Data Science Training - Day 5

Reinforcement learning

REINFORCEMENT LEARNING Accept data on the go - onine learning Adaptive
Upper Confidence Bound
Thompsan Sampling

- Adaptive learning!
- Provide data on the go
- Code the nature of behavior of the machine
- Lot of flexibility we require to code
- Not use any ready-made packages
- Use raw model
- Popularly used in robotics

At any given situation, robotics takes right step, then it is provided with positive reward. If robotics doesn't take proper steps to overcome the hurdle, then it is provided with negative reward.

Example 1:

Positive rewards: Provide smile and hug

Negative reward: Giving warning

Growing with good habits

Example 2: It is also used in e-commerce applications

Utilizing the existing information - people searching the product, bought, clicks –Product moves up

At the 66th minute, we declare the product as trending!

Trending product of the hours

Example 3: Crossing the bridge for the first time in the life.

None of them knows how to do that

First person is at risk

First person uses all his common sense and knowledge and takes steps

Second person uses the first person and perform the steps

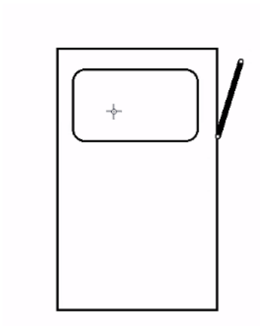
Third person sees the first and second person and executes the better step.

Best possible decision is made by watching out the previous step.

Thompsan Sampling:

Multi arm banded machine

Take a pully example: When we pull the pulley (black bolded one) - A number will be displayed in the LCD display.



These are the casino machines

Out of these four what is the best – won't be disclosed

Don’t know in which machine we should spend all these money

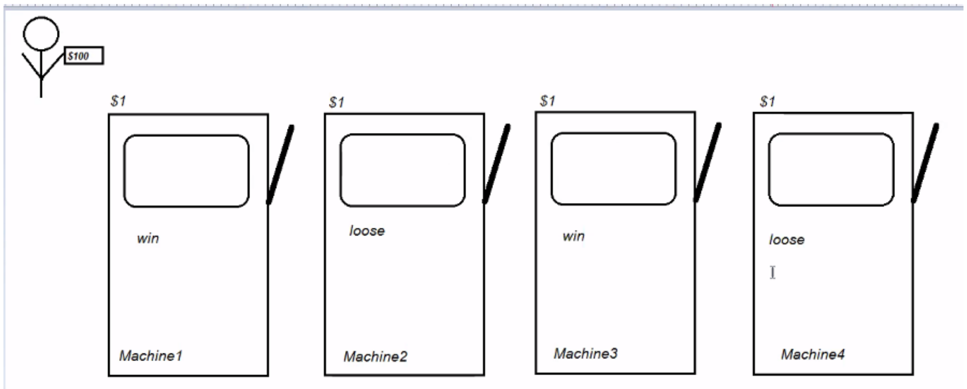
Only one machine will be good.

We will be in Delamo – which one is best?

Best is only own by the casino owners

You can see machine 4 is best after playing.

We will take decision after playing the game and based on the win.

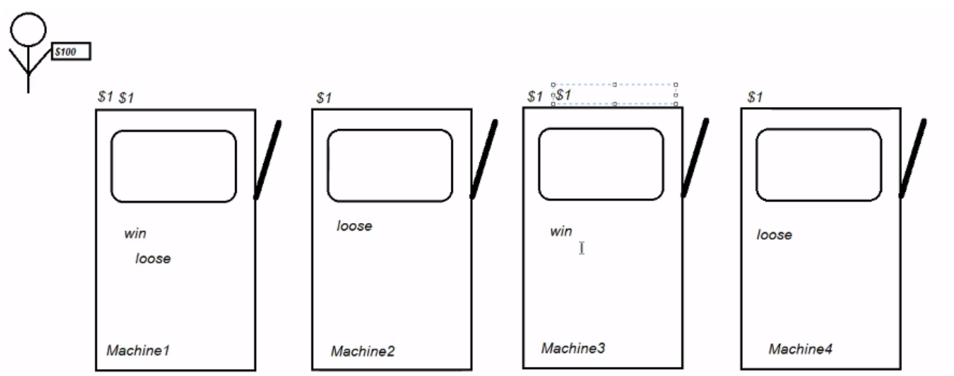


Our agenda is to find in which machine we win.

Now, where we will put the 5th dollar? Where to spend it?

We will go to the machine which we win

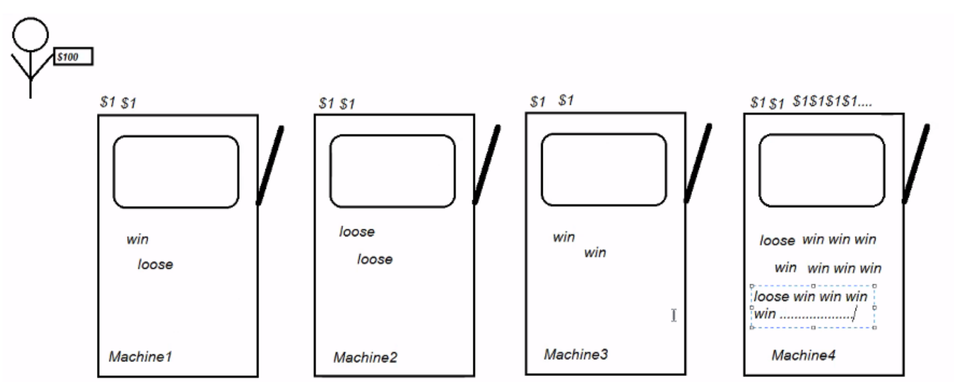
(machine 1 – lucky one consider – put the dollar – but you loose



Eventually play more and decide or find out the best machine

Here we perform **exploration – exploitation**.

Explore all our choices and then exploit our best choices.



What is the use case we are going to solve?

Product company: ABC company

New Product: xyz product

TO promote this, they add “AD agency”.

Start developing very attractive ads. Say they prepare 10 variations.

They call product owners to choose the best variates. They will say customers to choose the best.

Make use of social media platforms. They are giving us 10000 rounds Free

1 Round == 1 user connection the social media account.

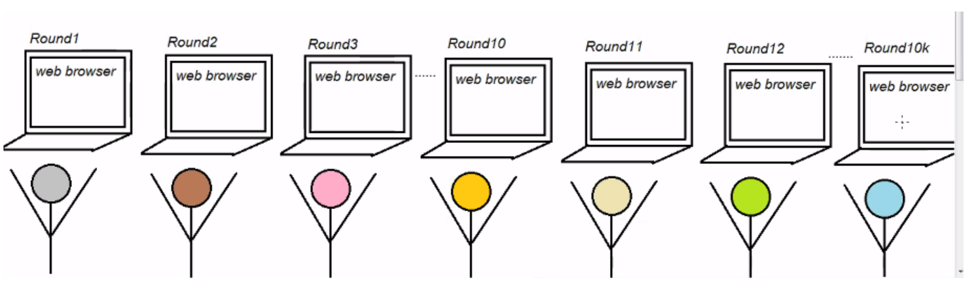
People are Unknowingly targeted for their product to do the survey.

Example:

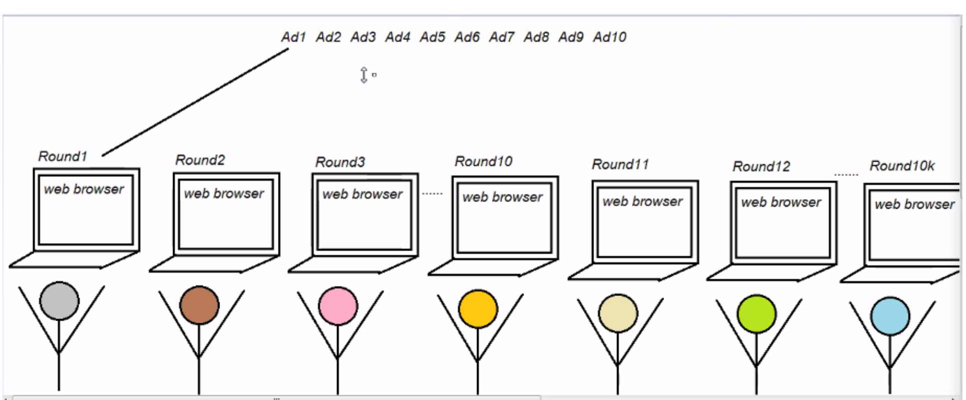
Youtube – ads floating in the starting – Fresh contents – something attracts you – you click – Interest captured with the click.

If you ignore that, considered as not interested.

If you keep on seeing it after 5 seconds, then you are showing the interested.



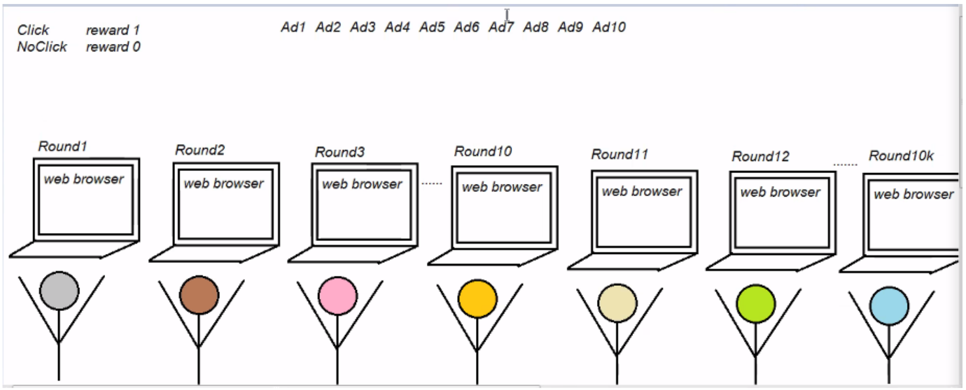
Independently connecting by the interest.



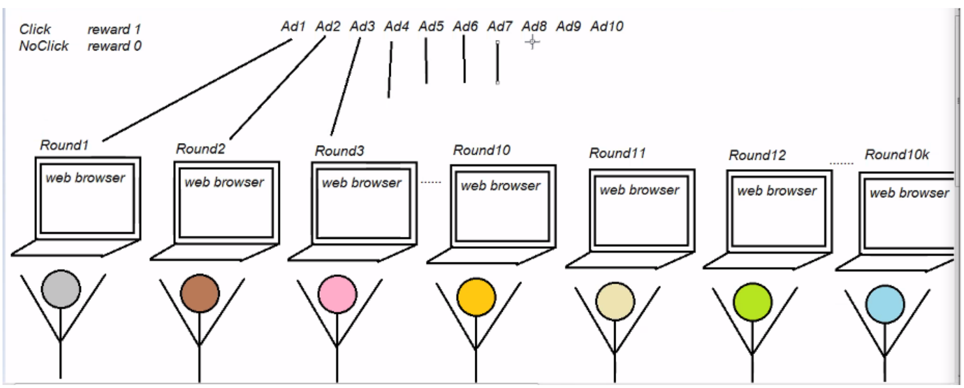
Only one add is displayed on the user

If someone likes it – click – reward 1

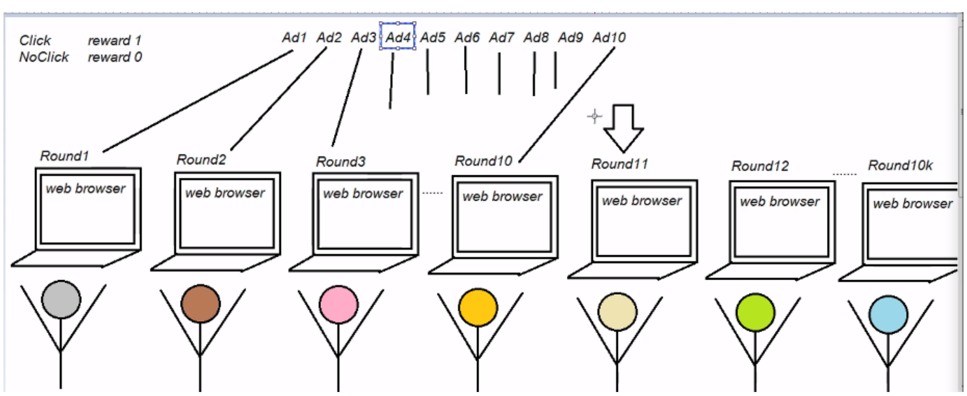
No click – reward 0



Give everyone one chance



Now, the question is what ad should I show him?

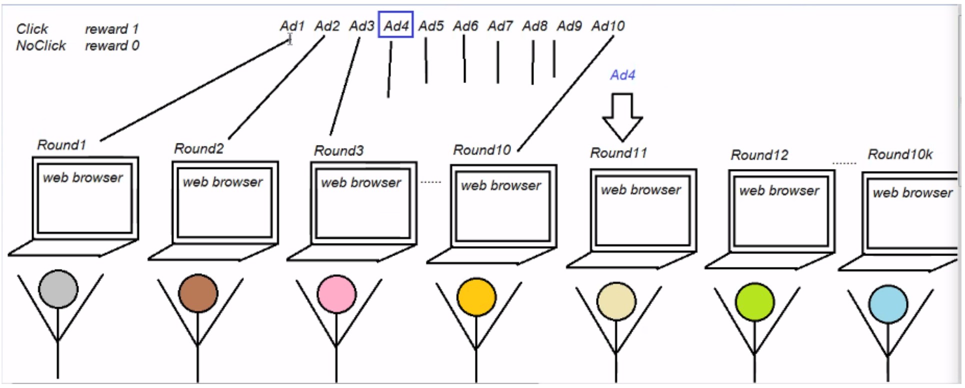


Consider Ad 4 is gained a click!

If none of them gets click, then we will give all ads another chance.

We will provide the Ad 4 (exploit) to other users and gain the click count.

Who has got the max reward? what if only ad4 gets more chance – logic becomes unbiased



Round 15 , Ad4 has got 6 chances it has earned 6 rewards

Average Reward = TotalRewards / NumberOfChancesToDisplay
= 6/6 = 1

Every time Ad 4 is given a chance to appear in the user browser, it gains the reward.

Round 20 , Ad4 has got 11 chances its has earned 6 rewards

Average Reward = 6/11 = 0.54

Rewards didn't increase but rounds got increased!

So, the average reward decreases.

Here, they won't survive the competitions.

How the ads get the chances, if the average reward is 0

When they will be getting chance?

Everyone will be given grace rewards.

Small bias is provided by this grace rewards.

graceRewards	*Number Of Chances a Ad has got* *If a Ad has got more chances to display "**graceRewards**" will be very small value* *If a Ad has got less chances to display "**graceRewards**" will be large value*

Upper Confidence Bound

- ❖ There are many algorithms to optimize the decision making behaviour of the agent, some perform better than others.
- ❖ A very popular method is the UCB exploration strategy
- ❖ This algorithm chooses the arm based on the average reward mean plus an exploration bonus.
- ❖ The exploration bonus is dependent on the number of times the action has been tried out before and the total number of action selections.

- We have **d** Ads that we display to users each time they connect to web page.
- Each time a user connects to this web page, that makes a **round**
- At **each round n**, we choose one Ad to display to the user.
- At each **round n**, **Ad i** gets reward
- if the user clicked on Ad $r_i(n) \in \{0, 1\}$: $r_i(n) = 1$
- if the user didnt then 0
- The goal is to **maximize the total reward** we get over many rounds

Step 1. At each round n, we consider two numbers for each ad i:

- $N_i(n)$ - the number of times the ad i was selected up to round n,
- $R_i(n)$ - the sum of rewards of the ad i up to round n.

Step 2. From these two numbers we compute:

- the average reward of ad i up to round n

$$\bar{r}_i(n) = \frac{R_i(n)}{N_i(n)}$$

- UCB $\bar{r}_i(n) + \Delta_i(n)$

$$\Delta_i(n) = \sqrt{\frac{3}{2}\frac{\log(n)}{N_i(n)}}$$

Grace reward is the delta i.

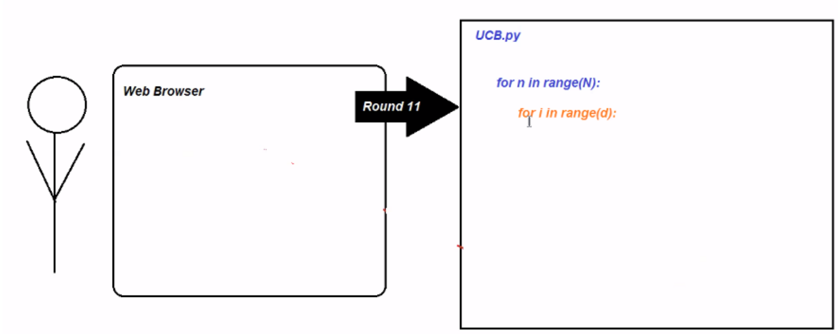
Less chance – more grace reward

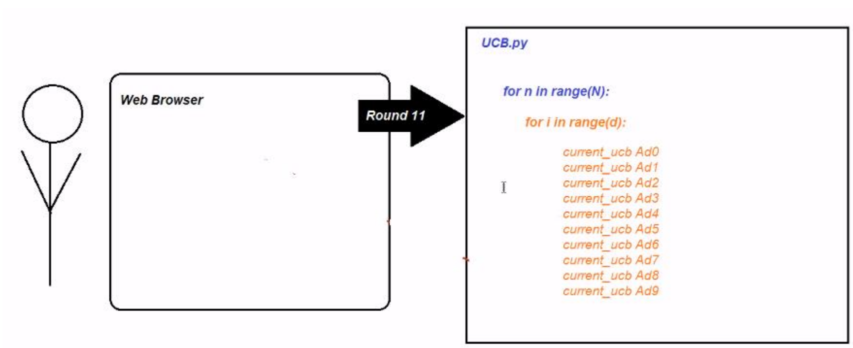
More chance – less grace reward

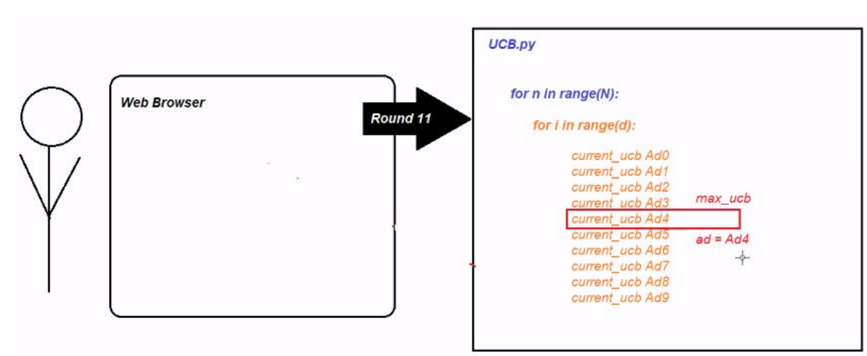
There are lots of things happen for not click. Though the Ads is potential, this grace marks will help to get the ads more clicks.

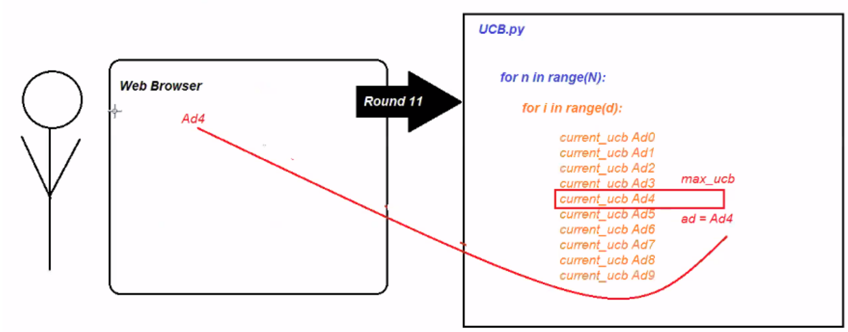
Situation – power cut, accidently click somewhere or another web browser opens.

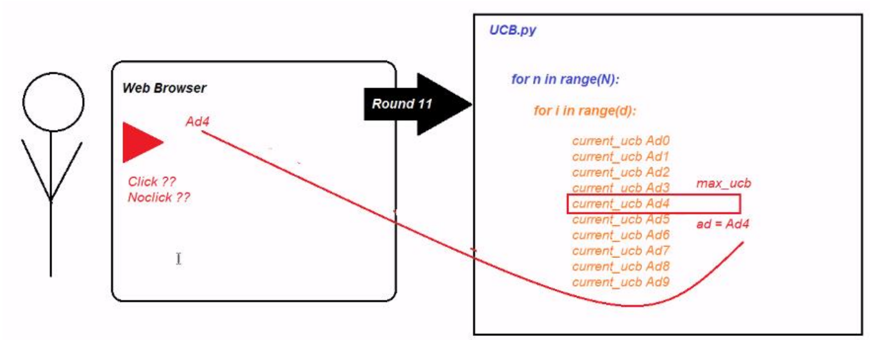
Step 3. We select the ad i that has the maximum UCB $\bar{r}_i(n) + \Delta_i(n)$.








Web Browser
Ad4
Round 11
UCB.py
for n in range(N):
for i in range(d):
current_ucb Ad0
current_ucb Ad1
current_ucb Ad2
current_ucb Ad3
current_ucb Ad4
current_ucb Ad5
current_ucb Ad6
current_ucb Ad7
current_ucb Ad8
current_ucb Ad9
max_ucb
ad = Ad4


Web Browser
Ad4
Click ??
Noclick ??
Round 11
UCB.py
for n in range(N):
for i in range(d):
current_ucb Ad0
current_ucb Ad1
current_ucb Ad2
current_ucb Ad3
current_ucb Ad4
current_ucb Ad5
current_ucb Ad6
current_ucb Ad7
current_ucb Ad8
current_ucb Ad9
max_ucb
ad = Ad4

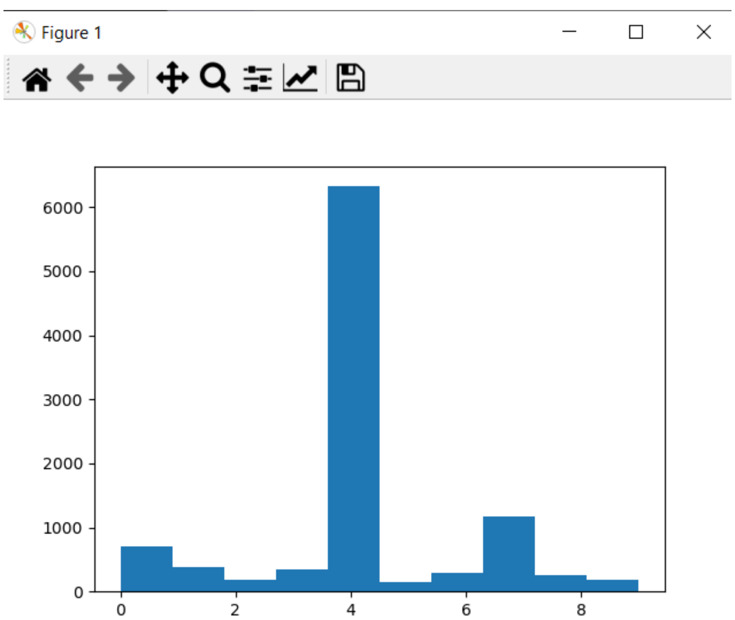
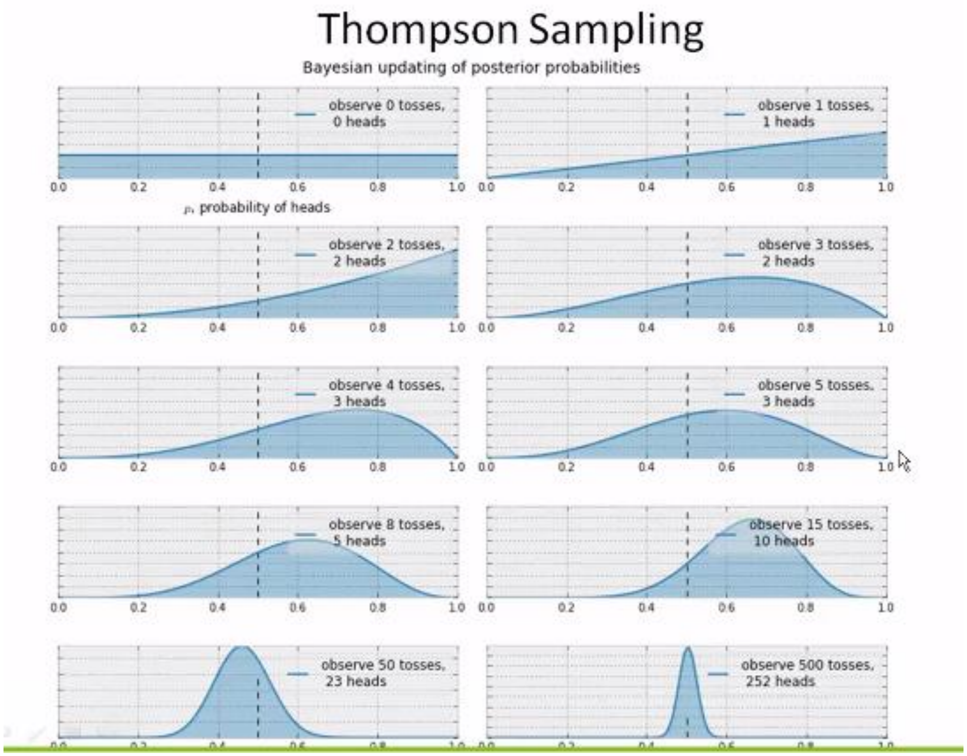


Figure 1
6000
5000
4000
3000
2000
1000
0
0
2
4
6
8



Probabilistic behavior

Posterior Probability

P(A|B) = P(A) * P(B|A) / P(B)

P(A|B) probability of Event A given Event B has occured

Probability of the Ad to get a click given it has got 10 noclicks

For each Ad we need to Capture

number of clicks
number of no-clicks

Thompson Sampling:

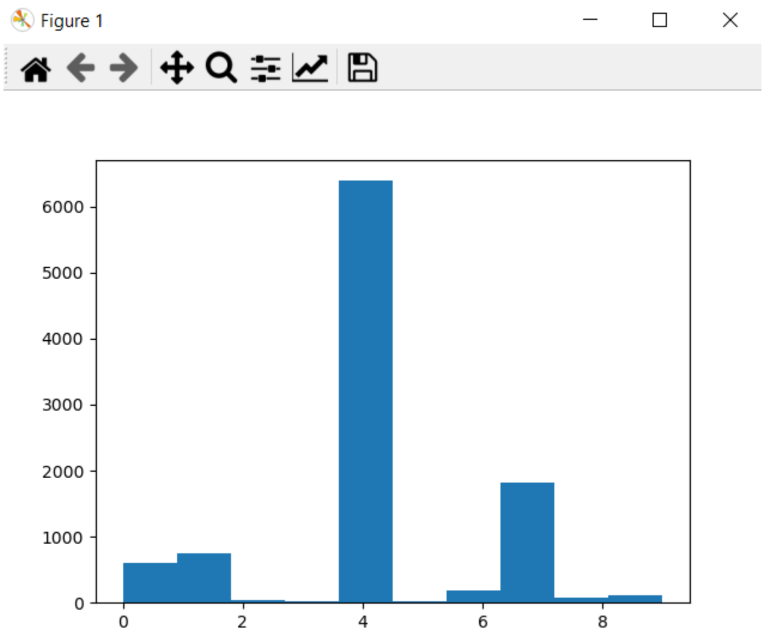
Thompson Sampling

Step 1. At each round n, we consider two numbers for each ad i:

- $N_i^1(n)$ - the number of times the ad i got reward 1 up to round n,
- $N_i^0(n)$ - the number of times the ad i got reward 0 up to round n.

Step 2. For each ad i, we take a random draw from the distribution below:

$$\theta_i(n) = \beta(N_i^1(n) + 1, N_i^0(n) + 1)$$



NLP:

NLTK – NL tool kit

Natural Lang Processing (nltk) Natural Lang. Tool-Kit

Review	Liked
Wow....Loved this place.	1
Crust is not that good :(	0
Great place will come back :) !!	1

Step1: Substitue all non alphabets with a space , using python re package

Wow....Loved this place
Crust is not that good
Great place will come back

Wow Loved this place
Crust is not that good
Great place will come back

Step2: Convert the review to lower case , python lower()

wow loved this place
crust is not that good
great place will come back

Step3: Convert the stmt into tokens of words , using split() method in python
Tokenization

[wow, loved, this, place]
[crust ,is ,not ,that ,good]
[great ,place, will, come, back]

Step4:Eliminate Stopwords , using nltk stopwords

[wow, loved, place]
[crust ,not ,good]
[great ,place, will, come, back]

Step5:Stemming of words, using nltk stemmer

love	loved	lovable	lovely
love	love	love	love

[wow, love, place]
[crust ,not ,good]
[great ,place]

Step6: Join the words back to stmt, using join() method in python

wow love place
crust not good
great place

Step7: Convert Text to number , using CountVectorizer & Calculation of TFIDF-(Term Frequency and inverse document Freq)

	crust	good	great	love	not	place	wow
wow love place	0	0	0	1	0	1	1
crust not good	1	1	0	0	1	0	0
great place	0	0	1	0	0	1	0

Step7: Convert Text to number , using CountVectorizer & Calculation of TFIDF-(Term Frequency and inverse document Freq)

	crust	good	great	love	not	place	wow	Liked
wow love place	0	0	0	1	0	1	1	1
crust not good	1	1	0	0	1	0	0	0
great place	0	0	1	0	0	1	0	1

X

Y

NLP

- TF-IDF
- TF => Trem Frequency
- IDF => Inverse document frequency

TF = Term Frequency

IDF = Inverse Document Frequency

TF-TDF = TF * IDF

TF

$$\frac{(Number\ of\ occurrences\ of\ a\ word\ in\ a\ document)}{(Number\ of\ words\ in\ that\ document)}$$

"to be or not to be"

to = $\frac{1+1}{6}$

to = 0.33

be = 0.33

or = 0.16

Words/ Documents	Document 1	Document 2	Document 3
going	0.16	0.16	0.12
to	0.16	0	0.12
today	0.16	0.16	0
i	0	0.16	0.12
am	0	0.16	0.12
it	0.16	0	0
is	0.16	0	0
rain	0.16	0	0

IDF

Formula

$$log\left(\frac{(Number\ of\ documents)}{(Number\ of\ documents\ containing\ word)}\right)$$

$$log\left(\frac{(Number\ of\ documents)}{(Number\ of\ documents\ containing\ word)}\right)$$

"to be or not to be"
"i have to be"
"you got to be"

to = $\log\left(\frac{3}{3}\right)$

to = 0

be = $\log\left(\frac{3}{3}\right)$

be = 0

have = $\log\left(\frac{3}{1}\right)$

Words	IDF Value
going	0
to	0.41
today	0.41
i	0.41
am	0.41
It	1.09
is	1.0[illegible]
rain	1.09

Words/ Documents	Document 1	Document 2	Document 3
going	0.16	0.16	0.12
to	0.16	0	0.12
today	0.16	0.16	0
i	0	0.16	0.12
am	0	0.16	0.12
it	0.16	0	0
is	0.16	0	0
rain	0.16	0	0

Words/ Documents	going	to	today	i	am	it	is	rain
Document 1	0	0.07	0.07	0	0	0.17	0.17	0.17
Document 2	0	0	0.07	0.07	0.07	0	0	0
Document 3	0	0.05	0	0.05	0.05	0	0	0

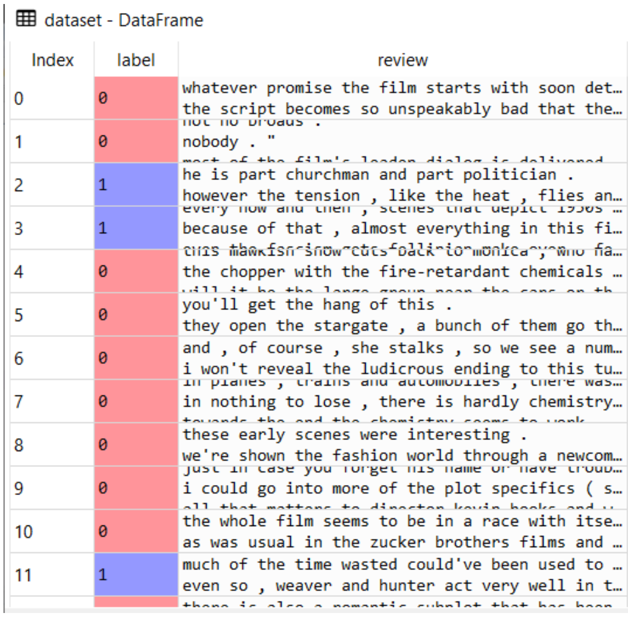
$$TFIDF(Word) = TF(Document, Word) * IDF(Word)$$

Step8: Split the data into Train-Test, choose a algo, check accuracy

Implementation:

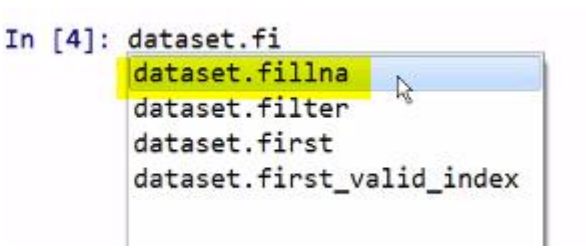
dataset - DataFrame

Index	label	review
0	neg	whatever promise the film starts with soon det… the script becomes so unspeakably bad that the…
1	neg	nobody . "
2	pos	he is part churchman and part politician . however the tension , like the heat , flies an…
3	pos	because of that , almost everything in this fi…
4	neg	the chopper with the fire-retardant chemicals …
5	neg	you'll get the hang of this . they open the stargate , a bunch of them go th…
6	neg	and , of course , she stalks , so we see a num… i won't reveal the ludicrous ending to this tu…
7	neg	in nothing to lose , there is hardly chemistry…
8	neg	these early scenes were interesting . we're shown the fashion world through a newcom…
9	neg	i could go into more of the plot specifics (s…
10	neg	the whole film seems to be in a race with itse…

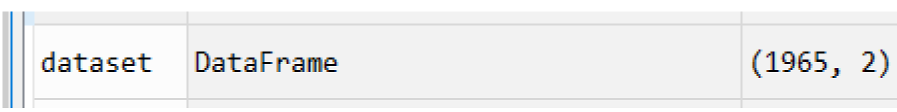


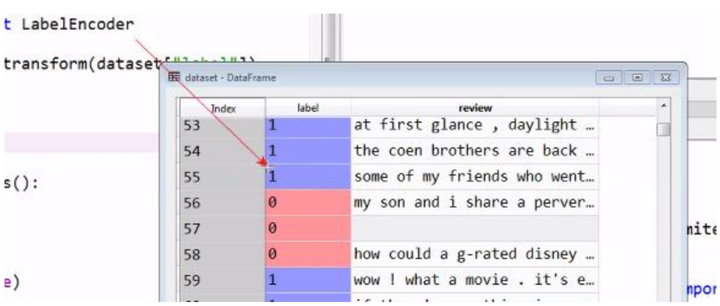
dataset - DataFrame

Index	label	review
0	0	whatever promise the film starts with soon det… the script becomes so unspeakably bad that the…
1	0	nobody . "
2	1	he is part churchman and part politician . however the tension , like the heat , flies an…
3	1	because of that , almost everything in this fi…
4	0	the chopper with the fire-retardant chemicals …
5	0	you'll get the hang of this . they open the stargate , a bunch of them go th…
6	0	and , of course , she stalks , so we see a num… i won't reveal the ludicrous ending to this tu…
7	0	in nothing to lose , there is hardly chemistry…
8	0	these early scenes were interesting . we're shown the fashion world through a newcom…
9	0	i could go into more of the plot specifics (s…
10	0	the whole film seems to be in a race with itse… as was usual in the zucker brothers films and …
11	1	much of the time wasted could've been used to … even so , weaver and hunter act very well in t…



After dropna:





Index	label	review
53	1	at first glance , daylight …
54	1	the coen brothers are back …
55	1	some of my friends who went…
56	0	my son and i share a perver…
57	0	
58	0	how could a g-rated disney …
59	1	wow ! what a movie . it's e…

Identify the blank spaces:

blanks - List (27 elements)

Index	Type	Size	Value
0	int	1	57
1	int	1	71
2	int	1	147
3	int	1	151
4	int	1	283
5	int	1	307
6	int	1	313
7	int	1	323
8	int	1	343
9	int	1	351
10	int	1	427
11	int	1	501

After dropping blanks:

dataset	DataFrame	(1938, 2)

```
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer    # class for stemming

#stopset = set(stopwords.words('english'))

stopset = set(stopwords.words('english')) - set(('over', 'under', 'below', 'more', 'most',

# ==============================================================================
# Cleaning of Text
# ==============================================================================

corpus = []          # variable corpus of type list is a collection of text, so this variat

for i in range(0, len(dataset)):

    review = re.sub('[^a-zA-Z]', ' ', dataset.iloc[i,1])          # cleaned reviews in vari
                    # [^a-zA-Z] indicates what we dont want to remove
                    # Replace the removed character by space
    review = review.lower()
```

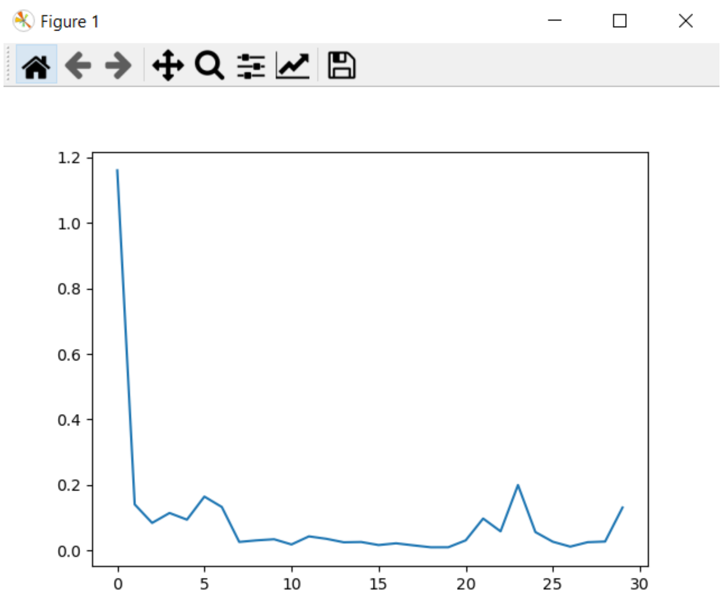
Variable explorer | File explorer

IPython console

Console 1/A

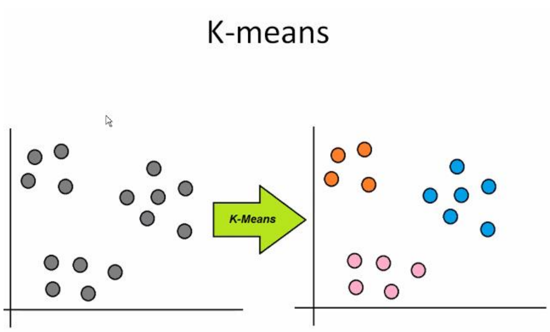
```
 'yours',
 'yourself',
 'yourselves',
 'he',
 'him',
 'his',
 'himself',
 'she',
 "she's",
 'her',
 'hers',
 'herself',
 'it',
 "it's",
 'its',
```

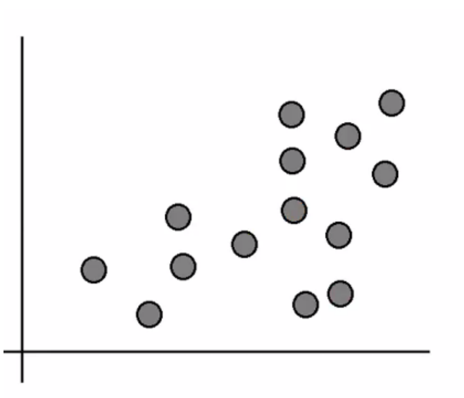
NLP using Deep learning:



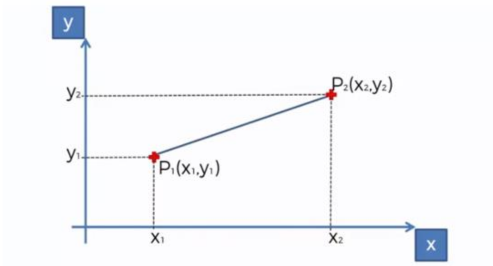
Unsupervised learning:

K means clustering

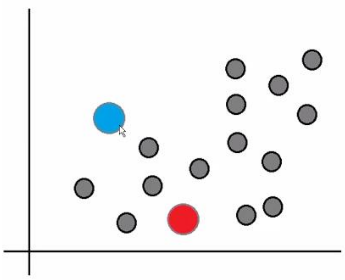
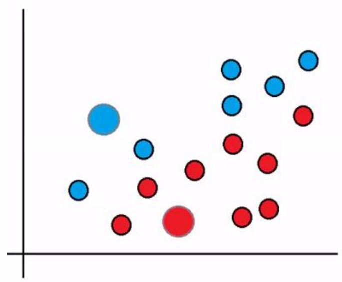
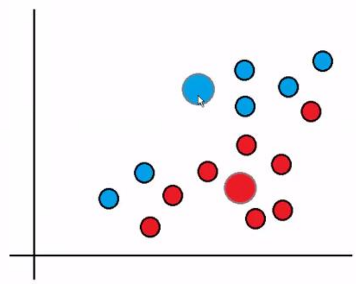


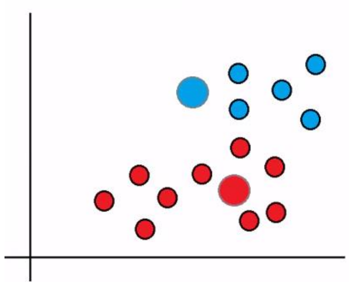
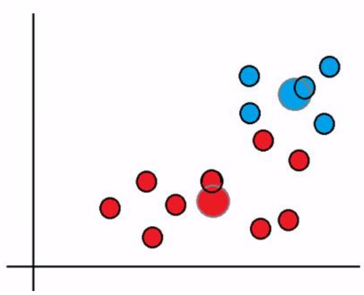
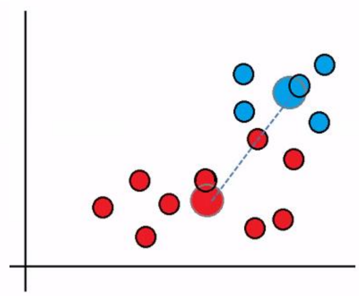


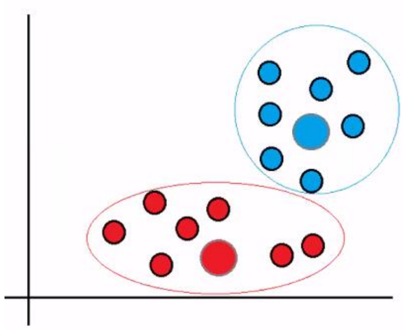
Euclidean Distance



Euclidean Distance between P_1 and $P_2 = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$



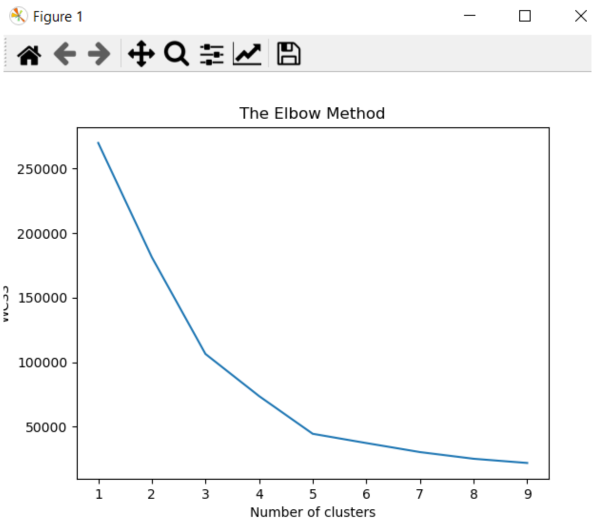
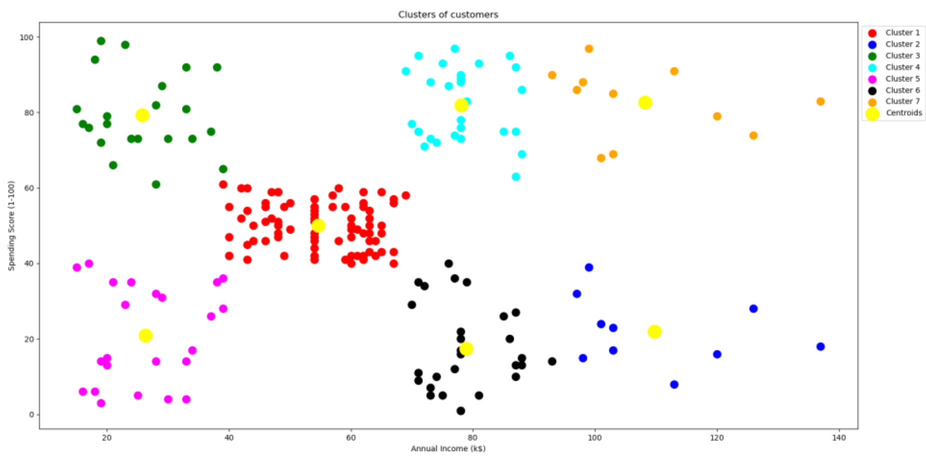
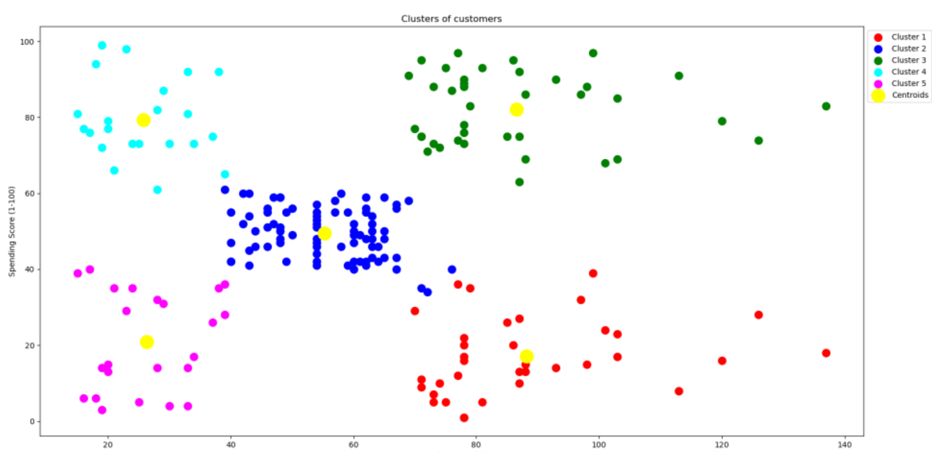
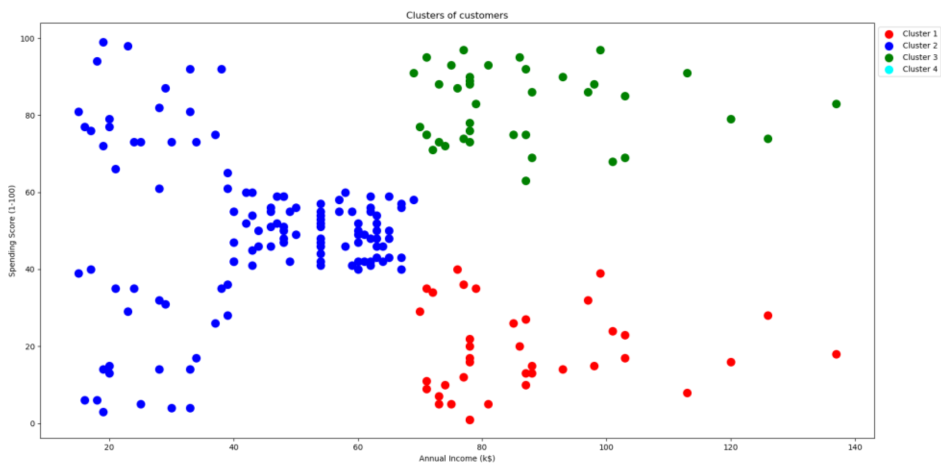


Figure 1
The Elbow Method
250000
200000
150000
100000
50000
1
2
3
4
5
6
7
8
9
Number of clusters


Clusters of customers
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster 5
Cluster 6
Cluster 7
Centroids
Spending Score (1-100)
Annual Income (k$)


Clusters of customers
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster 5
Centroids
Spending Score (1-100)
Annual Income (k$)



Trend stock prices – RNN:

No limitation in the type of data in neural network

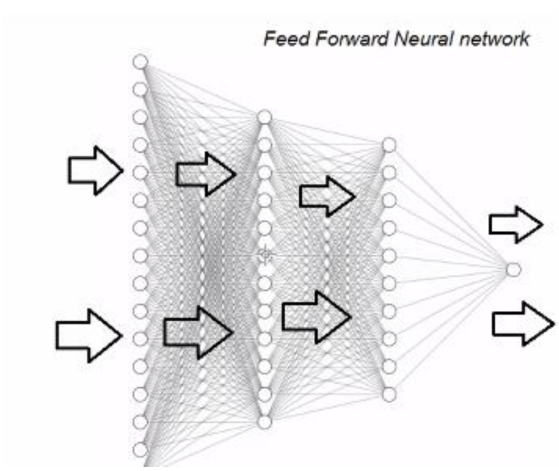
LSTM have different architecture. They have recursion inside them

Recursive functions – call function itself. Feed the input to it and process it.

Let's see what we have in recursion part:

Good when they don't depend on previous output

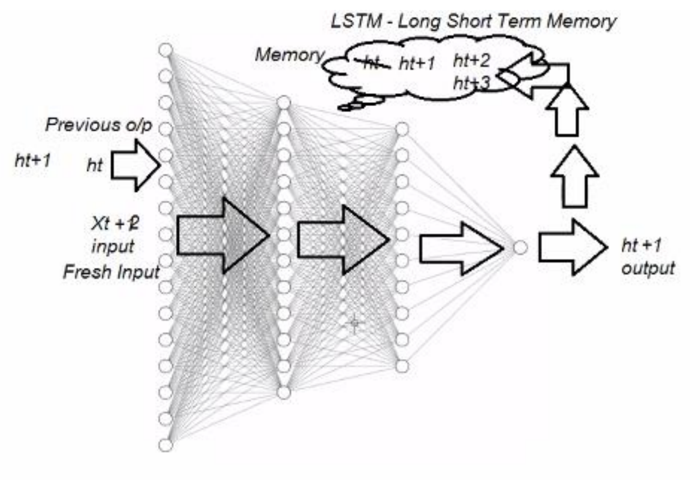
They don't have memory



Recursive NN– have reference of previous output

They will have memory to preserve it.

Preserve as long as you want!



TimeStep = 3 , lookback at 3 previous words

Predict next word by looking at previous 3 words

This
This is
This is my
This is my class
This is my class of
This is my clss of Data
This is my clss of Data science
This is my clss of Data science and
This is my clss of Data science and Machine
This is my clss of Data science and Machine Learning

Help you to preserve the previous output

Sequential processing

They have specialized architecture – Gateway architecture

Long time memory is the cell state.

What remains in the memory is decided by cell state

Cell state – consider it as conveyor belt. Put what you want in that belt

Forget gate – 0

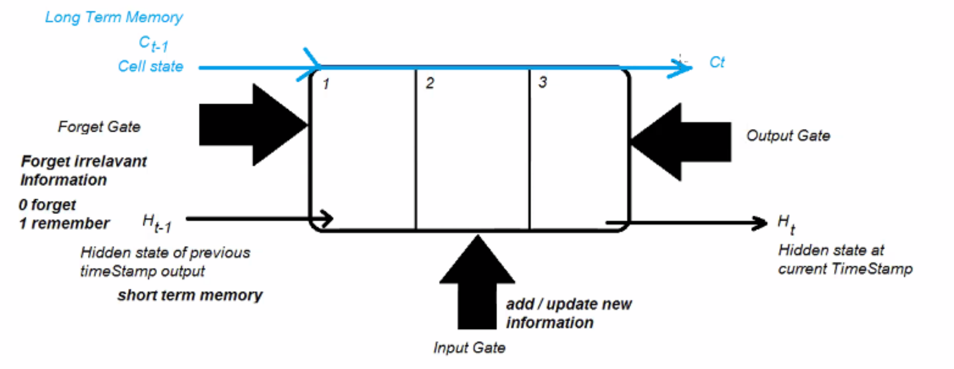
Remember gate – 1

Input gate – add the input data

Output gate – Put the output info in the belt

Long term memory and short-term memory is the immediate output.

To take references and make predictions



What is the use case of using the LSTM?

Stock price analysis:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	symbol	date	close	high	low	open	volume	adjClose	adjHigh	adjLow	adjOpen	adjVolume	divCash	splitFactor
0	AAPL	2015-05-2	132.045	132.26	130.05	130.34	4.6E+07	121.683	121.881	119.844	120.111	4.6E+07	0	1
1	AAPL	2015-05-2	131.78	131.95	131.1	131.86	3.1E+07	121.438	121.595	120.812	121.512	3.1E+07	0	1
2	AAPL	2015-05-2	130.28	131.45	129.9	131.23	5.1E+07	120.056	121.134	119.706	120.932	5.1E+07	0	1
3	AAPL	2015-06-C	130.535	131.39	130.05	131.2	3.2E+07	120.291	121.079	119.844	120.904	3.2E+07	0	1
4	AAPL	2015-06-C	129.96	130.655	129.32	129.86	3.4E+07	119.761	120.402	119.171	119.669	3.4E+07	0	1
5	AAPL	2015-06-C	130.12	130.94	129.9	130.66	3.1E+07	119.909	120.664	119.706	120.406	3.1E+07	0	1
6	AAPL	2015-06-C	129.36	130.58	128.91	129.58	3.8E+07	119.208	120.333	118.794	119.411	3.8E+07	0	1
7	AAPL	2015-06-C	128.65	129.69	128.36	129.5	3.6E+07	118.554	119.512	118.287	119.337	3.6E+07	0	1
8	AAPL	2015-06-C	127.8	129.21	126.83	128.9	5.3E+07	117.771	119.07	116.877	118.784	5.3E+07	0	1
9	AAPL	2015-06-C	127.42	128.08	125.62	126.7	5.6E+07	117.421	118.029	115.762	116.757	5.6E+07	0	1
10	AAPL	2015-06-1	128.88	129.34	127.85	127.92	3.9E+07	118.766	119.19	117.817	117.881	3.9E+07	0	1
11	AAPL	2015-06-1	128.59	130.18	128.475	129.18	3.5E+07	118.499	119.964	118.393	119.042	3.5E+07	0	1
12	AAPL	2015-06-1	127.17	128.33	127.11	128.185	3.7E+07	117.19	118.259	117.135	118.125	3.7E+07	0	1
13	AAPL	2015-06-1	126.92	127.24	125.71	126.1	4.4E+07	116.96	117.255	115.845	116.204	4.4E+07	0	1
14	AAPL	2015-06-1	127.6	127.85	126.37	127.03	3.1E+07	117.586	117.817	116.453	117.061	3.1E+07	0	1
15	AAPL	2015-06-1	127.3	127.88	126.74	127.72	3.3E+07	117.31	117.844	116.794	117.697	3.3E+07	0	1
16	AAPL	2015-06-1	127.88	128.31	127.22	127.23	3.5E+07	117.844	118.241	117.236	117.245	3.5E+07	0	1
17	AAPL	2015-06-1	126.6	127.82	126.4	127.71	5.5E+07	116.665	117.789	116.481	117.688	5.5E+07	0	1

Time step – consider

This will help to capture pattern from your data

Time step => Look back sequence

Predict next word looking at previous 3 words.

TimeStep => Look back sequence

Predict next word looking at previous 3 words TimeStep = previous 3 words == 3

Predict next word looking at previous 50 words TimeStep = previous 50 words == 50

Predict next days close price looking at previous 100 days close price of the stock
TimeStep = previous 100 days stock price == 100

If the t is the input data, then t+1 time will be the prediction.

This is the training data.

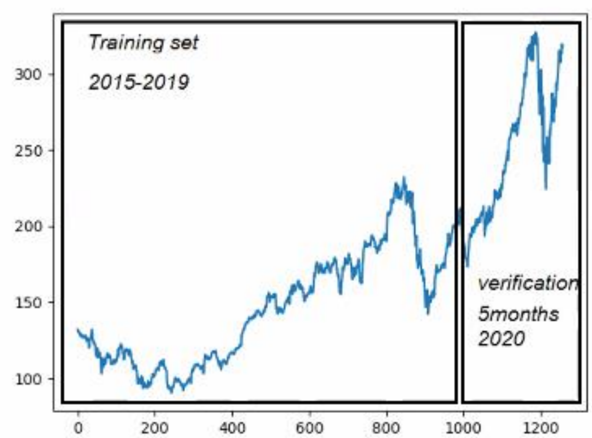
X matrix col = TimeStep

Input Data	Output Data
X_train	y_train
0 1 2 3 4 5 6 7 8........99	100
1 2 3 4 5 6 7 8 9......99 100	101
2 3 4 5 6 7 8 9.... 99 100 101	102

1..........................100 12th March ✓ 13thMarch 14th March

1..........................100 12th March ✓ 13thMarch ✗ 14th March ✗

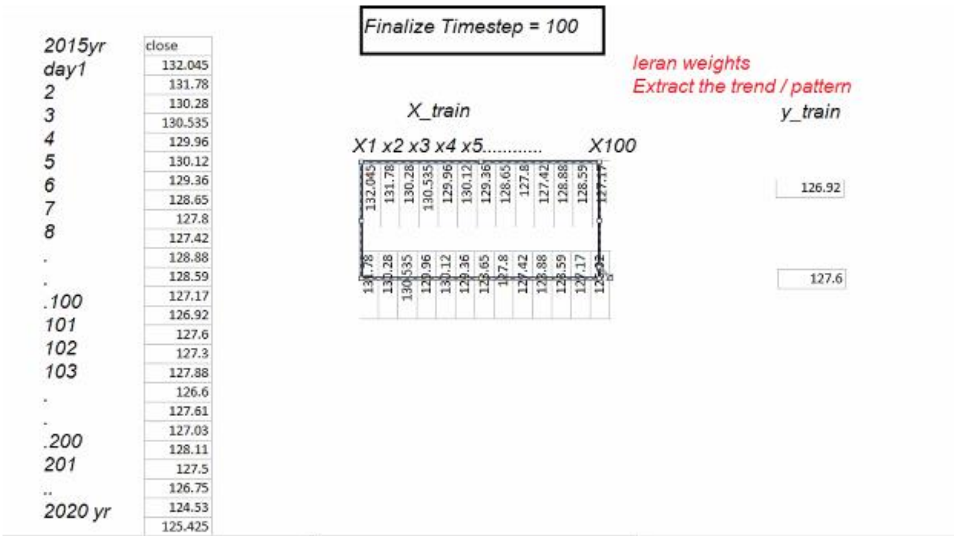
2015-2020 sample in hand

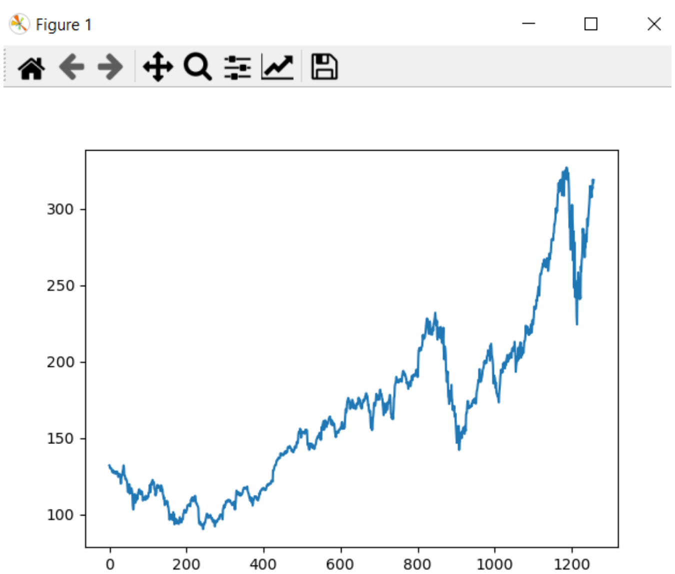


31st May 2020

1st Jan 2020

Close price Trend





!pip list

import pandas

pandas.__version__

https://github.com/meghakarale/DataScience-Reference-Repository

https://drive.google.com/drive/folders/1XazjLsUmnmnhw5tbOnPjlMpcADmtIPRV?usp=sharing

https://github.com/meghakarale/DataScience-Reference-Repository