# A/B Testing and Udacity Website

Yuanjing Zhu
Netid: yz792

In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

# Exercise 1: load the data

In [ ]:
```
df_treatment = pd.read_csv('https://github.com/nickeubank/MIDS_Data/blob/master'
                           '/udacity_AB_testing/experiment_data.csv?raw=true')
df_control = pd.read_csv('https://github.com/nickeubank/MIDS_Data/blob/master'
                         '/udacity_AB_testing/control_data.csv?raw=true')
```

# Exercise 2: explore the data

In [ ]:
```
df_treatment.sample(5)
```

Out[ ]:

| | Date | Pageviews | Clicks | Enrollments | Payments |
|---|---|---|---|---|---|
| **4** | Wed, Oct 15 | 9793 | 832 | 140.0 | 94.0 |
| **31** | Tue, Nov 11 | 9931 | 831 | NaN | NaN |
| **5** | Thu, Oct 16 | 9500 | 788 | 129.0 | 61.0 |
| **13** | Fri, Oct 24 | 9402 | 697 | 194.0 | 94.0 |
| **27** | Fri, Nov 7 | 9272 | 767 | NaN | NaN |

In [ ]:
```
df_control.sample(5)
```

Out[ ]:

| | Date | Pageviews | Clicks | Enrollments | Payments |
|---|---|---|---|---|---|
| **20** | Fri, Oct 31 | 8890 | 706 | 174.0 | 101.0 |
| **15** | Sun, Oct 26 | 8896 | 708 | 161.0 | 104.0 |
| **35** | Sat, Nov 15 | 8630 | 743 | NaN | NaN |
| **8** | Sun, Oct 19 | 8459 | 691 | 131.0 | 60.0 |
| **34** | Fri, Nov 14 | 9192 | 735 | NaN | NaN |

In [ ]:
```
print(f"The shape of the treatment dataframe is {df_treatment.shape}.")
```

```
print(f"The shape of the control dataframe is {df_control.shape}.")
```

```
The shape of the treatment dataframe is (37, 5).
The shape of the control dataframe is (37, 5).
```

Unit of observation: website traffic and payment information of Udacity website in a single day.

# Exercise 3: stack into a single dataframe

In [ ]:
```
df_treatment.loc[:,'treatment'] = 1
df_control.loc[:,'treatment'] = 0
df_all = pd.concat([df_treatment, df_control]).reset_index(drop=True)
df_all.head()
```

Out[ ]:

| | Date | Pageviews | Clicks | Enrollments | Payments | treatment |
|---|---|---|---|---|---|---|
| **0** | Sat, Oct 11 | 7716 | 686 | 105.0 | 34.0 | 1 |
| **1** | Sun, Oct 12 | 9288 | 785 | 116.0 | 91.0 | 1 |
| **2** | Mon, Oct 13 | 10480 | 884 | 145.0 | 79.0 | 1 |
| **3** | Tue, Oct 14 | 9867 | 827 | 138.0 | 92.0 | 1 |
| **4** | Wed, Oct 15 | 9793 | 832 | 140.0 | 94.0 | 1 |

# Exercise 4

- What outcome are they hoping will be impacted by their manipulation? / What is your Overall Evaluation Criterion (OEC)?

They are hoping to increase the percentage of students who enroll in the free trial, and ultimately buy and complete the course.

Therefore, the OEC is **(enrollment-payment)/click counts**

# Exercise 5: Sanity Checks

- Given Udacity's goals, what outcome are they hoping will not be impacted by their manipulation?

1. **The number of pageviews**

The number of pageviews should not be impacted by the manipulation, because the users view the page before they click on the "Start free trial" button and asked questions about time commitment.

2. **The number of clicks on the "Start free trial" button**

The number of clicks on the "Start free trial" button should not be impacted by the manipulation, because the question where the user is asked how much time they had available to devote to the course is asked after the user clicks on the "Start free trial" button.

# Exercise 6: Calculate the average number of pageviews for the treated group and for the control group.

In [ ]:
```
pageview_treatment = df_all.loc[df_all['treatment']==1, 'Pageviews']
avg_pv_treatment = pageview_treatment.mean()
pageview_control = df_all.loc[df_all['treatment']==0, 'Pageviews']
avg_pv_control = pageview_control.mean()
print(f"The average number of pageviews for the treatment group is \
      {round(avg_pv_treatment)}.")
print(f"The average number of pageviews for the control group is \
      {round(avg_pv_control)}.")
```

```
The average number of pageviews for the treatment group is 9315.
The average number of pageviews for the control group is 9339.
```

The average number of pageviews for the treatment group is 9315, and the average number of pageviews for the control group is 9339, which **is very close to each other**.

# Exercise 7: use a ttest to test the statistical significance of the differences

In [ ]:
```
from scipy.stats import ttest_ind
_, pvalue = ttest_ind(pageview_treatment, pageview_control)
print(f"The p-value for the difference in pageviews between \
      the treatment and control groups is {pvalue:.2f}.")
if pvalue < 0.05:
    print("It is statistically significant.")
else:
    print("It is not statistically significant.")
```

```
The p-value for the difference in pageviews between the treatment and control groups
is 0.89.
It is not statistically significant.
```

The p-value from t-test is 0.89 (>0.05), which means that the difference in pageviews between the two groups is **not statistically significant**.

# Exercise 8: What other measure is pre-treatment?

According to the description of the experiemnt and my answer to the question in Exercise 5, the number of clicks on the "Start free trial" button is another measure that is pre-treatment.

# Exercise 9: Check if the other pre-treatment variable is also balanced.

In [ ]:
```
_, pvalue = ttest_ind(df_all.loc[df_all['treatment']==1, 'Clicks'],\
                       df_all.loc[df_all['treatment']==0, 'Clicks'])
print(f"The p-value for the difference in clicks between the treatment \
      and control groups is {pvalue:.2f}.")
if pvalue < 0.05:
    print("It is statistically significant.")
else:
    print("It is not statistically significant.")
```

```
The p-value for the difference in clicks between the treatment and control groups is
0.93.
It is not statistically significant.
```

Since the p-value from t-test is 0.93 (>0.05), the difference in the number of clicks on the "Start free trial" button between the two groups is **not statistically significant**.

# Exercise 10: Test whether the OEC and the metric you don't want affected have different average values in the control group and treatment group.

In [ ]:
```
print("(enrollment-payment)/click:")
df_all['delta_per_click'] = (df_all['Enrollments'] - df_all['Payments'])\
      /df_all['Clicks']
avg_delta_per_click_treatment = df_all.loc[df_all['treatment']==1, \
                                          'delta_per_click'].mean()
avg_delta_per_click_control = df_all.loc[df_all['treatment']==0, \
                                        'delta_per_click'].mean()
print(f"The average difference between enrollment and payment per click \
      for the treatment group is {round(avg_delta_per_click_treatment, 2)}.")
print(f"The average difference between enrollment and payment per click\
      for the control group is {round(avg_delta_per_click_control, 2)}.")
_, pvalue = ttest_ind(df_all.loc[df_all['treatment']==1, 'delta_per_click'], \
                      df_all.loc[df_all['treatment']==0, 'delta_per_click'],\
                        nan_policy='omit')
print(f"The p-value for the difference in OEC between the \
      treatment and control groups is {pvalue:.2f}.")
if pvalue < 0.05:
    print("It is statistically significant.")
else:
    print("It is not statistically significant.")
```

```
(enrollment-payment)/click:
The average difference between enrollment and payment per click for the treatment gro
up is 0.09.
The average difference between enrollment and payment per clickfor the control group
is 0.1.
The p-value for the difference in OEC between the treatment and control groups is 0.1
3.
It is not statistically significant.
```

NOTE: there are missing values in the column, I added "nan_policy='omit'" in the ttest_ind function to ignore them.

The p-value of the difference between enrollment and payment per click is greater than 0.05, which means that the difference in the average values of the OEC between the two groups are **not statistically significant**. Therefore, Udacity **does not achieve their goal**.

# Exercise 11: re-estimating the effect of treatment on OEC using a linear regression.

In [ ]:
```
import statsmodels.formula.api as smf
res = smf.ols(formula='delta_per_click ~ treatment', data=df_all)
res.fit().summary()
print(res)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:        delta_per_click   R-squared:                       0.051
Model:                            OLS   Adj. R-squared:                  0.029
Method:                 Least Squares   F-statistic:                     2.356
Date:                Mon, 27 Mar 2023   Prob (F-statistic):              0.132
Time:                        22:27:29   Log-Likelihood:                 89.832
No. Observations:                  46   AIC:                            -175.7
Df Residuals:                      44   BIC:                            -172.0
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.1021      0.007     13.948      0.000       0.087       0.117
treatment     -0.0159      0.010     -1.535      0.132      -0.037       0.005
==============================================================================
Omnibus:                       14.160   Durbin-Watson:                   1.805
Prob(Omnibus):                  0.001   Jarque-Bera (JB):               15.205
Skew:                           1.227   Prob(JB):                     0.000499
Kurtosis:                       4.383   Cond. No.                         2.62
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spec
ified.
```

In [ ]:
```
print(f"p-value of treatment is {float(res.tables[1].data[2][4]):.2f}")
```

```
p-value of treatment is 0.13
```

Using a linear regression to estimate the effect of treatment on OEC, the p-value of the coefficient of the treatment group is 0.13 in 2 decimal places, which is the same as the p-value from t-test in Exercise 10.

# Exercise 12: add indicator variables for the day of each observation.

In [ ]:
```
res = smf.ols(formula='delta_per_click ~ treatment+Date', data=df_all)
res.fit().summary()
```

Out[ ]:

OLS Regression Results

| | | | |
|---|---|---|---|
| **Dep. Variable:** | delta_per_click | **R-squared:** | 0.806 |
| **Model:** | OLS | **Adj. R-squared:** | 0.602 |
| **Method:** | Least Squares | **F-statistic:** | 3.962 |
| **Date:** | Mon, 27 Mar 2023 | **Prob (F-statistic):** | 0.000978 |
| **Time:** | 22:27:30 | **Log-Likelihood:** | 126.29 |
| **No. Observations:** | 46 | **AIC:** | -204.6 |
| **Df Residuals:** | 22 | **BIC:** | -160.7 |
| **Df Model:** | 23 | | |
| **Covariance Type:** | nonrobust | | |

| | coef | std err | t | P>\|t\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| **Intercept** | 0.0978 | 0.004 | 21.790 | 0.000 | 0.089 | 0.107 |
| **Date[T.Fri, Nov 7]** | -7.013e-17 | 9.83e-17 | -0.714 | 0.483 | -2.74e-16 | 1.34e-16 |
| **Date[T.Fri, Oct 17]** | 0.0101 | 0.016 | 0.651 | 0.522 | -0.022 | 0.042 |
| **Date[T.Fri, Oct 24]** | 0.0547 | 0.016 | 3.518 | 0.002 | 0.022 | 0.087 |
| **Date[T.Fri, Oct 31]** | 0.0027 | 0.016 | 0.172 | 0.865 | -0.030 | 0.035 |
| **Date[T.Mon, Nov 10]** | 5.597e-18 | 1.95e-17 | 0.287 | 0.777 | -3.48e-17 | 4.6e-17 |
| **Date[T.Mon, Nov 3]** | 3.508e-18 | 1.01e-17 | 0.347 | 0.732 | -1.75e-17 | 2.45e-17 |
| **Date[T.Mon, Oct 13]** | -0.0129 | 0.016 | -0.833 | 0.414 | -0.045 | 0.019 |
| **Date[T.Mon, Oct 20]** | -0.0184 | 0.016 | -1.185 | 0.249 | -0.051 | 0.014 |
| **Date[T.Mon, Oct 27]** | 0.0429 | 0.016 | 2.760 | 0.011 | 0.011 | 0.075 |
| **Date[T.Sat, Nov 1]** | -0.0134 | 0.016 | -0.863 | 0.397 | -0.046 | 0.019 |
| **Date[T.Sat, Nov 15]** | 1.322e-17 | 5.71e-18 | 2.318 | 0.030 | 1.39e-18 | 2.51e-17 |
| **Date[T.Sat, Nov 8]** | 1.497e-17 | 9.58e-18 | 1.563 | 0.132 | -4.89e-18 | 3.48e-17 |
| **Date[T.Sat, Oct 11]** | 0.0084 | 0.016 | 0.543 | 0.592 | -0.024 | 0.041 |
| **Date[T.Sat, Oct 18]** | -0.0337 | 0.016 | -2.169 | 0.041 | -0.066 | -0.001 |
| **Date[T.Sat, Oct 25]** | -0.0208 | 0.016 | -1.337 | 0.195 | -0.053 | 0.011 |
| **Date[T.Sun, Nov 16]** | -1.22e-17 | 7.22e-18 | -1.690 | 0.105 | -2.72e-17 | 2.77e-18 |
| **Date[T.Sun, Nov 2]** | 0.0650 | 0.016 | 4.181 | 0.000 | 0.033 | 0.097 |
| **Date[T.Sun, Nov 9]** | -4.789e-18 | 4.22e-18 | -1.135 | 0.269 | -1.35e-17 | 3.96e-18 |
| **Date[T.Sun, Oct 12]** | -0.0245 | 0.016 | -1.579 | 0.129 | -0.057 | 0.008 |
| **Date[T.Sun, Oct 19]** | -0.0077 | 0.016 | -0.493 | 0.627 | -0.040 | 0.025 |
| **Date[T.Sun, Oct 26]** | -0.0121 | 0.016 | -0.779 | 0.444 | -0.044 | 0.020 |
| **Date[T.Thu, Nov 13]** | 2.52e-17 | 8.36e-18 | 3.014 | 0.006 | 7.86e-18 | 4.25e-17 |
| **Date[T.Thu, Nov 6]** | -9.574e-19 | 3.79e-18 | -0.252 | 0.803 | -8.82e-18 | 6.91e-18 |

| | | | | | | |
|---|---|---|---|---|---|---|
| **Date[T.Thu, Oct 16]** | -0.0127 | 0.016 | -0.818 | 0.422 | -0.045 | 0.020 |
| **Date[T.Thu, Oct 23]** | 0.0056 | 0.016 | 0.358 | 0.724 | -0.027 | 0.038 |
| **Date[T.Thu, Oct 30]** | 0.0689 | 0.016 | 4.437 | 0.000 | 0.037 | 0.101 |
| **Date[T.Tue, Nov 11]** | 3.248e-18 | 2.28e-18 | 1.423 | 0.169 | -1.49e-18 | 7.98e-18 |
| **Date[T.Tue, Nov 4]** | 3.922e-18 | 2.97e-18 | 1.321 | 0.200 | -2.24e-18 | 1.01e-17 |
| **Date[T.Tue, Oct 14]** | -0.0316 | 0.016 | -2.032 | 0.054 | -0.064 | 0.001 |
| **Date[T.Tue, Oct 21]** | 0.0043 | 0.016 | 0.274 | 0.786 | -0.028 | 0.036 |
| **Date[T.Tue, Oct 28]** | -0.0186 | 0.016 | -1.194 | 0.245 | -0.051 | 0.014 |
| **Date[T.Wed, Nov 12]** | 6.378e-19 | 1.03e-19 | 6.221 | 0.000 | 4.25e-19 | 8.5e-19 |
| **Date[T.Wed, Nov 5]** | -4.428e-19 | 1.5e-19 | -2.957 | 0.007 | -7.53e-19 | -1.32e-19 |
| **Date[T.Wed, Oct 15]** | -0.0031 | 0.016 | -0.200 | 0.844 | -0.035 | 0.029 |
| **Date[T.Wed, Oct 22]** | -0.0119 | 0.016 | -0.767 | 0.451 | -0.044 | 0.020 |
| **Date[T.Wed, Oct 29]** | 0.0568 | 0.016 | 3.652 | 0.001 | 0.025 | 0.089 |
| **treatment** | -0.0159 | 0.007 | -2.398 | 0.025 | -0.030 | -0.002 |

| | | | |
|---|---|---|---|
| **Omnibus:** | 3.871 | **Durbin-Watson:** | 1.863 |
| **Prob(Omnibus):** | 0.144 | **Jarque-Bera (JB):** | 3.826 |
| **Skew:** | -0.000 | **Prob(JB):** | 0.148 |
| **Kurtosis:** | 4.413 | **Cond. No.** | 8.53e+16 |

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 8.54e-33. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

The standard deviation after adding the variable for the day of each observation is 0.007, which is smaller than the standard deviation before adding the variable for the day of each observation (0.01). Therefore, the standard deviation of the residuals is **reduced**.

# Exercise 13: Given your results, what would you tell Udacity about their trial?

From the results of regression, only regressing OEC on the treatment indicator doesn't seem to be statistically significant, but when we add the day of each observation as a variable, the p-value of the coefficient of the treatment group is 0.025 (<0.05), which suggests it is statistically significant. With current results, it's possible that the treatment alone did not have a significant effect on the outcome, but the effect may have been influenced by other factors. As a result, it may be worthwhile for Udacity to investigate further and consider other factors such as the sample size before making any significant decisions based on these results.

# Exercise 14： add indicators for day of the week

In [ ]:
```
df_all['Day_of_week'] = df_all['Date'].str[:3]
df_all['Day_of_week'].value_counts()
```

Out[ ]:
```
Sat    12
Sun    12
Mon    10
Tue    10
Wed    10
Thu    10
Fri    10
Name: Day_of_week, dtype: int64
```

In [ ]:
```
smf.ols(formula='delta_per_click ~ treatment+Day_of_week',\
         data=df_all).fit().summary()
```

Out[ ]:

OLS Regression Results

| | | | |
|---|---|---|---|
| **Dep. Variable:** | delta_per_click | **R-squared:** | 0.215 |
| **Model:** | OLS | **Adj. R-squared:** | 0.070 |
| **Method:** | Least Squares | **F-statistic:** | 1.487 |
| **Date:** | Mon, 27 Mar 2023 | **Prob (F-statistic):** | 0.201 |
| **Time:** | 22:27:30 | **Log-Likelihood:** | 94.201 |
| **No. Observations:** | 46 | **AIC:** | -172.4 |
| **Df Residuals:** | 38 | **BIC:** | -157.8 |
| **Df Model:** | 7 | | |
| **Covariance Type:** | nonrobust | | |

| | **coef** | **std err** | **t** | **P>\|t\|** | **[0.025** | **0.975]** |
|---|---|---|---|---|---|---|
| **Intercept** | 0.1203 | 0.015 | 8.070 | 0.000 | 0.090 | 0.150 |
| **Day_of_week[T.Mon]** | -0.0186 | 0.020 | -0.940 | 0.353 | -0.059 | 0.021 |
| **Day_of_week[T.Sat]** | -0.0373 | 0.019 | -2.013 | 0.051 | -0.075 | 0.000 |
| **Day_of_week[T.Sun]** | -0.0173 | 0.019 | -0.934 | 0.356 | -0.055 | 0.020 |
| **Day_of_week[T.Thu]** | -0.0019 | 0.020 | -0.095 | 0.925 | -0.042 | 0.038 |
| **Day_of_week[T.Tue]** | -0.0378 | 0.020 | -1.905 | 0.064 | -0.078 | 0.002 |
| **Day_of_week[T.Wed]** | -0.0086 | 0.020 | -0.432 | 0.668 | -0.049 | 0.032 |
| **treatment** | -0.0159 | 0.010 | -1.569 | 0.125 | -0.036 | 0.005 |

| | | | |
|---|---|---|---|
| **Omnibus:** | 12.955 | **Durbin-Watson:** | 1.665 |
| **Prob(Omnibus):** | 0.002 | **Jarque-Bera (JB):** | 13.612 |
| **Skew:** | 1.110 | **Prob(JB):** | 0.00111 |
| **Kurtosis:** | 4.474 | **Cond. No.** | 9.27 |

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

After adding the indicator variables for the day of the week, the coefficient of treatment is -0.0159, and the standard deviation of the residuals is 0.01, which are the same as the results of the origianl regression with only treatment variable. However, the p-value of this model is 0.125, indicating that the coefficient of treatment is not statistically significant, which is different from the results of last model. This might mean that the impact of the treatment on OEC may have been dependent on the day of the week, and Udacity needs to further investigate this.

In [ ]: