# Pricing Optimization Strategy for PunchhMart

[**Pricing Optimization Strategy for PunchhMart**](#_1yoxjzdggr8l) **1**

[Problem Statement](#_ddojk0nudcnn) 2

[Exploratory Data Analysis](#_fr5kgozdk7ob) 2

[Price Segments](#_sc421e5x4tf) 2

[Profit Segments](#_uknrr1f75v3o) 3

[Incremental CVR Segments](#_32iq62ts82cg) 4

[Incremental Sales Segments](#_76ksrwpwdua3) 5

[POC for Profit Optimization](#_u6tpalvaahry) 5

[(A) Solve a standard Price Optimization problem](#_ctihmc9jr6wi) 5

[Process Flow:](#_qsx2yfi2ojn8) 6

[Demo:](#_cqlaybxbbwod) 6

[(B) Find the best Profit considering different combinations of price variations](#_id571xnyg5c2) 7

[Process Flow:](#_435tzoql713w) 7

[Profit calculation logic:](#_tgvo7rioo0mv) 7

[Assumptions:](#_i2hn416v2blm) 7

[Price Change calculation logic:](#_iti4npr8vpv0) 7

[Assumptions:](#_5tibt7ynwf5c) 8

[Main Workflow:](#_q3qcj8ee8xep) 8

[Assumptions:](#_kyh17q219cas) 8

[Demo](#_rorywnxaio54) 8

[Web App](#_le4eoagve5zz) 9

[(C) Create the demand function an item and find the maximum profit from predicted values](#_cii9sfi4ipj1) 9

[Process Flow](#_2xh7rpa3mfrx) 10

[Limitations:](#_2qnimrpgd7rk) 12

[Future Improvements](#_2tjj4yx5qich) 12

[Improve the custom price variation logic](#_x4fshno27m33) 12

[Improve Demand Forecast using Price Elasticity](#_dub9r6pviypw) 13

[Improve Demand Generation by considering additional signals like is\_promotion, is\_weekday , is\_holiday, weekly change in demand, elasticity, log-price ratio](#_2eb96vuvznau) 14

[Long-term vision - leverage relevant Marketing Signals and adopt Experimentation Strategy](#_1vlryifbhw1) 15

## Problem Statement

We need to optimize the pricing of 250 product items (see product\_sales.csv ) and maximize total profits for online grocery store, PunchhMart . From our own analysis, we have found that out of 100 people who visit your website, 5 would

end up making a transaction. From those who made a purchase, we have also obtained the general buying patterns, e.g. their avg units purchased of item\_id:1 are 0.6 (see the screenshot below).

To understand the pricing impact, we have tested different price points for each product. The impact can be broken down into two aspects:

● A lower price increases the volume of the purchased products. For instance, Item 1

might be a frequently used product. If you decrease its price, the customer’s

conversion rate would increase. The “incr\_cvr” column from the CSV file quantifies this

price elasticity effect. More detailed definitions below together with the screenshot.

● A decrease in the unit price, however, would also reduce your profit margin on the

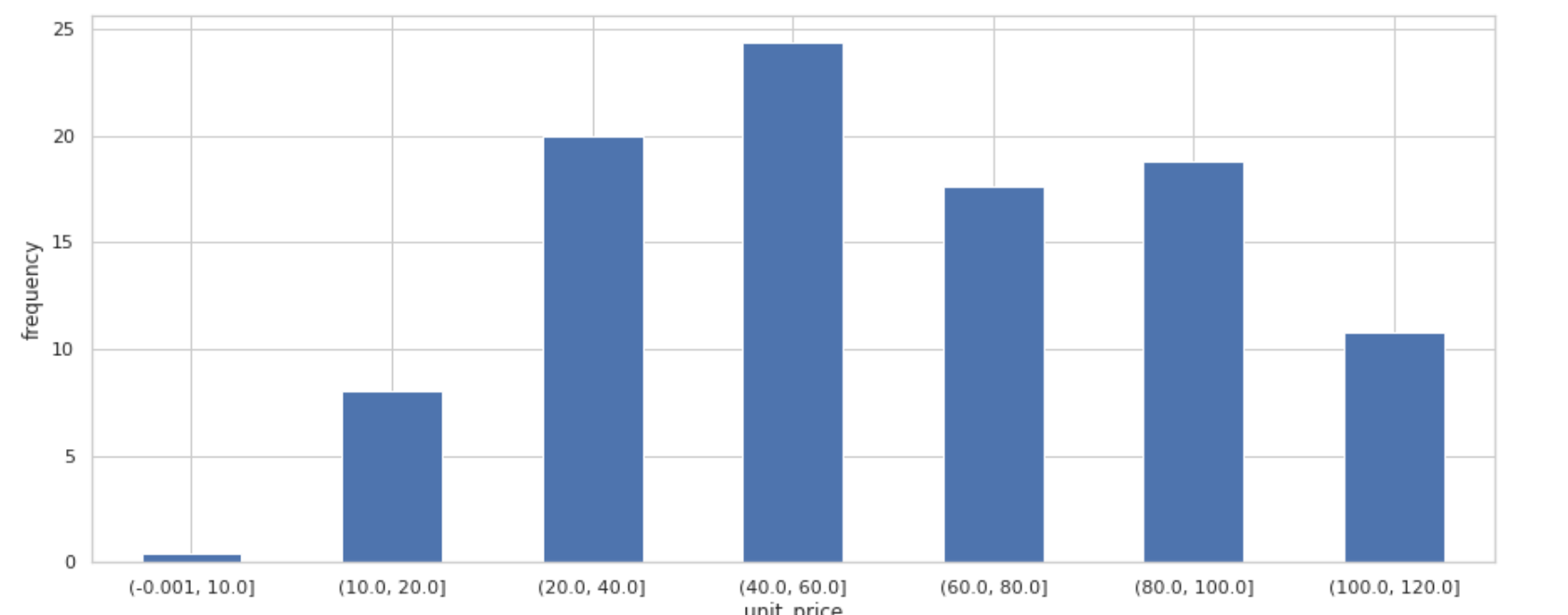
product.

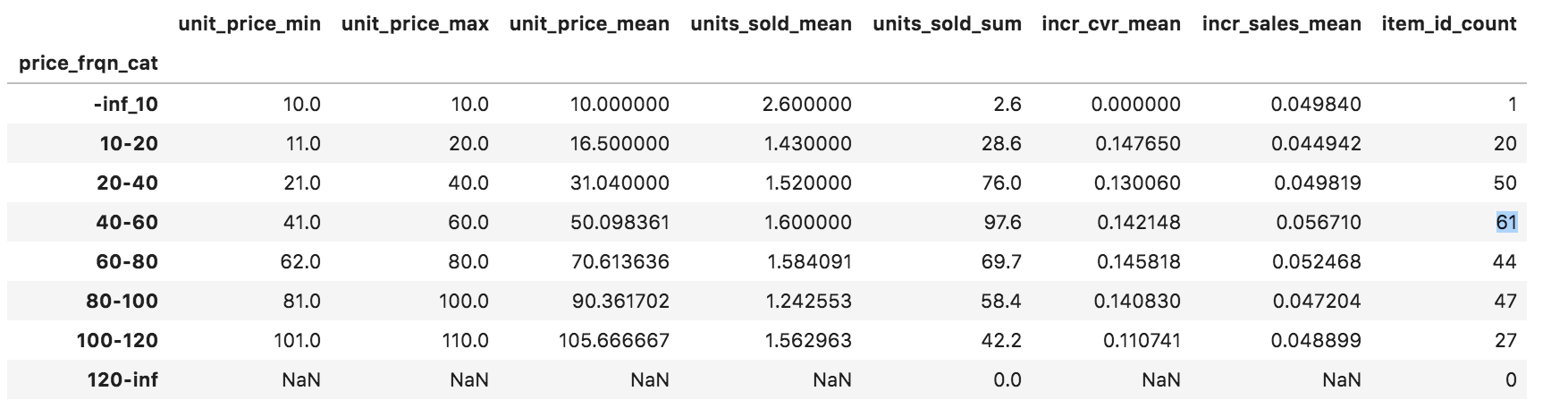
Therefore, our goal is to find the optimal price points for each item in order to maximize the total profits.

## Exploratory Data Analysis

Let’s analyze different segments (refer to **ItemSalesPriceEDA.ipynb**)

### Price Segments

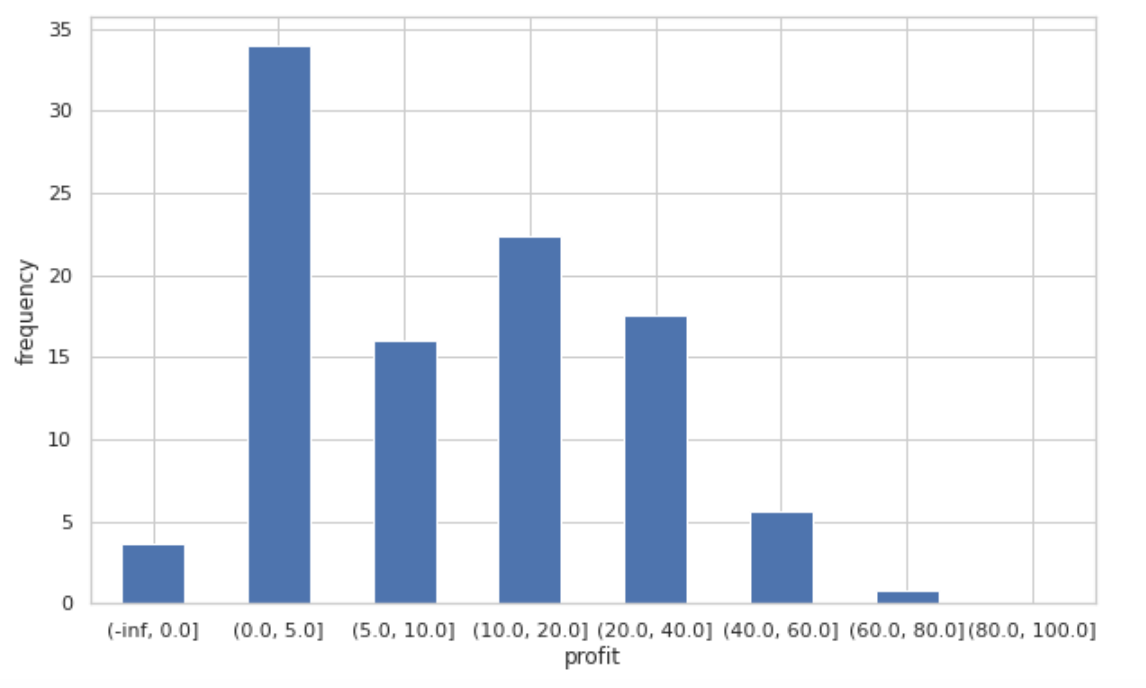


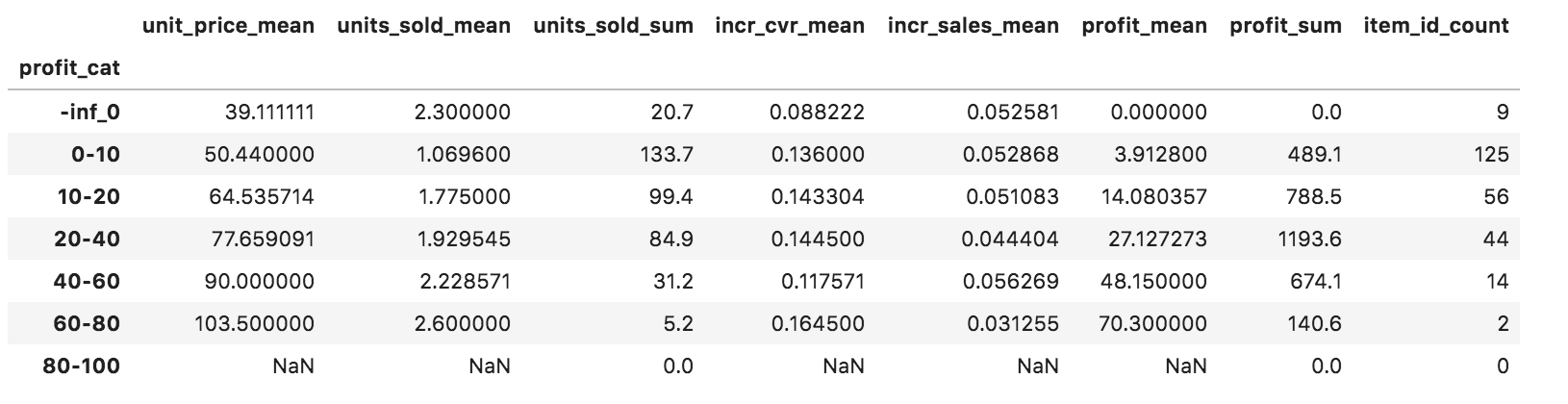


price range (40 - 60)

* contains highest# of items (61)
* highest# of avg units sold (1.6)
* pretty good avg price-elasticity ( 14.2 % incr\_cvr, 5.2 % incr\_sales)

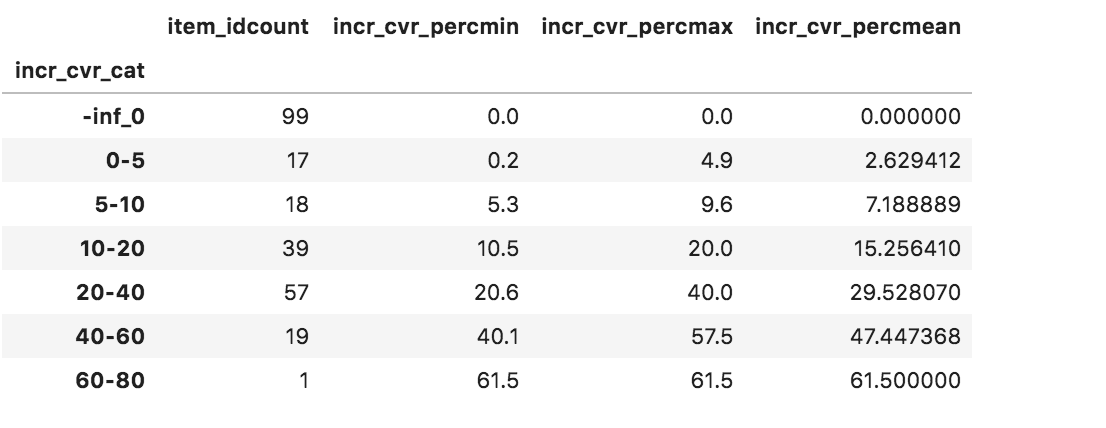
### Profit Segments





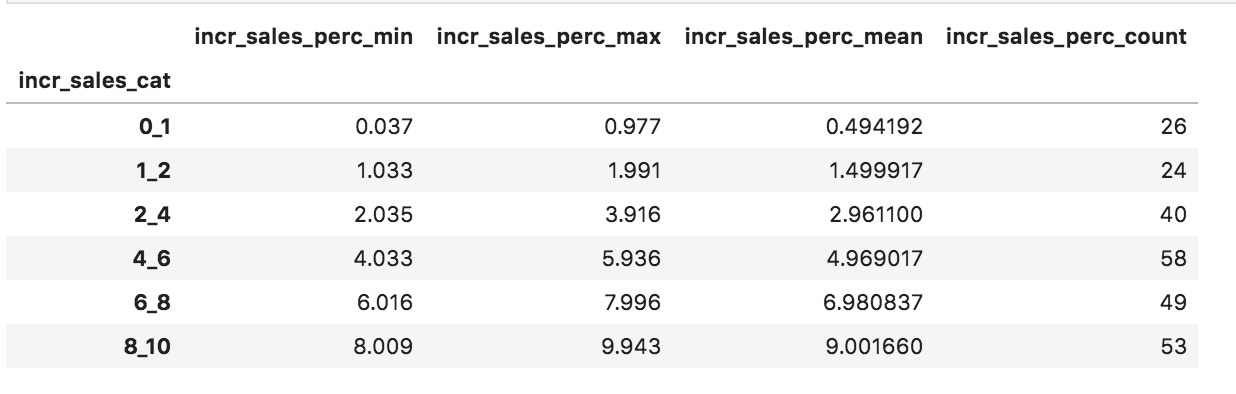
* 9 items don’t generate any profit with lowest incr cvr mean value (0.09)
* 125 items have very low avg profit margin ($3.9) and low average units sold (1.07)
* Further analysis needed to understand
  + the root cause of the problem
    - Is there higher procurement / manufacturing cost for these items
    - Is there higher propensity to churn for customers for these items
    - Do these items sell well only in a specific season ?
    - Were there any promotions for these items ? Check the Item age
    - Since these are online grocery items, what are the search demands for these items i.e. are these item\_keywords for the brands ranking in Google/Bing ?
    - Are consumers able to find these in ADs or Paid Social posts (in case already being promoted) ? In that case check the bidding strategy and keyword relevancy.
  + and then adopt some strategies to solve the problem
    - Can lowering price yield more profit as this group has moderate incr\_cvr ? Or maybe we can afford to increase the price a bit as the mean price already in the lower bracket , units\_sold is low and avg incr\_sales value looks good.
    - Since these 125 items have significant incr\_cvr and incr\_sales, they can be promoted (if not done already) in paid social and search engines to boost the cvr
  + Definitely additional marketing signals and user engagement metrics are needed to answer the above questions.

### Incremental CVR Segments



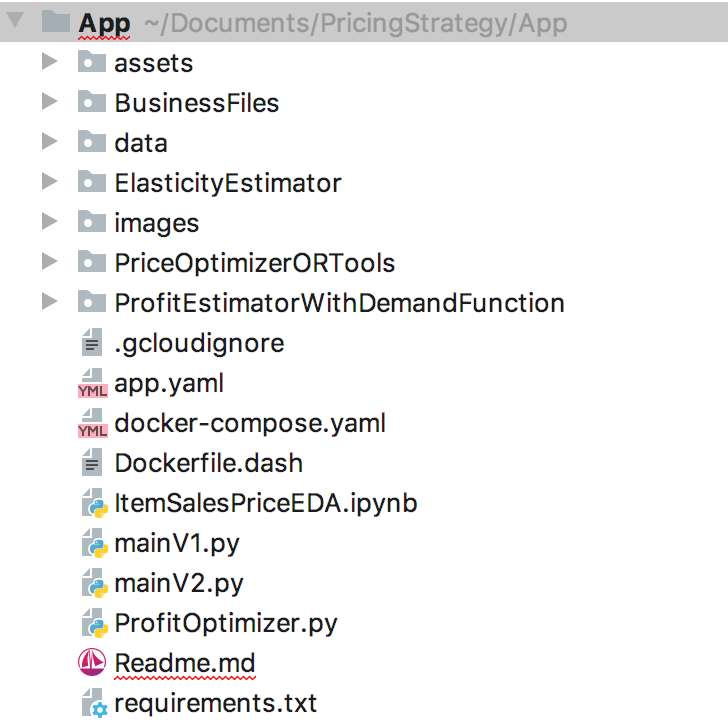
* 99 items do not have any incr\_cvr value either due to the fact that these are cold-start items without enough prior history of customer transactions or may be they are not getting enough customer visibility and can be candidates for Customer Acquisition Campaign
* We can take similar measures as mentioned in the strategies for boosting profit for low-profit items

### Incremental Sales Segments



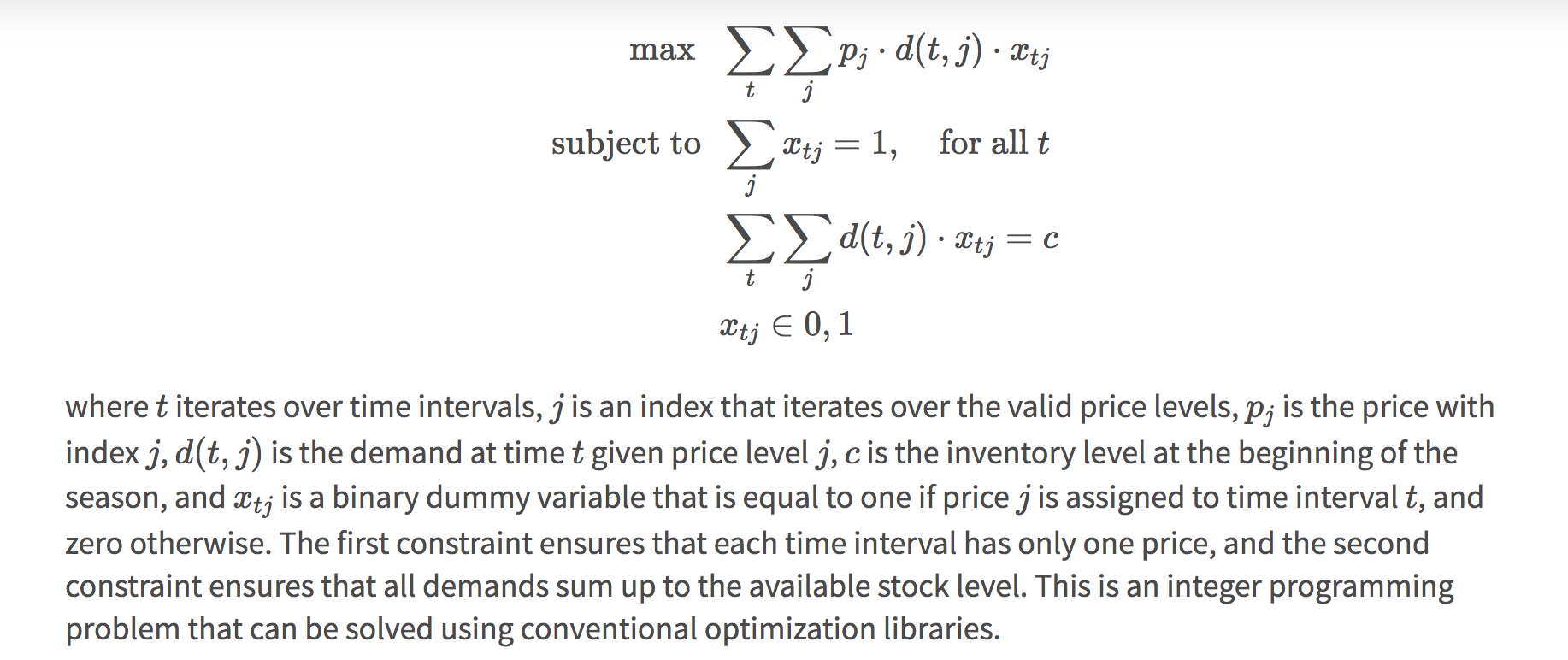
* 26 items have very low incr\_sales

## POC for Profit Optimization



### (A) Solve a standard Price Optimization problem

Expression of the optimization problem:



© <https://blog.griddynamics.com/deep-reinforcement-learning-for-supply-chain-and-price-optimization/>

#### 

#### **Process Flow:**

#Assumptions

# Since the optimizer can either maximize or minimize the price values, use maximization technique

price\_change\_multipler = np.array(data.shape[0] \* [0.9])

price\_lb = [float(i) **for** i **in** initial\_price\*0.9]

price\_ub = [float(i) **for** i **in** initial\_price\*1.2]

# Define the solver

solver = pywraplp.Solver(**'SolveStigler'**, pywraplp.Solver.GLOP\_LINEAR\_PROGRAMMING)

# Set the objective function

**for** i **in** range(0, len(data)):

target\_price[i] = solver.NumVar(price\_lb[i], price\_ub[i], str(data\_arr[i][0]))

objective.SetCoefficient(target\_price[i], float(price\_change\_multipler[i]))

objective.SetMaximization()

# Set the constraints: Sum (price\*quantity) >= Initial Profit ($3285.90)

constraints.append(solver.Constraint(float(np.sum(initial\_profit\_vals)),solver.infinity()))

**for** i **in** range(0, len(data)):

constraints[0].SetCoefficient(target\_price[i], float(adj\_quantity[i]))

#### **Demo:**

Run: **python PriceOptimizer.py**

Initial total price: $15155.00

Optimal total price: $18186.00

Initial total profit: $3285.90

Optimal total profit: $10929.95

**Limitations:**

Not sure if custom rules can be specified in OR Tools library

e.g.

* Decrease price for high-profit , high-volume , high incr\_cvr items
* Increase price for a low-profit , low volume items
* Increase price for very low incr\_cvr value etc.

We can apply custom constraint rules using scikit optimizer which offers some flexibility.

opt\_var = sco.minimize(min\_profit\_stat, initial\_weights, method='SLSQP', bounds=bnds, constraints=cons)

As a next step developed the following simple program with custom rules.

### (B) Find the best Profit considering different combinations of price variations

#### **Process Flow:**

##### **Profit calculation logic:**

* Calculate estimated\_price = original\_unit\_price \* (1+change\_in\_price)
* If there is a price reduction,

Calculate current incr\_sales per tx = incr\_sales\*abs(change\_in\_price)\*10

and current incr\_cvr = incr\_cvr\*abs(change\_in\_price)\*10

* If there is no price reduction, set current incr\_cvr = 0.05 (default val)
* Calculate estimated\_volume = original\_units\_sold \* (1+current incr\_sales per tx) \* (1 + current incr\_cvr )
* Calculate profit\_with\_price\_change\_effect = np.sum(estimated\_price \* estimated\_volume) - np.sum(original\_cost \* estimated\_volume)

###### Assumptions:

* Compute the next best price and volume against base values (original unit\_price & units\_sold)
* When price is not reduced, consider the default value of incr\_cvr i.e. assume there is a cvr increment by 5%

##### **Price Change calculation logic:**

* Iterate through all Items
  + Apply maximum reduction for products with higher propensity to boost sales and conversion i.e. for different buckets of higher incr\_sales and incr\_cvr values
  + Increase price slightly if there is no incr\_cvr / incr\_sales and units\_sold is high and unit\_price is high
  + Decrease price if there is no incr\_cvr / incr\_sales and units\_sold is comparatively lower and unit\_price is low as well
  + **Note:** many different combinations of price reduction rules can be created by blending high-price, high-volume, low-priced, low-volume items with the conversion rate factor and sales demand values.

###### **Assumptions:**

* **Price change boundary {-10%,20%}**

##### **Main Workflow:**

* Invoke Price Change calculation logic
* Check if new profit is better than old profit
* If not, keep iterating
  + Find new price\_change and new profit
* If new profit and old profit are same then the process has converged

###### **Assumptions:**

* retain the price changes even if it doesn't yield better profit

#### 

#### **Demo**

Prerequisites: Build Docker Image docker-compose build

kaniska\_mac$ docker-compose up -d

Starting Pricing-Strategy ... done

kaniska\_mac$ docker-compose ps

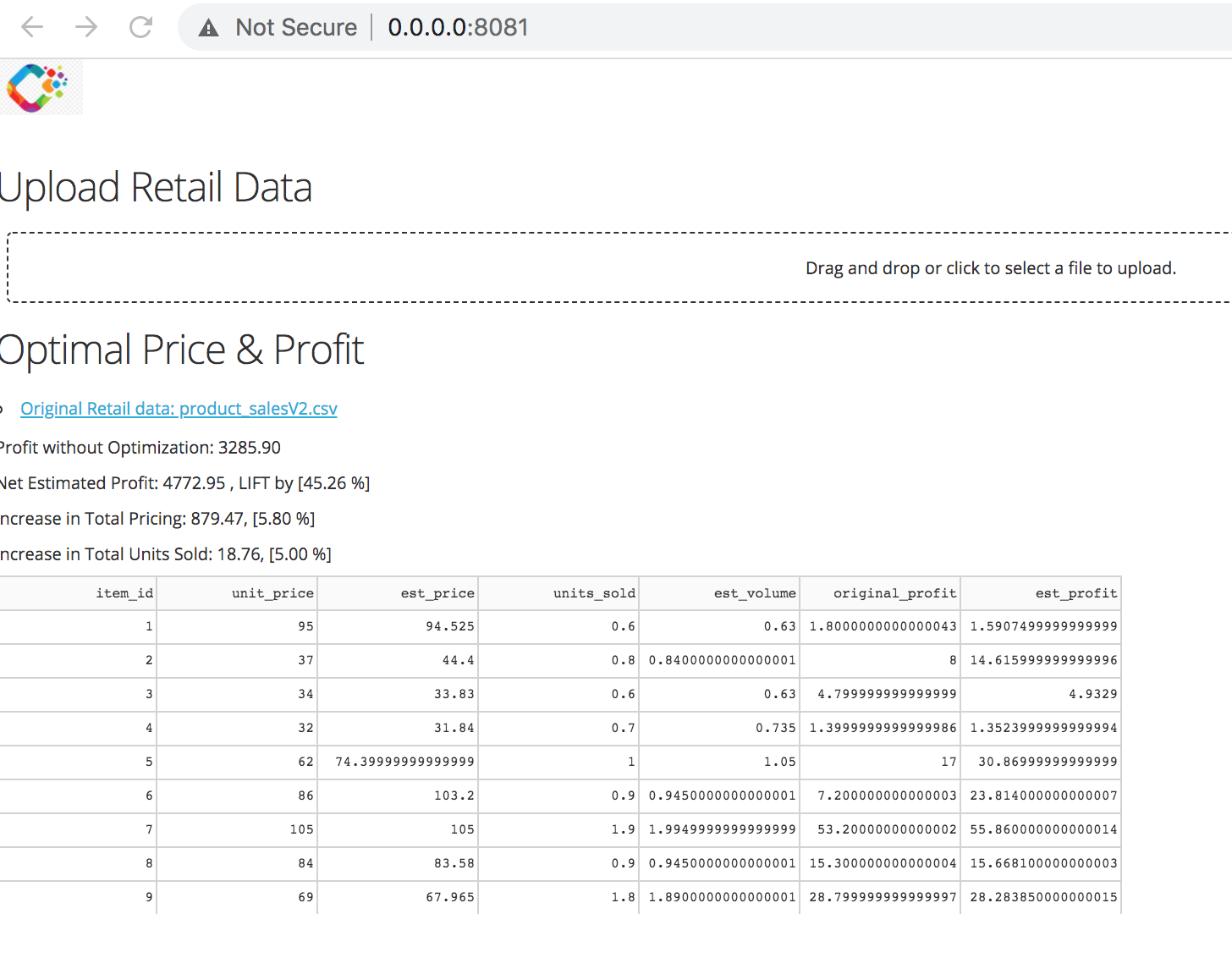
Name Command State Ports

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

Pricing-Strategy **python mainV2.py** Up 0.0.0.0:8081->8088/tcp

Profit without Optimization: 3285.90  
Net Estimated Profit: 4772.95 , LIFT by [**45.26 %**]  
Increase in Total Pricing: 879.47, [**5.80 %**]  
Increase in Total Units Sold: 18.76 %, [**5 %**]

##### Web App



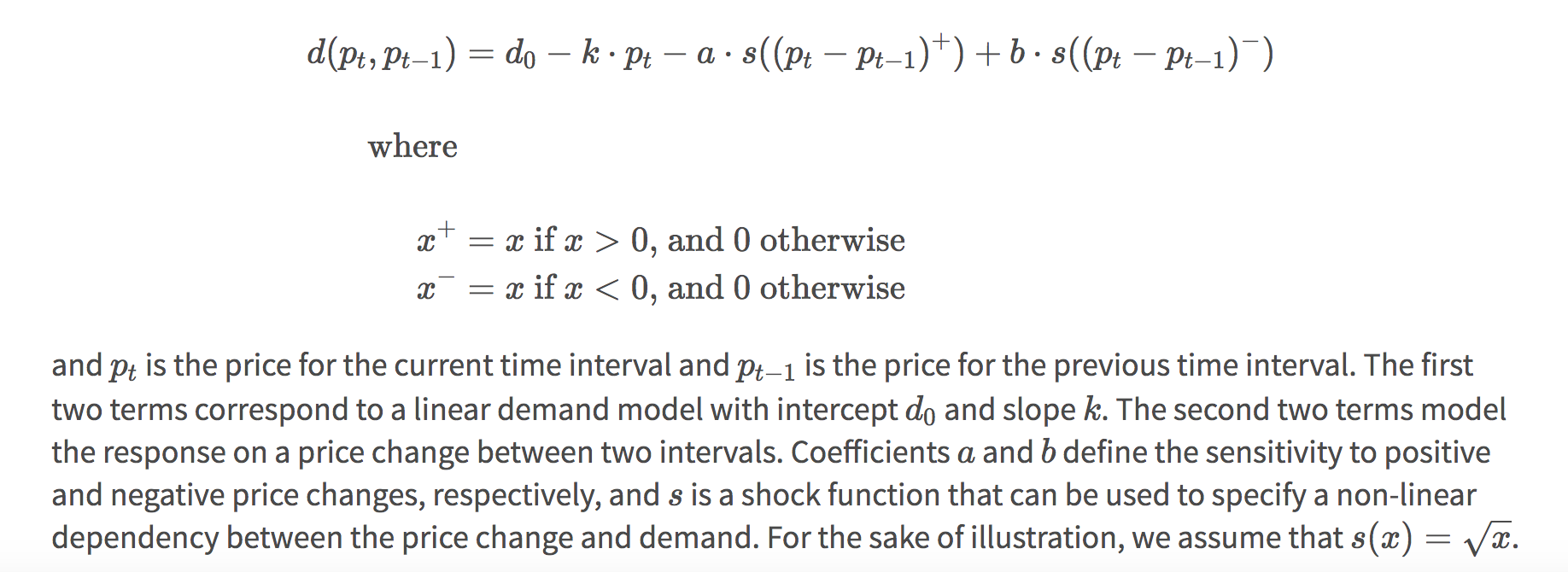
The above algorithm is a very rudimentary approach towards understanding the effect on sales by changing the price of a SKU and computing an optimal profit.

Now let’s switch gears and focus on how we can create a Demand Function

### (C) Create the demand function an item and find the maximum profit from predicted values

Expression of a generic demand function:

price decrease can create a temporary demand splash, while price increase can result in a temporary demand drop.



© <https://blog.griddynamics.com/deep-reinforcement-learning-for-supply-chain-and-price-optimization/>

#### Process Flow

Code: **DemandFunctionProfitEstimator.py**

itemA\_price\_lb = itemA\_base\_price\*0.90

itemA\_price\_ub = itemA\_base\_price\*1.2

itemA\_df = pd.DataFrame.empty

N = 100

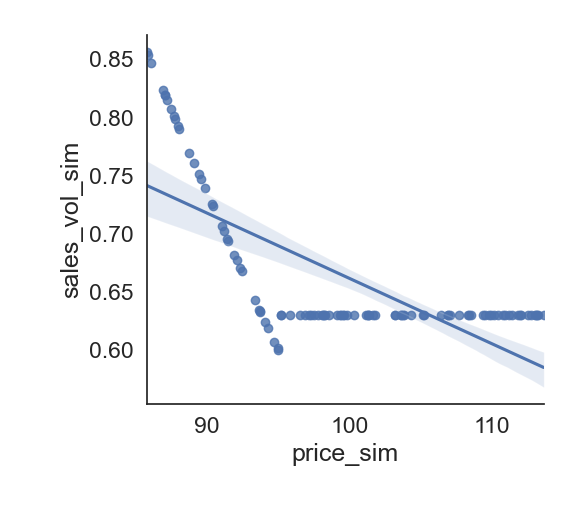
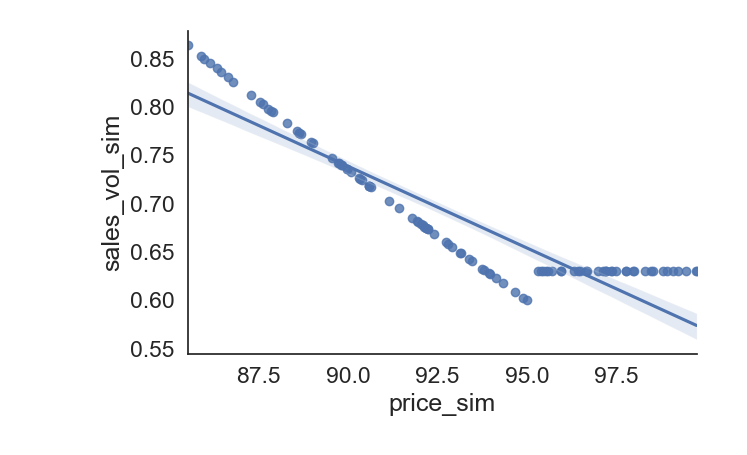
*# Generate 100 random price for the item => random.uniform(itemA\_price\_lb, itemA\_price\_ub)*

price\_sim = [random.uniform(itemA\_price\_lb, itemA\_price\_ub) **for** i **in** range(0,N)]

we calculate simulated profit for every time step for Item by applying the same logic as mentioned in Scenario (B)

*# plot demand curve*

These curves shows the relationship between simulated price and simulated sales volume



*# fit OLS model*

model = ols(**"sales\_vol\_sim ~ price\_sim"**, data = itemA\_df).fit()

*# print model summary*

print(model.summary())

R-squared: 0.558 indicating it explains 55.8% of model’s variability

p-value is 0 indicating the the Null Hypothesis that increase in price doesn’t reduce sales volume can be rejected

Intercept 1.3110 , price\_sim coefficient -0.0065

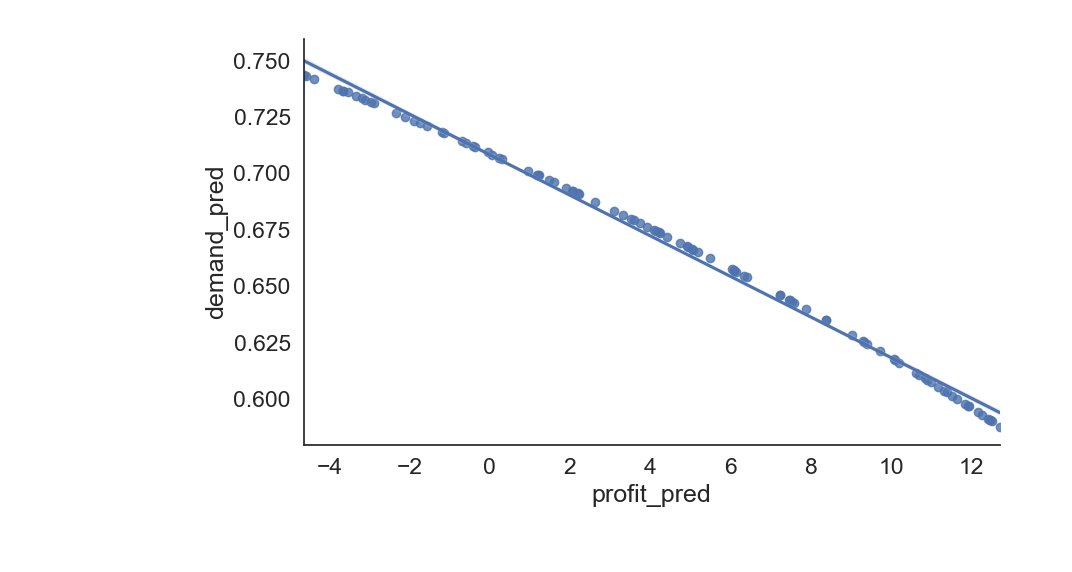
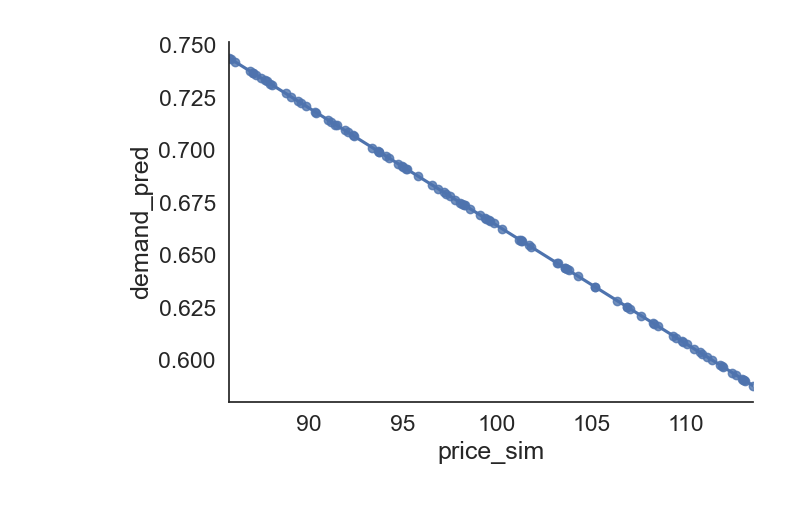
We can now compute the demand function by *plugging in regression coefficients: C= Intercept, S = Price Coeff* as follows *demand\_pred = C - S \* price*

We can now apply the demand function on a new set of price ranges for the Item.

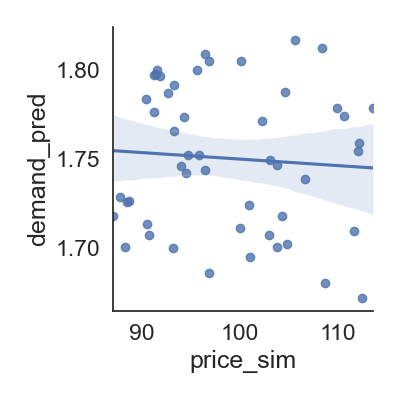
For the sake of brevity, applying on the simulated price values

itemA\_df[**'profit\_pred'**] = itemA\_df.demand\_pred \* (itemA\_df.price\_sim - itemA\_df.fixed\_cost)

We now fit the Demand Function against the same training dataset just for checking



Next we fit the curve on Test data (50 samples). As expected we don’t get a very good result as the Demand Function and OLS model is not optimized !



max\_val = itemA\_df2[itemA\_df2[**'profit\_pred'**] == itemA\_df2[**'profit\_pred'**].max()]

#### Limitations:

* This is just to explore the concept of Demand Function and show that if we can experiment with different price ranges for an item we can find a model with good coeff for determination and then use
* We haven’t removed any outlier values as we observed in EDA phase and didn’t change the price and quantity to log scale (log-log is recommended for Demand Function)
* we didn’t run additional iterations with different sample size , hence we didn’t get a better Coeff of Determination

## Future Improvements

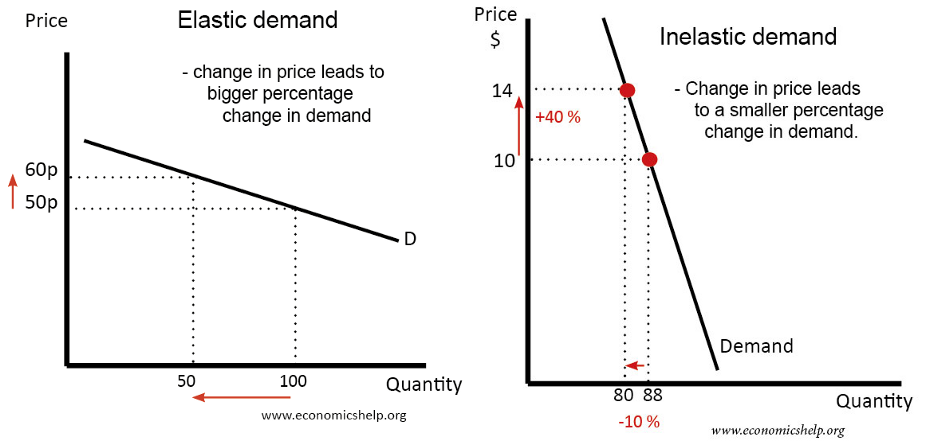
### **Improve the custom price variation logic**

The price variation logic as demonstrated in Approach-(B) in the POC section currently considers only one set of Business Rules with one specific range of threshold values for incr\_cvr param.

It can be further improved by running the Optimizer with different threshold values for different groups (incr\_cvr group, profit group, sales\_volume group) and then compute the max profit from the output of different combinations after removing outliers.

### **Improve Demand Forecast using Price Elasticity**

[Price elasticity of demand (PED)](https://en.wikipedia.org/wiki/Price_elasticity_of_demand) is a measure used in economics to show the responsiveness, or [elasticity](https://en.wikipedia.org/wiki/Elasticity_(economics)), of the quantity demanded of a good or service to a change in its price when nothing but the price changes. More precisely, it gives the percentage change in quantity demanded in response to a one percent change in price.



© <https://towardsdatascience.com/price-elasticity-data-understanding-and-data-exploration-first-of-all-ae4661da2ecb>

In marketing, it is how sensitive consumers are to a change in price of a product.

It gives answers to questions such as:

* “If I lower the price of a product, how much more will sell?”
* “If I raise the price of one product, how will that affect sales of the other products?”
* “If the market price of a product goes down, how much will that affect the amount that firms will be willing to supply to the market?”

Find the best Elasticity Coefficient based on simulated time series data.

Good amount of money are spent on promotions and campaigns

Estimating the demand against a price change can help us find the ROI of a promotion

**'elasticity\_l** = np.log(quantity\_delta/quantity) / price\_delta/price)

‘**'units\_sold\_l'** = np.log[**units\_sold**]

‘**'price\_l'** = np.log[**unit\_price**]

We can run GLM to generate a Normal distribution

**with** Model() **as** model:

GLM.from\_formula(**'units\_sold\_l ~ 'elasticity\_l'**, df\_l, family=**'normal'**)

trace = sample(4000, cores=4)

These observed data samples will now be our elasticity values which we obtained from GLM.

Note that we really need a considerable amount of observations!

Now we can configure and run Bayesian Model and create a distribution of the possible values of the elasticity conditions based on the observed and prior data.

Reference: <https://towardsdatascience.com/using-bayesian-modeling-to-improve-price-elasticity-accuracy-8748881d99ba>

### Improve Demand Generation by considering additional signals like is\_promotion, is\_weekday , is\_holiday, weekly change in demand, elasticity, log-price ratio

I have taken an excerpt from this notebook just to demonstrate how we can improve our prior Demand Function by considering various other features

© <https://github.com/shumingpeh/price-optimization/blob/master/ea-price-optimization-model.ipynb>

*X = sm.add\_constant(X)*

*y = self.itemA[['amount']]*

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

*$\beta\_{0} + \beta\_{1}P\_{i} + \beta\_{2}PromotionPresent + \beta\_{3}isWeekDay + \beta\_{4}isWeekEnd + \beta\_{5}isHoliday + \beta\_{6}PreviousPeriodDemand + \epsilon$*

*model.predict(X)*

*constant\_A = ( model\_param\_A\_df.const.values[0] + is\_weekday \* model\_param\_A\_df.is\_weekday.values[0]*

*constant\_numeric = np.array([[-1\*constant\_A],[-1\*constant\_B],[-1\*constant\_C]])*

*item\_A\_first = model\_param\_A\_df.weighted\_price\_usd\_A.values[0] \* 2*

*values = np.linalg.solve(model\_variables\_matrix, constant\_numeric)*

*original\_demand\_A = (*

*model\_param\_A\_df.const.values[0]*

*+ model\_param\_A\_df.is\_weekday.values[0]*

*+ model\_param\_A\_df.previous\_day\_demand.values[0] \* 16.44*

*+ model\_param\_A\_df.weighted\_price\_usd\_A.values[0] \* original\_price\_A*

*+ model\_param\_A\_df.weighted\_price\_usd\_B.values[0] \* original\_price\_B*

*+ model\_param\_A\_df.weighted\_price\_usd\_C.values[0] \* original\_price\_C*

*)*

*new\_demand\_A = (*

*model\_param\_A\_df.const.values[0]*

*+ model\_param\_A\_df.is\_weekday.values[0]*

*+ model\_param\_A\_df.previous\_day\_demand.values[0] \* 16.44*

*+ model\_param\_A\_df.weighted\_price\_usd\_A.values[0] \* 17.88810439*

*+ model\_param\_A\_df.weighted\_price\_usd\_B.values[0] \* 41.13139183*

*+ model\_param\_A\_df.weighted\_price\_usd\_C.values[0] \* 81.72909101*

*)*

*original\_total\_revenue = (*

*original\_demand\_A \* original\_price\_A*

*+ original\_demand\_B \* original\_price\_B*

*+ original\_demand\_C \* original\_price\_C*

*)*

*new\_total\_revenue = (*

*new\_demand\_A \* 17.88810439*

*+ new\_demand\_B \* 41.13139183*

*+ new\_demand\_C \* 81.72909101*

*)*

### **Long-term vision - leverage relevant Marketing Signals and adopt Experimentation Strategy**

Maximizing profits through price variation requires understanding of how the Items behave in the Market and Customer purchase process.

In the grand scheme of things; Item price, Item inventory, Item Ad spend, Item score, Item keyword rank in search-engine , Customer search intent , User-Item engagement metrics are all related to each other !

* Can we apply different price variation logic to different segments of products ?
* Is there any specific order of priority to promote items based on discount, newness and rating ?
* Is there a specific Price Mark Down / Up Business requirement ?
* Is the ATC / GMV for an item low even when more people are searching for it ?
* Is the Item rank in Search Engine is low even when clicks / impressions are high ?
* What is the optimal discount value for a certain group of products ?
* Are the items being categorized in correct catalogs ?
* Do we have Item Basket data ? Then we can apply a similar discount to the related group of items and boost sales of a group of items together.
* Certain types of Items are good candidates for New Customer Sign Up campaign
* Is a certain category Items showing the signs of Propensity to churn ? then they may be candidates for Customer Retention Campaign.
* Are the items being catered to Local Demand by leveraging location-specific discounts and demographic properties
* What is the customer satisfaction rate for the items ?
* Does an item have any seasonal pattern ?
* What is the current age and life-cycle phase of the Item ? Is it a Cold-start or long-lived item ?
* Are these items being marketed as per latest search trends and demands ?
* Are the items tuned to all types of faceted properties e.g. brand, gender, color, size ?
* Are we keeping track of OOS to avoid bouncing and opportunity loss ?
* Are we boosting perishable with shorter shelf life ?

There are various tools which provide access to different types of data based on a specific requirement !

E.g. Adobe Omniture => User Engagement metrics, eMarketeer => Ad spends, B2B, DSP , Flight Deck => supplier data, IRI => top brands, top items , Nielsen => POS data, Braze => Marketing attribution data , Brightedge => competitive keywords , SEMRush => Keyword Search Ranks , Google Ad => Keyword Ad ranks

Presence of different categories of Items and possibilities to combine multiple signals offer a great opportunity for implementing A/B Tests (Market-driven or simulated) and select right set of combinations of parameters to optimize profit.