What is the "Best" eBay Selling Price?

Don Lancaster
Synergetics, Box 809, Thatcher, AZ 85552

http://www.tinaja.com
don@tinaja.com
(928) 428-4073

By digging down into readily available **eBay** stats using some unusual tools and techniques, lots of potentially useful info can be extracted for serious **eBay Sellers**.

For instance…

> **Around 75 percent of eBay sales lose money.**
>
> **Less than 5 percent of eBay sales exceed $100.**
>
> **Price and sellthru are only weakly related; With an 0.4 sellthru being typical.**
>
> **A buyer is nearly THIRTY TIMES more likely to make a $20 purchase than a $100 one.**
>
> **The eBay market for big ticket items is much lower than most people suspect.**
>
> **The "best" eBay selling price is $24.95. And quite possibly the most profitable.**

Now, I would be the first to admit that you can make statistics show anything you want. Especially with limited sample sizes, hidden variables, plain old noise, any wrong assumptions, or preconceived notions of a desired outcome.

On the other hand, using stats to learn "**the nature of the beast**" will allow you to optimize your results. And possibly prevent you from making any stupid or counterproductive mistakes. Let's look at the statistical math and the obscure companion tools that strongly suggest the above claims…

Gathering the Data

Your first step should be gathering enough valid data. Yes, there are several new **eBay stat sites** including **this one**. But these may be doing different things in different ways than you need, so **it pays to generate your own "fresh" data**. One method might be to use the **eBay advanced search feature** and ask it for **all the completed U.S. auctions that have the word "of" in the title or description**. Reported in **descending cost order** at **100** entries per page.

This neglects a few subtleties such as category variations, non-closures, and partial Dutch sales, but it certainly seems a useful starting point. One that is easily refined later.

Doing so recently generated about **18,400** pages or a total of **1,840,000** closed transactions. A transaction here either represents a sale or a no sale. Let's call one of these pages a **bin** that holds the results of how buyers voted with both their wallets and their feet.

Each bin will have a **sellthru ratio**. Which we can define as **1/100th** of the total "green" sales in the bin. And will be a number from **0.0** to **1.0**. The total sales of each bin will be the average price per bin (very often a single value to the cent) multiplied by the sellthru ratio times the transactions per page.

Thus, in one of the **$23.55** bins having an **0.43** sellthru, the total delivered gross sales value will be **$23.55 * 0.43 * 100 = $989.00**.

Preliminary Data Processing

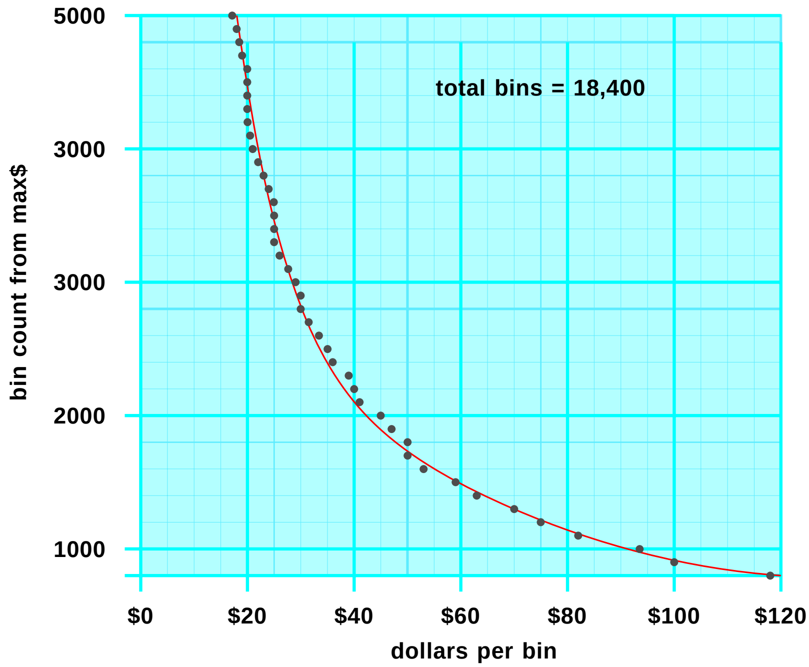
Now, 18,400 pages is a lot of data, so you might want to **downsample** to some reasonable number of values. Start your data base by selecting each bin that is a multiple of 100. Record the bin number (counting downwards from the highest price sales.) and the dollar value for that bin. Most bins will have a single dollar value. If not, use an appropriate average.

Then look closer at the page and count the "green" transactions. Divide this by one hundred to get your sellthru ratio. Enter this into your bin data base. Things will behave strangely different at very high or very low bin numbers, so **restrict your analysis to useful dollar values, say around $20 to $120.** Your data base of bin, dollar, and sellthru values might look like this...

```
/data1k5k [
            800  118.00   0.50
            900  100.00   0.39
           1000   93.50   0.73
             .  .  .  .  .  .  .
           4800   18.45   0.57
           4900   17.99   0.27
           5000   17.12   0.90
        ] store
```

Naturally, I will be using the incredibly superb **PostScript** language and **Distiller as a General Purpose host based PostScript interpreter**. Full code details appear in the **sourcecode** to this **GuruGram**. With additional tutorials and support **here** and specific details on our **Gonzo Utilities** **here**.

Your preliminary plot of bins versus useful dollars should look like this...



We have purposely plotted this "sideways", because we will later be much more interested in **bins per dollar** rather than our "raw data" dollars per bin. At first glance, our black data point dots look a little noisy. But most of this "noise" is easily explained away as the two **listing cost penalties** that **eBay** imposes above **$24.99** and **$49.99**.

We can immediately see several surprising results without any further analysis.

As we have seen **here** and **here**, a reasonable breakeven sale price for a serious long term eBay venture is **$19.63**. Well, **$19.63** occurs near bin **#4643** of about **18,400** total. Which implies that **(18,400 - 4663) = 74.8** or roughly **75 percent of eBay sales loose money**. When properly full burden true cost accounted.

Sales drop under a hundred dollars per bin after bin **#900**. A mere **900/18,400 = 4.89** or **less than 5 percent of eBay sales close above $100.**

We will also note that **the bins per dollar drops off astonishingly fast** with increasing price. Thus, **people are much more likely to make a low dollar buy**.

We will see exactly how much shortly after we have done…

Some Further Analysis

We have a slightly noisy plot of dollars versus bins. What we really need to continue is a clean plot of bins versus dollars. So we can determine the total dollars delivered for each dollar value. Approaches to do this might include filtering, equation solution, interpolation, or approximation.

I chose to "eyeball" a math function that reasonably closely seems to fit our data. This is the red curve shown in the above figure. Which seems to me to be in the "not half bad" class. A little trial and error thus saves us bunches of really ugly digital filtering and least squares work.

We first **normalize to a unity graph space**. An intermediate variable is first calculated to make the curve "run backwards" and get a normalized unity value at $17.50 or so. The math behind curve has first, second, and eleventh order components…

FIRST, calculate an intermediate variable…

y = (1 - (dollars/120))/0.85

THEN, find the bin number for that variable…

bin# = 800 + 4200*(0.55*y^11 + 0.35*y^2 + 0.07*y)

That eleventh order (!) term gives us a clue that the bins versus dollars will drop rather fast with increasing dollars. Especially in the $20 to $50 range.

This equation, of course, only applies to this particular data set on this particular day only over this particular dollar range.

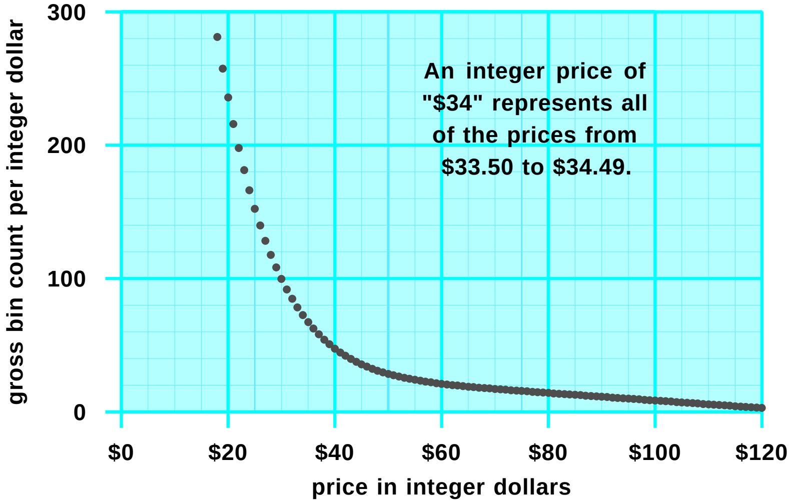
We now have a fairly accurate way to calculate the number of bins for a given dollar value. Let's next **quantize** our dollars to **integer** values. Then we will find the number of bins for the **next higher** and **next lower** integer dollar values.

Then we will take half of this as the number of bins per quantized dollar.

All of which is the equivalent of answering questions like "How many bins do prices from **$28.51** to **$29.50** collectively own?" Once we have the number of bins and the average quantized price, we can eventually find the **income for each price**.

Whose variation is our goal here.

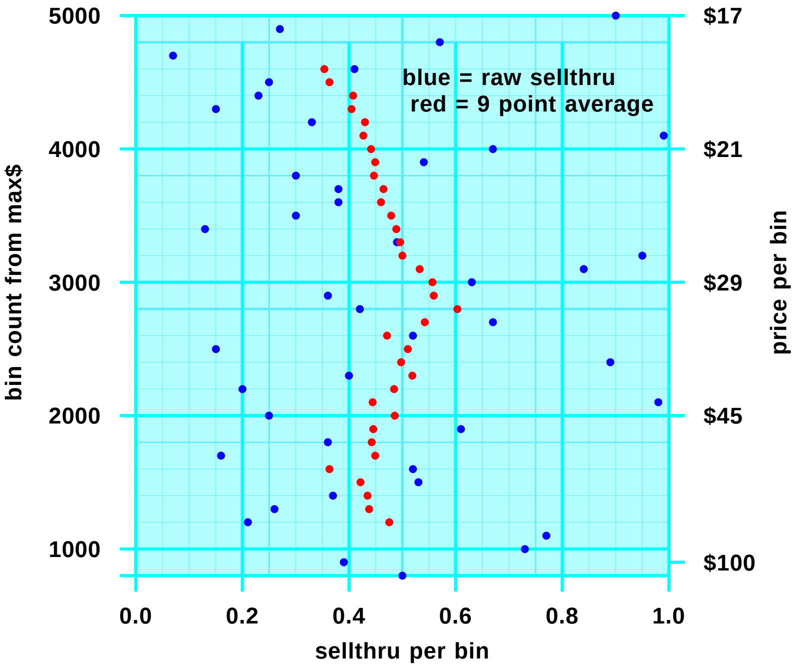
Yeah, we could have taken a derivative instead, but the actual differential seems more precise and less fraught with peril. For this plot of bins per dollar…



Before we can find the actual number of sales per integer dollar or the cash value of each integer dollar selected by our buyers, we will first have to look at…

The bad effects of sellthru

Our sellthru plot might look something like this…



Any bin consists of a mix of really good and really bad deals on the seller end. And grabbed and missed opportunities on the buyer end. One particular seller can easily dominate most of a bin if they have lots of identical prices closing on the same day. So, it is not uncommon for adjacent bins to have wildly different sellthru ratios.

The question we need to answer is "**How does sellthru vary with price?**"

The blue dots of our raw data here give us an initial plot that does not seem very encouraging. We obviously have lots of random events going on here. But many of these are "high frequency" noise that does not correlate with price at all.

What happens if we low pass filter these apparently random dots?

One easy filtering route is to use a **nine sample running average**. Which, with our red dots, does **suggest** a **mild** variation in sellthru versus pricing. Something like 0.4 at $20 rising to 0.59 at $30 and then dropping back down to $0.4 at $100.

We would expect the sellthrough to further head south above $100 because high price offers are best **prevented** from selling on their first few listings. Thus trading a few dollars of risk for hundreds of dollars of gain if your opening price was too low. In addition, some big ticket opening prices are way too high, and many high end sales do not actually close.

Yeah, we are literally on shaky ground here. Going back into our original data and averaging several adjacent bins would help out a lot. A further difficulty is that our plot is linear in noisy bin number space rather than clean dollar space.

We still might make what may well turn out to be a reasonable guess of sellthrough over the $20 to $100 range to be something like...

FIRST, calculate an intermediate variable...

y = dollars/120

THEN, find the sellthru for that variable...

sellthru = -2.9*(1-y)^3 + 2.9*(1-y)^2 + 0.4*y

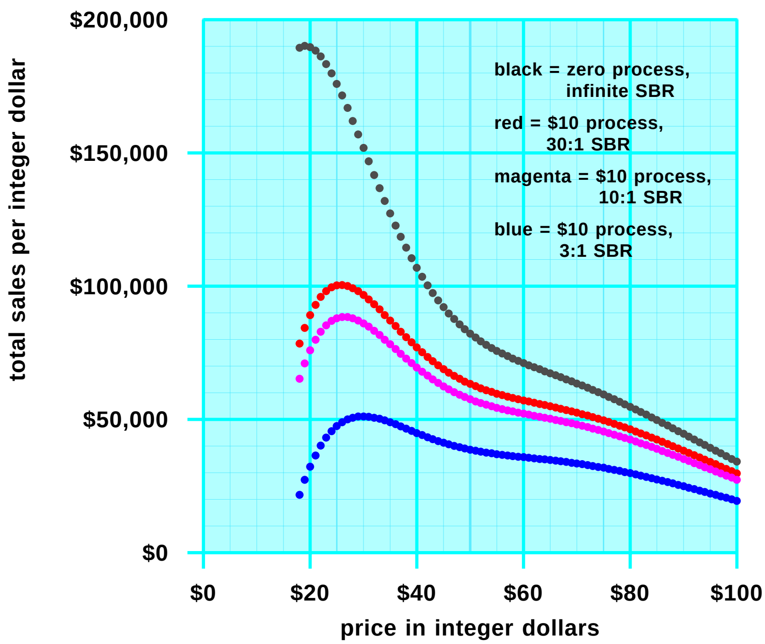
A repeat reminder that this sellthru is based on noisy data that you might want to improve upon later. And is **only** approximately valid for our current example over a price range of $20 to $100. Nonetheless, this gives us a necessary starting point to find the total income generated for any integer dollar value.

We are now ready to look at our...

Actual income plots

We can now at long last plot our gross and (potential) net incomes for each integer price value range. We do this by multiplying the integer price times the number of bins for that price times the sellthru for that price.

Giving us this **black dot** plot…



As previously noted, this curve is surprisingly steep. The total sales revenue for all $20 items on this day (representing prices from $19.50 to $20.49) was a huge $189,741. The sales revenue for all $100 items on this day was sharply lower at a mere $34,174. This is a ratio of 5.55 times. But since a $100 sale is five times more valuable than a $20 one, something like 27.76 or **NEARLY THIRTY TIMES** as many eBay sales get completed at a $20 price than at a $100 one.

Let's repeat that…

> **NEARLY THIRTY TIMES as many eBay sales get completed near a $20 price than near $100.**

The black dotted plot is **gross income only**. And is derived completely from "demand side" data. Now, **if** you had zero costs and zero time involved, you clearly should be selling at the lowest possible price.

The real world obviously does not work that way, so **you have to adjust the black plot for your operating costs**.

We might subtract out a fixed per-sale fulfillment cost of $10 plus an appropriate **sell/buy ratio** for your costs of materials sold. This gives you the red plot for a (highly recommended) SBR of 30, the magenta curve for a SBR of 10, or the blue curve for an (avoid at all costs) SBR of 3. Your mileage, of course, will vary sharply with your supply, operating, and fulfillment costs.

Plus your **Leavenworth Ratio** that determines how much you are net net net paying yourself compared to hourly rates at Folsom or Leavenworth. Do not forget to include your **pro-rated water bill** (and similar obscure costs) in all of your fully burdened calculations.

From which, we see that **the maximum dollar returns are apparently generated for eBay for final selling prices in the $24.99 range.**

Are We There Yet?

Sadly, there is a flaw in the above analysis. "Constant integer dollars" are more valuable at $20 than at $100. Because they represent **five percent** of all buys at $20 and only **one percent** of all buys at $100. A **log** or **constant percentage** distribution would probably be more suitable to accurately explore "best" eBay pricing.

Simply normalizing everything to $20 gives us a different plot with somewhat higher "best" eBay pricing. With peaks as high as $79. A better approach would be to use eBay's new **research service** and work in constant one percent price increments.

This awaits a future **GuruGram**.

For Additional Assistance

Similar tutorials and additional support materials are found on our **Auction Help** library page. Tested and proven sources that let you achieve a 30:1 or higher SBR are found **here**. Your own custom local or regional product finder can be created for you per **these details**. As always, **Custom Consulting** is available on a cash and carry or contract basis. As are seminars.

For details, you can email **don@tinaja.com**. Or call **(928) 428-4073**.