CENG463 ASSIGNMENT 1 REPORT

Members: Ceren Çağlayan & Elif Özyürek

ID: 270201059 & 280201079

Department: Computer Engineering

Lecture: Introduction to Machine Learning

Lecturer: Dr. Emrah İnan

TABLE OF CONTENTS

1. Creating the Dataset

The first step to create a dataset was to retrieve a certain amount of text data from academic papers on "brain injury", "type 2 diabetics" and "kidney failure" topics. In this assignment, the *Abstract* section was retrieved as most websites require a paid subscription to access the *Conclusion* section of academic papers.

The number of elements of the dataset was manipulated to inspect its effect on accuracy. Since a smaller number of data caused overfitting, we ended up with a model that pulls around 1400-1500 pieces of data for the dataset. (*see Figure.1*)

The function "*collect_data(keywords)*" is where the dataset is created using the "*paperscraper*" module and "*pandas*" library. *(see Figure.2)*. With the function *get_pubmed_papers(keyword, max_results=500),* the papers with the titles specified in the *keywords* list are extracted. To specify the number of papers to collect for each keyword, the *max_results* parameter is used. The function converts the collected data to a data frame, and the data is appended to the *dataset.csv* file.

2. Classification Techniques

To test the dataset, four classification techniques from the Sklearn library in Python were used:

2.1- K-Nearest Neighbors

2.2- Naive Bayes (Multinomial NB)

2.3- Random Forest

2.4- Support Vector Machine

Since the dataset consists of text data, *vectorizer* is used to convert text data into numerical vectors. The vectorizing process is shown in the code below.

```
      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=45)

      print("-----vectorizing process for text data-----")

      time.sleep(2)

      vectorizer = TfidfVectorizer(max_features=5000)

      X_train_vec = vectorizer.fit_transform(X_train)

      X_test_vec = vectorizer.transform(X_test)
```

2.1 K-Nearest Neighbors

K-Nearest Neighbors (KNN) is a machine learning algorithm that uses the value of a sample's k-nearest neighbors to determine classification or regression results. The n parameter determines the number of nearest neighbors to be considered when making predictions.

Results when the number of elements in the dataset = 1430:

```
----------K-Nearest Neighbors----------
------------------n=5------------------
K-Neighbor Model Results:
Train score: 0.3207
Test score: 0.3753
Accuracy: 0.3753
```

Comment: Default number of nearest neighbors in KNN model is 5. The model achieved low success in the training set, the performance on test data is also very low. Low scores in both the training and the test sets mean that there is *underfitting*.

2.2 Naive Bayes (Multinomial NB)

Naive Bayes is a machine learning algorithm used in classification problems. It has a probability-based structure and uses Bayes theorem. The alpha (a) parameter is used for Laplace smoothing, and it can be modified to prevent overfitting.

Results when the number of elements in the dataset = 1430:

```
---------------Naive Bayes---------------
-------------------a=1-------------------
Naive Bayes Model Results:
Train score: 0.9305
Test score: 0.8333
Accuracy: 0.8333
```

Comment: The model achieved very high success on the training data. The performance on the test data is close to the training score and is quite high. The algorithm *worked very well with the default alpha* (a) value 1.

2.3 Random Forest

Random Forest is an algorithm used in classification and regression problems by combining many decision trees and making a common prediction. Each decision tree is trained with random samples and random feature selections. The parameter *n* denotes the number of trees in the forest. It is less prone to overfitting compared to decision trees.

Results when the number of elements in the dataset = 1430:

```
-------------Random Forest-------------
------------------n=100-----------------
Random Forest Model Results:
Train score: 0.973
Test score: 0.8834
Accuracy: 0.8834
```

Comment: The *default value* of n, the number of trees in the forest, is 100. The model has a high score on the training data but there is a slight decrease in performance on test data. Since the gap between train and test scores is quite small, the model *can be considered successful*.

2.4 Support Vector Machine

Support Vector Machine (SVM) aims to find the best "hyperplane" that separates data points. The parameter *C* determines the *trade-off* between maximizing the space between the decision boundary and the closest points and allowing misclassifications. The performance of the algorithm also depends on the choice of kernel.

Results when the number of elements in the dataset = 1430:

```
---------Support Vector Machine--------
------------------C=1-------------------
Support Vector Machine Model Results:
Train score: 0.9711
Test score: 0.8372
Accuracy: 0.8372
```

Comment: The *default kernel choice* is “rbf” and the *default value for C* is 1. The algorithm has high scores in both the training and the test sets. Since the scores are high and the gap between the scores is low, it can be said that the algorithm worked *quite successfully*.

3. Regularization and Experimenting Parameters

Even though some algorithms performed successfully with the default parameter values, better accuracies can be obtained by changing these values. In the following sections, the parameters of each algorithm are changed and the results are observed.

3.1 K-Nearest Neighbors

In the K-Nearest Neighbors algorithm, changing the number of nearest neighbors affects the accuracy of the algorithm. In other words, better accuracy results can be found by changing the value of n.

If the selected n value is too small, overfitting can occur. (*see 2.1-K-Nearest Neighbors*) On the other hand, the generalization ability may be reduced if the value for n is too large. Therefore, it is important to choose an appropriate K value depending on the problem and the dataset.

The results for n values 5, 10, and 30 are shown below. When n=30, overfitting was eliminated and a good result was obtained.

When n = 1:

```
--------------Test 1 -> n=1--------------
K-Nearest Neighbors Model Results:
Train score: 0.971
Test score: 0.3566
Accuracy: 0.3566
```

Comment: The model achieved high success in training and fit the training data very well. However, the performance on test data is quite poor. The high score in the training set combined with a very low score on the test set means that there is *overfitting*. The model is unable to make accurate predictions when it is given new inputs.

When n = 10:

```
--------------Test 2 -> n=10-------------
K-Nearest Neighbors Model Results:
Train score: 0.6593
Test score: 0.345
Accuracy: 0.345
```

Comment: With a higher n value, a visible decrease in the train score was recorded, showing the inability to learn from the training set. The test scores were also very low, meaning that the model was not able to make accurate predictions. As can be seen from the two values, *an accurate model cannot be obtained* with this parameter value.

When n = 30:

```
--------------Test 3 -> n=30-------------
K-Nearest Neighbors Model Results:
Train score: 0.8142
Test score: 0.7576
Accuracy: 0.7576
```

Comment: With a higher number of neighbors, high train and test scores were observed. The *increase in accuracy* is an indication that using more neighbors increases the generalization ability of the model.

3.2 Naive Bayes (Multinomial NB)

The alpha parameter is used in regularization for the Naive Bayes algorithm. Smaller values for alpha mean less regularization and more complex models, while larger values mean simpler models.

For the default value of alpha, the model works quite successfully and the accuracy is very high.

When alpha = 0.1:

```
--------------Test 1 -> a=0.1------------
Naive Bayes Model Results:
Train score: 0.9454
Test score: 0.8125
Accuracy: 0.8125
```

Comment: As seen in the figure above, for a lower value of alpha, there is a slight decrease in the accuracy. It can be said that the generalization ability of the model is *not as good as when the default alpha value was used*.

When alpha = 30:

```
--------------Test 2 -> a=30-------------
Naive Bayes Model Results:
Train score: 0.9076
Test score: 0.8472
Accuracy: 0.8472
```

Comment: With an increased alpha value, the model has a lower train score due to the regularization reducing the focus on train data. Since regularization makes the model more resistant to overfitting, there is an *increase in the accuracy score*.

3.3 Random Forest

The *n* (*n_estimators* in Python) parameter in the Random Forest algorithm determines the number of trees and has an impact on accuracy. A higher number means better generalization, but there is a risk of overfitting when the value becomes too high.

The accuracy of the model was very high when the default value n=100 was used, but slightly better accuracy can be achieved by choosing a different number of trees.

When n = 20:

```
--------------Test 1 -> n=20-------------
Random Forest Model Results:
Train score: 0.973
Test score: 0.8438
Accuracy: 0.8438
```

Comment: For a smaller number of trees, in other words, a smaller value for the n parameter, a *slight decrease in the accuracy* can be observed.

When n = 150:

```
--------------Test 2 -> n=150------------
Random Forest Model Results:
Train score: 0.973
Test score: 0.8904
Accuracy: 0.8904
```

Comment: When the value of n is higher, meaning a higher number of trees were used in the forest, a *slightly higher accuracy* can be observed.

3.4 Support Vector Machines

*C i*s the regularization parameter applied by Support Vector Machines. While the lower C value provides a smoother model, the larger C value results in a model that is more complex.

The choice of *kernel type* also affects the accuracy value obtained with the Support Vector Classifier (SVC). Each kernel type may give different results depending on the requirements of the model. The effects of the following three types of kernels can be seen in this section: radial basis function (rbf), linear, and polynomial (poly).

The default kernel choice in SVC is the radial basis function (rbf) and the effects of changing the C value were observed using this kernel type.

For the observation of the effects of different kernel types, the C value is kept unchanged.

When C = 0.1 and kernel = "rbf":

```
--------------Test 1 -> C=0.1------------
Support Vector Machine Model Results:
Train score: 0.3467
Test score: 0.289
Accuracy: 0.289
```

Comment: With a decreasing C value, it seems that the model does not fit the training data well and fails to generalize to the test data, which results in *underfitting.*

When C = 20 and kernel = "rbf":

```
--------------Test 2 -> C=20-------------
Support Vector Machine Model Results:
Train score: 0.973
Test score: 0.8462
Accuracy: 0.8462
```

Comment: With an increased C value, the train and test scores have significantly higher values, and *the accuracy of the model becomes very high*. This proves that

increasing the regularization parameter improves both model fit and generalization, and the *tendency to overfit is lower*.

When C = 20 and kernel = "poly":

```
--------SVM (kernel changes, C=20)-------

--------------kernel -> poly-------------
Support Vector Machine Model Results:
Train score: 0.973
Test score: 0.6364
Accuracy: 0.6364
```

Comment: When the kernel type is switched to "polynomial", the *performance of the model decreases* in terms of test and accuracy scores. This indicates that using polynomial kernels does not effectively capture underlying patterns.

When C = 20 and kernel = "linear":

```
-------------kernel -> linear------------
Support Vector Machine Model Results:
Train score: 0.973
Test score: 0.8252
Accuracy: 0.8252
```

Comment: The accuracy result is *higher than* when the *"polynomial" kernel* is used. However, it is *slightly less than* the result when the *default kernel type "rbf"* was used. It can still be said that "linear" kernels can be used for linearly separable data.

Overall for SVM:

Radial basis function (rbf) is the most commonly used kernel type in SVM and fits most models. As can be seen from the previous experiments, the *highest accuracy was obtained with the "rbf" kernel and a higher value of C.*

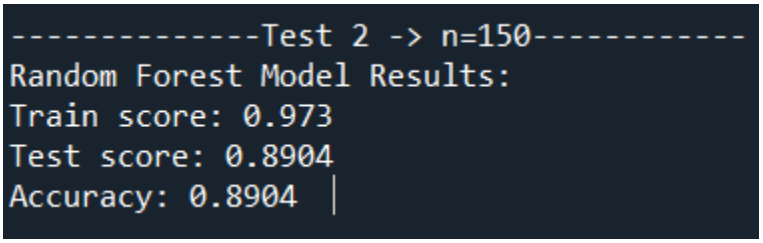
```
--------------Test 2 -> C=20-------------
Support Vector Machine Model Results:
Train score: 0.973
Test score: 0.8462
Accuracy: 0.8462
```

4. Conclusion

In this assignment, a dataset was created with the abstract sections of academic papers on the topics "brain injury", "type 2 diabetes", and "kidney failure". The following four classification techniques were selected: K-Nearest Neighbors, Naive Bayes (Multinomial NB), Random Forest, and Support Vector Machine (SVM).

The regularization parameters and other variables for each classification technique were modified to achieve the optimal complexity and the highest accuracy score. It was clear that understanding the characteristics of different algorithms and their sensitivity to parameters was a crucial part of developing effective machine-learning models.

Upon conducting several experiments, we observed that the *Random Forest* model with *150 trees (n_estimators = 150)* produced the *highest accuracy*. The accuracy scores for this model were high, and the overfitting issue did not occur. This is due to the fact that the Random Forest algorithm is *resistant to overfitting*. The train, test, and accuracy scores for n = 150 can be seen in the figure below.



APPENDIX

```
In [34]: dataset
Out[34]:
                                                  Title  ...           Label
0     Resveratrol Reduces Neuroinflammation and Hipp...  ...    brain injury
1     Pharmacokinetics of 4-Hydroxybenzaldehyde in N...  ...    brain injury
2     Dorsal striatal functional connectivity and re...  ...    brain injury
3     Pyroptosis in septic lung injury: Interactions...  ...    brain injury
4     Traumatic brain injury, abnormal growth hormon...  ...    brain injury
...                                                 ...  ...             ...
1425  Acidosis induces significant changes to the mu...  ...  kidney disease
1426  A Comprehensive Review on Molecular Mechanism ...  ...  kidney disease
1427  Spray-dried Solid Lipid Nanoparticles for Enha...  ...  kidney disease
1428  LiverTox: Clinical and Research Information on...  ...  kidney disease
1429                                          StatPearls  ...  kidney disease

[1430 rows x 4 columns]
```

Figure.1: *Dataset that has 1430 elements with 4 features*

```
In [33]: dataset.keys()
Out[33]: Index(['Title', 'Authors', 'Abstract', 'Label'], dtype='object')
```

Figure.2: *General structure of dataset*