This project is from the Codecademy Machine Learning career path.

### 18.3.5 A/B Testing at Nosh Mish Mosh

The Nosh Mish Mosh is a recipe and ingredient meal delivery service. They ship the raw materials and you get to cook them at your home! They’ve decided to hire a data analyst to help make product and interface decisions. Get started to help them figure out the amount of data they’ll need to make meaningful decisions.

## Nosh Mish Mosh: An Assortment of Edible Aliments

**1.**

We’ve collected customer data for the past week and exposed it through a Python library, so first import noshmishmosh.

**2.**

Next, we’ll need to do a little bit of data analysis — let’s use numpy to help. Import numpy into your workspace as np.

## A/B Testing at Nosh Mish Mosh

**3.**

Nosh Mish Mosh wants to run an experiment to see if we can convince more people to purchase meal plans if we use a more artisanal-looking vegetable selection. We’ve photographed these modern meals with blush tomatoes and graffiti eggplants, but aren’t sure if this strategy will sell enough units to benefit from establishing a business relationship with a new provider.

Before running this experiment, of course, we need to know the sample size that will be required to detect the difference we are hoping for. There are three things we need to know before we can determine that number.

* + the **Baseline Conversion Rate**
  + **Minimum Detectable Effect** (desired lift)
  + and the **Statistical Significance Threshold**

**4.**

Let’s get the ball rolling on finding those numbers! In order to get our baseline, we need to first know how many users visit the site in a typical week. Let’s grab that logged information, which is stored in noshmishmosh.customer\_visits. Assign that to a new variable called all\_visitors.

**5.**

Next we need to know how many visitors to the site ultimately end up buying a meal or set of meals in a typical week. We have that information saved into purchasing\_customers field on noshmishmosh. Save that information into a variable called paying\_visitors.

**6.**

Calculate the lengths of the two lists, saving the results into variables called total\_visitor\_count and paying\_visitor\_count, respectively.

**7.**

Now to get the baseline: Divide the number of purchasing visitors by the number of total visitors. Save the result in a variable called baseline\_percent. Since we want a percentage as our answer, multiply the result by 100.0.

**8.**

Print out the baseline\_percent so we know what to use for our baseline percentage in the A/B Sample Size Calculator.

## Mish Mosh B'Gosh: The Effect Size

**9.**

These rainbow fingerling potatoes don’t come cheap. We’d like to know for sure that, with this change, we’ll be pulling in at least $1240 more every week. In order to figure out how many more customers we need, we’ll have to investigate the average revenue generated from a given sale. Luckily we have a list of the money spent by each customer in a typical week: noshmishmosh.money\_spent. Save that list into a variable called payment\_history.

**10.**

We need to find how many purchases it would take to reach $1240 in additional revenue using our historical data.

Let’s start with computing the average payment per paying customer using np.mean, saving it as average\_payment.

**11.**

We want to know how many of these “usual” payments it would take to clear our $1240 mark. Round the number up using np.ceil (because that’s how many new customers it takes to bring in more than $1240). Save that value into a new\_customers\_needed variable.

**12.**

Now find the additional percent of weekly visitors who must make a purchase in order to make this change worthwhile. Do this by dividing the number of customers by the total visitor count for a typical week (calculated earlier), and multiplying by 100. Save the result in a variable called percentage\_point\_increase. Print percentage\_point\_increase to see what it is.

**13.**

In order to find our minimum detectable effect/desired lift, we need to express percentage\_point\_increase as a percent of baseline\_percent. You can do this by dividing percentage\_point\_increase by baseline\_percent and multiplying by 100.0.

Store the results in a variable called mde.

**14.**

Print out the result mde.

## Nosh Mish Mosh: Tying It All Together

**15.**

The last thing we need to calculate the sample size for Nosh Mish Mosh’s artisanal rebranding is our statistical significance threshold. We’d like to be fairly certain, but this isn’t going to be a million dollar decision, so let’s go with 10%.

**16.**

Now put it all together! Punch the baseline, the minimum detectable effect, and the statistical significance threshold into the calculator and evaluate how many people need to be shown the new assets before we can check if the results are a significant improvement. Save the results in a variable called ab\_sample\_size.

I created my own solution and compared it to the one generated with Perplexity. Here is my solution first.

# Step 1

import noshmishmosh

# Step 2

import numpy as np

# Step 4

all\_visitors = noshmishmosh.customer\_visits

# Step 5

paying\_visitors = noshmishmosh.purchasing\_customers

# Step 6

total\_visitor\_count = len(all\_visitors)

paying\_visitor\_count = len(paying\_visitors)

# Step 7: Percent of customers.

baseline\_percent = paying\_visitor\_count / total\_visitor\_count \* 100

# Step 8

print(f"Baseline Conversion Rate: {baseline\_percent:.2f}%")

# Step 9

payment\_history = noshmishmosh.money\_spent

# Step 10

average\_payment = np.mean(payment\_history)

# Step 11

new\_customers\_needed = np.ceil(1240 / average\_payment)

# Step 12: The total number or visitors does not change. The additional customers come from the same pool of total visitors. Hence, this is the number of visitors that need to be converted to paying visitors, i.e., customers.

percentage\_point\_increase = new\_customers\_needed / total\_visitor\_count \* 100

print(f"Percentage Point Increase: {percentage\_point\_increase:.2f}%")

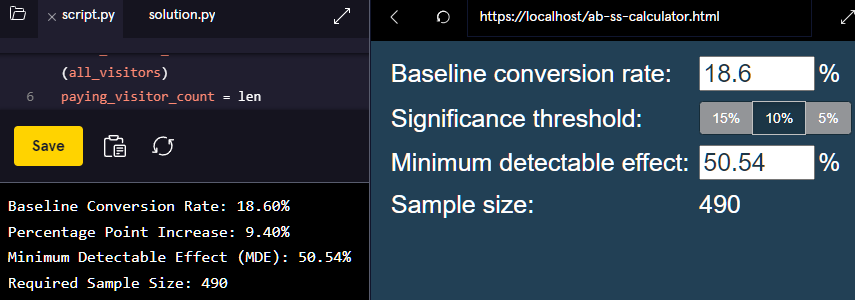
# Step 13

mde = percentage\_point\_increase / baseline\_percent \* 100

# Step 14

print(f"Minimum Detectable Effect (MDE): {mde:.2f}%")

# Step 16 Using the provided baseline calculator



ab\_sample\_size =  490

print(f"Required Sample Size: {ab\_sample\_size}")

This solution has been generated with Perplexity.

*# 1.*

**import** noshmishmosh

*# 2.*

**import** numpy **as** np

*# 4.*

all\_visitors = noshmishmosh.customer\_visits

*# 5.*

paying\_visitors = noshmishmosh.purchasing\_customers

*# 6.*

total\_visitor\_count = len(all\_visitors)

paying\_visitor\_count = len(paying\_visitors)

*# 7.*

baseline\_percent = (paying\_visitor\_count / total\_visitor\_count) \* 100.0

*# 8.*

**print**(f"Baseline Conversion Rate: {baseline\_percent:.2f}%")

*# 9.*

payment\_history = noshmishmosh.money\_spent

*# 10.*

average\_payment = np.mean(payment\_history)

*# 11.*

new\_customers\_needed = np.ceil(1240 / average\_payment)

*# 12.*

percentage\_point\_increase = (new\_customers\_needed / total\_visitor\_count) \* 100

**print**(f"Percentage Point Increase: {percentage\_point\_increase:.2f}%")

*# 13.*

mde = (percentage\_point\_increase / baseline\_percent) \* 100.0

*# 14.*

**print**(f"Minimum Detectable Effect (MDE): {mde:.2f}%")

*# 15.*

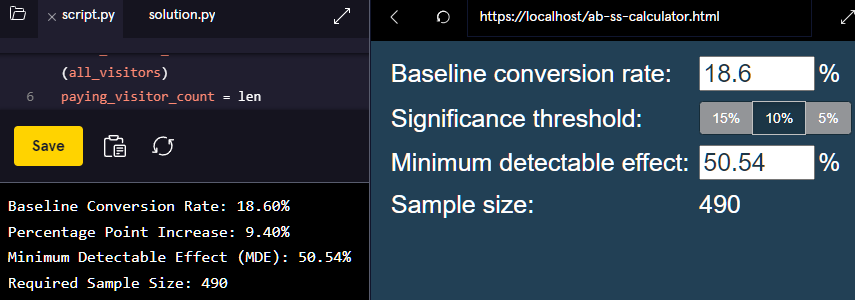
statistical\_significance\_threshold = 10 *# 10%*

*# 16.*

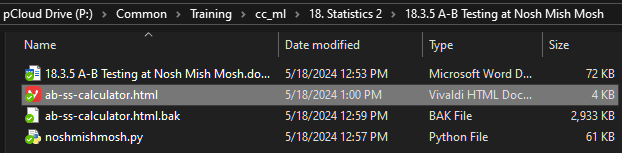
*# Assuming you have access to an A/B sample size calculator function*

ab\_sample\_size = calculate\_sample\_size(baseline\_percent, mde, statistical\_significance\_threshold)

In # 16 I am using the sample size calculator provided in Codecademy:



I saved the baseline calculator locally.



**print**(f"Required Sample Size: {ab\_sample\_size}")