Machine Learning and Financial Applications

Lecture 6
Logistic Regression in Finance

Liu Peng
liupeng@smu.edu.sg

Video tutorials:
https://youtu.be/I51DDBeZ-VU
https://youtu.be/N8v5L09jEpo?si=ubUms04BBT-nSHOg

Bayesian Inference: Updating Beliefs with Data

Bayes' Rule: A method to update a model's probability using new data.

Key Components:

1. **Prior ($P(\text{model})$):** Initial belief about the model before data.
2. **Likelihood ($P(\text{data}|\text{model})$):** Probability of data given the model.
3. **Posterior ($P(\text{model}|\text{data})$):** Updated belief about the model after data.

The Formula:

$$P(\text{model}|\text{data}) \propto P(\text{data}|\text{model}) \times P(\text{model})$$

(Posterior $\propto$ Likelihood $\times$ Prior)

Logarithmic Form:
For easier computation:

$$\log(\text{Posterior}) \propto \log(\text{Likelihood}) + \log(\text{Prior})$$

Summary: Bayesian inference systematically combines prior knowledge with new evidence to refine understanding.

LLMs as Bayesian Generative Models

LLMs: Probabilistic Generative Models:

- LLMs are **generative models** learning language probability ($P(\text{text})$) to create new text, often by predicting the next token.

Bayesian Concepts in LLM Generation:

1. **Prior (Initial Generative State):**
   - Pre-trained LLM knowledge forms the initial generative **prior**.
2. **Likelihood (Guiding Generation):**
   - Training maximizes **likelihood** for plausible text generation; likelihoods guide token choice.
3. **Posterior (Refined Generative Model):**
   - Fine-tuning/prompting creates specialized/contextual **posterior** generative models.

Generative Process (Bayesian Lens):

- LLMs generate text by sampling from learned distributions, aiming for outputs aligned with a contextually informed **posterior**.

Maximum A Posteriori (MAP)

- MAP finds model parameters maximizing log posterior probability: $\max_{\text{model}} \log P(\text{model}|\text{data})$
- Since $\log \text{Posterior} \propto \log \text{Likelihood} + \log \text{Prior}$, $\min_{\text{model}}(-\log P(\text{data}|\text{model}) - \log P(\text{model}))$

Interpreting Terms in LLM Training:

1. $-\log P(\text{data}|\text{model})$: **Loss Function**
   - Negative log-likelihood of training data.
   - Minimizing it makes training data probable (e.g., cross-entropy loss).
2. $-\log P(\text{model})$: **Regularization Term**
   - From log prior; penalizes complexity (e.g., L2 regularization from a Gaussian prior).
   - Prevents overfitting, improves generalization.

Supervised Learning: Objective is $\min_{\text{model}}(\text{Loss Function} + \text{Regularization})$.

Special Case: Maximum Likelihood Estimation (MLE)

- With a uniform prior ($\log P(\text{model})$ is constant), MAP becomes MLE: $\max_{\text{model}} \log P(\text{data}|\text{model})$ (minimizing only the loss function).

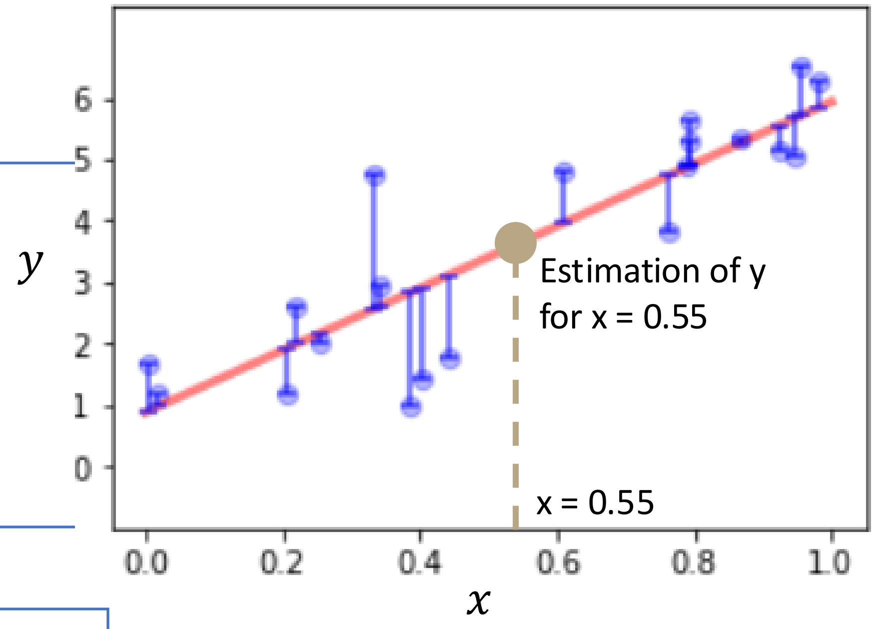
In-class quiz

Q1-3

Two categories of predictive models

An **regressor** predicts a numerical target, e.g.,

- monthly revenue
- GDP of a country
- crime rate of a region
- duration of a phone call



A **classifier** predicts a categorical target, e.g.,

- fraud, non-fraud
- default, non-default
- high risk, low-risk
- good, satisfactory, poor

Classification in financial applications

	Description	Solution	End goal
Internet fraud detection	▪ Stealing of credit card, login, personal details over internet ▪ A growing concern in banking and online payment, as it is hard to verify identity online	▪ Use classifier modelling in real time to **identify fraudulent transactions**, based on input variables, e.g., transaction amount, location, type of goods/services	▪ To flag/reject the suspicious transactions
Insurance fraud detection	Concealing, deceiving, and misrepresenting information to make a claim	▪ Use classifier modelling to **predict the likelihood of insurance fraud** based on input variables, e.g., size of claim, premium, previously reported fraud, insurer employment status, health conditions	▪ To flag/reject the suspicious claims

Is Linear Regression suitable as a classifier?

Simple linear regression

$$y = \beta_0 + \beta_1 x_1$$ in >= delta

Multiple linear regression

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k$$

Application

- Sales figure forecast
- No. of times being streamed for a new song on Spotify within the first month
- …

Advantages

- One simple equation
- Easy, fast and transparent to users

Limitations

- Target variable must be a continuous value
- Target variable can be $(-\infty, \infty)$; in practice usually within a specified range

Conclusion

- Is it the best model to predict categorical target variable?

Applying MLR to a classification problem

- **Social_Network_Ads.csv**
- Purchasing behaviour of people who have been exposed to a social media marketing advertisement campaign
- Input variables: Gender, Age, EstimatedSalary (UserID is not useful)
- Target variable: Purchased (0 or 1)

Convert 'Gender' column to 0 and 1

```
x_encoded = x_orig.copy()
x_encoded.loc[:,"Gender"] =
(x_encoded.loc[:,"Gender"] ==
"Female").astype(int)
```

Scale EstimatedSalary

```
x_encoded.loc[:,"EstimatedSalary"] =
x_encoded.loc[:,"EstimatedSalary"] / 1000
```

Train and Test - model evaluation method

- Training data
  - For training the predictive model – supervised or unsupervised learning?
  - Contains input variables and **known** target variable
  - Training data is **seen by the model**
- Testing data
  - For checking how well the model predicts target variable in the future
  - Contains input variables; but target variable is **hidden**
  - **NOTE:** Testing data is **unseen by the model during the training phase**
- How to select training and testing data sets
  - Usually by random sampling, e.g., randomly select 70% of the records as training data, and remaining 30% as testing data
  - Ratio of Training vs. Testing: usually more data in training set

Train and Test - implementation

Separate the data set into train vs. test subsets

```
x_train, x_test, y_train, y_test =
train_test_split(x_encoded, y,
test_size=0.3, random_state=12345)
```

stratify = y

Feed training data set into the model to get the model's prediction

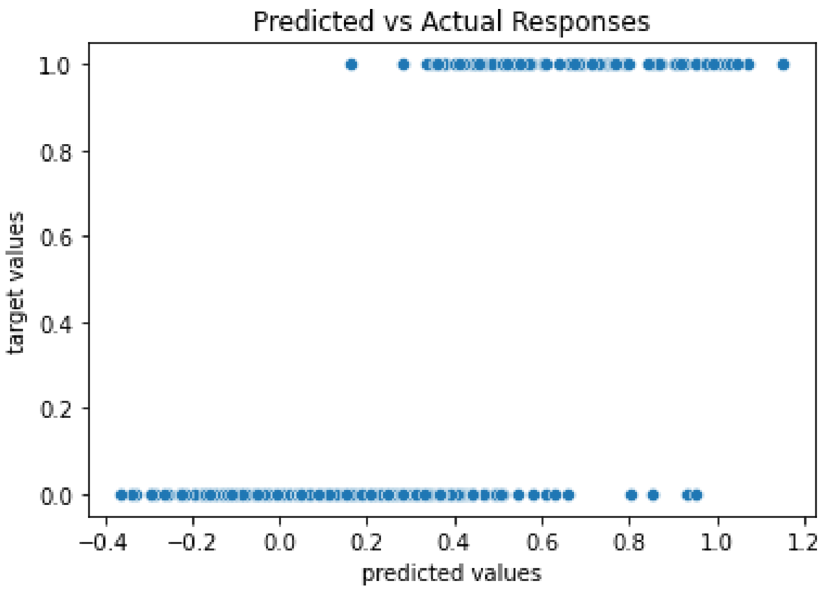
Feed testing data set into the model to get the model's prediction

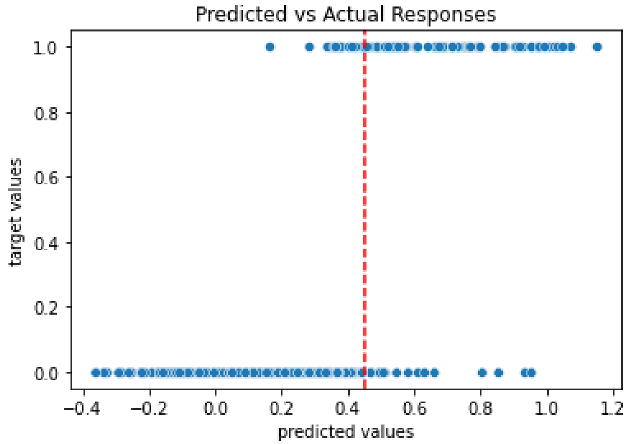
```
linreg_ols_pred_train =
linreg_ols.predict(x_train)
linreg_ols_pred_test =
linreg_ols.predict(x_test)
```

Transforming the model output to categorical target variable (1/2)

- How can we equate these predictions to the categorical target variable (Purchased should 0/1)?

- Find a threshold 0.45, based on the graph of actual values vs. predicted value of target variable ‘Purchased’





Transforming the model output to categorical target variable (2/2)

If a regression output is larger than 0.45, it would be transformed to True, then to a y value of 1

Compare the predicted y to the actual y value, to check accuracy

Do the same for test data set

```
y_pred_train = (linreg_ols_pred_train >
0.45).astype(int))

acc_train = y_pred_train == y_train
print("Accuracy:", acc_train.mean())

y_pred_test = (linreg_ols_pred_test >
0.45).astype(int)
acc_test = y_pred_test == y_test
print("Accuracy:", acc_test.mean())
```

Issues	▪ The cut-off value for regression output to be considered 1 or 0 was chosen by observation, without a guiding principle ▪ The Linear Regression model's output can be $(-\infty, \infty)$

Logistic Regression to the rescue

Logistic Regression

$$p = f(z) = \frac{1}{1 + e^{-z}}$$

where $z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k$

z is the linear combination of k input variables x_i, and it measures the overall effect of all the input variables

z is linear regression → place this z through an activation function to get a probability (a continuous value between 0 and 1)

Key facts

- Binary classification model to predict the probability of occurrence of an event, e.g.
  - probability of default on a loan
  - probability of buying insurance in response to an advertisement
- The target variable y is binary (1 or 0, Yes or No), depending on whether p is above or below a threshold
- y follows a Bernoulli distribution

$$y = \begin{cases} 1 & \text{with probability } p \\ 0 & \text{with probability } 1 - p \end{cases}$$

The loss function

- Quantifies the difference between the predicted probabilities and the actual class labels in the dataset.
- Minimize this cost function during the training process to obtain the best-fitting model.
- Binary cross-entropy loss:
  - **Loss(y, $\hat{y}$) = -[y * log($\hat{y}$) + (1 - y) * log(1 - $\hat{y}$)]**
- To obtain the overall cost function for logistic regression, average the binary cross-entropy loss over all data points in the training dataset:
  - **Cost = (1/N) * Σ Loss(y_i, $\hat{y}_i$)**

More on Logistic Regression

How Logistic Regression model is trained

Purpose of training is to obtain the coefficients β_i, $i = 0, 1, 2, \dots, k$

Once β_i is calculated, the logistic regression model $f(z)$ is trained

Usually, if $f(z) \geq 0.5$, then $\hat{y} = 1$; else, $\hat{y} = 0$

Exercise time to train logistic regression using logit() from statsmodels.formula.api

Assumptions

- The observations (rows) are independent of each other, and their target outcome follow the same Bernoulli distribution
- Little or no collinearity (i.e., low correlation) among the input variables
- No linear relationship between the target y and input variables
- Log odds of the probability of a target value y being 1 is linearly related to input variables

$$\text{log odds} = \log\left(\frac{p}{1-p}\right) = z$$
$$= \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k$$

(details later)

Sigmoid function and odds

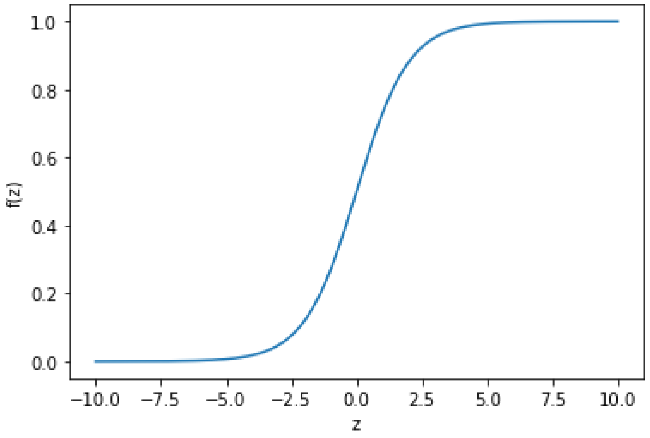
Definition of Sigmoid function

$$p = f(z) = \frac{1}{1+e^{-z}}$$

$z \rightarrow \infty, f(z) \rightarrow 1; z \rightarrow -\infty, f(z) \rightarrow 0$

Hence $f(z) \in [0,1]$

$f(z)$ represents the probability of an event



Definition of odds

$$\text{odds} = \frac{\textit{prob of event happening}}{\textit{prob of event not happening}} = \frac{p}{1-p}$$

$$p = \frac{1}{1+e^{-z}}$$

$$1-p = 1 - \frac{1}{1+e^{-z}} = \frac{e^{-z}}{1+e^{-z}}$$

$$\text{odds} = \frac{p}{1-p} = e^{z}$$

$$\text{i.e.} \log(\text{odds}) = \text{z}$$

Odds Ratio

- Suppose x_i is a binary input variable, x_i = 1 or 0
- Odds of x_i = 1: $\frac{p_1}{1-p_1} = e^z \,|\, x_i$ = 1, measures the chance of an event for x_i = 1, over the chance of a non-event
- Odds of x_i = 0: $\frac{p_0}{1-p_0} = e^z \,|\, x_i$ = 0, measures the chance of an event for x_i = 0 **over** the chance of a non-event
- **Odds Ratio of x_i:** the ratio of the odds of x_i = 1 to the odds of x_i = 0

$$\frac{p_1}{1-p_1} \Big/ \frac{p_0}{1-p_0} = \frac{e^{\beta_0+\beta_1 x_1+\ldots+\beta_i*1+\cdots+\beta_k x_k}}{e^{\beta_0+\beta_1 x_1+\ldots+\beta_i*0+\cdots+\beta_k x_k}} = e^{\beta_i}$$

- e^{β_i} measures the **quantified impact** of the **binary** input variable x_i on the odds of the outcome y being 1, while all other input variables remain unchanged
- Recall for Multiple Linear Regression:

$$y + \Delta y = \beta_0 + \beta_1 x_1 + \cdots + \beta_j (x_j + \Delta x_j) + \cdots$$

$$\Delta y = \beta_j \, \Delta x_j$$

How to interpret Odds Ratio

- For categorical input variable
  - E.g., gender (0: male, 1: female) may be related to whether the insurance is purchased (1: yes; 0: no)
  - Base category: male set as 0
  - If the estimated β of gender is 0.2, then OR = $e^{0.2} \approx 1.22$
  - **Odds** of **female** customers to purchase the insurance is **1.22 times** the **odds** of their **male** counterparts to purchase the insurance, assuming Ceteris Paribus
- For numerical input variable
  - E.g., age may be related to whether the insurance is purchased
  - No need to set base category
  - If the estimated β of age is 0.3, then OR = $e^{0.3} \approx 1.35$
  - **Odds** of a client to purchase is **1.35 times** the **odds** of similar people who are 1 year younger, assuming Ceteris Paribus
  - What if β is -0.3?

How to evaluate Logistic Regression model

Metrics

Accuracy rate
Error rate
Precision
Recall
Sensitivity (true positive rate)
Specificity (true negative rate)
AUC

Principle of parsimony

- If two competing models provide the similar level of fit to the data, the one with fewer input variables should be picked
- The most accurate model is not necessarily the best model

Model metrics – various rates

$n = a + b + c + d$

Confusion matrix

		Predicted $Y=0$	Predicted $Y=1$	Total
true	Non-Event $Y=0$	a	c	$a+c$
	Event $Y=1$	b	d	$b+d$
	Total	$a+b$	$c+d$	n

- **Accuracy (ACC) rate**
  - ACC = $(a+d)/n$
- **Error rate**
  - Error rate = 1 - ACC = $(b+c)/n$
- Positive predictive value (PPV), or **Precision**
  - **Precision** = $d/(c+d)$
  - Out of all the positive predictions, what percentage is actually positive?

True positive rate (TPR), or sensitivity, **Recall**

Recall = $d/(b+d)$

Out of all the events, what percentage are predicted correctly as positive?

True negative rate (TNR), or **Specificity**, selectivity

Specificity = $a/(a+c)$

Out of all the non-events, what percentage are predicted correctly as negative?

Exercise time to manually calculate ACC, Precision, Recall

How changing the classification threshold might impact the metrics of a classifier

Low threshold 0.3
- Accuracy: 0.79
- Precision: 0.79
- Recall: 0.68

Default threshold 0.5
- Accuracy: 0.74
- Precision: 0.85
- Recall: 0.46

High threshold 0.8
Accuracy: 0.69
Precision: 0.88
Recall: 0.3

In this particular example, increasing the threshold leads to

- Accuracy decreasing
- Recall decreasing
- Precision increasing

Implication for classifiers in general

- Precision and Recall usually have an inverse relationship with respect to the adjustment of the classification threshold
- Reviewing both precision and recall is useful for cases where there is a huge imbalance in the target variable's values

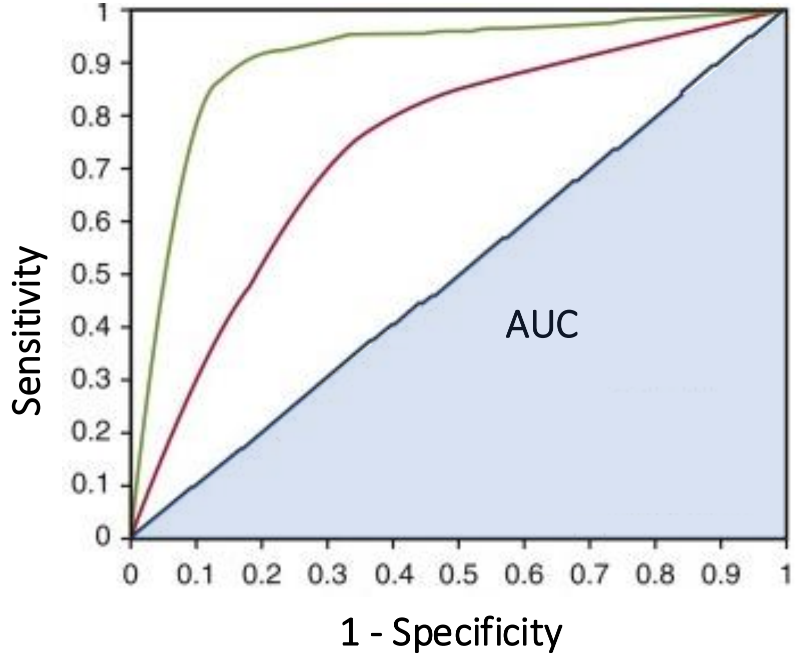
Optimize for Precision or Recall?

Recall (True positive rate TPR) = 1 – False Negative Rate (FNR)

	Positive class	Interpreting False Negative	How bad is FN?	Optimize for?
Spam filter	▪ Spam	▪ Spam goes to the inbox	▪ Acceptable	▪ Precision
Fraudulent transaction detector	Fraud	▪ Fraudulent transactions that are not detected	▪ Very bad	▪ Recall
Cancer Diagnose	▪ Cancer	▪ Test for cancer shows up as negative even though the patient has cancer	▪ Very bad	▪ Recall

Model metrics – Area under ROC curve (AUC)

- ROC curve (receiver operating characteristic curve)
  - Plot Sensitivity vs. (1-Specificity), i.e., TPR vs. (1-TNR), or TPR vs. FPR
- As the classification threshold goes up
  - FPR goes down
  - Leftward movement on the curve
- A perfect classifier (0,1) has AUC score of 1
  - 1 – specificity: 0, i.e., FPR is 0 (no false positive, i.e., all negative cases are not predicted as positive)
  - Sensitivity: 1, i.e., TPR is 1 (all positive cases are predicted as positive correctly)
- A randomly guessing classifier has AUC score of 0.5 (area under the **blue** ROC)



Dealing with imbalanced data

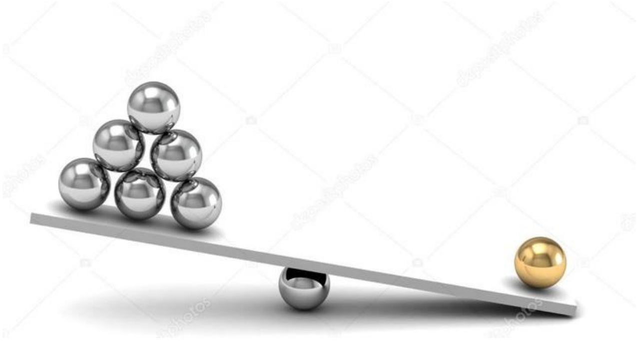
Issues

Target variable's values are not distributed equally

Very common in real-world applications

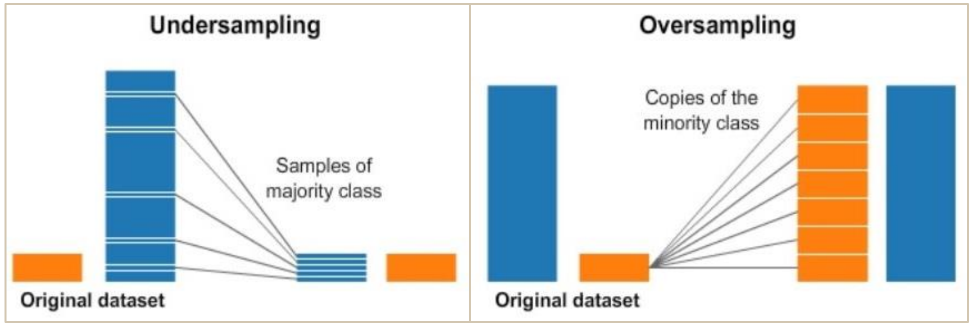
- Fraudulent transactions in banks
- Spam/phishing emails for employees in banks
- Identification of rare diseases, e.g., cancer
- Natural disasters, e.g., earthquakes

Classification performance may be **dominated by the majority class**, i.e., metric fool



Popular solutions

- Re-design the data collection or collect more data
- Change the performance evaluation method
- **Data resampling**: oversampling and/or under-sampling to make the distribution less imbalanced



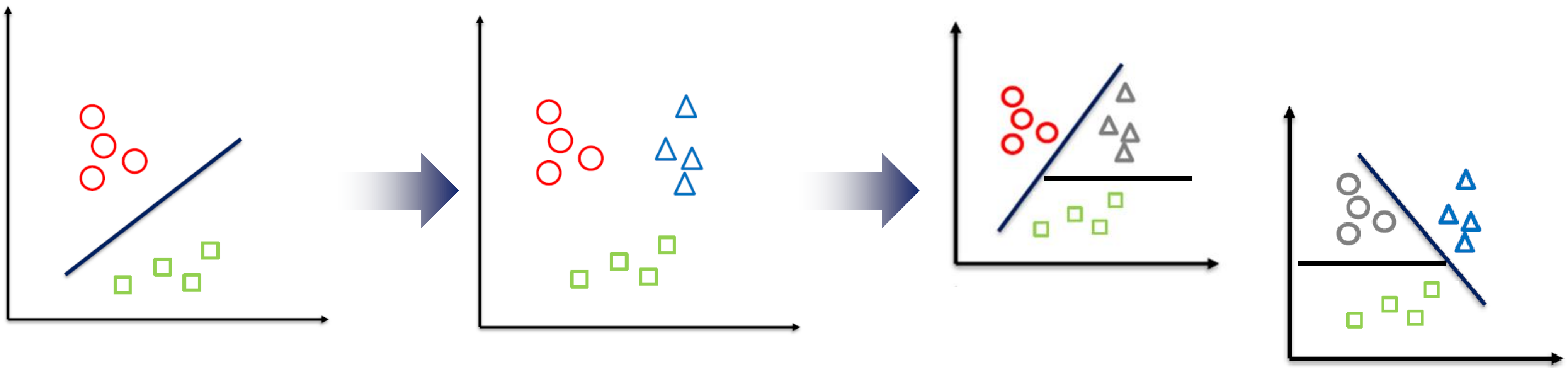
Extending binary classification to multiclass classification

Multiclass classification

- S&P bond rating: AAA, AA, A…
- Corporate client accounts in a bank: good credit, past due, overdue and doubtful

Multinomial logistic regression

- Generalizes logistic regression to multiclass problems
- Decomposed as a set of independent binary logistic regressions



Mathematical derivation to obtain the multinomial logistic regression

For binary target variable y

$$\log(\text{odd}) = \log\left(\frac{p}{1-p}\right) = \log\left(\frac{P(y=1)}{P(y=0)}\right)$$

$$= \text{z}$$

$$\frac{P(y=1)}{P(y=0)} = e^z$$

Since $P(y=1) + P(y=0) = 1$

Therefore:

$$P(y=1) = \frac{1}{1+e^{-z}}$$

$$P(y=0) = \frac{e^{-z}}{1+e^{-z}}$$

Assume target variable y have 3 values, 0, 1, 2

- Choose 0 as the pivot value
- $\frac{P(y=1)}{P(y=0)} = e^{z_1}$ (z_1 is a linear combination of all input x_i)
- $\frac{P(y=2)}{P(y=0)} = e^{z_2}$ (z_2 is another linear combination of all input x_i)

sigmoid

1/(1 + e^{-z})

- $P(y=2):P(y=1):P(y=0) = e^{z_2}:e^{z_1}:1$
- Since $P(y=2) + P(y=1) + P(y=0) = 1$
- Therefore:

$$\left.\begin{aligned} P(y=2) &= e^{z_2}/(e^{z_2} + e^{z_1} + 1) \\ P(y=1) &= e^{z_1}/(e^{z_2} + e^{z_1} + 1) \\ P(y=0) &= 1/(e^{z_2} + e^{z_1} + 1) \end{aligned}\right\} \text{softmax}$$

{z_1, z_2, z_3}

{0.1, 0.2, 0.7}

e^{z_1}/(e^{z_1} + e^{z_2} + e^{z_3})

In-class quiz

Q4-6