Classification

Vahid Partovi Nia

Lecture 03

6 November 2018

Outline

Supervised Learning

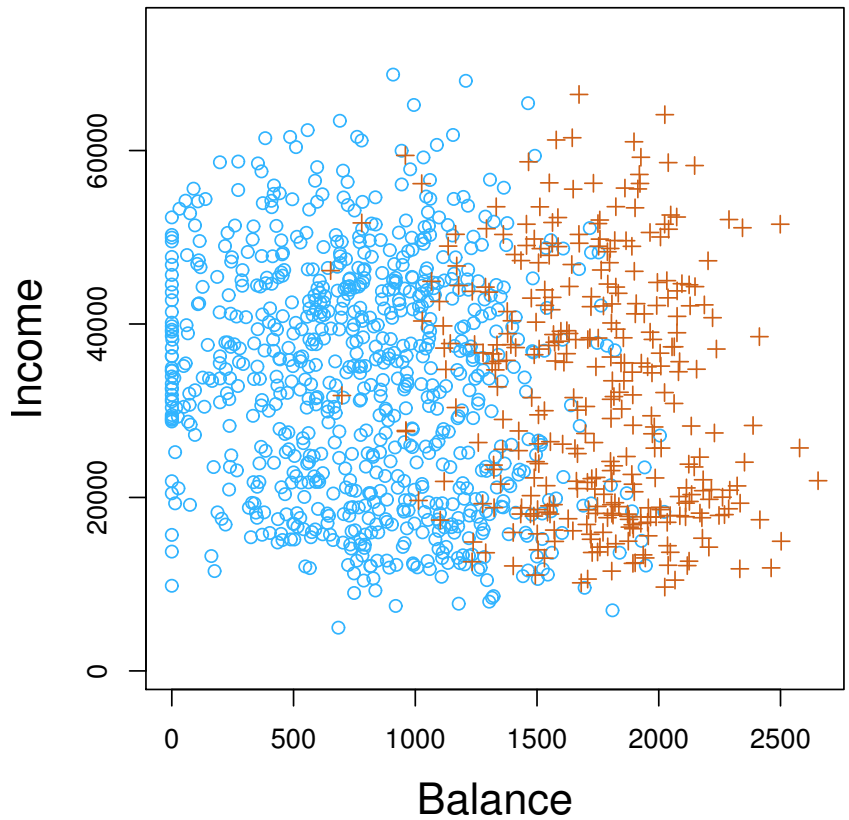
- Regression: y is continuous
- Classification: y is discrete

Default dataset

Default: After youve failed to make a payment on your credit card for 180 days, your issuer assumes youre probably never going to. The issuer closes your card, write off what you owe as bad debt, and sells your account to a collection agency.

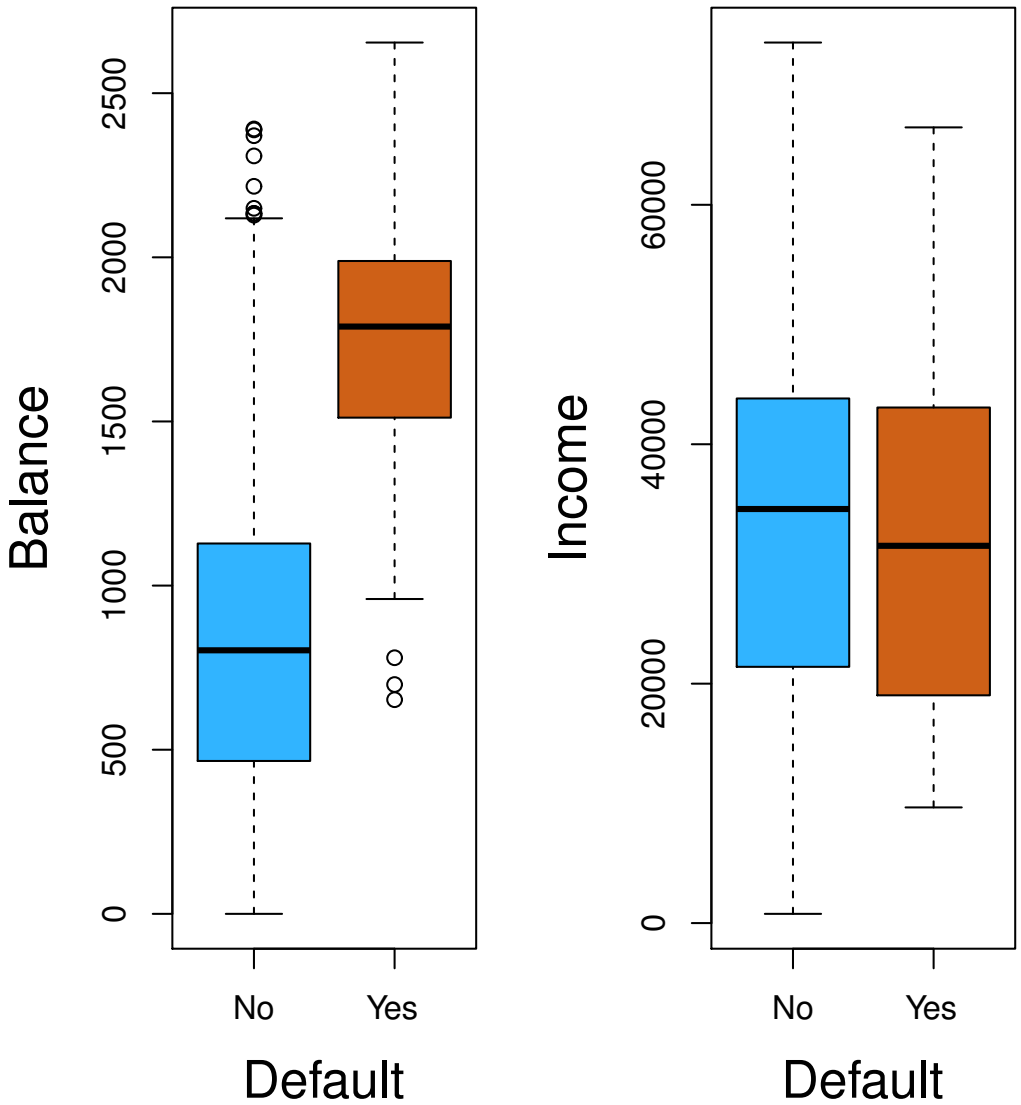
- y: default on credit card " No=0" or "Yes=1"
- x_1: Income
- x_2: Credit Balance

Scatterplot



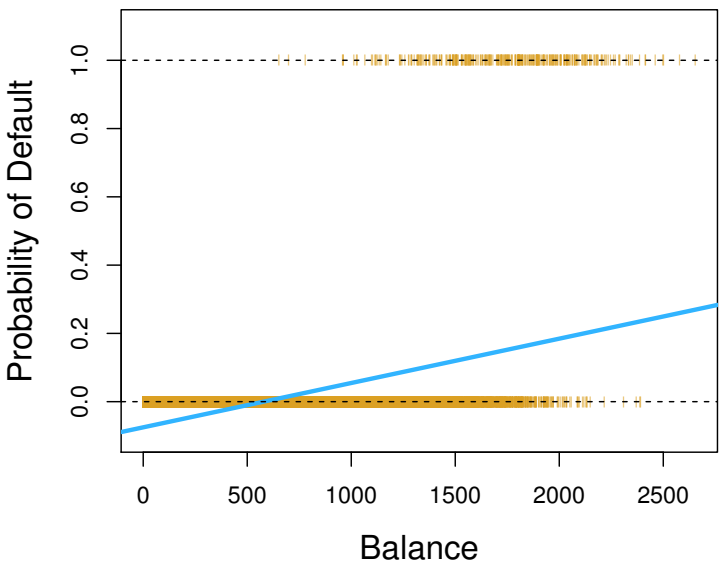
- No $= 0 =$ O
- Yes $= 1 =$ +

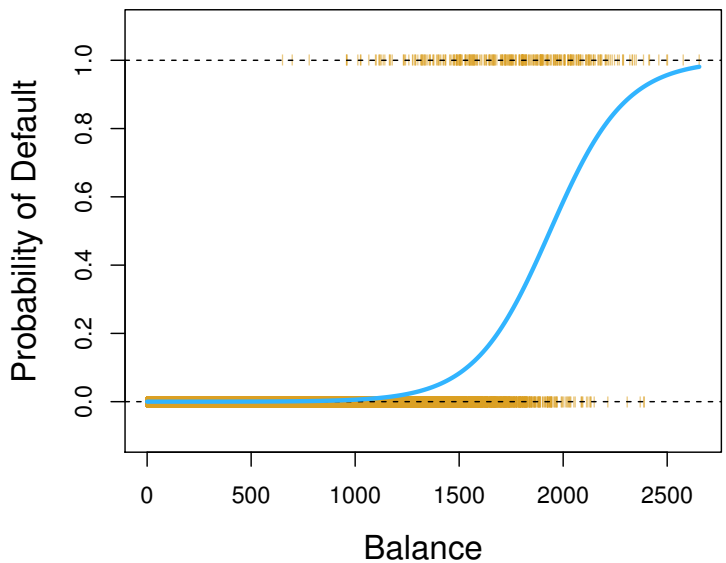
Boxplot



Regression for Classification

Simplify: let's focus only on "Balance" as the predictor of "default".





$$\begin{aligned} y_i &= \beta_0 + \varepsilon_i \\ \varepsilon_i &\sim N(0, \sigma^2) \end{aligned}$$

This means

$$y_i \sim N(\beta_0, \sigma^2)$$

$$y_i \sim B(p_0) = \begin{cases} y_i = 1 & \text{with probability } p_0 \\ y_i = 0 & \text{with probability } 1 - p_0 \end{cases}$$

One may define

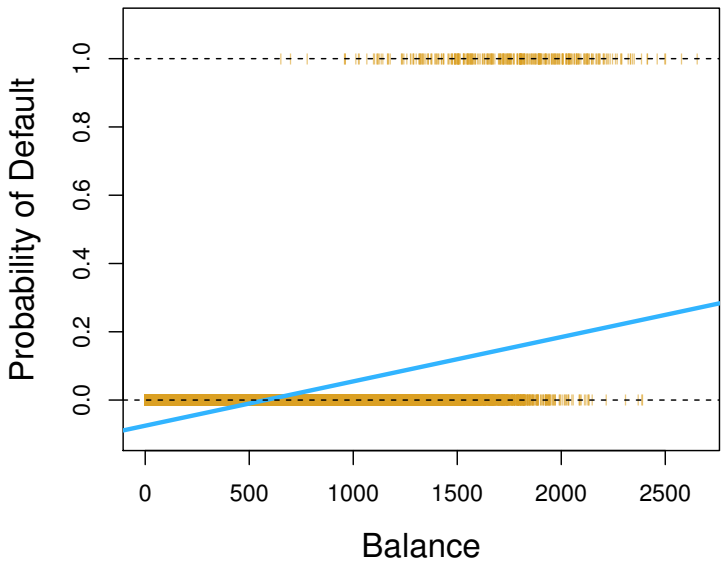
- $p_0 = p(\beta_0) = \frac{1}{1+e^{\beta_0}}$
- For $\beta_0 \in \mathbb{R}$, always $0 < p_0 < 1$.

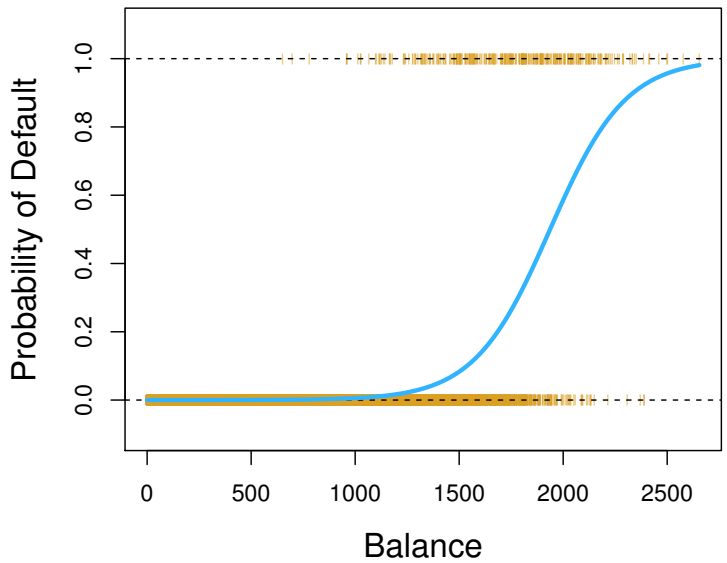
Simple Logistic Regression

$$\text{Default}_i \mid \text{Balance}_i \sim B(p_i) = \begin{cases} \text{Default}_i = 1 & \text{with probability } p_i \\ \text{Default}_i = 0 & \text{with probability } 1 - p_i \end{cases}$$

- $p_i = p(\beta_0 + \beta_1 \text{Balance}_i) = \frac{1}{1+e^{\beta_0+\beta_1 \text{Balance}_i}}$
- For $(\beta_0, \beta_1) \in \mathbb{R}^2$, always $0 < p_i < 1$.

Simple Regression





$$p_i = \beta_0 + \beta_1 x_i \qquad p_i = \frac{1}{1 + e^{\beta_0 + \beta_1 \text{Balance}_i}}$$

$$\log\left(\frac{p_i}{1 - p_i}\right) = \beta_0 + \beta_1 \text{Balance}_i$$

```
import pandas as pd
path='data/'
filename = path+'Default.xlsx'
default_data = pd.read_excel(filename)
```

```
import pandas as pd
path='data/'
filename = path+'Default.xlsx'
default_data = pd.read_excel(filename)


default_data['default_factor'] = default_data.default.factorize()[0]
default_data.head()
```

```
import pandas as pd
path='data/'
filename = path+'Default.xlsx'
default_data = pd.read_excel(filename)


default_data['default_factor'] = default_data.default.factorize()[0]
default_data.head()


from sklearn.linear_model import LogisticRegression

X = default_data[['balance']]
y = default_data['default_factor']


lr = LogisticRegression()
lr.fit(X, y)
```

Predict

Predict the probability of Default $=$ 'Yes' for $\text{Balance} = 1500$ and $\text{Balance} = 2000$ Implement simple logistic regression on credit data

Predict

Predict the probability of Default $=$ 'Yes' for $\text{Balance} = 1500$ and $\text{Balance} = 2000$ Implement simple logistic regression on credit data

```
import numpy as np
X_pred = np.array([1500, 2000]).reshape(-1,1)
print(lr.predict_proba(X_pred))
```

Multiple Logistic Regression

- y: default on credit card " No=0" or "Yes=1"
- x_1: Income
- x_2: Credit Balance

Coefficient interpretation

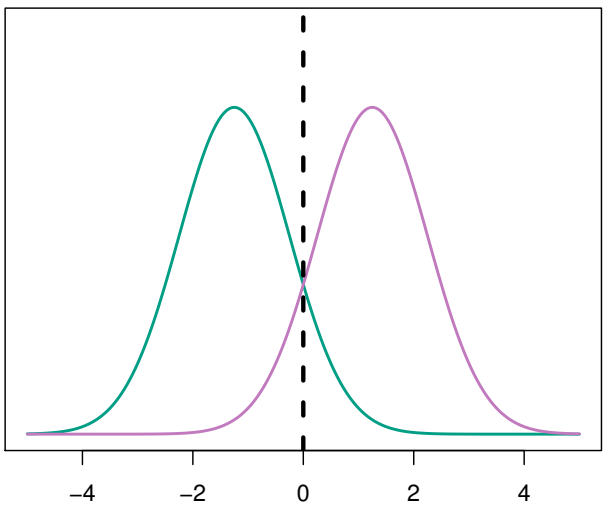
Implement simple logistic regression on credit data

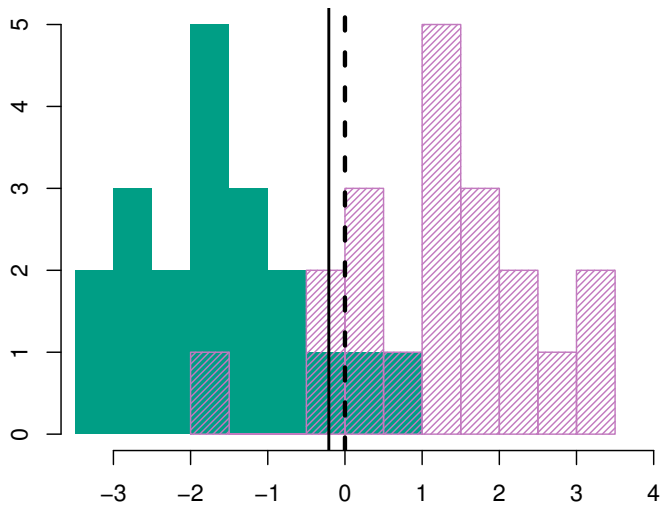
Linear Disceiminant

- Logistic Regression models $\text{Default}_i \mid \text{Balance}_i$
- Linear Discriminant models $\text{Balance}_i \mid \text{Default}_i$

Linear Discriminant

- $\text{Balance}_i \mid \text{Default}_i = 0 \sim N(\beta_0, \sigma^2)$
- $\text{Balance}_i \mid \text{Default}_i = 1 \sim N(\beta_1, \sigma^2)$





LDA

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
X = default_data[['balance']]
y = default_data['default_factor']
lda = LinearDiscriminantAnalysis()
lda.fit(X,y)
```

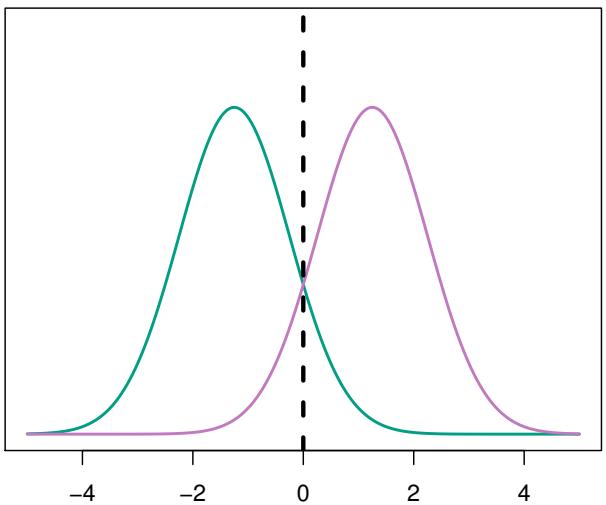
LDA

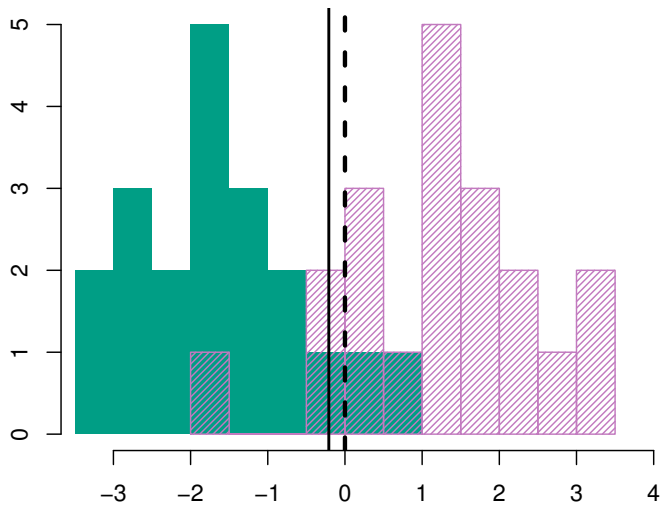
```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
X = default_data[['balance']]
y = default_data['default_factor']
lda = LinearDiscriminantAnalysis()
lda.fit(X,y)


X_pred = np.array([1500, 2000]).reshape(-1,1)
print(lda.predict_proba(X_pred))
```

Quadratic Discriminant

- $\text{Balance}_i \mid \text{Default}_i = 0 \sim N(\beta_0, \sigma_0^2)$
- $\text{Balance}_i \mid \text{Default}_i = 1 \sim N(\beta_1, \sigma_1^2)$





QDA

```
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis
X = default_data[['balance']]
y = default_data['default_factor']
qda = QuadraticDiscriminantAnalysis()
qda.fit(X,y)
```

QDA

```
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis
X = default_data[['balance']]
y = default_data['default_factor']
qda = QuadraticDiscriminantAnalysis()
qda.fit(X,y)


X_pred = np.array([1500, 2000]).reshape(-1,1)
print(qda.predict_proba(X_pred))
```

statsmodels

Logit Regression Results

Dep. Variable:	default_factor	**No. Observations:**	10000
Model:	Logit	**Df Residuals:**	9997
Method:	MLE	**Df Model:**	2
Date:	Mon, 02 Apr 2018	**Pseudo R-squ.:**	0.4594
Time:	21:46:38	**Log-Likelihood:**	-789.48
converged:	True	**LL-Null:**	-1460.3
		LLR p-value:	4.541e-292

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
Intercept	-11.5405	0.435	-26.544	0.000	-12.393	-10.688
balance	0.0056	0.000	24.835	0.000	0.005	0.006
income	2.081e-05	4.99e-06	4.174	0.000	1.1e-05	3.06e-05