Regulating Collective Investment Schemes Targeting Agricultural Commodities in India

CaSeiSm.com

DEMO

Our System
CaSeiSm.com

General Overview
CaSeiSm.com
- 5 main topics:
  - Recommender
  - Clustering
  - Searching & Ranking
  - Document Filtering
  - Finding independent features
- Highlights
  - Recommender
  - Searching & Ranking

Recommender
1. Input: author or paperID (converted to author)
2. Based on
  - coauthor distance: the closer, the better
  - citations: list with authors
3. Normalize and recommend top X
CaSeiSm.com
Let's look at the code

Questions?

Future Implementations
- More advanced GUI
- Optimization of the program

Searching & Ranking
1. Input: words!
2. Word frequency, word distance, document location, PageRanking
3. Sum output for each part + ranking
CaSeiSm.com
Let's look at the code

CaSeiSm.com

Overview

1. General overview assignment
2. Our System
3. Highlights
   - Recommender
   - Searching & Ranking
4. Future Implementations
5. Evaluation
6. Questions

Evaluation

Different skills - different tasks
Good book
Very relevant

All information at the start
Update CourSe Guide

Regulating Collective Investment Schemes Targeting Agricultural Commodities in India

CaseiSm.com

DEMO

CaseiSm.com

Overview

CaseiSm.com

Overview

1. General overview assignment
2. Our System
3. Highlights
   - Recommender
   - Searching & Ranking
4. Future Implementations
5. Evaluation
6. Questions

General Overview

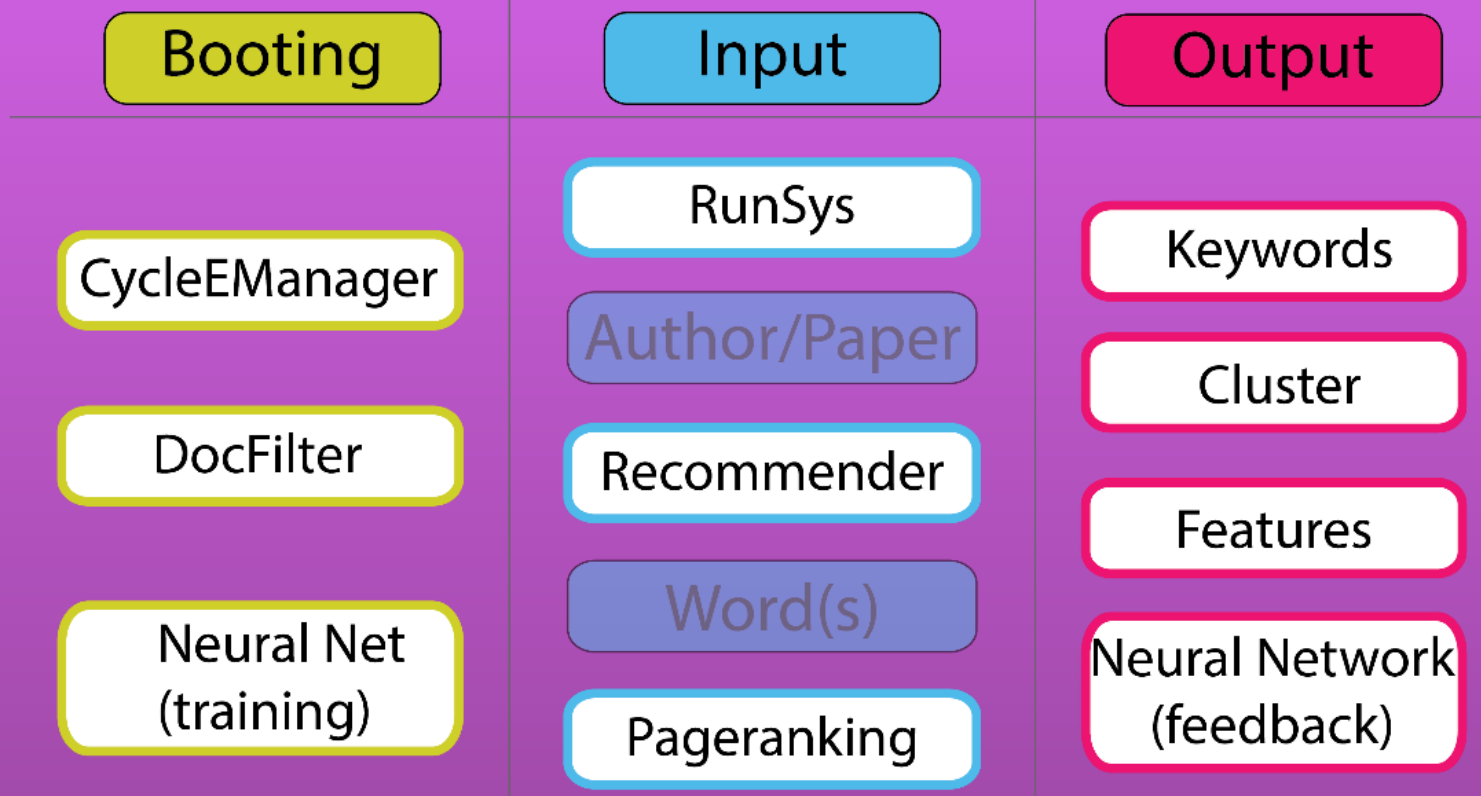
CaseiSm.com

- 5 main topics:
  - Recommender
  - Clustering
  - Searching & Ranking
  - Document Filtering
  - Finding Independent Features

- Highlights:
  - ✓ Recommender
  - ✓ Searching & Ranking

Our System

CaseiSm.com



Recommender

1. Input: author or paperID (converted to author)
2. Based on
   - coauthor distance: the closer, the better
   - citations: list with authors
3. Normalize and recommend top X

Caseism.com

Let's look at the code

Caseism.com

```
def Recommender(InData, SumFile, CitFile, AddSumFile, AuthorsFile, ListN=5):
    InDatA = []; RecomDat=[]; SumSet = SumFile.keys(); InDat=[]; InDat.append(InData); InDat = sum(InDat,[]);
    for i in range(len(InDat)):
        if InDat[i] in SumFile: #Check if input is PaperID, if so, add authors of paper to Author List
            InDatA += SumFile[InDat[i]][1]
        if InDat[i] in AuthorsFile: #If author, add author to Authors InData list
            InDatA.append(InDat[i])
    #Authors from Citations,Cited,COAuthors
    Authors = Counter(); #AuthorsFile[Author Name] = [PID lists in which the Author occurs, List of COAuths]
    for Author in InDatA: #Counts the ammount of collaborations + the amount of crossings of authors with
    citations/citings
        Authors.update(AuthorsFile[Author][1]) #CoAuthors #AuthorsFile[Author][0] #Papers of this Author
        Authors.update(sum(sum([[SumFile[CitID][1] for CitID in CitFile[i] if CitID in SumSet] for i in AuthorsFile[
        Author][0]],[]),[])) #Paper Cites
        Authors.update(sum(sum([[SumFile[CiteID][1] for CiteID in AddSumFile[i]['CiteDB'] if CiteID in SumSet] for i in
        AuthorsFile[Author][0]],[]),[]))
    for Author in InDatA:
        if Author in Authors:
            del Authors[Author]
    for Author in Authors.keys(): Authors[Author] = float(Authors[Author])/sum(Authors.values()); Authors[Author.replace
    (' ','_')] = Authors[Author]; del Authors[Author]; #Normalized Values
    RecomDat = Authors.most_common(ListN) #Recommended Authors and their Values:
    return RecomDat
```

Searching & Ranking

1. Input: word(s)
2. Word frequency, word distance, document location, PageRanking
3. Sum output for each part = ranking

CaseiSm.com

Let's look at the code

Caseism.com

```
def PPageRanking(PID, PageRank, InData, Weights, SumFile, AbsFile, AddSumFile):
    PRWrdCnt = 0.0; PRWrdLoc = 0.0; PRWrdDist = 0.0;
    WrdCnt, WrdNoD, WrdLoc = WordSplit(SumFile[PID][0] + ' ' + AbsFile[PID] if PID in AbsFile else SumFile[PID][0],
    Counter(),Counter());
    if len(set(InData) & set(WrdLoc.keys())) is len(set(InData)): #only if all the words are in the paper, other papers
    -> 0.0 + CitPR
        PRWrdCnt = sum([WrdCnt[InData[i]]/float(len(WrdCnt)) for i in range(len(InData)) if InData[i] in WrdCnt])
        PRWrdLoc = sum([1.0/(sum(WrdLoc[InData[i]])+(1.0*len(WrdLoc[InData[i]]))) for i in range(len(InData)) if InData[
        i] in WrdLoc])
        if len(InData) > 1:
            WrdDist = list(product(*[WrdLoc[InData[i]] for i in range(len(InData)) if InData[i] in WrdLoc]))
            PRWrdDist = 1.0/sum([abs(WrdDist[i][0]-WrdDist[i][1]) for i in range(len(WrdDist))])
    PRank = (((PRWrdCnt*Weights['PRWordFreq'])+(PRWrdLoc*Weights['PRWordLoc'])+(PRWrdDist*Weights['PRWordDist'])+(
    AddSumFile[PID]['CitPR']*Weights['PRCite']))/len(InData))/(4.0)
    PageRank.append([PRank,PID])
    return PageRank
```