STAT1005 Foundations of Data Science

Lecture (9): Classification & Logistic regression

Yuanhua Huang (黃淵華)

Office: Rm 1-05E, 1/F, JCBIR, 5 Sassoon Road (Medical Campus)
Q&A contact hours: Wed 3-5pm

Email: yuanhua@hku.hk | Web: https://web.hku.hk/~yuanhua

Nov 8, 2021

Department of Statistics & Actuarial Science 統計及精算學系
THE UNIVERSITY OF HONG KONG

HKU Med LKS Faculty of Medicine
School of Biomedical Sciences
香港大學生物醫學學院

Objectives today

1. Logistic regression
   i. Sigmoid / logit function
   ii. Gradient based optimization
   iii. Feature selection
2. Evaluation of classification performance
   i. Generalization, test set, and cross-validation
   ii. Confusion matrix, scoring metrices, and ROC curve
3. More classification methods
   i. Quick introduction to Naïve Bayes

- Wiki: https://en.wikipedia.org/wiki/Logistic_regression
- Scikit learn: https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
- Notebooks: https://github.com/huangyh09/foundation-data-science/

Acknowledgement: some notes from Prof. Jeff Yao (last year) and Prof. Nigel Goddard (U Edinburgh)

Data classification or data clustering

- Classification techniques: essential part of machine learning and data mining applications.
- Large proportion of data analysis problems are classification (60-80%)

  "In machine learning and statistics, classification is the problem of identifying to which of a set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known."

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

- Lots of classification solutions available: k-means, SVM, deep neural networks, random forests;
- But, Logistic Regression is a common and efficient regression method for solving classification problems.

Binary classification

Given a set of features (predictors) of a subject, the aim is to classify it into two categories (binary).

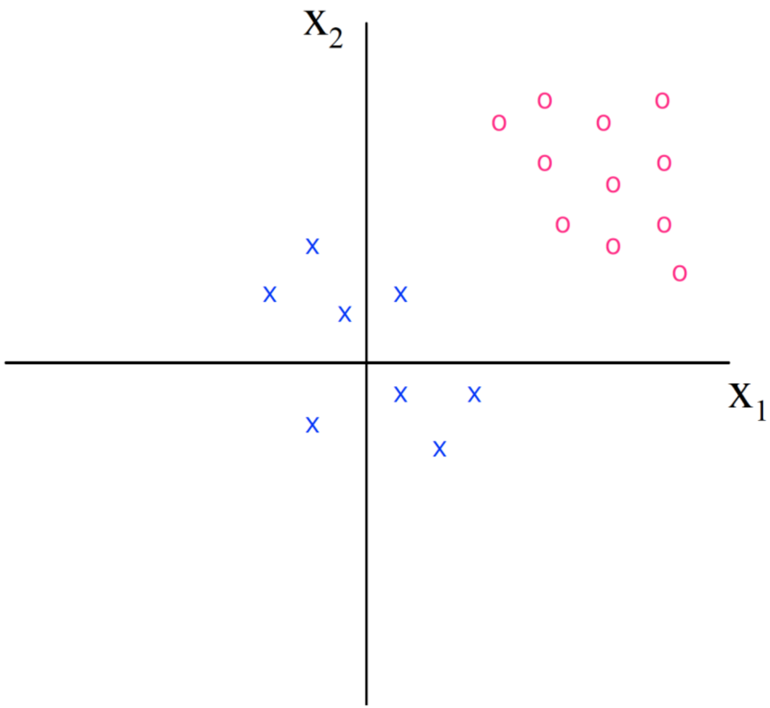
Examples

- Email filtering: is this email a spam?
  - {mass email, advertising business, commercial photos, senders, ... } { Yes, No }

- Admission: will an applicant be admitted to the prestigious Master of Data Science at HKU?
  - {Bachelor reputation, GPA, research experience, projects, English, ... } { Yes, No }

- Skin lesion detection: does this image come from a skin cancer?
  - {Colour values of 256 x 256 pixels, ... } { Yes, No }

1.1 Linear classifier | Example with two-dimensional data



In a two-class linear classifier, we learn a function

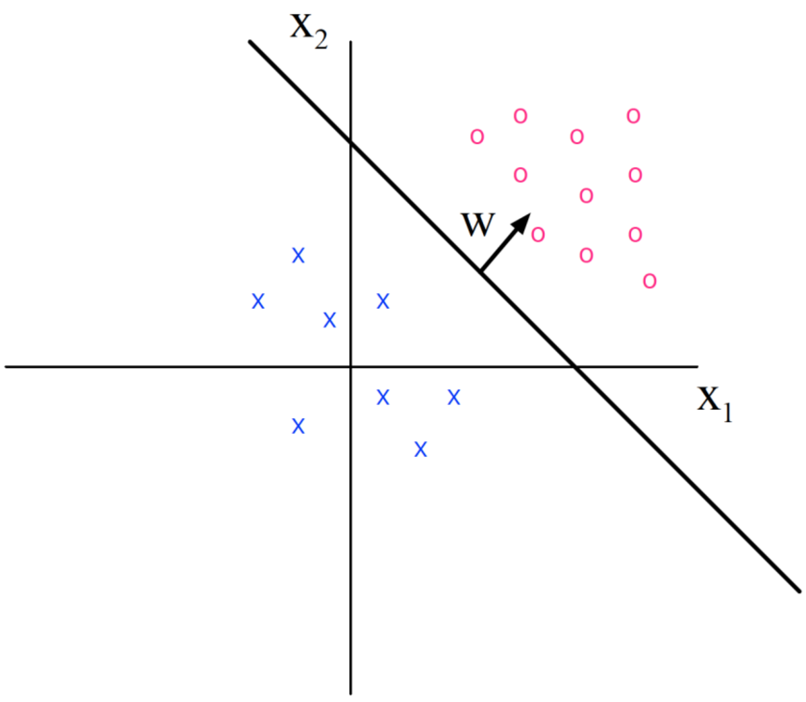
$$F(x_1, x_2|\boldsymbol{w}) = w_0 + w_1x_1 + w_2x_2$$

that represents how aligned the instance is with y = 1.

- $\boldsymbol{w} = (w_0, w_1, w_2)$ are parameters of the classifier that we learn from data.

- To do classification of an input x: $(x_1, x_2) \rightarrow (y = 1)$ if $F(x_1, x_2|\boldsymbol{w}) > 0$

1.1 Linear classifier | Example with two-dimensional data



- We have a linear classifier via function
  $F(x_1, x_2|\boldsymbol{w}) = w_0 + w_1x_1 + w_2x_2$

- To do classification of an input x:
  $(x_1, x_2) \rightarrow (y = 1)$ if $F(x_1, x_2|\boldsymbol{w}) > 0$

- The decision boundary here is
  $F(x_1, x_2) = w_0 + w_1x_1 + w_2x_2 = 0$

1.1 Linear classifier | Multi-dimensional predictors

- For *p* dimensional predictors (*p*>2), we still can have linear classifier. This boundary will be a hyperplane.
- Now, we rewrite in a vector form:

$$\mathbf{x} = (1, x_1, \dots, x_p), \mathbf{w} = (w_0, w_1, \dots, w_p)$$

- The decision boundary here is

$$F(\mathbf{x}|\mathbf{w}) = \mathbf{x}^{\mathsf{T}}\mathbf{w} = w_0 + w_1 x_1 + \cdots + w_p x_p = 0$$

- To do classification of an input x:

$$(x_1, x_2, \dots, x_p) \rightarrow (y = 1) \text{ if } F(\mathbf{x}|\mathbf{w}) > 0$$

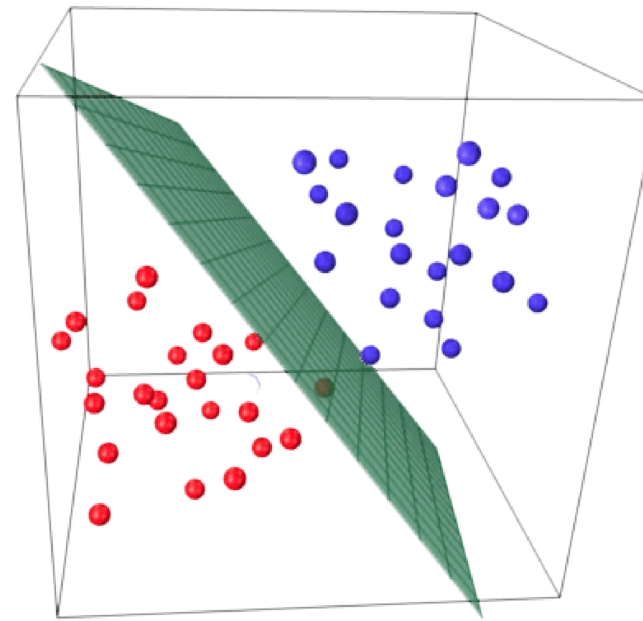


Figure from https://mlcourse.ai/articles/topic4-part2-logit/

1.1 Linear classifier | Probabilistic prediction

- We have defined a linear classifier of an input x:

$$\left(x_1, x_2, \dots, x_p\right) \rightarrow (y = 1) \text{ if } F(\mathbf{x}|\mathbf{w}) = \mathbf{x}^\top \mathbf{w} > 0$$

- Now we want to have a probabilistic outcome: $P(y = 1 \mid x_1, x_2, \dots, x_p)$
- We could simply try

$$P(y = 1 \mid x_1, x_2, \dots, x_p) = w_0 + w_1 x_1 + \cdots + w_p x_p$$

but it is stupid! The range is $[-\infty, +\infty]$, not valid for probability ranging [0, 1]

- Instead, what we will do is

$$P(y = 1 \mid \mathbf{x}) = f\left(w_0 + w_1 x_1 + \cdots + w_p x_p\right) = f(\mathbf{x}^\top \mathbf{w})$$

- Function $f()$ must return value between 0 and 1; It squashes the real line.
- Furthermore, the fact that probabilities sum to one means

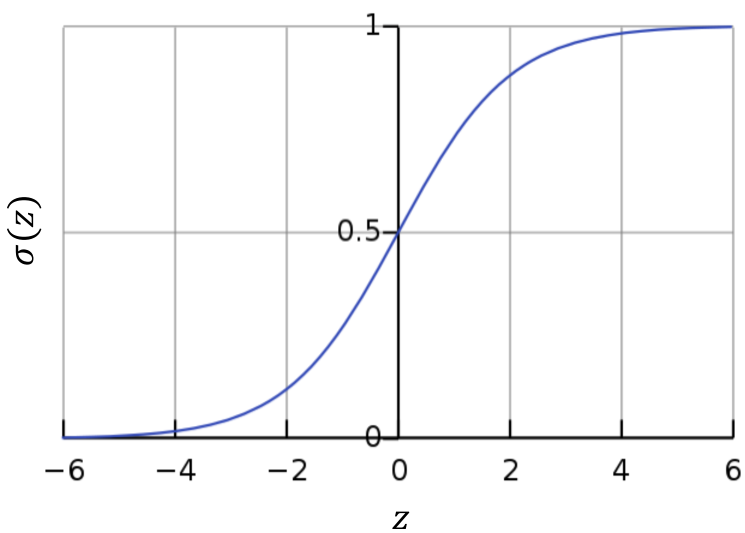
$$P(y = 0 \mid \mathbf{x}) = 1 - f(\mathbf{x}^\top \mathbf{w})$$

1.1 Linear classifier | Logistic function (sigmoid function)

- We need a function that returns probabilities (i.e., stays between 0 and 1).
- The logistic function provides this

$$P:\ f(z) = \sigma(z) \equiv \frac{1}{1+\exp(-z)}$$

- It has a "sigmoid" shape (i.e., S-like shape)



As z goes from $-\infty$ to $+\infty$, so f goes from 0 to 1, a "squashing function".

$Z = 0: \sigma(z) = 0.50$
$Z = 1: \sigma(z) = 0.73$
$Z = -1: \sigma(z) = 0.27$
$Z = 2: \sigma(z) = 0.88$

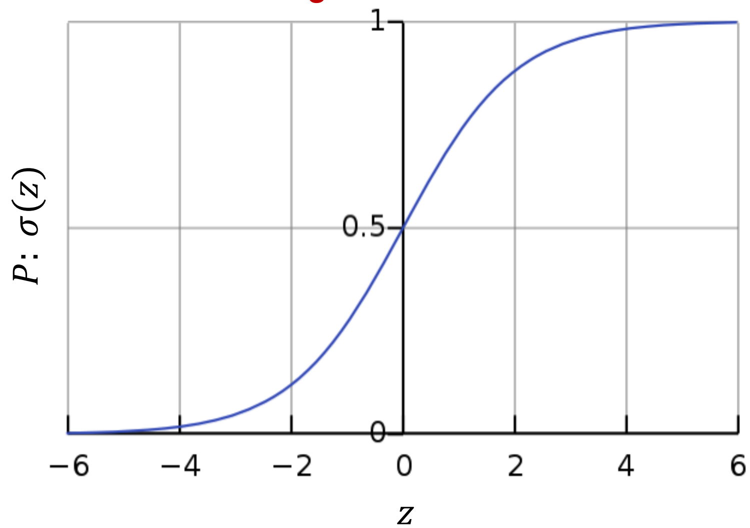
Classification:
$P = \sigma(z) > 0.5$:
$(x_1, x_2, \ldots, x_p) \rightarrow (y = 1)$

1.1 Linear classifier | Logistic function & logit function

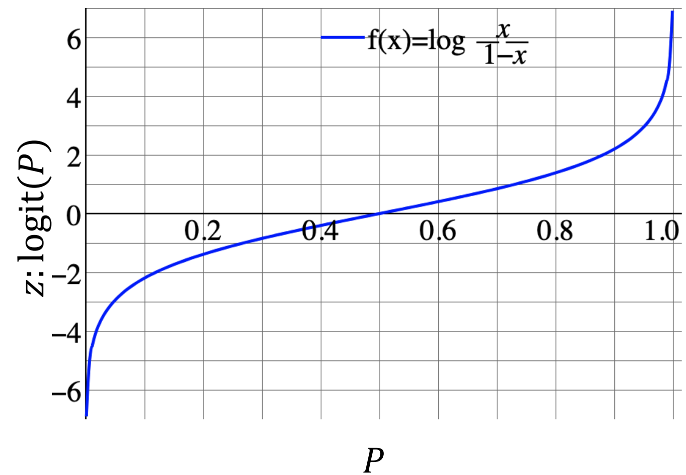
- Odds ratio between A (positive) and B (negative) events: P / (1-P)
- Logit function: z = log(P/(1-P)) = log(odds ratio)
- Z increases by dz means odds increases by exp(dz)

- Logit function is the inverse function of logistic function

Logistic function



Logit function



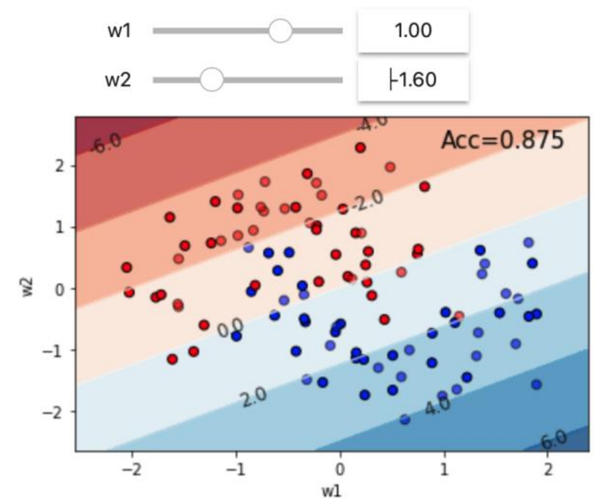
1.1 Linear classifier | Linear weights

- Linear weights + logistic squashing function == logistic regression.
- We model the probability of positive class as

$$P(y = 1|x) = \sigma(\mathbf{x}^\top \mathbf{w}) = \sigma(w_0 + w_1 x_1 + \cdots + w_p x_p)$$

- $\sigma(z) = 0.5$ when $z = 0$. Hence the decision boundary is given by $\mathbf{x}^\top \mathbf{w} = 0$.

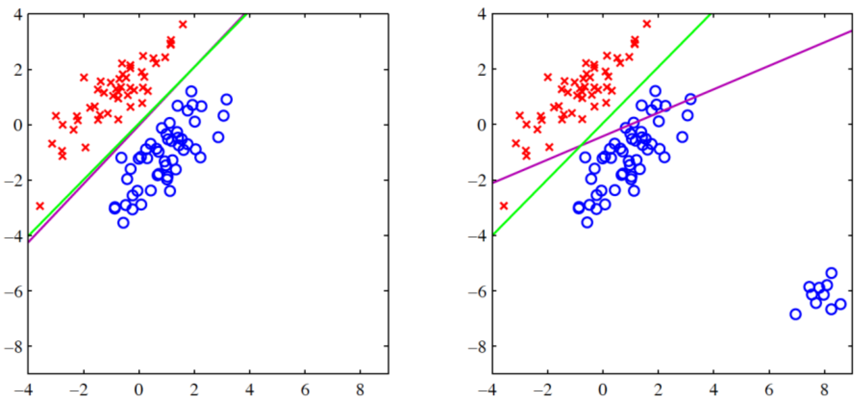
Decision boundary is a p-1 hyperplane for a p dimensional problem.



Colab Notebook:
https://bit.ly/3kcisoU

1.2 Fitting logistic regression | likelihood function

- How do we determine whether we have achieved a good decision hyperplane?
- Can we use least squares, which has analytical solution?
  - $SSE = \sum_{i=1}^{n}(y_i - \mathbf{x}^\top \mathbf{w})^2$
  - Not good in general, observed y is categorical (0 or 1) not numerical
- Another option: maximum likelihood (next slides)



Green: logistic regression (maximum likelihood);
Purple: least-squares regression;

Least squares method is sensitive to outliers, more common in X space.

Figure credit: Chris Bishop, PRML

1.2 Fitting logistic regression | likelihood function

- Assume data is independent and identically distributed.
- Call the data set $D = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots (\mathbf{x}_n, y_n)\}$
- The likelihood is

$$p(D|\mathbf{w}) = \prod_{i=1}^{n} p(y = y_i|\mathbf{x}_i, \mathbf{w})$$
$$= \prod_{i=1}^{n} p(y = 1|\mathbf{x}_i, \mathbf{w})^{y_i} \left(1 - p(y = 1|\mathbf{x}_i, \mathbf{w})\right)^{1-y_i}$$

Likelihood:
describes the joint probability of the observed data as a function of the parameters of the chosen statistical model

- Hence the log likelihood $L(\mathbf{w}) = \log p(D|\mathbf{w})$ is given by

$$L(\mathbf{w}) = \sum_{i=1}^{n} y_i \log \sigma(\mathbf{w}^\top \mathbf{x}_i) + (1 - y_i) \log(1 - \sigma(\mathbf{w}^\top \mathbf{x}_i))$$

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

1.2 Fitting logistic regression | maximum likelihood

- It turns out that the log likelihood function has a unique optimum (given sufficient training examples). It is convex.

- How to maximize? Necessary condition: all partial derivatives are 0.

$$\frac{\partial L(w_0, w_1, \dots, w_p)}{\partial w_i} = 0 \text{ for any dimension } i$$

- No closed-form solution is available for the optimum values of the parameters

$$\hat{\mathbf{w}} = \left(\hat{w}_0, \hat{w}_1, \dots, \hat{w}_p\right)$$

- Rather, we need numerical procedures to find these estimates of the parameters

1.2 Fitting logistic regression | gradient descent method

Actually, optimising the likelihood function is the general structure for learning algorithms

- Define the task: classification, discriminative
- Decide on the model structure: logistic regression model
- Decide on the score function: log likelihood
- Decide on optimization/search method to optimize the score function: numerical optimization routine. Note we have several choices here, gradient descent and its many variants (stochastic gradient descent, BFGS).

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

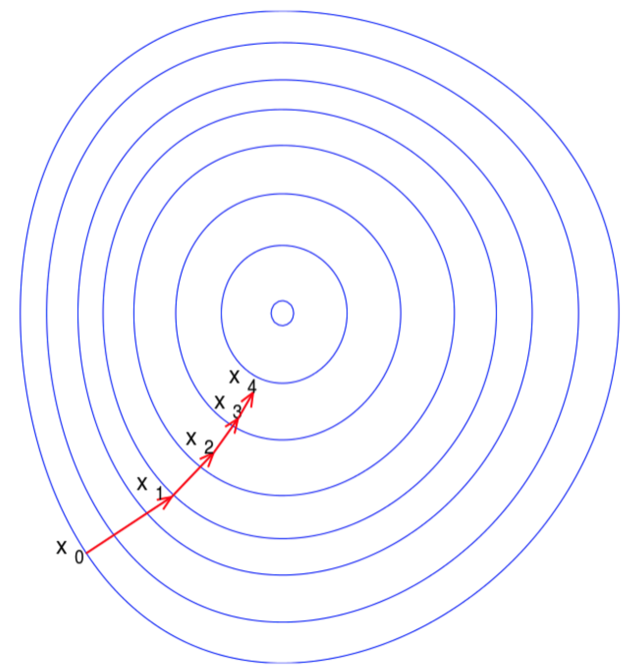


Illustration of gradient descent on a series of level sets. X are the parameter values, indexed by updating steps

1.3 Example | Diabetes diagnosis (PIMA Indians Diabetes)

- Diabetes is a chronic (long-lasting) health condition: body can't turn food (sugar) to energy, because either can't make enough or can't use insulin.

- ~700,000 individuals in Hong Kong (~10%) suffers from this disease.

- Can we diagnostically predict if this condition exists or not from easy-to-access measurements?

Example dataset (acknowledge to PIMA Indians Diabetes Database)

- 768 female individuals at least 21 years old of Pima Indian heritage;
- 8 medical predictor (independent) variables and 1 target (dependent) variable;
- Independent variables include their age, BMI, insulin level, glucose and so on.

1.3 Example | Diabetes diagnosis as a classification task

Diabetes diagnosis as a classification problem

- Given the input variables

$$X = \{\text{Pregnancies, Glucose, BP, Skin, Insulin, BMI, Pedigree, Age}\}$$

Should we classify (diagnose) the person to “having diabetes” (y)?

```
pima.head()
```

	pregnant	glucose	bp	skin	insulin	bmi	pedigree	age	diabetes
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

Aim: to build a model and fit to the observed data, so able to predict the output for a new person with values of input variables

1.3 Example | Diabetes diagnosis as a classification task

Logistic regression and the parameters:

$$P = \sigma(w_0 + w_1 * \text{Pregnacies} + w_2 * \text{Glucose} + w_3 * BP + w_4 * \text{Skin} + w_5 * \text{BMI} + w_6 * \text{Pedigree} + w_7 * \text{Age})$$

Fit parameters by maximizing likelihood function over training data.

Maximized value of

Log Liklihood = -277.93

LLR p-value = 6.246e-38 (~ 0)

(equivalent to the F-statistic in a linear regression model and its p-value)

```
log_reg.summary()
```

Logit Regression Results

Dep. Variable:	diabetes	**No. Observations:**	576
Model:	Logit	**Df Residuals:**	567
Method:	MLE	**Df Model:**	8
Date:	Sun, 07 Nov 2021	**Pseudo R-squ.:**	0.2600
Time:	10:36:00	**Log-Likelihood:**	-277.93
converged:	True	**LL-Null:**	-375.58
Covariance Type:	nonrobust	**LLR p-value:**	6.246e-38

	coef	std err	z	P>|z|	[0.025	0.975]
const	-8.4218	0.822	-10.240	0.000	-10.034	-6.810
pregnant	0.0869	0.036	2.448	0.014	0.017	0.157
glucose	0.0332	0.004	7.802	0.000	0.025	0.042
bp	-0.0112	0.006	-1.815	0.070	-0.023	0.001
skin	0.0059	0.008	0.728	0.466	-0.010	0.022
insulin	-0.0010	0.001	-1.011	0.312	-0.003	0.001
bmi	0.0880	0.017	5.103	0.000	0.054	0.122
pedigree	0.8935	0.342	2.613	0.009	0.223	1.564
age	0.0220	0.011	2.049	0.040	0.001	0.043

1.4 Model diagnosis | Compare linear & logistic regressions

	Linear regression	**Logistic regression**	**Comments**
Objective function for parameter estimation	Sum of squares $S(\alpha, \beta_{1,..,}\beta_p)$ to minimize; Closed-form solution for parameters	Likelihood function $L(\alpha, \beta_{1,..,}\beta_p)$ to maximize; Numerical solutions using numerical solvers	On training data
Significance check of individual parameters	p-value	p-value	On training data
Goodness-of-fit statistics	- Adjust R^2 - Fisher statistic and its p-value - Error (= SRE/mean)	- Log-likelihood value and its p-value - Prediction accuracy	On training data
Performance on test data	- Error (= SRE/mean)	- Prediction accuracy	On test data

1.4 Model diagnosis | feature selection in logistic regression

Fitted model with all the 8 input variables

- Severable variables have p-values > 5% (too large), which mean that they are not significant (given the presence in the model of other variables)
- By removing such redundancy, one may improve the model prediction performance on test data (new data)

```
log_reg.summary()
```

Logit Regression Results

Dep. Variable:	diabetes	**No. Observations:**	576
Model:	Logit	**Df Residuals:**	567
Method:	MLE	**Df Model:**	8
Date:	Sun, 07 Nov 2021	**Pseudo R-squ.:**	0.2600
Time:	10:36:00	**Log-Likelihood:**	-277.93
converged:	True	**LL-Null:**	-375.58
Covariance Type:	nonrobust	**LLR p-value:**	6.246e-38

	coef	std err	z	P>|z|	[0.025	0.975]
const	-8.4218	0.822	-10.240	0.000	-10.034	-6.810
pregnant	0.0869	0.036	2.448	0.014	0.017	0.157
glucose	0.0332	0.004	7.802	0.000	0.025	0.042
bp	-0.0112	0.006	-1.815	0.070	-0.023	0.001
skin	0.0059	0.008	0.728	0.466	-0.010	0.022
insulin	-0.0010	0.001	-1.011	0.312	-0.003	0.001
bmi	0.0880	0.017	5.103	0.000	0.054	0.122
pedigree	0.8935	0.342	2.613	0.009	0.223	1.564
age	0.0220	0.011	2.049	0.040	0.001	0.043

1.4 Model diagnosis | “Skin” removed

- Accuracy is calculated on 25% instances as test test

Model	log-L	Accuracy on test data	Comments
Full model	-277.93	80.2%	
`skin` removed	-278.20	80.7%	better

Can we further improve the model?

Logit Regression Results

Dep. Variable:	label	**No. Observations:**	576
Model:	Logit	**Df Residuals:**	568
Method:	MLE	**Df Model:**	7
Date:	Sun, 07 Nov 2021	**Pseudo R-squ.:**	0.2593
Time:	17:37:02	**Log-Likelihood:**	-278.20
converged:	True	**LL-Null:**	-375.58
Covariance Type:	nonrobust	**LLR p-value:**	1.472e-38

	coef	std err	z	P>|z|	[0.025	0.975]
const	-8.4226	0.823	-10.240	0.000	-10.035	-6.810
pregnant	0.0881	0.035	2.491	0.013	0.019	0.157
glucose	0.0327	0.004	7.823	0.000	0.025	0.041
bp	-0.0104	0.006	-1.714	0.087	-0.022	0.001
insulin	-0.0007	0.001	-0.775	$\boxed{0.438}$	-0.003	0.001
bmi	0.0921	0.016	5.621	0.000	0.060	0.124
pedigree	0.9124	0.342	2.668	0.008	0.242	1.583
age	0.0207	0.011	1.961	0.050	9.63e-06	0.041

1.4 Model diagnosis | “Skin” and “Insulin” removed

- Accuracy is calculated on 25% instances as test test

Colab Notebook: https://bit.ly/3ES8AIX

Model	log-L	Accuracy on test data	Comments
Full model	-277.93	80.2%	
`skin` removed	-278.20	80.7%	better
`skin & insulin` removed	-278.50	79.7%	Test performance slightly worse

Despite a small decrease in prediction performance, the 3rd model. It has all its variables being significant with acceptable p-values.

This is the preferred and recommended model

Logit Regression Results

Dep. Variable:	label	**No. Observations:**	576
Model:	Logit	**Df Residuals:**	569
Method:	MLE	**Df Model:**	6
Date:	Sun, 07 Nov 2021	**Pseudo R-squ.:**	0.2585
Time:	17:39:20	**Log-Likelihood:**	-278.50
converged:	True	**LL-Null:**	-375.58
Covariance Type:	nonrobust	**LLR p-value:**	3.304e-39

	coef	std err	z	P>|z|	[0.025	0.975]
const	-8.3490	0.815	-10.247	0.000	-9.946	-6.752
pregnant	0.0886	0.035	2.510	0.012	0.019	0.158
glucose	0.0317	0.004	8.011	0.000	0.024	0.039
bp	-0.0105	0.006	-1.736	0.083	-0.022	0.001
bmi	0.0909	0.016	5.594	0.000	0.059	0.123
pedigree	0.8810	0.340	2.593	0.010	0.215	1.547
age	0.0219	0.010	2.091	0.037	0.001	0.042

Part 2: Performance evaluation

2.1 Generalization | Training & future data

- Training data: $\{x_i, y_i\}$
  - Data used to train the model
- Future data: $\{x_i, ?\}$
  - Examples that our classifier has never seen before
- We care more on the errors in future data

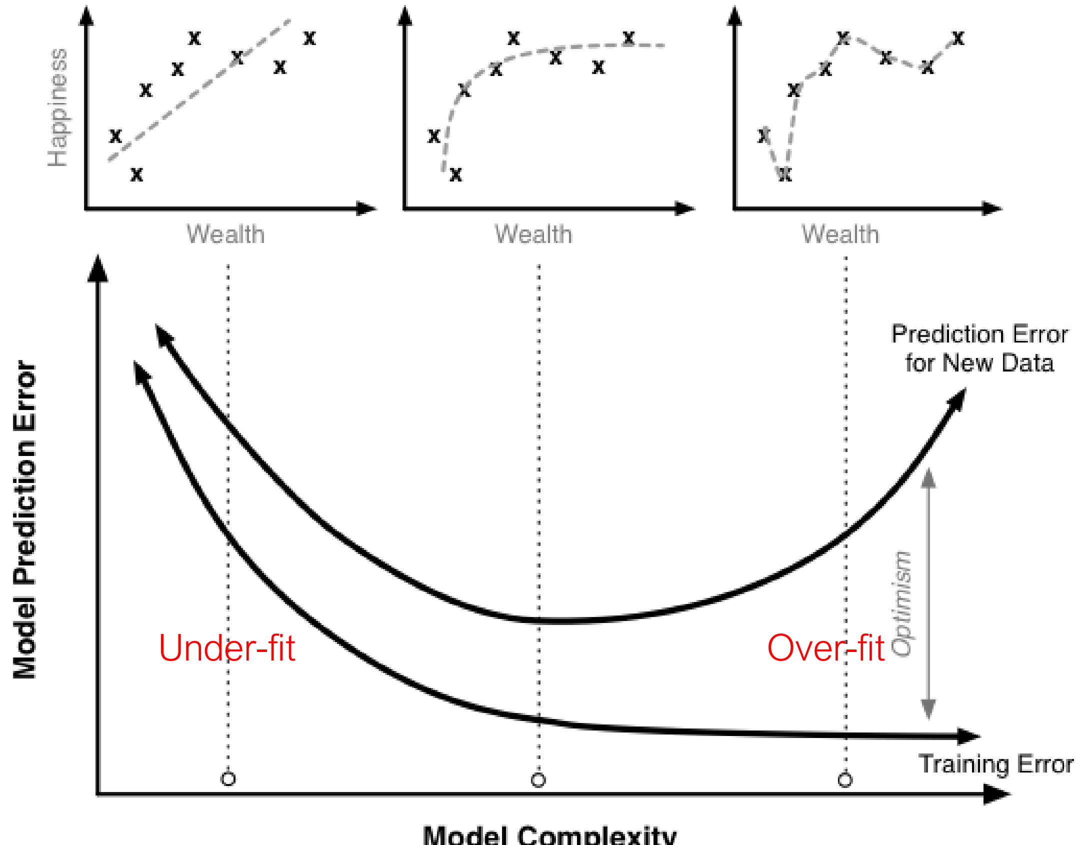


Figure credit: https://www.quora.com/What-is-generalization-in-machine-learning

2.1 Generalization | Evaluation on test data set

- Ideally, future data may be collected when classifier is trained, to evaluate the model performance and generalization

- Generally, collecting new data is costly. We could consider splitting the collected data into training and test data sets.
  - For example, 75% instances for training, and 25% instance for testing & evaluating the model performance
  - Ensure that the test data is not touched during training
  - Shuffle the data before splitting them to avoid bias

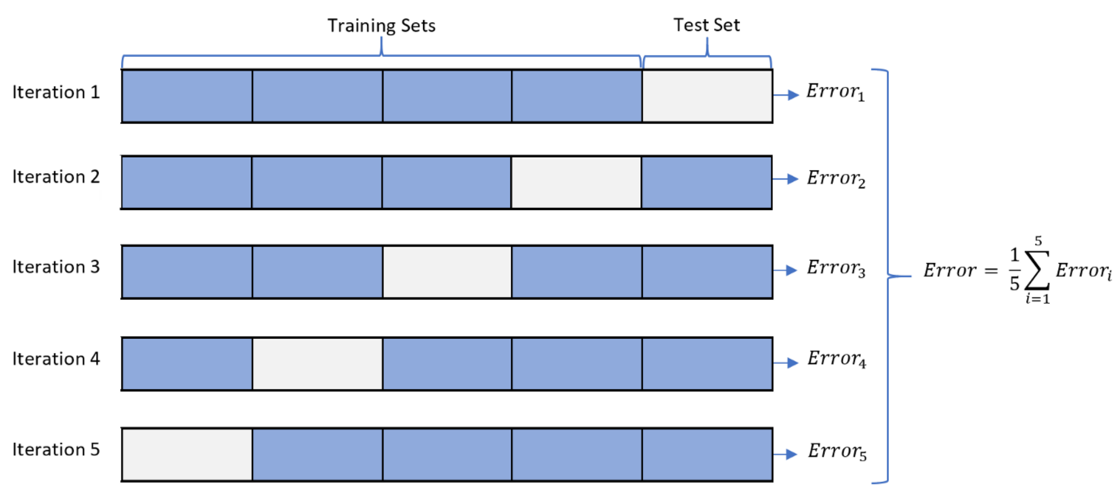
2.1 Generalization | Cross-validation

- When the collected data size is not big, the test set (e.g., 25%) may be too few to report the performance. Better to use all of them to evaluate the model.
- Cross-validation (with k-fold, e.g., 5-fold)
  - Alternately use subset (1/K of the full data) as test data set
  - Repeating K times, so as all data points are used for test
  - Combining all folds to have an overall evaluation

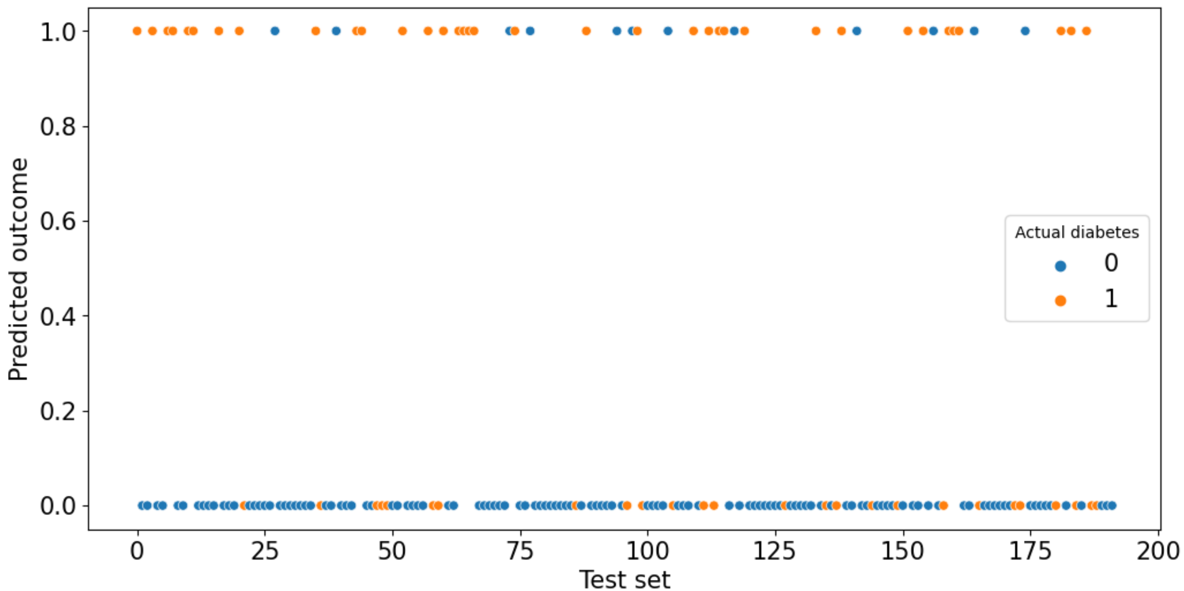
3, 5, or 10 folds are commonly used.

Leave-one-out: n-fold; n is the number instances.

The more folds, the larger the training set, but more computing time required.



2.2 Performance metrics | Example on diabetes



Scatter plot of actual response v.s. predicted response

If P>0.5: predict to be positive outcome

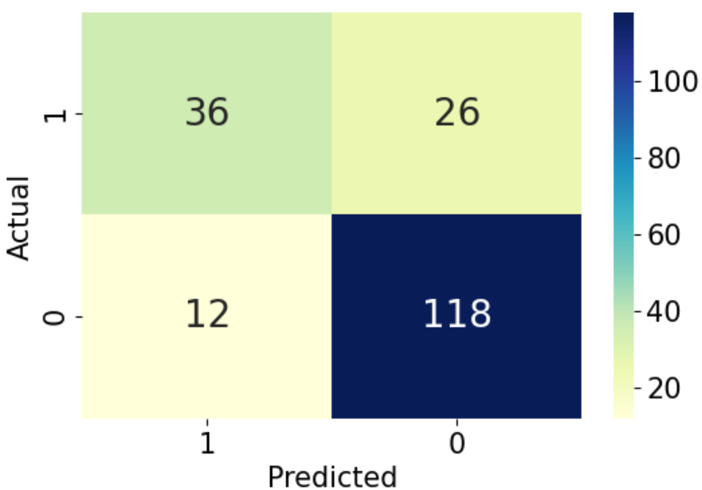
- Performance on test data
  75% - 25% train-test split
  576 -192 share
- Two type of errors:
  actual = 1 & predicted =0
  actual = 0 & predicted = 1
- Two type of correctness:
  actual = predicted = 0
  actual = predicted = 1

2.2 Performance metrics | confusion matrix & score metrics

- True positive rate (Power, Sensitivity, Hit rate, Recall): $TPR = \frac{TP}{TP+FN} = \frac{36}{36+26}$
- True negative rate (Specificity, 1-FPR): $TNR = \frac{TN}{TN+FP} = \frac{118}{118+12}$
- Precision (Positive Predictive Value; 1- false discovery rate):

$$Precision = \frac{TP}{TP + FP} = 1 - FDR = \frac{36}{36 + 12}$$

- Accuracy: (TP + TN) / all samples = (36+118) / (36+118+12+26)



		Predicted condition	
	Total population = P + N	Positive (PP)	Negative (PN)
Actual condition	Positive (P)	**True positive** (TP), hit	**False negative** (FN), type II error, miss, underestimation
	Negative (N)	**False positive** (FP), type I error, false alarm, overestimation	**True negative** (TN), correct rejection

2.2 Performance metrics | Choice of metrics

- Is sensitivity (true positive rate) or specificity (true negative rate) alone good enough to evaluate models (or thresholds)?
  - What if a model predict everything as positive --> perfect sensitivity
  - What if a model predict everything as negative --> perfect specificity

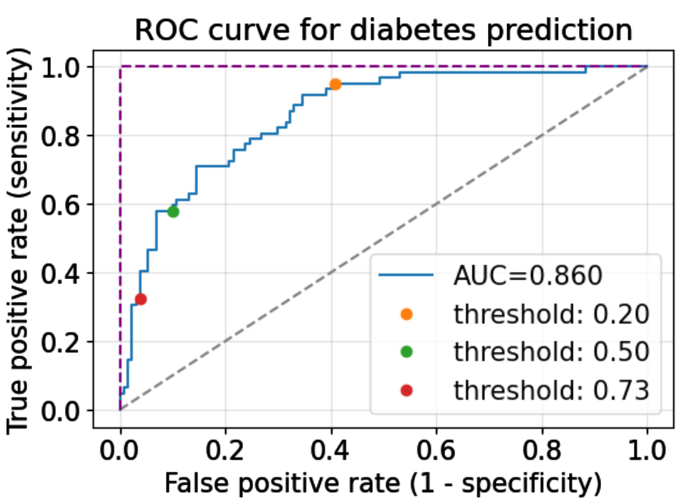
- Is accuracy alone good enough to evaluate models (or thresholds)?
  - Generally, yes. It balances both sensitive and specificity
  - However, when samples are imbalanced, accuracy can be biased.
  - If we predict all sample as non-diabetes, accuracy=130 / (130 + 62) = 67.7%

- When is precision desired? To control false discoveries.

- How to set a reasonable threshold to balance sensitivity and specificity?

2.2 Performance metrics | ROC Curve & AUC

- How to set a reasonable threshold to balance sensitivity and specificity?
  - Calculate sensitivity and specificity at each potential threshold
- We can also plot out the curve between specificity (usually 1 – specificity as x-axis) and sensitivity, when varying thresholds. This curve is called receiver operating characteristic (**ROC**) curve.
- Area Under the Curve (AUC) can be used as a summary metric of ROC curve.



Perfect ROC: towards the top left corner
Random ROC: along the diagonal

The larger the AUC, the better the performance.

Different threshold, different balance in sensitivity and specificity.

Colab Notebook:
https://bit.ly/3EUhEgo

Part 3: Naïve Bayes (quick introduction)

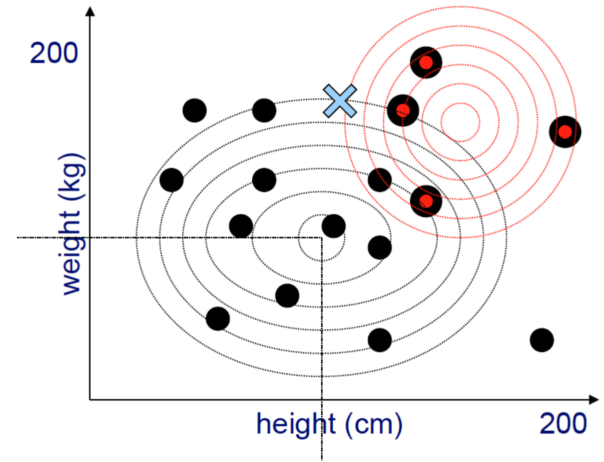
3 Bayesian model | Bayes' theorem

- In logistic regression, we defined the predicted probability as
$$p(y=1|\mathbf{x},\mathbf{w}) = \sigma(\mathbf{x}^\top \mathbf{w})$$

- We can also calculate with Bayes' theorem
$$p(y=1|\mathbf{x}) = \frac{p(\mathbf{x}|y=1)\,p(y=1)}{p(\mathbf{x})} = \frac{p(\mathbf{x}|y=1)\,p(y=1)}{\sum_{t\in\{0,1\}} p(\mathbf{x}|y=t)\,p(y=t)}$$

- Key part is to estimate the distribution of the data in each class:
$$p(\mathbf{x}|y=0), \qquad p(\mathbf{x}|y=1)$$

- Prior: $p(y=1)$ and $p(y=0)$



Bayes' theorem - probability **chain rule**

$$p(\mathbf{x},y=1) = p(y=1|\mathbf{x})\,p(\mathbf{x})$$
$$= p(\mathbf{x}|y=1)\,p(y=1)$$

Figure credit:
Dr Victor Lavrenko

3 Bayesian model | Naïve Bayes

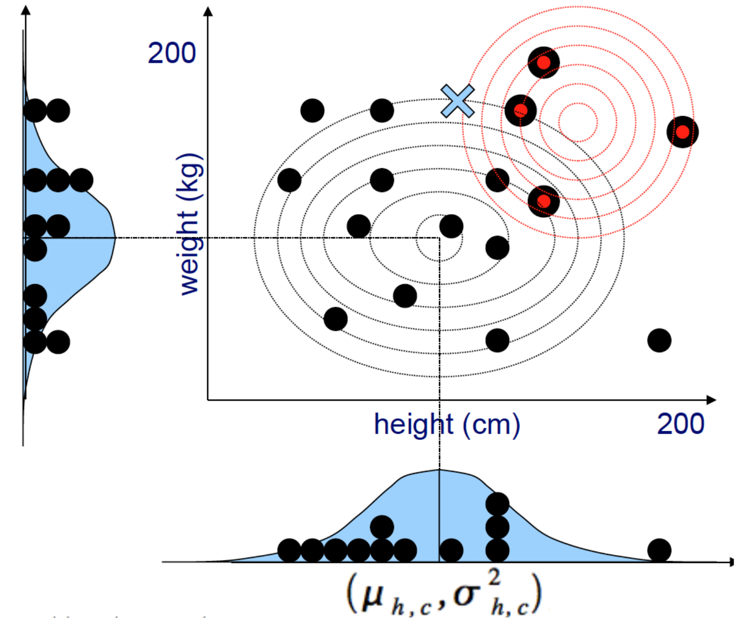
- Bayes' theorem

$$p(y=1|\mathbf{x}) = \frac{p(\mathbf{x}|y=1)\,p(y=1)}{\sum_{t\in\{0,1\}} p(\mathbf{x}|y=t)\,p(y=t)}$$

- Prior $p(y=1)$ is generally set manually. If no information, we set $p(y=0)=p(y=1)=0.5$

- The difficult part is estimating the high dimensional distributions.

- A **naïve assumption**: dimensions are independent in each class, so we can approximate the distribution via each dimension separately.



$$p(\mathbf{x}|y=t) = \prod_{j=1}^{p} p(x_j|y=t)$$

Figure credit:
Dr Victor Lavrenko

3 Bayesian model | Naïve Bayes – density estimation

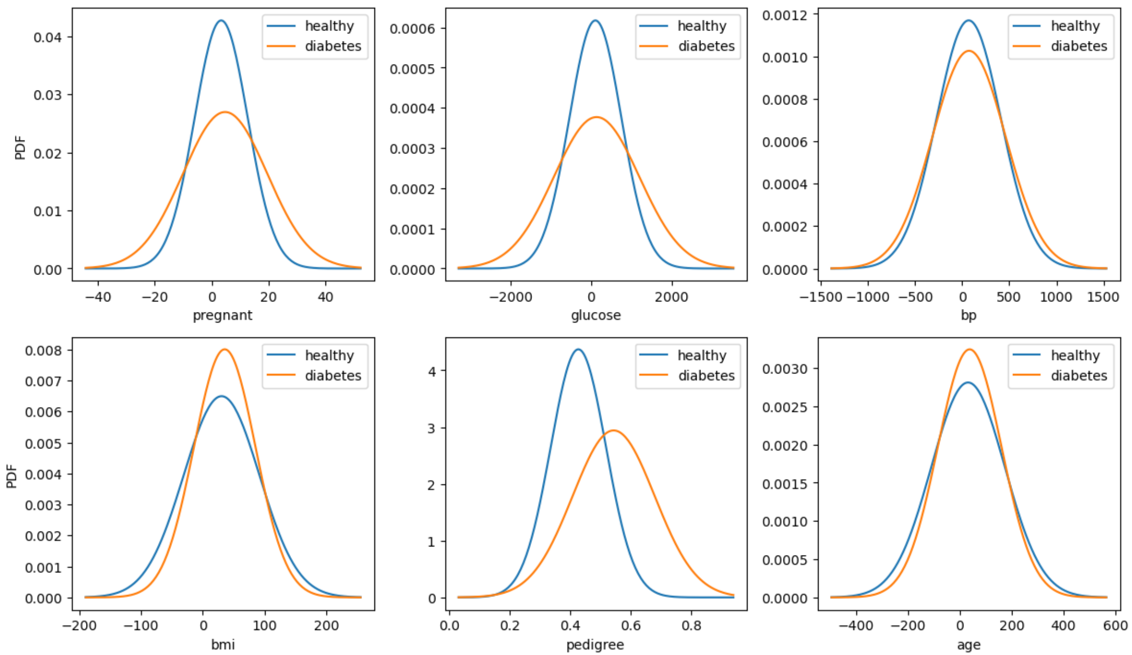
- With the conditional independence assumption, the modelling fitting is only about density estimation of single dimensional data
- Although the independence assumption is strong, Naïve Bayes actually works well in general.

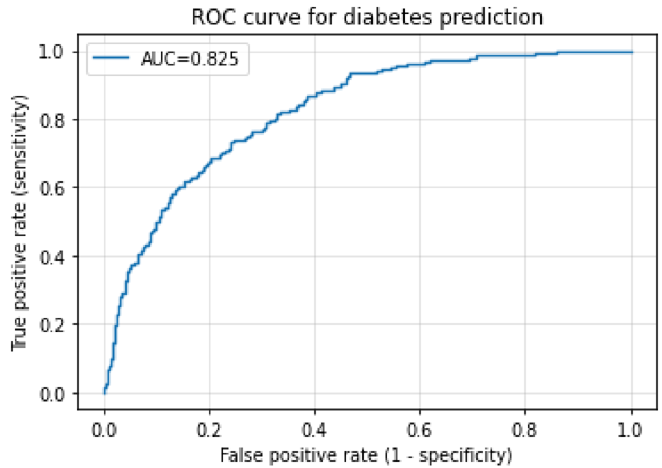
Common density estimation

- Recall the methods for density estimation in earlier lectures from Dr Lau.
- Gaussian distribution (normal distribution)
  - $X \sim N(\mu, \sigma^2)$. $\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_1$; $\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \hat{\mu})^2$
- Kernel-based methods (like the histogram curve)

Classification rule: $p(y = 1|\mathbf{x}) > 0.5 \quad \rightarrow \quad (y = 1)$

3 Bayesian model | Diabetes example





Colab Notebook:
https://bit.ly/3EUhEgo

We used normal distribution to approximate the density:
not always good, e.g., non-negative values.
Consider other distributions, e.g., log-normal, or transformation of the data, e.g., log transform.

Summary

1. Logistic regression
   i. Sigmoid / logit function
   ii. Gradient based optimization
   iii. Feature selection
2. Evaluation of classification performance
   i. Generalization, test set, and cross-validation
   ii. Confusion matrix, scoring metrices, and ROC curve
3. More classification methods
   i. Quick introduction to Naïve Bayes

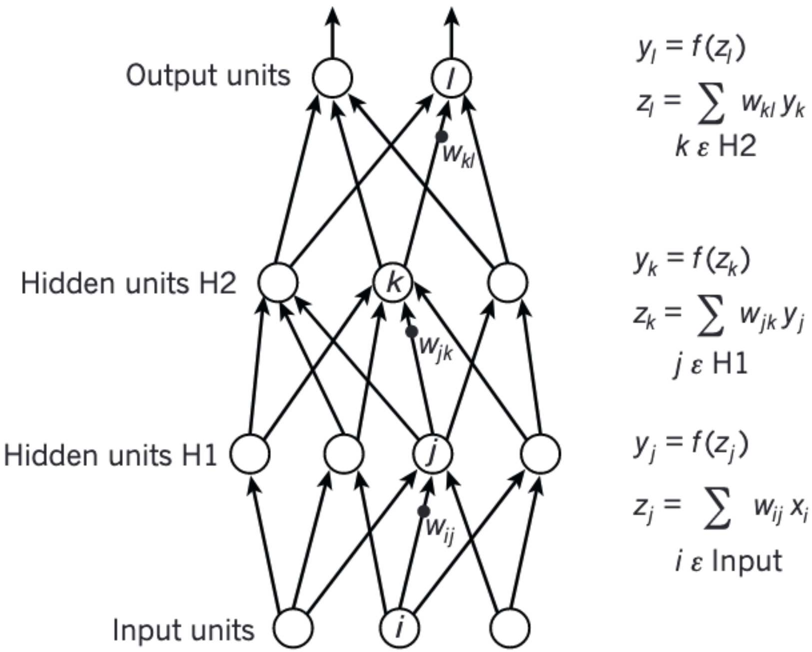
Resources & Acknowledgement

- IPython Notebook for this lecture note:
    - On Moodle
    - Also: https://github.com/huangyh09/foundation-data-science/

Other reference resources with acknowledgement:

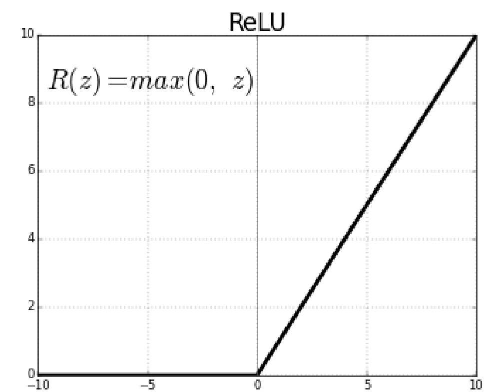
- Chapter 5, Bruces & Gedeck, Practical Statistics for Data Science

Inspiring future study | Neural network in one slide



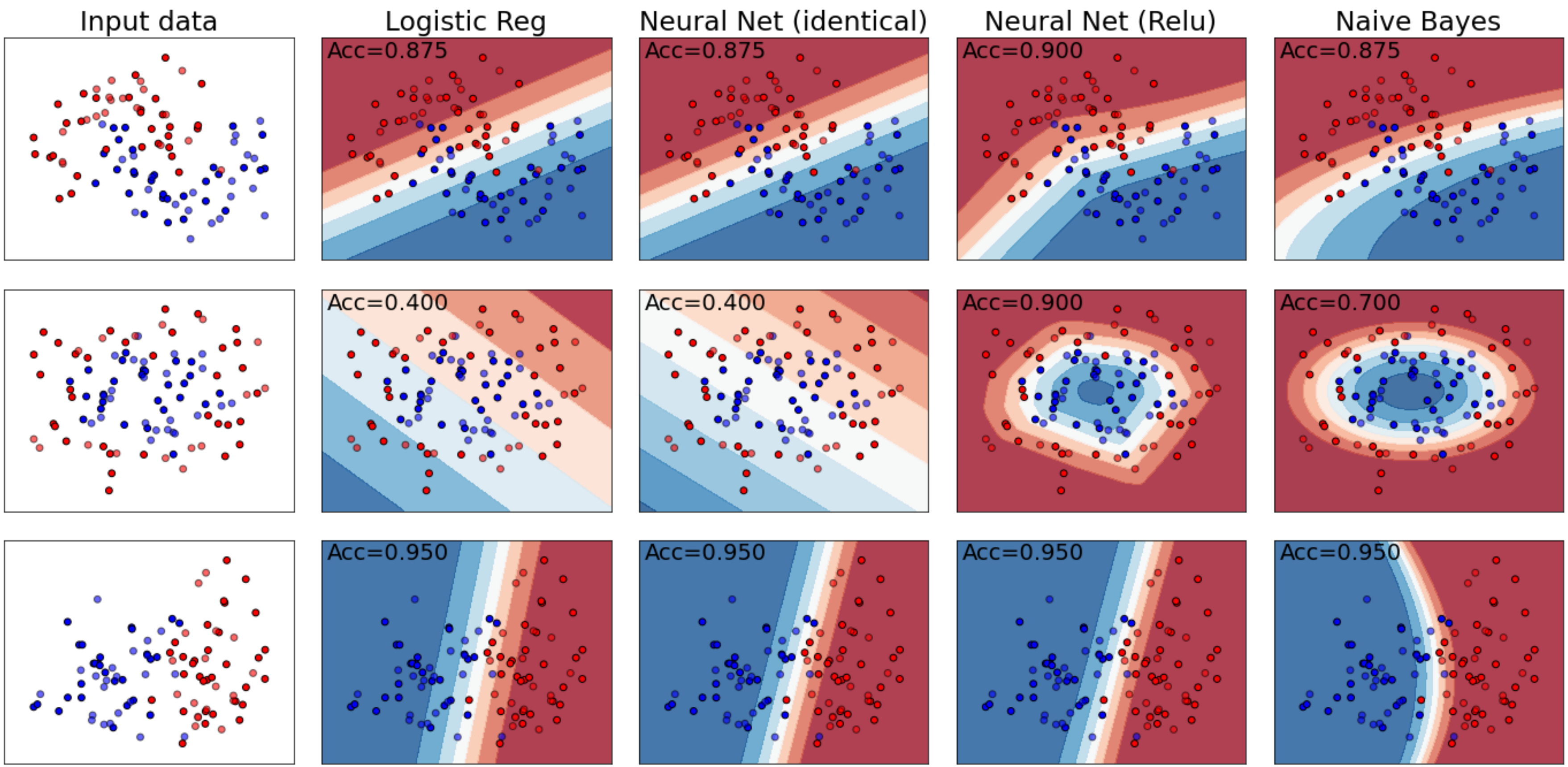
Activation function is critical. Examples:

- Rectified linear unit (ReLU) f(z) = max(0, z)
- Sigmoid function
- Hyperbolic tangent function
- Identical (not commonly used)



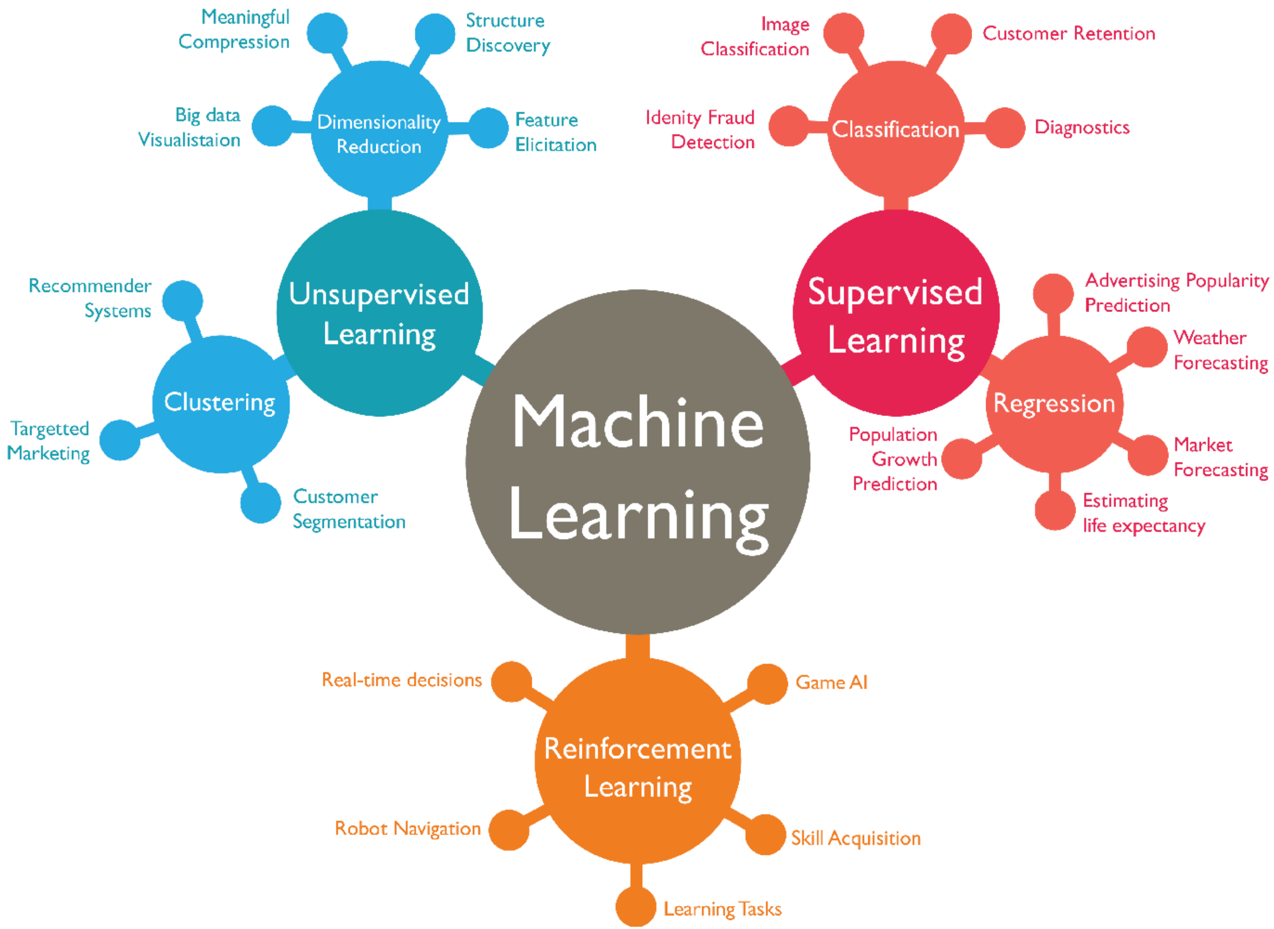
https://scikit-learn.org/stable/modules/neural_networks_supervised.html
LeCun, Bengio and Hinton, Deep learning, Nature (2015). https://doi.org/10.1038/nature14539
Online toy example: http://playground.tensorflow.org

Inspiring future study | More examples & methods



https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html
https://github.com/huangyh09/foundation-data-science/blob/main/w10-classification/P4_classifier_comparison.ipynb

Inspiring future study | Beginning of an exciting path



Inspiring future study | More readings

- Blei & Smyth, Science and Data Science, PNAS (2017).
- Domingos, A Few Useful Things to Know about Machine Learning

Practical books

- Bruces & Gedeck, Practical Statistics for Data Science, 2020 (2nd Edition).
- Raschka and Mirjalili, Python Machine Learning (E-book available at HKU lib)

Advanced books (some parts are intuitive, but some are more mathematical)

- Bishop, Pattern Recognition and Machine Learning (2006). Free online PDF thanks to the author and Microsoft research.
- Murphy, Probabilistic Machine Learning: An Introduction (2021). Free online PDF thanks to the author.

Thank you for your attention!

Please help provide your feedbacks to this course!