Experiment: 1.3

Student Name: Priyanshu Mathur	**UID:** 20BEC1073
Branch: Electronics and Communication	**Section/Group:** A
Semester: 7th	**Date of Performance:** 12/9/23
Subject Name: Artificial intelligence & Machine Learning	**Subject Code:** 20ECA-445

1. **Aim:** Write a program to study the various metrics for comparison of ANN.

2. **Apparatus/Tool Used:** Google Colab.

3. **Theory**:

Mean Absolute Error (MAE)

Mean Absolute Error (MAE) is a popular metric used to evaluate the performance of regression models in machine learning and statistics. It measures the average magnitude of errors between predicted and actual values without considering their direction.:

$$\mathrm{MAE} = \frac{1}{\mathrm{n}} \sum |\mathrm{y} - \hat{\mathrm{y}}|$$

Mean Squared Error (MSE)

Mean Squared Error (MSE) is another widely used metric for assessing the performance of regression models in machine learning and statistics. It measures the average squared difference between the predicted and actual values, thus emphasizing larger errors:

$$\mathrm{MSE} = \frac{1}{\mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{n}} \left(\mathrm{Y_i} - \hat{\mathrm{Y}}_\mathrm{i}\right)^2$$

Root Mean Squared Error (RMSE)

The Mean Squared Error (MSE) square root measures the average squared difference between the predicted and actual values. Root Mean Squared Error (RMSE) has the same unit as the target variable, making it more interpretable and easier to relate to the problem context than MSE.

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{\mathrm{i}=1}^{\mathrm{N}} ||\mathrm{y(i)} - \hat{\mathrm{y}}\mathrm{(i)}||^2}{\mathrm{n}}}$$

Classification metrics

Classification metrics assess the performance of machine learning models for classification tasks. They aim to assign an input data point to one of several predefined categories.

Accuracy

Accuracy is a fundamental evaluation metric for assessing the overall performance of a classification model. It is the ratio of the correctly predicted instances to the total instances in the dataset. The formula for calculating accuracy is:

$$\text{Accuracy} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{FP}+\text{TN}+\text{FN}}$$

Confusion Matrix

A confusion matrix, also known as an error matrix, is a tool used to evaluate the performance of classification models in machine learning and statistics. It presents a summary of the predictions made by a classifier compared to the actual class labels, allowing for a detailed analysis of the classifier's performance across different classes. The confusion matrix provides a comprehensive view of the model's performance, including each class's correct and incorrect predictions.

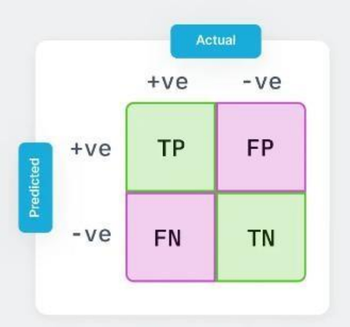


Fig 3.1 : Confusion Matrix[1]

- **TP: True Positives -** The number of patients with the disease correctly predicted as "yes."
- **TN: True Negatives -** The number of patients without the disease was correctly predicted as "no."
- **FP: False Positives -** The number of patients who don't have the disease but were incorrectly predicted as "yes."
- **FN: False Negatives -** The number of patients who have the disease but were incorrectly predicted as "no."

Precision (P) is the proportion of true positive predictions among all positive predictions. It is a measure of how accurate the positive predictions are.

Recall (R), also known as sensitivity or true positive rate (TPR), is the proportion of true positive predictions among all actual positive instances. It measures the classifier's ability to identify positive instances correctly.

4. Code

```
import numpy as np # linear algebra import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv) df =
pd.read_csv('Churn_Modelling.csv') df.head() # to see change in dataset df.drop(columns =
['RowNumber','CustomerId','Surname','Gender'],inplace=True) # to drop columns those are not in need.
df.head() df['Geography'].value_counts() df =
pd.get_dummies(df,columns=['Geography'],drop_first=True)
df.head()
X = df.drop(columns=['Exited']) y
= df['Exited'].values
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=0)
from sklearn.preprocessing import StandardScaler scaler = StandardScaler()
X_train_trf = scaler.fit_transform(X_train) X_test_trf
= scaler.transform(X_test) import tensorflow from
tensorflow import keras from tensorflow.keras import
Sequential from tensorflow.keras.layers import Dense
model = Sequential()
model.add(Dense(10,activation='relu',input_dim=10))
model.add(Dense(64,activation='relu'))
model.add(Dense(32,activation='relu'))
model.add(Dense(128,activation='relu'))
model.add(Dense(1,activation='sigmoid'))
model.summary()
model.compile(optimizer='Adam',loss='binary_crossentropy',metrics=['accuracy']) history =
model.fit(X_train,y_train,batch_size=50,epochs=100,verbose=1,validation_split=0.2) y_pred =
model.predict(X_test) y_pred = y_pred.argmax(axis=-1) from sklearn.metrics import
accuracy_score accuracy_score(y_test,y_pred) import matplotlib.pyplot as plt
plt.plot(history.history['loss']) plt.plot(history.history['val_loss'])
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy']) from sklearn.metrics import
confusion_matrix , classification_report
print(classification_report(y_test,y_pred)) import seaborn as sn cm =
tensorflow.math.confusion_matrix(labels=y_test,predictions=y_pred)
plt.figure(figsize = (4,4)) sn.heatmap(cm, annot=True, fmt='d')
plt.xlabel('Predicted') plt.ylabel('Truth')
```

5. Observation:

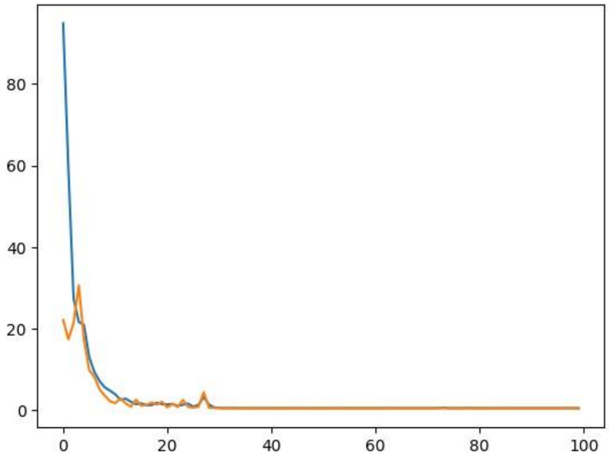


Fig.5.1: Val vs ValLoss.

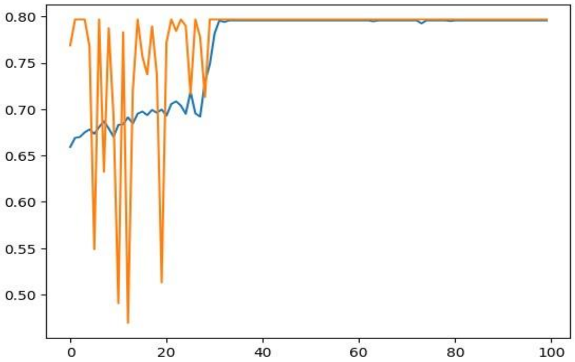
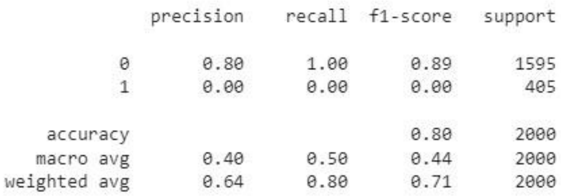


Fig.5.2: Accuracy vs val Accuracy



```
              precision    recall  f1-score   support

           0       0.80      1.00      0.89      1595
           1       0.00      0.00      0.00       405

    accuracy                           0.80      2000
   macro avg       0.40      0.50      0.44      2000
weighted avg       0.64      0.80      0.71      2000
```

Fig.5.3: Classification Report.

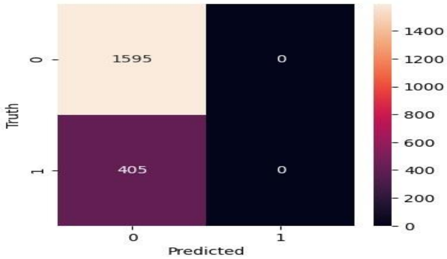


Fig.5.4: Confusion Matrix.

6. **Result:** Model is trained and get 80% accuracy and 80% precision.

7. **Conclusion**: This theoretical framework provides a structured approach to evaluate and compare ANN models, fostering transparency and reproducibility in the research and application of artificial neural networks. As ANNs continue to evolve, a standardized methodology for assessment becomes increasingly vital to make informed decisions regarding model selection and deployment.

8. Learning Outcomes:

- Learn about comprehensive understanding of ANN Metrics.
- Learn about confusion matrix.
- Learn to implement and find accuracy of the dataset.

9. References:

1. https://www.v7labs.com/blog/performance-metrics-in-machine-learning.