CS5560 Knowledge Discovery Management
Lab Assignment 4

Lab Submission before November 28th 2018 (11:59 PM).

Submit a short lab report including screenshots and data description to Turnitin . Post your GitHub URL with Source Code (No wiki) through Lab 4 form at https://goo.gl/forms/MP2n7LQGmm2atsZo2

1. Ontology Description

Trace the following Steps using the reference

1. **Determine the domain and scope of the ontology.**
   The domain of our ontology is Monarch Disease which is a semi-automatically constructed ontology that merges in multiple disease resources to yield a coherent merged ontology and its causes and treatments. Basically, for our Monarch Disease, there have two condition that can lead the disease, namely genetic causes and environmental causes. In addition, the ontology may contain some is that range of possible causes of Monarch Disease appears to be limited. For example, like gender.

2. **Consider reusing existing ontologies.**
   The reusing existing ontologies are some already existed ontologies that relevant to our ontology. Like "A Census of Disease Ontologies".

3. **Enumerate important terms.**
   For our ontology, like human, risk, cancer, diagnose, radiotherapy, patients, acromegaly, family, molecular, characteristics …

4. **Define the classes & class hierarchy.**
   For our ontology, we set diseases, environment, gene, chemical … as our classes. For the class hierarchy, we set some words that have some relatives with our classes.
   For example:
   -Disease
   -human
   -human disease causes and treatments.

5. **Define the properties of classes.**
   The properties of classes are property of sentences, like subject, object and verb.
   For example:

The oncofoetal antigen 5T4 is a promising T cell target in the context of colorectal cancer.

6. **Define the facets of the slots.**
   Facets are some words that seems like they belong to same mentioned under classes. For example, like our Monarch Disease which contain some diseases. They are mentioned to be as facets of slots and not act as subclasses.

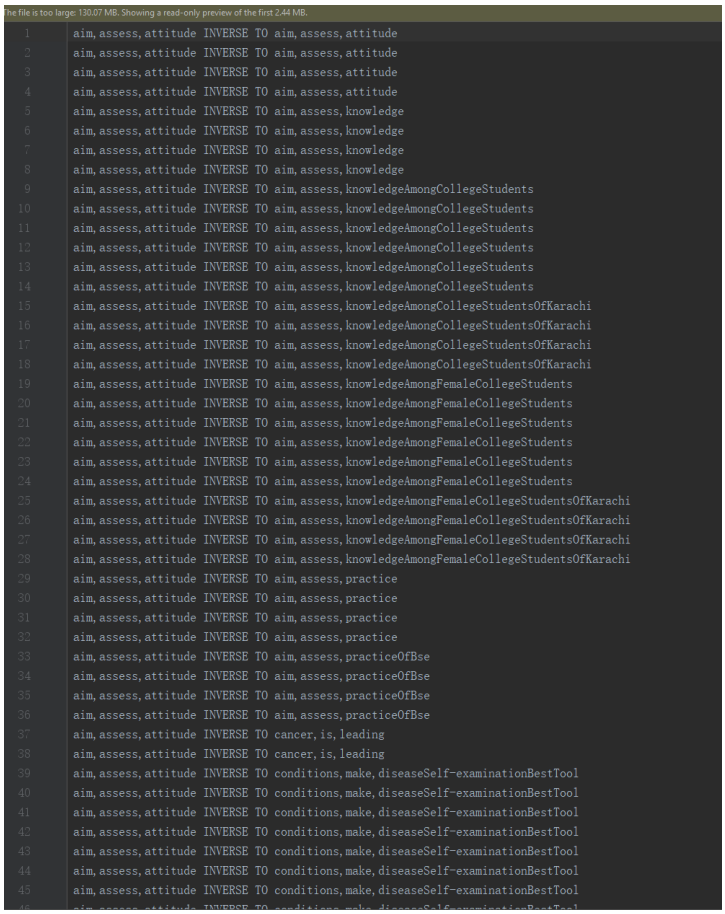
7. Create instances.
Instances of ontology are different words that have same meaning. For example, cancer, tumor.

Ontology Guide:
http://www.ksl.stanford.edu/people/dlm/papers/ontology101/ontology101-noy-mcguinness.html

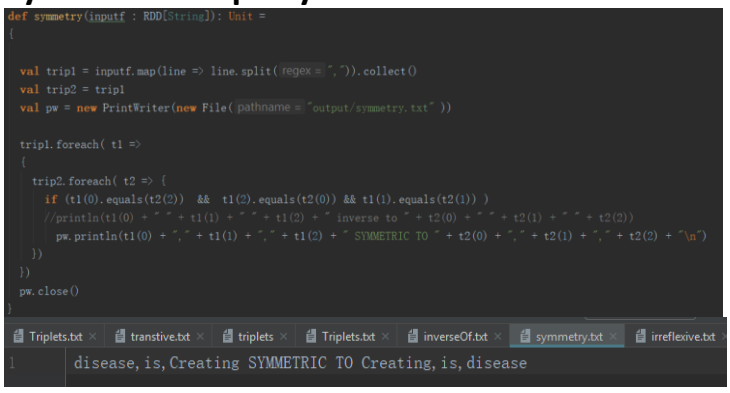
2. Characterize the triplets based on the following Rules (minimum 5) Triplets:

a. Inverse Of

```
def inverseof(inputf : RDD[String]): Unit =
{

  val trip1 = inputf.map(line => line.split( regex = ",")).collect()
  val trip2 = trip1
  val pw = new PrintWriter(new File( pathname = "output/inverseOf.txt" ))

  trip1.foreach( t1 =>
  {
    trip2.foreach( t2 => {
      if (t1(0).equals(t2(2))  &&  t1(2).equals(t2(0)) && !t1(1).equals(t2(1)) )
      println(t1(0) + " " + t1(1) + " " + t1(2) + " inverse to " + t2(0) + " " + t2(1) + " " + t2(2))
        pw.write(t1(0) + "," + t1(1) + "," + t1(2) + " INVERSE TO " + t2(0) + "," + t2(1) + "," + t2(2) + "\n")
    })
  })
  pw.close()
}
```



b. Symmetric Property



c. Transitive Property

```
def transtive(inputf : RDD[String]): Unit =
{

  val trip1 = inputf.map(line => line.split( regex = ",")).collect()
  val trip2 = trip1
  val pw = new PrintWriter(new File( pathname = "output/transtive.txt" ))

  trip1.foreach( t1 =>
  {
    trip2.foreach( t2 => {
      if (t1(2).equals(t2(0))   && t1(1).equals(t2(1)) &&  !t1(2).equals(t2(2))) {
        //println(t1(0) + " " + t1(1) + " " + t1(2) + " inverse to " + t2(0) + " " + t2(1) + " " + t2(2))
        println(t1(0) + "," + t1(1) + "," + t1(2) )
        println(t2(0) + "," + t2(1) + "," + t2(2) )
        println("Transtive :" )
        println(t1(0) + "," + t1(1) + "," + t2(2) )
        //println("\n")
      }
    })
  })
  pw.close()
}
```

Output\Triplets.txt | transtive.txt | triplets | output_ont\Triplets.txt | inverseOf.txt | symmetry.txt | irreflexive.txt

```
disease, is, cancer and cancer, is, causeOfDeath "TRANSITIVE TO" disease, is, causeOfDea
```

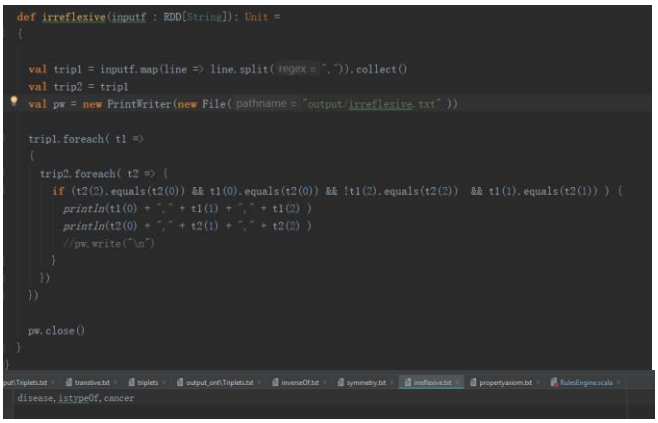
d. Property Chain Axiom

```
def propertyAxim(inputf : RDD[String]): Unit =
{

  val trip1 = inputf.map(line => line.split( regex = ",")).collect()
  val trip2 = trip1
  val trip3 = trip1
  val pw = new PrintWriter(new File( pathname = "output/propertyaxiom.txt" ))

  trip1.foreach( t1 =>
  {
    trip2.foreach( t2 => {
      if (t1(2).equals(t2(0))   && t1(1).equals(t2(1)) &&  !t1(2).equals(t2(2))) {
        trip3.foreach(t3 => {
          if (t1(0).equals(t3(0)) && t2(2).equals(t3(2)) && !t1(1).equals(t3(1)) ) {
            println(t1(0) + "," + t1(1) + "," + t1(2) )
            println(t2(0) + "," + t2(1) + "," + t2(2) )
            println("property axiom :" )
            println(t3(0) + "," + t3(1) + "," + t3(2) )
            //pw.write("\n")
          }
        }
        )
      }
    })
  })

  pw.close()
}
```

Output\Triplets.txt | transtive.txt | triplets | output_ont\Triplets.txt | inverseOf.txt | symmetry.txt | irreflexive.txt | propertyaxiom.txt

```
disease, is, cancer and cancer, is, causeOfDeath => disease, is, causeOfDeath
```

e. Irreflexive Property



3. Record unique features noted from your ontology

1. Entity recognition creation
2. Subclasses creation
3. Synonyms creation
4. Top words creation