Handout 1: Basic A/B Testing

Machine Learning for Smarter Innovation

1 Handout 1: Basic A/B Testing

1.1 Week 10: A/B Testing & Iterative Improvement

Skill Level: Basic | No Code Required | For Non-Technical Stakeholders

---

1.2 What is A/B Testing?

A/B testing is a scientific experiment method where you show two versions of something (A and B) to different groups of users, then measure which version performs better.

Real-World Example: - Version A: Blue "Buy Now" button - Version B: Green "Buy Now" button - Question: Which color gets more purchases?

Instead of guessing, you test both versions at the same time with real users and let data decide.

1.3 Why A/B Testing Matters

1.3.1 Companies That Test Win

Spotify: - Runs 1,000+ experiments per year - Tests every feature before full launch - Iterates 30% faster than competitors - Result: Industry-leading user retention

Amazon: - 35% of revenue from recommendations - Jeff Bezos: "Our success is a function of experiments" - Tests button colors, layouts, algorithms - Result: Highest e-commerce conversion rates

1.3.2 Companies That Don't Test Lose

Knight Capital (2012): - Deployed untested trading algorithm - Lost $440 million in 45 minutes - Company bankrupt - Lesson: Test before you deploy

Quibi (2020): - Launched $1.75B streaming service without testing - Assumed horizontal-only videos would work - Users wanted vertical videos on phones - Result: Shut down after 6 months

---

1.4 The Five-Step A/B Testing Process

1.4.1 Step 1: Observe

What to look for: - Metric declining (conversion rate dropping) - User complaints increasing - Competitor launching new feature - Hypothesis from user research

Example: "Our checkout conversion rate dropped from 8% to 6% last month. We need to fix this."

1.4.2 Step 2: Hypothesize

Format: "If [change], then [metric] will [improve] because [reason]."

Good Hypotheses: - "If we reduce form fields from 10 to 5, then checkout conversion will increase because users abandon long forms." - "If we add product reviews, then purchase rate will increase because users trust peer opinions."

Bad Hypotheses: - "Let's try a red button." (No reason) - "We should improve the site." (Too vague)

1.4.3 Step 3: Design

Key Questions:

A. What are you testing? - Control: Current 10-field checkout form - Treatment: New 5-field checkout form

B. What are you measuring? - Primary metric: Checkout completion rate - Secondary metric: Time to complete checkout - Guardrail metric: Error rate (must not increase)

C. How many users do you need? - Depends on current rate and desired improvement - Typical: 1,000 to 10,000 users per group - Calculator tools available online

D. How long will you run it? - Minimum: 1 week (capture weekly patterns) - Typical: 2-4 weeks - Never stop early just because you see a winner

1.4.4 Step 4: Implement

Technical Requirements: - Randomly assign 50% users to Control, 50% to Treatment - Track user experience consistently - Monitor for technical errors - Ensure users always see same version

Checklist: - [ ] Random assignment working correctly - [ ] Tracking implemented for all metrics - [ ] No technical errors in new version - [ ] Sample size calculator says we have enough users - [ ] Guardrail alerts configured

1.4.5 Step 5: Analyze

Look at three things:

A. Statistical Significance - Question: "Is the difference real or just random luck?" - Tool: P-value (should be less than 0.05) - Interpretation: 95% confident the difference is real

B. Practical Significance - Question: "Is the improvement big enough to matter?" - Example: 6.1% vs 6.0% conversion might be statistically significant but not worth the effort

C. Guardrails - Question: "Did we break anything else?" - Check: Error rates, page load time, support tickets - Rule: If guardrails fail, don't ship

---

1.5 Common Pitfalls and How to Avoid Them

1.5.1 Pitfall 1: Stopping Too Early

Wrong: "After 3 days, Treatment is winning! Let's ship it!"

Right: "Wait the full 2 weeks. Day-of-week effects could reverse the result."

Why it matters: - Monday behavior differs from Saturday - Early users differ from late users - Initial excitement fades

1.5.2 Pitfall 2: Testing Too Many Things at Once

Wrong: Change button color AND text AND position simultaneously.

Right: Test one change at a time.

Why it matters: If you change 3 things and conversion improves, which change caused it? You don't know.

1.5.3 Pitfall 3: Ignoring Sample Size

Wrong: "We tested on 100 users. Treatment won 52 to 48. Ship it!"

Right: "100 users is too small. We need 5,000 per group for a reliable result."

Why it matters: Small samples produce random noise, not real insights.

1.5.4 Pitfall 4: P-Hacking (Peeking)

Wrong: Check results every hour. Stop when you see p less than 0.05.

Right: Pre-commit to sample size and duration. Check once at the end.

Why it matters: If you check 20 times, you'll eventually see p less than 0.05 by random chance, even if there's no real effect.

1.5.5 Pitfall 5: Forgetting Segments

The Trap: Simpson's Paradox

Overall result: Control wins (8.5% vs 8.0%) - New users: Treatment wins (12% vs 10%) - Power users: Treatment wins (9% vs 8%)

How can Treatment win both segments but lose overall? Imbalanced sample sizes.

Solution: Always analyze by key segments (new/returning, mobile/desktop, geography).

1.6 Decision Framework

1.6.1 When to Ship

- **Strong Win:** p less than 0.01, improvement greater than 10%, guardrails pass
- **Decision:** Ship to 100% immediately
- **Example:** New recommendation algorithm increases revenue 15% with no downsides

1.6.2 When to Iterate

- **Marginal Win:** p less than 0.05, improvement 2-5%, some guardrail degradation
- **Decision:** Redesign to remove downsides, test again
- **Example:** New checkout increases conversion 3% but page load time increases 20%

1.6.3 When to Stop

- **Null Result:** p greater than 0.05, improvement less than 2%
- **Decision:** Stop testing, move to next idea
- **Example:** Button color change shows 0.5% improvement with p equals 0.4

1.6.4 When to Rollback

- **Guardrail Failure:** Primary metric improves but critical guardrail fails
- **Decision:** Immediate rollback, investigate root cause
- **Example:** Conversion increases 10% but error rate jumps from 0.1% to 5%

1.7 Key Metrics Explained

1.7.1 Primary Metrics (What You're Trying to Improve)

Conversion Rate: - Formula: Conversions / Visitors - Example: 500 purchases / 10,000 visitors equals 5% - Use when: You want more people to take action

Average Revenue Per User (ARPU): - Formula: Total revenue / Number of users - Example: $100,000 revenue / 10,000 users equals $10 ARPU - Use when: You want to increase revenue per person

Retention Rate: - Formula: Users returning next week / Total users - Example: 7,000 return / 10,000 original equals 70% - Use when: You want users to come back

1.7.2 Guardrail Metrics (What You Must Not Break)

Error Rate: - Formula: Errors / Total requests - Example: 10 errors / 10,000 requests equals 0.1% - Threshold: Must stay below 1%

Page Load Time: - Measure: 95th percentile latency - Example: 95% of pages load in under 2 seconds - Threshold: Must stay below 3 seconds

Support Ticket Rate: - Formula: Support tickets / Active users - Example: 50 tickets / 10,000 users equals 0.5% - Threshold: Must not increase more than 10%

1.8 Building an Experimentation Culture

1.8.1 Principles for Success

1. Make Experiments Easy - Pre-approved experiment framework - Self-service tools for product managers - No engineering bottlenecks - Goal: Launch experiment in less than 1 day

2. Celebrate Learning, Not Just Wins - Reward well-designed tests, even if they fail - Share null results publicly - Learn from failures - Mantra: "No such thing as a failed experiment"

3. Document Everything - What was tested - Why we tested it - What we learned - What we'll do next

4. Set Velocity Targets - Target: 10+ experiments per team per quarter - Measure: Time from idea to decision - Optimize: Reduce friction in testing process

1.8.2 Warning Signs of Poor Culture

- "Let's just ship it and see what happens" (No measurement)
- "We tested for 2 days and it's winning" (Stopped too early)
- "The CEO wants this feature, no test needed" (HiPPO: Highest Paid Person's Opinion)
- "Last test failed, so we're not testing anymore" (Learning aversion)

1.9 Real-World Success Stories

1.9.1 Netflix: 18% Engagement Increase

Challenge: Users overwhelmed by 5,000+ titles **Hypothesis:** Personalized micro-genres will help discovery **Test:** Control (standard categories) vs Treatment (personalized micro-genres) **Result:** 18% increase in viewing time, 25% increase in discovery **Learning:** Personalization at the category level, not just the item level

1.9.2 Booking.com: 3.5% Revenue Increase

Challenge: Users abandon checkout due to analysis paralysis **Hypothesis:** Showing scarcity signals will create urgency **Test:** Control (no scarcity) vs Treatment ("Only 2 rooms left!") **Result:** 3.5% increase in booking completion **Learning:** Authentic scarcity signals reduce abandonment

1.9.3 Etsy: 12% Search Improvement

Challenge: Search results miss good matches **Hypothesis:** ML ranking will surface better results **Test:** Control (rule-based) vs Treatment (ML-based) **Result:** 12% increase in search-to-purchase rate **Learning:** ML beats hand-crafted rules for complex ranking

---

1.10 Your First A/B Test: Checklist

1.10.1 Before You Start

- ☐ Clear hypothesis with expected direction
- ☐ Primary metric defined and measurable
- ☐ Guardrail metrics identified
- ☐ Sample size calculated (use online calculator)
- ☐ Duration planned (minimum 1 week)
- ☐ Stakeholder alignment on decision criteria

1.10.2 During the Test

- ☐ Random assignment is working (check split is 50/50)
- ☐ Both versions have no technical errors
- ☐ Guardrails are being monitored
- ☐ No peeking at results before planned end date

1.10.3 After the Test

- ☐ Statistical significance calculated (p-value)
- ☐ Practical significance assessed (is the lift big enough?)
- ☐ Guardrails reviewed (did we break anything?)
- ☐ Segment analysis completed (any surprising patterns?)
- ☐ Decision documented (ship/iterate/stop and why)
- ☐ Results shared with team

1.11 FAQ

Q: How long should I run my test? A: Minimum 1 week to capture day-of-week effects. Typical is 2-4 weeks. Use a sample size calculator to determine when you have enough users.

Q: Can I test more than 2 versions? A: Yes. A/B/C or A/B/C/D tests work the same way. Just need more users to maintain statistical power.

Q: What if my test shows no difference? A: That's valuable learning. Document it and move to the next idea. Null results prevent wasted effort scaling something that doesn't work.

Q: Should I test on 10% of users or 50%? A: Start small (10-20%) if you're worried about risk. But 50/50 splits reach statistical significance faster with fewer total users.

Q: What's a good win rate for experiments? A: Industry average is 1 in 7 tests produces a meaningful win (14%). If you're winning more than 50%, you're probably not taking enough risks.

Q: How do I convince my CEO to test instead of just shipping? A: Frame it as risk reduction. "Spending 2 weeks testing could save us from a $1M mistake. Netflix tests everything for this reason."

1.12 Key Takeaways

1. **A/B testing is the scientific method for product development**
   - Hypothesis, experiment, measure, learn
2. **Good experiments have three components**
   - Clear hypothesis with expected direction
   - Sufficient sample size for reliable results
   - Guardrail metrics to catch unintended consequences
3. **Most experiments fail, and that's good**
   - Industry average: 1 in 7 wins
   - Null results prevent scaling bad ideas
   - Celebrate learning, not just wins
4. **Common mistakes are predictable and avoidable**
   - Don't stop early
   - Don't test multiple things at once
   - Don't ignore sample size requirements
   - Don't peek at results repeatedly
5. **Speed wins over perfection**
   - Companies that iterate faster learn faster
   - Spotify: 1,000+ experiments per year
   - Your goal: 10+ experiments per quarter

Next Steps: - Read Handout 2 for Python implementation guide - Use online sample size calculators to plan your first test - Document your hypothesis and get stakeholder alignment - Run your first experiment this week

Remember: The best way to learn A/B testing is to run tests. Start small, document everything, and iterate.