Machine Learning and NLP

Name: Rudy Martinez

abc123: Lpe538

Blank notebook to be used for class exercises.

Exercise 1

Write code to load the data in the "iris.csv" into numpy arrays.

The frst 4 columns are the features/attributes. The last column is the class. Simply load the class as a list of strings. Don't forget to convert the dataset into a numpy array. You can use either DictVectorizer or the CSV method on the previous slide to load the features.

In [1]:
```
with open('iris.csv') as in_file:
    count = 0
    for row in in_file:
        print(row.strip())
        count += 1
        if count == 10:
            break
```

```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
```

In [118…
```
import csv
import numpy as np
from sklearn.feature_extraction import DictVectorizer

X_list_dict = []
```

```
Y_list = []

with open('iris.csv') as file:
    csv = csv.reader(file, delimiter = ',')

    for row in csv:
        features = {'feature_1': float(row[0]),
                    'feature_2': float(row[1]),
                    'feature_3': float(row[2]),
                    'feature_4': float(row[3])}
        X_list_dict.append(features)
        Y_list.append(row[4])

vec = DictVectorizer(sparse = False)
X_1 = vec.fit_transform(X_list_dict)
Y_1 = np.array(Y_list)
```

Exercise 2

Using the iris data you loaded in Exercise 1, do the following:

- Use train_test_split() to divide the iris dataset. (use 0.2 for the test size). Set random_state to 42.
- Train an SVM on the train split and evaluate using accuracy on the test split.
- Fiddle with the parameters of the SVM to see how it effects the performance.
- Calculate the accuracy on the train split. Is there a difference between the train/test accuracies?

Next, try using a different classifier, a random forest, and see how it compares to the SVM

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

Note that this is a toy dataset, so all scores will be high.

In [157…

```
from sklearn.svm import SVC, LinearSVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, f1_score
from sklearn.model_selection import train_test_split, GridSearchCV
import random

np.random.seed(42)
random.seed(42)

X_train, X_val, y_train, y_val = train_test_split(X_1, Y_1, test_size = 0.2, random_state = 42)
```

```
clf = SVC(C = 0.1)
clf.fit(X_train, y_train)

preds_1 = clf.predict(X_val)
print(f"Test: {accuracy_score(y_val, preds_1)}")

preds_2 = clf.predict(X_train)
print(f"Train: {accuracy_score(y_train, preds_2)}")
```

```
Test: 0.9666666666666667
Train: 0.925
```

Random Forest - Exercise 2

In [156…

```
np.random.seed(42)
random.seed(42)

X_train_, X_val_, y_train_, y_val_ = train_test_split(X_1, Y_1, test_size = 0.2, random_state = 42)


clf = RandomForestClassifier()
clf.fit(X_train_, y_train_)

preds_3 = clf.predict(X_val_)
print(f"Test: {accuracy_score(y_val_, preds_3)}")

preds_4 = clf.predict(X_train_)
print(f"Train: {accuracy_score(y_train_, preds_4)}")
```

```
Test: 1.0
Train: 1.0
```

Exercise 3

Using the train/test iris dataset split from exercise 2. Train a model on the training dataset using GridSearchCV with the SVC kernel parameters "rbf" and "linear",and C parameters 0.001, 0.01, 0.1, 1., and 10. Print the training and validation scores for the best set of parameters.

In [121…

```
X_train, X_test, y_train, y_test = train_test_split(X_1, Y_1, test_size = 0.2, random_state = 42)

parameters = {'kernel': ('linear', 'rbf'), 'C':[0.001, 0.01, 0.1, 1.,10.]}

c = SVC()
```

```
clf = GridSearchCV(c, parameters, cv = 3)

clf.fit(X_train, y_train)

print(f"Best Params: {clf.best_params_}")
print(f"Best Score: {clf.best_score_}\n")

preds_5 = clf.predict(X_test)
print(f"Test: {accuracy_score(y_test, preds_5)}")

preds_6 = clf.predict(X_train)
print(f"Train: {accuracy_score(y_train, preds_6)}")
```

```
Best Params: {'C': 1.0, 'kernel': 'linear'}
Best Score: 0.975

Test: 1.0
Train: 0.975
```

Exercise 4

The tab (\t) separated file "sentiment-twitter-data.tsv" contains tweets annotated for sentiment. Load the data then do the following:

- split the dataset into a train/test split.
- create a bag of words feature representation for the tweets using the CountVectorizer
- Use grid-search (CV) on the train split to find the best C parameters for a LinearSVC classifier. Only test 2 C values to reduce overhead (0.1 and 1.). Also, use a 2-fold CV, i.e., cv=2.
- report (print) the accuracy, micro F1, and macro F1 of the final classifier on the test data and train data
- How many features were created with the bag of words representation?

file path: ./sentiment-twitter-data.tsv

In [1]:
```
# This is a tab seperated file, so with csv reader use delimiter="\t"
with open('./sentiment-twitter-data.tsv') as in_file:
    count = 0
    for row in in_file:
        print(row.strip())
        count += 1
        if count == 10:
            break
```

```
264183816548130816      15140428        positive        Gas by my house hit $3.39!!!! I'm going to Chapel Hill
on Sat. :)
```

```
264249301910310912      18516728        negative        Iranian general says Israel's Iron Dome can't deal with
their missiles (keep talking like that and we may end up finding out)
264105751826538497      147088367       positive        with J Davlar 11th. Main rivals are team Poland. Hopefu
lly we an make it a successful end to a tough week of training tomorrow.
264094586689953794      332474633       negative        Talking about ACT's &amp;&amp; SAT's, deciding where I
want to go to college, applying to colleges and everything about college stresses me out.
254941790757601280      557103111       negative        They may have a SuperBowl in Dallas, but Dallas ain't w
inning a SuperBowl. Not with that quarterback and owner. @S4NYC @RasmussenPoll
264169034155696130      382403760       neutral Im bringing the monster load of candy tomorrow, I just hope it
doesn't get all squiched
263192091700654080      344222239       objective-OR-neutral    Apple software, retail chiefs out in overhaul:
SAN FRANCISCO Apple Inc CEO Tim Cook on Monday replaced the heads... http://t.co/X49ZEOsG
263398998675693568      812957996       positive        @oluoch @victor_otti @kunjand I just watched it! Sridev
i's comeback.... U remember her from the 90s?? Sun mornings on NTA ;)
260200142420992000      332530284       objective       #Livewire Nadal confirmed for Mexican Open in February:
Rafael Nadal is set to play at the Me... http://t.co/zgUXpcnC #LiveWireAthletics
264087629237202944      61903760        positive        @MsSheLahY I didnt want to just pop up... but yep we ha
ve chapel hill next wednesday you should come.. and shes great ill tell her you asked
```

Load the Data

In [183…

```python
import csv

X_text = []
y = []


with open('sentiment-twitter-data.tsv') as file:
    csv = csv.reader(file, delimiter = '\t')

    for row in csv:
        X_text.append(row[3])
        y.append(row[2])

X_text = np.array(X_text)
y = np.array(y)
```

Split the Data (Train / Test Split)

In [184…

```python
np.random.seed(42)
random.seed(42)

X_text_train, X_text_test, y_train, y_test = train_test_split(X_text, y, test_size = 0.2, random_state = 42)
```

Use grid-search (CV) on the train split to find the best C parameters for a LinearSVC classifier

In [185…
```
from sklearn.feature_extraction.text import CountVectorizer

vec = CountVectorizer(ngram_range = (1,1), min_df = 1)

X_train = vec.fit_transform(X_text_train)
X_test = vec.transform(X_text_test)

svc = LinearSVC()
params = {"C": [0.1, 1.]}

clf = GridSearchCV(svc, params, cv = 2)
clf.fit(X_train, y_train)
```

Out[185…
```
GridSearchCV(cv=2, estimator=LinearSVC(), param_grid={'C': [0.1, 1.0]})
```

Report (print) the accuracy, micro F1, and macro F1 of the final classifier on the test data and train data

In [187…
```
preds = clf.predict(X_test)
print(f"Accuracy Score: {accuracy_score(y_test, preds)}")

F1_Micro = f1_score(y_test, preds, average = 'micro')
print(f"F1 Micro Score: {F1_Micro}")

F1_Macro = f1_score(y_test, preds, average = 'macro')
print(f"F1 Micro Score: {F1_Macro}")
```

```
Accuracy Score: 0.465958775765146 77
F1 Micro Score: 0.46595877576514677
F1 Micro Score: 0.38576934100208116
```

Create a bag of words feature representation for the tweets using the CountVectorizer

In [201…
```
cnt_stats = vec.fit(X_text)
bagOfWords = vec.transform(X_text)

print(f"Vocabulary Content (First 1000): \n\n {cnt_stats.vocabulary_}"[0:1000])
```

```
Vocabulary Content (First 1000):

 {'gas': 7462, 'by': 3388, 'my': 12453, 'house': 8685, 'hit': 8497, '39': 441, 'going': 7739, 'to': 18593, 'cha
pel': 3821, 'hill': 8474, 'on': 13306, 'sat': 16151, 'iranian': 9281, 'general': 7517, 'says': 16187, 'israel':
9335, 'iron': 9294, 'dome': 5669, 'can': 3499, 'deal': 5134, 'with': 20214, 'their': 18304, 'missiles': 12056,
'keep': 10082, 'talking': 17970, 'like': 10836, 'that': 18241, 'and': 1580, 'we': 19903, 'may': 11650, 'end': 6
248, 'up': 19310, 'finding': 6885, 'out': 13468, 'davlar': 5091, '11th': 89, 'main': 11357, 'rivals': 15642, 'a
re': 1783, 'team': 18076, 'poland': 14244, 'hopefully': 8627, 'an': 1564, 'make': 11367, 'it': 9350, 'successfu
l': 17658, 'tough': 18736, 'week': 19940, 'of': 13160, 'training': 18800, 'tomorrow': 18653, 'about': 1034, 'ac
```

```
t': 1107, 'amp': 1549, 'deciding': 5173, 'where': 20035, 'want': 19798, 'go': 7703, 'college': 4283, 'applyin
g': 1738, 'colleges': 4287, 'everything': 6460, 'stresses': 17570, 'me': 11726
```

How many features were created with the bag of words representation?

In [209…

```
print(f"Feature Count: {len(vec.get_feature_names())}")
print(f"Bag of Words Shape: {bagOfWords.shape}")
```

```
Feature Count: 20898
Bag of Words Shape: (8002, 20898)
```

In [ ]: