Discriminant Analysis

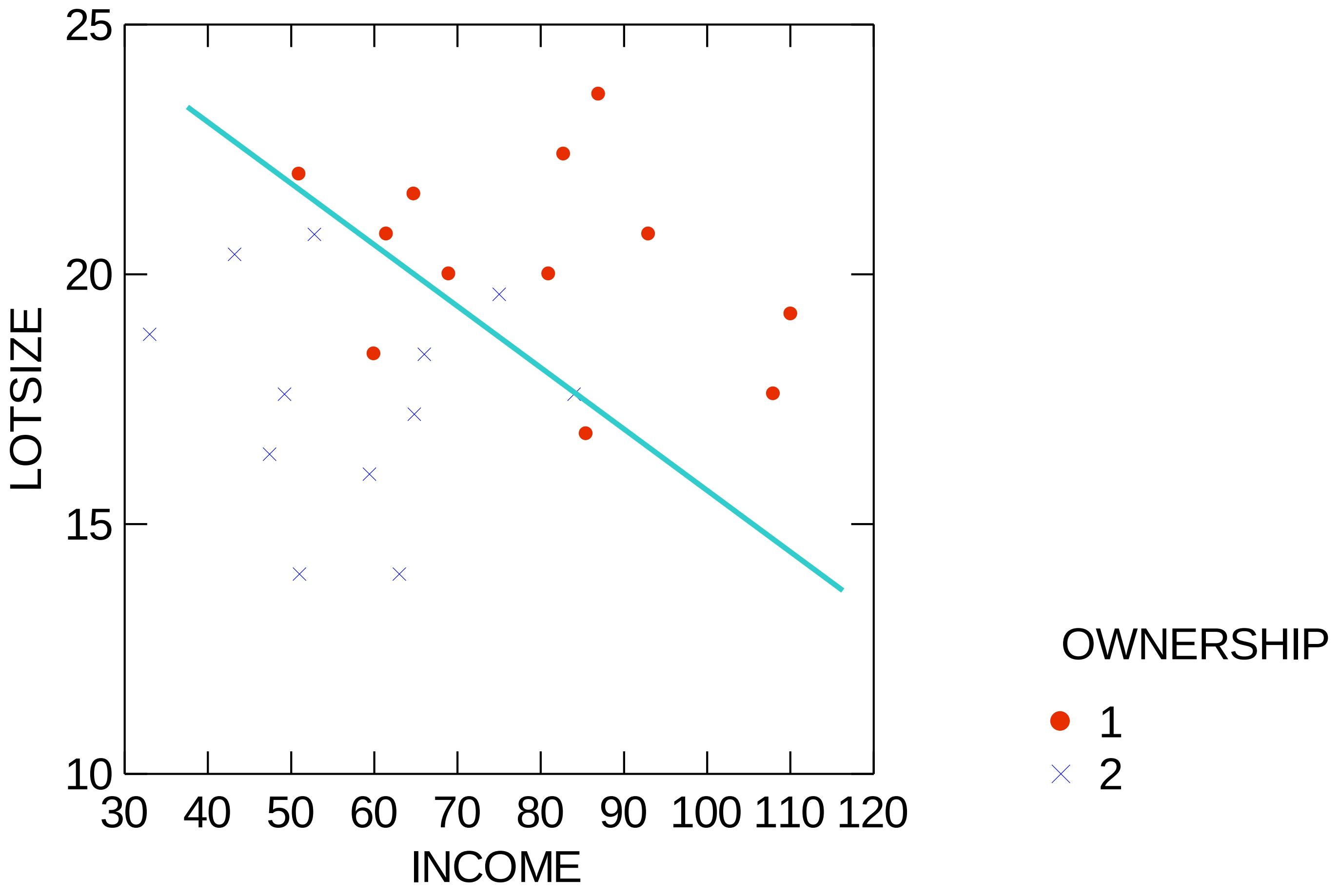
- Concerned with separating distinct sets of objects (observations) and allocating new objects (observations) to previously defined groups.

Classification for two population

- Separating two classes of objects
- Label the two classes Π_1 and Π_2
- Objects are classified on the basis of measurements on p associated variables

$\mathbf{X} = [X_1, X_2, \ldots X_p]$

Populations π_1 and π_2	Measured variables, $\mathbf{X}$
Solvent and Insolvent insurance company	Total assets, cost of stocks and bonds, market value of stocks and bonds, loss expenses, surplus, amount of premium.
Federalist papers written by James Madison and those written by Alexander Hamilton	Frequencies of different words and lengths of sentences
Purchasers of new products and laggards	Education, income, family size, amount of previous brand switching
Successful and unsuccessful students	Entrance examination scores, grade point average in school examination, number of school activities.
Good and poor credit risks	Income, age, number of credit cards, family size, occupation.



25
20
15
10
LOTSIZE
30
40
50
60
70
80
90
100
110
120
INCOME
OWNERSHIP
1
2

Classification principles

I. A good classification procedure should result in few misclassifications

- Probabilities of misclassification should be small
- For unequal population size, one has a greater likelihood of occurrence for larger populations; Include the concept of *prior probability*: let p_1 and p_2 be the prior probabilities of Π_1 and Π_2 respectively.

$$p_1 + p_2 = 1$$

		Classify as	
		π_1	π_2
True population	π_1		P(2|1)
	π_2	P(1|2)	

P(misclassified as π_1) = P(Observation comes from π_2 and is misclassified as π_1)
= P(1|2) p_2

P(misclassified as π_2) = P(Observation comes from π_1 and is misclassified as π_2)
= P(2|1) p_1

II. Another aspect of classification is cost

- Classifying a Π_1 object as belonging to Π_2 represents a more serious error than classifying a Π_2 object as belonging to Π_1

		Classify as	
		π_1	π_2
True population	π_1		C(2|1)
	π_2	C(1|2)	

A reasonable classification rule should have an Expected Cost of Misclassification (ECM) as small as possible.

$$\mathbf{ECM = C(2|1).P(2|1).}\boldsymbol{p_1}\mathbf{ + C(1|2).P(1|2).}\boldsymbol{p_2}$$

Respondent Number	Resort visit	Annual family income (000s)	Attitude towads travel	Importance attached to family skiing holiday	Household size	Age of head of household	Amount spent on family skiing
1	1	50.2	5	8	3	43	2
2	1	70.3	6	7	4	61	3
3	1	62.9	7	5	6	52	3
4	1	48.5	7	5	5	36	1
5	1	52.7	6	6	4	55	3
6	1	75	8	7	5	68	3
7	1	46.2	5	3	3	62	2
8	1	57	2	4	6	51	2
9	1	64.1	7	5	4	57	3
10	1	68.1	7	6	5	45	3
11	1	73.4	6	7	5	44	3
12	1	71.9	5	8	4	64	3
13	1	56.2	1	8	6	54	2
14	1	49.3	4	2	3	56	3
15	1	62	5	6	2	58	3

Respondent Number	Resort visit	Annual family income (000s)	Attitude towads travel	Importance attached to family skiing holiday	Household size	Age of head of household	Amount spent on family skiing
16	2	32.1	5	4	3	58	1
17	2	36.2	4	3	2	55	1
18	2	43.2	2	5	2	57	2
19	2	50.4	5	2	4	37	2
20	2	44.1	6	6	3	42	2
21	2	38.3	6	6	2	45	1
22	2	55	1	2	2	57	2
23	2	46.1	3	5	3	51	1
24	2	35	6	4	5	64	1
25	2	37.3	2	7	4	54	1
26	2	41.8	5	1	3	56	2
27	2	57	8	3	2	36	2
28	2	33.4	6	8	2	50	1
29	2	37.5	3	2	3	48	1
30	2	41.3	3	3	2	42	1

Respondent Number	Annual family income (000s)	Attitude towads travel	Importance attached to family skiing holiday	Household size	Age of head of household	Amount spent on family skiing
31	50.8	4	7	3	45	2
32	49.6	5	3	5	39	1
33	54.5	7	3	3	37	2
34	45	5	4	3	60	2
35	68	6	6	6	46	3
36	62.1	5	6	3	56	3
37	35	4	3	4	54	1
38	54	6	7	4	58	2
39	39.4	6	5	3	44	3
40	37	2	6	5	51	1

Devise a Discriminant Rule and based on the rule find whether the respondent Number 31-40 will visit the resort second time or not?

```
import pandas as pd
df = pd.read_csv("E:/MY DOCUMENTS/Desktop/Python/DAdata.csv")
# Dropping unnecessary columns
df.drop(['RespNo'], axis = 1, inplace=True)
# Dropping missing values rows
df.dropna(inplace=True)


from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
clf = LinearDiscriminantAnalysis()
X = df.iloc[:,1:].copy()
visit = df['visit'].copy()

clf.fit(X, visit)
```