```python
import pandas as pd
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler, LabelEncoder
import numpy as np
import matplotlib.pyplot as plt


df = pd.read_csv("googleplaystore.csv")
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     NaN       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            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.drop_duplicates(inplace=True)
```

```
df.dropna
```

```
<bound method DataFrame.dropna of
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     NaN       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

[10358 rows x 13 columns]>
```

```
df.replace("-", np.nan, inplace=True)



# Encode the categorical column 'App'
encoder = LabelEncoder()
gender_encoded = encoder.fit_transform(df["App"])
df["App"] = gender_encoded

# Encode the categorical column 'Category'
encoder = LabelEncoder()
gender_encoded = encoder.fit_transform(df["Category"])
df["Category"] = gender_encoded

# Encode the categorical column 'Rating'
encoder = LabelEncoder()
gender_encoded = encoder.fit_transform(df["Rating"])
df["Rating"] = gender_encoded

# Encode the categorical column 'Reviews'
encoder = LabelEncoder()
gender_encoded = encoder.fit_transform(df["Reviews"])
df["Reviews"] = gender_encoded


# Encode the categorical column 'Size'
encoder = LabelEncoder()
gender_encoded = encoder.fit_transform(df["Size"])
df["Size"] = gender_encoded


# Encode the categorical column 'Installs'
encoder = LabelEncoder()
gender_encoded = encoder.fit_transform(df["Installs"])
df["Installs"] = gender_encoded


# Encode the categorical column 'Type'
encoder = LabelEncoder()
```

```
gender_encoded = encoder.fit_transform(df["Type"])
df["Type"] = gender_encoded


# Encode the categorical column 'Genres'
encoder = LabelEncoder()
gender_encoded = encoder.fit_transform(df["Genres"])
df["Genres"] = gender_encoded


# Encode the categorical column 'Price'
encoder = LabelEncoder()
gender_encoded = encoder.fit_transform(df["Price"])
df["Price"] = gender_encoded


# Encode the categorical column 'Content Rating'
encoder = LabelEncoder()
gender_encoded = encoder.fit_transform(df["Content Rating"])
df["Content Rating"] = gender_encoded


# Encode the categorical column 'Last Updated'
encoder = LabelEncoder()
gender_encoded = encoder.fit_transform(df["Last Updated"])
df["Last Updated"] = gender_encoded


# Encode the categorical column 'Current Ver '
encoder = LabelEncoder()
gender_encoded = encoder.fit_transform(df["Current Ver"])
df["Current Ver"] = gender_encoded


# Encode the categorical column 'Android Ver '
encoder = LabelEncoder()
gender_encoded = encoder.fit_transform(df["Android Ver"])
df["Android Ver"] = gender_encoded

imputer = SimpleImputer(strategy="mean")
df = pd.DataFrame(imputer.fit_transform(df), columns=df.columns)


scaler = StandardScaler()
scaler.fit(df)
normalized_data = scaler.transform(df)

Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 - Q1
IQR
```

```
App                4810.50
Category             14.00
Rating                5.00
Reviews            3183.50
Size                259.75
Installs              7.00
Type                  0.00
Price                 0.00
Content Rating        0.00
Genres               59.00
Last Updated        486.75
Current Ver        1721.50
Android Ver           7.00
dtype: float64
```

```
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
```

```
data_clean = df[(df >= lower_bound) & (df <= upper_bound)]
data_clean
```

```
          App  Category  Rating  Reviews   Size  Installs  Type  Price
\
0      6963.0       1.0    29.0   1182.0   54.0       7.0   1.0   91.0

1      2632.0       1.0    27.0   5923.0   28.0      19.0   1.0   91.0

2      8657.0       1.0    35.0   5680.0  367.0      14.0   1.0   91.0

3      7828.0       1.0    33.0   1946.0  100.0      17.0   1.0   91.0

4      7023.0       1.0    31.0   5923.0   63.0      10.0   1.0   91.0

...       ...       ...     ...      ...    ...       ...   ...    ...

10353  8174.0      12.0    33.0   3471.0  239.0      13.0   1.0   91.0

10354  4609.0      12.0    38.0   3588.0  124.0       9.0   1.0   91.0

10355  6892.0      21.0    40.0   2854.0  413.0       3.0   1.0   91.0

10356  8395.0       4.0    33.0    355.0  461.0       3.0   1.0   91.0

10357  9487.0      19.0    33.0   3579.0   54.0       8.0   1.0   91.0


       Content Rating  Genres  Last Updated  Current Ver  Android Ver

0                 1.0     9.0         561.0        118.0         15.0

1                 1.0    12.0         481.0       1018.0         15.0
```

```
2                 1.0     9.0         116.0         464.0         15.0
3                 NaN     9.0         824.0        2765.0         18.0
4                 1.0    11.0         756.0         277.0         20.0
...               ...     ...           ...           ...          ...
10353             1.0    39.0         645.0         638.0         17.0
10354             1.0    39.0         691.0         113.0         17.0
10355             1.0    72.0         505.0         113.0          7.0
10356             NaN    19.0         496.0        2765.0          NaN
10357             1.0    68.0         646.0        2765.0          NaN

[10358 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
```

```
binary_df = df[df["Genres"] != 2]
```

```
X = binary_df.drop("Genres", axis=1)
y = binary_df["Genres"]
```

```
from sklearn.linear_model import Perceptron
```

```
ann = Perceptron(eta0=0.1, max_iter=500)
```

```
ann.fit(X, y)
```

```
Perceptron(eta0=0.1, max_iter=500)
```

```
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=42
)

from sklearn.linear_model import Perceptron

perceptron_model = Perceptron()
perceptron_model.fit(X_train, y_train)
```

```
Perceptron()
```

```
target = df["Genres"]
train_data, test_data, train_target, test_target = train_test_split(
    df, target, test_size=0.2, random_state=42
)
print(train_data.shape, test_data.shape, train_target.shape,
test_target.shape)
```

```
(8286, 13) (2072, 13) (8286,) (2072,)
```

```
X_train_scaled = scaler.fit_transform(train_data)
X_test_scaled = scaler.transform(test_data)

from sklearn.neural_network import MLPClassifier  # For scikit-learn's
MLP classifier

mlp = MLPClassifier(
    hidden_layer_sizes=(50, 50),
    activation="relu",
    solver="adam",
    alpha=0.001,
    batch_size=100,
    max_iter=1000,
)
mlp.fit(X_train_scaled, train_target)
```

```
MLPClassifier(alpha=0.001, batch_size=100, hidden_layer_sizes=(50,
50),
              max_iter=1000)
```

```
from keras.models import Sequential
from keras.layers import Dense

model = Sequential()
model.add(Dense(2, input_dim=3, activation="sigmoid"))
model.add(Dense(2, activation="sigmoid"))
```

```
C:\Users\HP\AppData\Roaming\Python\Python312\site-packages\keras\src\
layers\core\dense.py:87: UserWarning: Do not pass an
```

```
`input_shape`/`input_dim` argument to a layer. When using Sequential
models, prefer using an `Input(shape)` object as the first layer in
the model instead.
  super().__init__(activity_regularizer=activity_regularizer,
**kwargs)
```

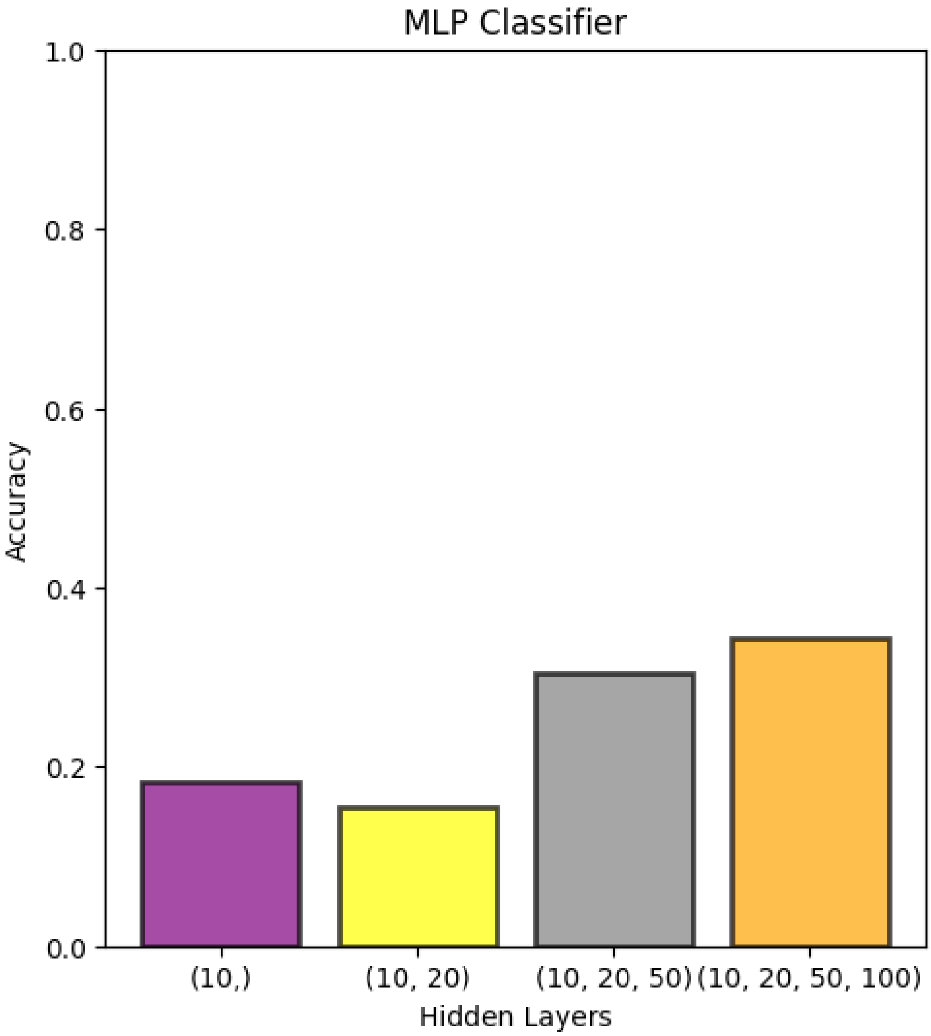
```python
model.compile(loss="mean_squared_error", optimizer="sgd",
metrics=["accuracy"])

models_MLP = [
    MLPClassifier(hidden_layer_sizes=(10,), max_iter=100),
    MLPClassifier(hidden_layer_sizes=(10, 20), max_iter=100),
    MLPClassifier(hidden_layer_sizes=(10, 20, 50), max_iter=100),
    MLPClassifier(hidden_layer_sizes=(10, 20, 50, 100), max_iter=100),
]


Models_Keras = [
    Sequential(
        [Dense(10, input_dim=7, activation="relu"), Dense(1,
activation="sigmoid")]
    ),
    Sequential(
        [
            Dense(10, input_dim=7, activation="relu"),
            Dense(20, activation="relu"),
            Dense(1, activation="sigmoid"),
        ]
    ),
    Sequential(
        [
            Dense(10, input_dim=7, activation="relu"),
            Dense(20, activation="relu"),
            Dense(50, activation="relu"),
            Dense(1, activation="sigmoid"),
        ]
    ),
    Sequential(
        [
            Dense(10, input_dim=7, activation="relu"),
            Dense(20, activation="relu"),
            Dense(50, activation="relu"),
            Dense(100, activation="relu"),
            Dense(1, activation="sigmoid"),
        ]
    ),
]

from sklearn.metrics import accuracy_score  # For calculating accuracy
```

```python
accuracy_mlp = []

for model in models_MLP:
    model.fit(train_data, train_target)

    y_pred = model.predict(test_data)

    accuracy_mlp.append(accuracy_score(test_target, y_pred))

    print(accuracy_score(test_target, y_pred))
```

```
C:\Users\HP\AppData\Roaming\Python\Python312\site-packages\sklearn\
neural_network\_multilayer_perceptron.py:691: ConvergenceWarning:
Stochastic Optimizer: Maximum iterations (100) reached and the
optimization hasn't converged yet.
  warnings.warn(

0.1829150579150579

C:\Users\HP\AppData\Roaming\Python\Python312\site-packages\sklearn\
neural_network\_multilayer_perceptron.py:691: ConvergenceWarning:
Stochastic Optimizer: Maximum iterations (100) reached and the
optimization hasn't converged yet.
  warnings.warn(

0.15444015444015444

C:\Users\HP\AppData\Roaming\Python\Python312\site-packages\sklearn\
neural_network\_multilayer_perceptron.py:691: ConvergenceWarning:
Stochastic Optimizer: Maximum iterations (100) reached and the
optimization hasn't converged yet.
  warnings.warn(

0.3055019305019305
0.34314671814671815

C:\Users\HP\AppData\Roaming\Python\Python312\site-packages\sklearn\
neural_network\_multilayer_perceptron.py:691: ConvergenceWarning:
Stochastic Optimizer: Maximum iterations (100) reached and the
optimization hasn't converged yet.
  warnings.warn(
```

```python
plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)
plt.bar(
    range(4),
    accuracy_mlp,
    color=["purple", "yellow", "gray", "orange"],
    alpha=0.7,
    edgecolor="black",
    linewidth=2,
)
plt.xticks(range(4), ["(10,)", "(10, 20)", "(10, 20, 50)", "(10, 20, 50, 100)"])
plt.title("MLP Classifier")
plt.xlabel("Hidden Layers")
plt.ylabel("Accuracy")
plt.ylim([0, 1])
```

```
(0.0, 1.0)
```



QUESTION 2

```
import queue


array = [[1,2,3],[4,5,6],[7,8,0]]
```

```
def calculate_manhattan(self):
        manhattan = 0
        for i in range(3):
            for j in range(3):
                if self.state[i][j] != 0:
                    x, y = divmod(self.state[i][j] - 1, 3)
                    manhattan += abs(x - i) + abs(y - j)
        return manhattan




def astar(graph, start, goal, heuristic):
    visited = set()
    pri_queue = queue.PriorityQueue()
    pri_queue.put((0 + heuristic[start], [start]))

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

        if current_node == goal:
            return current_path

        visited.add(current_node)

        for neighbor in graph.neighbors(current_node):
            if neighbor not in visited:
                g = graph[current_node][neighbor]['weight']
                new_path = current_path + [neighbor]
                pri_queue.put((g + heuristic[neighbor], new_path))

    return []
```