Inference Model & Belief Network

Presented By:
Shahzaib Irfan 2021-CS-7

Agenda

1. Why do we need this system
2. Introduction
3. Objectives
4. Scope
5. Assignment Modules
6. Data Flow Diagram
7. Conclusion

Introduction

This assignment introduces two probabilistic models:

- Inference Model
- Belief Network

This app computes relevance judgements, rank documents dynamically which simplifies exploration of retrieval model in real-world scenarios.

Objectives

- Goal:
  - Develop a basic search engine to rank documents using **Inference Model** & **Belief Network**.
  - Optimize user experience by providing relevant and accurate search results.

Scope

- Scope:
  - Build a intermediate level, web based search engine for large corpus.

Assignment modules

- Text Preprocessing
- Gather Data
- Initial Setup
- IDF Implementation
- Inference Model
- Belief Network
- Ranking Documents

Pre-Processing

Preprocessing is a crucial first step in building a search engine or any system dealing with large text data. It involves cleaning and organizing text to make it more "search-friendly."

Pre-Processing Technique:

- Tokenization

Pre-Processing (Continued…)

Tokenization:

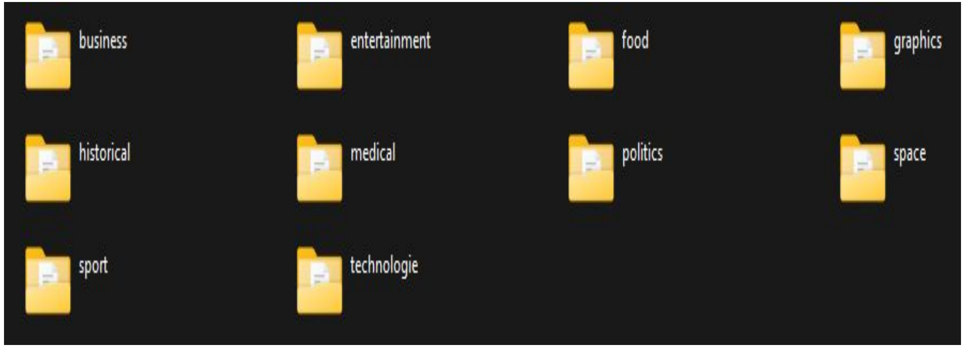
Split text into individual words or "tokens" that can be indexed.

Lets understand with an example:

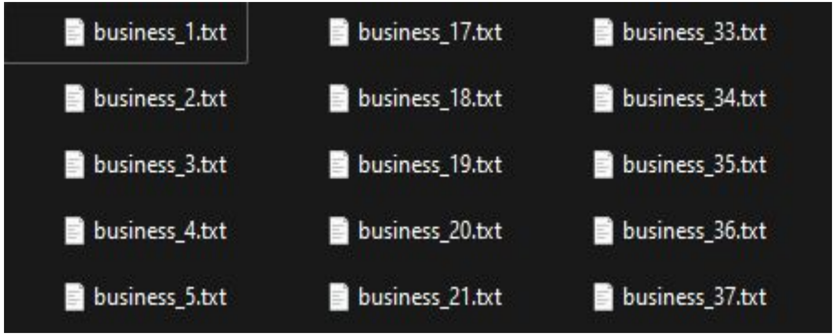
“Ali plays video games in the evening”⇨[“Ali”, “plays”, “video”, “games”, “in”, “the”, “evening”]

Gather Data

- Text files stored in a directory, containing different folders each representing a category.
- Each folder contains 100 text documents.



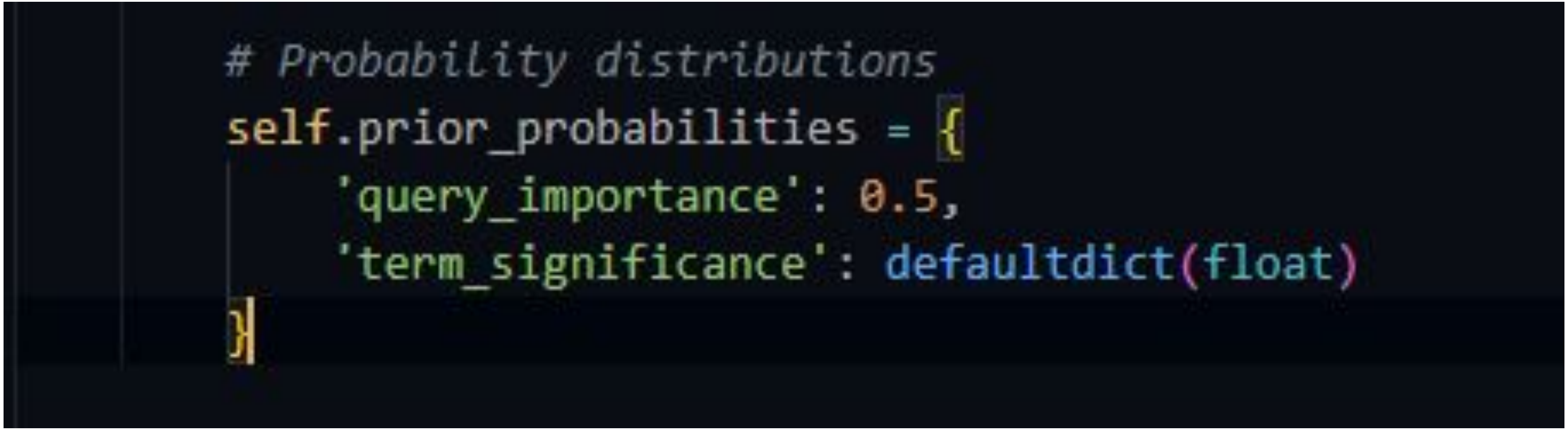
business
entertainment
food
graphics
historical
medical
politics
space
sport
technologie



business_1.txt
business_2.txt
business_3.txt
business_4.txt
business_5.txt
business_17.txt
business_18.txt
business_19.txt
business_20.txt
business_21.txt
business_33.txt
business_34.txt
business_35.txt
business_36.txt
business_37.txt

Initial Setup

- Prior Probablities:



```
# Probability distributions
self.prior_probabilities = {
    'query_importance': 0.5,
    'term_significance': defaultdict(float)
}
```

1. Query_importance: Initially we say, query is neutral.
2. Significance: Tells that how much relevant a term is.

IDF Implementation

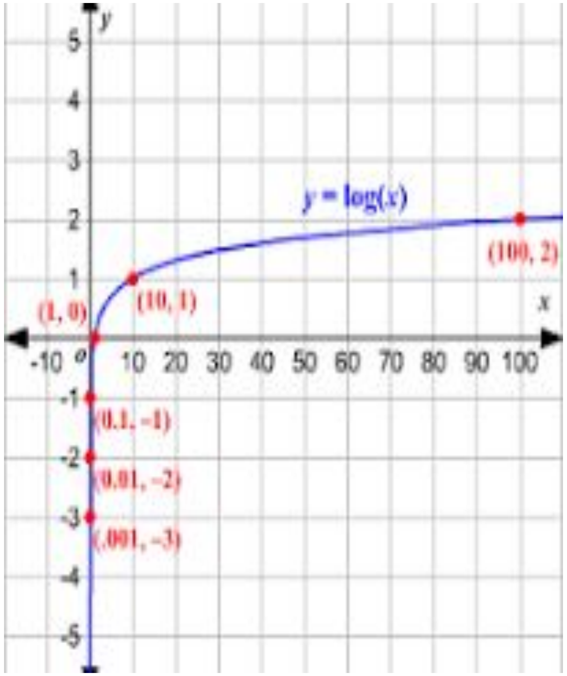
- IDF (Inverse Document Frequency):
  - Measures term importance across documents by giving higher scores to terms with fewer appearance in dataset.
  - Formula:
    - Term = log(Total Documents / (1 + Documents Containing Term)).

TF-IDF Implementation (Continued…)

- IDF (Formula Explanation):
  - **Total Documents**: Total documents in corpus.
  - **Documents Containing Term**: Simply it is count of terms in the complete corpus.
  - **Adding 1**: To avoid division by 0.
  - **Logarithmic Function**: To penalize common terms and reward rare terms.

TF-IDF Implementation (Continued…)

- IDF (Logarithm Function):
  - It penalizes higher values more and lower values less.
  - Example:
    - log(100) = 2
    - log(10) = 1



TF-IDF Implementation (Continued…)

- IDF (Calculated Example):
  - Consider two words “machine” and “data”.
  - Documents with “machine” = 9.
  - Documents with “data” = 499.
  - IDF[machine] = log(1000 /(1 + 9)) => log(100) => 2.
  - IDF[data] = log(1000 / (1 + 499)) => log(2) => 0.3.

So, word “machine” is more relevant to be retrieved because of its higher relevancy score.

Inference Model Steps

1. Calculate Prior Probability.
2. Calculate Overlap
   a. Overlap: **Query ∩ Document / Query**
      i. Overlap is the number of common terms between query and document.
      ii. Result is divided by the no. of query token to normalize result.
3. Calculate Relevance Score

Inference Model Steps (Continued)

1. Relevance Score(Formula):
    a. Relevance Score = **Prior Probability * (1 + Overlap)**
    b. Let’s understand this formula:
        i. Case 1: ∴ Assume Prior = 0.5(Neutral)
            1. No Overlap:
                a. Prior * 1 = Prior
            2. Partial Overlap:
                a. Prior * (1 + 0.5)
            3. Complete Overlap:
                a. Prior * (1 + 1)

Belief Network Steps

1. Calculate Prior Probability.
2. Calculate Overlap
    a. Overlap: Query ∩ Document / Query
        i. Overlap is the number of common terms between query and document.
        ii. Result is divided by the no. of query token to normalize result.
3. Implement Bayesian Probability

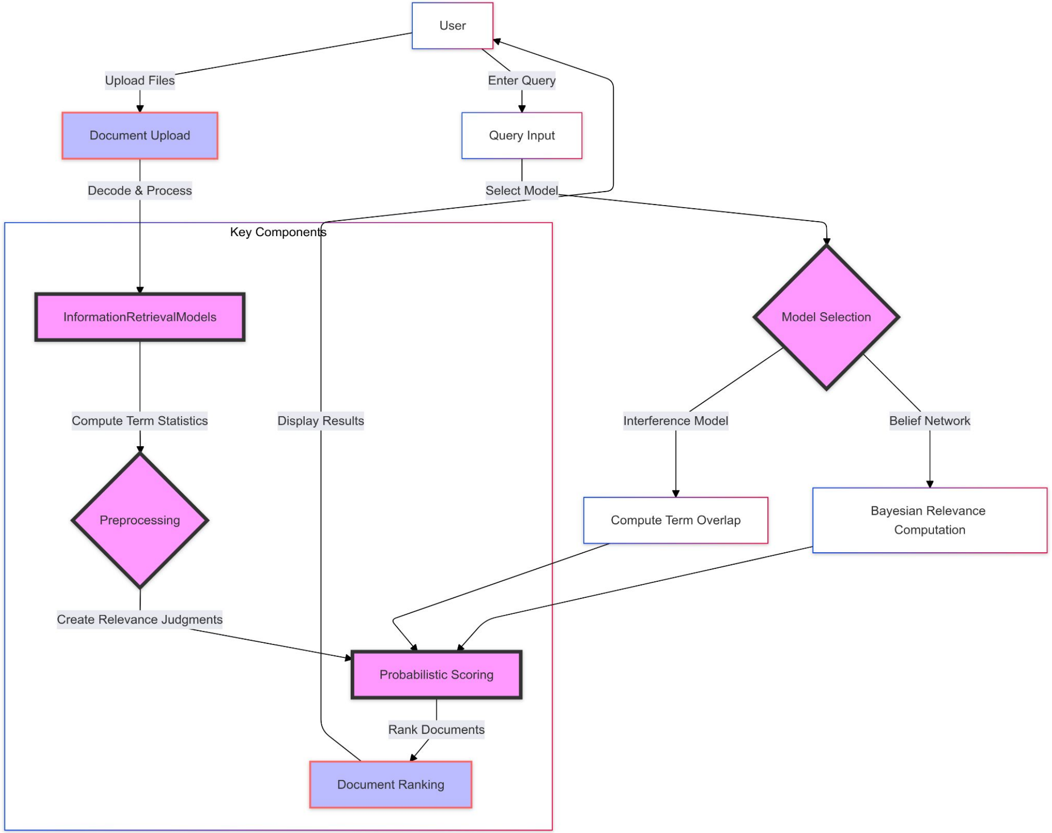
Belief Network Steps (Continued)

1. Bayesian Probability(Formula):
   a. **P(Relevance | Query) = P(Query | Relevance) * P(Relevance) / P(Query)**
   b. Let’s understand this formula:
      i. P(Query | Relevance):
         1. This is result of Step 2 in previous slide.
      ii. P(Relevance):
         1. This is initial relevance judgement score. It is calculated for the given document.
      iii. P(Query):
         1. This is Prior Probability

Ranking

- Documents are ranked on the basis of relevance score in decreasing order.

Data Flow Diagram


User
Upload Files
Enter Query
Document Upload
Query Input
Decode & Process
Select Model
Key Components
InformationRetrievalModels
Model Selection
Compute Term Statistics
Display Results
Interference Model
Belief Network
Preprocessing
Compute Term Overlap
Bayesian Relevance Computation
Create Relevance Judgments
Probabilistic Scoring
Rank Documents
Document Ranking