15	9	17	17	6	9	10	6
9	17	13	13	11	21	22	25	30	26	25	13	10
9.5	19	25	27	24	26	33	25	47	31	44	37	26
10	17	26	26	19	16	31	25	24	23	31	15	20
10.5	13	16	22	14	28	19	18	15	19	21	16	10
11	5	16	13	10	10	11	15	8	9	7	6	7
11.5	4	3	6	3	3	5	6	4	5	12	13	5
12	3	0	0	4	4	4	3	12	4	9	2	1
13	1	1	2	0	3	2	1	0	0	4	3	2
14	2	6	3	3	5	3	2	1	0	1	2	1
15	0	0	0	1	1	0	4	0	0	0	0	2
16	0	0	0	0	0	0	0	0	0	0	0	0

Now we can easily see the hidden message:

- [redacted]. So the chance that a man visits our store and buys a pair of shoes on these sizes is quite low, especially the oversize 15, and 16 — we have no unit sold for this size whole year!
- [redacted]. US man has foot size of 9.5 seems out number of other sizes. Focus to produce and store this size, we have great chance to sell shoes and get money!

But this is just a high level of examination. Keep diving deeper to reveal the entire of hidden messages our data set hiding.

Calculation

Because we have 17 different shoes sizes (men only — based on our plan), we need to calculate 17 different **CI (confidence intervals)**. Firstly let's calculate the **means** with Microsoft Excel using the =average(number1, [number2]...) function (in this post, I will not mention much about programming. Just stick with KISS principal)

	Men shoes, United States, 2016												Mean
Size (US) - Month	1	2	3	4	5	6	7	8	9	10	11	12	2016
6	4	1	3	1	3	3	3	4	3	7	3	0	2.91667
6.5	3	2	0	1	0	0	1	7	2	1	2	1	1.66667
7	0	0	1	0	6	4	4	2	3	0	0	0	1.66667
7.5	3	2	3	1	7	0	7	3	4	6	1	1	3.16667
8	7	9	7	3	12	2	9	4	7	5	2	6	6.08333
8.5	12	12	8	8	15	9	17	17	6	9	10	6	10.75000
9	17	13	13	11	21	22	25	30	26	25	13	10	18.83333
9.5	19	25	27	24	26	33	25	47	31	44	37	26	30.33333
10	17	26	26	19	16	31	25	24	23	31	15	20	22.75000
10.5	13	16	22	14	28	19	18	15	19	21	16	10	17.58333
11	5	16	13	10	10	11	15	8	9	7	6	7	9.75000
11.5	4	3	6	3	3	5	6	4	5	12	13	5	5.75000
12	3	0	0	4	4	4	3	12	4	9	2	1	3.83333
13	1	1	2	0	3	2	1	0	0	4	3	2	1.58333
14	2	6	3	3	5	3	2	1	0	1	2	1	2.41667
15	0	0	0	1	1	0	4	0	0	0	0	2	0.66667
16	0	0	0	0	0	0	0	0	0	0	0	0	0.00000
Total	110	132	134	103	160	148	165	178	142	182	125	98	

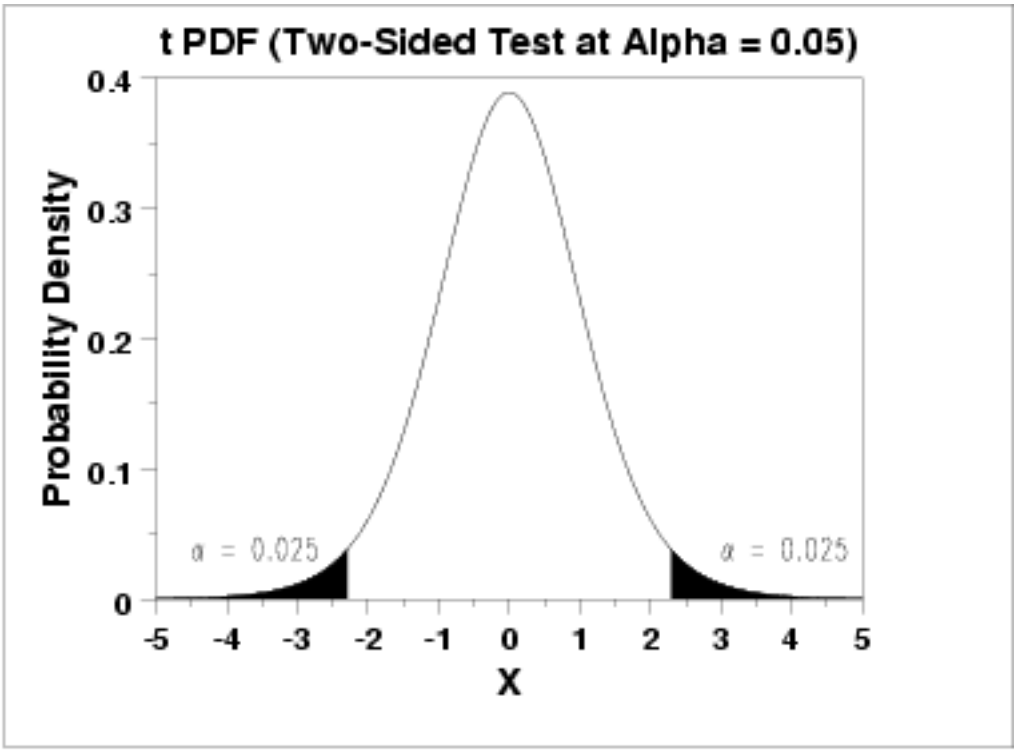
We do not know the population variance and our data set consists of only 12 observations (represented by 12 months of 2016), then we have to use the **T-Statistic**.

Let's **find the value in T-Statistic Table for 95% Confidence Interval with 11 Degrees of Freedom**!

$$t_{11,\,0.025} = ?$$

Let's explain a bit in case you concern:

- 11 is the Degree of Freedom of 12 months calculated using **df = n - 1** (with n=12)
- 0.025 is the Significant Level of 95% Confidence Interval calculated using **SL = (1 - CI) / 2**



Alright! Look at the T-Statistic Table below and we can find out the T-Value is 2.201 (right at the cross point between red column and blue row)

Degrees of freedom	Significance level					
	20% (0.20)	10% (0.10)	5% (0.05)	2% (0.02)	1% (0.01)	0.1% (0.001)
1	3.078	6.314	12.706	31.821	63.657	636.619
2	1.886	2.920	4.303	6.965	9.925	31.598
3	1.638	2.353	3.182	4.541	5.841	12.941
4	1.533	2.132	2.776	3.747	4.604	8.610
5	1.476	2.015	2.571	3.365	4.032	6.859
6	1.440	1.943	2.447	3.143	3.707	5.959
7	1.415	1.895	2.365	2.998	3.499	5.405
8	1.397	1.860	2.306	2.896	3.355	5.041
9	1.383	1.833	2.262	2.821	3.250	4.781
10	1.372	1.812	2.228	2.764	3.169	4.587
11	1.363	1.796	2.201	2.718	3.106	4.437
12	1.356	1.782	2.179	2.681	3.055	4.318
13	1.350	1.771	2.160	2.650	3.012	4.221
14	1.345	1.761	2.145	2.624	2.977	4.140
15	1.341	1.753	2.131	2.602	2.947	4.073
16	1.337	1.746	2.120	2.583	2.921	4.015
17	1.333	1.740	2.110	2.567	2.898	3.965
18	1.330	1.734	2.101	2.552	2.878	3.922
19	1.328	1.729	2.093	2.539	2.861	3.883
20	1.325	1.725	2.086	2.528	2.845	3.850
21	1.323	1.721	2.080	2.518	2.831	3.819
22	1.321	1.717	2.074	2.508	2.819	3.792
23	1.319	1.714	2.069	2.500	2.807	3.767
24	1.318	1.711	2.064	2.492	2.797	3.745
25	1.316	1.708	2.060	2.485	2.787	3.725
26	1.315	1.706	2.056	2.479	2.779	3.707
27	1.314	1.703	2.052	2.473	2.771	3.690
28	1.313	1.701	2.048	2.467	2.763	3.674
29	1.311	1.699	2.043	2.462	2.756	3.659
30	1.310	1.697	2.042	2.457	2.750	3.646
40	1.303	1.684	2.021	2.423	2.704	3.551
60	1.296	1.671	2.000	2.390	2.660	3.460
120	1.289	1.658	1.980	2.158	2.617	3.373
∝	1.282	1.645	1.960	2.326	2.576	3.291

$$t_{11,\,0.025} = 2.20$$

Now, let's calculate Standard Errors and Margin Errors!

In Microsoft Excel, you can calculate Standard Errors by using this formula:

=STDEV.S(number1, [number2]…)/SQRT(n)

with: number1, number2… are number of shoes was sold

n = 12 (months)

And Margin Errors = Standard Errors * t(11, 0.025)

	Mean	Standard error	Margin error
Size (US) - Month	2016	2016	2016
6	2.91667	0.51432	1.13201
6.5	1.66667	0.55505	1.22167
7	1.66667	0.60720	1.33644
7.5	3.16667	0.69449	1.52858
8	6.08333	0.88299	1.94346
8.5	10.75000	1.12226	2.47010
9	18.83333	1.96882	4.33338
9.5	30.33333	2.44743	5.38679
10	22.75000	1.56730	3.44963
10.5	17.58333	1.36769	3.01029
11	9.75000	1.00849	2.21968
11.5	5.75000	0.96236	2.11815
12	3.83333	1.01379	2.23136
13	1.58333	0.37856	0.83321
14	2.41667	0.49937	1.09911
15	0.66667	0.35533	0.78209
16	0.00000	0.00000	0.00000

Last one, calculate the Confidence Interval

$$\bar{X} - z_{\alpha/2}\left(\frac{\sigma}{\sqrt{n}}\right) < \mu < \bar{X} + z_{\alpha/2}\left(\frac{\sigma}{\sqrt{n}}\right)$$

$\bar{X}$ = sample mean
N = sample size
σ = standard deviation

For 90% confidence interval: $z_{\alpha/2} = 1.65$

95% confidence interval: $z_{\alpha/2} = 1.96$

99% confidence interval: $z_{\alpha/2} = 2.58$

I know that you are getting headache with above formula but wait! We already have the Mean and Margin Errors, right? Then we can calculate the CI as (Mean — ME; Mean + ME)

	Mean	Standard error	Margin error	95% CI	
Size (US) - Month	2016	2016	2016	2016	
6	2.91667	0.51432	1.13201	1.78466	4.04867
6.5	1.66667	0.55505	1.22167	0.44500	2.88833
7	1.66667	0.60720	1.33644	0.33023	3.00310
7.5	3.16667	0.69449	1.52858	1.63808	4.69525
8	6.08333	0.88299	1.94346	4.13987	8.02679
8.5	10.75000	1.12226	2.47010	8.27990	13.22010
9	18.83333	1.96882	4.33338	14.49995	23.16671
9.5	30.33333	2.44743	5.38679	24.94655	35.72012
10	22.75000	1.56730	3.44963	19.30037	26.19963
10.5	17.58333	1.36769	3.01029	14.57304	20.59362
11	9.75000	1.00849	2.21968	7.53032	11.96968
11.5	5.75000	0.96236	2.11815	3.63185	7.86815
12	3.83333	1.01379	2.23136	1.60197	6.06469
13	1.58333	0.37856	0.83321	0.75012	2.41655
14	2.41667	0.49937	1.09911	1.31756	3.51578
15	0.66667	0.35533	0.78209	-0.11542	1.44876
16	0.00000	0.00000	0.00000	0.00000	0.00000

In 95% of the cases, the true population mean of sales for each shoes will fall into the respective interval. The ceiling values (upper bound, or the higher values) of the CI shows us the maximum number of shoes needed. And vice versa for the flooring values as they are the minimum number of shoes was sold. As we don't want to be low in stock a possible solution to the problem is get as many pairs of shoes as the closest number to the ceiling limit of the Confidence Interval to maximum selling possibility. And of course, we need to store more than the flooring limit of the CI to ensure all customers (who are men, and have particular shoes size) can buy their favorite shoes instead of leaving our store without buying because the items are out of stocks.

Let's round up the CI and see:

	Number of pairs		Conclusions
Size (US) - Month	2016		2016
6	2	4	With 95% confidence, in the US, 2016, we sold at least 2 pair(s) of shoes, and upto 4 pair(s)
6.5	0	3	With 95% confidence, in the US, 2016, we sold at least 0 pair(s) of shoes, and upto 3 pair(s)
7	0	3	With 95% confidence, in the US, 2016, we sold at least 0 pair(s) of shoes, and upto 3 pair(s)
7.5	2	5	With 95% confidence, in the US, 2016, we sold at least 2 pair(s) of shoes, and upto 5 pair(s)
8	4	8	With 95% confidence, in the US, 2016, we sold at least 4 pair(s) of shoes, and upto 8 pair(s)
8.5	8	13	With 95% confidence, in the US, 2016, we sold at least 8 pair(s) of shoes, and upto 13 pair(s)
9	14	23	With 95% confidence, in the US, 2016, we sold at least 14 pair(s) of shoes, and upto 23 pair(s)
9.5	25	36	With 95% confidence, in the US, 2016, we sold at least 25 pair(s) of shoes, and upto 36 pair(s)
10	19	26	With 95% confidence, in the US, 2016, we sold at least 19 pair(s) of shoes, and upto 26 pair(s)
10.5	15	21	With 95% confidence, in the US, 2016, we sold at least 15 pair(s) of shoes, and upto 21 pair(s)
11	8	12	With 95% confidence, in the US, 2016, we sold at least 8 pair(s) of shoes, and upto 12 pair(s)
11.5	4	8	With 95% confidence, in the US, 2016, we sold at least 4 pair(s) of shoes, and upto 8 pair(s)
12	2	6	With 95% confidence, in the US, 2016, we sold at least 2 pair(s) of shoes, and upto 6 pair(s)
13	1	2	With 95% confidence, in the US, 2016, we sold at least 1 pair(s) of shoes, and upto 2 pair(s)
14	1	4	With 95% confidence, in the US, 2016, we sold at least 1 pair(s) of shoes, and upto 4 pair(s)
15	0	1	With 95% confidence, in the US, 2016, we sold at least 0 pair(s) of shoes, and upto 1 pair(s)
16	0	0	With 95% confidence, in the US, 2016, we sold at least 0 pair(s) of shoes, and upto 0 pair(s)
Total	105	175	

Did we solved the problem?

Nope! Don't blame on me

Alright. **This is the conclusion for the year of 2016 but now is 2019, so can we use it?**

Absolutely **yes! We can calculate for 2017, 2018 and then get the Mean of them and give the prediction for 2019**

The prediction for 2019 can be incorrect due to other factors such as: fashion trend this year is changed, people like using sneakers more than shoes; or there is another shoes store just opened next to us. But this post ONLY analyze on historical data

This post we used data of 2016, in The US, and analyzed for man sizes only. So if you're a store manager, you have to do the same analyze for women to get a bigger picture of your store

Extension! Let's deep dive a bit more

I wonder do you notice the number of 105 and 175 in the bottom on previous image of "Conclusions based on sample data set". What does it mean?

105: the minimum number of pairs of shoes was sold. So as a store manager, we have to prepare big inventory enough to store this number of shoes or we will run out of stock.

175: a big inventory is good, but it's no need to have a huge inventory! Big enough to store 175 pairs is enough. Remember, the bigger inventory you have, the more expensive you pay for management

What do you do for size US 16? If your eyes are keen enough, you can see there was no product sold with this size. Let's check the data for 2017, and 2018 as well. If no item was sold, you need to stop producing or importing this size.

What if your inventory is not big enough to store the minimum number of shoes? No worry. This is the number of sold items for whole year. Let's check the Heatmap on Tableau

on the top of this post, you can see the distribution during 12 months. Convert each number (per month) to frequency — how many percentage that month reflects to whole year. You can figure out which month we sold the most, double check the inventory, prepare the importing plan, then you have zero worry facing with the trend of supply and demand

Conclusion

1. Data analyzing cannot help you AVOID the problem, but help you REFLECT the historical data to predict the future trend.
2. In data science as well as statistics, we DON'T ENSURE anything, we just give the conclusion base on how much CONFIDENCE (this post is 95%)
3. And the last one, you can **increase your CONFIDENCE by analyzing the historical numbers frequently**. Keep it update yearly, or monthly, or weekly. The more frequently you update your data, the more accuracy you can predict

https://towardsdatascience.com/data-science-in-inventory-management-real-case-in-managing-a-warehouse-6259cad17c0e

Top 10 Challenges of Business Inventory Management

Business inventory management is a serious challenge for many companies. Although a tight **inventory control and a sales forecast are two strategies that help with inventory management**, companies are still facing many inventory challenges. Taking the time to look at and understand these challenges can be a turning point in solving them. Here they are.

- **Breaking down the inventory** into **safety stock**--*an additional quantity of an item held by a company in inventory in order to reduce the risk that the item will be out of stock*, **replenishment stock**-- *the rate at which inventory travels along the supply chain from the manufacturer to the supplier to warehousing, picking, and shipment locations*, and **normal stock** in order to maintain adequate levels for each of them.
- **Using statistical formulas that integrate sales forecast data to accurately calculate safety stock levels**.
- **Recalculating safety stock levels at least every six months** to improve the effectiveness of your inventory control.
- **Deciding who takes key inventory control decisions** in order to maintain cost-effective inventory levels as well as to ensure a fast and reliable customer service.
- **Assembling a team that decides when new products have to be ordered or manufactured** to keep up with the effects of major marketing campaigns.
- **Determining how often to order new inventory items when this isn't set by the supplier or the factory from which the products are obtained**, while taking into account changeover and inventory costs.
- **Deciding whether inventory ordering should be done regularly using data from analytical tools**, in order to continuously improve the inventory control
- **Establishing a process for determining why *excess stock*(***--inventory levels exceed forecasted demand)*** arises and figuring out how to deal** with it in a cost-effective way.

- Assigning the tasks of **identifying the root causes of obsolete stock** to different teams within your organization, which work concurrently, linking their efforts.
- Performing effective inventory control on all parts of your inventory, not only the finished goods.

In the end, it should be remembered that inventory control and management are on-going challenges that businesses have to address all the time. They cannot be solved once and for all, but require persistent work and innovative approaches, else they can become serious problems that have a major impact on the profitability of a business.

https://dashboardstream.com/top-10-challenges-of-business-inventory-management/

Inventory Management — Dealing with unpredictable demand

This article looks to tackle the challenges that come with Stochastic Demand of products. This is a case study that deals with distributions of demands for different products and uses Monte Carlo Simulation to best manage its inventory and make an expected profit.

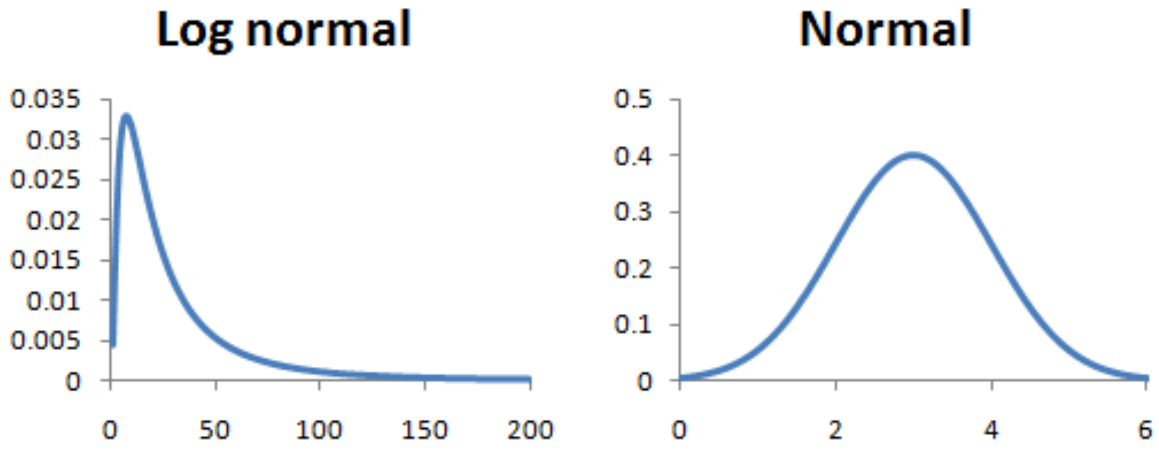
(Monte Carlo Simulation : https://www.youtube.com/watch?v=BfS2H1y6tzQ&ab_channel=Socratica)

Imagine you are a distributor of highly customized product and the demand for this product is therefore unique to every customer. It is not likely that you will receive an order for this product every day. Some products may vary due to seasonality, while others may have latent trends.

To mathematically model this stochastic demand, you must capture the sales information of each product at least for the last 12 months.

Now a customer can sometimes go into a store and not buy a particular item that they wanted. While it is difficult to model consumer behavior, we could provide a rough estimate that a customer has a probability 'p' of placing an order on any given day. This p can simply be calculated by dividing the number of orders last year by the number of working days.

Unless or until you have a contract with a particular client, another uncertainty is the order size. For this case study, an assumption was made that the **order size would follow a log-normal distribution whose distribution parameters are unknown** (which is often the case). Hence it is important to capture historical sales of the product.



(Normal distributions may present a few problems that log-normal distributions can solve. Mainly, **normal distributions can allow for negative random variables while log-normal distributions include all positive variables**. One of the most common applications where log-normal distributions are used in finance is in the analysis of stock prices --https://finance.zacks.com/stock-prices-considered-lognormal-10606.html

Note that **log-normal distributions are positively skewed with long right tails** due to low mean values and high variances in the random variables.)

Unless or until you have a contract with a particular client, another uncertainty is the order size. For this case study, **an assumption was made that the order size would follow a log-normal distribution whose distribution parameters are unknown** (which is often the case). Hence it is important to capture historical sales of the product.

This case looks at the sale of 4 different products and looks to adopt either **continuous review or periodic review policy to manage its inventory**. **The objective is to maximize its expected profit**.

https://www.ukessays.com/essays/business/continuous-review-periodic-review-system-1062.php

The **continuous review system** requires **knowing physical inventory all the times**, like using a barcode scanner every time cashier scans product purchased by the customer to update inventory. This method is more expensive to administer because inventory **needs to update each time something goes out of shelf or comes into the shelf**. It requires a lower level of safety since there is only uncertainty during the delivery period is the magnitude of demand. Therefore, during that time, the only safety stock required is for potential stockouts. (Wisner, Tan, Leong, Stanley, 2012, p.90).

The advantage of the continuous review system:

- Provide real-time updates of inventory counts
- Make easier to know when to order
- Provide accurate accounting

The disadvantage of the continuous system:

- Involves additional cost

The **periodic review system**, **evaluate inventory at specific times like counting inventory at the end of each month**. It is inexpensive to administer since counting takes place at a particular time, but a higher level of safety is required to buffer against uncertainty in demand over longer planning horizon. (Wisner et al., 2012, p.90).

The advantage of the periodic review system:

- Reduce time for business owner to analyze inventory counts
- Allows the business owner having more time to run another aspect of the business
- Simple to administer
- Save labor cost for counting

The disadvantage of the periodic review system:

- It may not provide an accurate inventory count when there is a high volume of sale
- It may also make accounting inaccurate
- There is little control over inventory movement

Product	Lead Time (days)	Volume (m^3)	Cost ($)	Selling Price ($)	Initial Inventory
1	9	0.57	12	16.1	2750
2	6	0.052	7	8.6	22500
3	16	0.53	6	10.2	5200
4	22	1.05	37	68	1400

Cubic metre

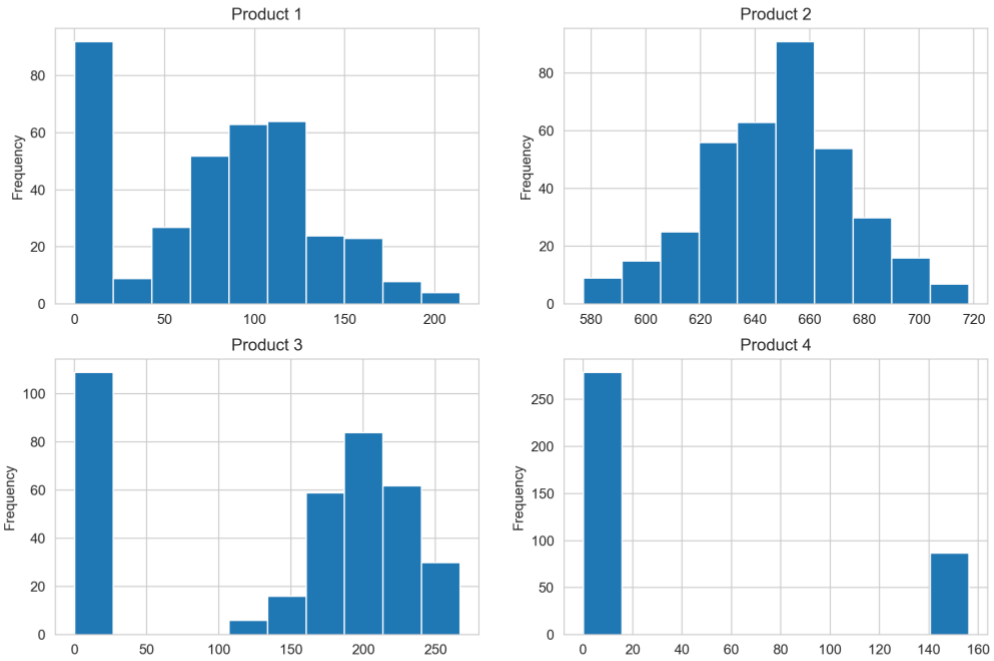
One cubic metre of concrete (representing the world annual production per inhabitant).

General information

Unit system	SI
Unit of	volume
Symbol	m³ or m³

https://en.wikipedia.org/wiki/Cubic_metre

Based on the past year sales, given below is the histogram of the **demand distribution of each product**.



You can see the stark differences in the demands for each product. For example, Product 2 is a high-volume product that gets bought every day (p = 1) and the mean order size is 649. Whereas Product 4 is purchased 24% of the time and its mean order size is around 150. The table below provides a summary of each product that can be calculated purely based on past sales data.

	Product 1	**Product 2**	**Product 3**	**Product 4**
Mean	103.5	648.55	201.68	150.07
Standard Deviation	37.32	26.45	31.08	3.22
Expected Proportion of Days	0.76	1.00	0.70	0.24
Expected Demand (Lead Time)	705	3891.31	2265.84	784.79
Standard Deviation (Lead Time)	165.01	64.78	383.33	299.72

It is important to understand the statistics of the demand during **lead time**.

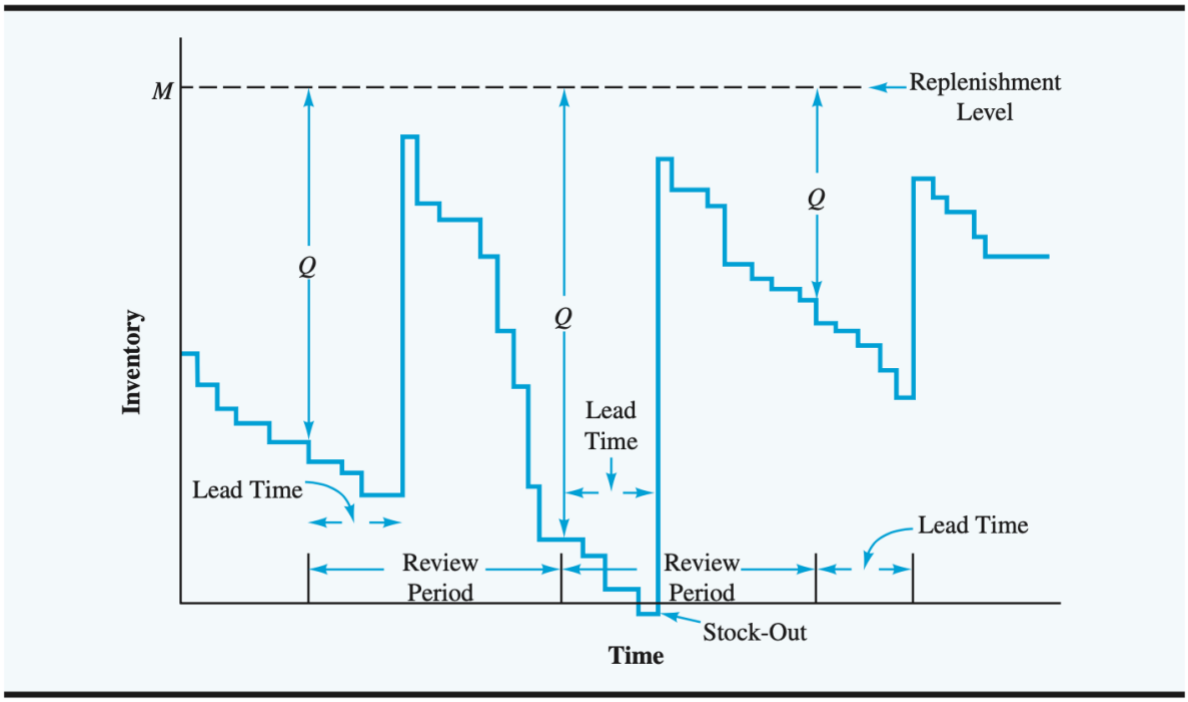
"A lead time is the **latency between the initiation and completion of a process**. For example, the **lead time between the placement of an order and**

delivery of new cars by a given manufacturer might be between 2 weeks and 6 months, depending on various particularities. One business dictionary defines **"manufacturing lead time" as the total time required to manufacture an item**, including order preparation time, queue time, setup time, run time, move time, inspection time, and put-away time. For make-to-order products, it is the time between release[vague] of an order and the production and shipment that fulfill that order. For make-to-stock products, it is the time taken from the release of an order to production and receipt into finished goods inventory.[1]"

To put these numbers into context, the lead time for Product 1 is 9 days (as mentioned earlier), so in those 9 days, the distributor can expect an average of 705 orders. This needs to be taken into consideration while placing the original order otherwise the distributor would always fall short of meeting the demand.

Periodic Review

With a periodic review system, the inventory is checked and reordering is done only at specified points in time. For example, inventory may be checked and orders placed on a weekly, biweekly, monthly, or some other periodic basis. When a firm or business handles multiple products, the periodic review system offers the advantage of requiring that orders for several items be placed at the same preset periodic review time. With this type of inventory system, the shipping and receiving of orders for multiple products are easily coordinated. Under the previously discussed order-quantity, reorder point systems, the reorder points for various products can be encountered at substantially different points in time, making the coordination of orders for multiple products more difficult. [1]



In the Periodic Review policy method, **the stock is replenished after a certain time period**. This time period is dependent on the review period and the lead time. **Monte Carlo technique was used to simulate the daily demand of each product in the store**.

	Product 1	Product 2	Product 3	Product 4
Mean	103.5	648.55	201.68	150.07
Standard Deviation	37.32	26.45	31.08	3.22
Expected Proportion of Days	0.76	1.00	0.70	0.24
Expected Demand (Lead Time)	705	3891.31	2265.84	784.79
Standard Deviation (Lead Time)	165.01	64.78	383.33	299.72

From the distributions plot in the previous section, we noticed that other than Product 2, not every product is bought on each and every day. **The expected proportion of orders calculated indicate the probability that the product would be bought on the day**. For example, for Product 1 the expected proportion is 0.76 which means that on **any given**

day within the year, there is a 76% chance that a customer will buy Product 1.

If a product is purchased, then the demand follows a log-normal distribution. The demand distribution from the previous year was converted into a log-normal distribution by taking the logarithm of the daily values. To simulate daily customer purchasing behaviour, a random number was picked from a uniform distribution within 0 and 1.

```
def daily_demand(mean, sd, probability):
    random_num = np.random.uniform(0, 1)
    if random_num > probability:
        return 0
    else:
        return np.exp(np.random.normal(mean, sd))
```

A Monte Carlo simulation was conducted to simulate the behaviour of demand and the calculation of profit for one realization. In the simulation, the algorithm iterates through each day trying to capture the inventory level for the product. This is to generate the product demand for that day.

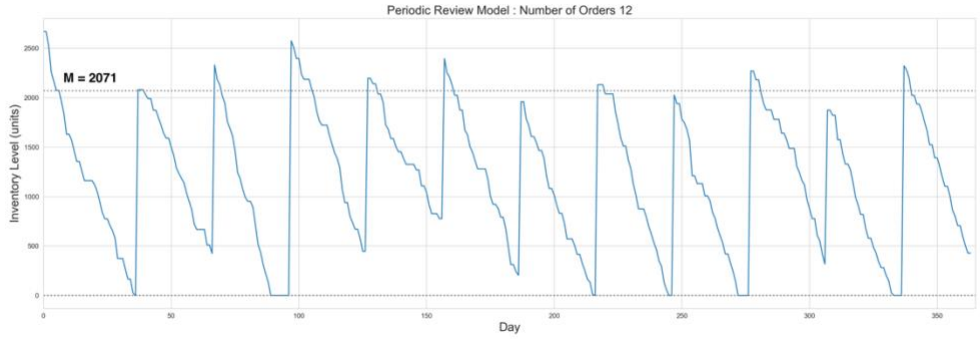
```
def monte_carlo_ray(M, product, review_period=30):
    inventory = product.starting_stock
    mean = product.mean
    sd = product.sd
    lead_time = product.lead_time
    probability = product.probability
    demand_lead = product.demand_lead

    q = 0
    stock_out = 0
    counter = 0
    order_placed = False

    # dictionary to store all the information
    data = {'inv_level': [], 'daily_demand': [], 'units_sold': [],
'units_lost': [], 'orders': []}

    for day in range(1, 365):
        day_demand = daily_demand(mean, sd, probability)
        data['daily_demand'].append(day_demand)

        if day % review_period == 0:
            # Placing the order
            q = M - inventory + demand_lead
            order_placed = True
            data['orders'].append(q)
```

```
        if order_placed:
            counter += 1

        if counter == lead_time:
            # Restocking day
            inventory += q
            order_placed = False
            counter = 0

        if inventory - day_demand >= 0:
            data['units_sold'].append(day_demand)
            inventory -= day_demand
        elif inventory - day_demand < 0:
            data['units_sold'].append(inventory)
            data['units_lost'].append(day_demand - inventory)
            inventory = 0
            stock_out += 1

        data['inv_level'].append(inventory)

    return data
```

The logic of the algorithm is as follows:

1. **If the demand can be completely serviced by the current inventory level** — the **inventory level is reduced by the demand and number of units sold on that day increments**
2. **If the demand cannot be serviced completely by the inventory level** — the inventory on hand would be the number of units sold on that day

In order to model the Periodic Review policy, the algorithm keeps track of the current day in the year. **If the day of the year is equal to the review period, then the order is placed to replenish the stock to an order up-to quantity M**. This value is the decision variable and is passed as an input to the algorithm. After the lead time has passed for that particular product, the inventory is updated by the order quantity that was placed. This is done for a duration of 365 days.



The inventory levels for one simulation of 365 days can be used to determine the profit the store would have made for that year. All the units that were sold are multiplied by the products selling price to calculate revenue.

The costs are broken down into three components,

1. product costs—Product costs are calculated by **multiplying the unit costs of each product to the aggregation of the units ordered**.
2. ordering costs—The ordering costs are calculated by **multiplying the number of times in that year an order was placed to the individual cost of ordering for that product**. The inventory levels for each day of the year were aggregated to indicate how much stock was held throughout the duration of the year.
3. holding costs—The holdings costs were then calculated by **multiplying the amount of stock held with the unit size of the product and the daily cost of holding a unit**.

These costs were subtracted from the revenue to give the corresponding profit for that one realization of the year. The number of units lost was aggregated and divided by the demand for the year to give the proportion of lost orders for that year.

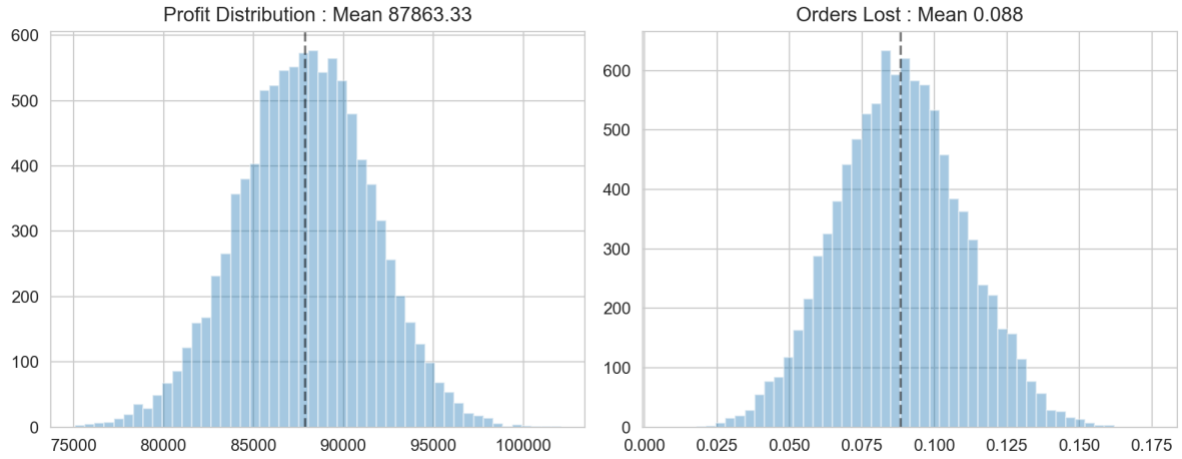
The mathematical formula for the Annual profit is given below.

$$\text{Annual profit }_i = SP_i \sum_{t=1}^{365} S_{i,t} - \left[\left(\frac{20V_i}{365} \right) \sum_{t=1}^{365} I_{i,t} + N_i C_{o,i} + \sum_{t=1}^{365} c_i P_{i,t} \right]$$

```
def calculate_profit(data, product):
```

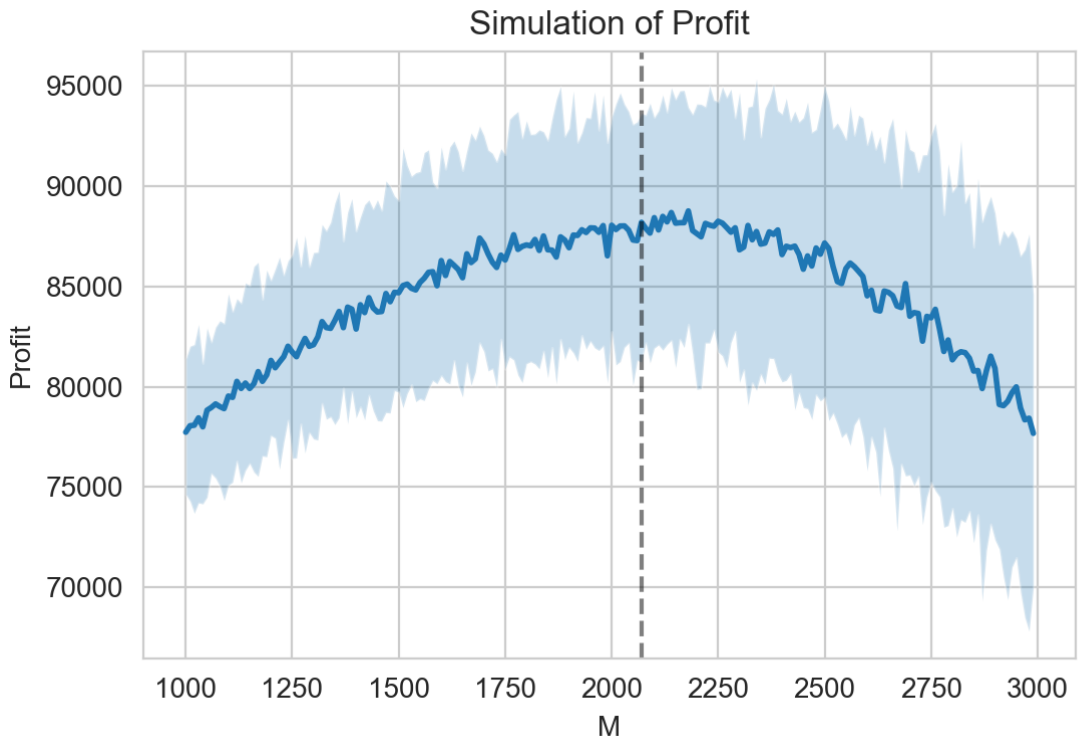
```
unit_cost = product.unit_cost
selling_price = product.selling_price
holding_cost = product.holding_cost
order_cost = product.ordering_cost
size = product.size
days = 365

revenue = sum(data['units_sold']) * selling_price
Co = len(data['orders']) * order_cost
Ch = sum(data['inv_level']) * holding_cost * size * (1 / days)
cost = sum(data['orders']) * unit_cost

profit = revenue - cost - Co - Ch

return profit
```

This simulation was carried out 10,000 times to give multiple realizations of profit and proportions of lost orders for each time. These results were used to plot a histogram and calculate the mean and standard deviation of profit and the proportion of lost orders for that particular order up-to point (M). The figure below is one such simulation for Product 1 with the order up-to quantity to be 2071.



Similarly, this exercise can be carried out for a range of values of M to determine the value that gives us the highest expected profit. As you can see from the figure below for Product 1, we simulated the results for a range of values between 1000 to 3000. This gave an optimum expected profit of $87,863 (in running the simulations again, $87,992 was optimal) for the order up-to point of 2071.

These experiments were carried out for the other products and their optimum was calculated accordingly. Since profit is linearly dependent on the demand and the costs associated with a particular product, maxima for that function can be calculated.



Using a one month review period, the table below lists out the optimal order points, the expected annual profits and the proportion of orders lost over the period of the year for each of the products.

	Product 1	**Product 2**	**Product 3**	**Product 4**
Optimal Order up-to point	2071	18424	4154	1305
Expected Annual Profit	$ 87,992	$ 372,188	$ 165,959	$ 320,507
Expected Standard Deviation	$ 5035	$ 1454	$ 6561	$ 29,864
Expected proportion of lost orders	0.077	0.01	0.05	0.074

Monte Carlo Simulation

https://pbpython.com/monte-carlo.html

Introduction

There are many sophisticated models people can build for solving a forecasting problem. However, they frequently stick to simple Excel models based on average historical values, intuition and some high level domain-specific heuristics. This approach may be precise enough for the problem at hand but there are alternatives that can add more information to the prediction with a reasonable amount of additional effort.

One approach that can produce a better understanding of the range of potential outcomes and help avoid the "flaw of averages" is a Monte Carlo simulation. The rest of this article will describe how to use python with pandas and numpy to build a Monte Carlo simulation to predict the range of potential values for a sales compensation budget. This approach is meant to be simple enough that it can be used for other problems you might encounter but also powerful enough to provide insights that a basic "gut-feel" model can not provide on its own.

Problem Background

For this example, we will try **to predict how much money we should budget for sales commissions for the next year**. This problem is useful for modeling because we have a defined formula for calculating commissions and we likely have some experience with prior years' commissions payments.

This problem is also important from a business perspective. **Sales commissions can be a large selling expense and it is important to plan appropriately for this expense**. In addition, the use of a Monte Carlo simulation is a relatively simple improvement that can be made to augment(เพิ่มพูน) what is normally an unsophisticated estimation process.

In this example, the sample sales commission would look like this for a 5 person sales force:

	A	B	C	D	E	F
1	**Sales Rep**	**Sales Target**	**Actual Sales**	**Percent To Plan**	**Commission Rate**	**Commission Amount**
2	1	$100,000	$88,000	88.0%	2.0%	$1,760
3	2	$200,000	$202,000	101.0%	4.0%	$8,080
4	3	$75,000	$90,000	120.0%	4.0%	$3,600
5	4	$400,000	$360,000	90.0%	0.0%	$0
6	5	$500,000	$350,000	70.0%	0.0%	$0
7						
8	Total	$1,275,000	$1,090,000			$13,440

In this example, the commission is a result of this formula:

Commission Amount = Actual Sales * Commission Rate

The commission rate is based on this Percent To Plan table:

H	I
Rate Schedule	
0 – 90%	2.00%
91-99%	3.00%
>= 100	4.00%

Before we build a model and run the simulation, let's look at a simple approach for predicting next year's commission expense.

Naïve Approach to the Problem

Imagine your task as Amy or Andy analyst is to tell finance how much to budget for sales commissions for next year. One approach might be to assume everyone makes 100% of their target and earns the 4% commission rate. Plugging these values into Excel yields this:

11	*Base Case*					
12	**Sales Rep**	**Sales Target**	**Actual Sales**	**Percent To Plan**	**Commission Rate**	**Commission Amount**
13	1	$100,000	$100,000	100.0%	4.0%	$4,000
14	2	$200,000	$200,000	100.0%	4.0%	$8,000
15	3	$75,000	$75,000	100.0%	4.0%	$3,000
16	4	$400,000	$400,000	100.0%	4.0%	$16,000
17	5	$500,000	$500,000	100.0%	4.0%	$20,000
18						
19	Total	$1,275,000	$1,275,000			$51,000

Imagine you present this to finance, and they say, “We never have everyone get the same commission rate. We need a more accurate model.”

For round two, you might try a couple of ranges:

21	*Scenario #1*					
22	**Sales Rep**	**Sales Target**	**Actual Sales**	**Percent To Plan**	**Commission Rate**	**Commission Amount**
23	1	$100,000	$95,000	95.0%	3.0%	$2,850
24	2	$200,000	$204,000	102.0%	4.0%	$8,160
25	3	$75,000	$60,000	80.0%	2.0%	$1,200
26	4	$400,000	$480,000	120.0%	4.0%	$19,200
27	5	$500,000	$400,000	80.0%	2.0%	$8,000
28						
29	Total	$1,275,000	$1,239,000			$39,410

Or another one:

31	*Scenario #2*					
32	**Sales Rep**	**Sales Target**	**Actual Sales**	**Percent To Plan**	**Commission Rate**	**Commission Amount**
33	1	$100,000	$105,000	105.0%	4.0%	$4,200
34	2	$200,000	$140,000	70.0%	2.0%	$2,800
35	3	$75,000	$74,250	99.0%	3.0%	$2,228
36	4	$400,000	$352,000	88.0%	2.0%	$7,040
37	5	$500,000	$550,000	110.0%	4.0%	$22,000
38						
39	Total	$1,275,000	$1,221,250			$38,268

Now, you have a little bit more information and go back to finance. This time finance says, “this range is useful but what is your confidence in this range? Also, we need you to do this for a sales force of 500 people and model several different rates to determine the amount to budget.” Hmmm… Now, what do you do?

This simple approach illustrates the basic iterative method for a Monte Carlo simulation. You iterate through this process many times in order to determine a range of potential commission values for the year. Doing this manually by hand is challenging. Fortunately, python makes this approach much simpler.

Monte Carlo

Now that we have covered the problem at a high level, we can discuss how Monte Carlo analysis might be a useful tool for predicting commissions expenses for the next year. At its simplest level, a Monte Carlo analysis (or simulation) involves running many scenarios with different random inputs and summarizing the distribution of the results.

Using the commissions analysis, we can continue the manual process we started above but run the program 100's or even 1000's of times and we will get a distribution of potential commission amounts. This distribution can inform the likelihood that the expense will be within a certain window. At the end of the day, this is a prediction so we will likely never predict it exactly. We can develop a more informed idea about the potential risk of under or over budgeting.

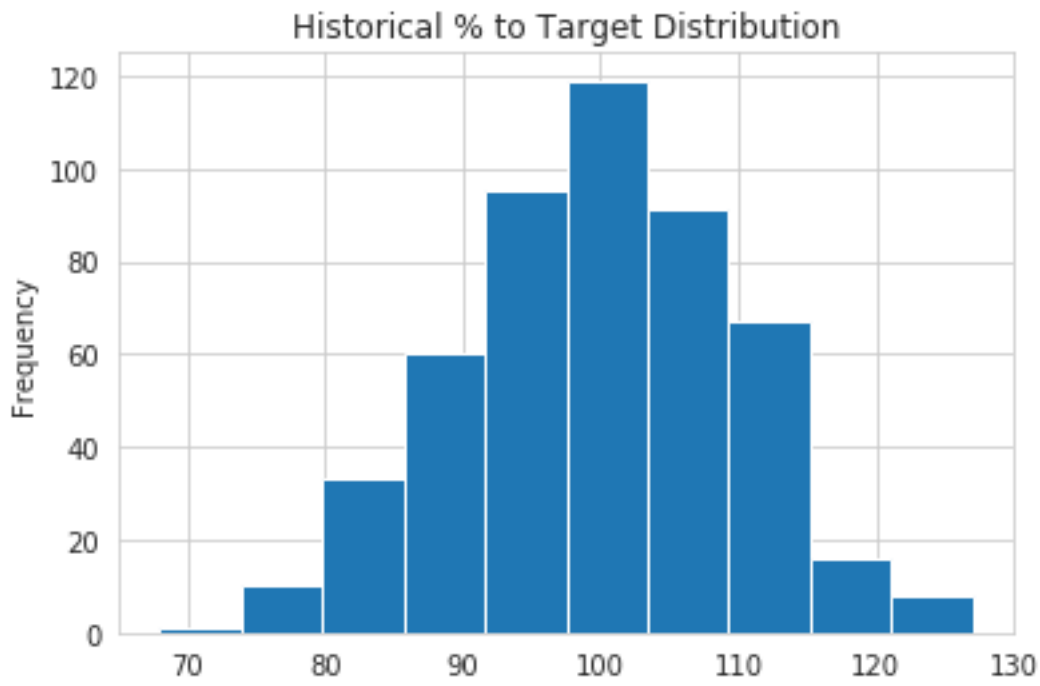
There are two components to running a Monte Carlo simulation:

- the equation to evaluate
- the random variables for the input

We have already described the equation above. Now we need to think about how to populate the random variables.

One simple approach would be to take a random number between 0% and 200% (representing our intuition about commissions rates). However, because we pay commissions every year, we understand our problem in a little more detail and can use that prior knowledge to build a more accurate model.

Because we have paid out commissions for several years, we can look at a typical historical distribution of percent to target:



This distribution looks like a normal distribution with a mean of 100% and standard deviation of 10%. This insight is useful because we can model our input variable distribution so that it is similar to our real world experience.

If you are interested in additional details for estimating the type of distribution, I found this article interesting.

Building a Python Model

We can use pandas to construct a model that replicates the Excel spreadsheet calculation. There are other python approaches to building Monte Carlo models but I find that this pandas method is conceptually easier to comprehend if you are coming from an Excel background. It also has the added benefit of generating pandas dataframes that can be inspected and reviewed for reasonableness.

First complete our imports and set our plotting style:

```
import pandas as pd
import numpy as np
import seaborn as sns
```

```
sns.set_style('whitegrid')
```

For this model, we will use a random number generation from numpy. The handy aspect of numpy is that there are several random number generators that can create random samples based on a predefined distribution.

As described above, we know that our historical percent to target performance is centered around a a mean of 100% and standard deviation of 10%. Let's define those variables as well as the number of sales reps and simulations we are modeling:

```
avg = 1
std_dev = .1
num_reps = 500
num_simulations = 1000
```

Now we can use numpy to generate a list of percentages that will replicate our historical normal distribution:

```
pct_to_target = np.random.normal(avg, std_dev, num_reps).round(2)
```

For this example, I have chosen to round it to 2 decimal places in order to make it very easy to see the boundaries.

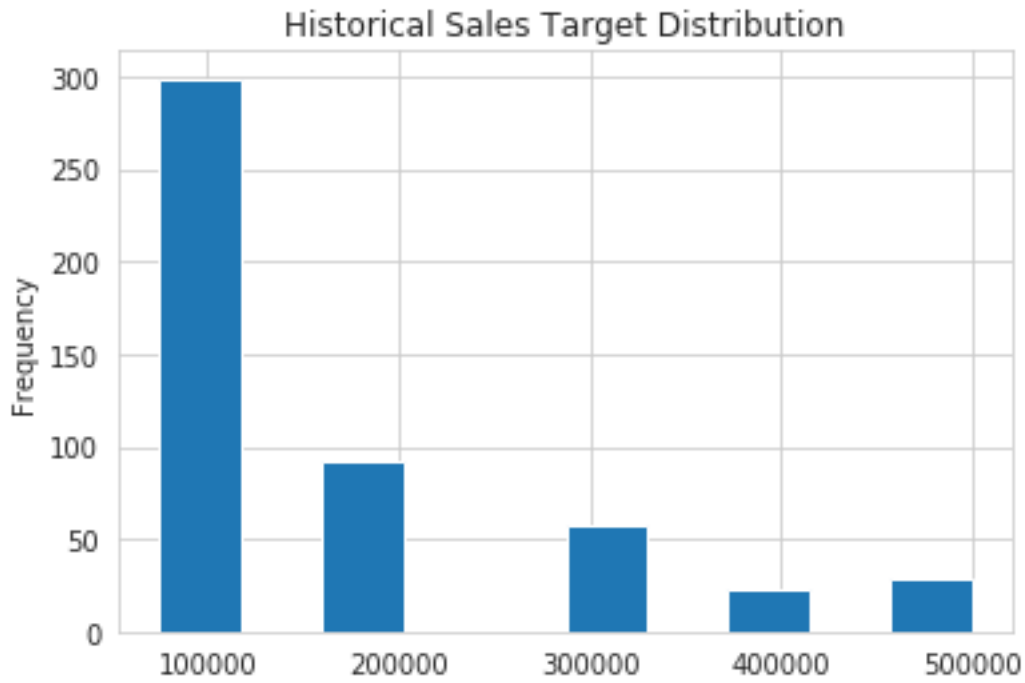
Here is what the first 10 items look like:

```
array([0.92, 0.98, 1.1 , 0.93, 0.92, 0.99, 1.14, 1.28, 0.91, 1.  ])
```

This is a good quick check to make sure the ranges are within expectations.

Since we are trying to make an improvement on our simple approach, we are going to stick with a normal distribution for the percent to target. By using numpy though, we can adjust and use other distribution for future models if we must. However, I do warn that you should not use other models without truly understanding them and how they apply to your situation.

There is one other value that we need to simulate and that is the actual sales target. In order to illustrate a different distribution, we are going to assume that our sales target distribution looks something like this:



This is definitely not a normal distribution. This distribution shows us that sales targets are set into 1 of 6 buckets and the frequency gets lower as the amount increases. This distribution could be indicative of a very simple target setting process where individuals are bucketed into certain groups and given targets consistently based on their tenure, territory size or sales pipeline.

For the sake of this example, we will use a uniform distribution but assign lower probability rates for some of the values.

Here is how we can build this using numpy.random.choice

```
sales_target_values = [75_000, 100_000, 200_000, 300_000, 400_000,
500_000]

sales_target_prob = [.3, .3, .2, .1, .05, .05]

sales_target = np.random.choice(sales_target_values, num_reps,
p=sales_target_prob)
```

Admittedly this is a somewhat contrived example but I wanted to show how different distributions could be incorporated into our model.

Now that we know how to create our two input distributions, let's build up a pandas dataframe:

```
df = pd.DataFrame(index=range(num_reps), data={'Pct_To_Target':
pct_to_target,
                                               'Sales_Target':
sales_target})


df['Sales'] = df['Pct_To_Target'] * df['Sales_Target']
```

Here is what our new dataframe looks like:

	Pct_To_Target	Sales_Target	Sales
0	0.92	100000	92000.0
1	0.98	75000	73500.0
2	1.10	500000	550000.0
3	0.93	200000	186000.0
4	0.92	300000	276000.0

You might notice that I did a little trick to calculate the actual sales amount. For this problem, the actual sales amount may change greatly over the years but the performance distribution remains remarkably consistent. Therefore, I'm using the random distributions to generate my inputs and backing into the actual sales.

The final piece of code we need to create is a way to map our `Pct_To_Target` to the commission rate. Here is the function:

```
def calc_commission_rate(x):
    """ Return the commission rate based on the table:
    0-90% = 2%
    91-99% = 3%
    >= 100 = 4%
    """
```

```
    if x <= .90:
        return .02
    if x <= .99:
        return .03
    else:
        return .04
```

The added benefit of using python instead of Excel is that we can create much more complex logic that is easier to understand than if we tried to build a complex nested if statement in Excel.

Now we create our commission rate and multiply it times sales:

```
df['Commission_Rate'] = df['Pct_To_Target'].apply(calc_commission_rate)
df['Commission_Amount'] = df['Commission_Rate'] * df['Sales']
```

Which yields this result, which looks very much like an Excel model we might build:

	Pct_To_Target	Sales_Target	Sales	Commission_Rate	Commission_Amount
0	97.0	100000	97000.0	.03	2910.0
1	92.0	400000	368000.0	.03	11040.0
2	97.0	200000	194000.0	.03	5820.0
3	103.0	200000	206000.0	.04	8240.0
4	87.0	75000	65250.0	.02	1305.0

There you have it!

We have replicated a model that is similar to what we would have done in Excel but we used some more sophisticated distributions than just throwing a bunch of random number inputs into the problem.

If we sum up the values (only the top 5 are shown above) in the `Commission_Amount` column, we can see that this simulation shows that we would pay $2,923,100.

Let's Loop

The real "magic" of the Monte Carlo simulation is that if we run a simulation many times, we start to develop a picture of the likely distribution of results. In Excel, you would need VBA or another plugin to run multiple iterations. In python, we can use a for loop to run as many simulations as we'd like.

In addition to running each simulation, we save the results we care about in a list that we will turn into a dataframe for further analysis of the distribution of results.

Here is the full for loop code:

```
# Define a list to keep all the results from each simulation that we
want to analyze
all_stats = []


# Loop through many simulations
for i in range(num_simulations):


    # Choose random inputs for the sales targets and percent to target
    sales_target = np.random.choice(sales_target_values, num_reps,
p=sales_target_prob)
    pct_to_target = np.random.normal(avg, std_dev, num_reps).round(2)


    # Build the dataframe based on the inputs and number of reps
    df = pd.DataFrame(index=range(num_reps), data={'Pct_To_Target':
pct_to_target,
                                                   'Sales_Target':
sales_target})


    # Back into the sales number using the percent to target rate
    df['Sales'] = df['Pct_To_Target'] * df['Sales_Target']


    # Determine the commissions rate and calculate it
    df['Commission_Rate'] =
df['Pct_To_Target'].apply(calc_commission_rate)
    df['Commission_Amount'] = df['Commission_Rate'] * df['Sales']
```

```
    # We want to track sales,commission amounts and sales targets over
all the simulations
    all_stats.append([df['Sales'].sum().round(0),
                      df['Commission_Amount'].sum().round(0),
                      df['Sales_Target'].sum().round(0)])
```

While this may seem a little intimidating at first, we are only including 7 python statements inside this loop that we can run as many times as we want. On my standard laptop, I can run 1000 simulations in 2.75s so there is no reason I can't do this many more times if need be.

At some point, there are diminishing returns. The results of 1 Million simulations are not necessarily any more useful than 10,000. My advice is to try different amounts and see how the output changes.

In order to analyze the results of the simulation, I will build a dataframe from `all_stats` :

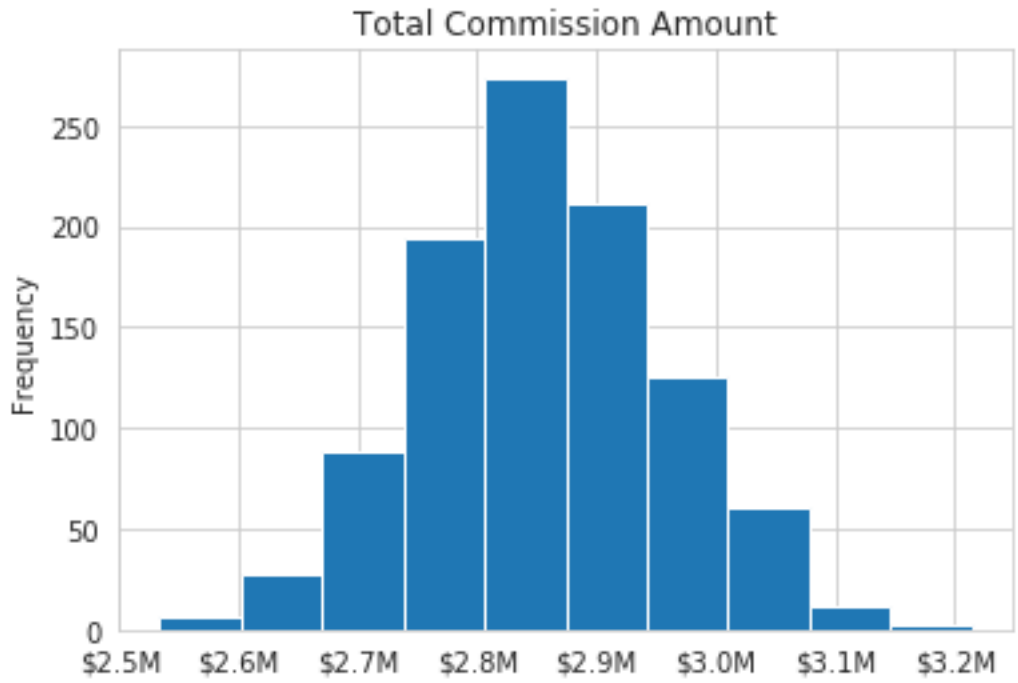
```
results_df = pd.DataFrame.from_records(all_stats, columns=['Sales',

'Commission_Amount',

'Sales_Target'])
```

Now, it is easy to see what the range of results look like:

```
results_df.describe().style.format('{:,}')
```

	Sales	Commission_Amount	Sales_Target
count	1,000.0	1,000.0	1,000.0
mean	83,617,936.0	2,854,916.1	83,619,700.0
std	2,727,222.9	103,003.9	2,702,621.8
min	74,974,750.0	2,533,810.0	75,275,000.0
25%	81,918,375.0	2,786,088.0	81,900,000.0
50%	83,432,500	2,852,165.0	83,525,000.0
75%	85,318,440.0	2,924,053.0	85,400,000.0
max	92,742,500.0	3,214,385.0	91,925,000.0

Graphically, it looks like this:



So, what does this chart and the output of describe tell us? We can see that the average commissions expense is $2.85M and the standard deviation is $103K. We can also see that the commissions payment can be as low as $2.5M or as high as $3.2M.

Based on these results, how comfortable are you that the expense for commissions will be less than $3M? Or, if someone says, “Let’s only budget $2.7M” would you feel comfortable that your expenses would be below that amount? Probably not.

Therein lies one of the benefits of the Monte Carlo simulation. You develop a better understanding of the distribution of likely outcomes and can use that knowledge plus your business acumen to make an informed estimate.

The other value of this model is that you can model many different assumptions and see what happens. Here are some simple changes you can make to see how the results change:

- Increase top commission rate to 5%
- Decrease the number of sales people
- Change the expected standard deviation to a higher amount
- Modify the distribution of targets

Now that the model is created, making these changes is as simple as a few variable tweaks and re-running your code. You can view the notebook associated with this post on github.

Another observation about Monte Carlo simulations is that they are relatively easy to explain to the end user of the prediction. The person receiving this estimate may not have a deep mathematical background but can intuitively understand what this simulation is doing and how to assess the likelihood of the range of potential results.

Finally, I think the approach shown here with python is easier to understand and replicate than some of the Excel solutions you may encounter. Because python is a programming language, there is a linear flow to the calculations which you can follow.

Conclusion

A Monte Carlo simulation is a useful tool for predicting future results by calculating a formula multiple times with different random inputs. This is a process you can execute in Excel but it is not simple to do without some VBA or potentially expensive third party plugins. Using numpy and pandas to build a model and generate multiple potential results and analyze them is relatively straightforward. The other added benefit is that analysts can run many scenarios by changing the inputs and can move on to much more sophisticated models in the future if the needs arise. Finally, the results can be shared with non-technical users and facilitate discussions around the uncertainty of the final results.

I hope this example is useful to you and gives you ideas that you can apply to your own problems. Please feel free to leave a comment if you find this article helpful for developing your own estimation models.

The Flaw of Averages

https://hbr.org/2002/11/the-flaw-of-averages

Consider the case of the statistician who drowns while fording a river that he calculates is, on average, three feet deep. If he were alive to tell the tale, he would expound on the "flaw of averages," which states, simply, that plans based on assumptions about average conditions usually go wrong. This basic but almost always unseen flaw shows up everywhere in business, distorting accounts, undermining forecasts, and dooming apparently well-considered projects to disappointing results.

Let's say that a company I'll call HealthCeuticals sells a perishable antibiotic. Although demand for the drug varies, for years the average monthly demand has been 5,000 units, so that's the quantity the company currently stocks. One day, the boss appears. "Give me a forecast of demand for next year," he says to his product manager. "I need it to estimate inventory cost for the budget." The product manager responds, "Demand varies from month to month. Here, let me give you a distribution." But the boss doesn't want a "distribution." "Give me a number!" he insists. "Well," the manager says meekly, "the average demand is 5,000 units a month. So, if you need a single number, go with 5,000."

The boss now proceeds to estimate inventory costs, which are calculated as follows: If monthly demand is less than the amount stocked, the firm incurs a spoilage cost of $50 per unsold unit. On the other hand, if the demand is greater than the amount stocked, the firm must air-freight the extra units at an increased cost of $150 each. These are the only two costs that depend on the accuracy of the forecast. The boss has developed a spreadsheet model to calculate the costs associated with any given demand and amount stocked. Since the average demand is 5,000 units, he plugs in 5,000. Since the company always stocks 5,000 units, the spreadsheet dutifully reports that for this average demand, the cost is zero: no spoilage or airfreight costs.

A bottom line based on average assumptions should be the average bottom line, right? It may miss fluctuations from month to month, but shouldn't you at least get the correct average cost by plugging in average demand? No. It's easy to see that the average cost can't be zero

by noting that when demand for HealthCeuticals' antibiotic deviates either up or down from the average, the company incurs costs.

Show Me the Number

Executives' desire to work with "a number," to plug in an average figure, is legendary. But whenever an average is used to represent an uncertain quantity, it ends up distorting the results because it ignores the impact of the inevitable variations. Averages routinely gum up accounting, investments, sales, production planning, even weather forecasting. Even the Generally Accepted Accounting Principles sanction the "flaw," requiring that uncertainties such as bad debt be entered a s single numbers. (To its credit, the SEC has proposed new rules that would begin to address this problem.)

In one celebrated, real-life case, relying on averages forced Orange County, California, into insolvency. In the summer of 1994, interest rates were low and were expected to remain so. Officials painted a rosy picture of the county's financial portfolio based on this expected future behavior of interest rates. But had they explicitly considered the well-documented range of interest-rate uncertainties, instead of a single, average interest-rate scenario, they would have seen that there was a 5% chance of losing $1 billion or more—which is exactly what happened. The average hid the enormous riskiness of their investments.

More recently, a failure to appreciate the flaw led to $2 billion in property damage in North Dakota. In 1997, the U.S. Weather Service forecast that North Dakota's rising Red River would crest at 49 feet. Officials in Grand Forks made flood management plans based on this single figure, which represented an average. In fact, the river crested above 50 feet, breaching the dikes, and unleashing a flood that forced 50,000 people from their homes.

Fixing the Flaw

Some executives are already attuned to the importance of acting on a range of relevant numbers—a distribution—rather than single values, and they employ statisticians who estimate the distributions of everything from stock prices to electricity demand. Increasingly, companies are also turning to software-based cures for the flaw. Many programs now simulate uncertainty, generating thousands of possible values for a given scenario—in essence, replacing the low-resolution "snapshot" of a single average number with a detailed "movie."

The movie comprises a whole range of possible values and their likelihood of occurring—the frequency distribution.

The simplest and most popular tool, called the Monte Carlo simulation, was described by David Hertz in a 1964 HBR article and popularized in financial circles by sophisticated users like Merck CFO and executive vice president Judy Lewent. Today, spreadsheet-based Monte Carlo simulation software is widely available and is being used in fields as diverse as petroleum exploration, financial engineering, defense, banking, and retirement portfolio planning. Wells Fargo Bank, for instance, used a Monte Carlo simulation to predict the cost of offering customers a variable-rate CD, whose return would increase if interest rates rose. A previous estimate based on three years of 1990s interest-rate data had shown that the cost would be about 0.10% for a five-year CD. But the Monte Carlo simulation, which combined interest-rate data going back to 1965 with models of customer behavior, found that the bank's cost could be eight times that amount. The alarming finding induced the bank to reconfigure its CD product to reduce the chance of unacceptable costs should interest rates rise.

Had the average-obsessed boss at HealthCeuticals used Monte Carlo simulation, he would have seen not only that the average inventory cost was not zero but that he shouldn't have been stocking 5,000 units in the first place. For executives like him who are still fond of single values, it's time for a shift in mind-set. Rather than "Give me a number for my report," what every executive should be saying is "Give me a distribution for my simulation."

~~4 Ways How to improve Inventory Management using AI~~

~~Biggest issues of inventory management and supply chain~~

- ~~**An overabundance of data**. Data inventory management becomes a tedious process due to how much of it is usually accumulated by any company. While carefully built and supported inventory management software can track and store data, it would still require tremendous efforts from the human staff to process all of it.~~
- ~~**Tracking issues**. It is getting harder to track every item in the inventory and to get relevant analysis out of it. Failure to follow every incoming and outgoing item can have~~

devastating effects on turnover, an adequate response to increasing or decreasing demand, timely delivery decisions, etc.

- **Difficulties with business planning**. Inventory management is a big part of successful planning for companies in any industry, especially the ones that deal with CPGs. Successful growth methodologies require real-time collection and processing of all data, and according to decisions and responses to situations.
- **High operational budgets**. As the size of inventory grows, the company struggles more with maintaining the cost-effectiveness across the branch. Whether it be the delivery service, analytical, and data mining teams or staff that keeps track of stored items, the funding increases alongside the size of the company. And often these numbers are far from being acceptable.
- **IT inventory management**. Keeping every IT asset within an organization requires thorough control. If done wisely, it will bring benefits to a company and reduce expenses on the assets. AI inventory management systems help companies to run this process properly.

4 ways to improve inventory management using AI

1. Dealing with planning, predictions, and modeling in the inventory management process

Inventory management is far beyond storing and delivering items from one place to another. It both generates and relies on substantial amounts of data to be effective in terms of time, money, and the workforce.

Overstocking and understocking are usually caused by failure to see and respond to the change in demand for a specific product. The **ability to predict** that, as a company, requires extremely competent analysts and experts in business modeling. There are other complicating factors, as more than one warehouse, location-specific demand, and more. Each product niche has its specifics that managers and analysts have to adapt to.

Artificial intelligence is capable of **providing insights** that were previously unavailable. It can take care of inventory management models, created for inventory regulations, with the help of advanced AI inventory models, which provide a solid control of the operations by

making and acting on the predictions. Moreover, AI can analyze more than 50 elements, which is vital for successful planning, stocking, and scheduling deliveries.

2. Data mining

An AI tech is showing serious competence in transforming the data into timely actions that can help the business to evolve or respond well to a particular situation. Say, a local hockey team progressed in the Stanley Cup, and there will be a much bigger demand for beverages and snacks in the next matchday in that specific region. Artificial Intelligence can analyze that and come up with a suggestion to overstock for that particular region.
Trained intelligence acts as a supervisor and a watchdog: weather conditions, events, economic situation, how trendy is this or that product, and much more. It can take the decisions made by the management from error-prone assumptions (even though backed up by data) to almost a guaranteed result. While, obviously, significantly reducing the workload of the said staff.

3. Stocking management and fulfillment

Inventory management has a significant impact on customer satisfaction and a sense of fulfillment. Planning errors or inadequate stock monitoring in any warehouse can lead to shortages and delays, negatively impacting the revenue.

As explained in previous sections, AI is already competent in analyzing customer behavior patterns and a big number of other factors that help to plan the stocking right. Furthermore, a well-trained intelligence can automate the process of stocking and increase the efficiency in delivery, suggesting the best routes.

AI inventory management minimizes the risks of mismanaging the stocking process and helps respond to the customer demand accordingly. With the insights provided by data mining, and AI can also help to establish efficient factory-to-warehouse transportation, which is extremely important for more volatile products that have shorter expiry time.

4. AI-based robotics

Robots are not a new thing on the market. Companies like Amazon are already using them in their day-to-day logistical tasks, and there's a number of benefits that put robots over human staff, especially in routine operations.

~~Robots can tirelessly move items around the warehouses, locate wares and scan their conditions. The machines can work around the clock and with more optimal time per action. This alone can save a big chunk of the operational budget and allows to allocate more employees to more urgent and vital tasks that require human cognition.~~
~~Artificial Intelligence can enhance the process further. United with an intelligence that is capable of analyzing the data, predicting the demand patterns, and suggesting optimal delivery routes, it becomes a potent tool that can completely automate internal processes in any warehouse.~~

~~https://medium.com/@inverita/4-ways-how-to-improve-inventory-management-using-ai-ee066fcbe529~~

See more for Monte Carlo Simulation :

- https://medium.com/towards-artificial-intelligence/monte-carlo-simulation-an-in-depth-tutorial-with-python-bcf6eb7856c8
- https://towardsdatascience.com/understanding-monte-carlo-simulation-eceb4c9cad4
- https://towardsdatascience.com/optimise-your-todo-list-with-monte-carlo-simulations-in-python-1682a8a5eb84
- https://towardsdatascience.com/using-a-monte-carlo-simulation-to-forecast-extreme-weather-events-d17671149d3e
- https://towardsdatascience.com/best-investment-portfolio-via-monte-carlo-simulation-in-python-53286f3fe93