Projektplan:

Fragestellung:

Welche Wikipedia-Artikel würden, nach den Kriterien von Webseiten-Rankings, die höchste Popularität bzw. Bewertung erlangen?
Welche Artikel sind am besten vernetzt?

Erläuterung: der Projektidee

Webseiten werden bspw. von Suchmaschinen nach diversen Faktoren bewertet bzw. gewichtet. Einer dieser Bewertungsmethoden ist, der von Google entwickelte, PageRank Algorithmus. Im Grunde gewichtet dieser Algorithmus Webseiten anhand der Anzahl der Links, die auf sie verweist.
Diese Gewichtung ist ein Wert zwischen 0 und 1, und gibt die Wahrscheinlichkeit an, mit der ein, im Internet zufällig surfender Nutzer, diese Webseite besucht. Der Grundgedanke ist dabei, dass durch eine Vielzahl von Links die auf eine bestimmte Webseite verweisen, Nutzer eher auf einen dieser Links klicken. Neben der Anzahl der Links gibt es, beim PageRank, noch einen weiteren Faktor: die Gewichtung der Webseite, auf der der Link platziert ist. Das bedeutet, je höher der PageRank der verlinkenden Webseite ist, umso mehr Einfluss hat dieser Link auf den PageRank der verlinkten Webseite.
Betrachtet man die Wikipedia, die bekannteste Online-Enzyklopädie, fällt auf, dass sie aufgrund ihres internen Aufbaus und Verlinkungsstruktur gute Voraussetzungen, zur Analyse der Gewichtung einzelner Artikel bietet. Somit lassen sich Analyse & Raking-Methoden von Suchmaschinen, die primär auf das Vergleichen von Webseiten ausleget sind, auf die Wikipedia übertragen. Der PageRank Algorithmus ließe sich somit an die Wikipedia anpassen und ausführen, dabei werden die Artikel der Wikipedia wie eigenständige Webseiten betrachtet.

Ziel des Projektes:

Das Ziel dieses Projektes ist es, die englischsprachige Wikipedia, mithilfe des PageRank Algorithmus zu analysieren. Dabei wird die zuerst die Wikipedia, genauer gesagt, die enthaltenen Artikel & ihre Verlinkungen ausgelesen und in eine Graph-Struktur überführt. Danach wird der PageRank Algorithmus auf diesen Graphen angewandt. Die Top-n Artikel, gemessen am PageRank und die best-vernetzten Artikel werden visualisiert.

Umsetzungsplan:

Das Projekt besteht aus drei wesentlichen Bestandteilen.

- Pre-Processing & Einlesen der Daten
- Überführen der Daten in eine Graph-Struktur & Analyse mittels PageRank
- Visualisierung der Ergebnisse

Die Implementierung erfolgt in Python 2.7 mithilfe einiger Bibliotheken, auf die ich näher in den einzelnen Projektteilen eingehen werde.

1. Preprocessing & Einlesen der Daten

Der erste Schritt besteht im Parsing und in der Verarbeitung der Wikipedia-Daten. Für dieses Projekt wird die englische Wikipedia verwendet, da sie mit ihren etwa 5 Mio. Artikeln (Stand August 2016) die größte Datenbasis besitzt.
Ein Dump der Datenbank ist frei zugänglich. Unter der URL:

https://dumps.wikimedia.org/enwiki/20160720/

lässt sich der Dump vom 20/07/2016 herunterladen. Ein aktuellerer Dump stand zu diesem Zeitpunkt nicht zur Verfügung. Es werden folgende Dateien benötigt:

enwiki-20160720-pagelinks.sql.gz ~4.8 GB beinhaltet Verlinkungsinformationen

enwiki-20160720-page.sql.gz ~1.3 GB enthält Artikel-Informationen

Beide Dateien bestehen aus jeweils einer gezippten SQL-Datei. Diese enthalten einer Reihe von SQL-Kommandos, die eine SQL Tabelle anlegen und den Inhalt der Datenbank mit dem jeweiligen Inhalt, über SQL INSERT Kommandos füllen.
Die Datei enwiki-20160720-pagelinks.sql.gz wird in Python zeilenweise eingelesen und mithilfe von regulären Ausdrücken und Python-Standardfunktionen werden aus den Informationen Tupel generiert, die aus Artikel-ID und Artikel-Überschrift bestehen. Die Tupel-Daten werden aus den INSERT Zeilen der SQL Datei extrahiert, indem SQL-spezifische Symbole und Befehle entfernt werden. Anhand der SQL Befehle zum Erstellen der Tabelle lässt sich der Aufbau der Tabelle ablesen. Die Daten, die an der ersten und dritten Stelle stehen, d.h. *page_id* und *page_title* werden extrahiert.
Nach dem *INSERT INTO `page` VALUES* folgt die Auflistung der einzufügenden Datensätze für die einzelnen Wikipedia-Artikel, wobei jeder Datensatz in Klammern steht und die Datensätze untereinander mit einem Komma getrennt werden. Der Parser wird, alle Symbole, die vor *VALUES stehen,* ignorieren. Danach werden die einzelnen Datensätze unterteilt, sodass am Ende eine Liste, bestehend aus den einzelnen Artikel-Datensätzen, vorliegt.
Da jeder Datensatz aus einer festen Zahl an Parametern besteht (s. Aufbau d. SQL Tabelle) und immer nur die Werte der ersten und dritten Stelle benötigt wird,
lassen sich diese einfach durch einen regulären Ausdruck, der bei einem Komma separiert, extrahieren.

Aufbau der SQL Tabelle:

```
CREATE TABLE `page` (
  `page_id` int(8) unsigned NOT NULL AUTO_INCREMENT,
  `page_namespace` int(11) NOT NULL DEFAULT '0',
  `page_title` varbinary(255) NOT NULL DEFAULT '',
  `page_restrictions` tinyblob NOT NULL,
  `page_counter` bigint(20) unsigned NOT NULL DEFAULT '0',
  `page_is_redirect` tinyint(1) unsigned NOT NULL DEFAULT '0',
```

```
  `page_is_new` tinyint(1) unsigned NOT NULL DEFAULT '0',
  `page_random` double unsigned NOT NULL DEFAULT '0',
  `page_touched` varbinary(14) NOT NULL DEFAULT '',
  `page_links_updated` varbinary(14) DEFAULT NULL,
  `page_latest` int(8) unsigned NOT NULL DEFAULT '0',
  `page_len` int(8) unsigned NOT NULL DEFAULT '0',
```

Auszug aus den INSERT-Befehl

```
INSERT INTO `page` VALUES
(10,0,'AccessibleComputing','',0,1,0,0.33167112649574004,'20131223211334',N
ULL,381202555,57),
(12,0,'Anarchism','',5252,0,0,0.786172332974311,'20140102071601',NULL,58875
8487,173521),(13,0,'AfghanistanHistory','',5,1,0,0.0621502865684687,'201401
01120449',NULL,74466652,57),
(14,0,'AfghanistanGeography','',0,1,0,0.952234464653055,'20140101143540',NU
LL,407008307,59),
(15,0,'AfghanistanPeople','',4,1,0,0.574721494293512,'20140101073926',NULL,
135089040,60),
(18,0,'AfghanistanCommunications','',8,1,0,0.7510681513241201,'201312191636
48',NULL,74466499,64) …..;
```

Nachdem die enwiki-20160720-page.sql.gz verarbeitet wurde, wird die enwiki-20160720-pagelinks.sql.gz eingelesen. Dabei wird, wieder mithilfe von regulären Ausdrücken und Python-Standardfunktionen, die Artikel-IDs der Quelle und der Artikeltitel des Ziels der Verlinkung extrahiert und in eine Datenstruktur abgespeichert. Nach Abgleich der extrahierten Artikel-Titel mit der, im Schritt zuvor generierten Tupel-Liste, wird die zugehörige Artikel-ID zu den Artikel-Titeln ausgelesen, eine neue Datenstruktur angelegt und diese Informationen darin gespeichert. Der Aufbau der Datenstruktur ist folgendermaßen:

[ID des Ziel-Artikels, Anzahl der eingehenden Links, Liste der IDs der verlinkenden Artikel]

Diese Datenstruktur ermöglicht eine einfache Überführung in einen Graphen.

Aufbau der SQL-Tabelle, pagelinks.sql

```
CREATE TABLE `pagelinks` (
  `pl_from` int(8) unsigned NOT NULL DEFAULT '0',
  `pl_namespace` int(11) NOT NULL DEFAULT '0',
  `pl_title` varbinary(255) NOT NULL DEFAULT '',
  UNIQUE KEY `pl_from` (`pl_from`,`pl_namespace`,`pl_title`),
  KEY `pl_namespace` (`pl_namespace`,`pl_title`,`pl_from`)
) ENGINE=InnoDB DEFAULT CHARSET=binary;
```

Aufbau der INSERT-Befehle

```
/*!40000 ALTER TABLE `pagelinks` DISABLE KEYS */;
INSERT INTO `pagelinks` VALUES (10,0,'Computer_accessibility'),
(12,0,'ism'),
(12,0,'1848_Revolution'),
(12,0,'1917_October_Revolution'),
(12,0,'1919_United_States_anarchist_bombings'),
(12,0,'19th_century_philosophy'),
(12,0,'6_February_1934_crisis'),
```

Überführen der Daten in eine Graphen 6 Analyse mittels PageRank

Nachdem die Daten aufbereitet wurden, muss diese in einen Graphen überführt werden. Dabei ist noch unklar, welche Python Bibliothek dafür verwendet wird. Zur Auswahl stehen NetworkX und Graph-Tool. Beide Bibliotheken bieten den benötigten Funktionsumfang und sind gut dokumentiert. Es muss ein Graph aufgebaut werden, dessen Knoten die IDs der Artikel und den PageRank Wert des Artikels enthalten und Kanten eine unidirektionale Verbindung zwischen Knoten bzw. Artikel enthalten, von der einer auf den anderen verlinkt. Ausgelesen wird der Graph letztendlich aus der Datenstruktur, die alle notwendigen Informationen enthält. Nach Abschluss dieses Schrittes entsteht ein Graph mit ~5 Mio. Knoten und mehreren Mio. Kanten.
Zuletzt muss der PageRank Algorithmus implementiert werden. Dabei wird die folgende Formel verwendet:

Formel:
PR(A) = (1-d) + d (PR(T1) / C(T1) + ... + PR(Tn)/C(Tn))
PR(A) = PageRank der Webseite A, welcher ermittelt wird
d= Dämpfungsfaktor
T1 - Tn = Webseiten die auf Webseite A verlinken
PR(T1) / C(T1) = PageRank der Seite T1 geteilt durch die Anzahl der Links, die von Tn ausgehen.

Der Algorithmus wird rekursiv, über 100 Iterationen ausgeführt. Für den Fall, das sich in den letzten Durchläufen, der PageRank noch stark verändert, ist auch eine höhere Anzahl an Iterationen denkbar. Dabei beginnt der Algorithmus damit, dass jeder Artikel einen PageRank von 1 hat. Danach wird, anhand der obigen Formel der PageRank ermittelt.
Am Ende werden alle Knoten durchlaufen, die Artikel-IDs und PageRank Werte extrahiert und in eine Liste, als Tupel <ArtikelID, PageRank>, absteigend sortiert, abgespeichert.

Visualisierung der Ergebnisse

Als letzter Schritt folgt die Visualisierung der Ergebnisse. Die Top-n Artikel werden, mithilfe -- der Python Bibliothek *bokeh* als eine Treemap dargestellt. Angezeigt werden Artikel-Name und PageRank Wert. Die best-vernetzten Artikel werden durch eine Bubble-Chart, mithilfe der Python Bibliothek pandas dargestellt.
So lässt sich, durch die unterschiedlichen Grüße der einzelnen Bubbles, einfach visuell darstellen, auf welchen Artikel am meisten gelinkt wird. Dabei ist die Anzahl der eingehenden und ausgehenden Verlinkungen sowie das Verhältnis zu zwischen eingehenden und ausgehenden Verbindungen zwei Faktoren, nachdem die Vernetzung eines Artikels ermittelt wird.

PageRank Algorithmus:
http://www.cs.princeton.edu/~chazelle/courses/BIB/pagerank.htm