

## Practical NO:- 2.

Aim:- Creating functions to compute various losses

### Theory:-

Deep learning models are neural networks with multiple layers that learn complex patterns from data.

#### \* Loss Function in Deep learning:-

loss function helps in updating the model's weights during training.

- (1) Mean Square Error (MSE) :- Commonly used in regression tasks, calculating the average squared difference between actual & the predicted values.
- (2) Mean Absolute Error (MAE) :- Measures the average absolute difference between actual & the predicted values, makes it less sensitive to outliers than MSE.
- (3) Categorical Cross Entropy :- used for multi-class classification problems, measuring how well the predicted probability distribution matches the actual labels.

#### Activation Function:-

- The ReLU (Rectified Linear Unit) activation function is used in the hidden layers. It introduces non-linearity into the model, helping it learn complex patterns.



## Neural Network Architecture:

- The deep learning model in this practical consists of

- Input layer

- Two hidden layers:-

• Hidden layer 1: 10 neurons

• Hidden layer 2: 8 neurons.

- Output layer.

Each neuron in one layer is connected to every neuron in the next layer using dense (fully connected) layers.

optimizer.

- Adam optimizer is used to adjust the network's weights efficiently during training. It combines the benefits of momentum and adaptive learning rates for faster convergence.

### Data set split

The data set is divided into 80% training & 20% testing to evaluate the model's performance.

### Implementation,

- The model is built using TensorFlow and Keras.

The key libraries used are:-

• Numpy & Pandas:- For handling numerical data.

• TensorFlow & Keras:- For defining and training the deep learning model.

Aim: Creating

Manasi Gaikwad

```
[10] import tensorflow as tf
from tensorflow.keras import layers
from tensorflow.keras import models
import pandas as pd
from sklearn.preprocessing import StandardScaler
```

```
# Load the data
df = pd.read_csv('data.csv')
```

```
# Drop the unnecessary columns
df.drop('id', axis=1, inplace=True)
```

```
[10] # Drop the unnecessary columns
df.drop('id', axis=1, inplace=True)
```

```
# Encode the categorical variables
for column in df.columns:
    if df[column].dtype == object:
```

```
# Separate the features and target variable
X = df[['feature1', 'feature2', 'feature3']]
y = df['target']
```

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Scale the features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```



Conclusion:-

The practical focuses on building a neural network for fraud detection by pre-processing data, using RELU activation, binary crossentropy loss, and evaluating performance with accuracy, MSE and MAE.

~~11/13/25~~