(Short version)
Quantitatively Dealing with Javascript Fatigue

You probably use at most 3 numbers when deciding whether a Node.js boilerplate or library is worthwhile: Github stars, npm downloads, and Hacker News votes. The rest of the time is spent manually reading Twitter or Medium posts, listening to podcasts, and even attending conferences like this one! Javascript Fatigue is a symptom; lack of objective metrics is a big cause.

As a former Wall Street guy learning to code, I wrestled with this the traditional way until I realized that we can exploit basic assumptions in Node.js and Github to quantify the cognitive load and community strength of Node.js projects.

If you've ever wished for a better way to deal with Javascript Fatigue, you'll gain some new perspectives on what you can quantify as I walk through a live demo implementation! How much faster will you learn, once you know how fast you will learn?

(Old Long version)
Evaluating Node.js Projects like Stocks

Developers of all stripes feel the weight of Javascript Fatigue, but have an intrinsic desire to find the next great game-changing boilerplate or library to add to their arsenal. However the primary metrics of discovery so far is manual evaluation proxied by Github stars, npm downloads, Hacker News votes, and well followed twitter accounts. What if there was an easier way to crunch more objective numbers?

To anyone from Wall Street, this is an extremely familiar problem. Every public company stock has its own story and ebbs and flows among a sea of peers and competitors. The choice between going broad and going deep is constrained by an analyst's time, just like a programmer who even before learning something has to figure out if something is worth learning. However while the financial world has developed Generally Accepted Accounting Principles that allow agnostic comparison between stocks, developers have not progressed beyond what basically amounts to counting bookmarks to quantify project quality.

This talk is explores building a system to exploit basic assumptions in the Node.js and Github ecosystem to do useful things like quantifying and comparing the cognitive load of boilerplates and libraries, or evaluating the strength of an open source community. Just like with stocks, numbers will never tell the whole story about any given project. But it is possible to use code to reveal useful information about code and we should do it more often.