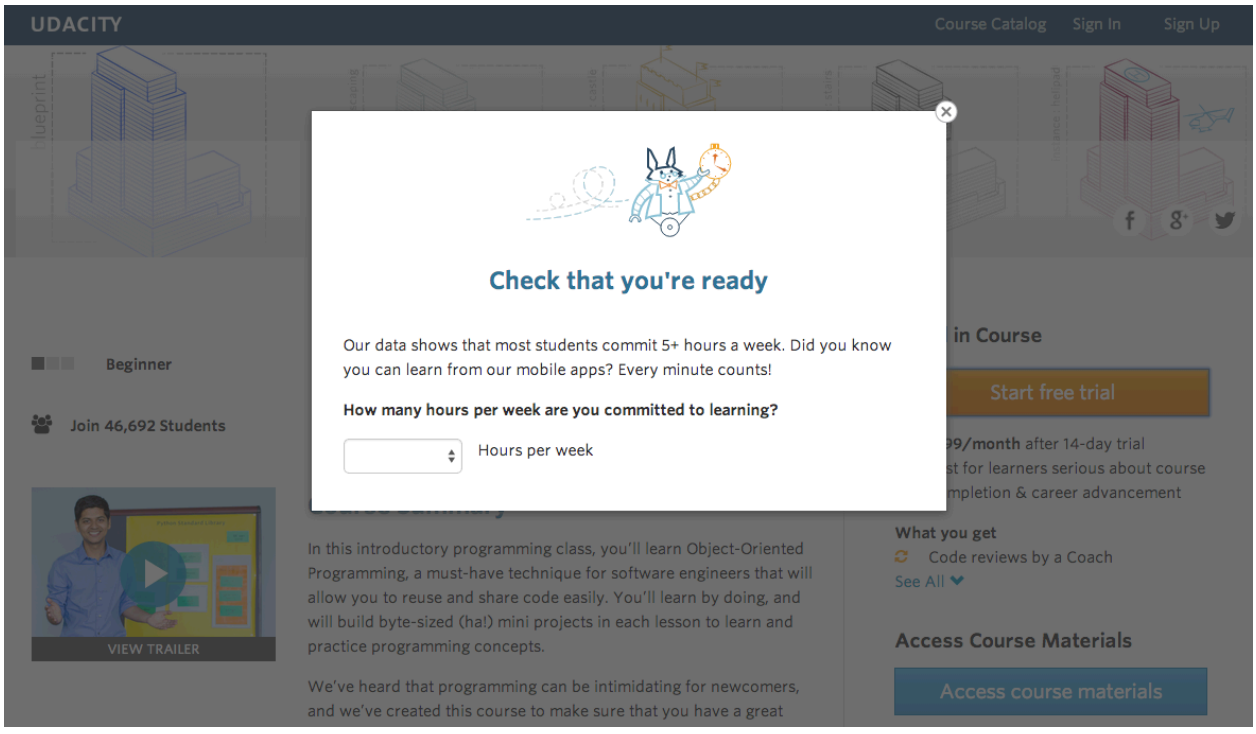
Experiment Overview: Free Trial Screener

At the time of this experiment, Udacity courses currently have two options on the course overview page: "start free trial", and "access course materials". If the student clicks "start free trial", they will be asked to enter their credit card information, and then they will be enrolled in a free trial for the paid version of the course. After 14 days, they will automatically be charged unless they cancel first. If the student clicks "access course materials", they will be able to view the videos and take the quizzes for free, but they will not receive coaching support or a verified certificate, and they will not submit their final project for feedback.

In the experiment, Udacity tested a change where if the student clicked "start free trial", they were asked how much time they had available to devote to the course. If the student indicated 5 or more hours per week, they would be taken through the checkout process as usual. If they indicated fewer than 5 hours per week, a message would appear indicating that Udacity courses usually require a greater time commitment for successful completion, and suggesting that the student might like to access the course materials for free. At this point, the student would have the option to continue enrolling in the free trial, or access the course materials for free instead. This screenshot shows what the experiment looks like.



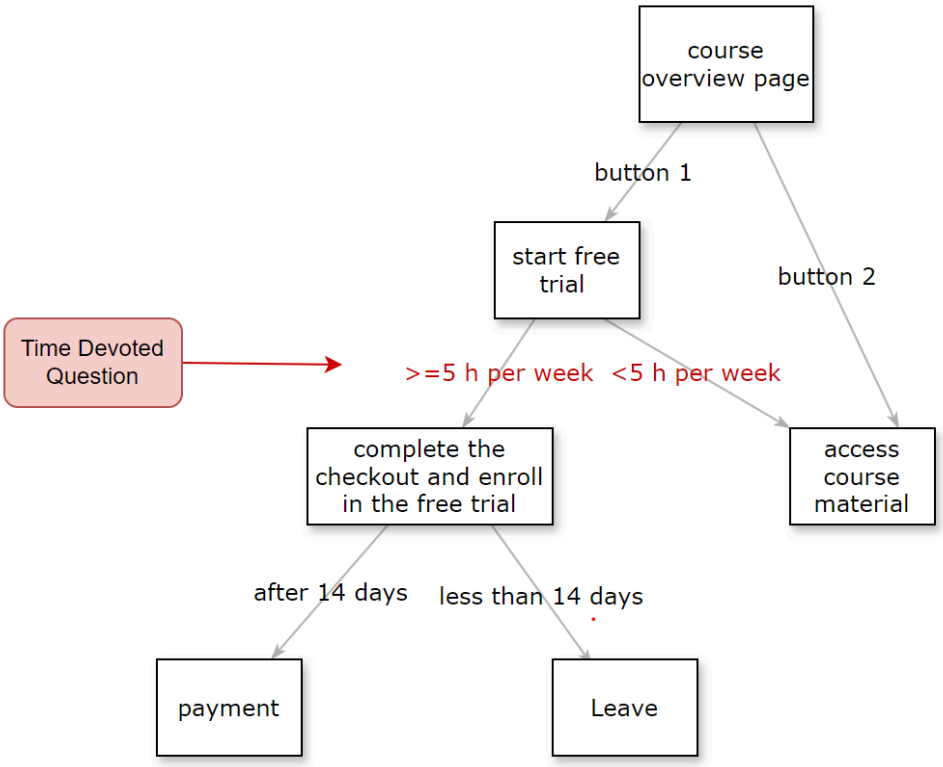
Experiment Design

1. Unit of Diversion (provided by Udacity)

The unit of diversion is **cookie**. Although if the students enrolled in the free trial, they will be tracked by user-id from that point forward. But for students who do not enroll, they will not be tracked by user-id, even if they signed in when they visited the course overview page. The same user-id can not enroll in the free trial twice.

2. Initial Hypothesis

The hypothesis is that the 'Time devoted Window' might set clearer expectations for students upfront, thus **reducing the number of frustrated students who left the free trial** because they don't have enough time. Also, this change won't significantly reduce the students who continue past the free trial and complete the course eventually. If the hypothesis holds true, Udacity would further improve the overall students experience and improve the coaches' capacity to support students who are likely to complete the course.(Provided by Udacity)



So, we can set some initial hypotheses based on the information provided. Then, we will refine the hypotheses later.

1. H0: The change has no effect on the number of students who enroll in the free trial
   H1: The change reduces the number of students who enroll in the free trial
2. H0: The change has no effect on the probability of students who leave the free trial
   H1: The change reduces the probability of students who leave the free trial
3. H0: The change has no effect on the probability of students who make at least one payment
   H1: The change reduces the probability of students who make at least one payment

3. Metric Choice

There are seven choices provided by Udacity below.

- **Number of cookies**: That is, number of unique cookies to view the course overview page. (dmin=3000)
- **Number of user-ids**: That is, number of users who enroll in the free trial. (dmin=50)
- **Number of clicks**: That is, number of unique cookies to click the "Start free trial" button (which happens before the free trial screener is triggered). (dmin=240)
- **Click-through-probability**: That is, number of unique cookies to click the "Start free trial" button divided by number of unique cookies to view the course overview page. (dmin=0.01)
- **Gross conversion**: That is, number of user-ids to complete checkout and enroll in the free trial divided by number of unique cookies to click the "Start free trial" button. (dmin= 0.01)
- **Retention**: That is, number of user-ids to remain enrolled past the 14-day boundary (and thus make at least one payment) divided by number of user-ids to complete checkout. (dmin=0.01)
- **Net conversion**: That is, number of user-ids to remain enrolled past the 14-day boundary (and thus make at least one payment) divided by the number of unique cookies to click the "Start free trial" button. (dmin= 0.0075)

dmin means the practical significance boundary for each metric, that is, the difference that would have to be observed before it was a meaningful change for the business, is given in parentheses. All practical significance boundaries are given as absolute changes.

3.1 Choosing Invariant Metrics

Invariant metrics are metrics that are expected not to change between control and experiment groups. They will be used for sanity checks.

Based on the metrics given above, we will use these three metrics as invariant metrics:

- **Number of cookies**
- **Number of clicks**
- **Click-through-probability**

The data for those three metrics can be collected before the change. So they should remain invariant. Though click-through-probability covers both number of cookies and number of clicks, it might unexpectedly vary during the sanity check, so it's a safer choice to check all those three metrics.

The reason why we don't use number of user-ids is that we only have the user-ids of students who enroll in the free trial, so we couldn't guarantee the distribution between control and experiment is the same and won't be affected by the change.

3.2 Choosing Evaluation Metrics

For evaluation metrics, we should not only care about the statistical significance, but also the practical significance(dmin), because it may be significant in statistics but not worth implementing in practice.

Based on the metrics given above, we will take those three metrics as evaluation metrics for the A/B test.

- **Gross conversion**: We expect this metric to decrease in the experiment group because the time question will filter out students who don't have enough time and the number of students enrolled in the free trial will decrease.
- **Retention**: We expect this metric to increase in the experiment group because the question has filtered out the students who don't have enough time, so the proportion of students remaining enrolled past the 14-day boundary will increase.
- **Net conversion**: We couldn't expect the direction because it's the production of the two above, which may vary.

3.3 Hypothesis Refined

Based on the metrics we choose and the initial hypotheses, we can refine our hypotheses.

1. H0: Gross Conversion(control) = Gross Conversion(treatment)
   H1: Gross Conversion(control) != Gross Conversion(treatment)
2. H0: Retention(control) = Retention(treatment)
   H1: Retention(control) != Retention(treatment)
3. H0: Net Conversion(control) = Net Conversion(treatment)
   H1: Net Conversion(control) != Net Conversion(treatment)

4. Measuring Standard Deviation

Baseline values provided by Udacity:

Unique cookies to view course overview page per day:	40000
Unique cookies to click "Start free trial" per day:	3200
Enrollments per day:	660
Click-through-probability on "Start free trial":	0.08
Probability of enrolling, given click:	0.20625
Probability of payment, given enroll:	0.53
Probability of payment, given click	0.1093125

Since each evaluation metric complies with the binomial distribution and the unit of analysis is the same as the unit of diversion, we can calculate the standard deviation analytically, and don't need to calculate the empirical variability.

Further, as n is relatively large in each case, according to the Central Limit Theorem, the sampling distribution of sample mean approaches a normal distribution, so we could calculate **the standard deviation of the sampling distribution of sample mean (standard error, in short) for each metric as a good estimate of the standard deviation of each metric**. (We could use the 3-standard-deviation rule to check if n is large enough.)

Given the sample size of cookies to view the course overview is 5000 (provided by Udacity), we could compute standard errors for each metric using the following formula:

$$SE = \sqrt{\frac{\hat{p} * (1-\hat{p})}{n}}$$

Evaluation Metrics	Standard Deviation
Gross Conversion	0.0202

Retention	0.0549
Net Conversion	0.0156

5. Sizing

5.1 Number of Samples vs. Power

We set the significance level to be 0.05 and the statistical power to be 0.80 (with beta to be 0.20). We use an online calculator to compute the number of samples needed.

Evaluation Metrics	Sample Size
Gross Conversion	645,875
Retention	4,741,212
Net Conversion	685,325

Given the calculation above, to test these three hypotheses, we need 4,741,212 cookies to view the course overview page.

We also need to decide whether to use Bonferroni correction at the experiment analysis stage. According to this paper, whether or not to use the Bonferroni correction depends on the circumstances of the study. It should be considered when it is imperative to avoid type I error or when we conduct a large number of tests without pre-planned hypotheses. For this analysis, we just have 3 hypotheses to test and being very conservative on the results is not required, so we decide not to use Bonferroni correction.

5.2 Duration vs. Exposure

From Udacity, given there are 40k pageviews per day, we could first assume to divert 100% of traffic to this experiment. With 100% of traffic, days needed to run this experiment is:

- For the experiment with gross conversion, retention and net conversion, we need: 119 days
- For the experiment with gross conversion and net conversion, we need: 17 days

However, if we are going to use retention as one of the metrics, the duration would be too long to be accepted. Because we couldn't run other experiments during this period and that would generate opportunity costs. Besides, if the change frustrates students and decreases the conversion rate, we couldn't notice that for more than 4 months and this will harm our business.

So we choose to divert 50% of traffic to this experiment and discard retention metric that could be inferred from other two metrics.

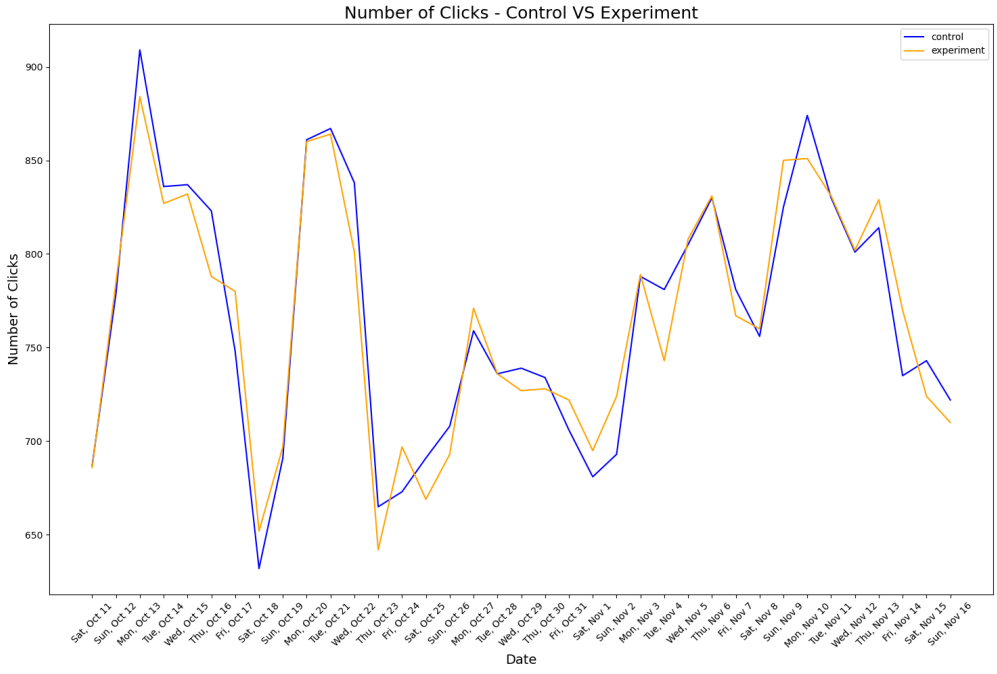
- For the experiment using 50% of traffic with gross conversion and net conversion, we need: **34 days**

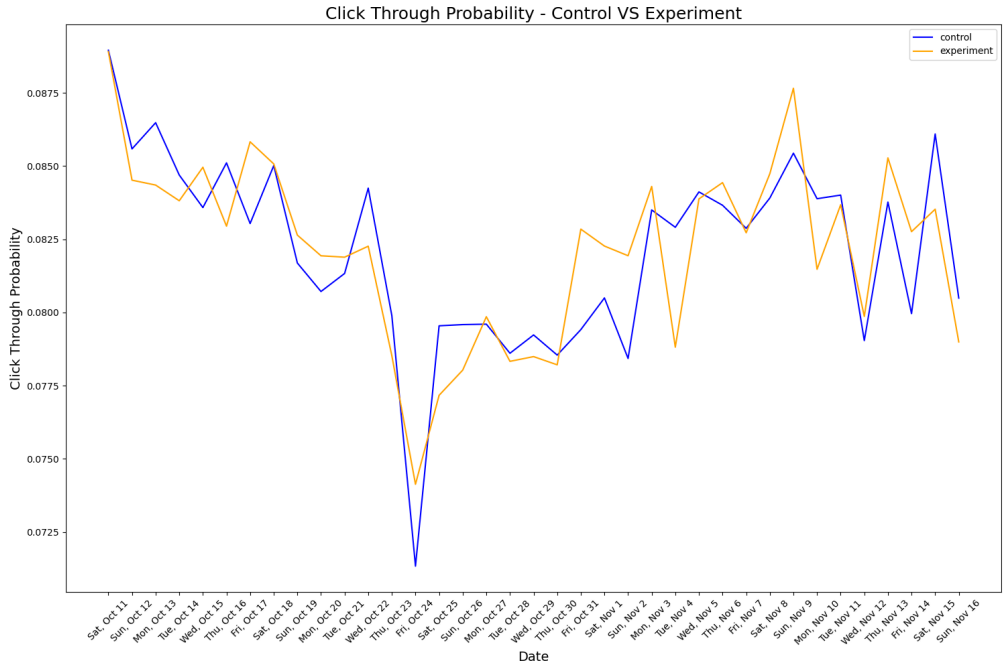
Experiment Analysis

1. Sanity Checks

1.1 Visualization

We could visualize the invariant metrics to get a clearer comparison between control and experiment group.





We can see that there is a dramatic drop on CTP on Oct 24. It's worth investigating further.

1.2 Sanity checks

Based on the data provided by Udacity, we collected 345543 cookies in the control group and 344660 cookies in the experiment group, so the total sample size is 690203. We now want to check if those cookies have been randomly assigned to the two groups, meaning there should not be a significant difference between the two groups. So the observed value should be within the confidence interval.

Cookies and Clicks: Because cookies and clicks should be randomly assigned to the control and experiment group, the probability of being in either group is 0.5. We could use binomial distribution to model the number of cookies/clicks in either group, let's say in the control group, given the total sample size. Further, because the sample size is relatively large (Central Limit Theorem), we launch a Z-test to check these metrics.

CTP: We usually assume CTP follows a normal distribution. Ideally, the difference of CTP between two groups also follows a normal distribution as a consequence of the linearity of variance for two normally distributed independent variables (assumed to be independent). Since we also assume the variances between two groups are equal, we would use pooled standard error as a better estimate of the standard error of the difference. (not equal use Welch's t test)

Invariant Metrics	CI_lower	CI_upper	Observed Value	Result
Cookies	0.4988	0.5012	0.5006	Pass

Clicks	0.4959	0.5042	0.5005	Pass
Difference_CTP	-0.001296	0.001296	0.0001	Pass

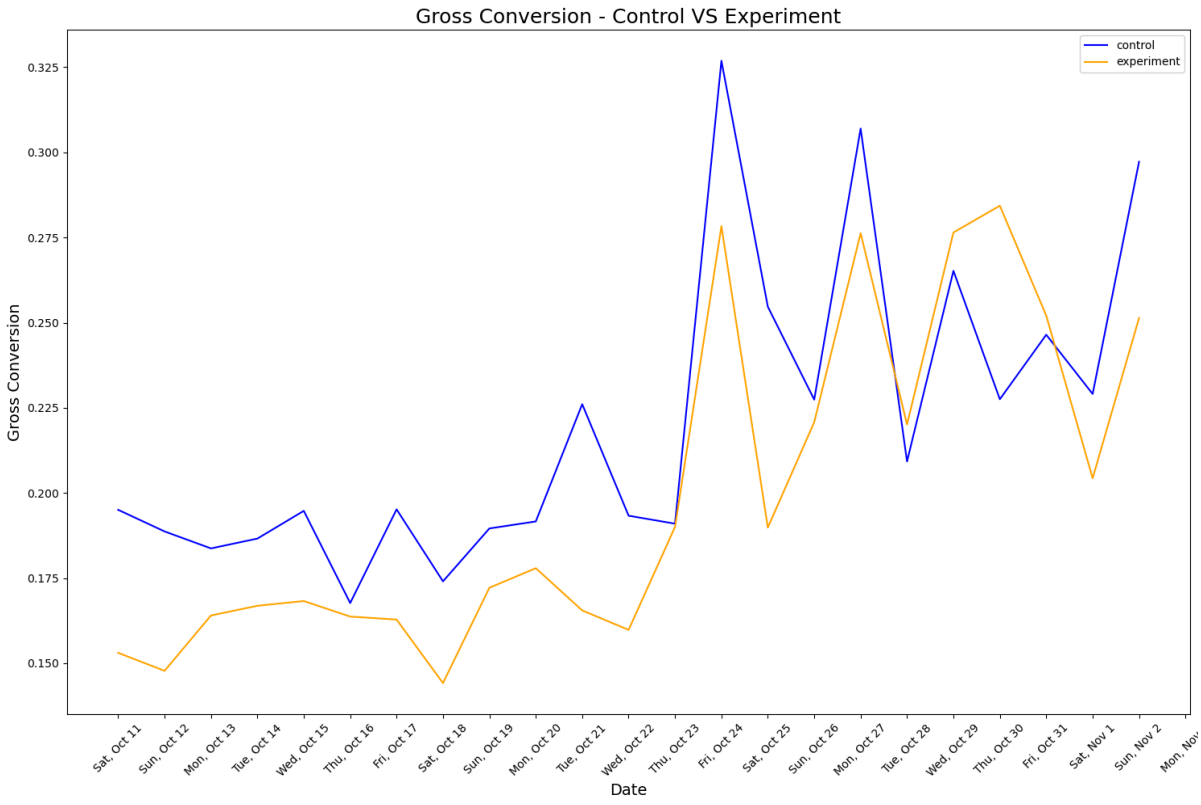
Based on Z-test results, the sanity check for invariant metrics has all passed.

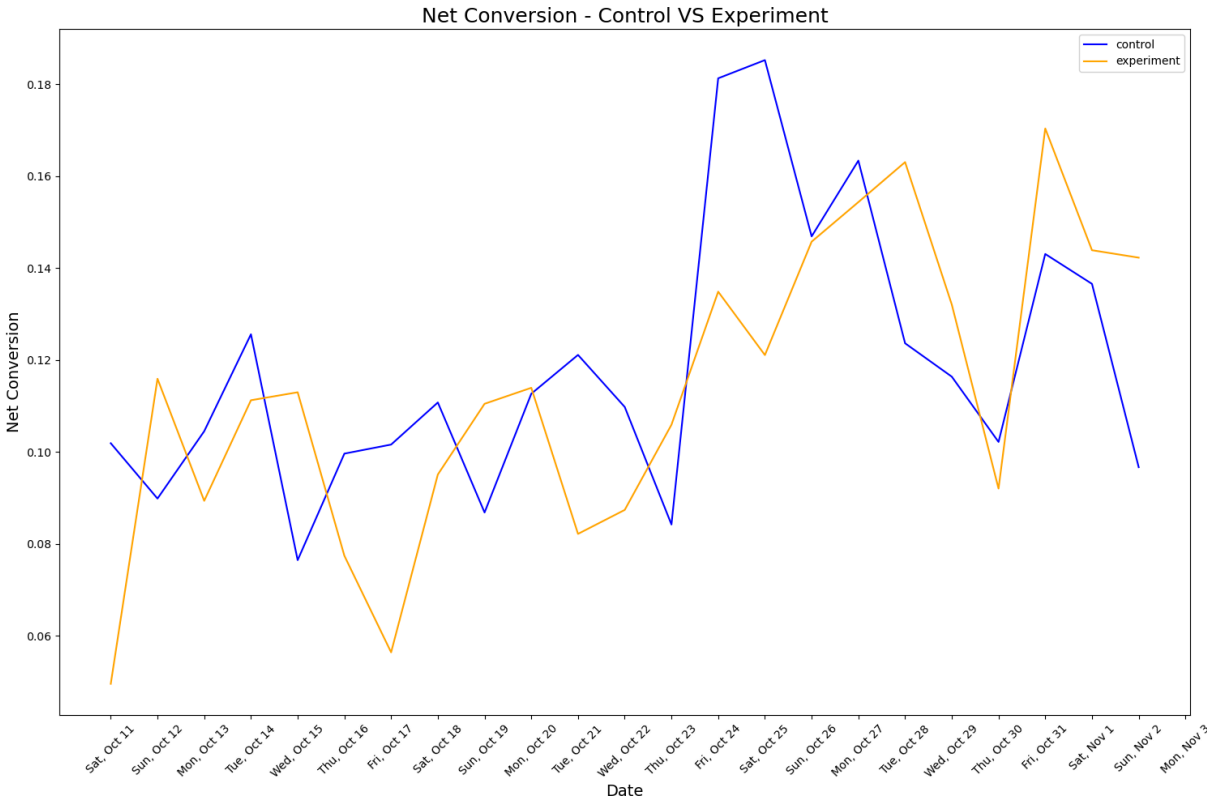
2. Result Analysis

2.1 Effect Size Tests

2.1.1 Visualization

Same as before, we visualize the experiment metrics to get a clearer comparison between two groups.





We could also observe a huge peak on both metrics on Oct 24. Based on what we observed on the CTP metric, we could infer that there is a relatively less number of clicks on that day. So we recommend Udacity teams to look at the whole traffic, not just this part of traffic, to find out the reason.

2.1.2 Practical and Statistical Significance Check

Both conversion metrics are proportions, similar to CTP, so we could check the significance of their difference the way we check CTP. Besides, we will check both statistical and practical significance, because practical significance decides whether to implement this change or not. (Notice: we need to calculate the CI around the observed value and check if the CI contains 0, not like before we do with invariant metrics.)

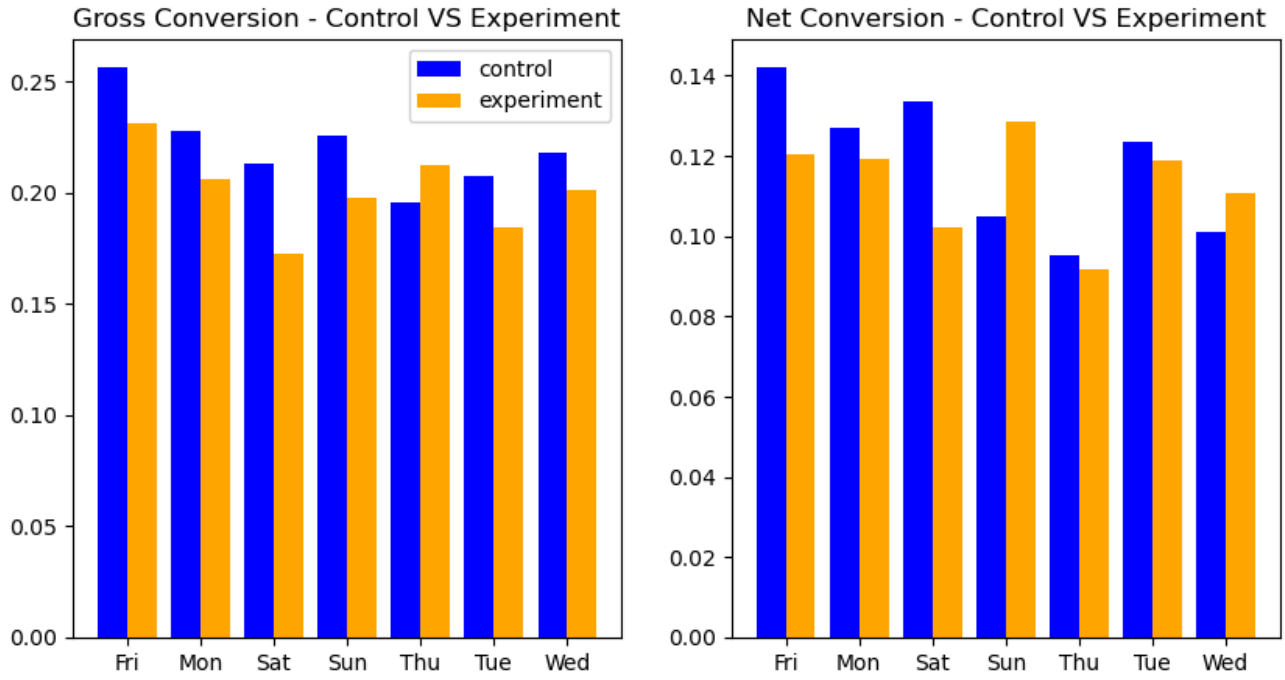
Because students make payments only after the 14 day trial, the payments and Enrollments are None in the last 14 days. However, we need 685,325 cookies to obtain a robust test result. But we only have 423,525 cookies since we don't have the last 14 day data, which is not enough. So we recommend extending the experiment period to get more data.

Evaluation Metrics	CI_lower	CI_upper	Observed Value	Statistic Result	dmin	Practical Result

Diff_GC	-0.0291	-0.0120	-0.0205	Pass	-0.01	Pass
Diff_NC	-0.0116	0.0019	-0.0049	Fail	0.0075	Fail

2.2 Weekly Analysis

In some cases, doing a sign test can help us examine the data day-by-day, but it requires the samples from two populations to be dependent or paired, which violates the assumption of our A/B test. So we are doing a weekly analysis, instead of a sign test, trying to find some patterns in our data.



From the charts above, we can see that this experiment for Gross Conversion rate was less effective on Thursday. As for Net Conversion rate, it was only effective on Sunday and Wednesday.

2.3 Results Summary

1. **For Gross Conversion**, we observed about a 2% drop which corresponds to the expectation we made at the hypothesis phase. And the 2% drop is both statistically and practically significant. Furthermore, the effect was obvious on Saturday, but almost had no impact on Thursday.
2. **For Net Conversion**, the observed value, about -0.5%, was not at the direction we expected, and it failed at both significance tests. Besides, as weekly analysis shows, we

only observed an increase on Wednesday and Sunday, but a decrease on the rest of five days.

3. Recommendation

This change does have an impact on reducing frustrated students and diverting them to access free course materials. But the overall goal of Udacity business is trying to increase their revenue, so user payments are much more important. In that case, though we observed a significant value on Gross Conversion rate, we still cannot decide to launch this change because Net Conversion rate is much more critical and it's not significant both statistically and practically. However, since we didn't collect enough data for the test of these two, we need further experiments to confirm our results.

Follow-Up Experiment

If we extended the experiment duration and the test for Net Conversion still failed, we could conduct other experiments and try to reduce the early cancellation (the cancellation before the end of the 14-day free trial period that triggered the payment). The considerations are from two aspects: before and after the enrollment.

- **Before the Enrollment**: we could reuse the experiment design we made in this test, and just make some modifications on the free-trial screener to see if they would improve the Gross Conversion rate and Net Conversion rate. For example, we could add some encouragement on the screener that would give students more incentive to finish the course. Like 'Complete the course and you're one step closer to your dream!', this could help students make up their mind to take the course. Second, for those who choose to not enroll in the course, we could add a button directing them to the free course materials page. This button improves the user experience and encourages more students to access the free materials. After change, we could also test the Gross Conversion rate and Net Conversion rate which would decrease and increase respectively.

- **After the Enrollment**: when students are watching online courses or doing assignments, we could add a pop-up window to ask if they need coaches when they cost more time than average on a certain course or a certain assignment. This could help with improving retention because students could learn more easily.
    - **Experiment Design:** For the students who have enrolled, randomly assign them to the experiment group and control group.
    - **Unit of Diversion:** user-ids
      Rationale: Because we only consider those who already enrolled in the course, we could track them using user-ids.

- **Hypothesis:**
  H0: The pop-up window has no effect on the number of students who leave the free trial.
  H1: The pop-up window reduces the number of students who leave the free trial.
  Rationale: With in time coaches, students are less likely to be frustrated by the difficulty of the course and thus more likely to remain after the trial ended.
- **Invariant Metrics:** user-ids
  Rationale: User-ids is what we could collect before the change based on the unit of diversion of this test.
- **Evaluation Metrics**: Retention
  Rationale: Retention is the number of user-ids to remain enrolled past the 14-day boundary (and thus make at least one payment) divided by number of user-ids to complete checkout.

Given the sanity check passed, if the Gross Conversion and Net Conversion in the first experiment or or the Retention in the second experiment is statistically and practically significant, and the resources and costs are acceptable for Udacity, we could launch this experiment.

Related Reference:
https://github.com/zyellieyan/AB-Testing-Project?tab=readme-ov-file
https://www.kaggle.com/code/mariusmesserschmied/udacity-a-b-testing-final-course-project/notebook

Appendix: Python Coding Implementation

1. For count metrics, SE = root (p*(1-p)/n); For proportion metrics, SE_pooled = root (s**2/n + s**2/n, s = root (p*(1-p))
2. n is large enough, use Z test
3. 14 days data is missing, recalculate sample size

Visualization for invariant variables

In [25]:
```
import pandas as pd
```

In [26]:
```
ctl = pd.read_csv('D:\\pandas\\YUQI\\Udacity\\Final Project Results - Control.csv')
ctl.head()
```

Out[26]:

	Date	Pageviews	Clicks	Enrollments	Payments
0	Sat, Oct 11	7723	687	134.0	70.0
1	Sun, Oct 12	9102	779	147.0	70.0
2	Mon, Oct 13	10511	909	167.0	95.0
3	Tue, Oct 14	9871	836	156.0	105.0
4	Wed, Oct 15	10014	837	163.0	64.0

In [28]:
```
ctl.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 37 entries, 0 to 36
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Date         37 non-null     object 
 1   Pageviews    37 non-null     int64  
 2   Clicks       37 non-null     int64  
 3   Enrollments  23 non-null     float64
 4   Payments     23 non-null     float64
dtypes: float64(2), int64(2), object(1)
memory usage: 1.6+ KB
```

In [29]:
```
exp = pd.read_csv('D:\\pandas\\YUQI\\Udacity\\Final Project Results - Experiment.csv')
exp.head()
```

Out[29]:

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

In [30]:
```
exp.info()
```

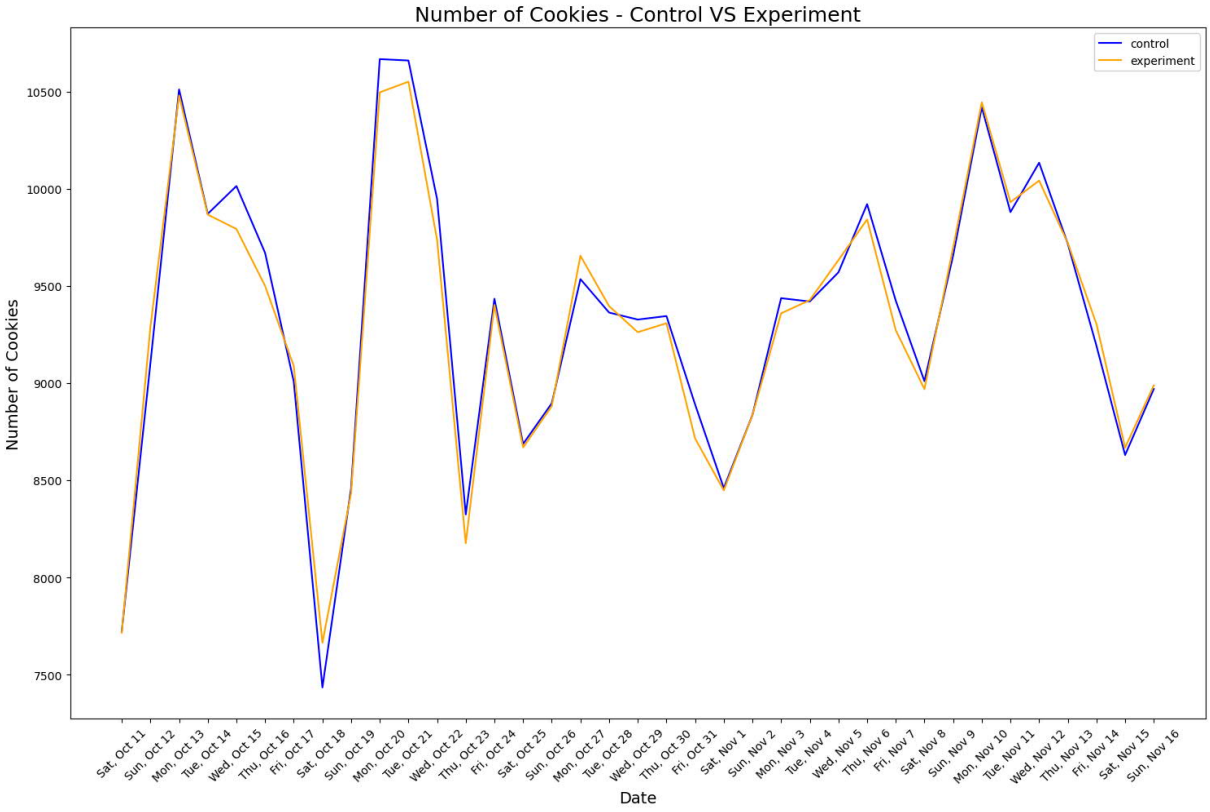
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 37 entries, 0 to 36
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Date         37 non-null     object 
 1   Pageviews    37 non-null     int64  
 2   Clicks       37 non-null     int64  
 3   Enrollments  23 non-null     float64
 4   Payments     23 non-null     float64
dtypes: float64(2), int64(2), object(1)
memory usage: 1.6+ KB
```

1. visualize invariant metrics to get a clearer comparison

In [31]:
```
from matplotlib import pyplot as plt
filepath = 'D:\\pandas\\YUQI\\Udacity\\'
```

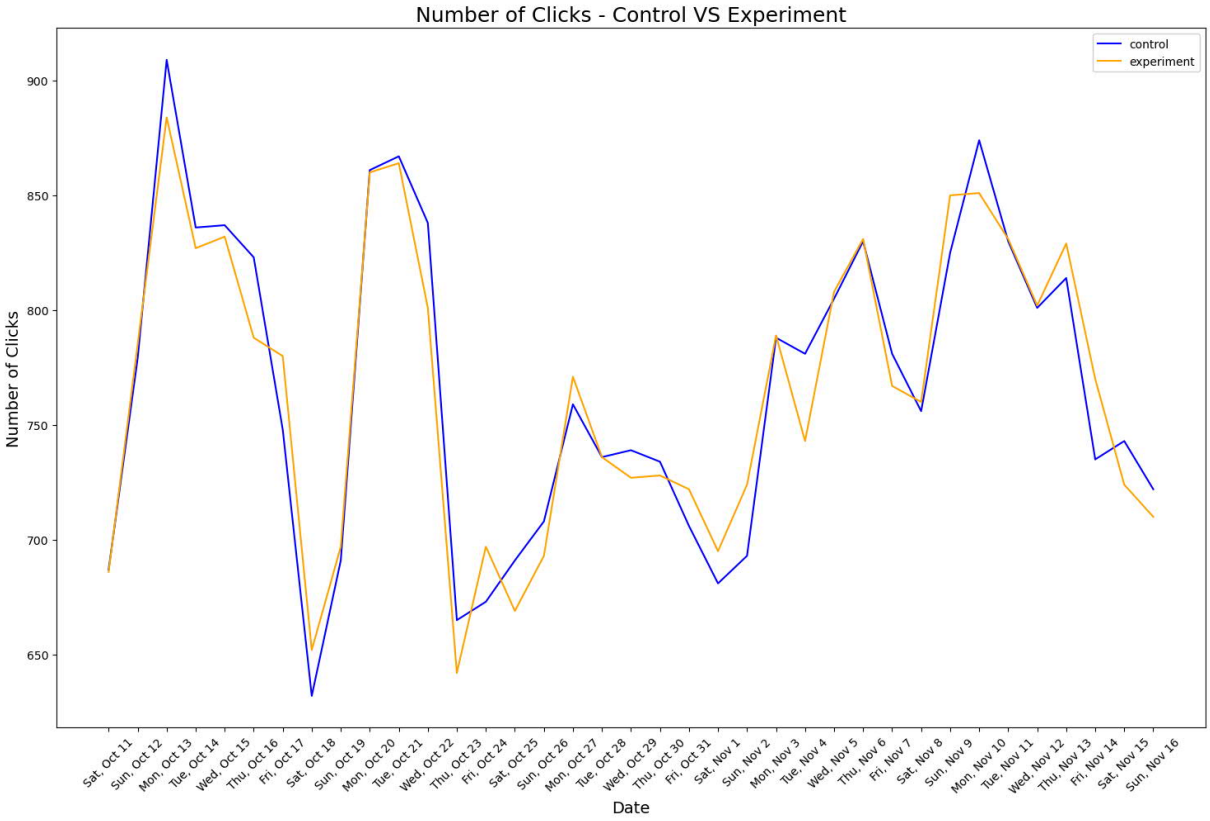
1.1 pageview

In [32]:
```
plt.figure(figsize=(18,11))
plt.plot(ctl['Date'], ctl['Pageviews'], color = 'blue', label = 'control')
plt.plot(exp['Date'], exp['Pageviews'], color = 'orange', label =  'experiment')
plt.xticks(rotation = 45)
plt.xlabel('Date', fontsize = 14)
plt.ylabel('Number of Cookies', fontsize = 14)
plt.title('Number of Cookies - Control VS Experiment', fontsize = 18)
plt.legend()
plt.savefig(filepath+'pageviews')
plt.show()
```



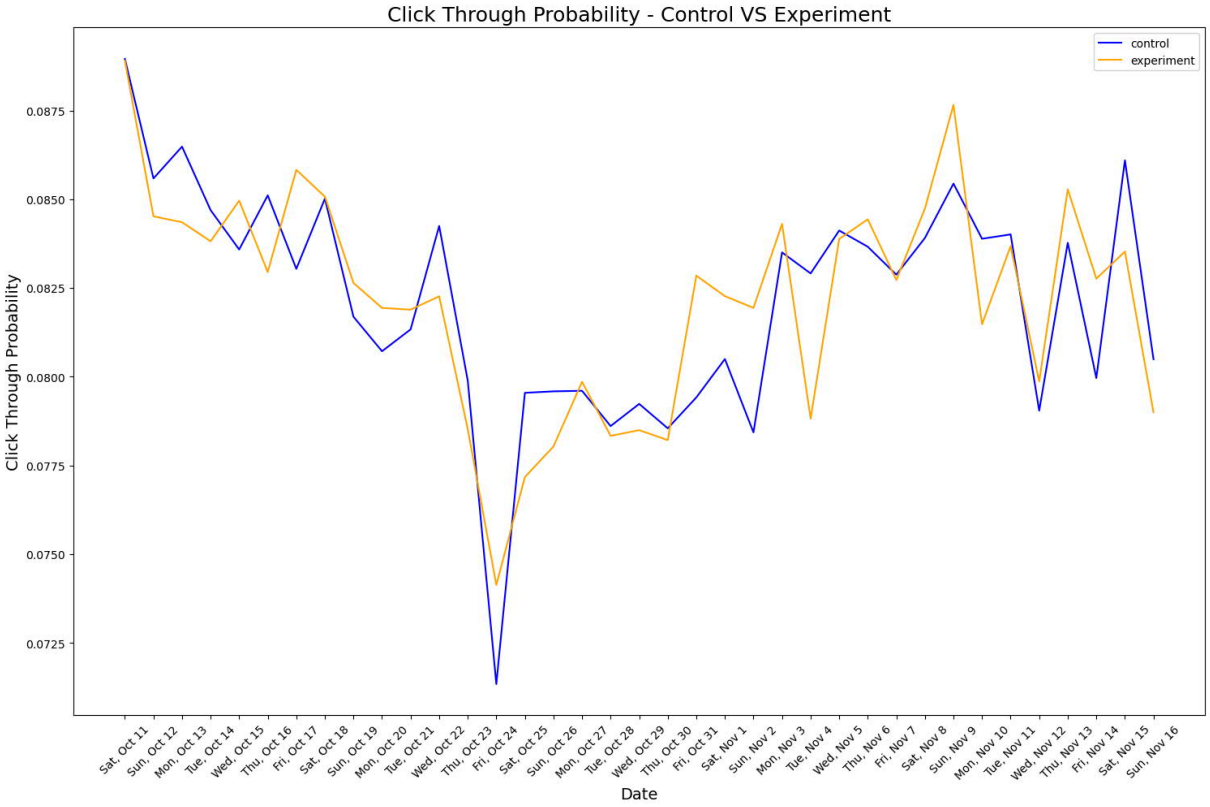
1.2 clicks

In [33]:
```
plt.figure(figsize=(18,11))
plt.plot(ctl['Date'], ctl['Clicks'], color = 'blue', label = 'control')
plt.plot(exp['Date'], exp['Clicks'], color = 'orange', label =  'experiment')
plt.xticks(rotation = 45)
plt.legend()
plt.xlabel('Date', fontsize = 14)
plt.ylabel('Number of Clicks', fontsize = 14)
plt.title('Number of Clicks - Control VS Experiment', fontsize = 18)
plt.savefig(filepath+'Clicks')
plt.show()
```



1.3 CTP

In [34]:
```
plt.figure(figsize=(18,11))
plt.plot(ctl['Date'], ctl['Clicks']/ctl['Pageviews'], color = 'blue', label = 'control')
plt.plot(exp['Date'], exp['Clicks']/exp['Pageviews'], color = 'orange', label =  'experiment')
plt.xticks(rotation = 45)
plt.xlabel('Date',fontsize = 14)
plt.ylabel('Click Through Probability',fontsize = 14)
plt.title('Click Through Probability - Control VS Experiment', fontsize = 18)
plt.legend()
plt.savefig(filepath+'CTP')
plt.show()
```



we can see there is a significant drop on Oct 23. We will look into that later.

In [35]:
```
import numpy as np
from scipy import stats
```

2. Define a Function for SE of pageviews and clicks, a Function for pooled SE of CTP

In [36]:
```
def SE(n,p) -> float:
    # for count metrics
    # n: sample size in total
    # p: probability
    return (p*(1-p)/n)**0.5

def pooled_SE(s_con, s_exp,n_con, n_exp) -> float:
    # for proportion metrics with different variance between 2 groups
    # s_con: estimated standard deviation of CTP in control group (use frequency as probability)
    # s_exp: estimated standard deviation of CTP in experiment group (root p*(1-p))
    # n_con: sample size of the denominator of the metric in control group
    # n_exp: sample size of the denominator of the metric in experiment group
    return (s_con**2/n_con + s_exp**2/n_exp)**0.5
```

Sanity check for invariant metrics

In [37]:
```
results = pd.DataFrame(columns=['metrics', 'CI_left', 'CI_right', 'observed', 'sanity check'])
results['metrics'] = ['Number of Cookies', 'Number of Clicks', 'Difference_CTP']
results
```

Out[37]:

	metrics	CI_left	CI_right	observed	sanity check
0	Number of Cookies	NaN	NaN	NaN	NaN
1	Number of Clicks	NaN	NaN	NaN	NaN
2	Difference_CTP	NaN	NaN	NaN	NaN

In [38]:
```
n_cookie = ctl['Pageviews'].sum()+exp['Pageviews'].sum()
n_click = ctl['Clicks'].sum()+exp['Clicks'].sum()
fq_con = ctl['Clicks'].sum()/ctl['Pageviews'].sum()
fq_exp = exp['Clicks'].sum()/exp['Pageviews'].sum()
observed = [ctl['Pageviews'].sum()/n_cookie,
            ctl['Clicks'].sum()/n_click,
            (fq_exp - fq_con)]

results['observed'] = [round(x, 4) for x in observed]
results
```

Out[38]:

	metrics	CI_left	CI_right	observed	sanity check
0	Number of Cookies	NaN	NaN	0.5006	NaN
1	Number of Clicks	NaN	NaN	0.5005	NaN
2	Difference_CTP	NaN	NaN	0.0001	NaN

1. Calculate CI = mean +- Z*SE

In [39]:
```
# se
se_cookie = SE(n_cookie, 0.5) # the probability of assigning to control is 0.5
se_click = SE(n_click, 0.5)
s_con = (fq_con*(1-fq_con))**0.5
s_exp = (fq_exp*(1-fq_exp))**0.5
se_pooled = pooled_SE(s_con, s_exp, ctl['Pageviews'].sum(), exp['Pageviews'].sum())

# CI
alpha = 0.05
se = [se_cookie, se_click, se_pooled]
mean = [0.5, 0.5, 0]
ci_left = []
ci_right = []
for i, j in zip(mean, se):
    ci_left.append(i - stats.norm.ppf(1-alpha/2)*j)
    ci_right.append(i + stats.norm.ppf(1-alpha/2)*j)

results['CI_left'] = ci_left
results['CI_right'] = ci_right
results
```

Out[39]:

	metrics	CI_left	CI_right	observed	sanity check
0	Number of Cookies	0.498820	0.501180	0.5006	NaN
1	Number of Clicks	0.495885	0.504115	0.5005	NaN
2	Difference_CTP	-0.001296	0.001296	0.0001	NaN

2. Sanity Check

In [40]:
```
sanity_check = []
for i in [0,1,2]:
    if ci_left[i] <= observed[i] <= ci_right[i]:
        sanity_check.append('pass')
    else:
        sanity_check.append('not pass')
results['sanity check'] = sanity_check
results
```

Out[40]:

	metrics	CI_left	CI_right	observed	sanity check
0	Number of Cookies	0.498820	0.501180	0.5006	pass
1	Number of Clicks	0.495885	0.504115	0.5005	pass
2	Difference_CTP	-0.001296	0.001296	0.0001	pass

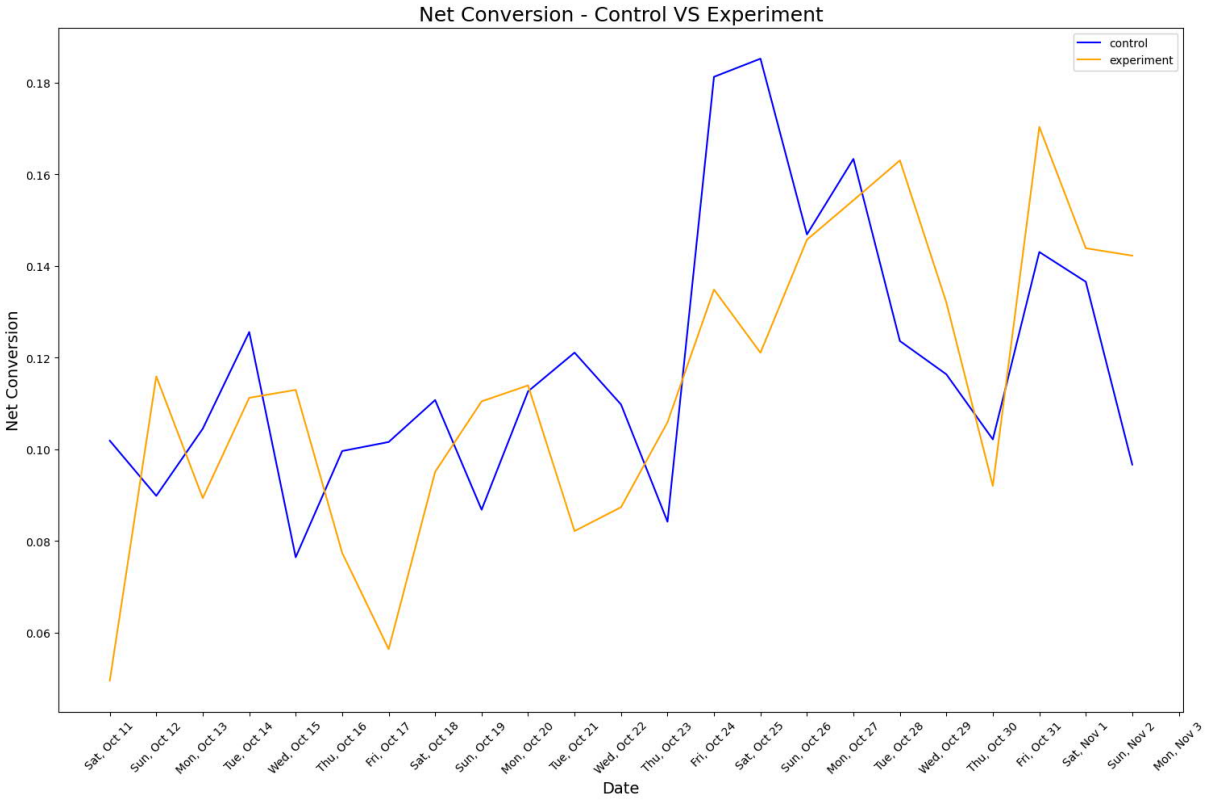
Visualize the experiment metrics

1.1 Gross Conversion

In [41]:
```
plt.figure(figsize=(18,11))
plt.plot(ctl['Date'], ctl['Enrollments']/ctl['Clicks'], color = 'blue', label = 'control')
plt.plot(exp['Date'], exp['Enrollments']/exp['Clicks'], color = 'orange', label =  'experiment')
plt.xticks(rotation = 45)
plt.xlabel('Date',fontsize = 14)
plt.ylabel('Gross Conversion',fontsize = 14)
plt.title('Gross Conversion - Control VS Experiment', fontsize = 18)
plt.legend()
plt.savefig(filepath+'Gross Conversion')
plt.show()
```

1.2 Net Conversion

In [42]:
```
plt.figure(figsize=(18,11))
plt.plot(ctl['Date'], ctl['Payments']/ctl['Clicks'], color = 'blue', label = 'control')
plt.plot(exp['Date'], exp['Payments']/exp['Clicks'], color = 'orange', label =  'experiment')
plt.xticks(rotation = 45)
plt.xlabel('Date',fontsize = 14)
plt.ylabel('Net Conversion',fontsize = 14)
plt.title('Net Conversion - Control VS Experiment', fontsize = 18)
plt.legend()
plt.savefig(filepath+'Net Conversion')
plt.show()
```



Practical and Statistical Significance check for experiment metrics

In [43]:
```
sig_check = pd.DataFrame(columns=['metrics', 'CI_left', 'CI_right', 'observed value', 'dmin', 'stat_check', 'prac
```

In [44]:
```
sig_check['metrics'] = ['Diff_GC', 'Diff_NC']
sig_check['dmin'] = [0.01, 0.0075]
sig_check
```

Out[44]:

	metrics	CI_left	CI_right	observed value	dmin	stat_check	prac_check
0	Diff_GC	NaN	NaN	NaN	0.0100	NaN	NaN
1	Diff_NC	NaN	NaN	NaN	0.0075	NaN	NaN

In [45]:
```
ctl['Pageviews'].iloc[:23].sum()+exp['Pageviews'].iloc[:23].sum()
```

Out[45]:
```
423525
```

We need 685,325 cookies to obtain a robust test result. But since payment can only be made after 14-day trial, we don't have the last 14 day data and thus we only have 423,525 cookies to check the Net Conversion metric, which is not enough

In [46]:
```
# observed value
gc_con = ctl['Enrollments'].iloc[:23].sum()/ctl['Clicks'].iloc[:23].sum()
gc_exp = exp['Enrollments'].iloc[:23].sum()/exp['Clicks'].iloc[:23].sum()

nc_con = ctl['Payments'].iloc[:23].sum()/ctl['Clicks'].iloc[:23].sum()
nc_exp = exp['Payments'].iloc[:23].sum()/exp['Clicks'].iloc[:23].sum() # calculate from 15th day
## may not have enough sample
diff_GC = gc_exp - gc_con
diff_NC = nc_exp - nc_con
sig_check['observed value'] = [diff_GC, diff_NC]

# SE & pooled SE
s_GC_con = (gc_con*(1-gc_con))**0.5
s_GC_exp = (gc_exp*(1-gc_exp))**0.5

s_NC_con = (nc_con*(1-nc_con))**0.5
s_NC_exp = (nc_exp*(1-nc_exp))**0.5

pooled_SE_GC = pooled_SE(s_GC_con, s_GC_exp, ctl['Clicks'].iloc[:23].sum(), exp['Clicks'].iloc[:23].sum())
pooled_SE_NC = pooled_SE(s_NC_con, s_NC_exp, ctl['Clicks'].iloc[:23].sum(), exp['Clicks'].iloc[:23].sum())

SE_pooled = [pooled_SE_GC, pooled_SE_NC]
```

1. Calculate CI = observed_value- Z*pooled_se

In [47]:
```
CI_left = []
CI_right = []
for i in range(2):
    CI_left.append(sig_check['observed value'][i] - stats.norm.ppf(1-alpha/2)*SE_pooled[i])
    CI_right.append(sig_check['observed value'][i] + stats.norm.ppf(1-alpha/2)*SE_pooled[i])

sig_check['CI_left'] = CI_left
sig_check['CI_right'] = CI_right

sig_check
```

Out[47]:

	metrics	CI_left	CI_right	observed value	dmin	stat_check	prac_check
0	Diff_GC	-0.029120	-0.011990	-0.020555	0.0100	NaN	NaN
1	Diff_NC	-0.011604	0.001857	-0.004874	0.0075	NaN	NaN

2. Check stats and prac significance

In [48]:
```
stat = []
prac = []
for i in range(2):
    if CI_left[i] <= 0 <= CI_right[i]:
        stat.append('not statistically significant')
    else:
        stat.append('statistically significant')

for i in range(2):
    if abs(sig_check['observed value'][i]) <= sig_check['dmin'][i]:
        prac.append('not practically significant')
    else:
        prac.append('practically significant')

sig_check['stat_check'] = stat
sig_check['prac_check'] = prac
```

In [49]:
```
sig_check
```

Out[49]:

	metrics	CI_left	CI_right	observed value	dmin	stat_check	prac_check
0	Diff_GC	-0.029120	-0.011990	-0.020555	0.0100	statistically significant	practically significant
1	Diff_NC	-0.011604	0.001857	-0.004874	0.0075	not statistically significant	not practically significant

Weekly analysis

In [73]:
```
df = pd.DataFrame(columns=['week_day', 'GC_con', 'GC_exp', 'NC_con', 'NC_exp'])
```

In [74]:
```
df['week_day'] = ctl['Date'].str[:3]
```

In [75]:
```
df['GC_con'] = ctl['Enrollments']/ctl['Clicks']
df['GC_exp'] = exp['Enrollments']/exp['Clicks']
df['NC_con'] = ctl['Payments']/ctl['Clicks']
df['NC_exp'] = exp['Payments']/exp['Clicks']
```

In [76]:
```
df.head()
```

Out[76]:

	week_day	GC_con	GC_exp	NC_con	NC_exp
0	Sat	0.195051	0.153061	0.101892	0.049563
1	Sun	0.188703	0.147771	0.089859	0.115924
2	Mon	0.183718	0.164027	0.104510	0.089367
3	Tue	0.186603	0.166868	0.125598	0.111245
4	Wed	0.194743	0.168269	0.076464	0.112981

In [81]:
```
weekday = df.groupby('week_day').mean()
weekday
```

Out[81]:

week_day	GC_con	GC_exp	NC_con	NC_exp
Fri	0.256180	0.231078	0.141981	0.120545
Mon	0.227446	0.206066	0.126848	0.119222
Sat	0.213220	0.172846	0.133614	0.102404
Sun	0.225736	0.198024	0.105066	0.128601
Thu	0.195392	0.212692	0.095342	0.091788
Tue	0.207303	0.184162	0.123449	0.118822
Wed	0.217761	0.201516	0.100874	0.110807

In [87]:
```
wd = weekday.index.tolist()
wd
```

Out[87]:
```
['Fri', 'Mon', 'Sat', 'Sun', 'Thu', 'Tue', 'Wed']
```

Visualize the weekly data

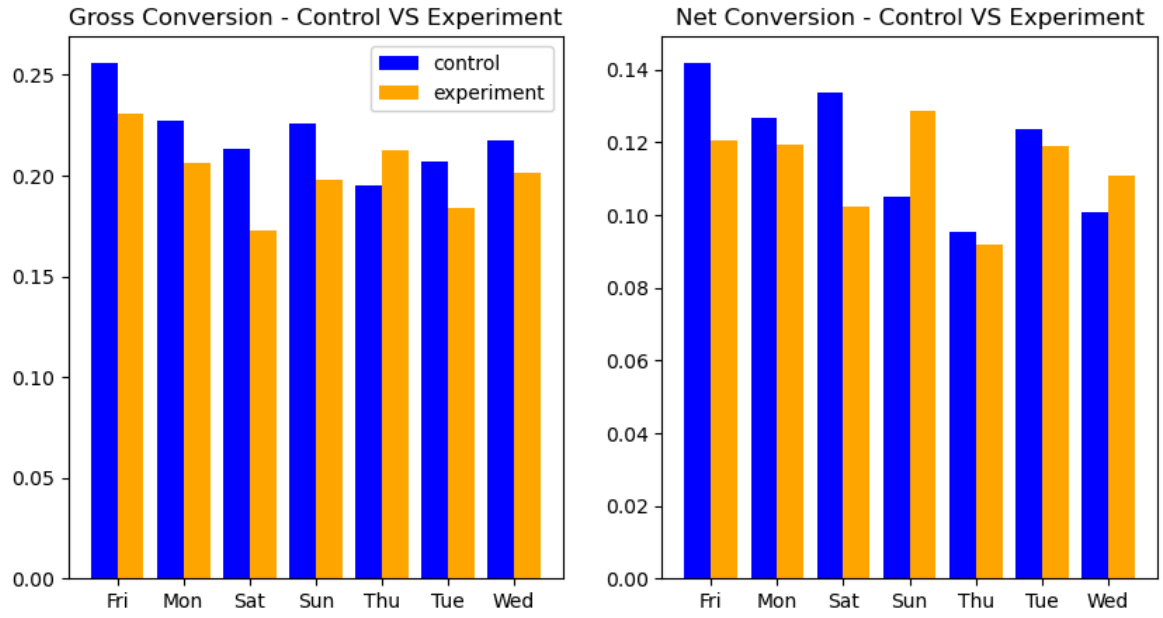
1.1 Gross Conversion

In [108]:
```
x = range(weekday.shape[0])
fig, (ax1, ax2) = plt.subplots(1, 2, figsize = (10, 5))

ax1.bar(x, weekday['GC_con'], color = 'blue', width = 0.4, label = 'control')
ax1.bar([i + 0.4 for i in x], weekday['GC_exp'], width = 0.4, color = 'orange', label = 'experiment')
ax1.set_xticks([i+0.2 for i in x])
ax1.set_xticklabels(wd)
ax1.legend()
ax1.set_title('Gross Conversion - Control VS Experiment')

ax2.bar(x, weekday['NC_con'], color = 'blue', width = 0.4, label = 'control')
ax2.bar([i + 0.4 for i in x], weekday['NC_exp'], width = 0.4, color = 'orange', label = 'experiment')
ax2.set_xticks([i+0.2 for i in x])
ax2.set_xticklabels(wd)
ax2.set_title('Net Conversion - Control VS Experiment')

plt.savefig(filepath+'weekly analysis')
plt.show()
```



In [ ]: