MBAN 5110: PREDICTIVE MODELING

SESSION 6: CLASSIFICATION MODELS

DR. ISIK BICER

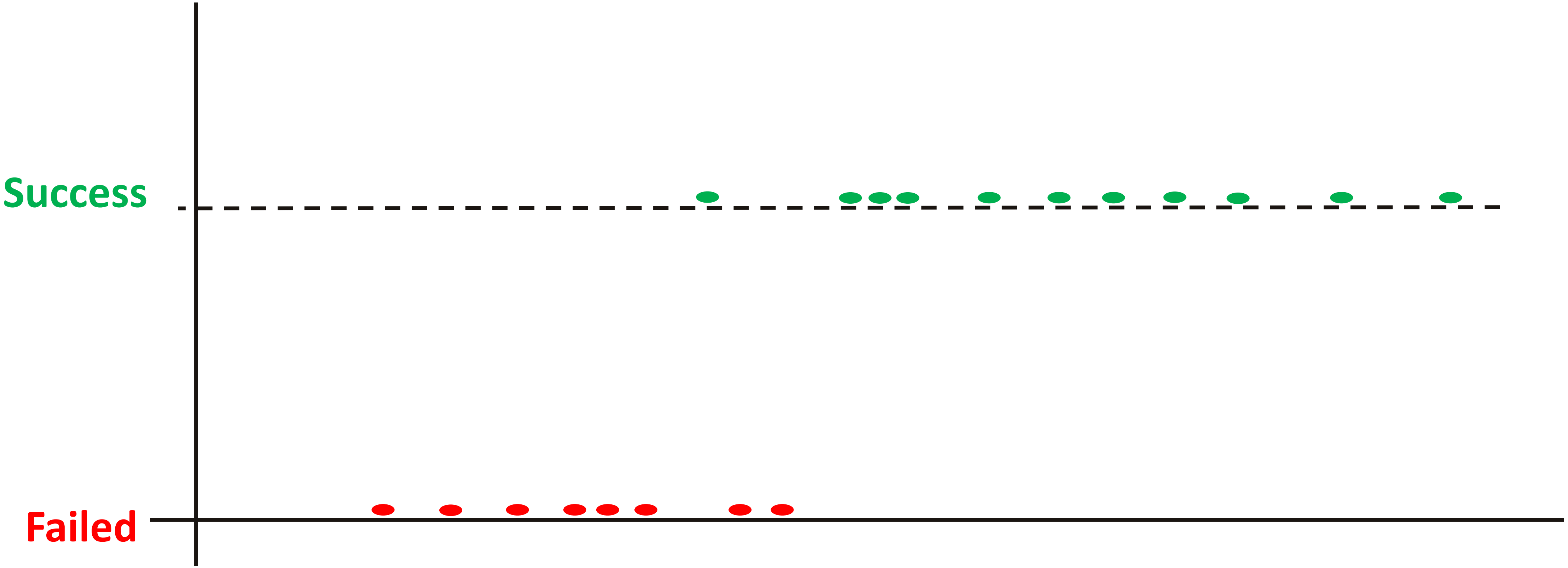
Schulich School of Business | YORK UNIVERSITÉ UNIVERSITY

PREDICTING QUALITATIVE RESPONSES

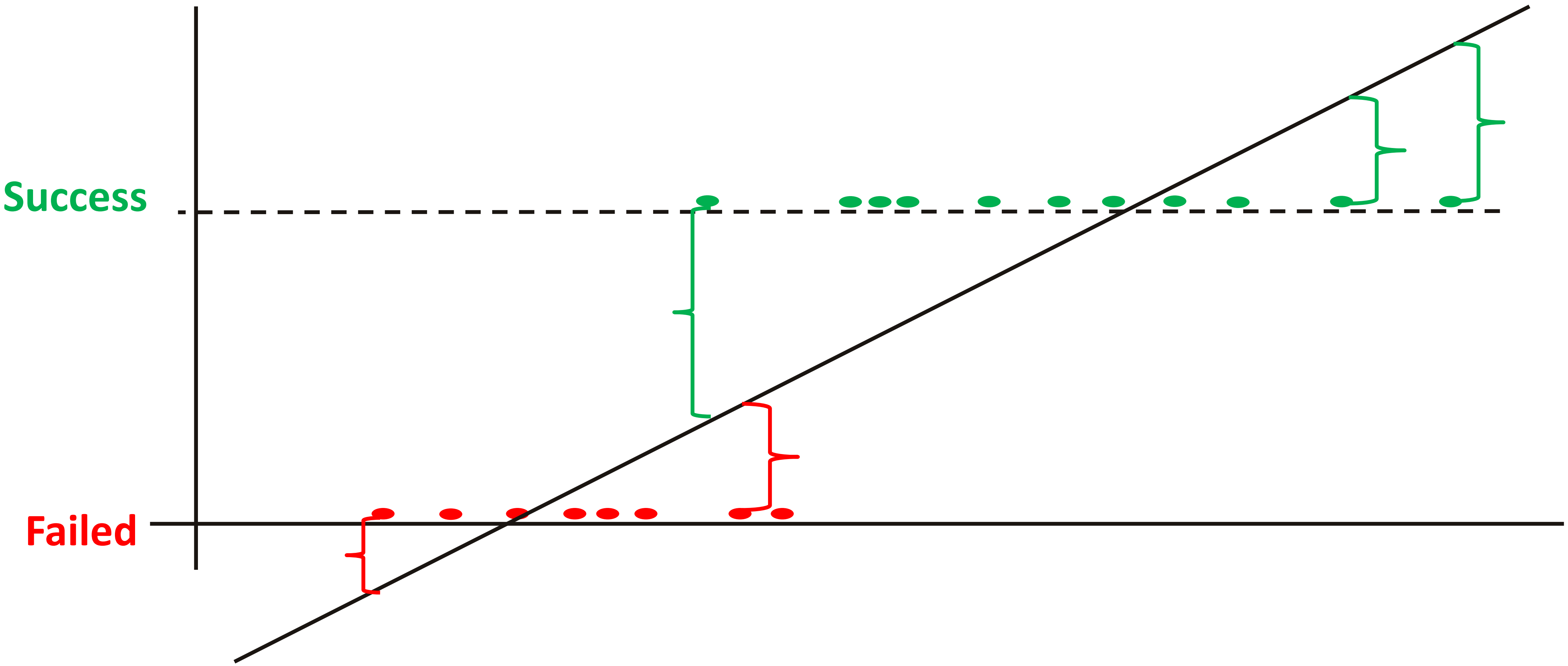
- Least squares for linear models
  - Dependent variable: Quantitative
  - Examples: Dependent variable is average income while independent variable is education level. Midterm grades and hours studied by each student.

- We sometimes observe qualitative data as dependent variable
  - Either passing or failing an exam
  - Whether rich or poor

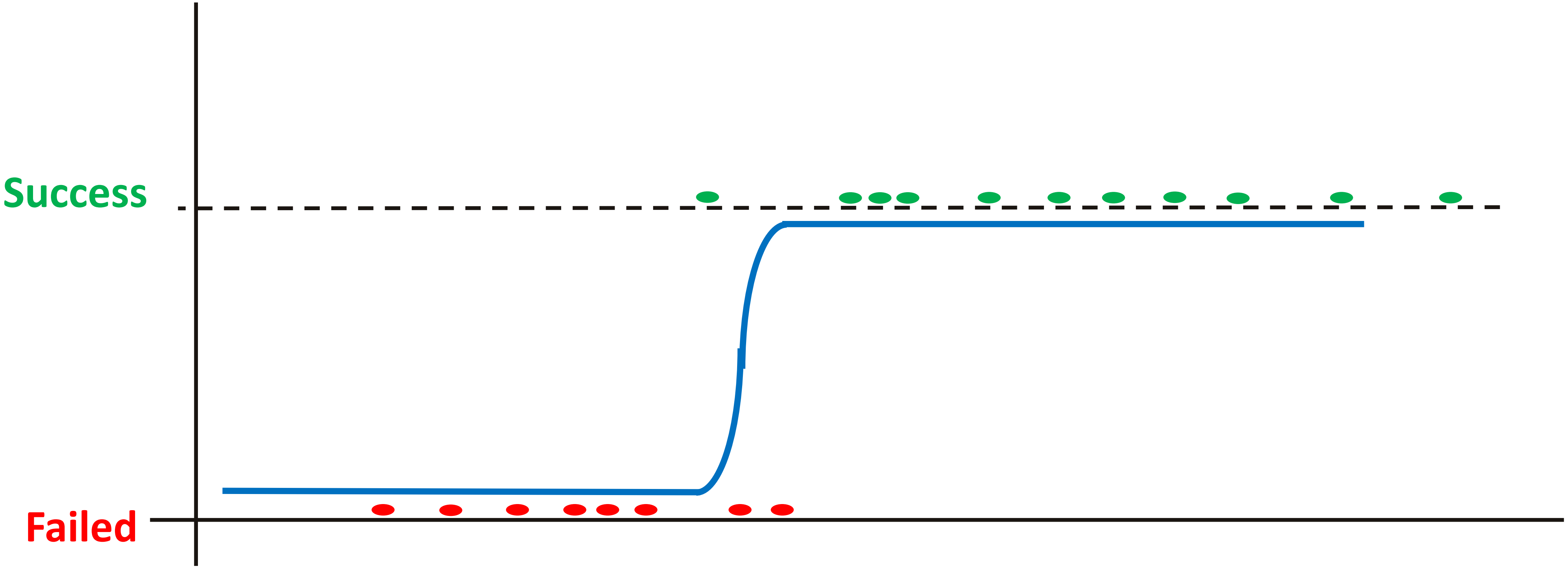
BINARY DEPENDENT VARIABLES



PROBLEM WITH A LINEAR MODEL



HOW TO FIT A CURVE THAT MINIMIZE THE ERRORS



LOGISTIC REGRESSION

$$y = \begin{cases} 0 \text{ if the student fails} \\ 1 \text{ if the student passes} \end{cases}$$

- x: hours studied for the exam (independent variable)
- Rather than modelling the dependent variable, we model its probability
- $\Pr(y = 1|x)$: Probability that a student passes if s/he spends x hours to study the exam

$$\Pr(y = 1|x) = f(x)$$

- $f(x)$: a function of x

CHARACTERISTICS OF THE FUNCTION

$$\Pr(y = 1|x) = f(x)$$

- $f(x)$: limited to be between 0 and 1
- There are many functions satisfying this limitation
- Logistic model is one of them, probably the most popular one

$$f(x) = \frac{e^{\beta_0+\beta_1 x}}{1 + e^{\beta_0+\beta_1 x}}$$

$$\underbrace{\ln\left(\frac{f(x)}{1 - f(x)}\right)}_{\boxed{\text{Logit}}} = \beta_0 + \beta_1 x$$

MULTI INDEPENDENT VARIABLES

$$y = \begin{cases} 0 \text{ if the student fails} \\ 1 \text{ if the student passes} \end{cases}$$

- x_1: hours studied for the exam
- x_2: GPA

$$f(x) = \frac{e^{\beta_0+\beta_1 x_1+\beta_2 x_2}}{1+e^{\beta_0+\beta_1 x_1+\beta_2 x_2}}$$

- The independent variables are listed in the exponential term

CATEGORICAL VARIABLES

- In a grocery store, a customer wants to buy cheese
- Four options: Cheddar, Gruyere, Gouda, and Feta

$$y = \begin{cases} 1 \text{ if Cheddar} \\ 2 \text{ if Gruyere} \\ 3 \text{ if Gouda} \\ 4 \text{ if Feta} \end{cases}$$

MULTINOMIAL LOGIT

- To develop a model to predict categorical variables with n categories, we have to run n-1 logistic regression model independently:
- Model (1): $\ln\left(\frac{\Pr(y=1|x)}{\Pr(y=4|x)}\right) = \beta_{1,0} + \beta_{1,1}x$
- Model (2): $\ln\left(\frac{\Pr(y=2|x)}{\Pr(y=4|x)}\right) = \beta_{2,0} + \beta_{2,1}x$
- Model (3): $\ln\left(\frac{\Pr(y=3|x)}{\Pr(y=4|x)}\right) = \beta_{3,0} + \beta_{3,1}x$

MULTINOMIAL LOGIT

$$\Pr(y=1|x)=\Pr(y=4|x)\,e^{\beta_{1,0}+\beta_{1,1}x}$$
$$\Pr(y=2|x)=\Pr(y=4|x)\,e^{\beta_{2,0}+\beta_{2,1}x}$$
$$\Pr(y=3|x)=\Pr(y=4|x)\,e^{\beta_{3,0}+\beta_{3,1}x}$$

- Total probability function:

$$\Pr(y=1|x)+\Pr(y=2|x)+\Pr(y=3|x)+\Pr(y=4|x)=1$$

$$\Pr(y=4|x)\left(1+\sum_{i=1}^{3} e^{\beta_{i,0}+\beta_{i,1}x}\right)=1$$

$$\Pr(y=4|x)=\frac{1}{1+\sum_{i=1}^{3} e^{\beta_{i,0}+\beta_{i,1}x}}$$

MULTINOMIAL LOGIT

$$\Pr(y = 1|x) = \frac{e^{\beta_{1,0}+\beta_{1,1}x}}{1+\sum_{i=1}^{3} e^{\beta_{i,0}+\beta_{i,1}x}}$$

$$\Pr(y = 2|x) = \frac{e^{\beta_{2,0}+\beta_{2,1}x}}{1+\sum_{i=1}^{3} e^{\beta_{i,0}+\beta_{i,1}x}}$$

$$\Pr(y = 3|x) = \frac{e^{\beta_{3,0}+\beta_{3,1}x}}{1+\sum_{i=1}^{3} e^{\beta_{i,0}+\beta_{i,1}x}}$$

$$\Pr(y = 4|x) = \frac{1}{1+\sum_{i=1}^{3} e^{\beta_{i,0}+\beta_{i,1}x}}$$

SOFTMAX NORMALIZATION

- Assume that $e^{\beta_{4,0}+\beta_{4,1}x} = 1$. Then,

$$\Pr(y = 1|x) = \frac{e^{\beta_{1,0}+\beta_{1,1}x}}{\sum_{i=1}^{4} e^{\beta_{i,0}+\beta_{i,1}x}}$$

$$\Pr(y = 2|x) = \frac{e^{\beta_{2,0}+\beta_{2,1}x}}{\sum_{i=1}^{4} e^{\beta_{i,0}+\beta_{i,1}x}}$$

$$\Pr(y = 3|x) = \frac{e^{\beta_{3,0}+\beta_{3,1}x}}{\sum_{i=1}^{4} e^{\beta_{i,0}+\beta_{i,1}x}}$$

$$\Pr(y = 4|x) = \frac{e^{\beta_{4,0}+\beta_{4,1}x}}{\sum_{i=1}^{4} e^{\beta_{i,0}+\beta_{i,1}x}}$$

SOFTMAX NORMALIZATION

- Now assume that we relax the constraint and keep the following expressions in their current forms:

$$\Pr(y = 1|x) = \frac{e^{\beta_{1,0}+\beta_{1,1}x}}{\sum_{i=1}^{4} e^{\beta_{i,0}+\beta_{i,1}x}}$$

$$\Pr(y = 2|x) = \frac{e^{\beta_{2,0}+\beta_{2,1}x}}{\sum_{i=1}^{4} e^{\beta_{i,0}+\beta_{i,1}x}}$$

$$\Pr(y = 3|x) = \frac{e^{\beta_{3,0}+\beta_{3,1}x}}{\sum_{i=1}^{4} e^{\beta_{i,0}+\beta_{i,1}x}}$$

$$\Pr(y = 4|x) = \frac{e^{\beta_{4,0}+\beta_{4,1}x}}{\sum_{i=1}^{4} e^{\beta_{i,0}+\beta_{i,1}x}}$$

- The coefficients are automatically normalized.

ESTIMATION OF COEFFICIENTS

- Let's create four dummy binary variables such that

  $y_1 = 1$ if $y = 1$, otherwise it is zero

  $y_2 = 1$ if $y = 2$, otherwise it is zero

  $y_3 = 1$ if $y = 3$, otherwise it is zero

  $y_4 = 1$ if $y = 4$, otherwise it is zero

- Given n observations of customer preferences, the maximum likelihood estimation is:

$$L(y, x) = \prod_{i=1}^{n} (\Pr(y_1|x))^{y_1} (\Pr(y_2|x))^{y_2} (\Pr(y_3|x))^{y_3} (\Pr(y_4|x))^{y_4}$$

ESTIMATION OF COEFFICIENTS

- The log-likelihood is then:

$$l(y,x) = \sum_{i=1}^{n}\{y_1 \ln(\Pr(y_1|x)) + y_2 \ln(\Pr(y_2|x)) + y_3 \ln(\Pr(y_3|x)) + y_4 \ln(\Pr(y_4|x))\}$$

- The optimal values of beta coefficients that maximize this function is found by numerical optimization methods.
  - Newton-cg
  - LBFGS

PREDICTION

- Returning back to original example

$$y = \begin{cases} 0 \text{ if the student fails} \\ 1 \text{ if the student passes} \end{cases}$$

- x: hours studied for the exam (independent variable)

$$\Pr(Success|x) = \frac{e^{\widehat{\beta_0}+\widehat{\beta_1}x}}{1 + e^{\widehat{\beta_0}+\widehat{\beta_1}x}}$$

- $\widehat{\beta_0}$ and $\widehat{\beta_1}$: estimates

MODEL ASSESSMENT

		What is observed in reality?		
		Success	Failure	
What model predicts	Success	12	6	Row Sum = 18
	Failure	8	2	Row Sum = 10
		Column Sum = 20	Column Sum = 8	Total = 28

PRECISION

- It gives the percentage of successes among those predicted as success by the model
- Model predicted that 18 students passed
- In reality, only 12 of these 18 students passed
- Precision = 12/18 = 66.7%

RECALL

- It is the ratio of the number of successes detected by the model to the total number of successes
- Model detected 12 of 20 successes
- Recall = 60%

PRECISION AND RECALL

- For an investor, precision is more important.
  - s/he wants to predict only a few of stocks with a high positive return in the stock market correctly
  - s/he doesn’t need to predict all stocks with a high return
  - If only few stocks with a high return can be correctly predicted, the investor invests in those stocks and doesn’t care for the rest

  - Once a clairvoyant tells me, Apple will certainly generate 20% return in the stock market in 2023, I don’t need to predict another stock correctly. I would only invest in Apple.

PRECISION AND RECALL

- For the Ontario government fighting the spread of COVID-19, recall is more important.
  - The government increases the number of daily tests to determine everyone with a positive test result and make them self-isolate themselves
  - If there are 400 active cases in the York region, the government wants to predict all of them correctly and ask them to stay at home

COVID EXAMPLE (CLASS EXERCISE)

		What is observed in reality?	
		Positive	**Negative**
What the tests predict	**Positive**	**120**	**5**
	Negative	**10**	**500**

- How many people were asked to stay at home although they are not positive?
- How many people are positive although they are not detected by the tests?
- Should we change the sensitivity of the test to reduce the number in top-right or bottom-left quadrant?

ACCURACY OF THE MODEL

- F_1 score to test the accuracy of the model

$$F_1 = \frac{2}{recall^{-1} + precision^{-1}}$$

EXAMPLE

		What is observed in reality?		
		Success	Failure	
What model predicts	Success	12	6	Row Sum = 18
	Failure	8	2	Row Sum = 10
		Column Sum = 20	Column Sum = 8	Total = 28

EXAMPLE

- Precision = 12/18
- Recall = 12/20

$$F_1 = \frac{2}{recall^{-1} + precision^{-1}} = \frac{2}{\frac{20}{12} + \frac{18}{12}} = \frac{24}{38} = 63\%$$

COVID EXAMPLE (CLASS EXERCISE)

		What is observed in reality?	
		Positive	Negative
What the tests predict	Positive	120	5
	Negative	10	500

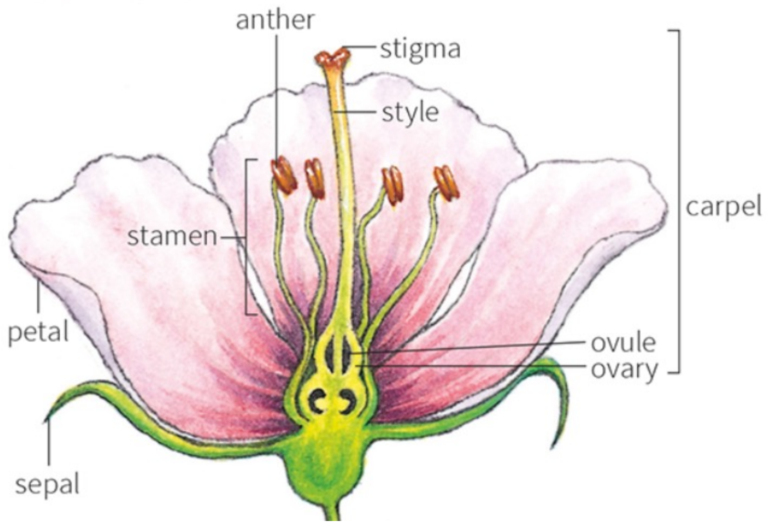
- What is the F1 score?

PROGRAMMING IN PYTHON

- Data set: Iris flowers datasets
- Very famous
- First used by Fisher (1936)
- Available on Python via sklearn
- Publicly available by UCI Machine Learning Laboratory

IRIS DATASET

- 4 attributes for Iris flowers
  - Sepal length
  - Sepal width
  - Pedal length
  - Pedal width
- Dependent variable (type of Iris flower)
  - 0 for Iris-Setosa
  - 1 for Iris-Versicolor
  - 2 for Iris-Virginica
- 150 observations (50 for each type)



LOADING DATASET ON PYTHON

```
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import f1_score
```

```
X,y = load_iris(return_X_y = True)
```

- load_iris is a function embedded in sklearn.datasets library
- The variable of this function is wheter return_X_y is true or false

IRIS DATASET

```
iris_dataframe = pd.DataFrame(X,columns=['sepal length in cm','sepal width in cm','petal length in cm',\
                                         'petal width in cm'])
iris_dataframe['iris class']=y
iris_dataframe['iris class'] = iris_dataframe['iris class'].replace(0,'Iris-Setosa')
iris_dataframe['iris class'] = iris_dataframe['iris class'].replace(1,'Iris-Versicolour')
iris_dataframe['iris class'] = iris_dataframe['iris class'].replace(2,'Iris-Virginica')
```

```
iris_dataframe
```

	sepal length in cm	sepal width in cm	petal length in cm	petal width in cm	iris class
0	5.1	3.5	1.4	0.2	Iris-Setosa
1	4.9	3.0	1.4	0.2	Iris-Setosa
2	4.7	3.2	1.3	0.2	Iris-Setosa
3	4.6	3.1	1.5	0.2	Iris-Setosa
4	5.0	3.6	1.4	0.2	Iris-Setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-Virginica
146	6.3	2.5	5.0	1.9	Iris-Virginica
147	6.5	3.0	5.2	2.0	Iris-Virginica
148	6.2	3.4	5.4	2.3	Iris-Virginica
149	5.9	3.0	5.1	1.8	Iris-Virginica

150 rows × 5 columns

FIRST ATTEMPT TO FIT THE LOGISTIC MODEL

- We received a warning message “STOP: TOTAL NO of ITERATIONS REACHED LIMIT”

```
model_classification = LogisticRegression(random_state=0).fit(X,y)
```

```
/Users/isikbicer/opt/anaconda3/lib/python3.9/site-packages/sklearn/li
lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

- Max iteration is limited to 100. So we have to increase it to 1000.

```
model_classification = LogisticRegression(random_state=0,max_iter = 1000).fit(X,y)
```

RESULTS

- Intercepts and coefficients

```
model_classification.intercept_
```

```
array([  9.85483916,   2.23332662, -12.08816578])
```

```
model_classification.coef_
```

```
array([[-0.42409864,  0.96688364, -2.51735629, -1.07931427],
       [ 0.53500171, -0.32104132, -0.20651985, -0.94434646],
       [-0.11090307, -0.64584232,  2.72387614,  2.02366073]])
```

- We have three intercepts and coefficient sets
- There are three classes
- Last instances are unnecessary

REMEMBER THE EARLIER SLIDE ON THEORY

$$\Pr(y = 0|x) = \frac{e^{\beta_{0,0}+\beta_{0,1}x}}{1 + \sum_{i=0}^{1} e^{\beta_{i,0}+\beta_{i,1}x}}$$

$$\Pr(y = 1|x) = \frac{e^{\beta_{1,0}+\beta_{1,1}x}}{1 + \sum_{i=0}^{1} e^{\beta_{i,0}+\beta_{i,1}x}}$$

$$\Pr(y = 2|x) = \frac{1}{1 + \sum_{i=0}^{1} e^{\beta_{i,0}+\beta_{i,1}x}}$$

- Here we use the last class $\Pr(y = 2|x)$ as a pivot
- Python uses another approach, which doesn’t use pivots
- It is also theoretically solid

PYTHON RESULTS

- When we don't pivot, the model becomes:

$$\Pr(y = 0|x) = \frac{e^{\beta_{0,0}+\beta_{0,1}x}}{\sum_{i=0}^{2} e^{\beta_{i,0}+\beta_{i,1}x}}$$

$$\Pr(y = 1|x) = \frac{e^{\beta_{1,0}+\beta_{1,1}x}}{\sum_{i=0}^{2} e^{\beta_{i,0}+\beta_{i,1}x}}$$

$$\Pr(y = 2|x) = \frac{e^{\beta_{2,0}+\beta_{2,1}x}}{\sum_{i=0}^{2} e^{\beta_{i,0}+\beta_{i,1}x}}$$

F1 SCORE

```
true_values = y
predictions = model_classification.predict(X)
f1_score(true_values,predictions,average='weighted')
```

```
0.9733226623982927
```

CLASS EXERCISE

- Obtain the confusion matrix from Python
- Calculate the precision, recall, and F1 Score manually