FUNKTIONALITÄTEN

wird in der Seminararbeit in Kapitel 4 behandelt

BENUTZUNG

Mit Anaconda Prompt in das Verzeichnis navigieren, in welches das zip-Archiv entpackt wurde.

runPrototype.bat eingeben und Enter drücken.

Den Anweisungen des Skripts folgen. (es wird zuerst nach einer Metriken csv Datei gefragt und dann nach einer Findings csv Datei. Beispielhafte Dateien befinden sich bereits in den Ordnern *FindingsCSVs* und *MetrikenCSVs*)

Bei größeren Datenmengen kann die Verarbeitung etwas dauern (z.B. braucht JSweet bei mir etwa 30 Sekunden). Nach der Verarbeitung öffnet sich die Excel Datei „Pivotbericht.xlsm“ und aktualisiert die Pivottabelle automatisch (auch die Aktualisierung kann ein paar Sekunden dauern (ganz unten in Excel ist eine schmale Leiste, welche anzeigt ob die Aktualisierung schon fertig ist).

Nun kann das Tool zur Analyse benutzt werden. Die beiden blauen Buttons starten Makros. Deren Verwendung versteht man leicht durch Ausprobieren.

HINTERGRUNDINFOS ZUM PYTHON SKRIPT

Das Python Tool verarbeitet csv Dateien, die Metriken von Methoden beinhalten und csv Dateien, die Findings enthalten.

Das entsprechende Trennzeichen wird automatisch erkannt, z.B. Komma, Semikolon oder Pipe.

Folgende Struktur müssen die Metriken-CSVs allerdings einhalten:
1. In der ersten Zeile müssen die Spaltenüberschriften stehen und ab der zweiten Zeile muss jede Zeile für eine Methode stehen
2. Es muss eine Spalte geben, ab der die Metriken beginnen und rechts davon darf auch nichts anderes mehr kommen als Metriken
3. Es muss eine Spalte oder eine Kombination von Spalten geben, die eine Methode eindeutig indentifizieren. Optimalerweise lässt sich daraus auch der Name der Methode herauslesen

Das Python Skript fängt falsche Nutzereingaben ab, lediglich nicht bei der Eingabe zur Bildung der ID-Spalte.

Nicht-numerische Daten in den Metriken Spalten werden durch Null ersetzt.

Für jede Methode wird die Mahalanobis Distanz berechnet, dies ist ein Standardverfahren der multivariaten Analyse.

Zu jeder Methode wird die Anzahl der Findings berechnet. Hierbei gelten folgende Annahmen:
1. Die Findings liegen als CSV Datei vor, wobei in der ersten Zeile Überschriften stehen müssen und ab dann jede Zeile für ein Finding steht
2. Es gibt eine Spalte, in der der Pfad der Klasse steht, und in diesem Pfad kommt der Substring „src\“ (oder „src/“) vor.
3. In derselben Spalte von Punkt 2 steht nach dem Klassennamen die Startzeile und Endzeile des Findings, getrennt durch ein Minuszeichen.

Anmerkung:
Mithilfe der Information des Pfades und der Start- und Endzeilen ordnet das Skript dann die Findings den Methoden zu. Hierbei mache ich mir zunutze, dass es in den von Sourcemeter erzeugten CSVs je eine Spalte mit Path, Start- und Endline gibt. Das ist alles hart gecodet.
Man kann also sagen, dass die Zuordnung der Findings nur mit Findings-CSVs von Teamscale funktioniert und mit Metriken-CSVs von Sourcemeter. Meiner Ansicht nach muss man das hart coden und kann es nicht dynamsich gestalten, da es ja sein könnte dass in einem anderen Metriken Tool die Start und Endzeile einer Methode in der gleichen Spalte stehen, oder einen anderen Namen/Index haben als in Sourcemeter. Gleiches gilt für die Findings-CSVs, dort wäre es auch möglich, dass eine anderen Tool als Teamscale nicht alle Infos in einer Spalte hat, sondern z.B. den Pfad in einer Spalte und die Start- und Endzeilen in einer extra Spalte. Theoretisch könnte man das alles den Nutzer fragen, dann wird es aber nervig zu benutzen.

Das Python Skript bildet anschießend noch für jede Metrik eine normierte Werte in je einer neuen Spalte. Diese neuen Spalten haben alle den Namen der ursprünglichen Spalte + „_normed“

Zuletzt wird das Ergebnis als .xlsx in den Ordner „BasisdatenFuerPivotbericht“ abgespeichert.