Predictive Model Logistic Regression

Dr Liew How Hui

Jan 2022

Review

Week 3:

- Validation Set / Train-Test Split / Holdout method
- ‘distance measure’ and kNN models

Relevant Practicals for this topic:

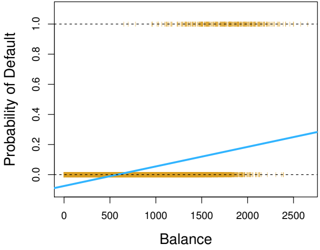
- p05_logreg1.R
- p06_logreg2.R

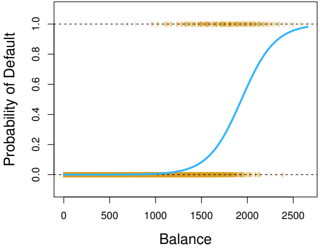
Outline

Using Linear Regression???

Using Linear Regression for Binary Classification may be a bad idea:

- Without cut-off, Y can be > 1 and < 0 but we want the output Y to be 0 or 1 only.
- Difficult to setup a cut-off as illustrated below





Theory

The *Logistic Regression (LR)* algorithm is a parametric method used for **binary** classification. It uses one-hot encoding to handle categorical features and it is better than kNN when the data is nearly linear and the feature dimension is large.

The assumption of LR is "the binary data are linearly separable with suitable parameters". Based on this assumption, a test input x would get a probability measure.

Theory (cont)

https://en.wikipedia.org/wiki/Logistic_function

$$S(x) = \frac{e^x}{1+e^x} = \frac{1}{1+e^{-x}}$$

Note that $0 < S(x) < 1$ for $-\infty < x < \infty$.

Cox (1958) proposed the "logistic regression" (LR) for binary classification problem:

$$\begin{aligned}
&\mathbb{P}(Y=1|X_1=x_1,\cdots,X_p=x_p)\\
=&S(\beta_0+\beta_1x_1+\cdots+\beta_px_p)\\
=&\frac{1}{1+\exp(-(\beta_0+\beta_1x_1+\cdots+\beta_px_p))}.
\end{aligned} \qquad (*)$$

Theory (cont)

Formula (*) can be written in vector form (using linear algebra)

$$\mathbb{P}(Y = 1|\mathrm{X} = \mathrm{x}) = S(\beta^T\tilde{\mathrm{x}})$$

where $\boldsymbol{\beta} = (\beta_0, \cdots, \beta_p)$ and $\widetilde{\mathrm{x}}_j = (1, \mathrm{x}_j)$.

Given an input x, the LR algorithm provides a prediction as follows (assuming the cut-off is 0.5):

$$h(\mathrm{x}) = \begin{cases} 1, & \mathbb{P}(Y = 1|X = \mathrm{x}) > 0.5 \\ 0, & \mathbb{P}(Y = 1|X = \mathrm{x}) \leq 0.5 \end{cases}$$

Theory (cont)

Estimating the parameters β_i from the given (observed) data (x_i, y_i), $i = 1, \cdots, n$ so that the **likelihood function** of $\beta_0, \cdots, \beta_p$:

$$L(\beta_0, \cdots, \beta_p;\; y_1, \cdots, y_n | \mathrm{x}_1, \cdots, \mathrm{x}_n) = \prod_{i=1}^{n} \mathbb{P}(Y = y_i | \mathrm{X} = \mathrm{x}_i) \tag{1}$$

is maximised using maximum likelihood estimation (MLE).

Y is binary and follows a **Bernoulli distribution**.

Theory (cont)

According to https://en.wikipedia.org/wiki/Bernoulli_distribution, $Y \sim Bernoulli(p_{\mathrm{x}} = \mathbb{P}(Y = 1|\mathrm{X} = \mathrm{x}))$, then the probability mass function of observing $y \in \{0, 1\}$ is

$$\mathbb{P}(y) = (p_{\mathrm{x}})^y (1 - p_{\mathrm{x}})^{1-y}.$$

$$\mathbb{P}(Y = y_i|\mathrm{X} = \mathrm{x}_i) = \left(\frac{e^{\widetilde{\mathrm{x}}_i^T \boldsymbol{\beta}}}{1 + e^{\widetilde{\mathrm{x}}_i^T \boldsymbol{\beta}}}\right)^{y_i} \left(1 - \frac{e^{\widetilde{\mathrm{x}}_i^T \boldsymbol{\beta}}}{1 + e^{\widetilde{\mathrm{x}}_i^T \boldsymbol{\beta}}}\right)^{1-y_i}$$

Theory (cont)

$$= e^{y_i \widetilde{\mathrm{x}}_i^T \boldsymbol{\beta}} \cdot (1 + e^{\widetilde{\mathrm{x}}_i^T \boldsymbol{\beta}})^{-y_i} \cdot (1 + e^{\widetilde{\mathrm{x}}_i^T \boldsymbol{\beta}})^{-(1-y_i)}$$

where $\boldsymbol{\beta} = (\beta_0, \cdots, \beta_p)$ and $\widetilde{\mathrm{x}}_i = (1, \mathrm{x}_i)$.

Substituting it into (1), we have

$$L(\beta_0, \cdots, \beta_p;\ y_1, \cdots, y_n | \mathrm{x}_1, \cdots, \mathrm{x}_n) = \prod_{i=1}^{n} (e^{y_i \widetilde{\mathrm{x}}_i^T \boldsymbol{\beta}})(1 + e^{\widetilde{\mathrm{x}}_i^T \boldsymbol{\beta}})^{-1}.$$

Taking natural log leads to

$$\ln L = \sum_{i=1}^{n} y_i \widetilde{\mathrm{x}}_i^T \boldsymbol{\beta} - \sum_{i=1}^{n} \ln(1 + e^{\widetilde{\mathrm{x}}_i^T \boldsymbol{\beta}}).$$

Theory (cont)

From Calculus, $f'(x^*) = 0$ at maximum:

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmax}}\, L = \underset{\boldsymbol{\beta}}{\operatorname{argmax}} \ln L \Rightarrow \frac{\partial}{\partial \boldsymbol{\beta}} (\ln L)\Big|_{\boldsymbol{\beta}=\hat{\boldsymbol{\beta}}} = 0$$

i.e.

$$\frac{\partial}{\partial \boldsymbol{\beta}} \left(\sum_{i=1}^{n} y_i \widetilde{\mathrm{x}}_i^T \boldsymbol{\beta} - \sum_{i=1}^{n} \ln(1 + e^{\widetilde{\mathrm{x}}_i^T \boldsymbol{\beta}}) \right) \Bigg|_{\boldsymbol{\beta}=\hat{\boldsymbol{\beta}}} = 0.$$

leading to

$$\sum_{i=1}^{n} y_i x_k^{(i)} - \sum_{i=1}^{n} \frac{x_k^{(i)} e^{\widetilde{\mathrm{x}}_i^T \hat{\boldsymbol{\beta}}}}{1 + e^{\widetilde{\mathrm{x}}_i^T \hat{\boldsymbol{\beta}}}} = 0, \quad k = 0, 1, \cdots, p$$

where $x_0^{(i)}$ is defined to be 1.

Hypothesis Testing and Inference

The *Z-statistic* tests the null hypothesis against the alternative hypothesis:

$$H_0 : \beta_i = 0 \quad \text{vs} \quad H_1 : \beta_i \neq 0.$$

https://en.wikipedia.org/wiki/Wald_test: With large "n",

$$\frac{\hat{\beta}_i - \beta_{i0}}{SE(\hat{\beta})} \sim Normal(0, 1),$$

The *standard error* $SE(\hat{\beta})$ is the inverse of the estimated information matrix with a shape $(p+1) \times (p+1)$:

$$SE(\hat{\beta}) = \left[\frac{\partial^2}{\partial \boldsymbol{\beta}^2} \left(\sum_{i=1}^{n} y_i \widetilde{\mathrm{x}}_i^T \boldsymbol{\beta} - \sum_{i=1}^{n} \ln(1 + e^{\widetilde{\mathrm{x}}_i^T \boldsymbol{\beta}}) \right) \right]^{-1}$$

Hypothesis Testing and Inference (cont)

- Z-statistic large $\Rightarrow$ *p-value* small,
  $\Rightarrow$ null hypothesis should be rejected (when p-value is less than some significance level, 5%, for example).
  $\Rightarrow$ X is associated with Y
  $\Rightarrow$ X is a significant factor.
- Z-statistic small $\Rightarrow$ *p-value* large
  $\Rightarrow$ null hypothesis should not be rejected (when (when p-value > 0.05).
  $\Rightarrow$ X and Y is most likely not related.
  $\Rightarrow$ X is an unimportant factor to Y.
- The interception $\hat{\beta}_0$ is typically not of interest and only for fitting data.

Hypothesis Testing and Inference (cont)

Logistic Regression in R (family=binomial):

```
glm(formula, family = gaussian, data, weights, subset,
    na.action, start = NULL, etastart, mustart, offset,
    control = list(...), model = TRUE, method = "glm.fit",
    x = FALSE, y = TRUE, contrasts = NULL, ...)

glm.fit(x, y, weights = rep(1, nobs),
        start = NULL, etastart = NULL, mustart = NULL,
        offset = rep(0, nobs), family = gaussian(),
        control = list(), intercept = TRUE)
```

Use family=binomial for LR; start = ‘guess’ for β_i;
glm.fit = iteratively reweighted least squares.
E.g.

```
library(ISLR)
Prob.Default = as.numeric(Default$default=="Yes")
plot(Default$balance,Prob.Default,xlab="Balance",
  pch='+', xlim=c(0,2750),ylim=c(-0.2,1.2))
glm.model = glm(default ~ balance, data=Default, family=binomial)
print(summary(glm.model)) #print(coef(glm.model))
```

Hypothesis Testing and Inference (cont)

```
Call:
glm(formula = default ~ balance, family = binomial, data = Default)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2697  -0.1465  -0.0589  -0.0221   3.7589

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.065e+01  3.612e-01  -29.49   <2e-16 ***
balance      5.499e-03  2.204e-04   24.95   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1596.5  on 9998  degrees of freedom
AIC: 1600.5

Number of Fisher Scoring iterations: 8
```

Hypothesis Testing and Inference (cont)

A $(1-\frac{\alpha}{2})\times 100\%$ confidence interval for β_i, $i=1,\cdots,p$, can be calculated as

$$\hat{\beta}_i \pm Z_{1-\alpha/2} SE(\hat{\beta}_i).$$

A 95% confidence interval is defined as a range of values such that with 95% probability, the range will contain the true unknown value of the parameter. In this case, $\alpha = 0.05$ so $Z_{1-\alpha/2}\approx 1.96$, therefore, the 95% confidence interval for β_i takes the form

$$[\hat{\beta}_i - 1.96\cdot SE(\hat{\beta}_i),\ \hat{\beta}_i + 1.96\cdot SE(\hat{\beta}_i)]. \tag{2}$$

Example

E.g. 3.3.1 (single numeric input)

Consider the logistic model for the **Default** data set:

$$\mathbb{P}(Y = 1|X) = \frac{1}{1 + \exp(-(-10.6513 + 0.0055\,\texttt{balance}))}$$

Predict the default probability for an individual with a balance of (a) \$1000, (b) \$2000.

Outline

Qualitative Predictors

When a predictor (or factor) is **qualitative**, we need to introduce **dummy variable(s)**: For example, the predictor "gender" has two levels 0 (male) and 1 (female), a new variable below is created

$$\text{gender1} = \begin{cases} 1, & \text{if gender} = 1 \\ 0, & \text{if gender} = 0 \end{cases}$$

Therefore, the logistic model is

$$\mathbb{P}(Y = 1|\mathrm{X} = \mathrm{x}) \\ = \frac{1}{1 + \exp(-(\beta_0 + \cdots + \beta_i \text{gender1} + \cdots))}$$

Qualitative Predictors (cont)

The coefficient associated with the dummy variable, "gender1" is interpreted as below.

β_i	OR	Relative probability of $\mathbb{P}(Y=1\vert\text{gender}=1)$	Probability to be classified into Class 1
Positive	≥ 1	Higher	female $>$ male
Negative	< 1	Lower	male $>$ female

where

$$\text{OR} = \frac{\frac{\mathbb{P}(Y=1|\text{gender}=1)}{\mathbb{P}(Y=0|\text{gender}=1)}}{\frac{\mathbb{P}(Y=1|\text{gender}=0)}{\mathbb{P}(Y=0|\text{gender}=0)}} = \frac{\exp(\cdots+\beta_i+\cdots)}{\exp(\cdots+0+\cdots)} = \exp(\beta_i)$$

OR=`https://en.wikipedia.org/wiki/Odds_ratio`

Qualitative Predictors (cont)

Example 3.1.6
Data: **Default** from ISLR
Formula: `default` $\sim$ `student`
The R script to fit the logistic model is listed below.

```
library(ISLR)
data(Default)
glm.model = glm(default ~ student, data=Default,
  family=binomial)
print(summary(glm.model))
```

Qualitative Predictors (cont)

The β_i coefficients and hypothesis testing results are:

```
Call:
glm(formula = default ~ student, family = binomial, data = Default)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.2970  -0.2970  -0.2434  -0.2434   2.6585

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.50413    0.07071  -49.55  < 2e-16 ***
studentYes   0.40489    0.11502    3.52 0.000431 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 2908.7  on 9998  degrees of freedom
AIC: 2912.7

Number of Fisher Scoring iterations: 6
```

Qualitative Predictors (cont)

Example 3.1.6 (cont)

(a) Find the odds ratio of default for a student with a non-student. Explain.

(b) Predict the probability of default for (i) student (ii) non-student.
Maths: (i) $\mathbb{P}(Y = 1|student = 1)$; (ii) $\mathbb{P}(Y = 1|student = 0)$

Qualitative Predictors (cont)

When a qualitative predictor X_i has $K > 2$ levels, $(K-1)$ **dummy variables** X_i.level2, $\cdots$, X_i.level$K$

$$\mathbb{P}(Y=1|\mathrm{X}) \\ = \frac{1}{1+\exp(-(\beta_0+\cdots+\beta_i^{(2)}x_i.\text{level2}+\cdots+\beta_i^{(K)}x_i.\text{level}K+\cdots))}$$

where

$$x_i.\text{level}k = \begin{cases} 1, & x_i = \text{level } k, \\ 0, & \text{otherwise}, \end{cases} \quad k = 2, \cdots, K.$$

For **one-hot encoding** the reference variable X_i may be kept. However, in the "nearly" one-hot encoding in LR, the reference variable is removed.

Examples (cont)

One of the reasons for LR to be widely used in practice is due to the interpretability of the model using the notion of **odds**:

$$\frac{\mathbb{P}(Y = 1|\mathrm{X} = \mathrm{x})}{\mathbb{P}(Y = 0|\mathrm{X} = \mathrm{x})} = \frac{\mathbb{P}(Y = 1|\mathrm{X} = \mathrm{x})}{1 - \mathbb{P}(Y = 1|\mathrm{X} = \mathrm{x})} = \exp(\tilde{\mathrm{x}}^T \boldsymbol{\beta}). \tag{3}$$

It quantifies the relative probability of odds as compared to $\mathbb{P}(Y = 0|X)$ as follows:

Value of odds	Relative Probability of $\mathbb{P}(Y = 1|X)$
≥ 1	Higher
< 1	Lower

Examples (cont)

By taking the logarithm of both sides of (3), we arrive at

$$\ln \frac{\mathbb{P}(Y = 1|\mathrm{X} = \mathrm{x})}{1 - \mathbb{P}(Y = 1|\mathrm{X} = \mathrm{x})} = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p. \quad (4)$$

The LHS is called the *log-odds* or *logit*, which is linear in X. Y can be inferred from inputs X.

Hence, LR "assumes" that the logit is linear in X. When assuming X to be quantitative, this means that a unit increase in X changes the logit by β_1 (4), or equivalently, it multiplies the odds by e^{β_1} (3). The amount that the default probability changes due to one-unit increase in X will depend on the current value of X.

Examples (cont)

E.g. 3.3.4: Suppose that the model is

```
Call: glm(formula=default~balance+income+student, family=binomial,
          data=Default)

Deviance Residuals:
    Min        1Q    Median        3Q       Max
-2.4691   -0.1418   -0.0557   -0.0203    3.7383

Coefficients: Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.087e+01  4.923e-01 -22.080  < 2e-16 ***
balance      5.737e-03  2.319e-04  24.738  < 2e-16 ***
income       3.033e-06  8.203e-06   0.370  0.71152
studentYes  -6.468e-01  2.363e-01  -2.738  0.00619 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2920.6  on 9999  degrees of freedom
Residual deviance: 1571.5  on 9996  degrees of freedom
AIC: 1579.5
Number of Fisher Scoring iterations: 8
```

Examples (cont)

E.g. 3.3.4 (cont)
Discuss the results involving the coefficients, odds and significance of each variable.

Solution

Coefficients: $\beta_0 = -10.8690$, $\beta_1 = 0.0057$, $\beta_2 = 0.0030$, $\beta_3 = -0.6468$.
Significance: Based on the p-value, we find that the intersection (bias), `balance` and `student` are significant while `income` is insignificant (according to the default $p = 0.05$).

Odds: The odds of the default increases with the balance but students has a lower odds compare to non-students.

Examples (cont): Final Exam Jan 2021, Q2(b)

The testing dataset of a social network advertisement is given in Table 2.2. The variables “Gender”, “Age” and “EstimatedSalary” are the predictors and the variable “Purchased” is the response. The “Gender” is a binary categorical data with levels “Male” and “Female”, the “Age” and the “EstimatedSalary” are quantitative data. The “Purchased” is a binary response with values 0 (representing “no purchase”, assuming **0 is the positive class**) and 1 (representing “purchase”).

Examples (cont)

Final Exam Jan 2021, Q2(b) continue

Table 2.2: The testing data of a social network advertisement.

Gender	Age	EstimatedSalary	Purchased
Male	29	80000	0
Male	45	26000	1
Female	48	29000	1
Male	45	22000	1
Female	47	49000	1
Male	48	41000	1
Male	46	23000	1
Male	47	20000	1
Male	49	28000	1
Female	47	30000	1

Examples (cont)

Final Exam Jan 2021, Q2(b) continue

Figure 2.1: The coefficients of the logistic regression based on an insurance claim data.

```
glm(formula=Purchased~., family=binomial, data=data.train)

Deviance Residuals:
    Min        1Q    Median       3Q      Max
-2.9882   -0.5640   -0.1372   0.5532   2.1820

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)     -1.188e+01  2.497e+00  -4.757 1.96e-06 ***
GenderMale       4.221e-01  5.927e-01   0.712 0.476319
Age              2.178e-01  4.751e-02   4.584 4.56e-06 ***
EstimatedSalary  3.868e-05  1.001e-05   3.863 0.000112 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
(Dispersion parameter for binomial family taken to be 1)
    Null deviance: 135.37  on 99  degrees of freedom
Residual deviance:  74.91  on 96  degrees of freedom
```

Examples (cont)

Final Exam Jan 2021, Q2(b) continue
Suppose a logistic regression model is trained and the coefficients are stated in Figure 2.1. Write down the **mathematical formula** of the logistic regression model and then use it to **predict** the variable “Purchase” of the insurance data in Table 2.2 as well as **evaluating** the performance of the model by calculating the confusion matrix, accuracy, sensitivity, specificity, PPV, NPV of the logistic model (assuming 0 is the positive class). [**Note**: The default cut-off is 0.5] (5 marks)

Examples (cont)

Final Exam Jan 2021, Q2(b) continue

Answer:

Let X_1 be Gender.Male, X_2 be Age and X_3 be EstimatedSalary and Y be the response variable "Purchased". The mathematical formula is

$$\mathbb{P}(Y=1|X_1=x_1,X_2=x_2,X_3=x_3)$$

$$=\frac{1}{1+\exp(-(-11.88+0.4221x_1+0.2178x_2+3.868\times 10^{-5}x_3))}$$ [1 mark]

Examples (cont)

Final Exam Jan 2021, Q2(b) continue

Answer (cont):

By using the formula, it is easy to construct the following table by using Excel:

Gender.Male	Age	EstimatedSalary	Probability	Predicted	Actual
1	29	80000	0.114325	0	0
1	45	26000	0.342715	0	1
0	48	29000	0.424609	0	1
1	45	22000	0.308756	0	1
0	47	49000	0.562649	1	1
1	48	41000	0.641615	1	1
1	46	23000	0.365990	0	1
1	47	20000	0.389908	0	1
1	49	28000	0.573792	1	1
0	47	30000	0.381544	0	1

With proper probability and predicted values in table.

Examples (cont)

Final Exam Jan 2021, Q2(b) continue

Answer (cont):

The confusion matrix is

		Actual Class	
		0 (not purchased)	1 (purchased)
Predicted Class	0 (not purchased)	1 (TP)	6 (FP)
	1 (purchased)	0 (FN)	3 (TN)

Examples (cont)

Final Exam Jan 2021, Q2(b) continue

Answer (cont):

Therefore, the performance metrics are

- accuracy $= \frac{1+3}{1+6+0+3} = 0.4$ .... [0.2 mark]
- sensitivity $= \frac{1}{1+1} = 1$ .... [0.2 mark]
- specificity $= \frac{3}{6+3} = 0.3333$ .... [0.2 mark]
- PPV $= \frac{1}{1+6} = 0.1429$ .... [0.2 mark]
- NPV $= \frac{3}{3+0} = 1$ .... [0.2 mark]

Example: Final Exam Jan 2019, Q5(b)

Table Q5(b) shows the results from a logistic regression to predict whether a customer churn happens.

Table Q5(b)

	Coefficient	p-value
Intercept	-7.6254	$<$0.0001
Gender_M	5.6211	0.0621
Age	0.3148	$<$0.0001
Payment_Cash	-0.7261	0.0012
Payment_Cheque	0.5024	0.0138
Income	-0.8521	0.0002

Final Exam Jan 2019, Q5(b) continue

(i) With 95% confidence and a cut-off of 0.7 for $Y = 1$, test the "reduced" model with the following test observations.

Obs	Gender	Age	Payment	Income	y
1	M	46	Card	1.6	0
2	F	52	Cash	8.5	1
3	F	54	Cheque	1.1	1
4	M	39	Cheque	7.4	0
5	F	55	Cash	9.4	1
6	M	49	Cheque	2.3	1
7	M	41	Cash	6.8	0
8	M	78	Card	8.1	1
9	F	42	Cash	2.1	1
10	M	37	Card	6.7	0

(13 marks)

(ii) Based on the answer in Q5(b)(i), construct a confusion matrix and calculate the five basic accuracy measures. (7 marks)

Examples (cont)

Final Exam May 2019, Q2
(a) The human resource department would like to determine potential employees for promotion. You have collected some data from previous employee promoting records as described below:

exp	Number of years of experience working in the company
sal_mth	Average monthly salary in last 12 months
sal_yr	Yearly salary in last 12 months
pjt	Is there any project involved? [Yes; No]
dpmt	Department [A; B; C; D]
emp_id	Employee ID
promote	Is the employee getting promoted? [Yes=1; No=0]

Examples (cont)

Final Exam May 2019, Q2 continue

A logistic regression has been constructed to predict the promotion of an employee. Table Q2(a) shows parts of the results of the logistic regression.

	Coefficient	*P*-value
Intercept	0.0035	$< 2e\text{-}16$
exp_yr	0.7124	$< 2e\text{-}16$
sal_mth	-0.0212	0.0057
sal_yr	-0.0363	0.0086
pjt_Yes	0.0330	0.2479
dpmt_B	1.0447	0.0002
dpmt_C	-1.5318	6.87e-05
dpmt_D	2.1539	0.0017
emp_id	-0.0279	0.5245

Table Q2(a)

Examples (cont)

Final Exam May 2019, Q2 continue

(i) Write the logistic regression model that compute the probability that an employee get promoted, $\mathbb{P}(Y = 1)$. (3 marks)

(ii) Calculate the odds and compare the probability of promotion for employee with 7 years of working experience and an employee with 2 years of working experience. (3 marks)

(iii) Calculate the odds and compare the probability of promotion for employee in different departments. Arrange the probability of promotion of department from lowest to highest. (8 marks)

Examples (cont)

Final Exam May 2019, Q2 continue
(c) State two possible issues found in the data. Suggest a suitable solution for each of the issue stated.

(4 marks)

Examples (cont)

Consider the weather data `http://storm.cis.fordham.edu/~gweiss/data-mining/weka-data/weather.arff`). Write an R script to test it using LOOCV.

Examples (cont)

Solution: A simple script is given below.

```
library(foreign)
d.f = read.arff("weather.arff")
### https://www.r-bloggers.com/predicting-creditability-using-logisti
errors = NULL
for(i in 1:nrow(d.f)) {
    d.f.test = d.f[ i,]
    d.f.tran = d.f[-i,]   # Leave-one-out
    logreg.model = glm(play~., family=binomial(link='logit'), data=d.
        control=list(maxit=50))
    # We can see that logistic regression fits the data poorly
    #print(summary(logreg.model))
    play.p = predict(logreg.model, newdata=d.f.test, type='response')
    play.p = ifelse(play.p > 0.5,"yes","no")
    errors[i] = (play.p!=d.f.test$play)
}
cat("error rate =", 100*sum(errors)/length(errors), "%\n")
```

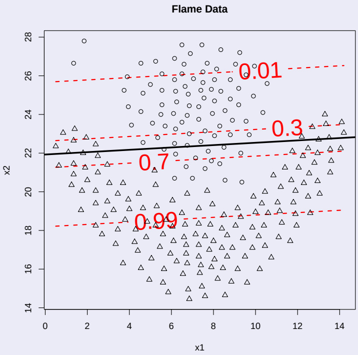
Examples (cont)

Not only that the error rate is 35.71% (high) but the coefficients in the logistic models are all having p-value much larger than 5% which indicates that logistic model is not suitable for modelling the weather data.

Examples (cont)

ROC Example

For the “flame” data, the “boundary” of the classifier is shown in the left figure below as the solid line:



Examples (cont)

ROC Example continue

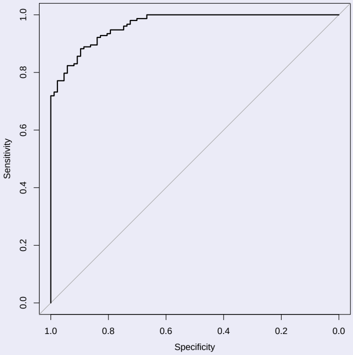
The dashed lines correspond to different "cut-off" 0.01, 0.3, 0.7 and 0.99.

The ROC curve can be understood as the result of varying the "cut-off" and calculating the "sensitivity" (TPR) and "specificity" mentioned in Topic 1. If we calculate out, we have

	0.01		0.3		0.7		0.99	
Predicted	1	2	1	2	1	2	1	2
1	19	0	64	6	79	23	87	80
2	68	153	23	147	8	130	0	73
	TPR = 0.2184	FPR = 0	0.7356	0.0392	0.9080	0.1503	1	0.5229

Examples (cont)

ROC Example continue



Outline

Multinomial LR

A general K-level qualitative response cannot be handled by the LR model.

We need `https://en.wikipedia.org/wiki/Multinomial_logistic_regression` (or Softmax regression):

$$\begin{cases} \ln \dfrac{\mathbb{P}(Y = 2|\mathrm{X} = \mathrm{x})}{\mathbb{P}(Y = 1|\mathrm{X} = \mathrm{x})} = \boldsymbol{\beta}_2 \cdot \mathrm{x} \\ \ln \dfrac{\mathbb{P}(Y = 3|\mathrm{X} = \mathrm{x})}{\mathbb{P}(Y = 1|\mathrm{X} = \mathrm{x})} = \boldsymbol{\beta}_3 \cdot \mathrm{x} \\ \qquad\qquad\qquad\qquad \cdots\cdots \\ \ln \dfrac{\mathbb{P}(Y = K|\mathrm{X} = \mathrm{x})}{\mathbb{P}(Y = 1|\mathrm{X} = \mathrm{x})} = \boldsymbol{\beta}_K \cdot \mathrm{x} \end{cases}$$

Multinomial LR (cont)

After some algebra, we have

$$\begin{aligned}\mathbb{P}(Y=1|\mathrm{X}=\mathrm{x}) &= \frac{1}{1+\sum_{i=2}^{K} e^{\boldsymbol{\beta}_i\cdot\mathrm{x}}} \\ \mathbb{P}(Y=j|\mathrm{X}=\mathrm{x}) &= \frac{e^{\boldsymbol{\beta}_j\cdot\mathrm{x}}}{1+\sum_{i=2}^{K} e^{\boldsymbol{\beta}_i\cdot\mathrm{x}}}, \quad j=2,\cdots,K.\end{aligned} \tag{5}$$

This model requires more data and LR, so when we have little data, this model won't work.

Multinomial LR (cont)

Note that LR can be regarded as a Multinomial LR when $K = 2$.
In R, the implementation is found in `nnet`:

```
multinom(formula, data, weights, subset, na.action,
         contrasts = NULL, Hess = FALSE, summ = 0,
         censored = FALSE, model = FALSE, ...)
```

We can compare the output of `glm` and `multinom` in Practical 3 for the data with $K = 2$.

Multinomial LR (cont)

In Python, it is implemented as a generalisation to **elastic net** instead of the LR we discussed:

```
class sklearn.linear_model.LogisticRegression(penalty='l2', *,
  dual=False, tol=0.0001, C=1.0, fit_intercept=True,
  intercept_scaling=1, class_weight=None, random_state=None,
  solver='lbfgs', max_iter=100, multi_class='auto', verbose=0,
  warm_start=False, n_jobs=None, l1_ratio=None)
```

When $C = \infty$, it approaches the LR. The LR and multinomial LR are implemented in Python as `Logit` and `MNLogit` in `statsmodels.discrete.discrete_model`.

Outline

(Feed-forward) ANN

Feed-forward Artificial Neural Networks (ANN) or multi-layer perceptron (MLP), "include" LR and multinomial LR as special cases.

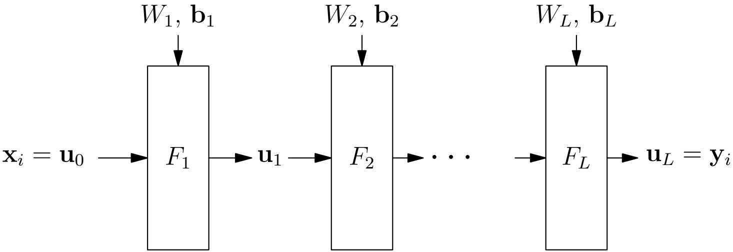
A multi-layer feed-forward ANN with input $\mathrm{x}_i \in \mathbb{R}^p$ and output is $\mathrm{y}_i \in \mathbb{R}^m$:

$$\begin{aligned}
\mathrm{u}_1 &= F_1(W_1\mathrm{u}_0 + \mathrm{b}_1), \quad \mathrm{u}_0 = \mathrm{x}_i \\
\mathrm{u}_2 &= F_2(W_2\mathrm{u}_1 + \mathrm{b}_2) \\
&\cdots \\
\hat{\mathrm{y}}_i = \mathrm{u}_L &= F_L(W_L\mathrm{u}_{L-1} + \mathrm{b}_L).
\end{aligned} \tag{6}$$

where L is the number of layers of ANN (with $L-1$ hidden layers).

ANN (cont)

Horizontal pictorial representation:



ANN (cont)

The algorithm to estimate the parameters W_ℓ and b_ℓ for the layer $\ell = 1, \ldots, L$ is the improvement of back-propagation algorithm:

1. $t = 0$;
2. Using the guess parameters $W_\ell^{(t)}$, $\mathrm{b}_\ell^{(t)}$, calculate all the intermediate states

$$\mathrm{u}_\ell^{(t)} = F_\ell(W_\ell^{(t)} \mathrm{u}_{\ell-1}^{(t)} + \mathrm{b}_\ell^{(t)})$$

   and the output $\hat{\mathrm{y}}_i$;

ANN (cont)

3. The output layer

$$\delta_L = \hat{\mathsf{y}}_i - \mathsf{y}_i$$

4. Back-Propagation (roughly): For ℓ from L to 1, do

$$\delta_{\ell-1} = \frac{\partial F_\ell}{\partial W_\ell}(\mathsf{u}_{\ell-1}^{(t)})\delta_\ell$$
$$W_\ell^{(t+1)} = W_\ell^{(t)} + \alpha \times \mathsf{u}_{\ell-1}^{(t)} \times \delta_{\ell-1}$$

5. $t = t + 1$ and go to step 2.

ANN (cont)

When $L = 1$, we obtain a `https://en.wikipedia.org/wiki/Perceptron`:

$$\mathrm{u}_1 = F_1(W_1 \mathrm{x}_i + \mathrm{b}_1). \tag{7}$$

We can see that when $m = 1$, $F_1(x) = S(x)$, we obtain the LR. When $m = K - 1$ ($K \geq 2$), we obtain the multinomial LR (which is how `nnet::multinom` was implemented).

ANN (cont)

When $L = 2$, we obtain an ANN with a single hidden-layer.

$$\begin{aligned} \mathrm{u}_1 &= F_1(W_1\mathrm{x}_i + \mathrm{b}_1) \\ \mathrm{y} = \mathrm{u}_2 &= F_1(W_2\mathrm{u}_1 + \mathrm{b}_2). \end{aligned} \tag{8}$$

This is implemented in R's `nnet` package as

```
nnet(x, y, weights, size, Wts, mask,
     linout = FALSE, entropy = FALSE, softmax = FALSE,
     censored = FALSE, skip = FALSE, rang = 0.7, decay = 0,
     maxit = 100, Hess = FALSE, trace = TRUE, MaxNWts = 1000,
     abstol = 1.0e-4, reltol = 1.0e-8, ...)
```

ANN (cont)

Python uses the more "precise name", i.e. MLP, for what we normally call "neural network".
It is implemented in Python's `sklearn` package as

```
class sklearn.neural_network.MLPClassifier(
  hidden_layer_sizes=(100,), activation='relu', *, solver='adam',
  alpha=0.0001, batch_size='auto', learning_rate='constant',
  learning_rate_init=0.001, power_t=0.5, max_iter=200,
  shuffle=True, random_state=None, tol=0.0001, verbose=False,
  warm_start=False, momentum=0.9, nesterovs_momentum=True,
  early_stopping=False, validation_fraction=0.1, beta_1=0.9,
  beta_2=0.999, epsilon=1e-08, n_iter_no_change=10, max_fun=15000)
```

It supports L layers and is more advanced than R's `nnet` which only supports 2 layers.