Dog Toy Recommender System

- BY QI SUN

CAPSTONE PROJECT FALL 2020

INSTRUCTOR: ANDY CATLIN

Introduction

The purpose of this project is to build a personalized dog toy recommender system for Chewy.com

Data source01: Scraped from Chewy.com
Size: Chewy data 2729 records

Column: 8

Column names: title, price, rating, link, category, sub-category, description, key benefits

Data source02: data from a survey that I created for this project

Survey:

- Survey tool: Qualtrics
- Survey question: How likely would your dog rank the following items? (5 being favorite, and 1 least)
- Toy numbers: 10
- Survey link: https://yeshiva.co1.qualtrics.com/jfe/form/SV_3wOuB8VPL8SQgYJ
- Total response: 21
- Export: csv

Survey Results:

#	Field	5		4		3		2		1		Not Sure		Total
1	KONG Dental Stick Dog Toy	14.29%	3	23.81%	5	14.29%	3	9.52%	2	23.81%	5	14.29%	3	21
2	Pet Qwerks Talking Babble Ball Dog Toy	28.57%	6	19.05%	4	19.05%	4	19.05%	4	9.52%	2	4.76%	1	21
3	JW Pet Whirlwheel Flying Disk Dog Toy	30.00%	6	20.00%	4	25.00%	5	15.00%	3	10.00%	2	0.00%	0	20
4	KONG Puppy Flyer Dog Toy	33.33%	7	9.52%	2	23.81%	5	28.57%	6	4.76%	1	0.00%	0	21
5	JW Pet iSqueak Funble Football Dog Toy	14.29%	3	28.57%	6	14.29%	3	23.81%	5	9.52%	2	9.52%	2	21
6	JW Pet iSqueak Bouncin' Baseball Dog Toy	14.29%	3	23.81%	5	33.33%	7	14.29%	3	4.76%	1	9.52%	2	21
7	ZippyPaws Hedgehog Plush Dog Toy	28.57%	6	19.05%	4	14.29%	3	14.29%	3	19.05%	4	4.76%	1	21
8	KONG Wubba Floppy Ears Dog Toy	33.33%	7	19.05%	4	14.29%	3	19.05%	4	9.52%	2	4.76%	1	21
9	JW Pet Treat Tower Treat Dispensing Dog Toy	47.62%	10	23.81%	5	19.05%	4	4.76%	1	0.00%	0	4.76%	1	21
10	KONG Wild Knots Eagle Dog Toy	42.86%	9	19.05%	4	19.05%	4	9.52%	2	4.76%	1	4.76%	1	21

Top 3 product rating for each user

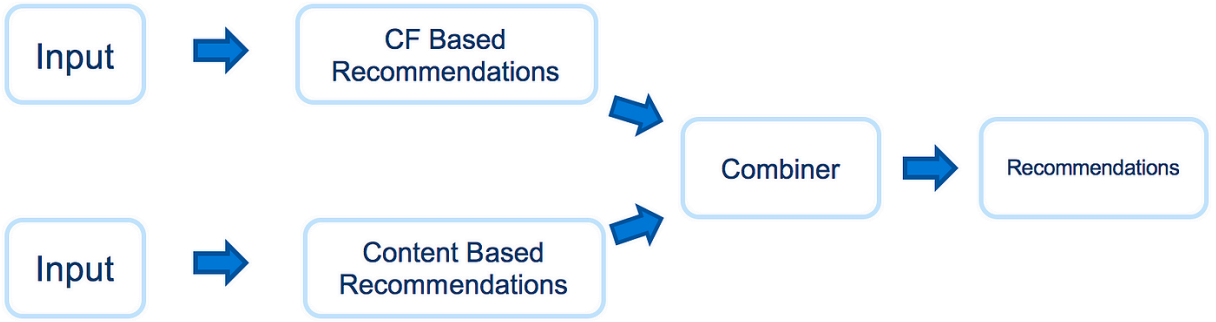
	userID	1st Max	Max1Value	2nd Max	Max2Value	3rd Max	Max3Value
0	1	347	5.0	354	5.0	688	4.0
1	2	347	5.0	354	5.0	997	5.0
2	3	688	5.0	347	4.0	416	4.0
3	4	688	5.0	1450	5.0	997	5.0
4	5	1450	5.0	416	4.0	354	4.0
5	6	347	5.0	1450	5.0	354	5.0
6	7	688	5.0	673	5.0	997	5.0
7	8	1303	5.0	189	4.0	393	4.0
8	9	189	5.0	354	5.0	416	4.0
9	10	347	5.0	673	5.0	393	5.0
10	11	688	5.0	354	5.0	1303	5.0
11	12	1303	5.0	688	4.0	416	4.0
12	13	347	5.0	673	5.0	997	5.0
13	14	347	5.0	688	5.0	673	5.0
14	15	673	5.0	1303	5.0	347	4.0
15	16	354	5.0	997	5.0	1303	5.0
16	17	673	5.0	393	5.0	416	5.0
17	18	189	5.0	673	5.0	416	5.0
18	19	393	5.0	1450	5.0	997	5.0
19	20	189	5.0	416	5.0	1450	5.0
20	21	688	5.0	1450	5.0	997	5.0

Average rating for each product

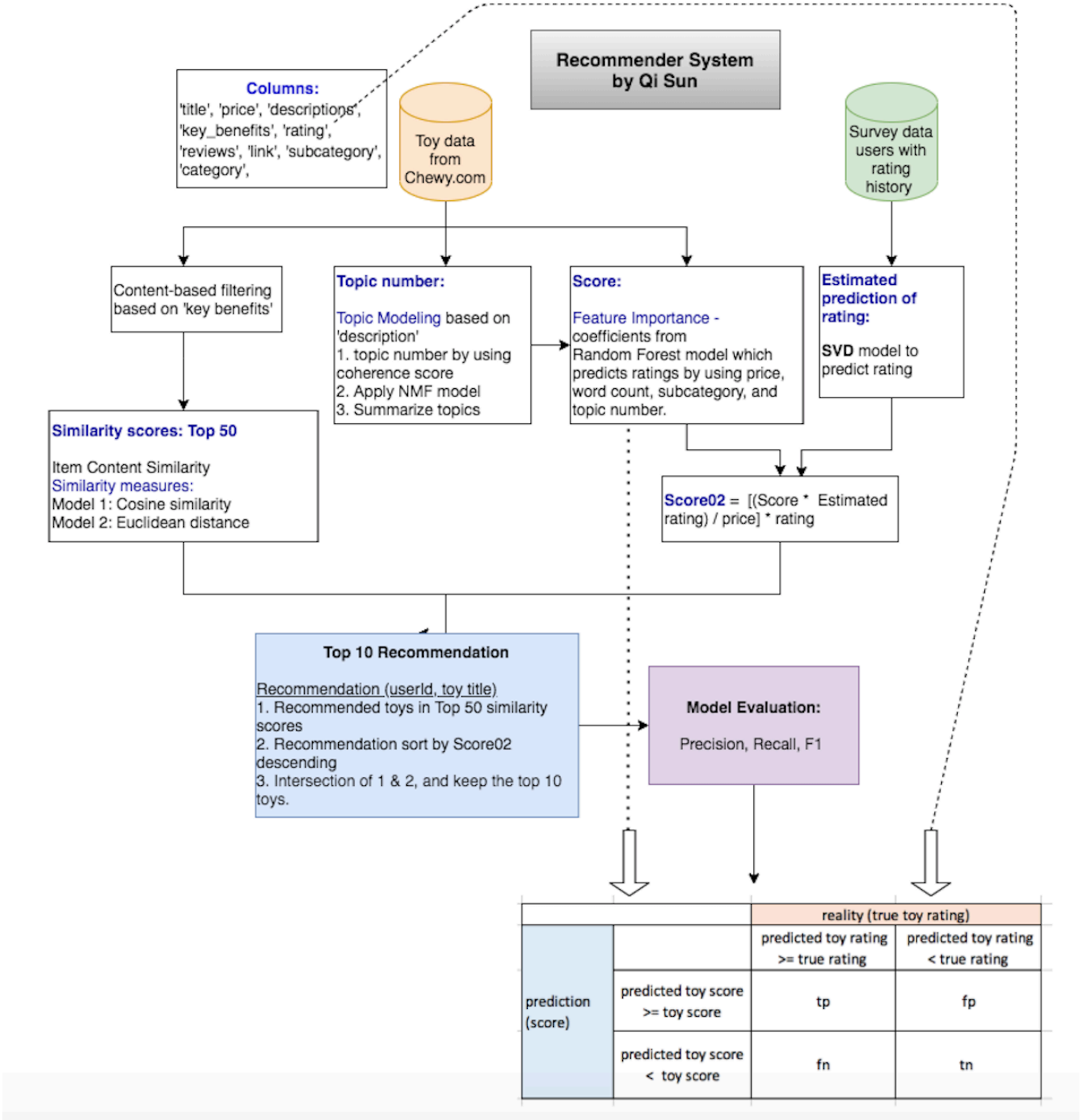
	productID	rating_mean
8	997	4.000000
9	1303	3.714286
3	673	3.380952
7	354	3.333333
2	688	3.285714
1	347	3.238095
6	1450	3.095238
5	416	3.000000
4	393	2.857143
0	189	2.523810

Method:

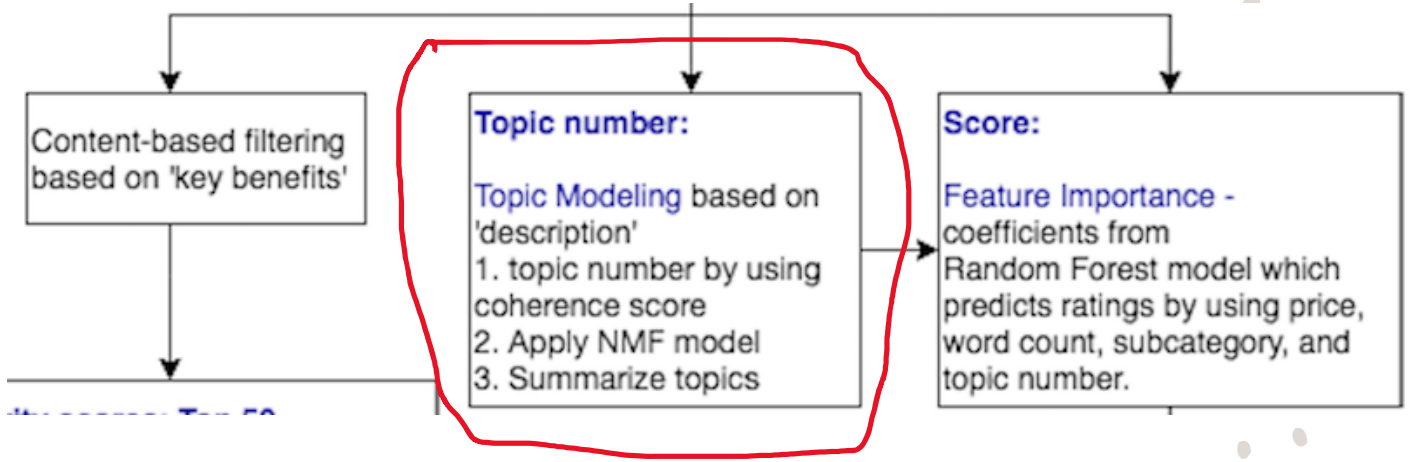
- Create a personalized hybrid recommender system
- Combine results of Content based and Collaborative Recommender Systems to get great results
- Contribution - differences between my project and all other existing recommender systems: Build topic modeling, calculate feature importance for numerical variables and set up a new model evaluation to create a more reliable recommender system
- Stakeholders: this model can be used by start-up companies with few users rating data



Architectures



Topic Modeling:

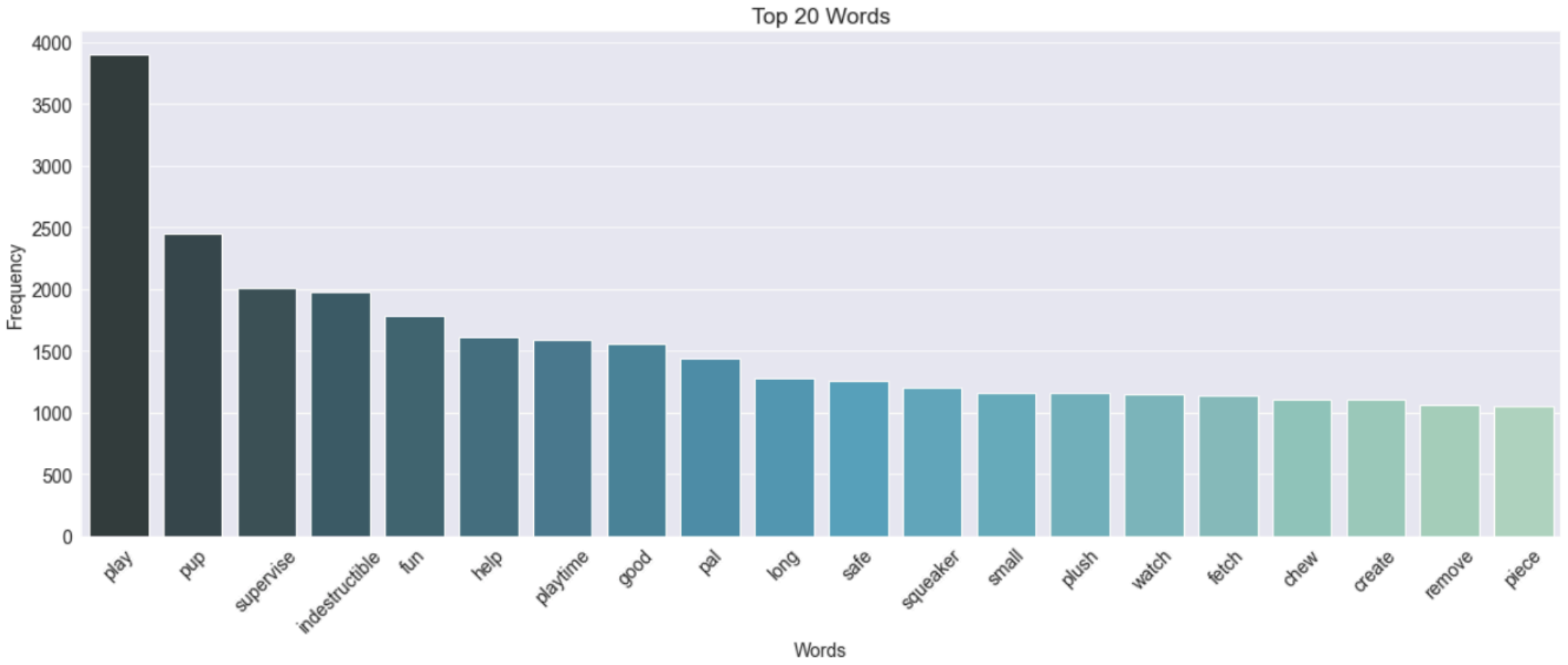


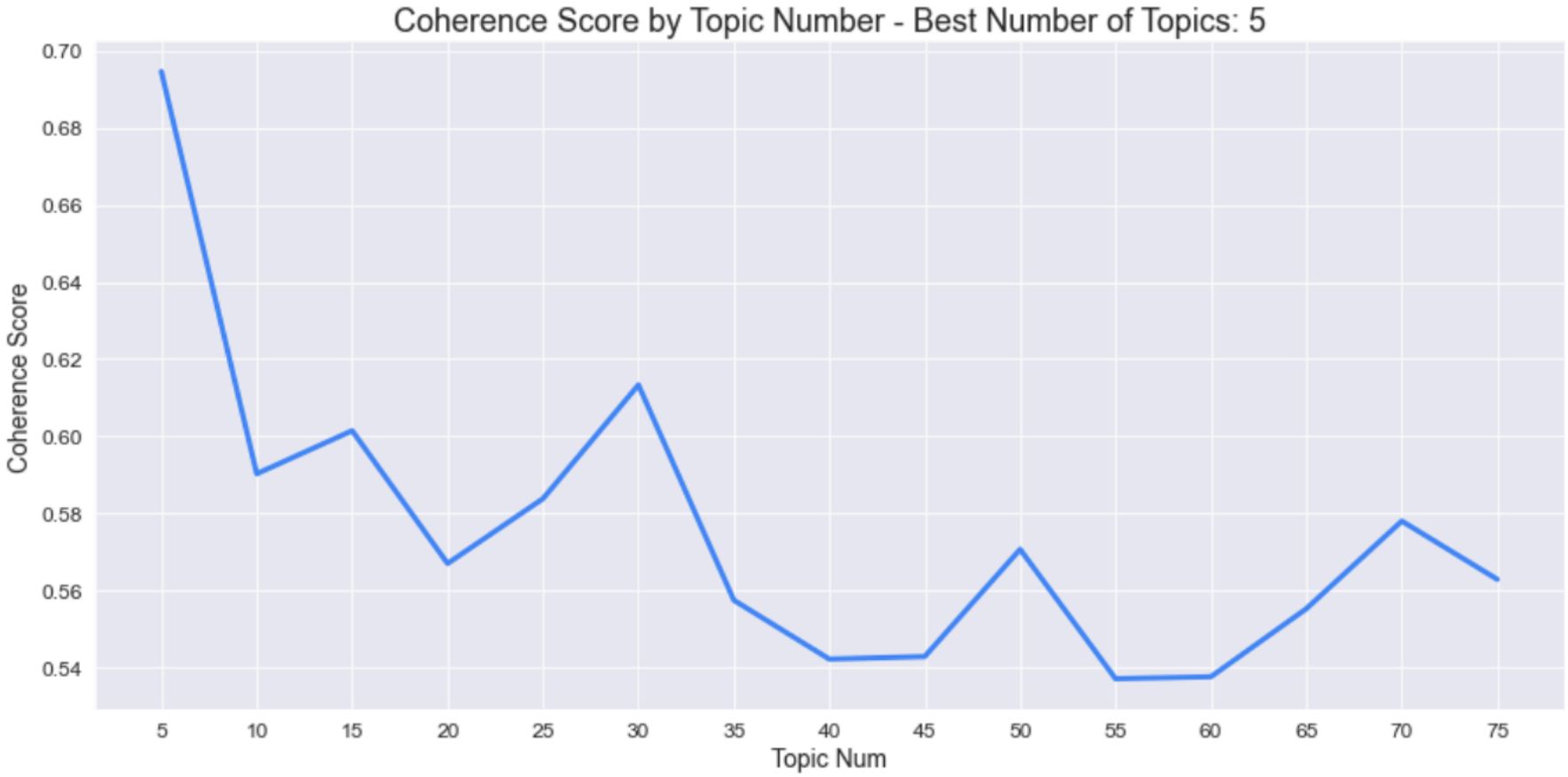
Apply a NMF model to the toy **description** and determine the number of topics and theme of each topic.

Here are the steps:

- Prepare the text (description) for topic modeling
- Find the best number of topics automatically
- Apply a NMF model
- Extract topics and Summarize those topics
- Find the highest quality topics among all the topics

bar chart for the top 20 most frequently occurring words




Coherence Score by Topic Number - Best Number of Topics: 5
Coherence Score
0.70
0.68
0.66
0.64
0.62
0.60
0.58
0.56
0.54
5
10
15
20
25
30
35
40
45
50
55
60
65
70
75
Topic Num

Results of the NMF model

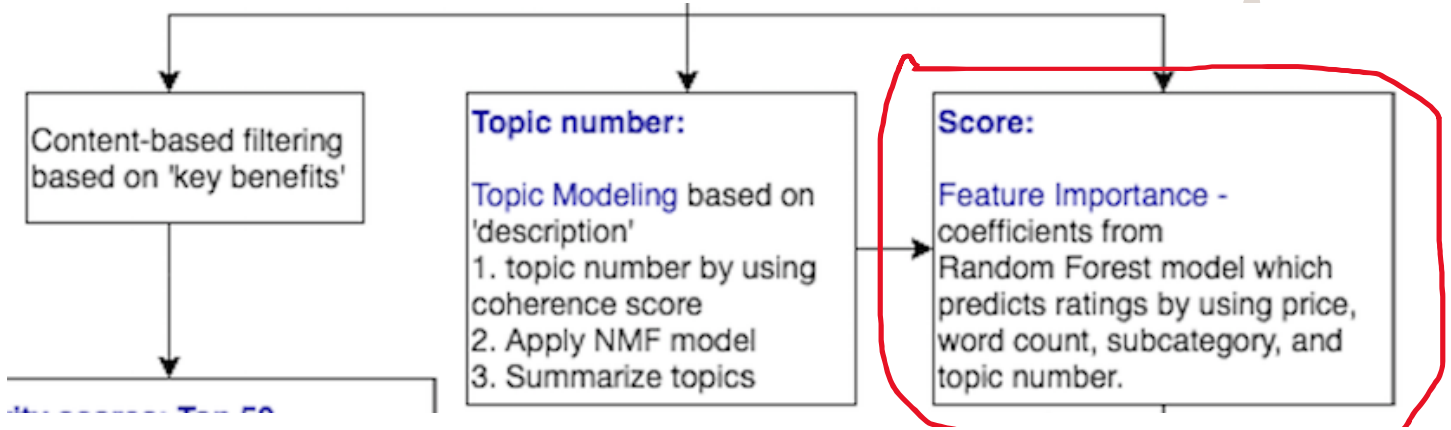
	0	1	2	3	4	5	6	7	8	9	topics
0	play	pal	long	safe	create	equal	truly	importantly	case	break	play pal long safe create equal truly importan...
1	internal	plush	damage	small	child	reach	blockage	discontinue	present	hazard	internal plush damage small child reach blocka...
2	tuffy	layer	washable	machine	sew	float	outdoor	indoor	protective	pouch	tuffy layer washable machine sew float outdoor...
3	ball	tennis	rubber	fetch	launcher	throw	bounce	durable	design	game	ball tennis rubber fetch launcher throw bounce...
4	treat	chew	flavor	bone	tooth	starmark	puppy	clean	dental	busy	treat chew flavor bone tooth starmark puppy cl...

From the results of top 10 words in each topic above, I summarized the topic manually below:

- The first topic is about toy with hign safty.
- The second topic is sbout toy of small size.
- The third topic is about tuffy and washable toy.
- The fourth topic is about fetch toy and toy shape.
- The fifth topic is about chew toy for puppy and dental cleaning.

Feature Importance

- There are some numerical variables in this dataset that I would like to calculate the weight of each numerical variable.
- **Random Forest** model to predict rating




```
imp_df(X.columns, rf.feature_importances_)
```

executed in 44ms, finished 20:48:13 2020-12-16

Feature	Importance
word_count	0.445138
price	0.370455
subcat02	0.098533
topic_num	0.085875

Findings:

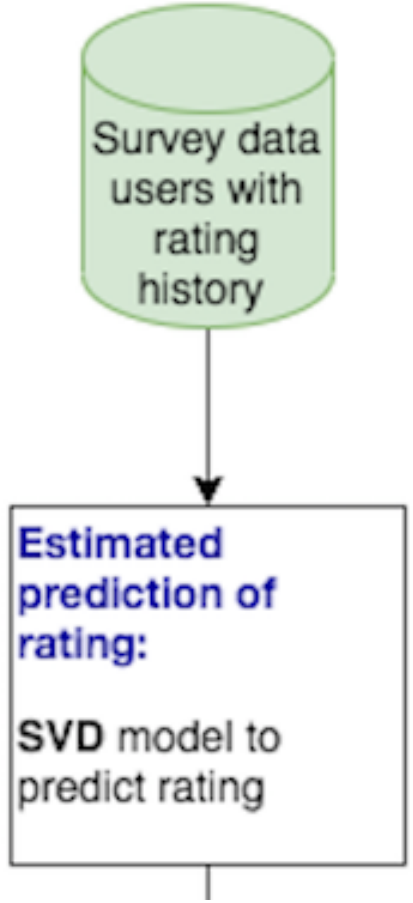
```
# calculate scores

df03['score'] = (
    0.370455 * df03['price'] + 0.445138 * df03['word_count'] +
    0.085875 * df03['topic_num'] + 0.098533 * df03['subcat02']
)
```

1. The variable og word_count has the most importance, which is the total number of words in a product key benefits. That means the more detailed benefits listed on the product webpage, the higher ratings that users gave.
2. Also, the price of a product is important to predict the ratings, which means dog toys with higher prices could make customers more satisfied. In general, the toy with a higher price should have a better quality and material, and much safer for dog.
3. The category and topic have least importance to predict the rating, which makes sense because a toy that a dog likes to play is not affected by the toy's category and topic.

SVD based Collaborative filtering

- **Surprise library**
- Surprise is a Python Scikit building and analyzing recommender systems that deal with explicit rating data.

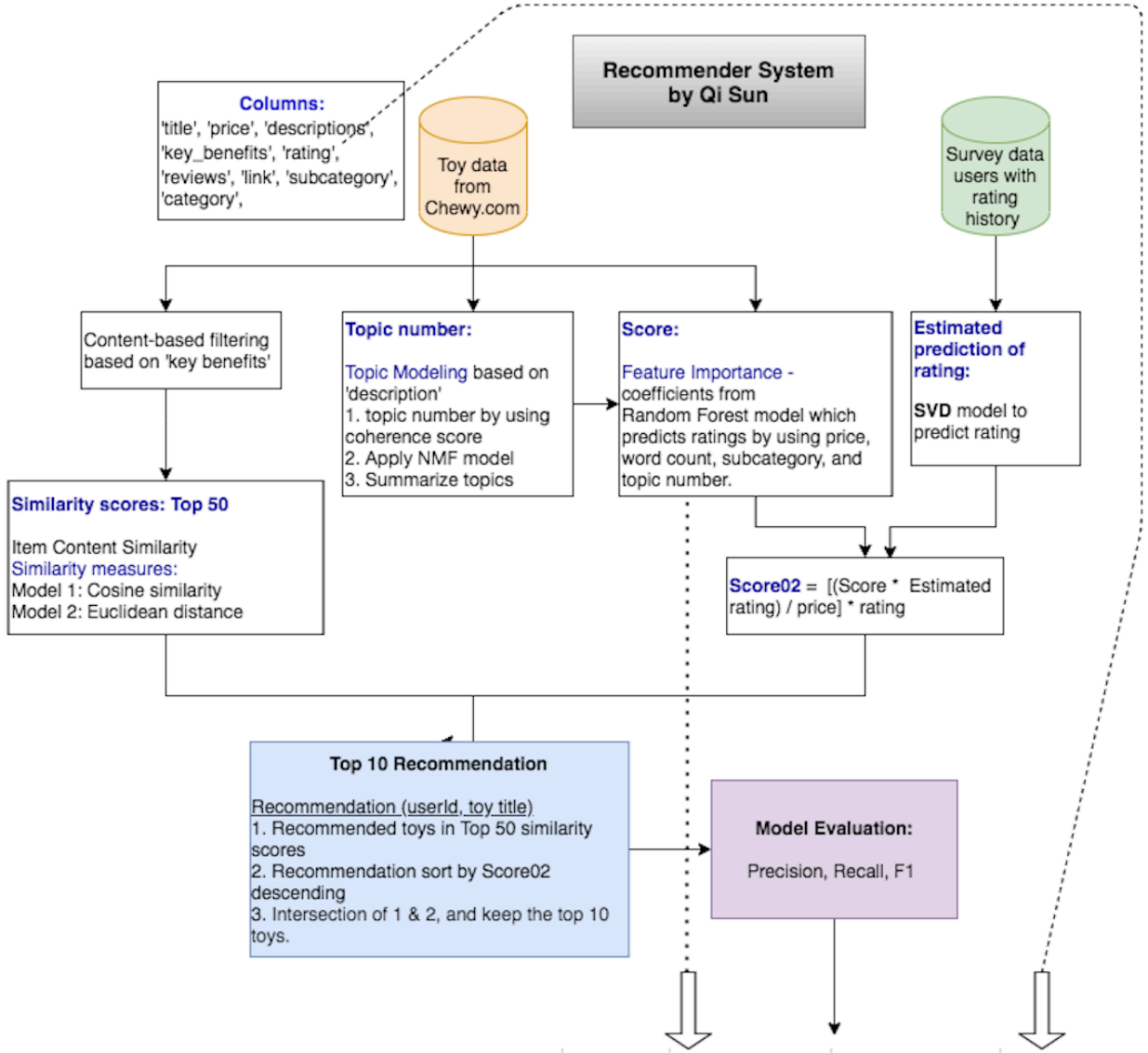


Example: to predict userID= 1 and item ID=100

```
svd.predict(1, 100)
```

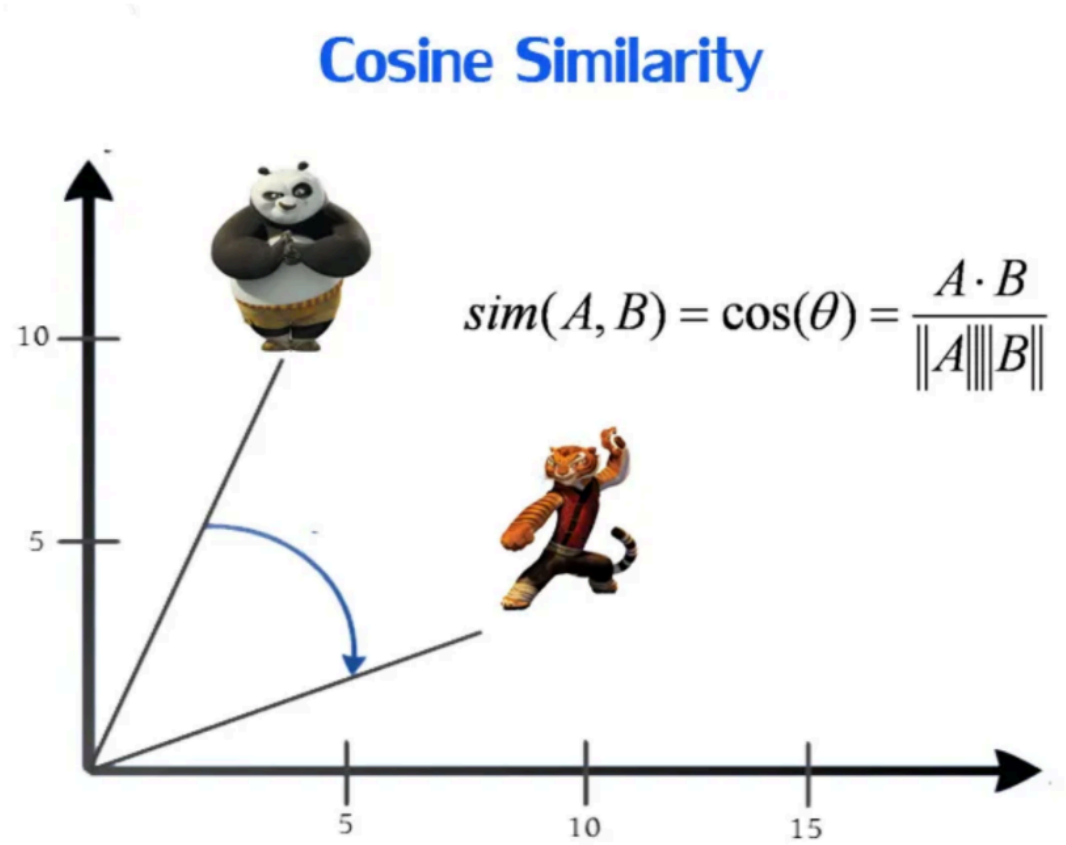
executed in 39ms, finished 22:12:58 2020-12-16

```
Prediction(uid=1, iid=100, r_ui=None, est=3.675533153691794, details={'was_impossible': Fals
e})
```


Recommender System by Qi Sun
Columns:
'title', 'price', 'descriptions', 'key_benefits', 'rating', 'reviews', 'link', 'subcategory', 'category',
Toy data from Chewy.com
Survey data users with rating history
Content-based filtering based on 'key benefits'
Topic number:
Topic Modeling based on 'description'
1. topic number by using coherence score
2. Apply NMF model
3. Summarize topics
Score:
Feature Importance - coefficients from Random Forest model which predicts ratings by using price, word count, subcategory, and topic number.
Estimated prediction of rating:
SVD model to predict rating
Similarity scores: Top 50
Item Content Similarity
Similarity measures:
Model 1: Cosine similarity
Model 2: Euclidean distance
Score02 = [(Score * Estimated rating) / price] * rating
Top 10 Recommendation
Recommendation (userId, toy title)
1. Recommended toys in Top 50 similarity scores
2. Recommendation sort by Score02 descending
3. Intersection of 1 & 2, and keep the top 10 toys.
Model Evaluation:
Precision, Recall, F1
reality (true toy rating)
predicted toy rating >= true rating
predicted toy rating < true rating
prediction (score)
predicted toy score >= toy score
tp
fp
predicted toy score < toy score
fn
tn

Hybrid recommendation system

- **Model 1: Hybrid recommendation system using Cosine Similarity**
- The cosine similarity metric finds the normalized dot product of the two attributes. By determining the cosine similarity, we would effectively try to find the cosine of the angle between the two objects. The cosine of 0° is 1, and it is less than 1 for any other angle.

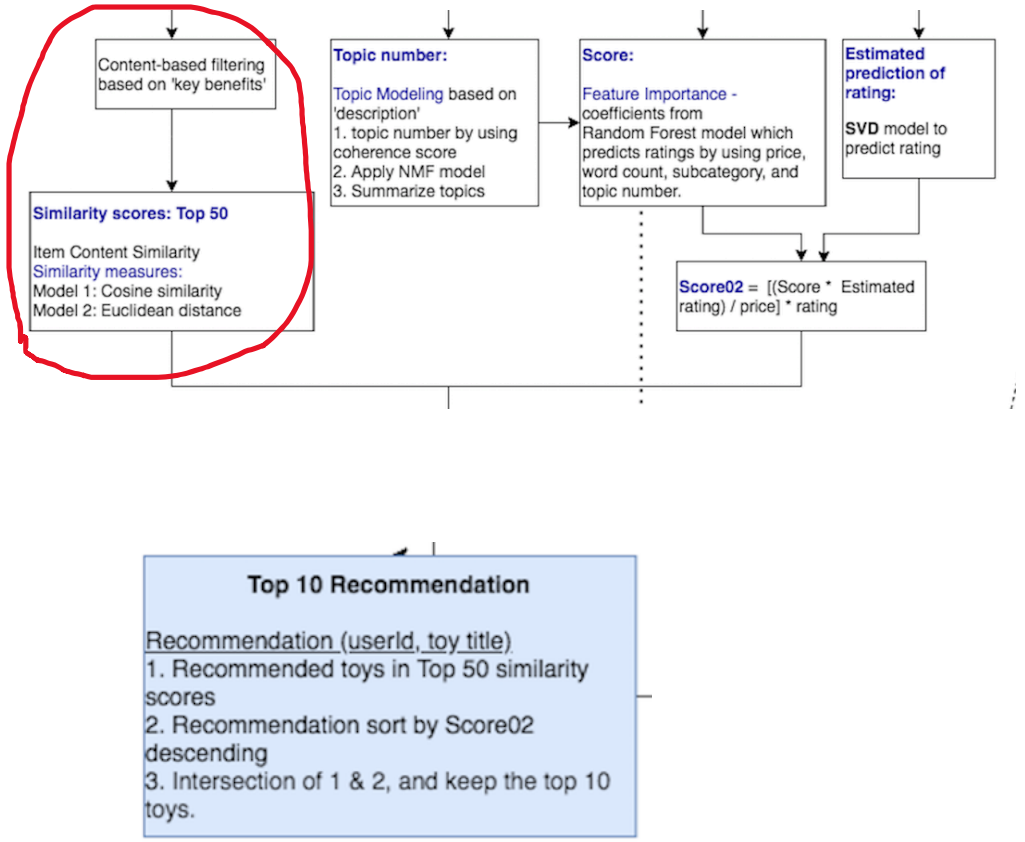


```python
def recommendations_cosine(userId, title):
    #print(df05.loc[df05['title']==title])
    #print('**' * 40)
    # finding cosine similarity for the vectors
    cosine_similarities = cosine_similarity(tfidf_matrix, tfidf_matrix)
    #Reverse mapping of the index
    indices = pd.Series(df04.index, index = df04['title']).drop_duplicates()

    idx = indices[title]
    sim_scores = list(enumerate(cosine_similarities[idx]))
    sim_scores = sorted(sim_scores, key = lambda x: x[1], reverse = True)
    sim_scores = sim_scores[1:51]
    toy_indices = [i[0] for i in sim_scores]
    toys02 = df04.iloc[toy_indices][['title', 'title02','rating', 'price','subcat','cat', 'top

    toys02['est'] = toys02['index'].apply(lambda x: svd.predict(userId, toys02.loc[x]['index']
    toys02['score02'] = ((toys02['est'] * toys02['score'])/toys02['price'])*toys02['rating']
    toys02 = toys02.sort_values('score02', ascending=False)

    cond1 = (df04.index.isin(toy_indices))
    index02 = df04[df04['title']==title]['index']
    #cond2 = (df04.topic_num.isin(df04.iloc[index02]['topic_num']))
    cond3 = (df04.rating > 0)
    recommend = toys02.loc[cond1 & cond3].sort_values(by='score02', ascending=False)
    toys_recommend = recommend[['title', 'index','rating', 'price','topic_num','score','link',
    return toys_recommend.head(10)
```



Model 1: Hybrid recommendation system using Cosine Similarity

Results:
Model 1: Hybrid recommendation system using Cosine Similarity

```
recommendations_cosine_output(1, 'KONG Tuggz Sloth Dog Toy')
```

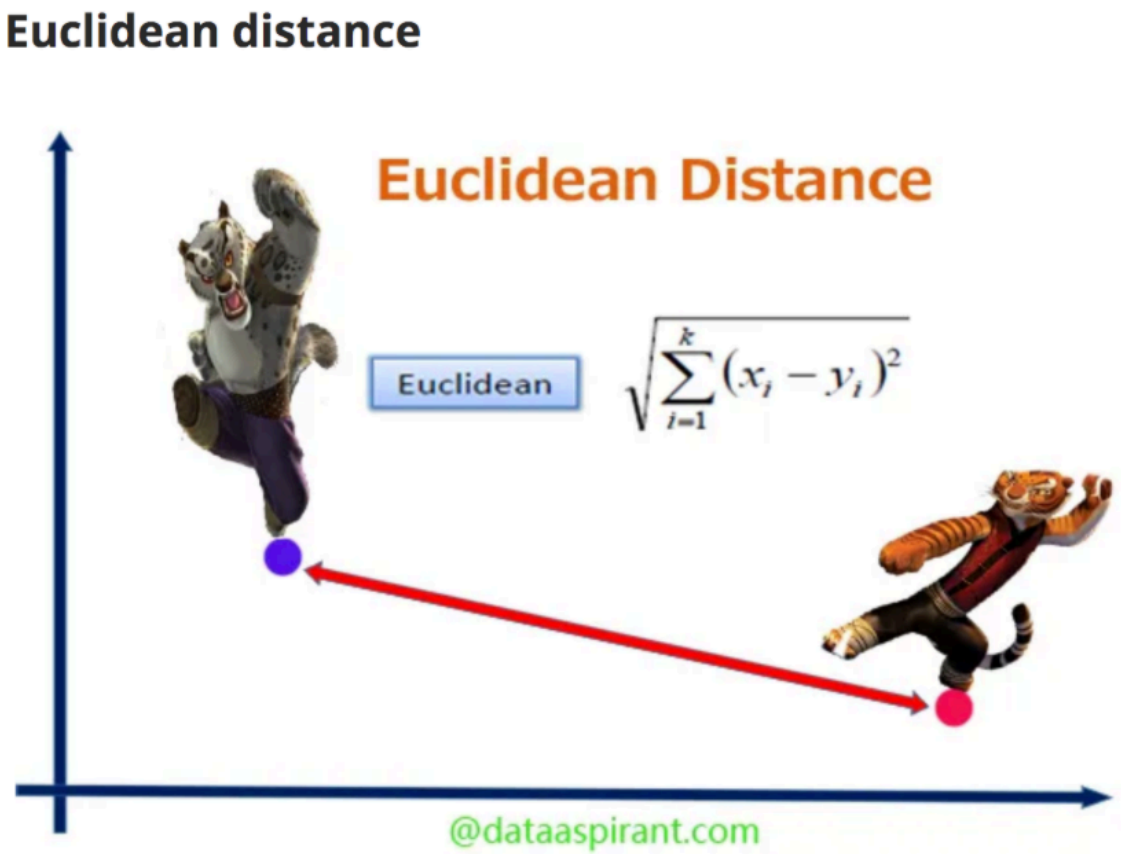
executed in 391ms, finished 02:46:05 2020-12-17

```
      index                        title      score  topic_num  price  rating
1447   1447  KONG Tuggz Sloth Dog Toy  79.256675          1  15.99     3.0
**********************************************************************************
```

	title	index	rating	price	topic_num	score	link	score02
1349	Charming Pet Animates Pig Squeaky Plush Dog Toy	1349	5.0	7.87	1	76.693719	https://www.chewy.com/charming-pet-animates-pi...	179.091681
1184	Frisco Wagazoo Plush Squeaking Elephant Dog Toy	1184	4.5	10.98	1	95.206216	https://www.chewy.com/frisco-wagazoo-plush-squ...	143.415411
1155	Frisco Wagazoo Plush Squeaking Giraffe Dog Toy	1155	4.5	10.98	1	95.206216	https://www.chewy.com/frisco-wagazoo-plush-squ...	143.415411
1493	Charming Pet Animates Lamb Squeaky Plush Dog Toy	1493	4.0	7.97	1	76.730764	https://www.chewy.com/charming-pet-animates-la...	141.544024
1132	Frisco Wagazoo Plush Squeaking Alligator Dog Toy	1132	4.3	10.98	1	95.206216	https://www.chewy.com/frisco-wagazoo-plush-squ...	137.041393
2213	Paws & Pals Y-Balls Rope Dog Toy	2213	2.5	6.95	4	103.220272	https://www.chewy.com/paws-pals-y-balls-rope-d...	136.471055
1265	KONG Sherps Donkey Dog Toy	1265	3.6	8.99	1	85.566250	https://www.chewy.com/kong-sherps-donkey-dog-t...	125.940570
1141	Frisco Wagazoo Plush Squeaking Triceratops Dog...	1141	4.2	12.08	1	95.613716	https://www.chewy.com/frisco-wagazoo-plush-squ...	122.186409
1412	Charming Pet Scrunch Bunch Bunny Squeaky Plush...	1412	5.0	9.59	1	61.751071	https://www.chewy.com/charming-pet-scrunch-bun...	118.335824
2274	Dogit Minty Knotted Rope Tough Dog Toy, Large	2274	4.1	5.99	4	46.332109	https://www.chewy.com/dogit-minty-knotted-rope...	116.562661

Hybrid recommendation system

- **Model 2: Hybrid recommendation system using Euclidean Distance**
- Euclidean distance is the most common use of distance measure. In most cases when people say about distance, they will refer to Euclidean distance. The Euclidean distance between two points is the length of the path connecting them.



Top 10 Recommendation

Recommendation (userId, toy title)
1. Recommended toys in Top 50 similarity scores
2. Recommendation sort by Score02 descending
3. Intersection of 1 & 2, and keep the top 10 toys.

```
from sklearn.metrics.pairwise import euclidean_distances

def recommendations_euclidean(userId, title):
    #print(df05.loc[df05['title']==title])
    #print('**' * 40)
    # finding cosine similarity for the vectors
    D = euclidean_distances(tfidf_matrix)
    #cosine_similarities = cosine_similarity(tfidf_matrix, tfidf_matrix)
    #Reverse mapping of the index
    indices02 = pd.Series(df04.index, index = df04['title']).drop_duplicates()

    idx02 = indices02[title]
    d_scores = list(enumerate(D[idx02]))
    d_scores = sorted(d_scores, key = lambda x: x[1], reverse = True)
    d_scores = d_scores[1:51]
    toy_indices11 = [i[0] for i in d_scores]
    toys12 = df04.iloc[toy_indices11][['title', 'title02','rating', 'price','subcat','cat', 't

    toys12['est'] = toys12['index'].apply(lambda x: svd.predict(userId, toys12.loc[x]['index']
    toys12['score02'] = ((toys12['est'] * toys12['score'])/toys12['price'])*toys12['rating']
    toys12 = toys12.sort_values('score02', ascending=False)

    cond11 = (df04.index.isin(toy_indices11))
    index12 = df04[df04['title']==title]['index']
    #cond12 = (df04.topic_num.isin(df04.iloc[index12]['topic_num']))
    cond13 = (df04.rating > 0)
    recommend11 = toys12.loc[cond11 & cond13].sort_values(by='score02', ascending=False)
    toys_recommend02 = recommend11[['title', 'index','rating', 'price','topic_num','score','li
    return toys_recommend02.head(10)
```

Results:
Model 2: Hybrid recommendation system using Euclidean Distance

```
recommendations_euclidean_output(1, 'KONG Tuggz Sloth Dog Toy')
```

executed in 525ms, finished 02:42:26 2020-12-17

```
      index                        title      score  topic_num  price  rating
1447   1447  KONG Tuggz Sloth Dog Toy  79.256675          1  15.99     3.0
**********************************************************************************
```

	title	index	rating	price	topic_num	score	link	score02
376	Ethical Pet Sensory Ball Tough Dog Chew Toy, C...	376	3.6	4.60	0	82.274071	https://www.chewy.com/ethical-pet-sensory-ball...	236.661711
1197	Frisco Summer Fun Plush Suntan Lotion Dog Toy	1197	4.7	5.99	1	75.997263	https://www.chewy.com/frisco-summer-fun-plush-...	219.174152
439	Ethical Pet Play Strong Rubber Ball Tough Dog ...	439	4.0	5.91	3	83.907268	https://www.chewy.com/ethical-pet-play-strong-...	208.733635
1386	ZippyPaws Polar Bear Miniz Dog Toy, 3 count	1386	5.0	5.72	0	64.682916	https://www.chewy.com/zippypaws-polar-bear-min...	207.818357
624	Nerf Dog Atomic Flyer Dog Toy, Large	624	3.9	7.99	0	96.537448	https://www.chewy.com/nerf-dog-atomic-flyer-do...	173.194457
1128	Frisco Retro Fanny Pack Plush with Rope Squeak...	1128	4.5	6.98	1	72.357772	https://www.chewy.com/frisco-retro-fanny-pack-...	171.459922
619	Chuckit! Flying Squirrel Dog Toy, Color Varies	619	4.3	7.99	0	81.402756	https://www.chewy.com/chuckit-flying-squirrel-...	161.020486
771	Planet Dog Orbee-Tuff Raspberry Treat Dispensi...	771	4.2	5.99	0	60.627157	https://www.chewy.com/planet-dog-orbee-tuff-ra...	156.246400
1339	Pet Qwerks Frog Sound Plush Dog Toy	1339	5.0	9.99	1	84.156153	https://www.chewy.com/pet-qwerks-frog-sound-pl...	154.814180
134	Planet Dog Orbee-Tuff Guru Treat Dispensing Do...	134	4.5	12.99	4	113.273238	https://www.chewy.com/planet-dog-orbee-tuff-gu...	144.228479

Model Evaluation:

I'll evaluate the learned recommender system by converting the ratings to negative and positive

		reality (true toy rating)	
prediction (score)		predicted toy rating >= true rating	predicted toy rating < true rating
	predicted toy score >= toy score	tp	fp
	predicted toy score < toy score	fn	tn

Metrics: Precision, Recall, F1

Recommendation is viewed as information retrieval task: Retrieve (recommend) all items which are predicted to be “good”.

- Precision: a measure of exactness, determines the fraction of relevant items retrieved out of all items retrieved

Precision = True positive / (True positive+False Positive)

- Recall: a measure of completeness, determines the fraction of relevant items retrieved out of all relevant items

Recall = True positive / (True Positive+False Negative)

- F1: The F1 Metric attempts to combine Precision and Recall into a single value for comparison purposes. May be used to gain a more balanced view of performance.

Compare model:

Model Evaluation:

Precision, Recall, F1

```
compare(1, 'KONG Tuggz Sloth Dog Toy')
```

executed in 9.64s, finished 10:08:03 2020-12-18

```
Cosine evaluation: (0.8333333333333334, 0.5555555555555556, 0.6666666666666667)
**********************************************************************
Euclidean evaluation: (1.0, 0.6, 0.7499999999999999)
```

```
compare(104, 'Charming Pet Ropes-A-Go-Go Gator Squeaky Plush Dog Toy')
```

executed in 9.59s, finished 10:08:14 2020-12-18

```
Cosine evaluation: (0.6, 0.375, 0.4615384615384615)
**********************************************************************
Euclidean evaluation: (0.6, 0.42857142857142855, 0.5)
```

```
compare(201, 'JW Pet Play Place Butterfly Puppy Teether, Color Varies')
```

executed in 9.93s, finished 10:08:27 2020-12-18

```
Cosine evaluation: (0.8888888888888888, 0.8888888888888888, 0.8888888888888888)
**********************************************************************
Euclidean evaluation: (0.875, 0.7777777777777778, 0.823529411764706)
```

```
compare(19, 'Petstages Dogwood Mesquite Tough Dog Chew Toy')
```

executed in 9.58s, finished 10:08:45 2020-12-18

```
Cosine evaluation: (0.75, 0.8571428571428571, 0.7999999999999999)
**********************************************************************
Euclidean evaluation: (0.7777777777777778, 0.875, 0.823529411764706)
```

Conclusion:

- **Limitation:**
- Few user data; recommendation mainly based on the item not user
- Preference ranking
- **Deployment:**
- Deploy both models, compare model performance, automatically select the best recommender model and generate results for a customer
- **Future improvement:**
- In order to build reliable recommender system, I need more user data and build more models by using different similarity measures

Thank you!