**AB testing**

A/B test of split test is a statistical experiment to compare 2 versions of products/ designs which known as control vs treatment. A/B includes

1. Develop new hypothesis
2. Design A/B test
3. Evaluation result
4. Making decision
5. Null hypothesis vs Alternative Ha
6. How long to run A/b test?   
   Sample size = 16 / with σ sample variance, δ different b/t treatment and control.

which involves information  
-type I error or power  
-significance level  
-minimum detectable effect

Note: σ sample variance is obtained from data, δ do not know before experiment so we use minimum detectable effect (e.g 0.1% increase in revenue)

Type 1 error vs Type 2 error

In statistics, **a type 1 error** means rejecting the null hypothesis when it's actually true, while **a type 2 error** means failing to reject the null hypothesis when it's actually false.

-----------------

------- Interview question---

Q: 10 tests are running with different landing page. 1 case won and the p value < 0.05. Should we make the change?

A: No

Since more than 2 variables, the probability of false discovery increase

For e: These are 3 groups, what is the chance of at least one false positive?

Pr(no false positive) = = = 0.857

Pr(at least 1 false positive) = 1 -Pr(no false positive) = 0.143

* Type I error > 14%

To deal with multiple testing problem is to use Bonferroni correction,  
new significant level = significant level/ number of test

**Real experience A/B test**

A retailer want to sell product online. A team want to provide a similar feature to show similar product in the shopping cart. The goal is to increase the revenue. However, there is a concern that this addition feature may distract customers and may delay or abandon check out. We need A/B test will be implemented with the following

Objective & Key metrics

Variants

Randomization units

Key metrics: Revenue

To deal with differences b/w group we need to normalize revenue by # of users in each group. So our metric is normalize revenue by # users or revenue per user

To guaranty a similar product, it should be generated by the same algorithm

In design test, we aim at the customers at check out. We determine how much increase in revenue per user to be considered outweigh the cost of (e.g) revenue increase $2 per user

* Power of the test (It is the probability of correctly rejecting the null hypothesis when it is false). Practical we use power of the test of 80%
* Significant level 5%

sample size = 16 / with σ sample variance, δ different b/t treatment and control

Bayesian test

<https://github.com/ajhalthor/bayesian-testing/>

the frequentist approach assigns probabilities to data, not to hypotheses, whereas the Bayesian approach assigns probabilities to hypotheses. Furthermore, Bayesian models incorporate prior knowledge into the analysis, updating hypotheses probabilities as more data become available

Q: What factor effect A/B Test

1. **Sample size**: ensure a large enough users to achieve statistically significant result
2. **Duration of the test**: Run the test for adequate period to account for variation in user behavior and external factor
3. **Statistical significance** Determine the level of confidence in the result (usually 95% or higher)
4. **Test validity**: Ensure the test design and execution are correct to avoid misleading result.

|  |  |
| --- | --- |
| parameter |  |
| Alpha | Smaller alpha high confidence level 🡺 more samples |
| Beta | Smaller beta, greater power 🡺 more sample |
| Variance | Smaller variance 🡺 less samples. Estimated from historical test |
| delta | Smaller delta more samples |

A/B test can be affected initially be the following factors but not last long

Novelty effect describes users who are interested in new features

Primacy effect describes users who are reluctant to adopt new features

To deal with:

+ run tests only on first time users since these effect do not affect new users

+ If the test is already running, compare the first time users to old users in treatments to get actual estimate the impact or no impact.

Q: We ran A/B test on a new feature. The test won and we relaunched the change. After a week the treatment effect quickly declined. How?

A: Novelty effect. This declines repeats until the effect wears off

Q: test a new feature to increase post created per user, assign each user randomly, test won by 1% in term of posts

What would happen after now feature is launched to all users?

* Will it be the same as 1%
* If not would be more or less?
* Assume no novelty effect

A: The difference will not be 1% due to

User behavior impacted by others

The effect can spill overs the control groups

In this case, the difference underestimates the treatment effect

Note; For 2 sided market as Uber, Lift, AirBnb. The resource shared among control and treatment group

The treatment group attracts more drivers so less drivers are available for control group

Q: Test a new feature to increase post created per user, assign each user randomly, test won by 1% in term of posts after now feature is launched to all users, we see The difference will not be 1% due to User behavior impacted by others The effect can spill overs the control groups How to deal with this case

A:

When introducing a new feature that aims to increase the number of posts created per user, and after observing a 1% increase during the testing phase, you might find that the impact is different once the feature is rolled out to all users. This discrepancy can be due to spillover effects, where the behavior of users in the control group is influenced by those in the treatment group.

### Dealing with Spillover Effects in A/B Testing

1. **Understand the Source of Spillover Effects**:
   * Identify how users in the control group might be influenced by those in the treatment group. For example, if users interact with each other, see each other's posts, or discuss the feature, this can lead to spillover.
2. **Designing the Experiment to Minimize Spillover**:
   * **Cluster Randomization**: Instead of randomizing individual users, randomize entire clusters of users. For example, if users are grouped by geography, social network, or another logical cluster, randomize at the cluster level to minimize interaction between treatment and control groups.
   * **Geographical Separation**: If feasible, assign users from different geographical locations to the treatment and control groups to reduce interaction.
   * **Network Analysis**: Use network analysis to identify and minimize connections between treatment and control groups.
3. **Measurement and Analysis Techniques**:
   * **Difference-in-Differences (DiD)**: Use DiD to control for time trends that affect both groups. This method can help isolate the impact of the new feature by comparing the pre-post difference in the treatment group against the control group.
   * **Instrumental Variables (IV)**: If there are strong instruments that affect the likelihood of receiving treatment but are not directly related to the outcome, IV methods can help address endogeneity issues caused by spillovers.
   * **Propensity Score Matching (PSM)**: Match users in the treatment and control groups on observable characteristics to create more comparable groups and reduce bias.
   * **Social Network Analysis**: Incorporate users' network structures into the analysis to account for indirect effects. This can help quantify and control for the impact of spillover effects.

### Example Scenario and Adjusted Analysis

Let’s assume we use cluster randomization and Difference-in-Differences (DiD) to analyze the impact. Here’s how you might approach it:

1. **Cluster Randomization**:
   * Randomly assign clusters (e.g., social groups, regions) to treatment and control groups.
2. **Data Collection**:
   * Collect data on the number of posts per user before and after the feature launch for both treatment and control clusters.
3. **Difference-in-Differences Analysis**:
   * Use DiD to compare the changes over time between treatment and control clusters.

import numpy as np

import pandas as pd

import statsmodels.api as sm

import matplotlib.pyplot as plt

# Simulate Data

np.random.seed(42)

n\_clusters = 50

cluster\_size = 100

# Generate random data for clusters

data = pd.DataFrame({

'cluster': np.repeat(np.arange(n\_clusters), cluster\_size),

'user': np.arange(n\_clusters \* cluster\_size),

'pre\_posts': np.random.poisson(lam=10, size=n\_clusters \* cluster\_size),

'post\_posts': np.random.poisson(lam=10, size=n\_clusters \* cluster\_size)

})

# Assign treatment randomly at the cluster level

data['treatment'] = np.where(data['cluster'] % 2 == 0, 1, 0)

# Apply treatment effect

data.loc[data['treatment'] == 1, 'post\_posts'] += 1

# Calculate difference in posts

data['diff'] = data['post\_posts'] - data['pre\_posts']

# Aggregate data by cluster

cluster\_data = data.groupby('cluster').agg({

'treatment': 'first',

'pre\_posts': 'mean',

'post\_posts': 'mean',

'diff': 'mean'

}).reset\_index()

# Difference-in-Differences Analysis

X = cluster\_data[['treatment']]

X = sm.add\_constant(X)

y = cluster\_data['diff']

model = sm.OLS(y, X).fit()

print(model.summary())

# Plotting the Results

plt.figure(figsize=(10, 6))

for name, group in cluster\_data.groupby('treatment'):

plt.plot(group['cluster'], group['diff'], marker='o', linestyle='', label='Treatment' if name == 1 else 'Control')

plt.axhline(y=0, color='grey', linestyle='--')

plt.title('Difference in Differences')

plt.xlabel('Cluster')

plt.ylabel('Difference in Posts (Post - Pre)')

plt.legend()

plt.show()

### Explanation

1. **Cluster Randomization**:
   * Clusters are randomized into treatment and control groups.
2. **Simulated Data**:
   * Simulate pre and post number of posts per user.
3. **Apply Treatment Effect**:
   * For simplicity, we assume a constant effect of adding one post in the treatment group.
4. **Difference-in-Differences Analysis**:
   * Perform DiD analysis by comparing the average difference in posts between treatment and control clusters.

Difference-in-Differences (DiD) is a statistical technique used in causal inference to estimate the treatment effect of a policy intervention or treatment. It is particularly useful in observational studies where random assignment is not possible. DiD compares the changes in outcomes over time between a treatment group (affected by the intervention) and a control group (not affected by the intervention).

### When to Use Difference-in-Differences

1. **Non-Randomized Studies**:
   * DiD is ideal for quasi-experimental designs where random assignment is not feasible, and you need to control for time-invariant unobserved heterogeneity.
2. **Pre-Post Design with a Control Group**:
   * Use DiD when you have data before and after the treatment for both the treatment and control groups. This allows you to control for trends that affect both groups equally.
3. **Policy Evaluations**:
   * Commonly used in policy evaluations where some regions or populations are exposed to a policy change (treatment group) and others are not (control group).
4. **Longitudinal Data**:
   * DiD requires data over multiple time periods. It leverages the repeated observations to account for unobserved confounders that are constant over time.

### Assumptions

1. **Parallel Trends Assumption**:
   * The key assumption is that in the absence of treatment, the average change in the outcome would have been the same for both the treatment and control groups. This implies that both groups would have followed parallel trends over time.
2. **No Spillover Effects**:
   * The treatment should not affect the control group. There should be no spillover effects where the treatment influences the control group’s outcome.

### Example Implementation in Python

Here’s a simplified implementation of DiD in Python:

1. **Simulate Data**: Create synthetic data for a treatment and control group over time.
2. **Conduct DiD Analysis**: Estimate the treatment effect using DiD.

import numpy as np

import pandas as pd

import statsmodels.api as sm

import matplotlib.pyplot as plt

# Step 1: Simulate Data

np.random.seed(42)

n = 1000 # Number of observations

time\_periods = 4 # Number of time periods

# Create a DataFrame to hold the data

data = pd.DataFrame({

'id': np.repeat(np.arange(n), time\_periods),

'time': np.tile(np.arange(time\_periods), n)

})

# Assign treatment group randomly (1 for treatment, 0 for control)

data['treatment'] = np.where(data['id'] % 2 == 0, 1, 0)

# Create an interaction term for treatment and post-intervention periods

data['post'] = np.where(data['time'] >= 2, 1, 0) # Assume intervention happens at time=2

data['treat\_post'] = data['treatment'] \* data['post']

# Generate outcome variable with some random noise

data['outcome'] = 50 + 5 \* data['treatment'] + 10 \* data['post'] + 15 \* data['treat\_post'] + np.random.normal(0, 5, size=n \* time\_periods)

# Step 2: Conduct DiD Analysis

# Create dummy variables for time periods

time\_dummies = pd.get\_dummies(data['time'], prefix='time', drop\_first=True)

data = pd.concat([data, time\_dummies], axis=1)

# Define the model

X = data[['treatment', 'post', 'treat\_post'] + list(time\_dummies.columns)]

X = sm.add\_constant(X)

y = data['outcome']

model = sm.OLS(y, X).fit()

# Output the results

print(model.summary())

# Step 3: Visualization

# Plot the average outcome by group and time

avg\_outcome = data.groupby(['time', 'treatment'])['outcome'].mean().unstack()

avg\_outcome.plot(kind='line', marker='o')

plt.title('Difference-in-Differences')

plt.xlabel('Time')

plt.ylabel('Average Outcome')

plt.legend(['Control', 'Treatment'])

plt.show()

 **Data Simulation**:

* We simulate data for 1000 individuals over 4 time periods.
* The treatment group is randomly assigned, and the intervention occurs at time=2.
* The outcome variable is generated with an interaction term to reflect the treatment effect.

 **Conduct DiD Analysis**:

* We use OLS regression to estimate the treatment effect, including interaction terms and time dummies.
* The treat\_post variable captures the DiD estimator.

+++++++++++++++++++++++++++++++++++++++++++++++++

**What is guard rail metrics**

Guardrail metrics (counter metric) are business metrics designs to indirectly measure business value and provide alerts about any potential misleading or erroneous results

Eg:

|  |  |  |
| --- | --- | --- |
|  | Month 1 | Month 2 |
| # app download | 100 | 50 |
| # sign up | 30 | 25 |
| #signup/#dowload | 30% | 50% |

Using #signup/#download can misleading as it shows the increase but actually performs decrease

Define a success metric steps

1. Clarify goal, metric, functionality. Mission and vision
2. Product/ customer goal
3. Funnel metrics: identify key actions, define a time frame
4. Priority
5. Critique metrics

**What is a difference bw A/A test and AB test**

Performing an A/A test is similar to that of an A/B test. Users are divived into similar size groups. Each group interacts with identical version of same piece of contents.

A/A is designed to help marketers examine the reliability of the A/B test used with the goal to find no difference b/w the control and variant

Q: **Using delta method for a/b test**

<https://medium.com/@ahmadnuraziz3/applying-delta-method-for-a-b-tests-analysis-8b1d13411c22>

Q: What is causal ML? and it used

<https://medium.com/datareply/causal-ml-the-future-of-decision-making-with-transparency-and-explainability-39e6f1aa2074>

**Retail optimization engine with Causal ML**

Let say we work for a pipeline to better price items for a large multinational fashion manufacturer and retailer.

Several traditional methods revolving around standard regression techniques to work out both pricing and promotions for items. While this worked well for them, they were at a loss to explain to the stores why they had chosen the prices they had obtained.

**Uplift modelling for automotive**

Let say we have two scenarios, where in one scenario a customer is offered complementary extended warranty, and in the other they must pay for it, how much more likely are they to buy the car? Or does it even have an effect?

using causality through **uplift modelling**, we can identify which variables to change if we want to maximize the probability of selling. By applying causal techniques, we simulate scenarios where factors (such as offering complementary warranty, or sending a newsletter) are applied or removed, and we can see the resulting increase or decrease in probability of buying. The true value for the client comes from knowing not only which discounts or marketing campaigns are necessary to get customers to buy, but whether they are even necessary i.e., if the customer would have bought anyway.

**Another application of causal ML is a alternative solution for A/B Test**

**Q: What are assumption on A/B test**

To apply an A/B test effectively, certain assumptions should be met to ensure the validity and reliability of the results:

**Random Assignment**: Participants should be randomly assigned to either the control group (A) or the treatment group (B). This randomization helps ensure that any differences observed between the groups are due to the treatment effect and not other factors.

**Independence**: Each participant's response in one group should be independent of the responses of others in the same group. This means that one participant's behavior or response should not influence another participant's behavior or response.

**Normality**: The distribution of the outcome variable (e.g., conversion rate, click-through rate) should be approximately normal within each group. If the outcome variable is highly skewed, transformation or non-parametric tests may be more appropriate.

**Homogeneity of Variance**: The variance of the outcome variable should be similar across the groups. This assumption is important for the validity of the statistical tests used to compare the means between groups (e.g., t-test, ANOVA).

**Large Sample Size**: While not always a strict requirement, larger sample sizes tend to provide more reliable results. Adequate sample sizes ensure that the observed effects are not simply due to random chance.

**Consistency of Treatment Application**: Ensure that the treatment is administered consistently across all participants in the treatment group to avoid confounding factors that could bias the results.

**No Interference**: Participants in one group should not be influenced by the treatment received by participants in the other group. This helps maintain the integrity of the experimental design.

**No History Effects**: External events or changes occurring during the experiment should not systematically affect one group more than the other. This ensures that any differences observed between groups are indeed due to the treatment and not other factors.

**Measurement Validity**: The outcome variable should accurately measure what it is intended to measure. Valid and reliable measurement tools are crucial for drawing meaningful conclusions from the experiment.

Q: Compare Causal ML vs A/B test

<https://towardsdatascience.com/using-causal-ml-instead-of-a-b-testing-eeb1067d7fc0>

Causal ML is a powerful tool because it is more flexible than A/B Testing, and it doesn’t require strong assumptions

The problem of AB test is to satisfy its assumption

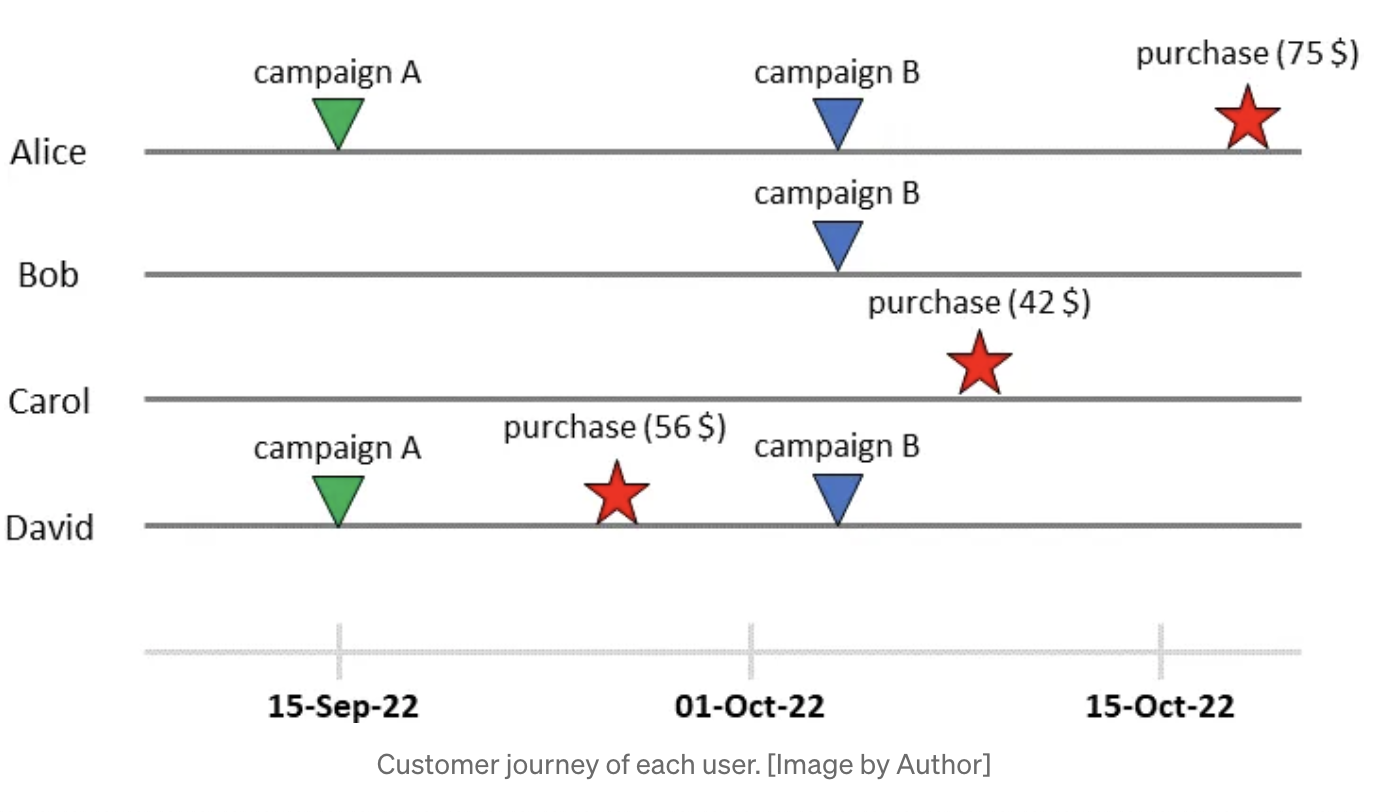
Often AB test is fine but One of its requirement is not running too many tests at the same time, because they can “contaminate” each other’s outcomes.

For example,

+ different teams sent conflicting marketing campaigns (e.g. retention campaigns and up-selling campaigns) to the same users.

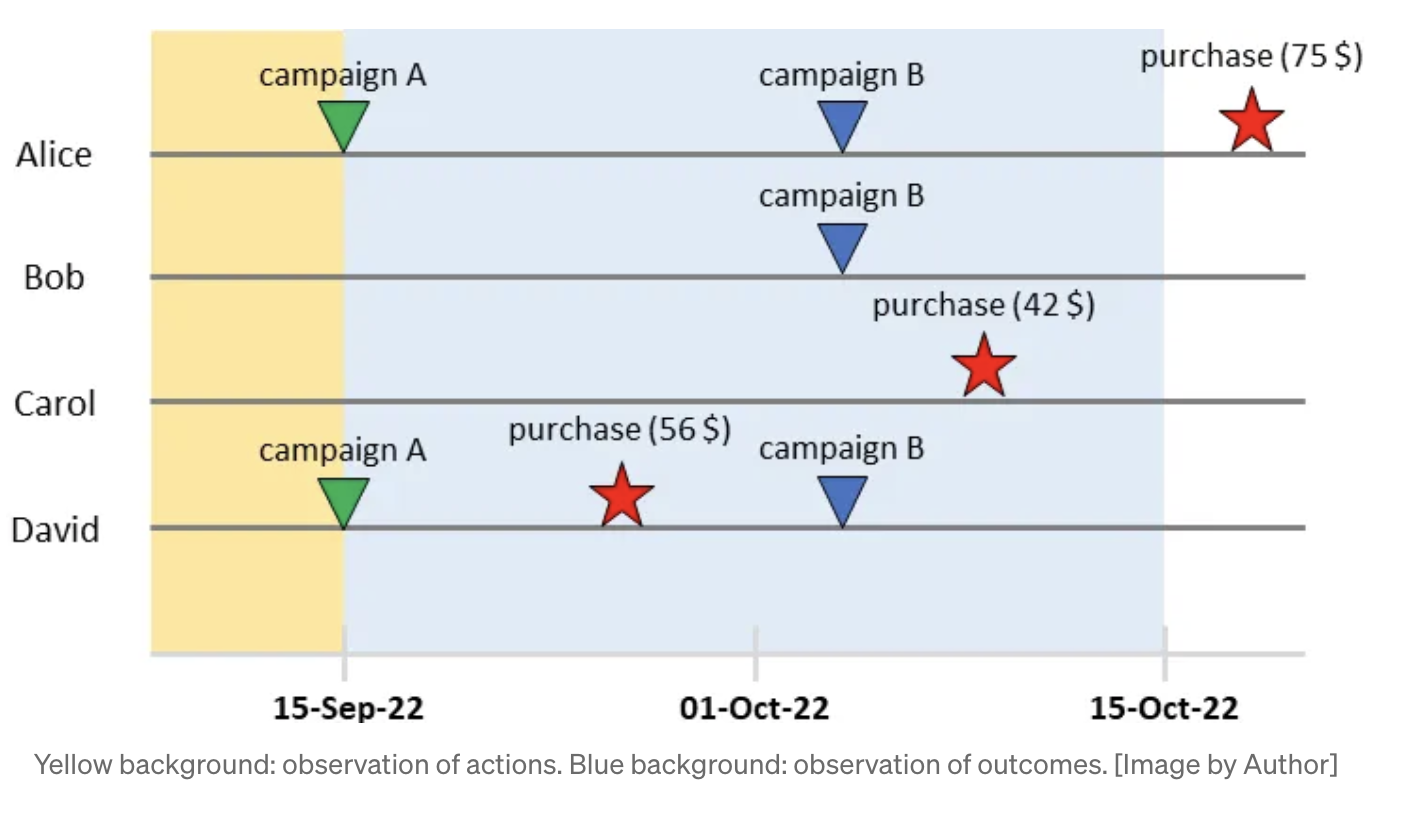
+ Some teams sent a marketing campaign to users that belong to the control group of another team.

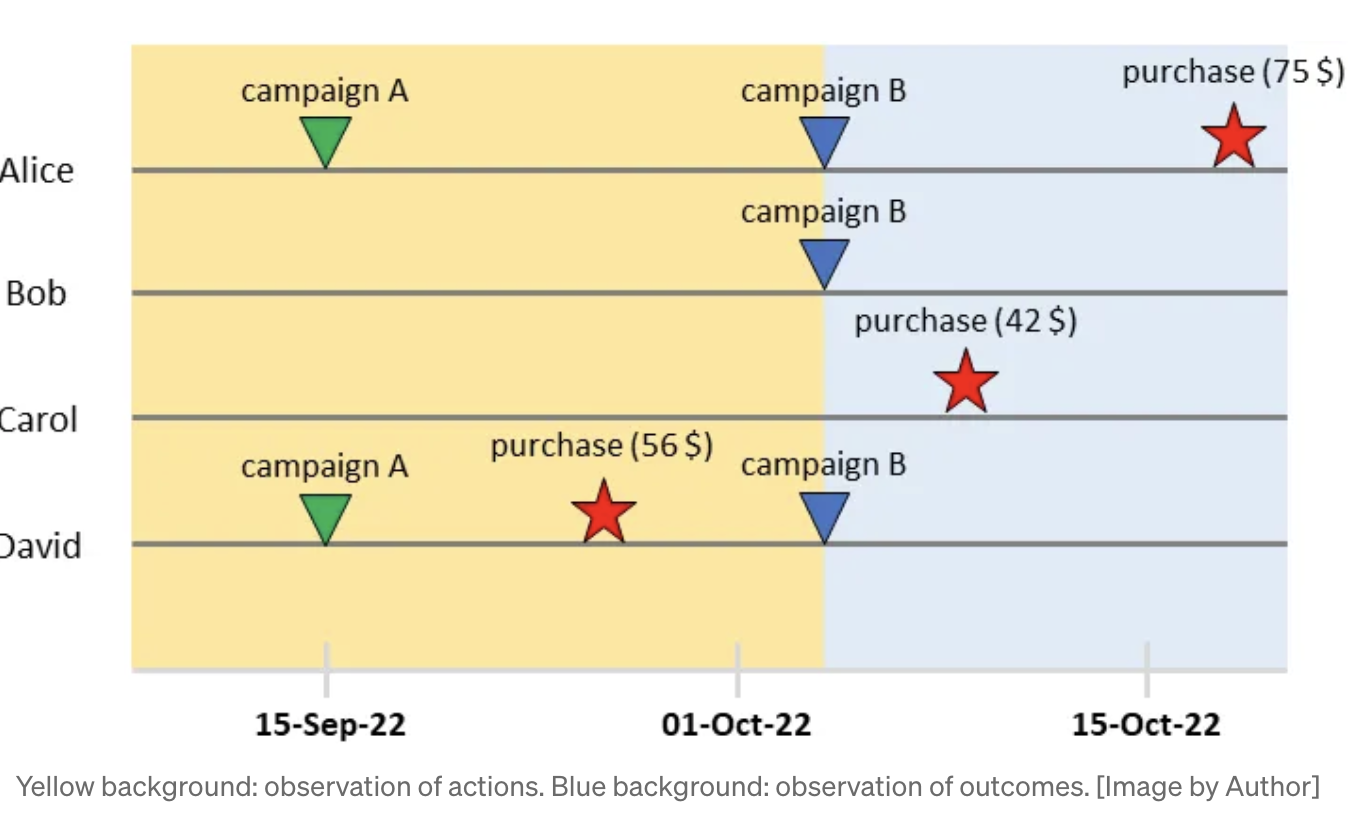
As a result, we are not sure which result is from which campaign



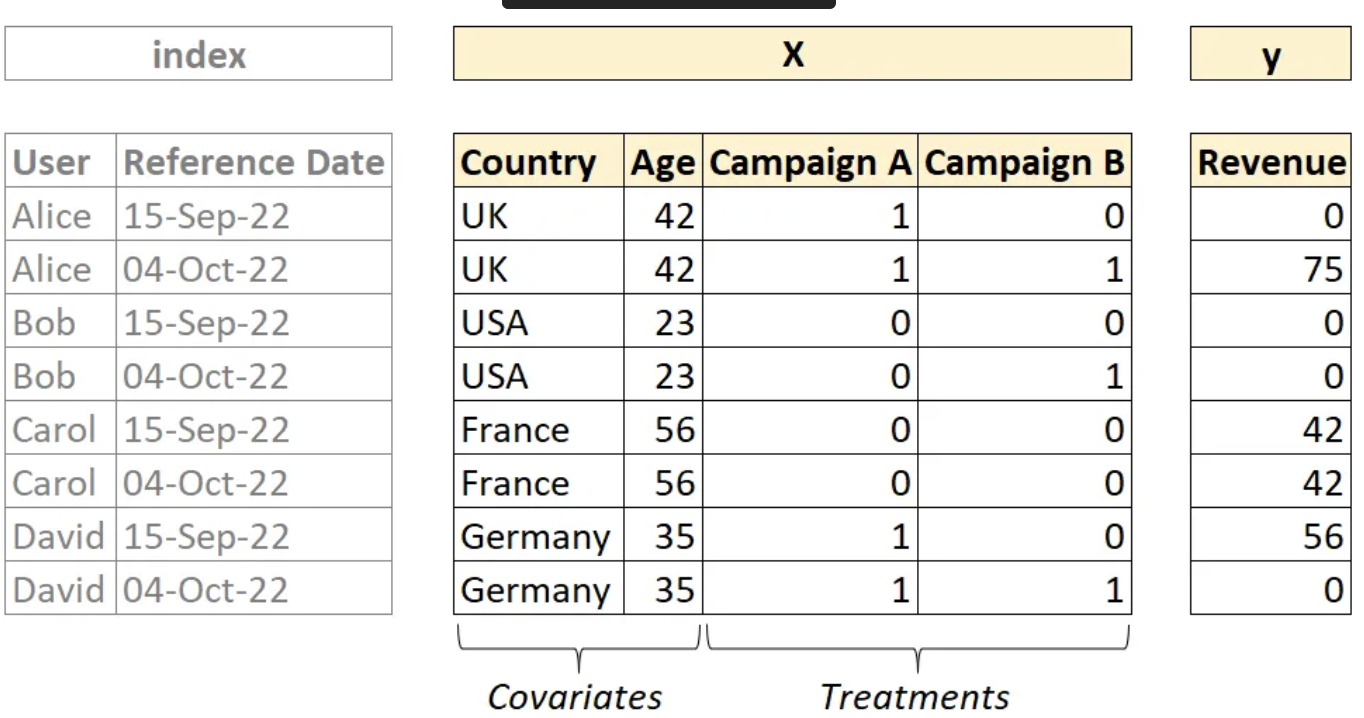
As we see from above, does purchase of $75 is a result of campaign A or B?

Causal Ml determines cutoff Oct 15 to examine the action belong to campaign A





Putting covariates, treatments and outcomes together brings us to rearrange our initial data in the following form:



**Explain Bayesian testing**

Implementing Bayesian A/B testing involves several steps. Below is a high-level outline of the process:

1. **Formulate Hypotheses**:
   * Define the null hypothesis *H*0​ (e.g., the conversion rates of variant A and variant B are equal) and the alternative hypothesis *H*1​ (e.g., the conversion rate of variant A is higher than variant B).
2. **Choose Prior Distributions**:
   * Select appropriate prior distributions for the parameters of interest (e.g., conversion rates). This could involve using non-informative priors, such as uniform or Jeffrey's priors, or informative priors based on historical data or expert knowledge.
3. **Collect Data**:
   * Randomly assign users or samples to variant A and variant B (control and treatment groups).
   * Collect data on the outcomes of interest (e.g., conversions) from both groups over a defined period.
4. **Update Posteriors with Data**:
   * Use Bayes' theorem to update the prior distributions with the observed data and compute posterior distributions for the parameters of interest (e.g., conversion rates for variant A and variant B).
   * This step involves calculating likelihood functions based on the observed data and combining them with the prior distributions to obtain posterior distributions using Markov chain Monte Carlo (MCMC) methods, variational inference, or other Bayesian estimation techniques.
5. **Compare Posterior Distributions**:
   * Compare the posterior distributions of the parameters between variant A and variant B to assess differences in conversion rates.
   * Calculate summary statistics (e.g., means, credible intervals) and perform hypothesis tests to determine the probability of superiority or inferiority of one variant over the other.
6. **Make Decisions**:
   * Based on the posterior distributions and hypothesis tests, make decisions about whether to accept or reject the null hypothesis and which variant to choose.
   * Consider factors such as effect size, uncertainty, and practical significance in decision-making.
7. **Monitor and Iterate**:
   * Continuously monitor the performance of the chosen variant and collect additional data over time.
   * Iterate the Bayesian A/B testing process as needed, updating priors, collecting new data, and refining hypotheses to optimize conversion rates.

It's important to note that implementing Bayesian A/B testing requires expertise in Bayesian statistics, probability theory, and data analysis. Additionally, appropriate software libraries or tools for Bayesian inference, such as PyMC3, Stan, or JAGS, may be needed to perform the computations efficiently.

Question: **implement a Bayesian A/b test in python in assumption that new feature increase the subscription**

Answer:

here's a basic implementation of a Bayesian A/B test in Python assuming that a new feature increases subscription rates. We'll use the PyMC3 library for Bayesian modeling:

import pymc3 as pm

import numpy as np

# Simulated data for control group (existing feature) and treatment group (new feature)

control\_data = np.random.binomial(n=1, p=0.1, size=1000) # Assuming a 10% subscription rate for control

treatment\_data = np.random.binomial(n=1, p=0.15, size=1000) # Assuming a 15% subscription rate for treatment

# Define Bayesian model

with pm.Model() as model:

# Prior distributions for conversion rates of control and treatment groups

p\_control = pm.Beta('p\_control', alpha=1, beta=1)

p\_treatment = pm.Beta('p\_treatment', alpha=1, beta=1)

# Likelihood of observed data

obs\_control = pm.Bernoulli('obs\_control', p=p\_control, observed=control\_data)

obs\_treatment = pm.Bernoulli('obs\_treatment', p=p\_treatment, observed=treatment\_data)

# Difference in conversion rates

delta = pm.Deterministic('delta', p\_treatment - p\_control)

# Define metrics for comparison

p\_better = pm.Deterministic('p\_better', delta > 0)

p\_superior = pm.Deterministic('p\_superior', delta > 0.05) # Define threshold for practical significance

# Run sampling

trace = pm.sample(draws=10000, tune=1000, cores=1)

# Summarize results

pm.summary(trace)

# Plot posterior distributions

pm.plot\_posterior(trace)

This code simulates data for a control group and a treatment group, with subscription rates of 10% and 15%, respectively. We then define a Bayesian model with Beta priors for the conversion rates of both groups. The observed data are modeled as Bernoulli random variables with the respective conversion rates. We calculate the difference in conversion rates (**delta**) and define metrics for comparison (**p\_better** and **p\_superior**). Finally, we run MCMC sampling using the PyMC3 library and summarize the results.

Please note that this is a simplified example for demonstration purposes. In a real-world scenario, you would replace the simulated data with actual data from your A/B test and adjust the model accordingly. Additionally, you may need to consider other factors such as sample size, prior information, and potential confounding variables.

-----------------------

---------

**implement Bayesian A/B test in python**

a basic implementation of Bayesian A/B testing in Python using the PyMC3 library:

import pymc3 as pm

import numpy as np

# Simulated data for A/B testing

np.random.seed(42)

n\_control = 1000

n\_treatment = 1000

control\_data = np.random.binomial(n=1, p=0.1, size=n\_control) # Conversion rate of 10% for control group

treatment\_data = np.random.binomial(n=1, p=0.12, size=n\_treatment) # Conversion rate of 12% for treatment group

# Define the Bayesian A/B testing model

with pm.Model() as ab\_model:

# Priors for the conversion rates of control and treatment groups

p\_control = pm.Beta('p\_control', alpha=1, beta=1)

p\_treatment = pm.Beta('p\_treatment', alpha=1, beta=1)

# Likelihood of observed data

control\_likelihood = pm.Bernoulli('control\_likelihood', p=p\_control, observed=control\_data)

treatment\_likelihood = pm.Bernoulli('treatment\_likelihood', p=p\_treatment, observed=treatment\_data)

# Difference in conversion rates

diff\_conversion = pm.Deterministic('diff\_conversion', p\_treatment - p\_control)

# Perform Bayesian inference

with ab\_model:

trace = pm.sample(draws=2000, tune=1000, chains=2)

# Plot posterior distributions

pm.plot\_posterior(trace, var\_names=['p\_control', 'p\_treatment', 'diff\_conversion'], ref\_val=0)

In this example:

* We simulate data for a control group and a treatment group with different conversion rates.
* We define a Bayesian model using PyMC3 with Beta priors for the conversion rates of both groups.
* We specify Bernoulli likelihoods for the observed data (conversion outcomes) in both groups.
* We compute the difference in conversion rates between the treatment and control groups as a deterministic variable.
* We perform Bayesian inference using Markov chain Monte Carlo (MCMC) sampling.
* Finally, we plot the posterior distributions of the conversion rates for both groups and the difference in conversion rates.

Question: Compare A/B test vs Bayesian A/B test

Answer:

Let's compare traditional A/B testing with Bayesian A/B testing across several aspects:

1. **Statistical Inference:**
   * **A/B Testing:** In traditional A/B testing, statistical inference is typically based on frequentist methods. Confidence intervals and p-values are commonly used to determine whether there is a statistically significant difference between the control and treatment groups.
   * **Bayesian A/B Testing:** Bayesian A/B testing provides a different approach to statistical inference. It allows for the direct estimation of probability distributions for parameters of interest (e.g., conversion rates), which can provide richer information about uncertainty and variability. Instead of p-values, Bayesian methods often use credible intervals and posterior probabilities.
2. **Handling Uncertainty:**
   * **A/B Testing:** Frequentist methods provide point estimates (e.g., mean conversion rate) along with confidence intervals, which capture uncertainty around the estimate. However, they do not directly quantify the probability that a parameter falls within a certain range.
   * **Bayesian A/B Testing:** Bayesian methods naturally quantify uncertainty by providing posterior distributions for parameters. This allows for more intuitive interpretation and decision-making, as probabilities can be assigned to statements such as "the treatment is better than the control with 95% probability."
3. **Flexibility and Adaptability:**
   * **A/B Testing:** Traditional A/B testing typically requires predefined sample sizes and fixed analysis plans. It can be challenging to adapt the experiment mid-way based on interim results, and multiple testing corrections may be necessary for sequential analyses.
   * **Bayesian A/B Testing:** Bayesian methods offer greater flexibility, as they allow for sequential analysis and updating of beliefs as new data becomes available. This enables adaptive experimentation, where sample sizes can be adjusted dynamically based on accumulating evidence, potentially leading to more efficient use of resources.
4. **Interpretation and Decision-making:**
   * **A/B Testing:** Frequentist methods provide clear rules for decision-making based on pre-specified significance levels (e.g., α = 0.05). However, interpretation can sometimes be challenging, especially when dealing with multiple comparisons or complex experimental designs.
   * **Bayesian A/B Testing:** Bayesian methods provide intuitive interpretations of results in terms of probabilities and distributions. Decision-making can be based on posterior probabilities directly, taking into account both the magnitude of effects and the uncertainty around them. This can lead to more informed and nuanced decisions, especially in situations where the stakes are high.

In summary, while traditional A/B testing and Bayesian A/B testing both aim to compare treatments and make data-driven decisions, they differ in their underlying statistical frameworks, handling of uncertainty, flexibility, and interpretation of results. The choice between the two approaches depends on factors such as the nature of the problem, available resources, and decision-maker preferences.

Let say we work for a pipeline to better price items for a large multinational fashion manufacturer and retailer. Several traditional methods revolving around standard regression techniques to work out both pricing and promotions for items. How Retail optimization engine with Causal ML can solve the problem of optimization price

Solution

Retail optimization, especially for pricing items effectively, is a complex problem that can be addressed using advanced methods like Causal Machine Learning (Causal ML). Here's how a retail optimization engine leveraging Causal ML can solve pricing optimization problems for a large multinational fashion manufacturer and retailer:

**Traditional Regression Techniques vs. Causal ML**

**Traditional Regression Techniques:**

* Use historical data to fit models that predict sales or revenue based on pricing and other features.
* Often rely on linear or logistic regression, decision trees, or other machine learning models to find correlations between price and sales.
* Can suffer from issues related to confounding variables and do not always provide causal insights.

**Causal ML:**

* Focuses on identifying causal relationships rather than mere correlations.
* Helps in understanding the actual impact of pricing changes on sales and revenue.
* Can handle confounding variables more effectively, providing more reliable insights for decision-making.

**Steps to Implement a Retail Optimization Engine Using Causal ML**

1. **Data Collection and Preparation:**
   * Collect historical data on prices, sales, promotions, inventory levels, customer demographics, and external factors (e.g., economic indicators, seasonal trends).
   * Clean and preprocess the data to handle missing values, outliers, and ensure consistency.
2. **Define Treatment and Outcome:**
   * **Treatment**: The pricing strategy or promotion applied to items.
   * **Outcome**: Sales volume, revenue, profit margins, or customer conversion rates.
3. **Identify Confounding Variables:**
   * Variables that influence both the treatment and the outcome, such as marketing efforts, competitor actions, and macroeconomic conditions.
   * Use domain knowledge and statistical methods to identify and control for these confounders.
4. **Causal Inference Techniques:**
   * **Randomized Controlled Trials (RCTs)**: If feasible, conduct A/B tests or controlled experiments where prices are varied for a subset of items or locations.
   * **Instrumental Variables (IV)**: Identify instruments that affect the treatment (price) but not the outcome (sales) directly, to account for endogeneity.
   * **Propensity Score Matching (PSM)**: Match treated and untreated items on observable characteristics to control for confounding.
   * **Difference-in-Differences (DiD)**: Compare changes in outcomes over time between treated and control groups to isolate the treatment effect.
   * **Regression Discontinuity Design (RDD)**: Exploit cutoff points (e.g., prices just above or below a threshold) to identify causal effects.
5. **Causal ML Models:**
   * **Causal Trees and Forests**: Use tree-based models designed to estimate heterogeneous treatment effects.
   * **Double Machine Learning (DML)**: Combines machine learning models with causal inference techniques to control for confounders and estimate treatment effects robustly.
   * **Bayesian Structural Time Series (BSTS)**: Model time series data to estimate the impact of interventions over time.
6. **Optimization and Decision Making:**
   * Use the estimated causal effects to build optimization models that maximize revenue or profit.
   * Implement algorithms like reinforcement learning to adapt pricing strategies dynamically based on real-time data and feedback.
   * Consider constraints such as inventory levels, competitor pricing, and customer preferences in the optimization process.

**Example Workflow Using Python**

Here's a simplified workflow to illustrate how Causal ML can be implemented for pricing optimization:

import pandas as pd

import numpy as np

import statsmodels.api as sm

from causalml.inference.meta import LRSRegressor

from sklearn.model\_selection import train\_test\_split

# Step 1: Data Collection and Preparation

data = pd.read\_csv('historical\_sales\_data.csv')

data = data.dropna()

# Step 2: Define Treatment and Outcome

treatment = data['price\_change']

outcome = data['sales']

# Step 3: Identify Confounding Variables

confounders = data[['promotion', 'season', 'competitor\_price']]

# Step 4: Causal Inference Techniques

# Using a meta-learner for causal inference

X = pd.concat([treatment, confounders], axis=1)

y = outcome

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

treatment\_train, treatment\_test = train\_test\_split(treatment, test\_size=0.2, random\_state=42)

# Step 5: Causal ML Models

# Using a meta-learner for causal inference

learner = LRSRegressor()

learner.fit(X\_train, treatment\_train, y\_train)

# Estimate treatment effects

te = learner.predict(X\_test)

# Step 6: Optimization and Decision Making

# Use the estimated treatment effects to inform pricing decisions

# This could involve creating a price optimization function

def optimize\_price(te, base\_price, margin):

optimal\_price = base\_price + te \* margin

return optimal\_price

data['optimal\_price'] = optimize\_price(te, data['base\_price'], data['profit\_margin'])

# Print the results

print(data[['item\_id', 'base\_price', 'optimal\_price']])

### Explanation of the Example

1. **Data Collection and Preparation**:
   * Load and clean historical sales data.
2. **Define Treatment and Outcome**:
   * Define the treatment (price change) and the outcome (sales).
3. **Identify Confounding Variables**:
   * Select relevant confounders that might influence both treatment and outcome.
4. **Causal Inference Techniques**:
   * Split the data into training and testing sets.
   * Use a meta-learner (e.g., LRSRegressor from causalml) to estimate the treatment effects.
5. **Optimization and Decision Making**:
   * Use the estimated treatment effects to optimize prices.
   * Define an optimization function to calculate the optimal price based on treatment effects and profit margins.

By using Causal ML, the retail optimization engine can provide more accurate and reliable pricing strategies that account for the true causal impact of price changes on sales, leading to better decision-making and improved profitability.

===================================================================

Q: **assuming you are interview for product data scientist position. Walk me through the following steps**: 1. Identify key metric 2. formulate hypothesis 3. step through component power analytics 4. propose possible solution

Answer:

1. **Identify Key Metric:** As a product data scientist, the first step is to understand the business objectives and identify the key metric(s) that align with those objectives. This could be revenue, user engagement, retention rate, conversion rate, or any other relevant metric depending on the product and business goals. For example, if we're working on an e-commerce platform, the key metric might be revenue per user.
2. **Formulate Hypothesis:** Once we have identified the key metric, we need to formulate a hypothesis about factors that could influence that metric. This involves brainstorming potential drivers or variables that might impact the key metric. For instance, in the case of revenue per user, hypotheses could include the impact of marketing campaigns, website usability, product pricing, or customer demographics. A hypothesis could be something like "Increasing the visibility of product recommendations on the homepage will lead to a higher average order value."
3. **Step Through Component Power Analytics:** With hypotheses in mind, we proceed to conduct exploratory data analysis to understand the current state of the data and its distribution. This involves examining trends, patterns, and correlations in the data. We can use statistical methods, visualization techniques, and machine learning models to gain insights into the relationships between different variables and the key metric. For example, we might analyze the click-through rates of different product recommendations or the impact of different marketing channels on user engagement and revenue.
4. **Propose Possible Solutions:** Based on the insights gained from the data analysis, we can propose possible solutions or interventions to test our hypotheses and improve the key metric. These solutions could involve changes to product features, marketing strategies, pricing strategies, user experience improvements, or other initiatives. For instance, if our analysis suggests that product recommendations have a significant impact on revenue per user, we might propose A/B testing different recommendation algorithms or placement strategies to optimize revenue.

Overall, the goal of this process is to leverage data-driven insights to inform decision-making and drive improvements in the key metric(s) of interest.

Top of Form

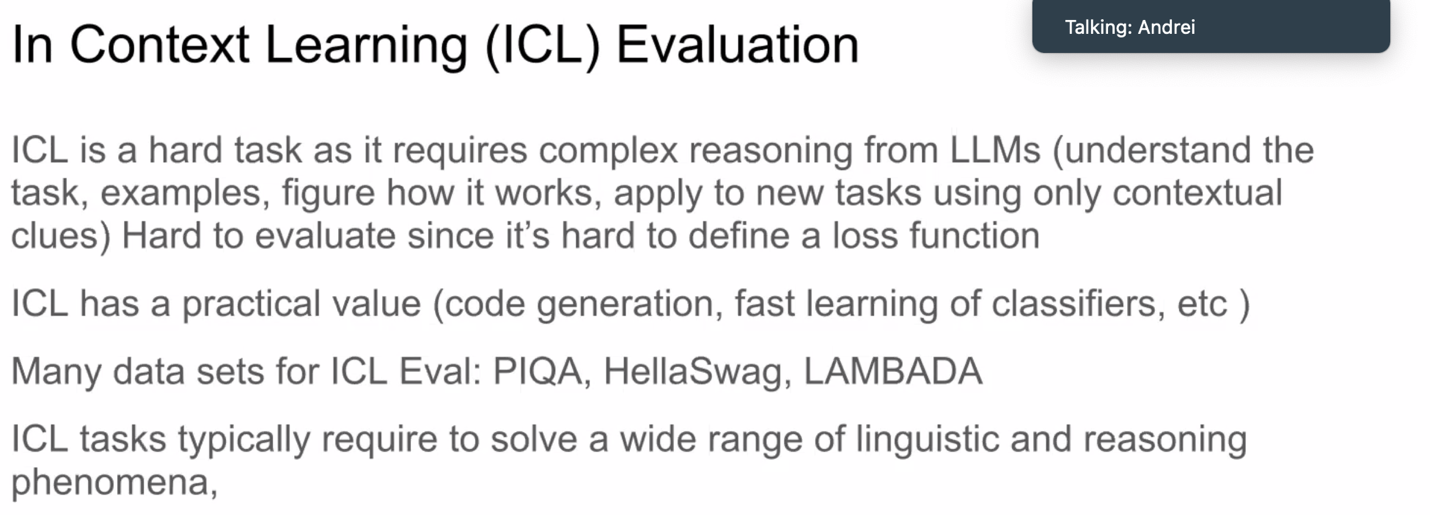
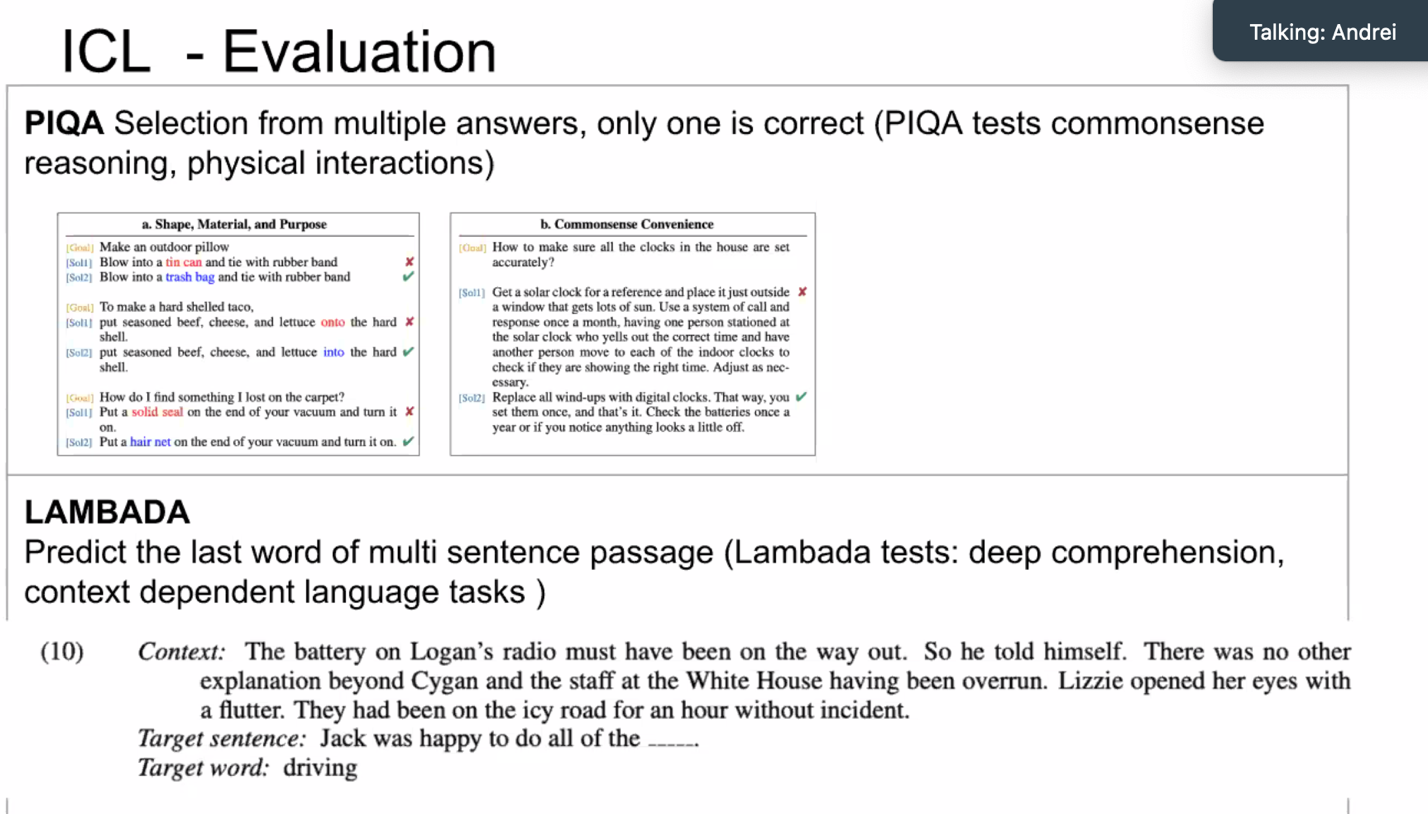
Q: **assuming you are interview for product data scientist position which acquisition of new customer is a key metric. Walk me through the following steps: 1. Identify key metric 2. formulate hypothesis 3. step through component power analytics 4. propose possible solution**Bottom of Form

Answer:

let's go through each step:

1. **Identify Key Metric:** In this case, the key metric is the acquisition of new customers. This metric indicates the effectiveness of the company's efforts in attracting and converting new users. It could be measured in terms of the number of new sign-ups, the number of new accounts created, or any other relevant metric that signifies the expansion of the customer base.
2. **Formulate Hypothesis:** We can start by brainstorming potential factors that could influence the acquisition of new customers. These factors might include marketing channels, advertising campaigns, product features, user demographics, pricing strategies, or competitive landscape. A hypothesis could be something like "Increasing targeted advertising on social media platforms will lead to a higher number of new customer sign-ups."
3. **Step Through Component Power Analytics:** In this step, we delve into the data to understand the current state of customer acquisition and its components. We can analyze the performance of different marketing channels (e.g., social media, search engine marketing, email campaigns), track user behavior on the website/app, and examine conversion funnels to identify potential bottlenecks or areas for improvement. We might use techniques such as cohort analysis, attribution modeling, and conversion rate optimization to gain insights into the effectiveness of various acquisition strategies.
4. **Propose Possible Solutions:** Based on the insights gained from the data analysis, we can propose potential solutions to increase new customer acquisition. These solutions could involve optimizing marketing campaigns to target specific demographics or interests, improving the user experience to increase conversion rates, experimenting with different pricing strategies or promotions, or exploring partnerships or collaborations to reach new audiences. For example, we might propose launching a referral program to incentivize existing customers to refer their friends or colleagues, thereby expanding the customer base through word-of-mouth marketing.

By following these steps, we can leverage data-driven insights to inform strategies and tactics aimed at increasing the acquisition of new customers.



There are two A/B test designs: Randomized vs Match Paired design

|  |  |
| --- | --- |
| Randomized design | Match paired design |
| Element is randomly select into treatment unit or control unit | Treatment unit match to control unit is matched on unit basis. Using a weighting of identified control variables |
|  | Is used when a volumn of observations is low  Concern for bias is high  Cost per observation is high |
| Do not set up before assign | Must be set up ahead of tine  LL UNITA MUAT B KNOWN BEFORE THE START OF THE Experiment |

**Setting Randomized design**

It is a case when treatment and control units are selected completely randomly and sufficient high in volume.

Control variables ensure both the treatment and control group are representative of the population. For example:

* Type of people who visiting the page
* It would be important ensure that we only count each person once regardless of how many time they click on the site
* Whether or not they already have a membership card

**There are 4 consideration:**

1. Unit of diversion, population, duration and size

The unit of diversion indicates how you will assign units to the control and treatment groups. You could randomly assign each visit to the site to eitjer control or treatment but the issue that the same user seeing both version of the site

So the unit of diversion is better with user so the same user can come back in same group, especially we will use IP address

1. Population for example all people come to the site and do not have app
2. Duration and size: Ensure control and treatment groups are representative of you population.  
   Duration can varies from one week to month while sample size may difference between control and treatment. We only make sure the size is large enough and each group can represent to population

Remember: A/B test here is test whether or not the mean values of the treatment and control groups are the same. With t-test, we try the calculate the p-value: the likehood the actual difference between the mean is 0. General, if the p-value is below 0,05 , indicate a confidence interval of 95%. T-test is use iwht small data sets

Setting for match paired testing

Select control units: we can use the following variables as control variables to match on:

Category Sales

Number of Products

State

Know that we **can set up more than one control units for each treatment unit**

Our goal is to find the balance between more matches and average between matches

How we select pair in A/B test with pair t test

In an A/B test, especially when using propensity score matching, we aim to compare the outcomes between two groups (treatment and control) while ensuring that these groups are balanced on certain covariates. Once we have identified matched pairs (where each treated subject is matched with a control subject based on similar propensity scores), we can use a paired t-test to compare the outcomes of interest.

### Steps to Perform Pair Selection and Paired t-test

1. **Prepare the Data and Estimate Propensity Scores**:
   * Using a logistic regression or generalized linear model to estimate the propensity scores.
2. **Perform Matching**:
   * Using nearest neighbor matching to create pairs.
3. **Conduct the Paired t-test**:
   * Comparing the outcomes between the treated and matched control subjects.

### Detailed Implementation

Here is a step-by-step Python example, including data preparation, propensity score estimation, matching, and performing a paired t-test.

#### **Step 1: Prepare Data and Estimate Propensity Scores**

import pandas as pd

import numpy as np

import statsmodels.api as sm

from sklearn.neighbors import NearestNeighbors

from scipy.stats import ttest\_rel

# Create a hypothetical dataset

np.random.seed(42)

data = pd.DataFrame({

'insurance\_program': np.random.binomial(1, 0.5, 1000),

'age': np.random.normal(50, 10, 1000),

'income': np.random.normal(50000, 10000, 1000),

'prev\_claims': np.random.poisson(2, 1000),

'num\_claims': np.random.poisson(3, 1000)

})

# Define covariates

covariates = ['age', 'income', 'prev\_claims']

X = data[covariates]

y = data['insurance\_program']

# Add a constant to the covariates matrix

X = sm.add\_constant(X)

# Fit the GLM model to estimate propensity scores

glm\_model = sm.GLM(y, X, family=sm.families.Binomial())

glm\_results = glm\_model.fit()

# Get the predicted propensity scores

data['propensity\_score'] = glm\_results.predict(X)

#### **Step 2: Perform Matching**

# Separate treated and untreated groups

treated = data[data['insurance\_program'] == 1]

untreated = data[data['insurance\_program'] == 0]

# Perform nearest neighbor matching

nn = NearestNeighbors(n\_neighbors=1)

nn.fit(untreated[['propensity\_score']])

distances, indices = nn.kneighbors(treated[['propensity\_score']])

# Get the matched untreated indices

matched\_indices = indices.flatten()

# Create matched DataFrames

matched\_treated = treated.reset\_index(drop=True)

matched\_untreated = untreated.iloc[matched\_indices].reset\_index(drop=True)

#### **Step 3: Conduct the Paired t-test**

# Combine into a DataFrame to see the matched pairs

matched\_pairs = pd.DataFrame({

'treated\_index': treated.index,

'control\_index': untreated.index[matched\_indices],

'treated\_num\_claims': matched\_treated['num\_claims'],

'control\_num\_claims': matched\_untreated['num\_claims']

})

# Perform paired t-test on the outcome variable (num\_claims)

t\_stat, p\_value = ttest\_rel(matched\_pairs['treated\_num\_claims'], matched\_pairs['control\_num\_claims'])

print(f"Paired t-test result: t-statistic = {t\_stat}, p-value = {p\_value}")

### Explanation

1. **Propensity Score Estimation**:
   * We use a generalized linear model (GLM) with a binomial family to estimate propensity scores based on covariates (**age**, **income**, **prev\_claims**).
2. **Matching Process**:
   * We use nearest neighbor matching to find the closest untreated (control) subject for each treated subject based on their propensity scores.
3. **Creating the Matched Pairs DataFrame**:
   * The **matched\_pairs** DataFrame includes the indices and outcome (**num\_claims**) of the treated subjects and their matched controls.
4. **Paired t-test**:
   * The paired t-test compares the means of the **num\_claims** variable between the treated and control groups within each matched pair.
   * This statistical test accounts for the pairing, which helps control for confounding variables that were used in the matching process.

### Interpretation of Results

* **t-statistic**: Measures the size of the difference relative to the variation in the sample data. A higher absolute value indicates a more significant difference.
* **p-value**: Indicates the probability of obtaining test results at least as extreme as the results actually observed, under the assumption that the null hypothesis is correct. A lower p-value suggests stronger evidence against the null hypothesis.

By matching treated and control subjects based on propensity scores and using a paired t-test, we ensure that we are comparing similar subjects, thereby reducing bias and increasing the validity of our A/B test results.