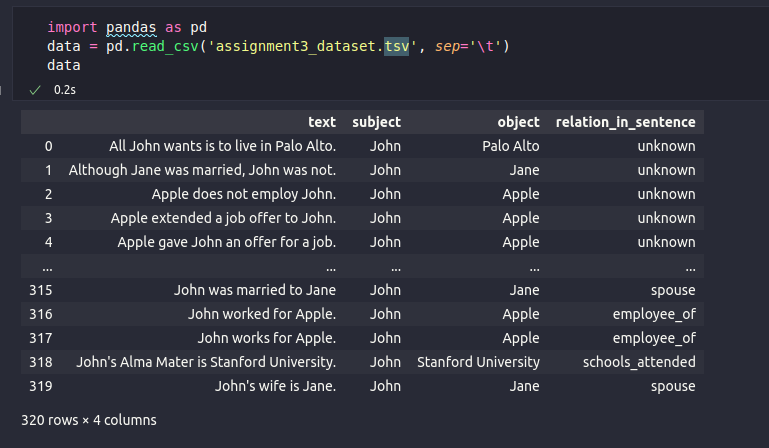
# Assignment 2: Knowledge Graph Population

## Task 1 (50%)

#### *Using an LLM of your choice, implement a prompt-based relation classifier that takes as input a sentence and two entities, and determines if and which of the above four relations are true in the sentence. If the input sentence does not express any of these relations the system should return “Unknown”. Use the provided dataset relation\_extraction\_dataset.tsv (available here) to evaluate the precision and recall of your classifier for each relation. In addition, inspect the errors the system makes and see if you can identify any systematic error patterns.*

#### *Answer:*

We load the data



Across the whole exercise we will use local models:

Models tried are:

* Llama3.2:3b-instruct-fp16

FIrst prompt tried:

| """Analyze the sentence to identify if there is a clear, explicitly stated relation between the subject and object entities. Return ONLY a single JSON key-value pair with "relation" as the key and the matched relation or "Unknown" as the value.  Input:  - Sentence: {row['text']}  - Subject: {row['subject']}  - Object: {row['object']}  Consider these relations ONLY if directly and unambiguously stated in the sentence:  - cities\_of\_residence: relates a person to cities where their physical residence is explicitly mentioned in a factual way  - employee\_of: relates a person to organizations where their employment status is explicitly mentioned through clear terms like "works for", "is employed by", "joined", etc.  - schools\_attended: relates a person to educational institutions where their student status is explicitly mentioned through clear terms like "studies at", "attended", "graduated from", etc.  - spouse: relates a person to persons where their marriage status is explicitly mentioned through clear terms like "married to", "wed", etc.  Return "Unknown" if ANY of these conditions exist:  1. The relation is implied but not explicitly stated  2. Any temporal ambiguity exists about when the relation occurred  3. The sentence contains qualifiers, modalities, or uncertainties  4. The relation direction is not 100% clear  5. Multiple interpretations of the relationship are possible  6. The sentence uses future tense or hypotheticals  7. The relationship is mentioned in passing or as background information  8. Any nuanced context that requires interpretation  9. The sentence describes wishes, plans, or intentions  10. The relation is negated or questioned  Rules  - Do not add any introduction or conclusion to the response  Response:""" |
| --- |

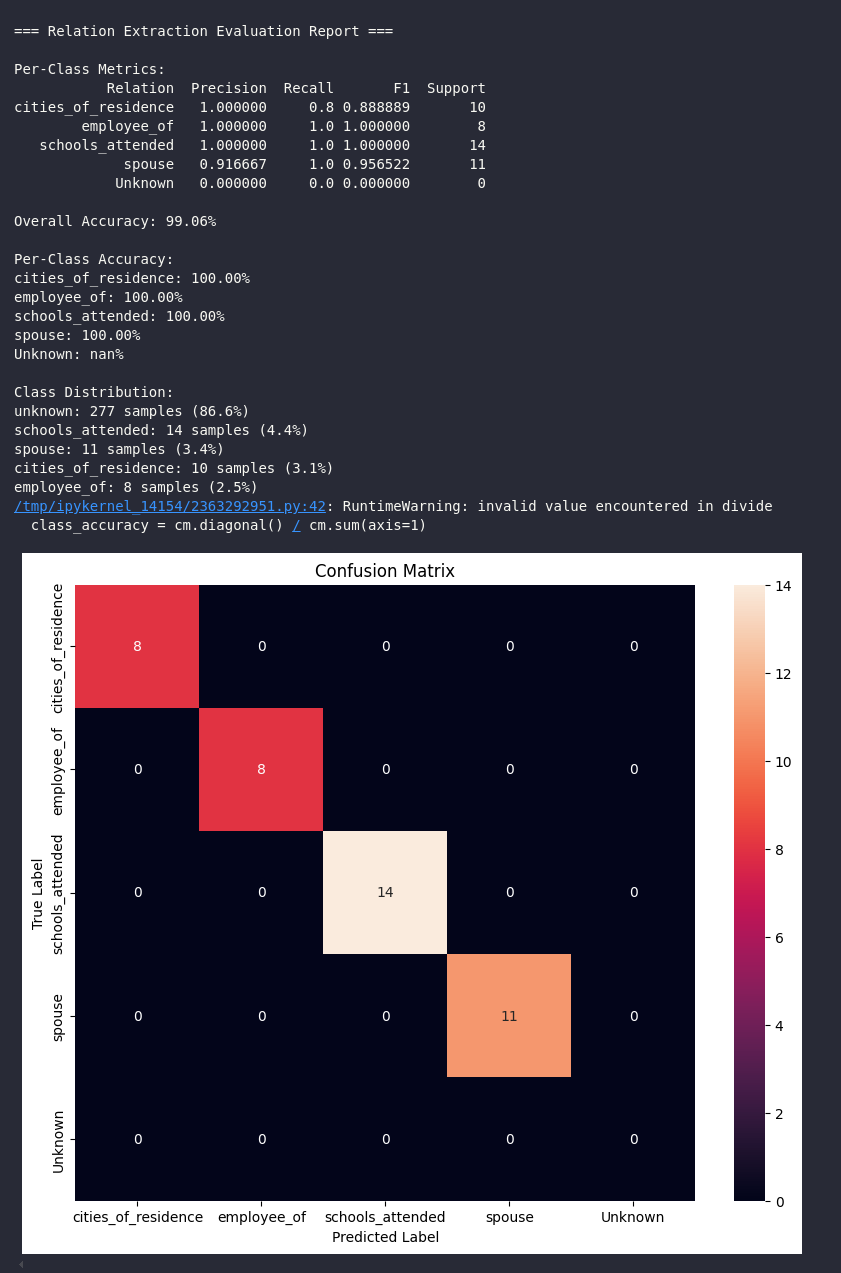
Evaluation:

|  |
| --- |

Adding few shots

| Analyze the sentence to identify if there is a clear, explicitly stated relation between the subject and object entities. Return ONLY a single JSON key-value pair with "relation" as the key and the matched relation or "Unknown" as the value.  Input:  - Sentence: {row['text']}  - Subject: {row['subject']}  - Object: {row['object']}  Consider these relations ONLY if directly and unambiguously stated in the sentence:  - cities\_of\_residence: relates a person to cities where their physical residence is explicitly mentioned in a factual way  - employee\_of: relates a person to organizations where their employment status is explicitly mentioned through clear terms like "works for", "is employed by", "joined", etc.  - schools\_attended: relates a person to educational institutions where their student status is explicitly mentioned through clear terms like "studies at", "attended", "graduated from", etc.  - spouse: relates a person to persons where their marriage status is explicitly mentioned through clear terms like "married to", "wed", etc.  Return "Unknown" if ANY of these conditions exist:  1. The relation is implied but not explicitly stated  2. Any temporal ambiguity exists about when the relation occurred  3. The sentence contains qualifiers, modalities, or uncertainties  4. The relation direction is not 100% clear  5. Multiple interpretations of the relationship are possible  6. The sentence uses future tense or hypotheticals  7. The relationship is mentioned in passing or as background information  8. Any nuanced context that requires interpretation  9. The sentence describes wishes, plans, or intentions  10. The relation is negated or questioned  Examples:  Elizabeth wants to live in New York. -> Unknown  Elizabeth lives in New York. -> cities\_of\_residence  Elizabeth is employed by Google. -> employee\_of  Elizabeth has been studying at Harvard. -> Unknown  Tesla would be lucky to have John as an employee -> Unknown  Jill and Jack are getting married. -> Unknown  Jill and Jack are married. -> spouse  Rules  - Do not add any introduction or conclusion to the response  Response: |
| --- |

Evaluation:



## Task 2 (50%)

#### *Answer*

Creating the dataset

| def generate\_test\_dataset():  """  Generate test dataset for uncertainty and advice/wish scenarios  """  # Test cases for uncertainty  uncertainty\_cases = [  # cities\_of\_residence uncertainty  ("It's possible that Maria lives in Seattle", "Maria", "Seattle", "cities\_of\_residence", "unknown"),  ("There's a chance John moved to Boston last year", "John", "Boston", "cities\_of\_residence", "unknown"),  ("Sarah might be residing in London now", "Sarah", "London", "cities\_of\_residence", "unknown"),  ("Reports suggest that Tom could be living in Paris", "Tom", "Paris", "cities\_of\_residence", "unknown"),  ("I heard James may have moved to Tokyo", "James", "Tokyo", "cities\_of\_residence", "unknown"),    # employee\_of uncertainty  ("Emma might work at Google", "Emma", "Google", "employee\_of", "unknown"),  ("There are rumors that David is employed by Microsoft", "David", "Microsoft", "employee\_of", "unknown"),  ("It's possible that Lisa joined Amazon recently", "Lisa", "Amazon", "employee\_of", "unknown"),  ("Sources suggest Alex could be working at Tesla", "Alex", "Tesla", "employee\_of", "unknown"),  ("I think Robert might be employed at Facebook", "Robert", "Facebook", "employee\_of", "unknown"),    # schools\_attended uncertainty  ("Kevin might be studying at Stanford", "Kevin", "Stanford", "schools\_attended", "unknown"),  ("There's a possibility that Anna enrolled at Harvard", "Anna", "Harvard", "schools\_attended", "unknown"),  ("I heard Peter could be attending MIT", "Peter", "MIT", "schools\_attended", "unknown"),  ("Reports indicate that Rachel might be at Yale", "Rachel", "Yale", "schools\_attended", "unknown"),  ("Sophie possibly studied at Oxford", "Sophie", "Oxford", "schools\_attended", "unknown"),    # spouse uncertainty  ("There are rumors that Mike and Jessica might be married", "Mike", "Jessica", "spouse", "unknown"),  ("It's possible that Eric and Diana got married", "Eric", "Diana", "spouse", "unknown"),  ("People say Chris and Laura could be married", "Chris", "Laura", "spouse", "unknown"),  ("I heard that Paul and Linda might have tied the knot", "Paul", "Linda", "spouse", "unknown"),  ("There's speculation that Mark and Amy are married", "Mark", "Amy", "spouse", "unknown"),  ]    # Test cases for advice/wish  advice\_wish\_cases = [  # cities\_of\_residence advice/wish  ("Jake should move to Chicago", "Jake", "Chicago", "cities\_of\_residence", "unknown"),  ("I wish Emily would live in San Francisco", "Emily", "San Francisco", "cities\_of\_residence", "unknown"),  ("It would be great if Daniel moved to Miami", "Daniel", "Miami", "cities\_of\_residence", "unknown"),  ("Helen ought to consider living in Vancouver", "Helen", "Vancouver", "cities\_of\_residence", "unknown"),  ("I hope Nathan relocates to Austin", "Nathan", "Austin", "cities\_of\_residence", "unknown"),    # employee\_of advice/wish  ("Susan should apply to work at IBM", "Susan", "IBM", "employee\_of", "unknown"),  ("I wish Brian would join Apple", "Brian", "Apple", "employee\_of", "unknown"),  ("It would be nice if Karen worked at Netflix", "Karen", "Netflix", "employee\_of", "unknown"),  ("Tim ought to consider working at Intel", "Tim", "Intel", "employee\_of", "unknown"),  ("I hope Michelle gets a job at Twitter", "Michelle", "Twitter", "employee\_of", "unknown"),    # schools\_attended advice/wish  ("Andrew should attend Princeton", "Andrew", "Princeton", "schools\_attended", "unknown"),  ("I wish Julia would study at Columbia", "Julia", "Columbia", "schools\_attended", "unknown"),  ("It would be great if Rick enrolled at Berkeley", "Rick", "Berkeley", "schools\_attended", "unknown"),  ("Emma ought to consider attending UCLA", "Emma", "UCLA", "schools\_attended", "unknown"),  ("I hope Patrick goes to Cambridge", "Patrick", "Cambridge", "schools\_attended", "unknown"),    # spouse advice/wish  ("Steve and Mary should get married", "Steve", "Mary", "spouse", "unknown"),  ("I wish Jack and Kate would tie the knot", "Jack", "Kate", "spouse", "unknown"),  ("It would be nice if Tom and Sarah got married", "Tom", "Sarah", "spouse", "unknown"),  ("Dave and Lisa ought to consider marriage", "Dave", "Lisa", "spouse", "unknown"),  ("I hope Bill and Nancy get married someday", "Bill", "Nancy", "spouse", "unknown"),  ]    # Combine all test cases  all\_cases = uncertainty\_cases + advice\_wish\_cases    # Create DataFrame  df = pd.DataFrame(all\_cases, *columns*=['text', 'subject', 'object', 'relation\_type', 'relation\_in\_sentence'])    return df |
| --- |

Testing with the previous best prompt

