This post was published to Code Blog at 9:35:00 AM 10/13/2016

Introductory Best Fit Model

Category Sitecore ; Data

 ​

Stored in the CodeShare repository under "AaronS" is a file called "introductory-R-and-best-fit" created back in July as part of a inquiry around Sitecore and performance.

Arthur mentioned during SDM training that any and all performance issues are the worst for troubleshooting; he's right. "This is slow" or "It's kind of laggy" or "Sometimes it takes a while" give us nowhere to start when trying to improve the customer experience.

On June 30th, 2016, the Digital I/S team in partnership with Digitas began making performance tuning adjustments to the strategic web hosting platform. The problem was - how did they know it was working? Casual browsers of Dow.com have no way to tell us that they see things running milliseconds faster, of course.

Because Dow.com has a robust Adobe Analytics implementation, we have access to page load time variables as experienced and reported back by end users' browsers. The next hurdle then is making sense of that much data; in the 2 months since performance tuning started, we collected over 1.6 million data points!

Using Microsoft R Open, an enhanced distribution of R, and a library called "RSiteCatalyst" developed by Randy Zwitch from Comcast, we're able to obtain all those data points and then transform them into something comprehensible.

In this case, all it took was 2 'for' loops and some basic math to prove that the teams had taken over an entire second off the average load time across the whole site, which is a 28% improvement, and continuing to show signs of improving more.

Below are the key snippets of code, with variable naming convention matching "data type"."descriptive name" - so for instance, we use "df.load.time.summary" for the summary of all load times, stored in a data frame, the most common and useful data type in R.

First, the for loops that enumerated the data, taking us from "25 instances of 4 seconds to load a page" and "4753 instances of taking 1.4 seconds to load a page" into a massive frame that summarized all the data points.

for (i in 1:num.dates) {

    df.daily.load.times <- df.page.groups[df.page.groups$datetime == vec.dates[i],]

    vec.load.times <- vector()

    # For every row of this daily frame, generate the load times into a vector for summary

    for (j in 1:nrow(df.daily.load.times)) {

        vec.load.times <- append(vec.load.times, rep(df.daily.load.times$name[j], df.daily.load.times$instances[j]))

    }

    # Bind the summary of this week's data to the previous summaries

    df.load.time.summary <- rbind(df.load.time.summary, summary(vec.load.times))

}

And below, the simple commands for creating a first-order linear model and second-order linear model to better show a consumer the trend.

# Linear model, first and second order, for the mean load time by the date

lin.mod <- lm(df.load.time.summary$Mean ~ time)

lin.mod2 <- lm(df.load.time.summary$Mean ~ I(time ^ 2) + time)

# Basic generic prediction of load time based on the linear model

pr.lm <- predict(lin.mod)

pr.lm2 <- predict(lin.mod2)

Ultimately, the graphs we generate show the improvement.

The black line is the actual average page load time for a given day. The blue line is the first-order model for page load time, and the green line is the second-order model. You can see the difference between first and second order models are nominal in this case, so we can base on either of these.

We can take this simple analysis in many directions from here, such as:

* Plotting using a box-and-whiskers plot, to give more granular insight into page loads
* As the amount of data increases, modeling new ways
* Take cross-sections of data to show improvements by section of websites (Sustainability, Markets, Careers, etc)
* Include more data sources and variables to prove downstream effects. We know this helps with Search Engine Optimization - but how much did it help and where?
* Improving the code to use 'sapply' instead of 'for' loops, and others
* Using a more robust statistic such as median, given that the data is very right-skewed
* And many more...