# DSCI 558: Building Knowledge Graphs
# Homework 5: Information Extraction II

Due: Tuesday, Mar 9th, 2021 11:59 PM

## Ground Rules

This homework must be done individually. You can ask others for help with the tools, however, the submitted homework has to be your own work.

## Summary

In this homework, you will extract data from unstructured text using Snorkel (https://snorkel.org/). You will use Snorkel to extract performances and their directors from the biographies collected in Homework 2. We provide a Python notebook (snorkel-0.7.ipynb), which contains code and instructions to accomplish this task.

- Once you finish accomplishing all the tasks, change the notebook's file name to Firstname_Lastname_hw05.ipynb and submit it. Make sure that you follow all the instructions in the #TODO cells.
- In this homework, you will need to submit a PDF version of your notebook as the report (Firstname_Lastname_hw05.pdf). Therefore, your answers to homework questions should be included in the notebook.

**You will need to use your 500 crawled biographies in Homework 2 for this homework. In case where you do not have enough extraction candidates, you are welcome to use the example biographies as well. Please always include your 500 crawled biographies.**

## Task 1 (2pts)

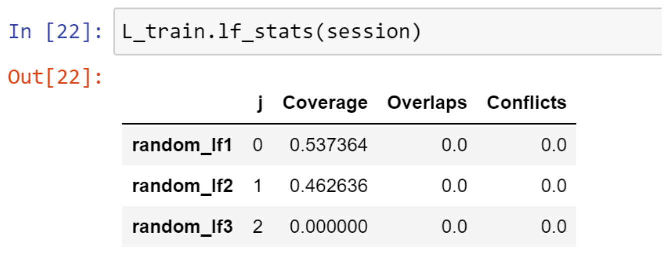
Submit your all your biographies (from Homework 2 and optionally the example data).
(Firstname_Lastname_hw05_all.csv)
Label 50 documents in a development set as instructed in the notebook. (This should be done after Task 2.1 for easier annotation)
Save your results to a csv file as shown in the notebook
(Firstname_Lastname_hw05_gold.dev.csv) using the code in the notebook.

## Task 2 (4pts)

**2.1.** Define (in the notebook) two matchers: one for performances and one for directors. You can re-use the extractors you have written in Homework 2 to create the matchers.

**2.2.** Define (in the notebook) at least three labeling functions (LFs), which Snorkel uses to label the training set.

**2.3.** Report the performance of your LFs **before** generative model training using the cell shown in Figure 1.

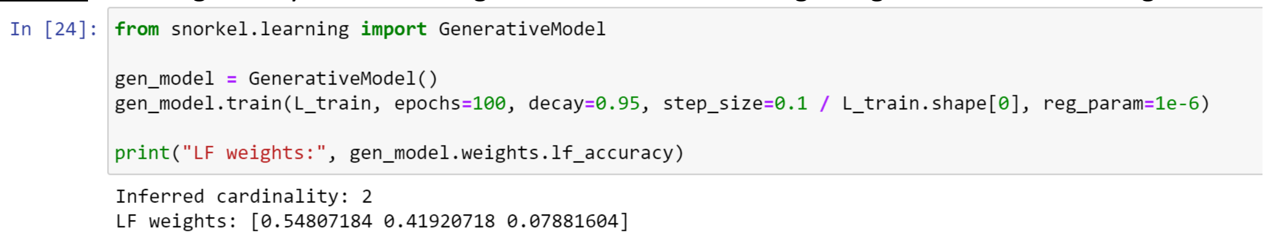


```
In [22]: L_train.lf_stats(session)
Out[22]:
```

| | j | Coverage | Overlaps | Conflicts |
|---|---|---|---|---|
| **random_lf1** | 0 | 0.537364 | 0.0 | 0.0 |
| **random_lf2** | 1 | 0.462636 | 0.0 | 0.0 |
| **random_lf3** | 2 | 0.000000 | 0.0 | 0.0 |

*Figure 1. Labeling Function Performance (Before Training)*

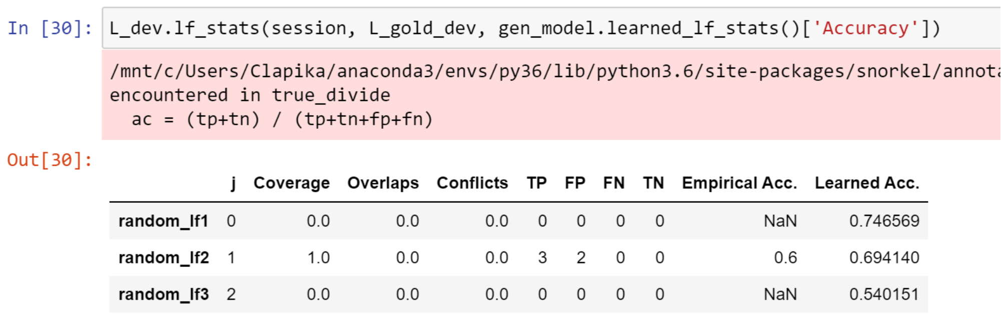
Report the weights of your LFs after generative model training using the cell shown in Figure 2.



```
In [24]: from snorkel.learning import GenerativeModel

         gen_model = GenerativeModel()
         gen_model.train(L_train, epochs=100, decay=0.95, step_size=0.1 / L_train.shape[0], reg_param=1e-6)

         print("LF weights:", gen_model.weights.lf_accuracy)

         Inferred cardinality: 2
         LF weights: [0.54807184 0.41920718 0.07881604]
```

*Figure 2. Labeling Function Weights*

Report the performance of your LFs **after** generative model training using the cell shown in Figure 3 on the labeled dev set.

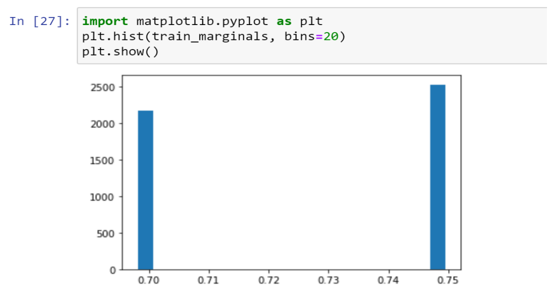


```
In [30]: L_dev.lf_stats(session, L_gold_dev, gen_model.learned_lf_stats()['Accuracy'])

         /mnt/c/Users/Clapika/anaconda3/envs/py36/lib/python3.6/site-packages/snorkel/annota
         encountered in true_divide
           ac = (tp+tn) / (tp+tn+fp+fn)
Out[30]:
```

| | j | Coverage | Overlaps | Conflicts | TP | FP | FN | TN | Empirical Acc. | Learned Acc. |
|---|---|---|---|---|---|---|---|---|---|---|
| **random_lf1** | 0 | 0.0 | 0.0 | 0.0 | 0 | 0 | 0 | 0 | NaN | 0.746569 |
| **random_lf2** | 1 | 1.0 | 0.0 | 0.0 | 3 | 2 | 0 | 0 | 0.6 | 0.694140 |
| **random_lf3** | 2 | 0.0 | 0.0 | 0.0 | 0 | 0 | 0 | 0 | NaN | 0.540151 |

*Figure 3. Labeling Function Performance (After Training)*

**2.4.** Report the distribution of the training marginals (as in Figure 4). The distribution is important because it tells you if your labeling functions are good.



```
In [27]: import matplotlib.pyplot as plt
         plt.hist(train_marginals, bins=20)
         plt.show()
```

*Figure 4. Training Marginals*

**2.5.** Explain (in your notebook) about your marginal distribution (max 3 sentences). Is it good or bad? Explain briefly.

## Task 3 (2pts)

**Distant supervision** generates training data automatically using an external, imperfectly aligned training resource, such as a Knowledge Base.

Define an additional **distant-supervision-based labeling function** which uses Wikidata or DBpedia. With the additional labeling function you added, please repeat steps 2.3, 2.4 & 2.5 and include your answer in the notebook (refer to these answers as 3.3, 3.4 & 3.5 respectively).

- Hint: You can use SPARQLWrapper (https://rdflib.github.io/sparqlwrapper/) to access DBpedia's SPARQL endpoint to query instances such as of types

## Task 4 (2pts)

Train a BiLSTM discriminative model (as demonstrated in the notebook).
Tune the hyper-parameters to get your **best F1 score** and include it your notebook, comment about your tuning process, explain your line of thought. Also include the labeling function analysis as shown in the notebook.
Extract and save the prediction of the dev set to Firstname_Lastname_hw05_pred.dev.csv with three columns: id (id of the biography), performance, director and your prediction.

## Submission Instructions

You must submit (via Blackboard) the following files/folders in a single .zip archive named Firstname_Lastname_hw05.zip:

- Firstname_Lastname_hw05.ipynb: The notebook contains code you wrote to accomplish the tasks
- Firstname_Lastname_hw05.pdf: A printed PDF version of the notebook
- Firstname_Lastname_hw05_all.csv: The crawled data from Homework 2
- Firstname_Lastname_hw05_gold.dev.csv: The labeled data from task 1
- Firstname_Lastname_hw05_pred.dev.csv: contains extracted relations from task 4
- source: This folder includes any additional code you may have wrote to accomplish the tasks (other than the notebook).