Splunk project:

amazon
reviews

SJSU
Dept of Computer Science

Submitted to :
Prof. Peter Zadrozny

Date:
Dec-7-2014

By:
Akshay Baheti
Manoj Gyanani
Dhruv Mevada

Index

1. Introduction

Our data consists of reviews from amazon. The data spans a period of 16 years, including approx 35 million reviews up to March 2013. Reviews include product and user information, ratings, and a plaintext review. Reviews provide information about the quality of products and what products are good for a certain group of people. Data was downloaded directly on the server from snap.stanford.edu/data/web-Amazon-links.html

Our analysis is primarily focused on all product reviews from amazon. Dataset files used :
all.txt.gz
related.txt.gz
brands.txt.gz
categories.txt.gz

1.1 Server Details

The project is carried out on one server with GoGrid. It is a XLarge server. We have not used a distributed set-up. The credentials to login to the Splunk server are given below.

URL: http://216.121.71.76:8000/
Username : admin
Password : Pictpune12345!
Splunk version : 6.2.0
License: Developer License
SSH Login: root/Pictpune12345!

1.2 Data Fields

1. all_reviews: It provides detailed information about reviews of all products sold by amazon. One review entry has the following fields:

product/**productId**: It describes unique id of the product.
product/**title**: It describes product name or title allotted to the product.
product/**price**: It describes price of the product.
review/**userId**: It describes id of the user who reviewed product.
review/**profileName**: It describes name of the user who reviewed product.
review/**helpfulness:** It describes fraction of users who found the review helpful.
review/**score**: It describes rating of the product.
review/**time**: It describes time of the review in unix format.
review/**summary**: It describes review of the summary.
review**/text**: It describes text of the review.

2. related.txt : It provides information about co-purchased products. It tells that what product(s) is/are also purchased by customer who purchased a product.
One entry in this file has the following fields.

productId: It describes product id of the product.
coPurchased: It describes product id(s) of the product(s) purchased by the customer who purchased the product with productid mentioned in productId field.

3. brands.txt: It provides information about product's brand. It tells what product belongs to which brand. Based on this data we can draw patterns on which brands is reviewed the most. Which brands have a good review,etc.
Fields are as explained below:

productId: It describes unique id of the product.
brand: It describes the brand name to which product belongs.

4. categories.txt: It provides category information for all products. It tells that product belongs to which category and falls under which sub category.
Fields are as described below:

productId: It describes unique id of the product.
pCategory: It describes primary category under which the product falls. Example Books
sCategory: It describes secondary category under which the product falls. Example Fiction,Thriller,Literature.

2. Loading the Data into Splunk

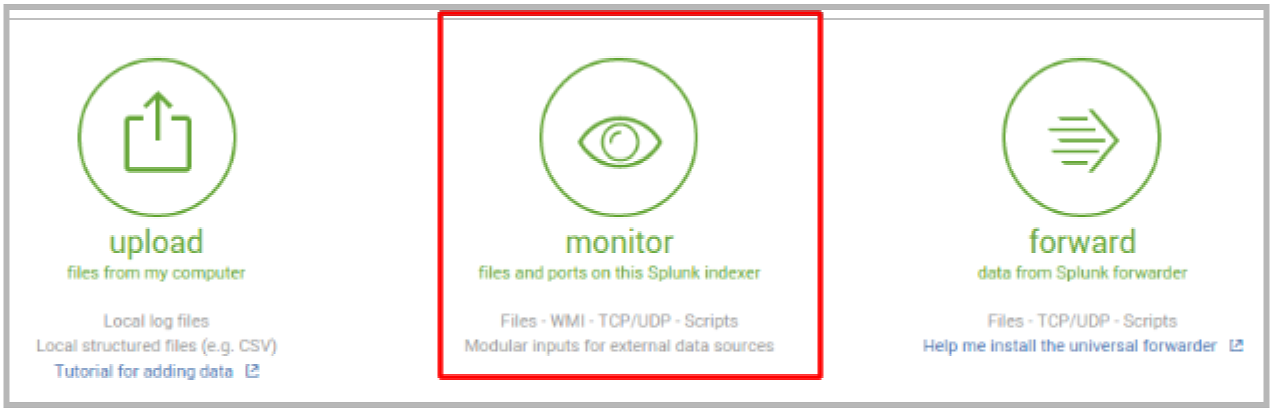
There are various ways of adding data to Splunk. Some of them are as below:

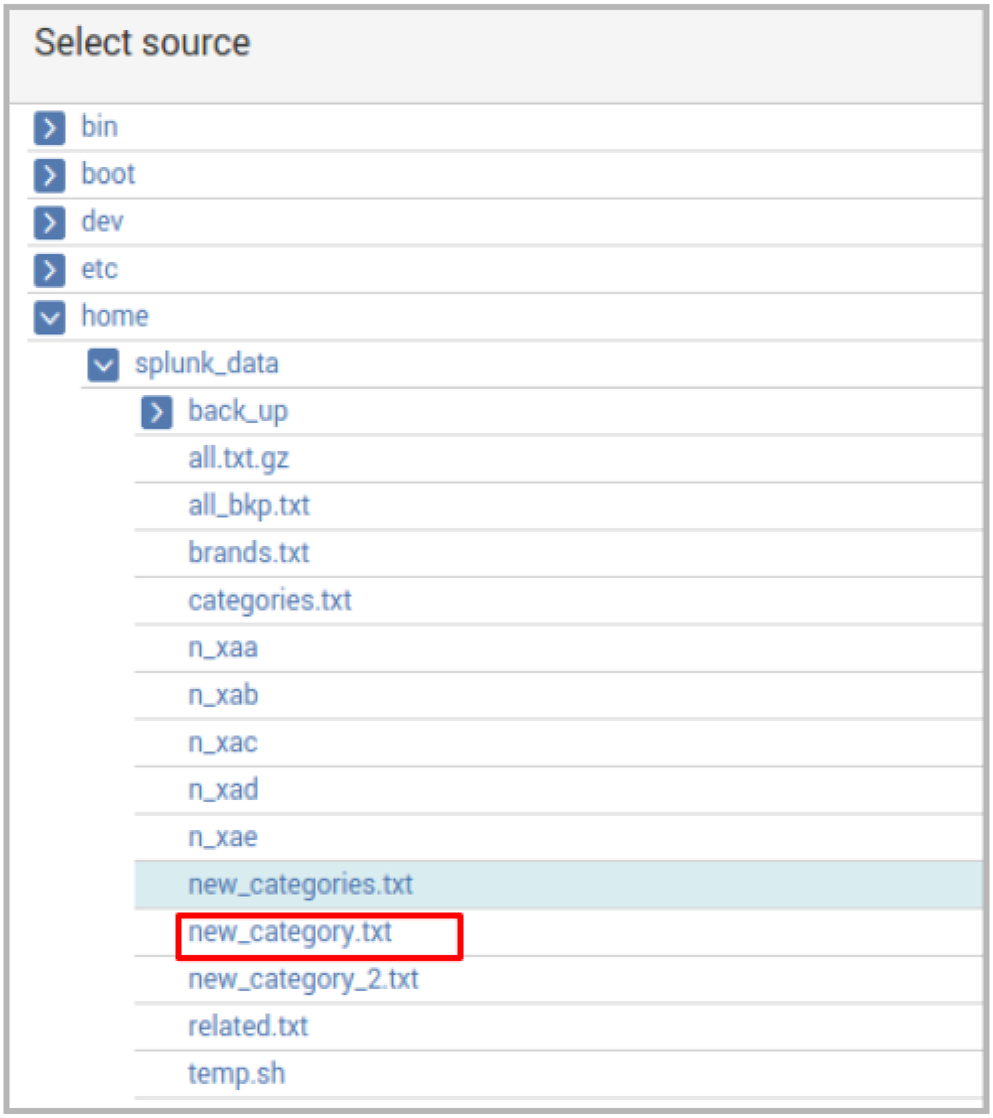
Apps: This offers pre-configuring inputs for various types of data sources.
Splunk Web: We can configure most inputs using Splunk Web data input pages. This provides a GUI-based approach to configure inputs. We can access the Add data landing page from either Splunk Home or System.
Splunk's CLI: We can use CLI (Command Line Interface) to configure most types of inputs.

2.1 Uploading data to Splunk using Splunk Web Interface

We first downloaded all the data on the server. Then uploaded the data to splunk using the Splunk Web Interface.

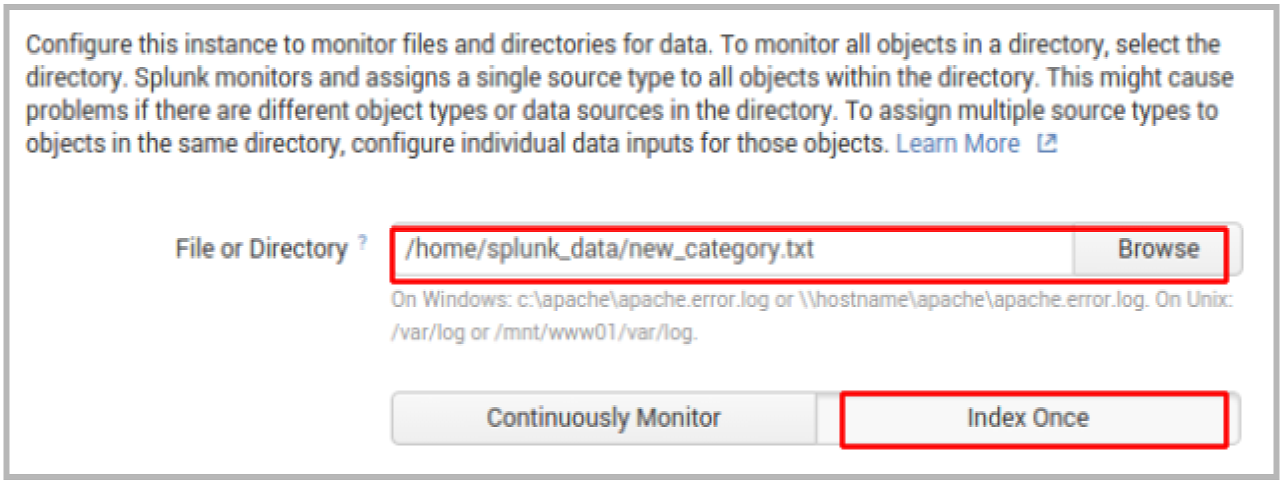
1. Upload via the monitor tab. Click on monitor, select “file and directories” and index once option. Select the file to be upload from the server. Steps are shown in the images below.


upload
files from my computer
Local log files
Local structured files (e.g. CSV)
Tutorial for adding data
monitor
files and ports on this Splunk indexer
Files - WMI - TCP/UDP - Scripts
Modular inputs for external data sources
forward
data from Splunk forwarder
Files - TCP/UDP - Scripts
Help me install the universal forwarder



Select source

- bin
- boot
- dev
- etc
- home
  - splunk_data
    - back_up
    - all.txt.gz
    - all_bkp.txt
    - brands.txt
    - categories.txt
    - n_xaa
    - n_xab
    - n_xac
    - n_xad
    - n_xae
    - new_categories.txt
    - new_category.txt
    - new_category_2.txt
    - related.txt
    - temp.sh



Configure this instance to monitor files and directories for data. To monitor all objects in a directory, select the directory. Splunk monitors and assigns a single source type to all objects within the directory. This might cause problems if there are different object types or data sources in the directory. To assign multiple source types to objects in the same directory, configure individual data inputs for those objects. Learn More

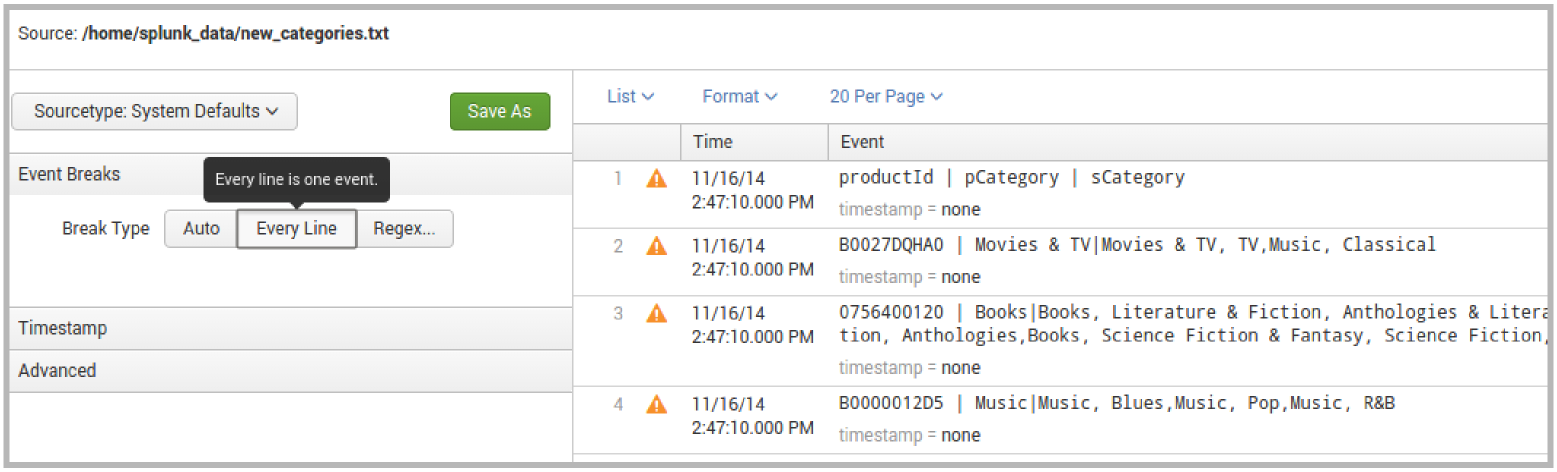
File or Directory ? /home/splunk_data/new_category.txt Browse

On Windows: c:\apache\apache.error.log or \\hostname\apache\apache.error.log. On Unix: /var/log or /mnt/www01/var/log.

Continuously Monitor | Index Once

2. Select the proper event break based on the file format

   a. Here every line as an event. So we select every line as an event break.



Source: **/home/splunk_data/new_categories.txt**

Sourcetype: System Defaults

Event Breaks

Every line is one event.

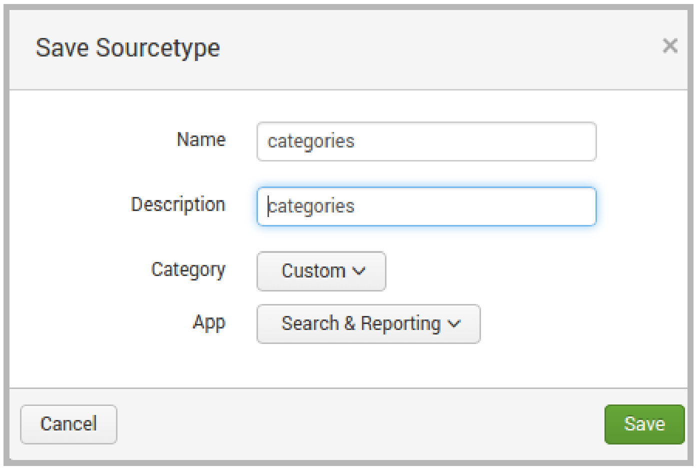
Break Type: Auto | **Every Line** | Regex...

Timestamp

Advanced

		Time	Event
1	⚠	11/16/14 2:47:10.000 PM	productId | pCategory | sCategory timestamp = none
2	⚠	11/16/14 2:47:10.000 PM	B0027DQHA0 | Movies & TV|Movies & TV, TV,Music, Classical timestamp = none
3	⚠	11/16/14 2:47:10.000 PM	0756400120 | Books|Books, Literature & Fiction, Anthologies & Litera tion, Anthologies,Books, Science Fiction & Fantasy, Science Fiction, timestamp = none
4	⚠	11/16/14 2:47:10.000 PM	B000012D5 | Music|Music, Blues,Music, Pop,Music, R&B timestamp = none

   b. Save the source type with an appropriate name by clicking on the “SaveAs” button.



Save Sourcetype

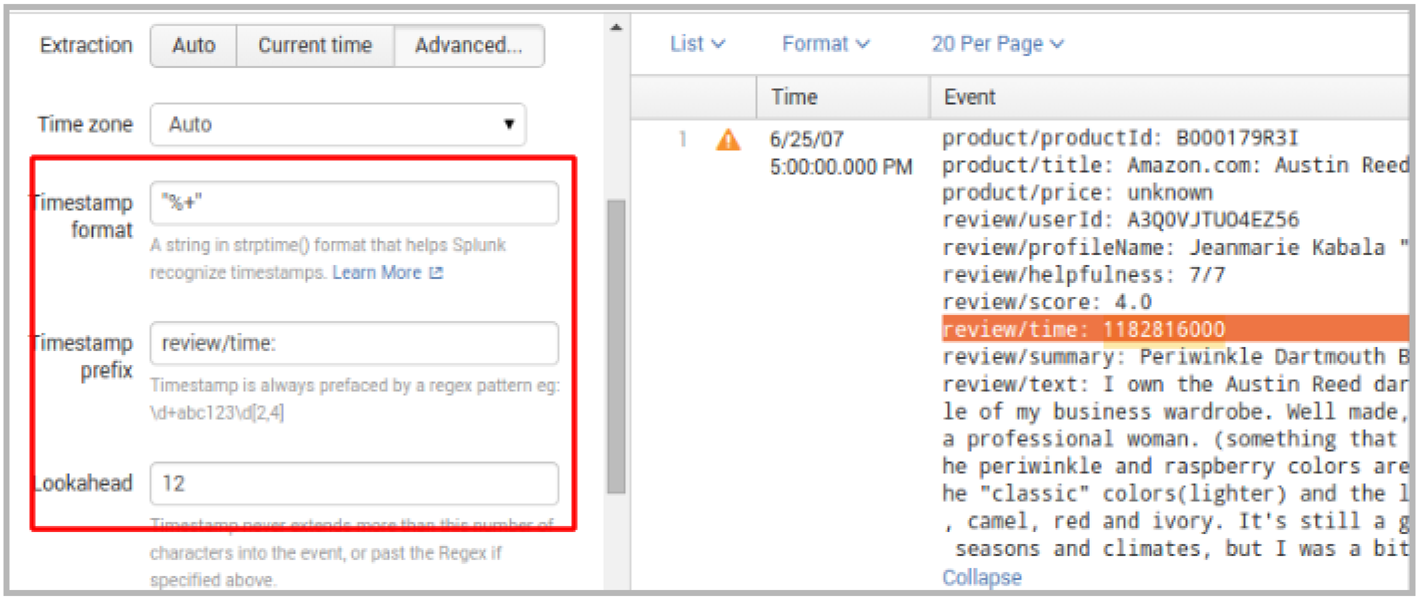
Name: categories

Description: categories

Category: Custom

App: Search & Reporting

c. For the main all_review file we used the timestamp of the review to index timestamp into splunk. Indexing the file over the the review time will help us easily narrow down queries to a particular time frame. To extract the timestamp from the data we must provide the timestamp prefix and timestamp format. We can see in the image below where the parameters must be configured.



Extraction: Advanced...

Time zone: Auto

Timestamp format: "%+"
A string in strptime() format that helps Splunk recognize timestamps. Learn More

Timestamp prefix: review/time:
Timestamp is always prefaced by a regex pattern eg: \d+abc123\d[2,4]

Lookahead: 12
Timestamp never extends more than this number of characters into the event, or past the Regex if specified above.

	Time	Event
1 ⚠	6/25/07 5:00:00.000 PM	product/productId: B000179R3I product/title: Amazon.com: Austin Reed product/price: unknown review/userId: A3Q0VJTUO4EZ56 review/profileName: Jeanmarie Kabala " review/helpfulness: 7/7 review/score: 4.0 review/time: 1182816000 review/summary: Periwinkle Dartmouth B review/text: I own the Austin Reed dar le of my business wardrobe. Well made, a professional woman. (something that he periwinkle and raspberry colors are he "classic" colors(lighter) and the l , camel, red and ivory. It's still a g seasons and climates, but I was a bit

For the all_review file we also had to configure the event break. We selected the **product/productId** as the event break as every review must begin with the “product/productId”.

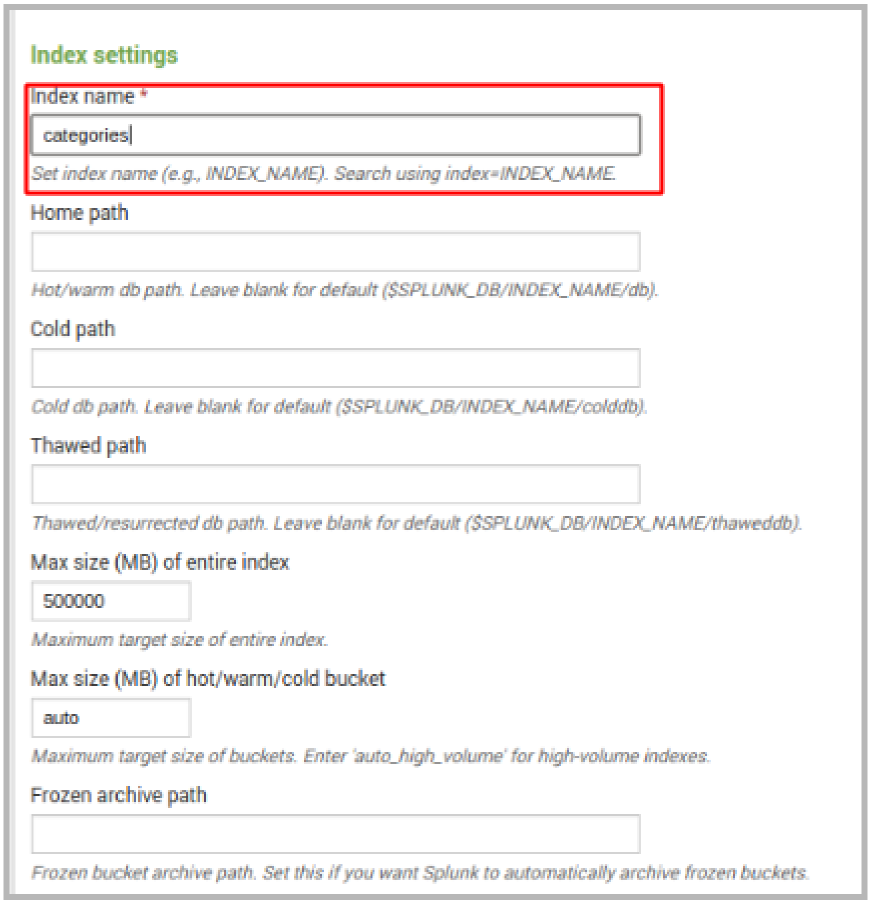
The configuration details for the all_review file are:

Parameter Name	Value	Interpretation
Timestamp Prefix:	**’%+’**	strptime uses this format to recognize unix timestamp
Timestamp format:	**review/time:**	In the data the timestamp is followed by this prefix
Lookahead:	**12**	Field indicates the number of character after prefix that can match the timestamp
Event break	**product/productId:.***	Every review must begin with this title
Max_days_ago	**-1**	Consider all possible dates back in time

3. Upload/Index the data

Once event breaks and timestamps are finalized. We need to decide weather to create a separate index for each data file or weather to upload everything in the main index. Creating separate index improves the speed of queries and makes it easy to manage data. So we decided to create 2 new indexes for the two major files all_reviews and categories. We indexed the other files in the main index. The following images describe the steps to create a index for the categories file.

a. First we create a new index.



Index settings

Index name *

categories

Set index name (e.g., INDEX_NAME). Search using index=INDEX_NAME.

Home path

Hot/warm db path. Leave blank for default ($SPLUNK_DB/INDEX_NAME/db).

Cold path

Cold db path. Leave blank for default ($SPLUNK_DB/INDEX_NAME/colddb).

Thawed path

Thawed/resurrected db path. Leave blank for default ($SPLUNK_DB/INDEX_NAME/thaweddb).

Max size (MB) of entire index

500000

Maximum target size of entire index.

Max size (MB) of hot/warm/cold bucket

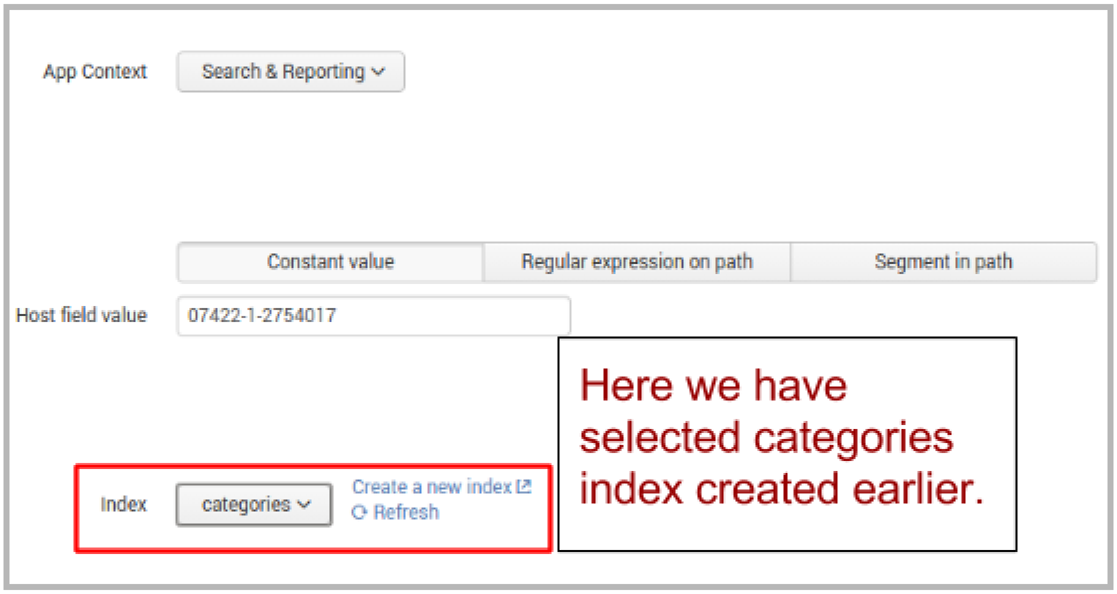
auto

Maximum target size of buckets. Enter 'auto_high_volume' for high-volume indexes.

Frozen archive path

Frozen bucket archive path. Set this if you want Splunk to automatically archive frozen buckets.

b. Then select the index while finalising the indexing option for file upload.



App Context: Search & Reporting

Host field value: 07422-1-2754017

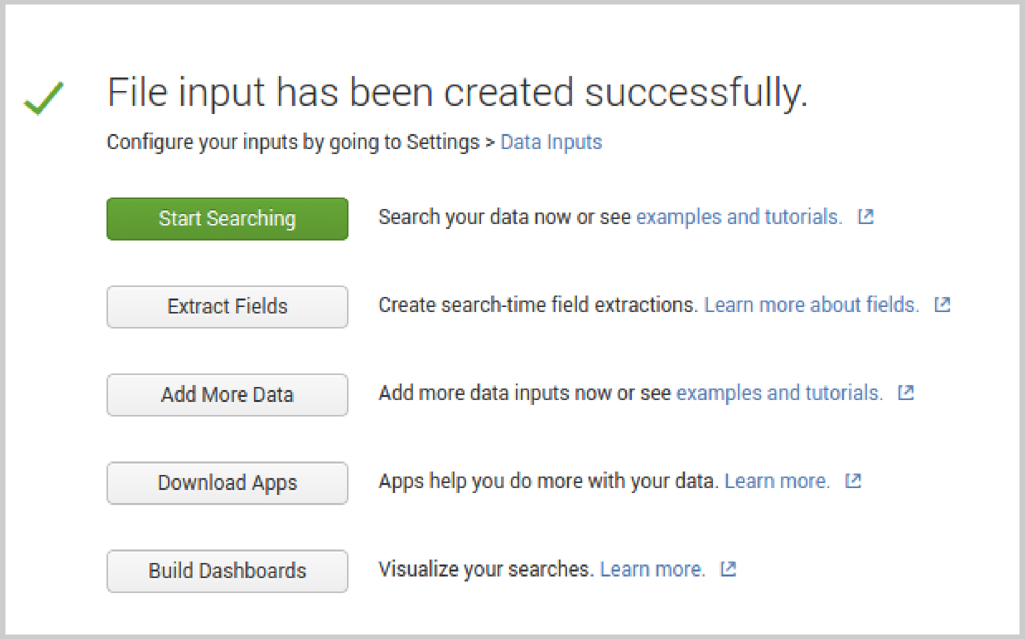
Index: categories

Here we have selected categories index created earlier.

Once uploaded splunk will index the file and add it to the index. The following table describes the file name and the corresponding index.

Filename	Index
all_reviews.txt	all_review
new_categories.txt	categories
brands.txt	main index
related.txt	main index

c. After the process you will the screen below.



File input has been created successfully.

Configure your inputs by going to Settings > Data Inputs

Search your data now or see examples and tutorials.

Create search-time field extractions. Learn more about fields.

Add more data inputs now or see examples and tutorials.

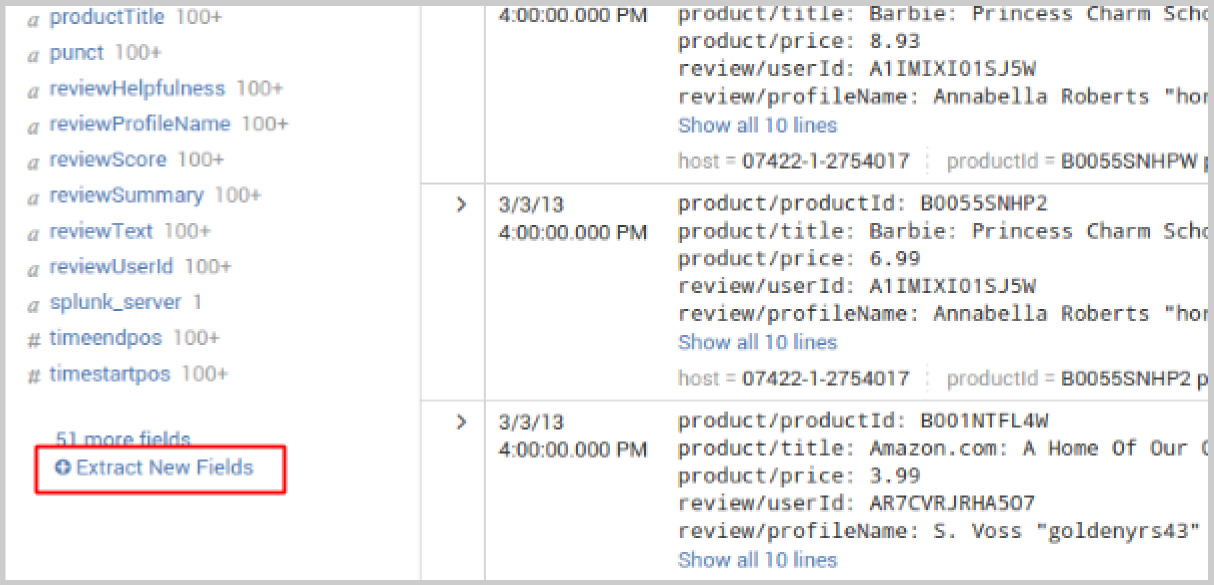
Apps help you do more with your data. Learn more.

Visualize your searches. Learn more.

2.2 Adding Field extractions

Since the data is in a text file with no standard format. We must add field extractions for the attribute names. We will first see how a field extraction is added and then will the field extractions added for the project.

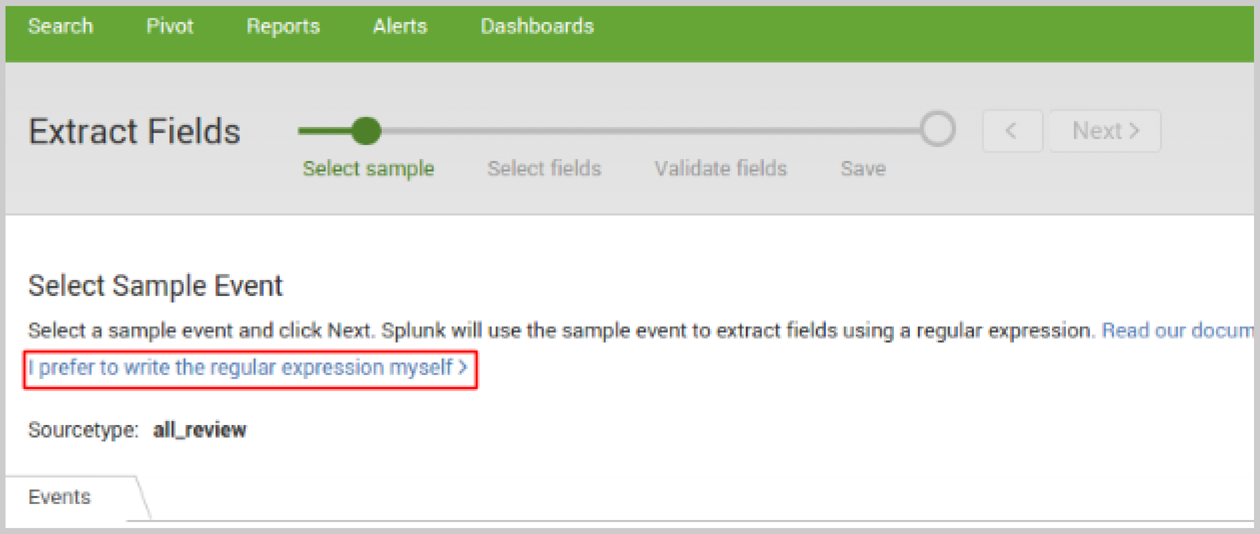
1. Select the sourcetype you want to add the field extraction for.
2. Then select Extract new fields as shown below.



```
a productTitle 100+
a punct 100+
a reviewHelpfulness 100+
a reviewProfileName 100+
a reviewScore 100+
a reviewSummary 100+
a reviewText 100+
a reviewUserId 100+
a splunk_server 1
# timeendpos 100+
# timestartpos 100+

51 more fields
Extract New Fields
```

```
4:00:00.000 PM   product/title: Barbie: Princess Charm Sch
                 product/price: 8.93
                 review/userId: A1IMIXI01SJ5W
                 review/profileName: Annabella Roberts "ho
                 Show all 10 lines
                 host = 07422-1-2754017   productId = B0055SNHPW

3/3/13           product/productId: B0055SNHP2
4:00:00.000 PM   product/title: Barbie: Princess Charm Sch
                 product/price: 6.99
                 review/userId: A1IMIXI01SJ5W
                 review/profileName: Annabella Roberts "ho
                 Show all 10 lines
                 host = 07422-1-2754017   productId = B0055SNHP2 p

3/3/13           product/productId: B001NTFL4W
4:00:00.000 PM   product/title: Amazon.com: A Home Of Our
                 product/price: 3.99
                 review/userId: AR7CVRJRHA5O7
                 review/profileName: S. Voss "goldenyrs43"
                 Show all 10 lines
```

3. Select the write the regular expression option as shown below



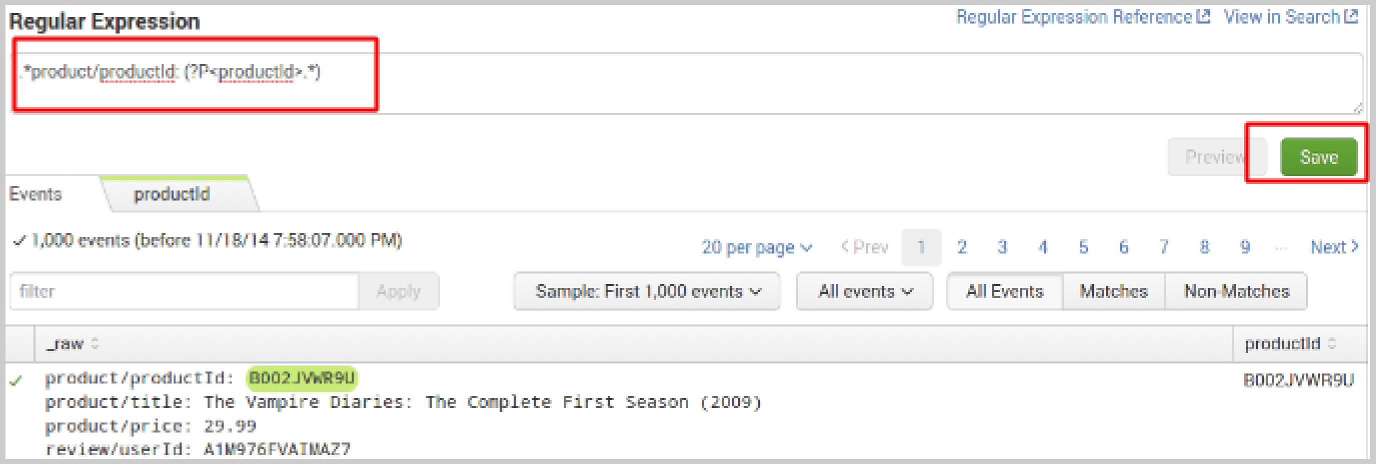
Select Sample Event

Select a sample event and click Next. Splunk will use the sample event to extract fields using a regular expression. Read our docum

I prefer to write the regular expression myself >

Sourcetype: **all_review**

4. Write the regular expression for your field and click preview to verify the extraction. Once verified save the extraction.



```
.*product/productId: (?P<productId>.*)
```

✓ 1,000 events (before 11/18/14 7:58:07.000 PM)

```
✓ product/productId: B002JVWR9U                                          B002JVWR9U
  product/title: The Vampire Diaries: The Complete First Season (2009)
  product/price: 29.99
  review/userId: A1M976FVAIMAZ7
```

Using the above mentioned process we have added field extraction for the following fields.

Interpretation for the regular expression : **.*product/productId: (?P<productId>[^\n]+)**

Any character followed by **product/productId:** followed one or more occurrence of a new line character will be named as **productId.**

List of field-extractions

Table Name:Field Name	**Regular Expression**
all_review:**productId**	.*product/productId: (?P<productId>[^\n]+)
all_review:**productPrice**	.*product/price: (?P<productPrice>[^\n]+)
all_review:**reviewUserId**	.*review/userId: (?P<reviewUserId>[^\n]+)
all_review**:reviewScore**	.*review/score: (?P<reviewScore>[^\n]+)
all_review:**reviewSummary**	.*review/summary: (?P<reviewSummary>[^\n]+)
all_review**:reviewText**	.*review/text: (?P<reviewText>[^\n]+)
all_review:**reviewHelpfulness**	.*review/helpfulness: (?P<reviewHelpfulness>[^\n]+)
all_review:**productTitle**	.*product/title: (?P<productTitle>[^\n]+)
all_review:**reviewProfileName**	.*review/profileName: (?P<reviewProfileName>[^\n]+)
categories:**productId**	^(?P<productId>\w+)
categories:**pCategory**	^[^|\n]*|(?P<pCategory>\s+\w+)
categories:**sCategory**	|+ (?P<sCategory>.*)
related:**productId**	^(?P<productId>\w+)\s+\w+\s+\w+\s+(?P<coPurchase>.+)
related:**coPurchased**	^(?P<productId>\w+)\s+\w+\s+\w+\s+(?P<coPurchase>.+)
brands:**productId**	^(?P<productId>\w+)
brands:**brandName**	^[^ \n]* (?P<brandName>.*)

3. Verification of Loaded Data

3.1 Data Pre-Processing

The categories file needed to be pre processed. The unprocessed format of the categories file is as follows.

```
B0027DQHA0  -> productId
  Movies & TV, TV          | ->category
  Music, Classical
0756400120
  Books, Literature & Fiction, Anthologies & Literary Collections, General
  Books, Literature & Fiction, United States
  Books, Science Fiction & Fantasy, Science Fiction, Anthologies
  Books, Science Fiction & Fantasy, Science Fiction, Short Stories
B0000012D5
  Music, Blues
  Music, Pop
  Music, R&B
B00024YAOQ
  Books, Business & Investing, Business Life, Motivation & Self-Improvement
```

Separating and pointing out the fields(categories) for the file was a tough task in splunk. So we decided to convert the file into the new format. The new format extracts the primary category and secondary category for each product.

B0027DQHA0 **| Movies & TV|** Movies & TV, TV,Music, Classical

0756400120 **| Books |**Books, Literature & Fiction, Anthologies & Literary Collections, General,Books, Literature & Fiction, United States,Books, Science Fiction & Fantasy, Science Fiction, Anthologies,Books, Science Fiction & Fantasy, Science Fiction, Short Stories

B0000012D5 **| Music |**Music, Blues,Music, Pop,Music, R&B

B00024YAOQ **| Books |**Books, Business & Investing, Business Life, Motivation & Self-Improvement

Here we separated three new fields with pipe.

productId | pCategory | sCategory

Now we can use primary category(pCategory) for building queries based on categories.

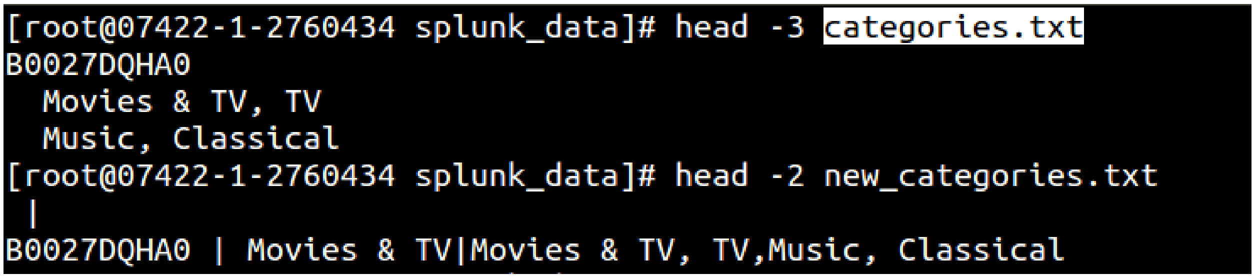
We wrote a script to perform the transformation.

```
#!/bin/sh
categories=""
while read line
do
 `echo $line | grep -q '[a-z]'` > /dev/null
 cond1=$?
 count=`echo $line | wc -c ` > /dev/null

 if [ "$cond1" -eq "1" -a "$count" -eq "11" ]; then
         echo "$product_id | $categories" >> new_categories.txt
         categories=""
         product_id="$line"
 else
         if [ -z "$categories" ]; then
                 pcat=`echo $line | awk -F "," '{print $1}'`
                 categories="$pcat|$line"
         else
                 categories="$categories,$line"
         fi
 fi
done < categories.txt
```

3.2 Verification of Data Preprocessing

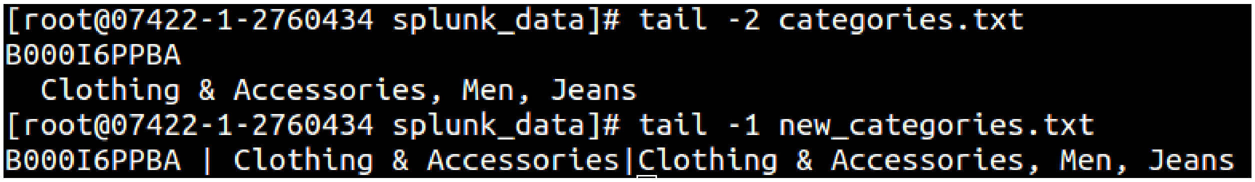
The verification of the ETL based on 3 checks. The two files in the ETL process are as follows.
categories.txt is the old file
new_categories.txt is the new file

1. We check the head on the old and the new files. We see that the first entry in both the files is same.



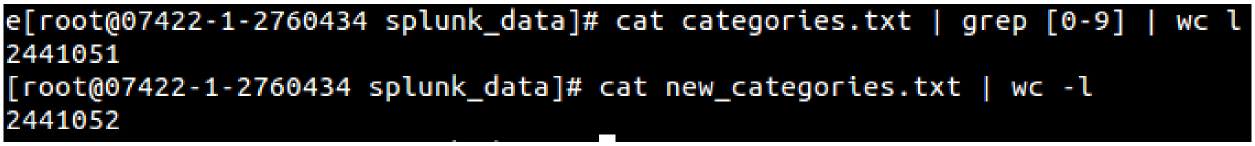
```
[root@07422-1-2760434 splunk_data]# head -3 categories.txt
B0027DQHA0
  Movies & TV, TV
  Music, Classical
[root@07422-1-2760434 splunk_data]# head -2 new_categories.txt
 |
B0027DQHA0 | Movies & TV|Movies & TV, TV,Music, Classical
```

2. The second check is based on last entry in the file. We see below that the last entry in both the files is the same.



```
[root@07422-1-2760434 splunk_data]# tail -2 categories.txt
B000I6PPBA
  Clothing & Accessories, Men, Jeans
[root@07422-1-2760434 splunk_data]# tail -1 new_categories.txt
B000I6PPBA | Clothing & Accessories|Clothing & Accessories, Men, Jeans
```

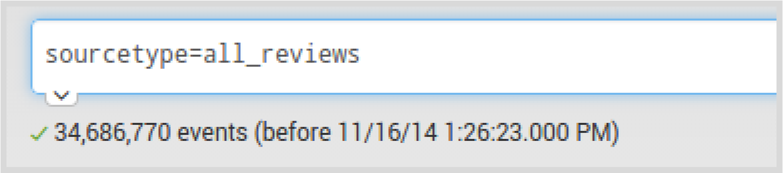
3.The third check is based on the number of products in each file i.e the count of entries in each file. Below we see that the number of valid entries in each file is the almost same.
One difference can be attributed to column name headers manually added in the new file.



```
e[root@07422-1-2760434 splunk_data]# cat categories.txt | grep [0-9] | wc l
2441051
[root@07422-1-2760434 splunk_data]# cat new_categories.txt | wc -l
2441052
```

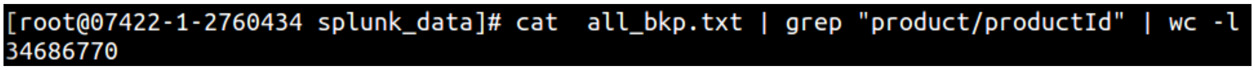
3.3 Verification of count of events

In this section we verify all the tables and their data.
all_reviews.txt is the file that contains all the reviews on amazon. One event in splunk corresponds to 10 lines in the file. We will now find the number of events in all_reviews in splunk .



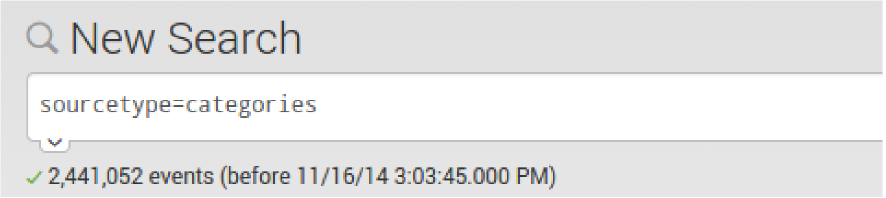
```
sourcetype=all_reviews
✓ 34,686,770 events (before 11/16/14 1:26:23.000 PM)
```

In image we observe the total events in splunk for the file are 34,686,770.
As each event has productId for sure. We can count the occurrence of productId in the all_reviews.txt.
We see that the occurrence of productId is same as the number of events in all_reviews in splunk .



```
[root@07422-1-2760434 splunk_data]# cat  all_bkp.txt | grep "product/productId" | wc -l
34686770
```

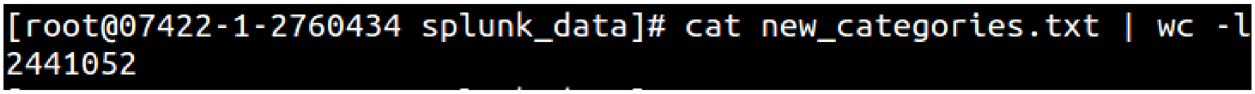
Now we verify the categories.txt. Field contains the productId and its categories.



```
sourcetype=categories
```

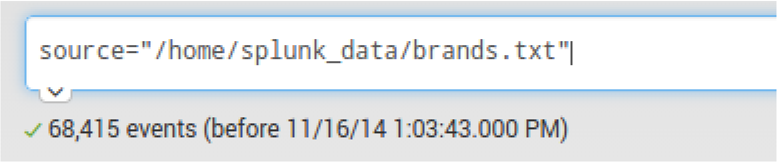
✓ 2,441,052 events (before 11/16/14 3:03:45.000 PM)

In the image above we observe the total events for the file are 2,441,052. As in the new_categories file each line corresponds to one event in splunk. We simply count the number of lines in the new files.



```
[root@07422-1-2760434 splunk_data]# cat new_categories.txt | wc -l
2441052
```

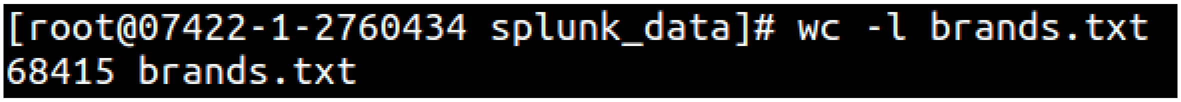
For the table brands.txt the total events that splunk could find are 68415.



```
source="/home/splunk_data/brands.txt"
```

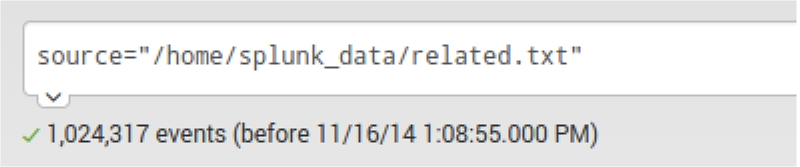
✓ 68,415 events (before 11/16/14 1:03:43.000 PM)

As each line is an event in the file. The number of lines gives us the count of events in the file.
i.e is 68415 which is same as the number of events splunk found.



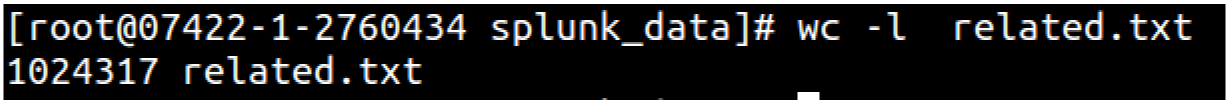
```
[root@07422-1-2760434 splunk_data]# wc -l brands.txt
68415 brands.txt
```

In the related.txt file which has all the coPurchased information for products. Here again each line accounts to one event. Splunk the count of events is 1,024,317 which is similar to what we found using the line count in linux.



```
source="/home/splunk_data/related.txt"
```

✓ 1,024,317 events (before 11/16/14 1:08:55.000 PM)



```
[root@07422-1-2760434 splunk_data]# wc -l  related.txt
1024317 related.txt
```

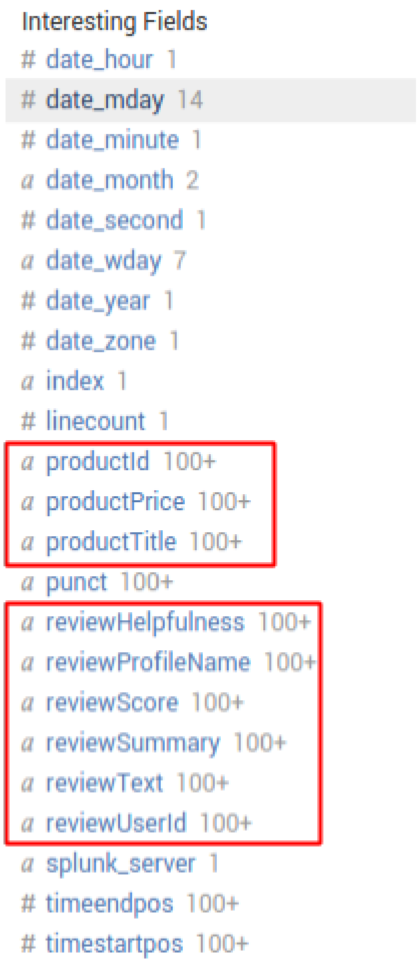
Now we have verified all the files with respect to the count.

3.4 Verification of fields for uploaded data

1. For the all review table, review/time is used as the timestamp for indexing. The other 9 fields were loaded into splunk.

product/productId: B000179R3I
product/title: Amazon.com: Austin Reed Dartmouth Jacket In Basics, Misses: Clothing
product/price: unknown
review/userId: A3Q0VJTUO4EZ56
review/profileName: Jeanmarie Kabala "JP Kabala"
review/helpfulness: 7/7
review/score: 4.0
review/time: 1182816000 --> used to indexing
review/summary: Periwinkle Dartmouth Blazer
review/text: I own the Austin Reed dartmouth blazer in every color in which they make it-- it is a staple of my business wardrobe. Well made, quality fabric, nicely tailored, classic lines, appropriate for a professional woman. (something that can be hard to find at times) It should be noted, however, that the periwinkle and raspberry colors are lovely, but the fabric and buttons are slightly different than the "classic" colors(lighter) and the linings and interfacings are not as substantial as the brown, navy, camel, red and ivory. It's still a good value, particularly as these are colors appropriate to warmer seasons and climates, but I was a bit surprised.

It can be observed that Interesting fields contain all 9 fields contained in all review table marked by red lines. Remaining one field ie review/time is used for indexing.



Interesting Fields

date_hour 1

date_mday 14

date_minute 1

a date_month 2

date_second 1

a date_wday 7

date_year 1

date_zone 1

a index 1

linecount 1

a productId 100+

a productPrice 100+

a productTitle 100+

a punct 100+

a reviewHelpfulness 100+

a reviewProfileName 100+

a reviewScore 100+

a reviewSummary 100+

a reviewText 100+

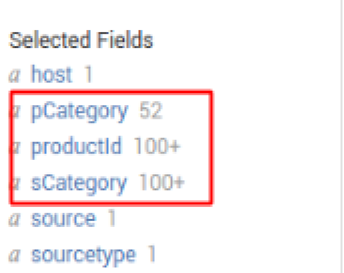
a reviewUserId 100+

a splunk_server 1

timeendpos 100+

timestartpos 100+

2. For Categories tables, after executing a search on splunk user interface, we can observe 3 entries appears in selected fields marked by red lines that tallies with fields in entries in our dataset.



Selected Fields

a host 1

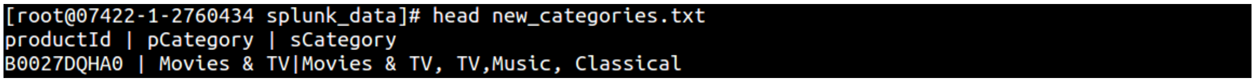
a pCategory 52

a productId 100+

a sCategory 100+

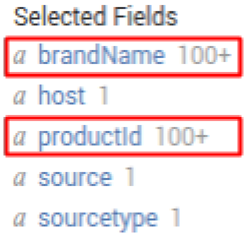
a source 1

a sourcetype 1

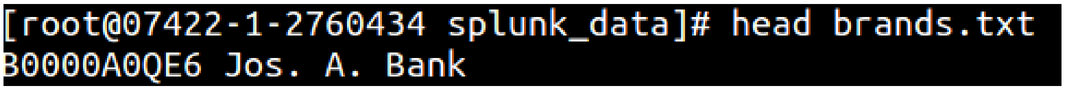


```
[root@07422-1-2760434 splunk_data]# head new_categories.txt
productId | pCategory | sCategory
B0027DQHA0 | Movies & TV|Movies & TV, TV,Music, Classical
```

3. For Brand table, on splunk user interface we can observe 2 column names appear in selected fields marked by red lines that tallies with fields in entries in our dataset.

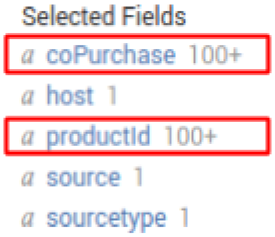


Selected Fields
a brandName 100+
a host 1
a productId 100+
a source 1
a sourcetype 1

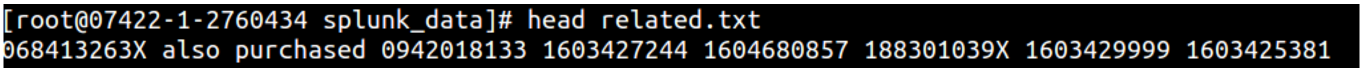


```
[root@07422-1-2760434 splunk_data]# head brands.txt
B0000A0QE6 Jos. A. Bank
```

4. For related table on splunk user interface, we can observe 2 column names appear in selected fields marked by red lines that tallies with fields in entries in our dataset.



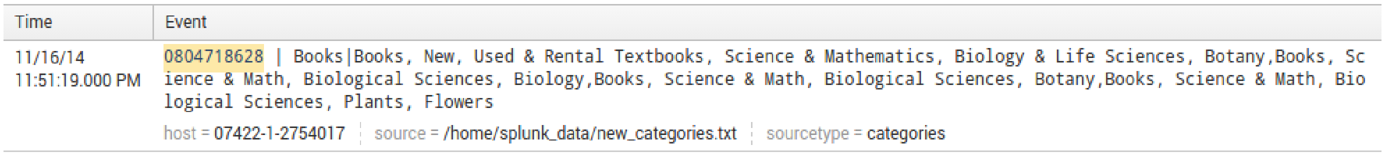
Selected Fields
a coPurchase 100+
a host 1
a productId 100+
a source 1
a sourcetype 1



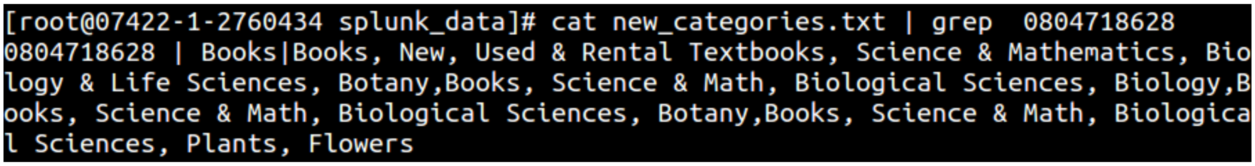
```
[root@07422-1-2760434 splunk_data]# head related.txt
068413263X also purchased 0942018133 1603427244 1604680857 188301039X 1603429999 1603425381
```

3.5 Verifying the data

1. To verify the integrity of the data indexed by Splunk, we select an entry randomly in the new_categories file how it is uploaded in splunk. In the images below we see that is exactly one entry for productId=**0804718628.**
   Query = **sourcetype=categories productId="0804718628"**



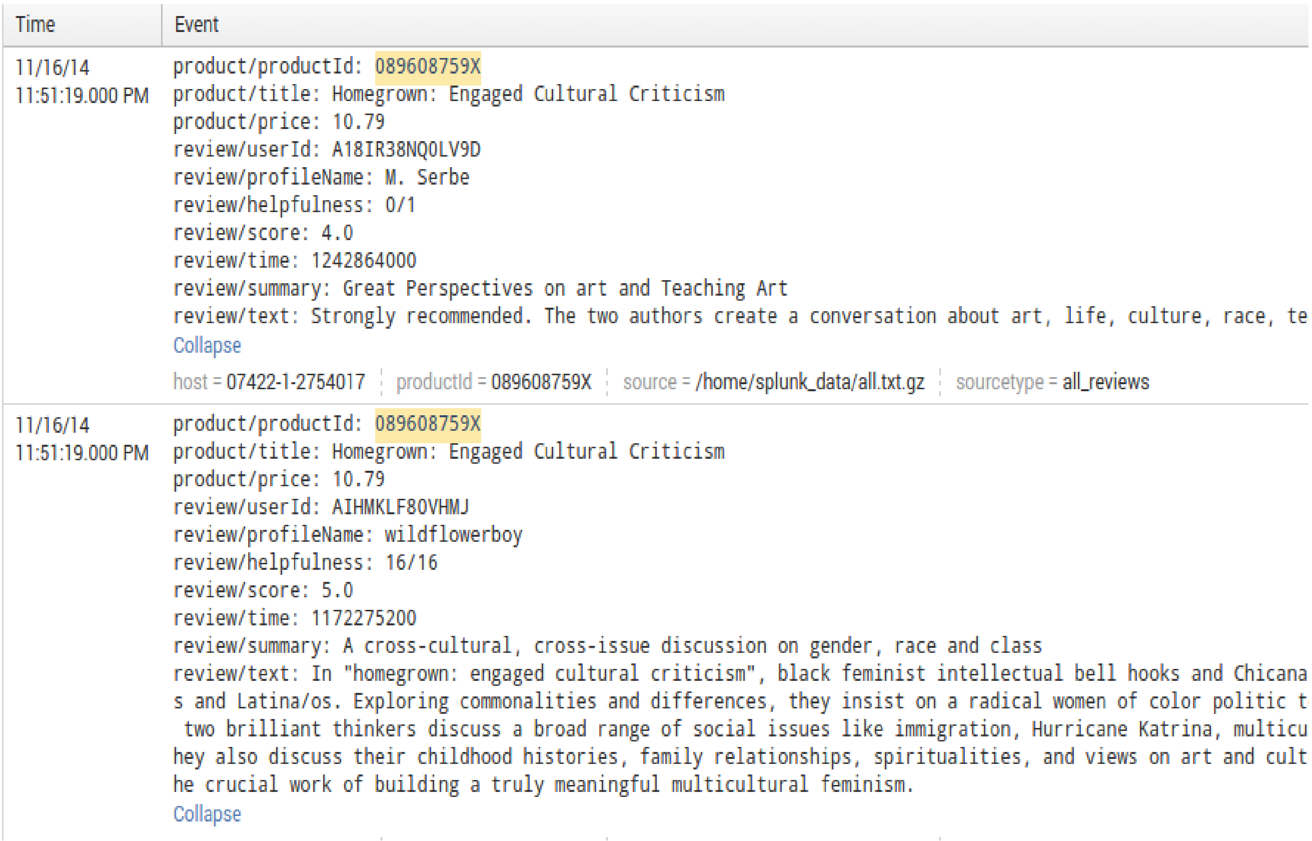
Time	Event
11/16/14 11:51:19.000 PM	0804718628 | Books|Books, New, Used & Rental Textbooks, Science & Mathematics, Biology & Life Sciences, Botany,Books, Science & Math, Biological Sciences, Biology,Books, Science & Math, Biological Sciences, Botany,Books, Science & Math, Biological Sciences, Plants, Flowers host = 07422-1-2754017 source = /home/splunk_data/new_categories.txt sourcetype = categories



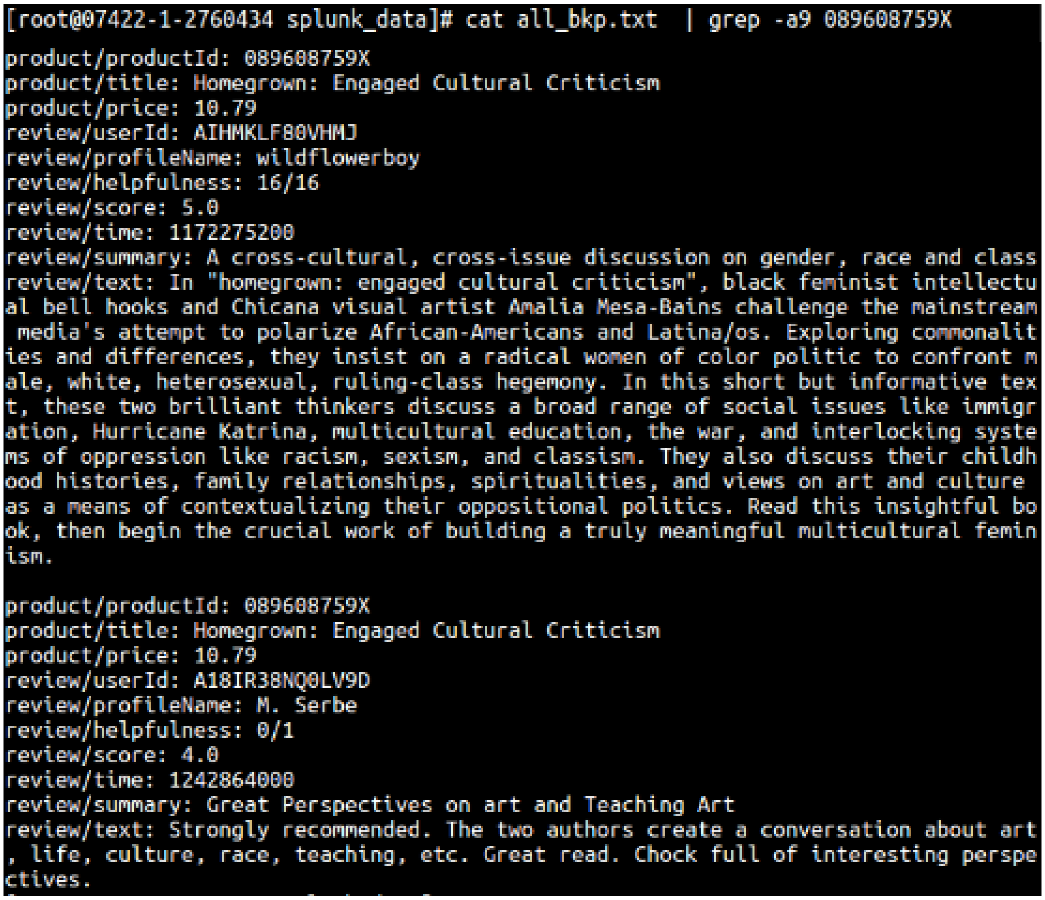
```
[root@07422-1-2760434 splunk_data]# cat new_categories.txt | grep  0804718628
0804718628 | Books|Books, New, Used & Rental Textbooks, Science & Mathematics, Bio
logy & Life Sciences, Botany,Books, Science & Math, Biological Sciences, Biology,B
ooks, Science & Math, Biological Sciences, Botany,Books, Science & Math, Biologica
l Sciences, Plants, Flowers
```

2. To verify all_reviews file. Again we will take some random product and check the data in both the text file and splunk. Here we have selected the product with productId = 089608759X. We see that there are two reviews for the product. We have found the similar reviews in both splunk and text file.

Query = **sourcetype=all_review productId="089608759X"**



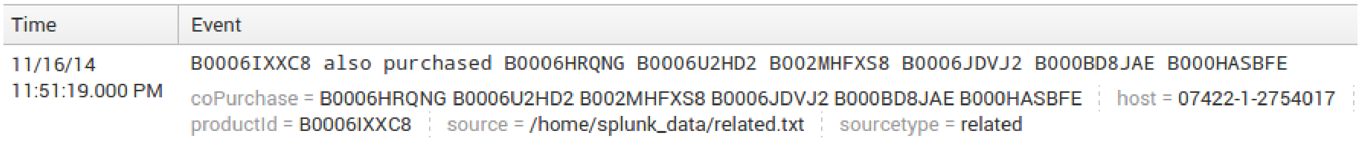
Time	Event
11/16/14 11:51:19.000 PM	product/productId: 089608759X product/title: Homegrown: Engaged Cultural Criticism product/price: 10.79 review/userId: A18IR38NQ0LV9D review/profileName: M. Serbe review/helpfulness: 0/1 review/score: 4.0 review/time: 1242864000 review/summary: Great Perspectives on art and Teaching Art review/text: Strongly recommended. The two authors create a conversation about art, life, culture, race, te Collapse host = 07422-1-2754017 ¦ productId = 089608759X ¦ source = /home/splunk_data/all.txt.gz ¦ sourcetype = all_reviews
11/16/14 11:51:19.000 PM	product/productId: 089608759X product/title: Homegrown: Engaged Cultural Criticism product/price: 10.79 review/userId: AIHMKLF80VHMJ review/profileName: wildflowerboy review/helpfulness: 16/16 review/score: 5.0 review/time: 1172275200 review/summary: A cross-cultural, cross-issue discussion on gender, race and class review/text: In "homegrown: engaged cultural criticism", black feminist intellectual bell hooks and Chicana s and Latina/os. Exploring commonalities and differences, they insist on a radical women of color politic t two brilliant thinkers discuss a broad range of social issues like immigration, Hurricane Katrina, multicu hey also discuss their childhood histories, family relationships, spiritualities, and views on art and cult he crucial work of building a truly meaningful multicultural feminism. Collapse



```
[root@07422-1-2760434 splunk_data]# cat all_bkp.txt  | grep -a9 089608759X
product/productId: 089608759X
product/title: Homegrown: Engaged Cultural Criticism
product/price: 10.79
review/userId: AIHMKLF80VHMJ
review/profileName: wildflowerboy
review/helpfulness: 16/16
review/score: 5.0
review/time: 1172275200
review/summary: A cross-cultural, cross-issue discussion on gender, race and class
review/text: In "homegrown: engaged cultural criticism", black feminist intellectu
al bell hooks and Chicana visual artist Amalia Mesa-Bains challenge the mainstream
 media's attempt to polarize African-Americans and Latina/os. Exploring commonalit
ies and differences, they insist on a radical women of color politic to confront m
ale, white, heterosexual, ruling-class hegemony. In this short but informative tex
t, these two brilliant thinkers discuss a broad range of social issues like immigr
ation, Hurricane Katrina, multicultural education, the war, and interlocking syste
ms of oppression like racism, sexism, and classism. They also discuss their childh
ood histories, family relationships, spiritualities, and views on art and culture
as a means of contextualizing their oppositional politics. Read this insightful bo
ok, then begin the crucial work of building a truly meaningful multicultural femin
ism.

product/productId: 089608759X
product/title: Homegrown: Engaged Cultural Criticism
product/price: 10.79
review/userId: A18IR38NQ0LV9D
review/profileName: M. Serbe
review/helpfulness: 0/1
review/score: 4.0
review/time: 1242864000
review/summary: Great Perspectives on art and Teaching Art
review/text: Strongly recommended. The two authors create a conversation about art
, life, culture, race, teaching, etc. Great read. Chock full of interesting perspe
ctives.
```

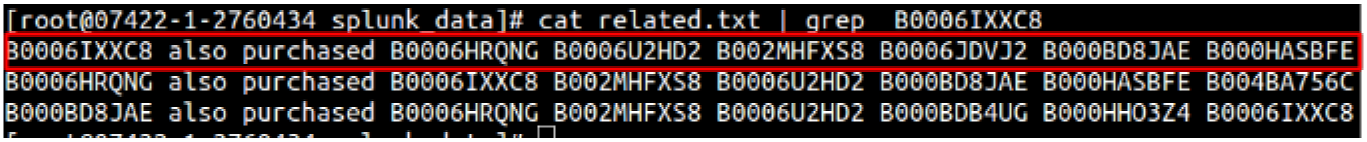
3. The related text file can verified again by select a random product and check the co purchased product for the product. We observe that we find 3 entries in both text file and splunk.

   Query = **sourcetype=related productId="B0006IXXC8"**



Time	Event

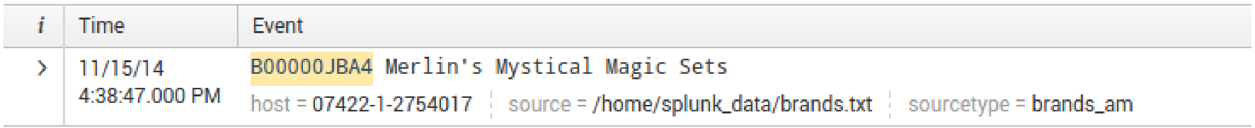
11/16/14
11:51:19.000 PM
	B0006IXXC8 also purchased B0006HRQNG B0006U2HD2 B002MHFXS8 B0006JDVJ2 B000BD8JAE B000HASBFE
	coPurchase = B0006HRQNG B0006U2HD2 B002MHFXS8 B0006JDVJ2 B000BD8JAE B000HASBFE ⋮ host = 07422-1-2754017 ⋮
	productId = B0006IXXC8 ⋮ source = /home/splunk_data/related.txt ⋮ sourcetype = related



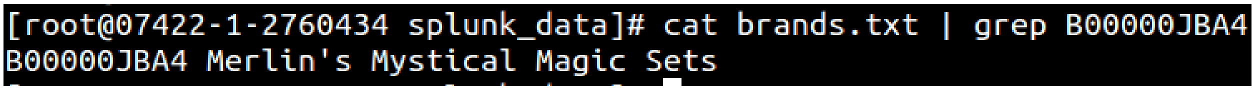
```
[root@07422-1-2760434 splunk_data]# cat related.txt | grep  B0006IXXC8
B0006IXXC8 also purchased B0006HRQNG B0006U2HD2 B002MHFXS8 B0006JDVJ2 B000BD8JAE B000HASBFE
B0006HRQNG also purchased B0006IXXC8 B002MHFXS8 B0006U2HD2 B000BD8JAE B000HASBFE B004BA756C
B000BD8JAE also purchased B0006HRQNG B002MHFXS8 B0006U2HD2 B000BDB4UG B000HHO3Z4 B0006IXXC8
```

4. For brands.txt we verify a random entry in both splunk and text file. We observe that both are same.

   Query : **sourcetype=brands productId="B00000JBA4"**



i	Time	Event
>	11/15/14 4:38:47.000 PM	B00000JBA4 Merlin's Mystical Magic Sets host = 07422-1-2754017 ┊ source = /home/splunk_data/brands.txt ┊ sourcetype = brands_am

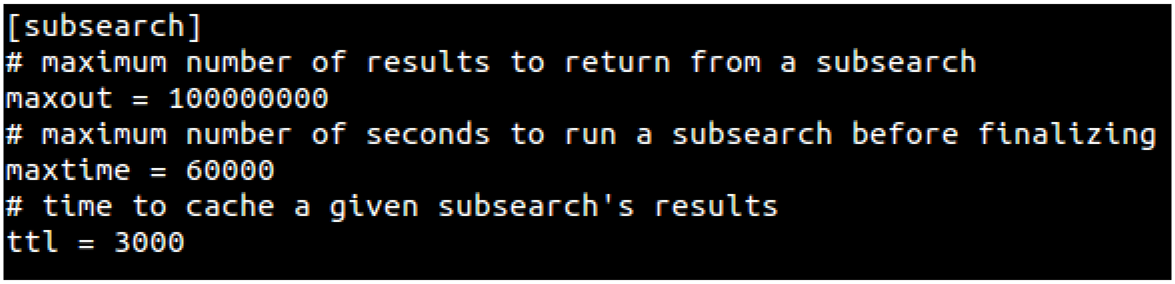


```
[root@07422-1-2760434 splunk_data]# cat brands.txt | grep B00000JBA4
B00000JBA4 Merlin's Mystical Magic Sets
```

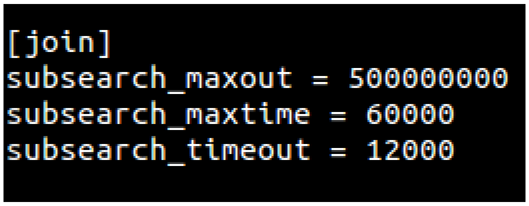
4. Building and Verifying Queries

The data spans over 16 years. But some queries are only evaluated over an year with the most reviews(i.e 2012). This is done to speed up the execution time and we believe one year gives us a idea of the information we are trying to extract.

Since the product categories and reviews are in separate tables. There are many queries that require a join and sub search operation. By default a subsearch timeout/maxout is low, the data/queries we have requires this values to be increased. So we made following changes in **./etc/system/default/limits.conf**

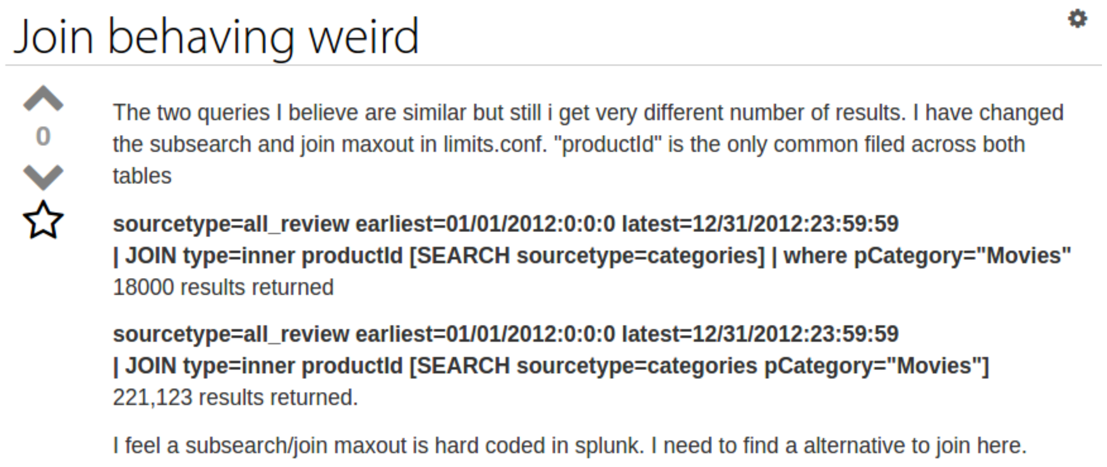


```
[subsearch]
# maximum number of results to return from a subsearch
maxout = 100000000
# maximum number of seconds to run a subsearch before finalizing
maxtime = 60000
# time to cache a given subsearch's results
ttl = 3000
```



```
[join]
subsearch_maxout = 500000000
subsearch_maxtime = 60000
subsearch_timeout = 12000
```

After making the changes in limits.conf we restarted splunk. After the change we noticed that warning for sub search maxout went. But still we found some inconsistencies in the results. So we posted a question on the splunk answers forum to better understand the issue.
http://answers.splunk.com/answers/202887/join-behaving-weird.html



Join behaving weird

The two queries I believe are similar but still i get very different number of results. I have changed the subsearch and join maxout in limits.conf. "productId" is the only common filed across both tables

sourcetype=all_review earliest=01/01/2012:0:0:0 latest=12/31/2012:23:59:59
| JOIN type=inner productId [SEARCH sourcetype=categories] | where pCategory="Movies"
18000 results returned

sourcetype=all_review earliest=01/01/2012:0:0:0 latest=12/31/2012:23:59:59
| JOIN type=inner productId [SEARCH sourcetype=categories pCategory="Movies"]
221,123 results returned.

I feel a subsearch/join maxout is hard coded in splunk. I need to find a alternative to join here.

From answers on the forum we assumed that splunk may have hardcoded a limit for sub search and join maxout. Hence we wrote queries that have a small result in sub search and join and observed the results to match. We did study methods to avoid a join like a lookup table and “sourcetype=X OR sourcetype=Y” but they did not meet the purpose. To our understanding today for some purposes JOIN is unavoidable.

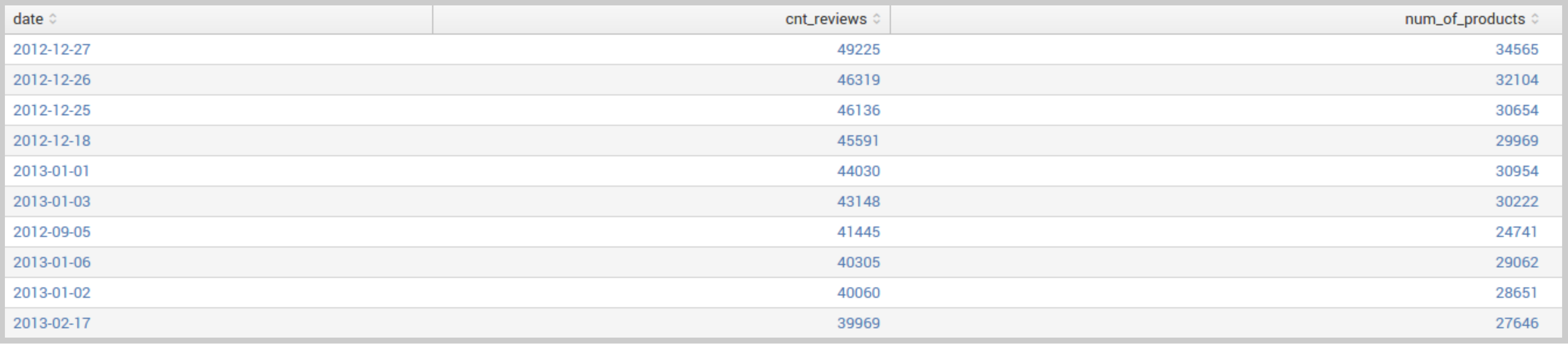
Most of our queries leverage the time indexing ability for splunk. Queires that analyze special period of the year like black friday week and holidays. These queries were found to be having fast response time.

4.1 Days with most reviews since 2012

Objective: To determine days on which products are reviewed the most since 2012. This gives us an rough idea of days when amazon sales the most products. Here we assume people review a product soon after they purchase it.

Description: Here we gathered all reviews and converted the timestamp for a review to a yyyy-mm-dd format to get the date. Then applied count on number of reviews by date. Finally sorted the data by count. We also counted the products reviewed to make the information more meaningful. We observed that most products of amazon are reviewed during the holidays. We also observe that Sep 05 has made it the list of top 10. This is most likely because Sep 03 was labour day.

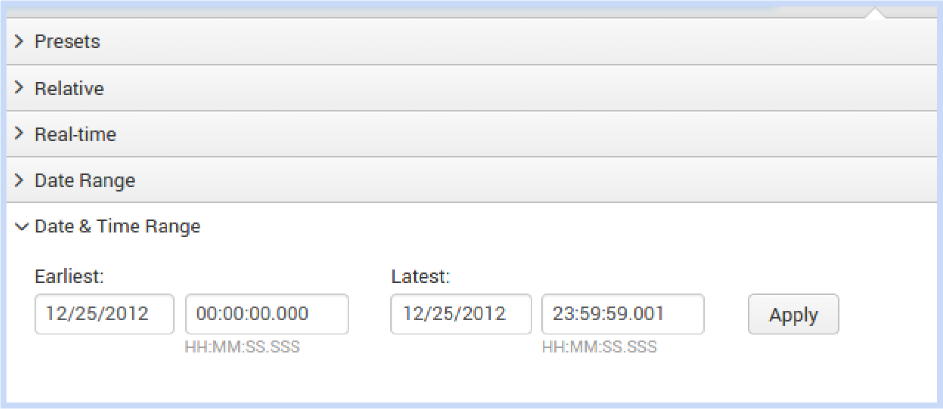
Query: **sourcetype=all_review earliest="01/01/2012:0:0:0"**
| convert timeformat="%Y-%m-%d" ctime(_time) AS date
| stats count as cnt_reviews,dc(productId) as num_of_products by date | sort -cnt_reviews | head 10



date	cnt_reviews	num_of_products
2012-12-27	49225	34565
2012-12-26	46319	32104
2012-12-25	46136	30654
2012-12-18	45591	29969
2013-01-01	44030	30954
2013-01-03	43148	30222
2012-09-05	41445	24741
2013-01-06	40305	29062
2013-01-02	40060	28651
2013-02-17	39969	27646

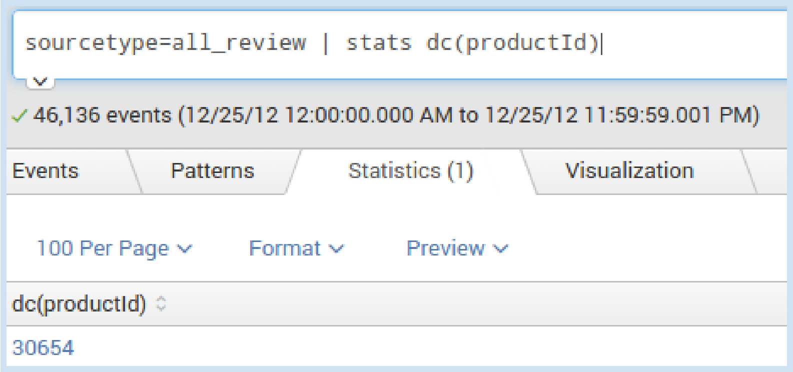
Verification: For verification, in the timepicker we selected the date as 25-Dec-2012. The total events observed were 46,136 and value of dc(productId) came out to be 30654. These values matched with the output of main query.

sourcetype=all_review | stats dc(productId)



Earliest: 12/25/2012 00:00:00.000

Latest: 12/25/2012 23:59:59.001



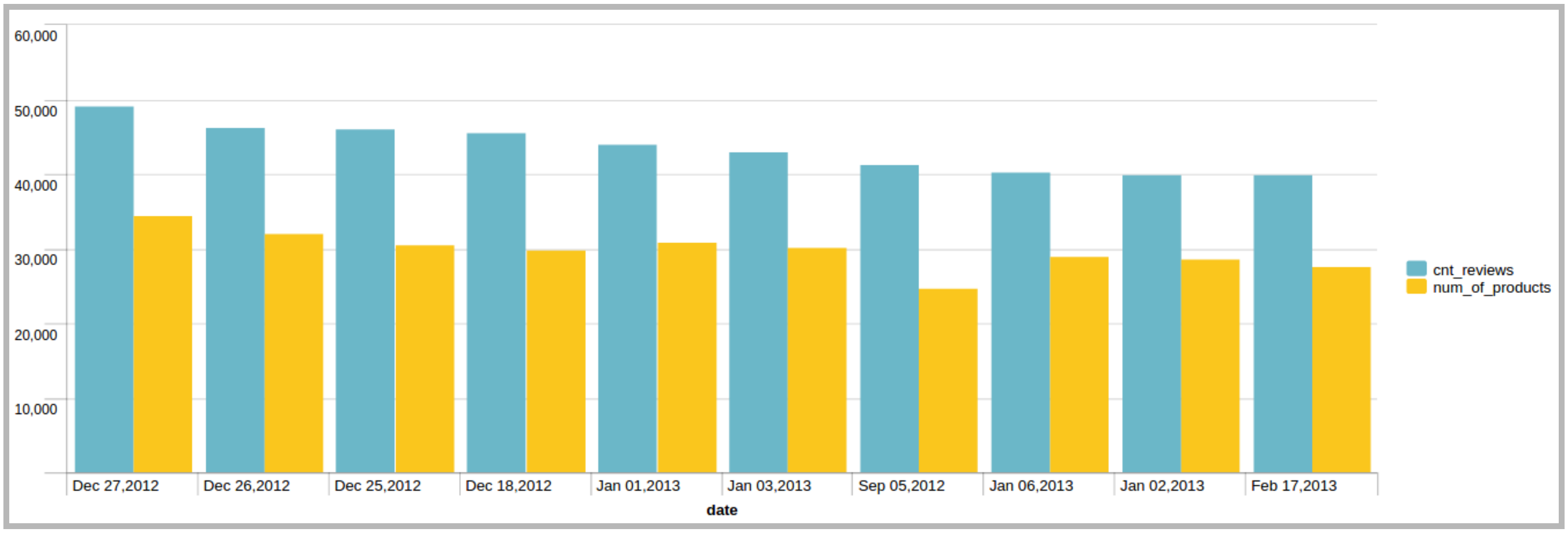
```
sourcetype=all_review | stats dc(productId)
```

✓ 46,136 events (12/25/12 12:00:00.000 AM to 12/25/12 11:59:59.001 PM)

dc(productId)
30654

Visualization: We reformatted the ctime date string to a more readable date.

Days with most reviews



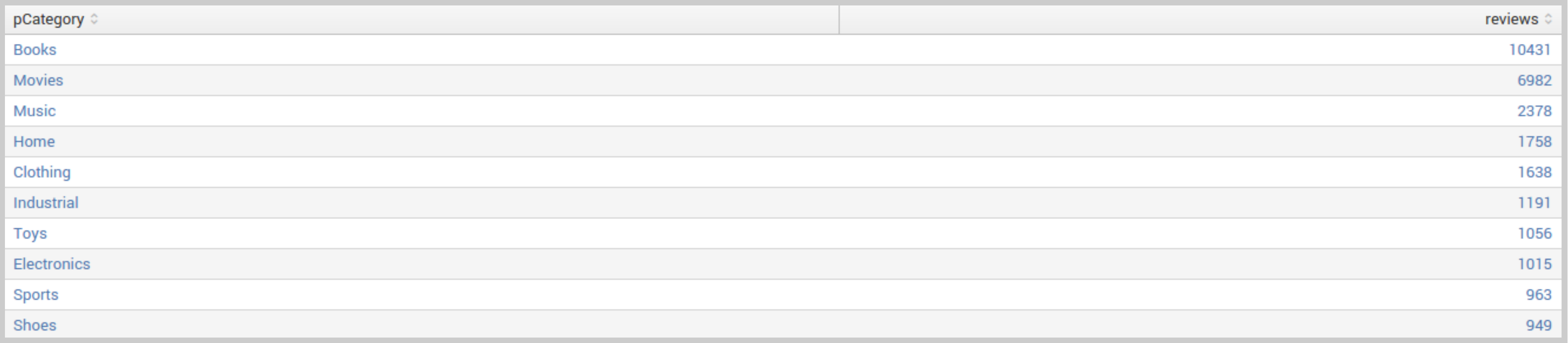
4.2 Categories for product on the day with most reviews.

Objective: Aim to find the categories reviewed on the day with the most reviews.

Description: We have to find which is most reviewed categories on 27-Dec-2012. This is the day when the most reviews were recorded. We first find all reviews for that day in a subsearch and then join with the categories table to find categories reviewed for that day. We observe that apart from book,movies and music which are always a part of the most reviewed products. toys make it to the top 10 list showing toys is popular product category during the holidays.

Query :

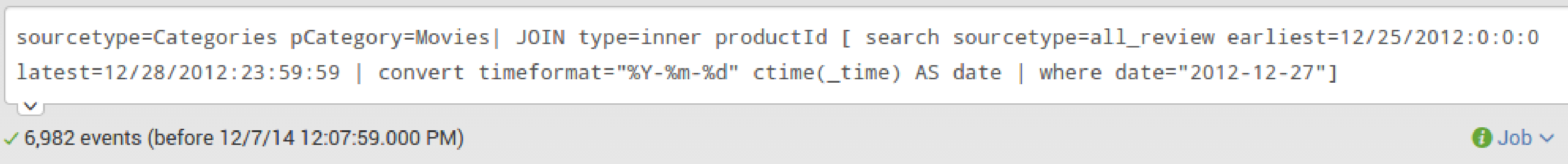
```
sourcetype=Categories
| JOIN type=inner productId [ search sourcetype=all_review earliest=12/25/2012:0:0:0
latest=12/28/2012:23:59:59 | convert timeformat="%Y-%m-%d" ctime(_time) AS date | where
date="2012-12-27"]
| stats count as reviews by pCategory
| sort -reviews | head 10
```



pCategory	reviews
Books	10431
Movies	6982
Music	2378
Home	1758
Clothing	1638
Industrial	1191
Toys	1056
Electronics	1015
Sports	963
Shoes	949

Verification: To verify the query. We take the join of review on27th Dec, with only the movies category. The result returned is similar to the count of movies reviewed in the main query.

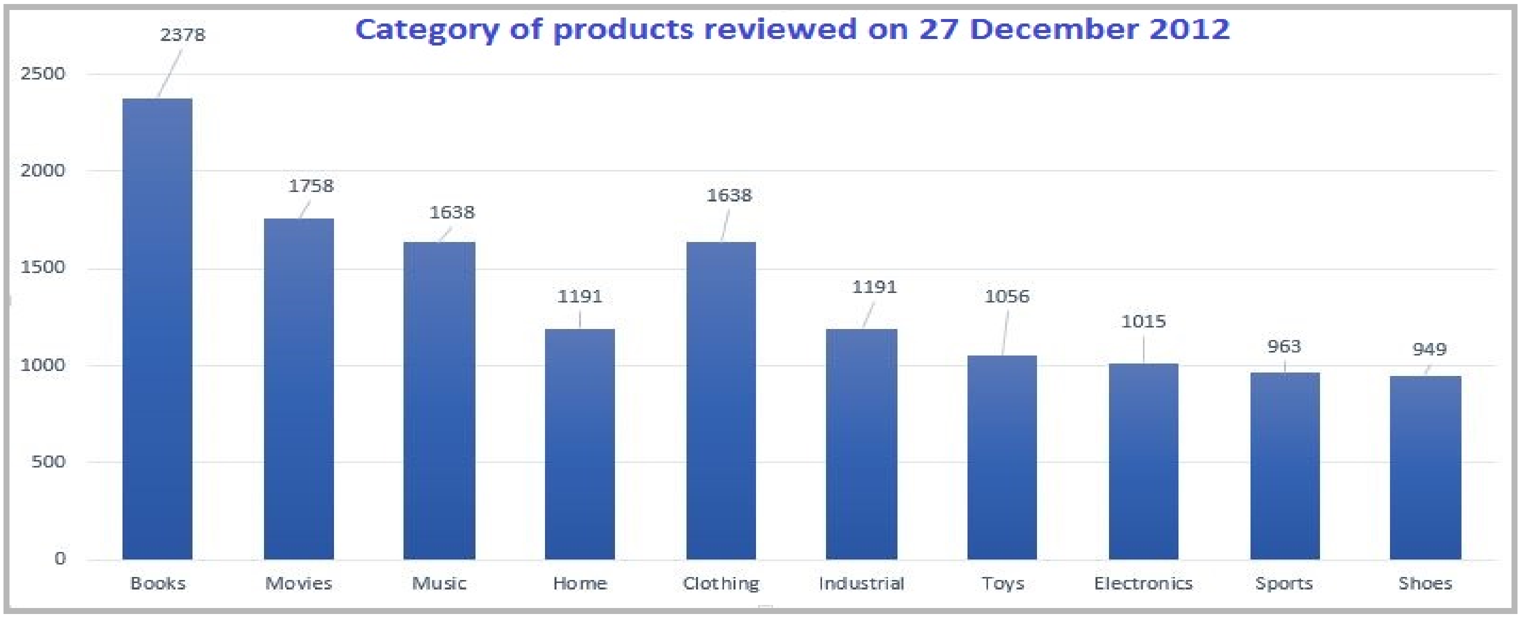
sourcetype=Categories pCategory=Movies
| JOIN type=inner productId [search sourcetype=all_review earliest=12/25/2012:0:0:0 latest=12/28/2012:23:59:59 | convert timeformat="%Y-%m-%d" ctime(_time) AS date | where date="2012-12-27"]



```
sourcetype=Categories pCategory=Movies| JOIN type=inner productId [ search sourcetype=all_review earliest=12/25/2012:0:0:0
latest=12/28/2012:23:59:59 | convert timeformat="%Y-%m-%d" ctime(_time) AS date | where date="2012-12-27"]
```

✓ 6,982 events (before 12/7/14 12:07:59.000 PM)

Visualization:

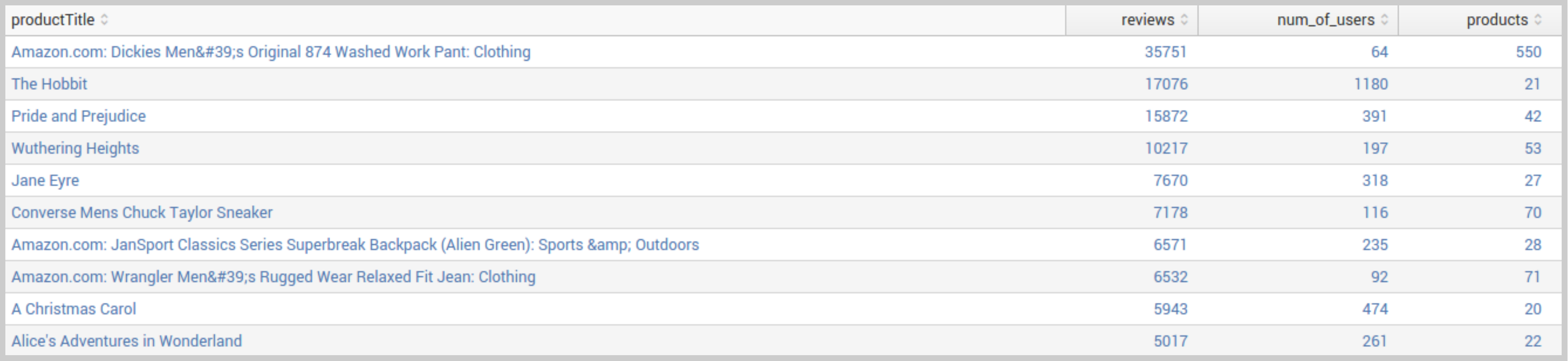


4.3 Most reviewed products for 2012

Objective: Aim is to find recent popular products on amazon. We also aim to find the user involvement with these products on amazon.

Description: Gathered all reviews for 2012 counted the reviews,number of products and and users. And sorted the result to get the most popular product. In the results we see strange values for the number of users reviewing the product. Later we analyzed the number of products with that title. This showed that when the number of users were less the number of products were more. Meaning either the same user goes and reviews all products with the same name(as we can have a one product by many different sellers) or amazon servers have some mechanism to replicate the reviews for all similar products.

Query: **sourcetype="all_review" earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59**
| stats count as reviews,dc(reviewUserId) as num_of_users,dc(productId) as products by productTitle
| sort -reviews | head 10

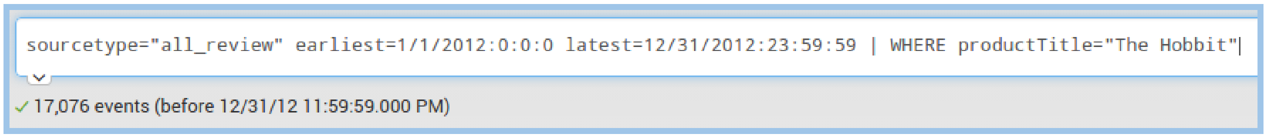


productTitle	reviews	num_of_users	products
Amazon.com: Dickies Men&#39;s Original 874 Washed Work Pant: Clothing	35751	64	550
The Hobbit	17076	1180	21
Pride and Prejudice	15872	391	42
Wuthering Heights	10217	197	53
Jane Eyre	7670	318	27
Converse Mens Chuck Taylor Sneaker	7178	116	70
Amazon.com: JanSport Classics Series Superbreak Backpack (Alien Green): Sports &amp; Outdoors	6571	235	28
Amazon.com: Wrangler Men&#39;s Rugged Wear Relaxed Fit Jean: Clothing	6532	92	71
A Christmas Carol	5943	474	20
Alice's Adventures in Wonderland	5017	261	22

Verification:

Part 1: For verification, we obtained reviews for product titled as “The Hobbit”. It matched with the count we obtained in the output of our main query.

sourcetype="all_review" earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59
| WHERE productTitle="The Hobbit"

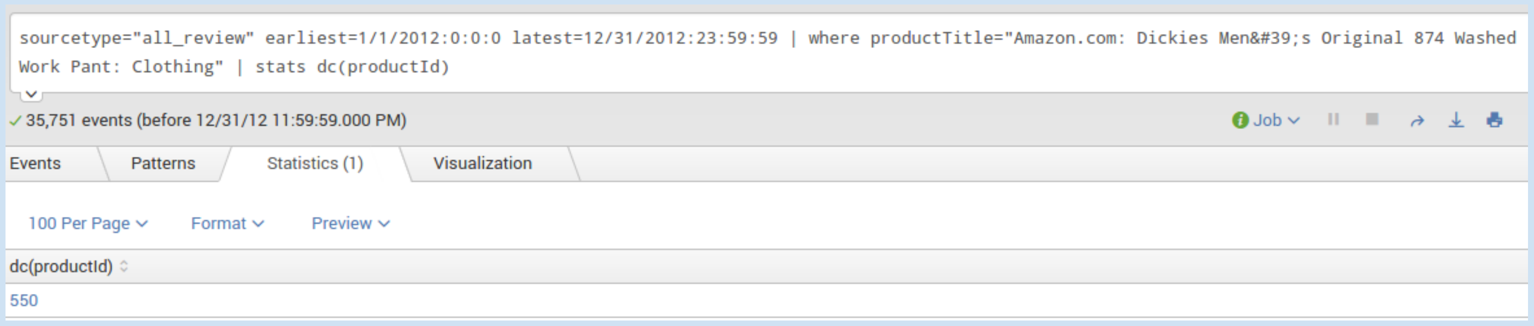


```
sourcetype="all_review" earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59 | WHERE productTitle="The Hobbit"
```

✓ 17,076 events (before 12/31/12 11:59:59.000 PM)

Part 2: We also verified the number of products for a particular product title by first getting count of products for a particular product title and later manually inspecting the entries to find an anomaly. We found the number to match the main query. Also did not notice any anomalies.

sourcetype="all_review" earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59
| where productTitle="Amazon.com: Dickies Men&#39;s Original 874 Washed Work Pant: Clothing"
| stats dc(productId)

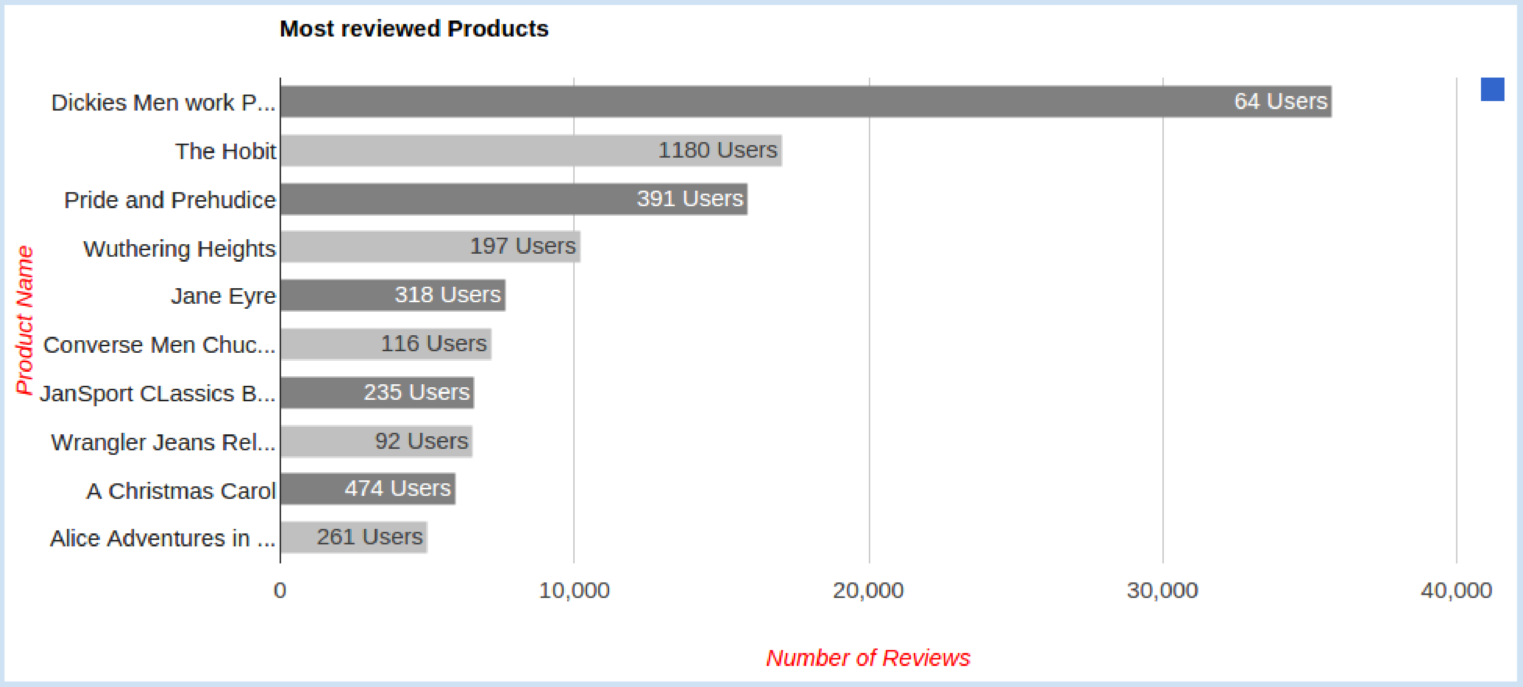


```
sourcetype="all_review" earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59 | where productTitle="Amazon.com: Dickies Men&#39;s Original 874 Washed
Work Pant: Clothing" | stats dc(productId)
```

✓ 35,751 events (before 12/31/12 11:59:59.000 PM)

dc(productId)
550

Visualization:

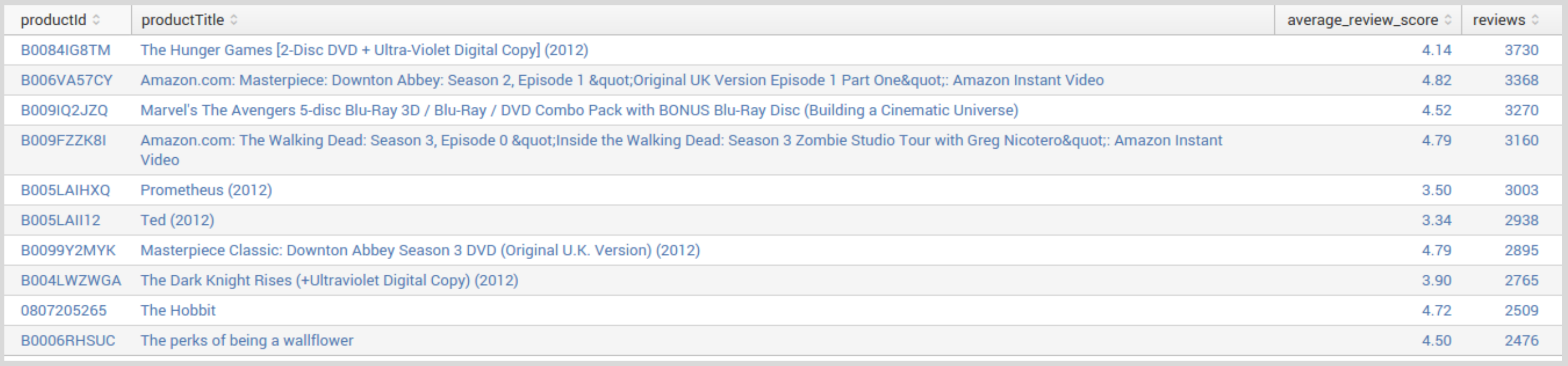


4.4 Average rating of most reviewed products

Objective: To determine average rating of products that are reviewed most on amazon.

Description: We aim to find the average rating of the most reviewed products(based on productId not the title) of all time. We gathered all reviews and calculated average review score, then applied count by product id and rounded off average review score up to two places after decimal point. Finally we sorted the result based on review count. We observed that the most reviewed products have a good rating. This makes sense for them to make it to the top 10 list because the reason they sold more and were reviewed more is that they are good products.

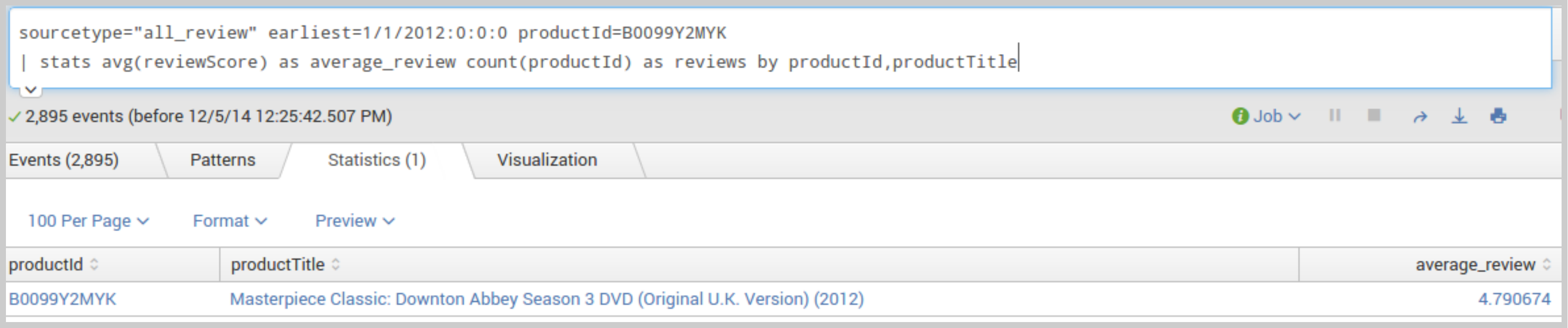
Query:
```
sourcetype="all_review" earliest=1/1/2012:0:0:0
| stats avg(reviewScore) as average_review_score count(productId) as reviews by productId,productTitle
| eval average_review_score= round(average_review_score,2)
| sort -reviews
| head 10
```



productId	productTitle	average_review_score	reviews
B0084IG8TM	The Hunger Games [2-Disc DVD + Ultra-Violet Digital Copy] (2012)	4.14	3730
B006VA57CY	Amazon.com: Masterpiece: Downton Abbey: Season 2, Episode 1 &quot;Original UK Version Episode 1 Part One&quot;: Amazon Instant Video	4.82	3368
B009IQ2JZQ	Marvel's The Avengers 5-disc Blu-Ray 3D / Blu-Ray / DVD Combo Pack with BONUS Blu-Ray Disc (Building a Cinematic Universe)	4.52	3270
B009FZZK8I	Amazon.com: The Walking Dead: Season 3, Episode 0 &quot;Inside the Walking Dead: Season 3 Zombie Studio Tour with Greg Nicotero&quot;: Amazon Instant Video	4.79	3160
B005LAIHXQ	Prometheus (2012)	3.50	3003
B005LAII12	Ted (2012)	3.34	2938
B0099Y2MYK	Masterpiece Classic: Downton Abbey Season 3 DVD (Original U.K. Version) (2012)	4.79	2895
B004LWZWGA	The Dark Knight Rises (+Ultraviolet Digital Copy) (2012)	3.90	2765
0807205265	The Hobbit	4.72	2509
B0006RHSUC	The perks of being a wallflower	4.50	2476

Verification: For verification, we calculated average value of reviews and number of reviews associated with productId titled B000N793XU. We found that both values matched with the values obtained in the output of main query.

```
sourcetype="all_review" productId=B000N793XU
| stats avg(reviewScore) as average_review count(productId) as reviews by productId
| eval average_review=round(average_review,2)
```

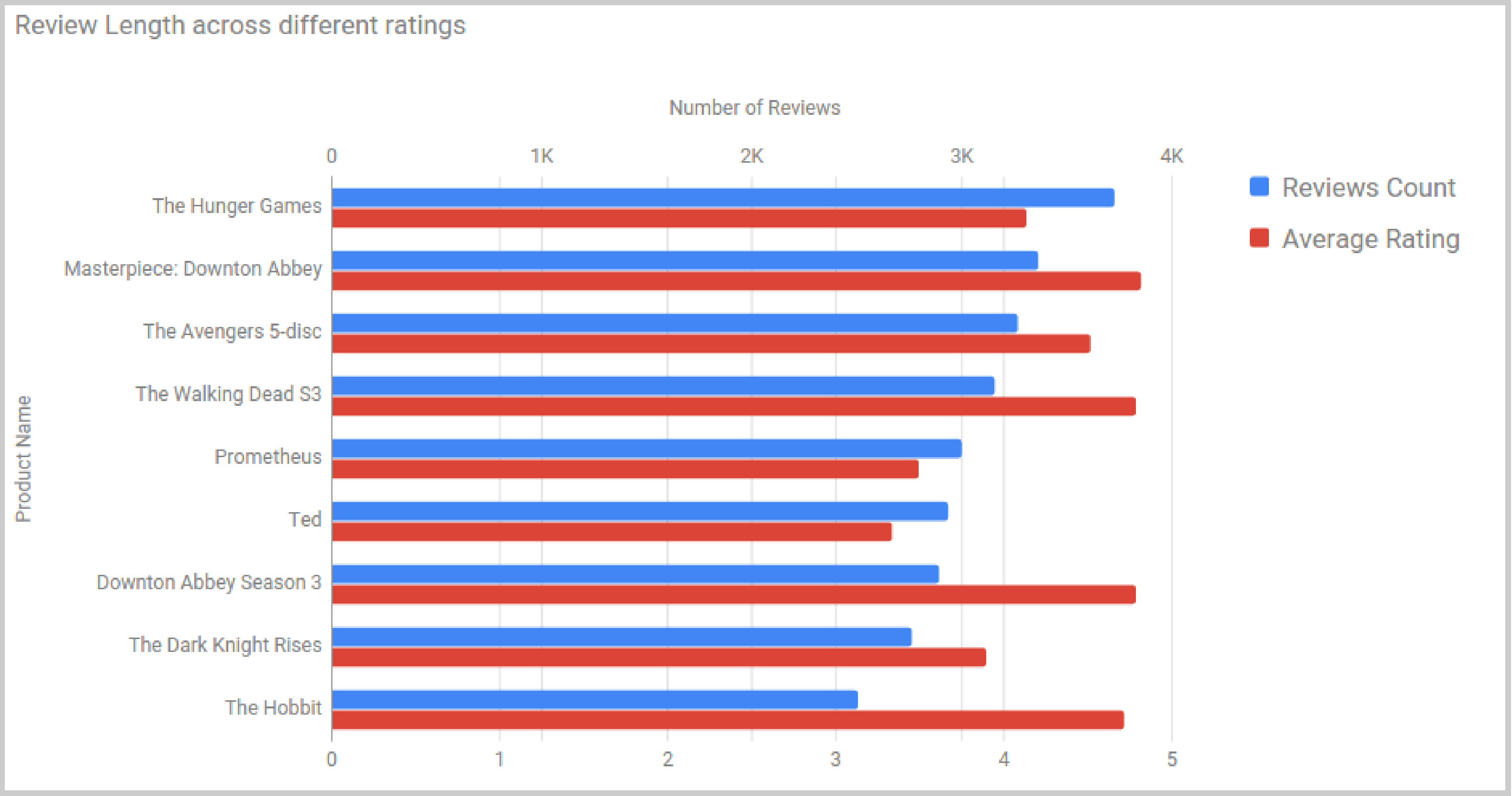


```
sourcetype="all_review" earliest=1/1/2012:0:0:0 productId=B0099Y2MYK
| stats avg(reviewScore) as average_review count(productId) as reviews by productId,productTitle
```

✓ 2,895 events (before 12/5/14 12:25:42.507 PM)

productId	productTitle	average_review
B0099Y2MYK	Masterpiece Classic: Downton Abbey Season 3 DVD (Original U.K. Version) (2012)	4.790674

Visualization:



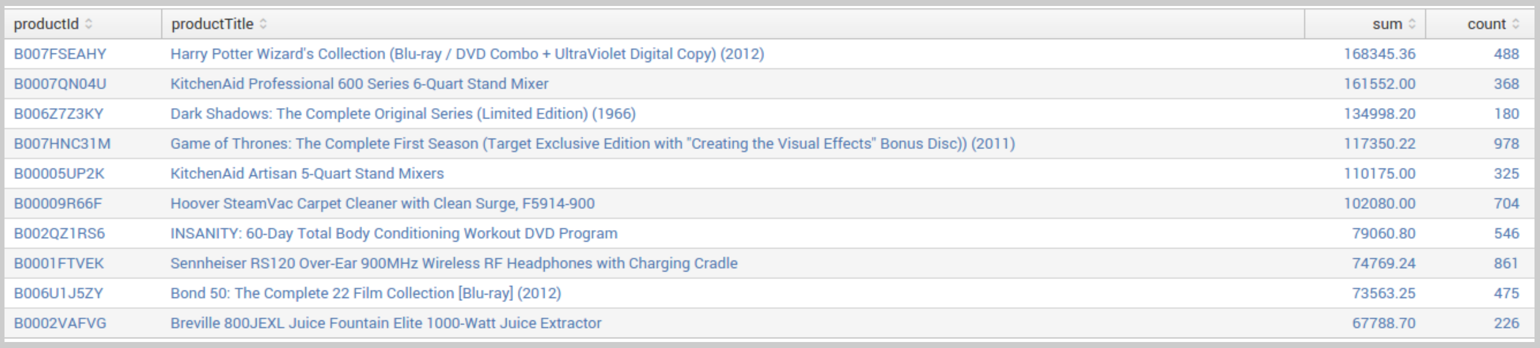
4.5 Most grossing item in the year 2012

Objective: To determine the most earning item on amazon based on the price in reviews.

Description: To find the most grossing item we simply calculated the sum of prices and sort the result based on the sum. Here we observe that surprisingly it's not a very common product with huge number of reviews making it to the top, but rather the highly priced DVD packs and kitchen products that earned the most. This is probably because the common product already have many reviews and there is not much to review in them. While the products in the list are the one's which has lots of content that can be reviewed like movie combo's.

Query:

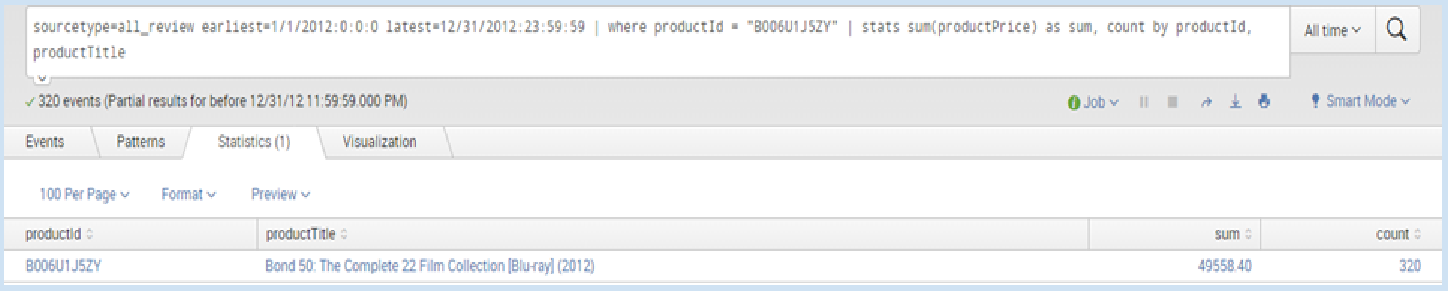
```
sourcetype=all_review earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59
| stats sum(productPrice) as sum, count by productId, productTitle
```



productId	productTitle	sum	count
B007FSEAHY	Harry Potter Wizard's Collection (Blu-ray / DVD Combo + UltraViolet Digital Copy) (2012)	168345.36	488
B0007QN04U	KitchenAid Professional 600 Series 6-Quart Stand Mixer	161552.00	368
B006Z7Z3KY	Dark Shadows: The Complete Original Series (Limited Edition) (1966)	134998.20	180
B007HNC31M	Game of Thrones: The Complete First Season (Target Exclusive Edition with "Creating the Visual Effects" Bonus Disc)) (2011)	117350.22	978
B00005UP2K	KitchenAid Artisan 5-Quart Stand Mixers	110175.00	325
B00009R66F	Hoover SteamVac Carpet Cleaner with Clean Surge, F5914-900	102080.00	704
B002QZ1RS6	INSANITY: 60-Day Total Body Conditioning Workout DVD Program	79060.80	546
B0001FTVEK	Sennheiser RS120 Over-Ear 900MHz Wireless RF Headphones with Charging Cradle	74769.24	861
B006U1J5ZY	Bond 50: The Complete 22 Film Collection [Blu-ray] (2012)	73563.25	475
B0002VAFVG	Breville 800JEXL Juice Fountain Elite 1000-Watt Juice Extractor	67788.70	226

Verification: For verification, we selected product id = B006U1J5ZY and calculated the values. This values matched with values in the output of main query.

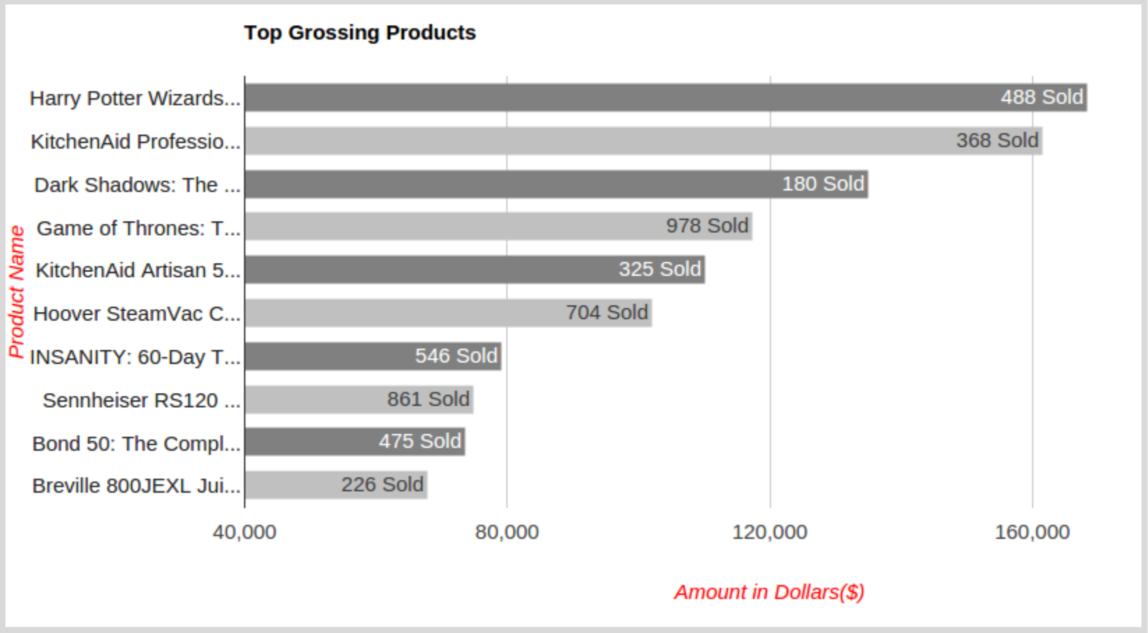
```
sourcetype=all_review earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59
| where productId = "B006U1J5ZY"
| stats sum(productPrice) as sum, count by productId, productTitle
```



```
sourcetype=all_review earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59 | where productId = "B006U1J5ZY" | stats sum(productPrice) as sum, count by productId, productTitle
```

productId	productTitle	sum	count
B006U1J5ZY	Bond 50: The Complete 22 Film Collection [Blu-ray] (2012)	49558.40	320

Visualization: Created using Google charts

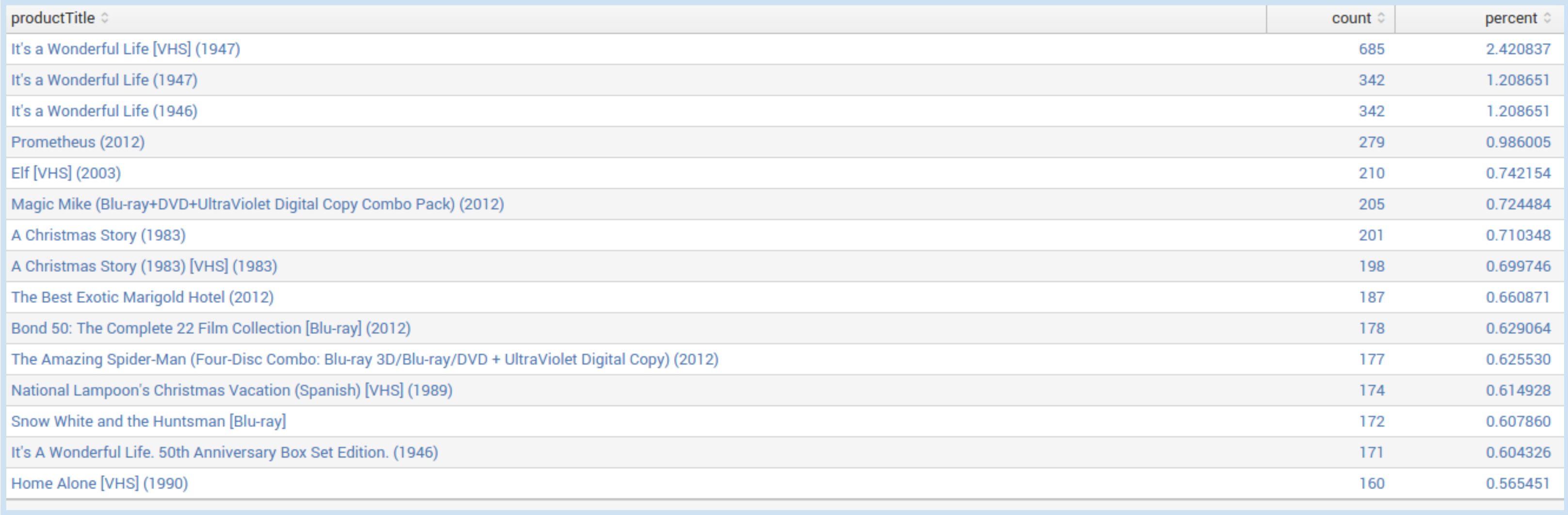


4.6 Most reviewed movies During the holidays

Objective: To determine movies that are reviewed most during holidays.

Description: Leveraging splunk's backbone of time series indexing. We aim to find the movies that are reviewed most during christmas and new year holidays. We gathered all reviews between 12-21-2012 and 1-3-2013, then applied join with categories table having primary category fields as movies. Sorted the result based on the most reviewed movies. American Christmas fantasy comedy "A wonderful Life " and "A Christmas story" made it to the top list. This shows people watch these classics even today and have strong opinion for them.

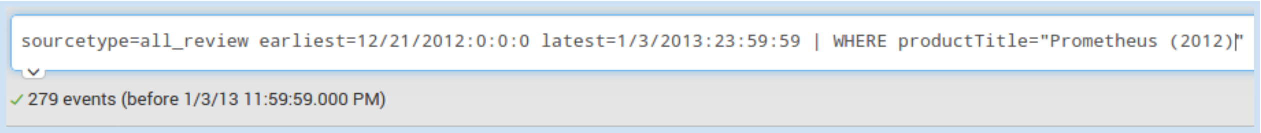
Query:
```
sourcetype=all_review earliest=12/21/2012:0:0:0 latest=1/3/2013:23:59:59
 | JOIN type=inner productId [SEARCH sourcetype=categories pCategory="Movies"]
| top limit=15 productTitle
```



productTitle	count	percent
It's a Wonderful Life [VHS] (1947)	685	2.420837
It's a Wonderful Life (1947)	342	1.208651
It's a Wonderful Life (1946)	342	1.208651
Prometheus (2012)	279	0.986005
Elf [VHS] (2003)	210	0.742154
Magic Mike (Blu-ray+DVD+UltraViolet Digital Copy Combo Pack) (2012)	205	0.724484
A Christmas Story (1983)	201	0.710348
A Christmas Story (1983) [VHS] (1983)	198	0.699746
The Best Exotic Marigold Hotel (2012)	187	0.660871
Bond 50: The Complete 22 Film Collection [Blu-ray] (2012)	178	0.629064
The Amazing Spider-Man (Four-Disc Combo: Blu-ray 3D/Blu-ray/DVD + UltraViolet Digital Copy) (2012)	177	0.625530
National Lampoon's Christmas Vacation (Spanish) [VHS] (1989)	174	0.614928
Snow White and the Huntsman [Blu-ray]	172	0.607860
It's A Wonderful Life. 50th Anniversary Box Set Edition. (1946)	171	0.604326
Home Alone [VHS] (1990)	160	0.565451

Verification: For verification, we calculated number of reviews associated with product titled as Prometheus. The number of events obtained matched with the count observed in the output of main query .

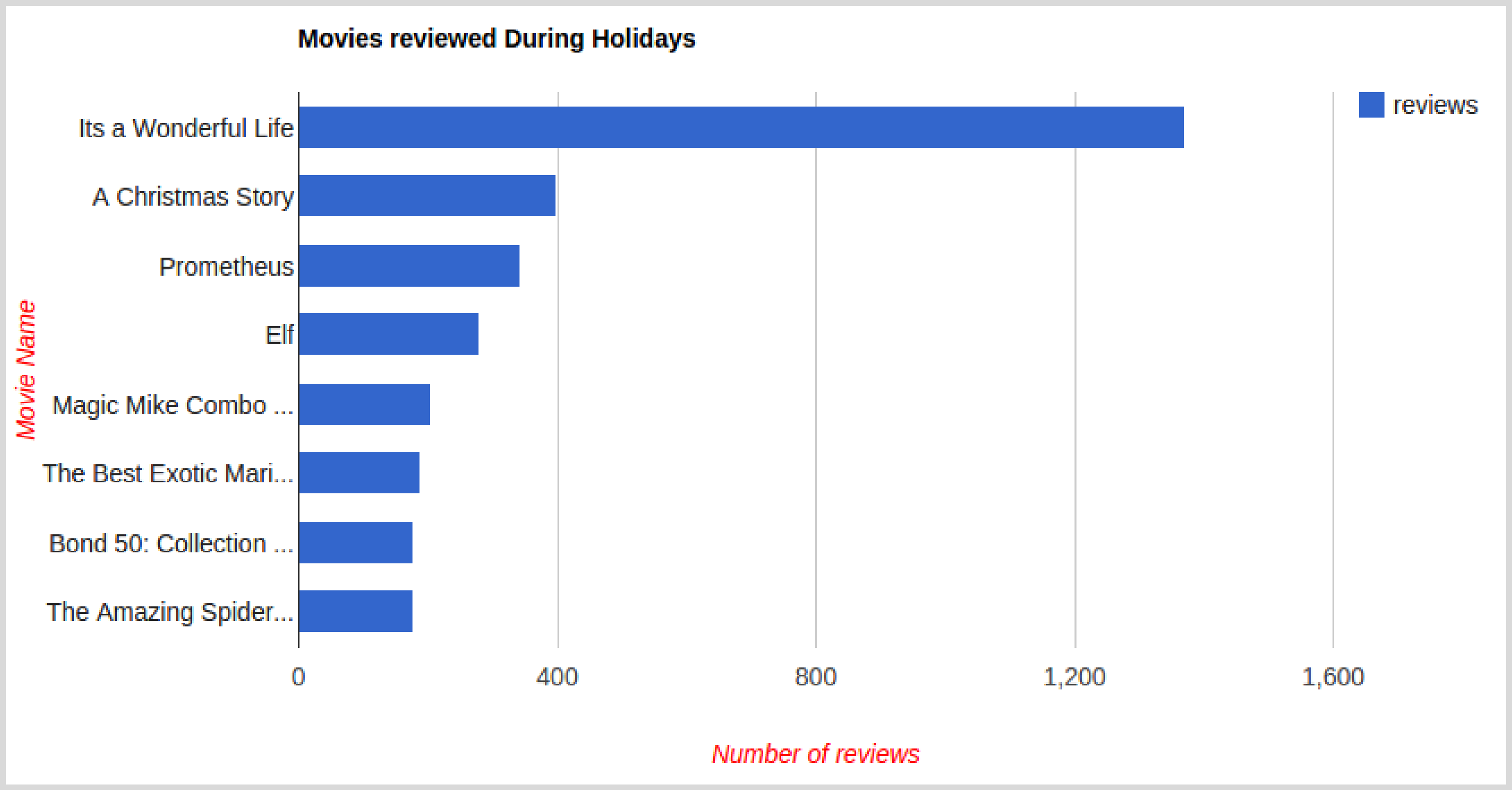
sourcetype=all_review earliest=12/21/2012:0:0:0 latest=1/3/2013:23:59:59
| WHERE productTitle="Prometheus (2012)"

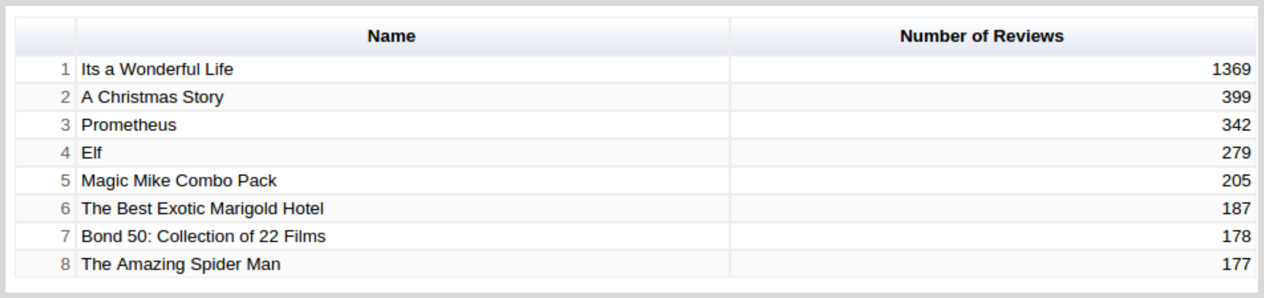


```
sourcetype=all_review earliest=12/21/2012:0:0:0 latest=1/3/2013:23:59:59 | WHERE productTitle="Prometheus (2012)"
```

✓ 279 events (before 1/3/13 11:59:59.000 PM)

Visualization :





	Name	Number of Reviews
1	Its a Wonderful Life	1369
2	A Christmas Story	399
3	Prometheus	342
4	Elf	279
5	Magic Mike Combo Pack	205
6	The Best Exotic Marigold Hotel	187
7	Bond 50: Collection of 22 Films	178
8	The Amazing Spider Man	177

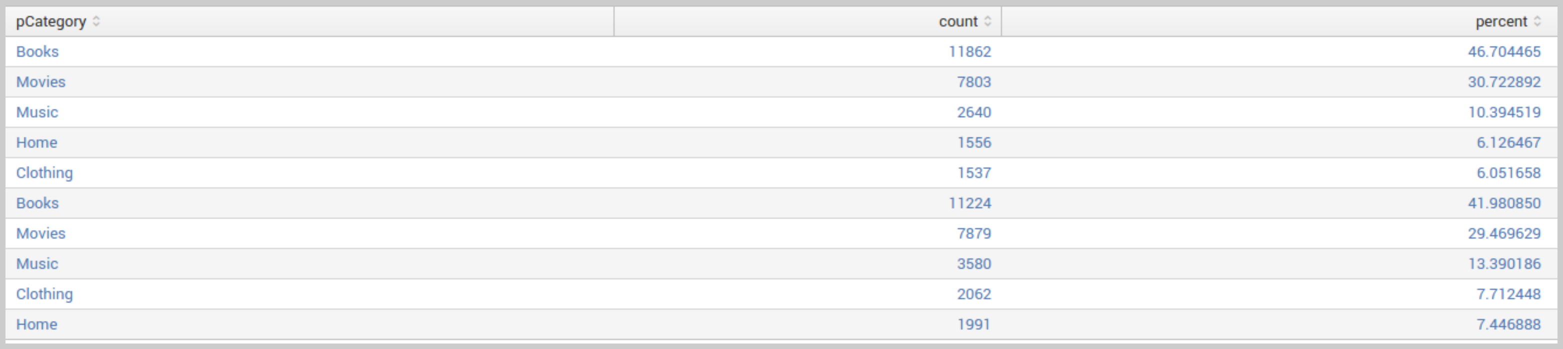
4.7 Change in product popularity across 2011 and 2012

Objective: We aim to find how user preference changed during the black friday week over the years.

Description: Firstly we find all the reviews for Black Friday in the year 2012 for popular product categories. Then compare the same information for the year 2011. We observe that the overall number of reviews have increased, and so has the number of books and movies reviews during these period. This shows that amazon is still popular for books and movie reviews.

Query:

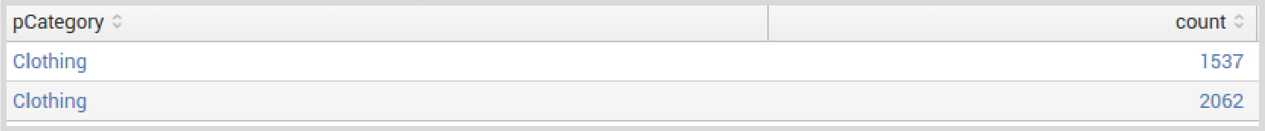
```
sourcetype=Categories pCategory="Books" OR pCategory="Movies" OR pCategory="Music" OR
pCategory="Clothing" OR pCategory="Home"
| JOIN type=inner productId [SEARCH sourcetype=all_review earliest=11/21/2012:0:0:0
latest=11/26/2012:23:59:59]
| top pCategory
| APPEND [SEARCH sourcetype=Categories pCategory="Books" OR pCategory="Movies" OR
pCategory="Music" OR pCategory="Clothing" OR pCategory="Home" | JOIN type=inner productId
[SEARCH sourcetype=all_review earliest=11/23/2011:0:0:0 latest=11/28/2011:23:59:59]
| top pCategory ]
```



pCategory	count	percent
Books	11862	46.704465
Movies	7803	30.722892
Music	2640	10.394519
Home	1556	6.126467
Clothing	1537	6.051658
Books	11224	41.980850
Movies	7879	29.469629
Music	3580	13.390186
Clothing	2062	7.712448
Home	1991	7.446888

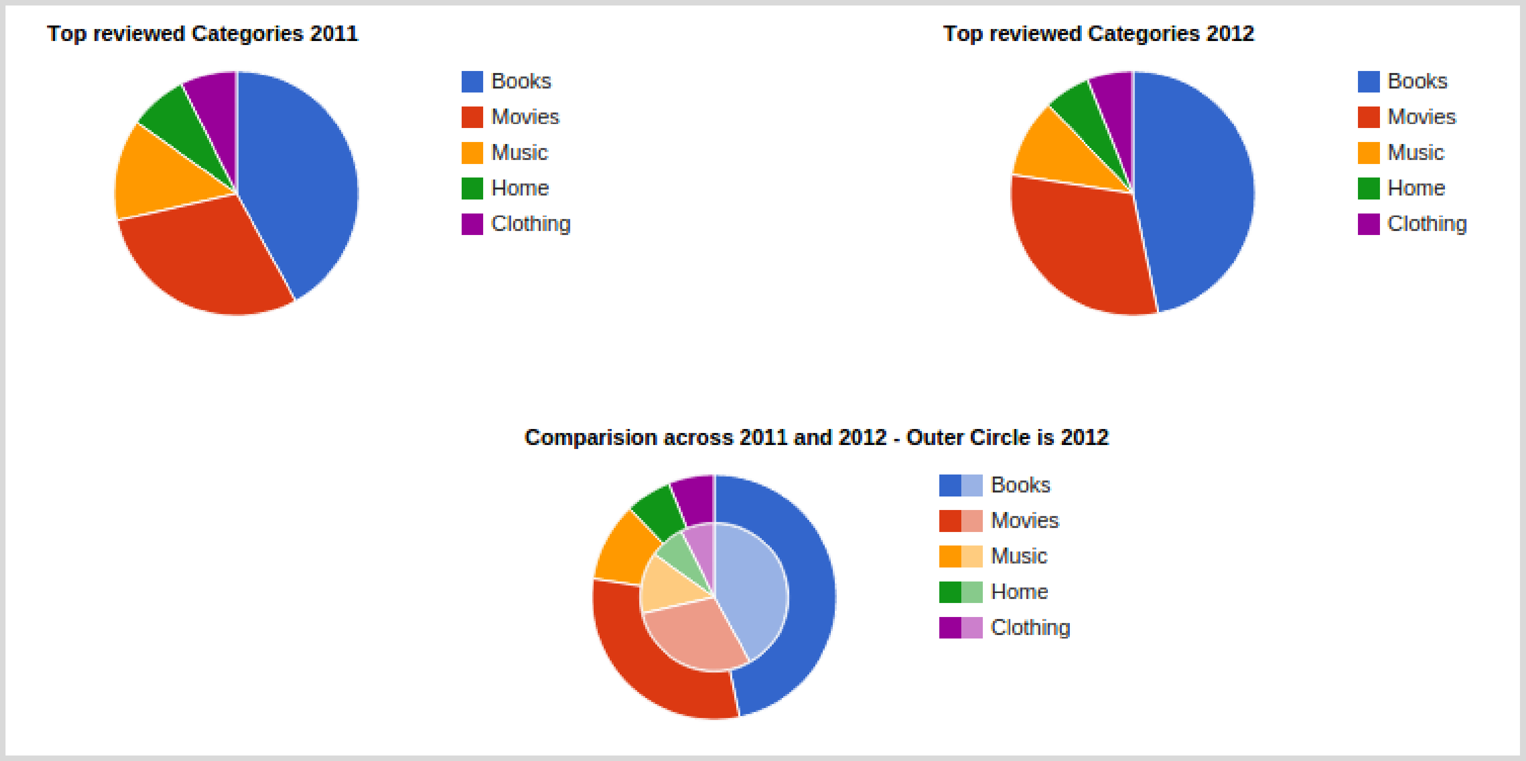
Verification: For verification, we selected primary category=Clothing and checked for the output. The output matched with the output obtained by main query.

```
sourcetype=Categories | where pCategory="Clothing" | JOIN type=inner productId [SEARCH
sourcetype=all_review earliest=11/21/2012:0:0:0 latest=11/26/2012:23:59:59]
| top pCategory
| APPEND [SEARCH sourcetype=Categories | where pCategory="Clothing" | JOIN type=inner
productId [SEARCH sourcetype=all_review earliest=11/23/2011:0:0:0 latest=11/28/2011:23:59:59]
| top pCategory ]
```



pCategory	count
Clothing	1537
Clothing	2062

Visualization:

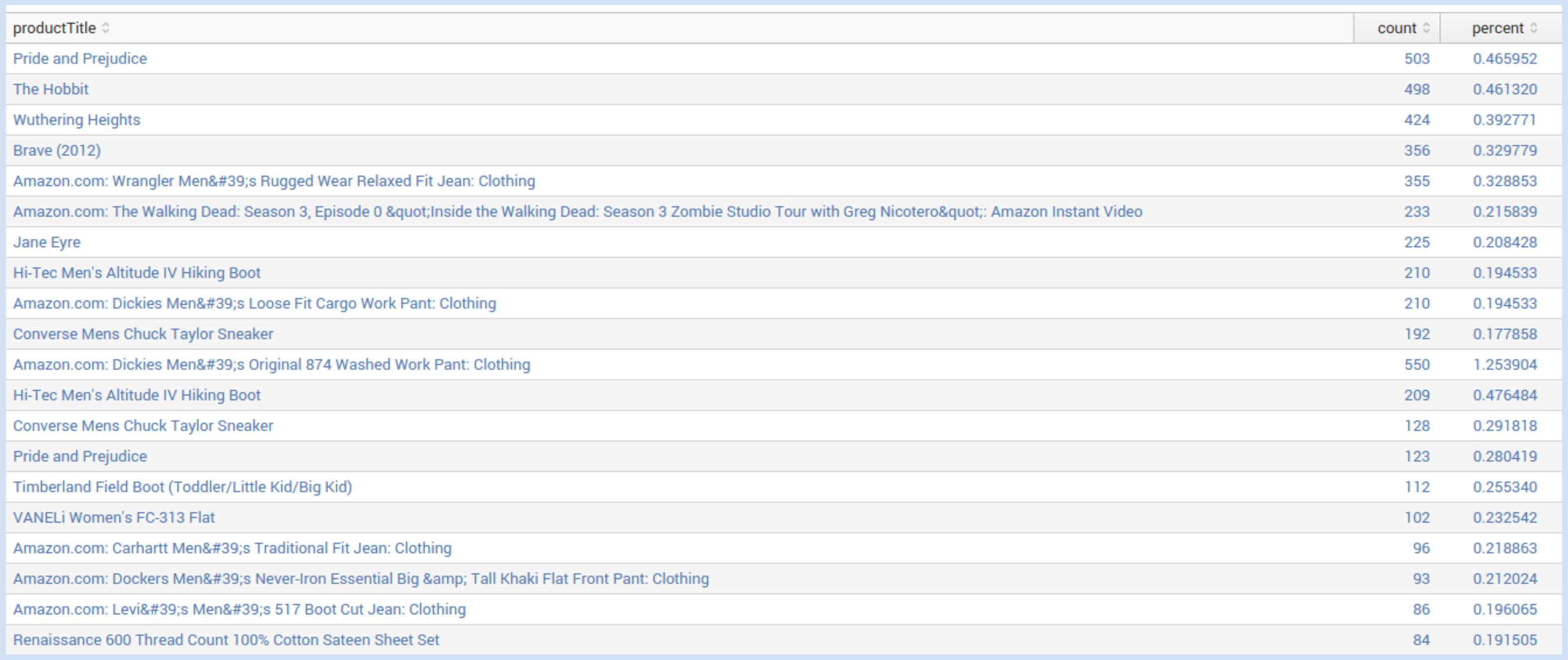


4.8 Popular products on black friday across year 2011 and 2012

Objective: To determine the most reviewed products during black friday's across 2 years.

Description: We aim to find the top 10 products that are reviewed during the Black Friday weeks of 2011 and 2012. We aim to check whether the same products are reviewed during each black friday. We gathered all reviews for 2011 & 2012 seperately and applied top by product title, then applied append to add both the results. We observe the pride and prejudice is the only product that makes it to the top ten for both the years. We observe here that the category of the products for the 2 years in very different. For 2012 movies are in the top 5 while for 2011 we have clothing dominating the top 5. These shows people are not interested in reviewing stuff they get good deals on.

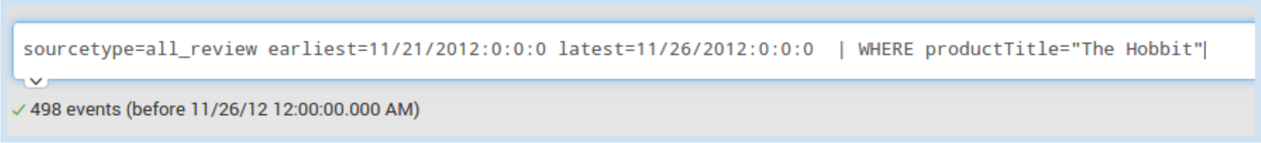
Query: **sourcetype=all_review earliest=11/21/2012:0:0:0 latest=11/26/2012:0:0:0**
| top productTitle
| APPEND [search sourcetype=all_review earliest=11/23/2011:0:0:0 latest=11/28/2011:0:0:0 | top productTitle]



productTitle	count	percent
Pride and Prejudice	503	0.465952
The Hobbit	498	0.461320
Wuthering Heights	424	0.392771
Brave (2012)	356	0.329779
Amazon.com: Wrangler Men&#39;s Rugged Wear Relaxed Fit Jean: Clothing	355	0.328853
Amazon.com: The Walking Dead: Season 3, Episode 0 &quot;Inside the Walking Dead: Season 3 Zombie Studio Tour with Greg Nicotero&quot;: Amazon Instant Video	233	0.215839
Jane Eyre	225	0.208428
Hi-Tec Men's Altitude IV Hiking Boot	210	0.194533
Amazon.com: Dickies Men&#39;s Loose Fit Cargo Work Pant: Clothing	210	0.194533
Converse Mens Chuck Taylor Sneaker	192	0.177858
Amazon.com: Dickies Men&#39;s Original 874 Washed Work Pant: Clothing	550	1.253904
Hi-Tec Men's Altitude IV Hiking Boot	209	0.476484
Converse Mens Chuck Taylor Sneaker	128	0.291818
Pride and Prejudice	123	0.280419
Timberland Field Boot (Toddler/Little Kid/Big Kid)	112	0.255340
VANELi Women's FC-313 Flat	102	0.232542
Amazon.com: Carhartt Men&#39;s Traditional Fit Jean: Clothing	96	0.218863
Amazon.com: Dockers Men&#39;s Never-Iron Essential Big &amp; Tall Khaki Flat Front Pant: Clothing	93	0.212024
Amazon.com: Levi&#39;s Men&#39;s 517 Boot Cut Jean: Clothing	86	0.196065
Renaissance 600 Thread Count 100% Cotton Sateen Sheet Set	84	0.191505

Verification: For verification, we selected product titled as “The Hobbit” and found output. The output found matched with the output obtained by main query.

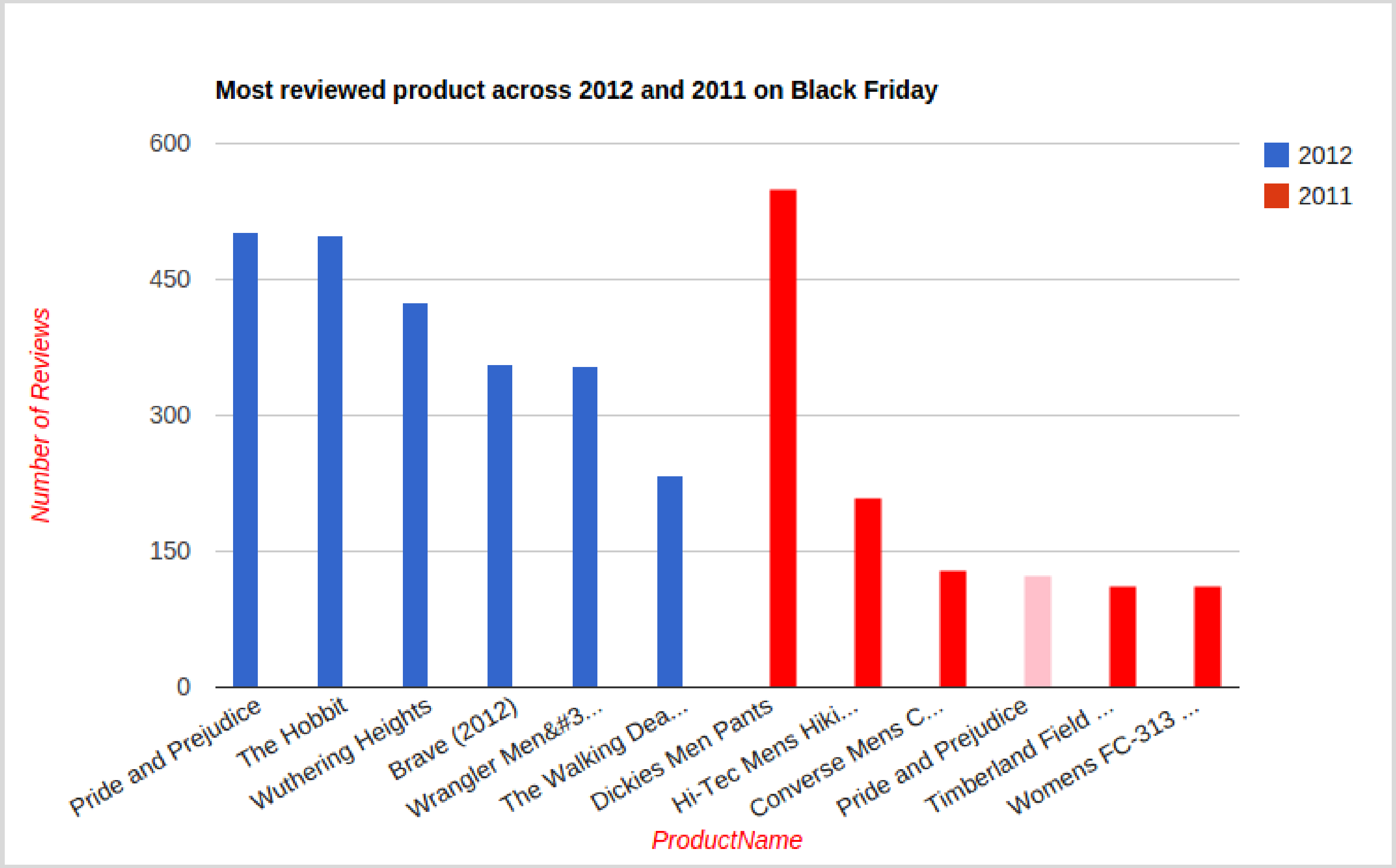
sourcetype=all_review earliest=11/21/2012:0:0:0 latest=11/26/2012:0:0:0 | WHERE productTitle="The Hobbit"



```
sourcetype=all_review earliest=11/21/2012:0:0:0 latest=11/26/2012:0:0:0  | WHERE productTitle="The Hobbit"
```

✓ 498 events (before 11/26/12 12:00:00.000 AM)

Visualization :

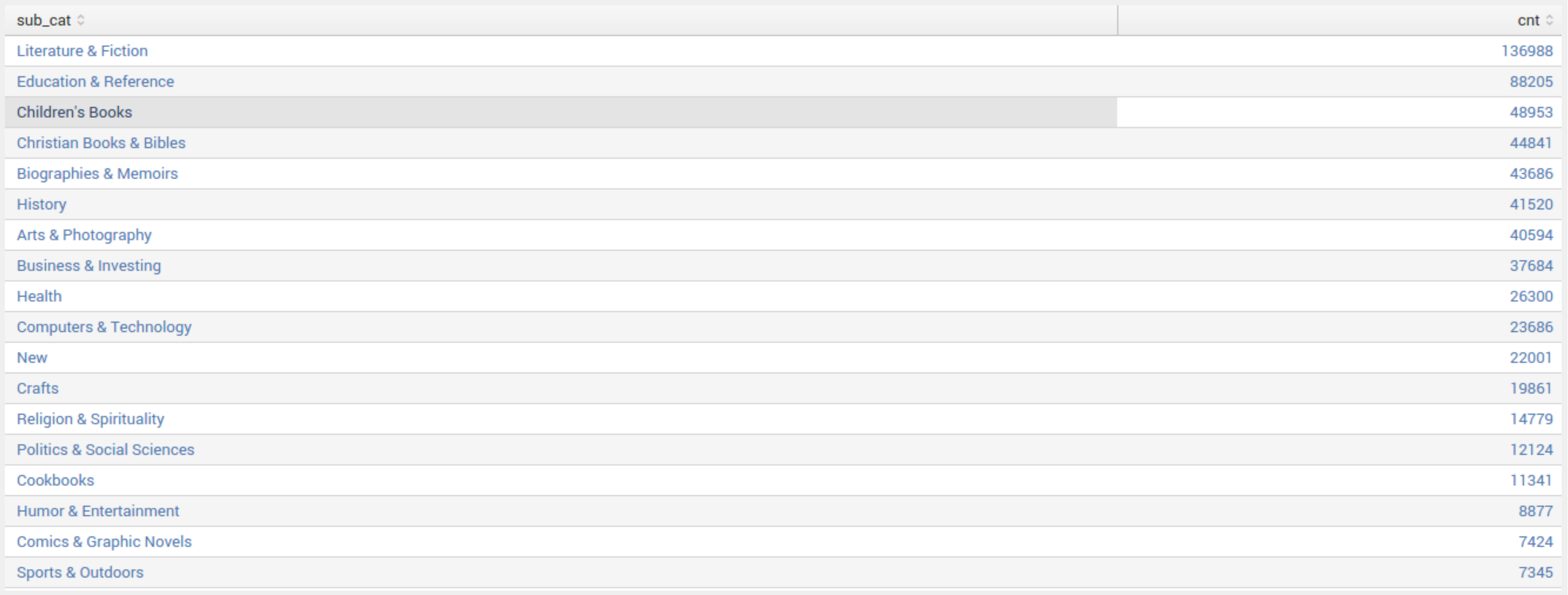


4.9 Sub categories of books on Amazon

Objective: Aim to find the sub categories of books on amazon. Since most reviews on amazon are on books. We thought exploring the subcategory may some give insight for the direction for next quiries.

Description: From the categories table we extracted the secondary category field. This field is a list of comma separated categories related to the book. The first value in the list is same as the primary category. Hence we have considered the second which is closest to the subcategory of the book.

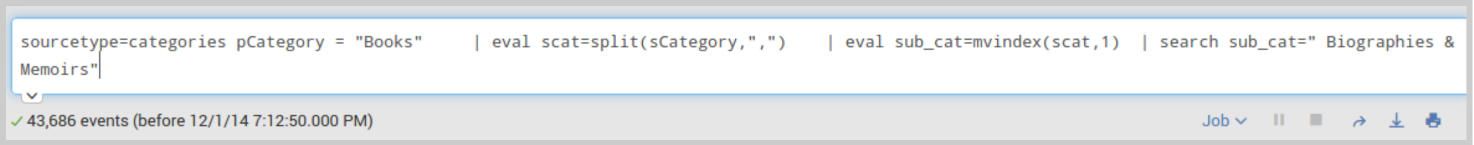
```
sourcetype=categories pCategory = "Books"
| eval scat=split(sCategory,",")
| eval sub_cat=mvindex(scat,1)
| stats count as cnt by sub_cat
| sort -cnt
```



sub_cat	cnt
Literature & Fiction	136988
Education & Reference	88205
Children's Books	48953
Christian Books & Bibles	44841
Biographies & Memoirs	43686
History	41520
Arts & Photography	40594
Business & Investing	37684
Health	26300
Computers & Technology	23686
New	22001
Crafts	19861
Religion & Spirituality	14779
Politics & Social Sciences	12124
Cookbooks	11341
Humor & Entertainment	8877
Comics & Graphic Novels	7424
Sports & Outdoors	7345

Verification: For verification, we calculated the number of books under the sub category "Biographies & Memoirs". It matched with the count obtained in the output of main query.

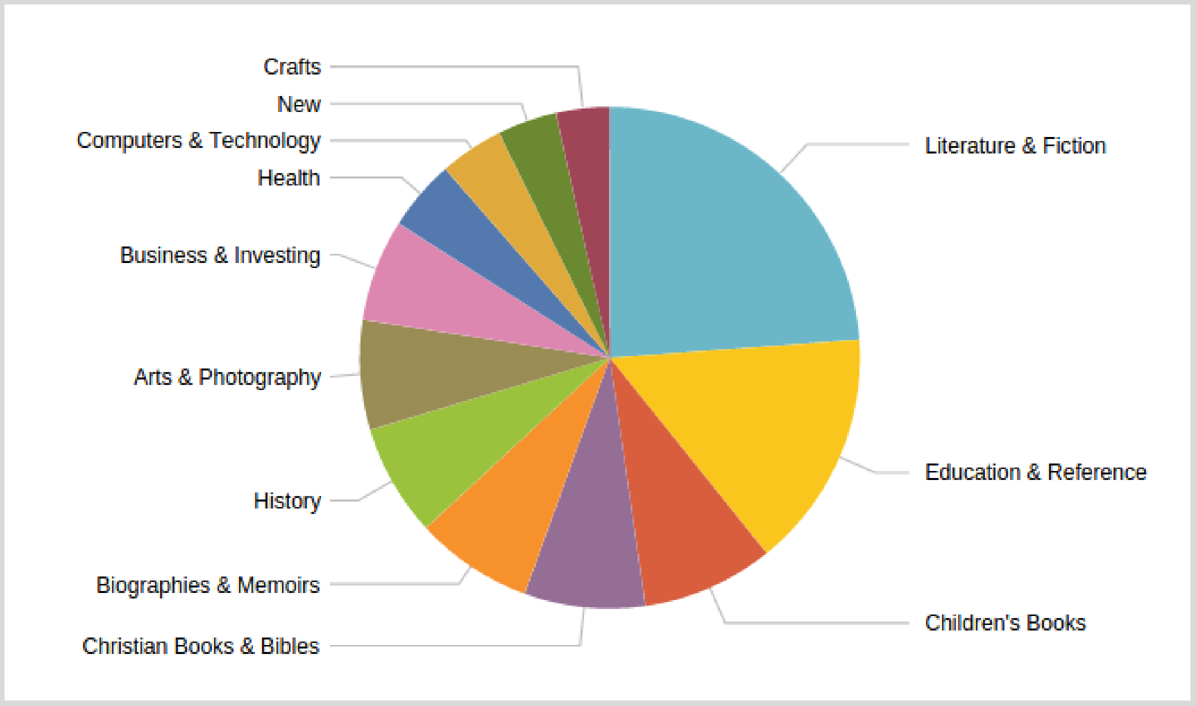
sourcetype=categories pCategory = "Books"
| eval scat=split(sCategory,",")
| eval sub_cat=mvindex(scat,1)
| search sub_cat=" Biographies & Memoirs"



```
sourcetype=categories pCategory = "Books"     | eval scat=split(sCategory,",")    | eval sub_cat=mvindex(scat,1)  | search sub_cat=" Biographies &
Memoirs"
```

✓ 43,686 events (before 12/1/14 7:12:50.000 PM)

Visualization: generated by Splunk



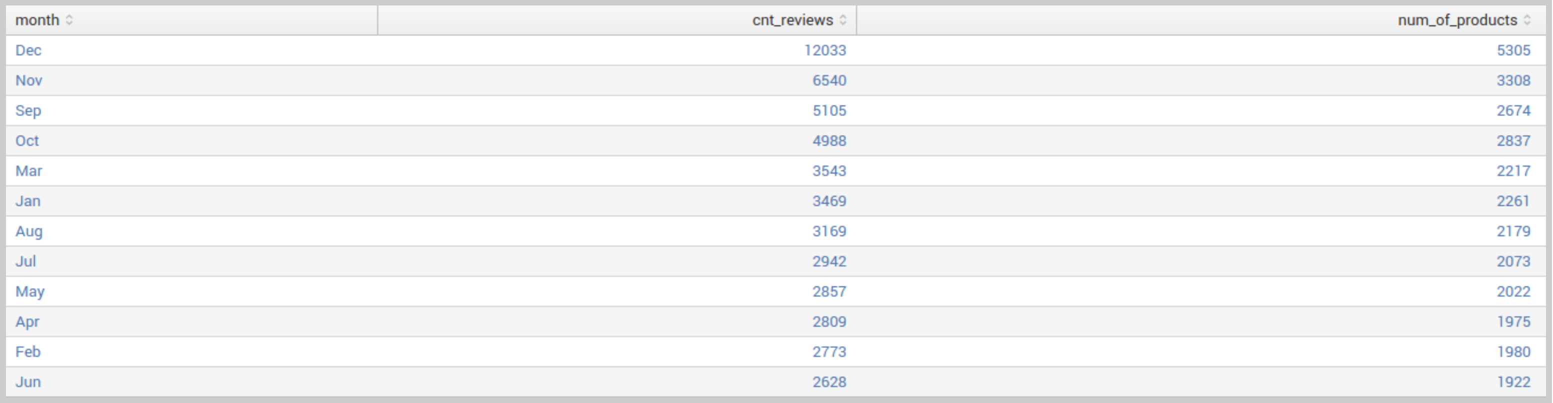
4.10 Pattern of reviews for Christian books and Bibles

Objective: To determine when “Christian books and Bibles” sell the most.

Description: In a sub search we extract all books under “Christian books and Bibles” and then find all reviews for these products. Later we group the reviews by month to find when they are reviewed the most. We observe as anticipated these books are most reviewed during December.

Query:

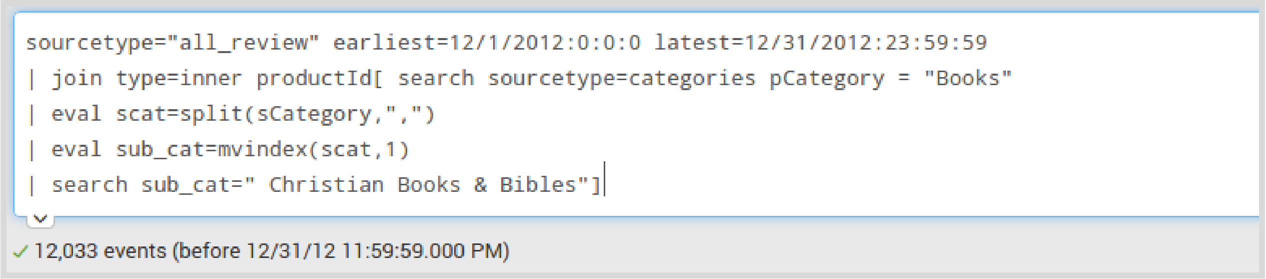
```
sourcetype="all_review" earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59
| join type=inner productId[ search sourcetype=categories pCategory = "Books"
| eval scat=split(sCategory,",")
| eval sub_cat=mvindex(scat,1)
| search sub_cat=" Christian Books & Bibles"]
|  convert timeformat="%b" ctime(_time) AS month
| stats count as cnt_reviews,dc(productId) as num_of_products by month | sort -cnt_reviews
```



month	cnt_reviews	num_of_products
Dec	12033	5305
Nov	6540	3308
Sep	5105	2674
Oct	4988	2837
Mar	3543	2217
Jan	3469	2261
Aug	3169	2179
Jul	2942	2073
May	2857	2022
Apr	2809	1975
Feb	2773	1980
Jun	2628	1922

Verification: To verify the the main query we change the date in the query to December and verify the count as expected.

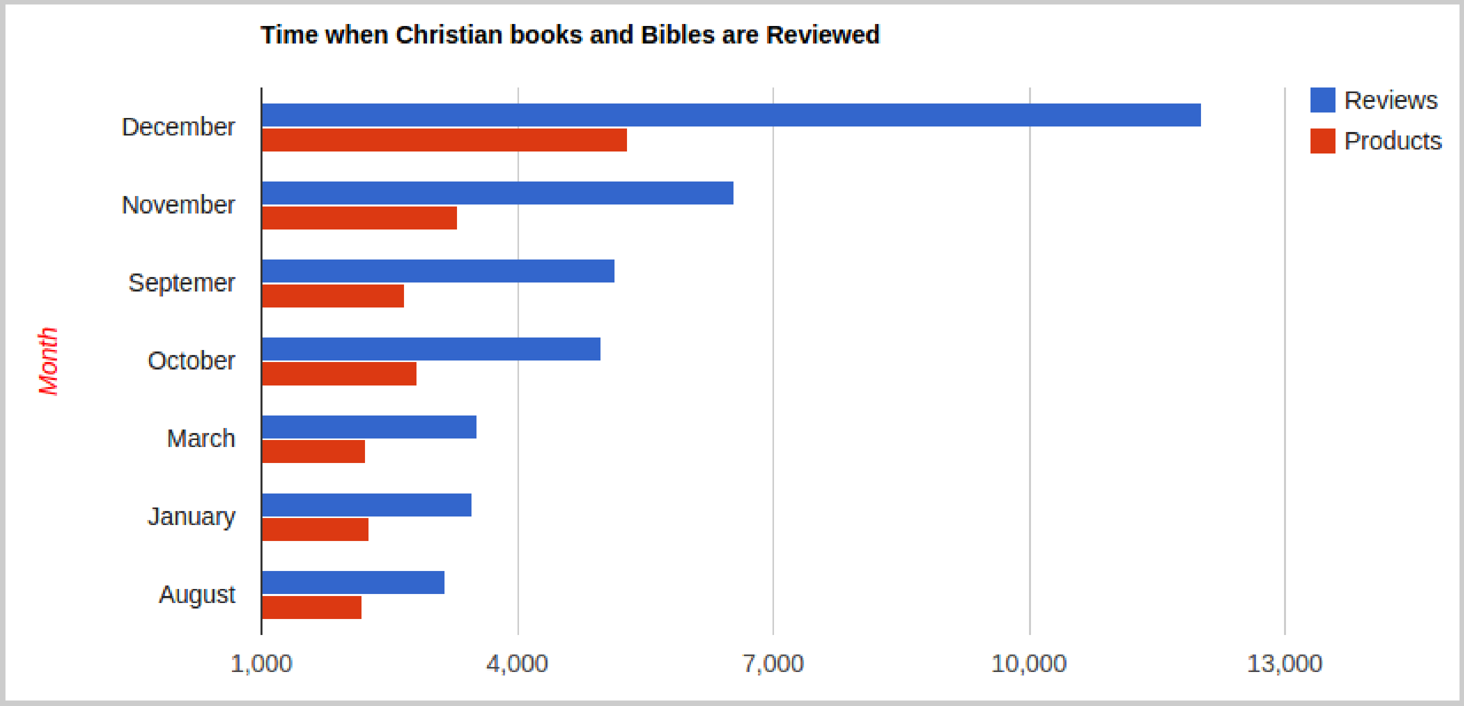
sourcetype="all_review" earliest=12/1/2012:0:0:0 latest=12/31/2012:23:59:59
| join type=inner productId[search sourcetype=categories pCategory = "Books"
| eval scat=split(sCategory,",")
| eval sub_cat=mvindex(scat,1)
| search sub_cat=" Christian Books & Bibles"]



```
sourcetype="all_review" earliest=12/1/2012:0:0:0 latest=12/31/2012:23:59:59
| join type=inner productId[ search sourcetype=categories pCategory = "Books"
| eval scat=split(sCategory,",")
| eval sub_cat=mvindex(scat,1)
| search sub_cat=" Christian Books & Bibles"]
```

✓ 12,033 events (before 12/31/12 11:59:59.000 PM)

Visualization:



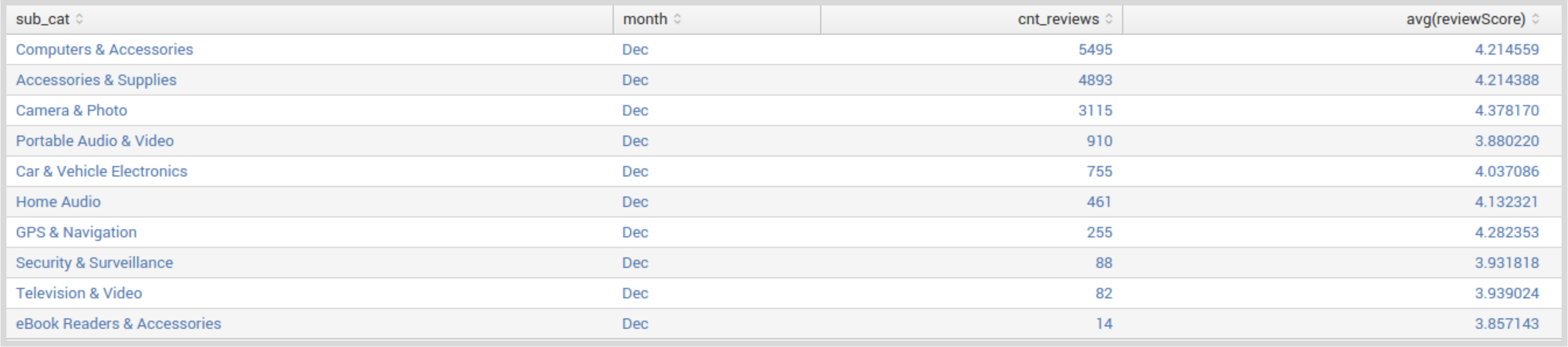
4.11 Pattern for reviews in Electronics and their rating

Objective: Aim to find which is the most popular category of products on amazon and when does it sell the most. Also what is the rating in that period.

Description: We first write a sub search for the categories table to extract all sub category of electronics. Later we join this with all_review table to get all the electronic reviews with the sub category. Now we calculate the average rating and number of reviews by month for the year 2012. This gives a subcategory of electronics with most reviewed month and average rating for that month. We observe that Electronics sell most in Nov/Dec i.e the shopping period with black friday and holidays and also most products are rated good during this period as they mainly a gift or on deals or are a planned/well thought purchases.

Query:

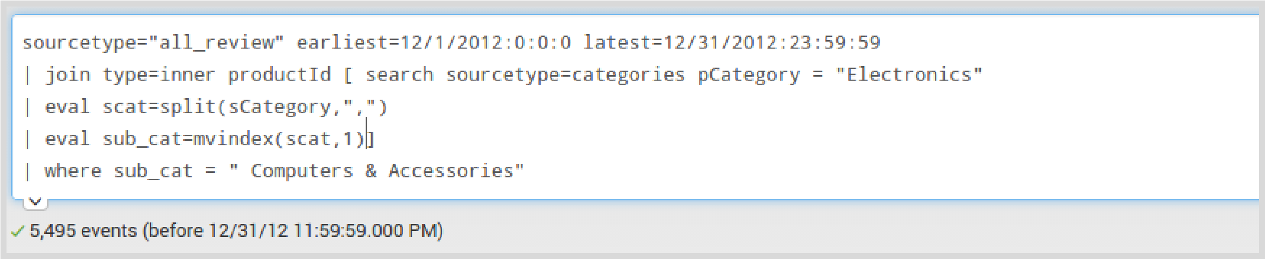
```
sourcetype="all_review" earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59
| join type=inner productId [ search sourcetype=categories pCategory = "Electronics"
| eval scat=split(sCategory,",")
| eval sub_cat=mvindex(scat,1) ]
|  convert timeformat="%b" ctime(_time) AS month
| stats count as cnt_reviews,avg(reviewScore) by sub_cat,month
| sort -cnt_reviews | dedup sub_cat
```



sub_cat	month	cnt_reviews	avg(reviewScore)
Computers & Accessories	Dec	5495	4.214559
Accessories & Supplies	Dec	4893	4.214388
Camera & Photo	Dec	3115	4.378170
Portable Audio & Video	Dec	910	3.880220
Car & Vehicle Electronics	Dec	755	4.037086
Home Audio	Dec	461	4.132321
GPS & Navigation	Dec	255	4.282353
Security & Surveillance	Dec	88	3.931818
Television & Video	Dec	82	3.939024
eBook Readers & Accessories	Dec	14	3.857143

Verification: To verify we calculated the count of all reviewed under the category "Computers and Accessories". We find them to match that of the main query.

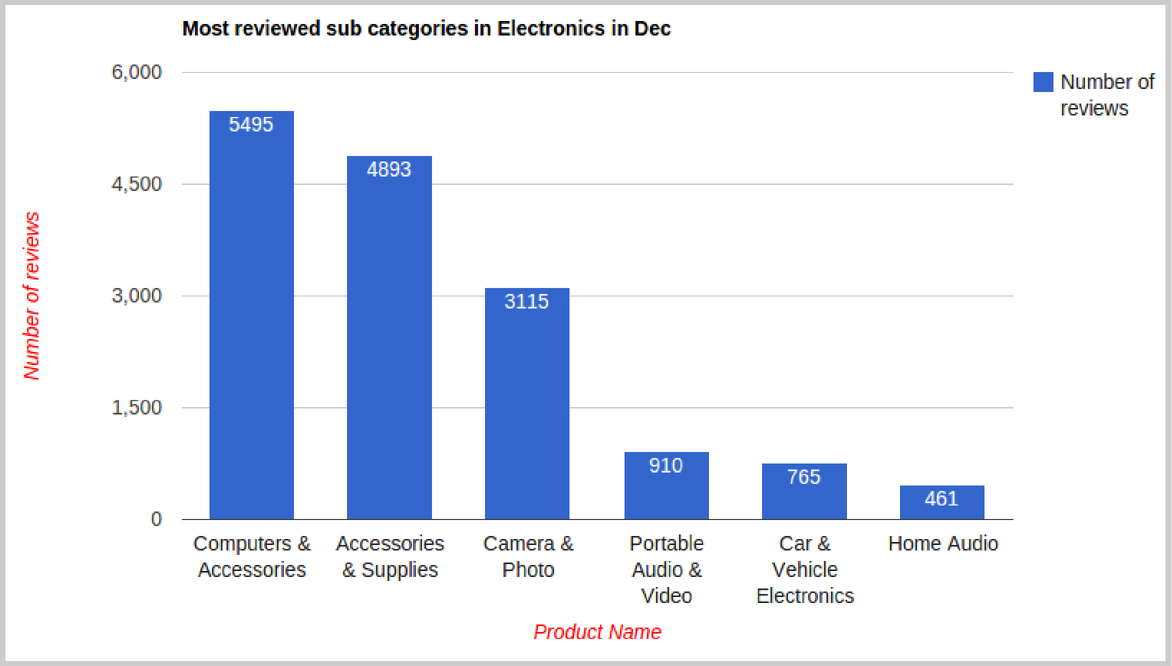
sourcetype="all_review" earliest=12/1/2012:0:0:0 latest=12/31/2012:23:59:59
| join type=inner productId [search sourcetype=categories pCategory = "Electronics"
| eval scat=split(sCategory,",")
| eval sub_cat=mvindex(scat,1)]
| where sub_cat = " Computers & Accessories"



```
sourcetype="all_review" earliest=12/1/2012:0:0:0 latest=12/31/2012:23:59:59
| join type=inner productId [ search sourcetype=categories pCategory = "Electronics"
| eval scat=split(sCategory,",")
| eval sub_cat=mvindex(scat,1)]
| where sub_cat = " Computers & Accessories"
```

✓ 5,495 events (before 12/31/12 11:59:59.000 PM)

Visualization:

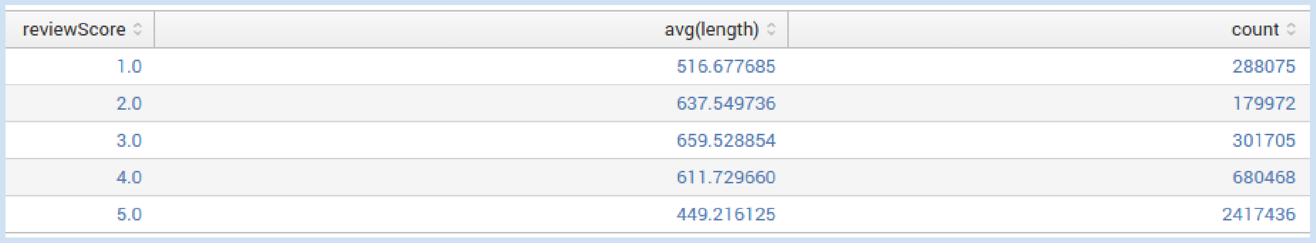


4.12 Average review text for a review rating

Objective: Aim to determine is there is relation between review rating and a review text. We need to see if a user if upset with the product does he write a long descriptive review.

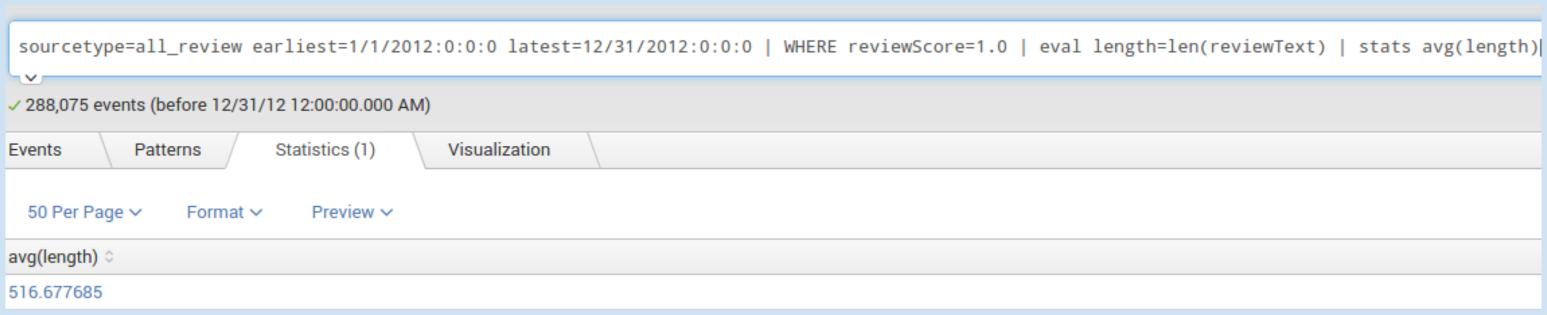
Description: Gathered all reviews between 1-1-2012 and 12-31-2012 and evaluated length of review text, then calculated average length by review score. We observe that a user writes the longest review when the rating is 3 i.e when he is just satisfied with the product not overwhelmed. Review might probably explain why he didn't feel it to be 5.

Query:**sourcetype=all_review earliest=1/1/2012:0:0:0 latest=12/31/2012:0:0:0**
| eval length=len(reviewText)
| stats avg(length),count by reviewScore



reviewScore	avg(length)	count
1.0	516.677685	288075
2.0	637.549736	179972
3.0	659.528854	301705
4.0	611.729660	680468
5.0	449.216125	2417436

Verification: For verification, we selected review score =1.0 and found output. The output found matched with the output obtained by main query
sourcetype=all_review earliest=1/1/2012:0:0:0 latest=12/31/2012:0:0:0 | WHERE reviewScore=1.0 | eval length=len(reviewText) | stats avg(length)

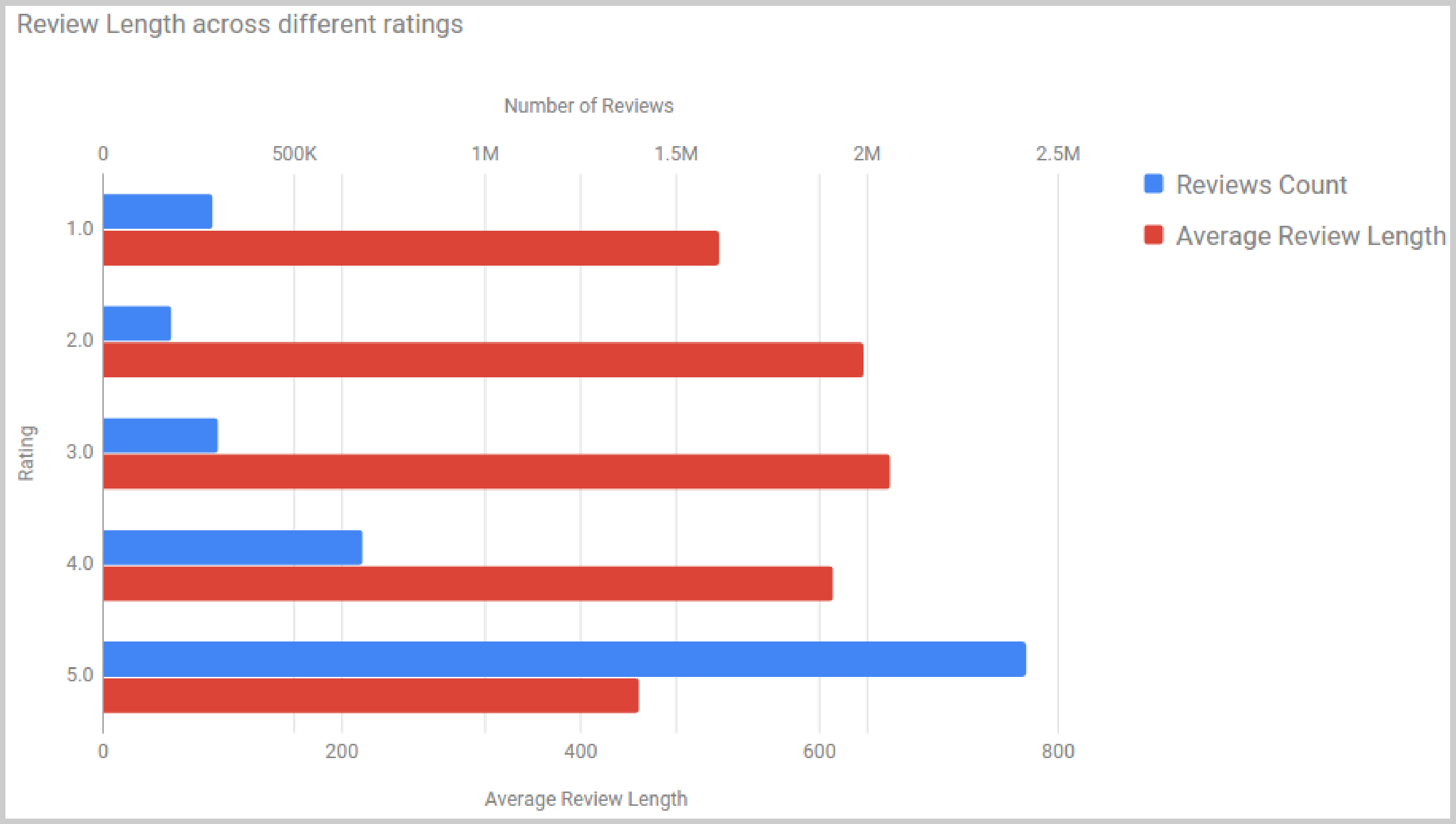


```
sourcetype=all_review earliest=1/1/2012:0:0:0 latest=12/31/2012:0:0:0 | WHERE reviewScore=1.0 | eval length=len(reviewText) | stats avg(length)
```

✓ 288,075 events (before 12/31/12 12:00:00.000 AM)

avg(length)
516.677685

Visualization:

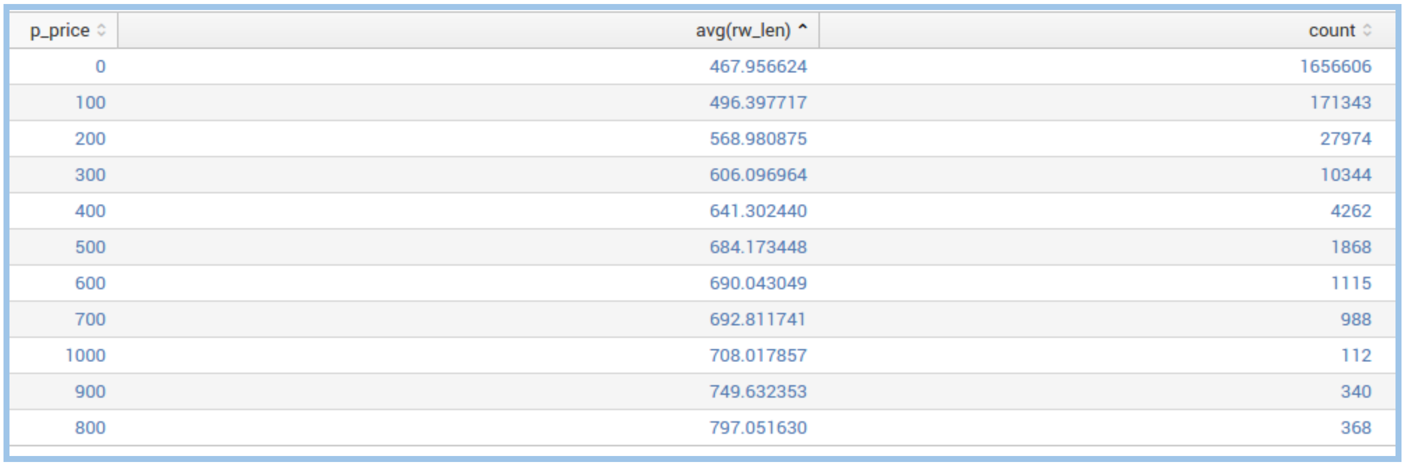


4.13 Average review length by price

Objective: To determine a relation between the review length and the price of the product. Confirm weather expensive products have more detailed reports.

Description: We find the average review length of products by price ranges, for the year 2012. Here p_price=0 includes the range of price 0-50 $ and p_price=100 include the range 50-149 $. We gathered all reviews, rounded off product price and calculated average review length. Here we observe that expensive product do indeed have long descriptive reviews.

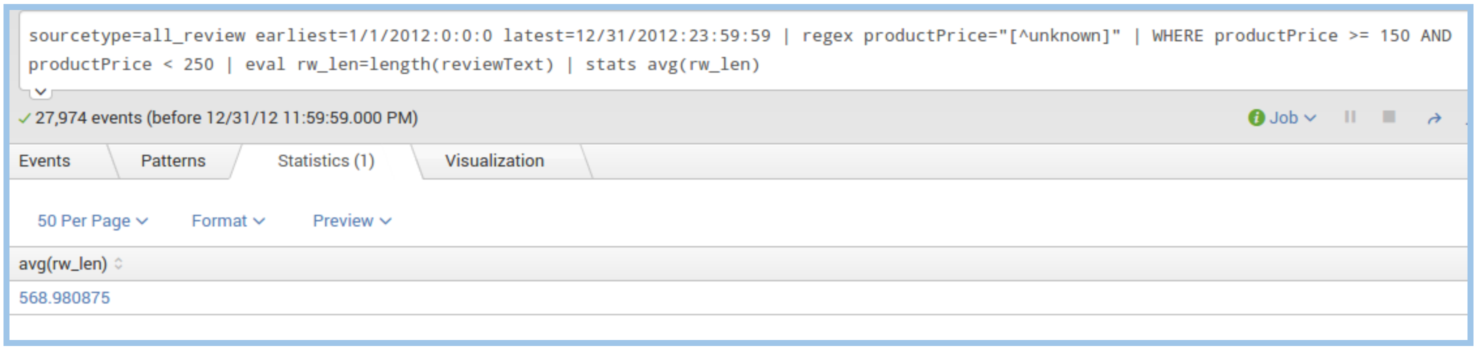
Query: **sourcetype=all_review earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59**
| regex productPrice="[^unknown]"
| eval p_price=round(productPrice,-2)
| eval rw_len=length(reviewText)
| stats avg(rw_len),count by p_price



p_price	avg(rw_len)	count
0	467.956624	1656606
100	496.397717	171343
200	568.980875	27974
300	606.096964	10344
400	641.302440	4262
500	684.173448	1868
600	690.043049	1115
700	692.811741	988
1000	708.017857	112
900	749.632353	340
800	797.051630	368

Verification: For verification we selected product price that is greater than or equal to 150 and less than 250 and found output. the output found matched with output obtained by main query

```
sourcetype=all_review earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59
 | regex productPrice="[^unknown]"
| WHERE productPrice >= 150 AND productPrice < 250
| eval rw_len=length(reviewText) | stats avg(rw_len)
```

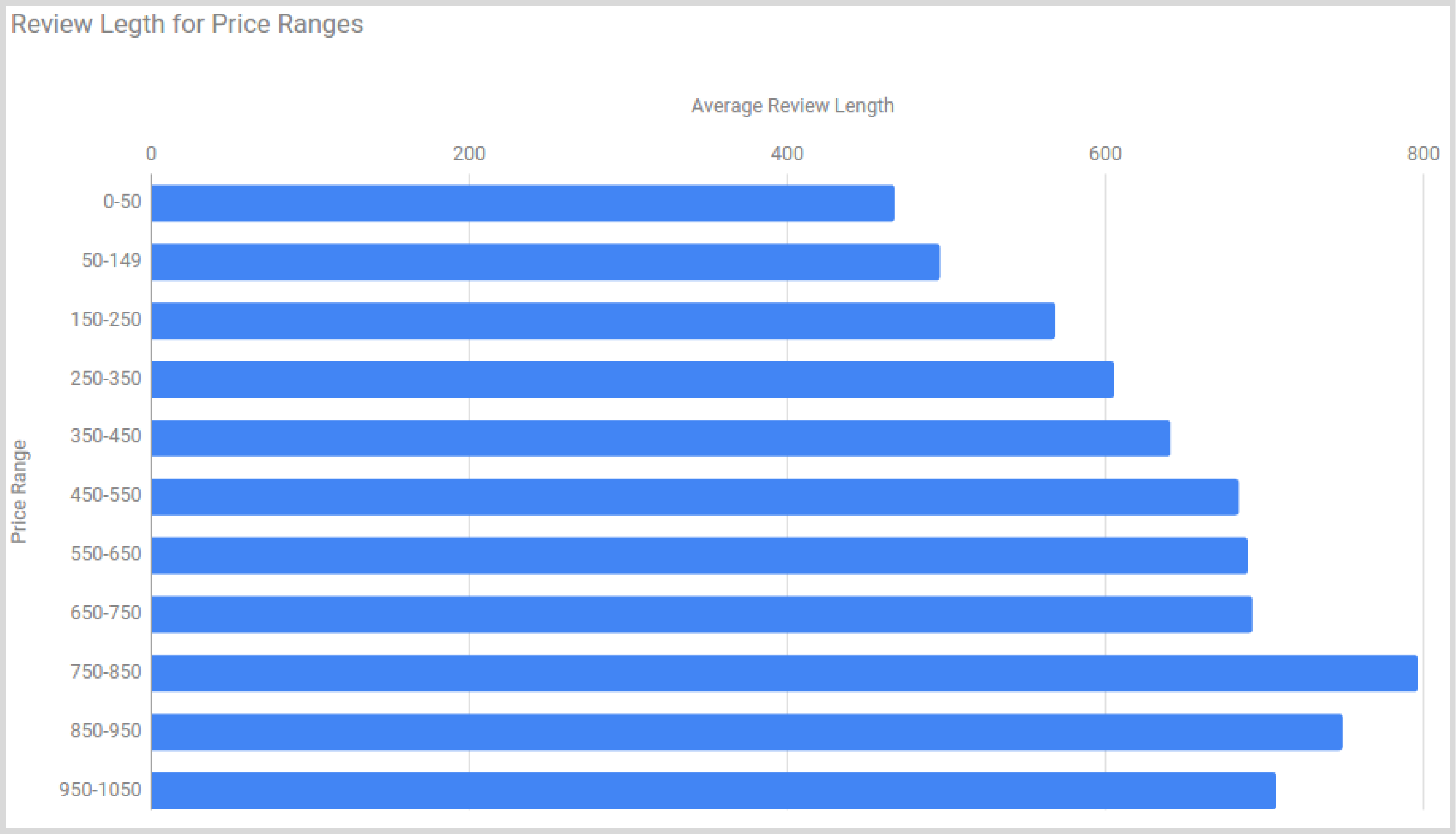


```
sourcetype=all_review earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59 | regex productPrice="[^unknown]" | WHERE productPrice >= 150 AND
productPrice < 250 | eval rw_len=length(reviewText) | stats avg(rw_len)
```

✓ 27,974 events (before 12/31/12 11:59:59.000 PM)

avg(rw_len)
568.980875

Visualization:



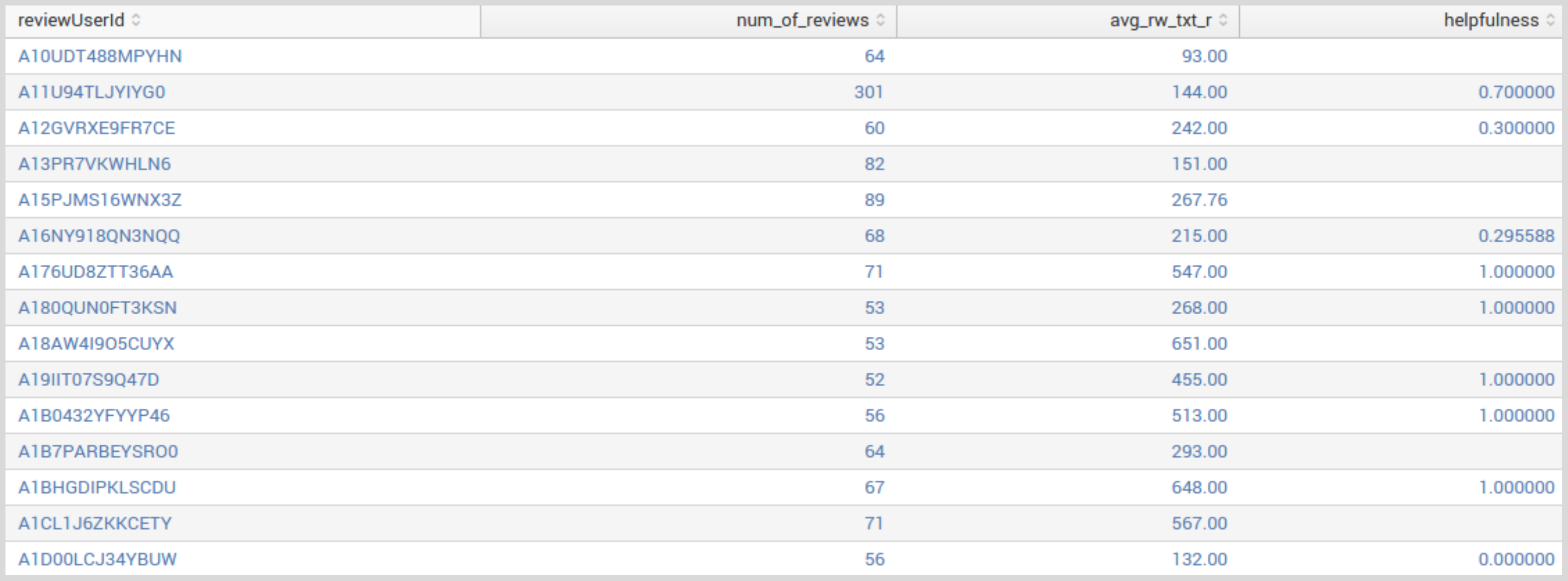
4.14 Users with the lowest rated reviews and helpfulness of the reviews

Objective: Aim is to find the users giving the lowest rating to products and to check weather the reviews are really helpful.

Description: First we calculate the helpfulness of each review by converting the string “1/3” to a division of 1/3. Then we find the users with the lowest average review. Finally we calculate the average helpfulness of the users. As the helpfulness is not a uniform parameters across users. Not all users have a helpfulness rating. But from the results we find we observe that the reviews are indeed helpful and not out of sheer disappointment of the product.

Query:

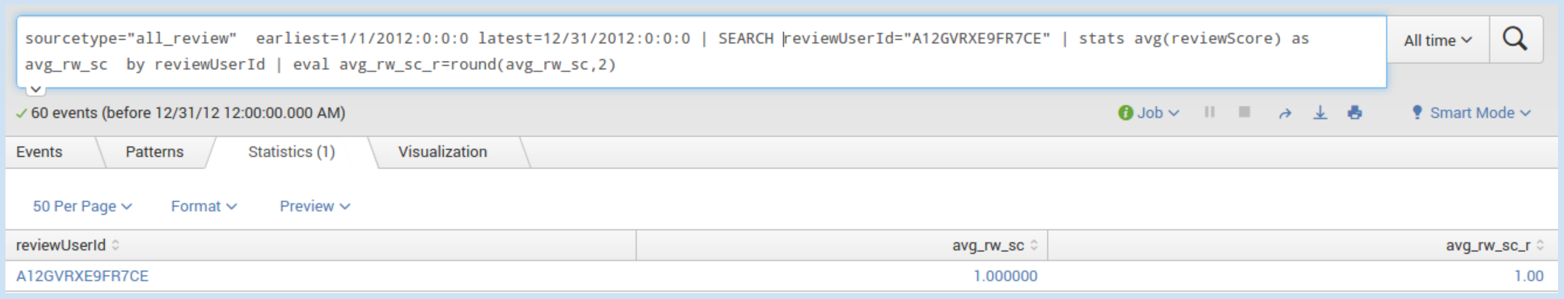
```
sourcetype="all_review"  earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59
| eval len_rw_txt=len(reviewText)
| eval help_split=split(reviewHelpfulness, "/")
| eval q=mvindex(help_split,0)
| eval d=mvindex(help_split,1)
| eval helpfulness=q/d
| eval helpfulness=round(helpfulness,1)
| stats avg(reviewScore) as avg_rw_sc avg(len_rw_txt) as avg_rw_txt count as num_of_reviews
avg(helpfulness) as helpfulness by reviewUserId
| eval avg_rw_sc_r=round(avg_rw_sc,2)
| eval avg_rw_txt_r=round(avg_rw_txt,2)
| WHERE num_of_reviews > 50
| sort avg_rw_sc_r
| head 15
| fields reviewUserId,num_of_reviews,avg_rw_txt_r,helpfulness
```



reviewUserId	num_of_reviews	avg_rw_txt_r	helpfulness
A10UDT488MPYHN	64	93.00	
A11U94TLJYIYG0	301	144.00	0.700000
A12GVRXE9FR7CE	60	242.00	0.300000
A13PR7VKWHLN6	82	151.00	
A15PJMS16WNX3Z	89	267.76	
A16NY918QN3NQQ	68	215.00	0.295588
A176UD8ZTT36AA	71	547.00	1.000000
A180QUN0FT3KSN	53	268.00	1.000000
A18AW4I9O5CUYX	53	651.00	
A19IIT07S9Q47D	52	455.00	1.000000
A1B0432YFYYP46	56	513.00	1.000000
A1B7PARBEYSRO0	64	293.00	
A1BHGDIPKLSCDU	67	648.00	1.000000
A1CL1J6ZKKCETY	71	567.00	
A1D00LCJ34YBUW	56	132.00	0.000000

Verification: For verification we selected user id = A12GVRXE9FR7CE and found output. The output found matched with the output obtained by main query.

```
sourcetype="all_review"  earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59
| SEARCH reviewUserId="A12GVRXE9FR7CE"
| stats avg(reviewScore) as avg_rw_sc  by reviewUserId
| eval avg_rw_sc_r=round(avg_rw_sc,2)
```

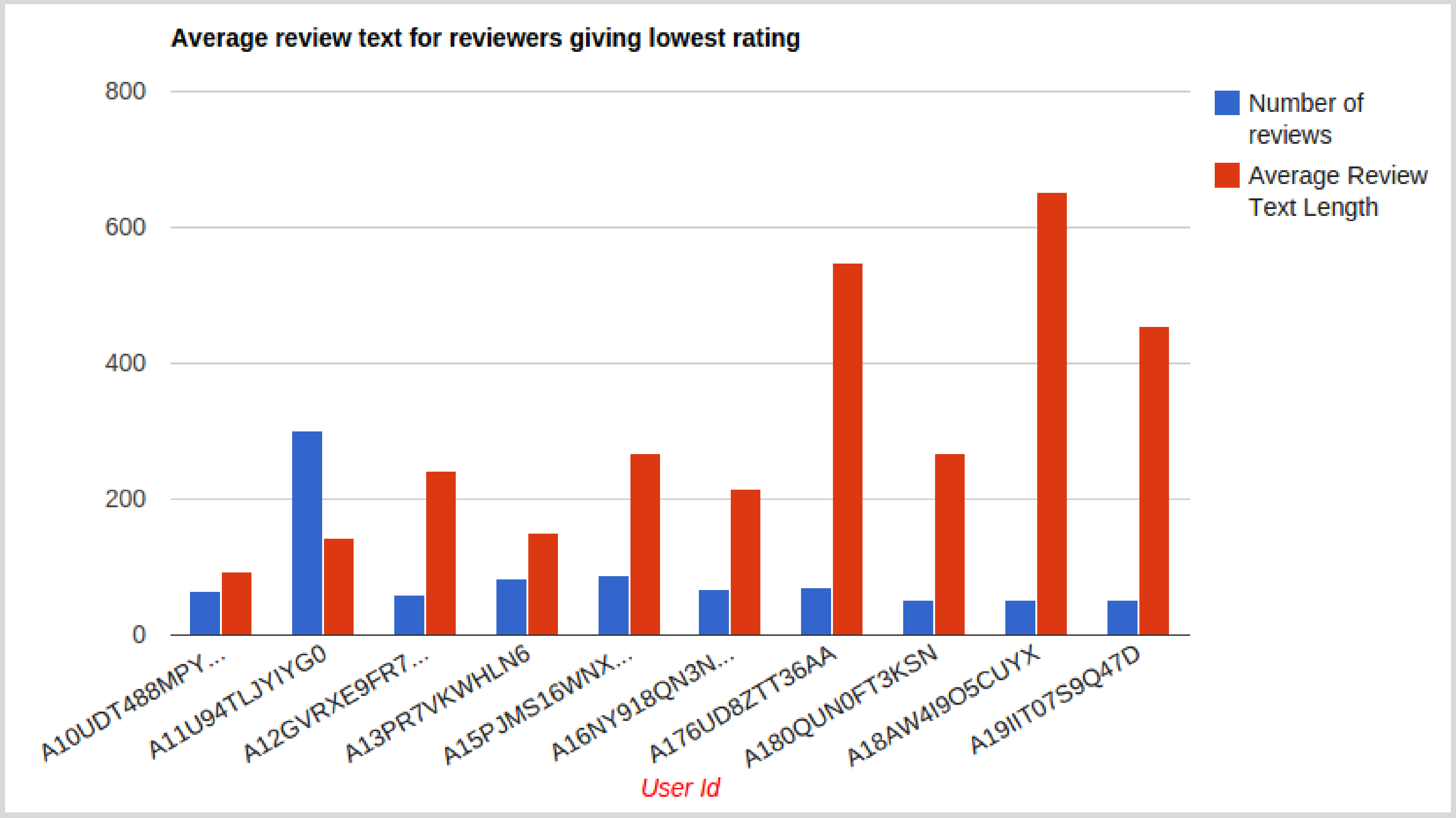


```
sourcetype="all_review"  earliest=1/1/2012:0:0:0 latest=12/31/2012:0:0:0 | SEARCH reviewUserId="A12GVRXE9FR7CE" | stats avg(reviewScore) as
avg_rw_sc  by reviewUserId | eval avg_rw_sc_r=round(avg_rw_sc,2)
```

✓ 60 events (before 12/31/12 12:00:00.000 AM)

reviewUserId	avg_rw_sc	avg_rw_sc_r
A12GVRXE9FR7CE	1.000000	1.00

Visualization:



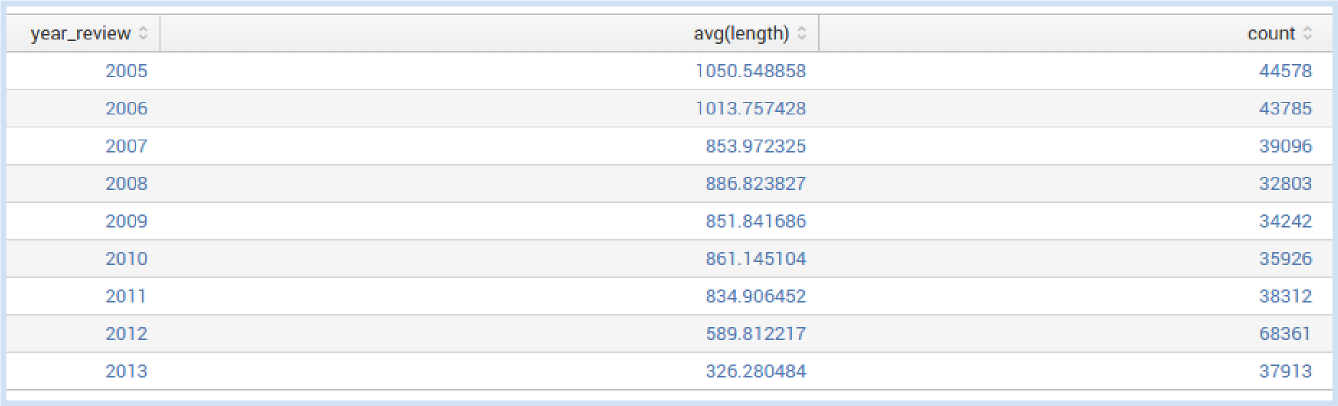
4.15 Review Text length change over the year for books since 2005

Objective: Aim to determine a nature of book reviewers on amazon as Books are the most reviewed category on amazon.

Description: Gathered all reviews since 1-1-2005, then we calculated average length of review text by year. We have observed that book reviews have decreased over the years only until 2012 after which we see a sudden jump in the number of reviews. We also observe that the reviews have become shorter and shorter every year.

Query:

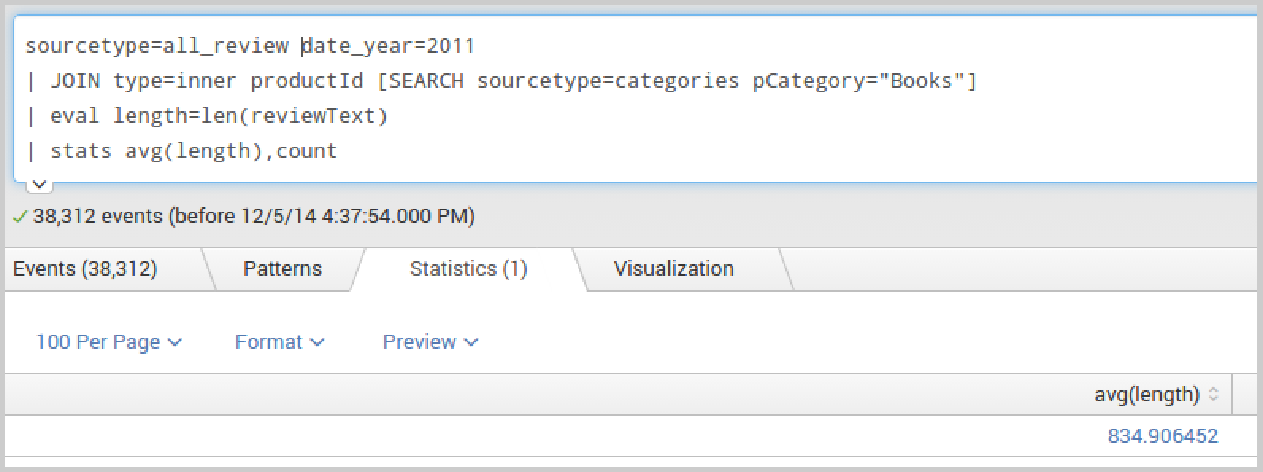
```
sourcetype=all_review earliest=1/1/2005:0:0:0
| rename date_year as year_review
| JOIN type=inner productId [SEARCH sourcetype=categories pCategory="Books"]
| eval length=len(reviewText)
| stats avg(length),count by year_review
```



year_review	avg(length)	count
2005	1050.548858	44578
2006	1013.757428	43785
2007	853.972325	39096
2008	886.823827	32803
2009	851.841686	34242
2010	861.145104	35926
2011	834.906452	38312
2012	589.812217	68361
2013	326.280484	37913

Verification: For verification, we selected date year=2011 and found output. the output matched with the output obtained by main query.

```
sourcetype=all_review
| WHERE date_year=2011
| JOIN type=inner productId [SEARCH sourcetype=categories pCategory="Books"]
| eval length=len(reviewText)
| stats avg(length),count
```

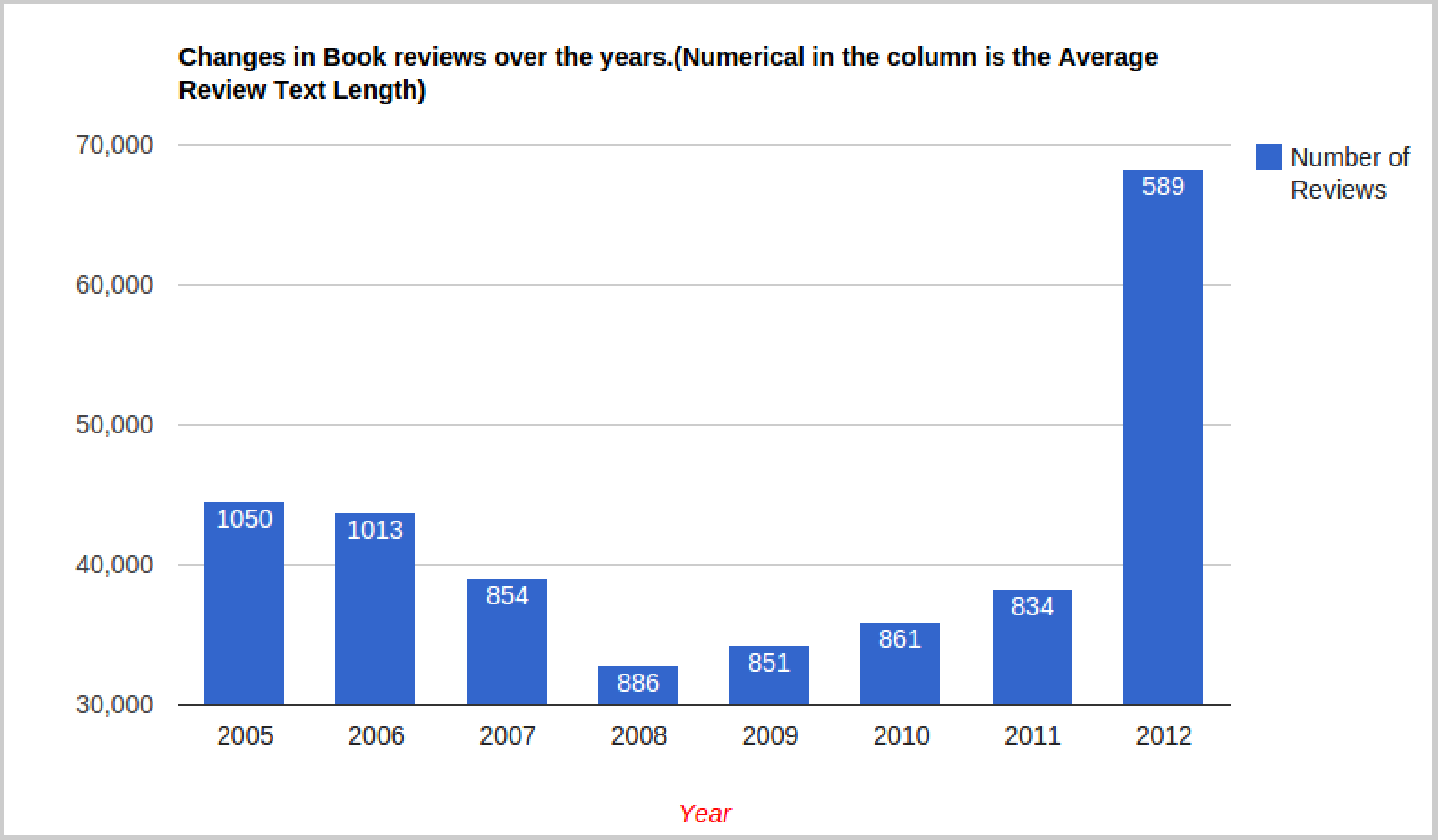


```
sourcetype=all_review date_year=2011
| JOIN type=inner productId [SEARCH sourcetype=categories pCategory="Books"]
| eval length=len(reviewText)
| stats avg(length),count
```

✓ 38,312 events (before 12/5/14 4:37:54.000 PM)

avg(length)
834.906452

Visualization:

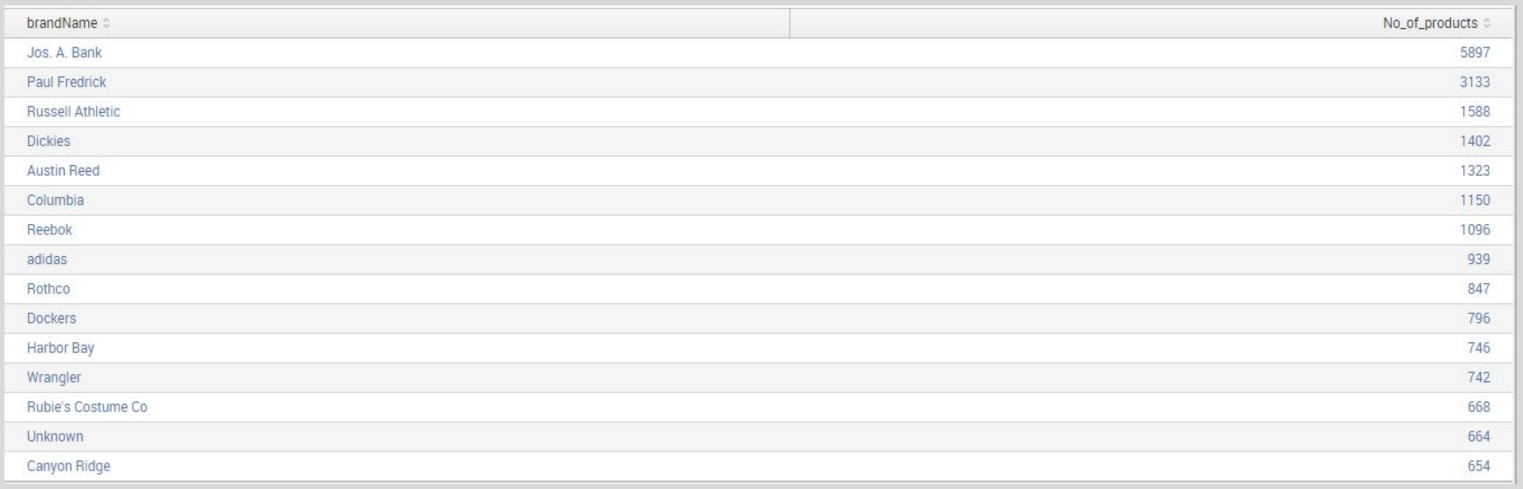


4.16 Which Brands have largest number of products

Objective: To determine the brands that have the most number of products.
Description: There are large number of brands that are sold on amazon. We aim to find the brands which have most products listed on amazon. We observe that some new brand or less known brands make it to the top 15. This shows unpopular brands use amazon as a platform to sell their products as people would have considered their product on amazon rather than on their personal website. On the other hand we see that some of the most popular products don't make it to the list as they may not consider amazon in their business model or might not have as many products as others.

Query:

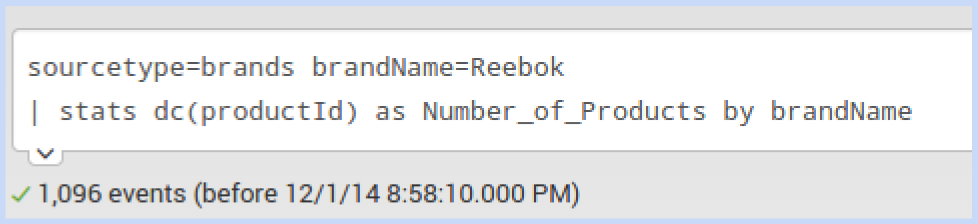
```
sourcetype=brands
| stats dc(productId) as "No_of_products" by brandName
| sort -No_of_products
| fields brandName, No_of_products
| head 15
```



brandName	No_of_products
Jos. A. Bank	5897
Paul Fredrick	3133
Russell Athletic	1588
Dickies	1402
Austin Reed	1323
Columbia	1150
Reebok	1096
adidas	939
Rothco	847
Dockers	796
Harbor Bay	746
Wrangler	742
Rubie's Costume Co	668
Unknown	664
Canyon Ridge	654

Verification: To verify the main query we check the number of products under the reebok brand. The output found matched with the output obtained by main query.

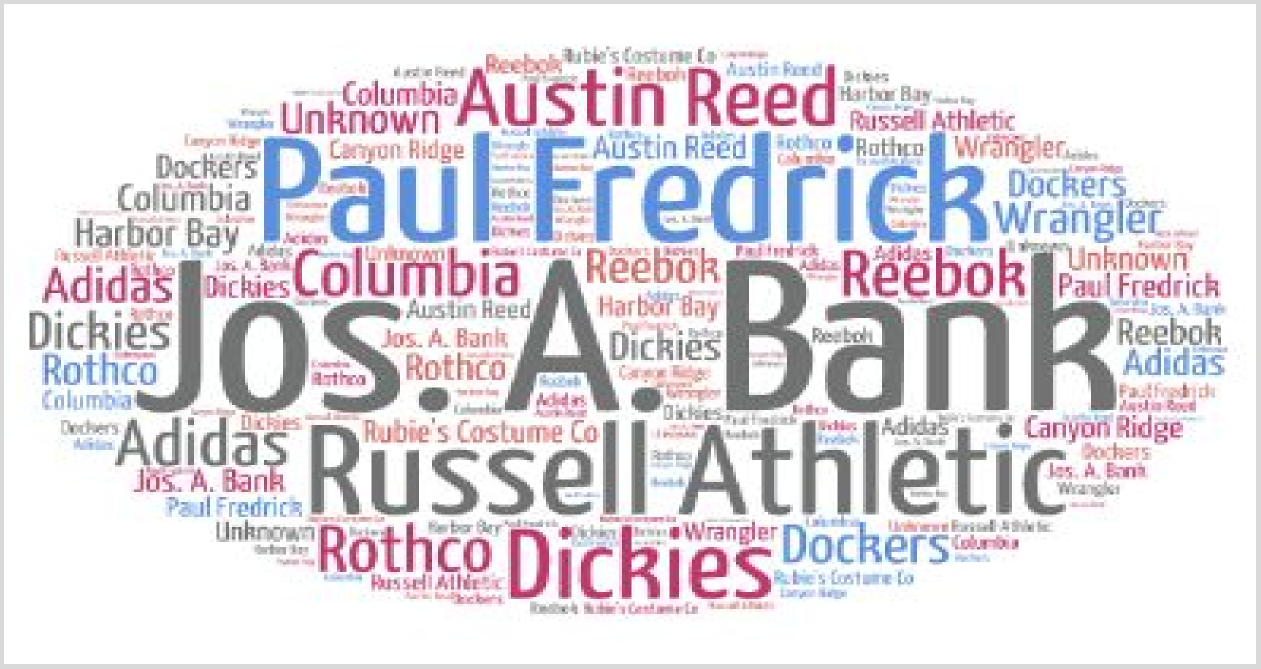
```
sourcetype=brands brandName=Reebok
| stats dc(productId) as Number_of_Products by brandName
```



```
sourcetype=brands brandName=Reebok
| stats dc(productId) as Number_of_Products by brandName
```

✓ 1,096 events (before 12/1/14 8:58:10.000 PM)

Visualization:



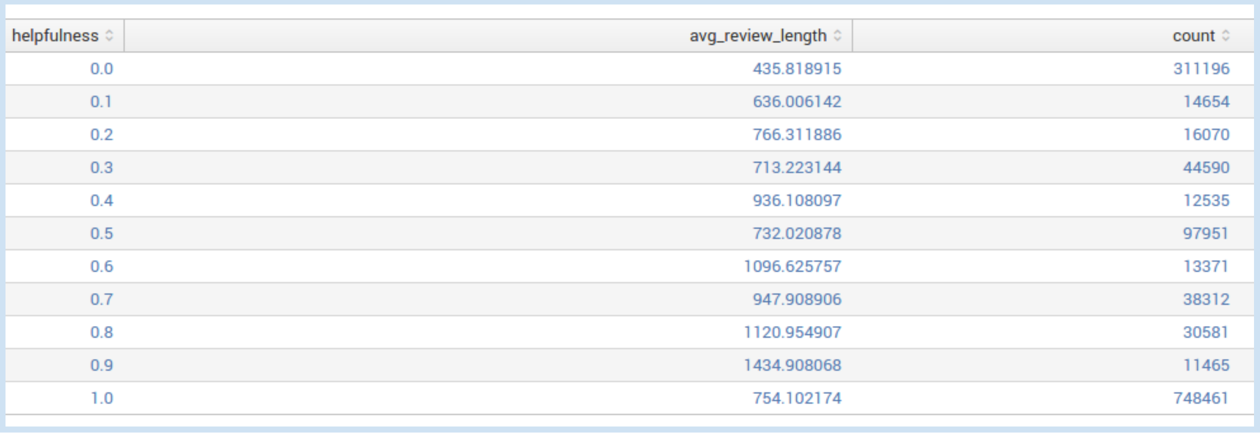
4.17 Review helpfulness relation to review-text length

Objective: Aim is to determine a relation between review length and helpfulness. Are long reviews more helpful than shorter one ?

Description: There is no uniformity in review-helpfulness filed in the reviews. The values vary from 0/0 ,3/5,6/8,etc. To make sense we removed all entries “0/0”. Then for the remaining entries we normalized to values rating bellow 1. So “0.9/1” indicates that the review is helpful,while a “0.1/1” indicates a less helpful review. Later we average review length by helpfulness. We observe a pattern as expected. The longer reviews are the more helpful reviews.

Query:

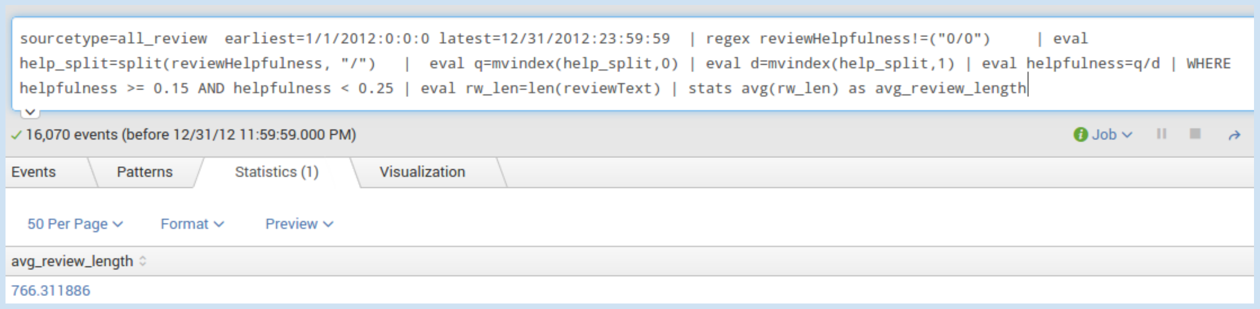
```
sourcetype=all_review  earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59
| regex reviewHelpfulness!=("0/0")
| eval help_split=split(reviewHelpfulness, "/")
|  eval q=mvindex(help_split,0) | eval d=mvindex(help_split,1)
| eval helpfulness=q/d
| eval helpfulness=round(helpfulness,1)
| eval rw_len=len(reviewText)
| stats avg(rw_len) as avg_review_length,count by helpfulness
```



helpfulness	avg_review_length	count
0.0	435.818915	311196
0.1	636.006142	14654
0.2	766.311886	16070
0.3	713.223144	44590
0.4	936.108097	12535
0.5	732.020878	97951
0.6	1096.625757	13371
0.7	947.908906	38312
0.8	1120.954907	30581
0.9	1434.908068	11465
1.0	754.102174	748461

Verification: For verification, we selected helpfulness value that is greater than equal to .15 and less than .25 and found output. The found output matched with the output obtained by main query.

sourcetype=all_review earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59
| regex reviewHelpfulness!=("0/0")
| eval help_split=split(reviewHelpfulness, "/")
| eval q=mvindex(help_split,0) | eval d=mvindex(help_split,1)
| eval helpfulness=q/d
| WHERE helpfulness >= 0.15 AND helpfulness < 0.25
| eval rw_len=len(reviewText)
| stats avg(rw_len) as avg_review_length

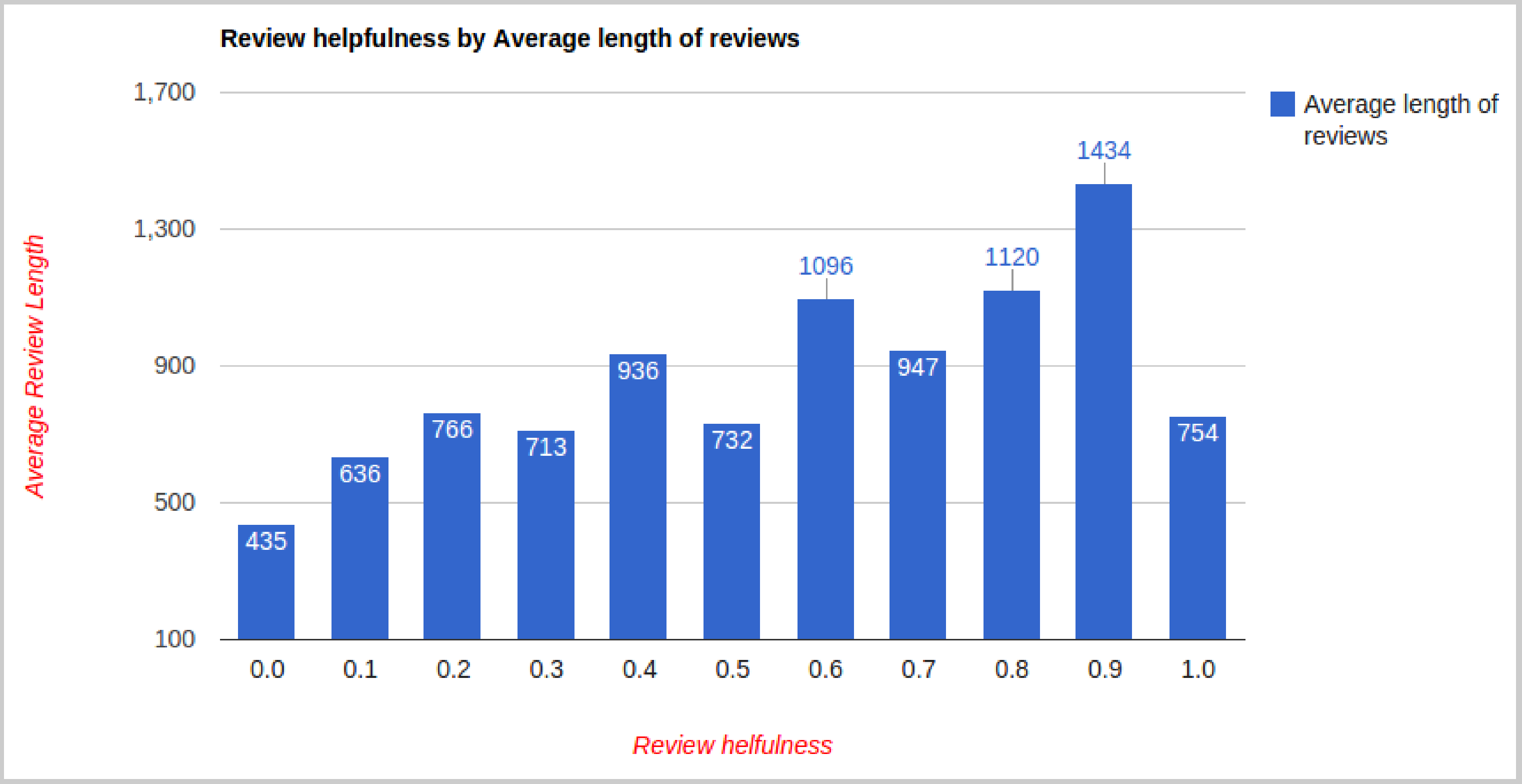


```
sourcetype=all_review  earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59  | regex reviewHelpfulness!=("0/0")     | eval
help_split=split(reviewHelpfulness, "/")   |  eval q=mvindex(help_split,0) | eval d=mvindex(help_split,1) | eval helpfulness=q/d | WHERE
helpfulness >= 0.15 AND helpfulness < 0.25 | eval rw_len=len(reviewText) | stats avg(rw_len) as avg_review_length
```

✓ 16,070 events (before 12/31/12 11:59:59.000 PM)

avg_review_length
766.311886

Visualization :



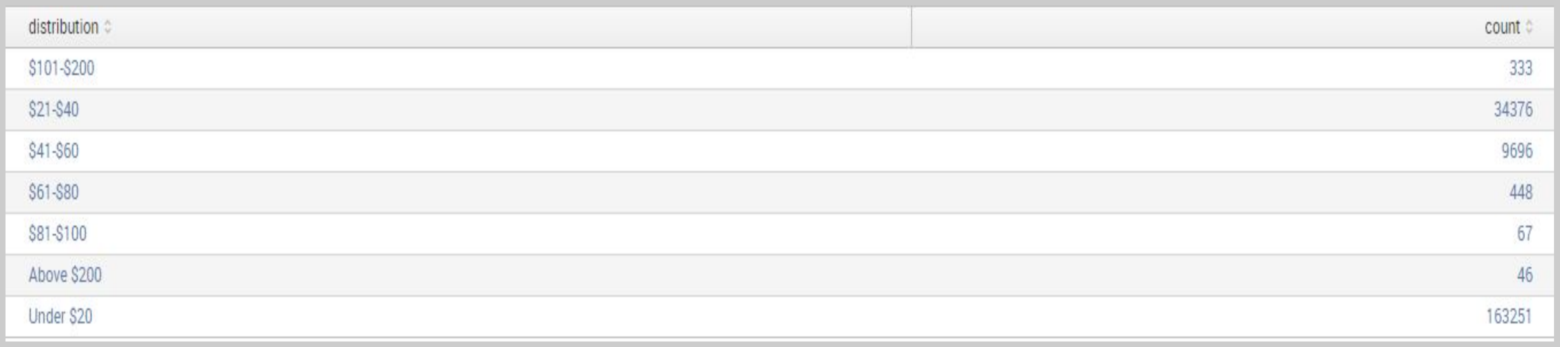
4.18 Distribution of prices of Music reviewed on Amazon

Objective: To determine the price distribution of Music reviewed on Amazon.

Description: There are wide variety of products sold on amazon that are related to music. We filtered all music reviews and then evaluated price distribution using case statement. As expected the cheaper music products are the most reviewed. The more expensive music products are likely the guitar and keyboards which have smaller market on amazon compared to music records.

Query:

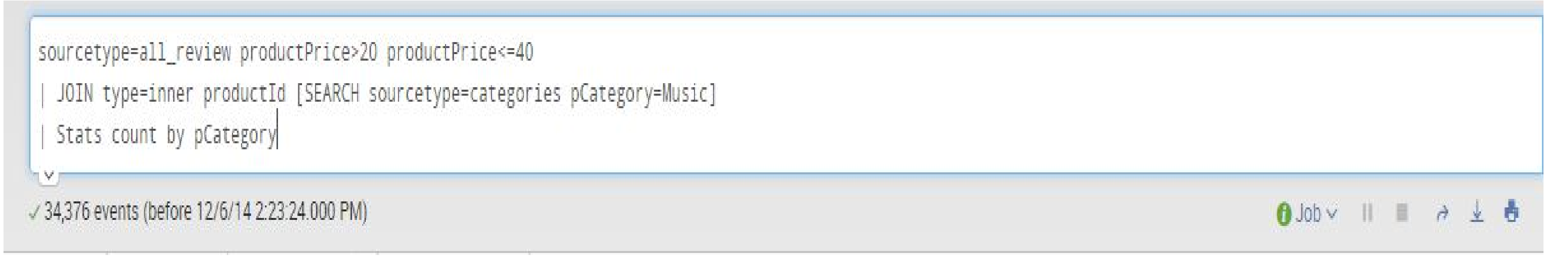
```
sourcetype=all_review
|JOIN type=inner productId [SEARCH sourcetype=categories pCategory=Music]
|eval distribution=case(productPrice<=20,"Under $20",productPrice>20 AND
productPrice<=40,"$21-$40",productPrice>40 AND productPrice<=60,"$41-$60",productPrice>60
AND productPrice<=80,"$61-$80",productPrice>80 AND
productPrice<=100,"$81-$100",productPrice>100 AND
productPrice<=200,"$101-$200",productPrice>200,"Above $200")
| stats count by distribution
```



distribution	count
$101-$200	333
$21-$40	34376
$41-$60	9696
$61-$80	448
$81-$100	67
Above $200	46
Under $20	163251

Verification: Verified Number of products sold in Music category priced between $21 and $40 inclusively. It matched with the number we got in our main query.

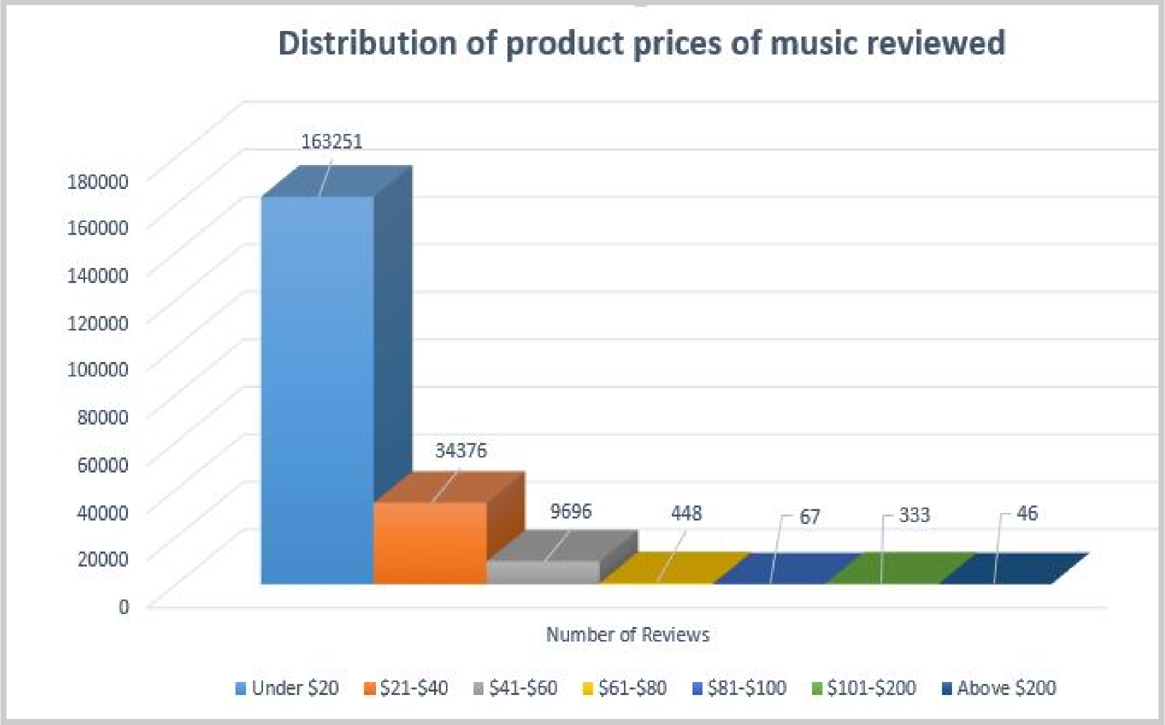
sourcetype=all_review productPrice>20 productPrice<=40
| JOIN type=inner productId [SEARCH sourcetype=categories pCategory=Music]
| Stats count by pCategory



```
sourcetype=all_review productPrice>20 productPrice<=40
| JOIN type=inner productId [SEARCH sourcetype=categories pCategory=Music]
| Stats count by pCategory
```

✓ 34,376 events (before 12/6/14 2:23:24.000 PM)

Visualization :



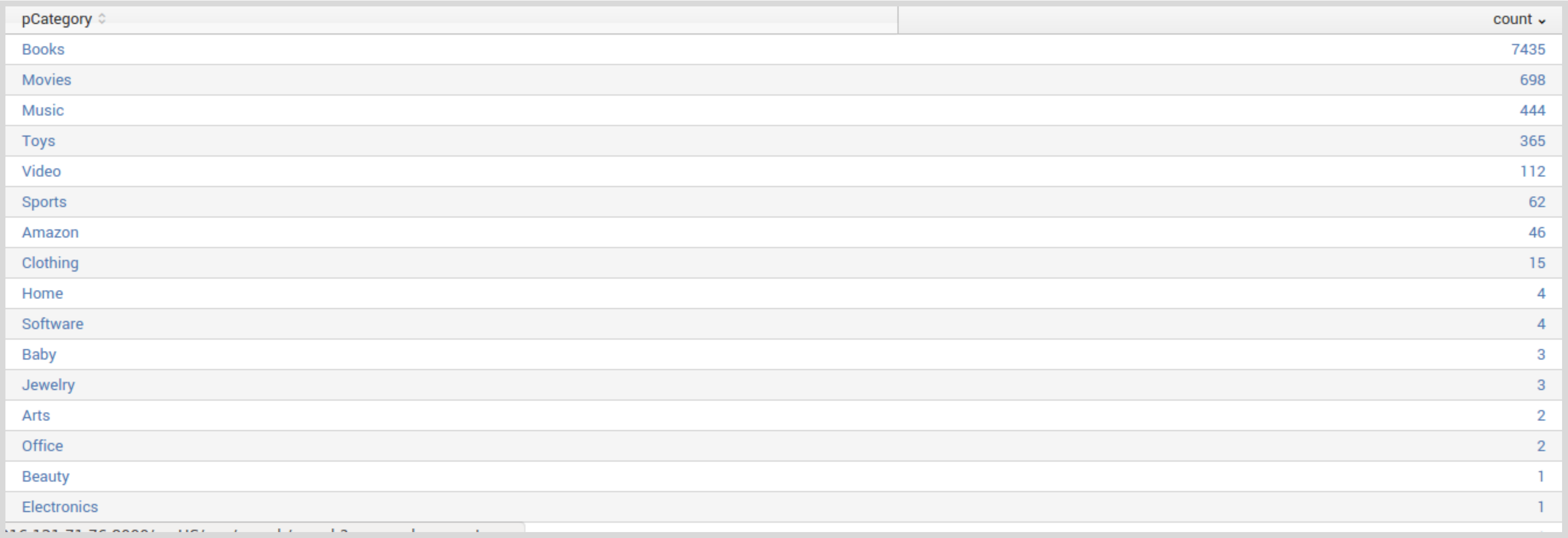
4.19 Product categories for most active user

Objective: Aim to find the products categories for the most active user.

Description: To find the product categories for the most active user, we need to first find the most active user. We have 3 nested sub searches. One to find the most active user. Then to find the products reviewed by the most active user. Then to find the categories of the products reviewed by the most active user. We find the most active user is a book reader. With close to 90 percents reviews are book reviews.

Query:

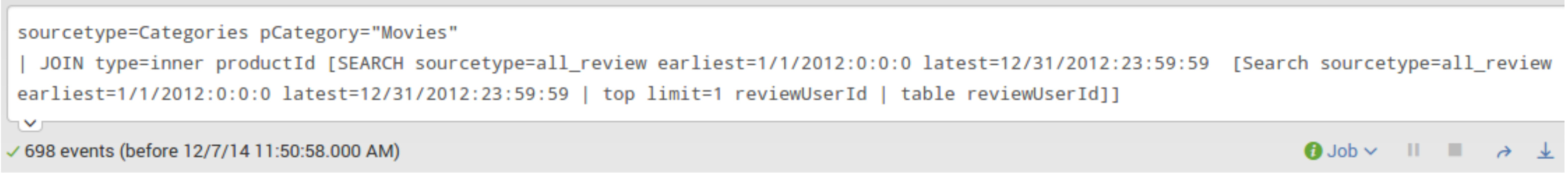
```
sourcetype=Categories
| JOIN type=inner productId [SEARCH sourcetype=all_review earliest=1/1/2012:0:0:0
latest=12/31/2012:23:59:59  [Search sourcetype=all_review earliest=1/1/2012:0:0:0
latest=12/31/2012:23:59:59 | top limit=1 reviewUserId | table reviewUserId]]
| stats count by pCategory
```



pCategory	count
Books	7435
Movies	698
Music	444
Toys	365
Video	112
Sports	62
Amazon	46
Clothing	15
Home	4
Software	4
Baby	3
Jewelry	3
Arts	2
Office	2
Beauty	1
Electronics	1

Verification: To verify the query we modify the third sub search to only join products for movies. This way verify whether the count of movies is similar to the main query.

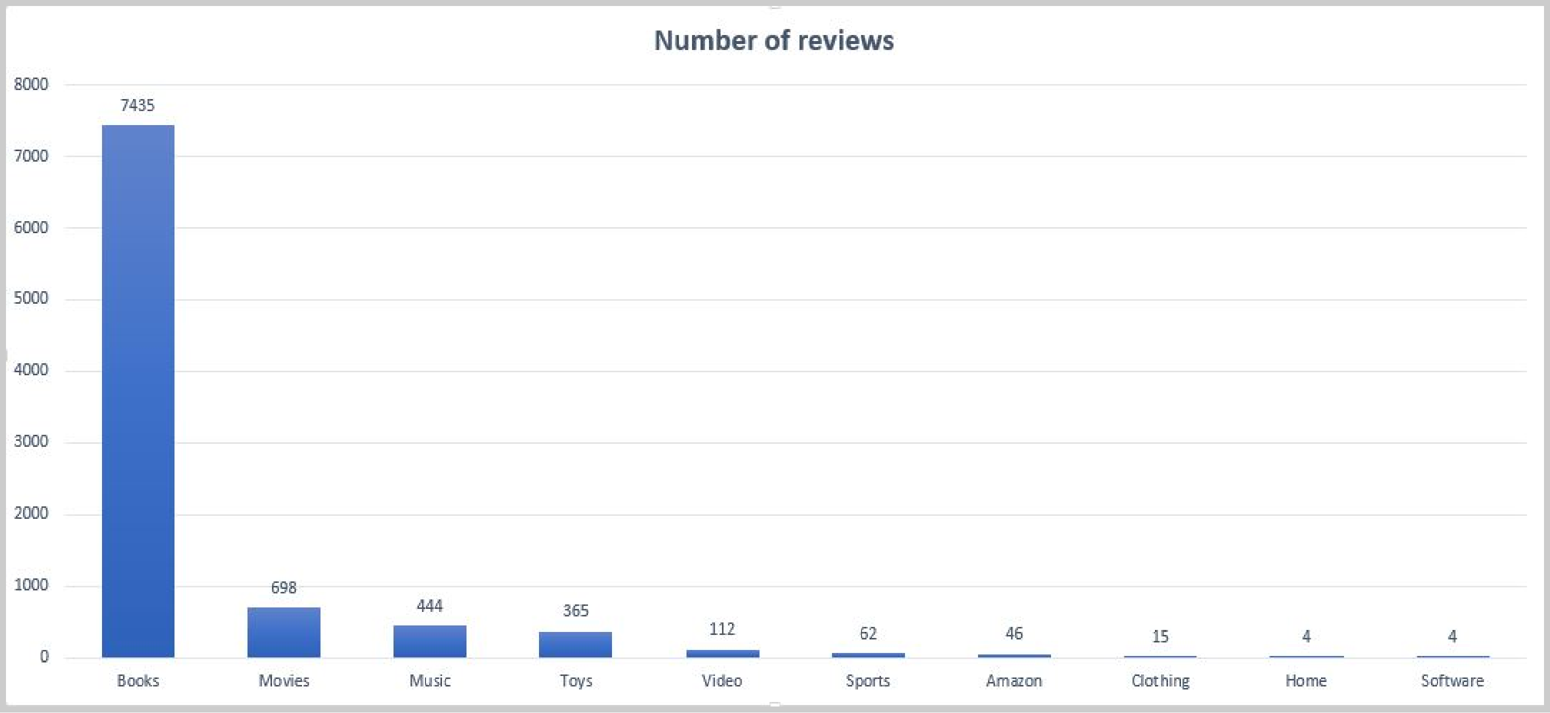
```
sourcetype=Categories pCategory="Movies"
| JOIN type=inner productId [SEARCH sourcetype=all_review earliest=1/1/2012:0:0:0
latest=12/31/2012:23:59:59  [Search sourcetype=all_review earliest=1/1/2012:0:0:0
latest=12/31/2012:23:59:59 | top limit=1 reviewUserId | table reviewUserId]]
```



```
sourcetype=Categories pCategory="Movies"
| JOIN type=inner productId [SEARCH sourcetype=all_review earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59  [Search sourcetype=all_review
earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59 | top limit=1 reviewUserId | table reviewUserId]]
```

✓ 698 events (before 12/7/14 11:50:58.000 AM)

Visualization:



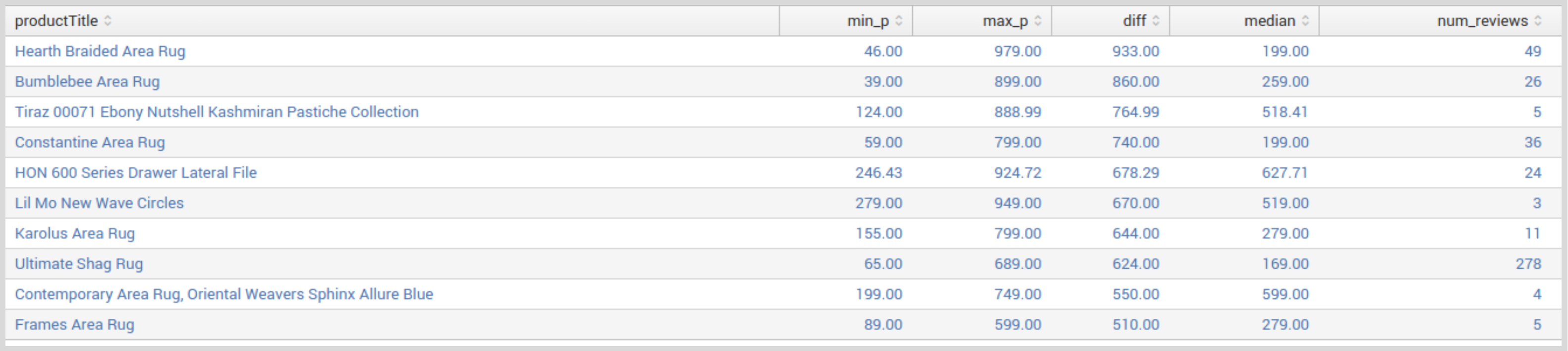
4.20 Most price variant product on Amazon

Objective: To determine the product on amazon with the most price difference.

Description: A product on amazon has many listing by different vendors resulting in different prices. We aim to find the product which has the maximum difference between minimum price and maximum price for a product. We first remove the reviews which have no price listed. Now we find the max and min price for a product and calculate the difference. We also calculate the median price for the product. In the final step we sort product by price difference. We observe that although the maximum price is very high for some products the median is way lower. These product might be sold from outside the US and can be used for money laundering may be.

Query:

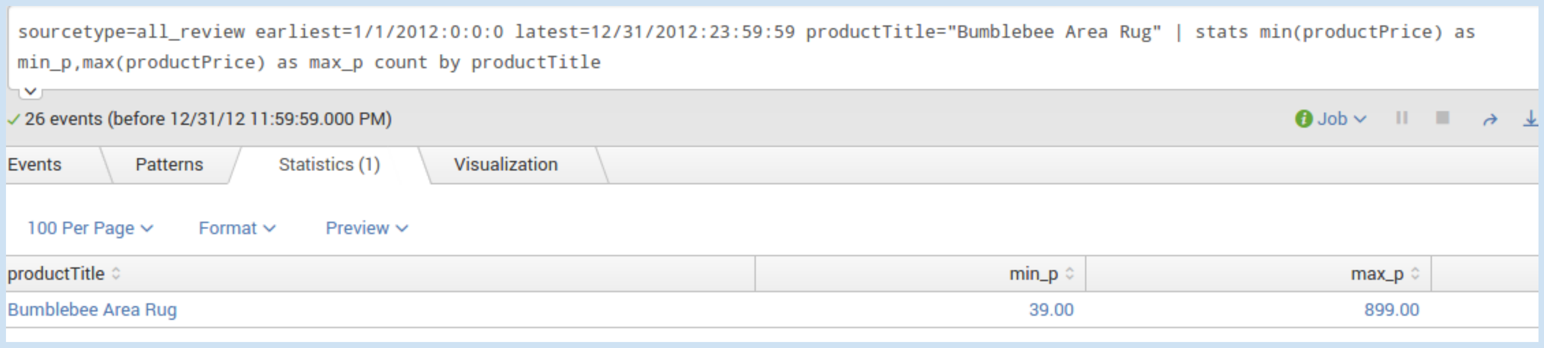
sourcetype=all_review earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59
| regex productPrice="[^unknown]"
| stats min(productPrice) as min_p,max(productPrice) as max_p,count as num_reviews,median(productPrice) as median by productTitle
| eval diff=max_p-min_p
| sort -diff
| fields productTitle,min_p,max_p,diff,median,num_reviews | head 10



productTitle	min_p	max_p	diff	median	num_reviews
Hearth Braided Area Rug	46.00	979.00	933.00	199.00	49
Bumblebee Area Rug	39.00	899.00	860.00	259.00	26
Tiraz 00071 Ebony Nutshell Kashmiran Pastiche Collection	124.00	888.99	764.99	518.41	5
Constantine Area Rug	59.00	799.00	740.00	199.00	36
HON 600 Series Drawer Lateral File	246.43	924.72	678.29	627.71	24
Lil Mo New Wave Circles	279.00	949.00	670.00	519.00	3
Karolus Area Rug	155.00	799.00	644.00	279.00	11
Ultimate Shag Rug	65.00	689.00	624.00	169.00	278
Contemporary Area Rug, Oriental Weavers Sphinx Allure Blue	199.00	749.00	550.00	599.00	4
Frames Area Rug	89.00	599.00	510.00	279.00	5

Verification: Here we verify the query by manually calculating the difference between maximum and minimum price for the "Bumblebee Area Rug".The output found matched with the output obtained by main query.

sourcetype=all_review earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59
productTitle="Bumblebee Area Rug"
| stats min(productPrice) as min_p,max(productPrice) as max_p count by productTitle

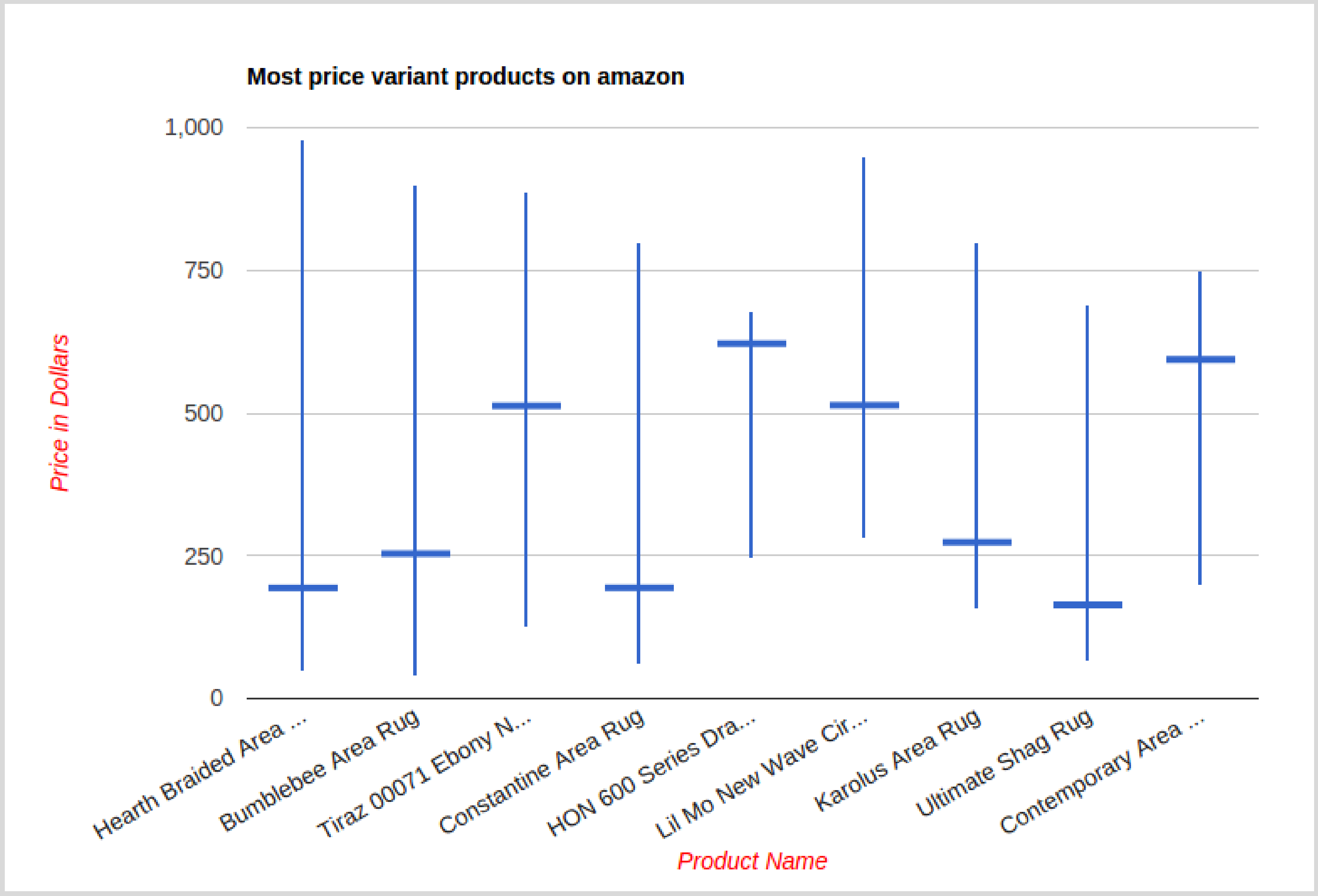


```
sourcetype=all_review earliest=1/1/2012:0:0:0 latest=12/31/2012:23:59:59 productTitle="Bumblebee Area Rug" | stats min(productPrice) as
min_p,max(productPrice) as max_p count by productTitle
```

✓ 26 events (before 12/31/12 11:59:59.000 PM)

productTitle	min_p	max_p
Bumblebee Area Rug	39.00	899.00

Visualization : The horizontal bar for each product indicates the median price of the product.


Most price variant products on amazon
Price in Dollars
Hearth Braided Area ...
Bumblebee Area Rug
Tiraz 00071 Ebony N...
Constantine Area Rug
HON 600 Series Dra...
Lil Mo New Wave Cir...
Karolus Area Rug
Ultimate Shag Rug
Contemporary Area ...
Product Name