Log clustering tool

Resumen

Contenido

1. Introducción

Hoy en día se genera una gran cantidad de archivos de logs en campos como la industria y la ciencia, ya que estos son una fuente de información muy importante y útil en muchas situaciones, Sin embargo, a medida que aumenta la complejidad de los sistemas, el análisis de los archivos de logs es cada vez más exigente y dificultoso, ya que hay que realizar un gran esfuerzo para recopilar, almacenar e indexar una gran cantidad de logs que se agrava más cuando estos logs no son heterogéneos (1).

Toda esta producción de logs hace, que, debido al tamaño de estos conjuntos de datos, que las soluciones de bases de datos convencionales no sean las adecuadas para el análisis de la información y en su lugar se consideren más apropiados bases de datos virtuales combinadas con sistemas de procesamiento distribuidos y paralelo.

Con todo esto, se puede suponer que el análisis de logs es un caso de uso de big data y por lo tanto es un gran desafío para su procesamiento, almacenamiento, variedad y su gestión con los recursos disponibles. Además, hay que añadir que cuando los logs provienen de múltiples fuentes surgen problemas con la extracción de contenido significativo y su correlación. Por todo esto surgen varias soluciones eficientes para tratar, reconocer y almacenar la información importante y que se puedan recuperar o migrar fácilmente entre los diferentes centros de datos.

En definitiva, el tratamiento de logs hoy en día tiene una gran importancia y por esta razón se han desarrollado muchos algoritmos para ello. Esto implica que tiene que existir computadoras muy potentes para dicho tratamiento, ya que una maquina genera muchos logs.

Con la monitorización de los archivos de logs se pueden detectar errores y/o anomalías en el funcionamiento de la máquina que genera los logs. Cuando se manifiesta un error, tiene que haber un experto que haya almacenado dicho error en una base de datos o algo similar, sin embargo, si el error que se ha generado no esta catalogado no puede ser detectado, pero llevar todo este trabajo a cabo lleva mucho tiempo y esfuerzo y además es propenso a que se cometan errores (2).

El objetivo de este trabajo es analizar logs a través de varios algoritmos de clusterización (que agrupa los logs en grupos y según sus patrón), para llevar a cabo estas tareas se trabajara con una base de datos no relacional (elasticsearch), con una herramienta de extracción, transformación y carga (logstash), con un framework de computación en clúster (apache spark) y con el lenguaje de programación Python y sus librerías, todas estas herramientas son open-source y están indicadas para trabajar con gran cantidad de datos.

2. Herramientas y metodología.

Para la realización de este trabajo se van a utilizar las herramientas que se describen a continuación.

2.1 Python.

Python (3) es un lenguaje de programación que surgió en 1991 con la idea de que su sintaxis haga que el código sea legible más fácilmente y que en la actualidad tiene dos versiones estables que son la 3.7 y la 2.7 (que es la que se utiliza en este trabajo), la razón por la que aún se está dando soporte a la versión 2.7 es porque de esta versión se pasa a la 3.0 y hay un gran cambio cuando uno se pone a desarrollar su código o quiere realizar una migración.

Python es un lenguaje interpretado (no hay que compilar el código antes de su ejecución) y multiplataforma por lo que se puede usar en varios sistemas operativos distintos como puede ser Windows, Ubuntu o Mac, además se pueden crear todo tipo de programas ya que no está diseñado para un único propósito. Asimismo, soporta la programación orientada a objetos y en muchos casos ofrece una manera sencilla de crear programas con componentes reutilizables.

Por último, dispone de muchas funciones incorporadas en el propio lenguaje y además existen muchas librerías que podemos importar en los programas para tratar temas específicos como la búsqueda de patrones usando expresiones regulares o hacer graficas de los datos obtenidos o disponibles. Las tres librerías principales que se utilizaran en este proyecto son:

- Matplotlib: librería que genera figuras de calidad en una gran variedad de formatos a partir de datos contenidos en listas o arrays.
- Re: librería que permite verificar si una expresión regular dada coincide con una cadena en particular.
- Pyspark: librería que permite utilizar todas las funciones de apache spark utilizando el lenguaje Python.

Solo cabe señalar que para desarrollar todo el código se hizo uso de un entorno de desarrollo integrado en inglés Integrated Development Environment (IDE) que es una aplicación informática que tiene muchas funcionalidades y servicios que facilitan el desarrollo del programa a realizar, en nuestro caso hicimos uso de dos IDE diferentes que fueron.

- Microsoft Visual Studio.
- Wing Python IDE.

2.2 Elasticsearch.

Elasticsearch (4) es una herramienta Open-source desarrollada por la compañía elastic, que nos permite indexar una gran cantidad de datos para, posteriormente, realizar consultas sobre ellos, ya sea realizando búsquedas aproximadas o un texto completo, ya que al estar la información almacenada indexada los resultados se obtienen de forma rápida. Elasticsearch funciona mediante una interfaz REST recibiendo y enviando datos en formato JSON y permite que pueda ser usadas por varias plataformas como puede ser Java, Python, .Net o un navegador con JavaScript, además la información que se almacena es persistente.

A continuación, se muestra un ejemplo de como se puede añadir un log y visualizar la información que se ha añadido en elasticsearch, para ver este ejemplo se hará uso del siguiente log:

Jul 26 09:36:29 RUE3 anacron[1102]: Job `cron.daily' terminated

Para insertar este log en elasticsearch basta con usar un método post y se inserta de forma directa. Seguidamente se hace una consulta a esta base de datos para ver como se a almacenado la información, esta consulta se puede realizar de varias formas diferentes y para este caso se ha usado el navegador web para conectarse a elasticsearch con la siguiente url *http://localhost:9200/trabajo_master/_search/?pretty* y el resultado obtenido ha sido el que se muestra en la figura 1.


```
{
  "took" : 6,
  "timed_out" : false,
  "_shards" : {
    "total" : 5,
    "successful" : 5,
    "skipped" : 0,
    "failed" : 0
  },
  "hits" : {
    "total" : 1,
    "max_score" : 1.0,
    "hits" : [
      {
        "_index" : "trabajo_master",
        "_type" : "doc",
        "_id" : "XdOqGmUB2v_tBaLzpYNl",
        "_score" : 1.0,
        "_source" : {
          "host" : "Javier",
          "@timestamp" : "2018-08-08T17:51:54.939Z",
          "message" : "Jul 26 09:36:29 RUE3 anacron[1102]: Job `cron.daily' terminated\r",
          "@version" : "1"
        }
      }
    ]
  }
}
```


Figura 1. Resultado de como se almaceno el log anterior en elasticsearch.

Como puede apreciarse en la figura 1 tras insertar el log en elasticsearch se ha creado de forma automática un archivo Json que indexa y almacena la información, además se puede ver como añade de forma automática varios campos como son @timestamp en el que almacena la hora en la que se ha insertado el evento y @version que almacena la versión del documento un campo, por último destacar que también se añade un campo _id que es un identificador de cada documento Json y que tiene que ser único.

2.3 Logstash.

Logstash (5) es una herramienta Open-source desarrollada por la compañía elastic que permite extraer, transformar y cargar la información en elasticsearch. Para poder realizar todo esto soporta varias entradas, códecs, filtros y salidas. La fuente donde se encuentra la información de datos es la entrada, los códecs sirven para cambiar formatos de entrada y/o salida, para transformar la información se utilizan los filtros de esta forma se procesan los eventos, finalmente las salidas son los destinos a los cuales se quieren enviar los datos tras ser procesados.

Vamos a ver un ejemplo del uso de logstash con el log definido anteriormente, en el cual puede apreciarse de una forma evidente que empieza por un mes seguido del día y de la hora, a la vista de esto se deduce que se puede obtener un campo extra en el que se almacene la fecha y hora en la que se generó el log (antes de insertarlo en elasticsearch), para realizar esto se hace uso de las funcionalidades disponibles y se escribe la secuencia tal como se muestra en la figura 2.

```
input
{
    stdin { }
}
filter
{
    grok
    {
        match => { "message" =>"(?<time_recive>[A-Za-z]{3}\s*[0-9]{2}\s*\S*)"}
    }
}

output
{
    elasticsearch
    {
        index => "trabajo_master"
    }
}
```

Figura 2. Código usado para transformar y obtener un campo extra del log de ejemplo.

En la figura 2 se observa como la entrada del log es por pantalla y a este se le aplica un filtro que almacena en la variable time_recive una fecha y hora tras hacer uso de una expresión regular y por último almacena toda la información en elasticsearch con el índice trabajo_master.

El documento que se ha insertado tendrá el mismo formato que el de la figura 1 salvo porque el _id ha cambiado (recordemos que es un identificador único) y porque se ha añadió un campo time_recive que contiene la información parseada, esto se puede apreciar en la figura 3.


```
{
  "_index" : "trabajo_master",
  "_type" : "doc",
  "_id" : "dZe1H2UBj2d1dmRNKp_9",
  "_score" : 1.0,
  "_source" : {
    "host" : "Javier",
    "@timestamp" : "2018-08-09T17:21:31.082Z",
    "@version" : "1",
    "time_recive" : "Jul 26 09:36:29",
    "message" : "Jul 26 09:36:29 RUE3 anacron[1102]: Job `cron.daily' terminated\r"
  }
}
```


Figura 3. Resultado de como se almaceno el log en elasticsearch tras usar logstash.

2.4 Kibana.

Kibana (6) es una herramienta Open-source desarrollada por la compañía elastic que permite realizar exploraciones visuales y análisis en tiempo real de los datos almacenados en Elasticsearch, ya que se pueden diseñar visualizaciones y dashboards.

A continuación, vemos un ejemplo en el que en un dashboards se muestran dos visualizaciones, en la primera de ellas se puede ver un histograma que muestra cuando se han ido recibiendo los logs en el tiempo y en la segunda visualización se muestra el log, todo esto se puede ver en la figura 4.

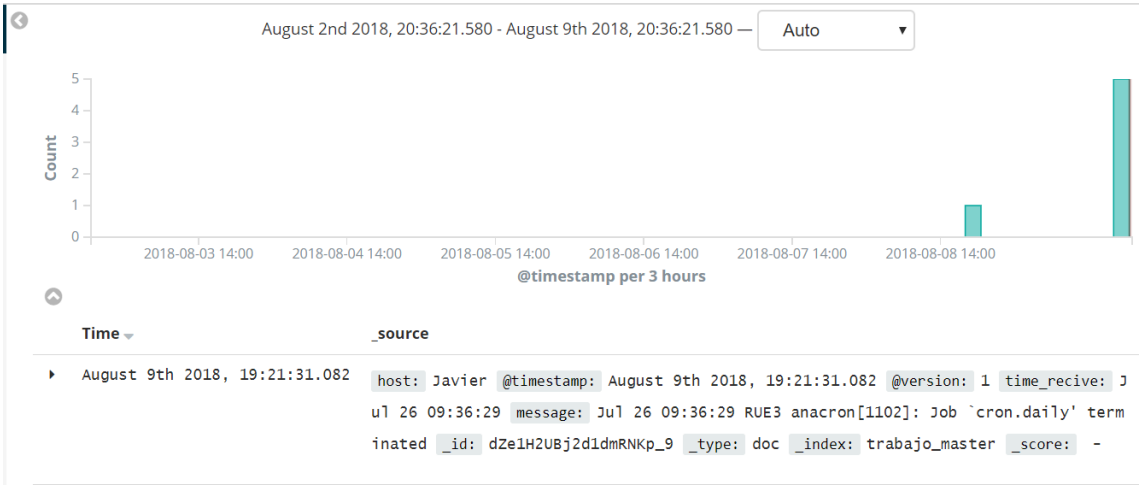


Figura 4. Dashboard que contiene dos visualizaciones sobre los logs.

2.5 Apache Spark.

Apache Spark (7) es una infraestructura informática de código abierto usado para trabajar con gran volumen de datos ya que gestiona el uso de éstos en memoria que surgió en el año 2009 dentro de un proyecto de investigación en la Universidad de Berkeley y que en el año 2013 fue donado a la fundación Apache Software Foundation, en agosto de 2018 se encuentra por la versión 2.3.1.

Las aplicaciones que usan Spark son realizadas de forma independientes y son coordinadas por el objeto SparkContext que se encuentra en el programa principal y que es capaz de conectarse a gestores de cluster que se encargaran de asignar los recursos que hay en el sistema para el mejor funcionamiento de la aplicación.

Para trabajar con Spark hay que conocer el concepto básico de Resilient Distribut Dataset (RDD) que son grupos de datos de lectura que están cargados en memoria (y que se pueden dividir para ser tratados de forma paralela) para realizar dos tipos diferentes de operaciones; acciones y/o transformaciones.

- **Acciones**: transmite el valor de un RDD a la aplicación, la función **count()** es un ejemplo de acción sobre un RDD ya que cuenta los elementos que posee el mismo, también la función **take(n)** devuelve un array con los primero **n** elementos del RDD y por último esta la función **collect()** que también devuelve en un array todos los elementos de un RDD
- **Transformaciones**: radica en obtener un nuevo RDD tras modificar el original, se pueden definir dos tipos diferentes de operaciones de transformación ya que lo mas probable es que los datos se encuentren en mas de un RDD, estos son:
    - **Narrow:** este tipo de operación se utiliza cuando los datos que se quieren tratar están en la misma distribución del RDD y no hace falta mezclarlos entre ellos, algunos ejemplos son **map(func)** que crea un nuevo RDD a partir de otro aplicando una transformación a cada elemento original o **filter(func)** que crea un RDD nuevo manteniendo solo los elementos del RDD original que cumplen una determinada condición
    - **Wide:** este tipo de operación se utiliza cuando los datos a tratar están situados en diferentes particiones de un RDD y es necesario que se mezclen estas particiones, algunos ejemplos son **groupByKey()** que agrupa los RDD o **reduceByKey()** que los reduce.

De esta forma, se pueden realizar operaciones de gran cantidad de datos de forma rápida y flexible a los fallos, además Spark cuenta con una API que permite realizar conexiones con repositorios de datos como Hadoop, Cassandra, SQL y también ser usado con otros lenguajes de programación como Python, R o Java.

2.5.1 Transformaciones y acciones utilizadas.

Spark utiliza un mecanismo de “evaluación perezosa” esto quiere decir que no se ejecuta una transformación en un RDD hasta que no se realiza alguna acción sobre el mismo.

A continuación, se muestran las operaciones de transformaciones y acciones utilizadas en este trabajo.

- **Transformación map()**: aplica una transformación a cada elemento del RDD original. A continuación, se muestra un ejemplo del funcionamiento.

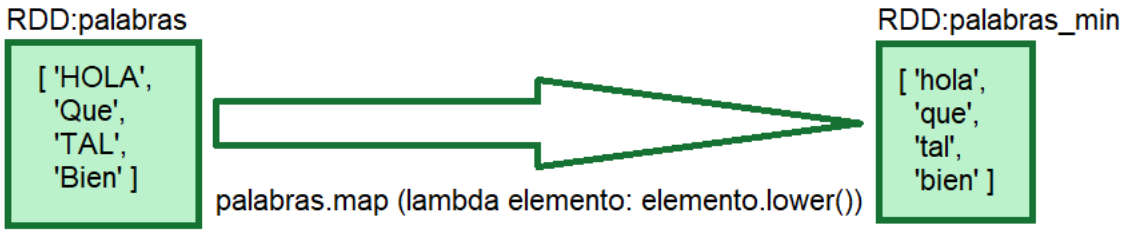


Figura 5. Ejemplo de funcionamiento de la transformación map(func) en un RDD primario.

- **Transformación flatMap():** aplica una transformación a cada elemento del RDD original pero cada elemento puede crear cero o más elementos. A continuación, se muestra un ejemplo en el que se le aplica la función Split (que devuelve una lista con las palabras de una cadena) a cada elemento del RDD.

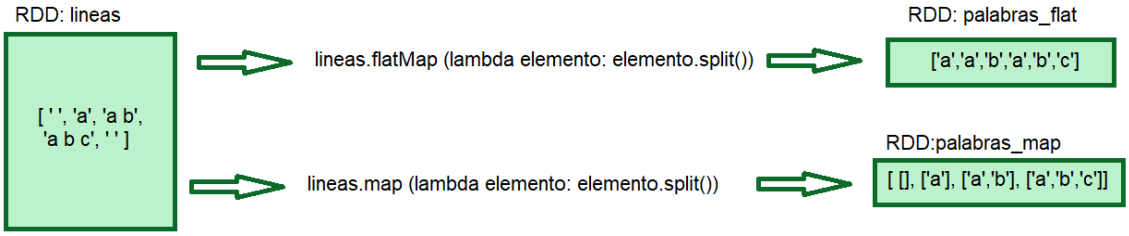


Figura 6. Ejemplo de funcionamiento de la transformación flatMap(func) y map(fun) en un RDD para comparar el funcionamiento de ambas.

- **Transformación filter():** filtra un RDD manteniendo solo los elementos que cumplen una condición.
- **Transformación union():** une dos RDD en uno solo. A continuación, se muestra un ejemplo en el que se hace uso de los métodos filter y union para transformar un RDD inicial.

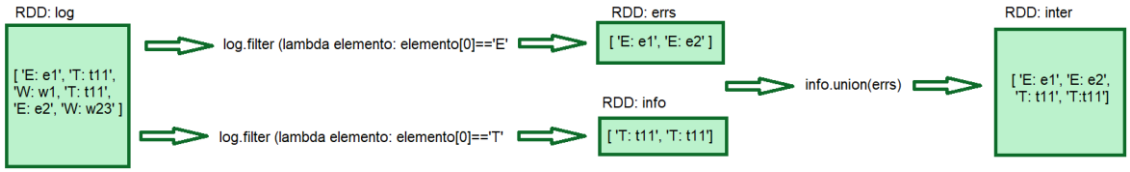


Figura 7. Ejemplo de funcionamiento de los métodos filter y union para transformar un RDD.

- **Transformación reduceByKey()**: Agrega todos los elementos del RDD hasta obtener un único valor por clave. A continuación, se muestra un ejemplo en el que se utiliza esta función para contar palabras.

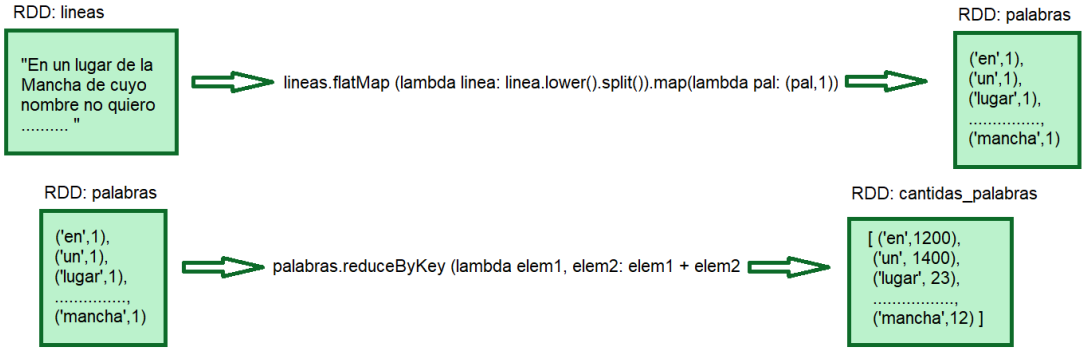


Figura 8. Ejemplo uso del método reduceByKey() para contar palabras.

- **Acción count()**: devuelve el número de elementos de un RDD.

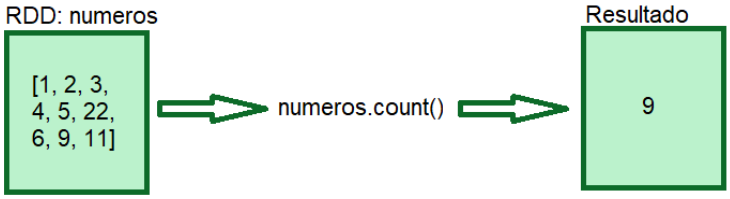


Figura 9. Ejemplo del método count() sobre un RDD.

- **Acción take(n)**: devuelve una lista con los primeros n elementos del RDD.

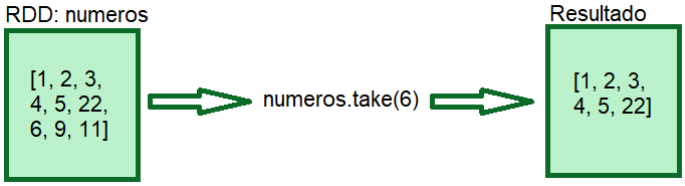


Figura 10. Ejemplo de aplicar la acción take(6) sobre un RDD.

- **Accion collect()**: devuelve en una lista todos los elementos del RDD.

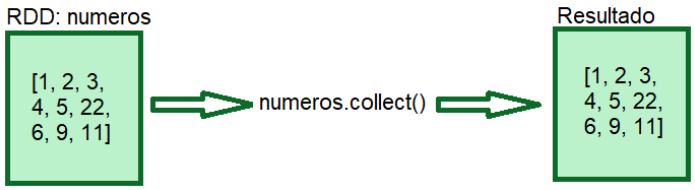


Figura 11. Ejemplo de aplicar la acción collect() sobre un RDD.

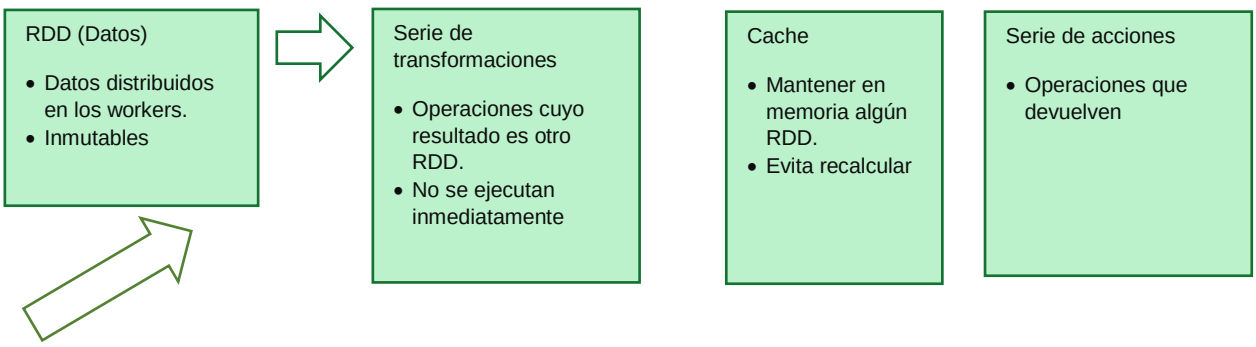
Para finalizar hay que destacar una de las características más importantes de Apache Spark y es la persistencia o cacheo de un dataset en memoria, por lo que cuando se persiste un RDD cada nodo almacena en memoria todas las particiones que posee para poder reutilizarlas al ejecutar otras acciones en dicho dataset, de esta forma las futuras acciones que se ejecuten serán más rápidas (hasta 10 veces más). Para poder convertir un RDD en persistente, hay que usar los métodos persist() o cache() y gracias a que Spark es tolerante a fallos si se pierde alguna partición de un RDD, esta se recalcula automáticamente utilizando las transformaciones que lo crearon originalmente.

2.6 Metodología.

La función split() devuelve una lista con las palabras de una cadena

Las transformaciones de Spark utilizan un mecanismo de “evaluacion perezosa”, es decir, las
transformaciones en un RDD no se ejecutan hasta que se realiza alguna accion sobre el mismo.
Las transformaciones que se puede realizar sobre un RDD son:

Un



Trabajaremos sobre colecciones de datos denominadas RDD:

- Son inmutables. Es decir una vez creados no se pueden modificar.
- Se pueden transformar para crear nuevos RDDso realizar acciones sobre ellos pero no modificar.
- Se guarda la secuencia de transformaciones para poder recuperar RDDsde forma eficiente si alguna máquina se cae

Logstash (5) es una herramienta Open-source desarrollada por la compañía elastic que permite

Elasticsearch (4) es una herramienta Open-source desarrollada por la compañía elastic,

ddddddd

ES se basa en Lucene pero expone su funcionalidad a través de una interfaz REST recibiendo y enviando datos en formato JSON y oculta mediante esta interfaz los detalles internos de lucene. Esta interfaz permite que pueda ser utilizada por cualquier plataforma no solo Java, puede usarse desde Python, .NET, PHP o incluso desde un navegador con JavaScript. Es persistente, es decir, que lo que indexemos en ella sobrevivirá a un reinicio del servidor.

aa

Elasticsearch es una potente herramienta que nos permite indexar una gran volumen de datos y posteriormente hacer consultas sobre ellos soportando entre otras muchas cosas búsquedas aproximadas, facetas y resaltado. Un uso puede ser hacer consultas de texto completo, al estar los datos indexados los resultados se obtienen de forma muy rápida.

`pyspark`

Una expresión regular (o RE) especifica un conjunto de cadenas que coincide con él; las funciones de este módulo le permiten verificar si una cadena en particular coincide con una expresión regular dada (o si una expresión regular dada coincide con una cadena en particular, que se reduce a la misma cosa).

`matplotlib`

generación de gráficos a partir de datos contenidos en listas o arrays en el lenguaje de programación Python y su extensión matemática NumPy.

e Python que produce figuras de calidad de publicación en una variedad de formatos impresos y entornos interactivos en todas las plataformas

Es un lenguaje de alto nivel, de uso general y que puede extenderse e incorporarse. Este lenguaje de programaci´on soporta varias filosof´ıas de programaci´on. Al tratarse de un lenguaje de programaci´on abierto, hay muchos paquetes en la nube que le dan una gran variedad de herramientas que hacen de Python un lenguaje muy extendido. Otra propiedad muy importante

a la hora de programar en este lenguaje es que es lenguaje interpretado, lo que permite crear y ejecutar programas muy r´apido. Python, adem´as, es un lenguaje escrito en C, lo que permite sacar el m´aximo rendimiento a la m´aquina donde se ejecuta. Por otro lado, este lenguaje es capaz de gestionar la memoria utilizada sin que el programador se est´e preocupando constantemente por este problema. A la hora de programar, si programamos con una herramienta como Eclipse, 2 tenemos a nuestra dispocisi´on unas buenas herramientas de depuraci´on, lo que permite entender bien c´omo funciona el c´odigo y ser capaz no solo de localizar los errores, sino de comprender por qu´e se comete dicho error. Para m´as informaci´on, v´ease [5].

Ccc

Además, el análisis de los archivos de registro presenta algunos desafíos adicionales. Dado que muchos sistemas están distribuidos y son heterogéneos, los registros de una serie de componentes deben correlacionarse primero (Oliner et al., 2012). Además, tal vez haya datos faltantes, duplicados o engañosos que hacen que el análisis de registros sea más complejo o incluso imposible. Además, muchas técnicas de modelado analítico y estadístico no siempre proporcionan ideas accionables (Oliner et al., 2012).

3. Bibliografía

1. *Performance evaluation of cloud-based log file analysis with Apache Hadoop and Apache Spark.* **Ilias Mavridis y Helen Karatza.** 2017, The Journal of Systems and Software, págs. 133-151.

2. *Scalability and Performance of Web Applications in a Compute Cloud.* **T. C. Chieu, A. Mohindra y A. A. Karve.** Beijing : s.n., 2011, IEEE 8th International Conference on e-Business Engineering, págs. 317-323.

3. **Python. [En línea] https://www.python.org/.**

4. ElasticSearch. [En línea] https://www.elastic.co/guide/en/elasticsearch/reference/current/index.html.

5. Logstash. [En línea] https://www.elastic.co/guide/en/logstash/current/index.html.

6. Kibana. [En línea] https://www.elastic.co/guide/en/kibana/current/index.html.

7. Apache Spark. [En línea] https://spark.apache.org/.