**Understanding Boxplot: Infinity Gauntlet of the Dataverse**

Use boxplots and Tukey’s Method to eliminate outliers in a snap of your fingers (or code)



CC image from [Flickr](https://www.flickr.com/photos/anazasicustom/45484665074/)

Boxplots can be intimidating to many beginners.

***It’s because they are jam-packed with statistical insights!***

But, if you are willing to dig just a bit deeper, they can reveal a treasure trove of information. Boxplots are powerful tools in statistics and data science.

*How powerful?*

Let’s put it this way:

*If Thanos were a data scientist, the boxplot would be his****Infinity Gauntlet***— with the power to summarize data into his fist, and to eliminate outliers with a snap of his fingers!

And just like in the Avengers, where infinite universal power is concentrated in 6 infinity stones —***the power of summarizing huge amounts of data is condensed into just 6 values.***

**Boxplots represent these 6 values visually.**

Using them you can:

* Get a great sense of numeric data
* Do a quick graphical examination
* Compare different groups within the data
* Use them to know and eliminate outliers from the data

***That’s what you call a data superpower!***

Let’s see how to understand a boxplot, the 6 values it uses to summarize the data, and how to use them to eliminate outliers with a **snap!**

**Understanding Boxplots**

Since we established the gauntlet analogy so thoroughly, let’s make the most of it to understand the infinity stones — uh sorry, I meant the 6 important values.

This image will make it a lot clearer!

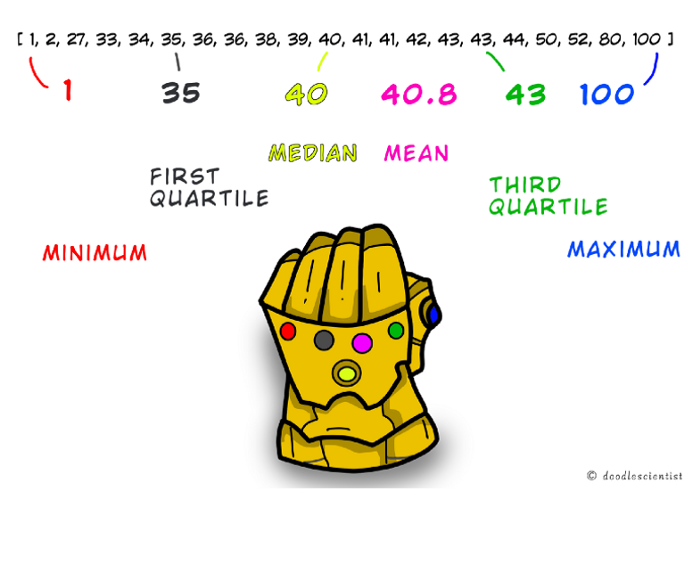


Image by the author

Take an ascending order **sorted list** of 21 numbers as seen in the image.

* The **Minimum**, represented by the stone on the pinky is the smallest value in the list — 1 in this case.
* The **Maximum** is the largest value, i.e. 100.
* The **Median**is thenumber right in the dead center of the list. 50% of the data lies on each side of the median. 40 is in the middle of the above list.
* There’s also the **Mean,**which is in the middle too. But not the middle of the list, but rather in the ***arithmetic middle*** of the values in the list — it is a sum of all values divided by the count of values in the list, which is 40.8 in this case.

Most of you already know the above values, as they are very commonly used. But what about the remaining two?

* The **First Quartile**(or Q1)is the value under which 25% of the data points lie. In a sense, it is the median of the first half of the data.
* The **Third Quartile** (or Q3) is the value under which 75% of the data points lie. It is the median of the second half of the data.

(Note: *the Median is itself also called the second quartile or Q2*)

That is it!

***Close your gauntlet fist and you have squeezed 21 numbers into just 6 values.***

And it doesn’t matter if it’s 21 or a billion, these 6 values are enough to give you a lot of insights.

Now let me show you how these values are visually represented.

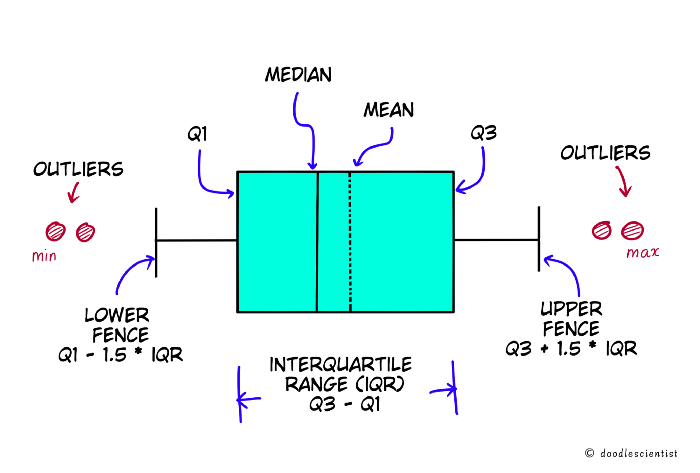


Image by the author

You see here that there is even more interesting info that we can get from this plot.

* **Inter Quartile Range (IQR)**is the difference between the third quartile and the first quartile: (Q3 — Q1). It gives the range of the middle half of the data.

**What are the T-shape protrusions on either side of the box?**

They are called whiskers or fences. They fence off the relevant data from the outliers.

* The**Lower Fence** is calculated as Q1 — (1.5 \* IQR)
* The **Upper Fence** is calculated as Q3 + (1.5 \* IQR)

Anything outside these limits is an outlier in the data.

***Phew!***

That is a lot of information in a single chart!

Now, let us use python to take the list and generate the chart automatically. It is really simple using Plotly and chart-studio.

I’ve exported the plot to chart studio so you can see the interactive version below.

Chart created by the author in Plotly Chart Studio

There you go, it’s so simple!

* In one glance, you can see the median, mean, range of the data, and outliers
* You can see that 50% of the data lies between values 35 and 43
* You can also infer some more characteristics of the data. The mean line is to the right of the median line, hence it is a ***right-skewed distribution***

I hope you understand the boxplot and its power now.

But, hang on a moment, that is only half the story.

**Now that you have the power of the figurative gauntlet, you need to snap your fingers too!**

Let us see how to use it to eliminate outliers from the data with a real-world example.

**Snap! Eliminate the Outliers**

You know now that the upper and lower fence calculated using the interquartile range can be conveniently used to separate the outliers from the data.

But did you wonder why the separation is 1.5 times the IQR on either side?

**The answer lies in statistics.**

According to the [68–95–99.7 rule](https://en.wikipedia.org/wiki/68%E2%80%9395%E2%80%9399.7_rule#:~:text=In%20statistics%2C%20the%2068%E2%80%9395,deviations%20of%20the%20mean%2C%20respectively.), most of the data (99.7%) lies within 3 standard deviations ( < 3σ) from the mean on either side of a standard distribution. Everything outside it is an outlier.

Now, the first quartile and the third quartile lie at 0.675 σ on either side of the mean.

Let's do some quick math.

Let X be the multiplying factor we need to calculate  
  
Lower Fence = Q1 - X \* IQR  
 = Q1 - X \* (Q3 - Q1)  
  
# The lower fence should be at -3σ  
# Q3 is 0.675σ and Q1 is -0.675σ  
  
-3σ = -0.675σ - X \* (0.675σ + 0.675σ)  
-3σ = -0.675σ -1.35σX  
  
X = 2.325 / 1.35  
 ~ 1.7  
  
# Similarly, it can be calculated for upper fence too!

We get a value of approximately 1.7, but one uses 1.5.

**Using this method to remove outliers is called Tukey’s Method.¹**

*(John Tukey, after whom this method is named, allegedly said 1.5 is chosen because 1 is too small and 2 is too large!)*

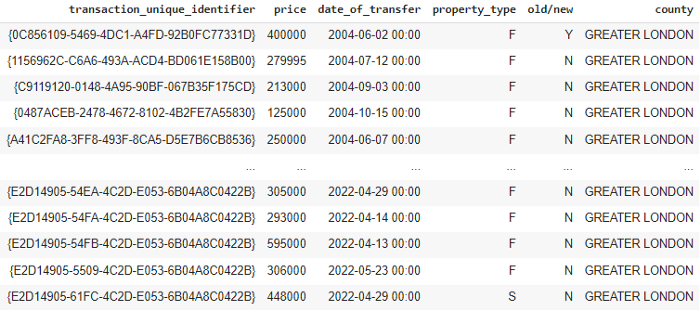
It is one of the simpler methods in statistics but works surprisingly well.

**Let's see a real-life example and build a figurative snap with python.**

I have used the public housing price data from England for 2022.²

# imports  
import chart\_studio  
import plotly.express as px  
import pandas as pd  
  
# Housing price data  
col\_names = ["transaction\_unique\_identifier",  
 "price",  
 "date\_of\_transfer",  
 "postcode",  
 "property\_type",  
 "old/new",  
 "duration",  
 "PAON",   
 "SAON",  
 "street",   
 "locality",   
 "town/city",   
 "district",   
 "county",   
 "PPD\_category\_type",   
 "record\_status\_monthly\_file\_only"]  
  
# Read data  
df = pd.read\_csv('http://prod.publicdata.landregistry.gov.uk.s3-website-eu-west-1.amazonaws.com/pp-2022.txt',  
 header = None,   
 names=col\_names)

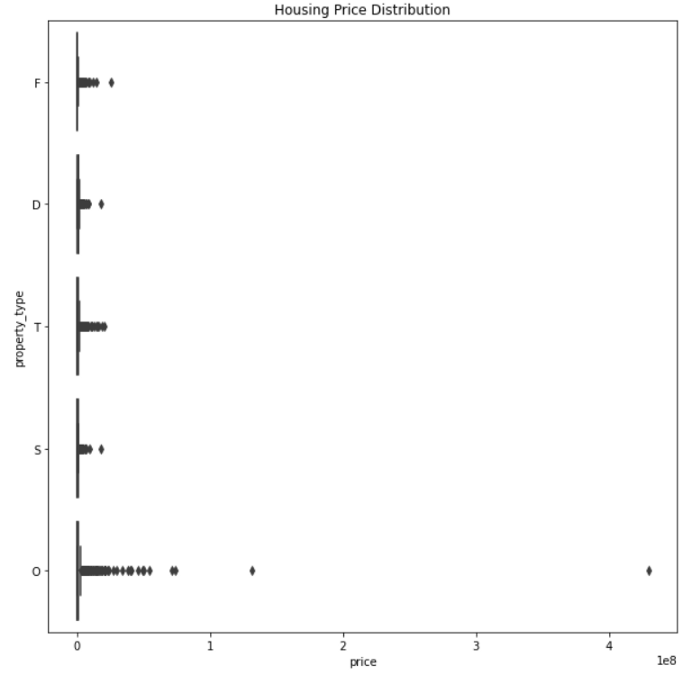
The first few columns look like below:



Screengrab by the author

Now, let’s quickly look at the property types for the county of Greater London and their prices using boxplots.

# Filter data for the county Greater London  
df= df[df["county"] == "GREATER LONDON"]  
  
# Boxplot of the fractional data  
sns.boxplot(x = df['price'],   
 y = df['property\_type'],   
 orient = "h").set(title='Housing Price Distribution')



Screengrab by the author

**That just looks awful, doesn’t it?**

There are huge outliers, such that we can't even see the *boxes* in the box plot!

That is not even good enough to create an interactive chart.

**Let’s create a neat little script using what you learned about IQR.**

Python has the *quantile function*, to get the part of the data within the defined quantiles.

# Create a function to get the outliers  
  
def get\_outliers\_IQR(df):  
  
 q1=df.quantile(0.25)  
 q3=df.quantile(0.75)  
   
 IQR=q3-q1  
  
 lower\_fence = q1-1.5\*IQR  
 upper\_fence = q3+1.5\*IQR  
  
 outliers = list(df[((df < lower\_fence) | (df > upper\_fence))])  
  
 return outliers

Let’s use it with our data frame to remove the outliers.

(**Please note:** *normally one would remove outliers from each group, but I am simplifying here to remove them from the entire price column*)

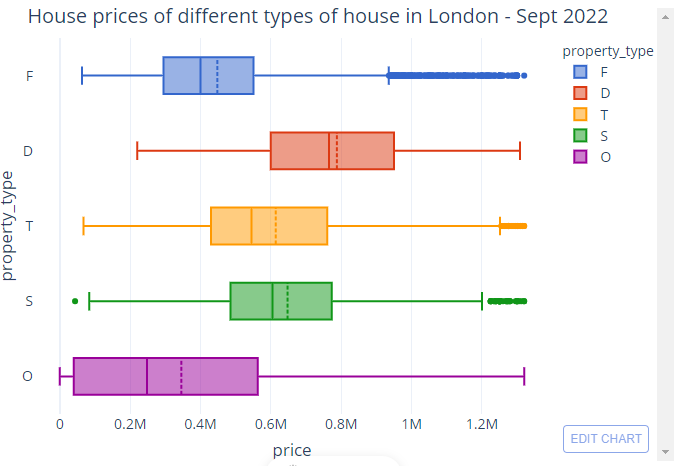
# Get outliers from the prices  
outliers = get\_outliers\_IQR(df['price'])  
  
# Remove outliers that we got in the outlier list - aka the Snap!  
df\_snap\_outliers = df[~df['price'].isin(outliers)]

Wait for it…

**Snap!**

This time, let's look at the interactive boxplot chart.

# Create boxplot from the list  
fig = px.box(data\_snap\_outliers,   
 x="price",   
 y="property\_type",   
 color="property\_type",  
 orientation='h',   
 template = "plotly\_white",  
 color\_discrete\_sequence= px.colors.qualitative.G10,  
 title="House prices of different types of house in London - Sept 2022"  
 )  
  
# There are many quartile calculation methods.   
# The one we discussed is calculated in plotly with quartilemethod = "inclusive"  
fig.update\_traces(quartilemethod="inclusive", boxmean = True)  
  
# Set margins, font and hoverinfo  
fig.update\_layout(margin=dict(l=1, r=1, t=30, b=1),   
 font\_family="Open Sans",   
 font\_size=14,  
 hovermode= "y"  
 )  
# Show plot  
fig.show()



Created by the author using Plotly Chart Studio

**Looks like the snap worked!**

Even at a glance, you can deduce a lot from the chart:

* The distribution is right-skewed for each category, which means that a lot more houses have a higher price
* The O-category houses have a much higher variability and range as compared to the other groups
* The D-category houses are generally more expensive

**Awesome right?**

You applied boxplots to visualize real-world housing data and used them to eliminate outliers!

**Improvements and Concluding Thoughts**

In this post, you learned about boxplots and outlier elimination using the analogy of the infinity gauntlet (I hope that you are Marvel fans😉).

I have given a simple explanation for beginners, but one can go even further to do an in-depth analysis using the plots.

However, one thing to remember is that **Boxplots** and the **Tukey Method** are just some of the many tools and methods in statistics.

You need to understand when they are most suitable to use.

For example, outliers may be sometimes useful and need not even be strictly eliminated.

Boxplots, likewise, also cannot be used always. They have a disadvantage in that we are not able to see how many data points are there within the group.

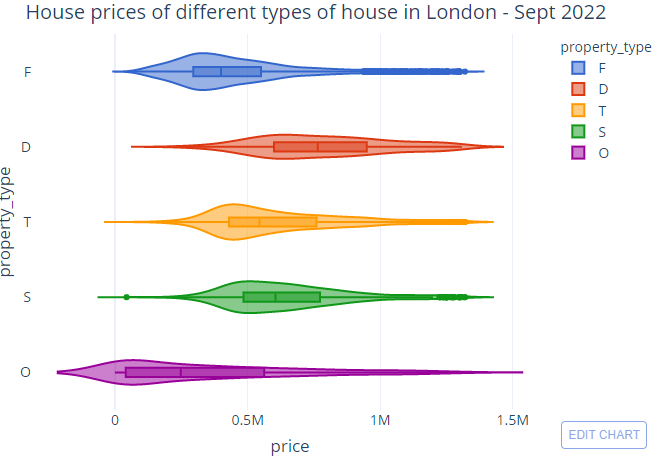
This can be solved either using the all-points parameter within Plotly’s boxplot function where we can see the data points along with the box or with a different kind of plot altogether — **a violin plot**.

A violin plot gives the data density along with the box plots, with the width of the violin indicating the frequency of data.

**In a sense, it is a combination of a boxplot and a histogram.**

Check out how the house category example looks with a violin plot:

# Create boxplot from the list  
fig = px.violin(data\_snap\_outliers,   
 x="price",   
 y="property\_type",   
 color="property\_type",  
 orientation='h',   
 template = "plotly\_white",  
 color\_discrete\_sequence= px.colors.qualitative.G10,  
 box = True,  
 title="House prices of different types of house in London - Sept 2022"  
 )  
  
# Set margins, font and hoverinfo  
fig.update\_layout(margin=dict(l=1, r=1, t=30, b=1),   
 font\_family="Open Sans",   
 font\_size=14,  
 hovermode= "y"  
 )  
# Show plot  
fig.show()



Created by the author using Plotly Chart Studio

One of the things we can deduce here is that even though the O-category prices have a much bigger range, fewer houses from this category have been sold as compared to the F-category — which has a higher data density.

*Cool isn't it?*

So why did we start with boxplots instead of violin plots?

Because it is essential to get the basics from the boxplot first, as a violin plot is only a better variation of it.

***But don’t worry, I will cover violin plots in detail in another post!***

I hope you enjoyed reading and learned a lot! I had a lot of fun writing this piece and would love to hear from you if you have any feedback.

Until then,

*Happy learning!*

*Fin.*