2017 Fall: COMP-SCI 5590/490 - Special Topics

Python Programming

Lab Assignment 3

Assignment Overview

The following assignment focus on to make one familiar with python machine learning

Lab Assignment

1) Pick any dataset from the dataset sheet in class sheet and make one prediction model using your imagination with **Linear Discriminant Analysis***. Some examples are:
   a. In the report provide convincible explanations about the difference of logistic regression and Linear Discriminant Analysis.
   b. You can also pick dataset of your own.

*Logistic Regression is a classification algorithm traditionally limited **to only two-class classification problems**. If you have **more than two classes then the Linear Discriminant Analysis algorithm is the preferred** linear classification technique.

Results:

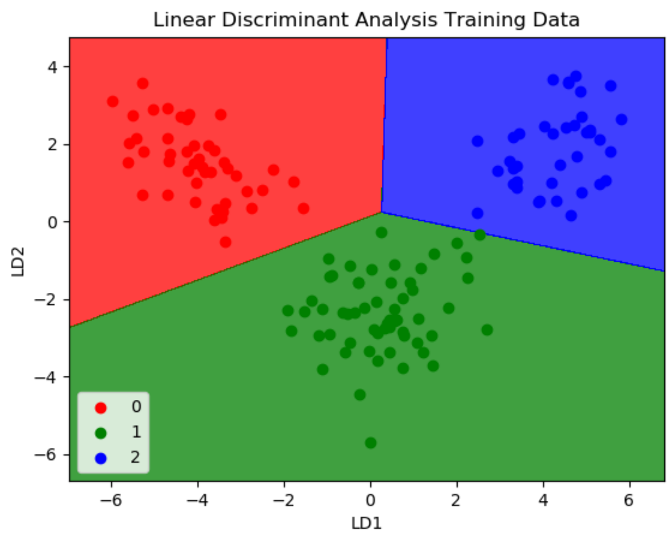
The following code was developed for the Linear Discriminant Analysis.

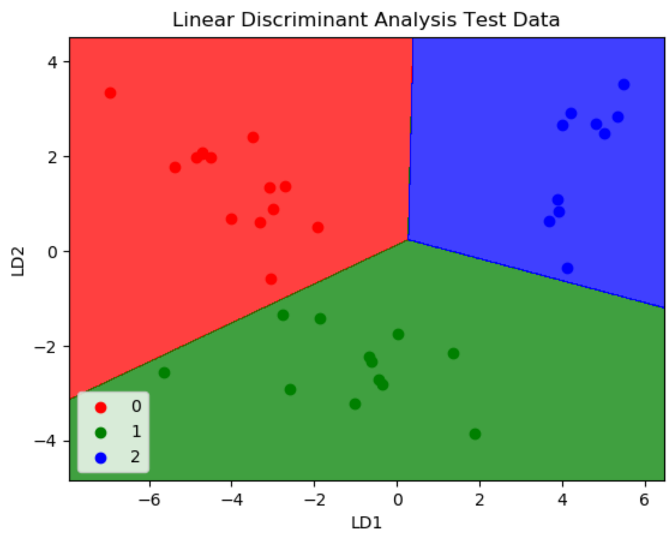
```
#Don Baker
#Comp-Sci 5590
#Lab 3, Question 1
#Pick any dataset and make a prediction model using LDA.  Compare the results to
logistic regression

#Guidance and example code taken from Learn&Teach YouTube videos at
https://www.youtube.com/channel/UCGD8R7ct8cQfMMuUFImqDdQ


#Import Packages/Libraries
import matplotlib.pyplot as plt
import numpy as np

#Import Wine data
from sklearn import datasets
wine = datasets.load_wine()

#Split the Wine data into training and testing data sets for cross-validation
from sklearn import model_selection as cv
X=wine.data
y=wine.target
splits=cv.train_test_split(X,y,test_size=0.20)
X_train,X_test,y_train,y_test=splits
```

```
#Scale the data to provide more accurate evalaution
from sklearn.preprocessing import StandardScaler
Scale_X=StandardScaler()
X_train=Scale_X.fit_transform(X_train)
X_test=Scale_X.transform(X_test)

#Complete Logistic Regression and Linear Discriminant Analysis to compare

from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
lda_wine=LDA(n_components=2)
X_train=lda_wine.fit_transform(X_train,y_train)
X_test=lda_wine.transform(X_test)

#Logistic Regression
from sklearn.linear_model import LogisticRegression
wine_LDA_classifier=LogisticRegression(random_state=5)
wine_LDA_classifier.fit(X_train,y_train)

#Predict the results of the model using the test set of data
y_pred_LDA=wine_LDA_classifier.predict(X_test)

#Create and display the confusion matrix for the analysis
from sklearn.metrics import confusion_matrix
confusion_LDA=confusion_matrix(y_test,y_pred_LDA)
print("The confusion matrix for the Linear Discriminat Analysis is:\n",confusion_LDA)

#Plot the training set results for the LDA
from matplotlib.colors import ListedColormap
X_set,y_set=X_train,y_train
X1,X2=np.meshgrid(np.arange(start=X_set[:,0].min()-
1,stop=X_set[:,0].max()+1,step=0.01),
                  np.arange(start=X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1,
step = 0.01))
plt.contourf(X1,X2,wine_LDA_classifier.predict(np.array([X1.ravel(),X2.ravel()]).T).re
shape(X1.shape),
             alpha=0.75,cmap=ListedColormap(('red','green','blue')))
plt.xlim(X1.min(),X1.max())
plt.ylim(X2.min(),X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set==j,0],X_set[y_set==j,1],
                c=ListedColormap(('red','green','blue'))(i),label=j)
plt.title('Linear Discriminant Analysis Training Data')
plt.xlabel('LD1')
plt.ylabel('LD2')
plt.legend()
plt.show()

#Plot the test set results for the LDA
from matplotlib.colors import ListedColormap
X_set,y_set=X_test,y_test
X1,X2=np.meshgrid(np.arange(start=X_set[:,0].min()-
1,stop=X_set[:,0].max()+1,step=0.01),
                  np.arange(start=X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1,
step = 0.01))
plt.contourf(X1,X2,wine_LDA_classifier.predict(np.array([X1.ravel(),X2.ravel()]).T).re
shape(X1.shape),
             alpha=0.75,cmap=ListedColormap(('red','green','blue')))
plt.xlim(X1.min(),X1.max())
plt.ylim(X2.min(),X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set==j,0],X_set[y_set==j,1],
                c=ListedColormap(('red','green','blue'))(i),label=j)
plt.title('Linear Discriminant Analysis Test Data')
plt.xlabel('LD1')
plt.ylabel('LD2')
plt.legend()
plt.show()
```

The following printout shows the results of the analysis.

```
"D:\Google Drive\UMKC\PhD\Classes\Python\Labs\Lab 3\Lab3\venv\Scripts\python.exe"
"D:/Google Drive/UMKC/PhD/Classes/Python/Labs/Lab 3/Lab3/Lab3 Q1.py"

The confusion matrix for the Linear Discriminat Analysis is:

 [[14  0  0]

 [ 0 12  0]

 [ 0  0 10]]


Process finished with exit code 0
```





2) Implement Support Vector Machine classification,
   1) Choose one of the dataset using the datasets features in the scikit-learn
   2) Load the dataset
   3) According to your dataset, split the data to 20% testing data, 80% training data(you can also use any other number)
   4) Apply SVC with Linear kernel
   5) Apply SVC with RBF kernel
   6) Report the accuracy of the model on both models separately and report their differences if there is
   7) Report your view how can you increase the accuracy and which kernel is the best for your dataset and why

Results:

This code runs an SVM analysis of the imported data using the Linear Kernel.

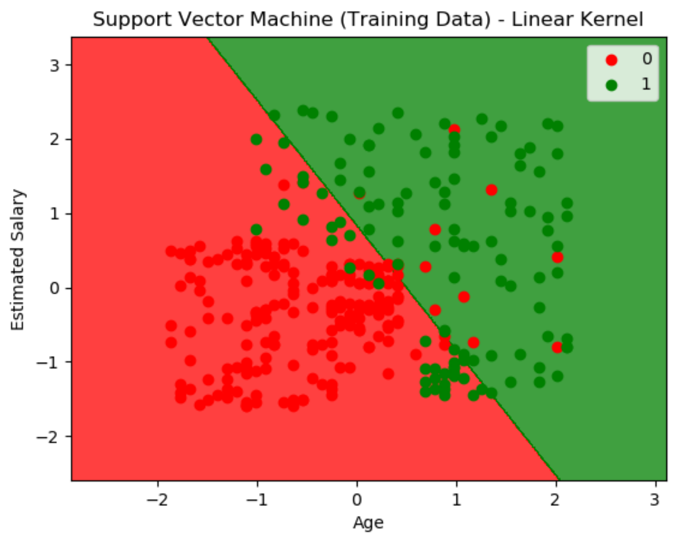
```
#Don Baker
#Comp-Sci 5590
#Lab 3, Question 2 (Linear Kernel)

#Implement Support Vector Machine classification,
#1)     Choose one of the dataset using the datasets features in the scikit-learn
#2)     Load the dataset
#3)     According to your dataset, split the data to 20% testing data, 80% training
data(you can also use any other number)
#4)     Apply SVC with Linear kernel
#5)     Apply SVC with RBF kernel
#6)     Report the accuracy of the model on both models separately and report their
differences if there is
```

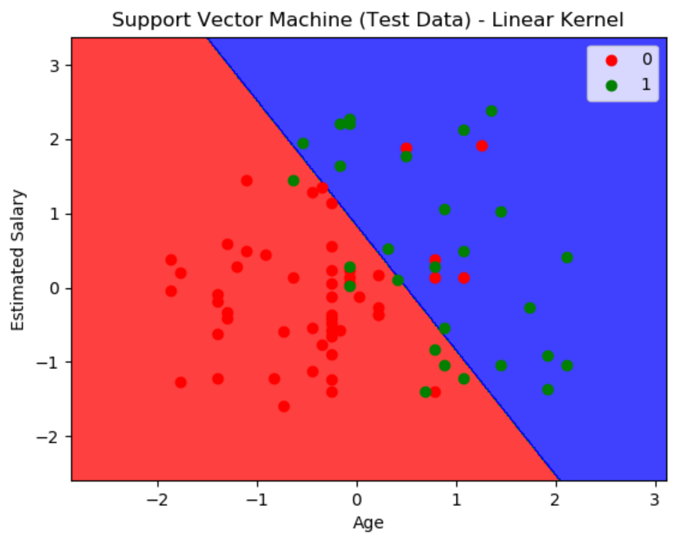
```
#7)    Report your view how can you increase the accuracy and which kernel is the best
for your dataset and why


#Guidance and example code taken from Learn&Teach YouTube videos at
https://www.youtube.com/channel/UCGD8R7ct8cQfMMuUFImqDdQ

#Import Packages/Libraries
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd

#Import Data
dataset=pd.read_csv('d:/Social_Network_Ads.csv')
X=dataset.iloc[:, [2,3]].values
y=dataset.iloc[:, 4].values

#Split the data into training and testing data sets for cross-validation
from sklearn import model_selection as cv
splits=cv.train_test_split(X,y,test_size=0.20)
X_train,X_test,y_train,y_test=splits

#Scale the data to provide more accurate evalaution
from sklearn.preprocessing import StandardScaler
Scale_X=StandardScaler()
X_train=Scale_X.fit_transform(X_train)
X_test=Scale_X.transform(X_test)

#Complete SVM using SVC - Linear Kernel

from sklearn.svm import SVC
SVC_classifier=SVC(kernel='linear', random_state=0)
SVC_classifier.fit(X_train,y_train)

#Predict the results of the model using the test set of data
y_pred_weather=SVC_classifier.predict(X_test)

#Create and display the confusion matrix for the analysis
from sklearn.metrics import confusion_matrix
confusion_SVC=confusion_matrix(y_test,y_pred_weather)
print("The confusion matrix for the SVM analysis is:\n",confusion_SVC)

#Plot the training set results for the SVM analysis
from matplotlib.colors import ListedColormap
X_set,y_set=X_train,y_train
X1,X2=np.meshgrid(np.arange(start=X_set[:,0].min()-
1,stop=X_set[:,0].max()+1,step=0.01),
                  np.arange(start=X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1,
step = 0.01))
plt.contourf(X1,X2,SVC_classifier.predict(np.array([X1.ravel(),X2.ravel()]).T).reshape
(X1.shape),
             alpha=0.75,cmap=ListedColormap(('red','green')))
plt.xlim(X1.min(),X1.max())
plt.ylim(X2.min(),X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set==j,0],X_set[y_set==j,1],
                c=ListedColormap(('red','green','blue'))(i),label=j)
plt.title('Support Vector Machine (Training Data) - Linear Kernel')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()

#Plot the test set results for the SVM analysis
from matplotlib.colors import ListedColormap
X_set,y_set=X_test,y_test
X1,X2=np.meshgrid(np.arange(start=X_set[:,0].min()-
1,stop=X_set[:,0].max()+1,step=0.01),
                  np.arange(start=X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1,
step = 0.01))
plt.contourf(X1,X2,SVC_classifier.predict(np.array([X1.ravel(),X2.ravel()]).T).reshape
(X1.shape),
             alpha=0.75,cmap=ListedColormap(('red','green','blue')))
plt.xlim(X1.min(),X1.max())
plt.ylim(X2.min(),X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set==j,0],X_set[y_set==j,1],
                c=ListedColormap(('red','green'))(i),label=j)
plt.title('Support Vector Machine (Test Data) - Linear Kernel')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

The results of this code are shown below:

```
The confusion matrix for the SVM analysis is:
 [[47  5]
 [ 8 20]]

Process finished with exit code 0
```

The following graphs illustrate the results of the SVM analysis for the training and test data.





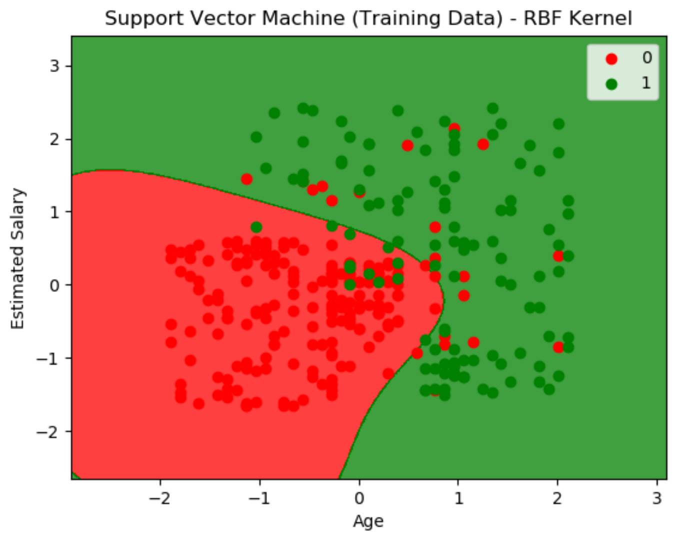
The following code is for the SVM analysis using the RBF kernel.

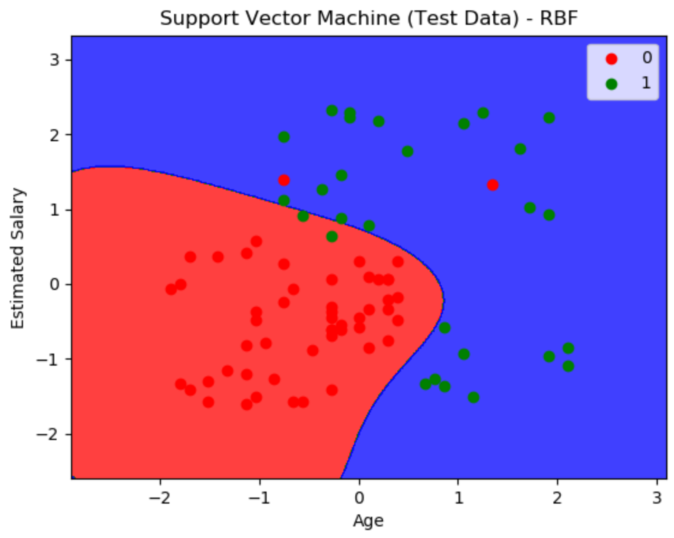
```
#Don Baker
#Comp-Sci 5590
#Lab 3, Question 2 (RBF Kernel)

#Implement Support Vector Machine classification,
#1)    Choose one of the dataset using the datasets features in the scikit-learn
#2)    Load the dataset
#3)    According to your dataset, split the data to 20% testing data, 80% training
data(you can also use any other number)
#4)    Apply SVC with Linear kernel
#5)    Apply SVC with RBF kernel
#6)    Report the accuracy of the model on both models separately and report their
differences if there is
#7)    Report your view how can you increase the accuracy and which kernel is the best
for your dataset and why


#Guidance and example code taken from Learn&Teach YouTube videos at
https://www.youtube.com/channel/UCGD8R7ct8cQfMMuUFImqDdQ

#Import Packages/Libraries
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd

#Import Data
dataset=pd.read_csv('d:/Social_Network_Ads.csv')
X=dataset.iloc[:, [2,3]].values
y=dataset.iloc[:, 4].values

#Split the data into training and testing data sets for cross-validation
from sklearn import model_selection as cv
splits=cv.train_test_split(X,y,test_size=0.20)
X_train,X_test,y_train,y_test=splits

#Scale the data to provide more accurate evalaution
```

```
from sklearn.preprocessing import StandardScaler
Scale_X=StandardScaler()
X_train=Scale_X.fit_transform(X_train)
X_test=Scale_X.transform(X_test)

#Complete SVM using SVC - RBF Kernel

from sklearn.svm import SVC
SVC_classifier=SVC(kernel='rbf', random_state=0)
SVC_classifier.fit(X_train,y_train)

#Predict the results of the model using the test set of data
y_pred_weather=SVC_classifier.predict(X_test)

#Create and display the confusion matrix for the analysis
from sklearn.metrics import confusion_matrix
confusion_SVC=confusion_matrix(y_test,y_pred_weather)
print("The confusion matrix for the SVM analysis is:\n",confusion_SVC)

#Plot the training set results for the SVM analysis
from matplotlib.colors import ListedColormap
X_set,y_set=X_train,y_train
X1,X2=np.meshgrid(np.arange(start=X_set[:,0].min()-
1,stop=X_set[:,0].max()+1,step=0.01),
                  np.arange(start=X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1,
step = 0.01))
plt.contourf(X1,X2,SVC_classifier.predict(np.array([X1.ravel(),X2.ravel()]).T).reshape
(X1.shape),
             alpha=0.75,cmap=ListedColormap(('red','green')))
plt.xlim(X1.min(),X1.max())
plt.ylim(X2.min(),X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set==j,0],X_set[y_set==j,1],
                c=ListedColormap(('red','green','blue'))(i),label=j)
plt.title('Support Vector Machine (Training Data) - RBF Kernel')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()

#Plot the test set results for the SVM analysis
from matplotlib.colors import ListedColormap
X_set,y_set=X_test,y_test
X1,X2=np.meshgrid(np.arange(start=X_set[:,0].min()-
1,stop=X_set[:,0].max()+1,step=0.01),
                  np.arange(start=X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1,
step = 0.01))
plt.contourf(X1,X2,SVC_classifier.predict(np.array([X1.ravel(),X2.ravel()]).T).reshape
(X1.shape),
             alpha=0.75,cmap=ListedColormap(('red','green','blue')))
plt.xlim(X1.min(),X1.max())
plt.ylim(X2.min(),X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set==j,0],X_set[y_set==j,1],
                c=ListedColormap(('red','green'))(i),label=j)
plt.title('Support Vector Machine (Test Data) - RBF')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

The results of the analysis are shown below in the printout and the graphs of the analysis for the training and test data.

```
The confusion matrix for the SVM analysis is:
 [[50  2]
 [ 2 26]]
```





Based on the results of the confusion matrices and the graphs, it appears that the RBF kernel provides a more accurate representation of the data.

3) Write a program
   Take an Input file. Use the simple approach below to summarize a text file:
- Read the file
- Using Lemmatization, apply lemmatization on the words
- Apply the bigram on the text
- Calculate the word frequency (bi-gram frequency) of the words (bi-grams)
- Choose top five bi-grams that has been repeated most
- Go through the original text that you had in the file
- Find all the sentences with those most repeated bi-grams
- Extract those sentences and concatenate
- Enjoy the summarization

Results:

The following code was used for the natural language analyses.

```
#Don Baker
#Comp-Sci 5590
#Lab 3, Question 3

#Take an Input file. Use the simple approach below to summarize a text file:
#- Read the file
#- Using Lemmatization, apply lemmatization on the words
#- Apply the bigram on the text
#- Calculate the word frequency (bi-gram frequency) of the words (bi-grams)
#- Choose top five bi-grams that has been repeated most
#- Go through the original text that you had in the file
#- Find all the sentences with those most repeated bi-grams
#- Extract those sentences and concatenate
#- Enjoy the summarization


#Import Packages
import nltk
from nltk.tokenize import TweetTokenizer
from nltk.stem import WordNetLemmatizer

#Read Text File
file_path = 'd:/Google Drive/UMKC/PhD/Classes/Python/Labs/Lab 3/Python_Lab3.txt'
text_file = open(file_path,'r')
text_data=text_file.read()
text_file.close()

#Using Lemmatization, apply lemmatization on the words

#First step is to tokenize the data file into individual words
tkn = TweetTokenizer() #create a tokenizer
token = tkn.tokenize(text_data)  #tokenize the data
print("Tokenize: \n",(token),"\n")
#print(token_results)

#Now lemmatize the tokenize text file (noun is th default)
wnl = WordNetLemmatizer()  #create lemmatizer
lem = [wnl.lemmatize(tkn) for tkn in token]
print("Lemmatize (nouns): \n",lem,"\n")

#Try lemmatizing looking for verbs
```

```
wnl = WordNetLemmatizer()  #create lemmatizer
lem = [wnl.lemmatize(tkn,pos="v") for tkn in token]
print("Lemmatize (verbs): \n",lem,"\n")

#Try lemmatizing looking for adjectives
wnl = WordNetLemmatizer()  #create lemmatizer
lem = [wnl.lemmatize(tkn,pos="a") for tkn in token]
print("Lemmatize (adjectives): \n",lem,"\n")

#Try lemmatizing looking for adverbs
wnl = WordNetLemmatizer()  #create lemmatizer
lem = [wnl.lemmatize(tkn,pos="r") for tkn in token]
print("Lemmatize (adverbs): \n",lem,"\n")

#Apply bigram on the text
#Grouping two words together. This includes special characters like '('
bigram = [tkn for tkn in nltk.bigrams(token)]
print("Bigram: \n",bigram)

#Calculate the Bigram Frequency
freq_bi=nltk.FreqDist(bigram)

#Find the 5 most common bigrams
bi_common=freq_bi.most_common(5)
print("\nThe 5 most common bigrams are:\n",bi_common)
```

The following printout illustrates the result of the program. The text file used was a converted Word file of the assignment.

```
"D:\Google Drive\UMKC\PhD\Classes\Python\Labs\Lab 3\Lab3\venv\Scripts\python.exe"
"D:/Google Drive/UMKC/PhD/Classes/Python/Labs/Lab 3/Lab3/Lab3 Q3.py"
Tokenize:
 ['2018', 'Spring', ':', 'COMP-SCI', '5590/490', '-', 'Special', 'Topics', 'Python',
'Programming', 'Lab', 'Assignment', '3', 'Assignment', 'Overview', 'The', 'following',
'assignment', 'focus', 'on', 'to', 'make', 'one', 'familiar', 'with', 'python',
'machine', 'learning', 'Lab', 'Assignment', '1', ')', 'Pick', 'any', 'dataset',
'from', 'the', 'dataset', 'sheet', 'in', 'class', 'sheet', 'and', 'make', 'one',
'prediction', 'model', 'using', 'your', 'imagination', 'with', 'Linear',
'Discriminant', 'Analysis', '*', '.', 'Some', 'examples', 'are', ':', 'a', '.', 'In',
'the', 'report', 'provide', 'convincible', 'explanations', 'about', 'the',
'difference', 'of', 'logistic', 'regression', 'and', 'Linear', 'Discriminant',
'Analysis', '.', 'b', '.', 'You', 'can', 'also', 'pick', 'dataset', 'of', 'your',
'own', '.', '*', 'Logistic', 'Regression', 'is', 'a', 'classification', 'algorithm',
'traditionally', 'limited', 'to', 'only', 'two-class', 'classification', 'problems',
'.', 'If', 'you', 'have', 'more', 'than', 'two', 'classes', 'then', 'the', 'Linear',
'Discriminant', 'Analysis', 'algorithm', 'is', 'the', 'preferred', 'linear',
'classification', 'technique', '.', '2', ')', 'Implement', 'Support', 'Vector',
'Machine', 'classification', ',', '1', ')', 'Choose', 'one', 'of', 'the', 'dataset',
'using', 'the', 'datasets', 'features', 'in', 'the', 'scikit-learn', '2', ')', 'Load',
'the', 'dataset', '3', ')', 'According', 'to', 'your', 'dataset', ',', 'split', 'the',
'data', 'to', '20', '%', 'testing', 'data', ',', '80', '%', 'training', 'data', '(',
'you', 'can', 'also', 'use', 'any', 'other', 'number', ')', '4', ')', 'Apply', 'SVC',
'with', 'Linear', 'kernel', '5', ')', 'Apply', 'SVC', 'with', 'RBF', 'kernel', '6',
')', 'Report', 'the', 'accuracy', 'of', 'the', 'model', 'on', 'both', 'models',
'separately', 'and', 'report', 'their', 'differences', 'if', 'there', 'is', '7', ')',
'Report', 'your', 'view', 'how', 'can', 'you', 'increase', 'the', 'accuracy', 'and',
'which', 'kernel', 'is', 'the', 'best', 'for', 'your', 'dataset', 'and', 'why', '3',
')', 'Write', 'a', 'program', 'Take', 'an', 'Input', 'file', '.', 'Use', 'the',
'simple', 'approach', 'below', 'to', 'summarize', 'a', 'text', 'file', ':', '-',
'Read', 'the', 'file', '-', 'Using', 'Lemmatization', ',', 'apply', 'lemmatization',
'on', 'the', 'words', '-', 'Apply', 'the', 'bigram', 'on', 'the', 'text', '-',
'Calculate', 'the', 'word', 'frequency', '(', 'bi-gram', 'frequency', ')', 'of',
'the', 'words', '(', 'bi-grams', ')', '-', 'Choose', 'top', 'five', 'bi-grams',
'that', 'has', 'been', 'repeated', 'most', '-', 'Go', 'through', 'the', 'original',
'text', 'that', 'you', 'had', 'in', 'the', 'file', '-', 'Find', 'all', 'the',
'sentences', 'with', 'those', 'most', 'repeated', 'bi-grams', '-', 'Extract', 'those',
'sentences', 'and', 'concatenate', '-', 'Enjoy', 'the', 'summarization', 'Submission',
'Guidelines', ':', '*', 'Submit', 'your', 'code', 'at', 'Github', 'and', 'properly',
'document', 'it', '.', 'Submit', 'screenshots', 'as', 'well', '.', '*', 'Properly',
'document', 'your', 'code', '*', 'Submit', 'your', 'report', 'to', 'UMKC',
'blackboard', 'assignment', '.', '*', 'Remember', 'report', 'similarity', 'to', 'be',
'less', 'than', '15', '%', '*', 'Use', 'following', 'link', 'to', 'submit', 'your',
'assignment', ':', 'https://goo.gl/forms/l9TitNZJ8yLCwGEW2']

Lemmatize (nouns):
 ['2018', 'Spring', ':', 'COMP-SCI', '5590/490', '-', 'Special', 'Topics', 'Python',
'Programming', 'Lab', 'Assignment', '3', 'Assignment', 'Overview', 'The', 'following',
'assignment', 'focus', 'on', 'to', 'make', 'one', 'familiar', 'with', 'python',
'machine', 'learning', 'Lab', 'Assignment', '1', ')', 'Pick', 'any', 'dataset',
'from', 'the', 'dataset', 'sheet', 'in', 'class', 'sheet', 'and', 'make', 'one',
'prediction', 'model', 'using', 'your', 'imagination', 'with', 'Linear',
'Discriminant', 'Analysis', '*', '.', 'Some', 'example', 'are', ':', 'a', '.', 'In',
'the', 'report', 'provide', 'convincible', 'explanation', 'about', 'the',
'difference', 'of', 'logistic', 'regression', 'and', 'Linear', 'Discriminant',
'Analysis', '.', 'b', '.', 'You', 'can', 'also', 'pick', 'dataset', 'of', 'your',
'own', '.', '*', 'Logistic', 'Regression', 'is', 'a', 'classification', 'algorithm',
'traditionally', 'limited', 'to', 'only', 'two-class', 'classification', 'problem',
'.', 'If', 'you', 'have', 'more', 'than', 'two', 'class', 'then', 'the', 'Linear',
'Discriminant', 'Analysis', 'algorithm', 'is', 'the', 'preferred', 'linear',
'classification', 'technique', '.', '2', ')', 'Implement', 'Support', 'Vector',
'Machine', 'classification', ',', '1', ')', 'Choose', 'one', 'of', 'the', 'dataset',
'using', 'the', 'datasets', 'feature', 'in', 'the', 'scikit-learn', '2', ')', 'Load',
'the', 'dataset', '3', ')', 'According', 'to', 'your', 'dataset', ',', 'split', 'the',
'data', 'to', '20', '%', 'testing', 'data', ',', '80', '%', 'training', 'data', '(',
'you', 'can', 'also', 'use', 'any', 'other', 'number', ')', '4', ')', 'Apply', 'SVC',
'with', 'Linear', 'kernel', '5', ')', 'Apply', 'SVC', 'with', 'RBF', 'kernel', '6',
')', 'Report', 'the', 'accuracy', 'of', 'the', 'model', 'on', 'both', 'model',
'separately', 'and', 'report', 'their', 'difference', 'if', 'there', 'is', '7', ')',
'Report', 'your', 'view', 'how', 'can', 'you', 'increase', 'the', 'accuracy', 'and',
'which', 'kernel', 'is', 'the', 'best', 'for', 'your', 'dataset', 'and', 'why', '3',
')', 'Write', 'a', 'program', 'Take', 'an', 'Input', 'file', '.', 'Use', 'the',
'simple', 'approach', 'below', 'to', 'summarize', 'a', 'text', 'file', ':', '-',
'Read', 'the', 'file', '-', 'Using', 'Lemmatization', ',', 'apply', 'lemmatization',
'on', 'the', 'word', '-', 'Apply', 'the', 'bigram', 'on', 'the', 'text', '-',
'Calculate', 'the', 'word', 'frequency', '(', 'bi-gram', 'frequency', ')', 'of',
'the', 'word', '(', 'bi-grams', ')', '-', 'Choose', 'top', 'five', 'bi-grams', 'that',
'ha', 'been', 'repeated', 'most', '-', 'Go', 'through', 'the', 'original', 'text',
'that', 'you', 'had', 'in', 'the', 'file', '-', 'Find', 'all', 'the', 'sentence',
'with', 'those', 'most', 'repeated', 'bi-grams', '-', 'Extract', 'those', 'sentence',
'and', 'concatenate', '-', 'Enjoy', 'the', 'summarization', 'Submission',
'Guidelines', ':', '*', 'Submit', 'your', 'code', 'at', 'Github', 'and', 'properly',
'document', 'it', '.', 'Submit', 'screenshots', 'a', 'well', '.', '*', 'Properly',
'document', 'your', 'code', '*', 'Submit', 'your', 'report', 'to', 'UMKC',
'blackboard', 'assignment', '.', '*', 'Remember', 'report', 'similarity', 'to', 'be',
'le', 'than', '15', '%', '*', 'Use', 'following', 'link', 'to', 'submit', 'your',
'assignment', ':', 'https://goo.gl/forms/l9TitNZJ8yLCwGEW2']

Lemmatize (verbs):
 ['2018', 'Spring', ':', 'COMP-SCI', '5590/490', '-', 'Special', 'Topics', 'Python',
'Programming', 'Lab', 'Assignment', '3', 'Assignment', 'Overview', 'The', 'follow',
'assignment', 'focus', 'on', 'to', 'make', 'one', 'familiar', 'with', 'python',
'machine', 'learn', 'Lab', 'Assignment', '1', ')', 'Pick', 'any', 'dataset', 'from',
'the', 'dataset', 'sheet', 'in', 'class', 'sheet', 'and', 'make', 'one', 'prediction',
'model', 'use', 'your', 'imagination', 'with', 'Linear', 'Discriminant', 'Analysis',
'*', '.', 'Some', 'examples', 'be', ':', 'a', '.', 'In', 'the', 'report', 'provide',
'convincible', 'explanations', 'about', 'the', 'difference', 'of', 'logistic',
'regression', 'and', 'Linear', 'Discriminant', 'Analysis', '.', 'b', '.', 'You',
'can', 'also', 'pick', 'dataset', 'of', 'your', 'own', '.', '*', 'Logistic',
'Regression', 'be', 'a', 'classification', 'algorithm', 'traditionally', 'limit',
'to', 'only', 'two-class', 'classification', 'problems', '.', 'If', 'you', 'have',
'more', 'than', 'two', 'class', 'then', 'the', 'Linear', 'Discriminant', 'Analysis',
'algorithm', 'be', 'the', 'prefer', 'linear', 'classification', 'technique', '.', '2',
')', 'Implement', 'Support', 'Vector', 'Machine', 'classification', ',', '1', ')',
'Choose', 'one', 'of', 'the', 'dataset', 'use', 'the', 'datasets', 'feature', 'in',
'the', 'scikit-learn', '2', ')', 'Load', 'the', 'dataset', '3', ')', 'According',
'to', 'your', 'dataset', ',', 'split', 'the', 'data', 'to', '20', '%', 'test', 'data',
',', '80', '%', 'train', 'data', '(', 'you', 'can', 'also', 'use', 'any', 'other',
'number', ')', '4', ')', 'Apply', 'SVC', 'with', 'Linear', 'kernel', '5', ')',
'Apply', 'SVC', 'with', 'RBF', 'kernel', '6', ')', 'Report', 'the', 'accuracy', 'of',
'the', 'model', 'on', 'both', 'model', 'separately', 'and', 'report', 'their',
'differences', 'if', 'there', 'be', '7', ')', 'Report', 'your', 'view', 'how', 'can',
'you', 'increase', 'the', 'accuracy', 'and', 'which', 'kernel', 'be', 'the', 'best',
'for', 'your', 'dataset', 'and', 'why', '3', ')', 'Write', 'a', 'program', 'Take',
'an', 'Input', 'file', '.', 'Use', 'the', 'simple', 'approach', 'below', 'to',
'summarize', 'a', 'text', 'file', ':', '-', 'Read', 'the', 'file', '-', 'Using',
'Lemmatization', ',', 'apply', 'lemmatization', 'on', 'the', 'word', '-', 'Apply',
'the', 'bigram', 'on', 'the', 'text', '-', 'Calculate', 'the', 'word', 'frequency',
'(', 'bi-gram', 'frequency', ')', 'of', 'the', 'word', '(', 'bi-grams', ')', '-',
'Choose', 'top', 'five', 'bi-grams', 'that', 'have', 'be', 'repeat', 'most', '-',
'Go', 'through', 'the', 'original', 'text', 'that', 'you', 'have', 'in', 'the',
'file', '-', 'Find', 'all', 'the', 'sentence', 'with', 'those', 'most', 'repeat', 'bi-
grams', '-', 'Extract', 'those', 'sentence', 'and', 'concatenate', '-', 'Enjoy',
'the', 'summarization', 'Submission', 'Guidelines', ':', '*', 'Submit', 'your',
'code', 'at', 'Github', 'and', 'properly', 'document', 'it', '.', 'Submit',
'screenshots', 'as', 'well', '.', '*', 'Properly', 'document', 'your', 'code', '*',
'Submit', 'your', 'report', 'to', 'UMKC', 'blackboard', 'assignment', '.', '*',
'Remember', 'report', 'similarity', 'to', 'be', 'less', 'than', '15', '%', '*', 'Use',
'follow', 'link', 'to', 'submit', 'your', 'assignment', ':',
'https://goo.gl/forms/l9TitNZJ8yLCwGEW2']

Lemmatize (adjectives):
 ['2018', 'Spring', ':', 'COMP-SCI', '5590/490', '-', 'Special', 'Topics', 'Python',
'Programming', 'Lab', 'Assignment', '3', 'Assignment', 'Overview', 'The', 'following',
'assignment', 'focus', 'on', 'to', 'make', 'one', 'familiar', 'with', 'python',
'machine', 'learning', 'Lab', 'Assignment', '1', ')', 'Pick', 'any', 'dataset',
'from', 'the', 'dataset', 'sheet', 'in', 'class', 'sheet', 'and', 'make', 'one',
'prediction', 'model', 'using', 'your', 'imagination', 'with', 'Linear',
'Discriminant', 'Analysis', '*', '.', 'Some', 'examples', 'are', ':', 'a', '.', 'In',
'the', 'report', 'provide', 'convincible', 'explanations', 'about', 'the',
'difference', 'of', 'logistic', 'regression', 'and', 'Linear', 'Discriminant',
'Analysis', '.', 'b', '.', 'You', 'can', 'also', 'pick', 'dataset', 'of', 'your',
'own', '.', '*', 'Logistic', 'Regression', 'is', 'a', 'classification', 'algorithm',
'traditionally', 'limited', 'to', 'only', 'two-class', 'classification', 'problems',
'.', 'If', 'you', 'have', 'more', 'than', 'two', 'classes', 'then', 'the', 'Linear',
'Discriminant', 'Analysis', 'algorithm', 'is', 'the', 'preferred', 'linear',
'classification', 'technique', '.', '2', ')', 'Implement', 'Support', 'Vector',
'Machine', 'classification', ',', '1', ')', 'Choose', 'one', 'of', 'the', 'dataset',
'using', 'the', 'datasets', 'features', 'in', 'the', 'scikit-learn', '2', ')', 'Load',
'the', 'dataset', '3', ')', 'According', 'to', 'your', 'dataset', ',', 'split', 'the',
'data', 'to', '20', '%', 'testing', 'data', ',', '80', '%', 'training', 'data', '(',
'you', 'can', 'also', 'use', 'any', 'other', 'number', ')', '4', ')', 'Apply', 'SVC',
'with', 'Linear', 'kernel', '5', ')', 'Apply', 'SVC', 'with', 'RBF', 'kernel', '6',
')', 'Report', 'the', 'accuracy', 'of', 'the', 'model', 'on', 'both', 'models',
'separately', 'and', 'report', 'their', 'differences', 'if', 'there', 'is', '7', ')',
'Report', 'your', 'view', 'how', 'can', 'you', 'increase', 'the', 'accuracy', 'and',
'which', 'kernel', 'is', 'the', 'best', 'for', 'your', 'dataset', 'and', 'why', '3',
')', 'Write', 'a', 'program', 'Take', 'an', 'Input', 'file', '.', 'Use', 'the',
'simple', 'approach', 'below', 'to', 'summarize', 'a', 'text', 'file', ':', '-',
'Read', 'the', 'file', '-', 'Using', 'Lemmatization', ',', 'apply', 'lemmatization',
'on', 'the', 'words', '-', 'Apply', 'the', 'bigram', 'on', 'the', 'text', '-',
'Calculate', 'the', 'word', 'frequency', '(', 'bi-gram', 'frequency', ')', 'of',
'the', 'words', '(', 'bi-grams', ')', '-', 'Choose', 'top', 'five', 'bi-grams',
'that', 'has', 'been', 'repeated', 'most', '-', 'Go', 'through', 'the', 'original',
'text', 'that', 'you', 'had', 'in', 'the', 'file', '-', 'Find', 'all', 'the',
'sentences', 'with', 'those', 'most', 'repeated', 'bi-grams', '-', 'Extract', 'those',
'sentences', 'and', 'concatenate', '-', 'Enjoy', 'the', 'summarization', 'Submission',
'Guidelines', ':', '*', 'Submit', 'your', 'code', 'at', 'Github', 'and', 'properly',
'document', 'it', '.', 'Submit', 'screenshots', 'as', 'well', '.', '*', 'Properly',
'document', 'your', 'code', '*', 'Submit', 'your', 'report', 'to', 'UMKC',
'blackboard', 'assignment', '.', '*', 'Remember', 'report', 'similarity', 'to', 'be',
'less', 'than', '15', '%', '*', 'Use', 'following', 'link', 'to', 'submit', 'your',
'assignment', ':', 'https://goo.gl/forms/l9TitNZJ8yLCwGEW2']

Lemmatize (adverbs):
 ['2018', 'Spring', ':', 'COMP-SCI', '5590/490', '-', 'Special', 'Topics', 'Python',
'Programming', 'Lab', 'Assignment', '3', 'Assignment', 'Overview', 'The', 'following',
'assignment', 'focus', 'on', 'to', 'make', 'one', 'familiar', 'with', 'python',
'machine', 'learning', 'Lab', 'Assignment', '1', ')', 'Pick', 'any', 'dataset',
'from', 'the', 'dataset', 'sheet', 'in', 'class', 'sheet', 'and', 'make', 'one',
'prediction', 'model', 'using', 'your', 'imagination', 'with', 'Linear',
'Discriminant', 'Analysis', '*', '.', 'Some', 'examples', 'are', ':', 'a', '.', 'In',
'the', 'report', 'provide', 'convincible', 'explanations', 'about', 'the',
'difference', 'of', 'logistic', 'regression', 'and', 'Linear', 'Discriminant',
'Analysis', '.', 'b', '.', 'You', 'can', 'also', 'pick', 'dataset', 'of', 'your',
'own', '.', '*', 'Logistic', 'Regression', 'is', 'a', 'classification', 'algorithm',
'traditionally', 'limited', 'to', 'only', 'two-class', 'classification', 'problems',
'.', 'If', 'you', 'have', 'more', 'than', 'two', 'classes', 'then', 'the', 'Linear',
'Discriminant', 'Analysis', 'algorithm', 'is', 'the', 'preferred', 'linear',
'classification', 'technique', '.', '2', ')', 'Implement', 'Support', 'Vector',
'Machine', 'classification', ',', '1', ')', 'Choose', 'one', 'of', 'the', 'dataset',
'using', 'the', 'datasets', 'features', 'in', 'the', 'scikit-learn', '2', ')', 'Load',
'the', 'dataset', '3', ')', 'According', 'to', 'your', 'dataset', ',', 'split', 'the',
'data', 'to', '20', '%', 'testing', 'data', ',', '80', '%', 'training', 'data', '(',
'you', 'can', 'also', 'use', 'any', 'other', 'number', ')', '4', ')', 'Apply', 'SVC',
'with', 'Linear', 'kernel', '5', ')', 'Apply', 'SVC', 'with', 'RBF', 'kernel', '6',
')', 'Report', 'the', 'accuracy', 'of', 'the', 'model', 'on', 'both', 'models',
'separately', 'and', 'report', 'their', 'differences', 'if', 'there', 'is', '7', ')',
'Report', 'your', 'view', 'how', 'can', 'you', 'increase', 'the', 'accuracy', 'and',
'which', 'kernel', 'is', 'the', 'best', 'for', 'your', 'dataset', 'and', 'why', '3',
')', 'Write', 'a', 'program', 'Take', 'an', 'Input', 'file', '.', 'Use', 'the',
'simple', 'approach', 'below', 'to', 'summarize', 'a', 'text', 'file', ':', '-',
'Read', 'the', 'file', '-', 'Using', 'Lemmatization', ',', 'apply', 'lemmatization',
'on', 'the', 'words', '-', 'Apply', 'the', 'bigram', 'on', 'the', 'text', '-',
'Calculate', 'the', 'word', 'frequency', '(', 'bi-gram', 'frequency', ')', 'of',
'the', 'words', '(', 'bi-grams', ')', '-', 'Choose', 'top', 'five', 'bi-grams',
'that', 'has', 'been', 'repeated', 'most', '-', 'Go', 'through', 'the', 'original',
'text', 'that', 'you', 'had', 'in', 'the', 'file', '-', 'Find', 'all', 'the',
'sentences', 'with', 'those', 'most', 'repeated', 'bi-grams', '-', 'Extract', 'those',
'sentences', 'and', 'concatenate', '-', 'Enjoy', 'the', 'summarization', 'Submission',
'Guidelines', ':', '*', 'Submit', 'your', 'code', 'at', 'Github', 'and', 'properly',
'document', 'it', '.', 'Submit', 'screenshots', 'as', 'well', '.', '*', 'Properly',
'document', 'your', 'code', '*', 'Submit', 'your', 'report', 'to', 'UMKC',
'blackboard', 'assignment', '.', '*', 'Remember', 'report', 'similarity', 'to', 'be',
'less', 'than', '15', '%', '*', 'Use', 'following', 'link', 'to', 'submit', 'your',
'assignment', ':', 'https://goo.gl/forms/l9TitNZJ8yLCwGEW2']

Bigram:
 [('2018', 'Spring'), ('Spring', ':'), (':', 'COMP-SCI'), ('COMP-SCI', '5590/490'),
('5590/490', '-'), ('-', 'Special'), ('Special', 'Topics'), ('Topics', 'Python'),
('Python', 'Programming'), ('Programming', 'Lab'), ('Lab', 'Assignment'),
('Assignment', '3'), ('3', 'Assignment'), ('Assignment', 'Overview'), ('Overview',
'The'), ('The', 'following'), ('following', 'assignment'), ('assignment', 'focus'),
('focus', 'on'), ('on', 'to'), ('to', 'make'), ('make', 'one'), ('one', 'familiar'),
('familiar', 'with'), ('with', 'python'), ('python', 'machine'), ('machine',
'learning'), ('learning', 'Lab'), ('Lab', 'Assignment'), ('Assignment', '1'), ('1',
')'), (')', 'Pick'), ('Pick', 'any'), ('any', 'dataset'), ('dataset', 'from'),
('from', 'the'), ('the', 'dataset'), ('dataset', 'sheet'), ('sheet', 'in'), ('in',
'class'), ('class', 'sheet'), ('sheet', 'and'), ('and', 'make'), ('make', 'one'),
('one', 'prediction'), ('prediction', 'model'), ('model', 'using'), ('using', 'your'),
('your', 'imagination'), ('imagination', 'with'), ('with', 'Linear'), ('Linear',
'Discriminant'), ('Discriminant', 'Analysis'), ('Analysis', '*'), ('*', '.'), ('.',
'Some'), ('Some', 'examples'), ('examples', 'are'), ('are', ':'), (':', 'a'), ('a',
'.'), ('.', 'In'), ('In', 'the'), ('the', 'report'), ('report', 'provide'),
('provide', 'convincible'), ('convincible', 'explanations'), ('explanations',
'about'), ('about', 'the'), ('the', 'difference'), ('difference', 'of'), ('of',
'logistic'), ('logistic', 'regression'), ('regression', 'and'), ('and', 'Linear'),
('Linear', 'Discriminant'), ('Discriminant', 'Analysis'), ('Analysis', '.'), ('.',
'b'), ('b', '.'), ('.', 'You'), ('You', 'can'), ('can', 'also'), ('also', 'pick'),
('pick', 'dataset'), ('dataset', 'of'), ('of', 'your'), ('your', 'own'), ('own', '.'),
('.', '*'), ('*', 'Logistic'), ('Logistic', 'Regression'), ('Regression', 'is'),
('is', 'a'), ('a', 'classification'), ('classification', 'algorithm'), ('algorithm',
'traditionally'), ('traditionally', 'limited'), ('limited', 'to'), ('to', 'only'),
('only', 'two-class'), ('two-class', 'classification'), ('classification',
'problems'), ('problems', '.'), ('.', 'If'), ('If', 'you'), ('you', 'have'), ('have',
'more'), ('more', 'than'), ('than', 'two'), ('two', 'classes'), ('classes', 'then'),
('then', 'the'), ('the', 'Linear'), ('Linear', 'Discriminant'), ('Discriminant',
'Analysis'), ('Analysis', 'algorithm'), ('algorithm', 'is'), ('is', 'the'), ('the',
'preferred'), ('preferred', 'linear'), ('linear', 'classification'),
('classification', 'technique'), ('technique', '.'), ('.', '2'), ('2', ')'), (')',
'Implement'), ('Implement', 'Support'), ('Support', 'Vector'), ('Vector', 'Machine'),
('Machine', 'classification'), ('classification', ','), (',', '1'), ('1', ')'), (')',
'Choose'), ('Choose', 'one'), ('one', 'of'), ('of', 'the'), ('the', 'dataset'),
('dataset', 'using'), ('using', 'the'), ('the', 'datasets'), ('datasets', 'features'),
('features', 'in'), ('in', 'the'), ('the', 'scikit-learn'), ('scikit-learn', '2'),
('2', ')'), (')', 'Load'), ('Load', 'the'), ('the', 'dataset'), ('dataset', '3'),
('3', ')'), (')', 'According'), ('According', 'to'), ('to', 'your'), ('your',
'dataset'), ('dataset', ','), (',', 'split'), ('split', 'the'), ('the', 'data'),
('data', 'to'), ('to', '20'), ('20', '%'), ('%', 'testing'), ('testing', 'data'),
('data', ','), (',', '80'), ('80', '%'), ('%', 'training'), ('training', 'data'),
('data', '('), ('(', 'you'), ('you', 'can'), ('can', 'also'), ('also', 'use'), ('use',
'any'), ('any', 'other'), ('other', 'number'), ('number', ')'), (')', '4'), ('4',
')'), (')', 'Apply'), ('Apply', 'SVC'), ('SVC', 'with'), ('with', 'Linear'),
('Linear', 'kernel'), ('kernel', '5'), ('5', ')'), (')', 'Apply'), ('Apply', 'SVC'),
('SVC', 'with'), ('with', 'RBF'), ('RBF', 'kernel'), ('kernel', '6'), ('6', ')'),
(')', 'Report'), ('Report', 'the'), ('the', 'accuracy'), ('accuracy', 'of'), ('of',
'the'), ('the', 'model'), ('model', 'on'), ('on', 'both'), ('both', 'models'),
('models', 'separately'), ('separately', 'and'), ('and', 'report'), ('report',
'their'), ('their', 'differences'), ('differences', 'if'), ('if', 'there'), ('there',
'is'), ('is', '7'), ('7', ')'), (')', 'Report'), ('Report', 'your'), ('your', 'view'),
('view', 'how'), ('how', 'can'), ('can', 'you'), ('you', 'increase'), ('increase',
'the'), ('the', 'accuracy'), ('accuracy', 'and'), ('and', 'which'), ('which',
'kernel'), ('kernel', 'is'), ('is', 'the'), ('the', 'best'), ('best', 'for'), ('for',
'your'), ('your', 'dataset'), ('dataset', 'and'), ('and', 'why'), ('why', '3'), ('3',
')'), (')', 'Write'), ('Write', 'a'), ('a', 'program'), ('program', 'Take'), ('Take',
'an'), ('an', 'Input'), ('Input', 'file'), ('file', '.'), ('.', 'Use'), ('Use',
'the'), ('the', 'simple'), ('simple', 'approach'), ('approach', 'below'), ('below',
'to'), ('to', 'summarize'), ('summarize', 'a'), ('a', 'text'), ('text', 'file'),
('file', ':'), (':', '-'), ('-', 'Read'), ('Read', 'the'), ('the', 'file'), ('file',
'-'), ('-', 'Using'), ('Using', 'Lemmatization'), ('Lemmatization', ','), (',',
'apply'), ('apply', 'lemmatization'), ('lemmatization', 'on'), ('on', 'the'), ('the',
'words'), ('words', '-'), ('-', 'Apply'), ('Apply', 'the'), ('the', 'bigram'),
('bigram', 'on'), ('on', 'the'), ('the', 'text'), ('text', '-'), ('-', 'Calculate'),
('Calculate', 'the'), ('the', 'word'), ('word', 'frequency'), ('frequency', '('),
('(', 'bi-gram'), ('bi-gram', 'frequency'), ('frequency', ')'), (')', 'of'), ('of',
'the'), ('the', 'words'), ('words', '('), ('(', 'bi-grams'), ('bi-grams', ')'), (')',
'-'), ('-', 'Choose'), ('Choose', 'top'), ('top', 'five'), ('five', 'bi-grams'), ('bi-
grams', 'that'), ('that', 'has'), ('has', 'been'), ('been', 'repeated'), ('repeated',
'most'), ('most', '-'), ('-', 'Go'), ('Go', 'through'), ('through', 'the'), ('the',
'original'), ('original', 'text'), ('text', 'that'), ('that', 'you'), ('you', 'had'),
('had', 'in'), ('in', 'the'), ('the', 'file'), ('file', '-'), ('-', 'Find'), ('Find',
'all'), ('all', 'the'), ('the', 'sentences'), ('sentences', 'with'), ('with',
'those'), ('those', 'most'), ('most', 'repeated'), ('repeated', 'bi-grams'), ('bi-
grams', '-'), ('-', 'Extract'), ('Extract', 'those'), ('those', 'sentences'),
('sentences', 'and'), ('and', 'concatenate'), ('concatenate', '-'), ('-', 'Enjoy'),
('Enjoy', 'the'), ('the', 'summarization'), ('summarization', 'Submission'),
('Submission', 'Guidelines'), ('Guidelines', ':'), (':', '*'), ('*', 'Submit'),
('Submit', 'your'), ('your', 'code'), ('code', 'at'), ('at', 'Github'), ('Github',
'and'), ('and', 'properly'), ('properly', 'document'), ('document', 'it'), ('it',
'.'), ('.', 'Submit'), ('Submit', 'screenshots'), ('screenshots', 'as'), ('as',
'well'), ('well', '.'), ('.', '*'), ('*', 'Properly'), ('Properly', 'document'),
('document', 'your'), ('your', 'code'), ('code', '*'), ('*', 'Submit'), ('Submit',
'your'), ('your', 'report'), ('report', 'to'), ('to', 'UMKC'), ('UMKC', 'blackboard'),
('blackboard', 'assignment'), ('assignment', '.'), ('.', '*'), ('*', 'Remember'),
('Remember', 'report'), ('report', 'similarity'), ('similarity', 'to'), ('to', 'be'),
('be', 'less'), ('less', 'than'), ('than', '15'), ('15', '%'), ('%', '*'), ('*',
'Use'), ('Use', 'following'), ('following', 'link'), ('link', 'to'), ('to', 'submit'),
('submit', 'your'), ('your', 'assignment'), ('assignment', ':'), (':',
'https://goo.gl/forms/l9TitNZJ8yLCwGEW2')]

The 5 most common bigrams are:
 [(('the', 'dataset'), 3), (('Linear', 'Discriminant'), 3), (('Discriminant',
'Analysis'), 3), (('.', '*'), 3), (('of', 'the'), 3)]

Process finished with exit code 0
```

4) Report your views on the k nearest neighbor algorithm when we change the K how it will affect the accuracy. Provide a good justification about the changes of the accuracy when we change the amount of K.
   For example: compare the accuracy when K=1 and K is a big number like 50, why the accuracy will change

Results:

Increasing the K value from one in the KNN analysis increases the number of neighboring data points in the analysis and provide greater accuracy. However, excessively high values of K cause the boundaries between the classes to become less distinct. The best value is greater than 1 but not too large. Research on the internet indicates that a value of around K=10 provides the best results.

Don Baker
Class ID 6

Submission Guidelines:

- Submit your code at Github and properly document it. Submit your screenshots as well.
- Properly document your code
- Submit only the code portion in text file to UMKC blackboard assignment.
- Remember code similarity to be less than 45%
- Use following link to submit your assignment:

https://goo.gl/forms/cxvY8Kg1pvNNzrpw1