COMPUTER ENGINEERING DEPARTMENT

Summer 2019

CMPE 256- Large Scale Analytics

Individual Project Report

Course based Recommender System

Jaykumar Patel (013756210)

Presented to:-

Prof. Shih yu chang

Problem Statement:

Students at various University find it difficult to choose a course and are not aware to choose the perfect subject for them from wide of courses. This project focusses on recommending student best courses for them based on the keyword or the technology they like to work in. This system recommends courses based on content based recommendation and uses various approach of the same.

Motivation:

Recommendation systems is the new trend in the tech industry. Each business needs to grow and these days this could be achieved mistreatment the recommendation system. Large datasets are analyzed to search out what number of alternative users are at risk of just like the similar things they could or may not have employed in the past. In this project, we are considering the dataset of courses of computer engineering department of San Jose State University.

Objective:

The objective of the project is to recommend the course to the user based on the title and the description of the course. I have tried to implement course recommendation system. The dataset already has the information such as Course_id, Course_name,Link to the course and discription of the course. It is the advanced approach than finding top N recommenders for the user. This approach will recommend the course to user based on his interest which is given as input to the recommendation system.

Approach:

Dataset is scraped from San jose state Website and further preprocessing is done on the data to clean the data and further used for various models. This extracted preprocessed data is then run on different algorithms to calculate most recommendable course to the user. Content based similarity is obtained by using differen models like cosine similarity, knn and Tf-idf top N items are recommended for each item.

Dataset:

I have done web scraping of data of the computer engineering department from San jose state website.
Contains 82 courses from San jose state Computer department with various details like:-

- Course ID
- Course name
- Course description
- Link to the course

In [2]:
```
colnames = ['course_id','course_name', 'link', 'course_details']
data = pd.read_csv('courses.csv',skiprows=[0],names=colnames)
data
```

Out[2]:

	course_id	course_name	link	course_details
0	CMPE 030	Programming Concepts and Methodology	http://info.sjsu.edu/web-dbgen/catalog/courses...	Introduction to programming; overview of compu...
1	CMPE 050	Object-Oriented Concepts and Methodology	http://info.sjsu.edu/web-dbgen/catalog/courses...	Application of object-oriented software engine...
2	CMPE 102	Assembly Language Programming	http://info.sjsu.edu/web-dbgen/catalog/courses...	Assembly programming; assembly-C interface; CP...
3	CMPE 110	Electronics for Computing Systems	http://info.sjsu.edu/web-dbgen/catalog/courses...	RC, RL and RLC circuit analysis, diodes and di...
4	CMPE 120	Computer Organization and Architecture	http://info.sjsu.edu/web-dbgen/catalog/courses...	Introduction to computer organization and arch...
5	CMPE 124	Digital Design I	http://info.sjsu.edu/web-dbgen/catalog/courses...	Combinational and sequential logic theory and ...
6	CMPE 125	Digital Design II	http://info.sjsu.edu/web-dbgen/catalog/courses...	Digital system building blocks, data path and ...
7	CMPE 126	Algorithms and Data Structure Design	http://info.sjsu.edu/web-dbgen/catalog/courses...	Object-oriented data organization and represen...
8	CMPE 127	Microprocessor Design I	http://info.sjsu.edu/web-dbgen/catalog/courses...	Microprocessor architecture and assembly langu...
9	CMPE 130	Advanced Algorithm Design	http://info.sjsu.edu/web-dbgen/catalog/courses...	Design and analysis of data structures and alg...
10	CMPE 131	Software Engineering I	http://info.sjsu.edu/web-dbgen/catalog/courses...	Why software engineering? What is software eng...
11	CMPE 132	Information Security	http://info.sjsu.edu/web-dbgen/catalog/courses...	A study of computer and network security from ...
12	CMPE 133	Software Engineering II	http://info.sjsu.edu/web-dbgen/catalog/courses...	Software Architecture, Software Technical Metr...
13	CMPE 135	Object-Oriented Analysis and Design	http://info.sjsu.edu/web-dbgen/catalog/courses...	Feasibility analysis and system requirements d...
14	CMPE 137	Wireless Mobile Software Engineering	http://info.sjsu.edu/web-dbgen/catalog/courses...	Mobility analysis, design principles, techniqu...
15	CMPE 138	Database Systems I	http://info.sjsu.edu/web-dbgen/catalog/courses...	File organization and storage structure, datab...
16	CMPE 139	Fundamentals of Data Mining	http://info.sjsu.edu/web-dbgen/catalog/courses...	Introduction to data management and data minin...

Content Based Filtering

To recommend courses based on the Description I have Implemented 2 models:

1. TF-IDF & Cosine similarity
2. K Nearest Neighbors

Feature Engineering and removing punctuation

feature engineering-remove punctuation

In [5]:
```
tokenizer = RegexpTokenizer ( r'\w+' )
tokenizer.tokenize ( pre_processing )
sentences = nltk.sent_tokenize ( pre_processing )
```

text processing

In [6]:
```
stemmer = PorterStemmer ()
for i in range ( len ( sentences ) ):
    wordsStemmer = nltk.word_tokenize ( sentences[i] )
    wordsStemmer = [stemmer.stem ( word ) for word in wordsStemmer]
    sentences[i] = ' '.join ( wordsStemmer )
```

Text processing using stemmer and Lemmatizer

text processing two words are same then it will normalization

In [7]:
```
lemmatizer = WordNetLemmatizer ()
for i in range ( len ( sentences ) ):
    wordslemmatizer = nltk.word_tokenize ( sentences[i] )
    wordslemmatizer = [lemmatizer.lemmatize ( word ) for word in wordslemmatizer]
    sentences[i] = ' '.join ( wordslemmatizer )
```

Vector Generation

vector generation

In [8]:
```
sentences = sentences[0].split ( '.' )
del sentences[-1]
stopWords = stopwords.words ( 'english' )

vectorizer = CountVectorizer ( stop_words=stopWords )

featurevectors = vectorizer.fit_transform ( col_course_details ).todense ()
```

TF-IDF & Cosine Similarity

Tf-idf is a transformation you apply to texts to get two real-valued vectors.

- TF: Term frequency. This is simply the frequency of a word in a document.
- IDF: Inverse Document Frequency . This is the universe of document frequency among the whole corpus of documents.

In [11]:
```
#getting the data
data_frame = pd.read_csv('courses.csv' , index_col = False)
data_frame = data_frame.loc[:, ~data_frame.columns.str.match('Unnamed')]

new_data_frame= data_frame[['course id','description','name']]


#Making into vectors
tfidfvectorizer = TfidfVectorizer()
tfidfmatrix = tfidfvectorizer.fit_transform(new_data_frame['description'])

data_frame = pd.DataFrame(tfidfmatrix.toarray())

# Caluculating similarity
cosine_sim = cosine_similarity(data_frame)
df_cosineSim = pd.DataFrame(cosine_sim)

#Recommendations
def recommendations(title, cosine_sim = cosine_sim):
    recommended_course_name={}
    idx = new_data_frame[new_data_frame['description'].str.contains(title, case=False)].index[0]

    score_series = pd.Series(cosine_sim[idx]).sort_values(ascending = False)

    top_5_indexes = list(score_series.iloc[1:6].index)

    for i in top_5_indexes:
        recommended_course_name[list(new_data_frame['course id'])[i]]=(list(new_data_frame['name'])[i])


    return (recommended_course_name)
```

Cosine similarity of any pair of vectors by taking their dot product and dividing that by the product of their norms. That yields the cosine of the angle between the vectors.If d2 and q are tf-idf vectors.

1. TF-IDF & Cosine similarity
2. K Nearest Neighbor

cosine similarity

In [9]:
```
def cosine(test2):
    global featurevectors
    cosine_similarities = linear_kernel ( test2, featurevectors ).flatten ()
    related_docs_indices = cosine_similarities.argsort ()[:-5:-1]
    related_docs_indices_list = related_docs_indices.tolist()

    course_name={}
    for i in related_docs_indices:
        course_name[col_course_id[i]]=col_course_name[i]

    result = []
    for i in related_docs_indices_list:
        result.append(col_course_details[i])

    return (course_name)
```

K Nearest Neighbors

For a given description of Courses, KNN algorithm uses Bag of Words model and Euclidean distance to recommend K nearest papers from the dataset.

A bag-of-words model is a way of extracting features from text for use in modeling, such as with machine learning algorithms.

Knn model

In [10]:
```
def build_model_knn(test2):
    neigh = NearestNeighbors ( n_neighbors=5 )
    global featurevectors
    neigh.fit ( featurevectors )
    NearestNeighbors ( algorithm='auto', leaf_size=30 )

    final_knn = neigh.kneighbors ( test2, return_distance=False )
    final_knn_list = final_knn.tolist()
    return final_knn_list
```

Final_output

In [12]:
```
#print(recommendations("cloud computing"))
```

In [16]:
```
input=["software testing"]
vector_input = vectorizer.transform ( input ).toarray ()
build_model_knn ( vector_input )
result_cosine =cosine ( vector_input )
result_tfidf=recommendations(input[0])
#print(result_cosine)
final_output={}
for i in result_cosine:
    if i in result_tfidf:
        final_output[i]=result_tfidf[i]
print(final_output)
```

```
{'CMPE 287': 'Software Quality Assurance and Testing', 'CMPE 187': 'Software Quality Engineering', 'CMPE 133': 'Software Engine
ering II'}
```

Conclusion

I have successfully implemented the course paper recommendation based on Content based approaches. Moreover provided the detail analysis and comparison report of total 2 models (KNN, TFIDF & Cosine similarity).