WORD EMBEDDINGS

Venkata Reddy Konasani

CONTENTS

What is Text mining?

What is text mining

- Making sense out of text data
- Datamining on text input
- Exploratory data analysis on text data
- Also known as Text Analytics
- What is NLP – Natural Language Processing
- Text data is not same as categorical data

Text Data Sources

- Customer Emails
- Customer feedback and reviews
- Blog articles
- Tweets and Facebook posts
- News articles
- Social media comments
- Customer verbatim in a survey
- Scanned documents of the physical forms

Numerical data is well structured

- rows and columns.
- For every record(row) we have information well organised in the form of columns.
- Each column captures a specific section of information
- Every record has almost all columns available
- Easy to perform mathematical and statistical computations

Text data is unstructured

- Most of the text data has one or two columns
- Whole data is in one column
- Each record might have different length
- Difficult to arrange it as a dataset
- Text data is not very well structured.
- Direct computation on text data not easy

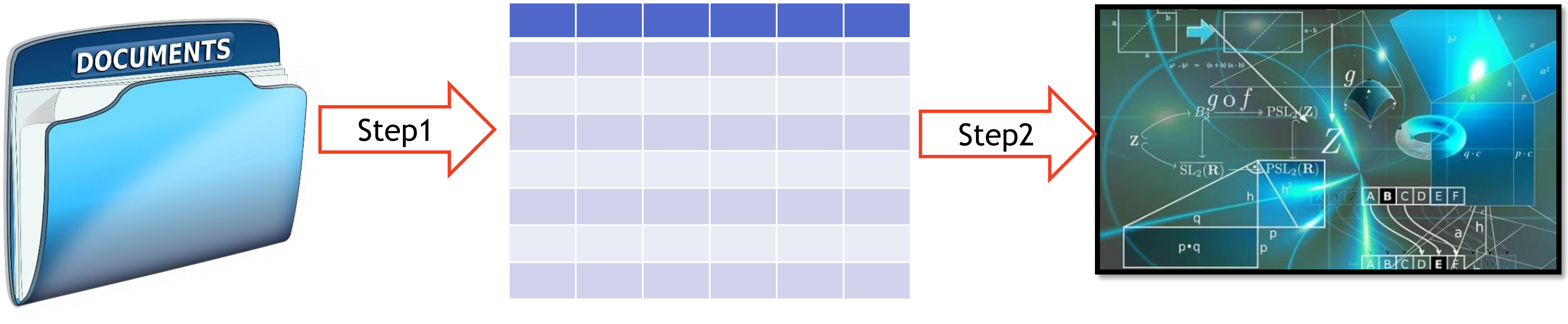
Computers don’t Understand Language

- Direct computation on text data not easy.
- Computers don’t understand a **Sentence** or a **Word** or any **Underlying Emotion**.
- We need to bring the data structure to a point where computers can convert text data into number, process the numbers, convert those numbers back to Text data.

Two Steps in NLP Model building

Step1=> Convert text data into numerical data

Step2=> Build models on numerical data



Giving structure to Unstructured text

- Two methods
  - Bag of words (One hot encoding)
  - Word2Vec

Document Term Matrix

Document Term Matrix

- Document – text document
- Can we consider each sentence as document? Can we call a sentence as a basic form of document
- We can create DTM and work with sklearn and other regular packages

- Doc1: Loved this place
- Doc2: At this place, crust is not good.
- Doc3: Loved it, good thin crust pizza.

Document Term Matrix

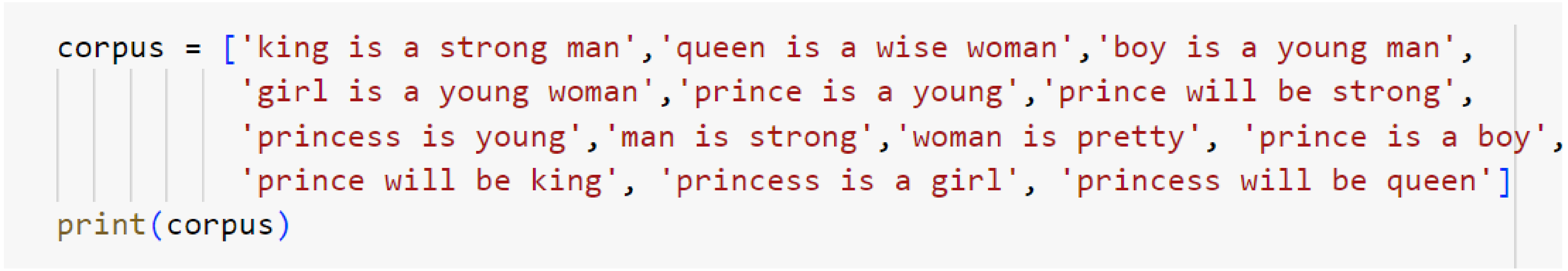
Doc1: Loved this place, good pizza
Doc2: At this place, crust is not good. pizza is not good.
Doc3: Loved it, good thin crust pizza.

Terms

Documents

	loved	this	place	at	crust	is	not	good	it	thin	pizza
Doc1	1	1	1					1			1
Doc2		1	1	1	1	2	2	2			1
Doc3	1				1			1	1	1	1

Example- Document Term Matrix



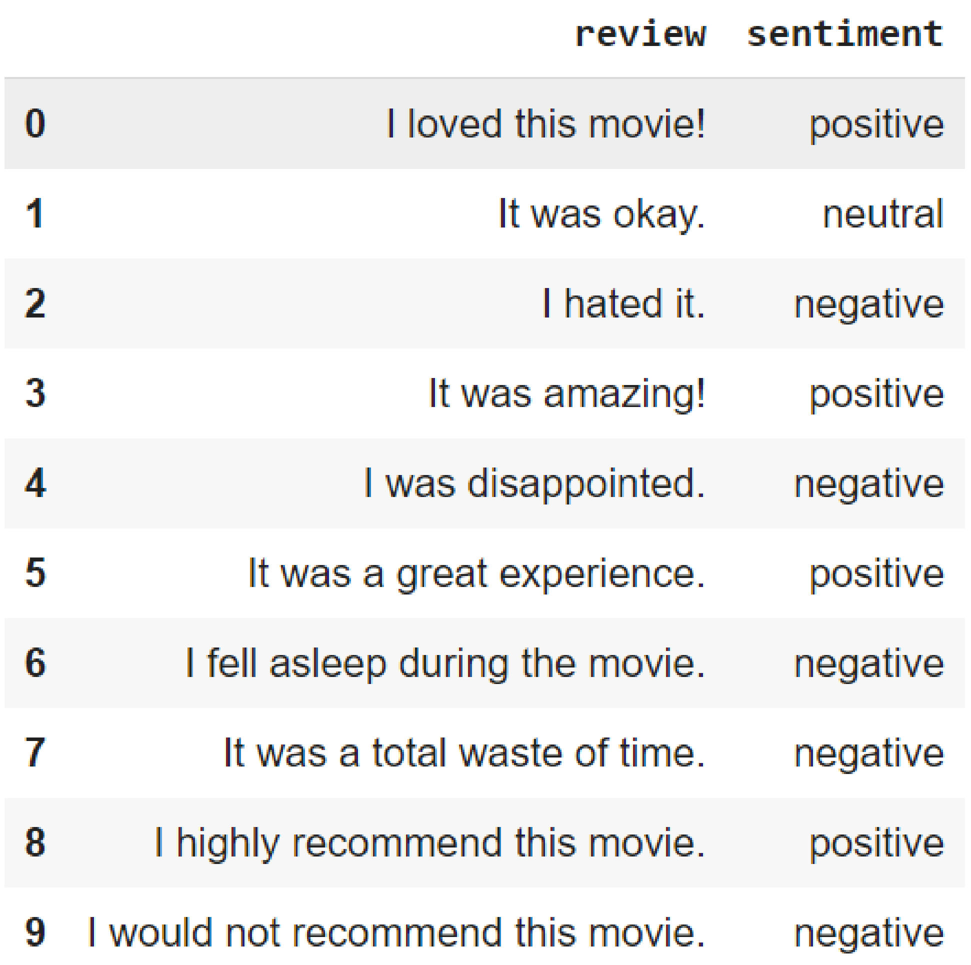
```python
corpus = ['king is a strong man','queen is a wise woman','boy is a young man',
          'girl is a young woman','prince is a young','prince will be strong',
          'princess is young','man is strong','woman is pretty', 'prince is a boy',
          'prince will be king', 'princess is a girl', 'princess will be queen']
print(corpus)
```

Example- Document Term Matrix

```python
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer()
DTM = cv.fit_transform(corpus)
DTM = pd.DataFrame(DTM.toarray(), columns=cv.get_feature_names_out())
DTM
```

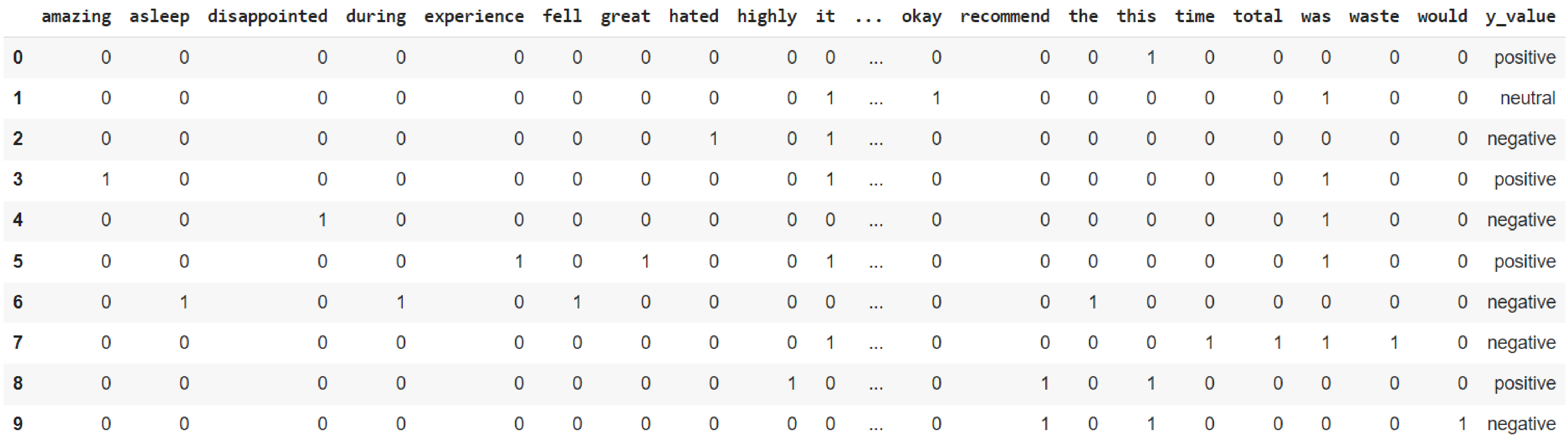
	be	boy	girl	is	king	man	pretty	prince	princess	queen	strong	will	wise	woman	young
0	0	0	0	1	1	1	0	0	0	0	1	0	0	0	0
1	0	0	0	1	0	0	0	0	0	1	0	0	1	1	0
2	0	1	0	1	0	1	0	0	0	0	0	0	0	0	1
3	0	0	1	1	0	0	0	0	0	0	0	0	0	1	1
4	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1

Sample Movie Review Data



	review	sentiment
0	I loved this movie!	positive
1	It was okay.	neutral
2	I hated it.	negative
3	It was amazing!	positive
4	I was disappointed.	negative
5	It was a great experience.	positive
6	I fell asleep during the movie.	negative
7	It was a total waste of time.	negative
8	I highly recommend this movie.	positive
9	I would not recommend this movie.	negative

Document Term Matrix On Review Data



	amazing	asleep	disappointed	during	experience	fell	great	hated	highly	it	...	okay	recommend	the	this	time	total	was	waste	would	y_value
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	1	0	0	0	0	0	positive
1	0	0	0	0	0	0	0	0	0	1	...	1	0	0	0	0	0	1	0	0	neutral
2	0	0	0	0	0	0	0	1	0	1	...	0	0	0	0	0	0	0	0	0	negative
3	1	0	0	0	0	0	0	0	0	1	...	0	0	0	0	0	0	1	0	0	positive
4	0	0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	1	0	0	negative
5	0	0	0	0	1	0	1	0	0	1	...	0	0	0	0	0	0	1	0	0	positive
6	0	1	0	1	0	1	0	0	0	0	...	0	0	1	0	0	0	0	0	0	negative
7	0	0	0	0	0	0	0	0	0	1	...	0	0	0	0	1	1	1	1	0	negative
8	0	0	0	0	0	0	0	0	1	0	...	0	1	0	1	0	0	0	0	0	positive
9	0	0	0	0	0	0	0	0	0	0	...	0	1	0	1	0	0	0	0	1	negative

WHY “BAG OF WORDS” TECHNIQUE FAILS?

- We want to perform some analysis on text data. How do we convert text into numerical data?
  - By keeping all the meaningful relations intact
  - By loosing very less information in that process of conversion

HOW DO YOU CONVERT THIS CORPUS INTO NUMBERS?

corpus = ['king is a strong man’, 'queen is a wise woman’, 'boy is a young man', 'girl is a young woman', 'prince is a young', 'prince will be strong', 'princess is young', 'man is strong', 'woman is pretty', 'prince is a boy', 'prince will be king', 'princess is a girl', 'princess will be queen']

IDENTIFY THE UNIQUE WORDS

young	man	king	woman	she	strong	prince	girl	wise	princess	pretty	he	boy	queen

These are all unique words, give one unique identify to each of these. Or simply perform one hot encoding.

ONE HOT ENCODING / BAG OF WORDS

- What is one hot encoding? Giving number 1 when the word appears and 0 when it doesn't appear.

Young	man	king	woman	she	strong	prince	girl	wise	princess	pretty	he	boy	queen
1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1

FEW ONE-HOT ENCODED EXAMPLES FROM OUR DATA

Young	prince	king	woman	princess
1	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	0
0	0	0	0	0
0	1	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	1
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

CORPUS AFTER CONVERTING INTO NUMBERS

Corpus_v1 =[king strong man, queen wise woman, boy young man, girl young woman, prince young king, he strong, princess young queen, she pretty, man strong, woman pretty, prince boy he king, princess girl she queen]

king	strong	man	..	..	..	..	..	..	..	..	..	she	queen
0	0	0	..	..	..	..	..	..	..	..	..	0	0
0	0	1	..	..	..	..	..	..	..	..	..	0	0
1	0	0	..	..	..	..	..	..	..	..	..	0	0
0	0	0	..	..	..	..	..	..	..	..	..	0	0
0	0	0	..	..	..	..	..	..	..	..	..	1	0
0	1	0	..	..	..	..	..	..	..	..	..	0	0
0	0	0	..	..	..	..	..	..	..	..	..	0	0
0	0	0	..	..	..	..	..	..	..	..	..	0	0
0	0	0	..	..	..	..	..	..	..	..	..	0	0
0	0	0	..	..	..	..	..	..	..	..	..	0	0
0	0	0	..	..	..	..	..	..	..	..	..	0	0
0	0	0	..	..	..	..	..	..	..	..	..	0	0
0	0	0	..	..	..	..	..	..	..	..	..	0	0
0	0	0	..	..	..	..	..	..	..	..	..	0	1

WHERE ONE-HOT ENCODING WORKS

- One hot encoding works perfectly in scenarios where we are converting categorical data into numerical data.
- It most of the classification problems, we convert the target variable into one-hot encoded values

Region
East
West
North
South
West

East	West	North	South
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1
0	1	0	0

WHERE ONE-HOT ENCODING DOES NOT WORK

- In pure text data, one hot encoding doesn’t preserve the relations and context.
- In our example
  - king, prince, man are very similar to each other they must have similar numerical values
  - Similarly queen, princess, woman are connected.
- All the vectors in the one hot encoded version are orthogonal. In the one hot encoded version the cosine similarity between
  - king and prince is zero
  - king and man is zero
  - queen and princess is zero

ISSUES WITH ONE-HOT ENCODING

- Failed to capture the relational structure of the corpus.
- One hot vector for every unique word, that leads to too many dimensions.
- One hot vector data is very sparse.
- Context and surrounding words are completely ignored

WE NEED A NEW REPRESENTATION

- That can preserve the word relations
- That has lesser dimensions
- A representation that doesn't just look at one word at a time, it should capture the context and surrounding words.
- A representation that shows king is close to prince and man; similarly queen, princess, woman are close to each other.

HOW TO GET THE CONTEXT

- By looking at the surrounding words
- Surrounding words is nothing but the context, if not full context, atleast a good representation of the context

THE IDEA TO KEEP THE CONTEXT

"You shall know a word by the company it keeps" (J. R. Firth 1957: 11)

king	**strong**	**man**

We can consider the window of surrounding words as context.

THE IDEA TO KEEP THE CONTEXT

"You shall know a word by the company it keeps" (J. R. Firth 1957: 11)

king	strong	man

king	strong
king	man

THE IDEA TO KEEP THE CONTEXT

“You shall know a word by the company it keeps” (J. R. Firth 1957: 11)

🟥 king	🟪 strong	🟦 man

king	strong
king	man
strong	king
strong	man

THE IDEA TO KEEP THE CONTEXT

"You shall know a word by the company it keeps" (J. R. Firth 1957: 11)

king	strong	man

king	strong
king	man
strong	king
strong	man
man	king
man	strong

THE IDEA TO KEEP THE CONTEXT

“You shall know a word by the company it keeps” (J. R. Firth 1957: 11)

king	**strong**	**man**

king	strong
king	man
strong	king
strong	man
man	king
man	strong

If king is input then strong and man should be output; if strong is input then king and man should be output.

WORD2VEC INTRODUCTION

- word2vec computes vector representation for words
- word2vec tries to convert words in to numerical vectors so that similar words share a similar vector representation.
- word2vec is the name of the concept and it is not a single algorithm
- word2vec is not a deep learning technique like RNN or CNN. (no unsupervised pre-training of layers)

WORD2VEC INTRODUCTION

- Word2vec comes from the idea of preserving local context
- It has two major steps
  - Create training samples
  - Use these samples to train the neural network model
- Word2Vec tries to create the training samples by parsing through the data with a fixed window size
- After creating training samples, we will use a single layer neural network to train the model

WORD2VEC – STEP1: CREATE TRAINING SAMPLES

king	strong	man

Input	Output
king	strong
king	man
strong	king
strong	man
man	king
man	strong

- We are considering window size as 2

WORD2VEC – STEP1: CREATE TRAINING SAMPLES

king	strong	man
queen	wise	women

- We are considering window size as 2

Input	Output
king	strong
king	man
strong	king
strong	man
man	king
man	strong
queen	wise
queen	women
wise	queen
wise	women
women	queen
women	wise

WORD2VEC – STEP2: BUILD A NEURAL NETWORK MODEL

Input	Output
king	strong
king	man
strong	king
strong	man
man	king
man	strong
queen	wise
queen	women
wise	queen
wise	women
women	queen
women	wise

- This input-output pair is called as word and context pair(context is a window of words in simple terms)
- Before building the neural network we need to perform one-hot encoding to these values

WORD2VEC – STEP2: BUILD A NEURAL NETWORK MODEL

- We train this model by taking input as word and output as context
- With the hope that all the words leading to same contexts will end up having similar weights

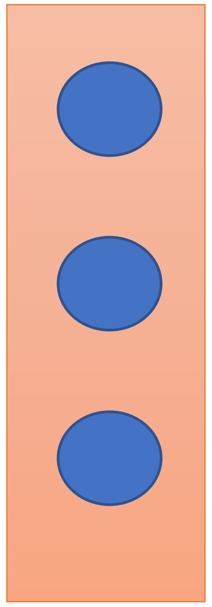
WORD2VEC – STEP2: BUILD A NEURAL NETWORK MODEL

Input
king
king
strong
strong
man
man
queen
queen
wise
wise
women
women

Output
strong
man
king
man
king
strong
wise
women
queen
women
queen
wise

WORD2VEC – STEP2: BUILD A NEURAL NETWORK MODEL

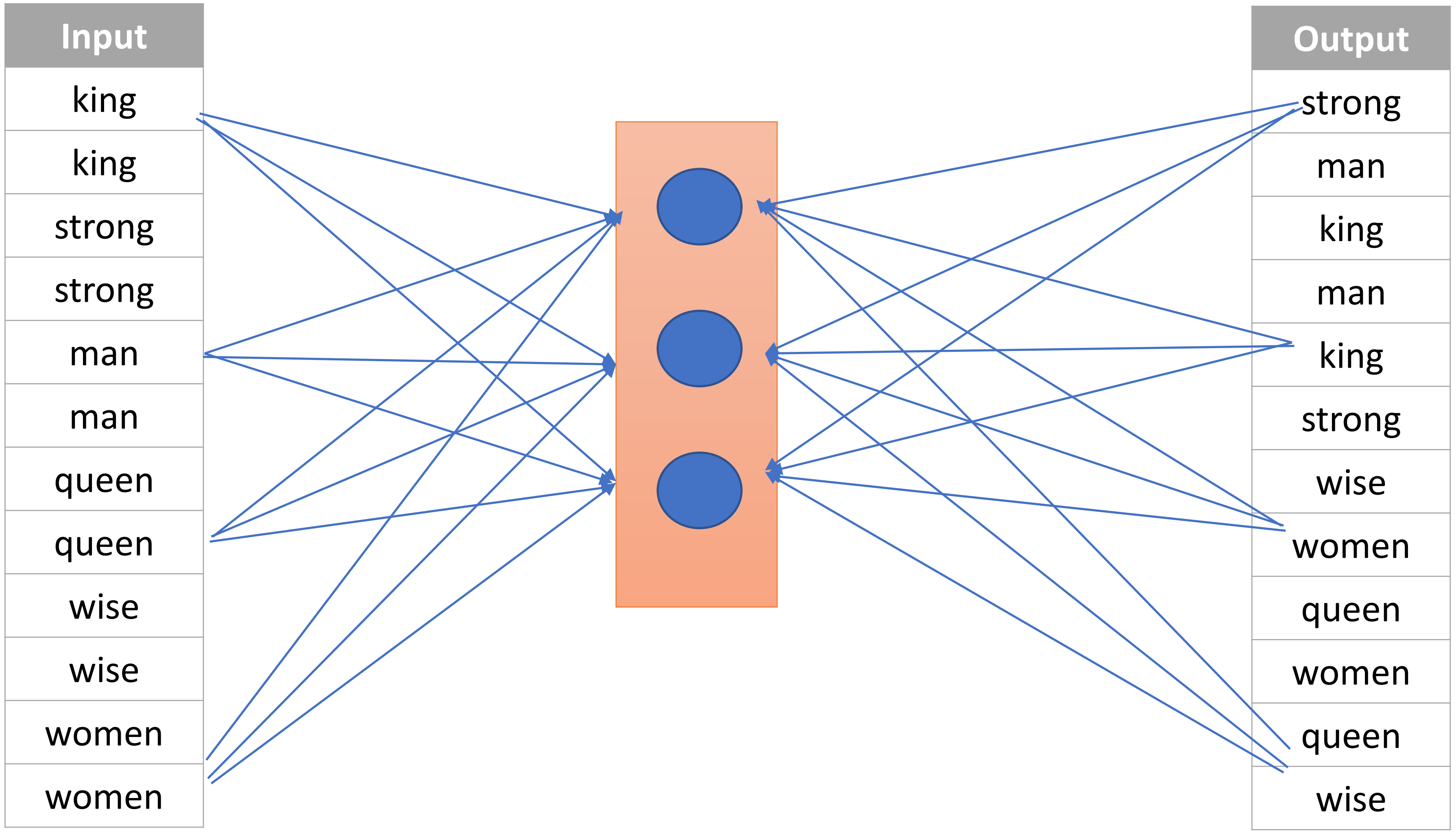
Input
king
king
strong
strong
man
man
queen
queen
wise
wise
women
women



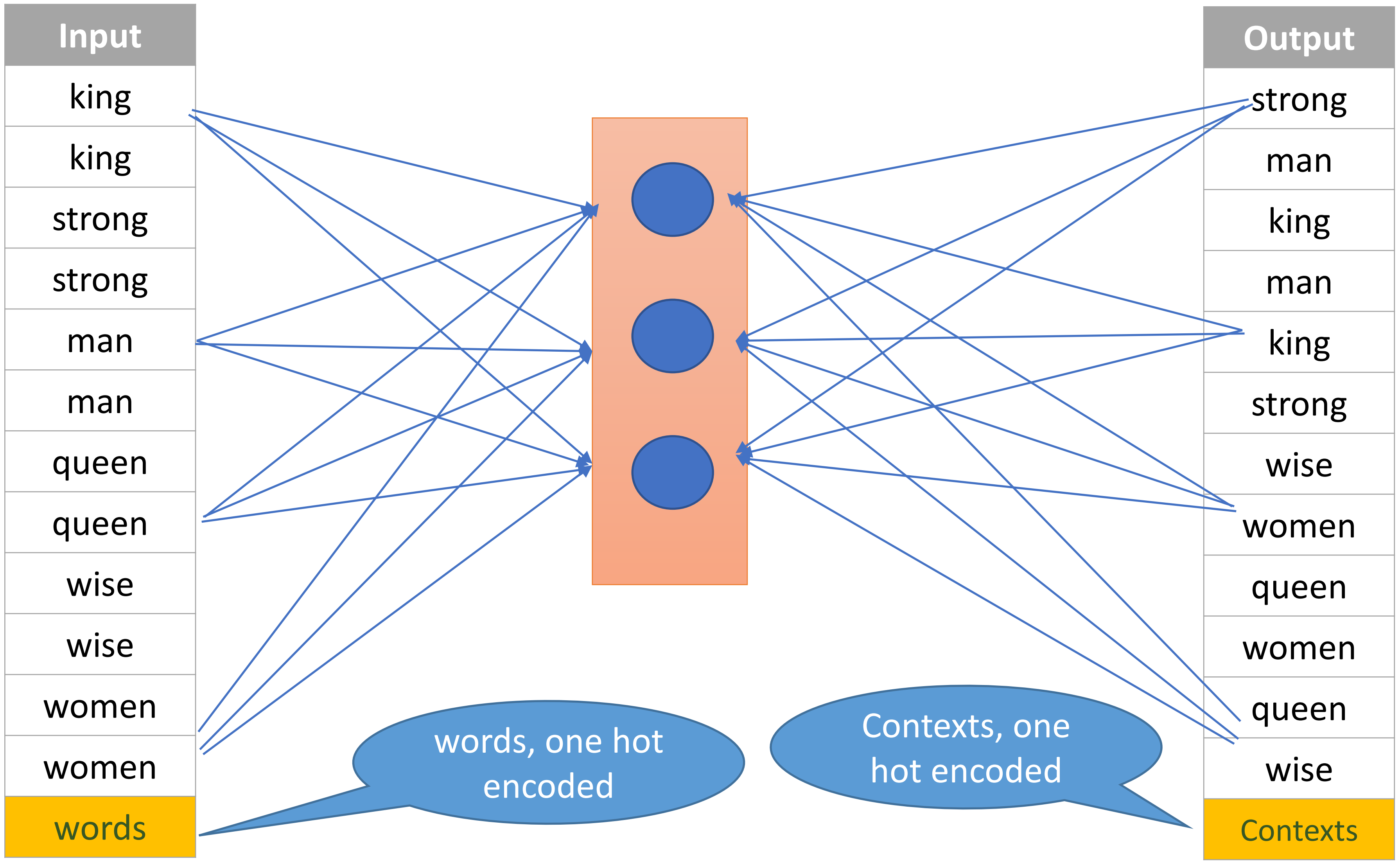
The number of hidden nodes will be the dimension of the final numerical vector space. This is a hyperparameter, we need to finetune it

Output
strong
man
king
man
king
strong
wise
women
queen
women
queen
wise

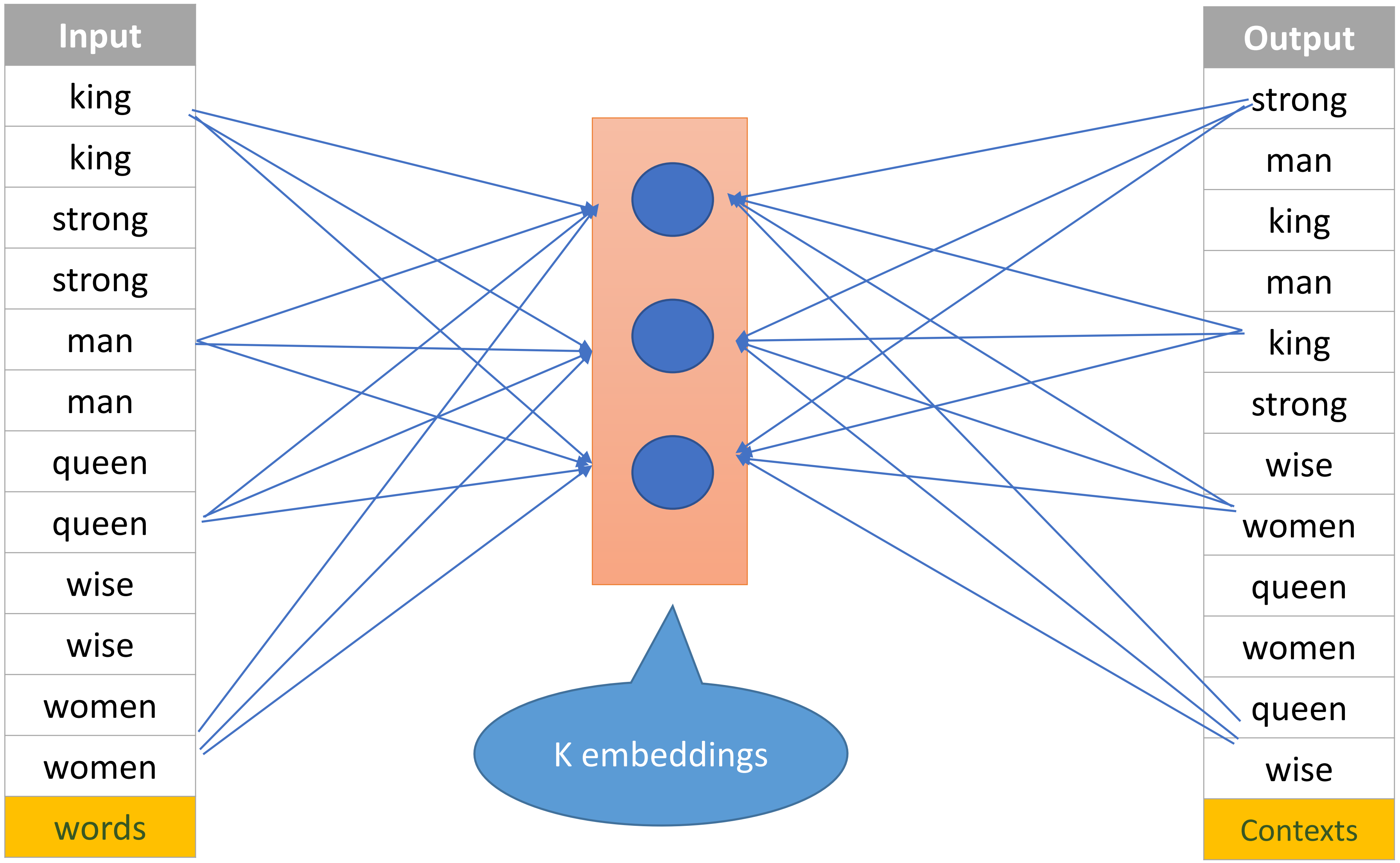
WORD2VEC – STEP2: BUILD A NEURAL NETWORK MODEL



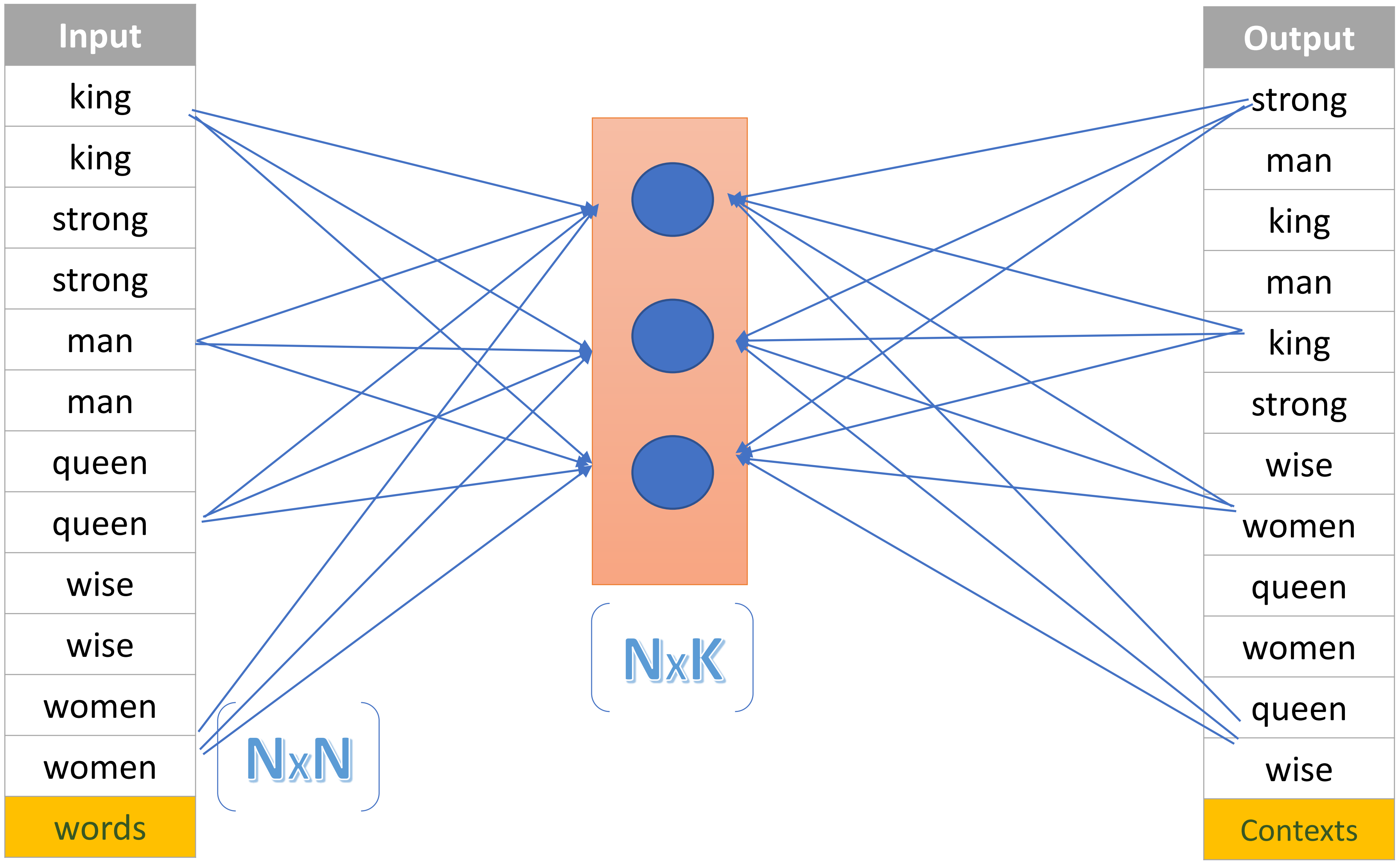
WORD2VEC – STEP2: BUILD A NEURAL NETWORK MODEL



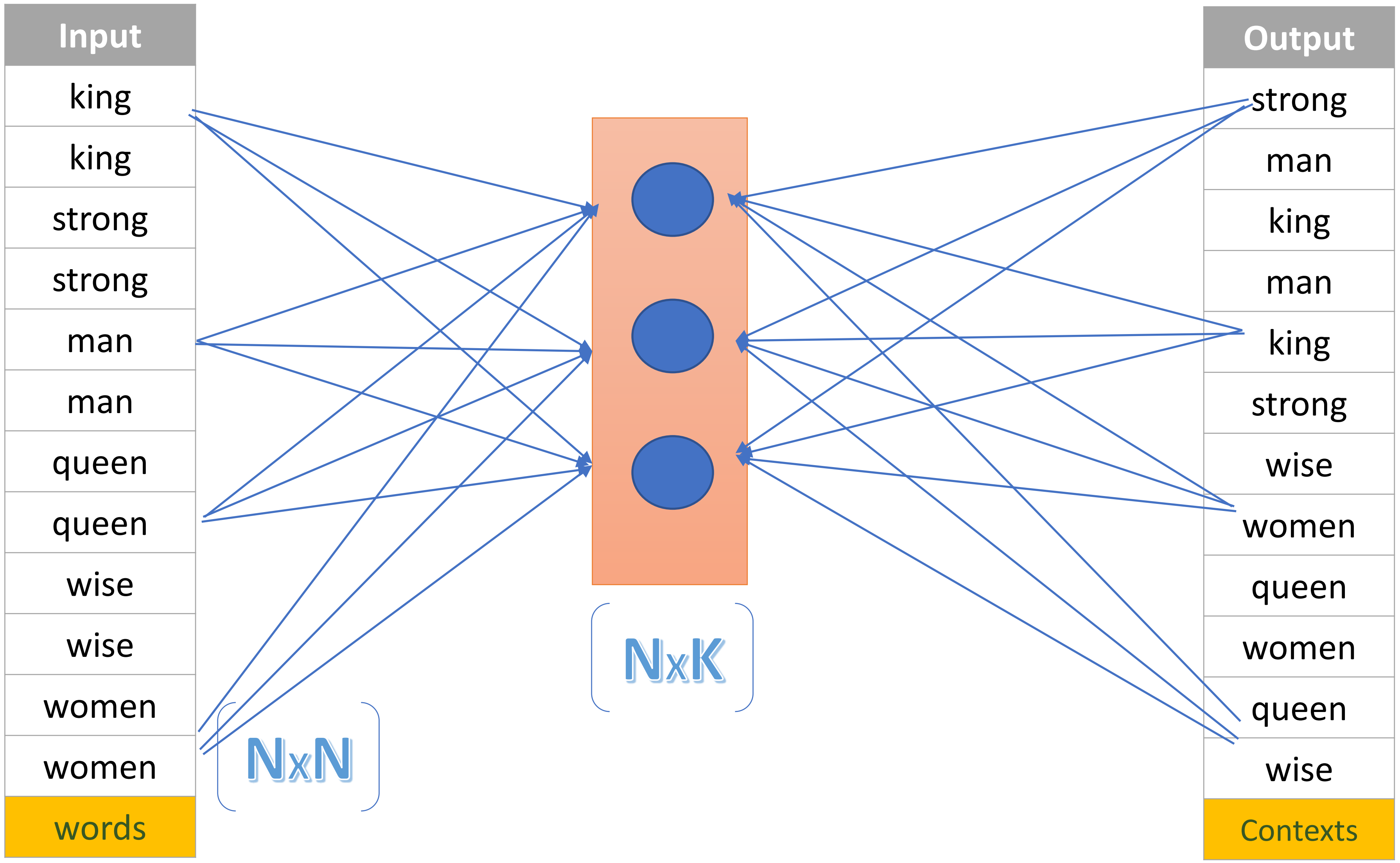
WORD2VEC – STEP2: BUILD A NEURAL NETWORK MODEL



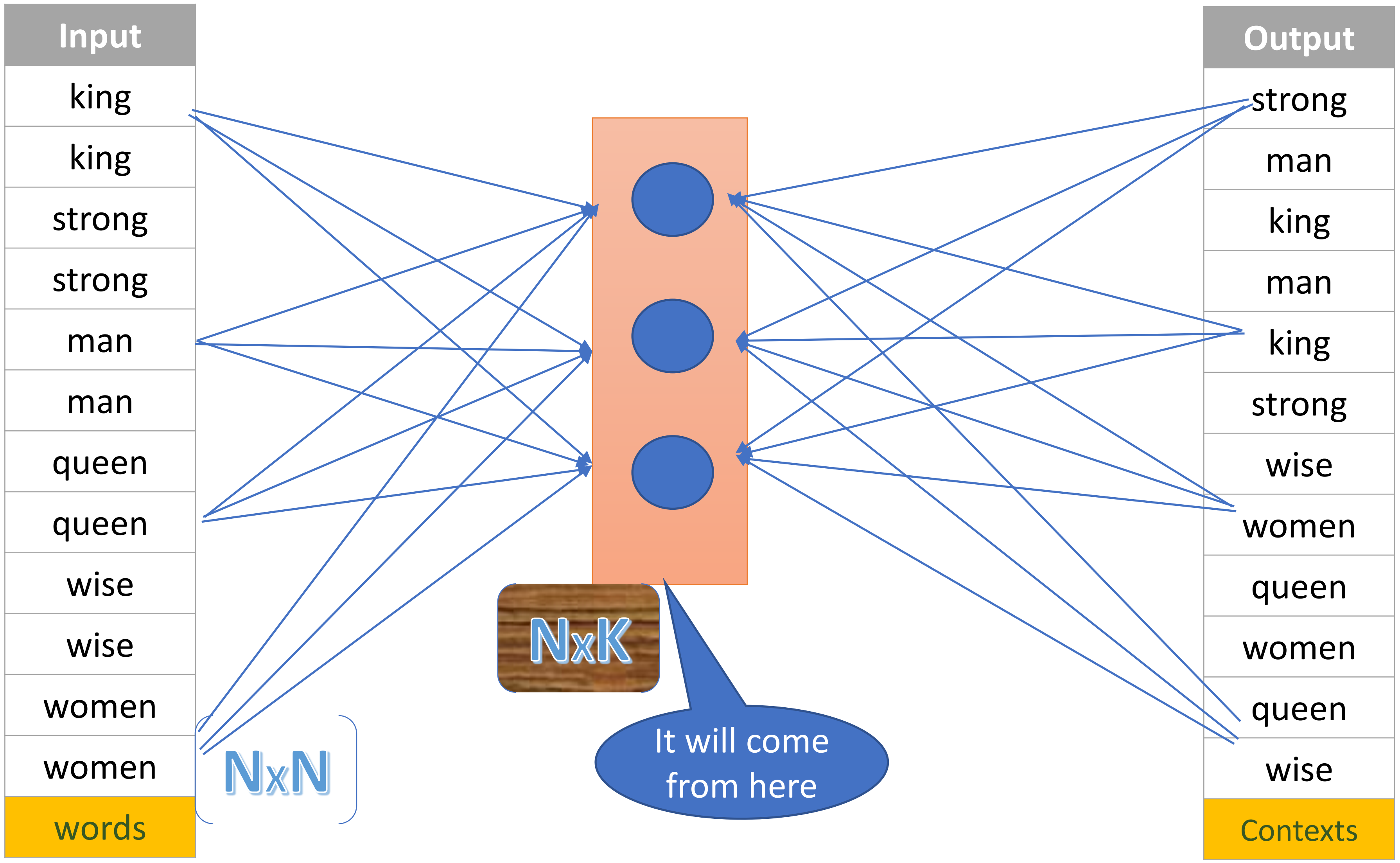
WORD2VEC – STEP2: MATRIX SIZE



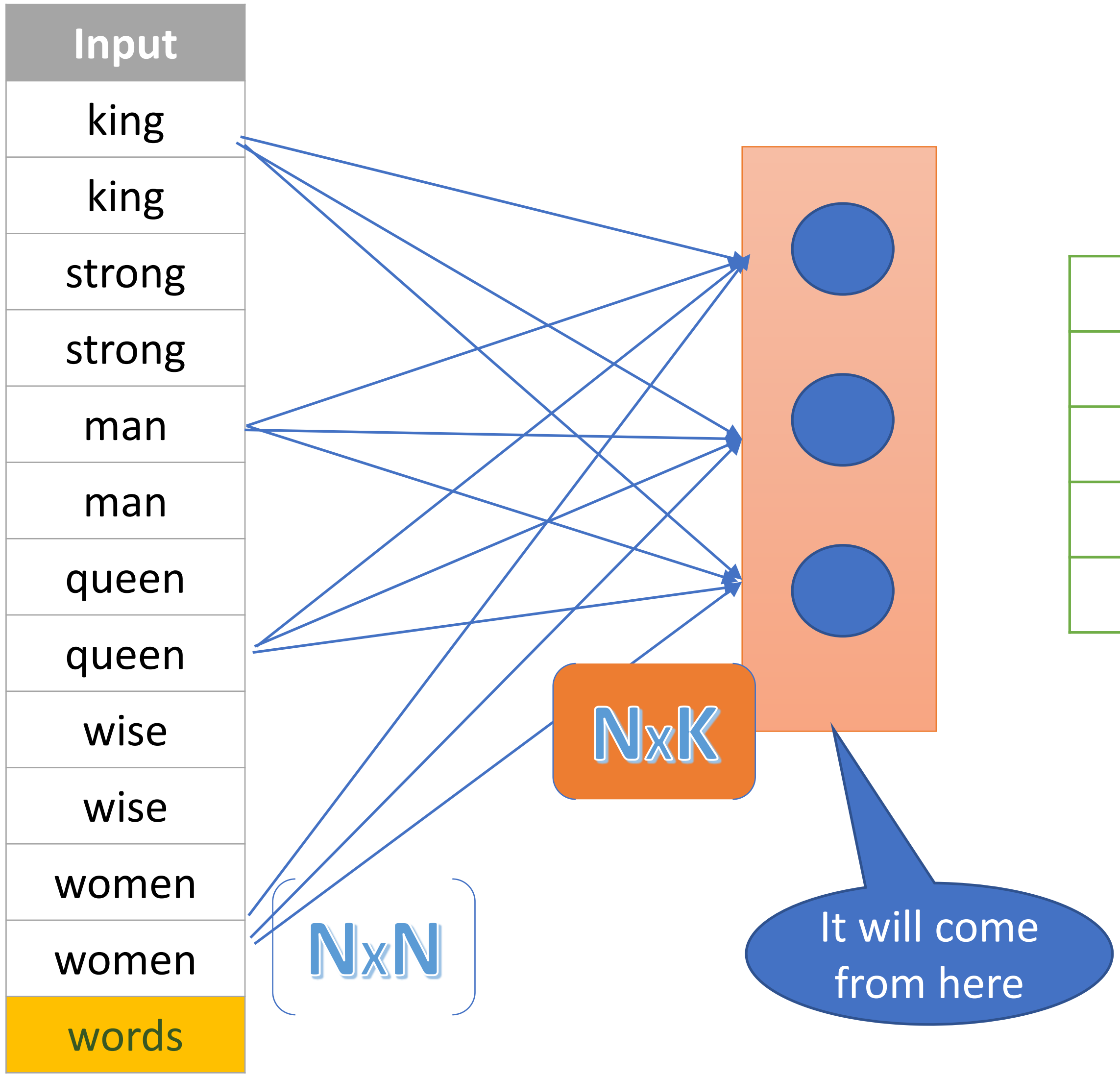
WHICH MATRIX IS THE FINAL RESULT OF WORD2VEC?



WHAT IS THE FINAL RESULT OF THE WORD2VEC



RESULT OF THE WORD2VEC

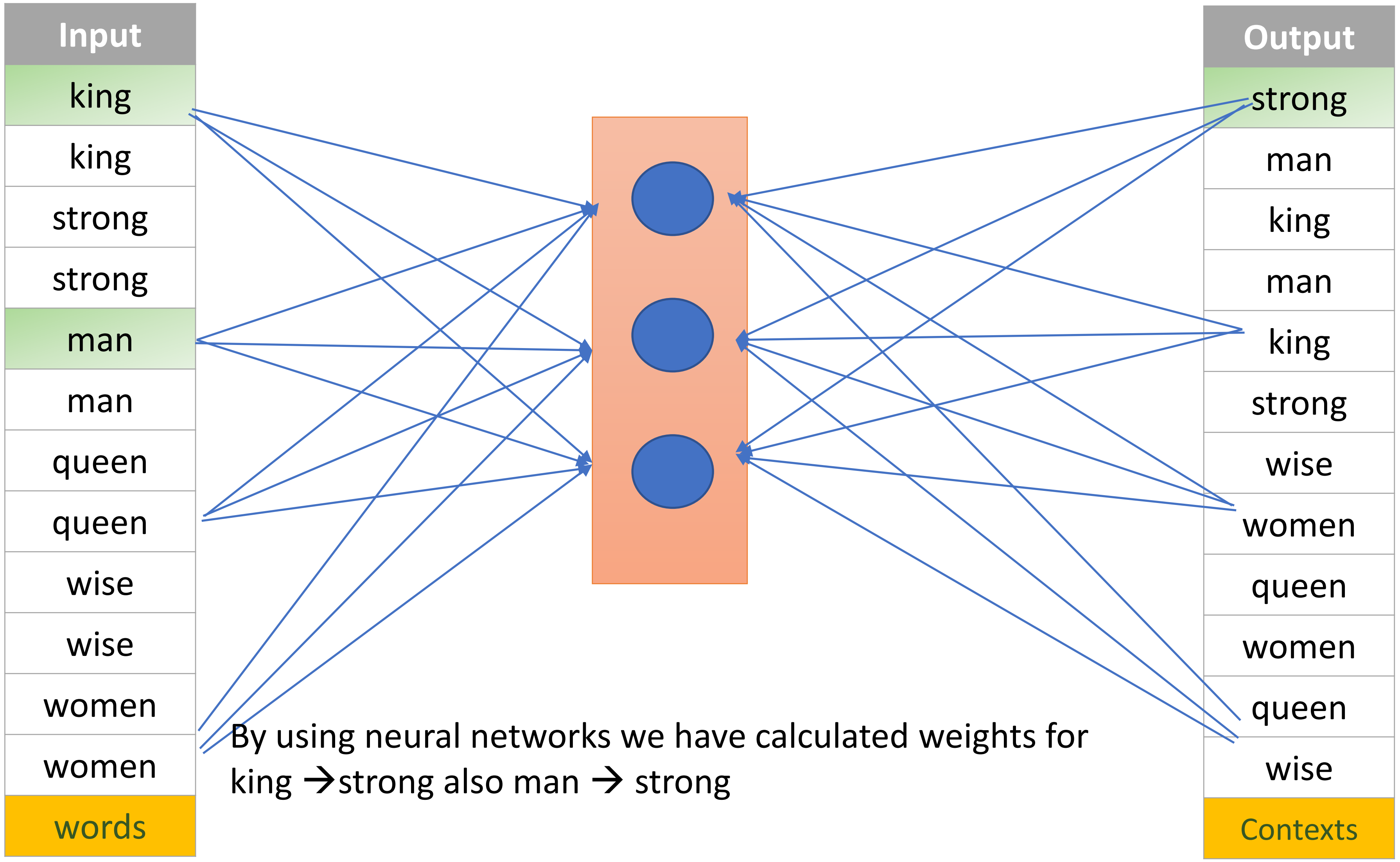


king	3.248315	-0.292609	2.028029
man	1.032173	3.037509	-1.810387
strong	3.659783	0.865091	-1.710116
queen	-3.151272	-2.009174	0.550185
woman	-2.420959	0.980081	-0.391963

WHY WORD2VEC WORKS BETTER THAN ONE HOT ENCODING

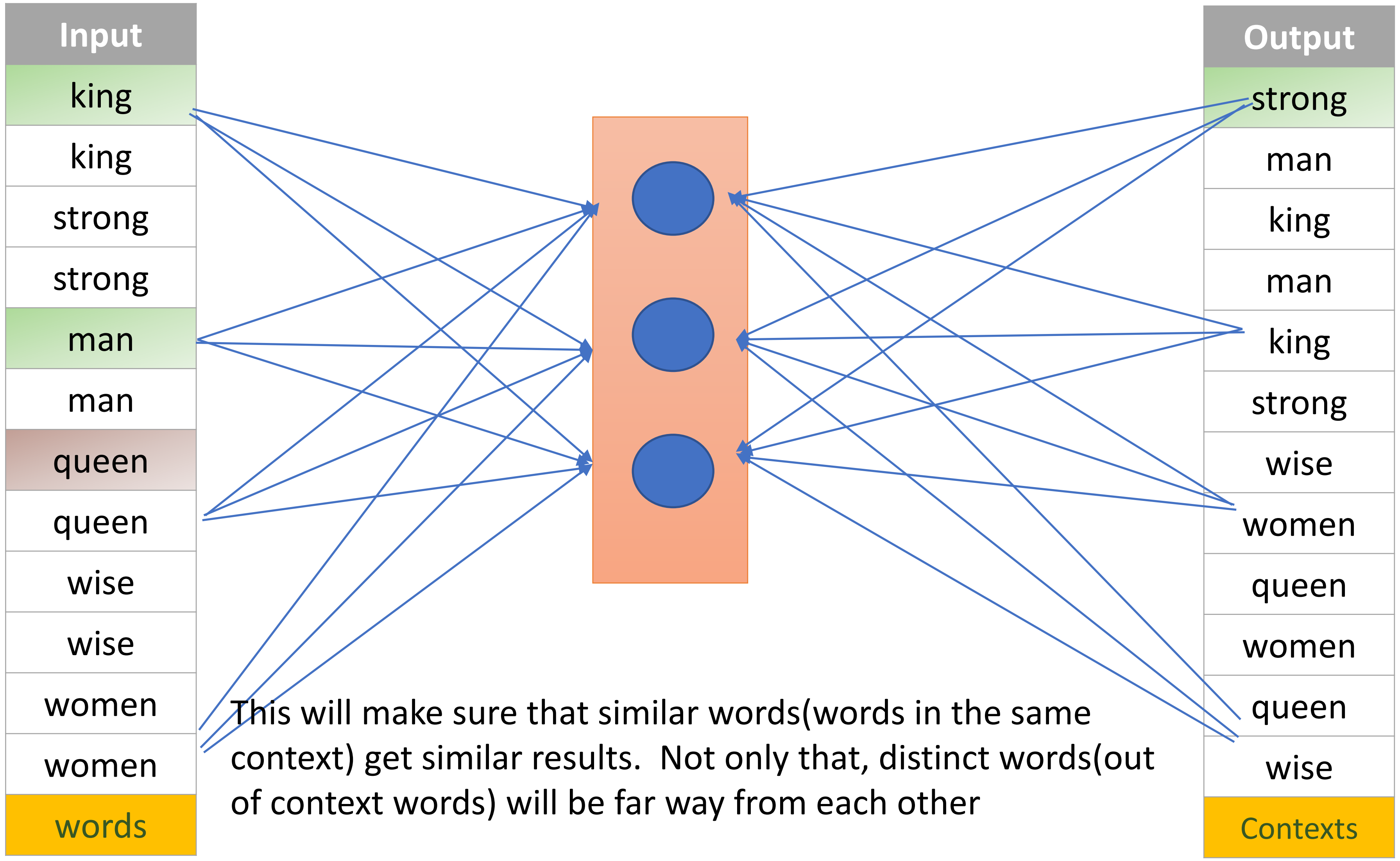
- We are focusing on the neighbouring words(contexts)
- If two words are having the same context then they should get same results from hidden layer
- In our example king is in the context of strength and man also has the context of strength.

WHY WORD2VEC WORKS BETTER THAN ONE HOT ENCODING



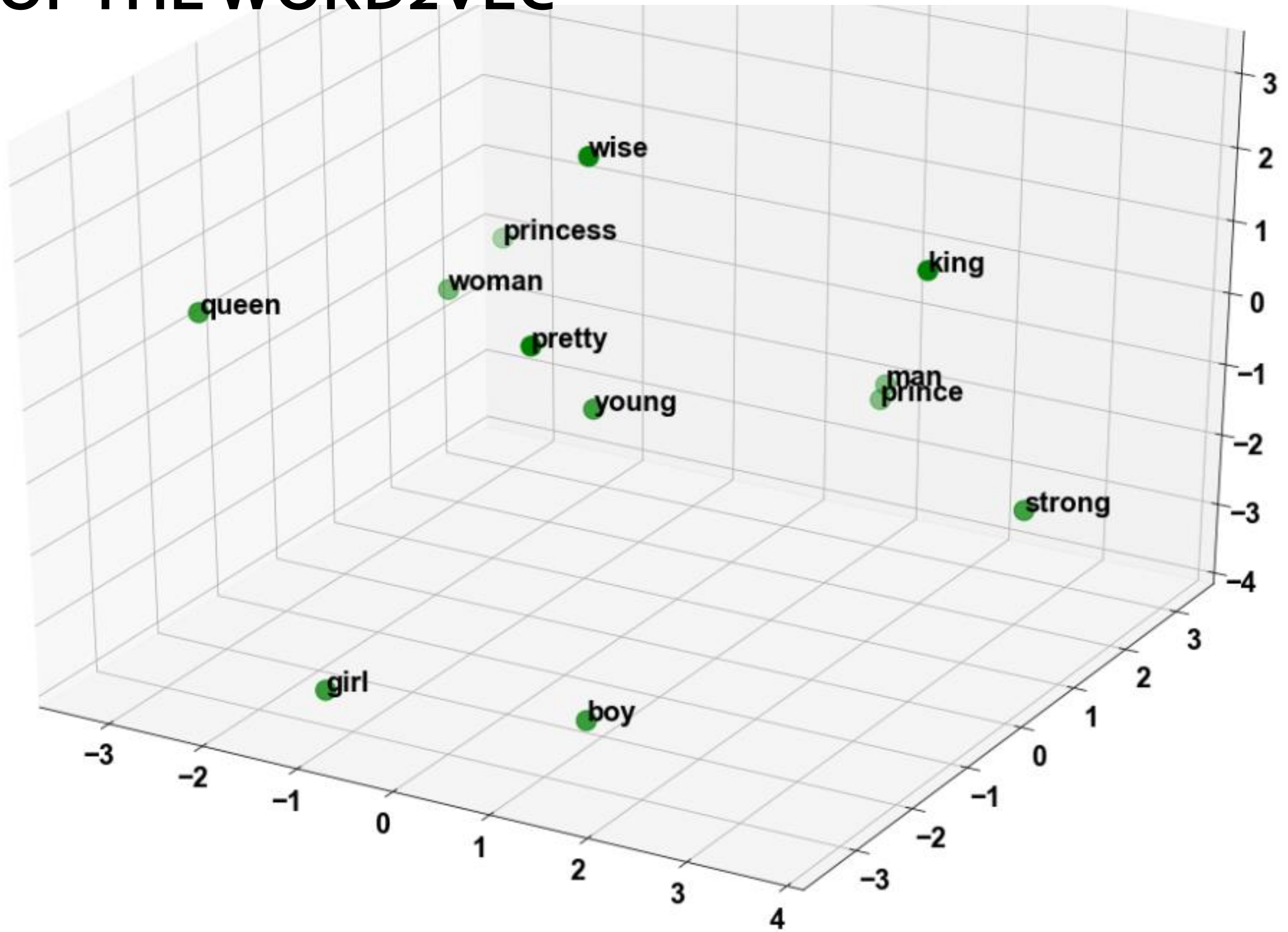
By using neural networks we have calculated weights for king →strong also man → strong

WHY WORD2VEC WORKS BETTER THAN ONE HOT ENCODING

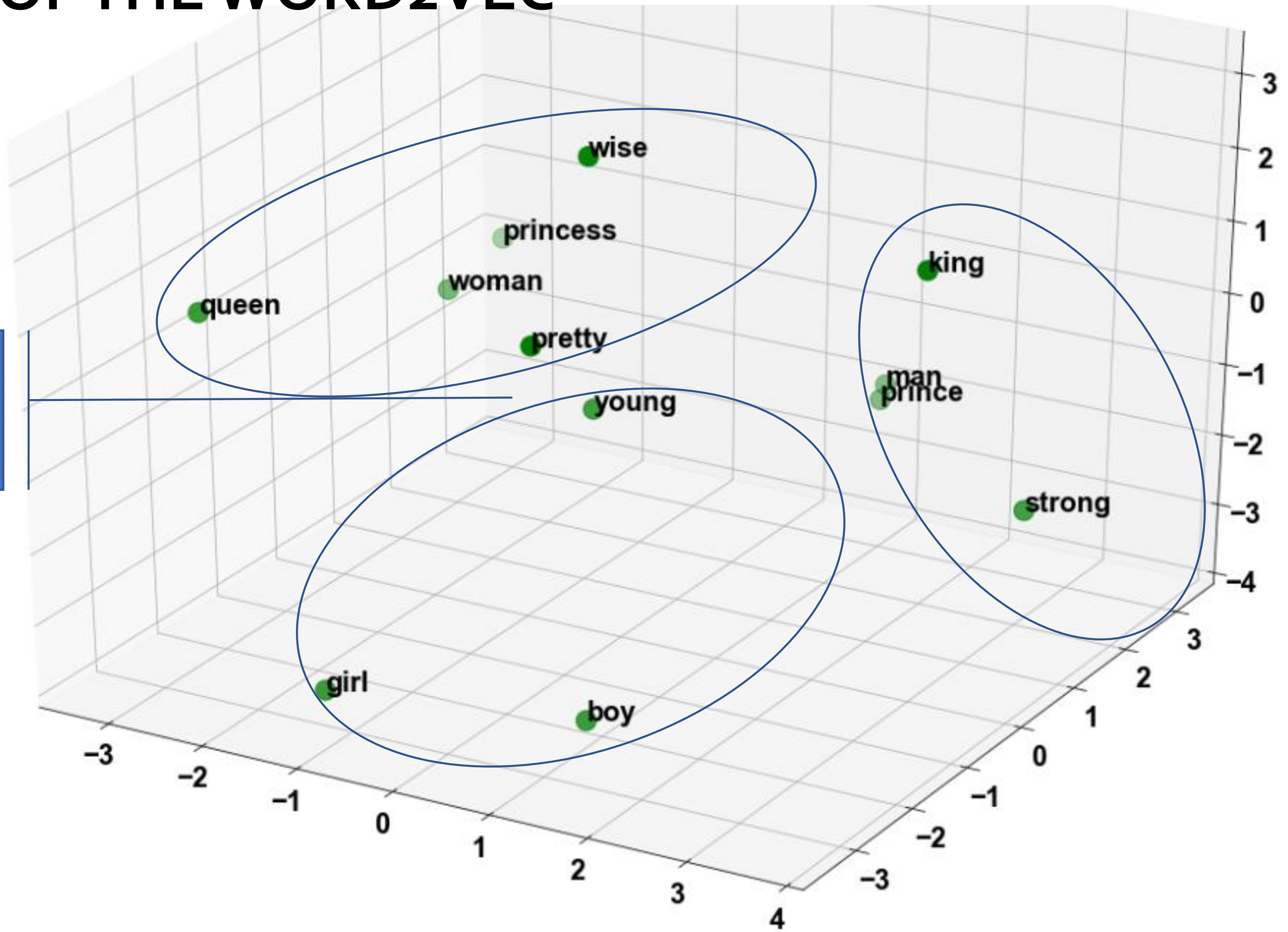


This will make sure that similar words(words in the same context) get similar results. Not only that, distinct words(out of context words) will be far way from each other

RESULT OF THE WORD2VEC

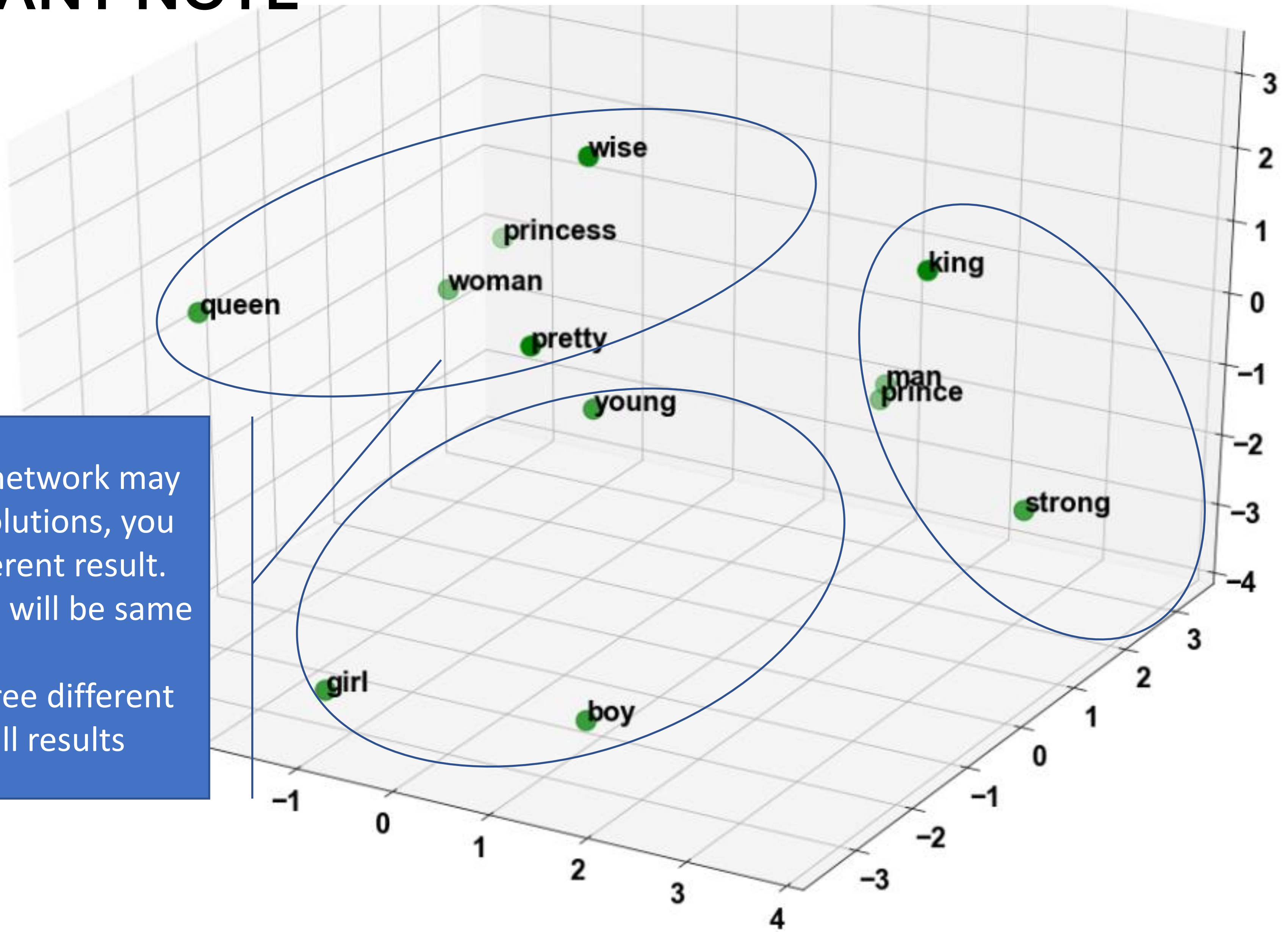


RESULT OF THE WORD2VEC



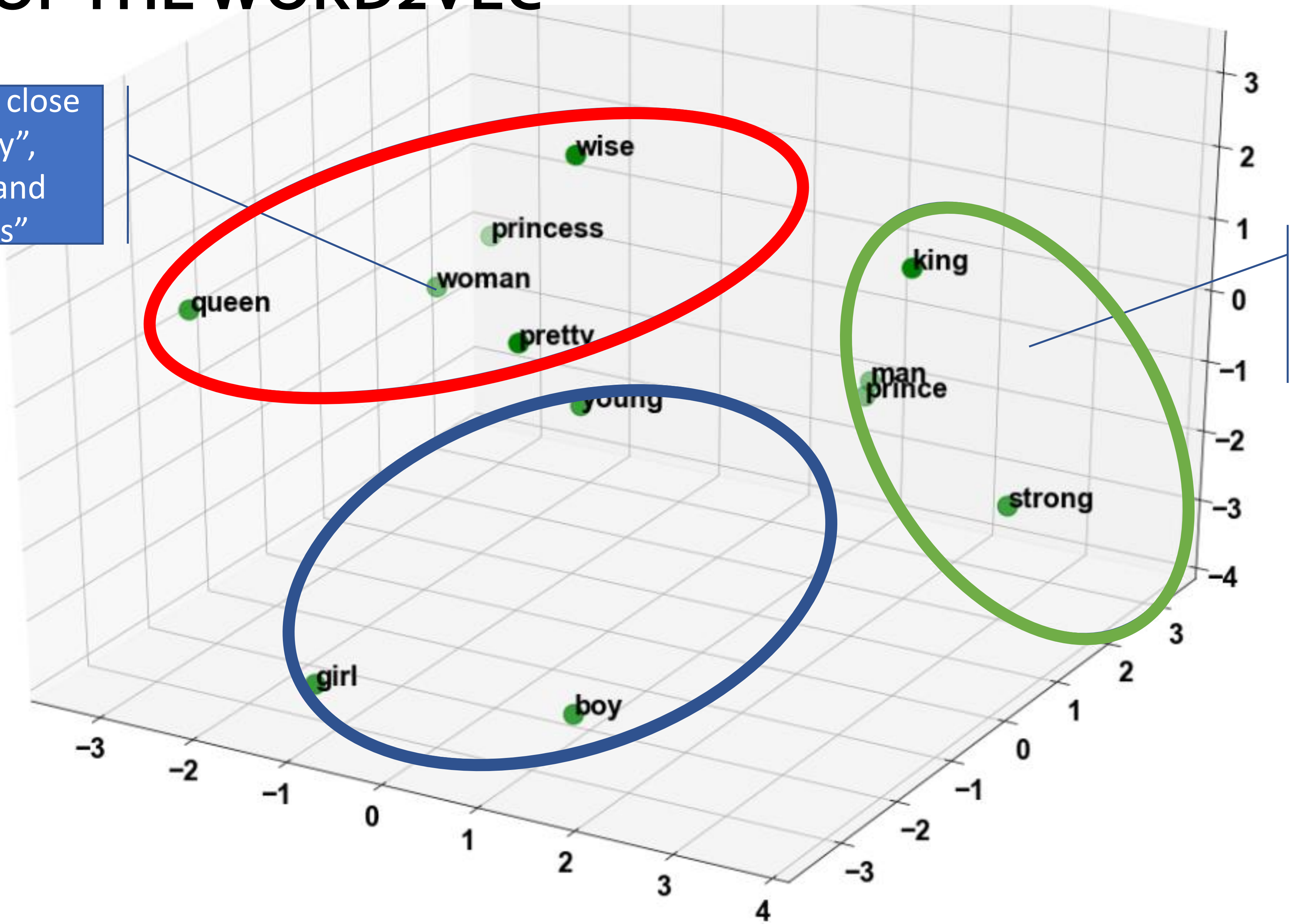
Three different clusters of words

IMPORTANT NOTE



- Since a neural network may have several solutions, you may get a different result. But the patterns will be same
- You will see three different clusters in all results

RESULT OF THE WORD2VEC

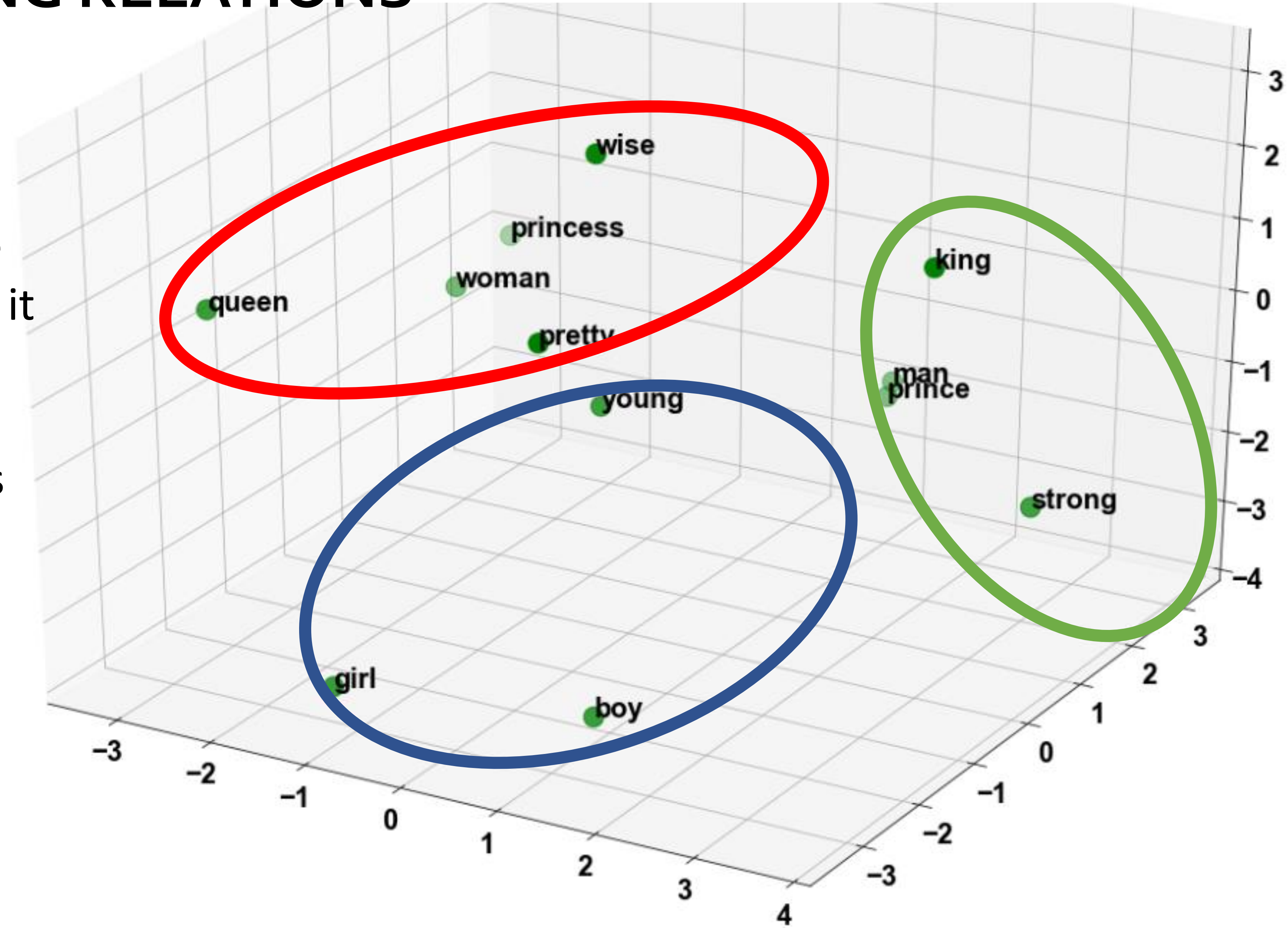


"woman" is close to "pretty", "queen" and "princess"

"King" is near to "man", "strong", "prince"

NOT JUST RETAINING RELATIONS

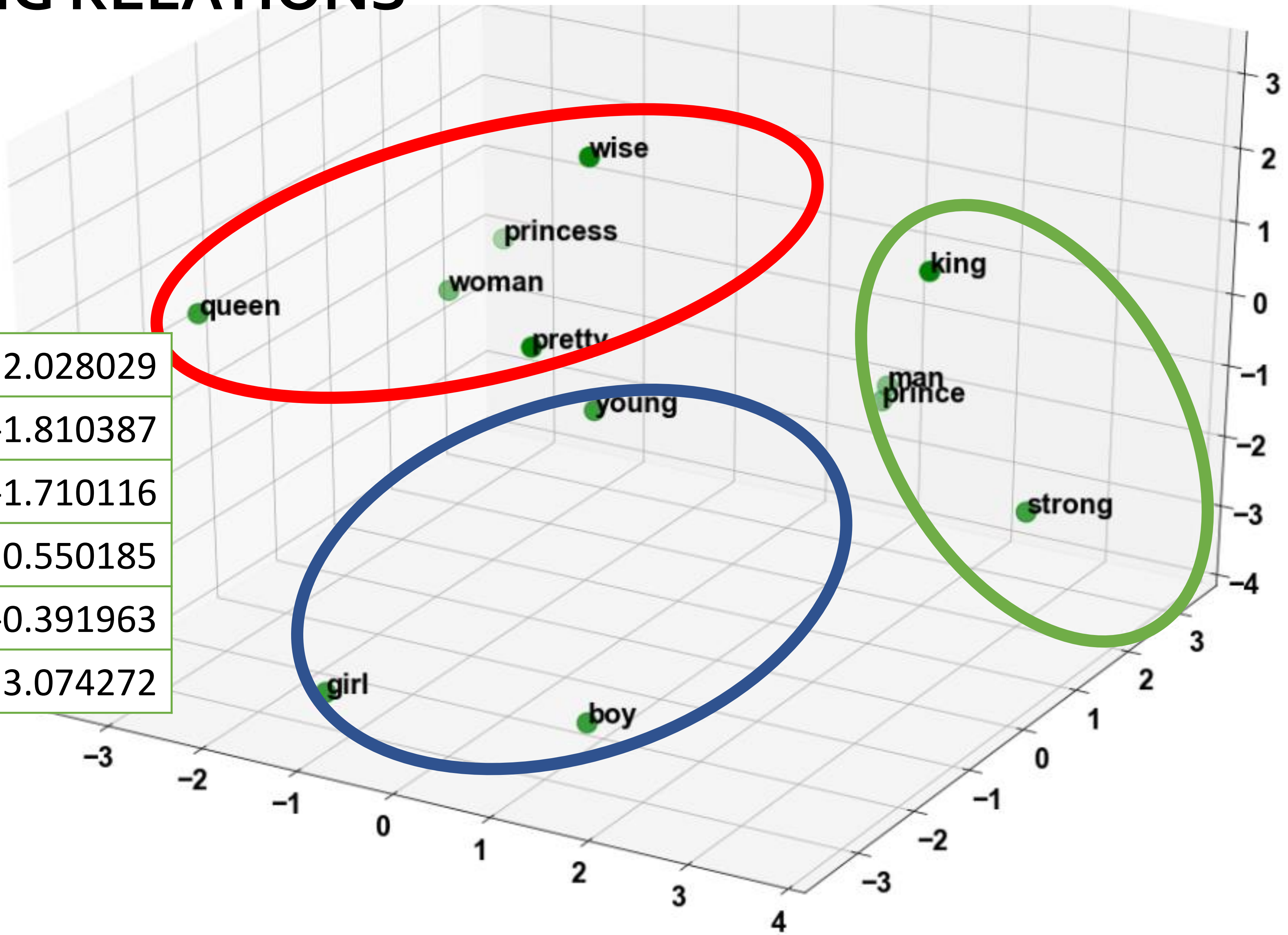
- What is the result of below equation?
- **king – man+ women + wise**
- Which context/cluster does it fall under?
- Red or green or blue?
- Lets us see the result of this operation on our vectors



NOT JUST RETAINING RELATIONS

- What is the result of below equation?
- **king – man+ women + wise**

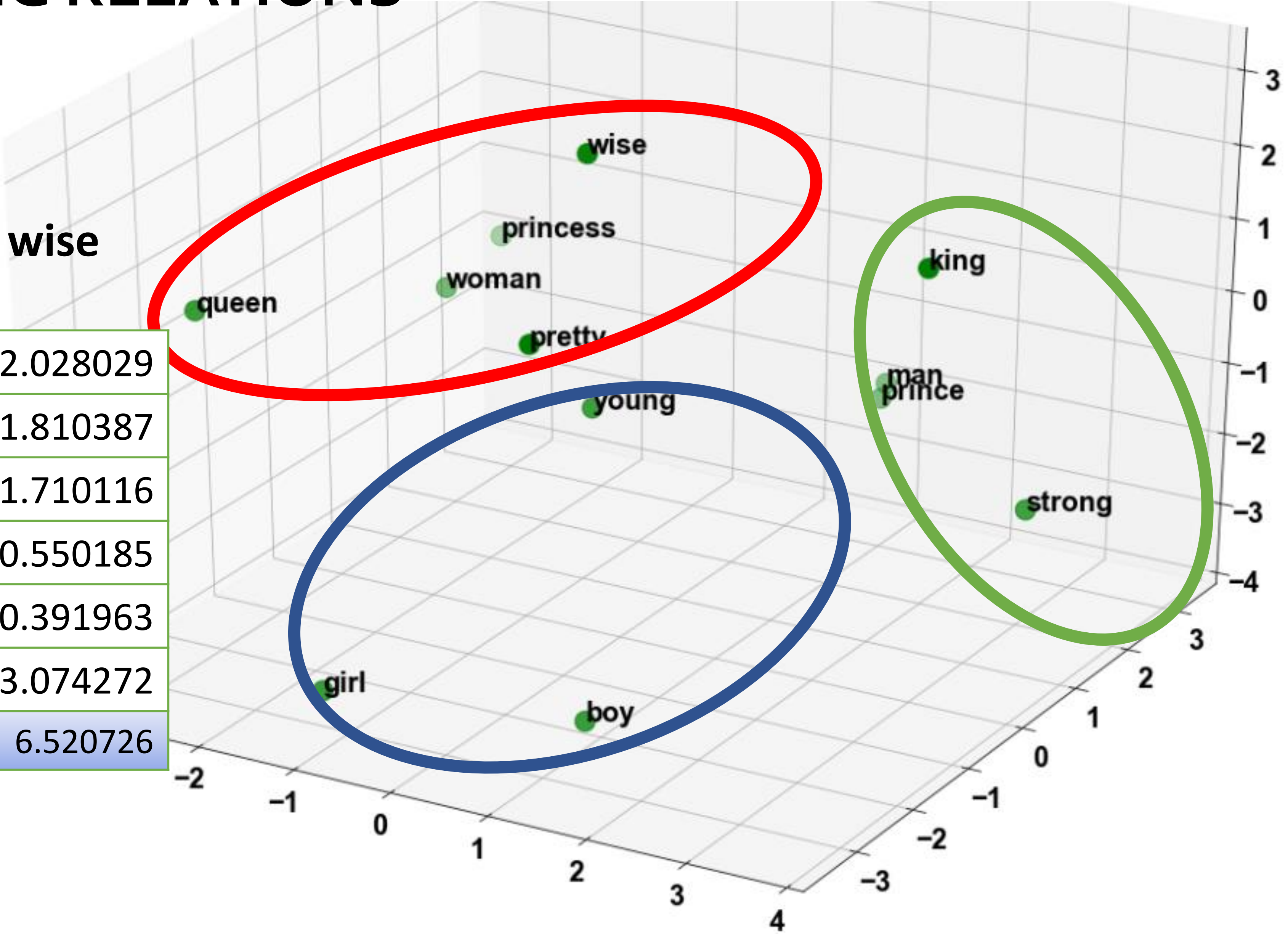
king	3.248315	-0.292609	2.028029
man	1.032173	3.037509	-1.810387
strong	3.659783	0.865091	-1.710116
queen	-3.151272	-2.009174	0.550185
woman	-2.420959	0.980081	-0.391963
Wise	0.273312	-0.993257	3.074272



NOT JUST RETAINING RELATIONS

- What is the result of below equation?
- **new=king – man+ women + wise**

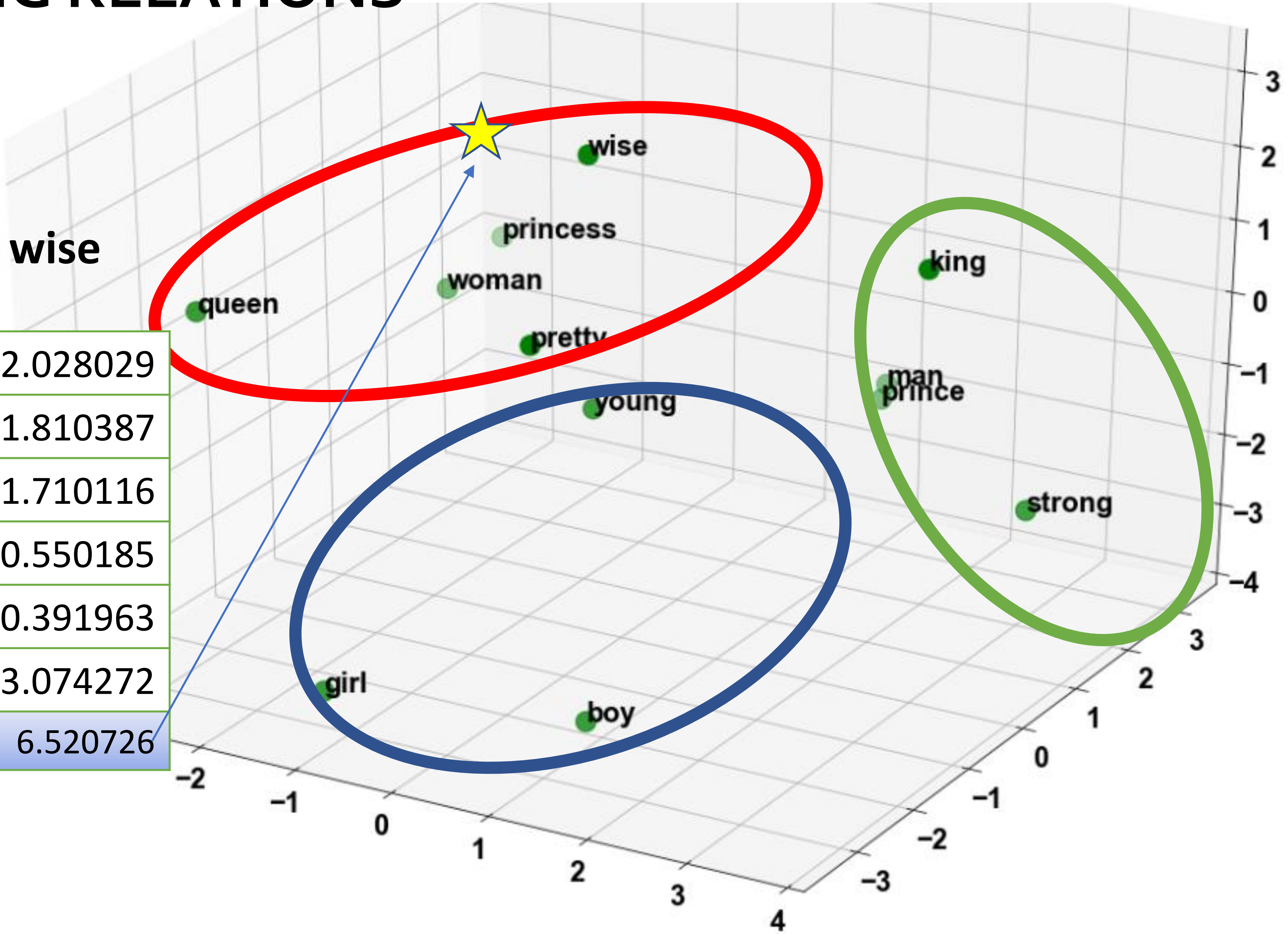
king	3.248315	-0.292609	2.028029
man	1.032173	3.037509	-1.810387
strong	3.659783	0.865091	-1.710116
queen	-3.151272	-2.009174	0.550185
woman	-2.420959	0.980081	-0.391963
Wise	0.273312	-0.993257	3.074272
new	0.06849611	-3.3432946	6.520726



NOT JUST RETAINING RELATIONS

- What is the result of below equation?
- **new=king – man+ women + wise**

king	3.248315	-0.292609	2.028029
man	1.032173	3.037509	-1.810387
strong	3.659783	0.865091	-1.710116
queen	-3.151272	-2.009174	0.550185
woman	-2.420959	0.980081	-0.391963
Wise	0.273312	-0.993257	3.074272
new	0.06849611	-3.3432946	6.520726



LAB: WORD2VEC MODEL BUILDING

```
statements = [
"Trees are tall",
"Trees are green",
"Trees are majestic",
"Trees are essential",
"Trees are diverse",
"Trees are oxygen-giving",
"computers are fast",
"computers are smart",
"computers are useful",
"computers are powerful",
"computers are everywhere",
"computers are changing"
]
```

GENSIM CODE

```
from gensim.models import Word2Vec
model = Word2Vec(documents, min_count=1, vector_size=3, window = 3)
#size:  size of word vector, hidden layer
#min-count: discard words that appear less than # times
#window: Context Window size
```

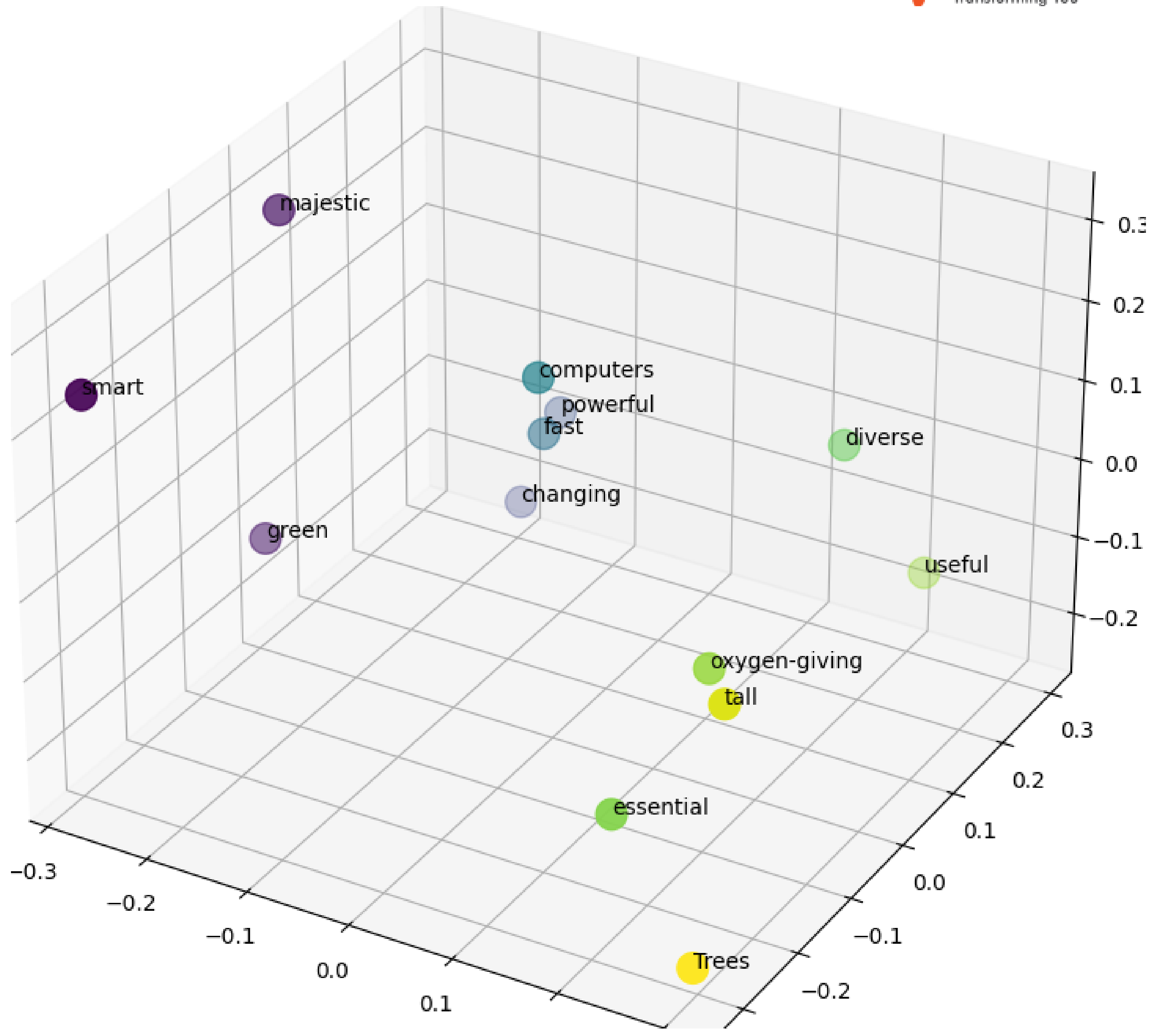
GENSIM RESULTS

```python
for word, vector in zip(model.wv.index_to_key, model.wv.vectors):
    print(word, vector)
```

```
computers [-0.01787424  0.00788105  0.17011166]
Trees [ 0.3003091  -0.31009832 -0.23722696]
useful [ 0.21529575  0.2990996  -0.16718094]
powerful [-0.12544572  0.24601682 -0.05111571]
changing [-0.15122044  0.21846838 -0.16200535]
fast [-0.06053392  0.09588599  0.03306246]
smart [-0.27617383 -0.3149606   0.24372554]
diverse [0.16900873 0.22525644 0.02542885]
oxygen-giving [ 0.21169634 -0.1135122  -0.03154671]
essential [ 0.19228578 -0.25072125 -0.13120346]
majestic [-0.2503861  -0.03100141  0.31793728]
green [-0.24397223 -0.07779229 -0.06459137]
tall [ 0.2692479  -0.19769652  0.00150541]
```

WORD EMBEDDINGS

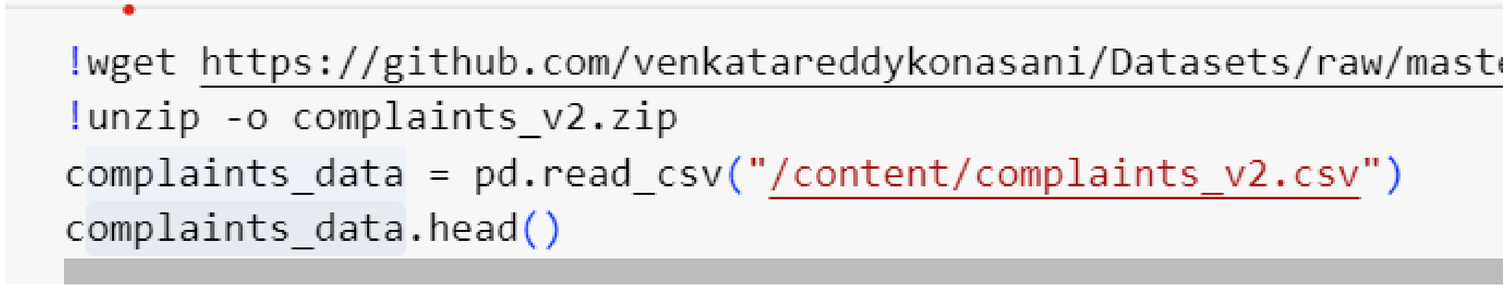
```
"Trees  tall",
"Trees  green",
"Trees  majestic",
"Trees  essential",
"Trees  diverse",
"Trees  oxygen-giving",
"computers  fast",
"computers  smart",
"computers  useful",
"computers  powerful",
"computers  everywhere",
"computers  changing"
```



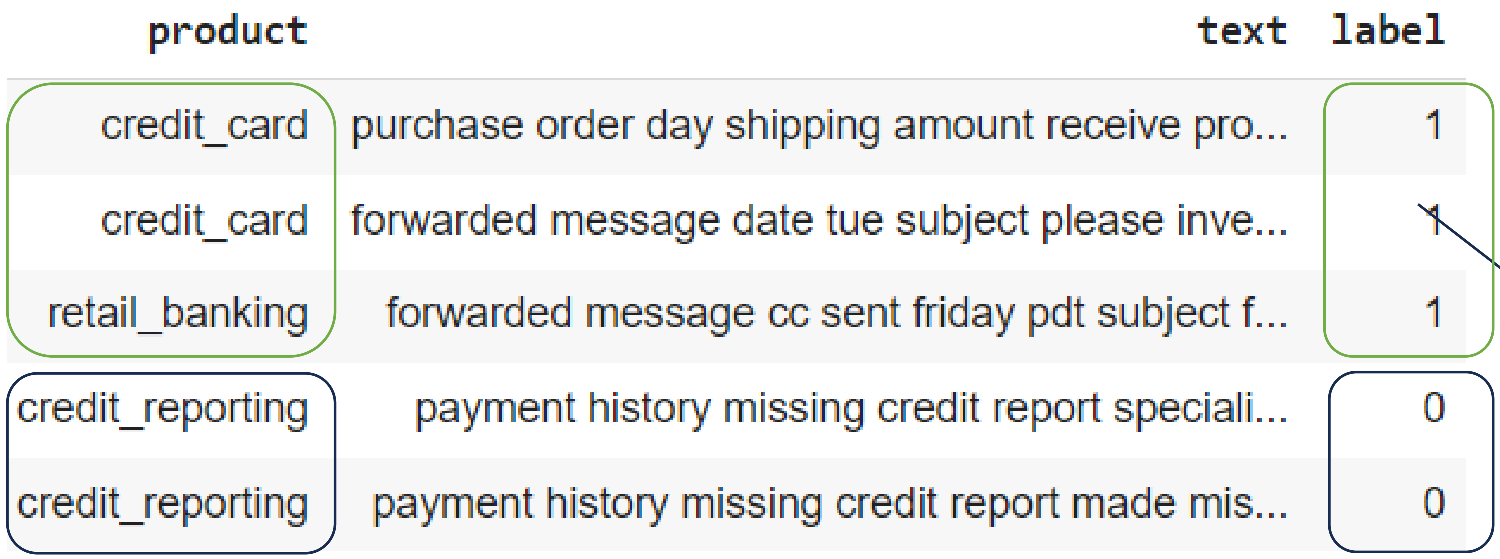
THE PROBLEM STATEMENT AND DATASET

- https://www.consumerfinance.gov/
- The Consumer Financial Protection Bureau (CFPB) acts as a mediator between financial institutions and consumers, facilitating dispute resolution when complaints arise.
- To improve efficiency and accuracy in handling customer complaints, they would like to automatically classify and route complaints to the appropriate teams based on their content and associated financial products.
- https://www.kaggle.com/datasets/adhamelkomy/bank-customer-complaint-analysis/data

ABOUT THE DATASET



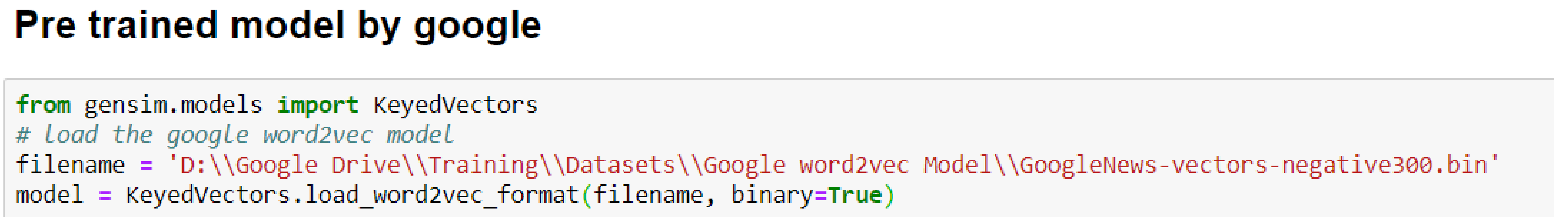
```
!wget https://github.com/venkatareddykonasani/Datasets/raw/maste
!unzip -o complaints_v2.zip
complaints_data = pd.read_csv("/content/complaints_v2.csv")
complaints_data.head()
```



product	text	label
credit_card	purchase order day shipping amount receive pro...	1
credit_card	forwarded message date tue subject please inve...	1
retail_banking	forwarded message cc sent friday pdt subject f...	1
credit_reporting	payment history missing credit report speciali...	0
credit_reporting	payment history missing credit report made mis...	0

- Bank customers complaints data
- Customer send email complaints for various products
- we are broadly classifying everything as “credit reporting” and “other complaints “

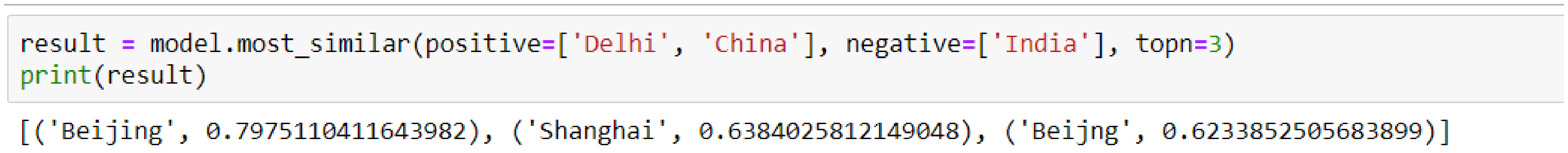
PRE-TRAINED MODELS BY GOOGLE



Pre trained model by google

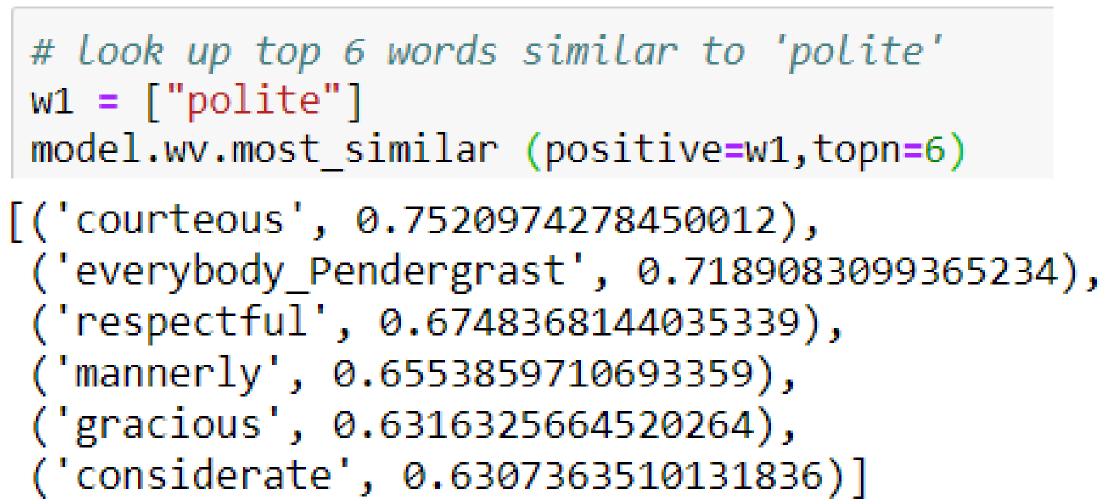
```
from gensim.models import KeyedVectors
# load the google word2vec model
filename = 'D:\\Google Drive\\Training\\Datasets\\Google word2vec Model\\GoogleNews-vectors-negative300.bin'
model = KeyedVectors.load_word2vec_format(filename, binary=True)
```

RESULTS OF GOOGLE MODEL



```
result = model.most_similar(positive=['Delhi', 'China'], negative=['India'], topn=3)
print(result)
```

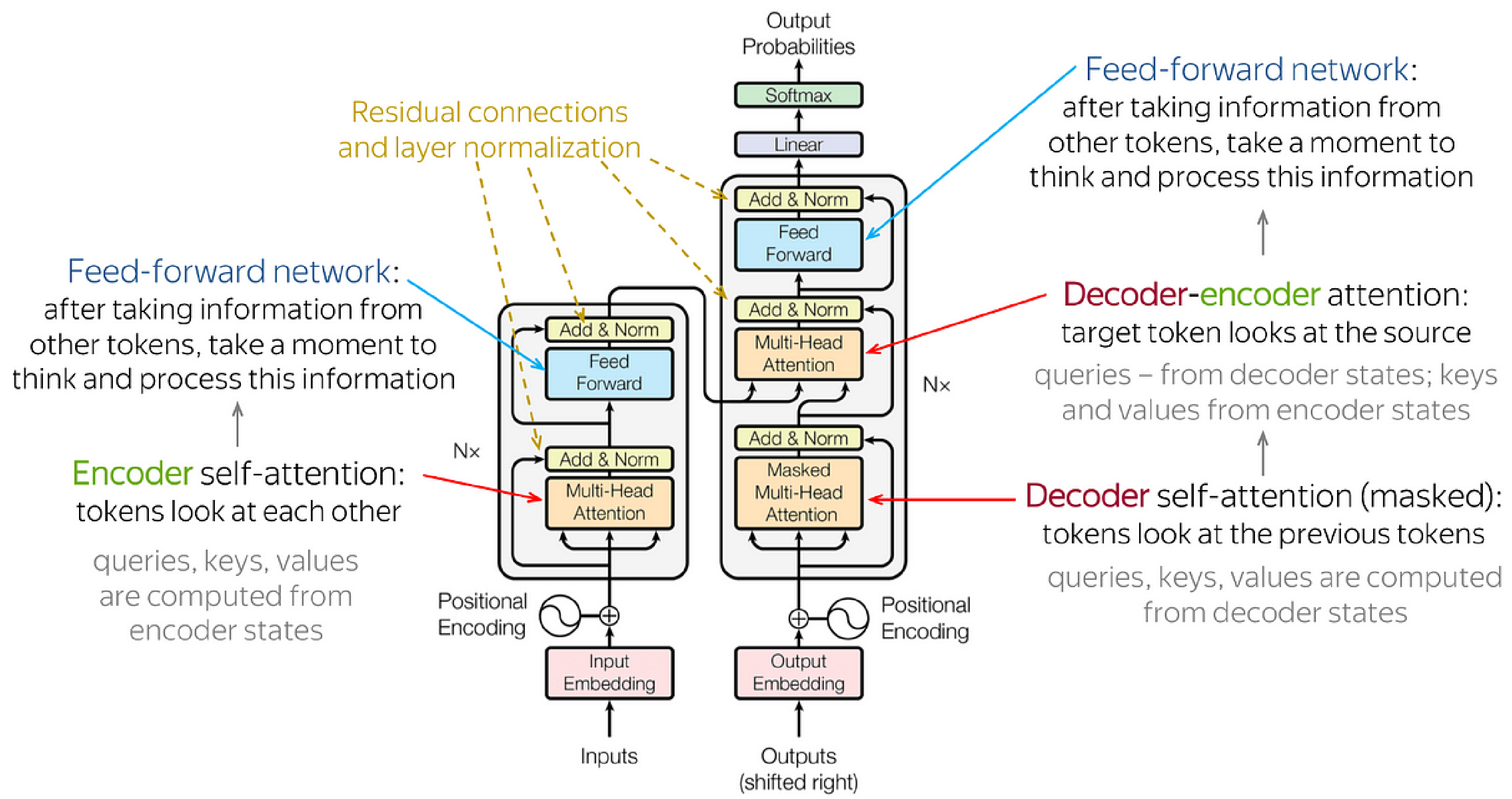
```
[('Beijing', 0.7975110411643982), ('Shanghai', 0.6384025812149048), ('Beijng', 0.6233852505683899)]
```



```
# look up top 6 words similar to 'polite'
w1 = ["polite"]
model.wv.most_similar (positive=w1,topn=6)
```

```
[('courteous', 0.7520974278450012),
 ('everybody_Pendergrast', 0.7189083099365234),
 ('respectful', 0.6748368144035339),
 ('mannerly', 0.6553859710693359),
 ('gracious', 0.6316325664520264),
 ('considerate', 0.6307363510131836)]
```

THE FUTURE – TRANSFORMERS & LLMS



Residual connections and layer normalization
Feed-forward network: after taking information from other tokens, take a moment to think and process this information
Encoder self-attention: tokens look at each other
queries, keys, values are computed from encoder states
Feed-forward network: after taking information from other tokens, take a moment to think and process this information
Decoder-encoder attention: target token looks at the source
queries – from decoder states; keys and values from encoder states
Decoder self-attention (masked): tokens look at the previous tokens
queries, keys, values are computed from decoder states
Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

THANK YOU