22P-9278-MuhammadShafeen

May 29, 2024

0.1 Muhammad Shafeen

0.2 22P-9278

0.3 AI-LAB EXAM FINAL

0.4 Question 1

0.5 Importing Libraries

[ ]:
```
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import Normalizer
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
import pandas as pd
from sklearn.preprocessing import LabelEncoder
import math
from scipy import stats
import numpy as np
import pandas as pd # for dealing with dataframes
from tensorflow.keras import Sequential # for creating a sequential model
from tensorflow.keras.layers import Dense # for creating layers in the model
import matplotlib.pyplot as plt # for plotting
import seaborn as sns # for plotting
from sklearn.preprocessing import StandardScaler, OneHotEncoder # for␣
 ↪preprocessing
from sklearn.impute import SimpleImputer # for preprocessing
from sklearn.compose import ColumnTransformer # for preprocessing
from sklearn.pipeline import Pipeline # for preprocessing
from sklearn.model_selection import train_test_split # for splitting the dataset
from sklearn.neural_network import MLPClassifier
from sklearn.preprocessing import OneHotEncoder
import networkx as nx
import math
import queue
```

[ ]:
```
path="googleplaystore.csv"
df=pd.read_csv(path)
```

[ ]:
```
df
```

[ ]:

	App	Category	Rating	Reviews	Size	Installs	Type	Price	Content Rating	Genres	Last Updated	Current Ver	Android Ver
0	Photo Editor & Candy Camera & Grid & ScrapBook	ART_AND_DESIGN	4.1	159	19M	10,000+	Free	0	Everyone	Art & Design	January 7, 2018	1.0.0	4.0.3 and up
1	Coloring book moana	ART_AND_DESIGN	3.9	967	14M	500,000+	Free	0	Everyone	Art & Design;Pretend Play	January 15, 2018	2.0.0	4.0.3 and up
2	U Launcher Lite - FREE Live Cool Themes, Hide …	ART_AND_DESIGN	4.7	87510	8.7M	5,000,000+	Free	0	Everyone	Art & Design	August 1, 2018	1.2.4	4.0.3 and up
3	Sketch - Draw & Paint	ART_AND_DESIGN	4.5	215644	25M	50,000,000+	Free	0	Teen	Art & Design	June 8, 2018	Varies with device	4.2 and up
4	Pixel Draw - Number Art Coloring Book	ART_AND_DESIGN	4.3	967	2.8M	100,000+	Free	0	Everyone	Art & Design;Creativity	June 20, 2018	1.1	4.4 and up
…	…	…	…	…	…	…	…	…	…	…	…	…	…
10836	Sya9a Maroc - FR	FAMILY	4.5	38	53M	5,000+	Free	0	Everyone	Education	July 25, 2017	1.48	4.1 and up
10837	Fr. Mike Schmitz Audio Teachings	FAMILY	5.0	4	3.6M	100+	Free	0	Everyone	Education	July 6, 2018		
10838	Parkinson Exercices FR	MEDICAL	NaN	3	9.5M	1,000+	Free	0	Everyone	Medical	January 20, 2017		
10839	The SCP Foundation DB fr nn5n	BOOKS_AND_REFERENCE	4.5	114	Varies with device	1,000+	Free	0	Mature 17+	Books & Reference	January 19, 2015		
10840	iHoroscope - 2018 Daily Horoscope & Astrology	LIFESTYLE	4.5	398307	19M	10,000,000+	Free	0	Everyone	Lifestyle	July 25, 2018		

```
10837                1.0          4.1 and up
10838                1.0          2.2 and up
10839  Varies with device  Varies with device
10840  Varies with device  Varies with device

[10841 rows x 13 columns]
```

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

[ ]:
```
App                0
Category           0
Rating          1474
Reviews            0
Size               0
Installs           0
Type               1
Price              0
Content Rating     1
Genres             0
Last Updated       0
Current Ver        8
Android Ver        3
dtype: int64
```

[ ]:
```
df=df.ffill()
```

[ ]:
```
df
```

[ ]:
```
                                                 App             Category  \
0         Photo Editor & Candy Camera & Grid & ScrapBook       ART_AND_DESIGN
1                                    Coloring book moana       ART_AND_DESIGN
2      U Launcher Lite - FREE Live Cool Themes, Hide …       ART_AND_DESIGN
3                                    Sketch - Draw & Paint       ART_AND_DESIGN
4                  Pixel Draw - Number Art Coloring Book       ART_AND_DESIGN
…                                                    …                  …
10836                                  Sya9a Maroc - FR              FAMILY
10837                  Fr. Mike Schmitz Audio Teachings              FAMILY
10838                           Parkinson Exercices FR             MEDICAL
10839                     The SCP Foundation DB fr nn5n  BOOKS_AND_REFERENCE
10840      iHoroscope - 2018 Daily Horoscope & Astrology           LIFESTYLE

       Rating Reviews                Size     Installs  Type Price  \
0         4.1     159                 19M      10,000+  Free     0
1         3.9     967                 14M     500,000+  Free     0
2         4.7   87510                8.7M   5,000,000+  Free     0
3         4.5  215644                 25M  50,000,000+  Free     0
4         4.3     967                2.8M     100,000+  Free     0
```

```
…       …      …                  …           …    …    …
10836    4.5     38                53M       5,000+  Free     0
10837    5.0      4               3.6M         100+  Free     0
10838    5.0      3               9.5M       1,000+  Free     0
10839    4.5    114  Varies with device       1,000+  Free     0
10840    4.5 398307                19M  10,000,000+  Free     0

      Content Rating                     Genres      Last Updated  \
0           Everyone               Art & Design   January 7, 2018
1           Everyone  Art & Design;Pretend Play  January 15, 2018
2           Everyone               Art & Design    August 1, 2018
3               Teen               Art & Design      June 8, 2018
4           Everyone    Art & Design;Creativity     June 20, 2018
…              …                          …              …
10836       Everyone                 Education     July 25, 2017
10837       Everyone                 Education      July 6, 2018
10838       Everyone                   Medical  January 20, 2017
10839     Mature 17+          Books & Reference  January 19, 2015
10840       Everyone                 Lifestyle     July 25, 2018

              Current Ver         Android Ver
0                   1.0.0       4.0.3 and up
1                   2.0.0       4.0.3 and up
2                   1.2.4       4.0.3 and up
3      Varies with device         4.2 and up
4                     1.1         4.4 and up
…                     …               …
10836                1.48         4.1 and up
10837                 1.0         4.1 and up
10838                 1.0         2.2 and up
10839  Varies with device  Varies with device
10840  Varies with device  Varies with device

[10841 rows x 13 columns]
```

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

[ ]:
```
App               0
Category          0
Rating            0
Reviews           0
Size              0
Installs          0
Type              0
Price             0
Content Rating    0
Genres            0
```

```
Last Updated      0
Current Ver       0
Android Ver       0
dtype: int64
```

[ ]:
```
df2=df
```

[ ]:
```
encoder=LabelEncoder()
df2["Category"] = encoder.fit_transform(df["Category"])
df2["Genres"] = encoder.fit_transform(df["Genres"])
df2["Type"] = encoder.fit_transform(df["Type"])
df2["Content Rating"] = encoder.fit_transform(df["Content Rating"])
#

# df2['Installs'] = preprocess_column(df['Installs'])
```

[ ]:
```
df2
```

[ ]:
```
                                                 App  Category  Rating  \
0         Photo Editor & Candy Camera & Grid & ScrapBook         1     4.1
1                                   Coloring book moana         1     3.9
2      U Launcher Lite - FREE Live Cool Themes, Hide …         1     4.7
3                                   Sketch - Draw & Paint         1     4.5
4                  Pixel Draw - Number Art Coloring Book         1     4.3
…                                                    …       …     …
10836                                   Sya9a Maroc - FR        12     4.5
10837                  Fr. Mike Schmitz Audio Teachings        12     5.0
10838                           Parkinson Exercices FR         21     5.0
10839                     The SCP Foundation DB fr nn5n         4     4.5
10840      iHoroscope - 2018 Daily Horoscope & Astrology        19     4.5

      Reviews                 Size     Installs  Type Price  Content Rating  \
0         159                 19M      10,000+     1     0               1
1         967                 14M     500,000+     1     0               1
2       87510                8.7M   5,000,000+     1     0               1
3      215644                 25M  50,000,000+     1     0               4
4         967                2.8M     100,000+     1     0               1
…       …                   …           …   …   …             …
10836      38                 53M       5,000+     1     0               1
10837       4                3.6M         100+     1     0               1
10838       3                9.5M       1,000+     1     0               1
10839     114  Varies with device       1,000+     1     0               3
10840  398307                 19M  10,000,000+     1     0               1

       Genres      Last Updated         Current Ver         Android Ver
0           9   January 7, 2018               1.0.0        4.0.3 and up
1          12  January 15, 2018               2.0.0        4.0.3 and up
```

```
2           9    August 1, 2018               1.2.4        4.0.3 and up
3           9      June 8, 2018  Varies with device          4.2 and up
4          11     June 20, 2018                 1.1          4.4 and up
…         …               …                   …                …
10836      39     July 25, 2017                1.48          4.1 and up
10837      39      July 6, 2018                 1.0          4.1 and up
10838      72   January 20, 2017                1.0          2.2 and up
10839      19   January 19, 2015  Varies with device  Varies with device
10840      68     July 25, 2018  Varies with device  Varies with device

[10841 rows x 13 columns]
```

[ ]:
```
df2=df2.drop(columns=["App","Installs","Price","Last Updated","Current
 ↪Ver","Android Ver","Size"])
```

[ ]:
```
df2
```

[ ]:
```
       Category  Rating Reviews  Type  Content Rating  Genres
0             1     4.1     159     1               1       9
1             1     3.9     967     1               1      12
2             1     4.7   87510     1               1       9
3             1     4.5  215644     1               4       9
4             1     4.3     967     1               1      11
…         …     …     …   …             …      …
10836        12     4.5      38     1               1      39
10837        12     5.0       4     1               1      39
10838        21     5.0       3     1               1      72
10839         4     4.5     114     1               3      19
10840        19     4.5  398307     1               1      68

[10841 rows x 6 columns]
```

[ ]:
```
# scalar=StandardScaler()
```

[ ]:
```
# df2.isnull().count()
# df2=df2.ffill()
```

[ ]:
```
df2
```

[ ]:
```
       Category  Rating Reviews  Type  Content Rating  Genres
0             1     4.1     159     1               1       9
1             1     3.9     967     1               1      12
2             1     4.7   87510     1               1       9
3             1     4.5  215644     1               4       9
4             1     4.3     967     1               1      11
…         …     …     …   …             …      …
10836        12     4.5      38     1               1      39
```

```
10837        12     5.0       4     1               1      39
10838        21     5.0       3     1               1      72
10839         4     4.5     114     1               3      19
10840        19     4.5  398307     1               1      68

[10841 rows x 6 columns]
```

[ ]:
```
df2.dtypes
```

[ ]:
```
Category            int64
Rating            float64
Reviews            object
Type               int64
Content Rating     int64
Genres             int64
dtype: object
```

[ ]:
```
df2
```

[ ]:
```
       Category  Rating Reviews  Type  Content Rating  Genres
0             1     4.1     159     1               1       9
1             1     3.9     967     1               1      12
2             1     4.7   87510     1               1       9
3             1     4.5  215644     1               4       9
4             1     4.3     967     1               1      11
…           …     …       …    …                 …      …
10836        12     4.5      38     1               1      39
10837        12     5.0       4     1               1      39
10838        21     5.0       3     1               1      72
10839         4     4.5     114     1               3      19
10840        19     4.5  398307     1               1      68

[10841 rows x 6 columns]
```

0.5.1 Using help from lab material

[ ]:
```
#
data = df2[['Category', 'Rating', 'Reviews', 'Type', 'Content␣
 ↪Rating','Genres']].copy()

data
```

[ ]:
```
       Category  Rating Reviews  Type  Content Rating  Genres
0             1     4.1     159     1               1       9
1             1     3.9     967     1               1      12
2             1     4.7   87510     1               1       9
3             1     4.5  215644     1               4       9
```

```
4              1     4.3     967     1                1      11
…          …      …      …      …                  …      …
10836         12     4.5      38     1                1      39
10837         12     5.0       4     1                1      39
10838         21     5.0       3     1                1      72
10839          4     4.5     114     1                3      19
10840         19     4.5  398307     1                1      68

[10841 rows x 6 columns]
```

[ ]:
```
data.dtypes
```

[ ]:
```
Category            int64
Rating            float64
Reviews            object
Type                int64
Content Rating      int64
Genres              int64
dtype: object
```

[ ]:
```
# df["Category"]=df["Category"].map({'male':1,'female':0}) #hot encoding ,␣
 ↪changing data into numerical
# df["Embarked"]=df["Embarked"].map({'S':2,'c':3,'Q':4}) #hot encoding ,␣
 ↪changing data into numerical
# df["Embarked"]=df["Embarked"].ffill()
# X_train_X=df[["Pclass","Age","SibSp","Parch","Fare"]]
# y_test_y=df["Survived"]
# X_train_X=X_train_X.ffill()
# y_test_y=y_test_y.ffill()
```

[ ]:
```
def extract_room_info(room_count):
    room = 0
    living_room = 0
    parts = str(room_count).split('M')
    if len(parts) > 0:
        if parts[0].isdigit():
            room = int(parts[0])
        if len(parts) > 1 and parts[1].isdigit():
            living_room = int(parts[1])
    return room, living_room
```

[ ]:
```
data['Reviews'], data['living_room'] = zip(*data['Reviews'].
 ↪apply(extract_room_info))
data.drop(['living_room'], axis=1,inplace=True)
data['Rating'], data['living_room'] = zip(*data['Rating'].
 ↪apply(extract_room_info))
data.drop(['living_room'], axis=1,inplace=True)
```

[ ]:
```
data.dtypes
```

[ ]:
```
Category          int64
Rating            int64
Reviews           int64
Type              int64
Content Rating    int64
Genres            int64
dtype: object
```

[ ]:
```
data
```

[ ]:
```
       Category  Rating  Reviews  Type  Content Rating  Genres
0             1       0      159     1               1       9
1             1       0      967     1               1      12
2             1       0    87510     1               1       9
3             1       0   215644     1               4       9
4             1       0      967     1               1      11
…           …       …        …   …               …       …
10836        12       0       38     1               1      39
10837        12       0        4     1               1      39
10838        21       0        3     1               1      72
10839         4       0      114     1               3      19
10840        19       0   398307     1               1      68

[10841 rows x 6 columns]
```

[ ]:
```
data.to_csv("newfile")
```

[ ]:
```
data.isnull().sum()
```

[ ]:
```
Category          0
Rating            0
Reviews           0
Type              0
Content Rating    0
Genres            0
dtype: int64
```

[ ]:
```
# df["Reviews"]=pd.to_numeric(df["Reviews"])
```

[ ]:
```
data.dtypes
```

[ ]:
```
Category          int64
Rating            int64
Reviews           int64
Type              int64
Content Rating    int64
```

```
Genres            int64
dtype: object
```

[ ]:
```
# Define preprocessing steps
numeric_features = [0, 1]  # Indices of numerical columns in data
numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),  # Fill missing values with␣
 ↪median
    ('scaler', StandardScaler())  # Scale data
])

categorical_features = [2, 3]  # Indices of categorical columns in data
categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='constant', fill_value='missing')),  #␣
 ↪Fill missing values with 'missing'
    ('onehot', OneHotEncoder())  # One-hot encode categorical variables
])

preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numeric_features),
        ('cat', categorical_transformer, categorical_features)
    ])
```

[ ]:
```
X=data[["Rating","Reviews","Type","Content Rating","Genres"]]
y=df["Category"]
```

[ ]:
```
X2=data[["Rating","Reviews","Type","Content Rating","Genres"]]
y2=df["Category"]
```

1 WITH KNN

1.0.1 Splitting data testing and training

[ ]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X2, y2, test_size=0.
 ↪2,random_state=0)
```

[ ]:
```
X_train, X_test, y_train, y_test = train_test_split(X2, y2, test_size=0.2,␣
 ↪random_state=42)
```

[ ]:
```
knn=KNeighborsClassifier(n_neighbors=3)
```

[ ]:
```
knn.fit(X_train,y_train)
```

[ ]:
```
KNeighborsClassifier(n_neighbors=3)
```

[ ]:
```
y_predict=knn.predict(X_test)
```

[ ]:
```
y_predict
```

[ ]:
```
array([15, 10, 16, …, 22, 26,  8])
```

[ ]:
```
accuracy = accuracy_score(y_test, y_predict)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.4310742277547257
```

[ ]:
```

```

2 WITH ANN

[ ]:
```
X.shape
```

[ ]:
```
(10841, 5)
```

[ ]:
```
y.shape
```

[ ]:
```
(10841,)
```

[ ]:
```
X_train.shape
```

[ ]:
```
(8672, 5)
```

[ ]:
```
y_train.shape
```

[ ]:
```
(8672,)
```

[ ]:
```
models = [
    Sequential([ # Model 1 with 1 hidden layers
        Dense(32, activation='relu', input_shape=(5,)),
        Dense(32, activation='relu'),
        Dense(1, activation='sigmoid')
    ],name="Recommendation_Model_1"),
    Sequential([ # Model 2 with 2 hidden layers
        Dense(64, activation='relu', input_shape=(5,)),
        Dense(100, activation='relu'),
        Dense(100, activation='relu'),
        Dense(1, activation='sigmoid')
    ],name="Recommendation_Model_2"),
    Sequential([ # Model 3 with many hidden layers and more neurons
        Dense(10, activation='relu', input_shape=(5,)),
        Dense(20, activation='relu'),
        Dense(30, activation='relu'),
        Dense(40, activation='relu'),
```

```
        Dense(50, activation='relu'),
        Dense(60, activation='relu'),
        Dense(70, activation='relu'),
        Dense(80, activation='relu'),
        Dense(90, activation='relu'),
        Dense(1, activation='sigmoid')
    ],name="Recommendation_Model_3"),
    Sequential([ # Model 4 with one hidden layer and many units
        Dense(100, activation='relu', input_shape=(5,)),
        Dense(1, activation='sigmoid')
    ],name="Recommendation_Model_4"),

]

# Compiling and  training models
histories = []
for model in models:
    model.compile(optimizer='adam', loss='binary_crossentropy',␣
 ↪metrics=['accuracy'])
    print(model.layers)
    history = model.fit(X_train, y_train, epochs=10, batch_size=32)
    histories.append(history)
```

```
[<keras.layers.core.dense.Dense object at 0x727eed61c940>,
<keras.layers.core.dense.Dense object at 0x727eed61c910>,
<keras.layers.core.dense.Dense object at 0x727eed61ce50>]
Epoch 1/10
271/271 [==============================] - 0s 469us/step - loss: -30683582.0000
- accuracy: 0.0052
Epoch 2/10
271/271 [==============================] - 0s 496us/step - loss: -299660960.0000
- accuracy: 0.0055
Epoch 3/10
271/271 [==============================] - 0s 650us/step - loss:
-1224982144.0000 - accuracy: 0.0055
Epoch 4/10
271/271 [==============================] - 0s 473us/step - loss:
-3301970432.0000 - accuracy: 0.0055
Epoch 5/10
271/271 [==============================] - 0s 543us/step - loss:
-6961060864.0000 - accuracy: 0.0055
Epoch 6/10
271/271 [==============================] - 0s 492us/step - loss:
-12209737728.0000 - accuracy: 0.0055
Epoch 7/10
271/271 [==============================] - 0s 492us/step - loss:
-19124813824.0000 - accuracy: 0.0055
Epoch 8/10
```

```
271/271 [==============================] - 0s 500us/step - loss:
-28160157696.0000 - accuracy: 0.0055
Epoch 9/10
271/271 [==============================] - 0s 487us/step - loss:
-39455440896.0000 - accuracy: 0.0055
Epoch 10/10
271/271 [==============================] - 0s 520us/step - loss:
-52923355136.0000 - accuracy: 0.0055
[<keras.layers.core.dense.Dense object at 0x727eef167eb0>,
<keras.layers.core.dense.Dense object at 0x727eeca099d0>,
<keras.layers.core.dense.Dense object at 0x727eecb26d30>,
<keras.layers.core.dense.Dense object at 0x727eed612fa0>]
Epoch 1/10
271/271 [==============================] - 0s 642us/step - loss:
-2393231104.0000 - accuracy: 0.0055
Epoch 2/10
271/271 [==============================] - 0s 657us/step - loss:
-117416288256.0000 - accuracy: 0.0055
Epoch 3/10
271/271 [==============================] - 0s 649us/step - loss:
-762796310528.0000 - accuracy: 0.0055
Epoch 4/10
271/271 [==============================] - 0s 709us/step - loss:
-2380052824064.0000 - accuracy: 0.0055
Epoch 5/10
271/271 [==============================] - 0s 717us/step - loss:
-5815969251328.0000 - accuracy: 0.0055
Epoch 6/10
271/271 [==============================] - 0s 793us/step - loss:
-12510161797120.0000 - accuracy: 0.0055
Epoch 7/10
271/271 [==============================] - 0s 777us/step - loss:
-23252053262336.0000 - accuracy: 0.0055
Epoch 8/10
271/271 [==============================] - 0s 805us/step - loss:
-39101558423552.0000 - accuracy: 0.0055
Epoch 9/10
271/271 [==============================] - 0s 678us/step - loss:
-60803990421504.0000 - accuracy: 0.0055
Epoch 10/10
271/271 [==============================] - 0s 684us/step - loss:
-88534383853568.0000 - accuracy: 0.0055
[<keras.layers.core.dense.Dense object at 0x727eed631160>,
<keras.layers.core.dense.Dense object at 0x727eed612d30>,
<keras.layers.core.dense.Dense object at 0x727eee3a4940>,
<keras.layers.core.dense.Dense object at 0x727eee3a4c40>,
<keras.layers.core.dense.Dense object at 0x727eee3a4f40>,
<keras.layers.core.dense.Dense object at 0x727eee39e280>,
```

```
<keras.layers.core.dense.Dense object at 0x727eee39e580>,
<keras.layers.core.dense.Dense object at 0x727eee39e880>,
<keras.layers.core.dense.Dense object at 0x727eee39eb80>,
<keras.layers.core.dense.Dense object at 0x727eee3a48b0>]
Epoch 1/10
271/271 [==============================] - 1s 825us/step - loss:
-129117705994240.0000 - accuracy: 0.0055
Epoch 2/10
271/271 [==============================] - 0s 792us/step - loss:
-1170646181802409984.0000 - accuracy: 0.0055
Epoch 3/10
271/271 [==============================] - 0s 816us/step - loss: nan - accuracy:
0.0013
Epoch 4/10
271/271 [==============================] - 0s 902us/step - loss: nan - accuracy:
1.1531e-04
Epoch 5/10
271/271 [==============================] - 0s 823us/step - loss: nan - accuracy:
1.1531e-04
Epoch 6/10
271/271 [==============================] - 0s 780us/step - loss: nan - accuracy:
1.1531e-04
Epoch 7/10
271/271 [==============================] - 0s 771us/step - loss: nan - accuracy:
1.1531e-04
Epoch 8/10
271/271 [==============================] - 0s 815us/step - loss: nan - accuracy:
1.1531e-04
Epoch 9/10
271/271 [==============================] - 0s 824us/step - loss: nan - accuracy:
1.1531e-04
Epoch 10/10
271/271 [==============================] - 0s 810us/step - loss: nan - accuracy:
1.1531e-04
[<keras.layers.core.dense.Dense object at 0x727eee3ad310>,
<keras.layers.core.dense.Dense object at 0x727eee2eae80>]
Epoch 1/10
271/271 [==============================] - 0s 495us/step - loss: -16439816.0000
- accuracy: 0.0055
Epoch 2/10
271/271 [==============================] - 0s 468us/step - loss: -83469072.0000
- accuracy: 0.0055
Epoch 3/10
271/271 [==============================] - 0s 465us/step - loss: -196996576.0000
- accuracy: 0.0055
Epoch 4/10
271/271 [==============================] - 0s 467us/step - loss: -371101760.0000
- accuracy: 0.0055
```

```
Epoch 5/10
271/271 [==============================] - 0s 480us/step - loss: -612741696.0000
- accuracy: 0.0055
Epoch 6/10
271/271 [==============================] - 0s 480us/step - loss: -912806720.0000
- accuracy: 0.0055
Epoch 7/10
271/271 [==============================] - 0s 460us/step - loss:
-1250795776.0000 - accuracy: 0.0055
Epoch 8/10
271/271 [==============================] - 0s 618us/step - loss:
-1625849344.0000 - accuracy: 0.0055
Epoch 9/10
271/271 [==============================] - 0s 460us/step - loss:
-2051796224.0000 - accuracy: 0.0055
Epoch 10/10
271/271 [==============================] - 0s 452us/step - loss:
-2515277312.0000 - accuracy: 0.0055
```

[ ]:
```
histories
```

[ ]:
```
[<keras.callbacks.History at 0x727eee2c3af0>,
 <keras.callbacks.History at 0x727eee29afa0>,
 <keras.callbacks.History at 0x727eee197ac0>,
 <keras.callbacks.History at 0x727eed649fd0>]
```

[ ]:
```
# # Extracting accuracies
# models = [
#     MLPClassifier(hidden_layer_sizes=(32,), activation='relu', max_iter=100),
#     MLPClassifier(hidden_layer_sizes=(64, 100, 100), activation='relu',
 ↪max_iter=100),
#     MLPClassifier(hidden_layer_sizes=(10, 20, 30, 40, 50, 60, 70, 80, 90),
 ↪activation='relu', max_iter=100),
#     MLPClassifier(hidden_layer_sizes=(100,), activation='relu', max_iter=100)
# ]
# plt.figure(figsize=(10, 6))

# # Define colors for bars
# colors = ['skyblue', 'lightgreen', 'lightcoral', 'lightseagreen']

# for i, history in enumerate(histories):
#     val_accuracy = history.history['val_accuracy'][-1]
#     plt.bar(i, val_accuracy, color=colors[i])

# plt.xticks(np.arange(len(models)), ['Model 1', 'Model 2', 'Model 3', 'Model
 ↪4'])
# plt.xlabel('Different Model Architecture')
```

```
# plt.ylabel('Accuracy')
# plt.title('Comparison of Different Neural Network Architectures')
# plt.ylim([0, 1])
# plt.show()
```

3 Question 2

[ ]:
```
data
```

[ ]:

	Category	Rating	Reviews	Type	Content Rating	Genres
0	1	0	159	1	1	9
1	1	0	967	1	1	12
2	1	0	87510	1	1	9
3	1	0	215644	1	4	9
4	1	0	967	1	1	11
…	…	…	…	…	…	…
10836	12	0	38	1	1	39
10837	12	0	4	1	1	39
10838	21	0	3	1	1	72
10839	4	0	114	1	3	19
10840	19	0	398307	1	1	68

[10841 rows x 6 columns]

[ ]:
```
twoDarray=np.array([[1,2,3],[4,0,5],[7,8,6]])
```

[ ]:
```
G = nx.Graph()
G.add_weighted_edges_from([('S', 'A', 1), ('S', 'G', 10), ('A', 'C', 1), ('A',␣
↪'B', 2), ('B', 'D', 5),('C', 'G', 4),('D', 'G', 2)])
```

3.1 Manhattan Huristic Formula

[ ]:
```
distance = 0
goal_position={{"0":(0,0),"1":(0,1),"2":(1,0)}}
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[1991], line 2
      1 distance = 0
----> 2 goal_position={{"0":(0,0),"1":(0,1),"2":(1,0)}}

TypeError: unhashable type: 'dict'
```

[ ]:
```
def astar(graph, start, goal, heuristic):
    visited = set()
```

```
    pri_queue = queue.PriorityQueue()  # Priority queue
    pri_queue.put((0 + heuristic[start], [start]))  # Initial state: f = g + h␣
 ↪= 0 + heuristic

    while not pri_queue.empty():
        f, current_path = pri_queue.get()
        current_node = current_path[-1]

        if current_node == goal:
            return current_path  # Goal found

        visited.add(current_node)

        for neighbor in graph.neighbors(current_node):
            if neighbor not in visited:
                g = graph[current_node][neighbor]['weight']  # Cost from start␣
 ↪to current node
                new_path = current_path + [neighbor]
                pri_queue.put((g + heuristic[neighbor], new_path))

    return []  # Goal not found
```

[ ]:
```
start_node = 'S'
goal_node = 'G'

# Define positions for the nodes (for Euclidean distance calculation)
pos = nx.spring_layout(G)

# Heuristic function using Euclidean distance
heuristic = {node: euclidean_distance(node, goal_node, pos) for node in G.nodes}

path = astar(G, start_node, goal_node, heuristic)
if path:
    print("Path from {} to {} found: {}".format(start_node, goal_node, ' -> '.
 ↪join(path)))
else:
    print("No path found from {} to {}".format(start_node, goal_node))



nx.draw(G, pos, with_labels=True, node_color='skyblue', node_size=1500,␣
 ↪edge_color='k', linewidths=1, font_size=15)
edge_labels = nx.get_edge_attributes(G, 'weight')
nx.draw_networkx_edge_labels(G, pos, edge_labels=edge_labels)
```