eda-and-model-training

December 23, 2023

1 Day - 19 _____#100DaysOfML

[20]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

[21]:
```
df = pd.read_csv('../datasets/new.csv')
```

[22]:
```
df.isnull().sum()
```

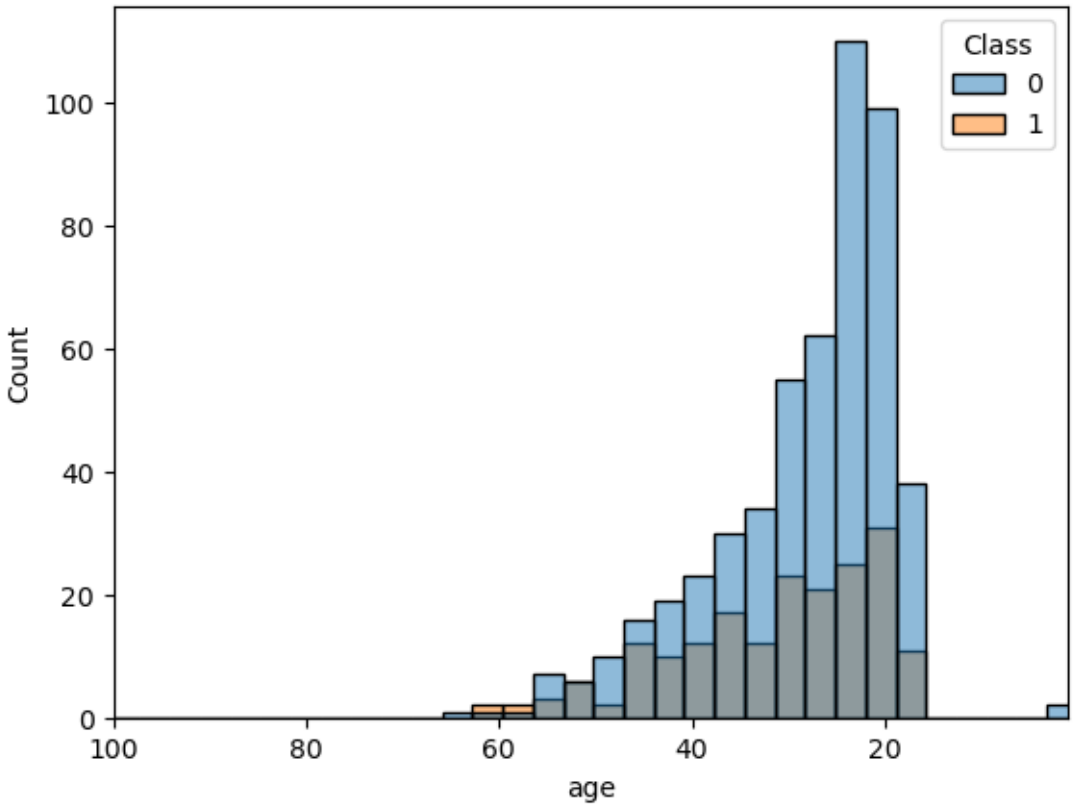
[22]:
```
A1_Score     0
A2_Score     0
A3_Score     0
A4_Score     0
A5_Score     0
A6_Score     0
A7_Score     0
A8_Score     0
A9_Score     0
A10_Score    0
age          0
gender       0
jaundice     0
autism       0
relation     0
Class        0
dtype: int64
```

[ ]:
```

```

[23]:
```
plt.xlim(100)
sns.histplot(data = df, x = 'age' ,hue = 'Class')
```

[23]:
```
<Axes: xlabel='age', ylabel='Count'>
```



[24]:
```
df.gender.value_counts()
```

[24]:
```
0    367
1    337
Name: gender, dtype: int64
```

[44]:
```
df
```

[44]:

	A1_Score	A2_Score	A3_Score	A4_Score	A5_Score	A6_Score	A7_Score	\
0	1	1	1	1	0	0	1	
1	1	1	0	1	0	0	0	
2	1	1	0	1	1	0	1	
3	1	1	0	1	0	0	1	
4	1	0	0	0	0	0	0	
..	…	…	…	…	…	…	…	
699	0	1	0	1	1	0	1	
700	1	0	0	0	0	0	0	
701	1	0	1	1	1	0	1	
702	1	0	0	1	1	0	1	
703	1	0	1	1	1	0	1	

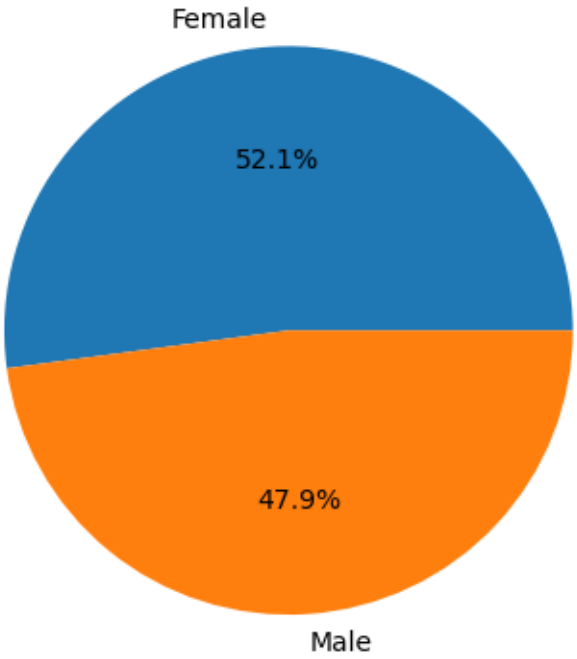
```
     A8_Score  A9_Score  A10_Score  age  gender  jaundice  autism  relation  \
0           1         0          0   26       1         0       0         0
1           1         0          1   24       0         0       1         0
2           1         1          1   27       0         1       1         1
3           1         0          1   35       1         0       1         0
4           1         0          0   40       1         0       0         2
..        ...       ...        ...  ...     ...       ...     ...       ...
699         1         1          1   25       1         0       0         0
700         1         0          1   34       0         0       0         1
701         1         0          1   24       1         0       0         2
702         0         1          1   35       0         0       0         0
703         1         1          1   26       1         0       0         0

     Class
0        0
1        0
2        1
3        0
4        0
..     ...
699      1
700      0
701      1
702      0
703      1

[704 rows x 16 columns]
```

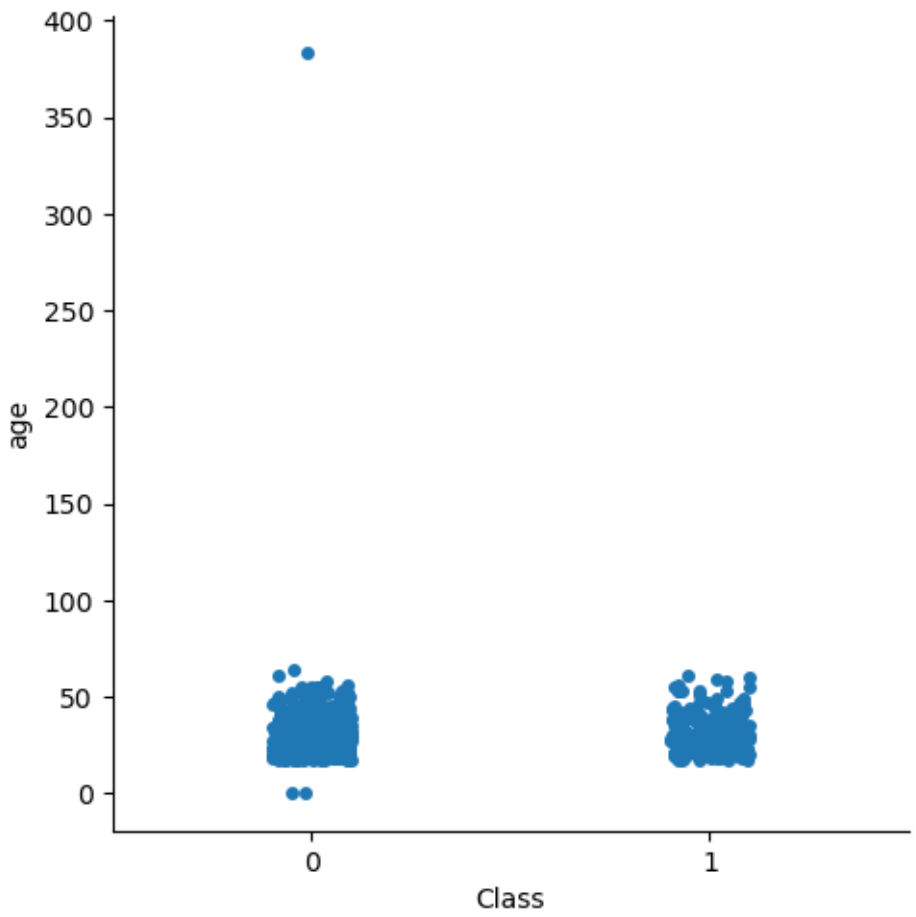
[25]:
```
plt.pie(df.gender.value_counts(), autopct='%1.1f%%', labels=['Female','Male'])
```

[25]:
```
([<matplotlib.patches.Wedge at 0x1b4f970ebc0>,
  <matplotlib.patches.Wedge at 0x1b4f970ead0>],
 [Text(-0.07357608612004667, 1.0975365868850366, 'Female'),
  Text(0.07357608612004629, -1.0975365868850366, 'Male')],
 [Text(-0.04013241061093454, 0.5986563201191109, '52.1%'),
  Text(0.04013241061093433, -0.5986563201191109, '47.9%')])
```



[26]:
```
sns.catplot(data=df, x="Class", y="age")
```

[26]: <seaborn.axisgrid.FacetGrid at 0x1b4f804e3b0>

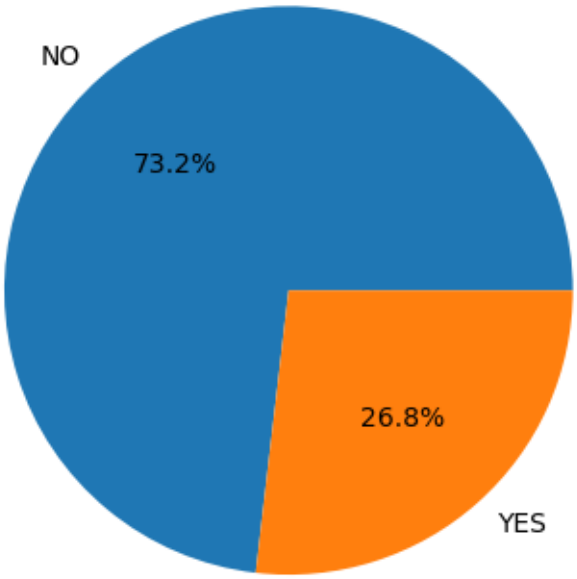


[27]:
```
df['Class'].value_counts()
```

[27]:
```
0    515
1    189
Name: Class, dtype: int64
```

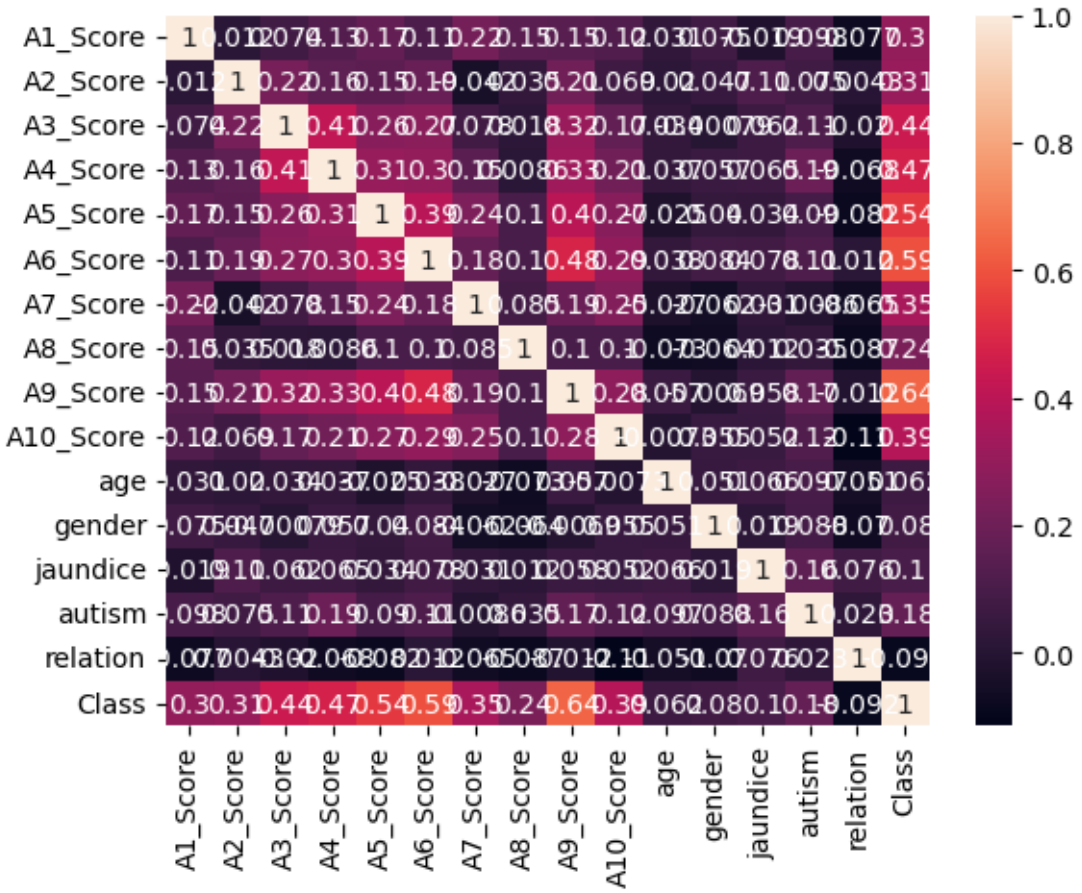
[28]:
```
plt.pie(df['Class'].value_counts(),labels=['NO','YES'],autopct='%1.1f%%')
```

[28]:
```
([<matplotlib.patches.Wedge at 0x1b4f66c7760>,
  <matplotlib.patches.Wedge at 0x1b4f66c77f0>],
 [Text(-0.7314111852459795, 0.8216067660974268, 'NO'),
  Text(0.7314111083215993, -0.8216068345770805, 'YES')],
 [Text(-0.39895155558871603, 0.4481491451440509, '73.2%'),
  Text(0.3989515136299632, -0.4481491824965893, '26.8%')])
```



[29]:
```
sns.heatmap(df.corr(),annot=True)
#No Correlarted columns
```

[29]: <Axes: >



[32]:
```
y
```

[32]:
```
0      0
1      0
2      1
3      0
4      0
      ..
699    1
700    0
701    1
702    0
703    1
Name: Class, Length: 704, dtype: int64
```

[31]:
```
y=df['Class']
X = df.copy()
X.drop('Class',axis = 1,inplace  = True)
```

[ ]:
```
X
```

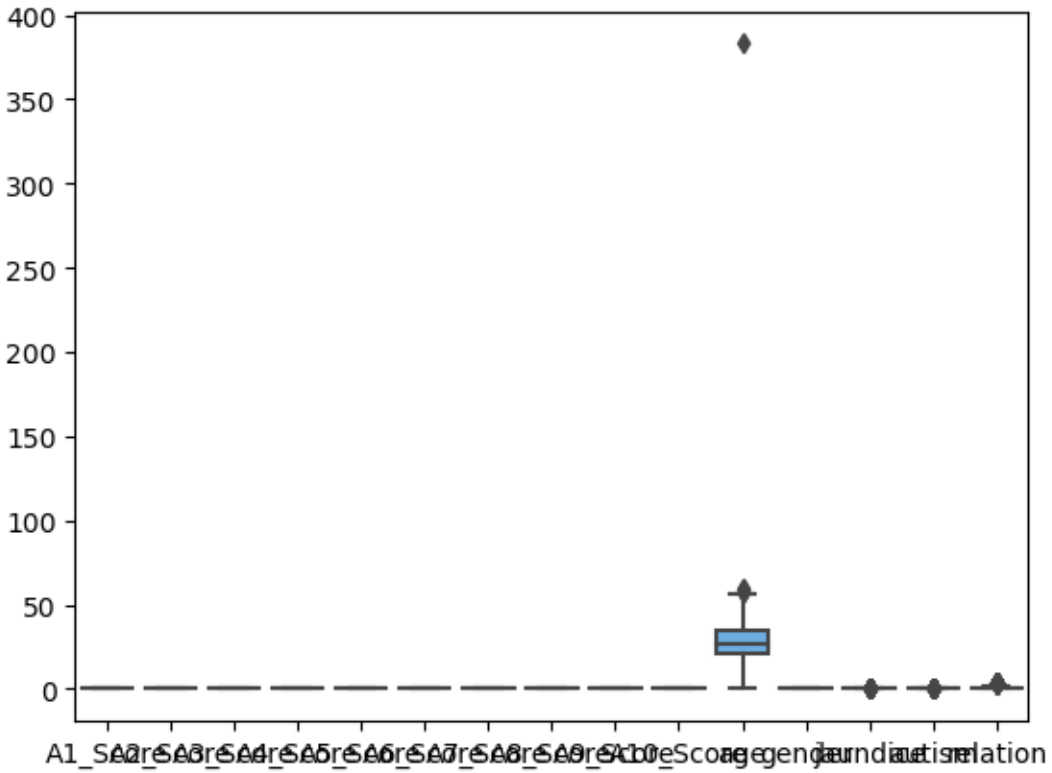
[33]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,␣
↪random_state=42)
```

```
<IPython.core.display.Javascript object>
```

[ ]:
```
X_train
```

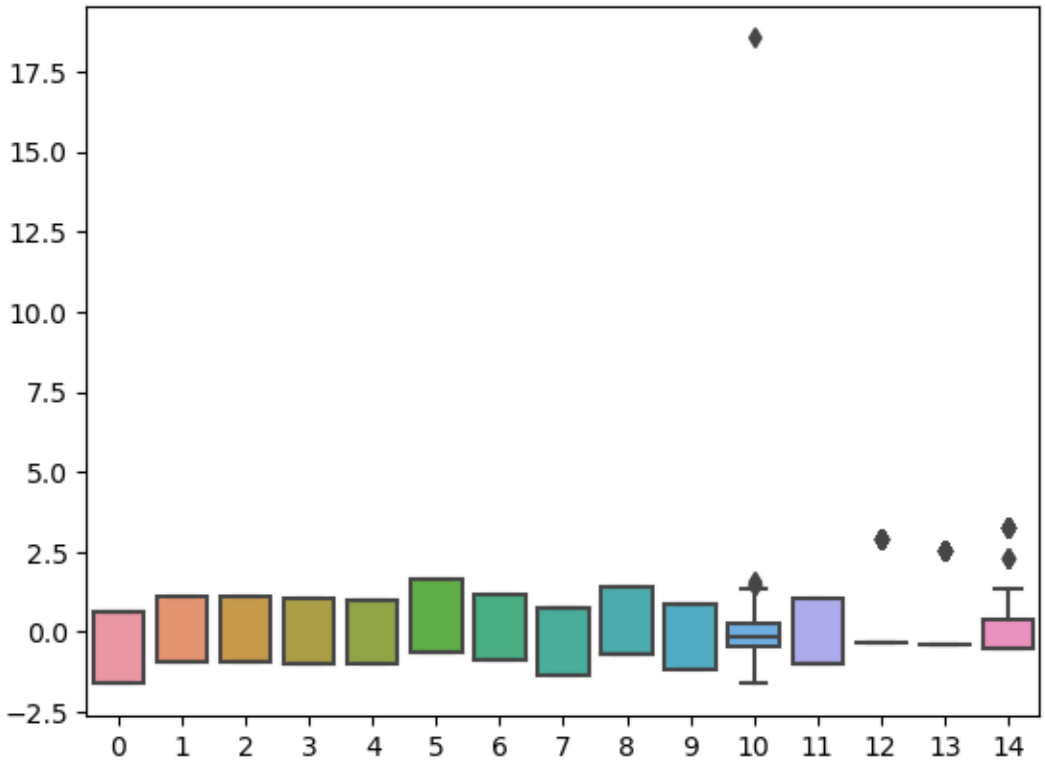
[43]:
```
# plt.subplot(1,2,1)
# plt.figure(figsize=(5,5))
sns.boxplot(X_train)
# plt.figure(figsize=(5,5))
# plt.subplot(1,2,2)
# sns.boxplot(X_train_scaled)
```

[43]: <Axes: >



[37]:
```
sns.boxplot(X_train_scaled)
```

[37]: <Axes: >



[36]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train_scaled = sc.fit_transform(X_train)
X_test_scaled = sc.transform(X_test)
```

[ ]:
```
from sklearn.linear_model import LogisticRegression
log_reg = LogisticRegression()
log_reg.fit(X_train_scaled,y_train)b
```

[ ]:
```
log_reg_pred = log_reg.predict(X_test_scaled)
```

[ ]:
```
from sklearn.metrics import accuracy_score
log_reg_acc = accuracy_score(y_test,log_reg_pred)
print(log_reg_acc)
```

[ ]:
```
from sklearn.linear_model import LinearRegression
linear = LinearRegression()
linear.fit(X_train_scaled,y_train)
```

[ ]:
```
from sklearn.ensemble import RandomForestClassifier
random = RandomForestClassifier()
random.fit(X_train_scaled,y_train)
```

[ ]:
```
random_pred = random.predict(X_test_scaled)
random_acc = accuracy_score(y_test,random_pred)
print(random_acc)
```