Experiment No. 10

Aim :

Exploratory Data Analysis of text

Theory :

Word Clouds are visual representations of words that give greater prominence to words that appear more frequently. This type of visualization can help presenters to quickly collect data from their audience, highlight the most common answers and present the data in a way that everyone can understand. It's an effective way to analyse the data we're dealing with. To generate a WordCloud it is very important to remove Stop Words like the, and , of , from ,his, her etc from the data , as these words are most frequently used , they might render the Wordcloud useless .

In this experiment we are dealing with corona virus report, thus a lot of preprocessing is required to make the Data Analysis more efficient.

PreProcessing :

We start be removing any non ascii letters and then replace the slangs with their actual words like , "can't" becomes "can not" , "he'll" becomes "he will" etc. After this is accomplished we go on to remove extra white spaces which maybe occurring. After the preprocessing is done.

WordCloud Formation :

To form a WordCloud , we import the wordcloud library and create an object and feed it the relevant data .

A wordcloud is generated based on the corpus of covid-19 dataset. Words like organism can be seen in bold in this word cloud.

For further Data analysis a frequency histogram of the most commonly used words in the tweet corpus were also plotted .

Data Retrieved from this can be used to find out about the trending topics, peoples feelings on a certain topic , general public sentiment at any given point etc.

Advantages of Word Clouds :

1. Analyzing customer and employee feedback.
2. Identifying new SEO keywords to target.

Drawbacks of Word Clouds :

1. Word Clouds are not perfect for every situation.
2. Data should be optimized for context.

Conclusion :

Thus , we were successfully able to create a wordcloud and a frequency plot of the text corpus

In [1]:

```
import numpy as np
import pandas as pd
from os import path
from PIL import Image
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
import matplotlib.pyplot as plt
#matplotlib inline

# df1 = pd.read_json (r'C:\Users\Bhavesh\Desktop\nlp\1.json')
# df1.head()
# df1.to_csv (r'C:\Users\Bhavesh\Desktop\nlp\c_1.csv', index = None)

df = pd.read_csv(r'C:\Users\Bhavesh\Desktop\nlp\corona-virus-report\metadata.csv', in
dex_col=0)
df.head()
```

In [1]:

Out[1]:

	sha	source_x	title	doi	pmcid	p
cord_uid						
vho70jcx	f056da9c64fbf00a4645ae326e8a4339d015d155	biorxiv	SIANN: Strain Identification by Alignment to N...	10.1101/001727	NaN	
i9tbix2v	daf32e013d325a6feb80e83d15aabc64a48fae33	biorxiv	Spatial epidemiology of networked metapopulati...	10.1101/003889	NaN	
62gfisc6	f33c6d94b0efaa198f8f3f20e644625fa3fe10d2	biorxiv	Sequencing of the human IG light chain loci fr...	10.1101/006866	NaN	
058r9486	4da8a87e614373d56070ed272487451266dce919	biorxiv	Bayesian mixture analysis for metagenomic comm...	10.1101/007476	NaN	
wich35l7	eccef80cfbe078235df22398f195d5db462d8000	biorxiv	Mapping a viral phylogeny onto outbreak trees ...	10.1101/010389	NaN	

In [3]:
```
df[["title","license","abstract","journal"]].head()
```

Out[3]:

	title	license	abstract	journal
cord_uid				
vho70jcx	SIANN: Strain Identification by Alignment to N...	biorxiv	Next-generation sequencing is increasingly bei...	NaN
i9tbix2v	Spatial epidemiology of networked metapopulati...	biorxiv	An emerging disease is one infectious epidemic...	NaN
62gfisc6	Sequencing of the human IG light chain loci fr...	biorxiv	Germline variation at immunoglobulin gene (IG)...	NaN
058r9486	Bayesian mixture analysis for metagenomic comm...	biorxiv	Deep sequencing of clinical samples is now an ...	NaN
wich35l7	Mapping a viral phylogeny onto outbreak trees ...	biorxiv	Developing methods to reconstruct transmission...	NaN

In [12]:
```
l = df.groupby("license")
l=l[["has_full_text"]]
l.describe().head()
```

Out[12]:

	has_full_text			
	count	unique	top	freq
license				
biorxiv	631	2	True	557
cc-by	8858	2	True	8622
cc-by-nc	1160	2	True	1084
cc-by-nc-nd	668	2	True	610
cc-by-nc-sa	472	2	True	345

In [13]:
```
d=l.mean().sort_values(by="has_full_text",ascending=False)
d.head()
```

Out[13]:

	has_full_text
license	
cc-by-sa	1.000000
pd	1.000000
cc0	0.992308
cc-by	0.973357
cc-by-nc	0.934483

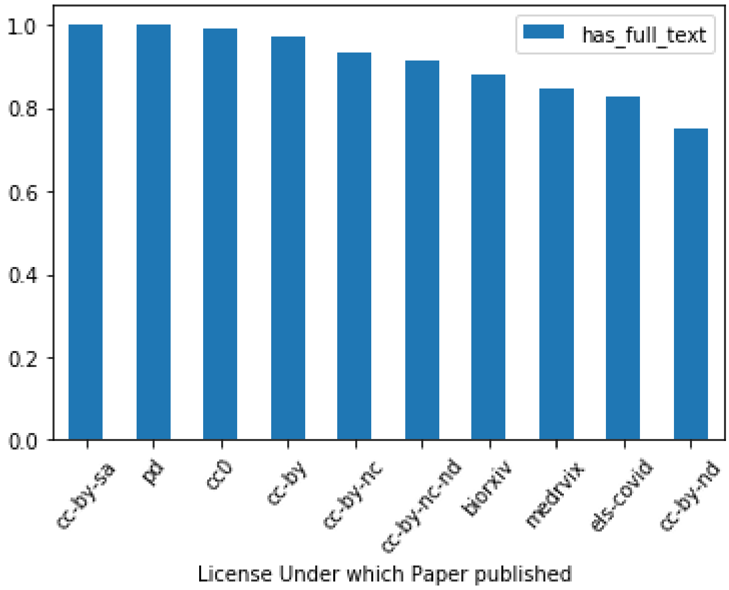
In [44]:

In [44]:

In [44]:

In [15]:
```
plt.figure(figsize=(200,200))
d.head(10).plot.bar()
plt.xticks(rotation=50)
plt.xlabel("License Under which Paper published ")
plt.ylabel("")
plt.show()
```

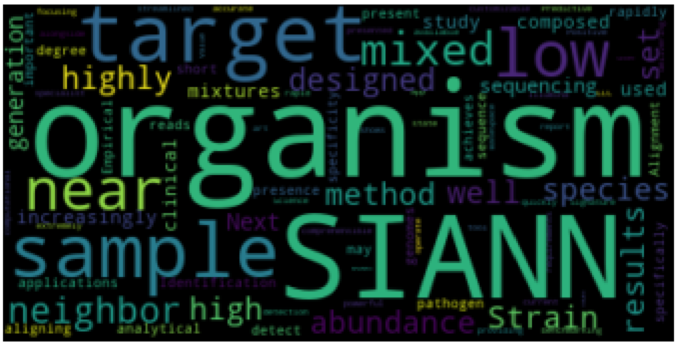
```
<Figure size 14400x14400 with 0 Axes>
```



In [27]:
```
text = df.abstract[0]
text
```

Out[27]:
```
'< g e n e r a t o r   o b j e c t   < g e n e x p r >   a t   0 x 0 0 0 0 0 2 1 0 2
3 2 1 C B 1 0 >'
```

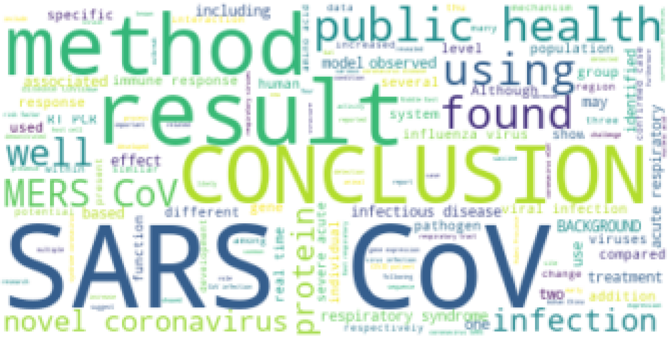
In [18]:
```
wordcloud = WordCloud().generate(text)
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```



In [ ]:
```
i=0
while i<1000 :
        row=df.abstract[i]
        text=text+str(row)
        i=i+1
print(done)
print ("There are {} words in the combination of 1000 rows of abstract.".format(len(t
ext)))
```

In [36]:

```
stopwords = set(STOPWORDS)
stopwords.update(["target", "organism", "sample", "strain", "mixed"])
wordcloud = WordCloud(stopwords=stopwords, background_color="white").generate(text)
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```



In [ ]: