Lab 2

Aim : Design and develop an Artificial Neural Network model which can predict the severity of accident

```
import pandas as pd
import numpy as np
import seaborn as sb
import matplotlib.pyplot as plt
```

```
df = pd.read_csv("Airplane Accidents.csv")
```

```
df
```

	id	safety_score	days_since_inspection	complaints	control_metric
0	1	49.223743	14	22	71.285324
1	2	62.465750	10	27	72.288055
2	3	63.059360	13	16	66.362810
3	4	48.082190	11	9	74.703735
4	5	26.484018	13	25	47.948950
...	...	...	...	...	...
9995	9996	56.118720	8	1	63.445763
9996	9997	40.365295	10	7	62.169550
9997	9998	27.853882	17	1	69.598910
9998	9999	56.210045	8	0	39.835915
9999	10000	50.000000	13	3	45.487694

10000 rows × 12 columns

```
#!pip install pandas-profiling
```

```
#pip install -U ydata-profiling
```

EDA

```
#! pip install https://github.com/pandas-profiling/pandas-profiling/archive/master.zip
```

```
from sklearn.preprocessing import LabelEncoder

# Assuming 'severity' is the column you want to label encode
column_to_encode = 'severity'

# Initialize the LabelEncoder
label_encoder = LabelEncoder()

# Fit and transform the 'severity' column
df[column_to_encode] = label_encoder.fit_transform(df[column_to_encode])

# Display the updated DataFrame
df
```

	id	safety_score	days_since_inspection	complaints	control_metric
0	1	49.223743	14	22	71.285324
1	2	62.465750	10	27	72.288055
2	3	63.059360	13	16	66.362810
3	4	48.082190	11	9	74.703735
4	5	26.484018	13	25	47.948950
...	...	...	...	...	...
9995	9996	56.118720	8	1	63.445763
9996	9997	40.365295	10	7	62.169550
9997	9998	27.853882	17	1	69.598910
9998	9999	56.210045	8	0	39.835915
9999	10000	50.000000	13	3	45.487694

10000 rows × 12 columns

```
# from pandas_profiling import ProfileReport
'''from ydata_profiling import ProfileReport
ProfileReport(df)'''
```

```
'from ydata_profiling import ProfileReport\nProfileReport(df)'
```

```
df=df.dropna()
```

Train Test split

```
# separating outcome and features
x=df.drop(columns='severity')
y=df.severity

# split the data into train and test
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=0)

#Standardise data
from sklearn.preprocessing import StandardScaler
st=StandardScaler()
# st.fit(x_train)
# st.fit(x_test)

x_train_std=st.fit_transform(x_train)
x_test_std=st.fit_transform(x_test)


x_train_std.shape
```

```
(7492, 11)
```

```
#!pip install livelossplot


# Import the libraries
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Input, Dense
from livelossplot import PlotLossesKerasTF
from tensorflow.keras.optimizers import SGD
```

Model Building

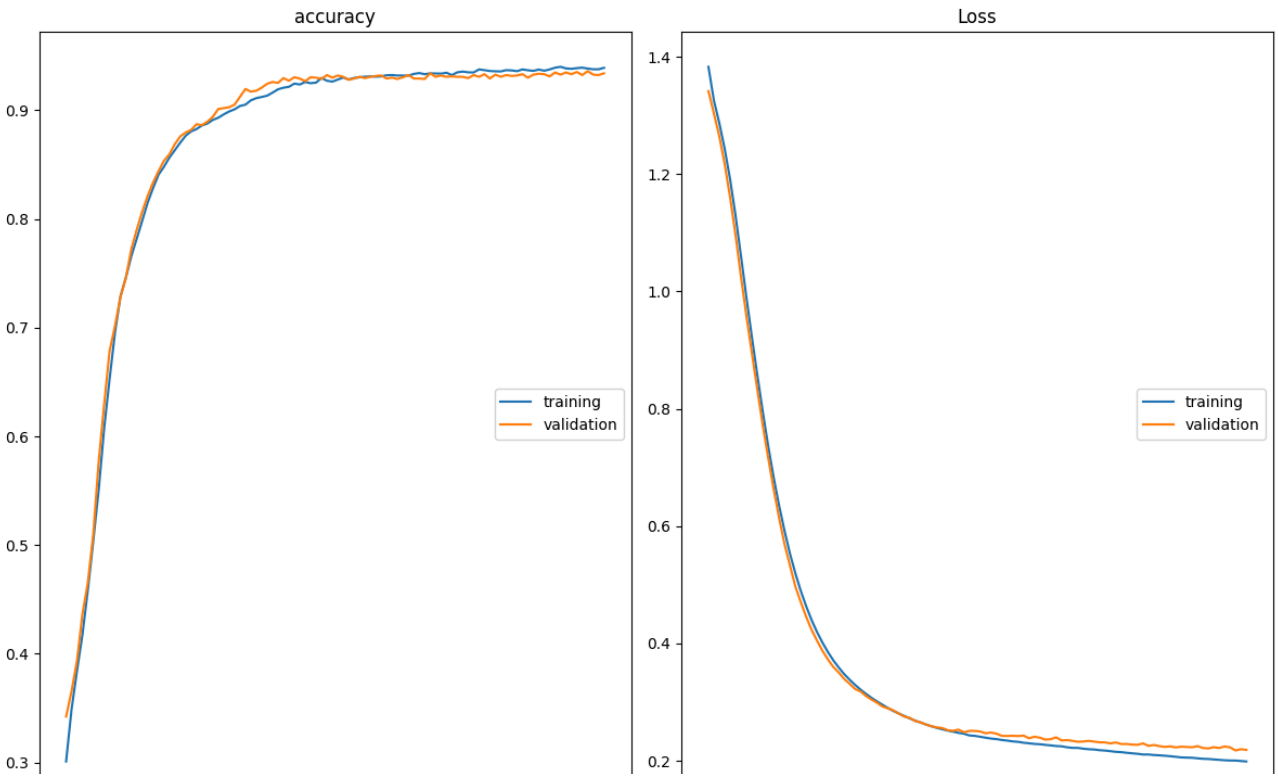
```
model=Sequential()

model.add(Input(shape=(11,),name="Input Layer"))
model.add(Dense(15,activation='relu',name="Hidden1"))
model.add(Dense(20,activation='relu',name="Hidden2"))
model.add(Dense(4,activation='softmax',name='OutputLayer'))
```

Batch size SGD

```
opt = tf.keras.optimizers.SGD(learning_rate=0.01)

model.compile(loss='sparse_categorical_crossentropy',optimizer=opt,metrics=['accuracy']
model.fit(x_train_std,y_train,epochs=100,batch_size=32, validation_data=(x_test_std,y_t
```


accuracy
0.9
0.8
0.7
0.6
0.5
0.4
0.3
training
validation
Loss
1.4
1.2
1.0
0.8
0.6
0.4
0.2
training
validation

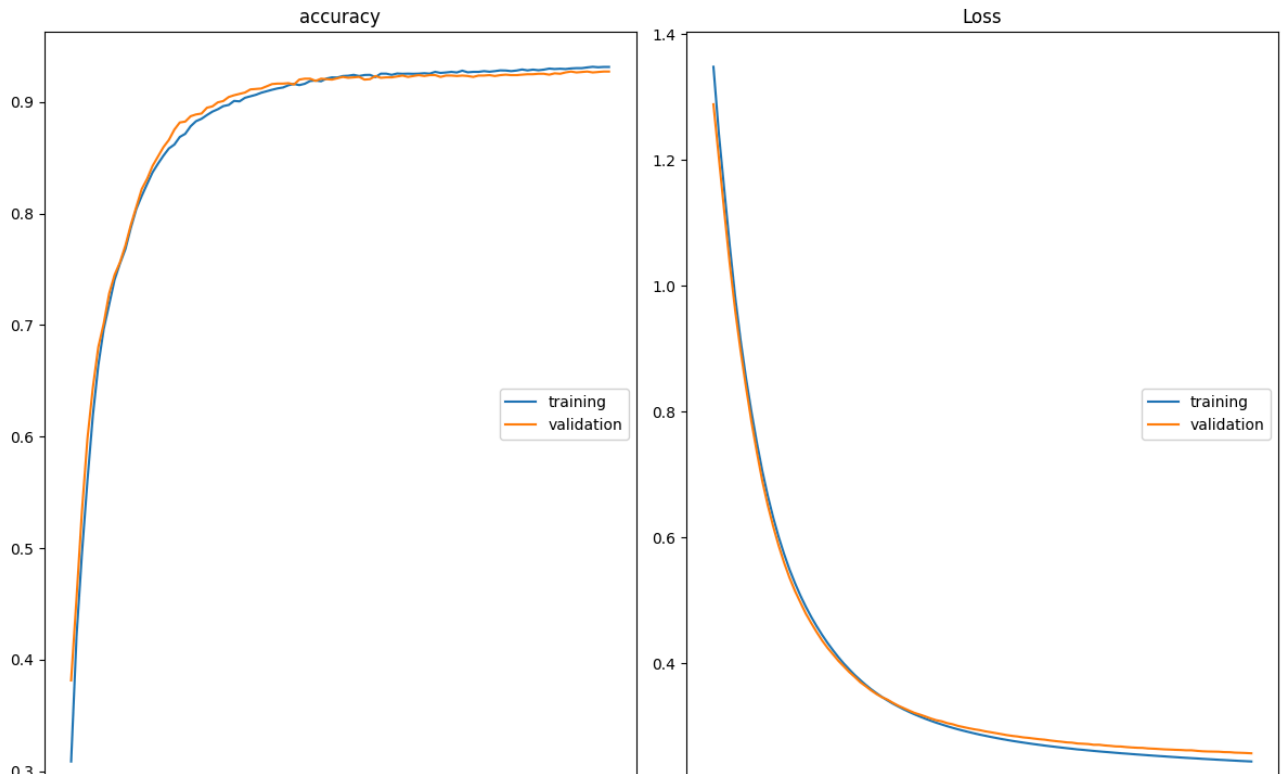
AdaGrad

```
model1=Sequential()

model1.add(Input(shape=(11,),name="Input Layer"))
model1.add(Dense(15,activation='relu',name="Hidden1"))
model1.add(Dense(20,activation='relu',name="Hidden2"))
model1.add(Dense(4,activation='softmax',name='OutputLayer'))

learning_rate=0.01
opt=tf.keras.optimizers.Adagrad(learning_rate=learning_rate)

model1.compile(loss='sparse_categorical_crossentropy',optimizer=opt,metrics=['accuracy'
model1.fit(x_train_std,y_train,epochs=100,batch_size=16,validation_data=(x_test_std,y_t
```



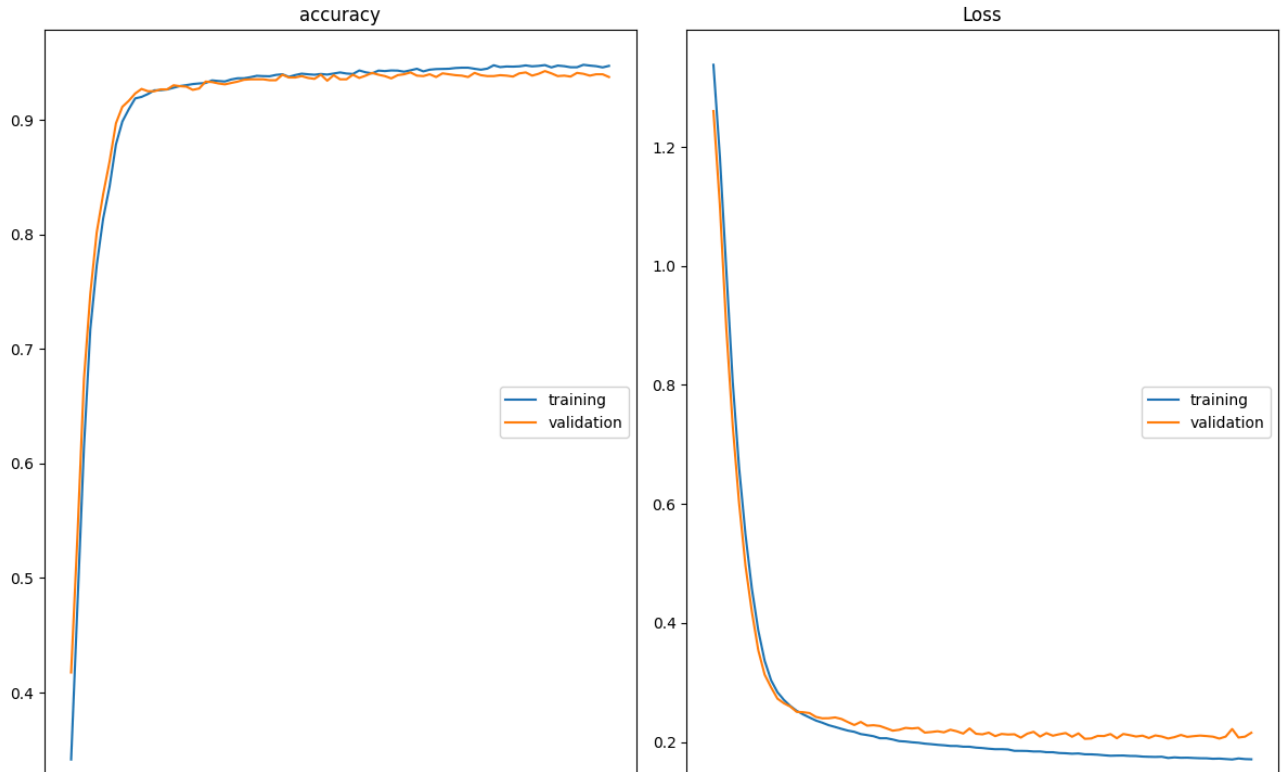
RMS Prop

```
model2=Sequential()

model2.add(Input(shape=(11,),name="Input Layer"))
model2.add(Dense(15,activation='relu',name="Hidden1"))
model2.add(Dense(20,activation='relu',name="Hidden2"))
model2.add(Dense(4,activation='softmax',name='OutputLayer'))

opt=tf.keras.optimizers.RMSprop()

model2.compile(loss='sparse_categorical_crossentropy',optimizer=opt,metrics=['accuracy'
model2.fit(x_train_std,y_train,epochs=85,batch_size=32,validation_data=(x_test_std,y_te
```



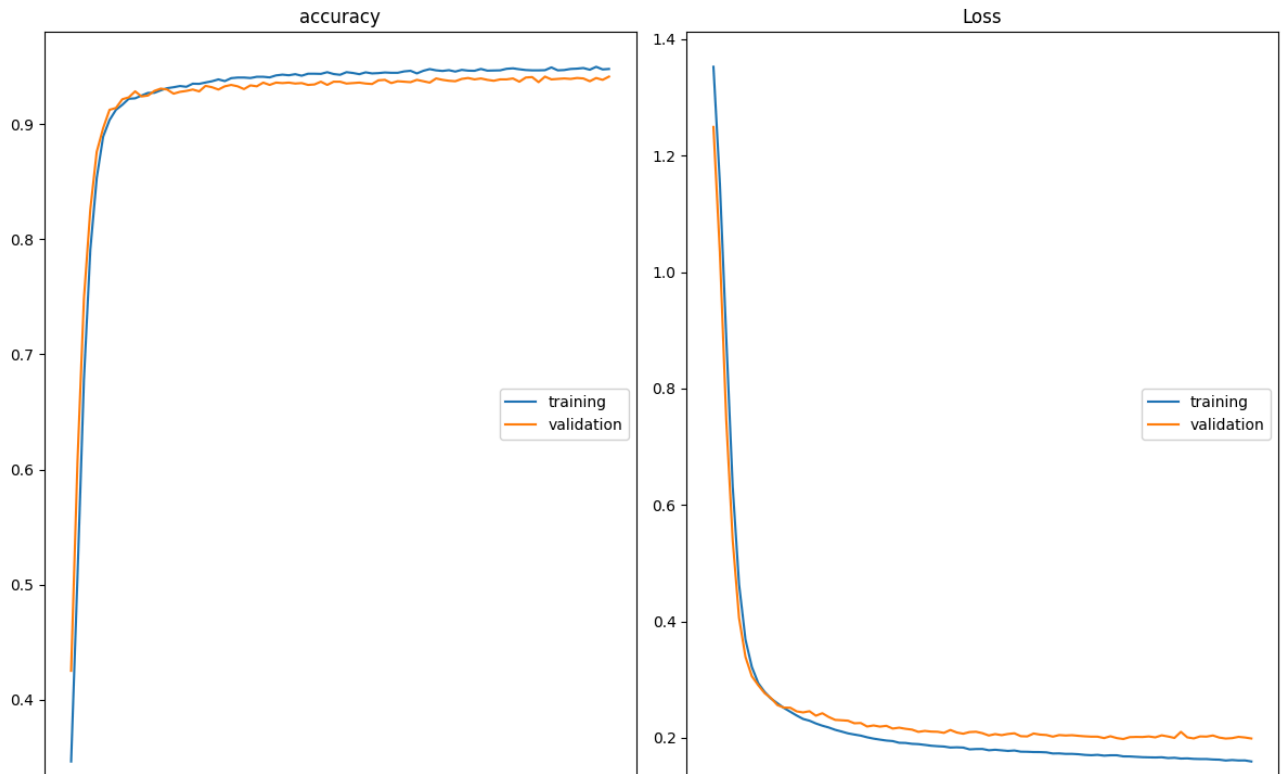
Adam

```
model3=Sequential()

model3.add(Input(shape=(11,),name="Input Layer"))
model3.add(Dense(15,activation='relu',name="Hidden1"))
model3.add(Dense(20,activation='relu',name="Hidden2"))
model3.add(Dense(4,activation='softmax',name='OutputLayer'))

opt=tf.keras.optimizers.Adam()

model3.compile(loss='sparse_categorical_crossentropy',optimizer=opt,metrics=['accuracy'
model3.fit(x_train_std,y_train,epochs=85,batch_size=24,validation_data=(x_test_std,y_te
```



```
from sklearn.metrics import classification_report

t= model.predict(x_test_std)
```

```
      79/79 [==============================] - 0s 1ms/step
```