In [1]:
```
import pandas as pd
import numpy as np

# Converting the Categorial data into Numberical data
from sklearn.preprocessing import LabelEncoder
```

In [2]:
```
from sklearn.model_selection import train_test_split

# Converting that converted Categorial data into number by using to_cate : w
from keras.utils.np_utils import to_categorical

#Using Sequential Architecture
from keras.models import Sequential

# Create Matrix Dense More 1 and less 0
from keras.layers import Dense
```

In [3]:
```
1 ls
```

```
 Volume in drive C has no label.
 Volume Serial Number is A007-E621

 Directory of C:\Users\tswar\Documents\DATA SETS

09/29/2022  02:37 PM    <DIR>          .
09/29/2022  02:37 PM    <DIR>          ..
09/29/2022  02:08 PM    <DIR>          .ipynb_checkpoints
05/07/2022  10:28 PM             4,213 Advertising.csv
09/29/2022  02:37 PM             7,363 ANN.ipynb
05/07/2022  10:25 PM            16,161 Bahaman.xlsx
09/29/2019  08:41 AM           938,020 Bengaluru_House_Data.csv
05/07/2022  10:25 PM             9,048 Bolt diameter.xlsx
05/07/2022  10:26 PM            35,203 boston.csv
05/07/2022  10:29 PM           684,858 Churn_Modelling (1).csv
05/07/2022  10:26 PM            29,822 claimants.csv
05/07/2022  10:25 PM            10,738 Contract Renewal.xlsx
09/20/2019  12:04 AM       150,828,752 creditcard.csv
07/20/2022  11:18 AM         9,224,265 Crop-wise_State-wise_Land holdings_Area_
Number.xlsx
08/09/2022  05:44 PM            26,842 Data Transformation.ipynb
08/04/2022  09:49 AM         5,057,493 data.csv
04/19/2022  05:39 PM            23,873 diabetes.csv
05/07/2022  10:29 PM         2,029,440 energy.xlsx
05/07/2022  10:25 PM             8,713 Fabric data.xlsx
08/31/2022  03:40 PM         5,694,462 Final.pkl
07/28/2022  08:59 AM         7,834,511 Fuel Consumption EDA.ipynb
09/01/2022  10:13 AM           136,968 Fuel_Consumption.ipynb
05/07/2022  10:28 PM            33,036 gene_expression.csv
08/09/2022  05:09 PM           171,155 GridSearchCV.ipynb
05/07/2022  10:28 PM           102,993 hearing_test.csv
10/24/2019  02:52 AM        41,399,186 Hindi_English_Truncated_Corpus.csv
05/07/2022  10:27 PM               297 homeprices (1).csv
05/07/2022  10:27 PM               297 homeprices.csv
09/07/2022  12:27 PM           251,015 IMG_20210202_121728.jpg
05/07/2022  10:24 PM               809 Indian Liver Patient Dataset -Descriptio
n.txt
05/07/2022  10:24 PM            23,814 Indian Liver Patient.csv
05/07/2022  10:25 PM            20,450 JohnyTalkers.xlsx
07/21/2022  03:12 PM           978,285 KingTran.csv
05/07/2022  10:29 PM             4,286 Mall_Customers.csv
05/14/2022  08:16 PM           132,037 Matplotlib.ipynb
05/20/2022  01:04 PM           930,983 mini -py.pdf
05/07/2022  10:28 PM            15,698 mouse_viral_study.csv
05/07/2022  10:28 PM           374,003 mushrooms.csv
04/06/2022  02:41 PM            73,008 MY2022 Fuel Consumption Ratings.csv
07/20/2022  11:24 AM         1,434,704 nigeria_houses_data.csv
08/31/2022  03:19 PM            20,220 ohe.joblib
05/07/2022  10:28 PM            13,519 penguins_size.csv
02/24/2022  10:55 AM        41,399,186 Project2(Hindi_English_Corpus).csv
05/07/2022  10:25 PM            13,309 Promotion.xlsx
08/27/2022  11:00 AM           280,236 PySpark.ipynb
08/05/2022  02:19 PM         2,315,464 Regression Overfitting.pdf
08/04/2022  11:38 AM            38,472 sentiment analysis.ipynb
06/15/2022  02:33 PM           643,041 simple-linear-regression.pdf
```

```
05/21/2020  07:27 AM           608,346 SOCR-HeightWeight.csv
05/07/2022  10:27 PM         2,635,787 stroke prediction.csv
05/07/2022  10:26 PM             8,008 tips.csv
05/07/2022  10:27 PM            61,194 titanic.csv
11/07/2021  01:07 PM            60,302 train.csv
08/09/2022  05:39 PM           157,028 Untitled.ipynb
07/27/2022  04:15 PM            10,806 Untitled1.ipynb
07/28/2022  10:06 PM        21,720,880 Untitled2.ipynb
08/23/2022  11:20 AM            94,444 Untitled3.ipynb
08/29/2022  09:23 PM           232,817 Untitled4.ipynb
08/31/2022  03:40 PM           339,470 Untitled5.ipynb
09/01/2022  05:24 PM             1,869 Untitled6.ipynb
09/17/2022  11:17 AM               588 Untitled7.ipynb
05/07/2022  10:27 PM            11,462 Wine.csv
              59 File(s)    299,213,249 bytes
               3 Dir(s)  92,792,340,480 bytes free
```

In [4]:
```
df= pd.read_excel("Crop-wise_State-wise_Land holdings_Area_Number.xlsx")
df.head(3)
```

Out[4]:

	2010-11	Unnamed: 1	Unnamed: 2	Unnamed: 3	Hectares	Hectares.1	Hectares.2	Hectares.3	Hectares.4
0	S. No.	Category	Crop	State	Avg area of marginal holdings	Avg area of small land holdings	Avg area of semi medium land holdings	Avg area of medium land holdings	Avg area of large land holdings
1	1	Cereals	Paddy	Andhra Pradesh & Telangana	0.484623	1.229168	2.379752	3.708802	6.291208
2	NaN	Cereals	Paddy	Assam	0.416651	1.245475	2.450684	4.666109	44.198373

In [5]:
```
l={j:i for i,j in zip(df.iloc[0],df.columns)}
```

In [6]:
```
l
```

Out[6]:
```
{'2010-11': 'S. No.',
 'Unnamed: 1': 'Category',
 'Unnamed: 2': 'Crop',
 'Unnamed: 3': 'State',
 'Hectares': 'Avg area of marginal holdings',
 'Hectares.1': 'Avg area of small land holdings',
 'Hectares.2': 'Avg area of semi medium land holdings',
 'Hectares.3': 'Avg area of medium land holdings',
 'Hectares.4': 'Avg area of large land holdings',
 'Hectares.5': 'Avg area of institutional land holdings',
 'Hectares.6': 'Avg area of total land holdings'}
```

In [7]:
```
df.rename(columns=l,inplace=True)
```

In [8]:
```
1 df
```

Out[8]:

	S. No.	Category	Crop	State	Avg area of marginal holdings	Avg area of small land holdings	Avg area of semi medium land holdings	Avg area of medium land holdings	Avg area of large land holdings	Av ins
0	S. No.	Category	Crop	State	Avg area of marginal holdings	Avg area of small land holdings	Avg area of semi medium land holdings	Avg area of medium land holdings	Avg area of large land holdings	Av in
1	1	Cereals	Paddy	Andhra Pradesh & Telangana	0.484623	1.229168	2.379752	3.708802	6.291208	
2	NaN	Cereals	Paddy	Assam	0.416651	1.245475	2.450684	4.666109	44.198373	3
3	NaN	Cereals	Paddy	Bihar	0.239013	1.131109	2.300838	4.440552	12.592312	
4	NaN	Cereals	Paddy	Chhattisgarh	0.446134	1.321186	2.375793	4.746052	12.273771	
...	...	...	...	...	...	...	...	...	...	
4322	NaN	NaN	Total Non-food Crops	Tamil Nadu	0.213988	0.696387	1.123242	2.060392	10.256437	
4323	NaN	NaN	Total Non-food Crops	Uttar Pradesh	0.286915	0.787019	1.251811	2.250594	5.10521	
4324	NaN	NaN	Total Non-food Crops	Uttarakhand	NaN	0.209174	0.426766	0.840273	3.66426	
4325	NaN	NaN	Total Non-food Crops	West Bengal	0.189433	0.842182	1.109985	1.342852	254.832707	8
4326	NaN	NaN	Total Non-food Crops	Others	0.185093	0.591636	0.711552	0.782285	1.816722	

4327 rows × 11 columns

In [9]:
```
1 df.columns
```

Out[9]:
```
Index(['S. No.', 'Category', 'Crop', 'State', 'Avg area of marginal holdings',
       'Avg area of small land holdings',
       'Avg area of semi medium land holdings',
       'Avg area of medium land holdings', 'Avg area of large land holdings',
       'Avg area of institutional land holdings',
       'Avg area of total land holdings'],
      dtype='object')
```

In [10]:
```
df.drop(columns=['S. No.', 'Category', 'State'],inplace=True)
```

In [11]:
```
df
```

Out[11]:

	Crop	Avg area of marginal holdings	Avg area of small land holdings	Avg area of semi medium land holdings	Avg area of medium land holdings	Avg area of large land holdings	Avg area of institutional land holdings	Avg area of total land holdings
0	Crop	Avg area of marginal holdings	Avg area of small land holdings	Avg area of semi medium land holdings	Avg area of medium land holdings	Avg area of large land holdings	Avg area of institutional land holdings	Avg area of total land holdings
1	Paddy	0.484623	1.229168	2.379752	3.708802	6.291208	5.184929	0.933116
2	Paddy	0.416651	1.245475	2.450684	4.666109	44.198373	31.087507	1.012407
3	Paddy	0.239013	1.131109	2.300838	4.440552	12.592312	0.944078	0.372548
4	Paddy	0.446134	1.321186	2.375793	4.746052	12.273771	9.469729	1.246811
...	...	...	...	...	...	...	...	...
4322	Total Non-food Crops	0.213988	0.696387	1.123242	2.060392	10.256437	8.777187	0.430444
4323	Total Non-food Crops	0.286915	0.787019	1.251811	2.250594	5.10521	0.747321	0.425144
4324	Total Non-food Crops	NaN	0.209174	0.426766	0.840273	3.66426	3.026786	0.888889
4325	Total Non-food Crops	0.189433	0.842182	1.109985	1.342852	254.832707	82.434179	0.200948
4326	Total Non-food Crops	0.185093	0.591636	0.711552	0.782285	1.816722	1.616312	0.570552

4327 rows × 8 columns

In [12]:
```
df.isnull().sum()
```

Out[12]:
```
Crop                                    0
Avg area of marginal holdings        1664
Avg area of small land holdings      1684
Avg area of semi medium land holdings    1729
Avg area of medium land holdings     1871
Avg area of large land holdings      2373
Avg area of institutional land holdings    2946
Avg area of total land holdings      1429
dtype: int64
```

In [13]:
```
df.drop(0,inplace=True)
```

In [14]:
```
df
```

Out[14]:

	Crop	Avg area of marginal holdings	Avg area of small land holdings	Avg area of semi medium land holdings	Avg area of medium land holdings	Avg area of large land holdings	Avg area of institutional land holdings	Avg area of total land holdings
1	Paddy	0.484623	1.229168	2.379752	3.708802	6.291208	5.184929	0.933116
2	Paddy	0.416651	1.245475	2.450684	4.666109	44.198373	31.087507	1.012407
3	Paddy	0.239013	1.131109	2.300838	4.440552	12.592312	0.944078	0.372548
4	Paddy	0.446134	1.321186	2.375793	4.746052	12.273771	9.469729	1.246811
5	Paddy	0.361901	1.051006	1.709356	3.008819	12.770878	2.807692	1.069407
...	...	...	...	...	...	...	...	...
4322	Total Non-food Crops	0.213988	0.696387	1.123242	2.060392	10.256437	8.777187	0.430444
4323	Total Non-food Crops	0.286915	0.787019	1.251811	2.250594	5.10521	0.747321	0.425144
4324	Total Non-food Crops	NaN	0.209174	0.426766	0.840273	3.66426	3.026786	0.888889
4325	Total Non-food Crops	0.189433	0.842182	1.109985	1.342852	254.832707	82.434179	0.200948
4326	Total Non-food Crops	0.185093	0.591636	0.711552	0.782285	1.816722	1.616312	0.570552

4326 rows × 8 columns

In [15]:
```
df1 =df.copy()
```

In [16]:
```
df1.isnull().sum()
```

Out[16]:
```
Crop                                    0
Avg area of marginal holdings        1664
Avg area of small land holdings      1684
Avg area of semi medium land holdings 1729
Avg area of medium land holdings     1871
Avg area of large land holdings      2373
Avg area of institutional land holdings 2946
Avg area of total land holdings      1429
dtype: int64
```

In [17]:
```
df1.dropna(inplace=True)
```

In [18]:
```
df1.isnull().sum()
```

Out[18]:
```
Crop                                    0
Avg area of marginal holdings           0
Avg area of small land holdings         0
Avg area of semi medium land holdings   0
Avg area of medium land holdings        0
Avg area of large land holdings         0
Avg area of institutional land holdings 0
Avg area of total land holdings         0
dtype: int64
```

In [19]:
```
df1.shape
```

Out[19]:
```
(1232, 8)
```

In [20]:
```
x = df1.drop("Crop",axis=1)
x
```

Out[20]:

	Avg area of marginal holdings	Avg area of small land holdings	Avg area of semi medium land holdings	Avg area of medium land holdings	Avg area of large land holdings	Avg area of institutional land holdings	Avg area of total land holdings
1	0.484623	1.229168	2.379752	3.708802	6.291208	5.184929	0.933116
2	0.416651	1.245475	2.450684	4.666109	44.198373	31.087507	1.012407
3	0.239013	1.131109	2.300838	4.440552	12.592312	0.944078	0.372548
4	0.446134	1.321186	2.375793	4.746052	12.273771	9.469729	1.246811
5	0.361901	1.051006	1.709356	3.008819	12.770878	2.807692	1.069407
...	...	...	...	...	...	...	...
4321	0.392997	1.042435	1.905515	3.918598	9.101863	5.086464	1.262967
4322	0.213988	0.696387	1.123242	2.060392	10.256437	8.777187	0.430444
4323	0.286915	0.787019	1.251811	2.250594	5.10521	0.747321	0.425144
4325	0.189433	0.842182	1.109985	1.342852	254.832707	82.434179	0.200948
4326	0.185093	0.591636	0.711552	0.782285	1.816722	1.616312	0.570552

1232 rows × 7 columns

In [21]:
```
y=df1["Crop"]
y
```

Out[21]:
```
1                     Paddy
2                     Paddy
3                     Paddy
4                     Paddy
5                     Paddy
                 ...
4321    Total Non-food Crops
4322    Total Non-food Crops
4323    Total Non-food Crops
4325    Total Non-food Crops
4326    Total Non-food Crops
Name: Crop, Length: 1232, dtype: object
```

In [22]:
```
y.nunique()
```

Out[22]:
```
183
```

In [23]:
```
X = np.array(x).astype('float32')
X
```

Out[23]:
```
array([[4.8462254e-01, 1.2291677e+00, 2.3797522e+00, ..., 6.2912078e+00,
        5.1849294e+00, 9.3311584e-01],
       [4.1665059e-01, 1.2454755e+00, 2.4506841e+00, ..., 4.4198372e+01,
        3.1087507e+01, 1.0124074e+00],
       [2.3901324e-01, 1.1311086e+00, 2.3008375e+00, ..., 1.2592312e+01,
        9.4407827e-01, 3.7254775e-01],
       ...,
       [2.8691533e-01, 7.8701931e-01, 1.2518107e+00, ..., 5.1052103e+00,
        7.4732095e-01, 4.2514360e-01],
       [1.8943344e-01, 8.4218192e-01, 1.1099848e+00, ..., 2.5483270e+02,
        8.2434181e+01, 2.0094769e-01],
       [1.8509305e-01, 5.9163648e-01, 7.1155167e-01, ..., 1.8167224e+00,
        1.6163125e+00, 5.7055187e-01]], dtype=float32)
```

In [24]:
```
Y = np.array(y)
Y
```

Out[24]:
```
array(['Paddy', 'Paddy', 'Paddy', ..., 'Total Non-food Crops',
       'Total Non-food Crops', 'Total Non-food Crops'], dtype=object)
```

In [25]:
```
le = LabelEncoder()
le
```

Out[25]:
```
▾ LabelEncoder

LabelEncoder()
```

In [26]:
```
le.fit(Y)
```

Out[26]:
```
LabelEncoder
LabelEncoder()
```

In [27]:
```
Y = le.transform(Y)
Y
```

Out[27]:
```
array([110, 110, 110, ..., 165, 165, 165])
```

In [28]:
```
Y = to_categorical(Y)
Y
```

Out[28]:
```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]], dtype=float32)
```

In [29]:
```
x_train,x_test,y_train,y_test =train_test_split(X,Y,test_size=0.3,random_sta
```

In [30]:
```
x_train.shape,y_train.shape
```

Out[30]:
```
((862, 7), (862, 183))
```

In [31]:
```
x_test.shape,y_test.shape
```

Out[31]:
```
((370, 7), (370, 183))
```

In [32]:
```
df1 = len(df1.columns)-1

model = Sequential()

#Input Layer
model.add(Dense(7, input_dim = df1, activation = 'relu'))

#Hidden Layers
model.add(Dense(10, activation = 'relu'))
model.add(Dense(10, activation = 'relu'))
model.add(Dense(10, activation = 'relu'))

#We create 183 Neuron because we have 183 output unnique value in target
model.add(Dense(183, activation = 'softmax'))

# We Have more label in target so we using caregorial_crossentropy
model.compile(loss = 'categorical_crossentropy', optimizer = 'adam', metrics
model.fit(x_train,y_train,epochs = 15, batch_size = 5)

```

```
Epoch 1/15
173/173 [==============================] - 2s 4ms/step - loss: 5.3663 - accurac
y: 0.0128
Epoch 2/15
173/173 [==============================] - 1s 4ms/step - loss: 5.1562 - accurac
y: 0.0093
Epoch 3/15
173/173 [==============================] - 1s 4ms/step - loss: 4.9889 - accurac
y: 0.0128
Epoch 4/15
173/173 [==============================] - 1s 4ms/step - loss: 4.8423 - accurac
y: 0.0244
Epoch 5/15
173/173 [==============================] - 1s 4ms/step - loss: 4.7390 - accurac
y: 0.0290
Epoch 6/15
173/173 [==============================] - 1s 4ms/step - loss: 4.6827 - accurac
y: 0.0278
Epoch 7/15
173/173 [==============================] - 1s 4ms/step - loss: 4.6449 - accurac
y: 0.0313
Epoch 8/15
173/173 [==============================] - 1s 4ms/step - loss: 4.6135 - accurac
y: 0.0348
Epoch 9/15
173/173 [==============================] - 1s 4ms/step - loss: 4.5883 - accurac
y: 0.0302
Epoch 10/15
173/173 [==============================] - 1s 4ms/step - loss: 4.5738 - accurac
y: 0.0452
Epoch 11/15
173/173 [==============================] - 1s 4ms/step - loss: 4.5581 - accurac
y: 0.0336
Epoch 12/15
173/173 [==============================] - 1s 4ms/step - loss: 4.5390 - accurac
y: 0.0360
Epoch 13/15
```

```
173/173 [==============================] - 1s 4ms/step - loss: 4.5199 - accurac
y: 0.0325
Epoch 14/15
173/173 [==============================] - 1s 4ms/step - loss: 4.5155 - accurac
y: 0.0360
Epoch 15/15
173/173 [==============================] - 1s 4ms/step - loss: 4.5047 - accurac
y: 0.0418
12/12 [==============================] - 1s 5ms/step - loss: 5.0801 - accuracy:
0.0108

loss: 5.080

accuracy: 0.011
```

`

Test Score

In [38]:
```
#Checking model Score
scores = model.evaluate(x_test,y_test)


for i, m in enumerate(model.metrics_names):
    print("\n%s: %.3f"% (m, scores[i]))
```

```
12/12 [==============================] - 0s 6ms/step - loss: 5.0801 - accuracy:
0.0108

loss: 5.080

accuracy: 0.011
```

Train Score

In [36]:
```
scoresT = model.evaluate(x_train,y_train)
```

```
27/27 [==============================] - 0s 5ms/step - loss: 4.4499 - accuracy:
0.0464
```

In [37]:
```
for i, m in enumerate(model.metrics_names):
    print("\n%s: %.3f"% (m, scoresT[i]))
```

```
loss: 4.450

accuracy: 0.046
```

In [ ]:
```

```

In [ ]:
```

```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```