ACE
ENGINEERING COLLEGE
INDEX

Experiment – 1: Preprocessing text document using NLTK of Python:

a. Stopword elimination

```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

def Stopword(sentence):
        stopword =set(stopwords.words('english'))
        word_tokens = word_tokenize(sentence)
        filtered_sentence = [w for w in word_tokens ifnot w.lower() in stopword]
        filtered_sentence = []
        for w in word_tokens:
                if w notin stopword:
                        filtered_sentence.append(w)
        #print(word_tokens)
        return(filtered_sentence)

sentence =input()
print()
print()
print(Stopword(sentence))
        This is a sample sentence showing off the stop words filtration.
```

OUTPUT

```
['This', 'sample', 'sentence', 'showing', 'stop', 'words', 'filtration', '.']
```

b. Stemming

```
from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize
ss = PorterStemmer()#(language = 'english')
sentence =input()
words = word_tokenize(sentence)
print()
print()
for w in words:
   print(w,":",ss.stem(w))
likes liked likely liking
```

OUTPUT:

```
likes  :  like
liked  :  like
likely : like
liking : like
```

c. Lemmatization

```
# import these modules
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize
import nltk
lemmatizer = WordNetLemmatizer()
sentence =input()
words = word_tokenize(sentence)
for w in words:
        print(w+":"+lemmatizer.lemmatize(w))
```

OUTPUT

```
Rocks  books
Rocks:Rocks
books:book
```

d. POS tagging

```
import re
def custom_word_tokenize(text):
    tokens = re.findall(r"\b\w+\b|\S", text)
    return tokens
text = "Laughed my heart out after ages!! Great acting by both the actors rest of the cast was also
great! Anushka was so pretty and loved the storyline dialogues and timing! I would go watch the
movie second time in theaters which is very rare for me! I loved it as much as Jaatiratnalu
Naveen polishetty you gotta do more movies!! Amazing acting!"
tokens = custom_word_tokenize(text)
print(tokens)
#OUTPUT:
['Laughed', 'my', 'heart', 'out', 'after', 'ages', '!', '!', 'Great', 'acting', 'by', 'both', 'the', 'actors', 'rest', 'of',
'the', 'cast', 'was', 'also', 'great', '!', 'Anushka', 'was', 'so', 'pretty', 'and', 'loved', 'the', 'storyline',
'dialogues', 'and', 'timing', '!', 'I', 'would', 'go', 'watch', 'the', 'movie', 'second', 'time', 'in', 'theaters',
'which', 'is', 'very', 'rare', 'for', 'me', '!', 'I', 'loved', 'it', 'as', 'much', 'as', 'Jaatiratnalu', ' ', ' ', 'Naveen',
'polishetty', 'you', 'gotta', 'do', 'more', 'movies', '!', '!', 'Amazing', 'acting', '!']
import re
def custom_pos_tag(text):
    sentences = re.split(r'(?<=[.!?])\s+', text)
    pos_tags = []
articles=['a','an','the','A','An','The']pronoun=['he','him','his','She','her','This','this','That','that','Them','th
em','It','it','I','i','me','you','yours','yourself','Your','my','mine','myself']
    adjective
    for sentence in sentences:
        words = sentence.split()
        for word in words:
            if word in articles:
                pos_tags.append((word,"Determiner"))
            elif word in pronoun:
                pos_tags.append((word,'Pronoun'))
            elif word[0].isupper():
```

```
            pos_tags.append((word, "Noun"))
        elif word.endswith("ed"):
            pos_tags.append((word, "Past Tense Verb"))
        elif word.endswith("ing"):
            pos_tags.append((word, "Adjective"))
        else:
            pos_tags.append((word, "Noun"))
    return pos_tags
text = "Laughed my heart out after ages!! Great acting by both the actors rest of the cast was also great! Anushka was so pretty and loved the storyline dialogues and timing! I would go watch the movie second time in theaters which is very rare for me! I loved it as much as Jaatiratnalu Naveen polishetty you gotta do more movies!! Amazing acting!"
tags = custom_pos_tag(text)
for word, tag in tags:
    print(f"{word}: {tag}")
```

#OUTPUT

```
Laughed: Noun
my: Pronoun
heart: Noun
out:  Noun
after: Noun
ages!!: Noun
Great: Noun
acting: Adjective
by: Noun
both: Noun
the: Determiner
actors: Noun
rest: Noun
of: Noun
the: Determiner
cast:  Noun
was:  Noun
also: Noun
great!: Noun
```

Anushka: Noun

was: Noun

so: Noun

pretty: Noun

and: Noun

loved: Past Tense Verb

the: Determiner

storyline: Noun

dialogues: Noun

and: Noun

timing!: Noun

I: Pronoun

would: Noun

go: Noun

watch: Noun

the: Determiner

movie: Noun

second: Noun

time: Noun

in: Noun

theaters: Noun

which: Noun

is: Noun

very: Noun

rare: Noun

for: Noun

me!: Noun

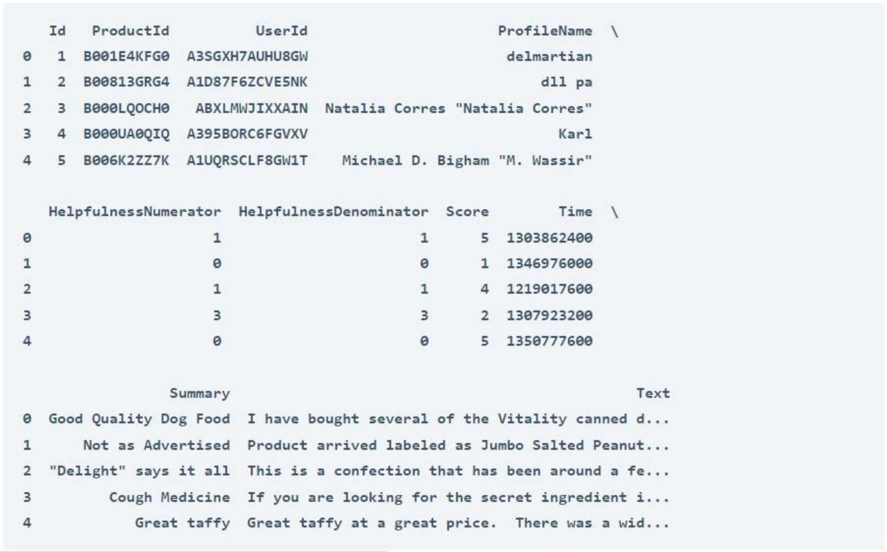
Experiment – 2: Sentiment analysis on customer review on products

Amazon Product Reviews Sentiment Analysis with Python

Amazon is an American multinational corporation that focuses on e-commerce, cloud computing, digital streaming, and artificial intelligence products. But it is mainly known for its e-commerce platform which is one of the biggest online shopping platforms today. There are somany customers buying products from Amazon that today Amazon earns an average of $ 638.1 million per day. So having such a large customer base, it will turn out to be an amazing data science project if we can analyze the sentiments of Amazon product reviews. So, in this article, I will walk you through the task of Amazon Product Reviews Sentiment Analysis with Python.

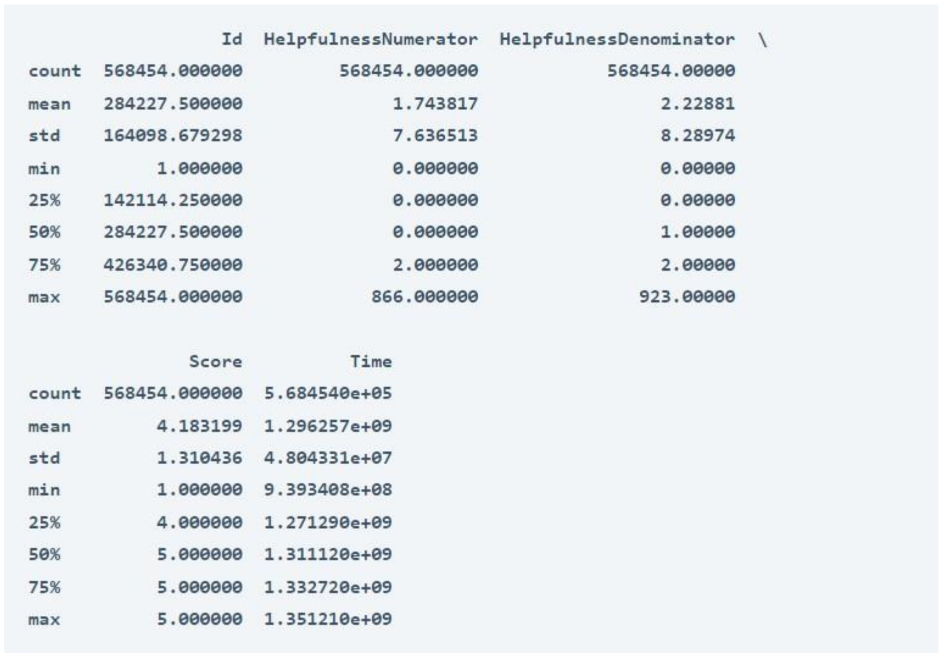
The dataset we're using for the task of Amazon product reviews sentiment analysis was downloaded from Kaggle. This dataset contains the product reviews of over 568,000 customers who have purchased products from Amazon. So let's start this task by importing the necessary Python libraries and the dataset:

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from nltk.sentiment.vader import SentimentIntensityAnalyzer
sentiments = SentimentIntensityAnalyzer()
data = pd.read_csv("Reviews.csv")
print(data.head())
```



```
   Id   ProductId          UserId                      ProfileName  \
0   1  B001E4KFG0  A3SGXH7AUHU8GW                       delmartian
1   2  B00813GRG4  A1D87F6ZCVE5NK                           dll pa
2   3  B000LQOCH0   ABXLMWJIXXAIN  Natalia Corres "Natalia Corres"
3   4  B000UA0QIQ  A395BORC6FGVXV                             Karl
4   5  B006K2ZZ7K  A1UQRSCLF8GW1T    Michael D. Bigham "M. Wassir"

   HelpfulnessNumerator  HelpfulnessDenominator  Score        Time  \
0                     1                       1      5  1303862400
1                     0                       0      1  1346976000
2                     1                       1      4  1219017600
3                     3                       3      2  1307923200
4                     0                       0      5  1350777600

                 Summary                                               Text
0  Good Quality Dog Food  I have bought several of the Vitality canned d...
1      Not as Advertised  Product arrived labeled as Jumbo Salted Peanut...
2  "Delight" says it all  This is a confection that has been around a fe...
3         Cough Medicine  If you are looking for the secret ingredient i...
4            Great taffy  Great taffy at a great price.  There was a wid...
```

print(data.describe())



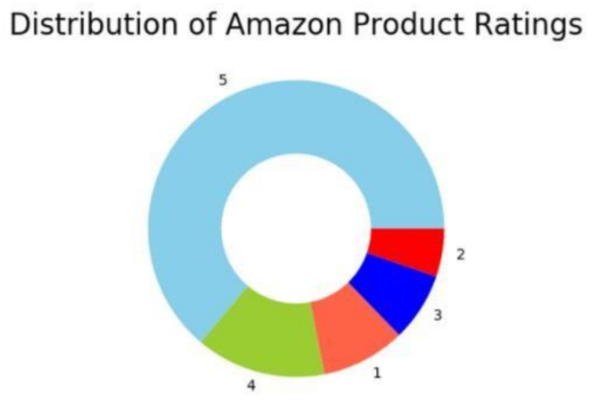
```
                  Id  HelpfulnessNumerator  HelpfulnessDenominator  \
count  568454.000000         568454.000000            568454.00000
mean   284227.500000              1.743817                 2.22881
std    164098.679298              7.636513                 8.28974
min         1.000000              0.000000                 0.00000
25%    142114.250000              0.000000                 0.00000
50%    284227.500000              0.000000                 1.00000
75%    426340.750000              2.000000                 2.00000
max    568454.000000            866.000000               923.00000

               Score          Time
count  568454.000000  5.684540e+05
mean        4.183199  1.296257e+09
std         1.310436  4.804331e+07
min         1.000000  9.393408e+08
25%         4.000000  1.271290e+09
50%         5.000000  1.311120e+09
75%         5.000000  1.332720e+09
max         5.000000  1.351210e+09
```

As this dataset is very large, it contains some missing values, so remove all the rows containing the missing values:

```
data = data.dropna()
```

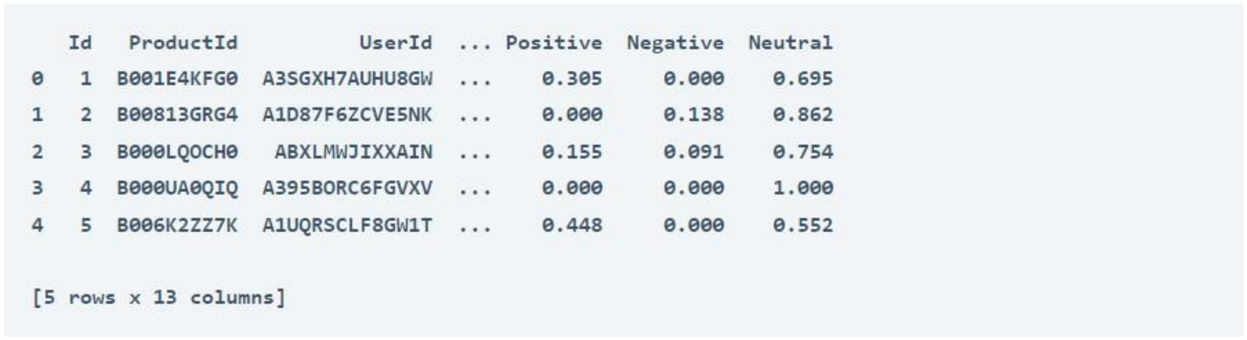
The Score column of this dataset contains the ratings that customers have given to the product based on their experience with the product. So let's take a look at the rating breakdown to see how most customers rate the products they buy from Amazon:

```
ratings = data["Score"].value_counts()
numbers = ratings.index
quantity = ratings.values
custom_colors = ["skyblue", "yellowgreen", 'tomato', "blue", "red"]
plt.figure(figsize=(10, 8))
plt.pie(quantity, labels=numbers, colors=custom_colors)
central_circle = plt.Circle((0, 0), 0.5, color='white')
fig = plt.gcf()
fig.gca().add_artist(central_circle)
plt.rc('font', size=12)
plt.title("Distribution of Amazon Product Ratings", fontsize=20)
plt.show()
```



According to the figure above, more than half of people rated products they bought from Amazon with 5 stars, which is good. Now, we're going to add three more columns to this dataset as Positive, Negative, and Neutral by calculating the sentiment scores of the customer reviews mentioned in the Text column of the dataset:

```
sentiments = SentimentIntensityAnalyzer()
data["Positive"] = [sentiments.polarity_scores(i)["pos"] for i in data["Text"]]
data["Negative"] = [sentiments.polarity_scores(i)["neg"] for i in data["Text"]]
data["Neutral"] = [sentiments.polarity_scores(i)["neu"] for i in data["Text"]]
print(data.head())
```



```
   Id   ProductId          UserId  ... Positive  Negative  Neutral
0   1  B001E4KFG0  A3SGXH7AUHU8GW  ...    0.305     0.000    0.695
1   2  B00813GRG4  A1D87F6ZCVE5NK  ...    0.000     0.138    0.862
2   3  B000LQOCH0   ABXLMWJIXXAIN  ...    0.155     0.091    0.754
3   4  B000UA0QIQ  A395BORC6FGVXV  ...    0.000     0.000    1.000
4   5  B006K2ZZ7K  A1UQRSCLF8GW1T  ...    0.448     0.000    0.552

[5 rows x 13 columns]
```

Now we can see how most people rated the products they bought from Amazon:

```
x = sum(data["Positive"])
y = sum(data["Negative"])
z = sum(data["Neutral"])
def sentiment_score(a, b, c):
    if (a>b) and (a>c):
        print("Positive  ")
    elif (b>a) and (b>c):
        print("Negative  ")
    else:
        print("Neutral  ")
sentiment_score(x, y, z)
```

Output:

```
Neutral ⍰
```

So, most people are neutral when submitting their experiences with the products they have purchased from Amazon. Now let’s see the total of all sentiment scores:

```
print("Positive: ", x)
print("Negative: ", y)
print("Neutral: ", z)
```

Output:

```
Positive:  109328.1269999992
Negative:  24033.022999999564
Neutral:  435043.95799998916
```

Experiment – 3: Search engine optimization- implement Spamdexing

Using Python scripts to analyse SEO (Search Engine Optimization) and broken links on your site

Python is all about automating repetitive tasks, leaving more time for your other Search Engine Optimization (SEO) efforts. Not many SEOs use Python for their problem-solving, even though it could save you a lot of time and effort. Python, for example, can be used for the following tasks:

- Data extraction
- Preparation
- Analysis & visualization
- Machine learning
- Deep learning

We'll be focussing mostly on data extraction and analysis in this article. The required modules will be indicated for each script.

Python SEO analyzer

A really useful script for analyzing your website is called **'SEO analyzer'**. It's an all round website crawler that analyses the following information:

- Word count
- Page Title
- Meta Description
- Keywords on-page
- Warnings
- Missing title
- Missing description
- Missing image alt-text

This is great for a quick analysis of your basic SEO problems. As page title, meta descriptions and on-page keywords are important ranking factors, this script is perfect for gaining a clear picture of any problems that might be in play.

Using the SEO analyzer

After having installed the necessary modules (BeautifulSoup 4 + urllib2) for this script and having updated your Python to version 3.4+, you are technically ready to use this script. Json or working variants, however, can be useful for exporting the data you gain from the SEO analyser. After having installed the script, these are the commands you can use:

seoanalyze http://internetvergelijnk.nl/

seoanalyze https://telefoonvergelijk.nl --sitemap https://telefoonvergelijk.nl/sitemap_index.xml

As seen in the examples above, for both internetvergelijk and telefoonvergelijk , it's possible to either crawl the website, or the XML sitemap of a website in order to do an SEO analysis. Another option is to generate HTML output from the analysis instead of using json. This can be done through the following command:

seoanalyze http://internetvergelijk.nl/ --output-format-html

If you have installed json and want to export the data, use the following command:

from seoanalyzer import analyse output = analyse(site, sitemap) print(output)

You can also choose for an alternative path, running the analysis as a script, as seen in the example below:

This will export the file into a html after having run the –output-format html script. This seoanalyze script is great for optimizing your page titles, meta descriptions, images and on-page keywords. It's also a lot faster than Screaming Frog, so if you're only looking for this information, running the seoanalyze script is more efficient.

Link status analyser

Another way to use Python for Search Engine Optimization is by using a script that crawls your website and analyses your URL status codes. This script is called Pylinkvalidator and can be found here). All it requires is BeautifulSoup if you're running it with Python 3.x. If you're runninga 2.x version like 2.6 or 2.7, you should not need BeajutifulSoup.

In order to speed up the crawling, however, it might be useful to install the following libraries:

1) lxml – Speeds up the crawling of HTML pages (requires C libraries) 1) gevent – enables pylinkvalidator to use green threads 1) cchardet – Speeds up document encoding detection

Do keep this in mind, they could be very useful for crawling larger websites, and just to enhance the link status analyser.

What this script essentially does, is crawl the entire URL structure of a website in order to analyse the status codes of each and every URL. This makes it a very long process for bigger websites, hence the recommendation of using the optional libraries to speed this up.

Using the link status analyser

Pylinkvalidator has a ton of different usage options for usage. For example, you can:

- Show progress
- Crawl the website and pages belonging to another host
- Only crawl a single page and the pages it links to
- Only crawl links, ignore others (images, stylesheets, etc.)
- Crawl a website with more threads or processes than default
- Change your user agent
- Crawl multiple websites
- Check robots.txt
- Crawling body tags and paragraph tags

Showing progress through **-P** or **--progress** is recommended, as without it, you will find yourself wondering when your crawl will be done without any visual signs. The command for crawling more threads **(-- workers='number of workers')** and processes **(-- mode=process -- workers='number of workers')** can be very useful as well.

Of course, the script has many more options to explore. The examples below show some of the possible uses:

pylinkvalidate.py -p http://www.example.com/

The function above crawls the website and shows progress.

pylinkvalidate.py -p workers=4 http://www.example.com/

This function crawls a website with multiple threads and shows progress.

pylinkvalidate.py -p --parser=lxml http://www.example.com/

This function uses the lxml library in order to speed up the crawl while showing progress.

pylinkvalidate.py -P --types=a http://www.example.com/

The function above only crawls links **(<a href>)** on your website, ignoring images, scripts, stylesheets and any other non-link attribute on your website. This is also a useful function when crawling the URLs of large websites. After the script has run its course, you'll get a list of URLs with status codes 4xx and 5xx that have been found by crawling your website. Along with that, you'll gain a list of URLs that link to that page, so you'll have an easier time fixing the broken links. The regular crawl does not show any 3xx status codes. For more detailed information about what URLs your pages can be reached from, try the following function:

pylinkvalidate.py --report-type=all http://www.example.com/

This give information about the status code of a page, and all the other pages that link to a page.

An incredibly useful **SEO** tool you can use to crawl your website for broken links (404) and server errors. Both of these errors can be bad for your SEO efforts, so be sure to regularly crawl your own website in order to fix these errors ASAP.

Conclusion

While these scripts are incredibly useful, there are a lot of various uses for Python in the world of SEO. Challenge yourself to create scripts that make your SEO efforts more efficient.

Spamdexing

Spamdexing (also known as **search engine spam**, **search engine poisoning**, **black-hat search engine optimization**, **search spam** or **web spam**) is the deliberate manipulation of **search engine indexes**. It involves a number of methods, such as **link building** and repeating unrelated phrases, to manipulate the relevance or prominence of resources indexed in a manner inconsistent with the purpose of the indexing system.

Spamdexing could be considered to be a part of **search engine optimization**, although there are many SEO methods that improve the quality and appearance of the content of web sites and serve content useful to many users.

Common Spamdexing techniques can be classified into two broad classes: **content spam** (or term spam) and **link spam**.

Content spam

These techniques involve altering the logical view that a search engine has over the page's contents. They all aim at variants of the vector space model for information retrieval on text collections.

- **Keyword stuffing**
- **Hidden or invisible text**
- **Meta-tag stuffing**
- **Doorway pages**
- **Scraper sites**
- **Article spinning**
- **Machine translation**

Link spam

Link spam is defined as links between pages that are present for reasons other than merit. Link spam takes advantage of link-based ranking algorithms, which gives websites higher rankings the more other highly ranked websites link to it. These techniques also aim at influencing other link- based ranking techniques such as the HITS algorithm.

Link farms

- **Private blog networks**
- **Hidden links**
- **Sybil attack**
- **Spam blogs**
- **Guest blog spam**
- **Buying expired domains**

Using world-writable pages

Web sites that can be edited by users can be used by spamdexers to insert links to spam sites if the appropriate anti-spam measures are not taken.

Automated spambots can rapidly make the user-editable portion of a site unusable. Programmers have developed a variety of automated spam prevention techniques to block or at least slow down spambots.

- **Spam in blogs**
- **Comment spam**
- **Wiki spam**
- **Referrer log spamming**
- **Countermeasures**

Experiment – 4: Use Google analytics tools to implement the following:

a. Conversion Statistics

A conversion in Google Analytics is an important action you want your users to complete on your website and is considered a key event to your business. Any action or engagement that happens on your website can be tracked and labelled as a conversion event.

Examples include:

- video plays
- newsletter signups
- free trials of your product
- form completions for demo or consultation

A conversion in GA4 is essentially any action that you define as being valuable to your business. You can create Events inside your GA4 account to ensure you"re tracking every conversion that"s valuable to your business. Shortly, we"ll walk you through exactly how to track conversions using GA4.

Track conversions with Google Analytics 4

To track and measure key actions on your website, we need to create a custom event in Google Analytics 4 and turn that into a conversion.

➢ First, click Configure and select Events.

  You"ll see different events. All of these are options predefined by Google Analytics 4.

    - click
    - first_visit
    - page_view
    - scroll
    - session_view

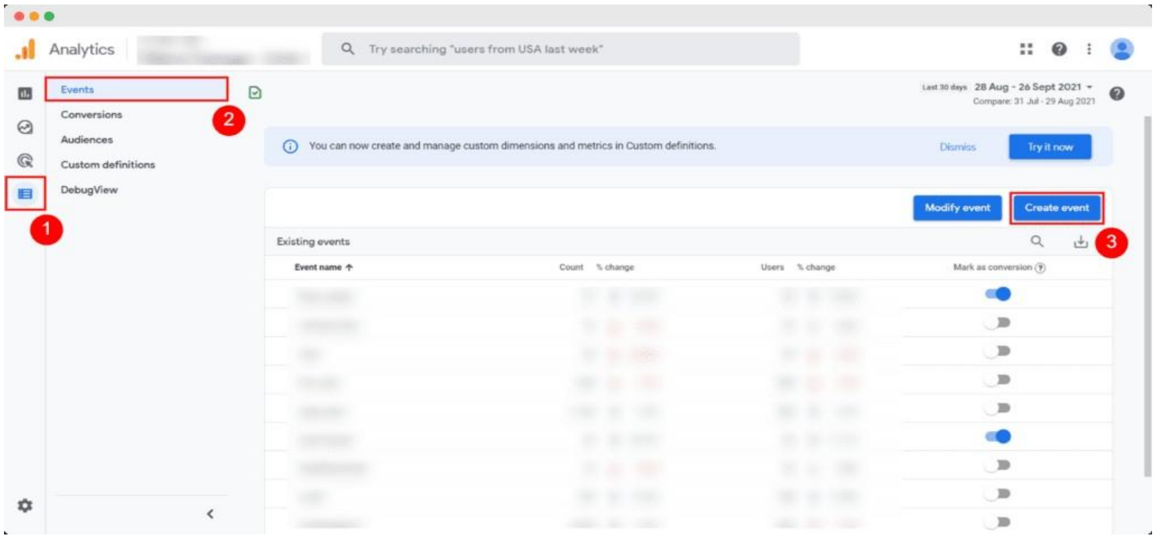
We can, set up a unique event to track a specific conversion. There are two ways to achieve this.

1. Create a new event based on an existing option
2. Configure a new tag in Google Tag Manager

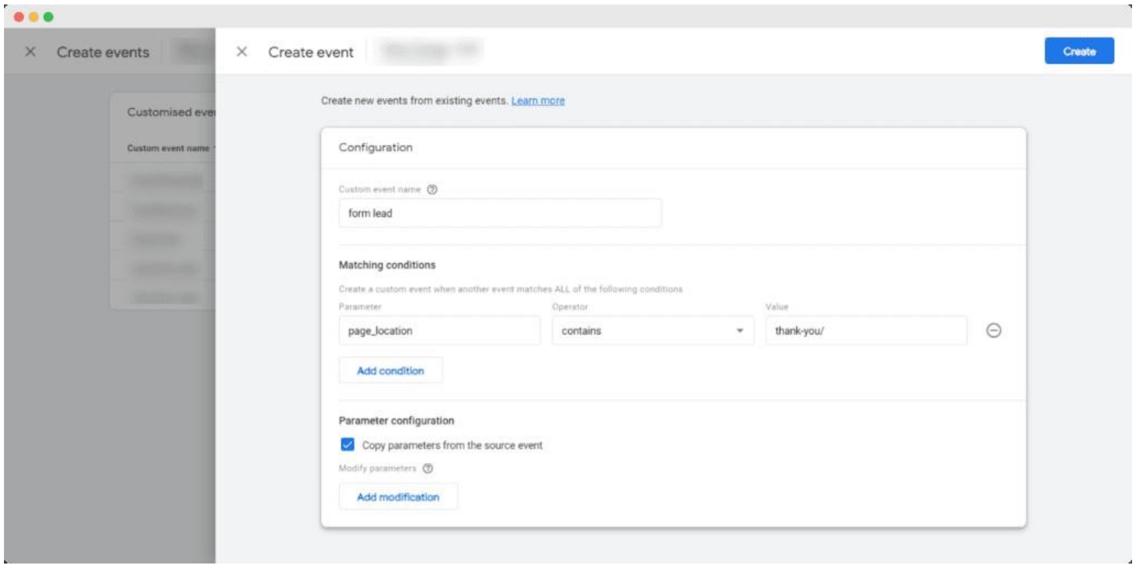
1. Create a new event based on an existing option

The first option is to create a new event based on an existing option in Google Analytics 4.
For example, we could take the predefined event "page_view" and trigger a conversion whenever a user lands on a specific thank you page. Here‟s how to get started.

1. Log into your GA4 property, go to **Configure** > „**Events**‟ > **Create Event**.


Analytics
Try searching "users from USA last week"
Events
Conversions
Audiences
Custom definitions
DebugView
1
2
3
Last 30 days 28 Aug - 26 Sept 2021
Compare: 31 Jul - 29 Aug 2021
You can now create and manage custom dimensions and metrics in Custom definitions.
Dismiss
Try it now
Modify event
Create event
Existing events
Event name
Count
% change
Users
% change
Mark as conversion

2. Give your custom event a name and set up your condition settings. In the example below, the configuration will trigger a new conversion event whenever someone views a thank you page after submitting a form.



Create new events from existing events. Learn more

Configuration

Custom event name: form lead

Matching conditions

Create a custom event when another event matches ALL of the following conditions

Parameter	Operator	Value
page_location	contains	thank-you/

Parameter configuration

☑ Copy parameters from the source event

Modify parameters

3. Once you‟ve set up your custom event, click **Create**.

4. Now go to Conversions. You should see your custom event listed. All you do is mark it as a conversion using the slider, and you‟re good to go. Don‟t panic if you don‟t see your custom event.

5. Sometimes it can take a while for it to appear.

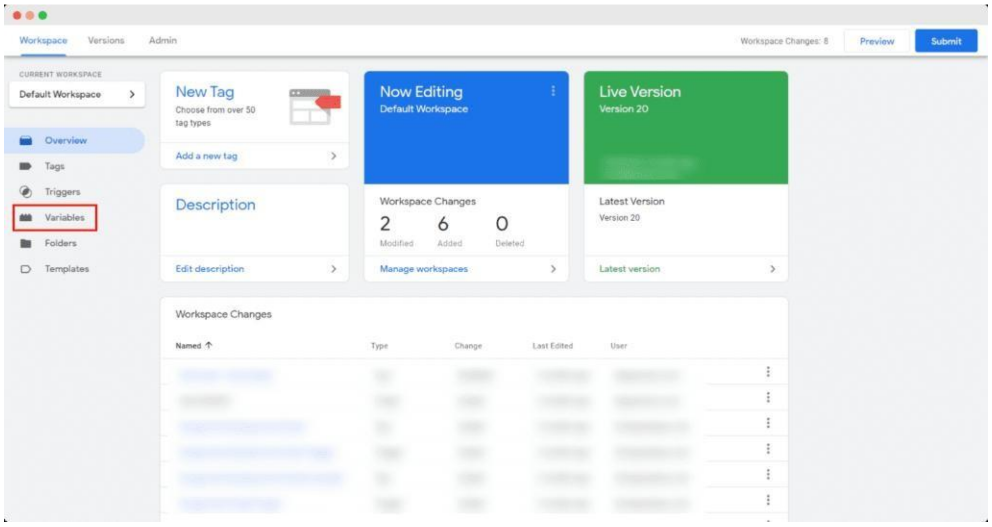
2. Configure a new tag in Google Tag Manager

The second option is to configure a new tag in Google Tag Manager that will successfully capture and report a unique event in GA4.

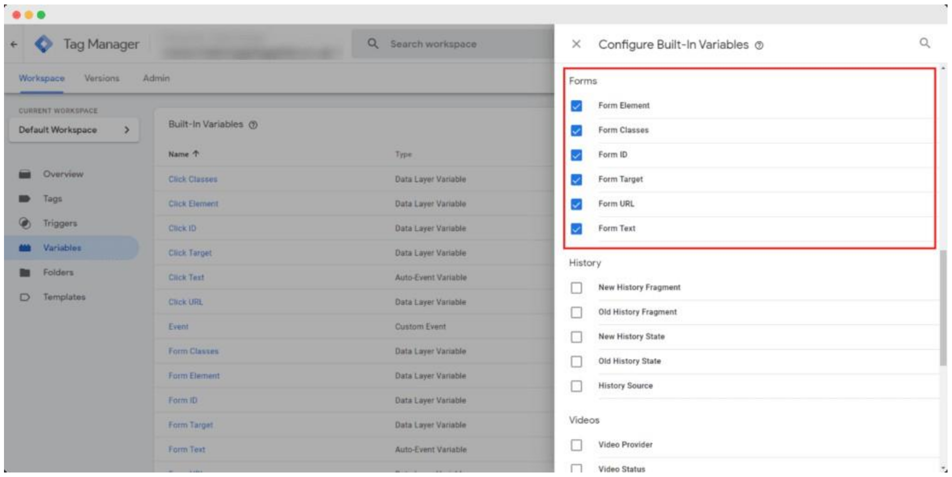
This method gives you a lot more flexibility and control over your conversions.

For this example, let‟s say you have a form on your website that doesn‟t redirect to a new page. You could configure your event in Google Tag Manager to track the submission button as a conversion instead.

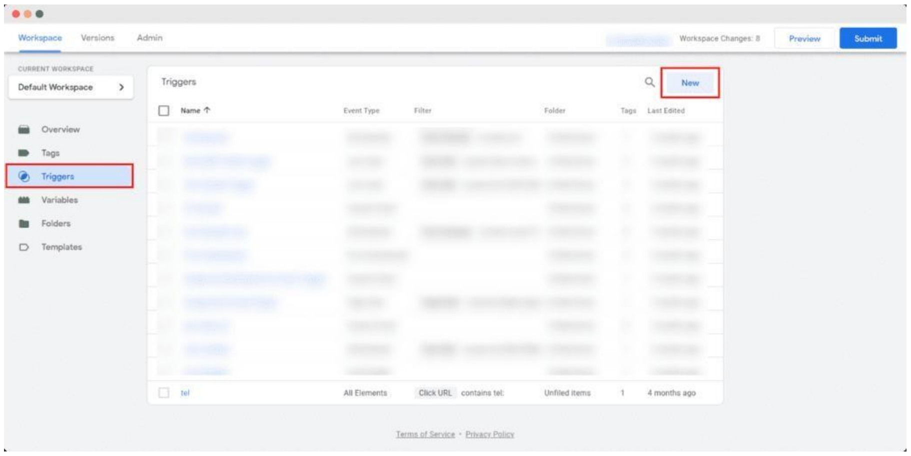
1. First, log into your Google Tag Manager and select „**Variables**‟.



2. If you‟re new to Google Tag Manager, then you‟ll want to click on „**Configure**‟ and ensure that all the boxes are checked under the „Forms‟ section.



3. Afterwards, you‟ll need to click on „**Triggers**‟. Here you‟ll want to create a generic form submission trigger. This will allow you to see the form submission event inside of Google Tag Manager, and you‟ll know which form id to fire your tag and conversion event in GA4.

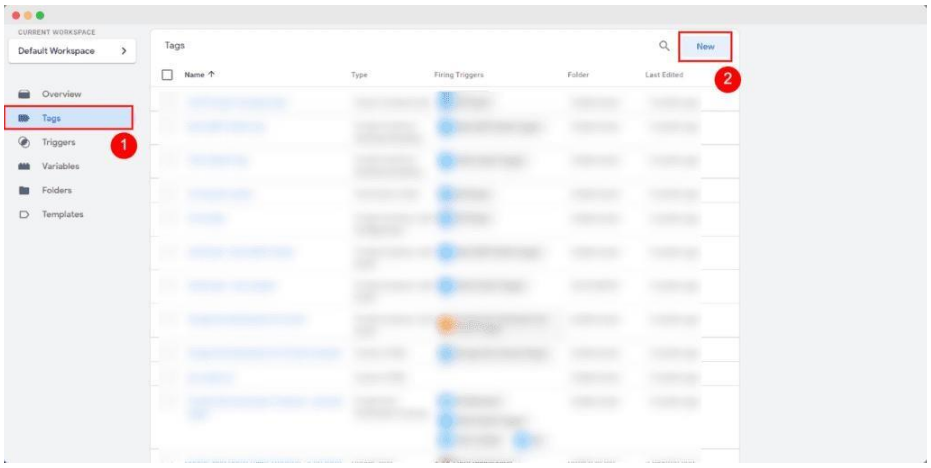


Once complete, go into preview mode, type in your domain and fill out a form.

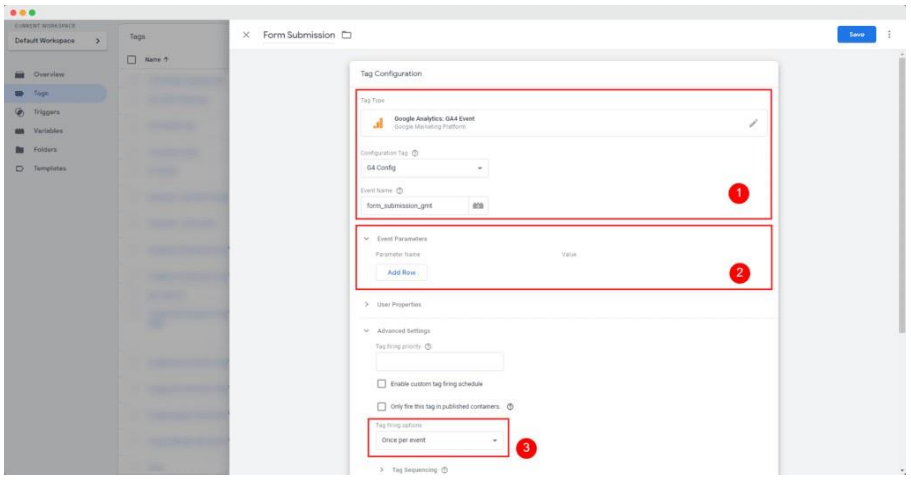
If successful, you should be able to see the form submit event and your variables in Google Tag Manager.

You"ll want to scroll down to Form ID and make a note of the code as you"ll need this later on in this set up.

4. Next, go to „**Tags**", select „**New**" and give your tag a name. Make sure to give it a name that is easy to remember.



5. Select „**Tag Configuration**", click „**Google Analytics: GA4 Event**" and complete the set up. Under „**Event Parameters**" you can assign a value which is displayed in Analytics every time the event is triggered. Before you hit save, make sure that this tag is only firing once per event. You don"t want someone to fill this in multiple times as it"ll lead to many data discrepancies further down the line.



Form Submission

Tag Configuration

Tag Type: Google Analytics: GA4 Event
Google Marketing Platform

Configuration Tag: G4 Config

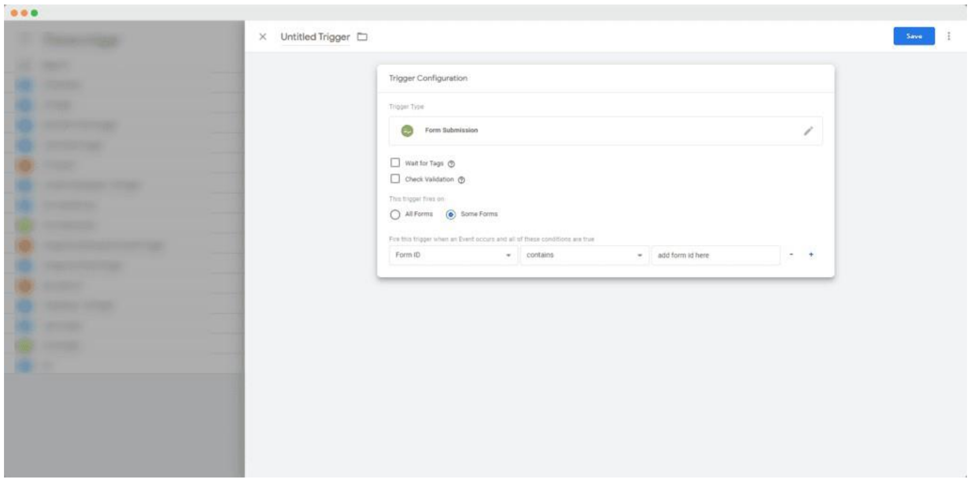
Event Name: form_submission_gmt

Event Parameters
Parameter Name | Value
Add Row

User Properties

Advanced Settings
Tag firing priority
☐ Enable custom tag firing schedule
☐ Only fire this tag in published containers.
Tag firing options: Once per event
Tag Sequencing

6. Now, for the trigger. Click the „+" in the top right hand corner. Select „**Form Submission**" and select „**Some Forms**". In the drop down menu, look for „**Form ID**". You"ll want to copy that code into the field that you took a note of earlier on in this step-by-step guide. Once complete, click „**Save**".



Untitled Trigger

Trigger Configuration

Trigger Type: Form Submission

☐ Wait for Tags
☐ Check Validation

This trigger fires on
All Forms [✓ Some Forms]

Fire this trigger when an Event occurs and all of these conditions are true
Form ID | contains | add form id here

7. Now, you"ll need to go back into preview mode to make sure that the tag is firing correctly. To test it"s working correctly, complete another form submission on your desired webpage. Go back to your form submit event you made earlier in step 3. If set up correctly, you"ll be able to find your submission displayed in Google Tag Manager. Go back to „**Tags**" and click „**Submit**".
8. Go back to your GA4 property. Be mindful that it can take a few hours for your new event to be displayed in your „All Events" view. Eventually, it"ll show and you can go ahead and mark it as a conversion. Once complete, you should be able to see your new event under „Conversions" as well as other reports in GA4.

Experiment – 4: Use Google analytics tools to implement the following:

b. Visitor Profiles

Through the Google Analytics dashboard, you can learn details about your website and app visitors and create advanced audience segments based on information such as: Age and gender, Locations and languages and Devices most used to browse.

A Dashing widget for displaying the number of visitors to your website over a specified timeframe, as reported by Google Analytics

Dependencies

google-api-ruby-client

Add it to dashing's gemfile:

```
gem 'google-api-client'
```

and run

```
bundle install.
```

Usage

To use this widget, you'll first need to set up a Google API project and attach it to the Google Analytics profile you wish to monitor.

1. Create and download a new private key for Google API access.

1. Go to https://code.google.com/apis/console
2. Click 'Create Project'
3. Enable 'Analytics API' service and accept both TOS's
4. Click 'API Access' in the left-hand nav menu
5. Click 'Create an OAuth 2.0 Client ID'
6. Enter a product name (e.g. Dashing Widget) - logo and url are optional
7. Click 'Next'
8. Under Application Type, select 'Service Account'
9. Click 'Create Client ID'
10. Click 'Download private key' NOTE: This will be your only opportunity to download this key.
11. Note the password for your new private key ('notasecret')
12. Close the download key dialog

2. Attach your Google API service account to your Google Analytics profile

Note: you will need to be an administrator of the Google Analytics profile

1. Log in to your Google Analytics account: http://www.google.com/analytics/
2. Click 'Admin' in the upper-right corner
3. Select the account containing the profile you wish to use
4. Select the property containing the profile you wish to use
5. Select the profile you wish to use
6. Click the 'Users' tab
7. Click '+ New User'
8. Enter the email address you copied from step 13 above
9. Click 'Add User'

3. Locate the ID for your Google Analytics profile

1. On your Google Analytics profile page, click the 'Profile Settings' tab
2. Under 'General Information' copy your Profile ID (e.g. 654321) - you'll need it in your ruby code later

4. Start coding (finally)

1. Copy the visitor_count.rb file in to your dashing jobs\ folder.
2. Update the service_account_email, key_file, key_secret and profileID variables

```
service_account_email = '[YOUR SERVICE ACCOUTN EMAIL]' # Email of service account
key_file = 'path/to/your/keyfile.p12' # File containing your private key
key_secret = 'notasecret' # Password to unlock private key
profileID = '[YOUR PROFILE ID]' # Analytics profile ID.
```

3. Add the widget HTML to your dashboard

```
<li data-row="1" data-col="1" data-sizex="1" data-sizey="1">
        <div data-id="visitor_count" data-view="Number" data-title="Visitors This
        Month"></div>
</li>
```

visitor_count.rb:

```
require 'google/api_client'
require 'date'

# Update these to match your own apps credentials
service_account_email = '[YOUR SERVICE ACCOUNT EMAIL]' # Email of service account
key_file = 'path/to/your/keyfile.p12' # File containing your private key
key_secret = 'notasecret' # Password to unlock private key
profileID = '[YOUR PROFILE ID]' # Analytics profile ID.

# Get the Google API client
client = Google::APIClient.new(:application_name => '[YOUR APPLICATION NAME]',
  :application_version => '0.01')

# Load your credentials for the service account
key = Google::APIClient::KeyUtils.load_from_pkcs12(key_file, key_secret)
client.authorization = Signet::OAuth2::Client.new(
  :token_credential_uri => 'https://accounts.google.com/o/oauth2/token',
  :audience => 'https://accounts.google.com/o/oauth2/token',
  :scope => 'https://www.googleapis.com/auth/analytics.readonly',
  :issuer => service_account_email,
  :signing_key => key)

# Start the scheduler
SCHEDULER.every '1m', :first_in => 0 do

  # Request a token for our service account
  client.authorization.fetch_access_token!

  # Get the analytics API
  analytics = client.discovered_api('analytics','v3')
  # Start and end dates
  startDate = DateTime.now.strftime("%Y-%m-01") # first day of current month
  endDate = DateTime.now.strftime("%Y-%m-%d") # now

  # Execute the query
  visitCount = client.execute(:api_method => analytics.data.ga.get, :parameters => {
    'ids' => "ga:" + profileID,
    'start-date' => startDate,
    'end-date' => endDate,
    # 'dimensions' => "ga:month",
    'metrics' => "ga:visitors",
    # 'sort' => "ga:month"
  })

  # Update the dashboard
  # Note the trailing to_i - See: https://github.com/Shopify/dashing/issues/33
send_event('visitor_count',   { current: visitCount.data.rows[0][0].to_i })
end
```

Experiment – 5: Use Google analytics tools to implement the Traffic Sources.

Do you want to know how to track website traffic with Google Analytics? If you use Google Analytics to see your website traffic, you'll get to see all kinds of data like how your visitors found your site, how long they spent on your site, and what pages they viewed.

When you know how your visitors found you, whether through Google search, social media, or other channel, you can make impactful decisions to help you grow your traffic.

Why Track Website Traffic with Google Analytics?

As a marketer, there are many benefits to being able to see your website traffic and where it came from. Here are some reasons for tracking your traffic:

- **Better Understand Your Visitors** – Tracking you traffic sources in Google analytics can help you identify your unique visitors' geographic location and which channels they use, so you can better understand them and their user behavior, provide targeted messages, and make improvements to your SEO strategies
- **Measure Your** Marketing Campaigns – If a campaign is built around driving traffic or completing actions on your website, then you can measure its effectiveness
- **Focus on Channels for Best Results** – By identifying which channel performed the best in getting visitors to your site, you can focus on it more to get even better results
- **Find New Content Topics** – People from different channels might be interested in specific topics, so you can discover new content ideas by checking your traffic sources and engagement rate for each page, then using that traffic data to perform keyword research
- **Identify Traffic Gaps on Your Site** – You can identify which channel doesn't perform well in attracting visitors, so you can optimize it.

How to Track Your Website Traffic in WordPress?

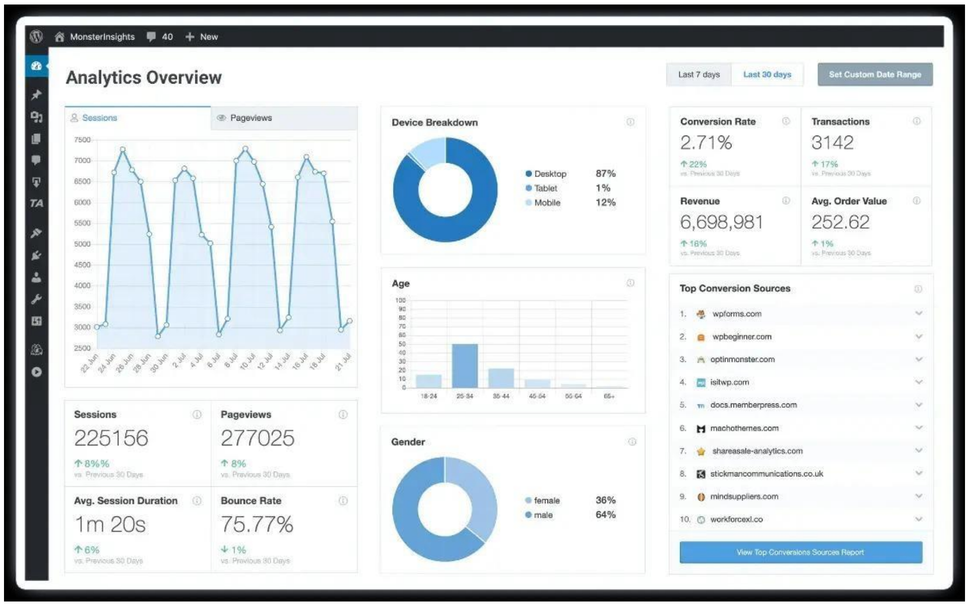
When it comes to using Google Analytics, many users find it overwhelming. That's because it requires some code for setting it up on your WordPress site.

Plus, you'll have to be an Analytics expert to find the right report and get the traffic data that you need for making decisions.

So, a much easier way of viewing your traffic sources, pageviews, and other engagement metrics in WordPress is through MonsterInsights.

It's the best WordPress plugin for Google Analytics and it makes using Analytics very easy. You don't have to worry about hiring a developer or someone who knows analytics.

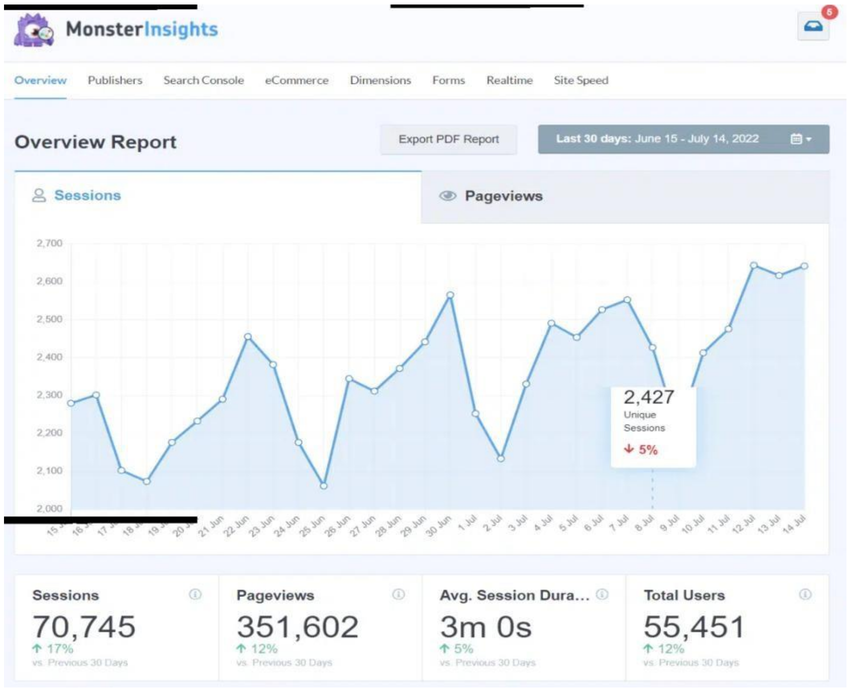
The plugin helps add Google Analytics to your website and then displays the most useful reports right inside your WordPress dashboard.



Once you've installed MonsterInsights on your website, you can see where your traffic is coming from. Let's look at a few of the reports that you can use to track website traffic.

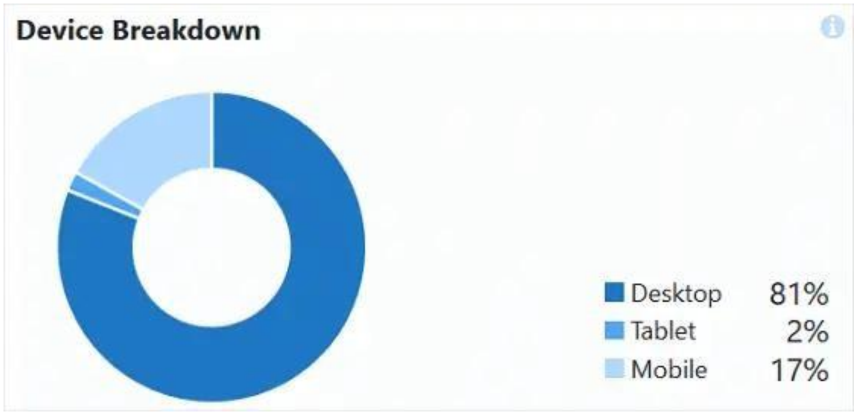
Overview Report

To start, go to **Insights » Reports » Overview**. Here you can see a traffic overview report and the overall performance of your website.

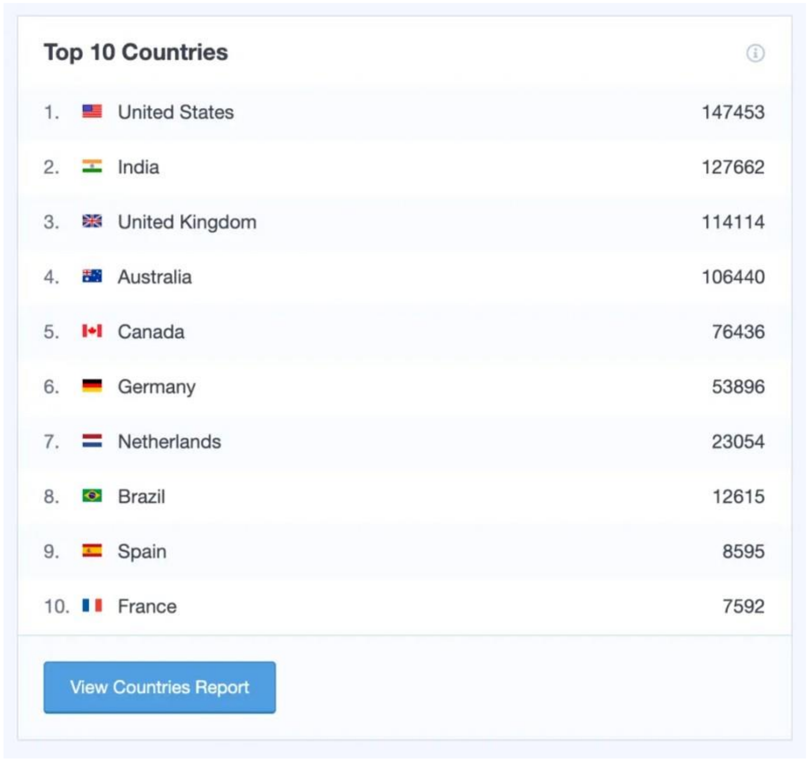


MonsterInsights
Overview
Publishers
Search Console
eCommerce
Dimensions
Forms
Realtime
Site Speed
Overview Report
Export PDF Report
Last 30 days: June 15 - July 14, 2022
Sessions
Pageviews
2,427 Unique Sessions 5%
Sessions 70,745 17% vs. Previous 30 Days
Pageviews 351,602 12% vs. Previous 30 Days
Avg. Session Dura... 3m 0s 5% vs. Previous 30 Days
Total Users 55,451 12% vs. Previous 30 Days

If you scroll down, you can see the **Device Breakdown** report that shows which device your visitors use to view your website.



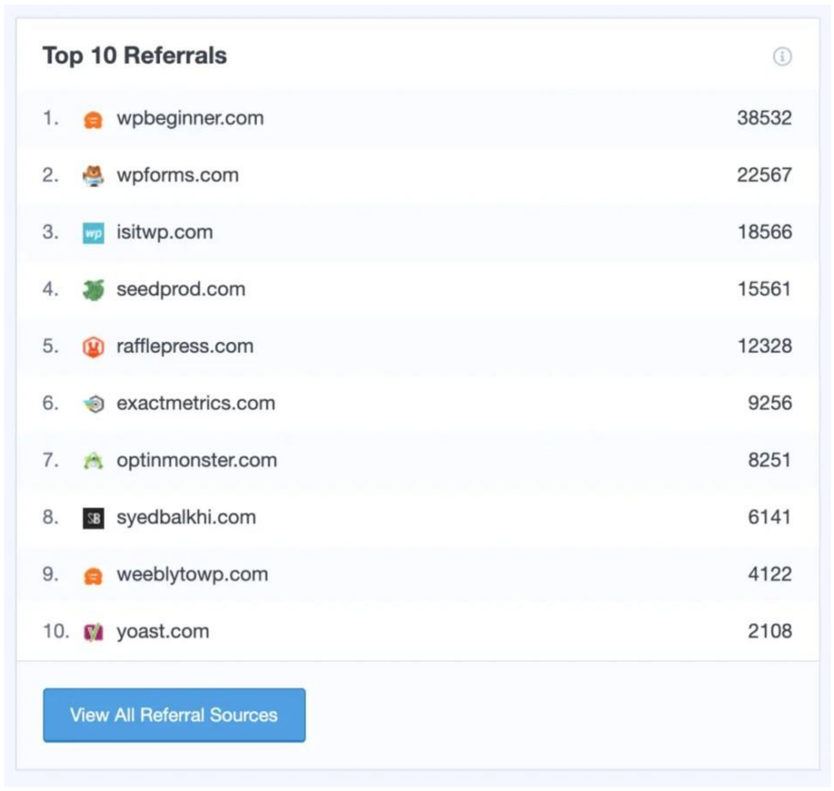
In the Overview report, you can also view the **Top 10 Countries** that your users are from. Using this report, you can create campaigns, messages, and content according to different regions.



Top 10 Countries

1.	United States	147453
2.	India	127662
3.	United Kingdom	114114
4.	Australia	106440
5.	Canada	76436
6.	Germany	53896
7.	Netherlands	23054
8.	Brazil	12615
9.	Spain	8595
10.	France	7592

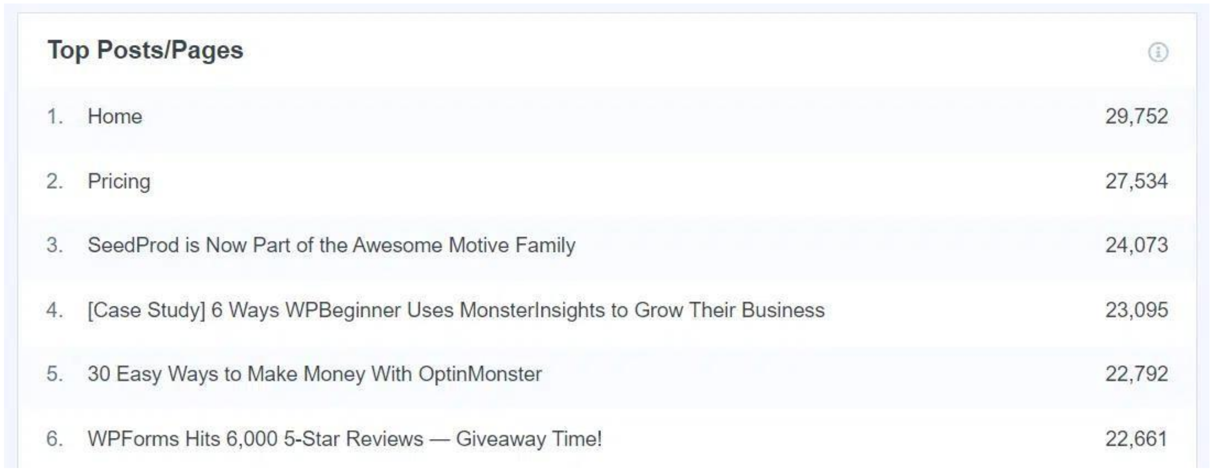
And next to countries, you can see the Top 10 Referrals report. This shows websites that send the most traffic to your site, including social media networks. You can form partnerships with these sites and continue to grow your traffic.



Top 10 Referrals

1.	wpbeginner.com	38532
2.	wpforms.com	22567
3.	isitwp.com	18566
4.	seedprod.com	15561
5.	rafflepress.com	12328
6.	exactmetrics.com	9256
7.	optinmonster.com	8251
8.	syedbalkhi.com	6141
9.	weeblytowp.com	4122
10.	yoast.com	2108

Wondering how to see page views on Google Analytics? At the bottom of the Overview report, we can find your top posts and pages.

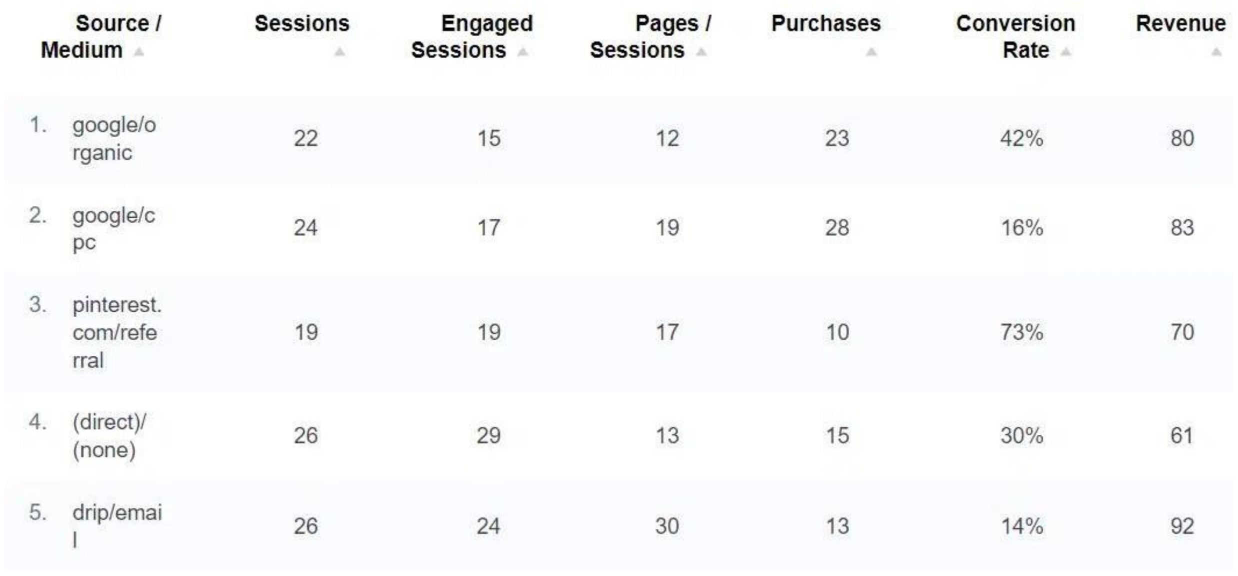


Top Posts/Pages

1.	Home	29,752
2.	Pricing	27,534
3.	SeedProd is Now Part of the Awesome Motive Family	24,073
4.	[Case Study] 6 Ways WPBeginner Uses MonsterInsights to Grow Their Business	23,095
5.	30 Easy Ways to Make Money With OptinMonster	22,792
6.	WPForms Hits 6,000 5-Star Reviews — Giveaway Time!	22,661

Source/Medium Report

Where exactly is your traffic coming from? With the Source/Medium report, you can see which channels are sending visitors your way. Plus, you’ll see data for each channel like engaged sessions, conversion rate, and revenue.

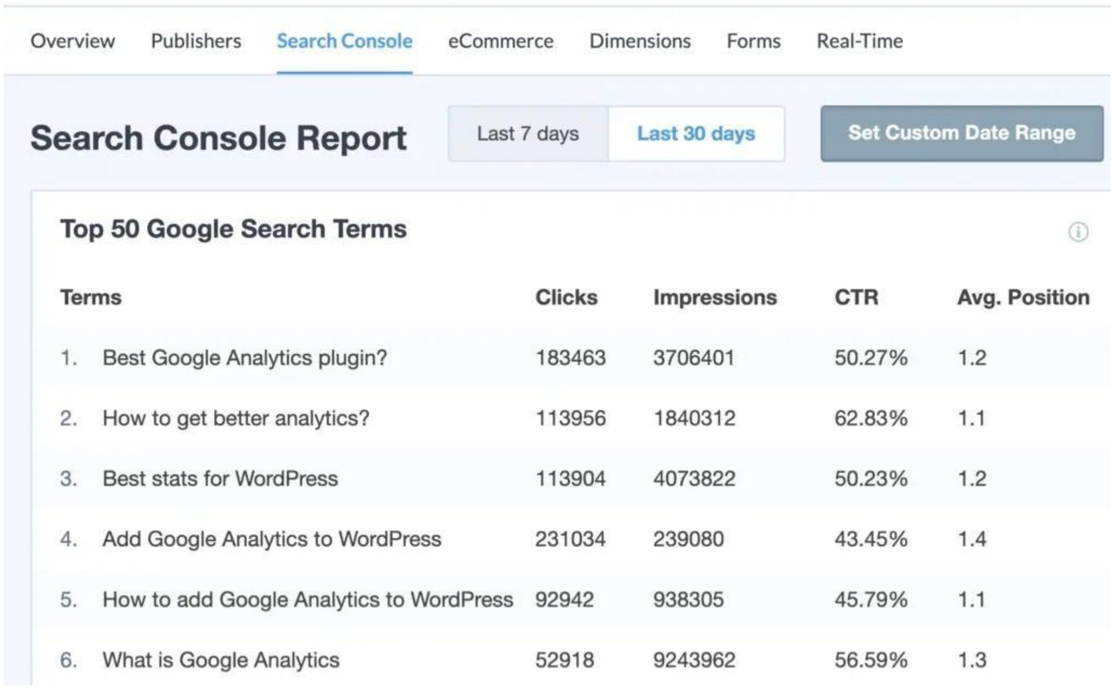


	Source / Medium	Sessions	Engaged Sessions	Pages / Sessions	Purchases	Conversion Rate	Revenue
1.	google/organic	22	15	12	23	42%	80
2.	google/cpc	24	17	19	28	16%	83
3.	pinterest.com/referral	19	19	17	10	73%	70
4.	(direct)/(none)	26	29	13	15	30%	61
5.	drip/email	26	24	30	13	14%	92

Search Console Report

Now, if your site gets some organic traffic, you should know which keywords your site is ranked for. To find that out, MonsterInsights offers a Search Console Report.

It shows the top 50 Google search terms for your website along with clicks, impressions, CTR (click-through-rate), and average position (keyword ranking).



Top 50 Google Search Terms

	Terms	Clicks	Impressions	CTR	Avg. Position
1.	Best Google Analytics plugin?	183463	3706401	50.27%	1.2
2.	How to get better analytics?	113956	1840312	62.83%	1.1
3.	Best stats for WordPress	113904	4073822	50.23%	1.2
4.	Add Google Analytics to WordPress	231034	239080	43.45%	1.4
5.	How to add Google Analytics to WordPress	92942	938305	45.79%	1.1
6.	What is Google Analytics	52918	9243962	56.59%	1.3

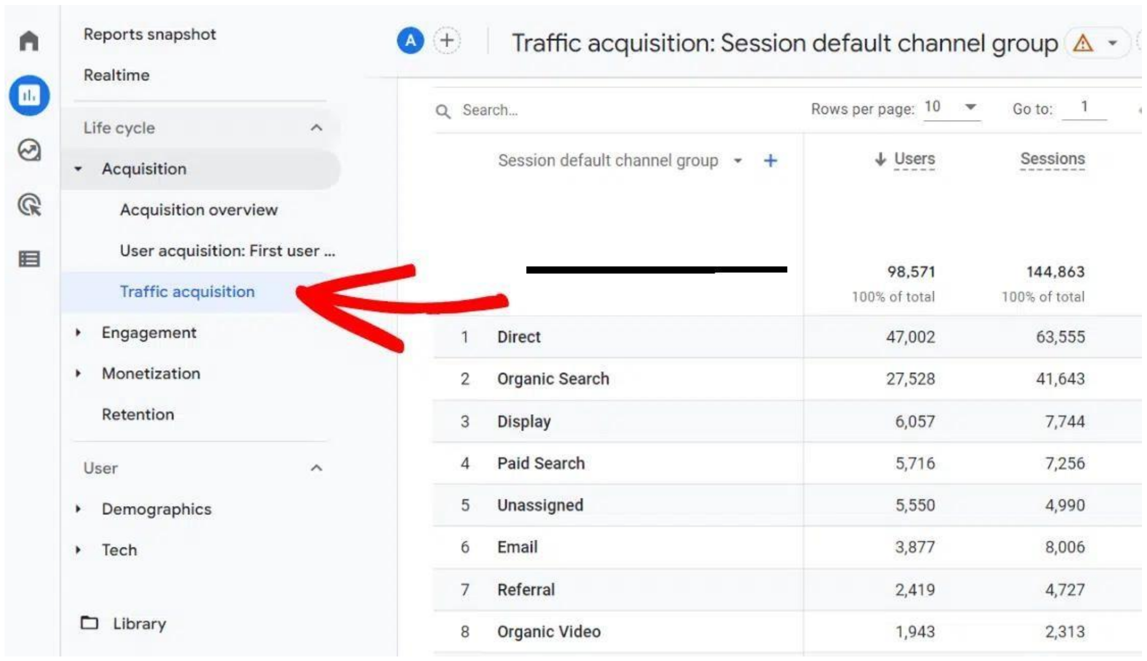
How to Track Website Traffic in Google Analytics?

MonsterInsights offers user-friendly reports inside your dashboard, so you don't have to leave your site. But what if you want to see where traffic is coming from in Google Analytics?

There are many reports that you can use to see website traffic statistics in Analytics. Having said that, it's very easy to get lost.

To help you find the data that matters, you can start by logging in to your Google Analytics account.

You can start by navigating to **Acquisition » Traffic Acquisition**:

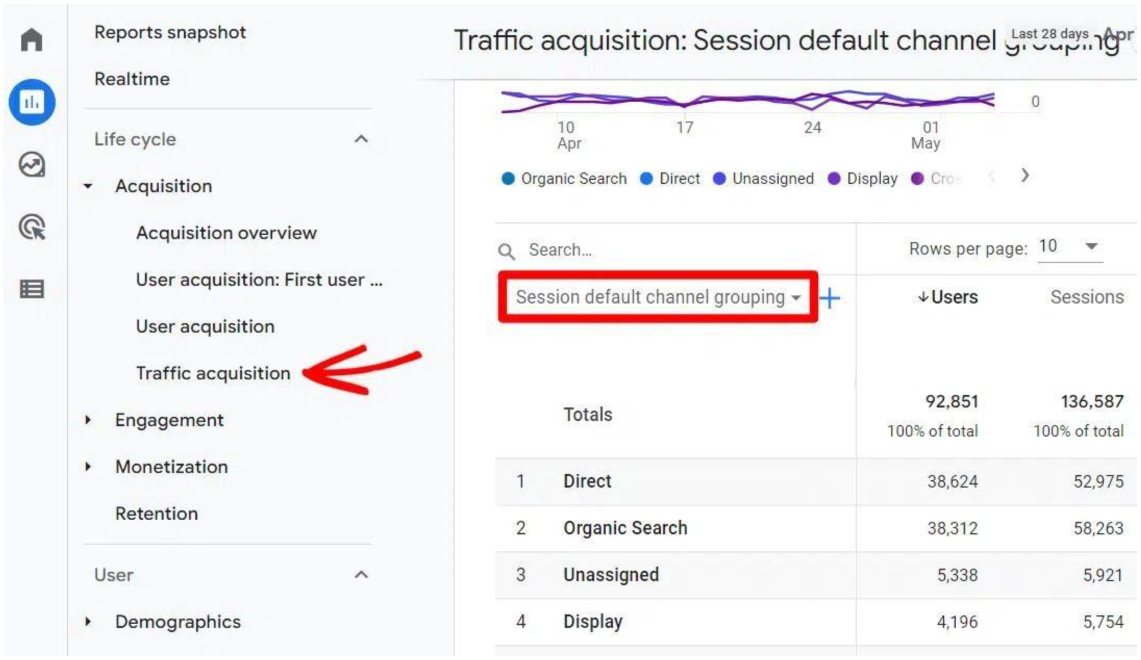


	Session default channel group	Users	Sessions
		98,571 100% of total	144,863 100% of total
1	Direct	47,002	63,555
2	Organic Search	27,528	41,643
3	Display	6,057	7,744
4	Paid Search	5,716	7,256
5	Unassigned	5,550	4,990
6	Email	3,877	8,006
7	Referral	2,419	4,727
8	Organic Video	1,943	2,313

In this Google Analytics web traffic report, you can see which channels are driving the most traffic to your website. For instance, you can see the number of visitors from organic searches.

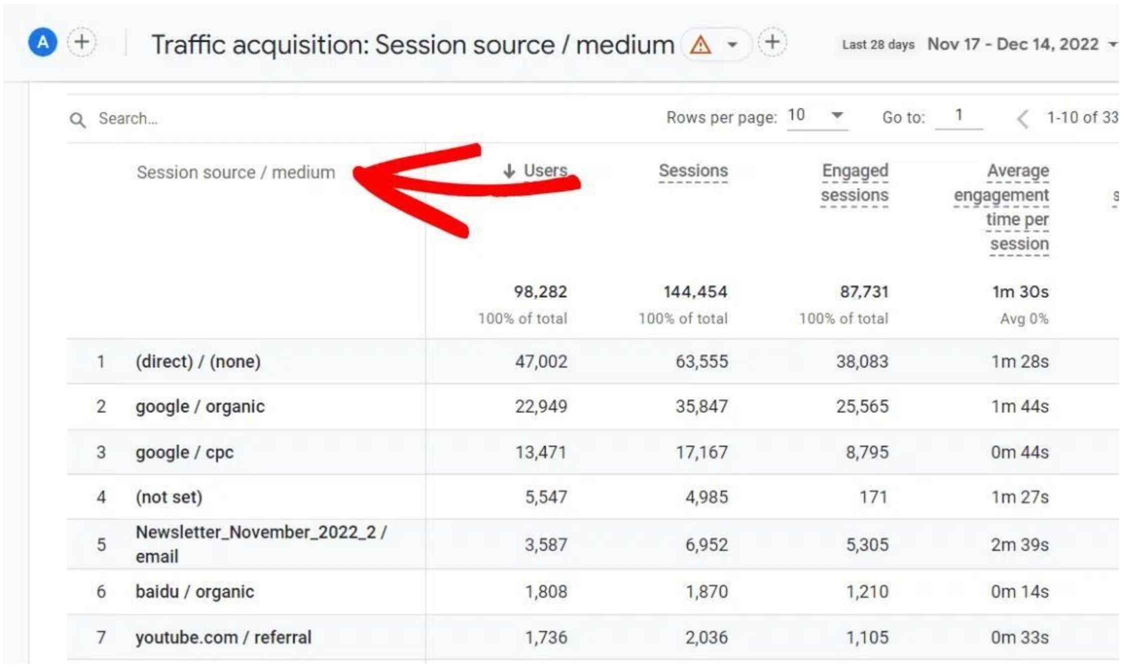
You'll also be able to see how much traffic is getting referred from other sites, how much direct traffic you're getting, and more.

To get an deeper look at exactly where your traffic is coming from, click the Session default channel group dropdown:



	Session default channel grouping	↓Users	Sessions
	Totals	92,851 100% of total	136,587 100% of total
1	Direct	38,624	52,975
2	Organic Search	38,312	58,263
3	Unassigned	5,338	5,921
4	Display	4,196	5,754

From there, select Session source/medium**:**

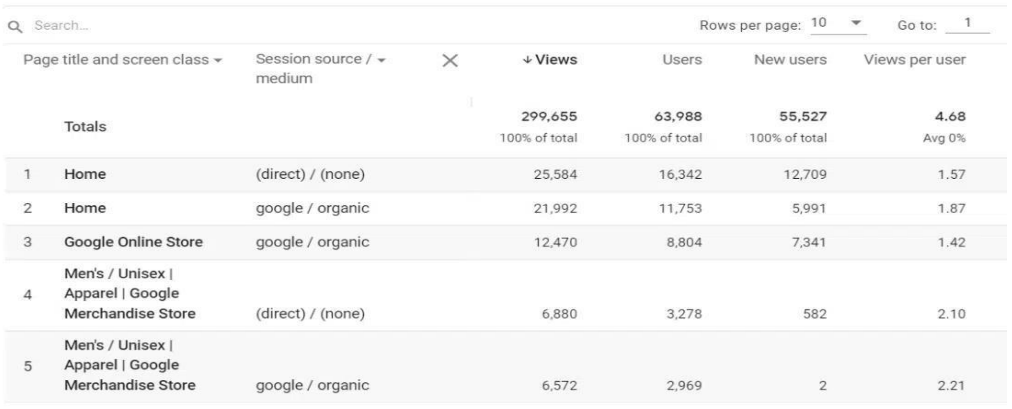

Traffic acquisition: Session source / medium
Last 28 days Nov 17 - Dec 14, 2022
Search...
Rows per page: 10
Go to: 1
1-10 of 33

	Session source / medium	↓ Users	Sessions	Engaged sessions	Average engagement time per session
		98,282 100% of total	144,454 100% of total	87,731 100% of total	1m 30s Avg 0%
1	(direct) / (none)	47,002	63,555	38,083	1m 28s
2	google / organic	22,949	35,847	25,565	1m 44s
3	google / cpc	13,471	17,167	8,795	0m 44s
4	(not set)	5,547	4,985	171	1m 27s
5	Newsletter_November_2022_2 / email	3,587	6,952	5,305	2m 39s
6	baidu / organic	1,808	1,870	1,210	0m 14s
7	youtube.com / referral	1,736	2,036	1,105	0m 33s

Now, you can see which search engines contributed to your organic traffic, which sites sent you referral traffic, and more.

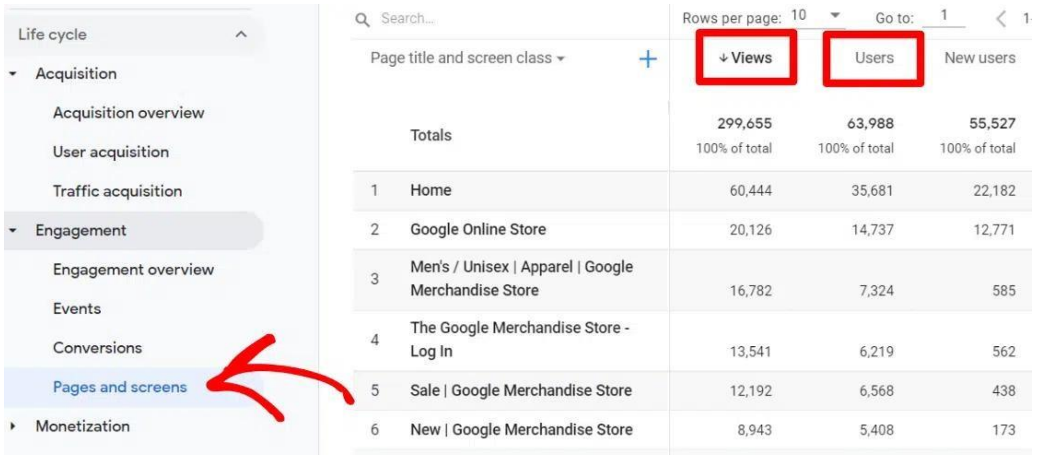
Now that you know how to see where traffic is coming from for your overall website, what if you want to use Google Analytics to find traffic for specific pages?

Google Analytics Traffic Sources for a Specific Page?



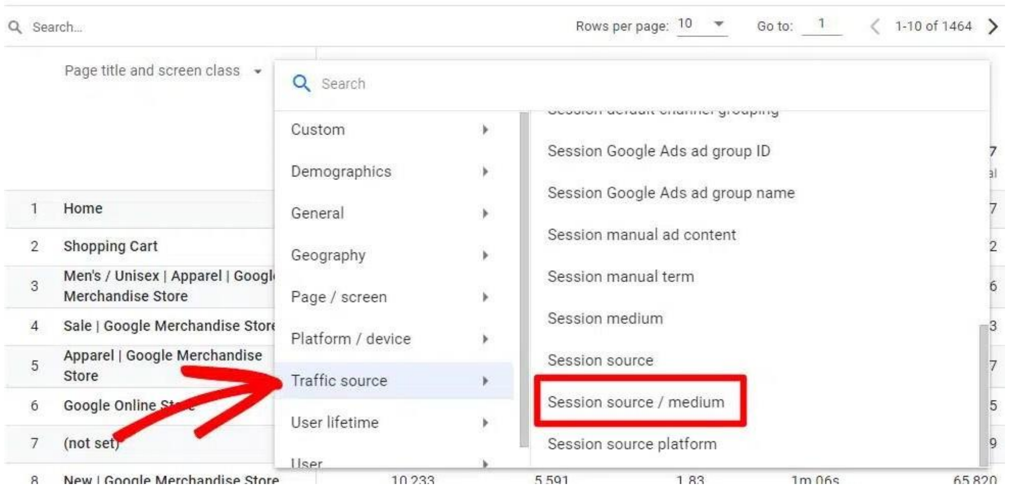
	Page title and screen class	Session source / medium	↓Views	Users	New users	Views per user
	Totals		299,655 100% of total	63,988 100% of total	55,527 100% of total	4.68 Avg 0%
1	Home	(direct) / (none)	25,584	16,342	12,709	1.57
2	Home	google / organic	21,992	11,753	5,991	1.87
3	Google Online Store	google / organic	12,470	8,804	7,341	1.42
4	Men's / Unisex | Apparel | Google Merchandise Store	(direct) / (none)	6,880	3,278	582	2.10
5	Men's / Unisex | Apparel | Google Merchandise Store	google / organic	6,572	2,969	2	2.21

To find site traffic sources and other web traffic analytics for a specific page in Google Analytics 4, navigate to Engagement » Pages and Screens. In the table, you'll see both Views and Users, so you can see how many views each page got, and how many users completed those views.



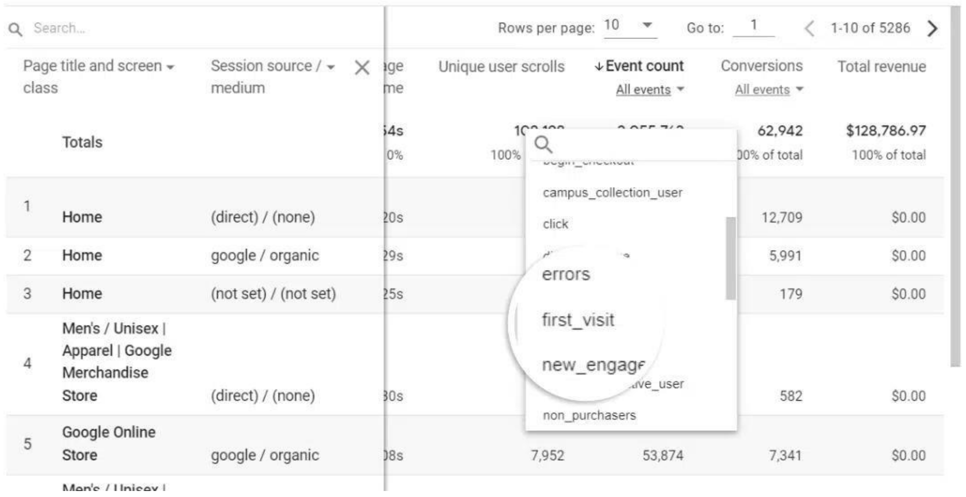
	Page title and screen class	↓Views	Users	New users
	Totals	299,655 100% of total	63,988 100% of total	55,527 100% of total
1	Home	60,444	35,681	22,182
2	Google Online Store	20,126	14,737	12,771
3	Men's / Unisex | Apparel | Google Merchandise Store	16,782	7,324	585
4	The Google Merchandise Store - Log In	13,541	6,219	562
5	Sale | Google Merchandise Store	12,192	6,568	438
6	New | Google Merchandise Store	8,943	5,408	173

To add source/medium to the report, click the **plus icon** above the Totals column. In the dropdown, choose **Traffic source** then **Session source / medium.**



	Page title and screen class					
1	Home					
2	Shopping Cart					
3	Men's / Unisex | Apparel | Googl Merchandise Store					
4	Sale | Google Merchandise Store					
5	Apparel | Google Merchandise Store					
6	Google Online Store					
7	(not set)					
8	New | Google Merchandise Store	10,233	5,591	1.83	1m 06s	65,820

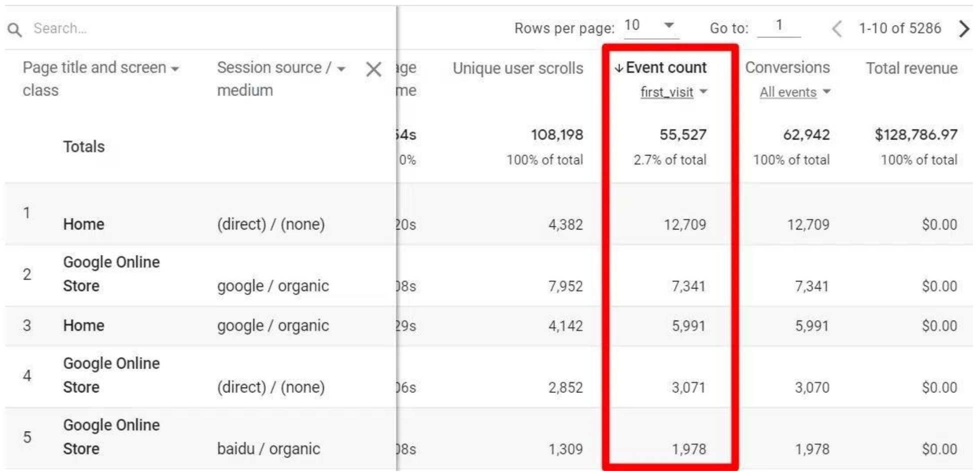
Now, you have a table of your pages by source/medium and views:
To change this table to view your landing pages by source/medium, scroll over the the **Event count** column. Click the down arrow next to **All Events** and select **first_visit**.



Rows per page: 10 Go to: 1 1-10 of 5286

	Page title and screen class	Session source / medium	ge me	Unique user scrolls	Event count All events	Conversions All events	Total revenue
	Totals		4s 0%	10 100%		62,942 00% of total	$128,786.97 100% of total
1	Home	(direct) / (none)	20s			12,709	$0.00
2	Home	google / organic	29s			5,991	$0.00
3	Home	(not set) / (not set)	25s			179	$0.00
4	Men's / Unisex | Apparel | Google Merchandise Store	(direct) / (none)	30s			582	$0.00
5	Google Online Store	google / organic	08s	7,952	53,874	7,341	$0.00

Then, click the **Event count** column title to sort by landing page visit events. Now you have your landing page visits report by source/medium.



Search... Rows per page: 10 Go to: 1 1-10 of 5286

	Page title and screen class	Session source / medium	ge me	Unique user scrolls	Event count first_visit	Conversions All events	Total revenue
	Totals		4s 0%	108,198 100% of total	55,527 2.7% of total	62,942 100% of total	$128,786.97 100% of total
1	Home	(direct) / (none)	20s	4,382	12,709	12,709	$0.00
2	Google Online Store	google / organic	08s	7,952	7,341	7,341	$0.00
3	Home	google / organic	29s	4,142	5,991	5,991	$0.00
4	Google Online Store	(direct) / (none)	06s	2,852	3,071	3,070	$0.00
5	Google Online Store	baidu / organic	08s	1,309	1,978	1,978	$0.00