Price Discrimination Simulator

Xiaoqian Ma (xm2146)
Zhicheng Wan (zw2275)
Zhou Zhou (zz2181)

Project Description

A simulation engine for pricing discrimination

Related Work

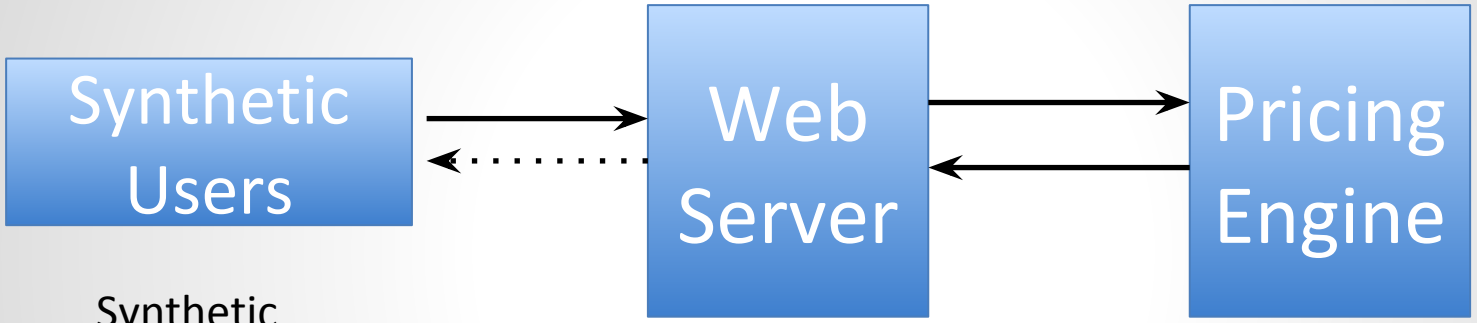
Mikians et al. [2]

- detection system as a watchdog

Hannak et al. [3]

- probed the existence of pricing steering

Design



Synthetic attributes:
Location
Race
Sex
Income

Web behavior:
Surfing history
Purchase history

Product attributes:
Inventory status
Competition
Cost

Synthetic Users

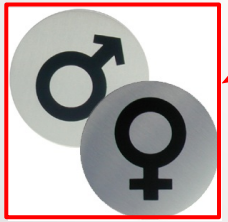
40000 zip code

40000 queries to Census.gov

Retrieved each record with 16 attributes

Real probability density function used

Synthetic Users



[1] = 'White, Not Hispanic or Latino'

[2] = 'Hispanic or Latino'

[3] = 'Black or African American'

[4] = 'American Indian and Alaska Native'

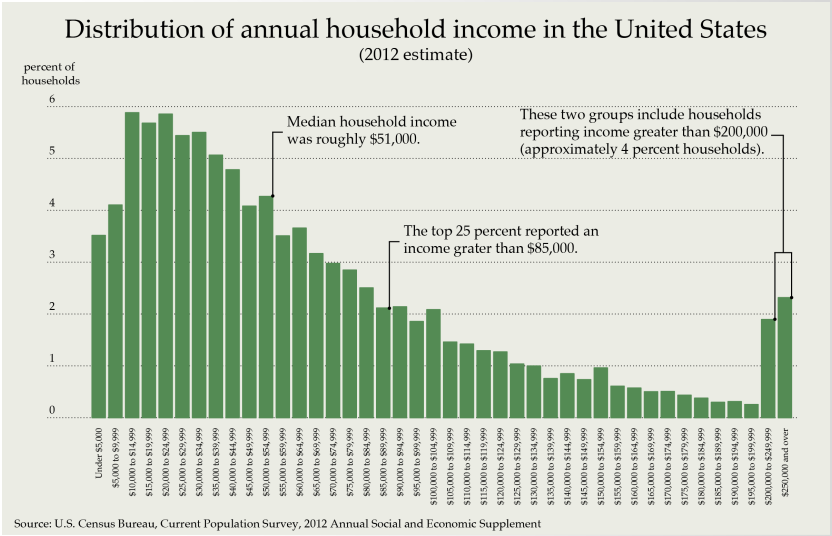
[5] = 'Asian'

[6] = 'Native Hawaiian and Other Pacific Islander'

[7] = 'Some Other Race'

[8] = 'Two or More Races'

Synthetic Users



A vector of 4 random seeds ----> A synthetic user

Tools: Flask, Heroku, MySQL, JSON

Middle Layer

Application Server

- Generate user: get new data from user pool
- Simulate: get the prices from pricing engine

Control Panel

- Adjust pricing coefficient

Development tools: django, sqlsite

Pricing Discrimination Simulator

Cost: $150 CompetitorPrice: $175

User 29

income:236106.0 sex:0 race:1 zipcode:89423 hasVisited:1
isVisitedPriceHighOrLow:1 hasPurchased:0
isPurchasedPriceHighOrLow:1 stock:41

Price: $156.0

User 30

income:8674.0 sex:1 race:1 zipcode:17603 hasVisited:0
isVisitedPriceHighOrLow:0 hasPurchased:1
isPurchasedPriceHighOrLow:1 stock:200

Price: $274.0

User 31

income:4935.0 sex:1 race:1 zipcode:95838 hasVisited:1
isVisitedPriceHighOrLow:1 hasPurchased:1
isPurchasedPriceHighOrLow:1 stock:197

Price: $285.0

User 32

income:23308.0 sex:1 race:2 zipcode:29651 hasVisited:0
isVisitedPriceHighOrLow:0 hasPurchased:0
isPurchasedPriceHighOrLow:1 stock:33

Price: $208.0

Pricing Coefficients Adjustment

Household Income(%): 10

Race(%): 52

Sex(%): 28

Zipcode(%): 81

Has Visited(%): 45

Is Visited High or Low(%): 68

Has Purchased(%): 34

Is Purchased High or Low(%): 62

Cost(%): 37

Competitor Price(%): 10

Stock(%): 79

save

Middle Layer(to be done)

Application Server

- Custom input user data
- Access control: administrator

Control Panel

- Choose industry category and price model
- Correlation analysis

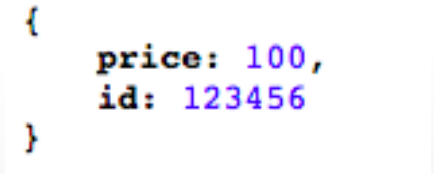
Pricing Engine - Communication

Two Set of APIs:

- System Configuration
- Specific Product Pricing Request

JSON Response:

- Client ID
- Price



Pricing Engine - Parameters

Domestic

Location

(Zipcode)

Shopping Behavior

Browsing History
(hasHistory, ProductPricingRange)

Purchase History
(hasHistory, ProductPricingRange)

Business

Cost

Inventory Level

Competition

(Competitor's Price)

Pricing Engine - Algorithm

Price =
LocationCoefficient(ZipCode)
*BrowsingHistoryCoefficient(hasVisited,isVisitedPriceHighOrLow)
*PurchaseHistoryCoefficient(hasPurchased,isPurchasedPriceHighOrLow)
*CompetitiveCoefficient(competitorPrice, stock)
*Cost

Pricing Engine - Algorithm

LocationCoefficient(ZipCode)

For each of the location, collect real online stores pricing statistics and use machine learning approach to calculate coefficient for every valid zip code

Online Store Explored

- Currently available: Best Buy,Amazon, Ebay
- Not available: Home Depot, Staples
- Possible candidate: Hotels.com, Expedia, Cheaptickets, Orbitz

Pricing Engine - Algorithm

BrowsingHistoryCoefficient(hasVisited,isVisitedPriceHighOrLow)
PurchaseHistoryCoefficient(hasPurchased,isPurchasedPriceHighOrLow)

Users are classified into High/Low categories according to the price of the products they have browsed/purchased

Price is inflated with a small markup for high valued customers

Initial pricing for first time visitor/buyer is also possible

Pricing Engine - Algorithm

CompetitiveCoefficient (competitorPrice, stock)

Competitive Algorithms for Online Pricing[1]

Adapted from IEOR research

Taking inventory level, highest price a customer is willing to pay(capped by the competitor's price)

Adjust Price with the listed Algorithm

1. Zhang, Yong, Yuxin Wang, Francis Y. L. Chin, and Hing-Fung Ting. "Competitive Algorithms For Online Pricing." Discrete Mathematics, Algorithms and Applications 04.02 (2012): 1250015. Web.

Algorithm 1. Pricing

Let y_j be the largest amount that user i is willing to buy given price 2^j and satisfying $y_j \leq m$.
Let $k = \arg\max_j y_j \cdot 2^j$
if $x_k \neq 0$ **then**
    Set unit price to be $p_i = 2^k$.
    Assign $m_i = \min\{x_k, y_k\}$ items to user i.
    Modify Available Amount of Items.
else $x_k = 0$
    Let $k' = \arg\max_{j>k} y_j \cdot 2^j$ such that $x_{k'} > 0$
    Set the unit price to be $p_i = 2^{k'}$.
    Assign $m_i = \min\{x_{k'}, y_{k'}\}$ items to user i.
    Modify Available Amount of Items.
end if

Algorithm 2. Modify Available Amount of Items: ($p_i = 2^k$ and $m_i = \min\{x_k, y_k\}$)

if $m_i = x_k$ **then**
    $x_j = 0$ for $0 \leq j \leq k$
else $m_i = y_k$
    Let $\ell = \arg\max_j x_k - x_j \geq y_k$.
    for $j = \ell + 1$ to k **do**
        $x_j = x_k - y_k$
    end for
end if

What’s next?

April 6th: Progress report

April 13th: Application server analysis module refined

April 20th: Pricing engine based on theoretical models

April 24th: Pricing engine based on machine learning, data should be gathered

April 27th: Pricing engine based on machine learning, models should be trained

Before Final: Synthetic user data, application server to be further refined; UI refined.

Thank you!