**Project Report: Data Mining with Declarative Programming**

CS 240: Databases and Knowledge Bases, Spring 2018

Zhengyuan Liu, 604945064, zhengyuanliu@ucla.edu

**Part 2: K-Nearest Neighbors Classifier (KNN)**

**1. Difficulties in implementing data mining algorithms in declarative systems**

(1) It is difficult to deal with and preprocess the raw dataset. For example, how to read data from file in Datalog, and how to work on arbitrary number of columns in SQL. Usually, other imperative programming language is needed to support the process. In this project, I used python to generate the script to preprocess and read the raw dataset in SQL, and then used the SQL to do the verticalization and KNN classification; In Datalog, I used python to generate the Datalog program with verticalized facts.

(2) There is no clear control flow statement like loop (for or while) in Datalog (DeALS), thus it is difficult to do the operations like sorting. Some straightforward processes in imperative programming may needs some unstraightforward way like negation or recursion to implement it, and may be

(3) It is inconvenient and inefficient to deal with intermediate data in SQL and Datalog, since intermediate table or rules/facts need to be explicitly created, which is very costly.

(4) Declarative programming is often (much) more time-consuming than imperative programming because first it is difficult to optimize the algorithm since most of time you just tell the programming language what to do rather than how to do it, and you cannot control the computation process as clear as imperative programming; second, processing intermediate data is much more costly in declarative programming, for example intermediate tables in SQL involve a lots of I/O, while imperative programming can just store them in the memory.

**2. Ease-of-programming in declarative systems**

(1) Declarative programming tells the programming language “what to do” rather than “how to do it”, which sometimes eases and simplifies work for programmers. For example, it is easy to get all the results for one goal by one statement in declarative systems, without using loop statements in imperative programming systems.

(2) Built-In Aggregates in declarative systems often reduce lots of codes and ease programmers, like sum<>, count<>.