MACHINE LEARNING

I.VINOTH D SILVA

FEB-4-2023

IMPORT THE LIBRARIES

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")
```

LOAD THE DATASET

In [2]:
```
df = pd.read_csv("Social_Network_Ads.csv")
```

In [3]:
```
df.head()
```

Out[3]:

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0

EDA

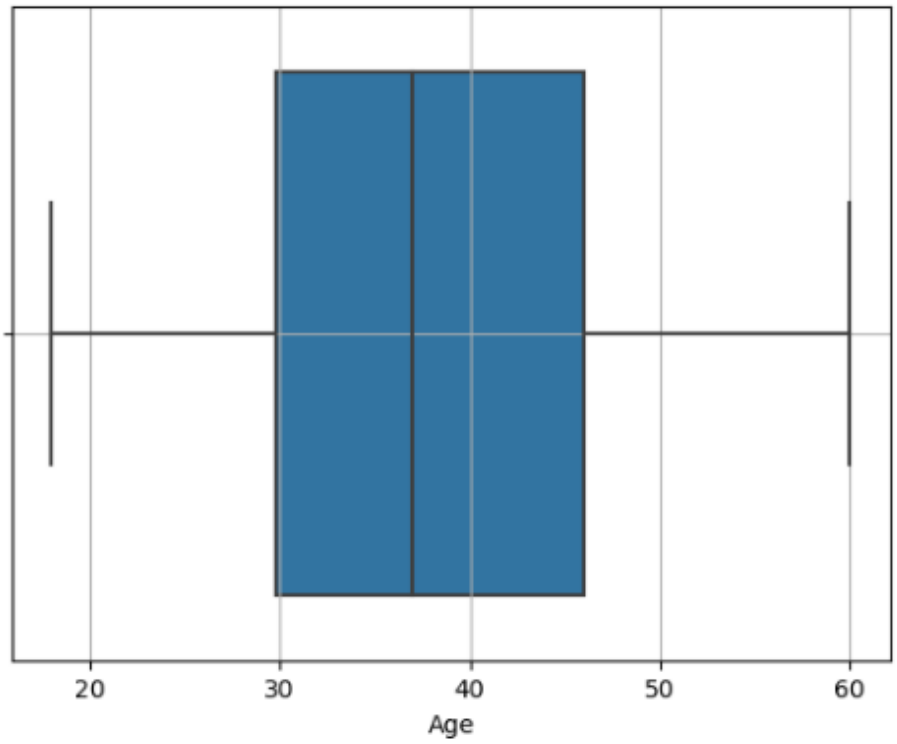
In [6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 5 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   User ID          400 non-null    int64 
 1   Gender           400 non-null    object
 2   Age              400 non-null    int64 
 3   EstimatedSalary  400 non-null    int64 
 4   Purchased        400 non-null    int64 
dtypes: int64(4), object(1)
memory usage: 15.8+ KB
```

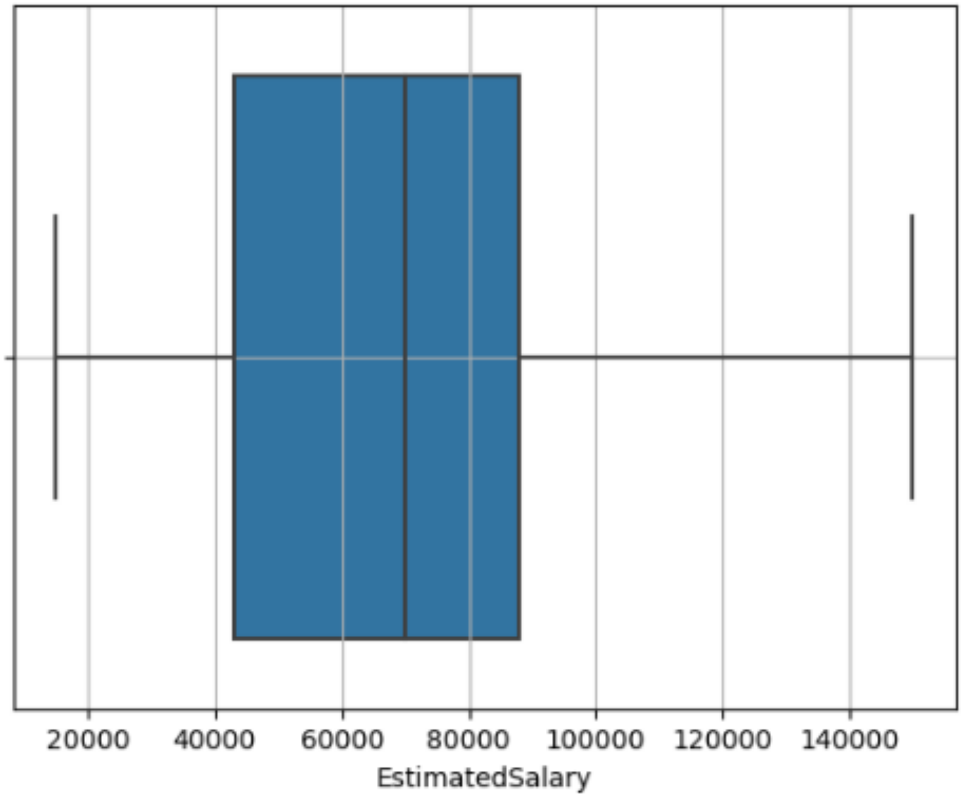
```
In [7]: df['Gender'].isna().sum()
Out[7]: 0

In [8]: df['Age'].isna().sum()
Out[8]: 0

In [9]: df['EstimatedSalary'].isna().sum()
Out[9]: 0
```

OUTLIERS

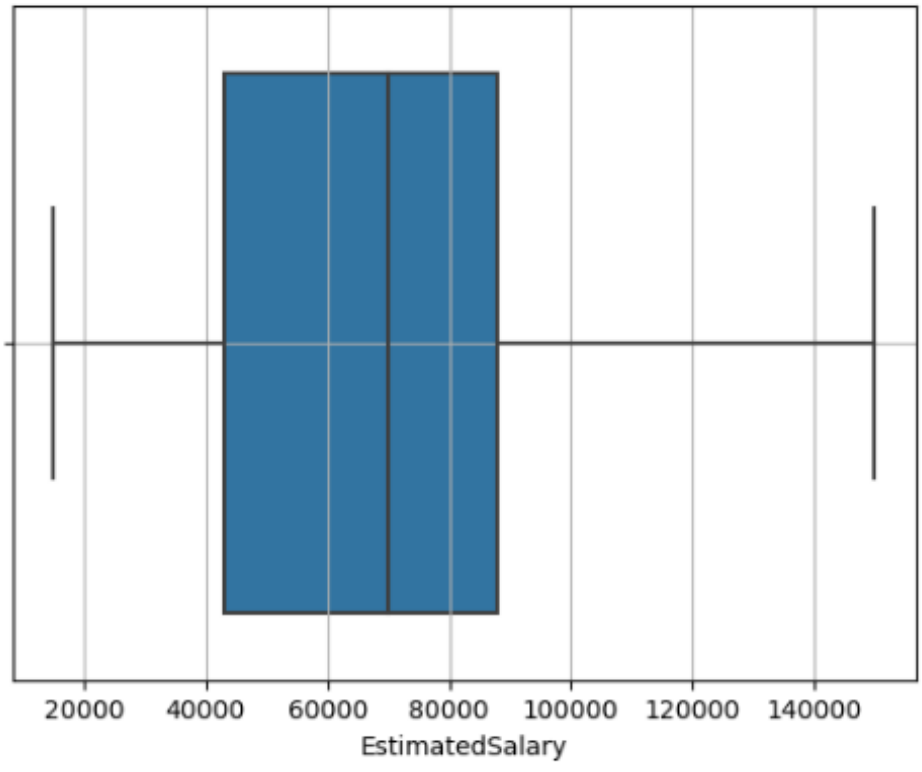
```
In [12]: sns.boxplot(df['Age'])
         plt.grid()
```



In [13]:
```
sns.boxplot(df['EstimatedSalary'])
plt.grid()
```



In [13]:
```
sns.boxplot(df['EstimatedSalary'])
plt.grid()
```

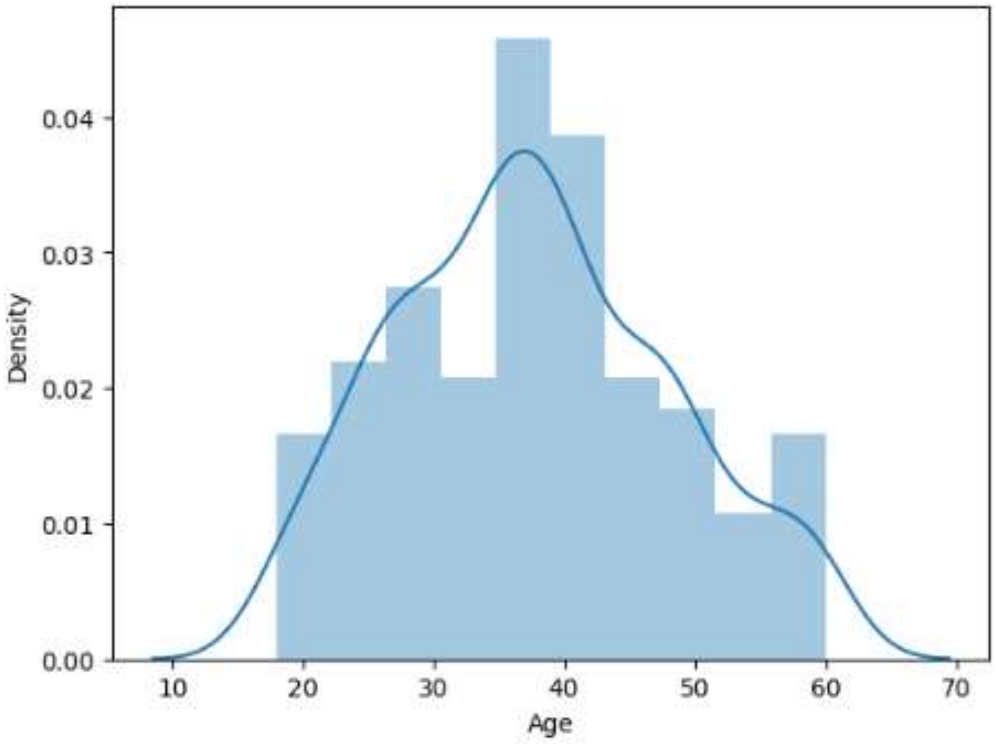


SKEW

In [60]:
```
from scipy.stats import skew
print(skew(df["Age"]))
sns.distplot(df["Age"])
```

```
0.23046904236325927
```

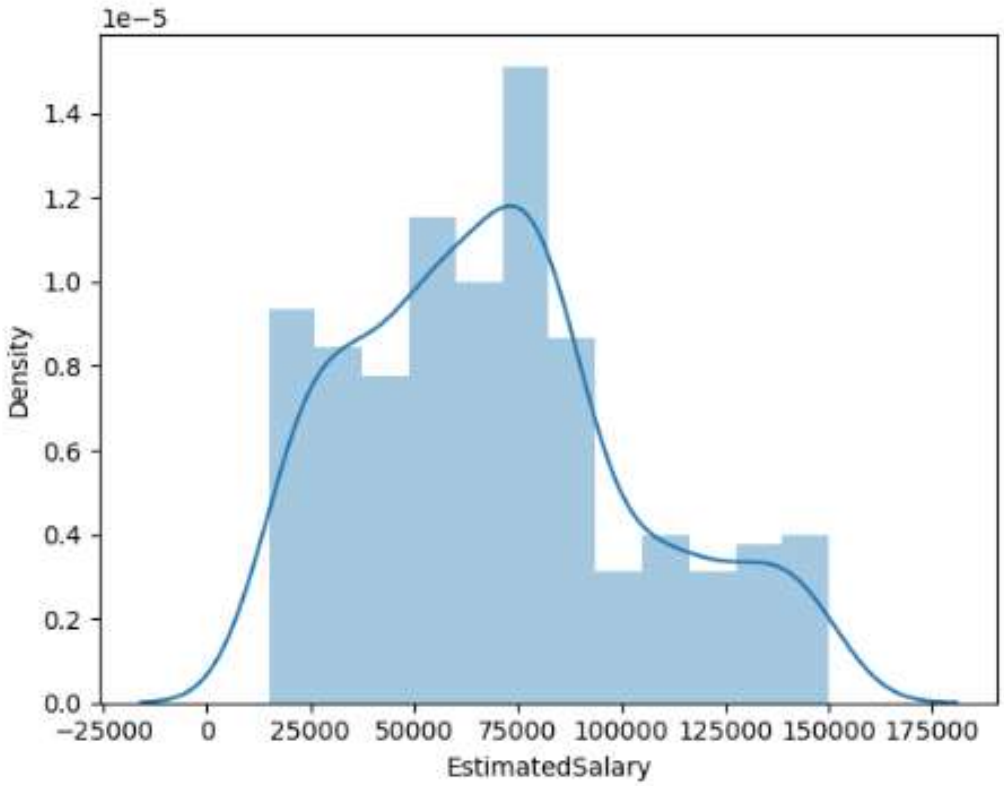
Out[60]: <AxesSubplot:xlabel='Age', ylabel='Density'>



In [61]:
```
print(skew(df["EstimatedSalary"]))
sns.distplot(df["EstimatedSalary"])
```

```
0.49316535320478544
```

Out[61]: <AxesSubplot:xlabel='EstimatedSalary', ylabel='Density'>



ENCODING

In [21]:
```
from sklearn.preprocessing import OrdinalEncoder
```

In [23]:
```
oe = OrdinalEncoder()
df['Gender'] = oe.fit_transform(df[['Gender']])
```

In [24]:
```
df['Gender']
```

Out[24]:
```
0      1.0
1      1.0
2      0.0
3      0.0
4      1.0
      ...
395    0.0
396    1.0
397    0.0
398    1.0
399    0.0
Name: Gender, Length: 400, dtype: float64
```

SCALING

In [29]:
```
x = df.iloc[:,[1,2,3]]
```

In [35]:
```
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler()
x = sc.fit_transform(x)
x =pd.DataFrame(x)
x
```

Out[35]:

	0	1	2
0	1.0	0.023810	0.029630
1	1.0	0.404762	0.037037
2	0.0	0.190476	0.207407
3	0.0	0.214286	0.311111
4	1.0	0.023810	0.451852
...	...	...	...
395	0.0	0.666667	0.192593
396	1.0	0.785714	0.059259
397	0.0	0.761905	0.037037
398	1.0	0.428571	0.133333
399	0.0	0.738095	0.155556

400 rows × 3 columns

In [38]:
```
x.columns = ['Gender','Age','EstimatedSalary']
x
```

Out[38]:

	Gender	Age	EstimatedSalary
0	1.0	0.023810	0.029630
1	1.0	0.404762	0.037037
2	0.0	0.190476	0.207407
3	0.0	0.214286	0.311111
4	1.0	0.023810	0.451852
...	...	...	...
395	0.0	0.666667	0.192593
396	1.0	0.785714	0.059259
397	0.0	0.761905	0.037037
398	1.0	0.428571	0.133333
399	0.0	0.738095	0.155556

400 rows × 3 columns

Feature and Target

In [39]:
```
feature = x
```

In [42]:
```
feature
```

Out[42]:

	Gender	Age	EstimatedSalary
0	1.0	0.023810	0.029630
1	1.0	0.404762	0.037037
2	0.0	0.190476	0.207407
3	0.0	0.214286	0.311111
4	1.0	0.023810	0.451852
...	...	...	...
395	0.0	0.666667	0.192593
396	1.0	0.785714	0.059259
397	0.0	0.761905	0.037037
398	1.0	0.428571	0.133333
399	0.0	0.738095	0.155556

400 rows × 3 columns

In [41]:
```
target = df["Purchased"]
```

In [43]:
```
target
```

Out[43]:
```
0      0
1      0
2      0
3      0
4      0
      ..
395    1
396    1
397    1
398    0
399    1
Name: Purchased, Length: 400, dtype: int64
```

SPLIT DATA

In [84]:
```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest = train_test_split(feature,target,test_size=0.3,random_state = 2)
```

MODEL BUILDING

KNeighborsClassifier

In [86]:
```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(xtrain,ytrain)
knnypred = knn.predict(xtest)
knnypred
```

Out[86]:
```
array([0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1,
       0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0,
       0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1,
       1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0,
       1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 0, 1, 0, 1, 0, 0, 1, 1, 0], dtype=int64)
```

MODEL EVALUATION

In [ ]:
```
from sklearn.metrics import classification_report
```

In [87]:
```
train = knn.score(xtrain, ytrain)
test = knn.score(xtest, ytest)
print(f"Training Accuracy : {train}\nTesting Accuracy : {test}\n\n")
```

```
Training Accuracy : 0.9214285714285714
Testing Accuracy : 0.9166666666666666
```

MODEL TRAINING

support Vector

In [88]:
```
from sklearn.svm import SVC
svc = SVC()
svc.fit(xtrain,ytrain)
svcypred = svc.predict(xtest)
svcypred
```

Out[88]:
```
array([0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0,
       0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0,
       0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1,
       1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0,
       1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 0, 1, 0, 1, 0, 0, 1, 1, 0], dtype=int64)
```

MODEL EVALUATION

In [ ]:
```
from sklearn.metrics import classification_report
```

In [89]:
```
train = svc.score(xtrain, ytrain)
test = svc.score(xtest, ytest)
print(f"Training Accuracy : {train}\nTesting Accuracy : {test}\n\n")
```

```
Training Accuracy : 0.9142857142857143
Testing Accuracy : 0.9083333333333333
```