BBC_Classification

November 20, 2022

1 CSCC11 - Introduction to Machine Learning, Fall 2022, Assignment 2

1.1 Authors

Shawn Santhoshgeorge (1006094673)
Anaqi Amir Razif (1005813880)

1.2 Part 2: Programming Component

1.2.1 Setup

[1]:

```
from helper import *
from NB import NB
from GCC import GCC
from KNN import KNN

OUTPUT = 'latex' # 'html' or 'latex' for create_report

binary_data = load_data(mode=Mode.BINARY, train_size=0.7, test_size=0.3)
freq_data = load_data(mode=Mode.FREQ, train_size=0.7, test_size=0.3)
```

1.2.2 Naive Bayes

[2]:

```
X_train, y_train, X_test, y_test, labels = binary_data

n_b = NB()
n_b.train(X_train, y_train, labels)

# Using Training Dataset
y_pred_train = n_b.predict(X_train)
label_acc_train, overall_acc_train = get_accuracy(labels, y_train, y_pred_train)

# Using Testing Dataset
y_pred_test = n_b.predict(X_test)
label_acc_test, overall_acc_test = get_accuracy(labels, y_test, y_pred_test)

create_report(label_acc_train, label_acc_test, overall_acc_train, overall_acc_test, output=OUTPUT)
```

Training Accuracy

Label	Accuracy
business	98.86 %
entertainment	96.24 %
politics	95.18 %
sport	99.17 %
tech	92.88 %

Testing Accuracy

Label	Accuracy
business	97.47 %
entertainment	95.0 %
politics	93.4 %
sport	98.67 %
tech	82.84 %

Training Overall Accuracy: 96.72 % Testing Overall Accuracy: 93.71 %

1.2.3 Gaussian Class Conditionals

[3]:

```python
X_train, y_train, X_test, y_test, labels = freq_data

g_cc = GCC()
g_cc.train(X_train, y_train, labels)

# Using Training Dataset
y_pred_train = g_cc.predict(X_train)
label_acc_train, overall_acc_train = get_accuracy(labels, y_train, y_pred_train)

# Using Testing Dataset
y_pred_test = g_cc.predict(X_test)
label_acc_test, overall_acc_test = get_accuracy(labels, y_test, y_pred_test)

create_report(label_acc_train, label_acc_test, overall_acc_train, overall_acc_test, output=OUTPUT)
```

Training Accuracy		Testing Accuracy	
Label	Accuracy	Label	Accuracy
business	100.0 %	business	93.67 %
entertainment	100.0 %	entertainment	90.83 %
politics	100.0 %	politics	87.74 %
sport	100.0 %	sport	95.33 %
tech	100.0 %	tech	90.3 %

Training Overall Accuracy: 100.0 % Testing Overall Accuracy: 91.62 %

1.2.4 K-Nearest Neighbours

[4]:

```python
for mode in Mode:
    if OUTPUT == 'html':
        display_html(f'<center> <h3> With {mode.value} Data </h3> </center>', raw=True)
    elif OUTPUT == 'latex':
        display(Math(r'\text{With }' + r'\text{' f'{mode.value}' + r'}' + r'\text{ Data}'))
    X_train, y_train, X_test, y_test, labels = freq_data if mode.value == Mode.FREQ.value else binary_data
    for k in [1, 3, 6]:
        if OUTPUT == 'html':
            display_html(f'<h4> For K = {k} </h4>', raw=True)
        elif OUTPUT == 'latex':
            display(Math(r'\text{For K = }' + f'{k}'))
        k_nn = KNN(k)
        k_nn.train(X_train, y_train)

        y_pred_train = k_nn.predict(X_train)
        label_acc_train, overall_acc_train = get_accuracy(labels, y_train, y_pred_train)

        y_pred_test = k_nn.predict(X_test)
        label_acc_test, overall_acc_test = get_accuracy(labels, y_test, y_pred_test)

        create_report(label_acc_train, label_acc_test, overall_acc_train, overall_acc_test, output=OUTPUT)
```

With Frequency Data

For K = 1

Training Accuracy		Testing Accuracy	
Label	Accuracy	Label	Accuracy
business	100.0 %	business	79.75 %
entertainment	100.0 %	entertainment	60.0 %
politics	100.0 %	politics	73.58 %
sport	100.0 %	sport	100.0 %
tech	100.0 %	tech	61.94 %

Training Overall Accuracy: 100.0 % Testing Overall Accuracy: 76.2 %

For K = 3

Training Accuracy		Testing Accuracy	
Label	Accuracy	Label	Accuracy
business	82.1 %	business	68.99 %
entertainment	71.43 %	entertainment	50.0 %
politics	71.06 %	politics	62.26 %
sport	99.72 %	sport	100.0 %
tech	74.53 %	tech	46.27 %

Training Overall Accuracy: 80.86 % Testing Overall Accuracy: 66.92 %

For K = 6

Training Accuracy

Label	Accuracy
business	68.75 %
entertainment	50.75 %
politics	56.91 %
sport	100.0 %
tech	43.45 %

Testing Accuracy

Label	Accuracy
business	56.96 %
entertainment	45.0 %
politics	51.89 %
sport	100.0 %
tech	35.07 %

Training Overall Accuracy: 66.22 % Testing Overall Accuracy: 59.28 %

With Binary Data

For K = 1

Training Accuracy

Label	Accuracy
business	100.0 %
entertainment	100.0 %
politics	100.0 %
sport	100.0 %
tech	100.0 %

Testing Accuracy

Label	Accuracy
business	41.14 %
entertainment	28.33 %
politics	42.45 %
sport	99.33 %
tech	30.6 %

Training Overall Accuracy: 100.0 % Testing Overall Accuracy: 50.0 %

For K = 3

Training Accuracy

Label	Accuracy
business	61.36 %
entertainment	43.61 %
politics	52.41 %
sport	100.0 %
tech	43.07 %

Testing Accuracy

Label	Accuracy
business	34.81 %
entertainment	14.17 %
politics	38.68 %
sport	100.0 %
tech	11.19 %

Training Overall Accuracy: 62.36 % Testing Overall Accuracy: 41.62 %

For K = 6

Training Accuracy

Label	Accuracy
business	45.17 %
entertainment	24.44 %
politics	32.8 %
sport	100.0 %
tech	11.61 %

Testing Accuracy

Label	Accuracy
business	25.32 %
entertainment	13.33 %
politics	25.47 %
sport	100.0 %
tech	5.22 %

Training Overall Accuracy: 46.11 % Testing Overall Accuracy: 35.93 %

1.2.5 Conclusion

Model	Training	Testing
Naive Bayes	96.72 %	93.71 %
Gaussian Class Conditionals	100.00 %	91.92 %
1-NN (Frequency Data)	100.00 %	76.2 %
3-NN (Frequency Data)	80.86 %	66.92 %
6-NN (Frequency Data)	66.22 %	59.28 %
1-NN (Binary Data)	100.00 %	50.00 %
3-NN (Binary Data)	62.36 %	41.62 %
6-NN (Binary Data)	46.11 %	35.93 %

Overall Accuracy Sorted By Testing Data

From the above table we can see that the Naive Bayes performed quite well compared to all other models here and would recommend using Naive Bayes for predicting the article labels. Some other things to note is how the K-NN models have quite varying performance based on if the data provided is Frequency or Binary and seems to work well with Frequency Data. Between all the models we can see that was able to accurately provide the label for 'sports' no matter if the data provided is Frequency or Binary, this could be becuase there are more of those articles in the dataset.