In [52]:
```
pwd
```

Out[52]:
```
'C:\\Users\\Sauda Maryam\\Documents'
```

In [53]:
```
import pandas as pd
import re
import string
import scipy
import numpy as np
import seaborn as sns
from sklearn import preprocessing

from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import BernoulliNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import LinearSVC
from sklearn.metrics import accuracy_score

from astropy.table import Table, Column
import matplotlib.pyplot as plt
```

Step 2: Read, Understand & Pre-process Train/Test Data

In [54]:
```
train_data = pd.read_csv("dataset/Gender_Identification_train.csv")
test_data = pd.read_csv("dataset/Gender_Identification_test.csv")
```

Train Data Set

In [55]:
```
train_data.head()
```

Out[55]:

	height	weight	hair	beard	scarf	gender
0	180.3000	196	Bald	Yes	No	Male
1	170.0000	120	Long	No	No	Female
2	178.5000	200	Short	No	No	Male
3	163.4000	110	Medium	No	Yes	Female
4	175.2222	220	Short	Yes	No	Male

Attribute Values of Train DataSet

In [56]:
```
train_data.columns
```

Out[56]:
```
Index(['height', 'weight', 'hair', 'beard', 'scarf', 'gender'], dtype='object')
```

Number Of Instances in Train DataSet

In [57]:
```
print("Number of instances in Train Dataset:")
print("-"*42)
print("-"*42)
print("Train Dataset:",len(train_data))
```

```
Number of instances in Train Dataset:
------------------------------------------
------------------------------------------
Train Dataset: 6
```

In [58]:
```
train_data.dtypes
```

Out[58]:
```
height    float64
weight      int64
hair       object
beard      object
scarf      object
gender     object
dtype: object
```

Test Data Set

In [59]:
```
test_data.head()
```

Out[59]:

	height	weight	hair	beard	scarf	gender
0	179.1	185	Long	Yes	No	Male
1	160.5	130	Short	No	No	Female
2	177.8	160	Bald	No	No	Male
3	161.1	100	Medium	No	No	Female

Attribute Values of Test DataSet

In [60]:
```
test_data.columns
```

Out[60]:
```
Index(['height', 'weight', 'hair', 'beard', 'scarf', 'gender'], dtype='object')
```

Number Of Instances in Test DataSet

In [61]:
```
print("Number of instances in Test Dataset:")
print("-"*42)
print("-"*42)
print("Train Dataset:",len(test_data))
```

```
Number of instances in Test Dataset:
------------------------------------------
------------------------------------------
Train Dataset: 4
```

In [62]:
```
test_data.dtypes
```

Out[62]:
```
height    float64
weight      int64
hair       object
beard      object
scarf      object
gender     object
dtype: object
```

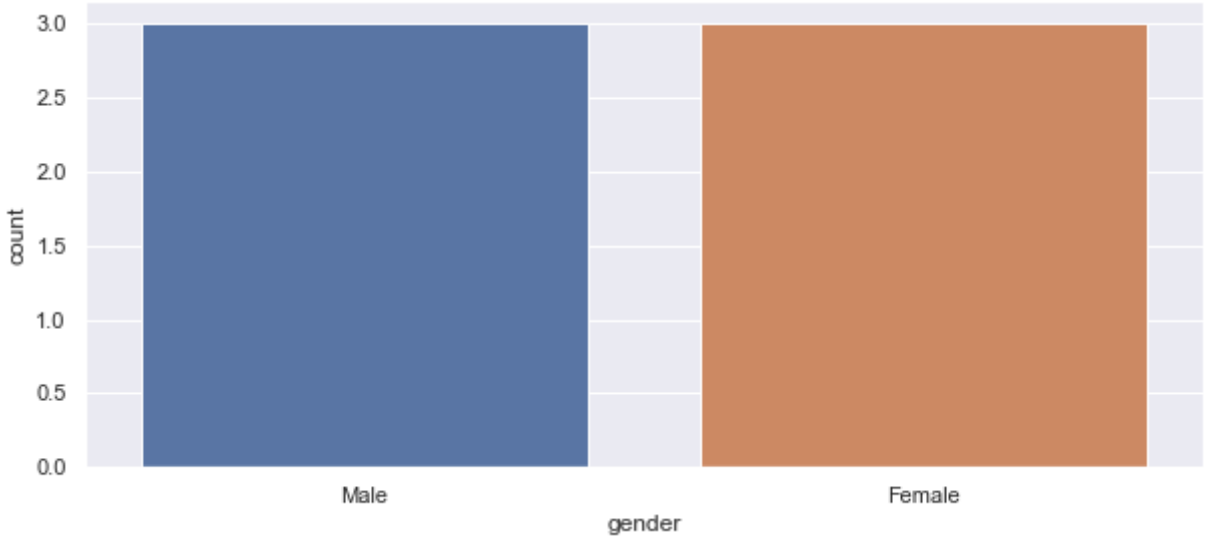
Total number of 'Males' and 'Females' in Train Dataset

In [63]:
```
print("Total n.o of MALES and FEMALES in train data set")

sns.countplot("gender" , data= train_data)
```

```
Total n.o of MALES and FEMALES in train data set
```

Out[63]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2010b27bef0>
```

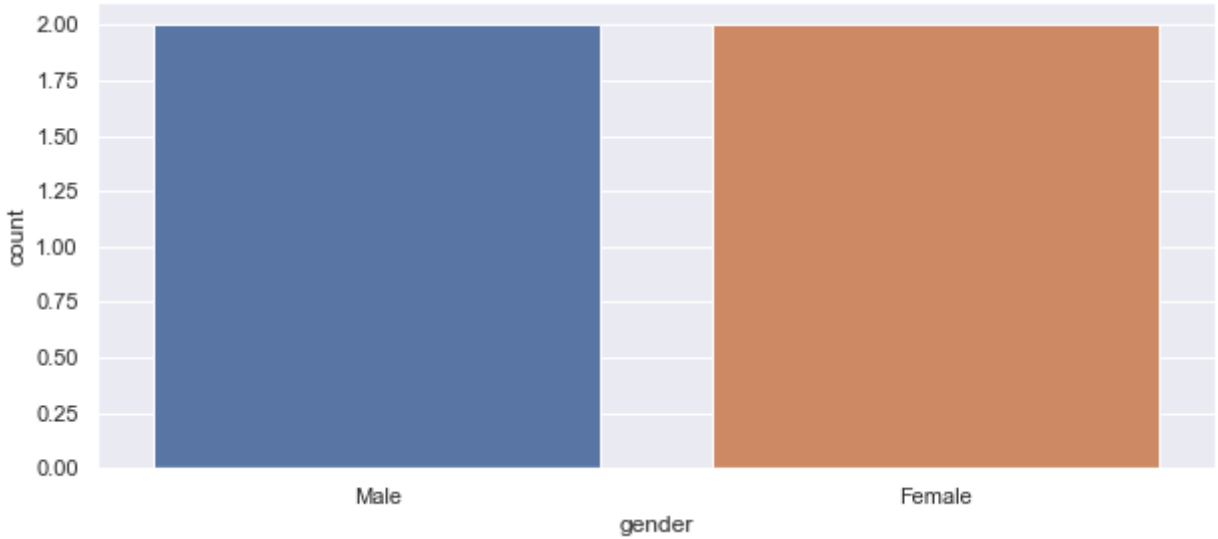


Total number of 'Males' and 'Females' in Test Dataset

In [113]:
```
print("Total n.o of MALES and FEAMLES in test data set")
sns.countplot("gender",data=test_data)
```

```
Total n.o of MALES and FEAMLES in test data set
```

Out[113]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2010c61d390>
```



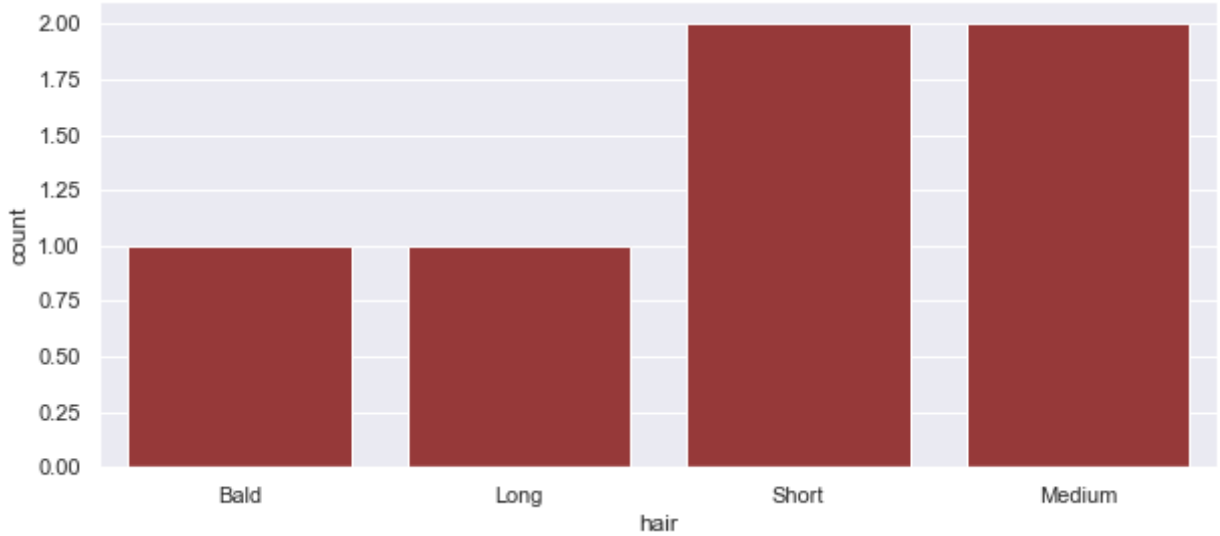
Number of people having various hair length in Train dataset:

In [65]:
```
print("N.o of people having varoius hair length in train dataset")

sns.countplot("hair",data=train_data,color = 'brown')
```

```
N.o of people having varoius hair length in train dataset
```

Out[65]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2010c47a6d8>
```

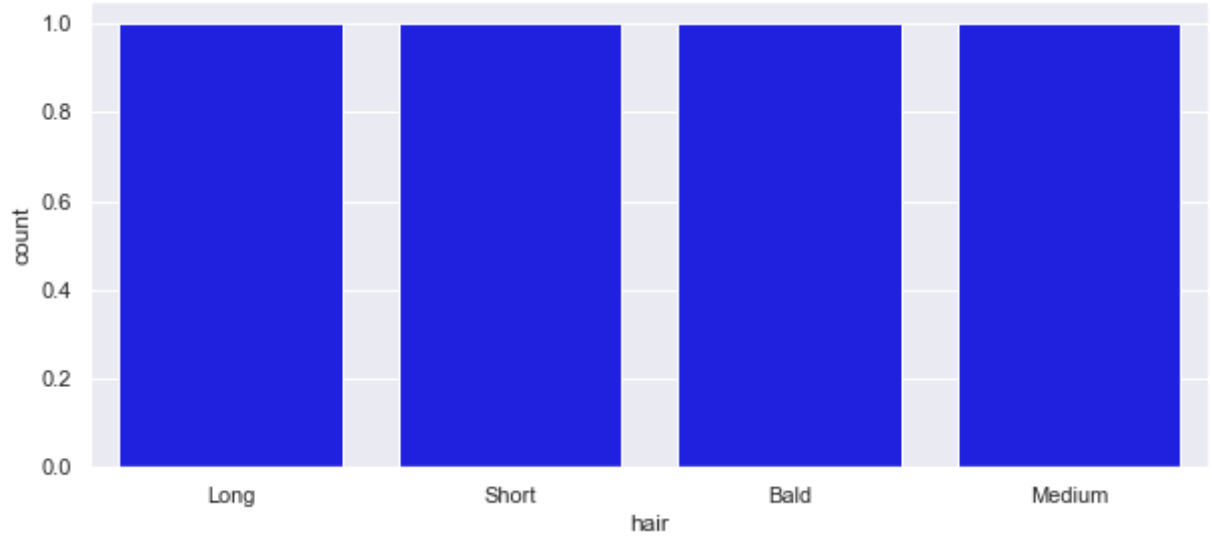


Number of people having various hair length in Test dataset:

In [66]:
```
print("N.o of people having varoius hair length in test dataset")
sns.set(rc={'figure.figsize':(10,4.27)})
sns.countplot("hair",data=test_data,color = 'blue')
```

```
N.o of people having varoius hair length in test dataset
```

Out[66]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2010c4e07b8>
```

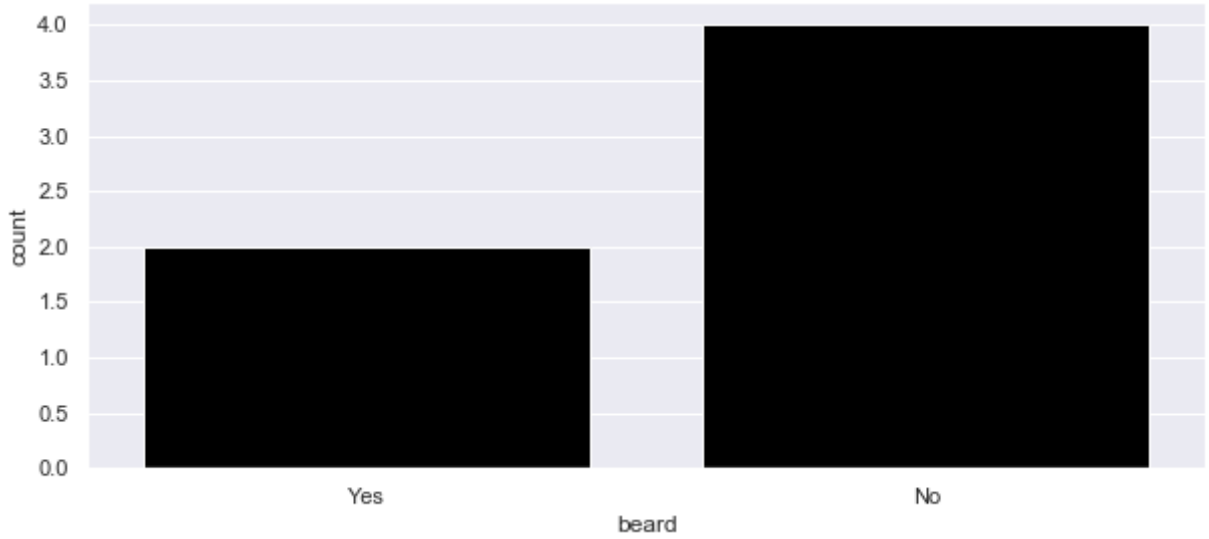


Number of people have/haven't beard in Train dataset:

In [67]:
```
print("N.o of people have/haven't beard in train dataset")
sns.set(rc={'figure.figsize':(10,4.27)})
sns.countplot("beard",data=train_data,color = 'black')
```

```
N.o of people have/haven't beard in train dataset
```

Out[67]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2010c52cb70>
```

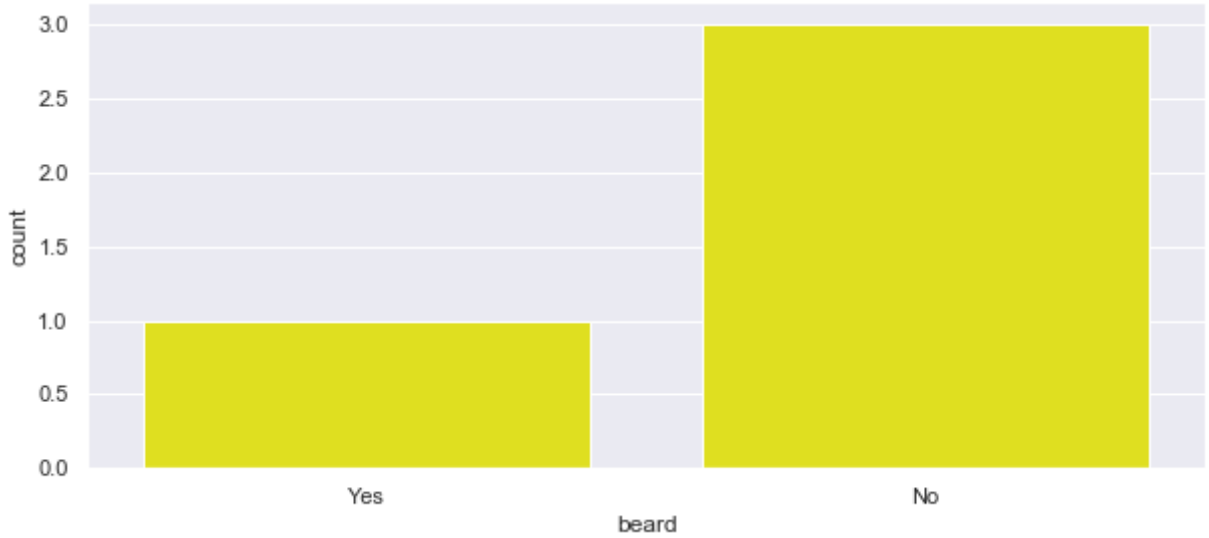


Number of people have/haven't beard in Train dataset:

In [68]:
```
print("N.o of people have/haven't beard in Test dataset")
sns.set(rc={'figure.figsize':(10,4.27)})
sns.countplot("beard",data=test_data,color = 'yellow')
```

```
N.o of people have/haven't beard in Test dataset
```

Out[68]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2010c5879b0>
```



Step 2.3: Pre-Process Data

Train data before pre-processing

In [69]:
```
print("Data before pre-processing :\n")
print(train_data,"\n")
```

```
Data before pre-processing :

     height  weight    hair beard scarf  gender
0  180.3000     196    Bald   Yes    No    Male
1  170.0000     120    Long    No    No  Female
2  178.5000     200   Short    No    No    Male
3  163.4000     110  Medium    No   Yes  Female
4  175.2222     220   Short   Yes    No    Male
5  165.0000     150  Medium    No   Yes  Female
```

Train data after pre-processing

In [70]:
```
train_data_pre = train_data.copy()

train_data_pre['height'] = round(train_data_pre['height'],2)
print("Data after pre-processing :\n")
print(train_data_pre)
train_data_pre.head()
```

```
Data after pre-processing :

   height  weight    hair beard scarf  gender
0  180.30     196    Bald   Yes    No    Male
1  170.00     120    Long    No    No  Female
2  178.50     200   Short    No    No    Male
3  163.40     110  Medium    No   Yes  Female
4  175.22     220   Short   Yes    No    Male
5  165.00     150  Medium    No   Yes  Female
```

Out[70]:

	height	weight	hair	beard	scarf	gender
0	180.30	196	Bald	Yes	No	Male
1	170.00	120	Long	No	No	Female
2	178.50	200	Short	No	No	Male
3	163.40	110	Medium	No	Yes	Female
4	175.22	220	Short	Yes	No	Male

In [ ]:
```

```

Step 3: Label Encoding for Test/Train Data

In [71]:
```
le_hair = preprocessing.LabelEncoder()
le_beard = preprocessing.LabelEncoder()
le_scarf = preprocessing.LabelEncoder()
le_gender = preprocessing.LabelEncoder()
```

Gender attribute encoding in train dataset

In [72]:
```
train_data['newencoded_gender'] = le_gender.fit_transform(train_data['gender'])

print("Gender attribute encoding in train dataset  :\n")
print(train_data[["gender","newencoded_gender"]])
```

```
Gender attribute encoding in train dataset  :

   gender  newencoded_gender
0    Male                  1
1  Female                  0
2    Male                  1
3  Female                  0
4    Male                  1
5  Female                  0
```

Scarf attribute encoding in train dataset

In [73]:
```
train_data['encoded_scarf'] = le_scarf.fit_transform(train_data['scarf'])

print("Scarf attribute encoding in train dataset  :\n")
print(train_data[["scarf","encoded_scarf"]])
```

```
Scarf attribute encoding in train dataset  :

  scarf  encoded_scarf
0    No              0
1    No              0
2    No              0
3   Yes              1
4    No              0
5   Yes              1
```

Beard attribute encoding in train dataset

In [74]:
```
train_data['en_beard'] = le_beard.fit_transform(train_data['beard'])

print("Beard attribute encoding in train dataset  :\n")
print(train_data[["beard","en_beard"]])
```

```
Beard attribute encoding in train dataset  :

  beard  en_beard
0   Yes         1
1    No         0
2    No         0
3    No         0
4   Yes         1
5    No         0
```

Hair attribute encoding in train dataset

In [75]:
```
train_data['en_hair'] = le_hair.fit_transform(train_data['hair'])

print("Hair attribute encoding in train dataset  :\n")
print(train_data[["hair","en_hair"]])
```

```
Hair attribute encoding in train dataset  :

     hair  en_hair
0    Bald        0
1    Long        1
2   Short        3
3  Medium        2
4   Short        3
5  Medium        2
```

Orignal Train Data Set

In [76]:
```
print("train dataset without encoding :\n")
print(train_data_pre,"\n")
```

```
train dataset without encoding :

   height  weight    hair beard scarf  gender
0  180.30     196    Bald   Yes    No    Male
1  170.00     120    Long    No    No  Female
2  178.50     200   Short    No    No    Male
3  163.40     110  Medium    No   Yes  Female
4  175.22     220   Short   Yes    No    Male
5  165.00     150  Medium    No   Yes  Female
```

Train Data after Label Encoding:

In [77]:
```
train_data_en = pd.read_csv("dataset/Gender_Identification_train.csv")
train_data_en['hair'] = le_hair.fit_transform(train_data_en['hair'])
train_data_en['beard'] = le_beard.fit_transform(train_data_en['beard'])
train_data_en['scarf'] = le_scarf.fit_transform(train_data_en['scarf'])
train_data_en['gender'] = le_gender.fit_transform(train_data_en['gender'])
train_data_en['height'] = round(train_data_en['height'],2)
print("train dataset with encoding :\n")
print(train_data_en)
```

```
train dataset with encoding :

   height  weight  hair  beard  scarf  gender
0  180.30     196     0      1      0       1
1  170.00     120     1      0      0       0
2  178.50     200     3      0      0       1
3  163.40     110     2      0      1       0
4  175.22     220     3      1      0       1
5  165.00     150     2      0      1       0
```

test dataset without encoding

In [78]:
```
print("test dataset without encoding :\n")
print(test_data,"\n")
```

```
test dataset without encoding :

   height  weight    hair beard scarf  gender
0   179.1     185    Long   Yes    No    Male
1   160.5     130   Short    No    No  Female
2   177.8     160    Bald    No    No    Male
3   161.1     100  Medium    No    No  Female
```

test dataset with encoding

In [79]:
```
test_data = test_data.copy()
test_data['hair'] = le_hair.fit_transform(test_data['hair'])
test_data['beard'] = le_beard.fit_transform(test_data['beard'])
test_data['scarf'] = le_scarf.fit_transform(test_data['scarf'])
test_data['gender'] = le_gender.fit_transform(test_data['gender'])
print("test dataset with encoding :\n")
print(test_data)
```

```
test dataset with encoding :

   height  weight  hair  beard  scarf  gender
0   179.1     185     1      1      0       1
1   160.5     130     3      0      0       0
2   177.8     160     0      0      0       1
3   161.1     100     2      0      0       0
```

Step 4: Feature Extraction – Changing Representation of Data “from String to Vector”

Step 5: Train Machine Learning Algorithms using Training Data

In [80]:
```
X_train = train_data_en[['height','weight','hair','beard','scarf']]
Y_train = train_data_en[['gender']]

X_test = test_data[['height','weight','hair','beard','scarf']]
Y_test = test_data[['gender']]
```

LogisticRegression

In [81]:
```
lr = LogisticRegression()
pred = lr.fit(X_train, Y_train).predict(X_test)
test_data['predicted_gender'] = pred

test_data['hair'] = le_hair.inverse_transform(test_data['hair'])
test_data['beard'] = le_beard.inverse_transform(test_data['beard'])
test_data['scarf'] = le_scarf.inverse_transform(test_data['scarf'])
test_data['gender'] = le_gender.inverse_transform(test_data['gender'])
test_data['predicted_gender'] = le_gender.inverse_transform(test_data['predicted_gender'])
print(test_data,"\n")

print("LogisticRegression accuracy : ",accuracy_score(Y_test, pred, normalize = True))
```

```
   height  weight    hair beard scarf  gender predicted_gender
0   179.1     185    Long   Yes    No    Male             Male
1   160.5     130   Short    No    No  Female           Female
2   177.8     160    Bald    No    No    Male           Female
3   161.1     100  Medium    No    No  Female           Female

LogisticRegression accuracy :  0.75

C:\Users\Sauda Maryam\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:432: FutureWarning: Default solver w
ill be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\Sauda Maryam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:724: DataConversionWarning: A column-vect
or y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

In [ ]:

LinearSVC

In [82]:
```
ls = LinearSVC()

pred_ls = ls.fit(X_train, Y_train).predict(X_test)
test_data['predicted_gender'] = pred_ls


print(test_data,"\n")

print("LinearSVC accuracy : ",accuracy_score(Y_test, pred_ls))
```

```
   height  weight    hair beard scarf  gender  predicted_gender
0   179.1     185    Long   Yes    No    Male                 1
1   160.5     130   Short    No    No  Female                 0
2   177.8     160    Bald    No    No    Male                 0
3   161.1     100  Medium    No    No  Female                 0

LinearSVC accuracy :  0.75

C:\Users\Sauda Maryam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:724: DataConversionWarning: A column-vect
or y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
C:\Users\Sauda Maryam\Anaconda3\lib\site-packages\sklearn\svm\base.py:929: ConvergenceWarning: Liblinear failed to conv
erge, increase the number of iterations.
  "the number of iterations.", ConvergenceWarning)
```

RandomForestClassifier

In [83]:
```
rfc = RandomForestClassifier()

pred_rfc = rfc.fit(X_train, Y_train).predict(X_test)
test_data['predicted_gender'] = pred_rfc


print(test_data,"\n")

print("RandomForestClassifier accuracy : ",accuracy_score(Y_test, pred_rfc))
```

```
   height  weight    hair beard scarf  gender  predicted_gender
0   179.1     185    Long   Yes    No    Male                 1
1   160.5     130   Short    No    No  Female                 0
2   177.8     160    Bald    No    No    Male                 1
3   161.1     100  Medium    No    No  Female                 0

RandomForestClassifier accuracy :  1.0

C:\Users\Sauda Maryam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:245: FutureWarning: The default value of n
_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\Sauda Maryam\Anaconda3\lib\site-packages\ipykernel_launcher.py:3: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  This is separate from the ipykernel package so we can avoid doing imports until
```

BernoulliNB

In [84]:
```
ber = BernoulliNB()

pred_ber = ber.fit(X_train, Y_train).predict(X_test)
test_data['predicted_gender'] = pred_ber

print(test_data,"\n")

print("BernoulliNB accuracy : ",accuracy_score(Y_test, pred_ber))
```

```
   height  weight    hair beard scarf  gender  predicted_gender
0   179.1     185    Long   Yes    No    Male                 1
1   160.5     130   Short    No    No  Female                 0
2   177.8     160    Bald    No    No    Male                 1
3   161.1     100  Medium    No    No  Female                 0

BernoulliNB accuracy :  1.0

C:\Users\Sauda Maryam\Anaconda3\lib\site-packages\sklearn\utils\validation.py:724: DataConversionWarning: A column-vect
or y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Selection of Best Model

In [85]:
```
print("Detailed performance of all the models")
print("====================================+")
print("+-----------------------------------+")
print("|        Model       |    Accuracy   |")
print("+-----------------------------------+")
print("| LogisticRegression |     ",accuracy_score(Y_test, pred, normalize = True),"    |")
print("| RandFrstClassifier |     ",accuracy_score(Y_test, pred_rfc),"     |")
print("|     LinearSVC      |     ",accuracy_score(Y_test, pred_ls),"    |")
print("|    BernoulliNB     |     ",accuracy_score(Y_test, pred_ber),"     |")
print("+-----------------------------------+")
print("")
print("                Best Model           ")
print("====================================+")
print("+-----------------------------------+")
print("|        Model       |    Accuracy   |")
print("+-----------------------------------+")
print("| RandFrstClassifier |     ",accuracy_score(Y_test, pred_rfc),"     |")
print("+-----------------------------------+")
```

```
Detailed performance of all the models
====================================+
+-----------------------------------+
|        Model       |    Accuracy   |
+-----------------------------------+
| LogisticRegression |      0.75     |
| RandFrstClassifier |      1.0      |
|     LinearSVC      |      0.75     |
|    BernoulliNB     |      1.0      |
+-----------------------------------+

                Best Model
====================================+
+-----------------------------------+
|        Model       |    Accuracy   |
+-----------------------------------+
| RandFrstClassifier |      1.0      |
+-----------------------------------+
```

In [86]:
```
print(train_data_en)
```

```
   height  weight  hair  beard  scarf  gender
0  180.30     196     0      1      0       1
1  170.00     120     1      0      0       0
2  178.50     200     3      0      0       1
3  163.40     110     2      0      1       0
4  175.22     220     3      1      0       1
5  165.00     150     2      0      1       0
```

In [87]:
```
test_data_en = pd.read_csv("dataset/Gender_Identification_train.csv")
test_data_en['hair'] = le_hair.fit_transform(test_data_en['hair'])
test_data_en['beard'] = le_beard.fit_transform(test_data_en['beard'])
test_data_en['scarf'] = le_scarf.fit_transform(test_data_en['scarf'])
test_data_en['gender'] = le_gender.fit_transform(test_data_en['gender'])
print("test dataset with encoding:")
print(test_data_en)
```

```
test dataset with encoding:
     height  weight  hair  beard  scarf  gender
0  180.3000     196     0      1      0       1
1  170.0000     120     1      0      0       0
2  178.5000     200     3      0      0       1
3  163.4000     110     2      0      1       0
4  175.2222     220     3      1      0       1
5  165.0000     150     2      0      1       0
```

In [88]:
```
combined_data = train_data_en.append(test_data_en)
print("Combinded Data (Train + Test) :")
print(combined_data)
```

```
Combinded Data (Train + Test) :
     height  weight  hair  beard  scarf  gender
0  180.3000     196     0      1      0       1
1  170.0000     120     1      0      0       0
2  178.5000     200     3      0      0       1
3  163.4000     110     2      0      1       0
4  175.2200     220     3      1      0       1
5  165.0000     150     2      0      1       0
0  180.3000     196     0      1      0       1
1  170.0000     120     1      0      0       0
2  178.5000     200     3      0      0       1
3  163.4000     110     2      0      1       0
4  175.2222     220     3      1      0       1
5  165.0000     150     2      0      1       0
```

In [89]:
```
X_train = combined_data[['height','weight','hair','beard','scarf']]
Y_train = combined_data[['gender']]
```

In [90]:
```
rfc = RandomForestClassifier()
pred_rfc = rfc.fit(X_train, Y_train)
```

```
C:\Users\Sauda Maryam\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:245: FutureWarning: The default value of n
_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
C:\Users\Sauda Maryam\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: DataConversionWarning: A column-vector y was
passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
```

Step 9: Make prediction on unseen/new data

Step 9.1: Load the Trained Model (saved in Step 8.3)

Step 9.2: Take Input from User

In [91]:
```
height = input ("Please enter your Height here (centimeters) :")
weight = input ("Please enter your Weight here (kg) :")
hair = input ("Please enter your Hair Length here (Bald/Long/Medium/Short) :")
beard = input ("Do you have beard? (yes/no) :")
scarf = input ("Do you wear scarf? (yes/no) :")
```

```
Please enter your Height here (centimeters) :180
Please enter your Weight here (kg) :60
Please enter your Hair Length here (Bald/Long/Medium/Short) :Long
Do you have beard? (yes/no) :no
Do you wear scarf? (yes/no) :yes
```

Step 9.3: Convert User Input into Feature Vector (Same as Feature Vector of Trained Model)

In [92]:
```
user_data = {
    'height' : [height],
    'weight' : [weight],
    'hair'   : [hair],
    'beard'  : [beard],
    'scarf'  : [scarf]
}

user_input = pd.DataFrame(user_data)
print("User input in actual DataFrame format:\n")
print(user_input)
```

```
User input in actual DataFrame format:

  height weight  hair beard scarf
0    180     60  Long    no   yes
```

In [93]:
```
user_input['hair'] = le_hair.fit_transform(user_input['hair'])
user_input['beard'] = le_beard.fit_transform(user_input['beard'])
user_input['scarf'] = le_scarf.fit_transform(user_input['scarf'])
print("User input in encoded DataFrame format : \n")
print(user_input)
```

```
User input in encoded DataFrame format :

  height weight  hair  beard  scarf
0    180     60     0      0      0
```

In [94]:
```
df_woe = pd.DataFrame(user_data)
print("User input in actual DataFrame format : \n")
print(df_woe,"\n")
print("User input in encoded DataFrame format : \n")
print(user_input)
```

```
User input in actual DataFrame format :

  height weight  hair beard scarf
0    180     60  Long    no   yes

User input in encoded DataFrame format :

  height weight  hair  beard  scarf
0    180     60     0      0      0
```

Step 9.4: Apply Trained Model on Feature Vector of Unseen Data and Output Prediction (Male/Female) to User

In [95]:
```
result = pred_rfc.predict(user_input)
output = le_gender.inverse_transform(result)
print ("Prediction : ",output)
```

```
Prediction :  ['Female']
```

In [ ]: