Part B - AdFisher

UPDATED 2018-10-26: Added clarifications and three preliminary ungraded questions with a solution to the first to guide students towards their answer to 8 (a).

UPDATED 2018-10-27: Reworded exercise 8 (a) to emphasize number of different treatments.

For this part of the homework, you will work with AdFisher. AdFisher is a tool to automate browser based experiments and run machine learning based statistical analyses on the collected data. It was described in the October 1 lecture and in the paper Automated Experiments on Ad Privacy Settings: A Tale of Opacity, Choice, and Discrimination. If you have not already done so, read the Methodology Section of that paper (Section 4).

Installation

The instructions here describe an installation of a virtual machine provisioning tool called Vagrant which is then used to install and run AdFisher. Alternatively you can install AdFisher and required components on your computer outside of a virtual machine. This is a lot more involved. See the last section of this homework for information on how to get started with that option.

- Download and install Vagrant and git.
  - Git from https://git-scm.com/downloads.

- Open a terminal.
  - Mac: On Mac computers, this is the Terminal.app utility found in /Applications/Utilities.
  - Windows: On Windows, open the Command Prompt.
  - Linux: Linux has a large variety of terminal emulators that come pre-installed. Consult your distribution’s documentation to determine your default emulator and how to invoke it.

- Clone the AdFisher git repository by running the following command in the terminal (the command is the text after >):

  ```
  > git clone https://github.com/cmu-transparency/tool-adfisher
  ```

- Start a virtual machine and install the other requirements. Navigate to the AdFisher directory in the tool-adfisher repository and run vagrant up.

  ```
  > cd tool-adfisher/AdFisher
  > vagrant up
  ```
This will run for a while but eventually will produce a VirtualBox window in which an instance of Linux is running. You are not done yet. After some more time the `vagrant up` command will complete. Wait for it to return before proceeding to the next step.

- Restart the virtual machine.

  > vagrant reload

  This should close the Linux window and then reopen it starting into a visual login interface. Log in using username `vagrant` and password `vagrant`.

- Disable screen lockout.

  Click on the gear and wrench icon the left side of the screen, then click on “Brightness & Lock”. Under “Turn screen off when inactive for”, select the option “Never”. Close the settings window but **do not** close the VirtualBox window. Browser instances will pop up there when running experiments.

- To run experiments you will need to start some python scripts. You can do this either from within the GUI of the virtual machine you have started in the previous step or alternatively you can connect to it from your local terminal over `ssh`.

  > vagrant ssh

  This will bring up a shell where you can execute commands on the virtual machine. You will first need to configure some display options. In the virtual machine, run the following:

  (on VM) > export DISPLAY=:0

  Note that you will need to run this command every time you connect to the VM (and want to run experiments).

**Testing the installation**

Navigate to `AdFisher/examples` and run `demo_exp.py` and `demo_analysis.py` to check if all dependencies have been correctly installed.

(on VM) > cd /vagrant/examples
(on VM) > python3 demo_exp.py
(on VM) > python3 demo_analysis.py

The first of the above scripts should start a few browser instances in the VirtualBox window and take about a minute or two to complete while the second should produce some statistics immediately. Assuming both of these succeed, you can proceed to the homework exercises below. Otherwise seek help of course staff (please indicate which of the above steps failed and how).

As part of this homework you will write some scripts making use of AdFisher. To edit scripts, you can edit the contents of the `tool-adfisher/AdFisher` directory on your host computer which should be simultaneously visible in the virtual machine under `/vagrant`. You can edit your scripts within or without the virtual machine but need execute them on the VM.

In the `examples` folder, you will find the script `test.substance.py`. This is a toned down version of the original experiment run as part of the paper study. This script no longer works due to the fact that Google no longer uses plain text ads, but you can still use this file for reference when writing your own code.
Warning: AdFisher is a tool originally developed several years ago and a lot of its functionality, especially that which depends on the structure of the web pages it crawls, will not operate correctly. Though we provide the full source code of the tool and the various experiments performed with it, most of those will likely fail due to being outdated and changes in the supporting libraries. We encourage you to explore the tool but you should not expect elements outside of the scope of this homework to be fully operational.

Updating AdFisher

The structures of the web pages that will be experimented with in this exercise change often. It might be necessarily for you to upload your copy of AdFisher. To do so you will need to commit your changes, if any, to a local repository and pull updates from us using the following git commands executed in the tool-adfisher directory:

```
> git stage -u
> git commit -m "a message describing your changes"
> git pull
```

Methodology

Familiarize yourself with Section 4 of Automated Experiments on Ad Privacy Settings: A Tale of Opacity, Choice, and Discrimination. The following concepts are relevant to this homework:

- experimental unit
- treatment, experimental treatment, control treatment
- hypothesis, null hypothesis
- significance testing
- permutation test
- p-value

3 Exercises

Alice is worried that her online activities are being used by Google target content recommendations to her. She uses her browser to visit websites and perform Google searches. She is worried some or all of these activities are affecting online news articles that are served to her. She hires you, a privacy technologist, to address her concerns. Your job is to design and run an experiment using AdFisher to either prove her right (or wrong). You will need to verify (or fail to verify) these two hypotheses:

- **Hypothesis 1**: Visits to websites related to TOPIC has an effect on news articles presented on Google News.
- **Hypothesis 2**: Google search queries related TOPIC have an effect on news articles presented on Google News.

As part of the hypothesis, you will also select a topic. Some examples are listed below but it is best if you pick a unique topic that you suspect might have impact on Google News stories. It is best if the students in the class pick a variety of topics so do try to not copy your classmates. Example topics:

- Substance abuse
- Loans, lotteries
- Democratic or republican politics (stick to one of the two)
- Stereotypical male interests or stereotypical female interests (stick to one of the two)
Exercise 6. Describe your topic.

(a) Find 3-5 websites closely relevant to your topic (for Hypothesis 1)
(b) Design 3-5 search queries closely related to your topic (for Hypothesis 2).

Replace the the contents of examples/sites.txt and examples/queries.txt with your topic-relevant items.

Exercise 7. Describe the experiments you can run to verify the hypotheses.

(a) What are the treatments in your experimental design? Be sure to mention experimental and control treatments.

(b) What measurements will you make?

(c) Let \( \vec{t}, \vec{y} \) be the list of the treatments assigned to all of your units and their measurements, respectively. What is your test statistic in terms of the treatments and measurement lists?

Exercise 8. The permutation test can be used to evaluate the significance of your test statistic. Let \( S(\vec{t}, \vec{y}) \) denote your test statistic. For a permutation \( \pi \), consider the value of \( S(\vec{t}, \pi(\vec{y})) \). That is, the value of your statistic in which the measurements are randomly shuffled. The p-value of the statistic \( S \) is the proportion of permutations \( \pi \) that produce statistic value at least as large as the observed, or \( S(\vec{t}, \pi(\vec{y})) \geq S(\vec{t}, \vec{y}) \). The proportion here is relative to all possible permutations \( \pi \).

Formally, for an experiment with \( m \) units, the p-value according to the permutation test is equal to

\[
\frac{1}{|S_m|} \sum_{\pi \in S_m} I[S(\vec{t}, \pi(\vec{y})) \geq S(\vec{t}, \vec{y})]
\]  

(1)

where \( S_m \) is the set of permutations of \( m \) elements, and \( I \) is the indicator operation (equal to 1 when argument is true and otherwise 0). Note that there are \( m! \) permutations of \( m \) elements where ! denotes the factorial operation.
1. What is the minimum p-value for a permutation test for an experiment with an arbitrary test statistic over \( m \) units and an arbitrary number of treatments?

**Answer:** There are \( m! \) permutations of \( m \) units and one of them does not change the order of its inputs. That is, \( \pi(y) = y \) so \( S(\bar{t}, \pi(y)) = S(\bar{t}, y) \) for this special identity permutation \( \pi \). Therefore, for any arbitrary test statistic over \( m \) units, at least \( \frac{1}{m!} \) fraction of the permutations will result in the statistic at least as large as the original.

AdFisher uses accuracy of a predictor of treatment from measurement as its test statistic. Let \( g \) be a model that guesses the treatment \( t \) given its measurement \( y \). The test statistic \( S \) is defined as follows.

\[
S(\bar{t}, \bar{y}) \overset{\text{def}}{=} \frac{1}{m} \sum_i I[g(y_i) = t_i]
\]  

2. (Ungraded Exercise) What is the minimum p-value for a permutation test for an experiment with two treatments with \( n \) units each? Note that in this case the elements of the treatments vector \( \bar{t} \) take on only one of two possible values.

AdFisher operates over blocks to minimize the effects of time as a confounding factor in its experiments. Let there be \( b \) blocks and \( 2n \) units in each block (\( n \) experimental treatments and \( n \) control treatments). The permutation test with blocks permutes the units within each block but not across blocks.

\[
\frac{1}{|S_{m/b}|} \sum_{\pi \in S_{m/b}} I \left[ S(\bar{t}, \pi(y)) \geq S(\bar{t}, \bar{y}) \right] 
\]  

Here \( S_{m/b} \) is the set of permutations of \( mb \) elements that respects block boundaries. This can be defined as follows:

\[
S_{m/b} \overset{\text{def}}{=} \{ \pi \in S_{mb} : \pi(\langle 1, \cdots, m \rangle) = \langle \pi_1(\langle 1, \cdots, m \rangle), \pi_2(\langle m+1, \cdots, 2m \rangle), \cdots, \pi_b(\langle m(b-1)+1, \cdots, mb \rangle) \rangle, \\
\pi_i \in S_m \} 
\]  

That is, a permutation that respects the block boundaries is a combination of \( b \) permutations of \( m \) elements, one for each block.

3. (Ungraded Exercise) How many permutations are there over \( b \) blocks of \( m \) units each. That is, what is the size of the set \( S_{m/b} \)?

(a) What is the minimum p-value according to the permutation test for an experiment with \( b \) blocks with exactly two treatments of \( n \) units each? Write your answer as a formula in terms of \( b \) and \( n \).
(b) In AdFisher, the default split is done so that first 80% of the blocks are used as training blocks and the last 20% are used as testing blocks.

As a result, if you use perform your experiments with 10 blocks, the test statistic would only apply to 2 of them. What is the number of agents (2n) and blocks (b) you are using in your experiment?

(c) For this selection, using the formula derived above, compute the minimum p-value you can obtain? For a significance level of 0.05 (i.e. you reject the null hypothesis only when the p-value is less than 0.05), is your choice of n and b good enough?

Exercise 9. Implement an AdFisher study. A template of a script is provided for you in AdFisher/examples/hwk3.py. The script is meant to be invoked in 4 different ways, for measurement and analysis, and for one of the two hypotheses:

```
> python3 hwk3.py measure 1
> python3 hwk3.py analyze 1
> python3 hwk3.py measure 2
> python3 hwk3.py analyze 2
```

(a) Fill in the missing details of hwk3.py script in the appropriate sections to develop the script to run your experiment. At least the places marked “TODO” need your attention.

Keep the invocation options “measure” and “analyze” and hypothesis number as part of your script and do not change the name of the log files produced as they will be part of your submission.

You may have to look at the other examples of measurements and analyses in the examples directory.

You should make use of these three methods:

- `BrowserUnit.visit_sites(site_file, delay)` (core/web/browser_unit.py)
  - `site_file` is the path to a file which contains a list of websites in each new line. This method drives the browser to visit all these websites.
  - Example use: `visit_sites(site_file="sites.txt", delay=5)`. 

• GoogleSearchUnit.search_and_click(query_file, clickdelay, clickcount) (core/web/google_search.py)

query_file is the path to a file which contains a list of queries in each new line. This method drives the browser to search for a query term, click on clickcount search results, with clickdelay seconds of delay between subsequent clicks.

Example use: search_and_click(query_file="queries.txt", clickdelay=20, clickcount=5).

• GoogleNewsUnit.get_news(type, reloads, delay)

(found in core/web/google_news.py)

This method collects news articles from Google News by reloading the page reloads times, with delay seconds between successive reloads. type should be ”top” to collect only the Top News (other methods are not supported as of now).

Example use: get_news(type="top", reloads=5, delay=10).

Note that GoogleNewsUnit is a subclass of GoogleSearchUnit which is a subclass of BrowserUnit. This means that GoogleNewsUnit units have access to all three of the above treatment/measurement methods.

Exercise 10. Run your experiments and report the test statistic (test-accuracy) and the p-value. Is the p-value significant?
Submission

You have to submit the following files:

• The main script `hwk3.py`.
• The sites `sites.txt` and queries `queries.txt` files you used to define your selected topic.
• The logs generated `hypothesis1.log` and `hypothesis2.log` as part of the experiments.

Put all the files into a zip file with the name `<your_andrew_id>_hwk3.zip` and submit it through Canvas. DO NOT submit the entire AdFisher folder.

Alternative Local Installation

You can also install AdFisher on your local computer without relying on a virtual machine. To do this follow these instructions. Note that this is a lot more involved than using the virtual machine and requires knowledge of your operating system.

• Clone AdFisher as in the previous section:

  > git clone https://github.com/cmu-transparency/tool-adfisher

• Consult the instructions in `tool-adfisher/AdFisher/README.md` along with the installation script `bootstrap.sh` used for the Vagrant-based installation. This script is is compatible with Linux distributions that use `apt` to manage packages. If yours is not one of those, you will need to adapt the lines making use of `apt-get` for your system’s package manager.