Hogwart's Sorting Hat Algorithm

The aim of this project was to build a machine-learning-powered Harry Potter's Sorting Hat that could tell which Hogwarts House you belong to based on given features. In this notebook I've implemented several multi-class classification algorithms in Python.

In [173…

```
pip install seaborn
```

```
Requirement already satisfied: seaborn in /srv/conda/envs/notebook/lib/python
3.6/site-packages (0.11.1)
Requirement already satisfied: pandas>=0.23 in /srv/conda/envs/notebook/lib/py
thon3.6/site-packages (from seaborn) (1.1.5)
Requirement already satisfied: scipy>=1.0 in /srv/conda/envs/notebook/lib/pyth
on3.6/site-packages (from seaborn) (1.5.3)
Requirement already satisfied: numpy>=1.15 in /srv/conda/envs/notebook/lib/pyt
hon3.6/site-packages (from seaborn) (1.19.5)
Requirement already satisfied: matplotlib>=2.2 in /srv/conda/envs/notebook/li
b/python3.6/site-packages (from seaborn) (3.3.4)
Requirement already satisfied: python-dateutil>=2.1 in /srv/conda/envs/noteboo
k/lib/python3.6/site-packages (from matplotlib>=2.2->seaborn) (2.8.1)
Requirement already satisfied: cycler>=0.10 in /srv/conda/envs/notebook/lib/py
thon3.6/site-packages/cycler-0.10.0-py3.6.egg (from matplotlib>=2.2->seaborn)
(0.10.0)
Requirement already satisfied: pillow>=6.2.0 in /srv/conda/envs/notebook/lib/p
ython3.6/site-packages (from matplotlib>=2.2->seaborn) (8.2.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in /sr
v/conda/envs/notebook/lib/python3.6/site-packages (from matplotlib>=2.2->seabo
rn) (2.4.7)
Requirement already satisfied: kiwisolver>=1.0.1 in /srv/conda/envs/notebook/l
ib/python3.6/site-packages (from matplotlib>=2.2->seaborn) (1.3.1)
Requirement already satisfied: six in /srv/conda/envs/notebook/lib/python3.6/s
ite-packages (from cycler>=0.10->matplotlib>=2.2->seaborn) (1.15.0)
Requirement already satisfied: pytz>=2017.2 in /srv/conda/envs/notebook/lib/py
thon3.6/site-packages (from pandas>=0.23->seaborn) (2021.1)
Note: you may need to restart the kernel to use updated packages.
```

In [174…

```
pip install statsmodels
```

```
Requirement already satisfied: statsmodels in /srv/conda/envs/notebook/lib/pyt
hon3.6/site-packages (0.12.2)
Requirement already satisfied: pandas>=0.21 in /srv/conda/envs/notebook/lib/py
thon3.6/site-packages (from statsmodels) (1.1.5)
Requirement already satisfied: scipy>=1.1 in /srv/conda/envs/notebook/lib/pyth
on3.6/site-packages (from statsmodels) (1.5.3)
Requirement already satisfied: patsy>=0.5 in /srv/conda/envs/notebook/lib/pyth
on3.6/site-packages (from statsmodels) (0.5.1)
Requirement already satisfied: numpy>=1.15 in /srv/conda/envs/notebook/lib/pyt
hon3.6/site-packages (from statsmodels) (1.19.5)
Requirement already satisfied: python-dateutil>=2.7.3 in /srv/conda/envs/noteb
ook/lib/python3.6/site-packages (from pandas>=0.21->statsmodels) (2.8.1)
Requirement already satisfied: pytz>=2017.2 in /srv/conda/envs/notebook/lib/py
thon3.6/site-packages (from pandas>=0.21->statsmodels) (2021.1)
Requirement already satisfied: six in /srv/conda/envs/notebook/lib/python3.6/s
ite-packages (from patsy>=0.5->statsmodels) (1.15.0)
Note: you may need to restart the kernel to use updated packages.
```

In [175…

```
import pandas as pd
import numpy as np

# data visualization
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# machine learning
from sklearn.preprocessing import StandardScaler

import sklearn.linear_model as skl_lm
from sklearn import preprocessing
from sklearn import neighbors
from sklearn.metrics import confusion_matrix, classification_report, precisio
from sklearn.model_selection import train_test_split


import statsmodels.api as sm
import statsmodels.formula.api as smf


# initialize some package settings
sns.set(style="whitegrid", color_codes=True, font_scale=1.3)

%matplotlib inline
```

In [176…

```
df = pd.read_csv('dataset.csv') #importing dataset
```

In [177…

```
df
```

Out[177…

	Index	Hogwarts House	First Name	Last Name	Birthday	Best Hand	Arithmancy	Astronomy	Her
0	0	Ravenclaw	Tamara	Hsu	2000-03-30	Left	58384.0	-487.886086	5.
1	1	Slytherin	Erich	Paredes	1999-10-14	Right	67239.0	-552.060507	-5.9
2	2	Ravenclaw	Stephany	Braun	1999-11-03	Left	23702.0	-366.076117	7.
3	3	Gryffindor	Vesta	Mcmichael	2000-08-19	Left	32667.0	697.742809	-6.
4	4	Gryffindor	Gaston	Gibbs	1998-09-27	Left	60158.0	436.775204	-7.8
...	...	...	...	...	...	...	...	...	
1595	1595	Gryffindor	Jung	Blank	2001-09-14	Right	49009.0	354.280086	-4.5
1596	1596	Slytherin	Shelli	Lock	1998-03-12	Left	63296.0	367.531174	6.(
1597	1597	Gryffindor	Benjamin	Christensen	1999-10-24	Right	63905.0	544.018925	-3.2
1598	1598	Hufflepuff	Charlotte	Dillon	2001-09-21	Left	82713.0	453.676219	3.
1599	1599	Hufflepuff	Kylie	Nowak	2000-08-21	Left	48639.0	688.911989	5.

1600 rows × 19 columns

In [178…

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1600 entries, 0 to 1599
Data columns (total 19 columns):
 #   Column                         Non-Null Count  Dtype  
---  ------                         --------------  -----  
 0   Index                          1600 non-null   int64  
 1   Hogwarts House                 1600 non-null   object 
 2   First Name                     1600 non-null   object 
 3   Last Name                      1600 non-null   object 
 4   Birthday                       1600 non-null   object 
 5   Best Hand                      1600 non-null   object 
 6   Arithmancy                     1566 non-null   float64
 7   Astronomy                      1568 non-null   float64
 8   Herbology                      1567 non-null   float64
 9   Defense Against the Dark Arts  1569 non-null   float64
 10  Divination                     1561 non-null   float64
 11  Muggle Studies                 1565 non-null   float64
 12  Ancient Runes                  1565 non-null   float64
 13  History of Magic               1557 non-null   float64
 14  Transfiguration                1566 non-null   float64
 15  Potions                        1570 non-null   float64
 16  Care of Magical Creatures      1560 non-null   float64
 17  Charms                         1600 non-null   float64
 18  Flying                         1600 non-null   float64
dtypes: float64(13), int64(1), object(5)
memory usage: 237.6+ KB
```

In [179…

```
#filling up the missing values using the averages of the columns
df['Arithmancy'].fillna((df['Arithmancy'].mean()), inplace=True)
df['Astronomy'].fillna((df['Astronomy'].mean()), inplace=True)
df['Herbology'].fillna((df['Herbology'].mean()), inplace=True)
df['Defense Against the Dark Arts'].fillna((df['Defense Against the Dark Arts
df['Divination'].fillna((df['Divination'].mean()), inplace=True)
df['Muggle Studies'].fillna((df['Muggle Studies'].mean()), inplace=True)
df['Ancient Runes'].fillna((df['Ancient Runes'].mean()), inplace=True)
df['History of Magic'].fillna((df['History of Magic'].mean()), inplace=True)
df['Transfiguration'].fillna((df['Transfiguration'].mean()), inplace=True)
df['Potions'].fillna((df['Potions'].mean()), inplace=True)
df['Care of Magical Creatures'].fillna((df['Care of Magical Creatures'].mean(
```

In [180…

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1600 entries, 0 to 1599
Data columns (total 19 columns):
 #   Column                         Non-Null Count  Dtype  
---  ------                         --------------  -----  
 0   Index                          1600 non-null   int64  
 1   Hogwarts House                 1600 non-null   object 
 2   First Name                     1600 non-null   object 
 3   Last Name                      1600 non-null   object 
 4   Birthday                       1600 non-null   object 
 5   Best Hand                      1600 non-null   object 
 6   Arithmancy                     1600 non-null   float64
 7   Astronomy                      1600 non-null   float64
 8   Herbology                      1600 non-null   float64
 9   Defense Against the Dark Arts  1600 non-null   float64
 10  Divination                     1600 non-null   float64
 11  Muggle Studies                 1600 non-null   float64
 12  Ancient Runes                  1600 non-null   float64
 13  History of Magic               1600 non-null   float64
 14  Transfiguration                1600 non-null   float64
 15  Potions                        1600 non-null   float64
 16  Care of Magical Creatures      1600 non-null   float64
 17  Charms                         1600 non-null   float64
```

```
 18  Flying                          1600 non-null   float64
dtypes: float64(13), int64(1), object(5)
memory usage: 237.6+ KB
```

In [181…

```
df.dtypes
```

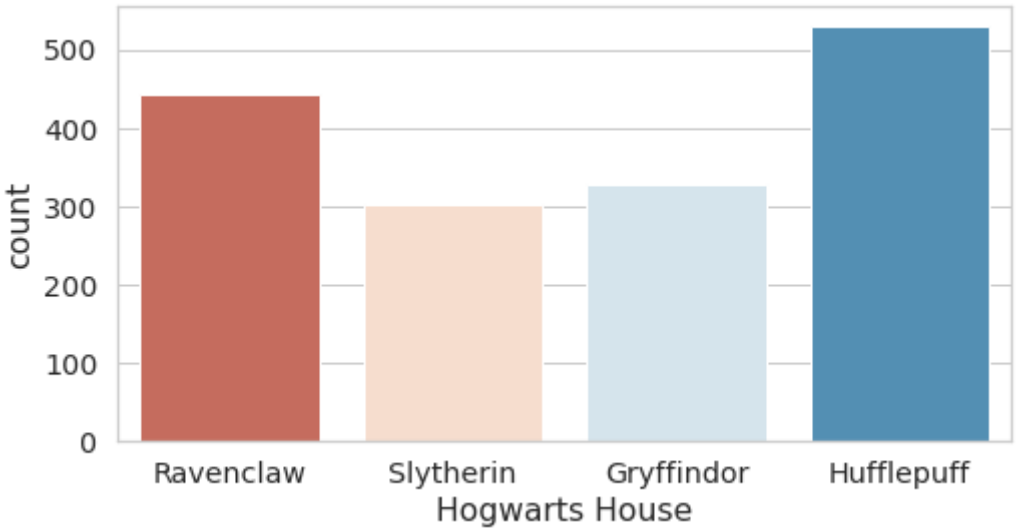
Out[181…

```
Index                              int64
Hogwarts House                    object
First Name                        object
Last Name                         object
Birthday                          object
Best Hand                         object
Arithmancy                       float64
Astronomy                        float64
Herbology                        float64
Defense Against the Dark Arts    float64
Divination                       float64
Muggle Studies                   float64
Ancient Runes                    float64
History of Magic                 float64
Transfiguration                  float64
Potions                          float64
Care of Magical Creatures        float64
Charms                           float64
Flying                           float64
dtype: object
```

In [182…

```
# visualize distribution of classes
plt.figure(figsize=(8, 4))
sns.countplot(df['Hogwarts House'], palette='RdBu')

# count number of obvs in each class
Ravenclaw, Slytherin, Gryffindor, Hufflepuff = df['Hogwarts House'].value_cou
print('Number of students in Ravenclaw: ', Ravenclaw)
print('Number of students in Slytherin: ', Slytherin)
print('Number of students in Gryffindor: ', Gryffindor)
print('Number of students in Hufflepuff: ', Hufflepuff)
print('')
print('% of cells labeled Ravenclaw', round(Ravenclaw / len(df) * 100, 2), '%
print('% of cells labeled Slytherin', round(Slytherin / len(df) * 100, 2), '%
print('% of cells labeled Gryffindor', round(Gryffindor / len(df) * 100, 2),
print('% of cells labeled Hufflepuff', round(Hufflepuff / len(df) * 100, 2),
```

```
Number of students in Ravenclaw:  529
Number of students in Slytherin:  443
Number of students in Gryffindor:  327
Number of students in Hufflepuff:  301

% of cells labeled Ravenclaw 33.06 %
% of cells labeled Slytherin 27.69 %
% of cells labeled Gryffindor 20.44 %
% of cells labeled Hufflepuff 18.81 %
/srv/conda/envs/notebook/lib/python3.6/site-packages/seaborn/_decorators.py:4
3: FutureWarning: Pass the following variable as a keyword arg: x. From versio
n 0.12, the only valid positional argument will be `data`, and passing other a
rguments without an explicit keyword will result in an error or misinterpretat
ion.
  FutureWarning
```

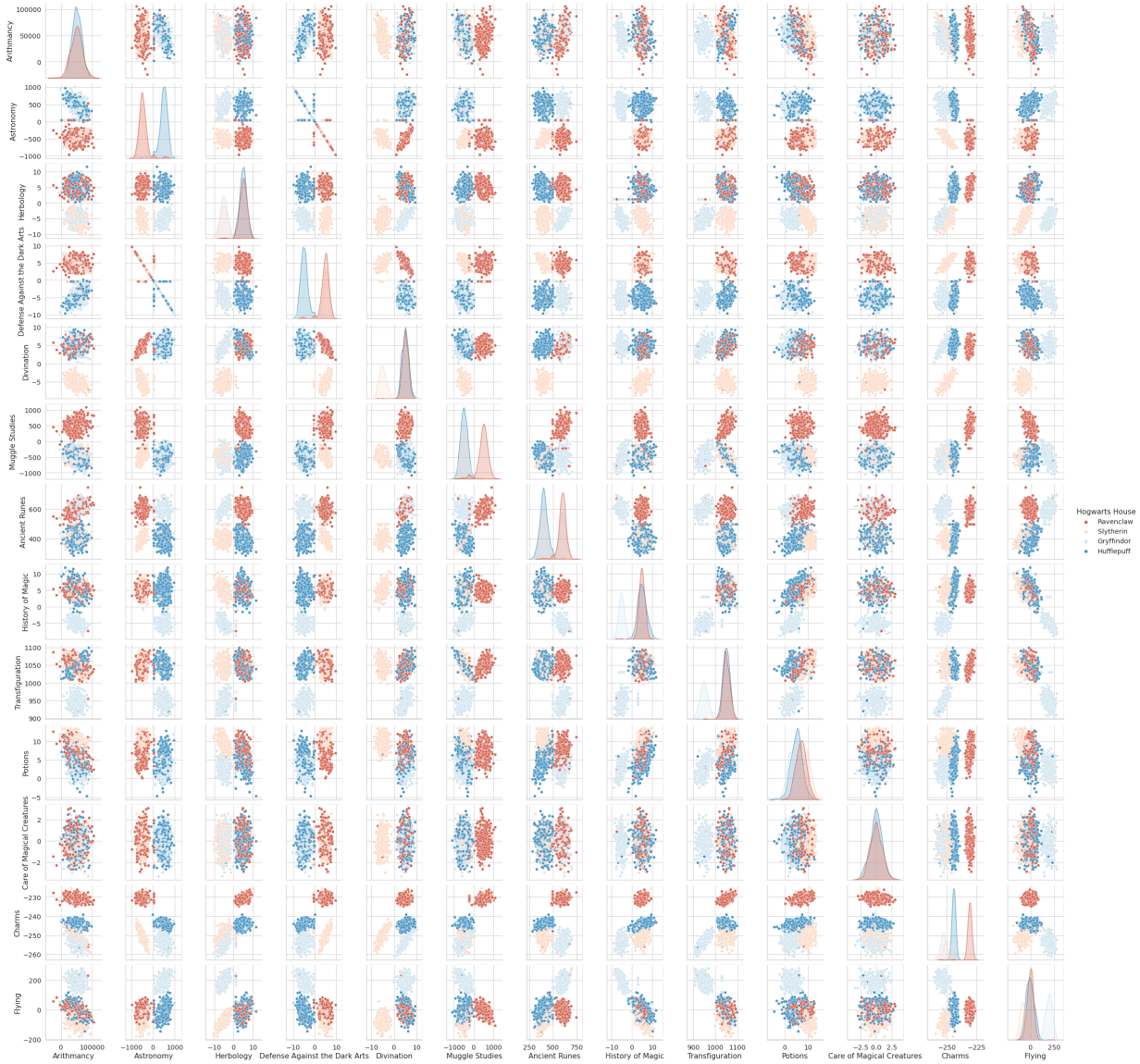


In [183…

```
# generate a scatter plot matrix with the columns
cols = ['Hogwarts House',
    'Best Hand',
        'Arithmancy',
        'Astronomy',
        'Herbology',
        'Defense Against the Dark Arts',
        'Divination',
        'Muggle Studies',
        'Ancient Runes',
        'History of Magic',
        'Transfiguration',
       'Potions',
       'Care of Magical Creatures',
       'Charms',
       'Flying']

sns.pairplot(data=df[cols], hue='Hogwarts House', palette='RdBu')
```

Out[183…

```
<seaborn.axisgrid.PairGrid at 0x7fd8656aada0>
```

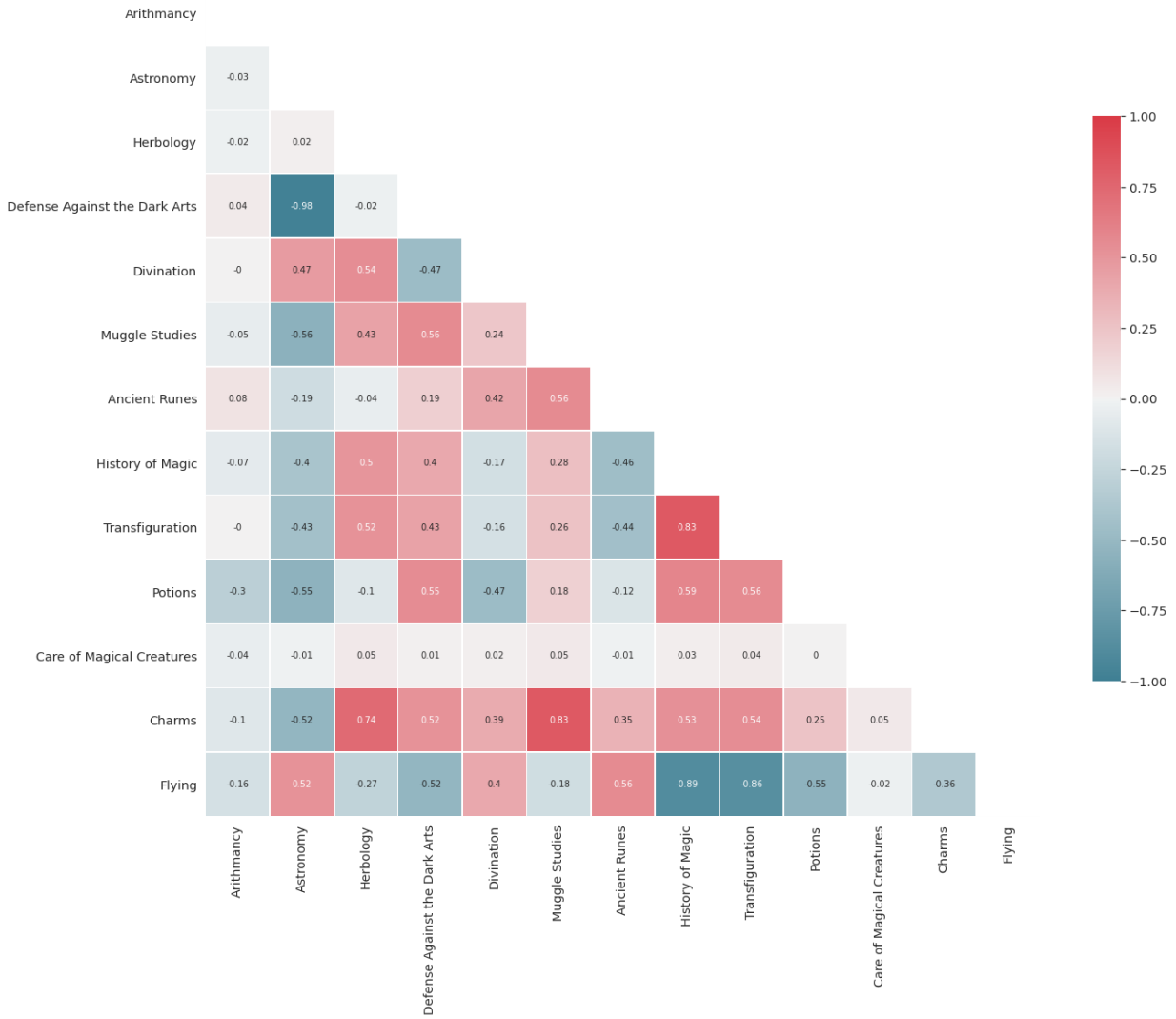


In [184…

```
df = df.drop('Index', axis=1) #removing index column
```

In [185…

```
# Generate and visualize the correlation matrix
corr = df.corr().round(2)

# Mask for the upper triangle
mask = np.zeros_like(corr, dtype=np.bool)
mask[np.triu_indices_from(mask)] = True

# Set figure size
f, ax = plt.subplots(figsize=(20, 20))

# Define custom colormap
cmap = sns.diverging_palette(220, 10, as_cmap=True)

# Draw the heatmap
sns.heatmap(corr, mask=mask, cmap=cmap, vmin=-1, vmax=1, center=0,
            square=True, linewidths=.5, cbar_kws={"shrink": .5}, annot=True)

plt.tight_layout()
```



In [186…
```
#for label encoding
df['Hogwarts House'].unique()
```

Out[186… array(['Ravenclaw', 'Slytherin', 'Gryffindor', 'Hufflepuff'], dtype=object)

In [187…
```
# Import label encoder
from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()
df['Hogwarts House']= label_encoder.fit_transform(df['Hogwarts House'])
df['Hogwarts House'].unique()
```

Out[187… array([2, 3, 0, 1])

Gryffindor-0 Hufflepuff-1 Ravenclaw-2 Slytherin-3

In [188…
```
df['Best Hand'].unique()
```

Out[188… array(['Left', 'Right'], dtype=object)

In [189…
```
df['Best Hand']= label_encoder.fit_transform(df['Best Hand'])
df['Best Hand'].unique()
```

Out[189… array([0, 1])

Left- 0 Right- 1

In [190…
```
df = df.drop('First Name', axis=1) #removing unecessary data
df = df.drop('Last Name', axis=1)
df = df.drop('Birthday', axis=1)
```

In [191…
```
# Split the data into training and testing sets
X = df
y = df['Hogwarts House']

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)
```

Support vector machine classifier

In [192…
```
# training a linear SVM classifier
from sklearn.svm import SVC
svm_model_linear = SVC(kernel = 'linear', C = 1).fit(X_train, y_train)
svm_predictions = svm_model_linear.predict(X_test)
```

In [193…
```
# model accuracy for X_test
accuracy = svm_model_linear.score(X_test, y_test)

# creating a confusion matrix
cm = confusion_matrix(y_test, svm_predictions)
```

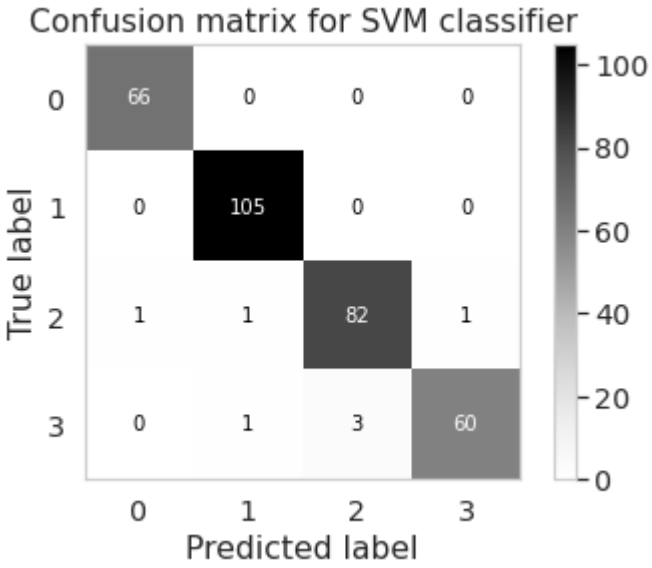
In [194…
```
print(accuracy)
```

```
0.978125
```

In [195…
```
print(cm)
```

```
[[ 66   0   0   0]
 [  0 105   0   0]
 [  1   1  82   1]
 [  0   1   3  60]]
```

In [196…
```
from sklearn.metrics import plot_confusion_matrix
matrix = plot_confusion_matrix(svm_model_linear, X_test, y_test,cmap=plt.cm.G
plt.title('Confusion matrix for SVM classifier')
plt.grid(False)
plt.show()
```



Decision tree classifier

In [197…
```
from sklearn.tree import DecisionTreeClassifier
```

In [198…
```
dtree_model = DecisionTreeClassifier(max_depth = 2).fit(X_train, y_train)
dtree_predictions = dtree_model.predict(X_test)
```

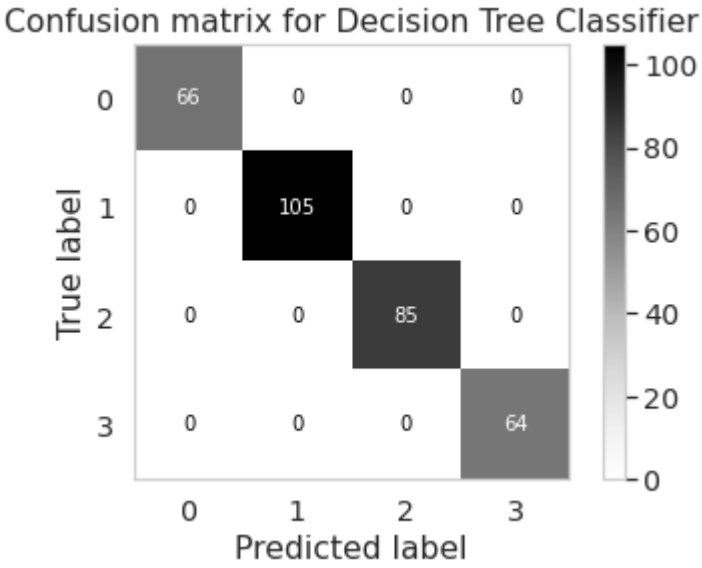
In [199…
```
cm1 = confusion_matrix(y_test, dtree_predictions)
cm1
```

Out[199…
```
array([[ 66,   0,   0,   0],
       [  0, 105,   0,   0],
       [  0,   0,  85,   0],
       [  0,   0,   0,  64]])
```

In [200…
```
accuracy1 = dtree_model.score(X_test, y_test)
accuracy1
```

Out[200…
```
1.0
```

In [201…
```
from sklearn.metrics import plot_confusion_matrix
matrix = plot_confusion_matrix(dtree_model, X_test, y_test,cmap=plt.cm.Greys)
plt.title('Confusion matrix for Decision Tree Classifier')
plt.grid(False)
plt.show()
```



KNN classifier

In [202…
```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors = 4).fit(X_train, y_train)
```

In [203…
```
accuracy2 = knn.score(X_test, y_test)
print (accuracy2)
```

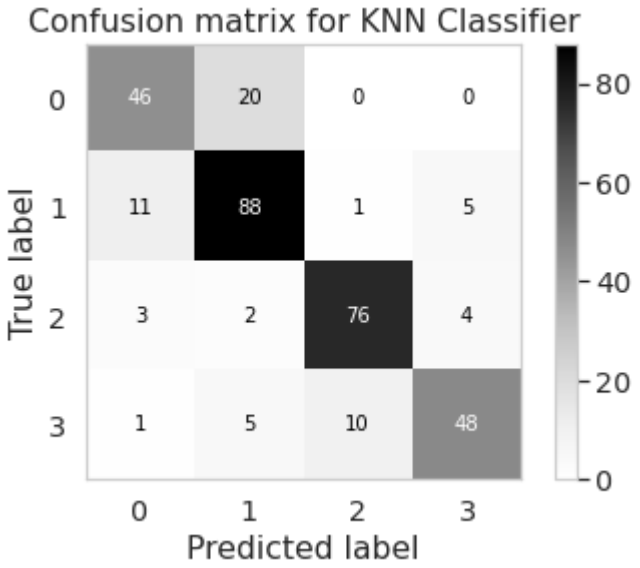
```
0.80625
```

In [204…
```
knn_predictions = knn.predict(X_test)
cm2 = confusion_matrix(y_test, knn_predictions)
```

In [205…
```
cm2
```

Out[205…
```
array([[46, 20,  0,  0],
       [11, 88,  1,  5],
       [ 3,  2, 76,  4],
       [ 1,  5, 10, 48]])
```

In [206…
```
matrix = plot_confusion_matrix(knn, X_test, y_test,cmap=plt.cm.Greys)
plt.title('Confusion matrix for KNN Classifier')
plt.grid(False)
plt.show()
```


Confusion matrix for KNN Classifier
True label
Predicted label
0
1
2
3
46
20
0
0
11
88
1
5
3
2
76
4
1
5
10
48
80
60
40
20
0