```
# All Libraries required for this lab are listed below. The libraries pre-installed on Skills Network Labs are commented.
# !pip install -qy pandas==1.3.4 numpy==1.21.4 seaborn==0.9.0 matplotlib==3.5.0 scikit-learn==0.20.1
# - Update a specific package
# !pip install pmdarima -U
# - Update a package to specific version
# !pip install --upgrade pmdarima==2.0.2
# Note: If your environment doesn't support "!pip install", use "!mamba install"
```

```
!pip install tqdm pmdarima
```

```
Requirement already satisfied: tqdm in /usr/local/lib/python3.11/dist-packages (4.67.1)
Collecting pmdarima
  Downloading pmdarima-2.0.4-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.manylinux_2_28_x86_64.whl.metadata (7.8 kB)
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.11/dist-packages (from pmdarima) (1.4.2)
Requirement already satisfied: Cython!=0.29.18,!=0.29.31,>=0.29 in /usr/local/lib/python3.11/dist-packages (from pmdarima) (3.0.12)
Requirement already satisfied: numpy>=1.21.2 in /usr/local/lib/python3.11/dist-packages (from pmdarima) (2.0.2)
Requirement already satisfied: pandas>=0.19 in /usr/local/lib/python3.11/dist-packages (from pmdarima) (2.2.2)
Requirement already satisfied: scikit-learn>=0.22 in /usr/local/lib/python3.11/dist-packages (from pmdarima) (1.6.1)
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.11/dist-packages (from pmdarima) (1.14.1)
Requirement already satisfied: statsmodels>=0.13.2 in /usr/local/lib/python3.11/dist-packages (from pmdarima) (0.14.4)
Requirement already satisfied: urllib3 in /usr/local/lib/python3.11/dist-packages (from pmdarima) (2.3.0)
Requirement already satisfied: setuptools!=50.0.0,>=38.6.0 in /usr/local/lib/python3.11/dist-packages (from pmdarima) (75.2.0)
Requirement already satisfied: packaging>=17.1 in /usr/local/lib/python3.11/dist-packages (from pmdarima) (24.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas>=0.19->pmdarima) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas>=0.19->pmdarima) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas>=0.19->pmdarima) (2025.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn>=0.22->pmdarima) (3.6.
Requirement already satisfied: patsy>=0.5.6 in /usr/local/lib/python3.11/dist-packages (from statsmodels>=0.13.2->pmdarima) (1.0.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas>=0.19->pmdarima)
Downloading pmdarima-2.0.4-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.manylinux_2_28_x86_64.whl (2.2 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.2/2.2 MB 18.0 MB/s eta 0:00:00
Installing collected packages: pmdarima
Successfully installed pmdarima-2.0.4
```

```
!pip install skillsnetwork
from tqdm import tqdm
import skillsnetwork
import numpy as np
import pandas as pd
#from itertools import accumulate
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.stats.api as sms
%matplotlib inline
from math import ceil

# You can also use this section to suppress warnings generated by your code:
def warn(*args, **kwargs):
    pass
import warnings
warnings.warn = warn
warnings.filterwarnings('ignore')

sns.set_context('notebook')
sns.set_style('white')
```

```
Requirement already satisfied: terminado>=0.8.3 in /usr/local/lib/python3.11/dist-packages (from notebook>=4.4.1->widgetsnbextension~
Requirement already satisfied: prometheus-client in /usr/local/lib/python3.11/dist-packages (from notebook>=4.4.1->widgetsnbextension
Requirement already satisfied: nbclassic>=0.4.7 in /usr/local/lib/python3.11/dist-packages (from notebook>=4.4.1->widgetsnbextension~
Requirement already satisfied: platformdirs>=2.5 in /usr/local/lib/python3.11/dist-packages (from jupyter-core>=4.6.0->jupyter-client
Requirement already satisfied: notebook-shim>=0.2.3 in /usr/local/lib/python3.11/dist-packages (from nbclassic>=0.4.7->notebook>=4.4.
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.11/dist-packages (from nbconvert>=5->notebook>=4.4.1->widgets
Requirement already satisfied: bleach!=5.0.0 in /usr/local/lib/python3.11/dist-packages (from bleach[css]!=5.0.0->nbconvert>=5->noteb
Requirement already satisfied: defusedxml in /usr/local/lib/python3.11/dist-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbex
Requirement already satisfied: jupyterlab-pygments in /usr/local/lib/python3.11/dist-packages (from nbconvert>=5->notebook>=4.4.1->wi
Requirement already satisfied: markupsafe>=2.0 in /usr/local/lib/python3.11/dist-packages (from nbconvert>=5->notebook>=4.4.1->widget
Requirement already satisfied: mistune<4,>=2.0.3 in /usr/local/lib/python3.11/dist-packages (from nbconvert>=5->notebook>=4.4.1->widg
Requirement already satisfied: nbclient>=0.5.0 in /usr/local/lib/python3.11/dist-packages (from nbconvert>=5->notebook>=4.4.1->widget
Requirement already satisfied: pandocfilters>=1.4.1 in /usr/local/lib/python3.11/dist-packages (from nbconvert>=5->notebook>=4.4.1->w
Requirement already satisfied: fastjsonschema>=2.15 in /usr/local/lib/python3.11/dist-packages (from nbformat->notebook>=4.4.1->widge
Requirement already satisfied: jsonschema>=2.6 in /usr/local/lib/python3.11/dist-packages (from nbformat->notebook>=4.4.1->widgetsnbe
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.1->jupyter-client>=6.1.12
Requirement already satisfied: argon2-cffi-bindings in /usr/local/lib/python3.11/dist-packages (from argon2-cffi->notebook>=4.4.1->wi
Requirement already satisfied: webencodings in /usr/local/lib/python3.11/dist-packages (from bleach!=5.0.0->bleach[css]!=5.0.0->nbcon
Requirement already satisfied: tinycss2<1.5,>=1.1.0 in /usr/local/lib/python3.11/dist-packages (from bleach[css]!=5.0.0->nbconvert>=5
Requirement already satisfied: attrs>=22.2.0 in /usr/local/lib/python3.11/dist-packages (from jsonschema>=2.6->nbformat->notebook>=4.
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in /usr/local/lib/python3.11/dist-packages (from jsonschema>=2.6-
Requirement already satisfied: referencing>=0.28.4 in /usr/local/lib/python3.11/dist-packages (from jsonschema>=2.6->nbformat->notebo
Requirement already satisfied: rpds-py>=0.7.1 in /usr/local/lib/python3.11/dist-packages (from jsonschema>=2.6->nbformat->notebook>=4
Requirement already satisfied: jupyter-server<3,>=1.8 in /usr/local/lib/python3.11/dist-packages (from notebook-shim>=0.2.3->nbclassi
Requirement already satisfied: cffi>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from argon2-cffi-bindings->argon2-cffi->notebo
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.11/dist-packages (from beautifulsoup4->nbconvert>=5->notebook>
Requirement already satisfied: typing-extensions>=4.0.0 in /usr/local/lib/python3.11/dist-packages (from beautifulsoup4->nbconvert>=5
Requirement already satisfied: pycparser in /usr/local/lib/python3.11/dist-packages (from cffi>=1.0.1->argon2-cffi-bindings->argon2-c
Requirement already satisfied: anyio>=3.1.0 in /usr/local/lib/python3.11/dist-packages (from jupyter-server<3,>=1.8->notebook-shim>=0
Requirement already satisfied: websocket-client in /usr/local/lib/python3.11/dist-packages (from jupyter-server<3,>=1.8->notebook-shi
Requirement already satisfied: sniffio>=1.1 in /usr/local/lib/python3.11/dist-packages (from anyio>=3.1.0->jupyter-server<3,>=1.8->no
Downloading skillsnetwork-0.21.10-py3-none-any.whl (26 kB)
Downloading jedi-0.19.2-py2.py3-none-any.whl (1.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.6/1.6 MB 20.9 MB/s eta 0:00:00
Installing collected packages: jedi, skillsnetwork
Successfully installed jedi-0.19.2 skillsnetwork-0.21.10
```

```
await skillsnetwork.download_dataset('https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMSkillsNetwork-GPXX022OEN/ab_data
df = pd.read_csv('ab_data.csv')
```

```
Downloading ab_data.csv: 100%                                    15901933/15901933 [00:02<00:00, 14655543.92it/s]
Saved as 'ab_data.csv'
```

```
df.sample(5)
```

	user_id	timestamp	group	landing_page	converted
268912	808944	2017-01-11 05:04:23.830118	treatment	new_page	0
174134	890133	2017-01-16 15:51:27.007177	treatment	new_page	0
191608	832333	2017-01-16 02:48:47.063543	treatment	new_page	0
383	808004	2017-01-10 15:45:10.577418	treatment	new_page	0
179812	679983	2017-01-13 16:41:57.944379	treatment	new_page	0

```
df['version'] = np.where(df['landing_page'] == 'new_page', 'dark_mode', 'light_mode')
df.head(5)
```

	user_id	timestamp	group	landing_page	converted	version
0	851104	2017-01-21 22:11:48.556739	control	old_page	0	light_mode
1	804228	2017-01-12 08:01:45.159739	control	old_page	0	light_mode
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0	dark_mode
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0	dark_mode
4	864975	2017-01-21 01:52:26.210827	control	old_page	1	light_mode

```
df['group'].value_counts()
```

group	count
treatment	147276
control	147202

dtype: int64

```
df['landing_page'].value_counts()
```

landing_page	count
old_page	147239
new_page	147239

dtype: int64

```
df['landing_page'].value_counts()
```

landing_page	count
old_page	147239
new_page	147239

dtype: int64

```
# filter the data based off of the version (dark or light mode)
old_conversion = df[df['version'] == 'light_mode']
new_conversion = df[df['version'] == 'dark_mode']

# get the conversion rates
light_converted = old_conversion['converted'].mean()
dark_converted = new_conversion['converted'].mean()

# filter the data based off of the version (dark or light mode)
old_conversion = df[df['version'] == 'light_mode']
new_conversion = df[df['version'] == 'dark_mode']

# get the conversion rates
light_converted = old_conversion['converted'].mean()
dark_converted = new_conversion['converted'].mean()

# print the results
print("The conversion rate in the group using light mode is: %.2f%%" % (100 * light_converted))
print("The conversion rate in the group using dark mode is: %.2f%%" % (100 * dark_converted))
```

```
The conversion rate in the group using light mode is: 12.05%
The conversion rate in the group using dark mode is: 11.88%
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 294478 entries, 0 to 294477
Data columns (total 6 columns):
 #   Column        Non-Null Count   Dtype
---  ------        --------------   -----
 0   user_id       294478 non-null  int64
 1   timestamp     294478 non-null  object
 2   group         294478 non-null  object
 3   landing_page  294478 non-null  object
 4   converted     294478 non-null  int64
 5   version       294478 non-null  object
dtypes: int64(2), object(4)
memory usage: 13.5+ MB
```

```
# check if some users appear multiple times
user_sessions = df['user_id'].value_counts()
```

```
multiple_times_user = user_sessions[user_sessions > 1].count()

multiple_times_user
```

```
np.int64(3894)
```

```
dr = user_sessions[user_sessions > 1].index
df = df[~df['user_id'].isin(dr)]

df.head(5)
```

	user_id	timestamp	group	landing_page	converted	version
0	851104	2017-01-21 22:11:48.556739	control	old_page	0	light_mode
1	804228	2017-01-12 08:01:45.159739	control	old_page	0	light_mode
2	661590	2017-01-11 16:55:06.154213	treatment	new_page	0	dark_mode
3	853541	2017-01-08 18:28:03.143765	treatment	new_page	0	dark_mode
4	864975	2017-01-21 01:52:26.210827	control	old_page	1	light_mode

```
df.shape[0]
```

```
286690
```

```
effect_size = sms.proportion_effectsize(0.13, 0.15)

sample_size = sms.NormalIndPower().solve_power(
    effect_size,
    power = 0.8,
    alpha = 0.05,
    ratio = 1
)

sample_size = ceil(sample_size)

sample_size
```

```
4720
```

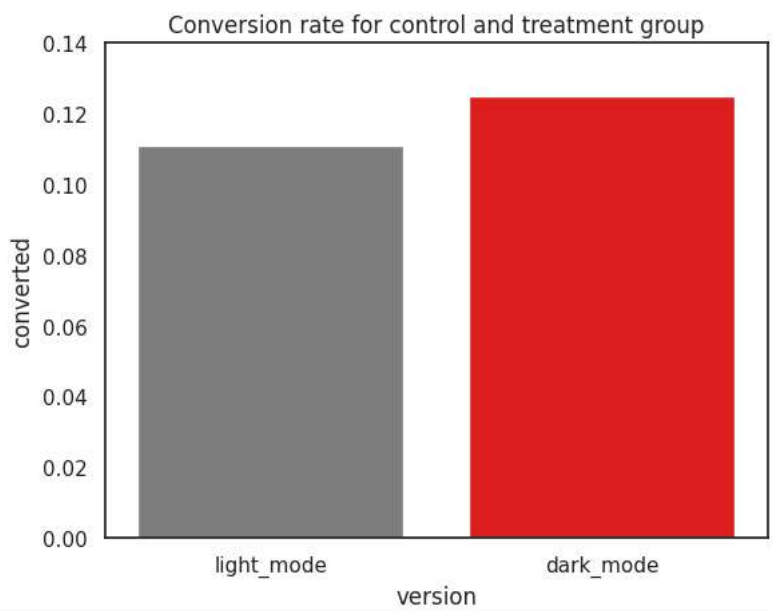
```
# The treatment and control samples
trt_sample = df[df['group']=='treatment'].sample(n=sample_size, random_state=888)

con_sample = df[df['group'] == 'control'].sample(n=sample_size, random_state=0)

# Combining into one dataframe and resetting the indices
df = pd.concat([con_sample, trt_sample], axis=0)
df.reset_index(drop=True, inplace=True)

df.sample(5)
```

	user_id	timestamp	group	landing_page	converted	version
1715	733274	2017-01-03 00:53:38.615763	control	old_page	0	light_mode
3109	839019	2017-01-08 18:31:13.183988	control	old_page	0	light_mode
5312	654782	2017-01-10 23:13:46.408952	treatment	new_page	1	dark_mode
6609	909312	2017-01-19 21:32:19.238471	treatment	new_page	0	dark_mode
3318	674441	2017-01-09 09:31:39.235924	control	old_page	0	light_mode

```
df.groupby(['group','version']).agg({'converted': 'mean'})
```

		converted
control	**light_mode**	0.111017
treatment	**dark_mode**	0.125000

```
sns.barplot(x = df['version'], y = df['converted'], palette = ['gray', 'red'], ci = False)
plt.ylim(0, 0.14)
plt.title('Conversion rate for control and treatment group')
plt.show()
```




```
conv_cont = df[df['group'] =='control']['converted']

conv_trt = df[df['group'] =='treatment']['converted']

n_cont = conv_cont.count()

n_trt = conv_trt.count()

num_converted = [conv_cont.sum(), conv_trt.sum()]

nobs = [n_cont, n_trt]

# p-value?
z_stat, pval = sms.proportions_ztest(num_converted, nobs=nobs)

pval
```

```
np.float64(0.03524195278525257)
```

```
# Downloading the dataset
await skillsnetwork.download_dataset('https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IND-GPXX0K4XEN/marketing_AB.csv')
marketing = pd.read_csv('marketing_AB.csv')
marketing.sample(5)
```

```
Downloading marketing_AB.csv: 100%  21980992/21980992 [00:02<00:00, 22041976.21it/s]
Saved as 'marketing_AB.csv'
```

	Unnamed: 0	user id	test group	converted	total ads	most ads day	most ads hour
350027	350027	1189077	ad	False	15	Tuesday	11
320382	320382	1531881	ad	False	1	Monday	19
588007	588007	1308514	ad	False	1	Tuesday	22
76478	76478	1374028	ad	False	90	Friday	18
445914	445914	1535199	ad	False	12	Sunday	15

```
# remove the first column
marketing = marketing.drop('Unnamed: 0', axis=1)

# check if some users appear multiple times
user_sess = marketing['user id'].value_counts()

dup_users = user_sess[user_sess > 1].count()
```

```
dup_users

# The following line was causing the error, remove it:
# marketing = marketing.drop('Unnamed: 0', axis=1)

# Display the marketing DataFrame:
marketing
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-29-7dd6a570a540> in <cell line: 0>()
      1 # remove the first column
----> 2 marketing = marketing.drop('Unnamed: 0', axis=1)
      3
      4 # check if some users appear multiple times
      5 user_sess = marketing['user id'].value_counts()

                              3 frames
/usr/local/lib/python3.11/dist-packages/pandas/core/indexes/base.py in drop(self, labels, errors)
   7068         if mask.any():
   7069             if errors != "ignore":
-> 7070                 raise KeyError(f"{labels[mask].tolist()} not found in axis")
   7071             indexer = indexer[~mask]
   7072         return self.delete(indexer)

KeyError: "['Unnamed: 0'] not found in axis"
```

Next steps: Explain error

```
# remove the first column if it exists
if 'Unnamed: 0' in marketing.columns:
    marketing = marketing.drop('Unnamed: 0', axis=1)

# check if some users appear multiple times
user_sess = marketing['user id'].value_counts()

dup_users = user_sess[user_sess > 1].count()

dup_users

# Display the marketing DataFrame:
marketing
```

	user id	test group	converted	total ads	most ads day	most ads hour
0	1069124	ad	False	130	Monday	20
1	1119715	ad	False	93	Tuesday	22
2	1144181	ad	False	21	Tuesday	18
3	1435133	ad	False	355	Tuesday	10
4	1015700	ad	False	276	Friday	14
...	...	...	...	...	...	...
588096	1278437	ad	False	1	Tuesday	23
588097	1327975	ad	False	1	Tuesday	23
588098	1038442	ad	False	3	Tuesday	23
588099	1496395	ad	False	1	Tuesday	23
588100	1237779	ad	False	1	Tuesday	23

588101 rows × 6 columns

```
effect = sms.proportion_effectsize(0.1, 0.15)

sample_size = sms.NormalIndPower().solve_power(
    effect,
    power = 0.8,
    alpha = 0.05,
    ratio = 1
)

sample_size = ceil(sample_size)
```

```
sample_size = ceil(sample_size)

sample_size
```

```
681
```

```
# sms.sms.proportions_ztest(num_converted, nobs=nobs) gives us two values: the z statistics score, and the p-value
# (This line is a comment and doesn't need correction)
```

```
converted_con = marketing[marketing['test group'] == 'ad']['converted']

converted_trt = marketing[marketing['test group'] == 'psa']['converted']

n_control = converted_con.count()

n_treatment = converted_trt.count()

num_converted = [converted_con.sum(), converted_trt.sum()]

nobs = [n_control, n_treatment]

# p-value?
z_stat, pval = sms.proportions_ztest(num_converted, nobs=nobs)

pval
```

```
np.float64(1.7052807161559727e-13)
```

Start coding or generate with AI.

Start coding or generate with AI.