
目 录

1	实验任务	1
1.1	任务提出	1
1.2	国际象棋规则	1
1.3	任务分析	2
2	数据构成	3
2.1	样本数据	3
3	数据处理	4
3.1	标签分类	4
3.2	五折交叉验证(5-Fold Cross Validation)	4
3.3	归一化	5
3.4	高斯核：从低维到高维的映射	5
4	训练实验	6
4.1	LIBSVM	6
4.2	训练参数设置	6
4.3	训练模型	7
5	测试实验	7
5.1	测试样本	7
6	实验评价	8
6.1	混淆矩阵	8
6.1	ROC 曲线(Receiver Operator Characteristic Curve).....	8
	附件 A 程序框图	1

1 实验任务

1.1 任务提出

国际象棋的残局：黑方剩下一个王，白方剩下一个兵和一个王。（如图 1）

当给出一组数据（三个棋子的位置），预测残局的结果。

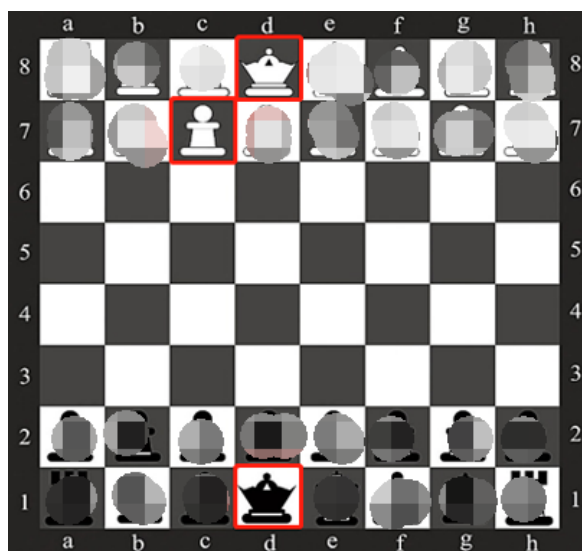


图 1 棋盘的坐标表示

1.2 国际象棋规则

以下介绍国际象棋规则，但本次实验并不需要借助这些规则，只是供给阅读这份报告的老师、同学更加了解这一部分的任务。

国际象棋的棋盘由 64 个黑白相间的八乘八网格组成，分为 1 至 8 行及 A 至 H 列。每位玩家开局时各有 16 个棋子：一王、一后、两车、两马、两象和八兵，各具不同功能与走法。

兵（黑白各八个）：第一步只能向前走一格或者两格，以后每次只能向前走一格，不能后退。但在吃对方子时，则是位于斜前方的那格去吃，并落在那格。若前方有棋子挡住，不能往前面走。当己方的兵走到对方的底线（即最远离己方的一行）时，称为“兵的升变”，此时，玩家可选择把该兵升级为车、马、象或后，但不能变王，也不能选择不变。

王（黑白各一个）：是国际象棋中最为重要的棋子，王被将死即告负。走法是每次横直斜走都可以，但是每次只能走一格。吃子与走法相同。

皇后（黑白各一个）：每次横直斜走都可以，但是每次可以走任意多步，是威力最大的棋子。吃子与走法相同。

相（黑白各两个）：每次只能斜着走若干步。黑方的相永远在黑色格子里，白方的相永远在白色格子里，永不相遇。

马（黑白各两个）：走“日”字。

城堡（黑白各两个）：每次横直走，每次可以走任意多步。

棋手行棋目标是将对方的王处在不可避免的威胁之下以将死对方，也可以通过对方自知无望、主动认输而获胜，另有相当多的情况可导致和局。

逼和：一方的王未被将军，但是移动到任意地方都会被对方将死，则此时是和局。黑方可以利用这一条规则可以与白方逼和。

1.3 任务分析

一个王和一个兵对单王有以下两种结果：

在某些位置下，白方可以将死黑方，白方胜。在其他一些位置，黑方和白方只能是和局。

所以可以分为两类，输入数据为三个棋子的位置坐标和标签。

该问题有数据，有标签，再来一个数据，预测它的标签。因为标签是离散值，所以，此问题为监督学习的分类问题。

我们使用 SVM 来求解划分此数据集的超平面。

给定一个特征空间上的训练数据集 $T = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$ ，其中 $x_i \in R^n$ ， $y_i \in \{-1, +1\}$ ，那么支持向量机(SVM)要解决的是以下的优化问题：

找一组 (w, b) ，满足：

$$\begin{aligned} \text{最小化: } & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^N \xi_i \\ \text{限制条件: } & y_i(w^T \Phi(x_i) + b) \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned} \tag{1-1}$$

即：

$$\begin{aligned} \min_{\omega, b, \xi} \quad & \frac{1}{2} w^T w + C \sum_{i=1}^N \xi_i \\ \text{subject to} \quad & y_i (w^T \Phi(x_i) + b) \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned} \quad (1-2)$$

其中，包含第一个超参数是 C ，超参数(Hyperparameter)是认为指定的，下面我们会在对它进行遍历求最优解。

2 数据构成

2.1 样本数据

数据来源于 UCI 机器学习数据库。数据样本链接为: [Click here](#)。总样本数为 28056 个，其中正样本 2796（和局），负样本 25260（白方胜）。数据格式如图 2。

1	a. 1. b. 3. c. 2	draw	4083	d. 1. g. 3. h. 1	six	23422	c. 1. g. 5. e. 2	fourteen
2	a. 1. c. 1. c. 2	draw	4084	d. 1. h. 2. f. 1	six	23423	c. 1. g. 5. e. 3	fourteen
3	a. 1. c. 1. d. 1	draw	4085	d. 1. h. 4. h. 1	six	23424	c. 1. g. 5. e. 4	fourteen
4	a. 1. c. 1. d. 2	draw	4086	d. 2. a. 1. f. 1	six	23425	c. 1. g. 5. e. 5	fourteen
5	a. 1. c. 2. c. 1	draw	4087	d. 2. a. 1. g. 1	six	23426	c. 1. g. 5. e. 6	fourteen
6	a. 1. c. 2. c. 3	draw	4088	d. 2. a. 1. h. 1	six	23427	c. 1. g. 5. e. 7	fourteen
7	a. 1. c. 2. d. 1	draw	4089	d. 2. a. 2. f. 1	six	23428	c. 1. g. 5. e. 8	fourteen
8	a. 1. c. 2. d. 2	draw	4090	d. 2. a. 2. g. 1	six	23429	c. 1. g. 5. f. 2	fourteen
9	a. 1. c. 2. d. 3	draw	4091	d. 2. a. 2. h. 1	six	23430	c. 1. g. 5. f. 3	fourteen
10	a. 1. c. 3. c. 2	draw	4092	d. 2. a. 3. f. 1	six	23431	c. 1. g. 5. f. 7	fourteen
11	a. 1. c. 3. d. 2	draw	4093	d. 2. a. 3. g. 1	six	23432	c. 1. g. 5. f. 8	fourteen
12	a. 1. c. 3. d. 3	draw	4094	d. 2. a. 3. g. 2	six	23433	c. 1. g. 5. g. 2	fourteen
13	a. 1. c. 3. d. 4	draw	4095	d. 2. a. 4. f. 1	six	23434	c. 1. g. 5. g. 3	fourteen
14	a. 1. c. 4. d. 3	draw	4096	d. 2. a. 4. f. 3	six	23435	c. 1. g. 5. g. 7	fourteen
15	a. 1. d. 1. c. 1	draw	4097	d. 2. a. 4. g. 1	six	23436	c. 1. g. 5. g. 8	fourteen
16	a. 1. d. 1. c. 2	draw	4098	d. 2. a. 4. h. 1	six	23437	c. 1. g. 6. a. 4	fourteen
17	a. 1. d. 1. d. 2	draw	4099	d. 2. a. 4. h. 2	six	23438	c. 1. g. 6. a. 5	fourteen
18	a. 1. d. 1. e. 1	draw	4100	d. 2. a. 4. h. 3	six	23439	c. 1. g. 6. a. 6	fourteen
19	a. 1. d. 1. e. 2	draw	4101	d. 2. a. 5. f. 1	six	23440	c. 1. g. 6. b. 3	fourteen
20	a. 1. d. 2. c. 1	draw	4102	d. 2. a. 5. g. 1	six	23441	c. 1. g. 6. c. 3	fourteen
21	a. 1. d. 2. c. 2	draw	4103	d. 2. a. 5. h. 1	six	23442	c. 1. g. 7. a. 5	fourteen
22	a. 1. d. 2. c. 3	draw	4104	d. 2. a. 6. a. 1	six	23443	c. 1. g. 7. a. 6	fourteen
23	a. 1. d. 2. d. 1	draw	4105	d. 2. a. 6. b. 1	six	23444	c. 1. g. 7. a. 7	fourteen
24	a. 1. d. 2. d. 2	draw	4106	d. 2. a. 6. f. 1	six			

图 2 数据示例

棋盘的坐标表示（如图 3）：

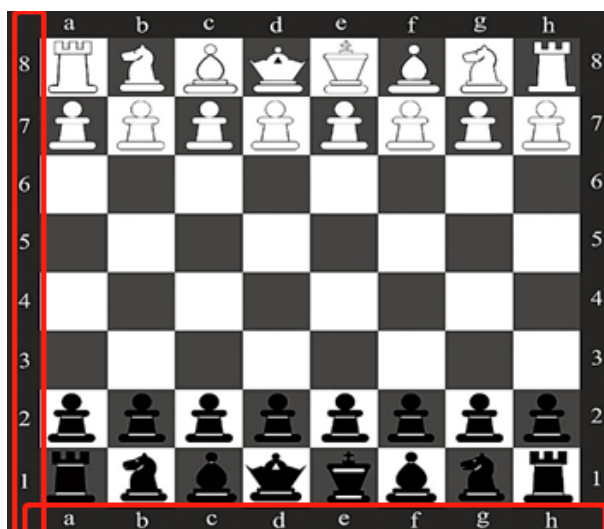


图 3 棋盘的坐标表示

标签有两类，一类代表的意思是和局，用“draw”表示。另一类是白方胜，one、two、...、fifteen 代表的是还有最快多少步可以将死黑方。在官方的文件是这样描述的：optimal depth-of-win for White in 0 to 16 moves, otherwise drawn {draw, zero, one, two, ..., sixteen}. 那就是说，最后的取胜路径（16 步之内），否则，则认为是和局。

3 数据处理

3.1 标签分类

先将输入的 28056 个样本数据的标签分为正负两类，和局记为正样本，共 2796 个；白方胜记为负样本，共 25260 个。

随后打乱数据样本。

3.2 五折交叉验证(5-Fold Cross Validation)

随机取 5000 个样本训练，其余 23056 个样本做(未知)测试。

对于 5000 个训练样本，使用五折交叉验证。（如图 4）

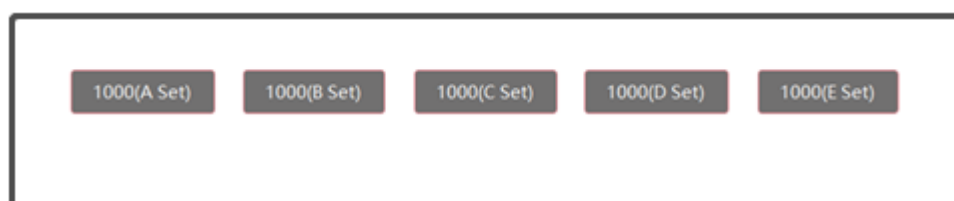


图 4 分集示例

把 5000 个训练样本平均分为五份，每份 1000 个，每一次用 4000 个训练数据训练，其余用 1000 个验证。也就是说，接下来，ABCD 训练，用 E 验证；用 ABCE 训练，用 D 验证；用 ABDE 训练，用 C 验证；用 ACDE 训练，用 B 验证；用 BCDE 训练，用 A 验证。这样将会做出五个模型，最终准确率是五个模型准确率的平均。

3.3 归一化

要对每一个特征做归一化。输入的每组位置坐标是一个六维向量，我们在程序中，把 a 到 h 转化为 1 到 8，每个维度是从 1 到 8 的自然数。

不建议把数据直接输进去，而是对每一维的数据做归一化，

$$newX = \frac{X - mean(X)}{std(X)} \quad (1-3)$$

减掉每个维度的均值，再除以每个维度的方差。这就可以做成一个均值为 0，方差为 1 的一个高斯分布。因为在实际采出来的特征中，我们并不知道特征的性质。

对训练样本取均值和方差，对训练样本和测试样本做归一化。值得注意的是，因为测试样本是未知的，所以测试样本做归一化用的也是训练样本取均值和方差。

3.4 高斯核：从低维到高维的映射

利用核函数，将低维的线性不可分的数据，通过核函数的映射，映射到高维，可能变成线性可分的数据集。

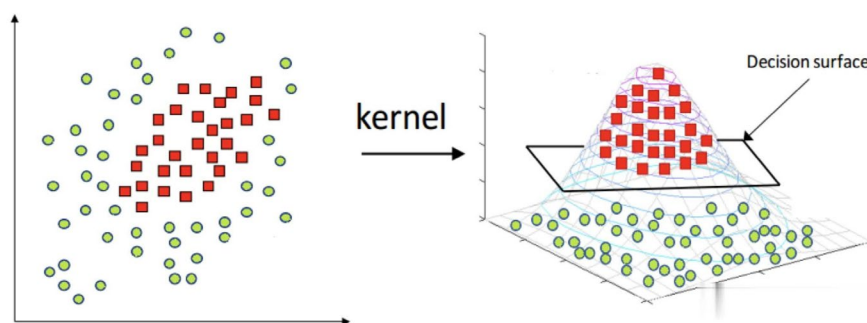


图 5 从低维到高维的映射

本实验选择高斯核函数 (RBF-Radial Basis Function):

$$K(x, y) = e^{-\frac{\|x-y\|^2}{2\sigma^2}} = e^{-\gamma\|x-y\|^2} \quad (1-4)$$

其中，核的超参数是 σ ，令 $\gamma = -\frac{1}{2\sigma^2}$ 。 γ 为本实验中第二个需要优化的超参数。

4 训练实验

4.1 LIBSVM

LIBSVM是台湾大学林智仁 (Lin Chih-Jen) 教授等开发设计的一个简单、易于使用和快速有效的SVM模式识别与回归的软件包。该软件对SVM提供了很多的默认参数，利用这些默认参数可以解决很多问题。我们可以通过设置一下参数来训练获得一些超参数和模型。

4.2 训练参数设置

我们利用`libsvmtrain()`函数返回训练好的SVM分类器模型，参数设置如下：

训练参数设置：

`Svmtrain(yTraining, xTraining, cmd)`

`cmd=['-t 2 -c ', num2str(C(i)), ' -g ', num2str(gamma(j)), ' -v 5'];`

其中，`yTraining` 是真实的标签值，`xTraining` 是输入训练样本，`cmd` 为训练参数配置量，我们选择的 `cmd` 训练参数为：

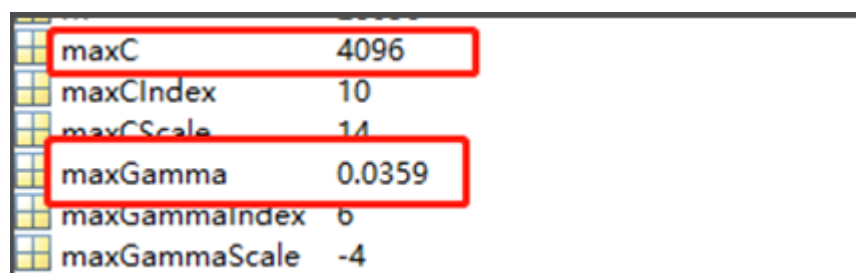
-s svm 类型：	0 - C-SVC
-t 核函数类型：	2 - RBF(径向基)核函数
-c cost：设置 C-SVC 的参数	CValue
-g r(gamma)：核函数中的 gamma 函数设置	GammaValue
-v n：n-fold 交互检验模式，n 为 fold 的个数	-v 5 - 5fold 交互检验模式

设置超参数的区间范围，遍历区间，在 $C \in [2^{-5}, 2^{15}]$ ， $\gamma \in [2^{-15}, 2^3]$ 上做搜索和遍历，每隔两个幂次数取一个值，求最大的识别率，以及以及最大识别率下的 C 和 γ ，在选出一组最大识别率下的 C 和 γ 后，再在那个 (C, γ) 的周围框出一个为 10×10 的邻域，再把

这个邻域做细分，再遍历，选出一个更优的一组 C 和 γ 。

4.3 训练模型

当确定最优的超参数 C 和 γ 后，再把所有训练数据（5000 个）放进去，做一个最后的训练，训练出一个模型。



maxC	4096
maxCIndex	10
maxCScale	14
maxGamma	0.0359
maxGammaIndex	6
maxGammaScale	-4

图 6 训练后获得超参数

5 测试实验

5.1 测试样本

输入 23056 个测试样本(xTesting)做测试，用预测值(yPred)和真值(yTesting)做比较，得出识别率。0 表示全部不识别，1 表示全部识别。

```
.....  
optimization finished, #iter = 329430  
nu = 0.024624  
obj = -374832.583919, rho = 61.554409  
nSV = 208, nBSV = 80  
Total nSV = 208  
Accuracy = 99.4015% (22918/23056) (classification)  
>>
```

图 7 测试实验结果

其中，#iter 为迭代次数， rho 为判决函数的偏置项 b ，决策函数 $wx+b$ 中的常数项 $b=6.2863$ 。

支持向量：288 个，其中 208 个正样本，80 个负样本。只有支持向量对最后算 $w^T \Phi(x_i) + b$ 有贡献，训练样本为 5000 个，最后得到的 288 个支持向量占不到 $\frac{1}{10}$ ，因为支持向量是寻找规律的东西，如果支持向量大于 $\frac{1}{10}$ ，如果支持向量特别多，接近 $\frac{1}{2}$ 的话，

那么该模型是不可用的。因为每次打乱的不一样，所以每次跑出来的结果也略微有区别。

6 实验评价

6.1 混淆矩阵

对于一个系统好不好，不能脱离实际条件，而只看识别率。一般应该用混淆矩阵来衡量。通过实验的结果我们可以得到混淆矩阵如下：

表 6.1 混淆矩阵（个数）：

		预测	
		正样本	负样本
实际	正样本	2264 (TP)	47 (FN)
	负样本	91 (FP)	20654 (TN)

其中，TP(truePositive)是将正样本识别为正样本的数量或概率；FP(falsePositive)是将负样本识别为正样本的数量或概率。TN是将负样本识别为负样本的数量或概率；FN是将正样本识别为负样本的数量或概率。

百分比表示如下：

表 6.2 混淆矩阵（概率/百分数）

		预测	
		正样本	负样本
实际	正样本	97.96% (TP)	2.04% (FN)
	负样本	0.44% (FP)	99.56% (TN)

6.1 ROC 曲线(Receiver Operator Characteristic Curve)

系统的性能可以用唯一的数来表示，这个数叫做等错误率(Equal Error Rate)，我们以 FP 为横坐标，TP 为纵坐标，可以画出 ROC 曲线，同时做出 EER 直线，如图 8：

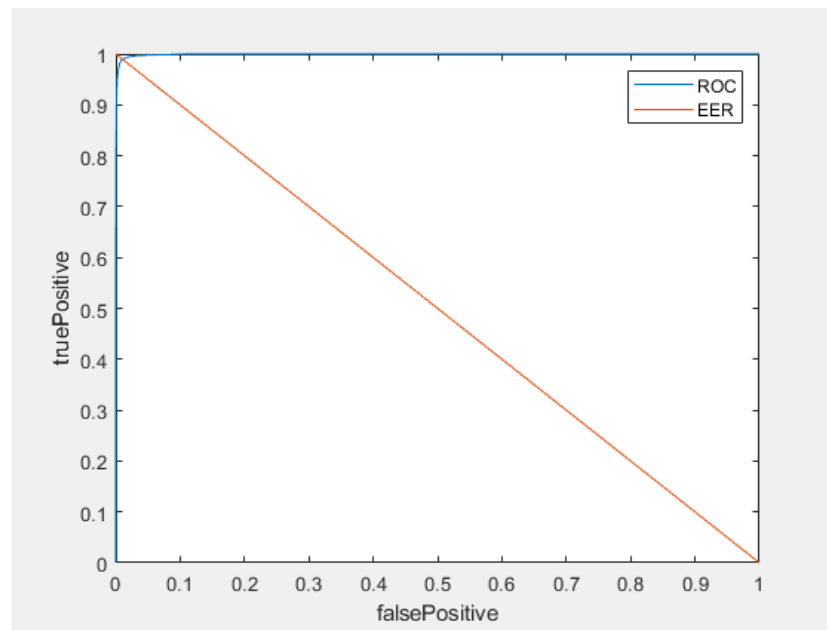


图 8 ROC 和 EER

交点是 FN(将负样本识别为负样本的数量或概率)和 FP(将负样本识别为正样本的数量或概率)相等时的错误率 (因为 $FN=1-TP$)，当 EER 越接近于 0。系统性能越好，因为预测对了更多的真的正值。

也可以采用第二种评判方法：Area Under Curve，即在 ROC 曲线下，与坐标轴围起来的面积越大，则系统越好。

由此可见，我们的系统性能是比较可靠的。

附件 A 程序框图

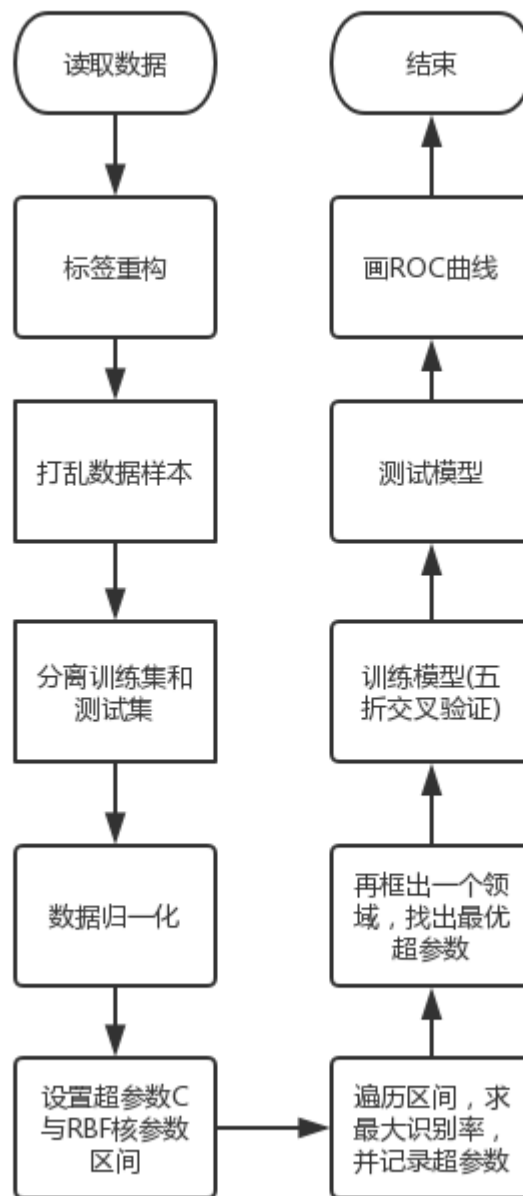


图 9 程序框图