```
In [2]: %run Imports.ipynb
```

Introduction

In this report, NLP is used to analyze the words from a CNBC article about covid-19 for document importance. This report uses K-means clustering to determine the document importance classes from the tf-idf matrix, KNN oversampling to resolve class imbalance, and multinomial naive bayes to classify the words for document importance.

Import data.

```
In [104]: url='https://www.cnbc.com/2021/04/18/half-of-us-adults-have-received-at-least-one-covid-vaccine-shot.html'
          r1 = requests.get(url)
          page = r1.content
```

```
In [105]: soup = BeautifulSoup(page, 'html5lib')
```

```
In [106]: tag = soup.find_all('div', class_='group')
```

```
In [116]: tag=tag[1:5]
```

```
In [114]: data=''
          for i in tag:
              a=i.text
              data=data+a
```

```
In [119]: data=data.replace('\'','')
          data=data.replace('\xa0',' ')
```

```
In [120]: data
```

```
Out[120]: 'Half of all U.S. adults have now received at least one Covid-19 vaccine dose, according to data from the Centers for Disease Control and Prevention, marking a major mi
          lestone in the countrys largest vaccine campaign.More than 129 million people ages 18 and older have received at least one shot, or 50.4% of the total adult population,
          according to the CDC. More than 83 million adults, or 32.5% of the total adult population, are fully vaccinated with one of the three vaccines approved in the U.S.The m
          ilestone comes one day after the global death toll from the virus topped 3 million people, according to data from the Johns Hopkins University, with global deaths avera
          ging about 12,000 each day.In the U.S., the rate of daily new Covid-19 cases nationwide remains high. The country is reporting about 68,000 new infections each day on a
          verage. CDC data shows an average of 3.3 million daily vaccine doses reported administered in the past week.White House Covid-19 Response Coordinator Jeff Zients has sa
          id the pause in Johnson & Johnson vaccinations, which occurred after reports of six cases of rare brain blood clots, would not slow down the vaccine campaign since the
          country has enough supply of Pfizer and Moderna vaccines.White House chief medical advisor Dr. Anthony Fauci said Sunday he thinks the U.S. will likely resume use of th
          e Johnson & Johnson vaccine with a warning or restriction attached and anticipates a decision as soon as Friday, when the CDCs vaccine advisory panel meets to discuss r
          esumption."My estimate is that we will continue to use it in some form," Fauci said during an interview on NBCs "Meet the Press." "I doubt very seriously if they just c
          ancel it. I dont think thats going to happen."'
```

Data Mining

```
In [396]: len(data.split(' '))
```

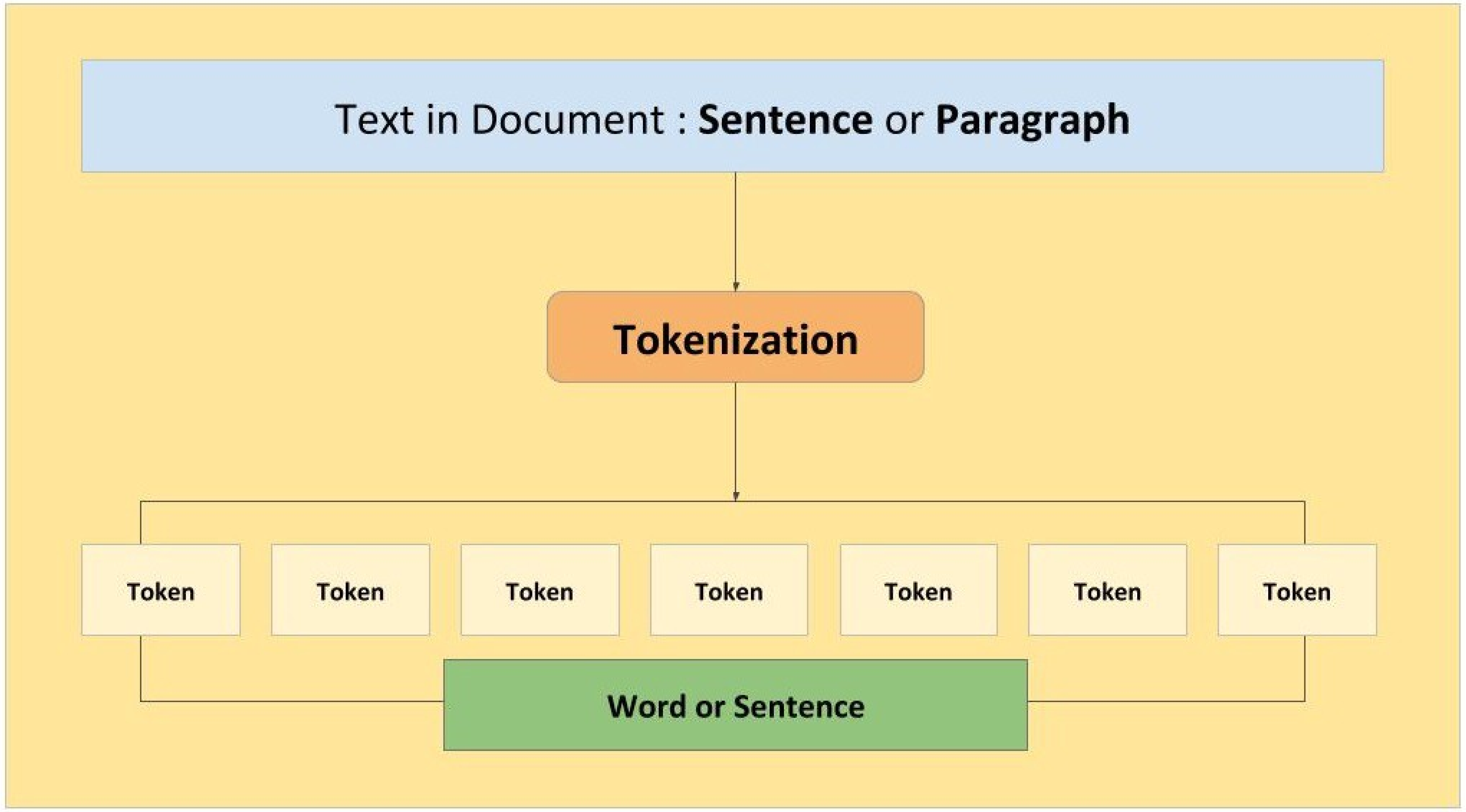
```
Out[396]: 288
```

Data Cleaning

Tokenization seperates a string of words into a list.

```
In [397]: Image(filename='tokenization.jpeg')
```

```
Out[397]:
```



Tokenize data.

```
In [398]: tokenizer = RegexpTokenizer('\w+')
          tokens = tokenizer.tokenize(data)
```

Stop words are a set of commonly used, unimportant, words in a language. Remove stopwords from data.

```
In [399]: stopwords_list = stopwords.words('english')
          stopwords_list += list(string.punctuation)
```

```
In [476]: stopped_tokens = [w.lower() for w in tokens if w not in stopwords_list]
```

us is coming out as two seperate letters.

```
In [477]: stopped_tokens
```

```
Out[477]: ['half',
           'u',
           's',
           'adults',
           'received',
           'least',
           'one',
           'covid',
           '19',
           'vaccine',
           'dose',
           'according',
           'data',
           'centers',
           'disease',
           'control',
           'prevention',
           'marking',
           'major',
           'milestone',
           'country',
           'largest',
           'vaccine',
           'campaign',
           'more',
           '129',
           'million',
           'people',
           'ages',
           '18',
           'older',
           'received',
           'least',
           'one',
           'shot',
           '50',
           '4',
           'total',
           'adult',
           'population',
           'according',
           'cdc',
           'more',
           '83',
           'million',
           'adults',
           '32',
           '5',
           'total',
           'adult',
           'population',
           'fully',
           'vaccinated',
           'one',
           'three',
           'vaccines',
           'approved',
           'u',
           's',
           'the',
           'milestone',
           'comes',
           'one',
           'day',
           'global',
           'death',
           'toll',
           'virus',
           'topped',
           '3',
           'million',
           'people',
           'according',
           'data',
           'johns',
           'hopkins',
           'university',
           'global',
           'deaths',
           'averaging',
           '12',
           '000',
           'day',
           'in',
           'u',
           's',
           'rate',
           'daily',
           'new',
           'covid',
           '19',
           'cases',
           'nationwide',
           'remains',
           'high',
           'the',
           'country',
           'reporting',
           '68',
           '000',
           'new',
           'infections',
           'day',
           'average',
           'cdc',
           'data',
           'shows',
           'average',
           '3',
           '3',
           'million',
           'daily',
           'vaccine',
           'doses',
           'reported',
           'administered',
           'past',
           'week',
           'white',
           'house',
           'covid',
           '19',
           'response',
           'coordinator',
           'jeff',
           'zients',
           'said',
           'pause',
           'johnson',
           'johnson',
           'vaccinations',
           'occurred',
           'reports',
           'six',
           'cases',
           'rare',
           'brain',
           'blood',
           'clots',
           'would',
           'slow',
           'vaccine',
           'campaign',
           'since',
           'country',
           'enough',
           'supply',
           'pfizer',
           'moderna',
           'vaccines',
           'white',
           'house',
           'chief',
           'medical',
           'advisor',
           'dr',
           'anthony',
           'fauci',
           'said',
           'sunday',
           'thinks',
           'u',
           's',
           'likely',
           'resume',
           'use',
           'johnson',
           'johnson',
           'vaccine',
           'warning',
           'restriction',
           'attached',
           'anticipates',
           'decision',
           'soon',
           'friday',
           'cdc',
           'vaccine',
           'advisory',
           'panel',
           'meets',
           'discuss',
           'resumption',
           'my',
           'estimate',
           'continue',
           'use',
           'form',
           'fauci',
           'said',
           'interview',
           'nbc',
           'meet',
           'press',
           'i',
           'doubt',
           'seriously',
           'cancel',
           'i',
           'think',
           'going',
           'happen']
```

us appears four times.

```
In [478]: pattern = 'U.S.'
          p = re.compile(pattern)
          p.findall(data)

Out[478]: ['U.S.', 'U.S.', 'U.S.', 'U.S.']
```

Use n-gram to attach u and s.

```
In [479]: n_grams = ngrams(stopped_tokens, 2)
          us=[''.join(grams) for grams in n_grams][1:3][0]
          us

Out[479]: 'us'
```

Remove u and s from stopped_tokens.

In [480]:
```
for i in range(4):
    stopped_tokens.remove('u')
    stopped_tokens.remove('s')
```

Append us to stopped_tokens four times.

In [481]:
```
for i in range(4):
    stopped_tokens.append(us)
```

Lemmatization is the morphological analysis of words, and is used to remove inflectional endings of words and to return the base form of a word. Lemmatize stopped_tokens.

Word	Stem	Lemma
Studies	Studi	Study
Studying	Study	Study

In [482]:
```
words=[]
lemmatizer = WordNetLemmatizer()
for i in stopped_tokens:
    s=lemmatizer.lemmatize(i)
    words.append(s)
```

Data Exploration

In [486]:
```
len(words)
```

Out[486]:
```
198
```

In [485]:
```
freqdist = FreqDist(words)
most_common = freqdist.most_common()
most_common
```

```
Out[485]: [('vaccine', 8),
           ('adult', 4),
           ('one', 4),
           ('million', 4),
           ('johnson', 4),
           ('u', 4),
           ('covid', 3),
           ('19', 3),
           ('according', 3),
           ('data', 3),
           ('country', 3),
           ('cdc', 3),
           ('day', 3),
           ('3', 3),
           ('said', 3),
           ('received', 2),
           ('least', 2),
           ('milestone', 2),
           ('campaign', 2),
           ('more', 2),
           ('people', 2),
           ('total', 2),
           ('population', 2),
           ('the', 2),
           ('global', 2),
           ('death', 2),
           ('000', 2),
           ('daily', 2),
           ('new', 2),
           ('case', 2),
           ('average', 2),
           ('white', 2),
           ('house', 2),
           ('fauci', 2),
           ('think', 2),
           ('use', 2),
           ('meet', 2),
           ('i', 2),
           ('half', 1),
           ('dose', 1),
           ('center', 1),
           ('disease', 1),
           ('control', 1),
           ('prevention', 1),
           ('marking', 1),
           ('major', 1),
           ('largest', 1),
           ('129', 1),
           ('age', 1),
           ('18', 1),
           ('older', 1),
           ('shot', 1),
           ('50', 1),
           ('4', 1),
           ('83', 1),
           ('32', 1),
           ('5', 1),
('fully', 1),
('vaccinated', 1),
('three', 1),
('approved', 1),
('come', 1),
('toll', 1),
('virus', 1),
('topped', 1),
('john', 1),
('hopkins', 1),
('university', 1),
('averaging', 1),
('12', 1),
('in', 1),
('rate', 1),
('nationwide', 1),
('remains', 1),
('high', 1),
('reporting', 1),
('68', 1),
('infection', 1),
('show', 1),
('dos', 1),
('reported', 1),
('administered', 1),
('past', 1),
('week', 1),
('response', 1),
('coordinator', 1),
('jeff', 1),
('zients', 1),
('pause', 1),
('vaccination', 1),
('occurred', 1),
('report', 1),
('six', 1),
('rare', 1),
('brain', 1),
('blood', 1),
('clot', 1),
('would', 1),
('slow', 1),
('since', 1),
('enough', 1),
('supply', 1),
('pfizer', 1),
('moderna', 1),
('chief', 1),
('medical', 1),
('advisor', 1),
('dr', 1),
('anthony', 1),
('sunday', 1),
('likely', 1),
('resume', 1),
('warning', 1),
('restriction', 1),
('attached', 1),
('anticipates', 1),
('decision', 1),
('soon', 1),
('friday', 1),
('advisory', 1),
('panel', 1),
('discus', 1),
('resumption', 1),
('my', 1),
('estimate', 1),
('continue', 1),
('form', 1),
('interview', 1),
('nbc', 1),
('press', 1),
('doubt', 1),
('seriously', 1),
('cancel', 1),
('going', 1),
('happen', 1)]
```

Feature Engineering

TF-IDF determines how relevant a word is to a document in a collection of documents. This is done by multiplying term frequency of how many times a term, t, appears in a document, d, with the inverse document frequency of the terms across a set of documents. IDF is calculated by taking the log of one plus the number of documents, n, divided by one plus the document frequency of the terms, and the result added by one.

$$TFIDF = TF(t,d) \times IDF(t)$$

$$IDF(t) = log(\frac{1+n}{1+DF(t)}) + 1$$

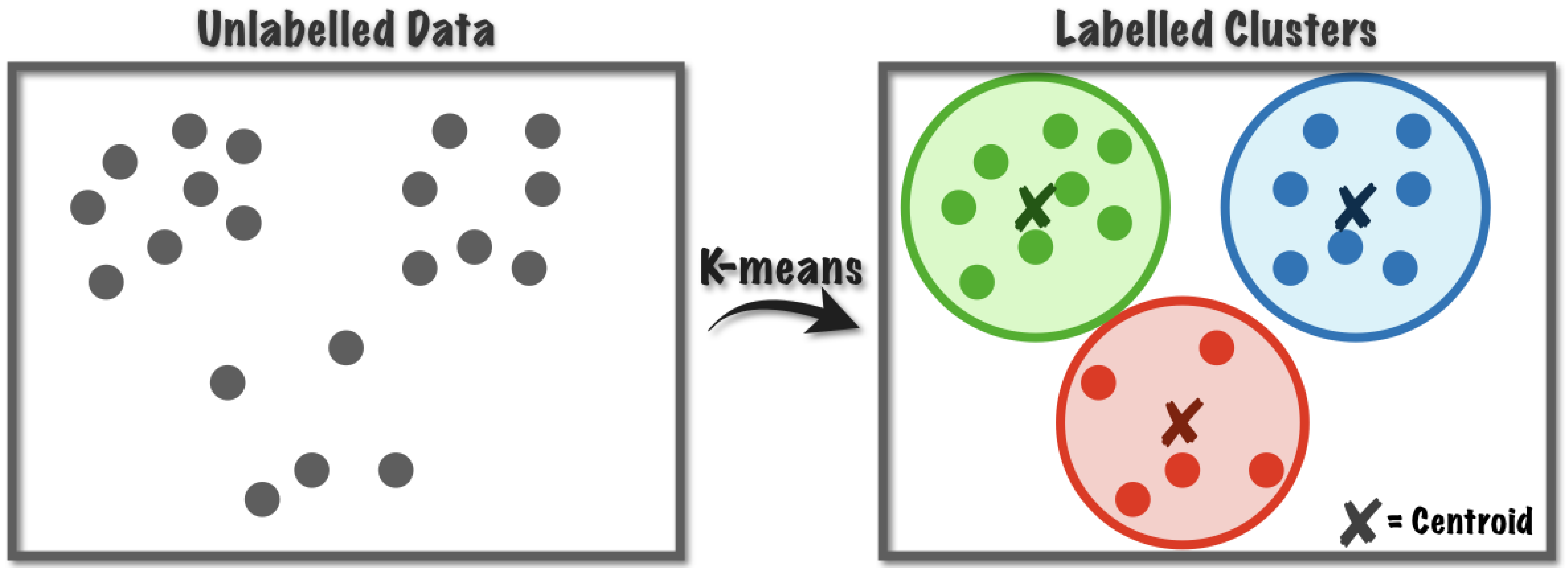
In [491]:
```
vectorizer = TfidfVectorizer()
tf_idf_matrix = vectorizer.fit_transform(words)
```

k-means clustering is a method of vector quantization that partitions n observations into k clusters in which each observation belongs to the cluster with the nearest mean, and will be used to determine class labels for the tf_idf_matrix. Centroids are initially randomly assigned, and then shift with each iteration as they are recalculated to match the center of the points assigned to their cluster. C is the number of clusters, N is the number of data points, and $||xi$–v$j||$ is the euclidean distance between the observations and centroids.

$$argmin(S) = \sum_i^N \sum_j^C (||x_i - v_j||)^2$$

```
In [494]: Image(filename='k_means.png')
```

Out[494]:



Check 2 through 7 clusters to see which amount of clusters is optimal.

```
In [296]: k_means_2 = KMeans(n_clusters=2).fit(tf_idf_matrix)
          k_means_3 = KMeans(n_clusters=3).fit(tf_idf_matrix)
          k_means_4 = KMeans(n_clusters=4).fit(tf_idf_matrix)
          k_means_5 = KMeans(n_clusters=5).fit(tf_idf_matrix)
          k_means_6 = KMeans(n_clusters=6).fit(tf_idf_matrix)
          k_means_7 = KMeans(n_clusters=7).fit(tf_idf_matrix)

          k_list = [k_means_2, k_means_3, k_means_4, k_means_5, k_means_6, k_means_7]
```

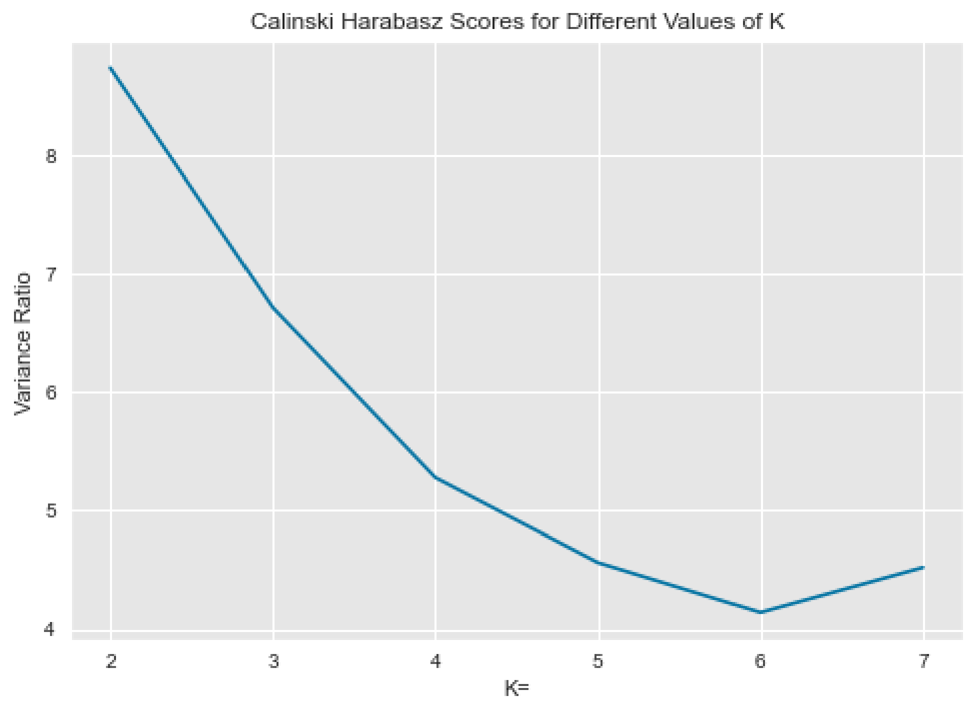
The calinski harabasz score is used to determine how many clusters the data has. SW is the is the overall within-cluster variance, and SB is the overall between-cluster variance.

$$\frac{S_B}{S_W} \times \frac{N - C}{C - 1}$$

```
In [297]: CH_score = []

          for model in k_list:
              labels = model.labels_
              CH_score.append(calinski_harabasz_score(tf_idf_data_train.toarray(), labels))
```

```
In [298]: plt.plot([2, 3, 4, 5, 6, 7], CH_score)
          plt.xticks([2,3,4,5,6,7])
          plt.title('Calinski Harabasz Scores for Different Values of K')
          plt.ylabel('Variance Ratio')
          plt.xlabel('K=')
          plt.show()
```



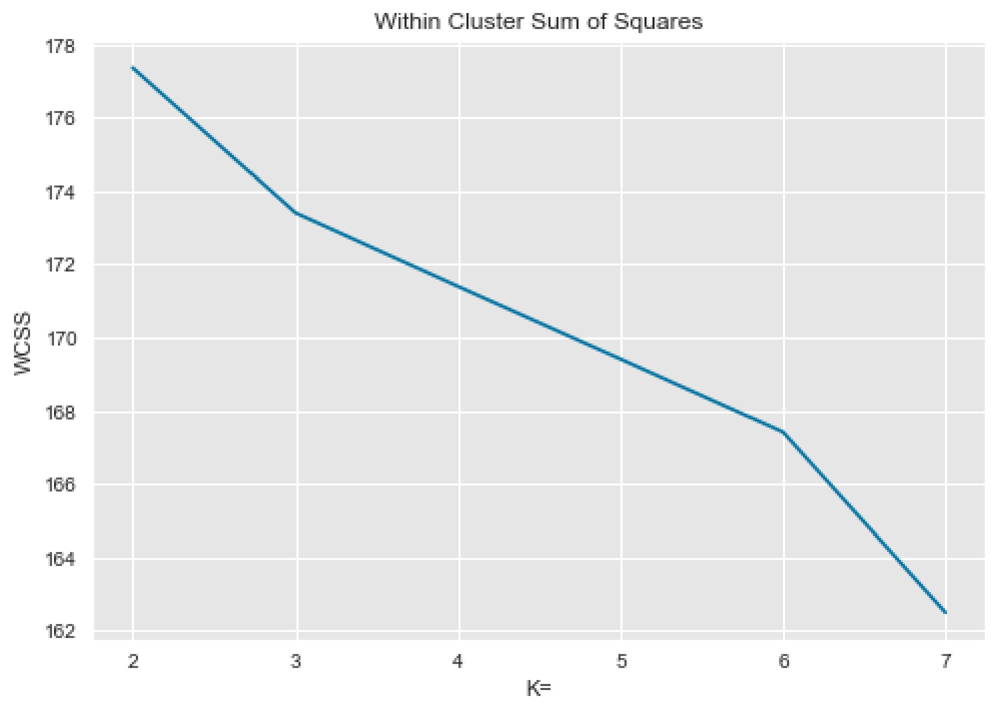
WCSS is the sum of squares of the distances of each data point in all clusters to their respective centroids.

$$\sum_j^C (\sum_i^N \sqrt{(||x_i - v_j||)^2})$$

```
In [299]: wcss_score = []

          for model in k_list:
              labels = model.labels_
              wcss_score.append(model.inertia_)
```

In [300]:
```
plt.plot([2, 3, 4, 5, 6, 7], wcss_score)
plt.xticks([2,3,4,5,6,7])
plt.title('Within Cluster Sum of Squares')
plt.ylabel('WCSS')
plt.xlabel('K=')
plt.show()
```



Both metrics show the elbow at K being 3 clusters.

In [303]:
```
k_means = KMeans(n_clusters=3)

k_means.fit(tf_idf_matrix)

cluster_assignments = k_means.predict(tf_idf_matrix)
```

View index of each document importance word label.

In [516]:
```
for i,l in enumerate(cluster_assignments):
    print(i,l)
```

```
0 0
1 0
2 0
3 0
4 0
5 0
6 0
7 0
8 2
9 1
10 0
11 0
12 0
13 0
14 0
15 0
16 0
17 0
18 0
19 0
20 0
21 0
22 1
23 0
24 0
25 0
26 0
27 0
28 0
29 0
30 0
31 0
32 0
33 0
34 0
35 0
36 0
37 0
38 0
39 0
40 0
41 0
42 0
43 0
44 0
45 0
46 0
47 0
48 0
49 0
50 0
51 0
52 0
53 0
54 0
55 1
56 0
```

```
57 0
58 0
59 0
60 0
61 0
62 0
63 0
64 0
65 0
66 0
67 0
68 0
69 0
70 0
71 0
72 0
73 0
74 0
75 0
76 0
77 0
78 0
79 0
80 0
81 0
82 0
83 0
84 0
85 0
86 0
87 0
88 0
89 0
90 2
91 0
92 0
93 0
94 0
95 0
96 0
97 0
98 0
99 0
100 0
101 0
102 0
103 0
104 0
105 0
106 0
107 0
108 0
109 0
110 0
111 0
112 1
113 0
```

```
114 0
115 0
116 0
117 0
118 0
119 0
120 0
121 2
122 0
123 0
124 0
125 0
126 0
127 0
128 0
129 0
130 0
131 0
132 0
133 0
134 0
135 0
136 0
137 0
138 0
139 0
140 0
141 1
142 0
143 0
144 0
145 0
146 0
147 0
148 0
149 1
150 0
151 0
152 0
153 0
154 0
155 0
156 0
157 0
158 0
159 0
160 0
161 0
162 0
163 0
164 0
165 0
166 0
167 0
168 1
169 0
170 0
```

```
171 0
172 0
173 0
174 0
175 0
176 0
177 1
178 0
179 0
180 0
181 0
182 0
183 0
184 0
185 0
186 0
187 0
188 0
189 0
190 0
191 0
192 0
193 0
194 0
195 0
196 0
197 0
198 0
199 0
200 0
201 0
```

Dose, country, and pause are the words with the greatest document importance.

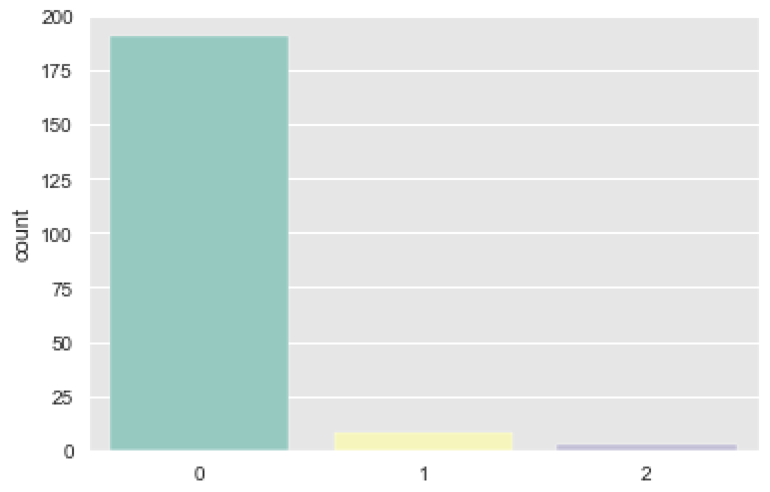
```
In [521]: for i,l in enumerate(words):
              if i in [8,90,121]:
                  print(l)
```

```
dose
country
pause
```

Check cluster distribution.

```
In [495]: sns.countplot(cluster_assignments, palette='Set3')
```

```
Out[495]: <AxesSubplot:ylabel='count'>
```



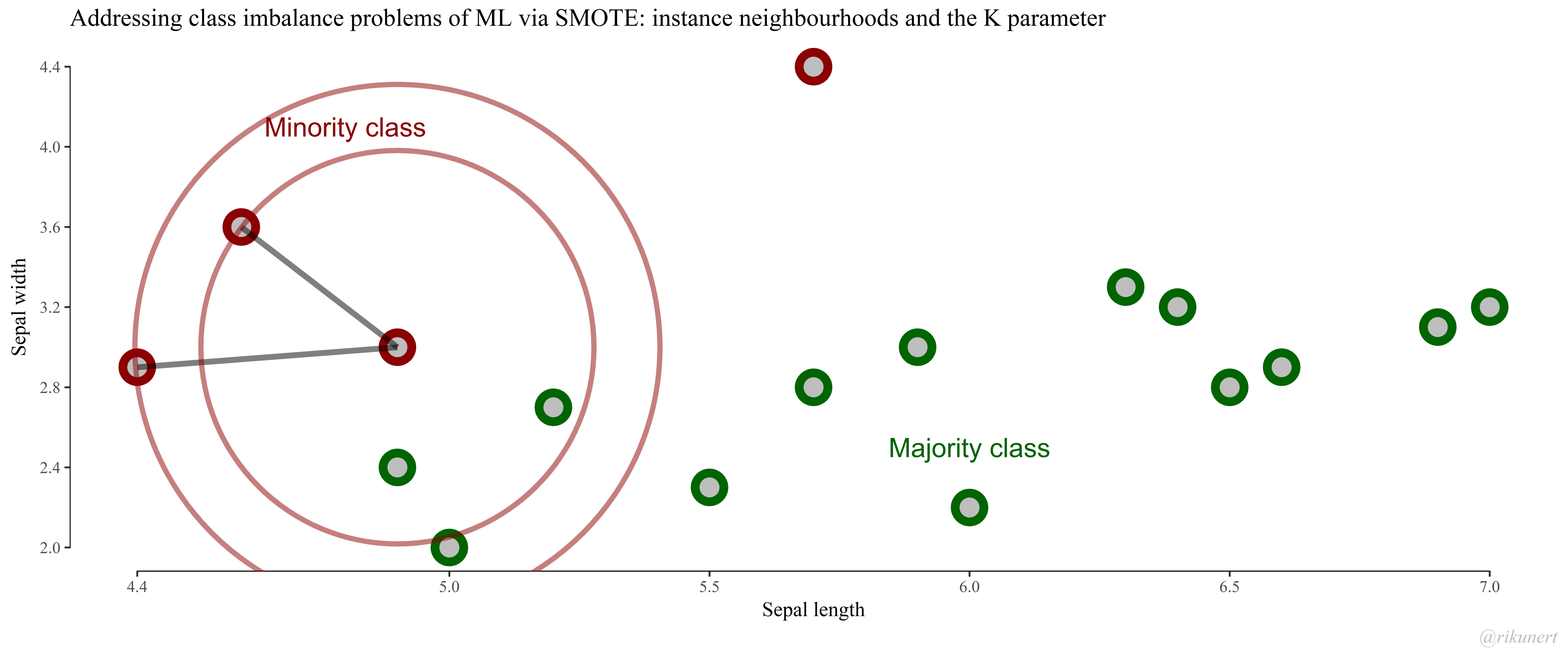
There is a class imbalance that will be resolved using Smote.

KNN uses data to classify new data points based on a distance function. Classification is done by a majority vote from the neighbors of the new data points. At K = 2, both the closest and the second closest neighbours are considered.

$$\sqrt{\sum_i^N (||q_i - p_i||)^2}$$

```
In [497]: Image(filename='SMOTE.png')
```

Out[497]:



```
In [339]: smote = SMOTE(k_neighbors=2)
          X_resampled, y_resampled = smote.fit_sample(tf_idf_matrix.toarray(), cluster_assignments)
```

Make 80/20 train test split.

```
In [340]: X_train, X_test, y_train, y_test = train_test_split(X_resampled, y_resampled, test_size=0.2, random_state=42)
```

Naive Bayes Text Importance Classification Modeling

Naive Bayes applies Bayes' theorem with the naive assumption of conditional independence between the features to classify the label based on the posterior pobability.

$$P(Y|X) = \frac{P(X|Y) * P(Y)}{P(X)}$$

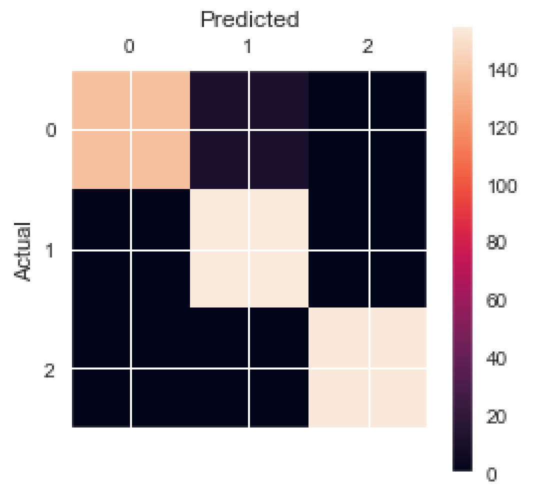
$$Posterior = \frac{Prior * Likelihood}{Evidence}$$

In [510]:
```
nb = MultinomialNB()

nb.fit(X_train, y_train)
preds = nb.predict(X_train)
```

In [511]:
```
con_mat(y_train,preds)
```

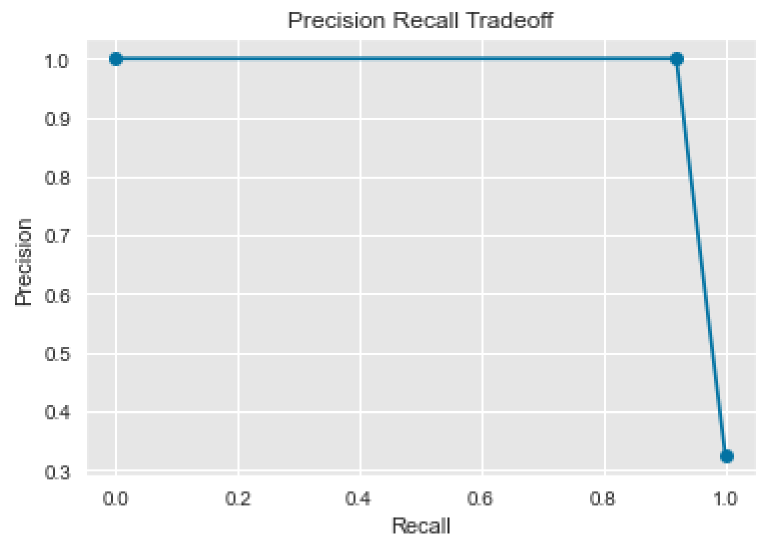
```
Confusion Matrix
----------------
     0    1    2
0  137   12    0
1    0  155    0
2    0    0  154
```

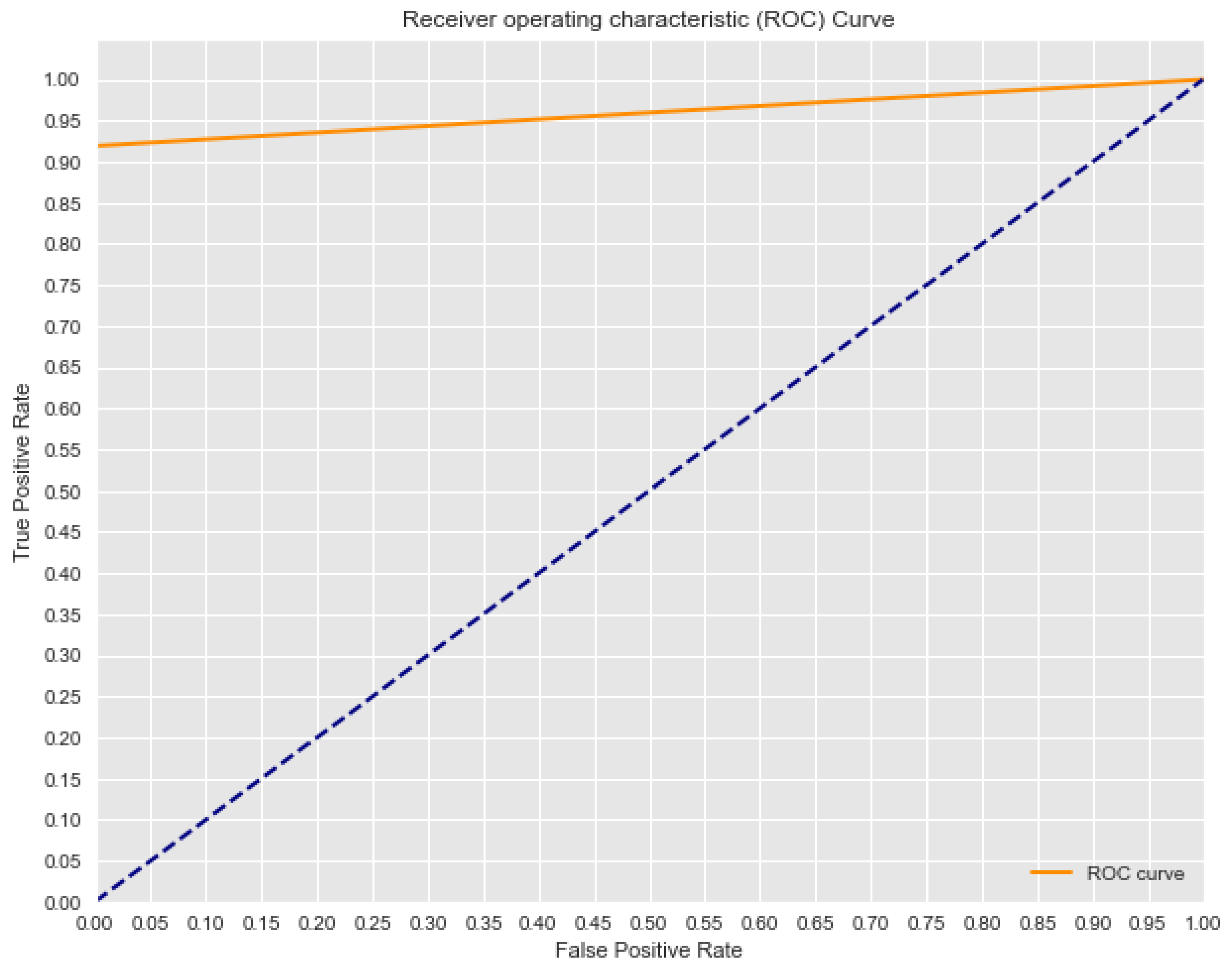


In [501]:
```
actual_class0=y_train==0
actual_class1=y_train==1
actual_class2=y_train==2
predictions_class0=preds==0
predictions_class1=preds==1
predictions_class2=preds==2
a=[actual_class0,actual_class1,actual_class2]
p=[predictions_class0,predictions_class1,predictions_class2]
```

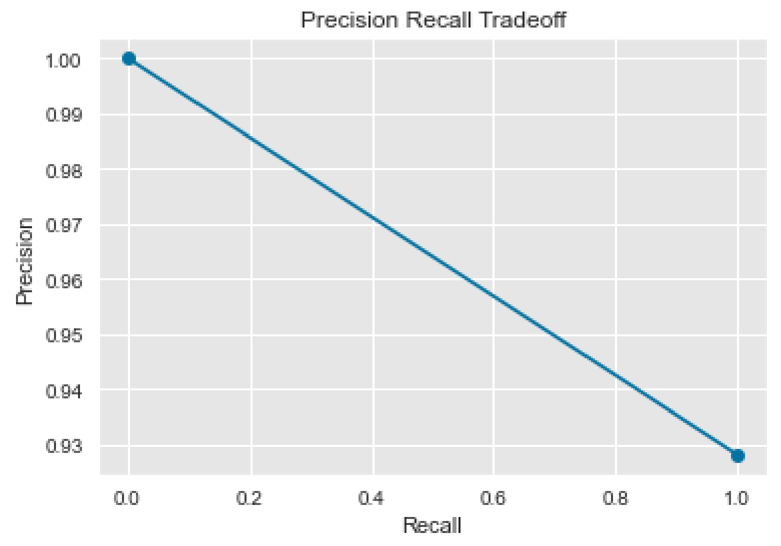
```
In [506]: multiclass(nb,X_train,a,p,y_train,'train')
```

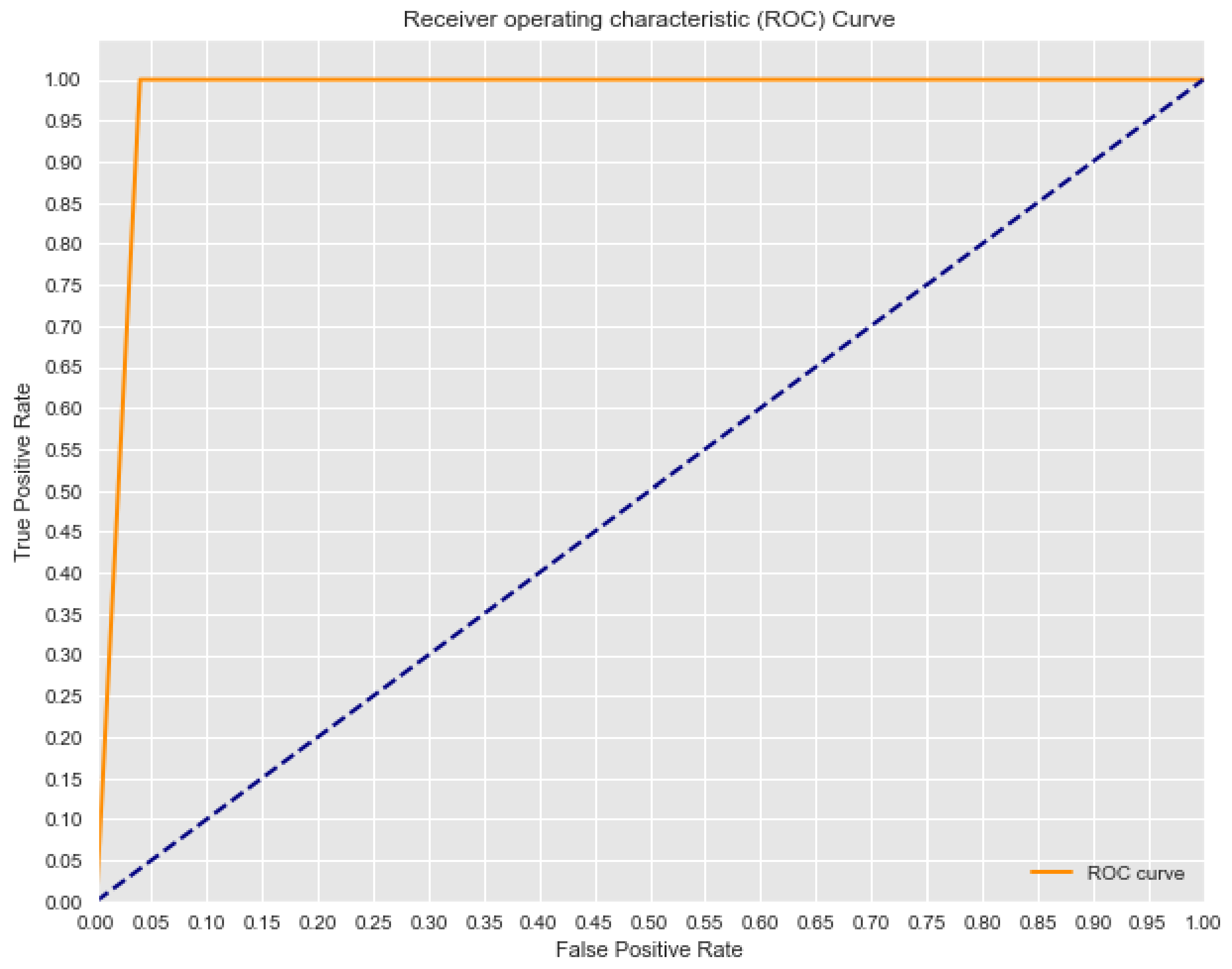
```
Class:0
Precision Score: 1.0
Recall Score: 0.9194630872483222
F1 Score: 0.958041958041958
Accuracy Score: 0.9737991266375546
Specificity Score: 0.9626168224299065
AUC: 0.9597315436241611
```



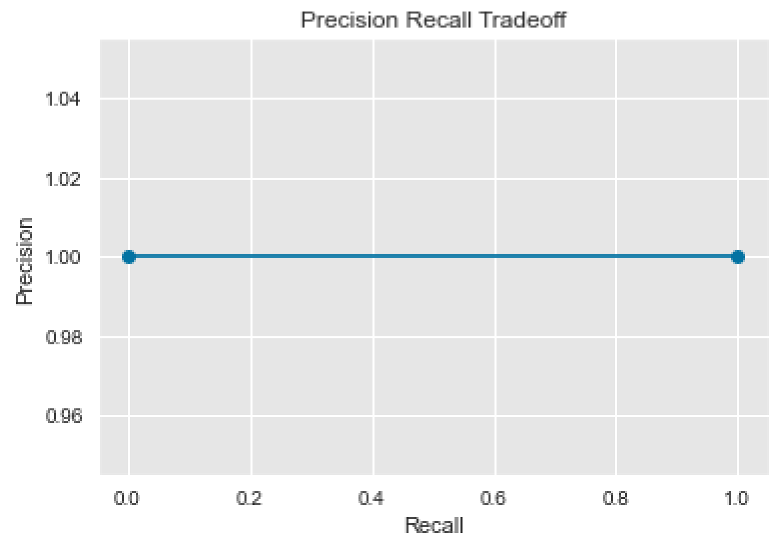


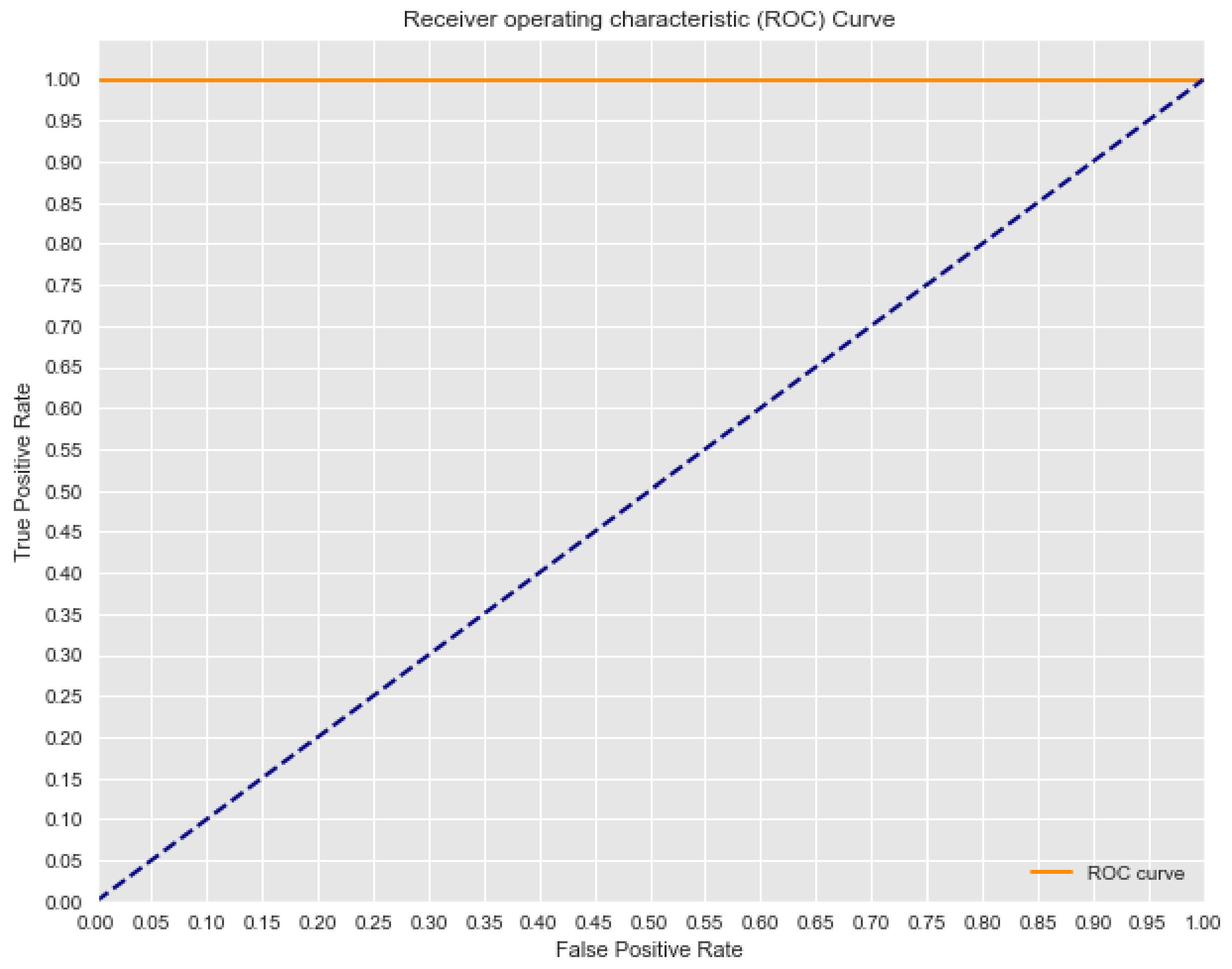
```
Class:1
Precision Score: 0.9281437125748503
Recall Score: 1.0
F1 Score: 0.9627329192546584
Accuracy Score: 0.9737991266375546
Specificity Score: 1.0
AUC: 0.9801980198019802
```





```
Class:2
Precision Score: 1.0
Recall Score: 1.0
F1 Score: 1.0
Accuracy Score: 1.0
Specificity Score: 1.0
AUC: 1.0
```





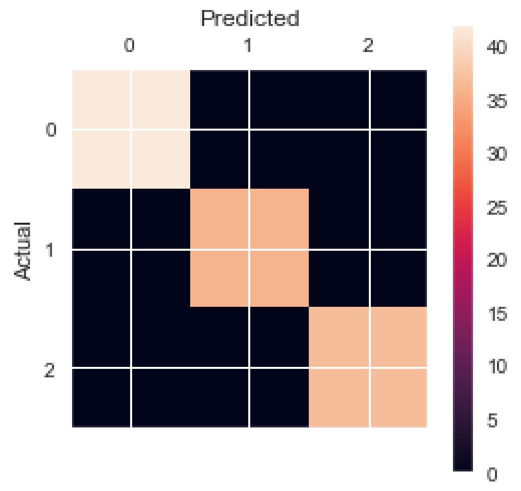
```
____________________________________________________
Class 0 Cross Validated ROC AUC Score: 0.8478193701723115
Class 1 Cross Validated ROC AUC Score: 0.958028846153846
Class 2 Cross Validated ROC AUC Score: 0.9651515151515152
```

Test model.

In [513]:
```
nb.fit(X_test, y_test)
preds = nb.predict(X_test)
```

In [514]:
```
con_mat(y_test,preds)
```

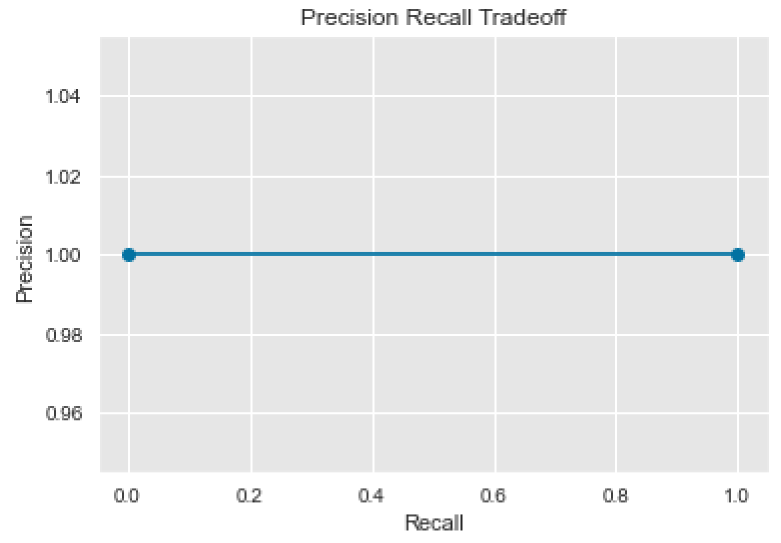
```
Confusion Matrix
----------------
    0   1   2
0  42   0   0
1   0  36   0
2   0   0  37
```

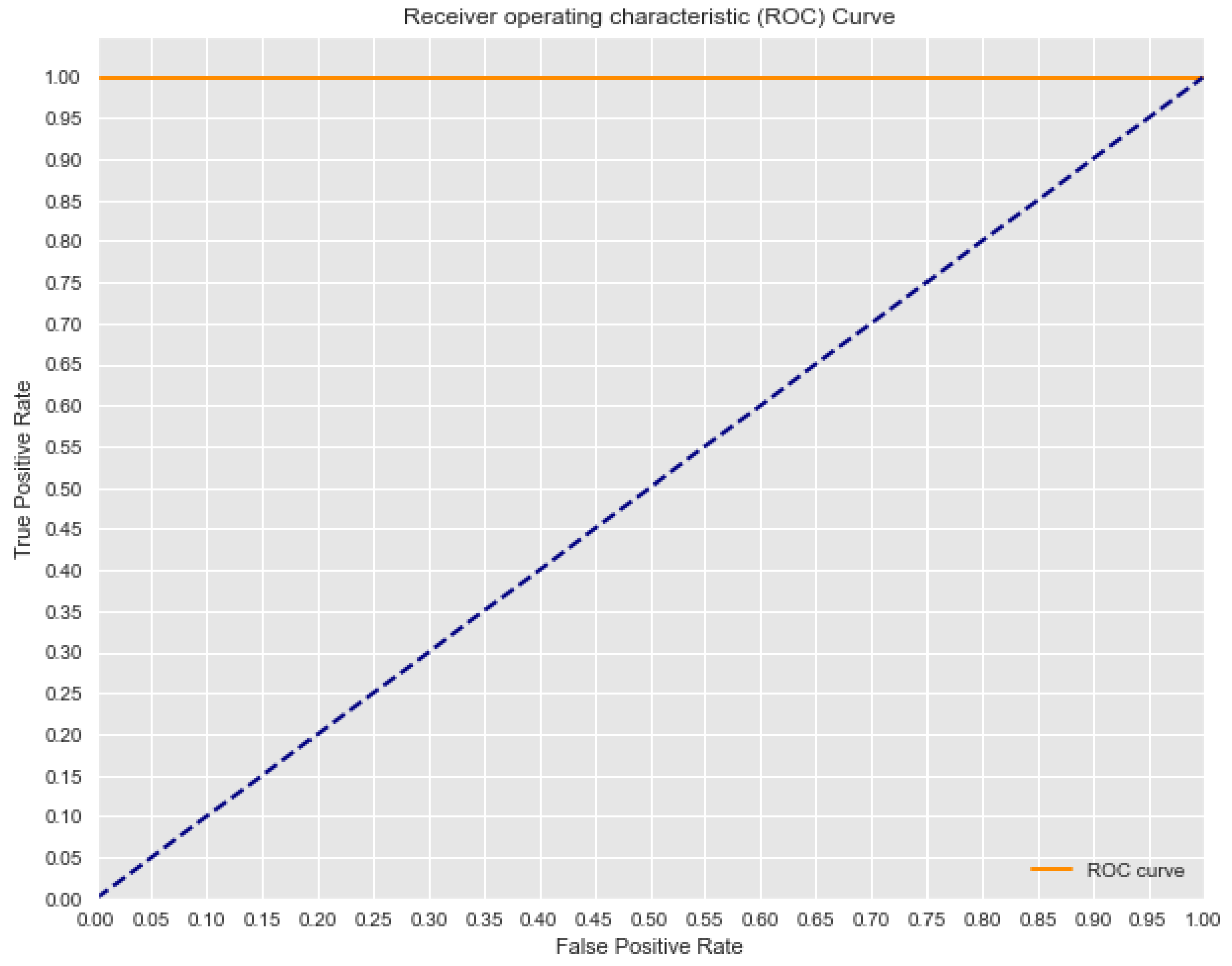


In [508]:
```
actual_class0=y_test==0
actual_class1=y_test==1
actual_class2=y_test==2
predictions_class0=preds==0
predictions_class1=preds==1
predictions_class2=preds==2
a=[actual_class0,actual_class1,actual_class2]
p=[predictions_class0,predictions_class1,predictions_class2]
```

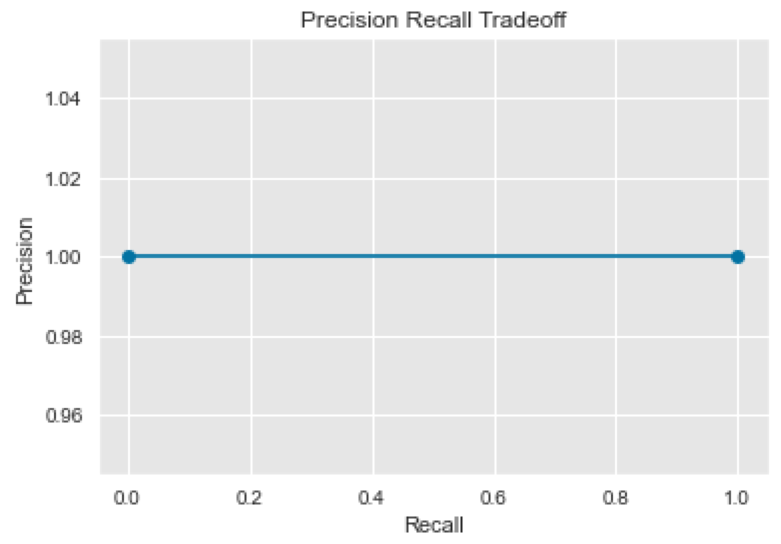
```
In [509]: multiclass(nb,X_test,a,p,y_test,'test')
```

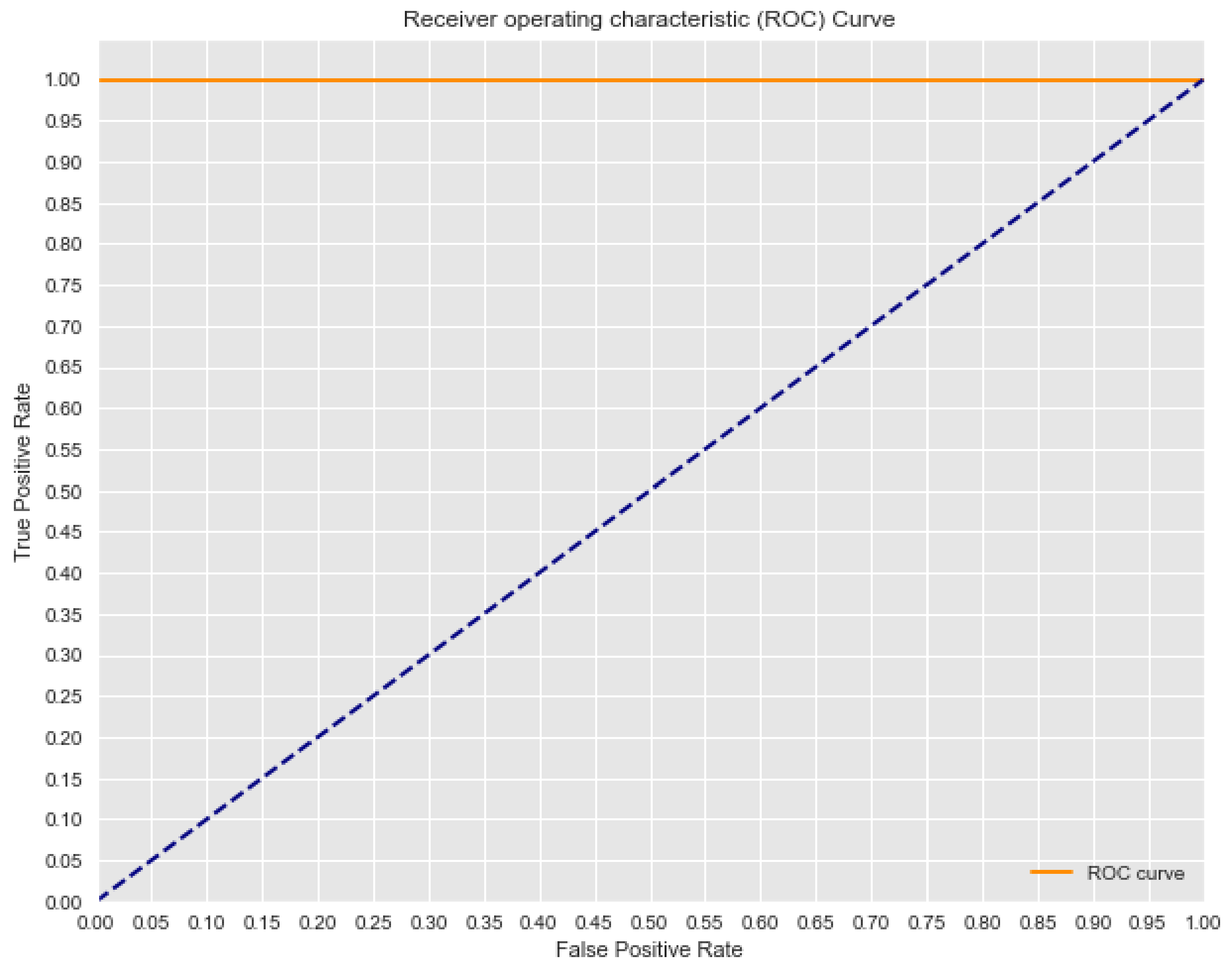
```
Class:0
Precision Score: 1.0
Recall Score: 1.0
F1 Score: 1.0
Accuracy Score: 1.0
Specificity Score: 1.0
AUC: 1.0
```



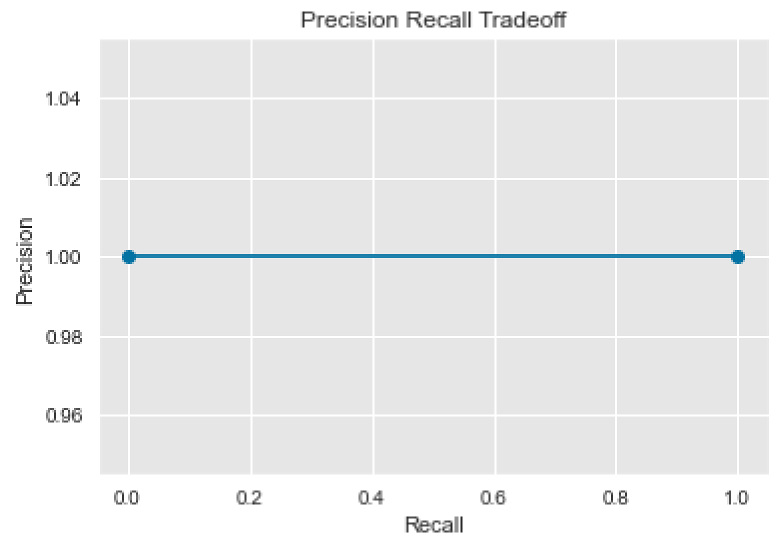


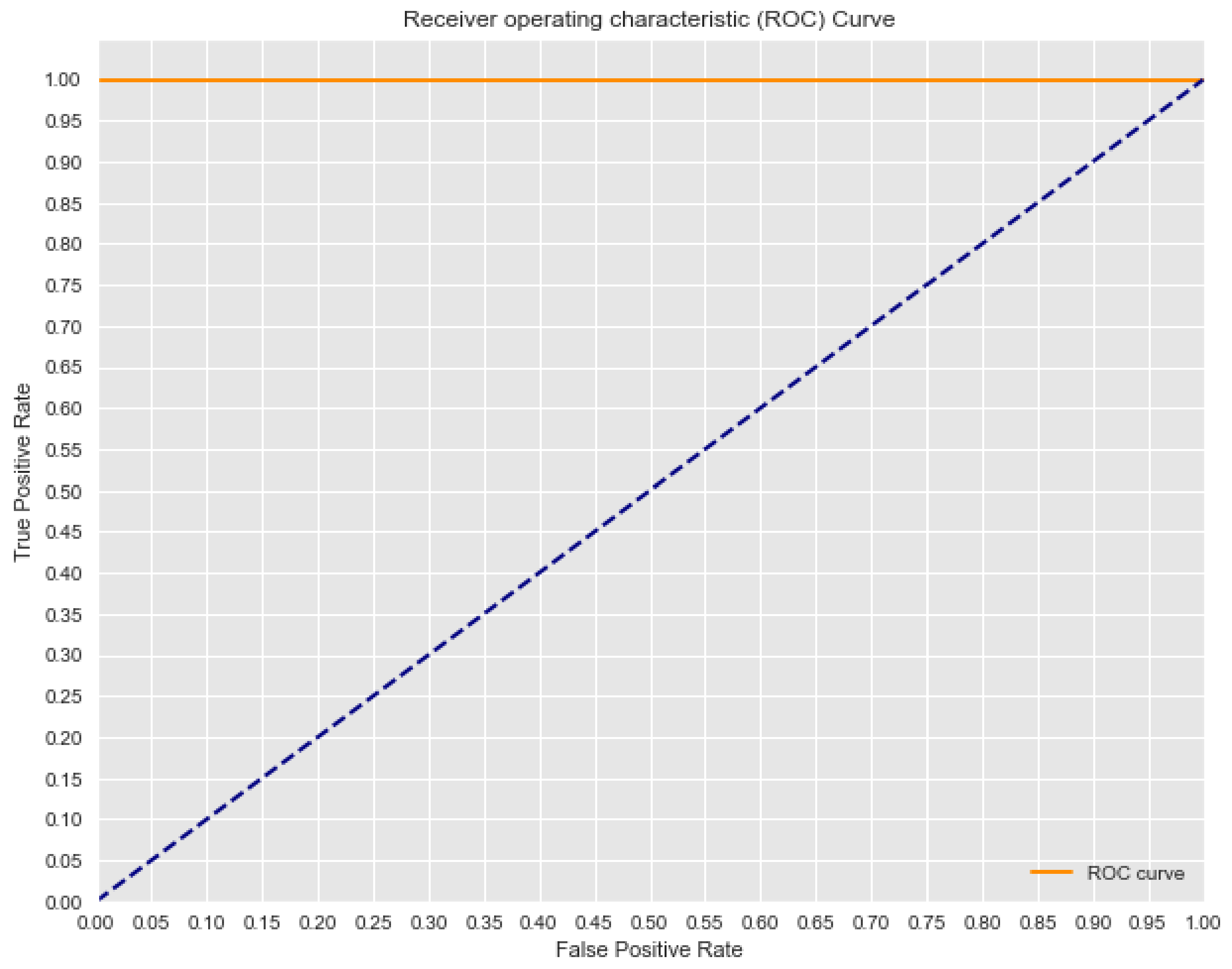
```
Class:1
Precision Score: 1.0
Recall Score: 1.0
F1 Score: 1.0
Accuracy Score: 1.0
Specificity Score: 1.0
AUC: 1.0
```





```
Class:2
Precision Score: 1.0
Recall Score: 1.0
F1 Score: 1.0
Accuracy Score: 1.0
Specificity Score: 1.0
AUC: 1.0
```





Conclusion

Dose, country, and pause are the words with the greatest document importance. The naive bayes model is able to classify text importance with a cross validated ROC AUC Score for class 0 of 85, for class 1 of 96, and for class 2 of 97.

```
In [ ]:
```