The number of bee colonies in the US: Cyclical movement throughout the year

1. Introduction and aims

The main idea of this project is to explore the cyclical movement of the number of bee colonies in the US throughout the year. I will try to find interesting data that have correlations and give us new conclusions. The main goal is to find data that will provide valuable conclusions so that the beekeeping community can benefit from my findings.

In my research, I will use different techniques that I have learned in this module. One of the most important starting points is that I find good data for my research.

2. Ethical consideration

When it comes to the ethical aspect of using the data from third parties, none of the websites I used as a source of data has a clause in their terms and conditions that forbids downloading the data. All data that I gathered is allowed for public use. While collecting data, I kept in my mind all the important principles of data ethics, which are: ownership, transparency, privacy, intention, and outcomes.

3. Import libraries and modules

3.1 Libraries import chronologically

Here is the list of all library resources that I have used in the project:

import nltk

import matplotlib.pyplot as plt

from nltk.tokenize import word_tokenize

from nltk.corpus import stopwords

from nltk.stem import PorterStemmer

from nltk.stem import LancasterStemmer

from nltk.stem import WordNetLemmatizer

import pandas as pd

import matplotlib.pyplot as plt

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

4. Pre-processing Text

4.1 Classification

Since I did not find the exact data about bee colonies and their loss numbers, I will use similar data for this pre-procesing text part of the task.

In [1]:
```
# import of libraries
import nltk
import matplotlib.pyplot as plt
```

Here I have six stories. The three sories are related with the word 'bee', and three other stories are related with the word 'honey'. I have imported them here.

In [2]:
```
# import of data sets
stories = [""] * 6

# Three about bee
stories[0] = "Bees feed on nectar and pollen, the former primarily as an energy
stories[1] = "Bee pollination is important both ecologically and commercially, a
stories[2] = "Bees are winged insects closely related to wasps and ants, known f

# Three about honey
stories[3] = "Honey gets its sweetness from the monosaccharides fructose and glu
stories[4] = "Honey use and production have a long and varied history as an anci
stories[5] = "Honey is a sweet, viscous food substance made by honey bees and so
```

In [17]:
```
# Define a helper function to plot freq dist of words in a list of lists
def plotfreq(x, title=""):
    # Flatten list of lists
    flat = []
    for i in x:
        flat += i

    # Get freq distribution of tokens
    f = nltk.FreqDist(flat)

    # Plot freq dist
    plt.figure(figsize=(12,3))

    plt.title(title)
    f.plot()
```

4.2 Tokenize

Our first step is to tokenize the stories, splitting them up into words.

In [18]:
```
# import of libraries
from nltk.tokenize import word_tokenize

tokens = []
for story in stories:
    words = word_tokenize(story)
    tokens.append(words)
```

In [19]:
```
tokens
```

Out[19]:
```
[['Bees',
  'feed',
  'on',
  'nectar',
  'and',
  'pollen',
  ',',
  'the',
  'former',
  'primarily',
  'as',
  'an',
  'energy',
  'source',
  'and',
  'the',
  'latter',
  'primarily',
  'for',
  'protein',
  'and',
  'other',
  'nutrients',
  '.',
  'Most',
  'pollen',
  'is',
  'used',
  'as',
  'food',
  'for',
  'their',
  'larvae',
  '.',
  'Vertebrate',
  'predators',
  'of',
  'bees',
  'include',
  'primates',
  'and',
  'birds',
  'such',
  'as',
  'bee',
  'eaters',
  ';',
  'insect',
  'predators',
  'include',
  'beewolves',
  'and',
  'dragonflies',
  '.'],
 ['Bee',
  'pollination',
  'is',
  'important',
  'both',
  'ecologically',
  'and',
  'commercially',
  ',',
```

```
 'and',
 'the',
 'decline',
 'in',
 'wild',
 'bees',
 'has',
 'increased',
 'the',
 'value',
 'of',
 'pollination',
 'by',
 'commercially',
 'managed',
 'hives',
 'of',
 'honey',
 'bees',
 '.',
 'The',
 'analysis',
 'of',
 '353',
 'wild',
 'bee',
 'and',
 'hoverfly',
 'species',
 'across',
 'Britain',
 'from',
 '1980',
 'to',
 '2013',
 'found',
 'the',
 'insects',
 'have',
 'been',
 'lost',
 'from',
 'a',
 'quarter',
 'of',
 'the',
 'places',
 'they',
 'inhabited',
 'in',
 '1980',
 '.'],
['Bees',
 'are',
 'winged',
 'insects',
 'closely',
 'related',
 'to',
 'wasps',
 'and',
 'ants',
 ',',
```

```
 'known',
 'for',
 'their',
 'role',
 'in',
 'pollination',
 'and',
 ',',
 'in',
 'the',
 'case',
 'of',
 'the',
 'best',
 'known',
 'bee',
 'species',
 ',',
 'the',
 'western',
 'honey',
 'bee',
 ',',
 'for',
 'producing',
 'honey',
 '.',
 'Bees',
 'are',
 'a',
 'monophyletic',
 'lineage',
 'within',
 'the',
 'superfamily',
 'Apoidea',
 '.',
 'They',
 'are',
 'presently',
 'considered',
 'a',
 'clade',
 ',',
 'called',
 'Anthophila',
 '.',
 'There',
 'are',
 'over',
 '16,000',
 'known',
 'species',
 'of',
 'bees',
 'in',
 'seven',
 'recognized',
 'biological',
 'families',
 '.'],
['Honey',
 'gets',
```

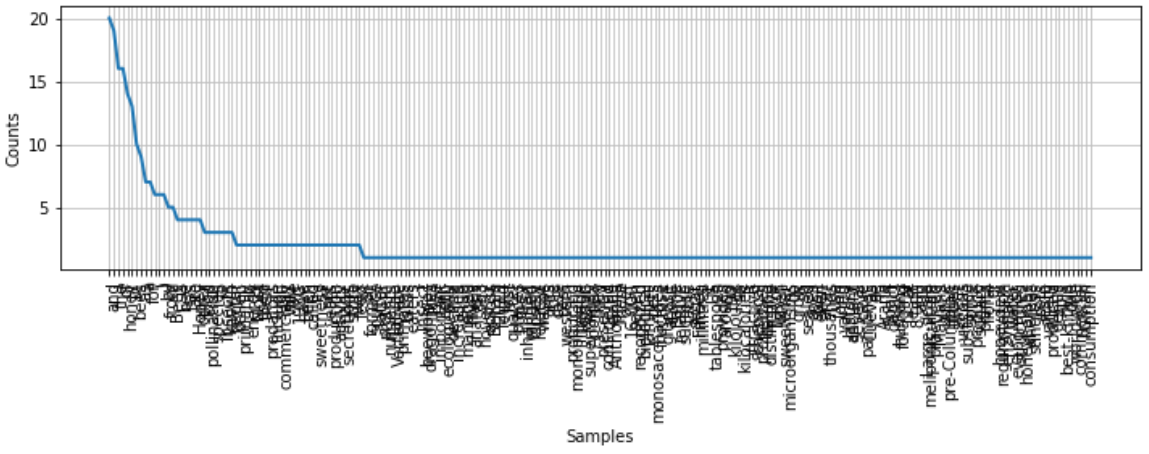
```
'its',
'sweetness',
'from',
'the',
'monosaccharides',
'fructose',
'and',
'glucose',
',',
'and',
'has',
'about',
'the',
'same',
'relative',
'sweetness',
'as',
'sucrose',
'(',
'table',
'sugar',
')',
'.',
'Fifteen',
'millilitres',
'(',
'1',
'US',
'tablespoon',
')',
'of',
'honey',
'provides',
'around',
'190',
'kilojoules',
'(',
'46',
'kilocalories',
')',
'of',
'food',
'energy',
'.',
'It',
'has',
'attractive',
'chemical',
'properties',
'for',
'baking',
'and',
'a',
'distinctive',
'flavor',
'when',
'used',
'as',
'a',
'sweetener',
'.',
'Most',
'microorganisms',
```

```
 'do',
 'not',
 'grow',
 'in',
 'honey',
 ',',
 'so',
 'sealed',
 'honey',
 'does',
 'not',
 'spoil',
 ',',
 'even',
 'after',
 'thousands',
 'of',
 'years',
 '.'],
['Honey',
 'use',
 'and',
 'production',
 'have',
 'a',
 'long',
 'and',
 'varied',
 'history',
 'as',
 'an',
 'ancient',
 'activity',
 '.',
 'Several',
 'cave',
 'paintings',
 'in',
 'Cuevas',
 'de',
 'la',
 'Araña',
 'in',
 'Spain',
 'depict',
 'humans',
 'foraging',
 'for',
 'honey',
 'at',
 'least',
 '8,000',
 'years',
 'ago',
 '.',
 'Large-scale',
 'meliponiculture',
 'has',
 'been',
 'practiced',
 'by',
 'the',
 'Mayans',
```

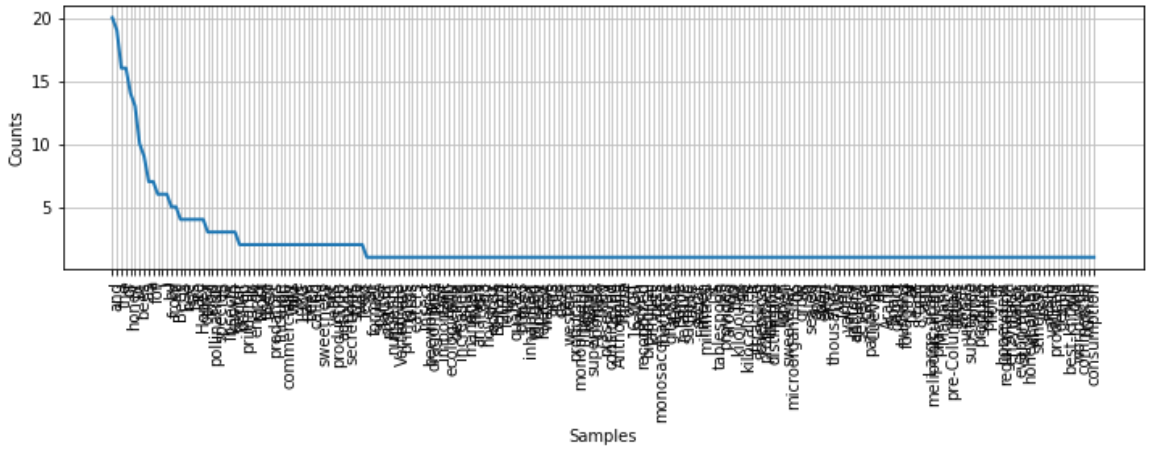
```
 'since',
 'pre-Columbian',
 'times',
 '.'],
['Honey',
 'is',
 'a',
 'sweet',
 ',',
 'viscous',
 'food',
 'substance',
 'made',
 'by',
 'honey',
 'bees',
 'and',
 'some',
 'other',
 'bees',
 '.',
 'Bees',
 'produce',
 'honey',
 'from',
 'the',
 'sugary',
 'secretions',
 'of',
 'plants',
 '(',
 'floral',
 'nectar',
 ')',
 'or',
 'from',
 'secretions',
 'of',
 'other',
 'insects',
 '(',
 'such',
 'as',
 'honeydew',
 ')',
 ',',
 'by',
 'regurgitation',
 ',',
 'enzymatic',
 'activity',
 ',',
 'and',
 'water',
 'evaporation',
 '.',
 'Honey',
 'bees',
 'store',
 'honey',
 'in',
 'wax',
 'structures',
```

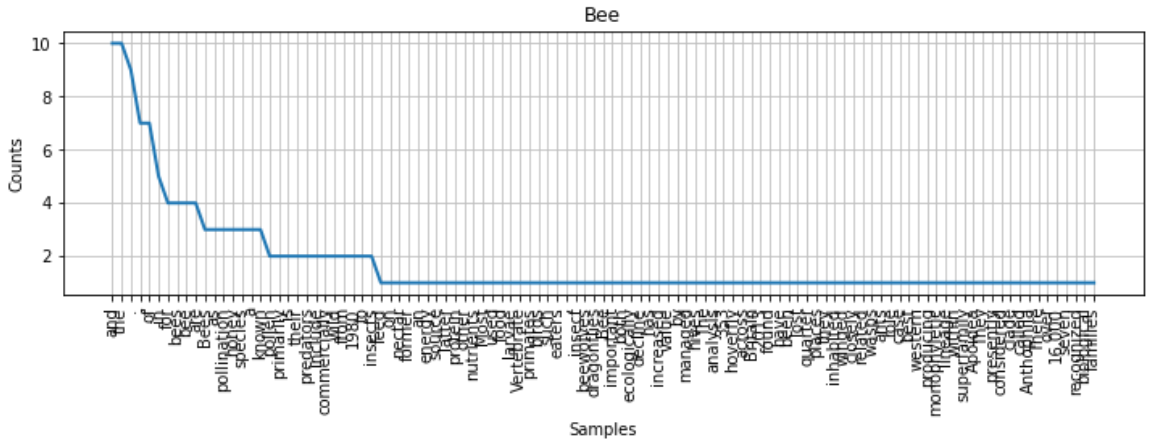
```
 'called',
 'honeycombs',
 ',',
 'whereas',
 'stingless',
 'bees',
 'store',
 'honey',
 'in',
 'pots',
 'made',
 'of',
 'wax',
 'and',
 'resin',
 '.',
 'The',
 'variety',
 'of',
 'honey',
 'produced',
 'by',
 'honey',
 'bees',
 '(',
 'the',
 'genus',
 'Apis',
 ')',
 'is',
 'the',
 'best-known',
 ',',
 'due',
 'to',
 'its',
 'worldwide',
 'commercial',
 'production',
 'and',
 'human',
 'consumption',
 '.']]
```

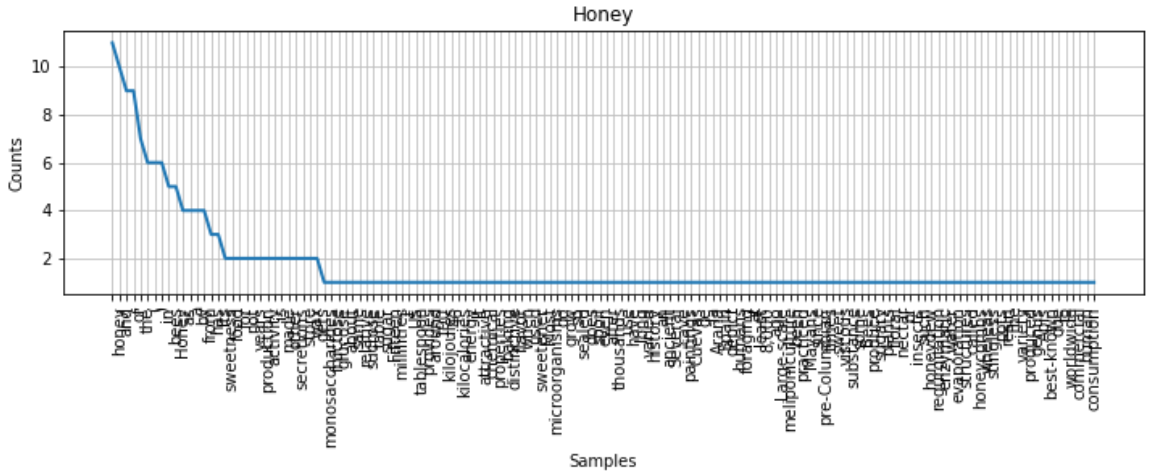
In [20]:
```
# plot of tokens
plotfreq(tokens)
```



In [21]:
```
plotfreq(tokens)
plotfreq(tokens[0:3], "Bee") # bee
plotfreq(tokens[3:6], "Honey") # honey
```







The aim here is to see the most frequent words and how high the 'bee' and 'honey' words are. It can be said that the higher they are, the more frequent they are.

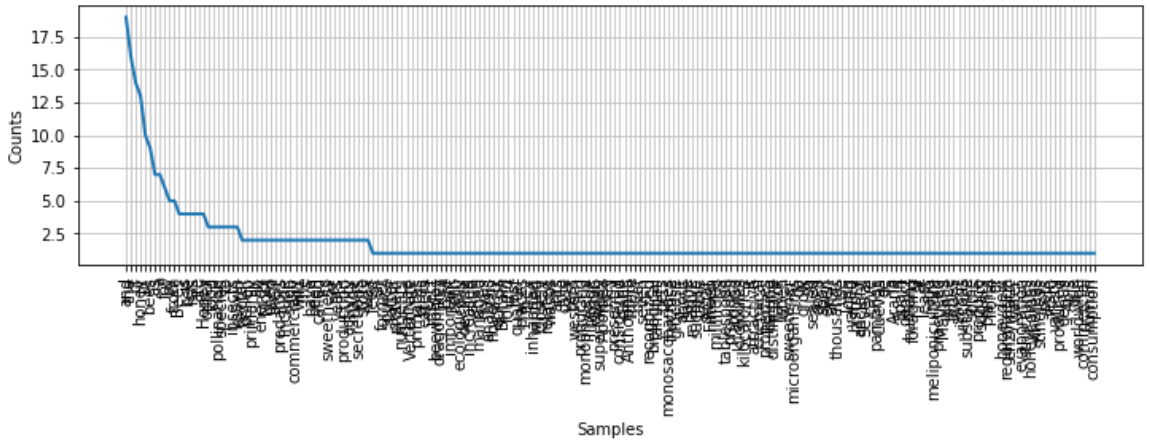
4.3 Remove punctuation

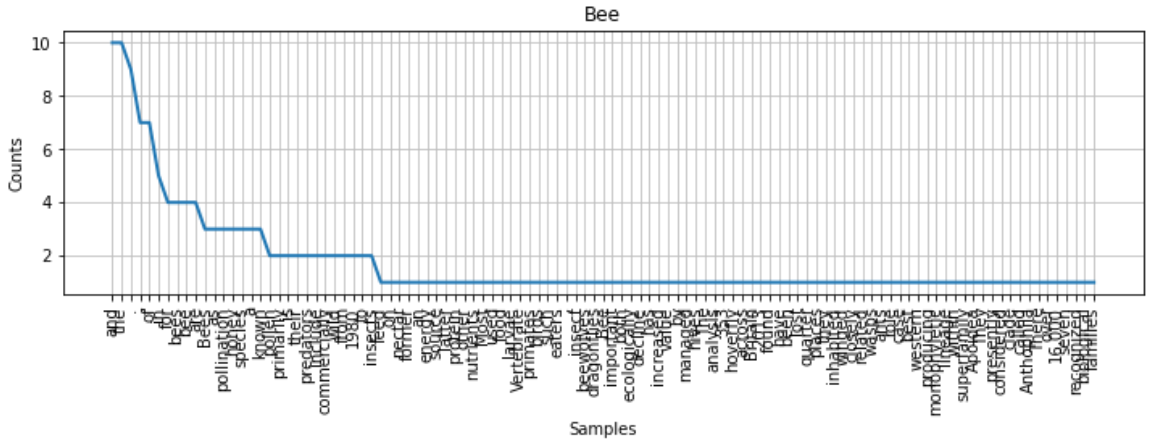
Let's first remove the punctuation.

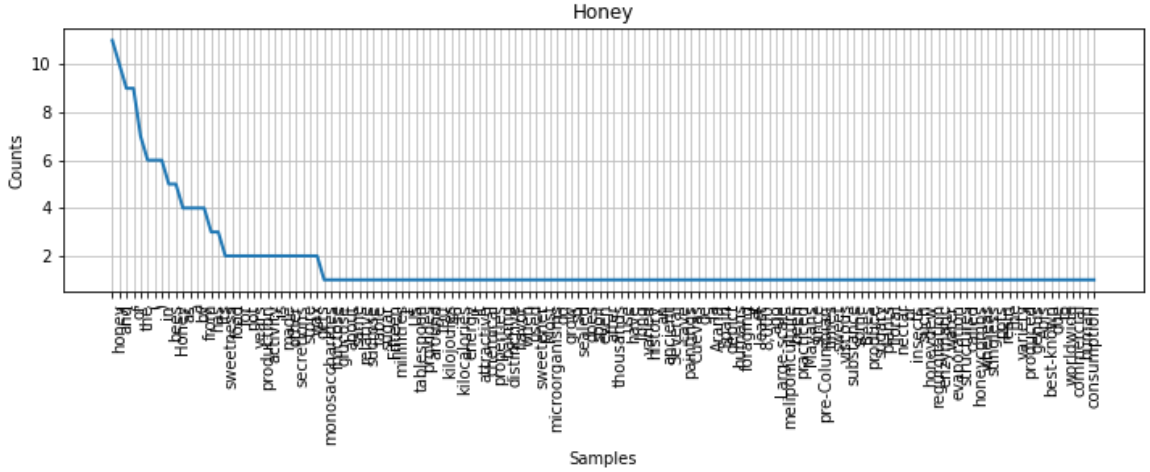
In [33]:
```
# removing punctuation
cleaned_tokens = []
for t in tokens:
    cleaned = [word for word in t if word.isalpha()]
    cleaned_tokens.append(cleaned)
```

The word frequency plots are now cleaner:

In [34]:
```
plotfreq(cleaned_tokens)
plotfreq(tokens[0:3], "Bee") # bee
plotfreq(tokens[3:6], "Honey") # honey
```







I tried to get better data, but noise is still too big. I will try with the next process.

4.4 Remove stopwords

Now let's remove the stopwords. These are common English words that have little information value in natural language understanding. Similar stop word lists exist in other languages (just substitute the language string).

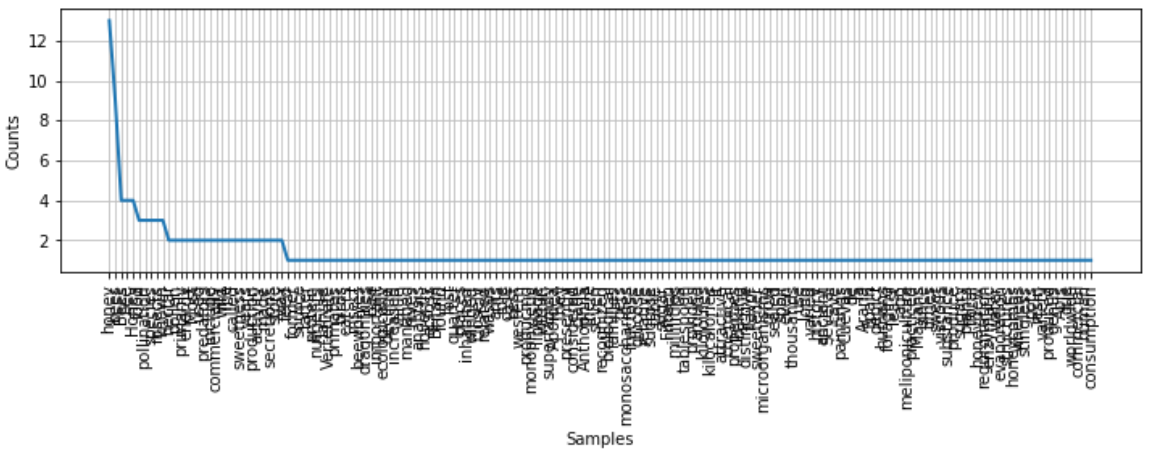
In [35]:
```
# import of libraries
from nltk.corpus import stopwords
stop_words = stopwords.words('english')
print(stop_words)
```

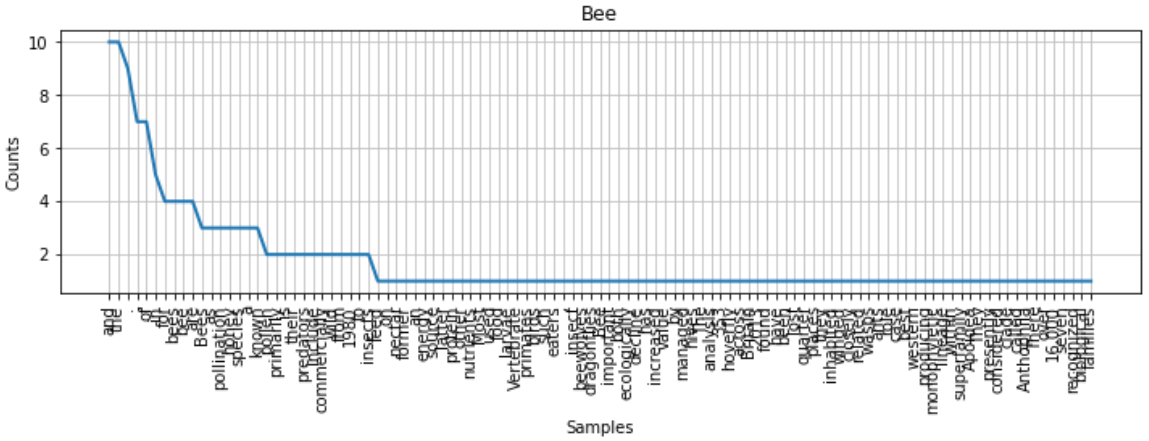
```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "
you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'hi
m', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'i
ts', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which',
'who', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', '
was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does
', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as',
'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between
', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from
', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further',
'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'b
oth', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not
', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will',
'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're',
've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doe
sn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "
isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'sha
n', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won
', "won't", 'wouldn', "wouldn't"]
```

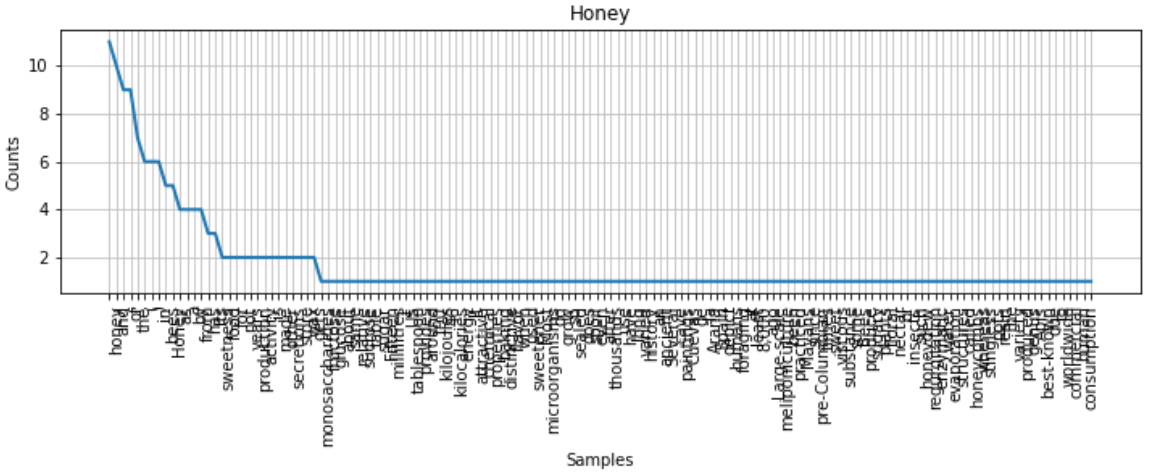
In [36]:
```
# removing the stopwords
stopwords_removed_tokens = []
for t in cleaned_tokens:
    cleaned = [word for word in t if not word in stop_words]
    stopwords_removed_tokens.append(cleaned)
```

Now looking at the word frequencies, we are starting to see more meaningful words appear at the top. However, for the bee stories, the bee-related words are appearing with low frequencies so are not being considered as very meaningful.

In [37]:
```
plotfreq(stopwords_removed_tokens)
plotfreq(tokens[0:3], "Bee") # bee
plotfreq(tokens[3:6], "Honey") # honey
```







The data from previous step and this step is not very different. I can say that my stories do not have many stopwords.

4.5 Stem

I can start to address the issue above where the bee-related words are not appearing at the top of the list, by using stemming I will try to gat better results.

Let's stem all words and then do the frequency count.

In [38]:
```
# import of libraries
from nltk.stem import PorterStemmer
ps = PorterStemmer()
stemmed_tokens = []
for t in stopwords_removed_tokens:
    cleaned = [ps.stem(word) for word in t]
    stemmed_tokens.append(cleaned)
```

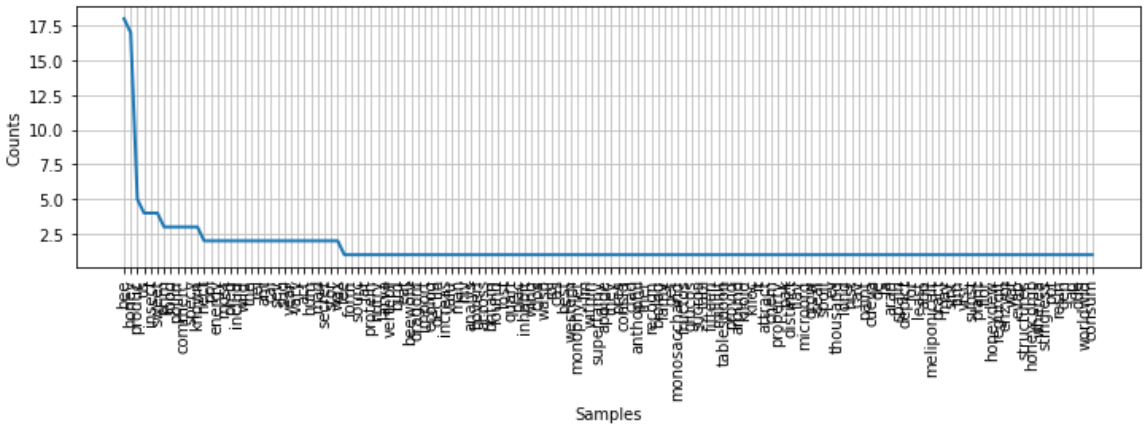
First I stem the words with PorterStemmer. Then I checked the plots.

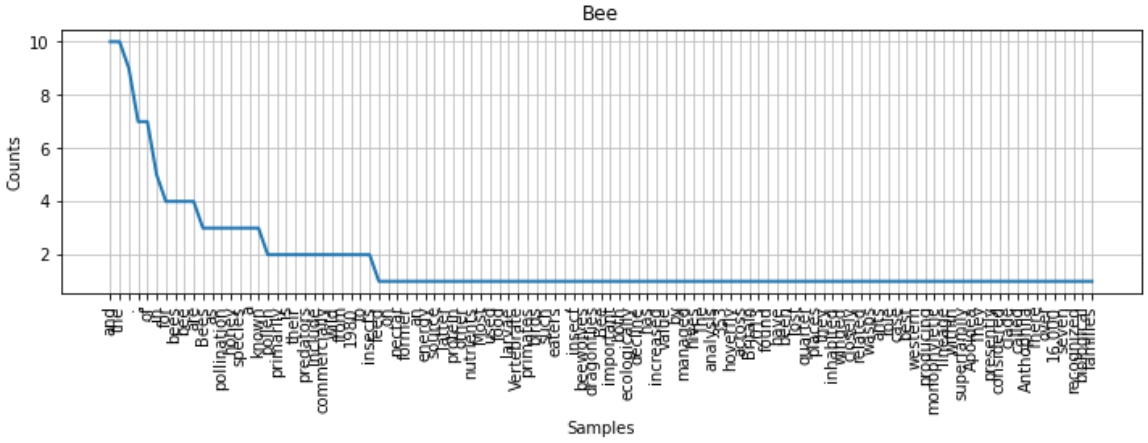
The PorterStemmer did not give me the best results. Let's try with LancasterStemmer, it is sharper.

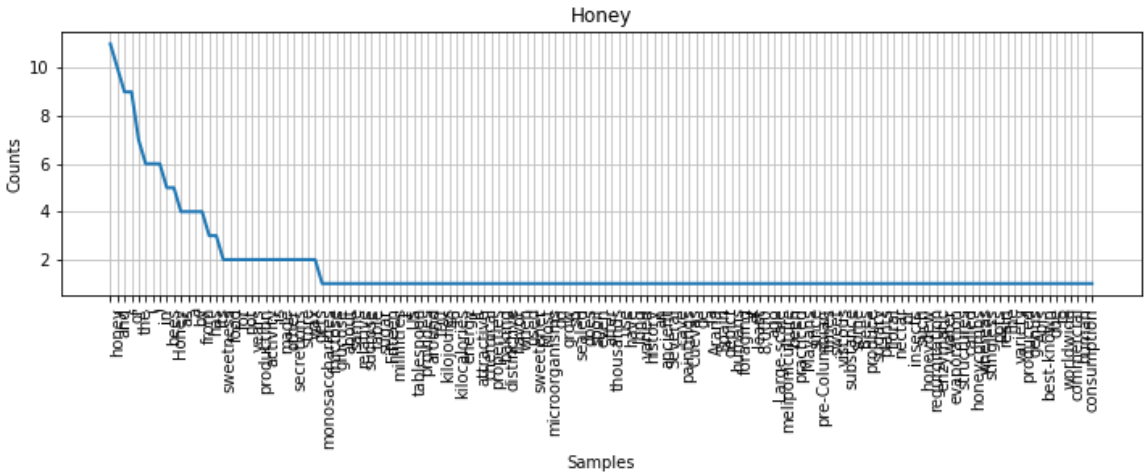
In [39]:
```
# import of libraries
from nltk.stem import LancasterStemmer
ls = LancasterStemmer()
stemmed_tokens = []
for t in stopwords_removed_tokens:
    cleaned = [ls.stem(word) for word in t]
    stemmed_tokens.append(cleaned)
```

Now I will plot again after LancasterStemmer.

In [40]:
```
plotfreq(stemmed_tokens)
plotfreq(tokens[0:3], "Bee") # bee
plotfreq(tokens[3:6], "Honey") # honey
```







After first stemming of the words with PorterStemmer, we can see that our words are moving to the left.

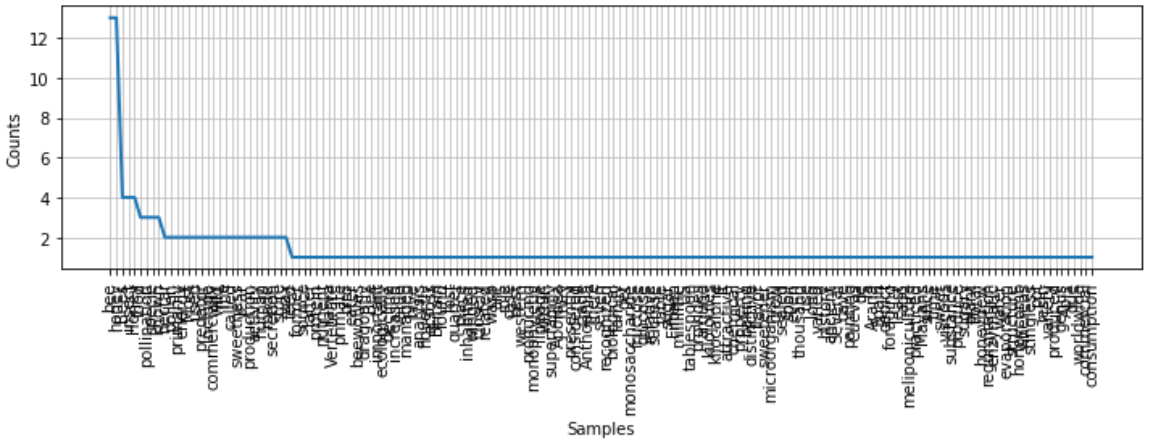
With the second stemming we got better results. Bee word is closer to the top, and honey word is in the number one position.

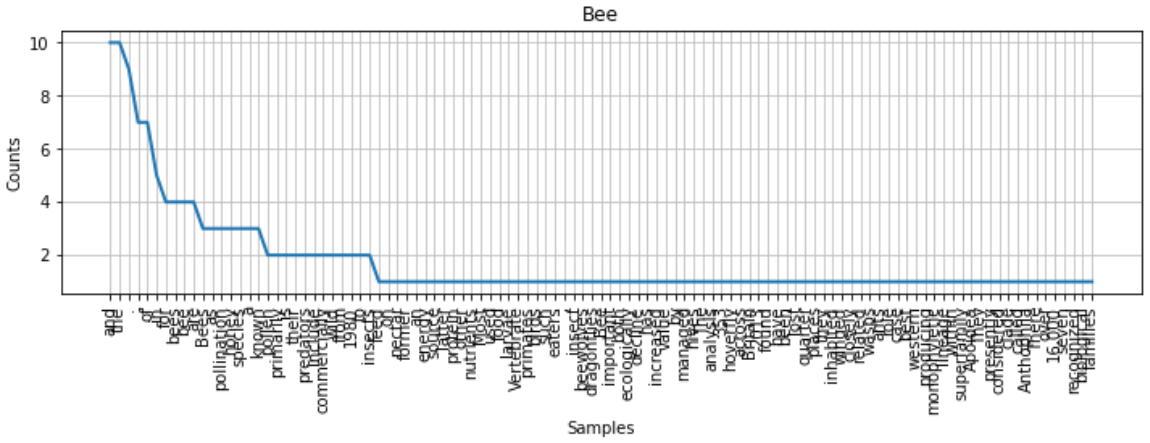
4.6 Lemmatization

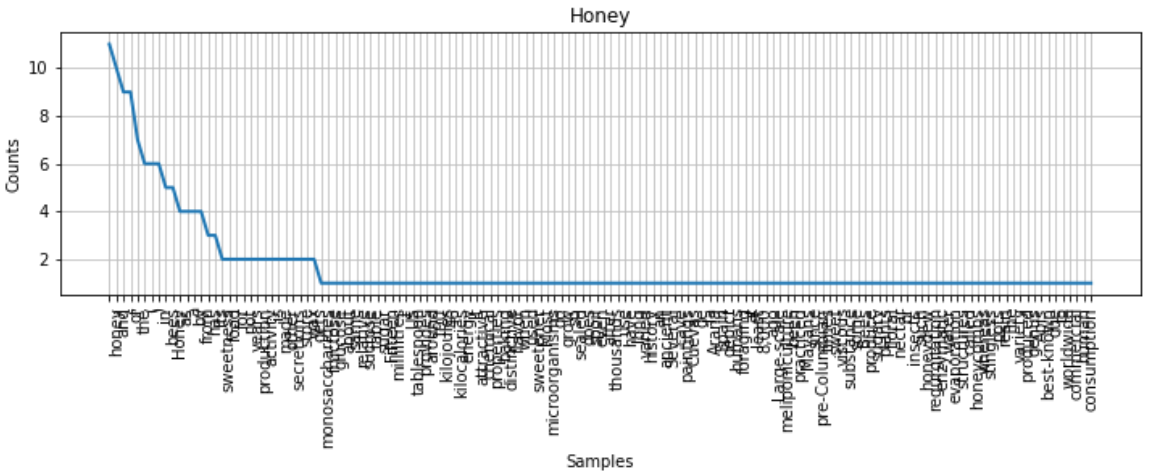
I will do the lemmatization with my data sets, and I will try to get better results especially for 'bee' word.

In [41]:
```
# import of libraries and lemmatization
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
lemmatized_tokens = []
for t in stopwords_removed_tokens:
    cleaned = [lemmatizer.lemmatize(word) for word in t]
    lemmatized_tokens.append(cleaned)
```

In [42]:
```
plotfreq(lemmatized_tokens)
plotfreq(tokens[0:3], "Bee") # bee
plotfreq(tokens[3:6], "Honey") # honey
```







Does lemmatization give us better rules? No, not for this classifcation task! But maybe if I had more stories it would perform better.

Summary

Bee

For the bee word I did get good results but maybe not the best. For the best results I should probably use better filters from the libraries and bigger sample - more stories.

Honey

For honey word I can say that I got much better results than for bee word. My honey word is on the top. That means that it is the most frequent word.

5. Exploratory data analysis

Importing data set, and checking if it is okay.

In [2]:
```
# import pandas
import pandas as pd

# import csv
df = pd.read_csv("Bee-Colony-Data-USDA-(No US Totals).csv")
```

I can see how my data set looks like, and how I will use the data. I will check first ten rows with all columns.

In [3]:
```
# First 10 rows
df.head(10)
```

Out[3]:

	LineNum	State	Colonies at start of Month Range	Maximum Colonies	Lost Colonies	Percent Lost Colonies	Added Colonies	Renovated Colonies	Per Renov
0	0	Alabama	5500	5500	650	12	800	200	
1	1	Arizona	22000	22000	2500	11	430	90	
2	2	Arkansas	28000	28000	6500	23	20	20	
3	3	California	1140000	1580000	235000	15	83000	86000	
4	4	Colorado	5000	7500	320	4	0	0	
5	5	Connecticut	3100	3100	270	9	170	10	
6	6	Florida	300000	315000	46000	15	41000	16500	
7	7	Georgia	120000	129000	14500	11	19500	8000	
8	8	Hawaii	16500	16500	390	2	660	3700	
9	9	Idaho	132000	145000	12500	9	4600	1100	

In [4]:
```
# We can pick some arbitrary slices to explore at more depth.

df[26:39]
```

Out[4]:

	LineNum	State	Colonies at start of Month Range	Maximum Colonies	Lost Colonies	Percent Lost Colonies	Added Colonies	Renovated Colonies	R Ren
26	26	New Mexico	4800	4800	170	4	0	0	
27	27	New York	24000	24000	3400	14	110	80	
28	28	North Carolina	10500	14500	1700	12	750	890	
29	29	North Dakota	92000	102000	1100	1	0	1300	
30	30	Ohio	10500	12500	3500	28	1700	440	
31	31	Oklahoma	21000	55000	50	0	20	0	
32	32	Oregon	89000	95000	7500	8	8000	1800	
33	33	Pennsylvania	15000	15000	2900	19	370	320	
34	34	South Carolina	12000	13000	690	5	2000	340	
35	35	South Dakota	18500	18500	1100	6	0	0	
36	36	Tennessee	6000	7000	1500	21	240	320	
37	37	Texas	260000	330000	17000	5	61000	35000	
38	38	Utah	8500	11000	1800	16	900	0	

After examinations I can see that my data is well structured. Now I will try to find interesting data points so I can compare them later.

In [88]:
```
# maximum of maximum colonies in month range
df["Maximum Colonies "].max()
```

Out[88]:
```
1710000
```

We can say that beekeeping in the US is a good scaled business, since there are good examples of beekeepers who evolved.

In [89]:
```
# minimum of maximum colonies in month range
df["Maximum Colonies "].min()
```

Out[89]:
```
1700
```

In [90]:
```
# maximum of added colonies in month range
df["Added Colonies"].max()
```

Out[90]:
```
240000
```

In [91]:
```
# minimum of added colonies in month range
df["Added Colonies"].min()
```

Out[91]:
```
0
```

In [92]:
```
# maximum of colonies at start of month range
df["Colonies at start of Month Range"].max()
```

Out[92]:
```
1350000
```

In [93]:
```
# minimum of colonies at start of month range
df["Colonies at start of Month Range"].min()
```

Out[93]:
```
1300
```

In [94]:
```
# mean of maximum colonies
df["Maximum Colonies "].mean()
```

Out[94]:
```
79582.56521739131
```

In [95]:
```
# mean of added colonies
df["Added Colonies"].mean()
```

Out[95]:
```
7838.0
```

In [96]:
```
# mean of colonies at start of month range
df["Colonies at start of Month Range"].mean()
```

Out[96]:
```
61826.30434782609
```

After I examinated max, min, mean values of different categories in the columns, I will now compare some of these values. I hope that this data will give me some new points of views.

In [97]:
```
# standard deviation of lost colonies
df["Lost Colonies"].std()
```

Out[97]:
```
28656.765729561714
```

In [98]:
```
# standard deviation of added colonies
df["Added Colonies"].std()
```

Out[98]:
```
25682.41563479627
```

Here above I tried to get some additional numbers on standard deviation of lost colonies and added colonies.

In [100…
```
# describe will relate other values in table with colonies at start of month ran
df["Colonies at start of Month Range"].describe()
```

Out[100]:
```
count    2.300000e+02
mean     6.182630e+04
std      1.723093e+05
min      1.300000e+03
25%      7.000000e+03
50%      1.575000e+04
75%      3.625000e+04
max      1.350000e+06
Name: Colonies at start of Month Range, dtype: float64
```

In [101…
```
# describe will relate other values in table with renovated colonies
df["Renovated Colonies"].describe()
```

Out[101]:
```
count       230.000000
mean       5374.217391
std       20903.064944
min           0.000000
25%          60.000000
50%         340.000000
75%        1800.000000
max      240000.000000
Name: Renovated Colonies, dtype: float64
```

Here I have some general numbers on my data set. I am still struggling to find the best samples/columns for comparisson.

In [102…
```
# as 2D representation, values names in rows and columns 230x11
df.shape
```

Out[102]:
```
(230, 11)
```

In [103…
```
# in short, the complete table
df
```

Out[103]:

	LineNum	State	Colonies at start of Month Range	Maximum Colonies	Lost Colonies	Percent Lost Colonies	Added Colonies	Renovated Colonies	Rer
0	0	Alabama	5500	5500	650	12	800	200	
1	1	Arizona	22000	22000	2500	11	430	90	
2	2	Arkansas	28000	28000	6500	23	20	20	
3	3	California	1140000	1580000	235000	15	83000	86000	
4	4	Colorado	5000	7500	320	4	0	0	
...	...	...	...	...	...	...	...	...	
225	229	Washington	50000	114000	3100	3	5000	3300	
226	230	West Virginia	7500	7500	570	8	1600	1500	
227	231	Wisconsin	27000	53000	1700	3	11500	4600	
228	232	Wyoming	17500	24000	1600	7	2300	2100	
229	233	Other States 5/	6500	7440	250	3	1270	1500	

230 rows × 11 columns

In [104…

```
# with this function we can se the states, by name
df["State"]
```

Out[104]:

```
0              Alabama
1              Arizona
2             Arkansas
3           California
4             Colorado
             ...
225         Washington
226      West Virginia
227          Wisconsin
228            Wyoming
229    Other States 5/
Name: State, Length: 230, dtype: object
```

In [107…

```
# this way we can sort data to see from high to low the percent renovated
df.sort_values(by=['Percent Renovated'], ascending=False)
```

Out[107]:

	LineNum	State	Colonies at start of Month Range	Maximum Colonies	Lost Colonies	Percent Lost Colonies	Added Colonies	Renovated Colonies	Pe Reno
222	226	Utah	21000	31000	2700	9	7000	13500	
153	156	Louisiana	44000	48000	4800	10	9000	20000	
193	197	Idaho	48000	81000	14500	18	32000	29000	
221	225	Texas	360000	385000	25000	6	75000	136000	
198	202	Kentucky	11000	11000	1200	11	4200	3500	
...	...	...	...	...	...	...	...	...	
176	179	Utah	12000	17000	400	2	4200	0	
45	45	Other States 5/	6070	6070	590	10	70	30	
173	176	South Dakota	20000	20000	1200	6	20	0	
123	125	Oklahoma	6500	37000	2500	7	460	0	
39	39	Vermont	6500	6500	260	4	50	0	

230 rows × 11 columns

In [108…

```
# the threshold values of maximum colonies of more than 1M
df[df["Maximum Colonies "] > 300000]
```

Out[108]:

	LineNum	State	Colonies at start of Month Range	Maximum Colonies	Lost Colonies	Percent Lost Colonies	Added Colonies	Renovated Colonies	Pe Renov
3	3	California	1140000	1580000	235000	15	83000	86000	
6	6	Florida	300000	315000	46000	15	41000	16500	
37	37	Texas	260000	330000	17000	5	61000	35000	
49	50	California	1140000	1580000	235000	15	83000	86000	
52	53	Florida	300000	315000	46000	15	41000	16500	
83	84	Texas	260000	330000	17000	5	61000	35000	
95	97	California	700000	1300000	136000	10	160000	37000	
121	123	North Dakota	420000	420000	27000	6	3000	4300	
141	144	California	1350000	1710000	230000	13	240000	66000	
144	147	Florida	295000	310000	30000	10	41000	8000	
175	178	Texas	235000	340000	35000	10	83000	24000	
187	191	California	1200000	1200000	74000	6	184000	240000	
190	194	Florida	325000	325000	39000	12	36000	23000	
213	217	North Dakota	67000	395000	17500	4	29000	37000	
221	225	Texas	360000	385000	25000	6	75000	136000	

I did not find the category that has some big deviations within itself. Let's try with the amalgamations of insights, and maybe I will get some interesting outputs.

In [109…

```
# amalgamations of things,to get the most interest insights
#filtering by two criteriums
df.loc[(df["Maximum Colonies "] > 500000) & (df["Percent Lost Colonies"] < 15)]
```

Out[109]:

	LineNum	State	Colonies at start of Month Range	Maximum Colonies	Lost Colonies	Percent Lost Colonies	Added Colonies	Renovated Colonies	Pe Renov
95	97	California	700000	1300000	136000	10	160000	37000	
141	144	California	1350000	1710000	230000	13	240000	66000	
187	191	California	1200000	1200000	74000	6	184000	240000	

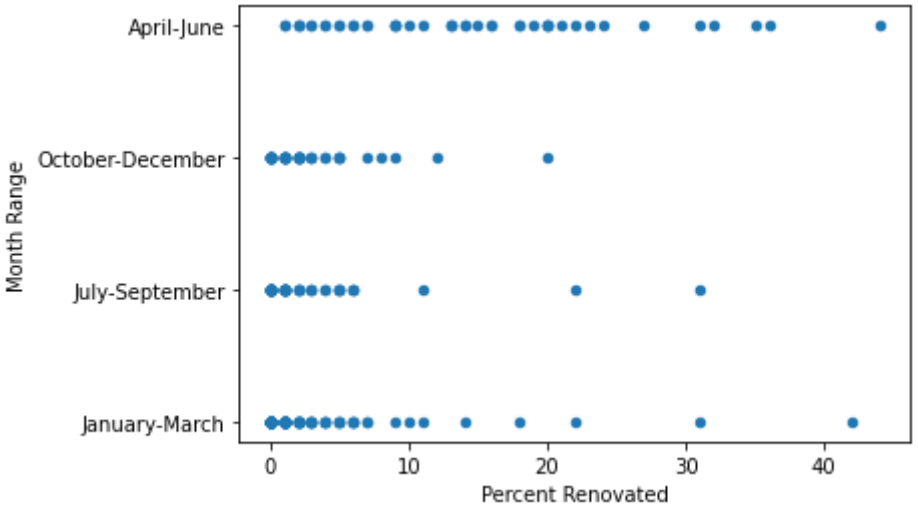
Here I can see something interesting with data. The California is a state with highest amount of bee colonies and the lowest percentage of the lost colonies. This means that California is probably one of the best states for beekeeping, and this conclusion is valuable information for anyone who is in beekeeping business. This also means that bees likes California a lot.

In [110…

```
# we have two parameters to plot scatter graph
df.plot.scatter(x = 'Percent Renovated', y = 'Month Range')
```

Out[110]:

```
<AxesSubplot:xlabel='Percent Renovated', ylabel='Month Range'>
```

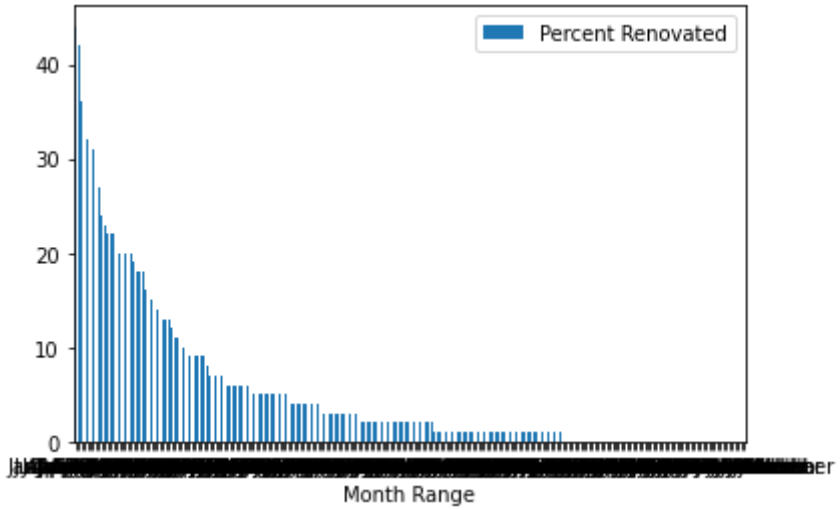


In [111…

```
df.sort_values(by='Percent Renovated', ascending=False).plot(kind="bar", rot=0,
```

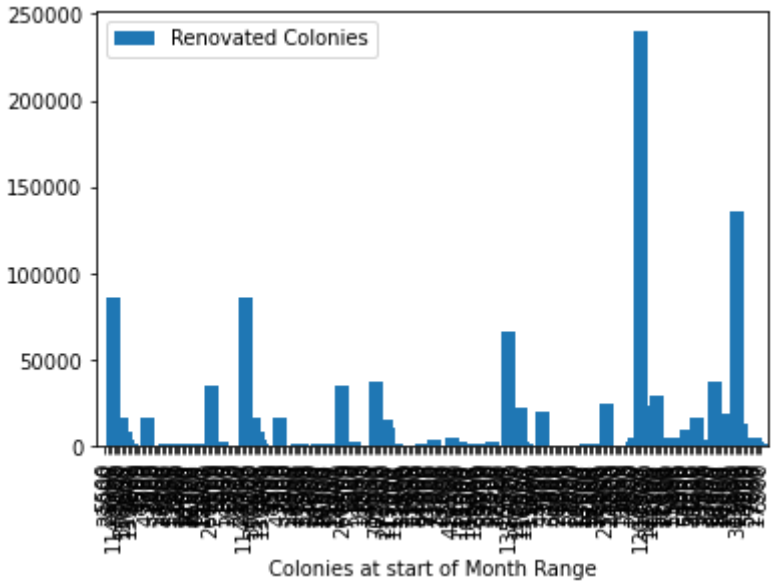
Out[111]:

```
<AxesSubplot:xlabel='Month Range'>
```



In [119…
```
# pandas.DataFrame.plot #many options for ploting
df.plot.bar(width=4.9,x='Colonies at start of Month Range',y="Renovated Colonies
```

Out[119]:
```
<AxesSubplot:xlabel='Colonies at start of Month Range'>
```



I treid to present some data with three different plots, but niether of them are great representation of my data. I will try with other graphs in next section.

Summary

First I have imported the data set that I found online. Then I uploaded and examined the data that I have. I calculated some values from the different data categories, and the most significant conclusion that I had was the state in the US that is maybe the best for beekeeping business when it comes to losses of bee colonies.

6. Data examination and plotting

In this section I will try to represent data with more complex ploting. The end result should be that I get some additional and interesting findings based on my data, but with more sophisticated data representation tools.

Let's see our data set.

In [5]:
```
df
```

Out[5]:

	LineNum	State	Colonies at start of Month Range	Maximum Colonies	Lost Colonies	Percent Lost Colonies	Added Colonies	Renovated Colonies	Ren
0	0	Alabama	5500	5500	650	12	800	200	
1	1	Arizona	22000	22000	2500	11	430	90	
2	2	Arkansas	28000	28000	6500	23	20	20	
3	3	California	1140000	1580000	235000	15	83000	86000	
4	4	Colorado	5000	7500	320	4	0	0	
...	...	...	...	...	...	...	...	...	
225	229	Washington	50000	114000	3100	3	5000	3300	
226	230	West Virginia	7500	7500	570	8	1600	1500	
227	231	Wisconsin	27000	53000	1700	3	11500	4600	
228	232	Wyoming	17500	24000	1600	7	2300	2100	
229	233	Other States 5/	6500	7440	250	3	1270	1500	

230 rows × 11 columns

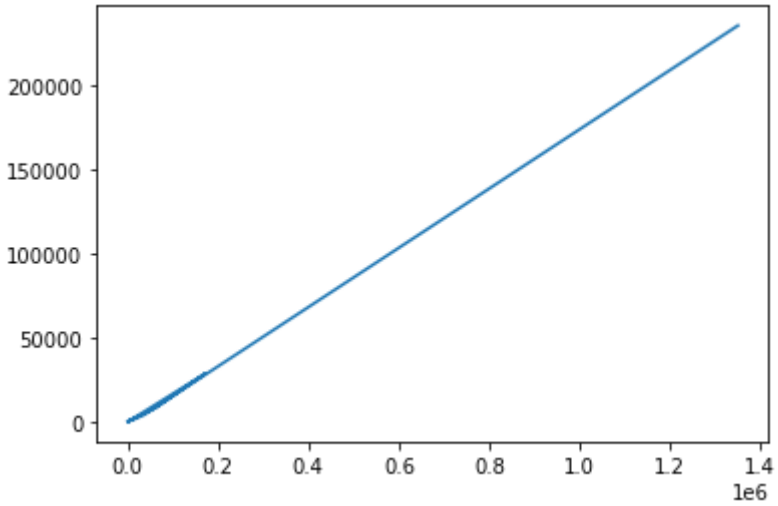
First I will go with the simple plot chart.

In [16]:
```
# Import the library
import matplotlib.pyplot as plt

# Prepare the data
x = df["Colonies at start of Month Range"].describe()
y = df["Lost Colonies"].describe()

# Plot the data (by default it plots a line chart)
plt.plot(x, y)

# Show the plot
plt.show()
```



This straight line is showing us consistency. Whitch implicates that beekeepers are doing a good job. Since we can see that they are fulfilling their losses in the equivalent ratio.

Here I will use more diverse chart.

In [22]:
```
# Load numpy and pandas for list and data set manipulation
import numpy as np
import pandas as pd

# Load matplotlib and seaborn for data visualisation
import matplotlib.pyplot as plt
import seaborn as sns
```

In [23]:
```
# Set style for Seaborn
sns.set(style="ticks", color_codes=True)
```

In [24]:
```
bees_data = pd.read_csv('Bee-Colony-Data-USDA-(No US Totals).csv')
```

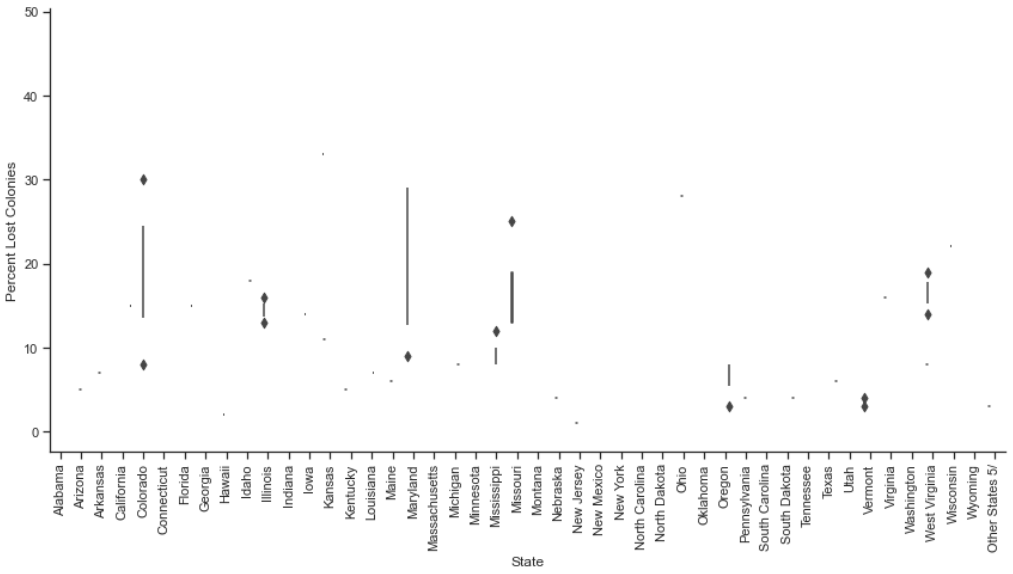
In [25]:
```
bees_data
```

Out[25]:

	LineNum	State	Colonies at start of Month Range	Maximum Colonies	Lost Colonies	Percent Lost Colonies	Added Colonies	Renovated Colonies	Ren
0	0	Alabama	5500	5500	650	12	800	200	
1	1	Arizona	22000	22000	2500	11	430	90	
2	2	Arkansas	28000	28000	6500	23	20	20	
3	3	California	1140000	1580000	235000	15	83000	86000	
4	4	Colorado	5000	7500	320	4	0	0	
...	...	...	...	...	...	...	...	...	
225	229	Washington	50000	114000	3100	3	5000	3300	
226	230	West Virginia	7500	7500	570	8	1600	1500	
227	231	Wisconsin	27000	53000	1700	3	11500	4600	
228	232	Wyoming	17500	24000	1600	7	2300	2100	
229	233	Other States 5/	6500	7440	250	3	1270	1500	

230 rows × 11 columns

In [30]:
```
sns.catplot(kind="boxen", x="State", y="Percent Lost Colonies", hue="Maximum Col
plt.xticks(rotation=90)
plt.show()
```


Percent Lost Colonies
50
40
30
20
10
0
Alabama
Arizona
Arkansas
California
Colorado
Connecticut
Florida
Georgia
Hawaii
Idaho
Illinois
Indiana
Iowa
Kansas
Kentucky
Louisiana
Maine
Maryland
Massachusetts
Michigan
Minnesota
Mississippi
Missouri
Montana
Nebraska
New Jersey
New Mexico
New York
North Carolina
North Dakota
Ohio
Oklahoma
Oregon
Pennsylvania
South Carolina
South Dakota
Tennessee
Texas
Utah
Vermont
Virginia
Washington
West Virginia
Wisconsin
Wyoming
Other States 5/
State
Maximum Colonies
1700
3100
3800
4000
4100
4300
4400
4500
4800
4900
5000
5500
6000
6070
6500
7000
7440
7500
8000
8200
8500
9000
9500
10000
10500
11000
11500
11510
12000
12500
13000
13500
14000
14500
15000
15500
16000
16500
17000
17500
18000
18500
19000
19500
20000
21000
22000
23000
24000
25000
26000
27000
28000
29000
31000
32000
33000
36000
37000
38000
39000
43000
46000
48000
51000
53000
55000
56000
57000
65000
68000
74000
75000
79000
81000
82000
84000
93000
95000
97000
102000
108000
114000
120000
121000
126000
129000
130000
131000
143000
145000
168000
189000
196000
205000
240000
300000
310000
315000
325000
330000
340000
385000
395000
420000
1200000
1300000
1580000
1710000

I did get the wide scoope of my data with this chart, but still my data does not fit perfectly in any graphical data representation that I tried by now.

The last representation of my data I will have with most interesting charts to me.

In [37]:
```
# Load numpy and pandas for list and data set manipulation
import numpy as np
import pandas as pd

# Load matplotlib and seaborn for data visualisation
import matplotlib.pyplot as plt
import seaborn as sns
```

In [39]:
```
# Set style for Seaborn
sns.set(color_codes=True)
```

In [40]:
```
bees_data = pd.read_csv('Bee-Colony-Data-USDA-(No US Totals).csv')
```

In [41]:
```
bees_data
```

Out[41]:

	LineNum	State	Colonies at start of Month Range	Maximum Colonies	Lost Colonies	Percent Lost Colonies	Added Colonies	Renovated Colonies	Rer
0	0	Alabama	5500	5500	650	12	800	200	
1	1	Arizona	22000	22000	2500	11	430	90	
2	2	Arkansas	28000	28000	6500	23	20	20	
3	3	California	1140000	1580000	235000	15	83000	86000	
4	4	Colorado	5000	7500	320	4	0	0	
...	...	...	...	...	...	...	...	...	
225	229	Washington	50000	114000	3100	3	5000	3300	
226	230	West Virginia	7500	7500	570	8	1600	1500	
227	231	Wisconsin	27000	53000	1700	3	11500	4600	
228	232	Wyoming	17500	24000	1600	7	2300	2100	
229	233	Other States 5/	6500	7440	250	3	1270	1500	

230 rows × 11 columns

In [35]:
```
# Select a subset of the features
subset = bees_data[['Colonies at start of Month Range', 'Maximum Colonies ', 'Lo
       'Percent Lost Colonies', 'Added Colonies', 'Renovated Colonies', 'Percent

# Plot
sns.pairplot(subset)
```

Out[35]:
```
<seaborn.axisgrid.PairGrid at 0x138d895bb50>
```

Mostly the data is all over the place, and there are some interesting comparisons of data.

Summary

I have tried some interesting tools for data representation. With my data, I did not get the representations that I wanted. The point of this attempt is that the next time I will try to find different data in some way.

7. Conclusions

In this project, I represented different analytical skills. For the first part, I did not find the stories directly related to my main data set. However, I had interesting stories about bees and honey for my pre-processing text. My main data set for the rest of the project was on the cyclical movement of the number of bee colonies in the US throughout the year.

With the pre-processing text part, I can say that end result is satisfying. I was able to pass all steps. The change was obvious. In the end, I got the bee and honey word filtered properly.

With the exploratory data analysis part of the project, I was struggling to get interesting data for graphical representation. At the end of this section, my conclusion was really interesting. With this information, as the end result of my work, I can say to any beekeeper that California is one of the best states in the US for beekeeping. California is a state with the highest amount of bee colonies in general, and the lowest percentage of the lost colonies.

In the last part, I have represented data with more complex libraries for data representation.

8. References and Resources

Used for 4. Pre-processing Text

https://en.wikipedia.org/wiki/Bee

https://en.wikipedia.org/wiki/Honey

Used for 5. Exploratory data analysis and 6. Data examination and plottin

Bee-Colony-Data-USDA-(No US Totals).csv on website https://www.kaggle.com/datasets/ellies15/bee-colony-data-usda-20192020