Leslie Rodrigues 10-10613
Erick Silva 11-10969
Georvic Tur 12-11402

Análisis de *Clusters* Aplicado a Repositorios Públicos de Github

Objetivo

Determinar si los *clusters* de lenguajes de programación usados de manera conjunta en repositorios de Github revelan los perfiles de programación de los usuarios.

Justificación

Los empleadores en el área de las ciencias de la computación necesitan determinar de manera rápida quiénes, entre todos sus candidatos, cumplen de mejor manera con su perfil profesional buscado.

Técnicas y herramientas utilizadas

ALGORITMOS:

- Expectation Maximization
- DBSCAN
- K-Means
- Jerárquico Aglomerativo
- Jerárquico Divisivo

Clustering

Conjunto de Datos

Se usó el API de Github para coleccionar los datos, obteniendo familias de usuarios al recorrer los seguidores de usuarios en bfs para recolectar los lenguajes usados en sus repositorios.

RECOLECCION DE DATOS

1 raiz

- Datos de 491 usuarios, 8127 repositorios y se encontraron alrededor de 200 lenguajes.
- Datos de 1403 usuarios, 25898 repositorios, con 254 lenguajes.

2 raíces:

2300 usuarios, 38558 repositorios, con 265 lenguajes.

PREPROCESAMIENTO

Para diferentes experimentos se realizaron diferentes medidas de distancias y se excluyeron o no lenguajes menos usados

La data de Github también se encuentra disponible por el API BigQuery de google, pero desde 2015 este no hace un buen tracking de los usos de cada lenguaje.

EXPERIMENTOS

Primer Experimento

Frecuencias de los lenguajes en el conjunto de datos usado en este experimento

Lenguaje	frecuencia
JavaScript	2118
HTML	1655
CSS	1623
Python	1457
Shell	1252
Ruby	628
Makefile	625

Lenguaje	frecuencia
C++	592
C	590
Objective-C	377
PHP	372
Batchfile	217
TeX	197
CoffeeScript	187

Lenguaje	frecuencia
Perl	183
C#	157
Jupyter Notebook	149
Go	139
Swift	116
Scala	111

Primer Experimento

Cada instancia representa a un lenguaje

PHP, Perl, **Java**, **CSS, HTML**, C++, Batchfile, Groff, Fortran, Processing, CMake, **Kotlin**, **FreeMarker**, **Python**, **Ruby**, Makefile, **JavaScript**, Objective-C++, Objective-C, NSIS, Assembly, **Shell**, C, **Handlebars**

Lua, **Dart**, **LiveScript**, **Haxe**, **Clojure**, **JSONiq ***, **Elm**, **Rust**, **ColdFusion**, Go, Vala, **D**, Cucumber, Cirru, **Scala**, **Verilog**, Liquid, GLSL, LSL, PowerShell, TeX, **VHDL**, Lean, **Haskell**, **R**, Mask, ABAP, Swift, OpenSCAD, Forth, **Elixir**, Groovy, **OCaml**, **Scheme**, C#, **TypeScript**, Io, Tcl, Nix, XQuery, 'Protocol Buffer', **Matlab**, **Erlang**, **ActionScript**, COBOL, Pascal, Ada, AutoHotkey, '**Common Lisp**', **Julia**, 'Visual Basic', **CoffeeScript**, Eiffel,

K = 3 con EM. Sólo se muestran algunos clusters

Primer Experimento

PHP, Lua, Clojure, D, Scala, TeX, Haskell, **R**, **Ruby**, Elixir, OCaml, Scheme, C#, **Matlab**, Erlang, 'Common Lisp'

CSS, HTML, Python, JavaScript, Shell

Dart, LiveScript, Haxe, **JSONiq**, **Elm**, Rust, ColdFusion, Go, Vala, Cucumber, Cirru, **Verilog**, Liquid, GLSL, LSL, PowerShell, **VHDL**, Lean, Mask, ABAP, OpenSCAD, Forth, Groovy, **TypeScript**, Io, Tcl, Nix, XQuery, 'Protocol Buffer', ActionScript, COBOL, Pascal, Ada, AutoHotkey, Julia, 'Visual Basic', **CoffeeScript**, Eiffel

---

Perl, Java, **C++**, Batchfile, **Makefile, Objective-C, Assembly, C**

K = 10 con EM. Sólo se muestran algunos clusters

Primer Experimento

CSS, HTML, Python, Makefile, **JavaScript, Shell**

Perl, **C++,** Batchfile, **Objective-C, Assembly, C**

D, TeX, **R**, **Elixir, Scheme**, **Matlab**, **'Common Lisp'**

PHP, Lua, **Dart, LiveScript, Haxe**, **Clojure**, **JSONiq, Elm**, Rust, **ColdFusion**, Go, Vala, Cucumber, Cirru, Verilog, Liquid, GLSL, LSL, PowerShell, VHDL, Lean, **Haskell**, Mask, ABAP, Ruby, OpenSCAD, Forth, Groovy, **OCaml,** C#, Io, Tcl, Nix, XQuery, 'Protocol Buffer', **Erlang**, **ActionScript,** COBOL, Pascal, Ada, AutoHotkey, Julia, 'Visual Basic', **CoffeeScript**, Eiffel

K = 20 con K-Means. Sólo se muestran algunos clusters

Segundo Experimento

Cada instancia representa a un repositorio.

Se usó EM y K-Means con el conjunto de datos de 8127 repositorios.

Casi todos los lenguajes se agrupaban en un solo cluster.

Tercer Experimento - Lenguajes por usuario

En este experimento se trata de usar las probabilidades condicionales de que cada lenguaje sea usado por un usuario para calcular relaciones entre ellos. Cosas a notar:

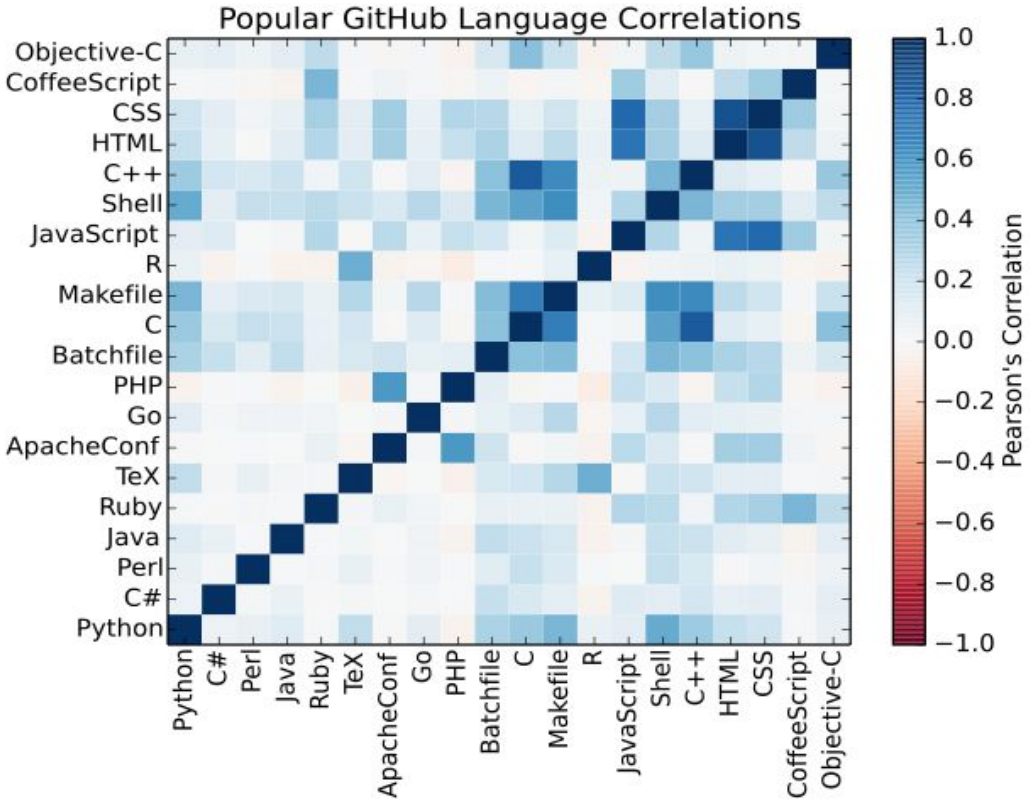
- La probabilidad condicional no es simétrica.
- Los algoritmos de clustering con distancias no simétricas suelen obtener malos resultados.
- Esta medida puede ser afectada por la cantidad de usos de un lenguaje, principalmente por lenguajes que tengan muy pocos usos.

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

Tercer Experimento - Tabla de Probabilidades Condicionales

Lenguaje	PHP	CSS	Makefile	C	Java	HTML	JS	Shell	Ruby	Python
PHP	1.00	0.93	0.55	0.47	0.54	0.89	0.96	0.82	0.59	0.68
CSS	0.53	1.00	0.54	0.48	0.52	0.91	0.96	0.80	0.58	0.69
Makefile	0.53	0.92	1.00	0.69	0.62	0.92	0.94	0.94	0.65	0.86
C	0.48	0.88	0.74	1.00	0.67	0.86	0.90	0.91	0.64	0.83
Java	0.51	0.87	0.62	0.62	1.00	0.86	0.89	0.84	0.59	0.75
HTML	0.52	0.94	0.56	0.49	0.53	1.00	0.94	0.80	0.57	0.70
JS	0.53	0.93	0.54	0.48	0.52	0.88	1.00	0.79	0.57	0.67
Shell	0.52	0.89	0.62	0.56	0.56	0.87	0.92	1.00	0.61	0.75
Ruby	0.54	0.92	0.61	0.56	0.56	0.88	0.95	0.88	1.00	0.72
Python	0.50	0.89	0.66	0.59	0.58	0.88	0.90	0.87	0.59	1.00

Tercer Experimento - Gráfico de Correlaciones de Pearson

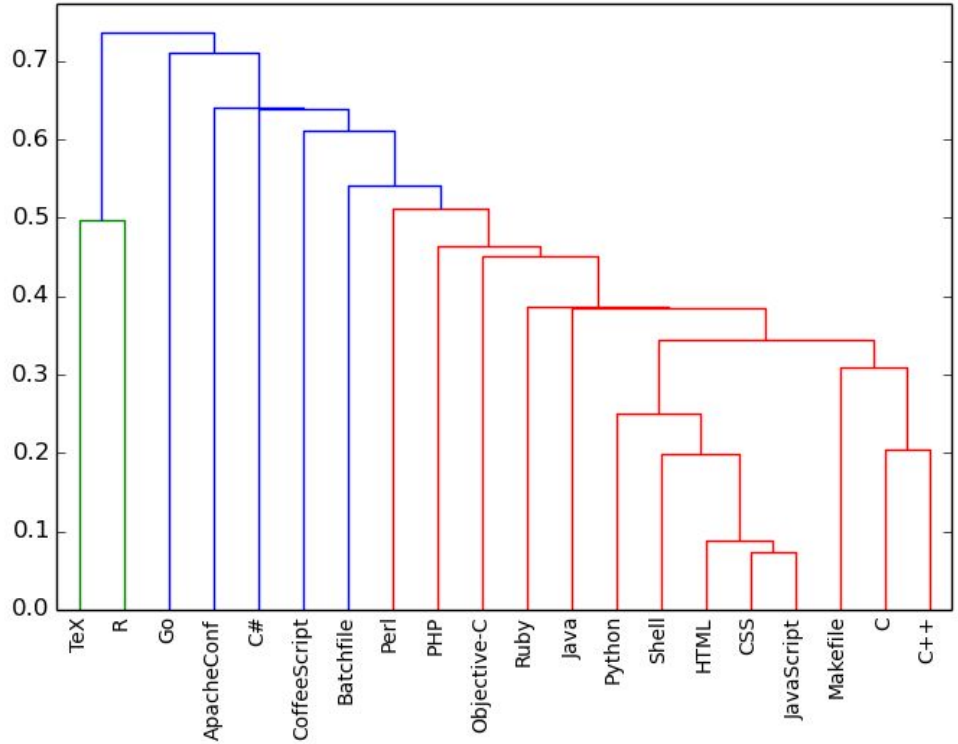


Tercer Experimento - Medidas de distancia

Notamos que la probabilidad condicional no es simétrica, así que necesitamos una medida de distancia simétrica para poder usar algoritmos de clustering:

- Para obtener distancia en lugar de similitud se usa (1 - P(A|B))
- Para hacer a la matriz simétrica, para todo A y B su distancia es max(1-P(A|B), 1-P(B|A)).
- El usar el máximo nos permite disminuir el efecto que tienen los lenguajes que se usan mucho en esta distancia.

Tercer Experimento - Dendograma



Tercer Experimento - DBSCAN

Vemos que lenguajes muy poco conocidos toman la mayoría de los clusters.

- Eps = 0.5
- Min_samples = 2

Contenido
Agda, Turing, Cool, J, Oz, Golo, ooc, XProc, Pike, Modula-2, Tea, Ioke, LOLCODE, Opa, Boo, Idris, Fantom, Lasso, Omgrofl, Befunge, Squirrel, Factor, PureScript, Csound, Dogescript
JSONiq, Io, VHDL, Nix, Vala, XQuery, Forth, ABAP, Verilog, COBOL, Eiffel, Mask, Ada, Lean, Cirru, OpenSCAD, AutoHotkey, Liquid, Elm, LSL, Haxe
Ruby, HTML, Java, PHP, Shell, Python, Objective-C, CSS, JavaScript, C++, C, Makefile
Nemerle, BlitzBasic

IGOR Pro, ChucK, KRL
UrWeb, Cycript, Fancy
SAS, Module Management System
REXX, Ragel
Myghty, Genshi
PicoLisp, GDScript
TeX, R
Lex, Yacc

Conclusiones

El agrupamiento a nivel de lenguajes y de usuarios es el más efectivo.

Entre los veinte lenguajes más usados en el conjunto de datos es la relativa ausencia de correlaciones negativas.

La cantidad de lenguajes muy usados es mucho menor que la cantidad lenguajes.