Discriminant Analysis

Introduction

Perform Discriminant analysis using linear discriminant analysis from sklearn.discriminant_analysis.

The Data

The data set contain details of number of visits (1 or 2) to resort by the customer with some features.

```
import pandas as pd
df= pd.read_csv(r"C:\Users\my pc\Desktop\MBA - BA II\Multivariate analysis lab\4.DA\DAdata.csv")
df.columns = df.columns.str.replace(" ", "_")
df=df.rename(columns = {'Annual_family_income_(000s)':'Annual_family_income'})
#Dropping unnecessary columns
df.drop(['Respondent_Number'],axis = 1, inplace=True)
df.info()

Data columns (total 7 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                 --------------  -----
 0   Resort_visit                            30 non-null     int64
 1   Annual_family_income                    30 non-null     float64
 2   Attitude_towads_travel                  30 non-null     int64
 3   Importance_attached_to_family_skiing_holiday  30 non-null     int64
 4   Household_size                          30 non-null     int64
 5   Age_of_head_of_household                30 non-null     int64
 6   Amount_spent_on_family_skiing           30 non-null     int64

#split the feature and target variable
x = df.drop(['Resort_visit'],axis = 1)
x.info()
y = df['Resort_visit']
```

Summary statistics and visualization of dataset

Group Frequency :

```
#group frequency
count = df.groupby(['Resort_visit']).size()
print(count)

output

Resort_visit
1                15
2                15
```

Here we observed that, we have equal number of data on both class.

Group mean

```
#group mean
class_feature_means = pd.DataFrame(columns=y)
for c, rows in df.groupby('Resort_visit'):
   class_feature_means[c] = rows.mean()
class_feature_means = class_feature_means.drop('Resort_visit')
class_feature_means

output:
```

Resort_visit	1	2
Annual_family_income	60.520000	41.913333
Attitude_towads_travel	5.400000	4.333333
Importance_attached_to_family_skiing_holiday	5.800000	4.066667
Household_size	4.333333	2.800000
Age_of_head_of_household	53.733333	50.133333
Amount_spent_on_family_skiing	2.600000	1.400000

These are mean value of all feature class 1 and class 2

Perform one-way MANOVA

We conduct this monova analysis to our find data is statistically significant to perform to lda.

```
from statsmodels.multivariate.manova import MANOVA
fit = MANOVA.from_formula('Annual_family_income + Attitude_towads_travel +\
                  Importance_attached_to_family_skiing_holiday+\
                   Household_size+\
                     Age_of_head_of_household + \
                        Amount_spent_on_family_skiing ~ Resort_visit', data=df)
print(fit.mv_test())

output:

                                    Multivariate linear model
============================================================

------------------------------------------------------------
     Intercept                Value             Num DF  Den DF     F Value Pr > F
------------------------------------------------------------
     Wilks' lambda            0.0525            6.0000  23.0000    69.1554 0.0000
     Pillai's trace           0.9475            6.0000  23.0000    69.1554 0.0000
     Hotelling-Lawley trace   18.0405           6.0000  23.0000    69.1554 0.0000
     Roy's greatest root      18.0405           6.0000  23.0000    69.1554 0.0000
------------------------------------------------------------

------------------------------------------------------------
     Resort_visit             Value             Num DF  Den DF     F Value Pr > F
------------------------------------------------------------
```

```
    Wilks' lambda                 0.3021          6.0000          23.0000   8.8556          0.0000
    Pillai's trace        0.6979          6.0000          23.0000   8.8556          0.0000
    Hotelling-Lawley trace        2.3102          6.0000          23.0000   8.8556          0.0000
    Roy's greatest root           2.3102          6.0000          23.0000   8.8556          0.0000
============================================================
```

The Wilks' lambda test statistics is statistically significant [Wilks' lambda = 0.3021, F(6, 23) = 8.8556, p = 0.000] and indicates that resort visit has a statistically significant association with all the features.

Linear Discriminant Analysis

Here we will perform the linear discriminant analysis (LDA) using sklearn to see the differences between each group. LDA will discriminate the groups using information from both the dependent variables.

```
#LinearDiscriminantAnalysis
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
import seaborn as sns
import numpy as np
lda =LinearDiscriminantAnalysis(n_components = 1)
da =lda.fit(x,y)
y_pred = lda.predict(x)
print(y_pred)

output:

[1 1 1 2 1 1 2 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
```

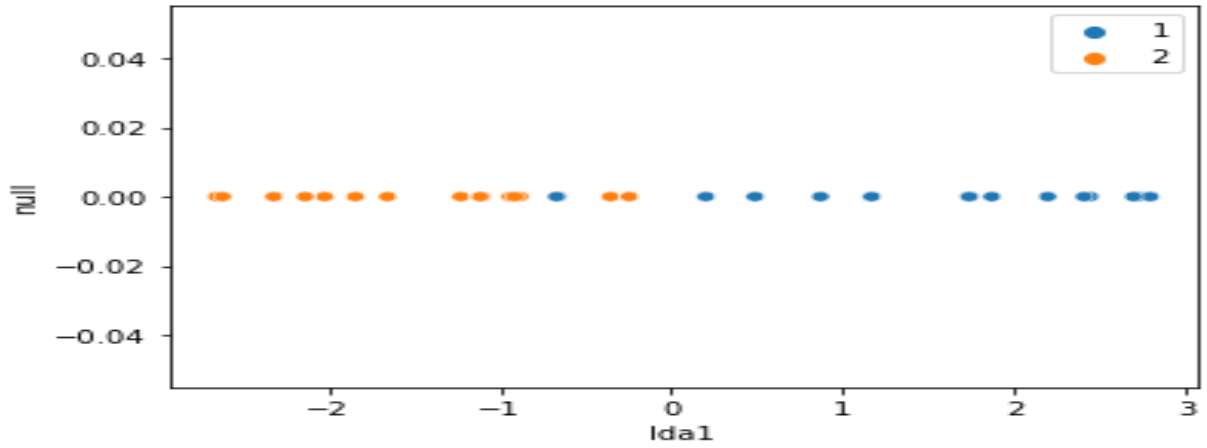
```
# get Prior probabilities of groups:
da.priors_

output:

array([0.5, 0.5])
```

Plot

```
#plot
X_new = pd.DataFrame(da.transform(x), columns=["lda1"])
val = 0. # this is the value where you want the data to appear on the y-axis.
X_new['null'] = np.zeros_like(X_new) +val;
sns.scatterplot(data=X_new, x="lda1", y="null", hue=df.Resort_visit.tolist(),palette=["C0", "C1"])
```

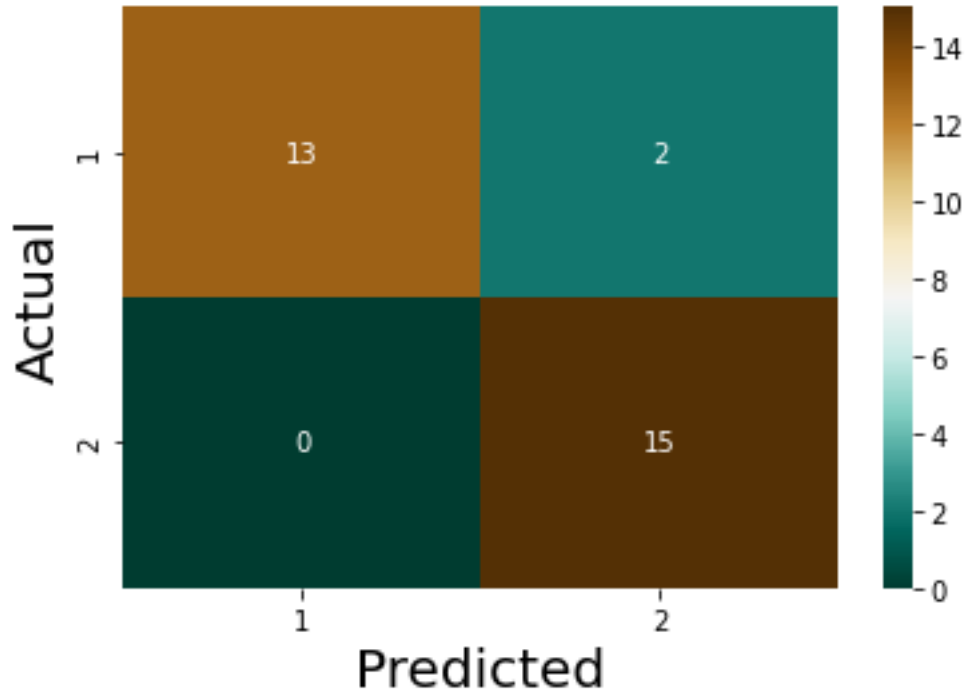


Confusion Matrix

```
from sklearn import metrics
cm=metrics.confusion_matrix(y,y_pred)
cm
x_axis = [1,2]
y_axis = [1,2]
p=sns.heatmap(cm, annot=True, cmap='BrBG_r',xticklabels=x_axis,yticklabels=y_axis)
p.set_xlabel("Predicted", fontsize = 20)
p.set_ylabel("Actual", fontsize = 20)

output:

array ([[13,  2],
        [0, 15]]
```



Using test data set predict the number of visit

#test dataset

```
df_test= pd.read_csv(r"C:\Users\my pc\Desktop\MBA - BA II\Multivariate analysis lab\4.DA\DAdata_test.csv")
df_test.columns = df_test.columns.str.replace(" ", "_")
df_test=df_test.rename(columns = {'Annual_family_income_(000s)':'Annual_family_income'})
df_test.drop(['Respondent_Number'],axis = 1, inplace=True)
test_pred = lda.predict(df_test)
test_pred

output :

array([1, 2, 2, 2, 1, 1, 2, 1, 1, 2]
```

Conclusion

We done the Discriminant analysis on given data set and predict the number of visit of resort using Linear discriminant analysis model.

Python code

```
import pandas as pd
df= pd.read_csv(r"C:\Users\my pc\Desktop\MBA - BA II\Multivariate analysis lab\4.DA\DAdata.csv")
df.columns = df.columns.str.replace(" ", "_")
df=df.rename(columns = {'Annual_family_income_(000s)':'Annual_family_income'})
```

```
#Dropping unnecessary columns
df.drop(['Respondent_Number'],axis = 1, inplace=True)
df.info()
#split the feature and target variable
x = df.drop(['Resort_visit'],axis = 1)
x.info()
y = df['Resort_visit']

#group frequency
count = df.groupby(['Resort_visit']).size()
print(count)
#group mean
class_feature_means = pd.DataFrame(columns=y)
for c, rows in df.groupby('Resort_visit'):
   class_feature_means[c] = rows.mean()
class_feature_means = class_feature_means.drop('Resort_visit')
class_feature_means



from statsmodels.multivariate.manova import MANOVA
fit = MANOVA.from_formula('Annual_family_income + Attitude_towads_travel +\
                    Importance_attached_to_family_skiing_holiday+\
                     Household_size+\
                       Age_of_head_of_household + \
                          Amount_spent_on_family_skiing ~ Resort_visit', data=df)
print(fit.mv_test())

#LinearDiscriminantAnalysis
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
import seaborn as sns
import numpy as np
lda =LinearDiscriminantAnalysis(n_components = 1)
da =lda.fit(x,y)
y_pred = lda.predict(x)
print(y_pred)

# get Prior probabilities of groups:
da.priors_

#plot
X_new = pd.DataFrame(da.transform(x), columns=["lda1"])
val = 0. # this is the value where you want the data to appear on the y-axis.
X_new['null'] = np.zeros_like(X_new) +val;
sns.scatterplot(data=X_new, x="lda1", y="null", hue=df.Resort_visit.tolist(),palette=["C0", "C1"])



from sklearn import metrics
```

```
cm=metrics.confusion_matrix(y,y_pred)
cm
x_axis = [1,2]
y_axis = [1,2]
p=sns.heatmap(cm, annot=True, cmap='BrBG_r',xticklabels=x_axis,yticklabels=y_axis)
p.set_xlabel("Predicted", fontsize = 20)
p.set_ylabel("Actual", fontsize = 20)


#test dataset
df_test= pd.read_csv(r"C:\Users\my pc\Desktop\MBA - BA II\Multivariate analysis lab\4.DA\DAdata_test.csv")
df_test.columns = df_test.columns.str.replace(" ", "_")
df_test=df_test.rename(columns = {'Annual_family_income_(000s)':'Annual_family_income'})
df_test.drop(['Respondent_Number'],axis = 1, inplace=True)
test_pred = lda.predict(df_test)
test_pred
```