G. H. Raisoni College Of Engineering And Management, Wagholi Pune

2021- 2022

Group C :-Assignment no :-19

Department	CE [SUMMER 2022 (Online)]		
Term / Section	III/B	Date Of **submission**	13-12-2021
Subject Name /Code	Python for Data Science / UCSP204		
Roll No.	SCOB77	Name	Pratham Rajkumar pitty
Registration Number	2020AC0E1100107		

Group C : Assignment No 12

Aim:

Using Naives Bayes, Decision Tree, KNN predict if a customer with certain age and salary will purchase a product or not

Theory:-

- **Naives Bayes:-**

It is a simple technique for constructing classifier models that assign class labels to problem instances, represented as vectors of feature values

- **Decision tree :-**

It is a flowchart - like structure in which each internal node represents a "test" on attribute

- **KNN:**

It stands K-Nearest Neighbour. It is one of the simplest machine learning algorithm based on supervised learning

Colclusion :-

Hence we conclude that using Naive Bayes, Decision Tree, KNN predict if a customer with certain age and salary will purchase a product or not.

INFOSYS STOCK MARKET PRICE PREDICTION USING LSTM(Long short-term memory)

FROM [01-07-2015 t0 01-07-2020(5 Years)]

DATASET FROM NSE INDIA(National Stock Exchange of India Ltd)

In [14]:

```
print("****************************************")
print("SCOB77_Pratham pitty_Group c Assignment 19")
print("****************************************")
```

```
****************************************
SCOB77_Pratham pitty_Group c Assignment 19
****************************************
```

In [2]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from keras.models import Sequential
from keras.layers import Dense, LSTM, Dropout
```

In [4]:

```
INFOSYS = pd.read_csv("/content/INFY.NS.csv")
```

In [5]:

```
INFOSYS.head(10)
```

Out[5]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2015-07-01	494.500000	502.500000	493.000000	498.700012	415.561249	6880852.0
1	2015-07-02	499.500000	500.700012	492.524994	494.000000	411.644745	4007568.0
2	2015-07-03	494.000000	496.500000	491.000000	495.149994	412.603058	2695306.0
3	2015-07-06	492.500000	494.000000	487.500000	491.649994	409.686493	4305602.0
4	2015-07-07	492.500000	495.000000	489.500000	490.250000	408.519897	3497418.0
5	2015-07-08	489.500000	489.500000	477.399994	478.750000	398.937103	7024178.0
6	2015-07-09	478.750000	483.000000	467.024994	469.000000	390.812531	8587772.0
7	2015-07-10	474.049988	476.049988	466.325012	468.750000	390.604187	7411522.0
8	2015-07-13	470.500000	477.500000	467.024994	475.075012	395.874756	6531768.0
9	2015-07-14	477.500000	486.750000	472.750000	485.625000	404.665955	5367074.0

In [6]:

```
INFOSYS.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1228 entries, 0 to 1227
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       1228 non-null   object 
 1   Open       1227 non-null   float64
 2   High       1227 non-null   float64
 3   Low        1227 non-null   float64
 4   Close      1227 non-null   float64
 5   Adj Close  1227 non-null   float64
 6   Volume     1227 non-null   float64
dtypes: float64(6), object(1)
memory usage: 67.3+ KB
```

In [9]:

```
INFOSYS_NEW=INFOSYS.dropna(axis=0)
INFOSYS_NEW
```

Out[9]:

	Date	Open	High	Low	Close	Adj Close	Volume
0	2015-07-01	494.500000	502.500000	493.000000	498.700012	415.561249	6880852.0
1	2015-07-02	499.500000	500.700012	492.524994	494.000000	411.644745	4007568.0
2	2015-07-03	494.000000	496.500000	491.000000	495.149994	412.603058	2695306.0
3	2015-07-06	492.500000	494.000000	487.500000	491.649994	409.686493	4305602.0
4	2015-07-07	492.500000	495.000000	489.500000	490.250000	408.519897	3497418.0
...	...	...	...	...	...	...	...
1223	2020-06-23	695.700012	724.500000	692.099976	720.650024	720.650024	12157390.0
1224	2020-06-24	716.900024	726.799988	709.400024	714.150024	714.150024	10220838.0
1225	2020-06-25	704.849976	705.599976	693.500000	700.500000	700.500000	14945284.0
1226	2020-06-26	710.200012	751.599976	710.000000	748.200012	748.200012	26030458.0
1227	2020-06-29	735.400024	744.700012	730.099976	731.750000	731.750000	8954428.0

1227 rows × 7 columns

In [11]:

```
INFOSYS_NEW.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1227 entries, 0 to 1227
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       1227 non-null   object 
 1   Open       1227 non-null   float64
 2   High       1227 non-null   float64
 3   Low        1227 non-null   float64
 4   Close      1227 non-null   float64
 5   Adj Close  1227 non-null   float64
 6   Volume     1227 non-null   float64
dtypes: float64(6), object(1)
memory usage: 76.7+ KB
```

In [52]:

```
INFOSYS_NEW['Close']=pd.to_numeric(INFOSYS_NEW.Close,errors='coerce')
INFOSYS_NEW = INFOSYS_NEW.dropna()
INFOSYS_TRAIN = INFOSYS_NEW.iloc[:,4:5].values
```

In [37]:

```
INFOSYS_NEW.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1227 entries, 0 to 1227
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   Date       1227 non-null   object 
 1   Open       1227 non-null   float64
 2   High       1227 non-null   float64
 3   Low        1227 non-null   float64
 4   Close      1227 non-null   float64
 5   Adj Close  1227 non-null   float64
 6   Volume     1227 non-null   float64
dtypes: float64(6), object(1)
memory usage: 76.7+ KB
```

In [53]:

```
sc = MinMaxScaler(feature_range=(0,1))
INFOSYS_TRAIN = sc.fit_transform(INFOSYS_TRAIN)
INFOSYS_TRAIN.shape
```

Out[53]:

```
(1227, 1)
```

In [54]:

```
x_train = []
y_train = []

for i in range (60,1227):
  x_train.append(INFOSYS_TRAIN[i-60:i,0])
  y_train.append(INFOSYS_TRAIN[i,0])

x_train, y_train = np.array(x_train), np.array(y_train)
```

In [55]:

```
x_train = np.reshape(x_train,(x_train.shape[0],x_train.shape[1],1))
x_train.shape
```

Out[55]:

```
(1167, 60, 1)
```

In [58]:

```
model = Sequential()

model.add(LSTM(units=100, return_sequences= True, input_shape=(x_train.shape[1],1)))
model.add(Dropout(0.2))

model.add(LSTM(units=100, return_sequences= True))
model.add(Dropout(0.2))


model.add(LSTM(units=100, return_sequences= True))
model.add(Dropout(0.2))


model.add(LSTM(units=100, return_sequences= False))
model.add(Dropout(0.2))


model.add(Dense(units=1))
model.compile(optimizer='adam', loss='mean_squared_error')
```

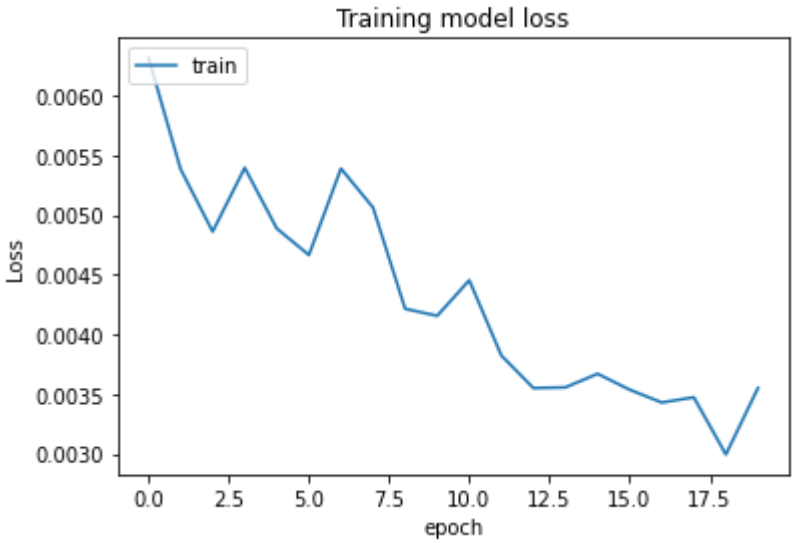
In [60]:

```
hist = model.fit(x_train, y_train, epochs=20, batch_size=32, verbose=2)
```

```
Epoch 1/20
 - 9s - loss: 0.0063
Epoch 2/20
 - 9s - loss: 0.0054
Epoch 3/20
 - 9s - loss: 0.0049
Epoch 4/20
 - 9s - loss: 0.0054
Epoch 5/20
 - 9s - loss: 0.0049
Epoch 6/20
 - 9s - loss: 0.0047
Epoch 7/20
 - 8s - loss: 0.0054
Epoch 8/20
 - 9s - loss: 0.0051
Epoch 9/20
 - 9s - loss: 0.0042
Epoch 10/20
 - 8s - loss: 0.0042
Epoch 11/20
 - 9s - loss: 0.0045
Epoch 12/20
 - 9s - loss: 0.0038
Epoch 13/20
 - 9s - loss: 0.0036
Epoch 14/20
 - 9s - loss: 0.0036
Epoch 15/20
 - 9s - loss: 0.0037
Epoch 16/20
 - 9s - loss: 0.0035
Epoch 17/20
 - 9s - loss: 0.0034
Epoch 18/20
 - 9s - loss: 0.0035
Epoch 19/20
 - 9s - loss: 0.0030
Epoch 20/20
 - 9s - loss: 0.0036
```

In [61]:

```
plt.plot(hist.history['loss'])
plt.title('Training model loss')
plt.ylabel('Loss')
plt.xlabel('epoch')
plt.legend(['train'],loc='upper left')
plt.show()
```



In [76]:

```
INFOSYS_NEW.columns
```

Out[76]:

```
Index(['Date', 'Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume'], dtype
='object')
```

In [92]:

```
INFOSYS_TEST = pd.read_csv("/content/INFY.NS.csv")
INFOSYS_TEST["Close"]=pd.to_numeric(INFOSYS_TEST.Close, errors='coerce')
INFOSYS_TEST = INFOSYS_TEST.dropna()
INFOSYS_TEST = INFOSYS_TEST.iloc[:,4:5]
y_test = INFOSYS_TEST.iloc[60:,0:].values

inputClosing = INFOSYS_TEST.iloc[:,0:].values
inputClosing_scaled = sc.transform(inputClosing)
inputClosing_scaled.shape
x_test = []
length = len(INFOSYS_TEST)
timestep = 60
for i in range (timestep, length):
  x_test.append(inputClosing_scaled[i-timestep:i,0])
x_test = np.array(x_test)
x_test = np.reshape(x_test, (x_test.shape[0],x_test.shape[1],1))
x_test.shape
```

Out[92]:

```
(1167, 60, 1)
```

In [88]:

```
y_predict = model.predict(x_test)
y_predict
```

Out[88]:

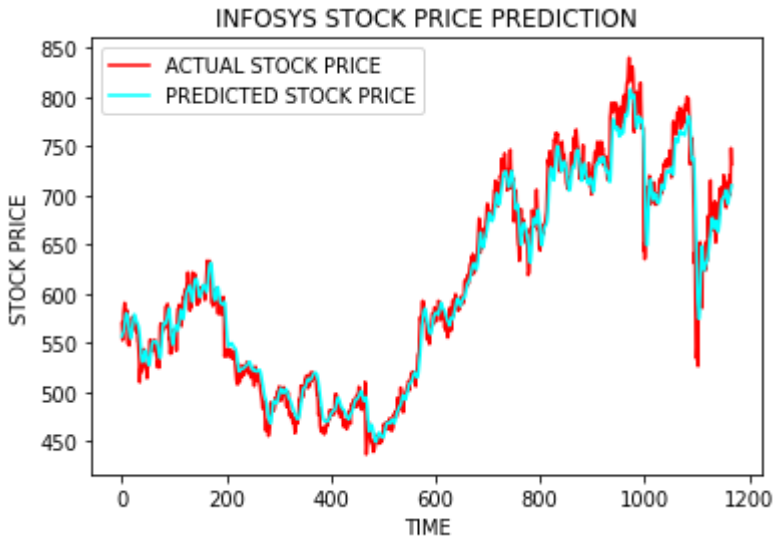
```
array([[0.29796383],
       [0.30271685],
       [0.30667093],
       ...,
       [0.6671327 ],
       [0.6693521 ],
       [0.67803097]], dtype=float32)
```

In [89]:

```
predicted_price = sc.inverse_transform(y_predict)
```

In [90]:

```
plt.plot(y_test, color='red', label='ACTUAL STOCK PRICE')
plt.plot(predicted_price, color='cyan', label='PREDICTED STOCK PRICE')
plt.title('INFOSYS STOCK PRICE PREDICTION')
plt.xlabel('TIME')
plt.ylabel('STOCK PRICE')
plt.legend()
plt.show()
```



In [2]:

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

In [5]:

```
df = pd.read_csv("titanic_test.csv")
df.head()
```

Out[5]:

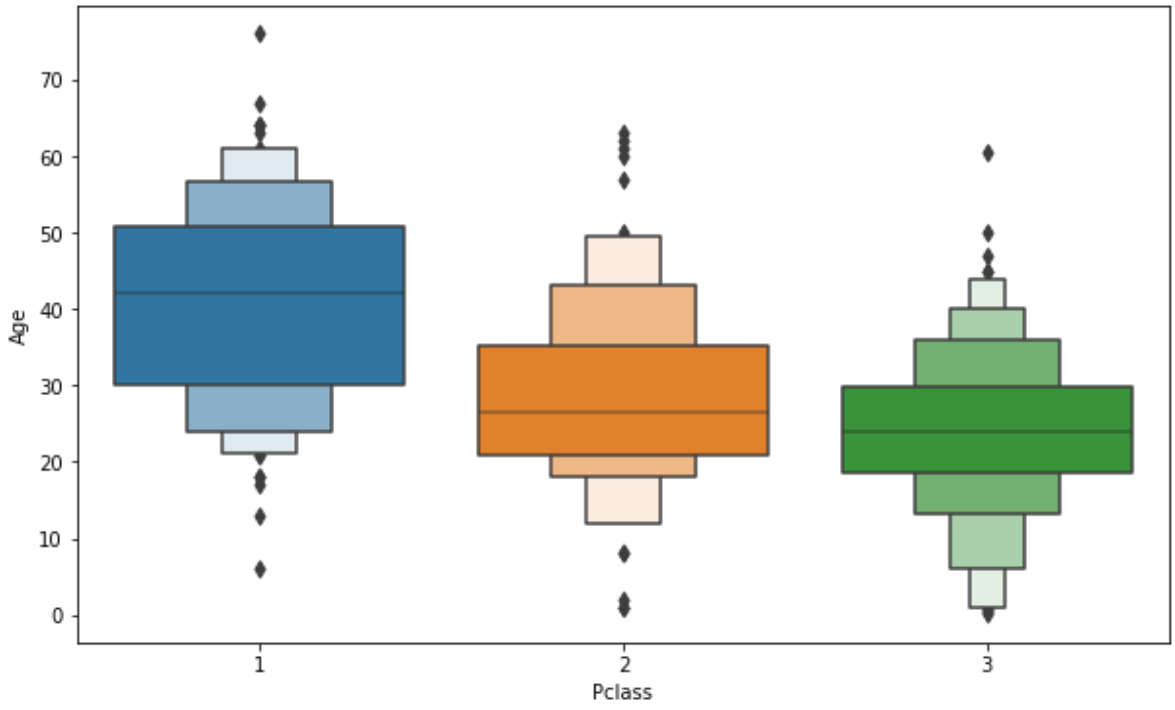
	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

In [6]:

```
figsize = plt.figure(figsize=(10,6))
sns.boxenplot(x='Pclass',y='Age',data=df)
```

Out[6]:

```
<AxesSubplot:xlabel='Pclass', ylabel='Age'>
```

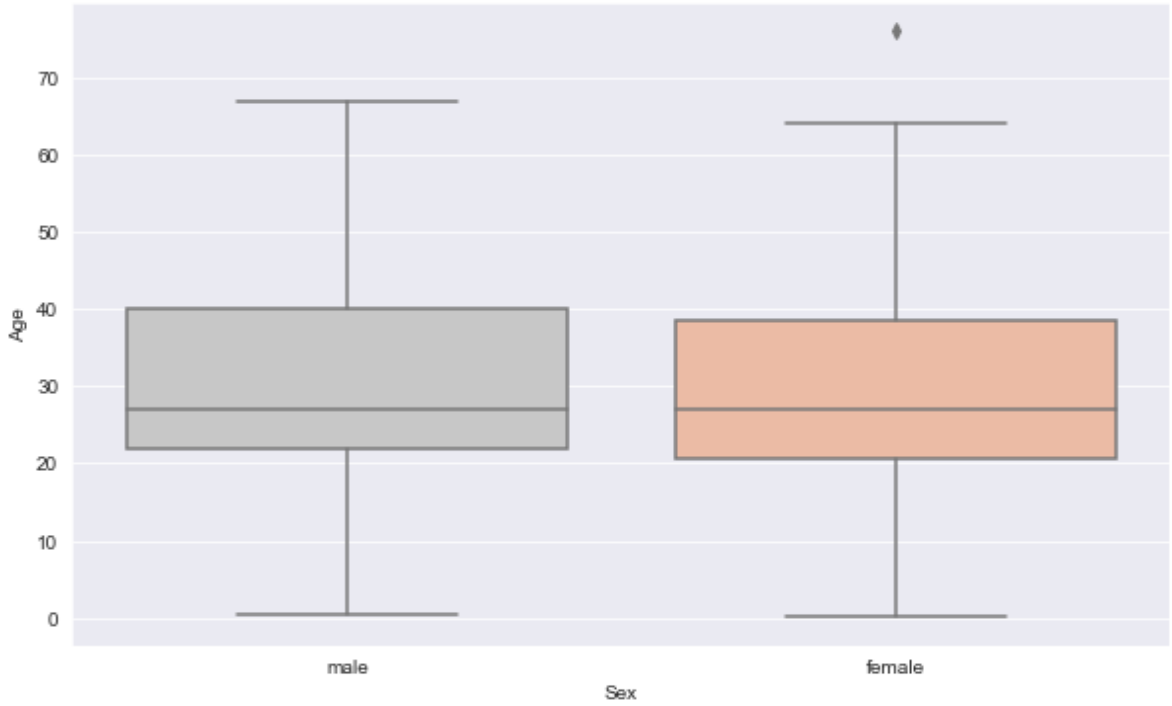


In [7]:

```
sns.set_style("darkgrid")
figsize = plt.figure(figsize=(10,6))
sns.boxplot(x='Sex',y='Age',data=df,palette='RdGy_r')
```

Out[7]:

```
<AxesSubplot:xlabel='Sex', ylabel='Age'>
```

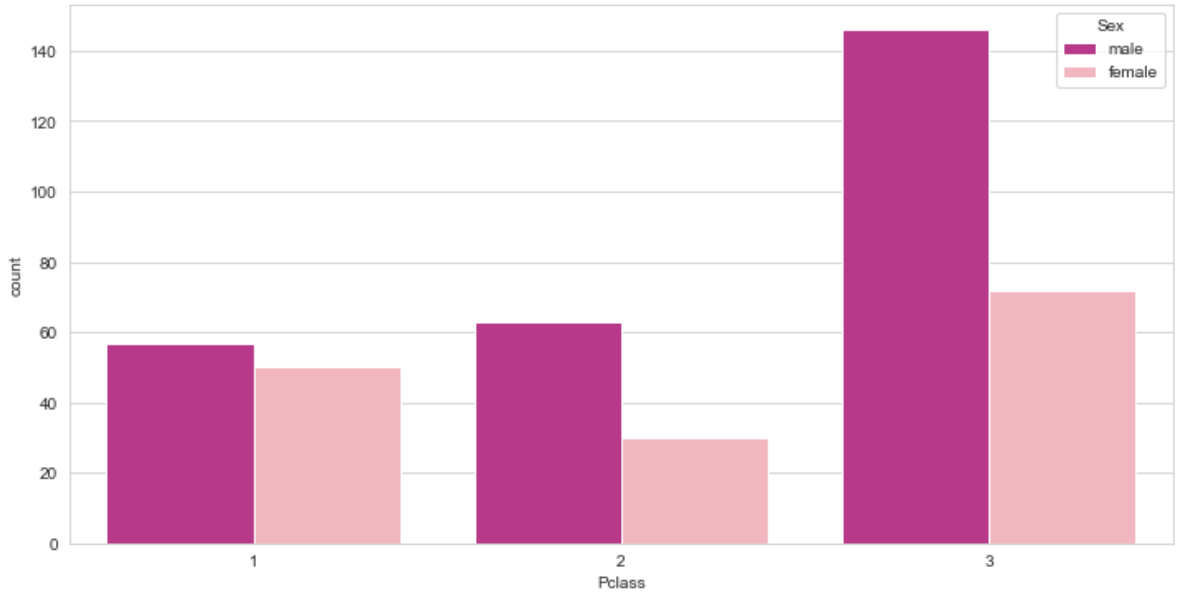


In [8]:

```
sns.set_style("whitegrid")
figure = plt.figure(figsize=(12,6))
sns.countplot(x="Pclass",data=df,hue="Sex",palette="RdPu_r")
```

Out[8]:

```
<AxesSubplot:xlabel='Pclass', ylabel='count'>
```

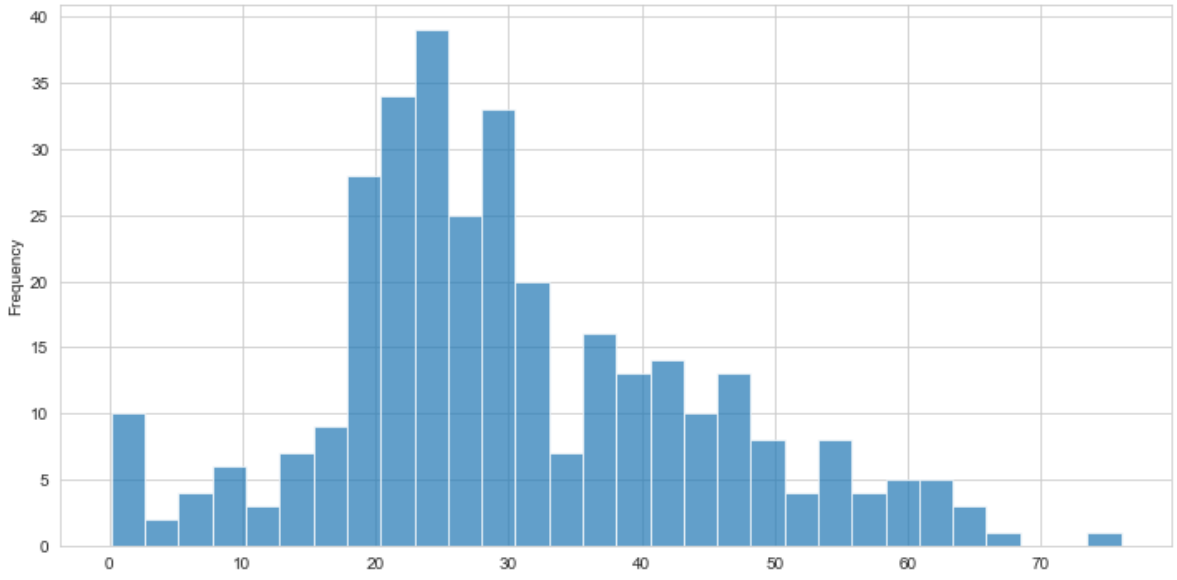


In [9]:

```
figsize = plt.figure(figsize=(12,6))
df["Age"].plot.hist(bins=30,alpha=0.7)
```

Out[9]:

```
<AxesSubplot:ylabel='Frequency'>
```



In [10]:

```
df.isnull().sum()
```

Out[10]:

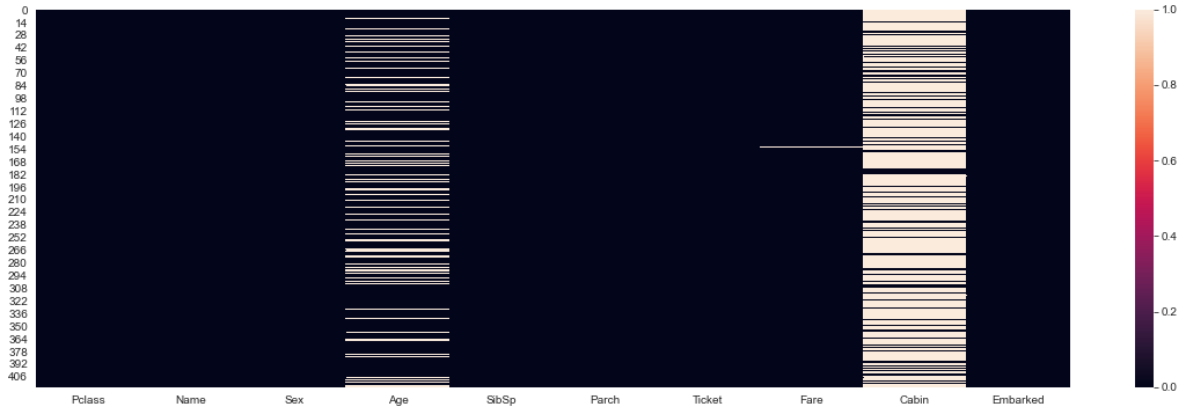
```
Pclass        0
Name          0
Sex           0
Age          86
SibSp         0
Parch         0
Ticket        0
Fare          1
Cabin       327
Embarked      0
dtype: int64
```

In [11]:

```
figsize = plt.figure(figsize=(20,6))
sns.heatmap(df.isnull())
```

Out[11]:

```
<AxesSubplot:>
```



In [12]:

```
figsize = plt.figure(figsize=(20,6))
df['Age'].plot(kind='kde',)
```

Out[12]:

```
<AxesSubplot:ylabel='Density'>
```