实证分析中的稳健性检验：基于机器学习方法

课堂展示

R 语言编程基础与金融数据分析第八组小组展示

吉林大学经济学院

2024 年 11 月 4 日

1. 研究背景
2. 基准回归说明
3. 线性回归参数估计的误差修正
4. 使用非参数方法对模型参数估计进行优化
5. 使用非监督学习对模型参数估计进行优化

1 研究背景

2 基准回归说明

3 线性回归参数估计的误差修正

4 使用非参数方法对模型参数估计进行优化

5 使用非监督学习对模型参数估计进行优化

线性回归是最优预测吗？

直观上，数据可视为信号与噪声的组合。当样本内的拟合得越来越完美时，这意味着回归函数也拟合了大量的噪声（因为回归函数“跟”数据跟得太紧）；而噪声对于样本外的预测毫无意义。因此，在过拟合的情况下，虽然在样本内的拟合优度很高，但模型在样本外的预测能力反而下降。然而，机器学习目的恰恰是样本外的预测能力，即将模型运用于其未见过的数据(unseendata)时，所具有的推广预测能力，即模型的泛化能力(unseendata)。另一方面，对于模型已经见过并据此进行优化的训练数据，即使拟合得再好，也说明不了问题：它可能只是记住了训练数据，比如此例中 M=9 的多项式回归情形。

线性回归是最优预测吗？

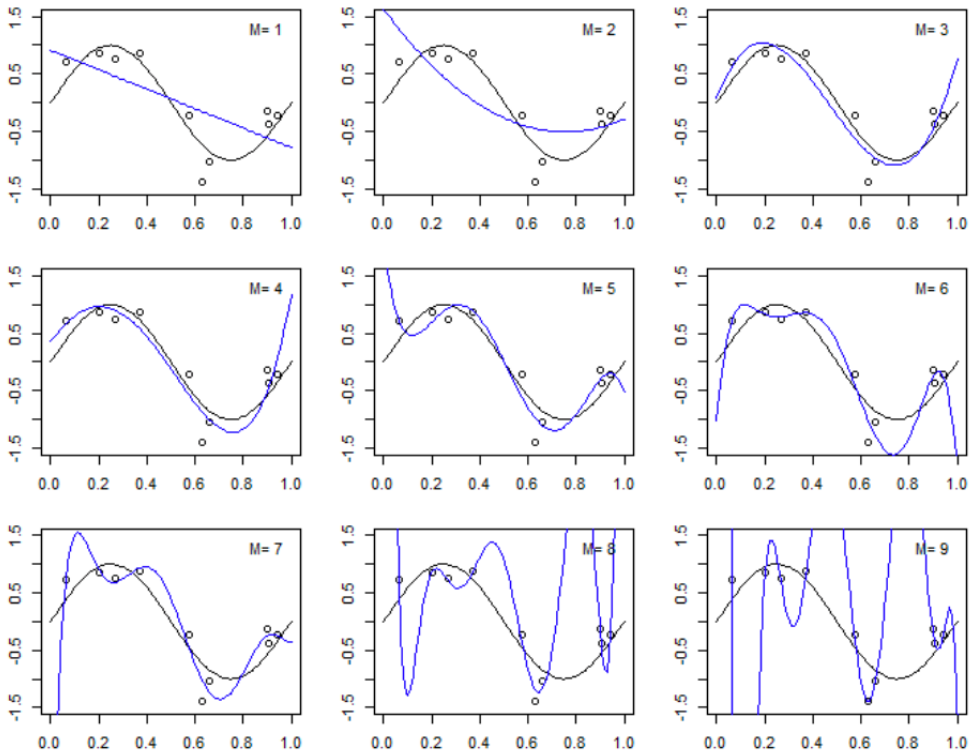

图 1: 线性回归是最优预测吗？

偏差与方差的权衡

通常来说，偏差平方与方差之间存在着此消彼长替代关系。当模型过于简单（比如多项式拟合案例中 M 为 1 或 2），则估计量的偏差较大，但方差较小；此时存在欠拟合。随着模型的复杂程度增加，估计量越来越接近于数据，故偏差变小；但由于数据包含噪声，紧跟数据使得方差随之增大，此时易出现过拟合。因此，在选择模型复杂程度时，存在偏差与方差的权衡（bias-variance trade-off），故须选择合适的模型复杂程度，使得模型的均方误差来达到最小值（比如上例中的 M=3）。

偏差与方差的权衡

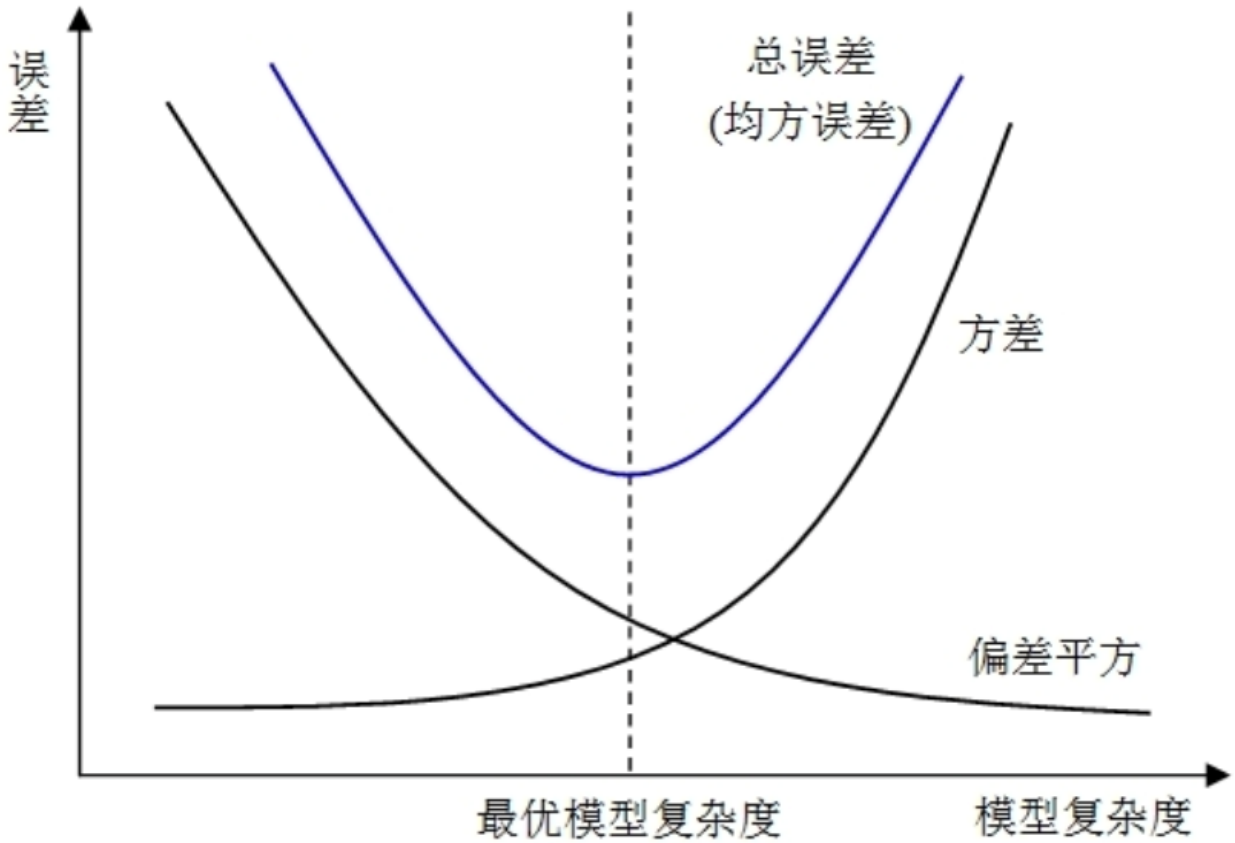

图 2: 偏差与方差的权衡

1 研究背景

2 基准回归说明

3 线性回归参数估计的误差修正

4 使用非参数方法对模型参数估计进行优化

5 使用非监督学习对模型参数估计进行优化

基准模型设定

设定以下模型研究企业数字化转型对于企业创新能力的影响：

$$\text{Innovation}_{i,t+1} = \alpha_0 + \alpha_1 \text{ Digit}_{i,t} + \sum \alpha_i \text{ Controls}_{i,t} + \mu_t + \lambda_i + \varepsilon_{i,t}$$

其中，$\text{Innovation}_{i,t+1}$ 代表企业创新能力，$\text{Digit}_{i,t}$ 代表企业数字化转型程度，$\text{Controls}_{i,t}$ 包含一系列控制变量，μ_t 表示年份固定效应，λ_i 表示企业固定效应。本文关注的主要系数为 α_1 。若企业数字化转型对企业创新具有显著的促进作用，则 α_1 应显著为正。

主要变量定义

主要变量定义如图所示

Variable	变量定义
Innovation	企业创新，用未来一期专利申请总数+1 的自然对数表示
Lndgt	数字化转型程度，用年报中数字化转型相关词汇的词频+1 的自然对数表示
Size	企业规模，用营业收入的自然对数表示
LEV	资产负债率，等于总负债/总资产
PPE	固定资产占比，等于固定资产/总资产
ROA	总资产报酬率，等于净利润/总资产
Cash	经营活动现金净额/总资产
SOE	产权性质，若为国有企业，则取值为 1，否则为 0
Boardsize	董事会规模，用董事会人数的自然对数表示
Indboard	独立董事占比
BM	账面市值比，等于股东权益总额/公司市值
Dual	两职合一，若董事长与总经理为同一人，则取值为 1，否则为 0
Age	企业年龄，等于企业成立年数

图 3: 主要变量定义

数据集结构

数据清洗结果如图所示，R 代码与原始数据详见主页

```
> data
# A tibble: 30,448 × 19
   股票代码 股票简称.x  年份 行业名称 行业代码 省份  Innovation Digit  Size   Lev      PPE
      <dbl> <chr>      <dbl> <chr>    <chr>    <chr>       <dbl> <dbl> <dbl> <dbl>    <dbl>
 1        2 万科A       2012 房地产业 K70      广东…        1.79  1.10  25.4 0.783 0.00426
 2        2 万科A       2013 房地产业 K70      广东…        1.61  1.61  25.6 0.780 0.00444
 3        2 万科A       2014 房地产业 K70      广东…        3.26  3.18  25.7 0.772 0.00454
 4        2 万科A       2015 房地产业 K70      广东…        3.26  1.61  26.0 0.777 0.00804
 5        2 万科A       2016 房地产业 K70      广东…        2.94  1.10  26.2 0.805 0.00820
 6        2 万科A       2017 房地产业 K70      广东…        3.47  1.39  26.2 0.840 0.00609
 7        2 万科A       2018 房地产业 K70      广东…        3.69  2.08  26.4 0.846 0.00755
 8        2 万科A       2019 房地产业 K70      广东…        3.74  1.61  26.6 0.844 0.00717
 9        2 万科A       2020 房地产业 K70      广东…        3.89  2.20  26.8 0.813 0.00673
10        2 万科A       2021 房地产业 K70      广东…        3.66  2.40  26.8 0.797 0.00661
# ℹ 30,438 more rows
# ℹ 8 more variables: ROA <dbl>, Cash <dbl>, SOE <dbl>, Boardsize <dbl>, Indboard <dbl>,
#   BM <dbl>, Dual <dbl>, Age <dbl>
```

图 4: 数据集结构

基准回归结果

基准回归结果如图所示，Do 文档与 dta 文件详见主页

Fixed-effects (within) regression — Number of obs = 30,448
Group variable: id — Number of groups = 4,397

R-squared:
Within = 0.1995
Between = 0.0354
Overall = 0.0476

Obs per group:
min = 1
avg = 6.9
max = 11

F(12, 4396) = 166.98
corr(u_i, Xb) = -0.3378 — Prob > F = 0.0000

(Std. err. adjusted for 4,397 clusters in id)

innovation	Coefficient	Robust std. err.	t	P>|t|	[95% conf.	interval]
digit	.0649232	.0106135	6.12	0.000	.0441154	.0857309
size	.3086007	.0213392	14.46	0.000	.2667651	.3504363
lev	-.0005683	.0041583	-0.14	0.891	-.0087207	.0075841
ppe	-.2427198	.1277654	-1.90	0.058	-.4932045	.0077648
roa	.0171784	.0180771	0.95	0.342	-.0182619	.0526187
cash	-.1950347	.082702	-2.36	0.018	-.3571723	-.032897
soe	.0103067	.0297877	0.35	0.729	-.0480922	.0687056
boardsize	.1062044	.0897143	1.18	0.237	-.0696808	.2820896
indboard	-.0003287	.0025724	-0.13	0.898	-.0053719	.0047144
bm	.1619138	.0421268	3.84	0.000	.079324	.2445036
dual	.0471588	.023517	2.01	0.045	.0010536	.0932639
age	.0801267	.0038337	20.90	0.000	.0726107	.0876427
_cons	-5.086779	.5185527	-9.81	0.000	-6.103404	-4.070155
sigma_u	1.5798192					
sigma_e	.81313919					
rho	.79056374	(fraction of variance due to u_i)				

图 5: 基准回归结果

1 研究背景

2 基准回归说明

3 线性回归参数估计的误差修正

4 使用非参数方法对模型参数估计进行优化

5 使用非监督学习对模型参数估计进行优化

岭回归

对于任意估计量 $\hat{\boldsymbol{\beta}}$, 其均方误差可分解为方差与偏差平方之和:

$$\mathrm{MSE}(\hat{\boldsymbol{\beta}}) \equiv E\left[(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})'\right] = \mathrm{Var}(\hat{\boldsymbol{\beta}}) + [\mathrm{Bias}(\hat{\boldsymbol{\beta}})][\mathrm{Bias}(\hat{\boldsymbol{\beta}})]'$$

因此, 使均方误差最小化, 可视为在方差与偏差之间进行权衡 (trade-off)。比如，一个无偏估计量（偏差为 0），如果方差很大，则可能不如一个虽然有偏差但是方差却很小的估计量。在（严格）多重共线性的情况下，虽然 OLS 估计量无偏，但其方差太大（无穷大），而岭回归虽有少量偏差，但可大幅减少方差, 这使得岭回归估计量的均方误差 (MSE) 可能比 OLS 更小, 参见下图。

岭回归

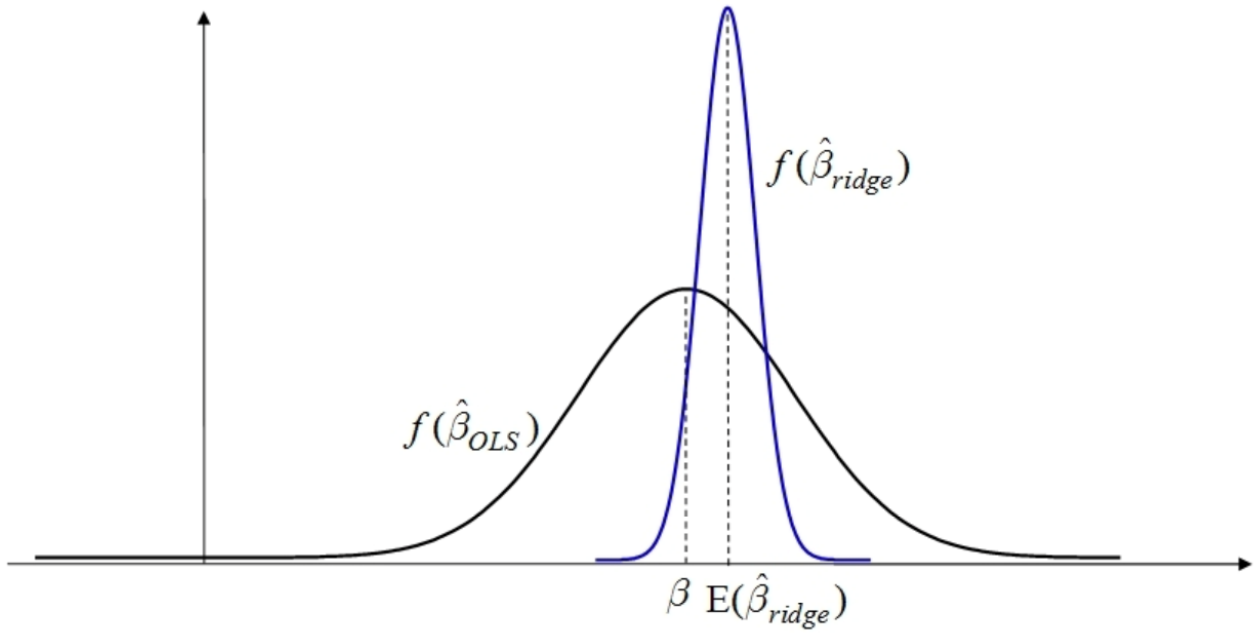

图 6: 使用岭回归进行误差修正

套索回归

Tibshirani （1996）提出"套索估计量"（Least Absolute Shrinkage and Selection Operator, 简记 LASSO), 将岭回归惩罚项中 2-范数改为 1 -范数:

$$\min_{\beta} L(\beta) = \underbrace{(\boldsymbol{y} - \boldsymbol{X}\beta)'(\boldsymbol{y} - \boldsymbol{X}\beta)}_{\text{SSR}} + \underbrace{\lambda\|\beta\|_1}_{\text{panelty}}$$

其中, $\|\beta\|_1 = \sum_{j=1}^{p} |\beta_j|$ 为参数向量 β 的 1-范数（L_1 norm）

类似于岭回归, Lasso 最小化问题也可等价写为如下约束极值问题:

$$\min_{\beta}(\boldsymbol{y} - \boldsymbol{X}\beta)'(\boldsymbol{y} - \boldsymbol{X}\beta)$$

$$\text{s.t. } \|\beta\|_1 \leqslant t$$

其中，$t \geqslant 0$ 为某常数。

弹性网回归

Zou and Hastie（2005）将 Lasso 与岭回归相结合，提出弹性网（elastic net）估计量。在弹性网估计量损失函数中，同时包含 L_1 与 L_2 惩罚项：

$$\min_{\boldsymbol{\beta}}(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})'(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})+\lambda_1\|\boldsymbol{\beta}\|_1+\lambda_2\|\boldsymbol{\beta}\|_2^2$$

其中，$\lambda_1 \geqslant 0$ 与 $\lambda_2 \geqslant 0$ 都是调节参数。由于 λ_1 与 λ_2 的取值范围均为无穷，不便于使用交叉验证选择其最优值。为此，定义 $\lambda \equiv \lambda_1+\lambda_2, \alpha \equiv \frac{\lambda_1}{\lambda}$，可将损失函数等价地写为

$$\min_{\boldsymbol{\beta}}(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})'(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})+\lambda\left[\alpha\|\boldsymbol{\beta}\|_1+(1-\alpha)\|\boldsymbol{\beta}\|_2^2\right]$$

其中，$\lambda \geqslant 0$ 与 $0 \leqslant \alpha \leqslant 1$ 为调节参数。由于调节参数 α 的取值局限于区间 $[0,1]$，故便于通过交叉验证选择其最优值。

弹性网回归

显然，如果 $\alpha = 0$，则弹性网退化为岭回归；而如果 $\alpha = 1$，则弹性网退化为 Lasso。因此，岭回归与 Lasso 都是弹性网的特例。如果 $0 < \alpha < 1$，则弹性网为岭回归与 Lasso 之间的折中。

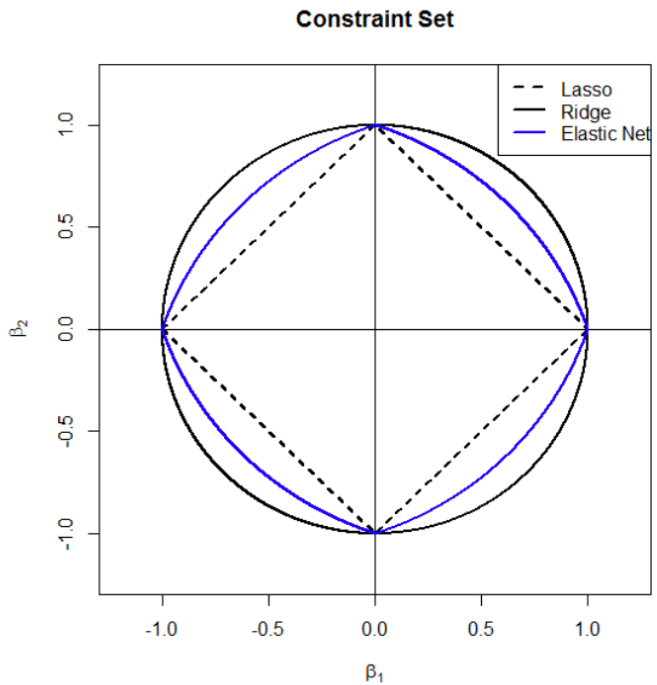

图 7: 弹性网回归的几何解释

Lasso 回归与弹性网回归结果

Lasso 回归与弹性网回归结果如图所示，R 代码与原始数据详见主页

```
15 x 1 sparse Matrix of class "dgCMatrix"
                    s1
(Intercept)  2.767961014
Digit        0.123564265
Size         0.757993225
Lev          .
PPE         -0.000522285
ROA          .
Cash        -0.046287744
SOE          .
Boardsize    .
Indboard     .
BM          -0.079218247
Dual         .
Age         -0.246227581
Id           .
Year         0.190665034
```

图 8: Lasso 回归结果

```
15 x 1 sparse Matrix of class "dgCMatrix"
                    s1
(Intercept)  2.767961014
Digit        0.125660924
Size         0.742877070
Lev          .
PPE         -0.003882644
ROA          .
Cash        -0.051310192
SOE          .
Boardsize    .
Indboard     .
BM          -0.075428644
Dual         .
Age         -0.240701008
Id           .
Year         0.189944438
```

图 9: 弹性网回归结果

1 研究背景

2 基准回归说明

3 线性回归参数估计的误差修正

4 使用非参数方法对模型参数估计进行优化

5 使用非监督学习对模型参数估计进行优化

分类树

Breiman et al.（1984）研究了 UCSD 医学中心的一个案例。当心梗病人进入 UCSD 医学中心后 24 小时内，测量 19 个变量，包括血压、年龄以及 17 个排序或虚拟变量。数据集中包含 215 个病人，其中 37 人为高危病人。研究目的是为了快速预测哪些心梗病人为“高危病人”（无法活过 30 天），哪些是“低危病人”（可活过 30 天），从而更好地配置医疗资源。

分类树

Breiman et al.（1984）建立了如下分类树

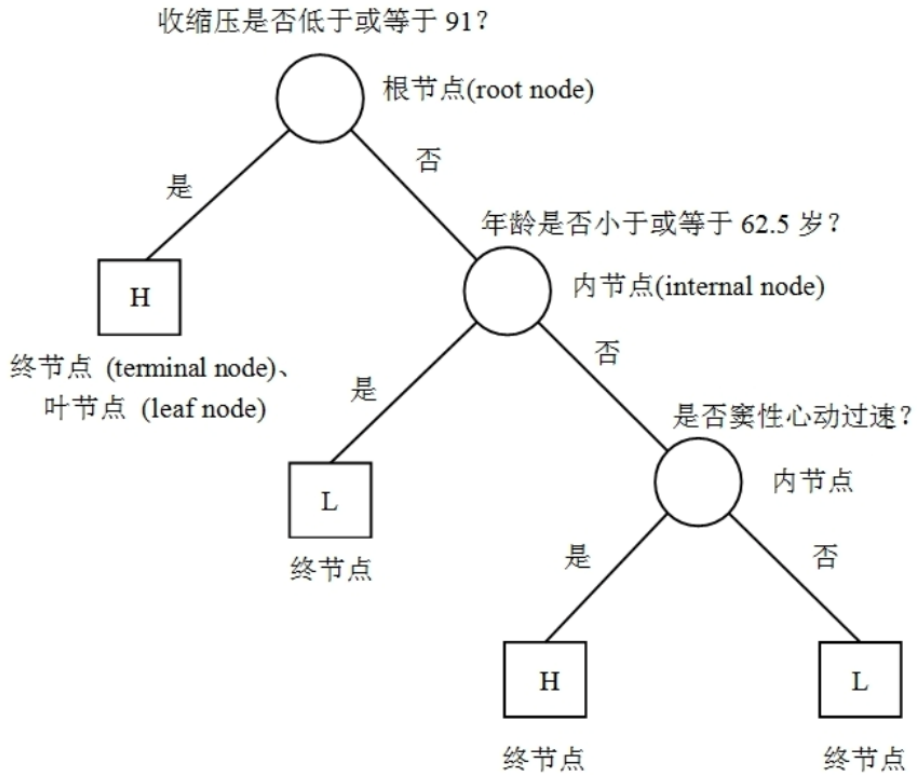

图 10: 两分类决策树

分类树

对于三维以上的特征空间，我们依然可用树状结构来表示，因为决策树每次仅使用一个变量进行分裂（splitting）决策树模型将特征空间分割为若干（超）矩形的终节点。在进行预测时，每个终节点只有一个共同的预测值。对于分类问题，此预测值为该终节点所有训练样本的最常见类别（most commonly occurring class）。对于回归问题，此预测值为该终节点所有训练样本的平均值。因此，在数学上，决策树为“分段常值函数”（piecewise constant function）

分类树

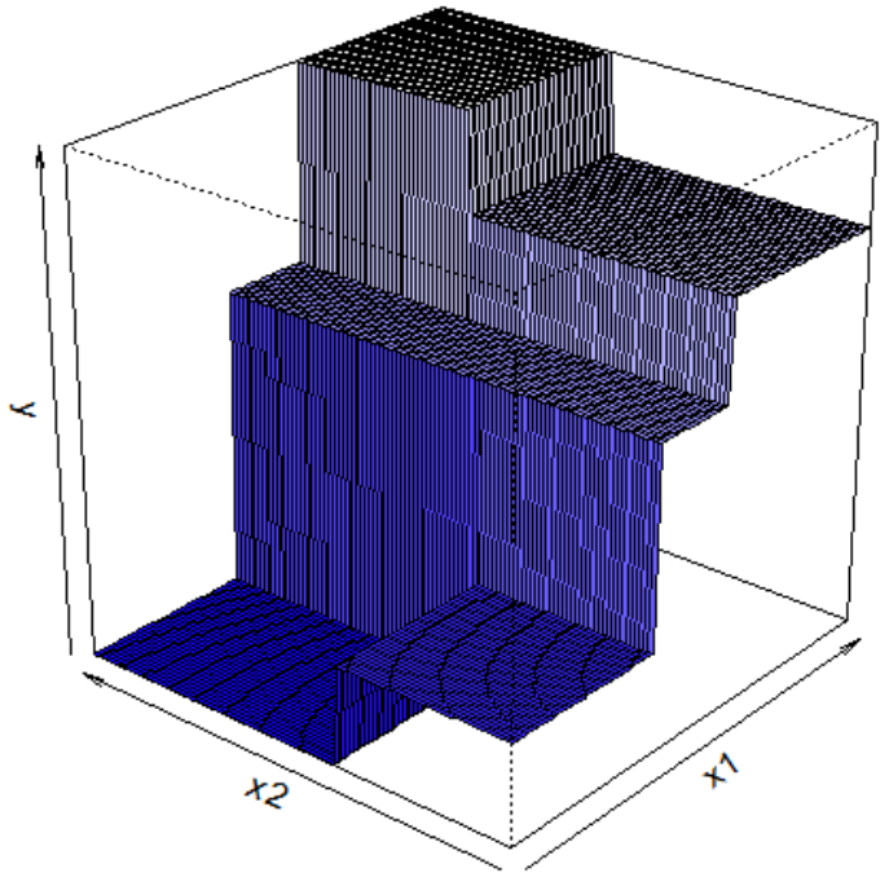

图 11: 多分类决策树

分类树

在估计决策树模型时，面临一个选择，即何时停止节点分裂。一个较好的解决方法是，先让决策树尽情生长，记最大的树为 $T_{\max}$，再进行" 修枝" （pruning），以得到一个" 子树" （subtree） T 。对于任意子树 $T \subseteq T_{\max}$ ，定义其" 复杂性" （complexity）为子树 T 的终节点数目（number of terminal nodes），记为 $|T|$ 。为避免过拟合，不希望决策树过于复杂，故惩罚其规模 $|T|$ ：

$$\min_T \underbrace{R(T)}_{\text{cost}} + \lambda \cdot \underbrace{|T|}_{\text{complxity}}$$

其中，$R(T)$ 为原来的损失函数，比如 $0-1$ 损失函数 （$0-1$ loss），即在训练样本中，如果预测正确，则损失为 0 ，而若预测错误则损失为 1 $\lambda \geqslant 0$ 称为调节参数（tuning parameter），也称其为" 复杂性参数" （complexity parameter，简记 CP ）。λ 控制对决策树规模 $|T|$ 的惩罚力度，可通过交叉验证确定。这种修枝方法称为成本—复杂性修枝 (cost-complexity pruning)。

分类树

对于回归问题，其响应变量 y 可为连续变量。因此，对于回归树，可使用"最小化残差平方和"作为节点的分裂准则。这意味着，在进行节点分裂时，希望分裂后残差平方和下降最多，即两个子节点的残差平方和的总和最小。为避免过拟合，对于回归树，也要使用惩罚项来进行修枝，即最小化如下目标函数:

$$\min_{T} \underbrace{\sum_{m=1}^{|T|} \sum_{x_i \in R_m} \left(y_i - \hat{y}_{R_m}\right)^2}_{\text{cost}} + \lambda \cdot \underbrace{|T|}_{\text{complexity}}$$

其中，R_m 为第 m 个终节点，而 $\hat{y}_{R_m}$ 为该终节点的预测值(此终节点的样本均值)。$\sum_{x_i \in R_m} \left(y_i - \hat{y}_{R_m}\right)^2$ 为第 m 个终节点的残差平方和，然后对所有终节点 $m = 1, 2, \cdots, T$ 进行加总，即为成本 $R(T)$ 。

随机森林

与单棵决策树相比，随机森林包含很多决策树（比如 500 棵决策树的平均），故无法像单棵决策树那样进行解释。对于随机森林，应如何度量“变量重要性”（Variable Importance）？由于在每次节点分裂，仅使用一个变量，因此容易区分每个变量的贡献，考察该分裂变量使得残差平方和（或者基尼指数）下降多少。具体来说，对于每个变量，在随机森林的每棵决策树，可度量由该变量所导致的分裂准则函数的下降幅度。然后，针对此下降幅度，对每棵决策树进行平均，即为对该变量重要性的度量。将每个特征变量的重要性依次排列画图，即为变量重要性图（Variable Importance Plot）。

AdaBoost

最早的提升法为 Freund and Schapire（1996，1997）提出的自适应提升法（Adaptive Boosting，简记 AdaBoost），仅适用于分类问题。对于分类问题，考虑依次种 M 棵。对于第 m 棵树中错误分类的观测值，则在其之后的第 m+1 棵树加大其权重，以此类推。

AdaBoost

自从 Freund and Schapire（1996）提出 AdaBoost 算法后，即取得了很好的预测效果。但学界一直不明白为何 AdaBoost 的预测效果如此好。直至 Friedman et al.（2000），才给出 AdaBoost 的统计解释。关键结论是，AdaBoost 算法可以等价地视为前向分段加法模型（forward stagewise additive modeling），并使用了指数损失函数（exponential loss function）。发现 AdaBoost 算法的数学本质后，即开启了改进与推广 AdaBoost 之门。比如，AdaBoost 算法使用指数损失函数，那么在前向分段加法模型中，使用其他损失函数，即可得到不同的算法。最初的 AdaBoost 算法仅适用于分类问题，这是因为它使用指数损失函数。而对于回归问题，很自然地可以考虑使用“误差平方”（squared loss）的损失函数。

决策树与随机森林回归结果

决策树与随机森林回归结果如图所示，R 代码与原始数据详见主页

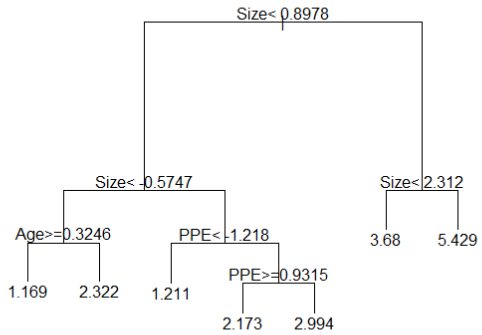

图 12: 决策树回归结果

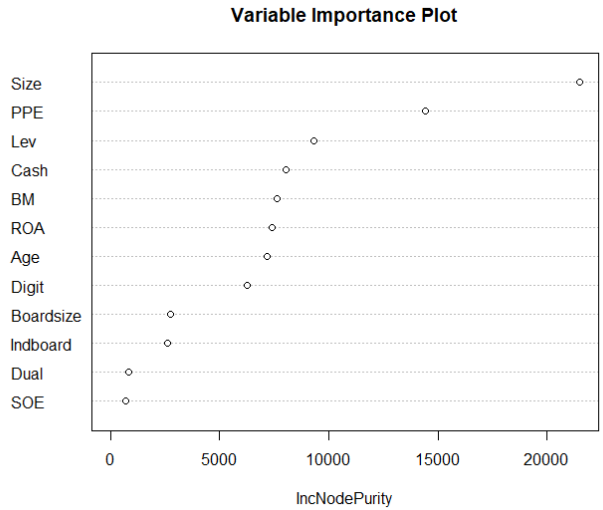

图 13: 随机森林回归结果

AdaBoost 回归结果

AdaBoost 回归结果如图所示，R 代码与原始数据详见主页

```
> summary(model)
                  var   rel.inf
Size             Size 77.273018
Age               Age 11.645551
PPE               PPE  8.803018
Year             Year  2.278412
Digit           Digit  0.000000
Lev               Lev  0.000000
ROA               ROA  0.000000
Cash             Cash  0.000000
SOE               SOE  0.000000
Boardsize Boardsize  0.000000
Indboard   Indboard  0.000000
BM                BM  0.000000
Dual             Dual  0.000000
Id                 Id  0.000000
```

图 14: AdaBoost 回归结果

1 研究背景

2 基准回归说明

3 线性回归参数估计的误差修正

4 使用非参数方法对模型参数估计进行优化

5 使用非监督学习对模型参数估计进行优化

向量机

假设有 p 个特征变量，则分离超平面 L 的方程可写为

$$\beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p = \beta_0 + \boldsymbol{\beta}'\boldsymbol{x} = 0$$

显然，此分离超平面 L 将 p 维特征空间（feature space）一分为二。可见如果 $\beta_0 + \boldsymbol{\beta}'\boldsymbol{x} > 0$，则观测值 $\boldsymbol{x}$ 落于超平面 L 的一边。反之，如果 $\beta_0 + \boldsymbol{\beta}'\boldsymbol{x} < 0$，则观测值 $\boldsymbol{x}$ 落于超平面 L 的另一边。进一步，$\left|\beta_0 + \boldsymbol{\beta}'\boldsymbol{x}\right|$（绝对值）的大小可用于度量观测值 $\boldsymbol{x}$ 到超平面 L 距离远近。如果两类数据之间存在分离超平面，则称数据为线性可分（linearly separable）。然而，在线性可分的情况下，分离超平面一般并不唯一，因为总可以稍微移动超平面，而依然将两类数据分离。

向量机

针对分离超平面不唯一的问题，一种解决方法是使得分离超平面离两类数据尽量远。具体来说，希望在两类数据之间有一条“隔离带”，而且这条隔离带要越宽越好。这就是所谓最大间隔分类器（maximal margin classifier），俗称为最宽街道法（widest street approach），即在两类数据之间建一条最宽的街道。

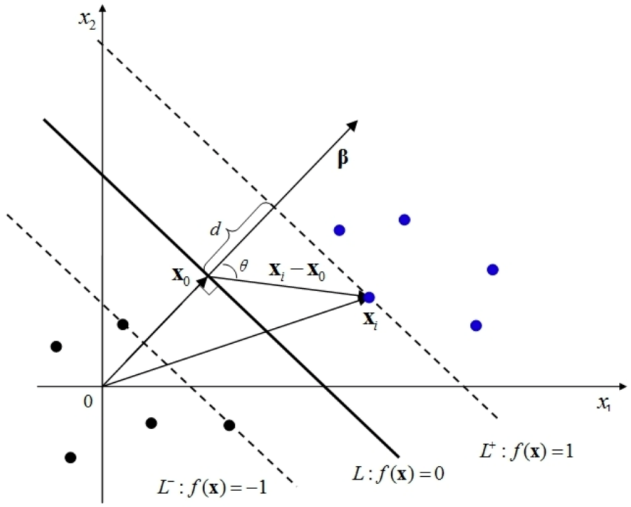

图 15: 分离超平面

向量机

更一般地，对于决策边界非线性数据，考虑对特征向量 $\boldsymbol{x}_i$ 进行变换，比如将 $\boldsymbol{x}_i$ 变换为 $\boldsymbol{\varphi}(\boldsymbol{x}_i)$；其中，$\boldsymbol{\varphi}(\boldsymbol{x}_i)$ 为多维函数（维度可以高于 $\boldsymbol{x}_i$），甚至可以无限维函数。这意味着，可将训练样本 $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$ 变换为 $\{\boldsymbol{\varphi}(\boldsymbol{x}_i), y_i\}_{i=1}^n$，目的是希望在 $\boldsymbol{\varphi}(\boldsymbol{x}_i)$ 的特征空间（feature space）中，可以得到线性可分的情形，参见下图。但难点在于，对于高维数据，一般不知道变换 $\boldsymbol{\varphi}(\cdot)$ 的具体形式。

向量机

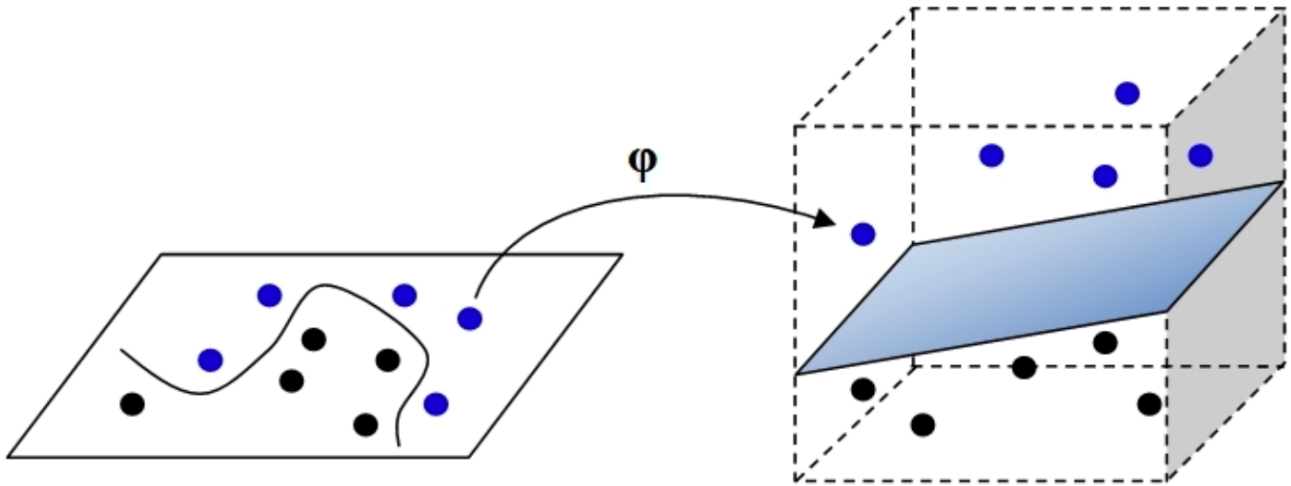

图 16: 核变换

感知机

感知机算法的直观解释为，当一个样本点被错误分类，即出现在分离超平面的错误一侧时，则调整参使得分离超平面向该误分类点一侧移动，以减少此误分类点与超平面的距离，直至正确分类为止。可以证明，对于线性可分的数据，则感知机一定会收敛，参见李航 (2019)。这表明，只要给予足够的数据，感知机具备学得参数 (w,b) 的能力，仿佛拥有“感知”世界的能力（比如，自动将事物分类），故名“感知机”。然而，从不同的初始值出发，一般会得到不同的分离超平面，无法得到唯一解。另一方面，如果数据为线性不可分，则感知机的算法不会收敛。感知机更严重的缺陷是，它的决策边界依然为线性函数。

感知机

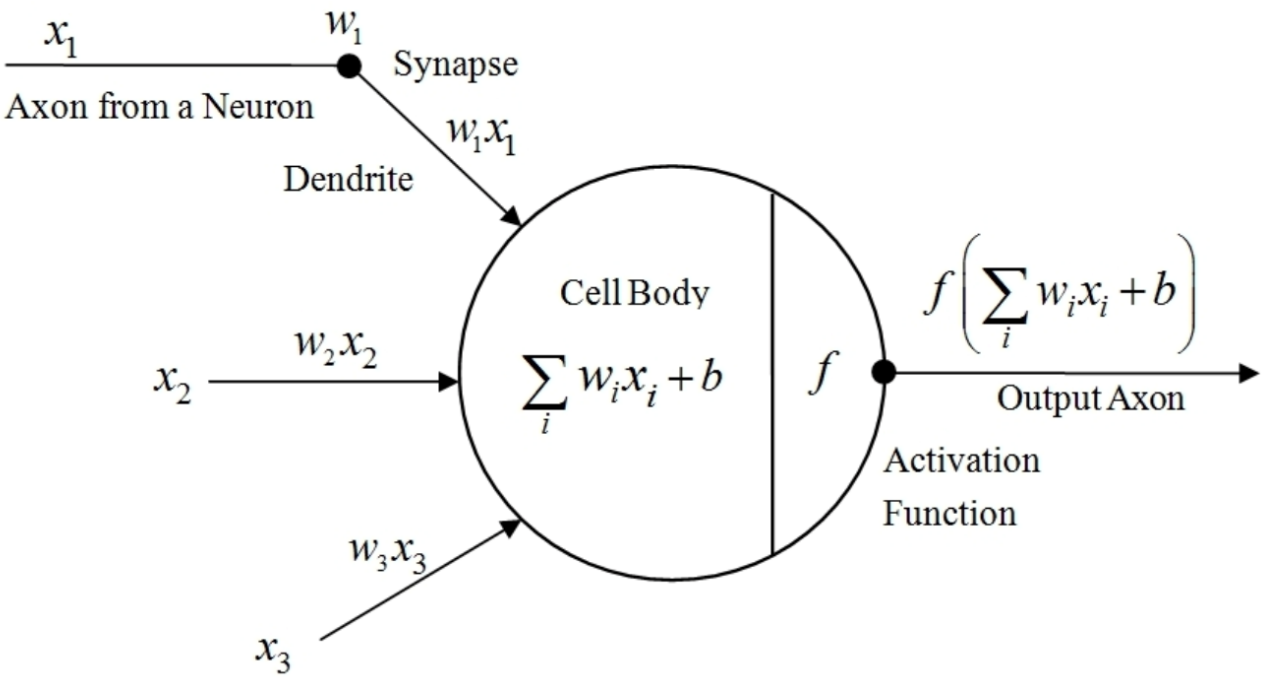

图 17: 感知机核变换

神经网络

只要引入多层神经元，经过两个及以上的非线性激活函数迭代之后，即可得到非线性决策边界。在此，非线性的激活函数是关键；因为如果使用线性的激活函数，则无论叠加或嵌套多少次(相当于微积分的复合函数)，所得结果一定还是线性函数。首先，考虑具有多个输出结果（multi-output）的感知机；其次，图中的多个输出结果可重新作为输入变量，经过加权求和后，再次施以激活函数。

神经网络

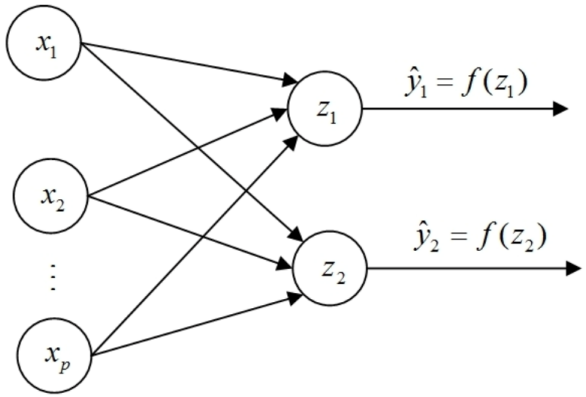

图 18: 多个输出结果的感知机

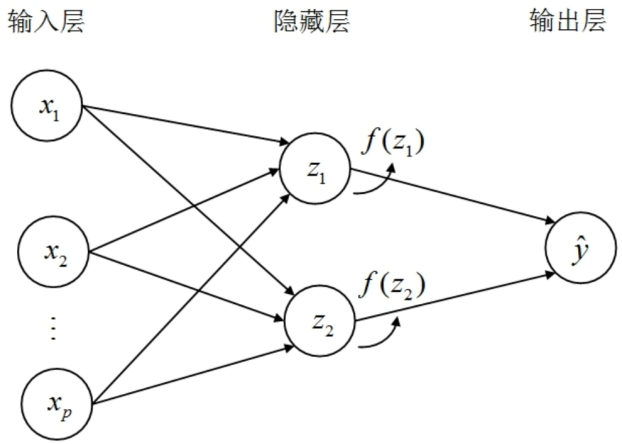

图 19: 多层感知机

神经网络

只要引入多层神经元，经过两个及以上的非线性激活函数迭代之后，即可得到非线性决策边界。在此，非线性的激活函数是关键；因为如果使用线性的激活函数，则无论叠加或嵌套多少次(相当于微积分的复合函数)，所得结果一定还是线性函数。首先，考虑具有多个输出结果（multi-output）的感知机；其次，图中的多个输出结果可重新作为输入变量，经过加权求和后，再次施以激活函数。

神经网络

只要引入多层神经元，经过两个及以上的非线性激活函数迭代之后，即可得到非线性决策边界。在此，非线性的激活函数是关键；因为如果使用线性的激活函数，则无论叠加或嵌套多少次(相当于微积分的复合函数)，所得结果一定还是线性函数。首先，考虑具有多个输出结果（multi-output）的感知机；其次，图中的多个输出结果可重新作为输入变量，经过加权求和后，再次施以激活函数。

神经网络

如图所示，即为基本的神经网络结构。这种标准神经网络，称为前馈神经网络（feed forward neural network），因为输入从左向右不断前馈；也称为全连接神经网络（fully connected neural network），因为相邻层的所有神经元都相互连接。当然，针对特殊数据类型可能还需要一些特别的网络结构，比如，卷积神经网络（适用于图像识别）、循环神经网络（适用于自然语言等时间序列）等。另外，如果神经网络的隐藏层很多，则称为深度神经网络（deep neural networks），简称深度学习（deep learning）。

神经网络

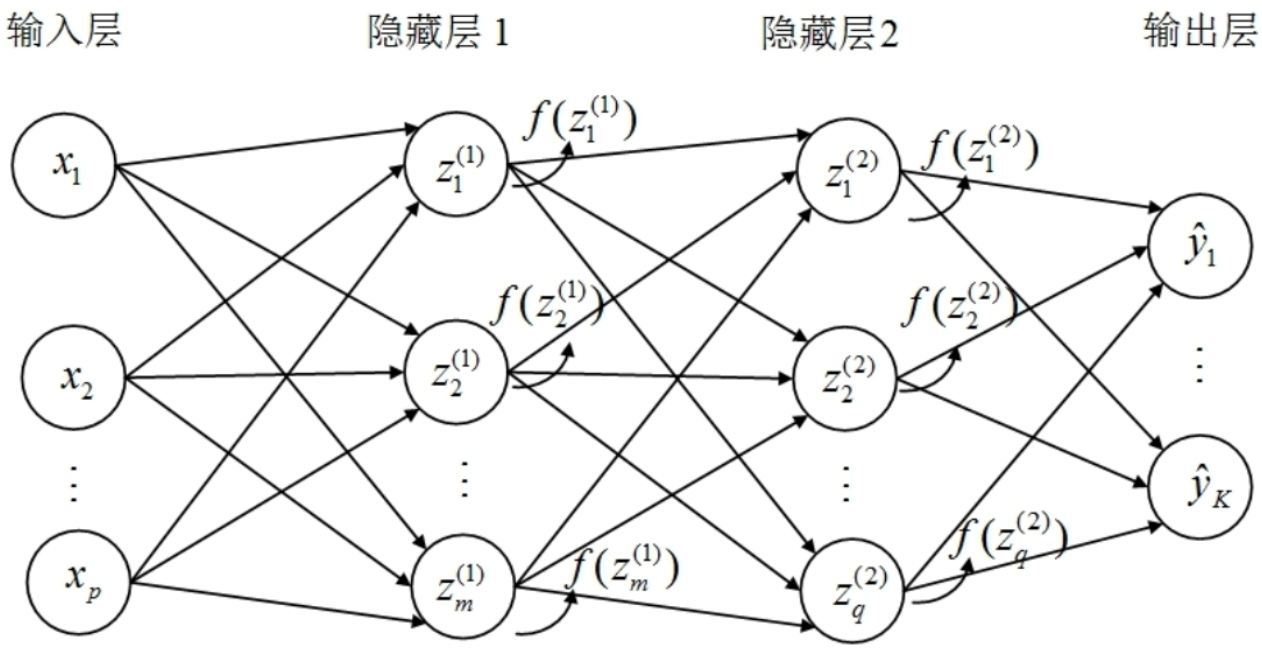

图 20: 基本的神经网络

神经网络

Cybenko（1988）与 Hornik, Stinchcombe and White（1989）使用“泛函分析”(functional analysis)，证明了神经网络的“通用近似定理”(universal approximation theorem)。其主要结论为，包含单一隐藏层的前馈神经网络模型，只要其神经元数目足够多，则可以以任意精度逼近任何一个在有界闭集上定义的连续函数。通用近似定理表明，形如 G(x) 的函数在定义于有界闭集上的连续函数的集合中是“稠密的”(dense)，这意味着对于任意有界闭集上的连续函数，都可以找到形如 G(x) 的函数（也即单隐层的前馈神经网络），使二者的距离任意接近。

向量机回归结果

向量机回归结果如图所示，R 代码与原始数据详见主页

```
Call:
svm(formula = Innovation ~ Digit + Size + Lev + PPE + ROA + Cash + SOE +
    Boardsize + Indboard + BM + Dual + Age + Id + Year, data = data, type = "eps-regressio
n")


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.07142857
    epsilon:  0.1


Number of Support Vectors:  27064
```

图 21: 向量机回归结果

Thanks!