

Applied Natural Language Processing
Prof. Ramaseshan Ramachandran
Visiting Professor at Chennai Mathematical Institute
Indian Institute Technology, Madras

Lecture - 18
Query Processing

(Refer Slide Time: 00:15)

The slide titled "SUMMARY OF SVD" contains the following bullet points:

- ▶ Find a new set of dimensions or attributes that capture the variability of the data
- ▶ Identify the strongest pattern in the data
- ▶ Most variability is captured by a small fraction of the total set of dimensions
- ▶ Patterns among the terms are captured by the left-singular matrix
- ▶ Patterns among the documents are captured by the right-singular matrix
- ▶ The eigen vectors associated with the largest eigen value indicates the direction of largest variance¹

Handwritten red annotations on the slide include a vertical list of terms on the right side and a box around the last bullet point. The presenter, Prof. Ramaseshan Ramachandran, is visible in the bottom left corner. The slide footer includes the citation: ¹Pang-Ning Tan et al, "Introduction to Data Mining" 2007, the slide number 12 / 13, and the NPTEL logo.

Alright to summarize the SVD (Refer Time: 00:19) model as we mentioned earlier it's about reducing the dimension right. So, what we going to doing here is to find a new set of dimensions or attributes that capture the variability of the data. So, it does not matter how big is your term document corpus all right. You can then always reduce it in terms of the dimensions. For example, if you have document sizes about 2000 plus and then you have around 20,000 words as vocabulary unique words, you can represent that in terms of a matrix that is where you construct your TF, IDF as we discussed earlier.

So, once we have done this and then in the transform domain what we are having is there is a left singular matrix, right singular matrix, and singular value matrix and we spoke about that earlier right. And then when you reduce the singular matrix size that is it some (Refer Time: 01:22) from 2000 to 200, we still would be able to reconstruct the original matrix with some small error right. So, that is what we had shown earlier especially with respect to the image.

So, even though we had a high-resolution image after the transformation, we truncated the SVD to a smaller number and then we showed that by just slightly increasing that SVD (Refer Time: 01:53) from 10 to 20 instead of 700 and odd numbers. We were able to reconstruct that image very clearly with about 50 eigenvalue right.

So, that is the power of this instead of having a very large dimension in the domain that we are looking at the image. We would be able to translate that into the SVD dimension or the SVD domain where the value could be reduced and then we would be able to reconstruct the original image with a very small number of Eigenvalues. So, we demonstrated with respect to the image.

So, if you look at the image you know it is possible for you to reconstruct the image with a few patterns you know especially if you look at a grey level image or a black and white black and white image. There could be a certain kernel that we can always find or the pattern that you can always find and you can just mix and match those patterns to form an image.

So, will that work in the case of a natural language text, how do you deal with that? You know there are patterns within the text that we have we should be able to reduce the dimension, but how do you deal with those small ones you know where the frequency of the word is only one or two say out of 20,000 words that you will find as vocabulary or unique words or tokens. Only about 100 or 200 of them would be having a very high-frequency rest of them would be having very small numbers especially the vocabulary with the count of one of the words with the count of one would be having a large set.

Will, that contribute anything to the; will that contribute anything in our case especially in the query modeling and terms of retrieving the document based on keywords or trying to classify the document based on the set of keywords? Do not we consider them as noise? And now will they really have some impact in in the dimension that we are talking about. Let us see how we can do with by taking some small example ok. As I am going to be talking about two examples to showcase that so, we want to identify the strong pattern in the data; so that is the idea right. So, if you look at the Eigenvalues, we know that it is in the descending order with respect to the magnitude correct.

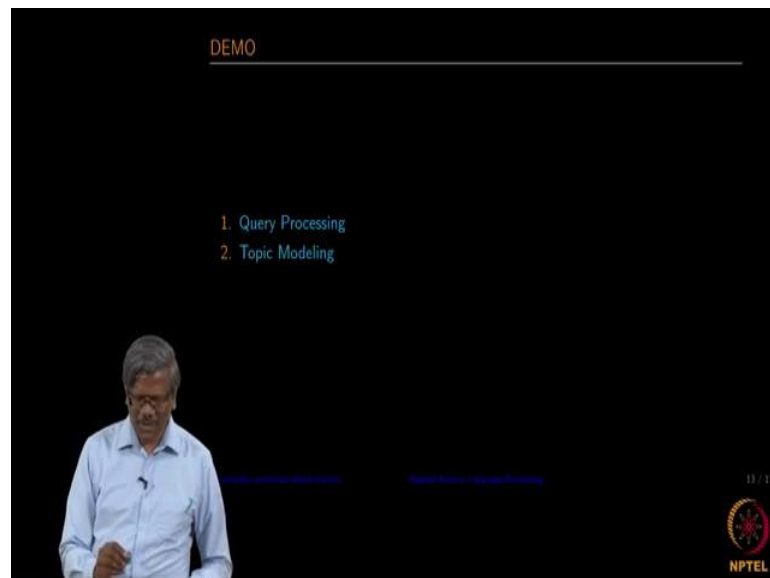
So, this contributes the maximum and then we know that this contributes to the lowest that is why we are able to cut and then reduce the dimension and we still have we are still

reconstructing the original document with a minimal error right. So, we also know that the most variability is captured by the small set of values patterns among the terms are captured by the left singular values, we have the left singular value here right U and then we have the right singular matrix right.

So, the patterns related to the documents are captured by this. So, if you have terms and document organization where each one or each cell in this matrix represents the word in that particular document correct and then the left singular one captures the term related features. The Eigenvectors associated with the largest Eigenvalues indicate the direction of the largest variance; so this captures the largest variance. If you take these matrixes right and then find the Eigenvectors associated with this one, you will find or the vectors would indicate the direction of the largest variance.

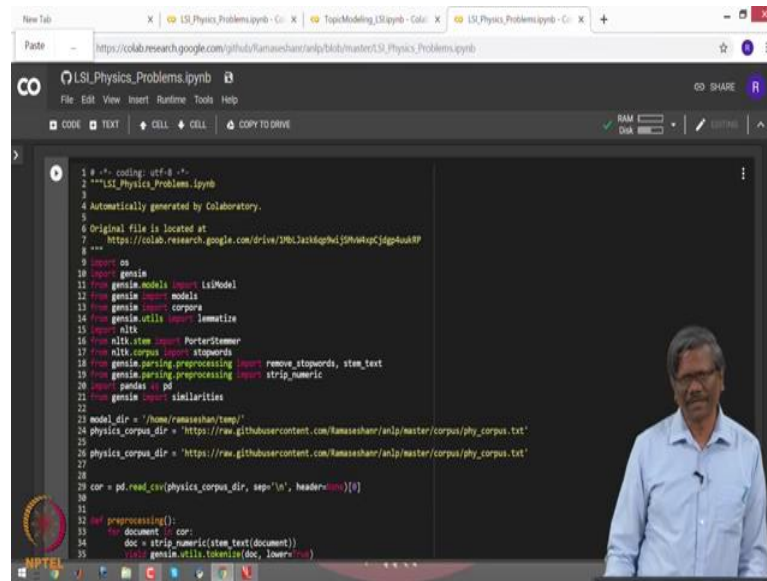
So, again if you take the second one which is orthogonal to the previous one that we computed that will give you the rest of the direction not the rest of some direction and the third one will give you the next level forth one the next level and so on so forth. And then the last one will be the last one and probably would give you a very little or very small number of directions. So, we have seen this through the image demo.

(Refer Slide Time: 07:48)



So, now let us look at it through a text I am going to be showing two demos. So, I am opening it in the collab.

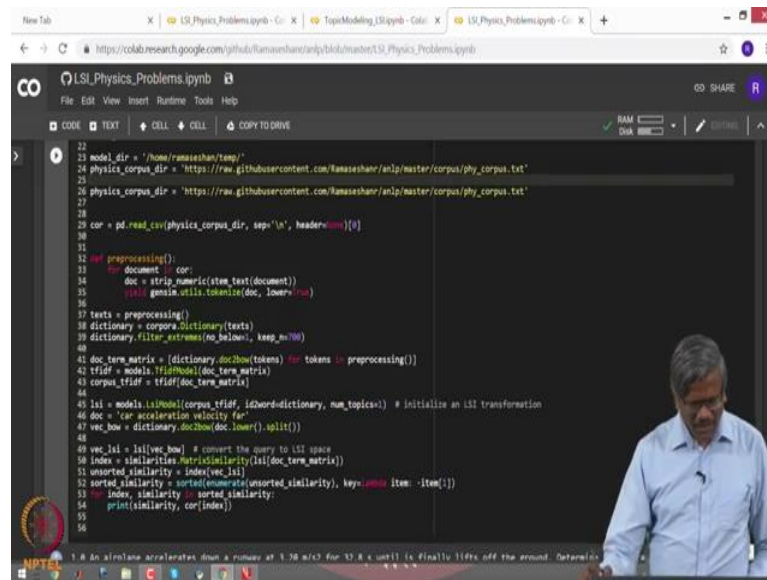
(Refer Slide Time: 07:55)



```
1 # -*- coding: utf-8 -*-
2 """LSI_Physics_Problems.ipynb
3
4 Automatically generated by Colaboratory.
5
6 Original file is located at
7 https://colab.research.google.com/drive/2N6JzckqPhi59hWtqCj4p6u8k8P
8 """
9 import os
10 from gensim.models import LsiModel
11 from gensim.models import models
12 from gensim.corpora import corpora
13 from gensim.util import lemmatize
14 from nltk.tokenize import PorterStemmer
15 from nltk.tokenize import PorterStemmer
16 from nltk.tokenize import PorterStemmer
17 from nltk.tokenize import PorterStemmer
18 from gensim.parsing.preprocessing import remove_stopwords, stem_text
19 from gensim.parsing.preprocessing import strip_numeric
20 from gensim import pd
21 from gensim.similarity import similarities
22
23 model_dir = "/home/ramaseshan/keep/"
24 physics_corpus_dir = "https://raw.githubusercontent.com/Ramaseshan/nlp/master/corpus/phy_corpus.txt"
25
26 physics_corpus_dir = "https://raw.githubusercontent.com/Ramaseshan/nlp/master/corpus/phy_corpus.txt"
27
28
29 cor = pd.read_csv(physics_corpus_dir, sep='\n', header=None)[0]
30
31
32 def preprocessing():
33     document = cor
34     doc = strip_numeric(stem_text(document))
35     gensim.util.tokenize(doc, lowercase=True)
36
37 texts = preprocessing()
38 dictionary = corpora.Dictionary(texts)
39 dictionary.filter_extremes(no_below=1, keep_n=100)
40
41 doc_term_matrix = (dictionary.doc2bow(tokens) for tokens in preprocessing())
42 tfidf = models.TfidfModel(doc_term_matrix)
43 corpus_tfidf = tfidf(doc_term_matrix)
44
45 lsi = models.LsiModel(corpus_tfidf, id2word=dictionary, num_topics=1) # Initialize an LSI transformation
46 doc = "car acceleration velocity fast"
47 vec_bow = dictionary.doc2bow(doc.lower().split())
48
49 vec_lsi = lsi[vec_bow] # convert the query to LSI space
50 index = similarities.MatrixSimilarity(lsi[doc_term_matrix])
51 unsorted_similarity = index[vec_lsi]
52 sorted_similarity = sorted(enumerate(unsorted_similarity), key=lambda item: -item[1])
53 index_similarity = sorted_similarity
54 print(similarity, cor[index])
55
56
```

So, what I have in this demo is I have a set of documents these documents are coming from the kinematics problem statements ok. So, I have about 271 documents related to the problems in kinematics, and that what I am using it as my corpus. So, this you may also use it this available publicly you can use this ok.

(Refer Slide Time: 08:23)



```
22
23 model_dir = "/home/ramaseshan/keep/"
24 physics_corpus_dir = "https://raw.githubusercontent.com/Ramaseshan/nlp/master/corpus/phy_corpus.txt"
25
26 physics_corpus_dir = "https://raw.githubusercontent.com/Ramaseshan/nlp/master/corpus/phy_corpus.txt"
27
28
29 cor = pd.read_csv(physics_corpus_dir, sep='\n', header=None)[0]
30
31
32 def preprocessing():
33     document = cor
34     doc = strip_numeric(stem_text(document))
35     gensim.util.tokenize(doc, lowercase=True)
36
37 texts = preprocessing()
38 dictionary = corpora.Dictionary(texts)
39 dictionary.filter_extremes(no_below=1, keep_n=100)
40
41 doc_term_matrix = (dictionary.doc2bow(tokens) for tokens in preprocessing())
42 tfidf = models.TfidfModel(doc_term_matrix)
43 corpus_tfidf = tfidf(doc_term_matrix)
44
45 lsi = models.LsiModel(corpus_tfidf, id2word=dictionary, num_topics=1) # Initialize an LSI transformation
46 doc = "car acceleration velocity fast"
47 vec_bow = dictionary.doc2bow(doc.lower().split())
48
49 vec_lsi = lsi[vec_bow] # convert the query to LSI space
50 index = similarities.MatrixSimilarity(lsi[doc_term_matrix])
51 unsorted_similarity = index[vec_lsi]
52 sorted_similarity = sorted(enumerate(unsorted_similarity), key=lambda item: -item[1])
53 index_similarity = sorted_similarity
54 print(similarity, cor[index])
55
56
```

So, here this is the location of the corpus I think there is the duplicate here we can remove this and then I am reading that corpus from the location using pandas function called read CSV. So, this reads the file and each document is separated by a new line and

there is no header related to that and then I have a small preprocessing function that takes away all the numeric values and then it tokenizes after that.

I am just creating a text initially which contains all the tokens related to 271 documents and then I create a dictionary with respect to the vocabulary of the words used there ok. And then if you go and then look at the document dictionary right, it is very important for you to do that; that is an important step. You have to find out how many words are there in the vocabulary, so we need to get the count of that. So, in this case, it is about 900 plus and I have restricted the count to 700 see here, I am not using any stop words or any other mechanism in the pre-processing, I am just removing the numeric and then tokenizing that.

So, out of 900 I am just saying let us keep only about 700. So, now, your document size is 271, the number of words is 700 and then convert the dictionary into a doc the bag of words ok, for every document in or every token in the text that we have created here. So, we are creating a bag of words right. So, that contains your document term matrix as well and then using that text that has already created by pre-processing, we also need to find out the term of idf right.

So, that is our aim, so the entire matrix now contains the tf idf values. So, your column contains the documents, the rows or the words and every cell that you have in that matrix is a tf if it contains the tf IDF value ok. So, once you create the tf if now we can use an LSI. So, here I am using Gensim as the library. So, I do not have to write a lot of code in terms of creating the SVD and so on, maybe in additionally you may want to write five or 6 lines if you use (Refer Time: 11:46). So, in this case I relied on the Gensim library to do the job for me. So, there are several other libraries available you can try each one of them as well ok.

So, what I have done is I have just created tf IDF and model and then I am using the LSI which will now create our U sigma and V right. So, here to show that the number of topics here refers to the sigma values or the k values that we are talking about in the Eigen matrix ok. So, I just kept it as one I have just taken only the top one here.

So, you remember even in the image processing is started with the lowest one and then reconstructed the image and then see we added one more and then slowly as we add

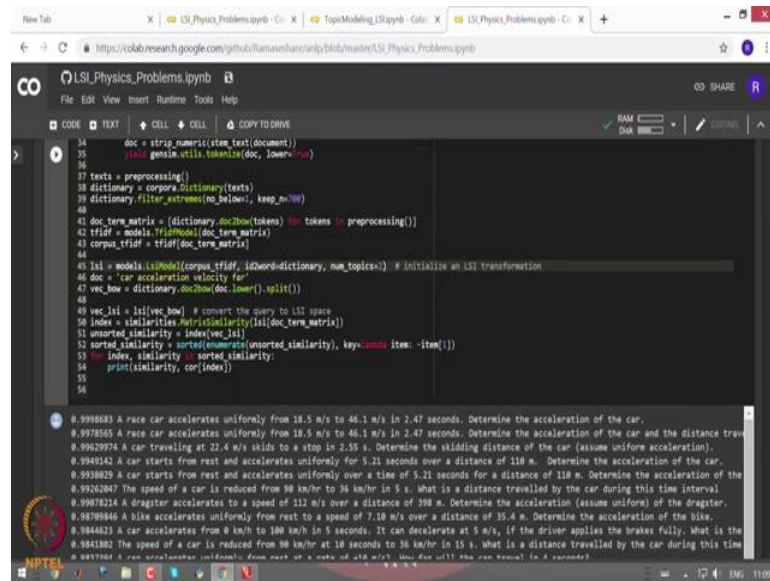
more and more, the image started becoming clear and clearer and clearer you know from a blurry picture to clearer picture within about 80 or 90 Eigenvalue.

So, in this case, again I am doing the same thing I am starting with one and let us see what happens. So, in this case, what I am also doing is inside of classifying the document I am going to use the query function I am going to have three or four query parameters, in this case, I have four and then query the transform domain and finally, get the values in the domain that we are dealing with to find out what documents are (Refer Time: 13:36) based on the query. So, so for you to do that the query has to be converted into the SVD domain right or the LSI domain. So, we first create that LSI model using the sigma as one and then using the dictionary a which contains the word id to the word and then using the tf idf. And then the query document contains car acceleration velocity far.

So, now, I have to create a query in the LSI domain as well. So, again I create a dictionary of that and then create a vector for the query in the LSI domain that is available in this. And then there is a similarity function that is also available as part of the Gensim library and then it tells it goes and then looks at all the documents related to the query and then provides (Refer Time: 14:44) for each one of those ok.

You remember the cosine (Refer Time: 14:48) that we computed earlier is a very similar fashion this constructs all the similar values and then keeps it and you can only list you know 5 of them, 10 of them, 20 of them in any order that you want to list them ok. So, it first gets you the similarities for the query, and then you sort it and then finally, print this.

(Refer Slide Time: 15:25)

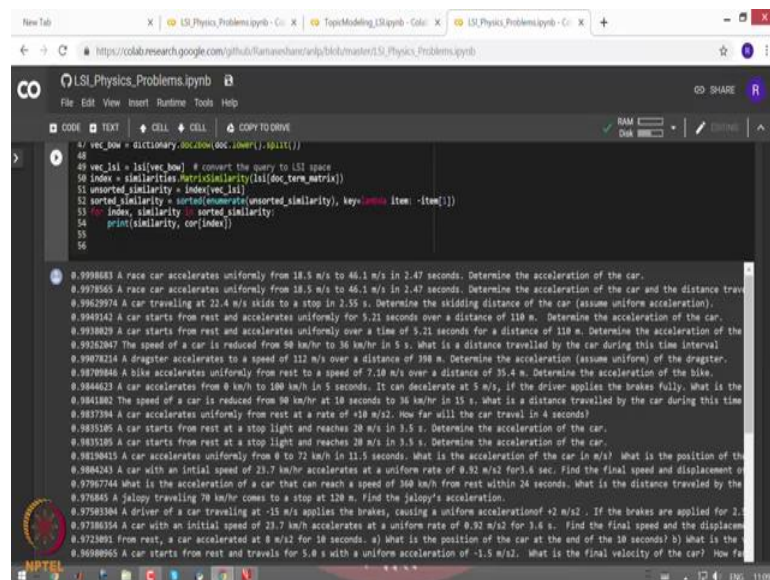


```
34 doc = str.strip_numeric(steer_text(document))
35 #add gensim.utils.tokenize(doc, lower=True)
36
37 tests = preprocessing()
38 dictionary = corpora.Dictionary(texts)
39 dictionary.filter_extremes(no_below=1, keep_n=None)
40
41 doc_term_matrix = (dictionary.doc2bow(tokens) for tokens in preprocessing())
42 tfidf = models.TfidfModel(doc_term_matrix)
43 corpus_tfidf = tfidf(doc_term_matrix)
44
45 lsi = models.LsiModel(corpus_tfidf, id2word=dictionary, num_topics=5) # initialize an LSI transformation
46 doc = "car acceleration velocity far"
47 vec_low = dictionary.doc2bow(doc.lower()).split()
48
49 vec_lsi = lsi[vec_low] # convert the query to LSI space
50 index = similarities.MatrixSimilarity(lsi[doc_term_matrix])
51 unsorted_similarity = index[vec_lsi]
52 sorted_similarity = sorted(enumerate(unsorted_similarity), key=lambda item: -item[1])
53 index, similarity = sorted_similarity
54 print(similarity, cor[index])
55
56
```

0.9998038 A race car accelerates uniformly from 18.5 m/s to 46.1 m/s in 2.47 seconds. Determine the acceleration of the car.
0.9978555 A race car accelerates uniformly from 18.5 m/s to 46.1 m/s in 2.47 seconds. Determine the acceleration of the car and the distance traveled.
0.9962974 A car traveling at 22.4 m/s skids to a stop in 2.35 s. Determine the skidding distance of the car (assume uniform acceleration).
0.9949142 A car starts from rest and accelerates uniformly for 5.21 seconds over a distance of 118 m. Determine the acceleration of the car.
0.9938829 A car starts from rest and accelerates uniformly over a time of 5.21 seconds for a distance of 118 m. Determine the acceleration of the car.
0.9926287 The speed of a car is reduced from 90 km/hr to 36 km/hr in 5 s. What is a distance travelled by the car during this time interval.
0.99078214 A dragster accelerates to a speed of 112 m/s over a distance of 398 m. Determine the acceleration (assume uniform) of the dragster.
0.98789846 A bike accelerates uniformly from rest to a speed of 7.10 m/s over a distance of 35.4 m. Determine the acceleration of the bike.
0.9844623 A car accelerates from 0 km/h to 100 km/h in 5 seconds. It can decelerate at 5 m/s². If the driver applies the brakes fully, what is the
0.9841882 The speed of a car is reduced from 90 km/hr at 10 seconds to 36 km/hr in 15 s. What is a distance travelled by the car during this time
0.9837394 A car accelerates uniformly from rest at a rate of +10 m/s². How far will the car travel in 4 seconds?
0.9835155 A car starts from rest at a stop light and reaches 20 m/s in 3.5 s. Determine the acceleration of the car.
0.9833825 A car starts from rest at a stop light and reaches 20 m/s in 3.5 s. Determine the acceleration of the car.
0.98198415 A car accelerates uniformly from 0 to 72 km/h in 11.5 seconds. What is the acceleration of the car in m/s²? What is the position of the
0.9804243 A car with an initial speed of 23.7 km/hr accelerates at a uniform rate of 0.92 m/s² for 3.6 sec. Find the final speed and displacement of
0.97907846 What is the acceleration of a car that can reach a speed of 300 km/h from rest within 24 seconds. What is the distance traveled by the
0.978465 A jalopy traveling 70 km/hr comes to a stop at 120 m. Find the jalopy's acceleration.
0.97581384 A driver of a car traveling at -15 m/s applies the brakes, causing a uniform acceleration of +2 m/s². If the brakes are applied for 2.1
0.97386354 A car with an initial speed of 23.7 km/h accelerates at a uniform rate of 0.92 m/s² for 3.6 s. Find the final speed and the displacement
0.9723891 From rest, a car accelerated at 8 m/s² for 10 seconds. a) What is the position of the car at the end of the 10 seconds? b) What is the
0.96989955 A car starts from rest and travels for 5.0 s with a uniform acceleration of -1.5 m/s². What is the final velocity of the car? How far

So, in this case I have used my k as 1 you see that all it says that all documents are equal. So, it has not found anything, it is not very similar to what we saw in the image, is not it. So, there we saw at least some portions of the image being reconstructed recognize whereas in this case you know you can see all kinds of jumble even though you see a car is in the second one, but the similar similarity values are all equal that is not the right thing. So, let us now try to increase it by another and let us see what happens.

(Refer Slide Time: 16:01)

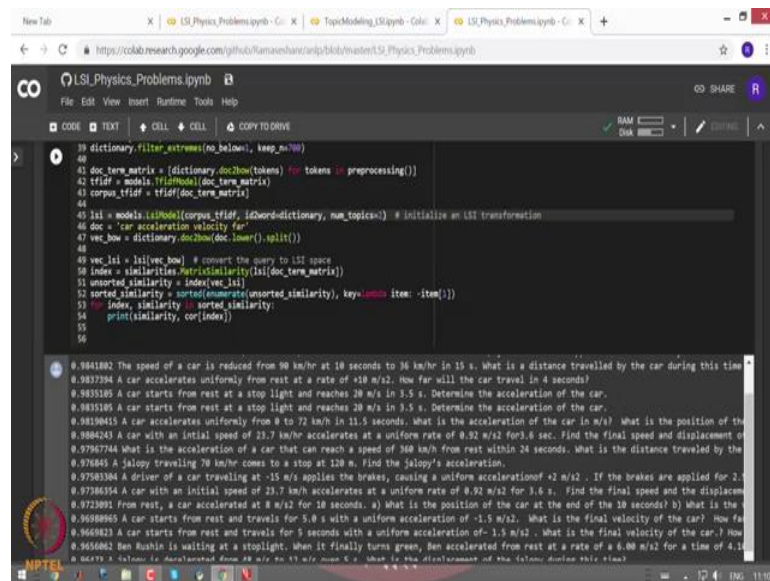


```
47 vec_low = dictionary.doc2bow(doc.lower()).split()
48
49 vec_lsi = lsi[vec_low] # convert the query to LSI space
50 index = similarities.MatrixSimilarity(lsi[doc_term_matrix])
51 unsorted_similarity = index[vec_lsi]
52 sorted_similarity = sorted(enumerate(unsorted_similarity), key=lambda item: -item[1])
53 index, similarity = sorted_similarity
54 print(similarity, cor[index])
55
56
```

0.9998038 A race car accelerates uniformly from 18.5 m/s to 46.1 m/s in 2.47 seconds. Determine the acceleration of the car.
0.9978555 A race car accelerates uniformly from 18.5 m/s to 46.1 m/s in 2.47 seconds. Determine the acceleration of the car and the distance traveled.
0.9962974 A car traveling at 22.4 m/s skids to a stop in 2.35 s. Determine the skidding distance of the car (assume uniform acceleration).
0.9949142 A car starts from rest and accelerates uniformly for 5.21 seconds over a distance of 118 m. Determine the acceleration of the car.
0.9938829 A car starts from rest and accelerates uniformly over a time of 5.21 seconds for a distance of 118 m. Determine the acceleration of the car.
0.9926287 The speed of a car is reduced from 90 km/hr to 36 km/hr in 5 s. What is a distance travelled by the car during this time interval.
0.99078214 A dragster accelerates to a speed of 112 m/s over a distance of 398 m. Determine the acceleration (assume uniform) of the dragster.
0.98789846 A bike accelerates uniformly from rest to a speed of 7.10 m/s over a distance of 35.4 m. Determine the acceleration of the bike.
0.9844623 A car accelerates from 0 km/h to 100 km/h in 5 seconds. It can decelerate at 5 m/s². If the driver applies the brakes fully, what is the
0.9841882 The speed of a car is reduced from 90 km/hr at 10 seconds to 36 km/hr in 15 s. What is a distance travelled by the car during this time
0.9837394 A car accelerates uniformly from rest at a rate of +10 m/s². How far will the car travel in 4 seconds?
0.9835155 A car starts from rest at a stop light and reaches 20 m/s in 3.5 s. Determine the acceleration of the car.
0.9833825 A car starts from rest at a stop light and reaches 20 m/s in 3.5 s. Determine the acceleration of the car.
0.98198415 A car accelerates uniformly from 0 to 72 km/h in 11.5 seconds. What is the acceleration of the car in m/s²? What is the position of the
0.9804243 A car with an initial speed of 23.7 km/hr accelerates at a uniform rate of 0.92 m/s² for 3.6 sec. Find the final speed and displacement of
0.97907846 What is the acceleration of a car that can reach a speed of 300 km/h from rest within 24 seconds. What is the distance traveled by the
0.978465 A jalopy traveling 70 km/hr comes to a stop at 120 m. Find the jalopy's acceleration.
0.97581384 A driver of a car traveling at -15 m/s applies the brakes, causing a uniform acceleration of +2 m/s². If the brakes are applied for 2.1
0.97386354 A car with an initial speed of 23.7 km/h accelerates at a uniform rate of 0.92 m/s² for 3.6 s. Find the final speed and the displacement
0.9723891 From rest, a car accelerated at 8 m/s² for 10 seconds. a) What is the position of the car at the end of the 10 seconds? b) What is the
0.96989955 A car starts from rest and travels for 5.0 s with a uniform acceleration of -1.5 m/s². What is the final velocity of the car? How far

So, we can see that there are similarity value changes and we start seeing the query related a document right. So, car acceleration velocity far, so we have a race car the first one contains all the values related to that, second one third the speed of the car. So, there is a dragster also. So, what has happened here? So, it is able to find a document that does not have the word car you remember the idea of having the distributed embedding is to find similar terms. So, it is somehow able to find the dragster similar to the car ok.

(Refer Slide Time: 16:53)



```
39 dictionary.filter_extremes(no_below=1, keep_n=10)
40
41 doc_term_matrix = [dictionary.doc2bow(tokens) for tokens in preprocessing]
42 tfidf = models.TfidfModel(doc_term_matrix)
43 corpus_tfidf = tfidf[doc_term_matrix]
44
45 lsi = models.LsiModel(corpus_tfidf, idf=word_dictionary, num_topics=5) # initialize an LSI transformation
46 doc = 'car acceleration velocity far'
47 vec_bow = dictionary.doc2bow(doc.lower().split())
48
49 vec_lsi = lsi[vec_bow] # convert the query to LSI space
50 index = similarities.MatrixSimilarity(lsi[doc_term_matrix])
51 unsorted_similarity = index[vec_lsi]
52 sorted_similarity = sorted(zip(index.sorted_similarity, key=lambda item: -item[1]))
53 index, similarity = sorted_similarity
54 print(similarity, cor[index])
55
56
```

0.9841882 The speed of a car is reduced from 90 km/hr at 10 seconds to 36 km/hr in 15 s. What is a distance travelled by the car during this time?

0.9837394 A car accelerates uniformly from rest at a rate of +18 m/s². How far will the car travel in 4 seconds?

0.9833169 A car starts from rest at a stop light and reaches 20 m/s in 3.5 s. Determine the acceleration of the car.

0.9831195 A car starts from rest at a stop light and reaches 20 m/s in 3.5 s. Determine the acceleration of the car.

0.98190415 A car accelerates uniformly from 0 to 72 km/h in 11.5 seconds. What is the acceleration of the car in m/s²? What is the position of the car at the end of the 11.5 seconds?

0.9804243 A car with an initial speed of 23.7 km/hr accelerates at a uniform rate of 0.92 m/s² for 3.6 sec. Find the final speed and displacement of the car.

0.97967744 What is the acceleration of a car that can reach a speed of 360 km/h from rest within 24 seconds. What is the distance traveled by the car during this time?

0.976845 A jalopy traveling 70 km/hr comes to a stop at 120 m. Find the jalopy's acceleration.

0.97583384 A driver of a car traveling at -15 m/s applies the brakes, causing a uniform acceleration of +2 m/s². If the brakes are applied for 2.5 s, what is the final velocity of the car?

0.97390594 A car with an initial speed of 23.7 km/h accelerates at a uniform rate of 0.92 m/s² for 3.6 s. Find the final speed and the displacement of the car.

0.9723092 From rest, a car accelerated at 8 m/s² for 10 seconds. a) What is the position of the car at the end of the 10 seconds? b) What is the final velocity of the car?

0.9698095 A car starts from rest and travels for 5.0 s with a uniform acceleration of -1.5 m/s². What is the final velocity of the car? How far does the car travel during this time?

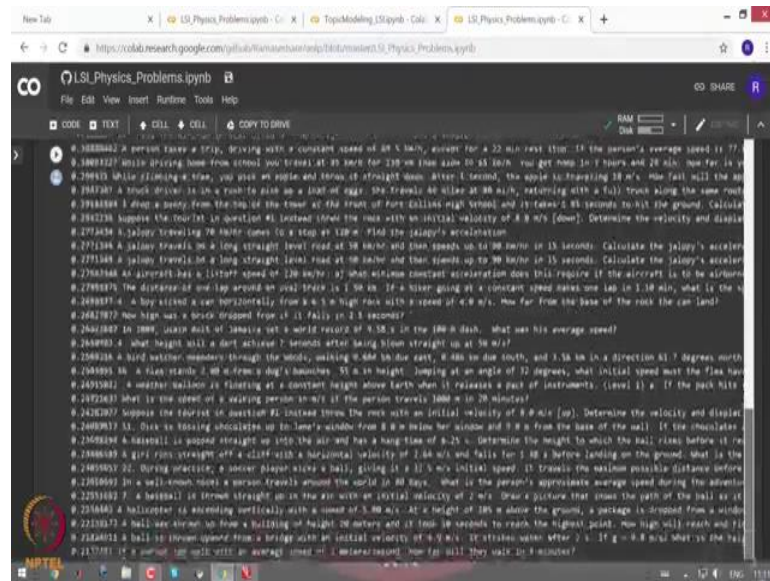
0.9656823 A car starts from rest and travels for 5 seconds with a uniform acceleration of -1.5 m/s². What is the final velocity of the car? How far does the car travel during this time?

0.9654862 Ben Rushie is waiting at a stoplight. When it finally turns green, Ben accelerated from rest at a rate of a 6.00 m/s² for a time of 4.10 s. How far does the car travel during this time?

0.96473 A jalopy is accelerated from 48 m/s to 13 m/s over 5 s. What is the displacement of the jalopy during this time?

And then we can see many car-related documents here even we can see a jalopy there and then there is a bike we saw somewhere. So, if the document that contains; the document that contains similar values as in the query is now listed I have just increased the sigma value by 2. So, by just increasing this to let us say 5 so, it is now, even more, sharpening the results better than what we saw earlier. So, you can now see that right, so lots of cars in front of the query ok.

(Refer Slide Time: 17:53)



So, if you go down here you will see the similarity value getting reduced and then towards the end just because it managed to find a few keywords, it gave very smaller values for those documents right. So, you can increase it to 10 or 15 let us say we will increase it to 15 and see what happens. So, it is changing it slightly, but we can see all the documents related to that are found ok.

So, you also want to go through this and then look at the data change the query parameters and then see whether you are getting this value or not ok. So, you can look at the corpus closely, change the query parameters and then change these parameters as well and then find out how this particular application behaves ok. So, this is one example that I have shown where we have a tf IDF based term-document matrix.

We used LSI and then in the reduced domain we tried to find the similarities between the query and the document; between the query and the query and every document in the corpus.