IT-641 Deep Learning

Lab 2

1.Introduction

Classification and Regression

In this lab tutorial we shall utilize the machine learning pipeline to develop machine learning models on classification and regression tasks. In order to determine which model works best upon our data we need to define certain evaluation metrics for our machine learning models. We shall implement classification and regression models and evaluate them in this lab

2.Model Evaluation Metrics

Evaluation metrics for Regression

1. Mean Squared Error (MSE):

   It represents the squared distance between actual and predicted values. We perform squared to avoid the cancellation of negative terms and it is the benefit of MSE.

   MSE = $\frac{1}{n}(\sum(y - ypred)^2$

   sklearn implementation:

```
from sklearn.metrics import mean_squared_error
#checking the insample error
y_pred = model.predict(x_train)

error = mean_squared_error(y_train,y_pred)
```

2.R2 Score: In regression, the R2 coefficient of determination is a statistical measure of how well the regression predictions approximate the real data points. An R2 of 1 indicates that the regression predictions perfectly fit the data

sklearn implementation:

```
from sklearn.metrics import r2_score
#checking the insample error
y_pred = model.predict(x_train)

r2 = r2_score(y_train,y_pred)
```

3. Mean Absolute Error(MAE): MAE is a very simple metric which calculates the absolute difference between actual and predicted values.

$$\text{MAE} = \frac{1}{n}(|y - ypred|)$$

```
from sklearn.metrics import mean_absolute_error
#checking the insample error
y_pred - model.predict(x_train)

mae = mean_absolute_error(y_train,y_pred)
```

<u>Evaluation metrics for classification</u>

1.Confusion Matrix:

A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the total number of target classes. The matrix compares the actual target values with those predicted by the machine learning model

```
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sb

y_pred = model.predict(x_train)
plt.figure(figsize = (20,10))
sb.heatmap(confusion_matrix(y_train,y_pred))
plt.title("Confusion Matrix")
plt.show()
```

2. Precision Score:
Precision for a class is the number of true positives (i.e. the number of items correctly labelled as belonging to the positive class) divided by the total number of elements labelled as belonging to the positive class

```
from sklearn.metrics import precision_score
y_pred = model.predict(x_train)
precision = precision_score(y_train,y_pred)
```

3. Recall score:
We define recall as the number of true positives divided by the number of true positives plus the number of false negatives.

```
from sklearn.metrics import recall_score
y_pred = model.predict(x_train)
recall = recall_score(y_train,y_pred)
```

4. F1 Score:
Harmonic Mean of precision and recall. A low F1 score is an indication of both poor precision and poor recall.

```
from sklearn.metrics import f1_score
y_pred = model.predict(x_train)
f1 = f1_score(y_train,y_pred)
```

5. Accuracy Score:

```
from sklearn.metrics import accuracy_score
y_pred = model.predict(x_train)
acc = accuracy_score(y_train,y_pred)
```

3. Dataset:

1. **Hearts dataset:** predict heart disease using features like age, sex, chestpressure, cholesterol levels, restecg etc.
2. **House Price Prediction:** Predict house prices using advanced regression techniques

4. Tasks:

For the given datasets perform the following tasks:

1. Load Data and Find out if it has any missing values
2. Correct the missing values
3. Identify and encode necessary features
4. Identify and normalize necessary features
5. Split the dataset into train set (75%) test set (15%) an validation set (10%)
6. For Regression Task: Write a code manually for Linear Regression and compare the results with sklearns linear regression model.
7. For Classification Task: Write manual code for logistic regression using Gradient Descent and compare it with sklearns Logistic Regression
8. Use all the respective model performance criteria and compare to model
9. Discuss under-fitting and overfitting based upon results

5. Submission Details

1. The assignment can be submitted as IPython Notebook as well as PDF

2. The submitted file must be of format:

   "STUDENTID_FIRSTNAME_LASTNAME.ipynb" or
   "STUDENTID_FIRSTNAME_LASTNAME.pdf"