GENERAL SIR JOHN KOTELAWALA DEFENCE UNIVERSITY

DEPARTMENT OF COMPUTER ENGINEERING

Machine Learning

Assignment 1

Data Analysis Report

Name: W.M.K Walisundara

Registration Number: D/BCE/21/0016

Field: Computer Engineering

Introduction

Brief Description of the Dataset

Dataset contains 210 soft X-ray images of wheat kernels (varieties: Kama, Rosa, Canadian) from experimental fields in Lublin, Poland.

Dataset can be found at: Seeds - UCI Machine Learning Repository

To construct the data, seven geometric parameters of wheat kernels were measured in this dataset:

1. Area (A)

2. Perimeter (P)

3. Compactness ($C = \frac{4\pi A}{P^2}$)

4. Length of kernel

5. Width of kernel

6. Asymmetry coefficient

7. Length of kernel groove

Exploratory Data Analysis

Data Understanding

- Reporting the data types, number of instances, data quality issues and any missing values.
- Features and target variable(s) in the dataset

Data Preprocessing

- Checking for the duplicates, missing values calculation, scaling, encoding categorical variables
- Data Standardization: This process centers the feature around zero and scales it to have a unit variance.
- Data Normalization: Data normalization is a similar technique to standardization, but instead of using the mean and standard deviation, it scales the features to a specific range, usually [0, 1] or [-1, 1].

Data Visualization

- Visualize the target variable(s) and their relationship with the features.

Preprocessing and Data Engineering

Initial overview of the dataset

```
RangeIndex: 210 entries, 0 to 209
Data columns (total 8 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   area                    210 non-null    float64
 1   perimeter               210 non-null    float64
 2   compactness             210 non-null    float64
 3   length of kernel        210 non-null    float64
 4   width of kernel         210 non-null    float64
 5   asymmetry coefficient   210 non-null    float64
 6   length of kernel groove  210 non-null    float64
 7   type                    210 non-null    int64  
dtypes: float64(7), int64(1)
memory usage: 13.3 KB
None
```

Handling Missing values

- There are no missing values in the dataset.

```
area                    0
perimeter               0
compactness             0
length of kernel        0
width of kernel         0
asymmetry coefficient   0
length of kernel groove  0
type                    0
dtype: int64
```

Handling duplicate values

- There are no duplicate values in the dataset.

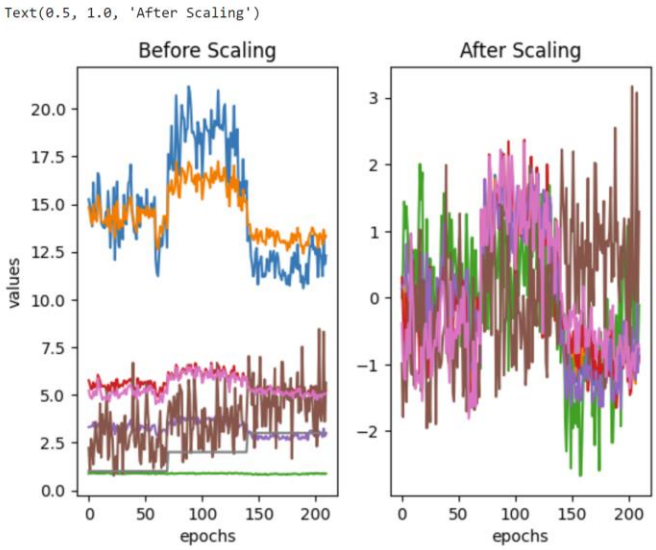
```
Empty DataFrame
Columns: [area , perimeter , compactness, length of kernel, width of kernel, asymmetry coefficient, length of kernel groove, type]
Index: []
```

Descriptive Statistics

	area	perimeter	compactness	length of kernel	width of kernel	asymmetry coefficient	length of kernel groove	type
count	210.000000	210.000000	210.000000	210.000000	210.000000	210.000000	210.000000	210.000000
mean	14.847524	14.559286	0.870999	5.628533	3.258605	3.700201	5.408071	2.000000
std	2.909699	1.305959	0.023629	0.443063	0.377714	1.503557	0.491480	0.818448
min	10.590000	12.410000	0.808100	4.899000	2.630000	0.765100	4.519000	1.000000
25%	12.270000	13.450000	0.856900	5.262250	2.944000	2.561500	5.045000	1.000000
50%	14.355000	14.320000	0.873450	5.523500	3.237000	3.599000	5.223000	2.000000
75%	17.305000	15.715000	0.887775	5.979750	3.561750	4.768750	5.877000	3.000000
max	21.180000	17.250000	0.918300	6.675000	4.033000	8.456000	6.550000	3.000000

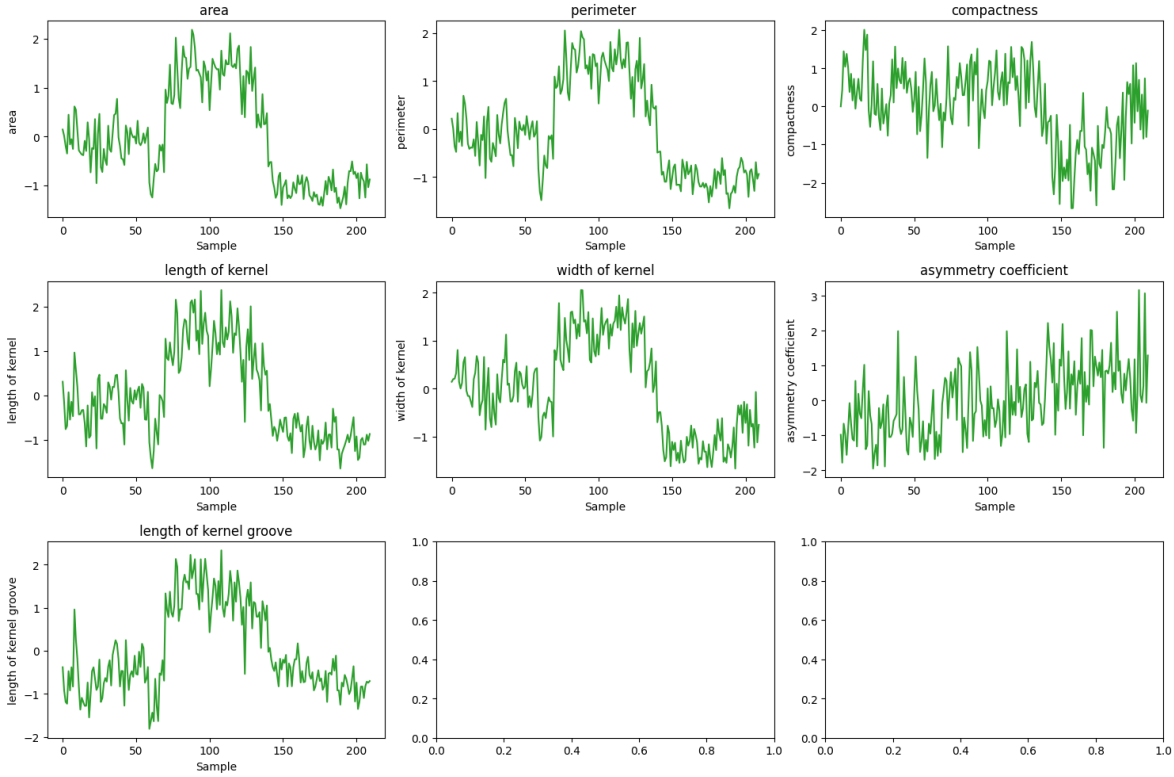
Data Standardization/Normalization

This uses to create a level playing field for all features. For example, In the dataset, some might have higher values compared to others. To avoid that this technique used.



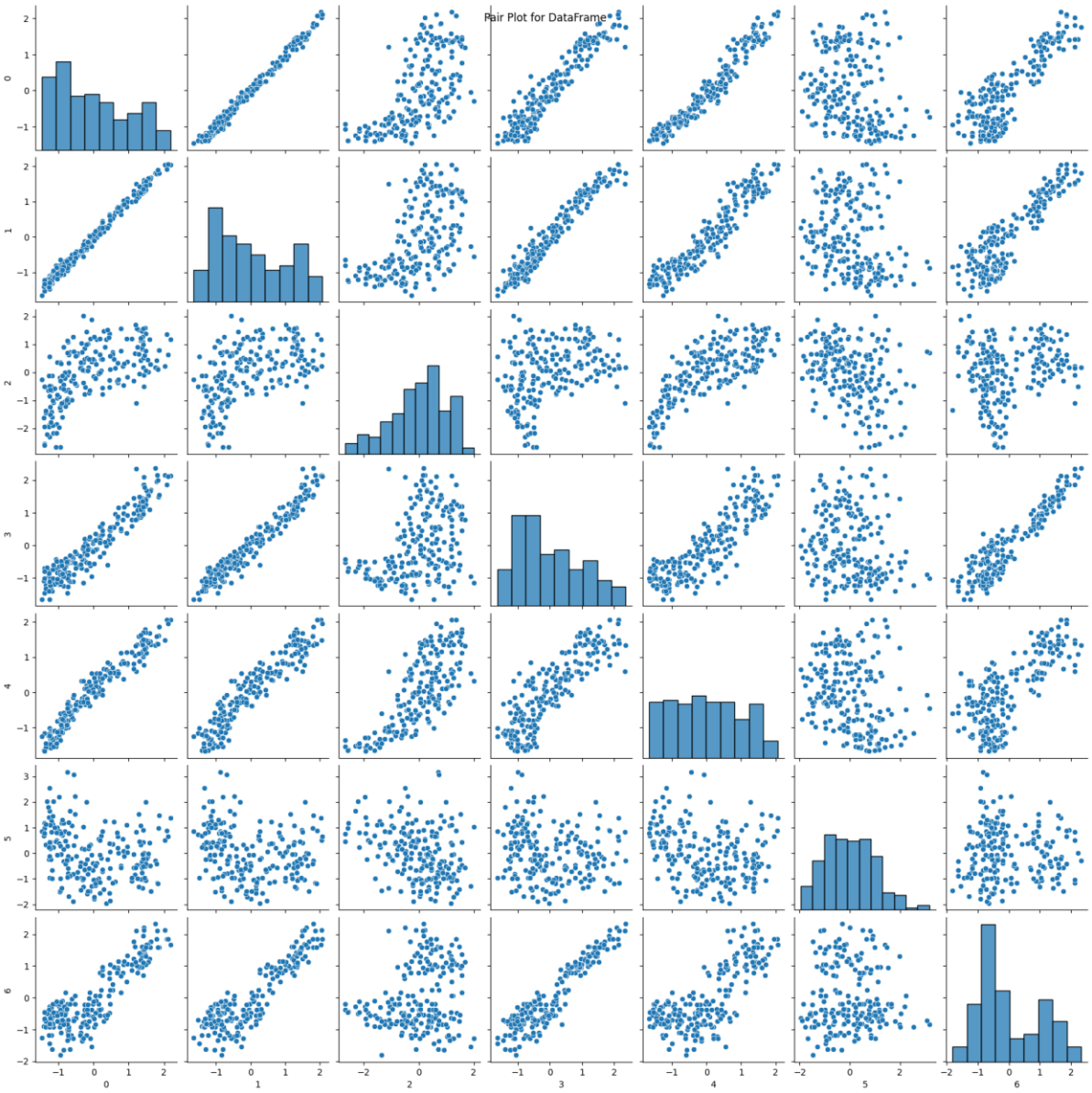
Univariate Analysis

Univariate analysis is an exploratory data analysis technique that focuses on understanding the distribution and characteristics of a single variable (feature) in a dataset.



Bivariate analysis

Bivariate analysis is an exploratory data analysis technique that examines the relationship between two variables (features) in a dataset.



Evaluating skewness

According to this dataset is relatively symmetrical.

```
0    0.399889
1    0.386573
2   -0.537954
3    0.525482
4    0.134378
5    0.401667
6    0.561897
dtype: float64
```

Deciding The Model Architecture

There are 4 models, in each model,

- ***No: of epochs: 100***
- ***Initial learning rate at 0.001***
- ***For each model optimizers were different.***

Model 1

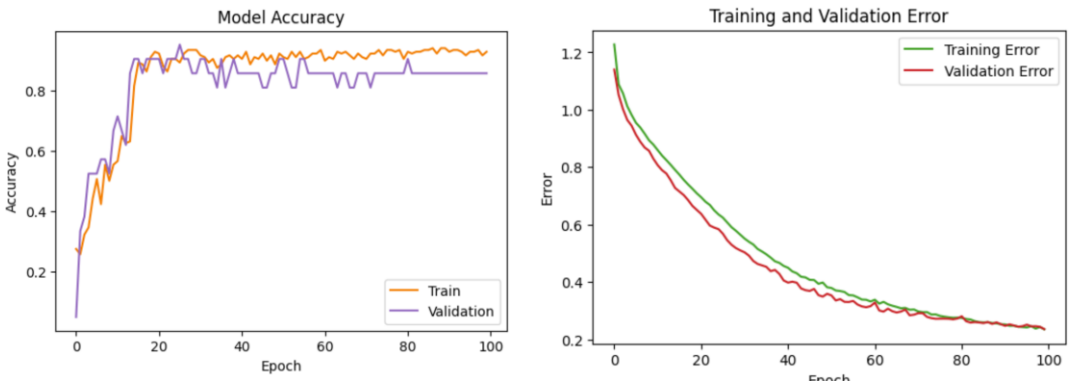
Optimizer – Adam

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 20)                160

 dense_1 (Dense)             (None, 20)                420

 dense_2 (Dense)             (None, 3)                 63

=================================================================
Total params: 643 (2.51 KB)
Trainable params: 643 (2.51 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```




```
1/1 [==============================] - 0s 28ms/step - loss: 0.2981 - accuracy: 0.9048
Test Accuracy: 0.9048
```

Model 2

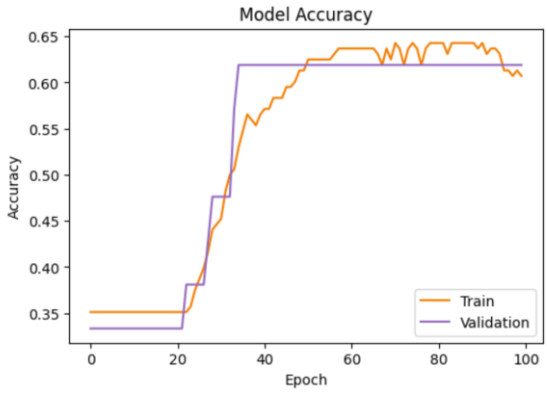
Optimizer – Adagrad

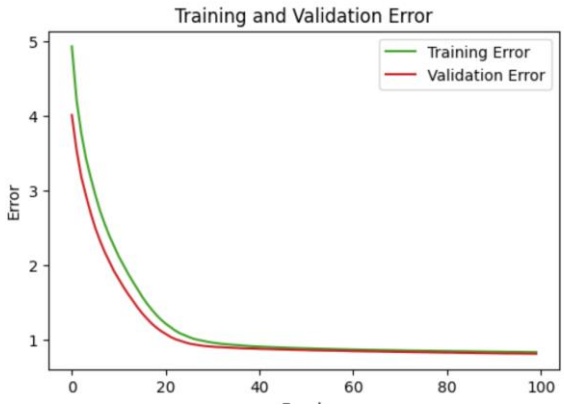
```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_3 (Dense)             (None, 30)                240

 dense_4 (Dense)             (None, 20)                620

 dense_5 (Dense)             (None, 3)                 63

=================================================================
Total params: 923 (3.61 KB)
Trainable params: 923 (3.61 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```






```
1/1 [==============================] - 0s 36ms/step - loss: 0.8617 - accuracy: 0.4762
Test Accuracy: 0.4762
```

Model 3

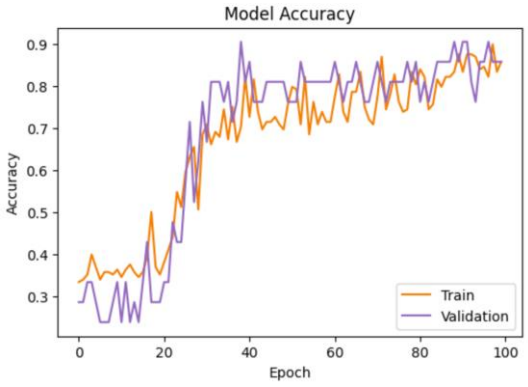
Optimizer – SGD

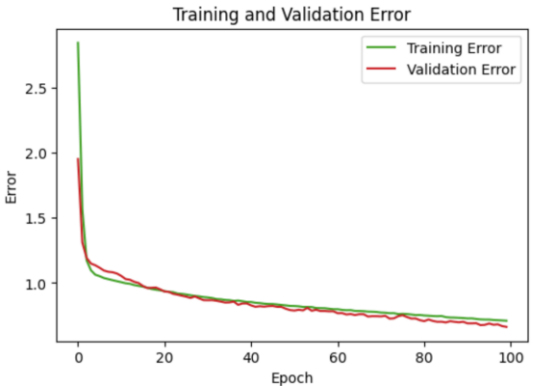
```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_6 (Dense)             (None, 10)                80

 dense_7 (Dense)             (None, 10)                110

 dense_8 (Dense)             (None, 3)                 33

=================================================================
Total params: 223 (892.00 Byte)
Trainable params: 223 (892.00 Byte)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```






```
1/1 [==============================] - 0s 29ms/step - loss: 0.7248 - accuracy: 0.8095
Test Accuracy: 0.8095
```

Model 4

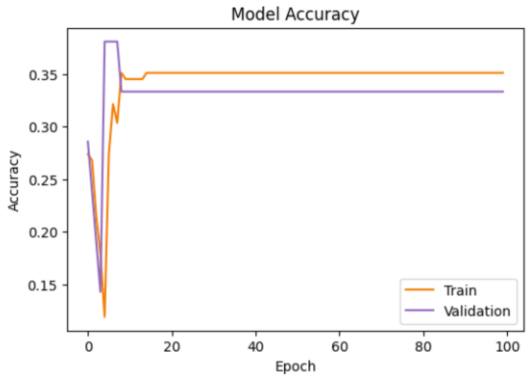
Optimizer – RMSprop

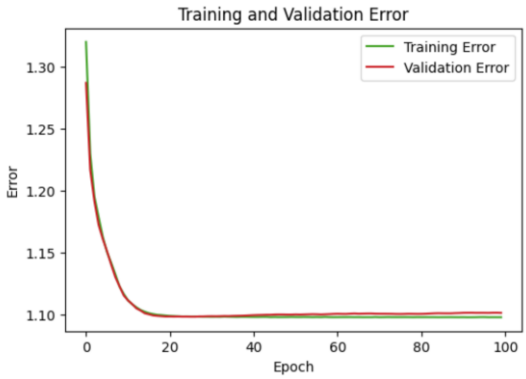
```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_9 (Dense)             (None, 6)                 48

 dense_10 (Dense)            (None, 10)                70

 dense_11 (Dense)            (None, 3)                 33

=================================================================
Total params: 151 (604.00 Byte)
Trainable params: 151 (604.00 Byte)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```






```
1/1 [==============================] - 0s 25ms/step - loss: 1.1106 - accuracy: 0.1905
Test Accuracy: 0.1905
```

Conclusion

The results demonstrate the impact of both model architecture and optimizer selection on model performance. Through a comprehensive exploratory data analysis and model development process.

For these Artificial Nural networks (ANN) model was conducted with various neuron sizes in the hidden layers. As well as model were used different optimizers such as Adam, SGD, Adagrad and RmsProp.

In conclusion, For ANNs , both model architecture and optimizer play a crucial role in how an ANN learns and performs.