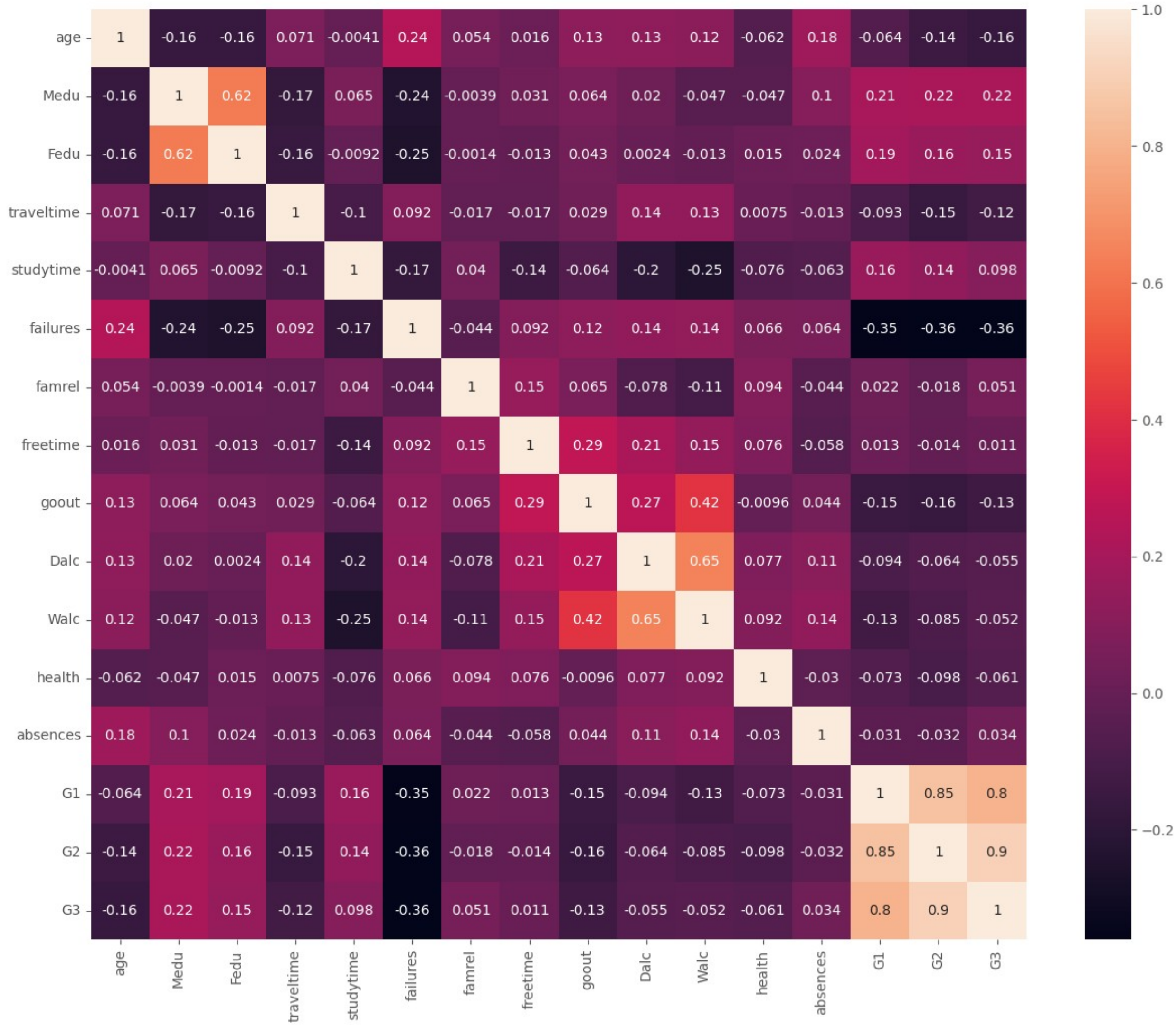
```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import preprocessing
import numpy as np
import pylab as pl
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from sklearn.utils import shuffle
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import cross_val_score, GridSearchCV
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import pandas as pd
from sklearn.metrics import mean_absolute_error, mean_squared_error
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

import warnings
warnings.filterwarnings("ignore")

plt.style.use('ggplot')

student_mat = pd.read_csv('student-mat.csv')
student_mat
```

```
    school sex  age address famsize Pstatus  Medu  Fedu      Mjob
Fjob  \
0       GP   F   18       U     GT3       A     4     4   at_home
teacher
1       GP   F   17       U     GT3       T     1     1   at_home
other
2       GP   F   15       U     LE3       T     1     1   at_home
other
3       GP   F   15       U     GT3       T     4     2    health
services
4       GP   F   16       U     GT3       T     3     3     other
other
..     ...  ..  ...     ...     ...     ...   ...   ...       ...
...
390     MS   M   20       U     LE3       A     2     2  services
services
391     MS   M   17       U     LE3       T     3     1  services
services
392     MS   M   21       R     GT3       T     1     1     other
other
393     MS   M   18       R     LE3       T     3     2  services
```

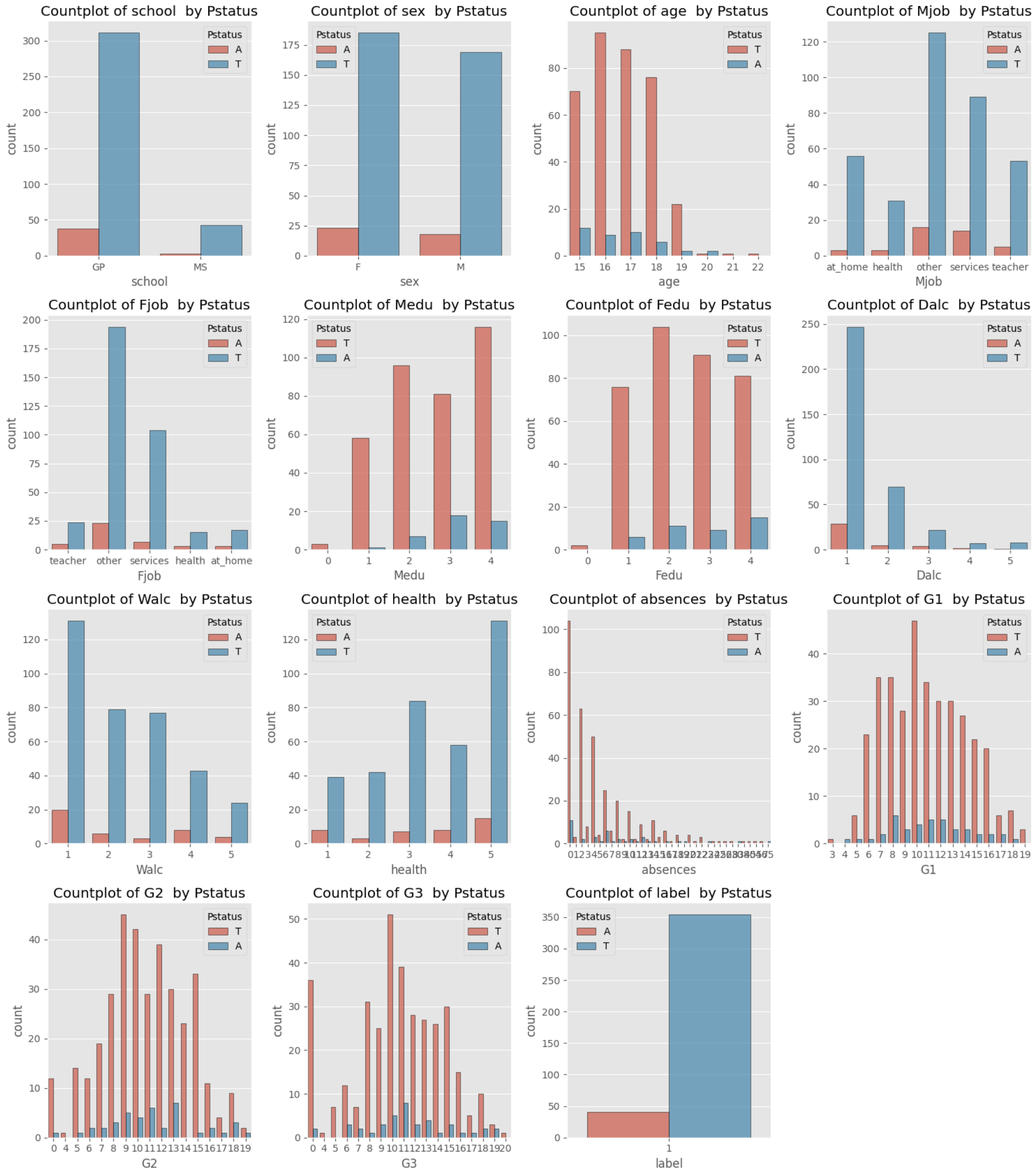
```
other
394     MS    M    19         U      LE3        T      1      1      other
at_home

     ... famrel freetime  goout  Dalc  Walc health absences  G1  G2
G3
0    ...      4        3      4     1     1      3        6   5   6
6
1    ...      5        3      3     1     1      3        4   5   5
6
2    ...      4        3      2     2     3      3       10   7   8
10
3    ...      3        2      2     1     1      5        2  15  14
15
4    ...      4        3      2     1     2      5        4   6  10
10
..   ...    ...      ...    ...   ...   ...    ...      ...  ..  ..  .
.
390  ...      5        5      4     4     5      4       11   9   9
9
391  ...      2        4      5     3     4      2        3  14  16
16
392  ...      5        5      3     3     3      3        3  10   8
7
393  ...      4        4      1     3     4      5        0  11  12
10
394  ...      3        2      3     3     3      5        5   8   9
9

[395 rows x 33 columns]

plt.figure(figsize=(15,12))
sns.heatmap(student_mat.corr(numeric_only=True),annot=True)
plt.show()
```



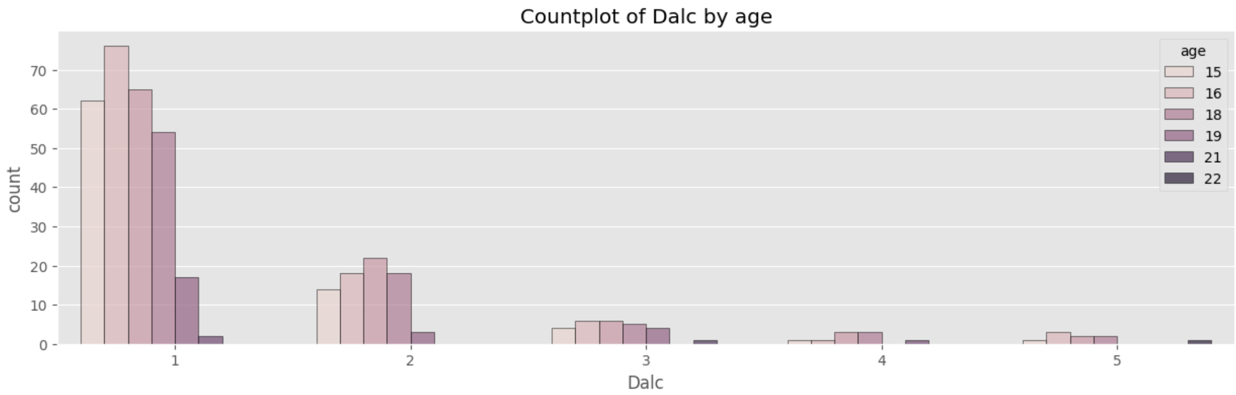
```
student_mat['label']="1"

student_mat.head()

  school sex  age address famsize Pstatus  Medu  Fedu     Mjob
Fjob  ...  \
0     GP   F   18       U     GT3       A     4     4  at_home
teacher  ...
1     GP   F   17       U     GT3       T     1     1  at_home
other  ...
2     GP   F   15       U     LE3       T     1     1  at_home
other  ...
3     GP   F   15       U     GT3       T     4     2   health
services  ...
4     GP   F   16       U     GT3       T     3     3    other
other  ...
```

```
  freetime goout  Dalc  Walc  health absences  G1  G2  G3 label
0        3     4     1     1       3        6   5   6   6     1
1        3     3     1     1       3        4   5   5   6     1
2        3     2     2     3       3       10   7   8  10     1
3        2     2     1     1       5        2  15  14  15     1
4        3     2     1     2       5        4   6  10  10     1

[5 rows x 34 columns]
```

```
plt.figure(figsize=[15,17])
fft=['school','sex','age','Mjob','Fjob','Medu','Fedu','Dalc','Walc','health','absences','G1','G2','G3','label']
n=1
for f in fft:
    plt.subplot(4,4,n)
    sns.countplot(x=f, hue='Pstatus', edgecolor="black", alpha=0.7, data=student_mat)
    sns.despine()
    plt.title("Countplot of {}  by Pstatus".format(f))
    n=n+1
plt.tight_layout()
plt.show()
```

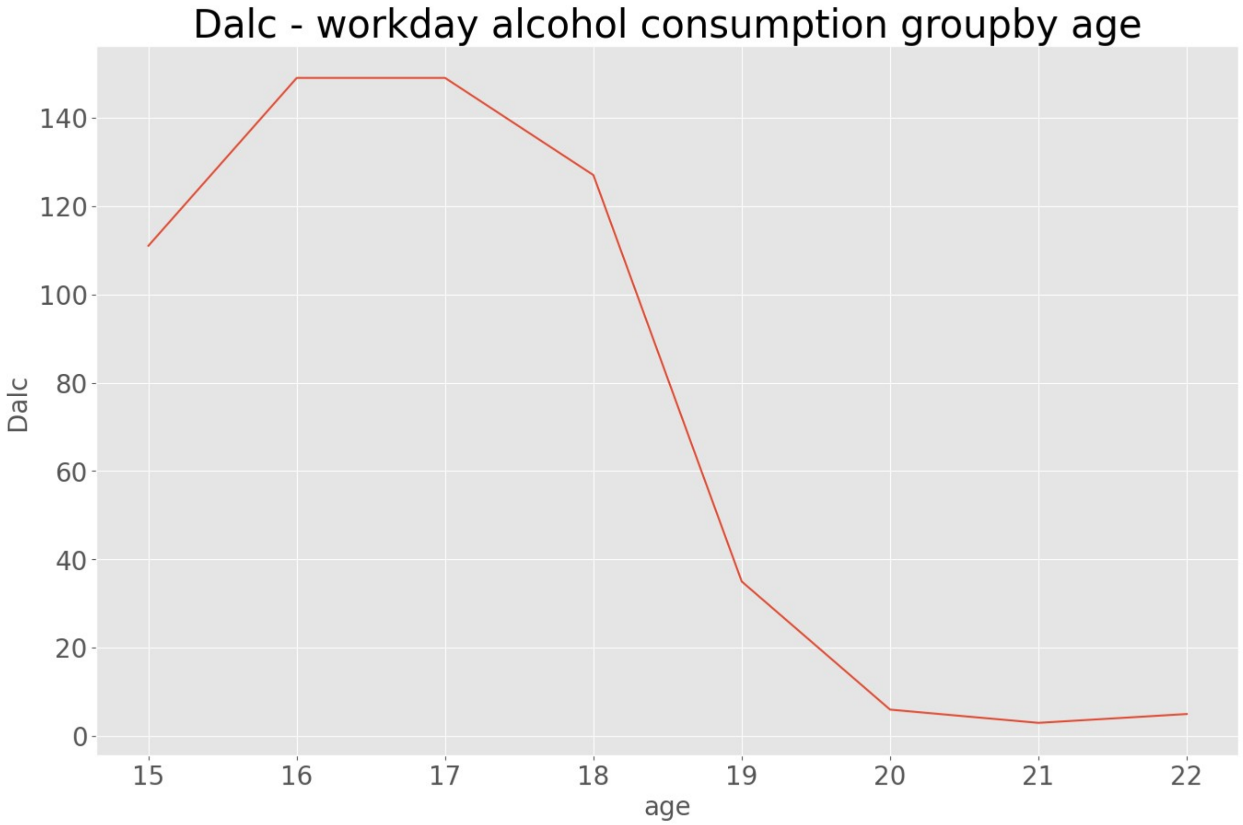


```
plt.figure(figsize=[15,4])
sns.countplot(x='Dalc', hue='age',edgecolor="black", alpha=0.7,
data=student_mat)
sns.despine()
plt.title("Countplot of Dalc by age")
plt.show()
```

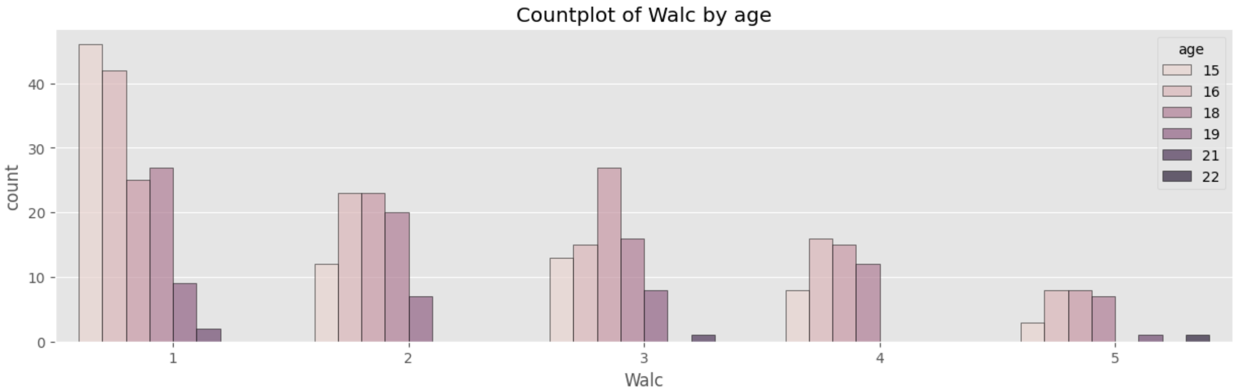


```
fig = plt.figure(figsize=(16,10))
ax = fig.add_subplot(111)
dfg = student_mat.groupby('age').sum()['Dalc']
dfg.plot(kind='line', title='Dalc - workday alcohol consumption
groupby age', fontsize=20)

plt.ylabel('Dalc ')
ax.title.set_fontsize(30)
ax.xaxis.label.set_fontsize(20)
ax.yaxis.label.set_fontsize(20)
print('Max: ' + str(dfg.max()) + ' ocurred in ' + str(dfg.loc[dfg ==
dfg.max()].index.values[0:]))
print('Max: ' + str(dfg.min()) + ' ocurred in ' + str(dfg.loc[dfg ==
dfg.min()].index.values[0:]))
print('Mean: ' + str(dfg.mean()))
```

```
Max: 149 ocurred in [16 17]
Max: 3 ocurred in [21]
Mean: 73.125
```



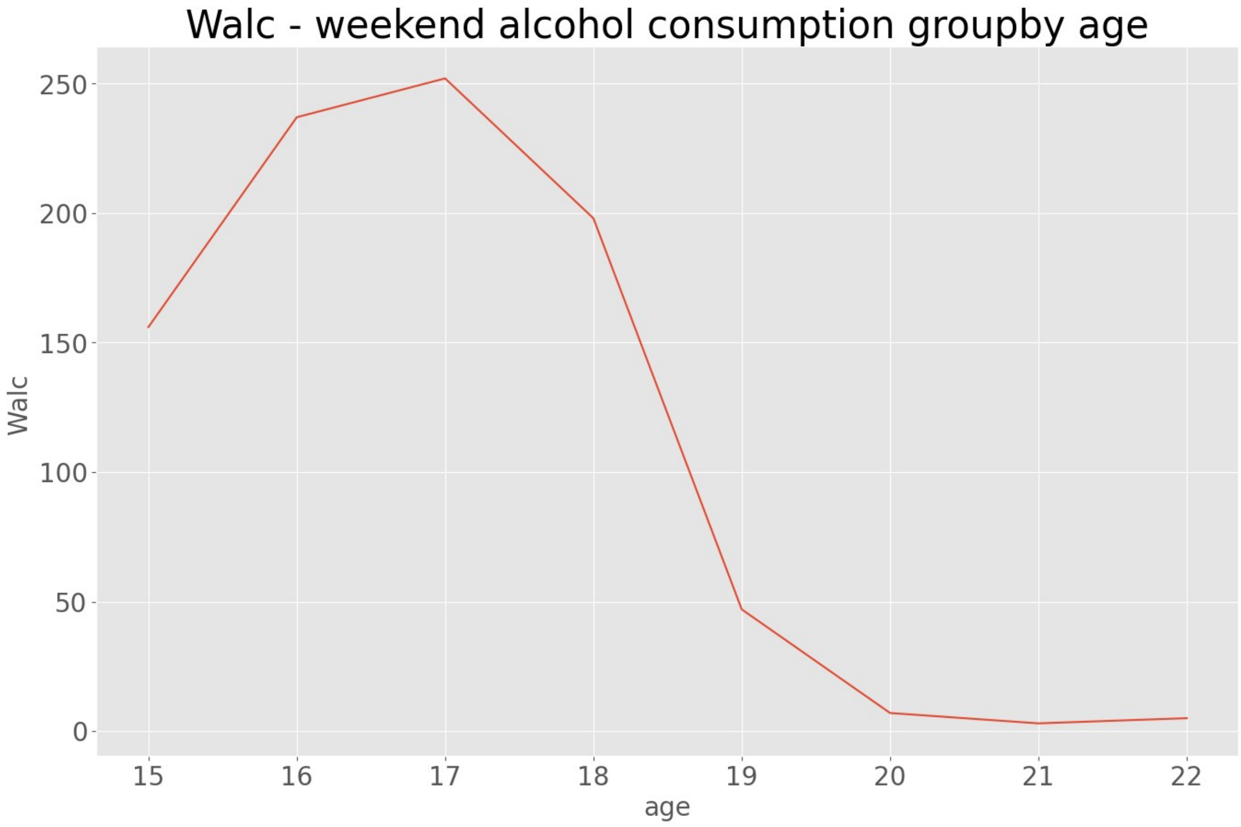
```
plt.figure(figsize=[15,4])
sns.countplot(x='Walc', hue='age',edgecolor="black", alpha=0.7,
data=student_mat)
sns.despine()
plt.title("Countplot of Walc by age")
plt.show()
```



```
fig = plt.figure(figsize=(16,10))
ax = fig.add_subplot(111)
dfg = student_mat.groupby('age').sum()['Walc']
```

```
dfg.plot(kind='line', title='Walc - weekend alcohol consumption
groupby age', fontsize=20)

plt.ylabel('Walc ')
ax.title.set_fontsize(30)
ax.xaxis.label.set_fontsize(20)
ax.yaxis.label.set_fontsize(20)
print('Max: ' + str(dfg.max()) + ' ocurred in ' + str(dfg.loc[dfg ==
dfg.max()].index.values[0:]))
print('Max: ' + str(dfg.min()) + ' ocurred in ' + str(dfg.loc[dfg ==
dfg.min()].index.values[0:]))
print('Mean: ' + str(dfg.mean()))
```

```
Max: 252 ocurred in [17]
Max: 3 ocurred in [21]
Mean: 113.125
```



```
student_por = pd.read_csv('student-por.csv')
student_por
student_por['label']="0"
student_por.head()
```

```
  school sex  age address famsize Pstatus  Medu  Fedu     Mjob
Fjob  ...  \
0     GP   F   18       U     GT3       A     4     4  at_home
teacher  ...
1     GP   F   17       U     GT3       T     1     1  at_home
other  ...
2     GP   F   15       U     LE3       T     1     1  at_home
other  ...
3     GP   F   15       U     GT3       T     4     2   health
services  ...
4     GP   F   16       U     GT3       T     3     3    other
other  ...

  freetime goout  Dalc  Walc  health absences  G1  G2  G3 label
0        3     4     1     1       3         4   0  11  11     0
1        3     3     1     1       3         2   9  11  11     0
2        3     2     2     3       3         6  12  13  12     0
3        2     2     1     1       5         0  14  14  14     0
4        3     2     1     2       5         0  11  13  13     0

[5 rows x 34 columns]
```

```
Data= student_mat._append([student_mat,student_por])
```

```
x = Data.iloc[:, [3]].values
Data
```

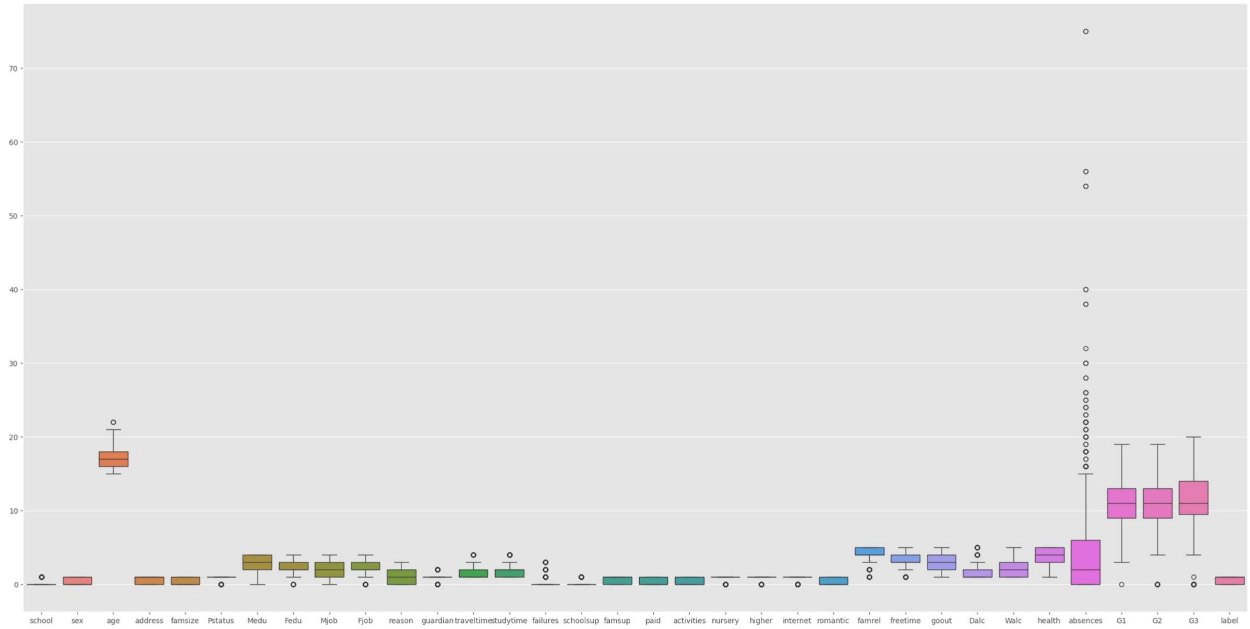
```
    school sex  age address famsize Pstatus  Medu  Fedu      Mjob
Fjob  \
0       GP   F   18       U     GT3       A     4     4   at_home
teacher
1       GP   F   17       U     GT3       T     1     1   at_home
other
2       GP   F   15       U     LE3       T     1     1   at_home
other
3       GP   F   15       U     GT3       T     4     2    health
services
4       GP   F   16       U     GT3       T     3     3     other
other
..     ...  ..  ...     ...     ...     ...   ...   ...       ...
...
644     MS   F   19       R     GT3       T     2     3  services
other
645     MS   F   18       U     LE3       T     3     1   teacher
services
646     MS   F   18       U     GT3       T     1     1     other
other
647     MS   M   17       U     LE3       T     3     1  services
services
648     MS   M   18       R     LE3       T     3     2  services
other
```

```
     ... freetime goout  Dalc  Walc  health absences  G1  G2  G3 label

0    ...        3     4     1     1       3        6   5   6   6     1

1    ...        3     3     1     1       3        4   5   5   6     1

2    ...        3     2     2     3       3       10   7   8  10     1

3    ...        2     2     1     1       5        2  15  14  15     1

4    ...        3     2     1     2       5        4   6  10  10     1

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

644  ...        4     2     1     2       5        4  10  11  10     0

645  ...        3     4     1     1       1        4  15  15  16     0

646  ...        1     1     1     1       5        6  11  12   9     0

647  ...        4     5     3     4       2        6  10  10  10     0

648  ...        4     1     3     4       5        4  10  11  11     0


[1439 rows x 34 columns]
```

```python
##Convert string to numeric
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
def FunLabelEncoder(df):
    for c in df.columns:
        if df.dtypes[c] == object:
            le.fit(df[c].astype(str))
            df[c] = le.transform(df[c].astype(str))
    return df

Data = FunLabelEncoder(Data)
Data.info()
Data.iloc[0:4,:]
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1439 entries, 0 to 648
Data columns (total 34 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   school      1439 non-null   int64
 1   sex         1439 non-null   int64
 2   age         1439 non-null   int64
 3   address     1439 non-null   int64
 4   famsize     1439 non-null   int64
 5   Pstatus     1439 non-null   int64
 6   Medu        1439 non-null   int64
 7   Fedu        1439 non-null   int64
 8   Mjob        1439 non-null   int64
 9   Fjob        1439 non-null   int64
 10  reason      1439 non-null   int64
 11  guardian    1439 non-null   int64
 12  traveltime  1439 non-null   int64
 13  studytime   1439 non-null   int64
 14  failures    1439 non-null   int64
 15  schoolsup   1439 non-null   int64
 16  famsup      1439 non-null   int64
 17  paid        1439 non-null   int64
 18  activities  1439 non-null   int64
 19  nursery     1439 non-null   int64
 20  higher      1439 non-null   int64
 21  internet    1439 non-null   int64
 22  romantic    1439 non-null   int64
 23  famrel      1439 non-null   int64
 24  freetime    1439 non-null   int64
 25  goout       1439 non-null   int64
 26  Dalc        1439 non-null   int64
 27  Walc        1439 non-null   int64
 28  health      1439 non-null   int64
 29  absences    1439 non-null   int64
 30  G1          1439 non-null   int64
 31  G2          1439 non-null   int64
 32  G3          1439 non-null   int64
 33  label       1439 non-null   int64
dtypes: int64(34)
memory usage: 393.5 KB

   school  sex  age  address  famsize  Pstatus  Medu  Fedu  Mjob  Fjob
...  \
0       0    0   18        1        0        0     4     4     0     4
...
1       0    0   17        1        0        1     1     1     0     2
...
2       0    0   15        1        1        1     1     1     0     2
...
3       0    0   15        1        0        1     4     2     1     3
...

   freetime  goout  Dalc  Walc  health  absences  G1  G2  G3  label
0         3      4     1     1       3         6   5   6   6      1
1         3      3     1     1       3         4   5   5   6      1
2         3      2     2     3       3        10   7   8  10      1
3         2      2     1     1       5         2  15  14  15      1

[4 rows x 34 columns]
```

```
plt.figure(figsize=(30,15))
sns.boxplot(data=Data)
plt.show()
```

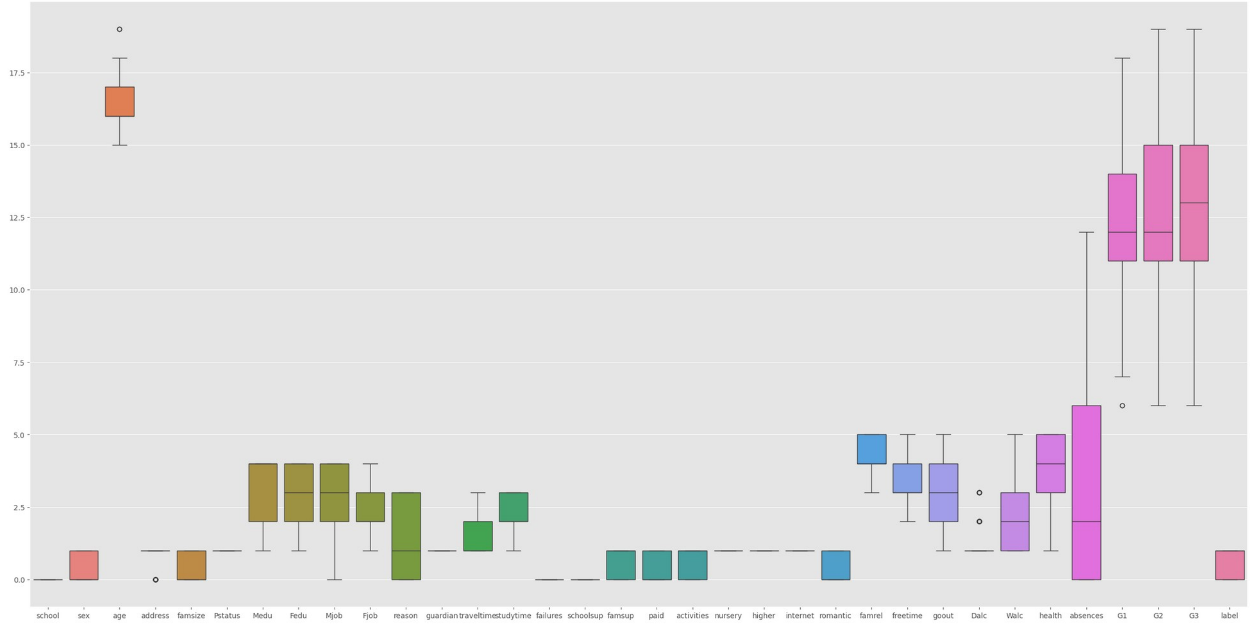


```
Q1 = Data.quantile(0.25)
Q3 = Data.quantile(0.75)
IQR= Q3-Q1
#print the IQR
print(IQR)
```

```
school         0.0
sex            1.0
age            2.0
address        1.0
famsize        1.0
Pstatus        0.0
Medu           2.0
Fedu           1.0
Mjob           2.0
Fjob           1.0
reason         2.0
guardian       0.0
traveltime     1.0
studytime      1.0
failures       0.0
schoolsup      0.0
famsup         1.0
paid           1.0
activities     1.0
nursery        0.0
```

```
higher        0.0
internet      0.0
romantic      1.0
famrel        1.0
freetime      1.0
goout         2.0
Dalc          1.0
Walc          2.0
health        2.0
absences      6.0
G1            4.0
G2            4.0
G3            4.5
label         1.0
dtype: float64
```

```
ul =Q3 + 1.5*IQR
ll =Q1 - 1.5*IQR

Data= Data[~((Data <ll) |(Data> ul)).any(axis=1)]

plt.figure(figsize=(30,15))
sns.boxplot(data=Data)
plt.show()
```



```
from sklearn.model_selection import train_test_split
Y = Data['label']
X = Data.drop(columns=['label'])
X_train, X_test, Y_train, Y_test = train_test_split(X, Y,
test_size=0.15, random_state=9)
```

```
print('X train shape: ', X_train.shape)
print('Y train shape: ', Y_train.shape)
print('X test shape: ', X_test.shape)
print('Y test shape: ', Y_test.shape)
```

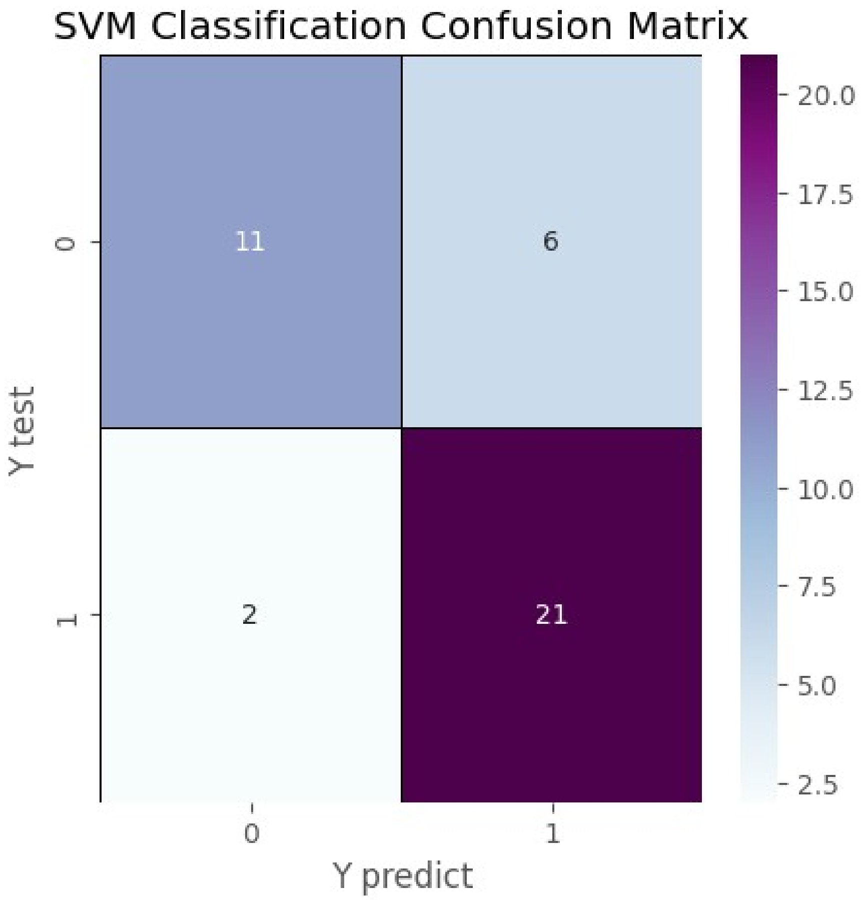
```
X train shape:  (225, 33)
Y train shape:  (225,)
X test shape:  (40, 33)
Y test shape:  (40,)
```

```
X_train
```

```
     school  sex  age  address  famsize  Pstatus  Medu  Fedu  Mjob
Fjob  ...  \
139       0    0   15        1        0        1     4     4     4
4  ...
357       0    0   18        1        0        1     4     3     2
2  ...
347       0    1   18        1        0        1     4     3     4
2  ...
258       0    1   18        1        0        1     2     1     2
2  ...
15        0    0   16        1        0        1     4     4     1
2  ...
..      ...  ...  ...      ...      ...      ...   ...   ...   ...    .
..  ...
302       0    0   17        1        0        1     4     2     2
2  ...
209       0    0   17        0        0        1     4     3     4
2  ...
388       0    0   18        1        0        1     3     3     3
3  ...
267       0    0   18        0        0        1     4     4     4
2  ...
215       0    0   17        1        1        1     3     2     2
2  ...

     famrel  freetime  goout  Dalc  Walc  health  absences  G1  G2  G3

139       4         3      2     1     1       5         0  16  16  15

357       4         3      4     1     1       5         2  14  15  17

347       5         4      5     2     3       5         0  10  10   9

258       5         2      4     1     2       4         8  15  14  14

15        4         4      4     1     2       2         4  14  14  14

..      ...       ...    ...   ...   ...     ...       ...  ..  ..  ..
```

```
302        4         3      3     1     1       3         0  15  12  14

209        4         4      2     1     1       4         6   7   7   7

388        5         3      4     1     1       4         8  10  11  12

267        4         3      4     2     2       4         8  12  10  11

215        4         4      4     1     3       1         2  14  15  15


[225 rows x 33 columns]
```

```
from sklearn.ensemble import BaggingClassifier
from sklearn.multiclass import OneVsRestClassifier
from sklearn.svm import SVC
svmcla =
OneVsRestClassifier(BaggingClassifier(SVC(C=10,kernel='rbf',random_sta
te=9, probability=True),
                                               n_jobs=-1))

svmcla.fit(X_train, Y_train)


Y_predict2 = svmcla.predict(X_test)

test_acc_svmcla = round(svmcla.fit(X_train,Y_train).score(X_test,
Y_test)* 100, 2)
train_acc_svmcla = round(svmcla.fit(X_train, Y_train).score(X_train,
Y_train)* 100, 2)

# The confusion matrix
svmcla = confusion_matrix(Y_test, Y_predict2)
f, ax = plt.subplots(figsize=(5,5))
sns.heatmap(svmcla, annot=True, linewidth=0.7, linecolor='black',
fmt='g', ax=ax, cmap="BuPu")
plt.title('SVM Classification Confusion Matrix')
plt.xlabel('Y predict')
plt.ylabel('Y test')
plt.show()
```



```
model = pd.DataFrame({
    'Model': ['SVM'],
    'Train Score': [train_acc_svmcla],
    'Test Score': [test_acc_svmcla]
})
model.sort_values(by='Test Score', ascending=False)
```

```
  Model  Train Score  Test Score
0   SVM        82.22        80.0
```

```
from sklearn.metrics import average_precision_score
average_precision = average_precision_score(Y_test, Y_predict2)

print('Average precision-recall score: {0:0.2f}'.format(
      average_precision))
```

```
Average precision-recall score: 0.76
```

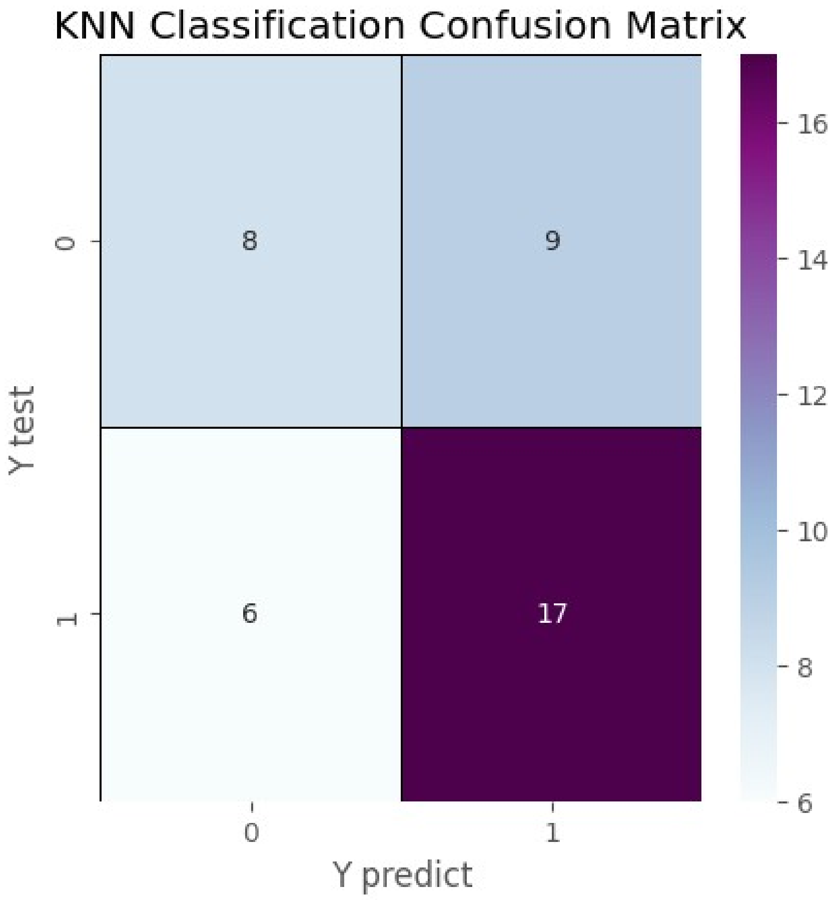
```
from sklearn.ensemble import BaggingClassifier
from sklearn.multiclass import OneVsRestClassifier
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train, Y_train)
```

```
y_pred = knn.predict(X_test)
y_pred
```

```
array([1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1,
0,
       1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1])
```

```
test_acc_svmcla = round(knn.fit(X_train,Y_train).score(X_test,
Y_test)* 100, 2)
train_acc_svmcla = round(knn.fit(X_train, Y_train).score(X_train,
Y_train)* 100, 2)
```

```
knn = confusion_matrix(Y_test, y_pred)
f, ax = plt.subplots(figsize=(5,5))
sns.heatmap(knn, annot=True, linewidth=0.7, linecolor='black',
fmt='g', ax=ax, cmap="BuPu")
plt.title('KNN Classification Confusion Matrix')
plt.xlabel('Y predict')
plt.ylabel('Y test')
plt.show()
```



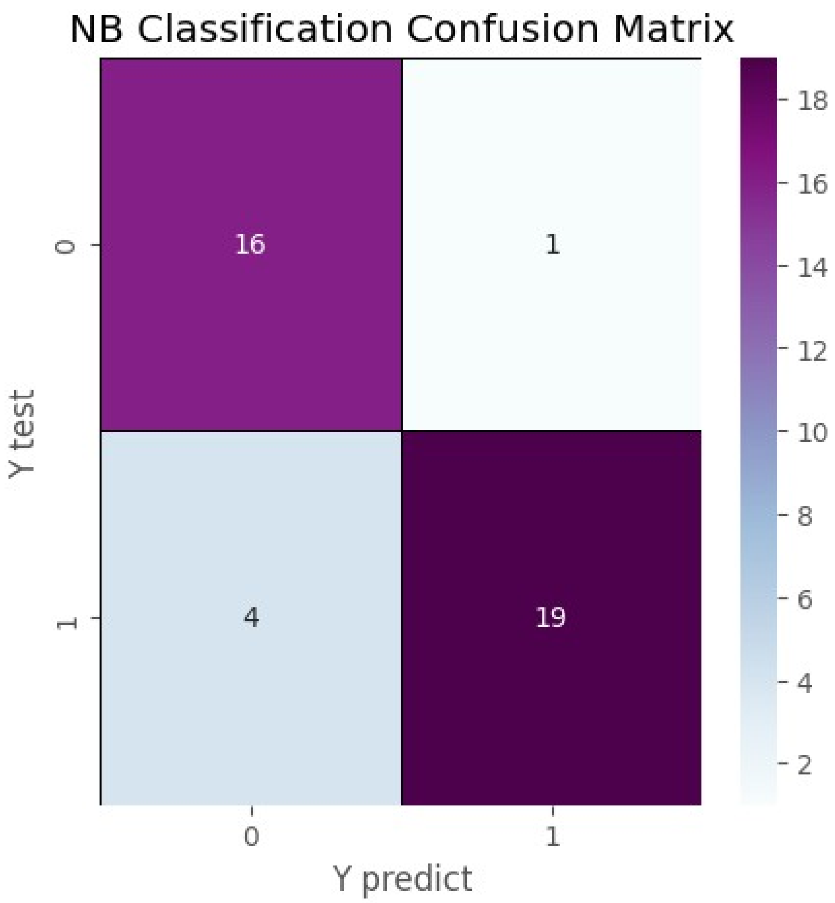
```
model = pd.DataFrame({
    'Model': ['KNN'],
    'Train Score': [train_acc_svmcla],
    'Test Score': [test_acc_svmcla]
})
model.sort_values(by='Test Score', ascending=False)
```

```
  Model  Train Score  Test Score
0   KNN         76.0        62.5
```

```
from sklearn.metrics import average_precision_score
average_precision = average_precision_score(Y_test, y_pred)

print('Average precision-recall score: {0:0.2f}'.format(
      average_precision))
```

```
Average precision-recall score: 0.63
```

```
from sklearn.ensemble import BaggingClassifier
from sklearn.multiclass import OneVsRestClassifier
from sklearn.naive_bayes import GaussianNB
nb = GaussianNB()
nb.fit(X_train, Y_train)
y_pred1 = nb.predict(X_test)

test_acc_svmcla = round(nb.fit(X_train,Y_train).score(X_test, Y_test)*
100, 2)
train_acc_svmcla = round(nb.fit(X_train, Y_train).score(X_train,
Y_train)* 100, 2)

nb = confusion_matrix(Y_test, y_pred1)
f, ax = plt.subplots(figsize=(5,5))
sns.heatmap(nb, annot=True, linewidth=0.7, linecolor='black', fmt='g',
ax=ax, cmap="BuPu")
plt.title('NB Classification Confusion Matrix')
plt.xlabel('Y predict')
plt.ylabel('Y test')
plt.show()
```



```
model = pd.DataFrame({
    'Model': ['NB'],
    'Train Score': [train_acc_svmcla],
    'Test Score': [test_acc_svmcla]
})
model.sort_values(by='Test Score', ascending=False)
```

```
  Model  Train Score  Test Score
0    NB        75.11        87.5
```

```
from sklearn.metrics import average_precision_score
average_precision = average_precision_score(Y_test, y_pred1)

print('Average precision-recall score: {0:0.2f}'.format(
      average_precision))
```

```
Average precision-recall score: 0.88
```