Mario Fischer, Patrick Lürwer

R-LEUCHTUNGEN!

Teil 3: Welche Content-Art performt außergewöhnlich gut?

In den Ausgaben bis 54 bis 58 der Website Boosting konnten Sie in der Serie „R4SEO“ von Patrick Lürwer nachvollziehen, wie man die kostenlose Software R verwendet, was sie leistet und wie man sie nutzbringend für die eigene Arbeit für SEO bzw. die Aufklärung im Online-Marketing einsetzen kann. R wurde ja ursprünglich für Statistik entwickelt. Wer das bisher als Entschuldigung verwendet hat, es deswegen links liegen zu lassen, dem sei versichert, dass er damit komplett falschliegt.

R kann für den Einsatz im Unternehmen, allen Bereichen voran „Online“, sehr viel mehr leisten, als statistische Berechnungen durchzuführen. Genau genommen ist es ein wirklich nützliches Helferlein bei allen Aufgaben im Umgang mit größeren Datenmengen, mit Daten, die man erst in eine gewisse Struktur bringen muss, und bei der automatischen oder halb automatischen Datenbeschaffung aus praktisch fast allen Quellen aus dem Web!

Für die interessierten Einsteiger, aber auch für alle, die nach der Serie von Patrick Lürwer „R-Blut“ geleckt haben, startete in der Ausgabe 62 eine neue anwendungsorientierte Serie. Sie werden in jeder Ausgabe erfahren, wie sie ohne Programmierkenntnisse jeweils ein definiertes und in der Online-Praxis häufiger auftretendes Problem rund um das Thema Daten und Auswertungen lösen können. Und keine Sorge, die kleinen Hilfe-Tutorials nehmen Sie Schritt für Schritt an der Hand, sodass Sie auch als Neuling von der Power von R profitieren können. Was hält Sie also ab, das einfach mal auszuprobieren? Die einzelnen Schritte müssen Sie übrigens nicht im Detail verstanden haben. Die nachfolgende Auswertung kann man auch mit reinem Copy & Paste erzeugen – und das in nur wenigen Minuten.

Diesmal geht es darum, schnell zu erkennen, welches Content-Format besonders gut via SEO performt.

DER AUTOR

Mario Fischer ist Herausgeber und Chefredakteur der Website Boosting und seit der ersten Stunde des Webs von Optimierungsmöglichkeiten fasziniert. Er berät namhafte Unternehmen aller Größen und Branchen und lehrt im neu gegründeten Studiengang E-Commerce an der Hochschule für angewandte Wissenschaften in Würzburg.

DER AUTOR

Patrick Lürwer ist Senior Analyst bei get:traction GmbH. Dort ist er für die Datenerfassung, -aufbereitung und -analyse zuständig. Sein tägliches Handwerkszeug sind R, Python und KNIME.

Die Herausforderung

Zu ermitteln, welche Formate wirklich zu echten Verbesserungen beim Ranking führen, ist meist gar nicht so einfach. Zu den High-Performance-Content-Formaten (HPCF) zählen diejenigen, die für mindestens 20 Prozent der Keywords in den Top 10 ranken, also auf Platz 1–10. Hanns Kronenberg hatte hierzu bereits im April 2018 wichtige Gedanken veröffentlicht (*http://einfach.st/sistrix5*). Dreht man die Betrachtung um, könnte man es als den Lieblingscontent von Google und wahrscheinlich auch der (signalerzeugenden) Besucher bezeichnen.

Google findet dieses Content-Format für die betroffene Domain offenbar so relevant, dass es von den Algorithmen stark bevorzugt wird gegenüber den anderen Seiten der Domain. Manche Seiten erzielen sogar einen Top-10-Anteil von 60 bis 80 Prozent. Diese sollte man natürlich gut kennen. Häufig bildet man unterschiedliche Formate in verschiedenen und nach dem Format benannten Verzeichnissen ab, z. B. als Blog (/blog/), als FAQ (/faq/), Tutorials (/tutorial/), als Forum (/forum/), News (/news/) oder andere. Auf die HPCF sollte man sich im Kern konzentrieren. Hier ist die Wahrscheinlichkeit deutlich höher, dass ein Beitrag mit einem Thema/Keyword in gute Rankingpositionen vordringen kann bzw. fast automatisch vordringt, wenn die Domain stark genug ist.

Kronenberg schrieb dazu:

„[…] Erfolgsfaktor sind High-Performance-Content-Formate, die bei fast allen überdurchschnittlich erfolgreichen Websites zu finden sind. Sie sprengen die vorhandenen Grenzen von technischem SEO und ermöglichen ein außergewöhnliches SEO-Wachstum.

Wir reden hier nicht mehr über Wachstumsraten von 10 oder 20 Prozent im Jahr. Mit HPCF sind auch bei technisch bereits gut optimierten Websites Steigerungen der Sichtbarkeit von 100 Prozent, 200 Prozent oder auch 500 Prozent möglich. Die 100 Domains auf der Liste SEO-Newcomer 2017 haben innerhalb eines Jahres ein Wachstum zwischen 168 und 2.130 Prozent hingelegt.“ (Quelle: www.sistrix.de/news/high-performance-content-formate)

Mittels eines kleinen Skripts in R, das Sie wirklich nur kopieren müssen, können Sie die Daten entsprechend aufbereiten und am Ende auch in einem kleinen Portfolio nach Verzeichnissen sortieren. Was Sie dazu benötigen, ist lediglich ein Account beim SEO-Toolanbieter SISTRIX, um über die API die entsprechenden Sichtbarkeitskennzahlen abfragen zu können bzw. zu dürfen.

Den API-Key bei SISTRIX holen

In der SISTRIX-Toolbox findet man rechts oben die Accountsteuerung und weitere prinzipielle Einstellmöglichkeiten. Im vorletzten Punkt, direkt über dem Log-out, ist der Zugang zum API-Key, mit dem man Programmen Dritter (hier R) erlauben kann, für den eigenen Account Daten von der Toolbox abzurufen (Abbildung 1). Daraufhin erscheint der aus Ziffern und Zahlen bestehende Key (Abbildung 2), den man sich von dort einfach kopiert und am besten in einer Textdatei speichert, damit man ihn nicht ständig neu holen muss. Hat man bereits Daten über die API von SISTRIX geholt, erscheint darunter im Log, was wann übertragen wurde und wie viele Credits verbraucht wurden. Pro gebuchtes Modul schreibt SISTRIX jede Woche 10.000 Credits neu gut bzw. füllt die verbrauchten wieder auf. Fragt man die Daten z. B. für fünf Verzeichnisse ab, werden 15 Credits verbraucht, je einer für die Werte „domain.kwcount.seo“, „domain.kwcount.seo.top10“ und „domain.sichbarkeitsindex“ pro Verzeichnis. Also hier fünf (Verzeichnisse) mal drei Kennmetriken = 15 Abfragen.

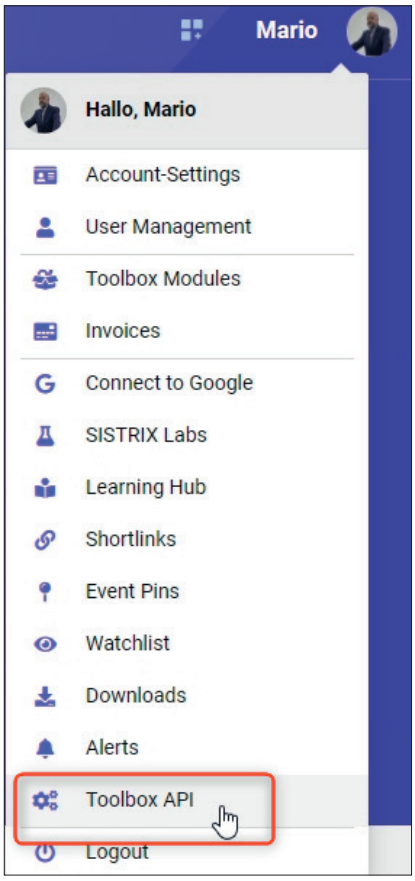


Abb.1: Wo findet man den API-Key bei SISTRIX?

Das kann man bei nicht exzessiven Abfragen sicherlich getrost ignorieren. Pro Woche lassen sich so 666 Abfragen machen, sofern man nur ein Modul gebucht hat. Bei allen fünf SISTRIX-Modulen und 50.000 Credits pro Woche wären theoretisch weit über 3.000 Abfragen möglich. Sind die Credits verbraucht, ist für den Rest der Woche allerdings Schluss (Abbildung 3).

INFO

Wer R noch nicht installiert hat, kann das schnell und einfach bewerkstelligen. Unter cran.r-project.org/bin/ findet man die aktuellen Versionen für Windows (32/64 Bit), MacOS (X) und Linux. Dort im Unterverzeichnis „base“ liegen dann die installierbaren Dateien. Nach der Installation holt man sich am besten gleich unmittelbar die Software „R-Studio“, die sich als eine Art Hülle um den Kern von R als grafische Benutzeroberfläche legt und viele Tools und Eingabehilfen zur Verfügung stellt. Die Bedienung von R wird mit R-Studio fast zum Kinderspiel. Man findet sie unter: *rstudio.com/products/rstudio/download/*

Dies nur als Hinweis, wenn Sie selbst mit dem Code spielen wollen und automatisiert programmgesteuert ggf. deutlich mehr als die 15 Datensätze abholen würden.

Ab in R(-Studio) und los gehts

Wie schon in den ersten beiden Teilen in den letzten Ausgaben erwähnt, erhält R seine große Stärke durch die vielen Funktionserweiterungen, Bibliotheken bzw. Librarys genannt. Jede dieser Librarys muss man bei der allerersten Verwendung in R nachladen bzw. installieren. Das ist allerdings ein einmaliger Vorgang. R merkt sich, welche Librarys man installiert hat, und fortan muss man sie nach dem Aufruf des Programms bei Bedarf nur starten mit dem Befehl library(name_der_library).

Die Library „tidyverse“ wurden schon in den beiden letzten Teilen dieser Serie verwendet. Wer diese ausprobiert hat, in dessen R-Paket ist sie also schon enthalten. Wer jetzt erst einsteigt, muss sie mittels des Befehls „install.packages“ noch erstmalig installieren. Dazu gibt man beide oder nur die zweite Zeile mit der neuen Library „lubridate“ direkt in R ein:

- **install.packages(„tidyverse“)**
- **install.packages(„lubridate“)**

Neu sind die folgenden Pakete:

- **install.packages(„jsonlite“)**
- **install.packages(„httr“)**

HINWEIS

Bitte beachten Sie die Feinheit, dass beim Installieren einer Library deren Name innerhalb der Klammer mit Anführungs- und Schlusszeichen geschrieben wird, also z. B. install.packages(„tidyverse“), beim Aufrufen aber ohne diese, also library(tidyverse). Bei einem Copy & Paste muss man die Anführungszeichen daher manuell entfernen.

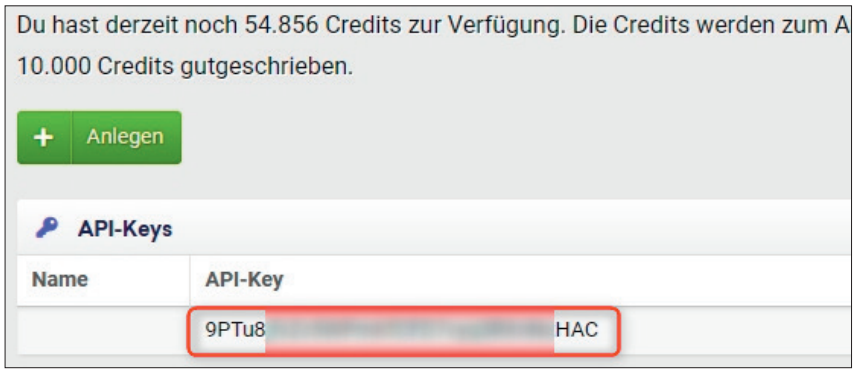


Abb. 2: Den API-Key kopiert man sich von dieser Stelle (hier wurde der mittlere Teil unkenntlich gemacht)

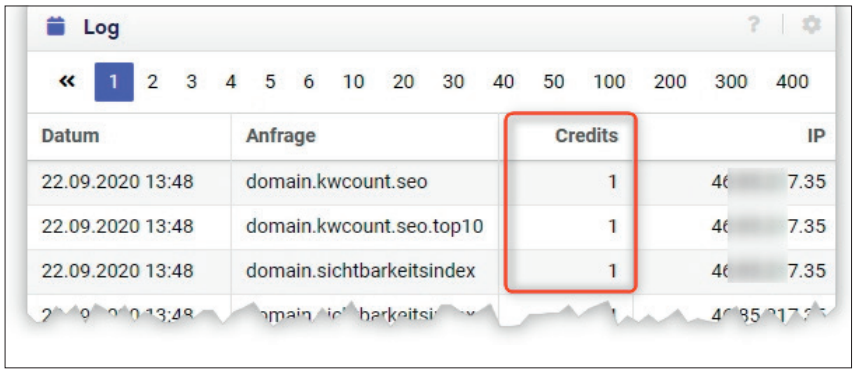


Abb. 3: Pro Verzeichnis werden für die drei nötigen Kennzahlen je drei Credits verbraucht

Diese müssen noch installiert werden, falls man nicht in der Vergangenheit schon mal damit gearbeitet hat. Aber keine Sorge, ein erneutes (Darüber-)Installieren ist völlig unproblematisch. Im Zweifel installieren Sie einfach alle vier Librarys zur reinen Sicherheit. Wen es interessiert: „jsonlite“ hilft dabei, die von API zurückgelieferten Daten effizient in einen sog. Dataframe zu packen und „httr“ unterstützt bei der Abfrage der URLs.

Alternativ zur Kommandozeileneingabe kann man in R-Studio (siehe Kasten) unter „Tools“ und „Install Packages“ auch mit der Maus menügeführt arbeiten. Fängt man an, im entsprechenden Feld (Abbildung 4, Ziffer 1) den Namen zu tippen, erscheint bereits ein Vorschlag, den man einfach übernehmen kann.

Jetzt müssen die beiden eben oder schon vorher installierten Librarys tatsächlich noch aufgerufen bzw. in den Arbeitsspeicher von R geladen werden. Das geschieht mittels des einfachen Befehls „library“ gefolgt vom Namen der Erweiterung in Klammern. In unserem Fall also:

- **library(tidyverse)**
- **librady(lubridate)**
- **library(httr)**
- **library(jsonlite)**

Dieser Aufruf ist im kopierbaren Code bereits vorhanden. Haben Sie die vier Librarys einmal installiert, können Sie in unserem Code am Anfang der vier Zeilen das Zeichen # setzen. Dieses kennzeichnet in R eine Kommentarzeile und veranlasst R, die Zeilen einfach zu ignorieren. Das spart beim erneuten Verwenden Zeit, weil nicht jedes Mal alles installiert wird.

TIPP

Ab und zu erscheinen in R bzw. R-Studio „Warnmeldungen“. Diese können sie in der Regel getrost ignorieren, weil es wirklich nur Hinweise auf z. B. abweichende Versionen von R und einer Erweiterung sind. Solange Sie keine Fehlermeldungen erhalten, läuft alles weiter.

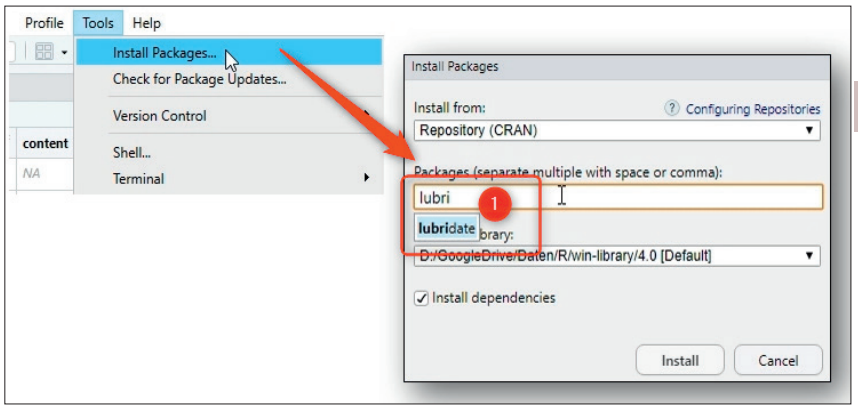


Abb. 4: Auch das geht: Erweiterungen per Menü installieren

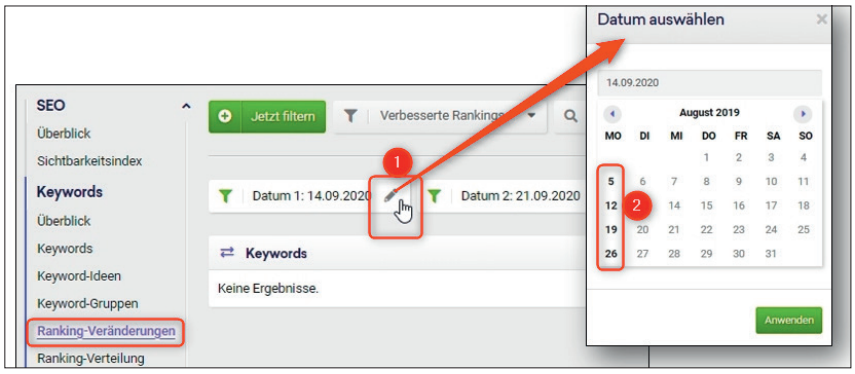


Abb. 5: So finden Sie ein gültiges Crawldatum bei SISTRIX

Was müssen Sie anpassen, bevor das Skript läuft?

Das sind nur zwei Dinge und es ist schnell erledigt. Zuerst tragen Sie den bei SISTRIX kopierten eigenen API-Key in die entsprechende Stelle im Skript ein. Dazu suchen Sie die Stelle

```
API_KEY <- „MEIN_API_KEY“
```

und fügen für MEIN_API_KEY Ihren Key ein. Das sieht dann so aus (der Key hier wurde durch das Einfügen von mehreren X unkenntlich gemacht):

```
API_KEY <- „9P44sf6gXXXXXXXXXXXXXXXXzghHs1L“
```

Anschließend wird die betroffene Domain eingetragen. Auch dazu müssen Sie einfach nur den beispielhaften Eintrag überschreiben unter

```
DOMAIN <- „https://www.domain.de“
```

Jetzt geben Sie das Datum an, für das Sie die Auswertung haben möchten. Beispielhaft ist in unserem Code hier der 22. September 2020 hinterlegt. Dieses Datum überschreiben Sie einfach nach dem Muster Jahr, Monatszahl mit führender Null, also zweistellig, und analog dazu als Letztes das Tagesdatum.

```
DATE <- „2020-09-22“
```

Achtung – eine kleine Fußangel gibt leider noch. SISTRIX stellt nicht für jedes beliebige Datum Werte zur Verfügung. Seit Mitte Juni dieses Jahres gibt es zwar für jeden Tag neue Werte. Davor wurde aber in der Regel immer montags ein neues Datenset eingestellt. Wie in Abbildung 5 gezeigt, wären im August 2019 der 5., 12., 19 und der 26. ein gültiges Datum. Diese sind gegenüber den anderen Tagen fett dargestellt. Für den 12. August 2019 wäre der Eintrag im Skript also z. B. Date <- „2019-08-12“.

Jetzt müssen Sie noch die abzufragenden Verzeichnisse eintragen. Wie oben bereits erklärt, funktioniert das natürlich nur dann vernünftig, wenn Sie Ihre Content-Formate auch sauber in Verzeichnissen organisiert haben. An der Stelle

```
PATHS <- c(
 „https://www.domain.de/de/verzeichnis1/“,
 „https://www.domain.de/de/verzeichnis2/“,
 „https://www.domain.de/de/verzeichnis3/“,
 „https://www.domain.de/de/verzeichnis4/“,
 „https://www.domain.de/de/verzeichnis5/“
)
```

überschreiben Sie einfach die Platzhalter *www.domain.de/de/verzeichnis1/* bis zu *.../verzeichnis5/* mit Ihren eigenen. Wollen Sie weniger Verzeichnisse abfragen, dann löschen Sie einfach die entsprechenden Zeilen. Sollen es mehr sein, dann fügen Sie jeweils eine Zeile hinzu nach dem Muster:

```
„https://www.domain.de/de/verzeichnis-N/“
```

Wollte man z. B. wissen, wie performant bei Aldi Süd die einzelnen Content-Formate funktionieren, würde man die folgenden Einträge verwenden:

```
PATHS <- c(
 „https://www.aldi-sued.de/de/angebote/“,
 „https://www.aldi-sued.de/de/infos/“,
 „https://www.aldi-sued.de/de/ratgeber/“,
 „https://www.aldi-sued.de/de/rezepte/“,
 „https://www.aldi-sued.de/de/sortiment/“
)
```

TIPP

Damit Sie den Code nicht einzeln abtippen müssen, können Sie ihn unter *http://einfach.st/rcode4* als Textdatei downloaden und sich am besten alles auf einmal herauskopieren und in R einfügen. Dann noch Ihren eigenen SISTRIX-API-Code einsetzen, die Verzeichnisse angeben, Return drücken und das fertig berechnete Portfolio erscheint.

Das war schon alles, was zur Vorbereitung gemacht werden muss. Am besten speichern Sie sich Ihr modifiziertes Skript als .R-Datei in Ihrem aktuellen Arbeitsbereich ab (STRG + S oder über das Menü File > Save) und können dann später bei Bedarf darauf zugreifen, ohne erneut etwas ändern zu müssen. Interessant sind ja wie immer die Veränderungen im Zeitverlauf. Wenn Sie das Skript also in definierten Intervallen mit dem jeweiligen Datumseintrag laufen lassen und die Ergebnisse jeweils archivieren oder gegenüberstellen, entsteht eine aufschlussreiche Zeitreihe – sofern Sie parallel tatsächlich auch Ihre Maßnahmen an der Website bzw. diesen Verzeichnissen zurückverfolgen können. Nur dann können Sie Maßnahme und Veränderung einigermaßen in Beziehung zueinander setzen.

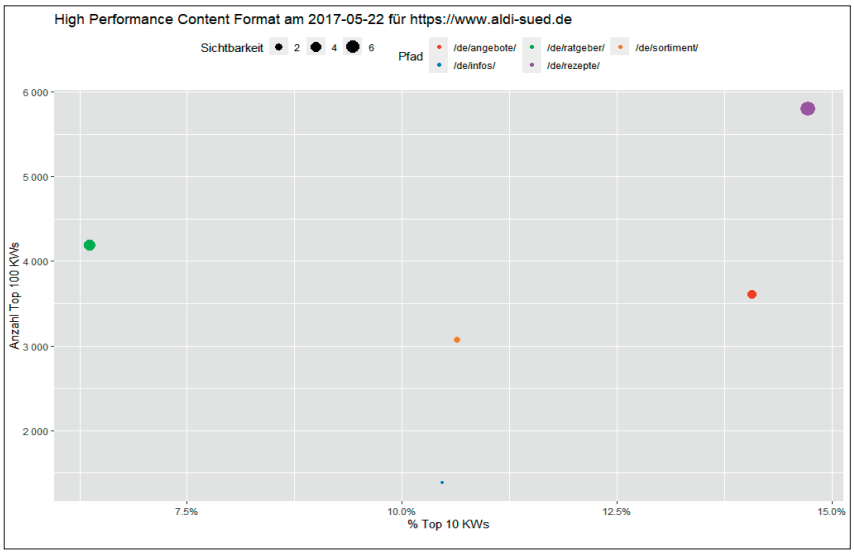


Abb. 6: Beispiel: Welche Content-Art lief im Mai 2017 bei Aldi Süd besonders gut?

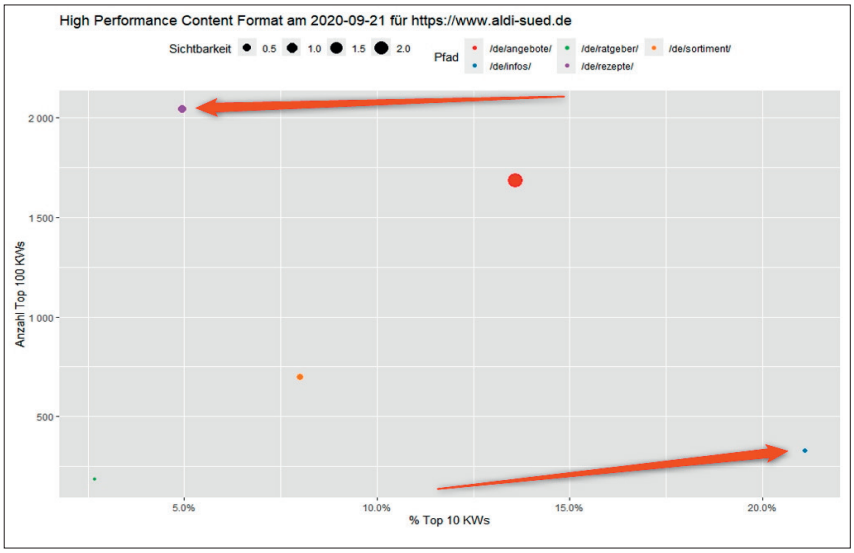


Abb. 7: Wie hat sich die Performance der Content-Formate bei Aldi seither (zum 21. September 2020) entwickelt?

Interpretation der Grafik

Mit dem vorliegenden Skript wurde mit den weiter oben beispielhaft erwähnten fünf Content-Art-Verzeichnissen von aldi-sued.de für den 22. Mai 2017 eine High-Performance-Format-Grafik erzeugt (Abbildung 6). Generell gilt von der Lesart: Rechts oben ist gut, links unten ist schlecht. Je weiter rechts ein Kreis bzw. das dahinter liegende Content-Format liegt, desto mehr Top-10-Rankings hat es. Rein theoretisch verteilen sich 100 % aller Rankings auf die zehn Ergebnisseiten der Top 100. Damit wäre bei Gleichverteilung statistisch gesehen jede Ergebnisseite 1–10 mit je zehn Prozent Rankinganteil dabei. Liegt dieser über diesen 10 % für eine Seite, hier natürlich Seite 1, bedeutet das, dass die Content-Art – genauer das Verzeichnis – im Schnitt besser performt als andere.

In der Abbildung 6 ist dann auch recht eindeutig zu sehen, dass die Content-Art „Rezepte“ (lila) sehr gut von Google angenommen wurde. 14 % der in diesem Verzeichnis über URLs vereinten Rankings sind unter den Top 10. Der Ratgeber (grün) hat nur unterdurchschnittlich viele Top-10-Rankings (6 %), aber viele Treffer in den Top 100. Das bedeutet, dass zwar hier wohl quantitativ sehr viel in Content investiert wurde und damit auch viele Rankings erzielt werden – aber eben nur in den Top 100 auf Plätzen, die in der Regel nahezu keinerlei Traffic bringen. Dort findet man also eher Masse statt Klasse.

Lässt man die Berechnung für ein aktuelles Datum, hier der 21. September 2020, erneut laufen, erkennt man unschwer die gravierenden Veränderungen (Abbildung 7). Betrachtet man die y-Achse mit der Anzahl an Top-100-Keywords, erkennt man eine extreme Abnahme der Rankings. Ging die Achse zuerst bis 6.000, sind es aktuell nur noch 2.000. Die vormals performante Content-Art der Rezepte (lila) ist drastisch eingebrochen, während die Inhalte des Verzeichnisses /infos/ deutlich dazu gewonnen haben.

```
> df
                                path       date     si kw_top_10 kw_top_100 pct_top_10
1  https://www.aldi-sued.de/de/angebote/ 2020-09-21 2.3271        229       1685 0.13590504
2     https://www.aldi-sued.de/de/infos/ 2020-09-21 0.0938         69        327 0.21100917
3  https://www.aldi-sued.de/de/ratgeber/ 2020-09-21 0.0170          5        186 0.02688172
4   https://www.aldi-sued.de/de/rezepte/ 2020-09-21 0.4836        101       2040 0.04950980
5 https://www.aldi-sued.de/de/sortiment/ 2020-09-21 0.1769         56        699 0.08011445
```

Abb. 8: Mit dem Befehl „df“ werden die Werte des Data-Frames ausgegeben

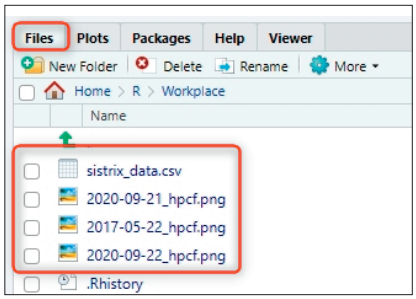


Abb. 9: Von „Plots“ nach „Files“ klicken – dort liegen die Dateien fix und fertig

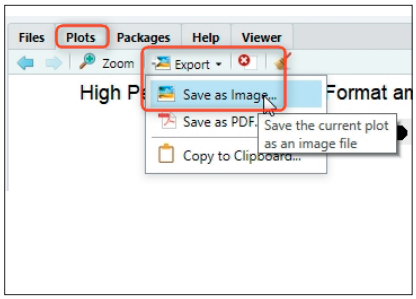


Abb. 10: Grafiken (Plots) lassen sich auch individuell speichern

	A	B	C	D	E	F
1	path	date	si	kw_top_10	kw_top_100	pct_top_10
2	https://www.aldi-sued.de/de/angebote/	21.09.2020	2,327	229	1.685	13,6%
3	https://www.aldi-sued.de/de/infos/	21.09.2020	0,094	69	327	21,1%
4	https://www.aldi-sued.de/de/ratgeber/	21.09.2020	0,017	5	186	2,7%
5	https://www.aldi-sued.de/de/rezepte/	21.09.2020	0,484	101	2.040	5,0%
6	https://www.aldi-sued.de/de/sortiment/	21.09.2020	0,177	56	699	8,0%

Abb. 11: Der Inhalt der Datei „sistrix_data.csv“ zeigt die einzelnen Werte (si = Sichtbarkeitsindex des Verzeichnisses)

Wo liegen Daten und Grafik?

Wenn Sie nach der Berechnung wissen möchten, welche Daten zugrunde liegen, genügt es, in R einfach den Namen des Data-Frames einzugeben. In unserem Fall ist es „df“ ohne Anführungszeichen. Danach listet R die Werte auf (Abbildung 8).

Gleichzeitig schreibt R diese Werte aber auch dauerhaft in die Datei „sistrix_data.csv“ und legt diese im Arbeitsverzeichnis von R ab (Abbildung 9). Ebenso wird standardmäßig die erzeugte Abbildung mit dem gewählten Datum im Namen dort abgespeichert.

Die dafür verantwortlichen Befehle ganz am Ende des Skripts sind

```
ggsave(pasteO(DATE, „_hpcf.png“),
w = 8, h = 6, type = „cairo-png“)
```

für die Abbildung und

```
write_csv2(df, „sistrix_data.csv“)
```

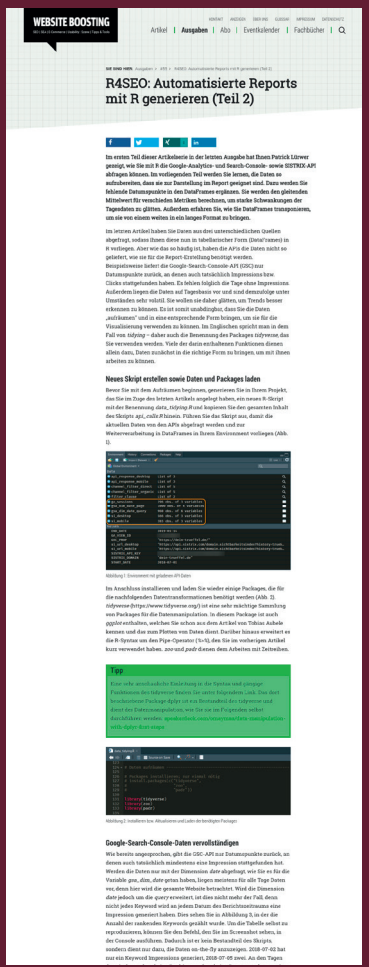
für das Erzeugen einer CSV-Datei mit deutschen Trennzeichen für Zahlen (Abbildung 11). In dieser Datei findet man bei Bedarf die genauen Einzelwerte. Möchte man sich eine Zeitreihe erstellen, kopiert man diese Datei am besten in ein Archivverzeichnis um und kann somit später über eine intelligente Verknüpfung über mehrere Datumswerte hinweg Bewegungsanalysen erstellen.

Fazit

Zu wissen, welche Content-Formate besonders gut bei Google performen, ist durchaus essenziell für eine Erfolgssteuerung beim SEO. Mittels des vorliegenden Skripts sind Sie in der Lage, vergleichsweise schnell und – einmal vorbereitet – innerhalb weniger Sekunden aussagekräftige Visualisierungen zu bestimmten Tagen zu erstellen. Schneller und transparenter dürfte es wahrscheinlich nicht gehen. Und als kleinen Nebeneffekt haben Sie sich vielleicht noch ein Stück weiter in das mächtige Tool R eingearbeitet bzw. verstehen erneut ein paar Kniffe mehr. Worauf warten Sie also noch? ¶

TIPP

Wer Lust bekommen hat, tiefer in R einzusteigen, dem sei die fünfteilige Serie von Patrick Lürwer in der Website Boosting, Ausgaben 54 bis 58, empfohlen. Diese sind bereits frei online und als PDF-Version verfügbar.



Zur Auswahl für die kostenlosen Ausgaben (derzeit Ausgabe 1–58) kommen Sie direkt über *www.websiteboosting.com/magazin.html*.

CODE

```
# Das erste Mal installieren:
install.packages(„jsonlite“)
install.packages(„httr“)

# Nachfolgended werden die Librarys in den Speicher von R geladen:
library(tidyverse)
library(lubridate)
library(httr)
library(jsonlite)

# 1. Config ----------------------------------------------------------------
# Eigenen Sistrix API-Key eintragen an Stelle von: MEIN_API_KEY
# Den API Key bekommt man in der Sistrix Toolbox bei den Einstellungen
API_KEY <- „MEIN_API_KEY“

# 2. Domain definieren
DOMAIN <- „https://www.domain.de/“

# 3. Abzufragendes Datum definieren
# Das Beispieldatum hier bei Bedarf einfach überschreiben
DATE <- „2020-09-21“

# 4. Abzufragende Verzeichnisse definieren
# Dazu werden die Pfade unten entsprechend überschrieben, hier 5 als Vorlage
# Für mehr oder weniger Verzeichnisse Zeilen hinzufügen oder löschen und
# beim letzten Verzeichnis kein Komma mehr dahinter machen!
PATHS <- c(
  „https://www.domain.de/de/verzeichnis1/“,
  „https://www.domain.de/de/verzeichnis2/“,
  „https://www.domain.de/de/verzeichnis3/“
)

# Function definition zur Abfrage der jeweiligen Endpoints ----------
get_sistrix_data <- function(path, date, endpoint = c(„sichtbarkeitsindex“, „kwcount.seo.top10“, „kwcount.seo“)) {
        message(„Get data: „, path)
        query_param <- list(
                path = path,
                date = date,
                format = „json“,
                api_key = API_KEY
        )
        r_json <- fromJSON(
                content(
                        GET(paste0(„https://api.sistrix.com/domain.“, endpoint),
                        query = query_param),
                        as = „text“))
        r_df <- r_json[[„answer“]][[1]][[1]]
        if (is.null(r_df)) {
                r_df <- tibble(path = path, date = date, value = 0)
        } else {
                r_df[„path“] <- path
        }
        return(r_df)
        Sys.sleep(1)
}
```