Laboratory Tutorial 13: Supervised Learning and Logistic Regression

In this laboratory tutorial you will work with Supervised Machine Learning, specifically with Logistic Regression model for classification problems.

Preamble

In this lab worksheet, you will gain practice on one of the groups of Supervised Machine Learning - Classification. In particular, you will learn how to develop a classification algorithm called Logistic Regression Model for classification problems. You will first use some historical data, split it into training and testing datasets before running the classifier, and finally evaluate the classifier using confusion matrix.

You will need Jupyter notebook for Python 3.

Exercise 13.1: Classification

Suppose you are asked to build an email spam classifier. What could be the training data, features and the output label? How would you evaluate the classifier?

Exercise 13.2: Confusion Matrix

Each of the following confusion matrices relate to a classification model.

N=43	Predicted Class (0)	Predicted Class (1)
Actual Class (0)	3	20
Actual Class (1)	2	18

Confusion Matrix 1

N=56	Predicted Class (0)	Predicted Class (1)
Actual Class (0)	28	2
Actual Class (1)	1	25

Confusion Matrix 2

N=47	Predicted Class (0)	Predicted Class (1)
Actual Class (0)	3	18
Actual Class (1)	24	2

Confusion Matrix 3

(a) For each of the above confusion matrix calculate the Positive Predicted Value.

(b) F1 Score

F1 Score is the weighted average of Precision and Recall. This score takes both false positives and false negatives into account. It performs well on imbalance dataset. To calculate F1 score we need the following formula:

F1 Score = (2*Precision*Recall)/(Precision + Recall)

Calculate the F1 score for Confusion Matrix 2.

(c) For each of the confusion matrix calculate the accuracy and error to two decimal places.

(d) From (c), based on your results, which classification model has the:
   (i) Highest error rate?
   (ii) Lowest error rate?

Exercise 13.3: Running Logistic Regression model in Python

Using scikit-learn library we will build a logistic regression model in python. We will first generate and split a dataset into training dataset and test dataset. We will then train the logistic regression model (as a classifier), make some predictions, and finally perform the evaluation. **Note:** before you actually complete this exercise make sure you have clearly understood training and testing dataset from the lecture notes. If you are struggling please ask me or the GTAs for help.

1. Open Jupyter
2. Import the following library:

```
from sklearn.datasets import make_classification
# helps to generate a dataset that can be used for logistic regression
```

3. Next, we will generate a dataset

```
X,Y=make_classification(n_samples=20,n_features=1,n_classes=2,n_clusters_per_class=1,
flip_y=0.03,n_informative=1,n_redundant=0,n_repeated=0)

#X is the feature, and Y is the categorical output, with class: 0 and 1

#Basically, we have 20 random values and each value is  categorised as  0 or 1

#If you are struggling to understand feature and output then check your lecture slides.
```

4. Let us see what we have for X and Y

```
print(X)

print(Y)


#We can see that whenever a value for x is negative the class (in terms of y)  is 0, and
#whenever it is positive the class is 1. Note: If the value of x is 0, we will simply treat it as
positive
```

5. We can also visualise our data

```
#import matplotlib to visualise our data

from matplotlib import pyplot as plt

plt.scatter(X,Y,cmap = 'Blue')

plt.show()
```

6. Data split

```
#Split the data into training dataset (80%), and test dataset (20%).

#In terms of the 20% test dataset we will test whether the logistic regression model which
we are going to build is predicting well or not

#We need to import train_test_split

# If you are struggling to understand training and test dataset then check your lecture slides.

from sklearn.model_selection import train_test_split

X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.20)
```

7. X_train and Y_train

```
X_train

#There should be 16 values (relates to feature). This is 80% of the data that  is trained
```

Y_train

#There should be 16 values (relates to output: as 0 or 1), this is 80% of the data that is trained.

#We can observe that in the X_train when a value for x is negative, if you mapped it with the value of Y_train it is 0. When a value for x is positive or 0, then it is 1

8. X_test (as feature) and Y_test (as actual output)

```
X_test
#There should be 4 values (20% of the testing data)
Y_test
#There should be 4 values (20% of the testing data)


#Similar to 7) Map X_test with Y _test
```

We will now check whether the logistic regression model that we are going to build correctly predict the output (as 0 or 1) or not. In other words, is the predicting output same as actual output (as in step 8). To do this we need both the training and testing data.

9. Performing Logistic Regression

```
from sklearn.linear_model import LogisticRegression
logistic_regression_model=LogisticRegression() #we create a Logistic Regression object


#Train the model(80% of the data)
logistic_regression_model.fit(X_train,Y_train) # fit the object to our data.
```

10. Test

```
prediction=logistic_regression_model.predict(X_test)
print(prediction) # predicted output
```

11. Compare with Y_test (actual output)

```
Y_test
```

#Check if there is any difference between the actual and predicted output. If not, this means the logistic regression model is predicting well. In reality, the regression model will still make some errors (which we will find out in step 13))

12. Similar to Linear Regression (which you covered in Data1) we can calculate the coefficient and intercept

```
coeff=logistic_regression_model.coef_
print(coeff)
intercept=logistic_regression_model.intercept_
print(intercept)
```

13. Evaluate the Logistic Regression model using a confusion matrix

```
from sklearn.metrics import confusion_matrix
confusion_matrix(Y_test,prediction)
#Output
array([[2,0],
       [0,2]])
#Where TN=2, FP=0, FN=0, and TP=2

#Accuracy manually
#Accuracy = (TP+TN)/(TN+FP+FN+TP) = (2+2)/(2+0+0+2) = 1

#Accuracy using scikit learn
from sklearn.metrics import accuracy_score
model_accuracy=accuracy_score(Y_test,prediction)
model_accuracy

#Output = 1
```

14. In step 13 since the accuracy value is close or equal to 1, this means the Logistic Regression model is predicting very well and should not be discarded.

15. Since the Logistic Regression model is predicting well, we can now predict a new instance of data. Let's find the value of y with x = -2.069

```
logistic_regression_model.predict([[-2.069]])


#Output (i.e., value of y) should be 0
```

Summary

In this tutorial you have learned how to use Logistic Regression model for classification problems. Specifically, using scikit-learn library in Python you have learned how to build Logistic Regression model using training and testing dataset, and also how to evaluate the classifier.

Reading

Textbook: Elements of Statistical Learning

- https://web.stanford.edu/~hastie/Papers/ESLII.pdf
- Chapter: 13