利用 SVM 预测真核生物剪接位点

（生信基地 1801，徐伟，U201812207）

摘要：在真核生物基因转录过程中会发生 mRNA 的剪接，其剪切位点在不同基因组和不同 mRNA 中都具有保守性。通过已有的实验已经证实剪接位点中的 donor 位点后通常是 GT 序列碱基，因此我们根据这个特征利用算法来识别剪接位点。目前已有许多机器学习的算法来进行位点识别，如支持向量机 SVM，贝叶斯网络，WMM，WAM 算法。本文通过 SVM（Support Vector Machines）算法预测真核生物剪接位点中的 donor 位点，并通过 python 实现。利用 KR set（ Kulp & Reese）作为训练集，BG set（Burset & Guigo）作为测试集，对模型进行训练和检验。

关键词：真核生物剪接位点预测，donor 位点，机器学习，SVM 算法

1. 引言

真核生物剪接发生在基因转录的过程中，在许多真核细胞中，基因的编码序列被一段段非编码序列隔开，这些割裂基因的编码序列被称为外显子，而非编码序列被称为内含子，在多细胞生物中，例如人类中，有内含子的基因数目非常多，每个基因所含的内含子也很多（最多的情况下可以达到 362 个）。当一个有内含子的基因转录时，RNA 最初包括了这些内含子。这些内含子被除去以得到成熟的 mRNA，内含子被除去的过程称为剪接。内含子和外显子边界的序列可以帮助细胞识别内含子并将其切除。这些剪接序列大多数存在于内含子中。这些序列根据它们相对于内含子末端的位置被称为 3’剪接位点（acceptor 位点）和 5’剪接位点（donor 位点）[1]。

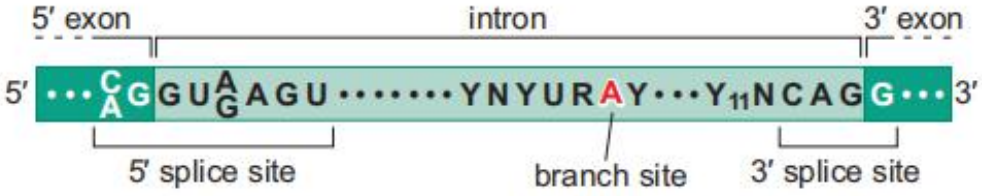


图 1，mRNA 序列内含子与外显子边界。这里是 mRNA 序列，所以 5’剪切位点，即 donor 位点边界为 GU，与我们已知的基因序列为 GT 相同。他们即为 donor 位点的共有序列（consensus sequences）[2]

本文利用 donor 位点的共有序列，通过机器学习算法 SVM 提取其附近序列特征，从而用于真核基因剪接位点预测。支持向量机（SVM）是一种二分类模型，它的基本模型是定义在特征空间上的间隔最大的线性分类器，间隔最大使它有别于感知机；SVM 还包括核技巧，这使它成为实质上的非线性分类器。SVM 的的学习策略就是间隔最大化，可形式化为一个求解凸二次规划的问题，也等价于正则化的合页损失函数的最小化问题。SVM 的的学习算法就是求解凸二次规划的最优

化算法[3]。

2. 目标

本次上机实操目的为通过 python 实现 SVM 算法预测真核剪切位点，从而加深对机器学习和 SVM 算法的理解，通过其预测性能对 SVM 算法进行性能评估，在后续与其他算法进行比较，学习了解 SVM 算法的优劣势，并在实践的过程中锻炼编程能力。

3. 基本思想与假设

理论上，贝叶斯分类器具有最优的性能，即所实现的分类错误率或风险在所有可能的分类器中是最小的。但贝叶斯分类要求确定各个类别的条件概率密度（似然函数）和先验概率。

对于高维特征 x=[x1,⋯,xd]，该似然函数 P(x|ωi) = P(x1,⋯,xd|ωi)是一个高维分布，因此，应用贝叶斯分类器的前提是：

①或者对先验概率和类条件概率密度函数有充分的先验知识；

②或者有足够满足要求的样本，可以较好地进行概率密度估计；

如果这些前提条件不满足，则设计出的分类器往往不具有最优的性质，有时甚至比采用简单方法设计的分类器的效果更差。

所以，在一些实际问题中，后续介绍的基于样本直接设计分类器的方法经常得到更广泛的应用。其基本思想是：不去恢复类条件概率密度，而是利用样本直接设计分类器，即：首先给出类的判别函数，再根据样本集确定判别函数中的未知参数。

在本文中，我们首先获得剪切位点序列和非剪切序列的类别，根据 SVM 模型训练出判别函数，再据此训练好参数的模型用测试集预测其性能。

4. 算法原理[4]

4.1 超平面与支持向量

给定一个训练集数据，分类学习最基本的想法就是基于训练集在平面上找到一条直线使得两类数据正好能够完全分开。如果训练集数据是高维的，则这条直线称为超平面。需要在样本空间中找到一个划分超平面，将不同类别的样本分开。但能将训练样本分开的超平面有很多种。

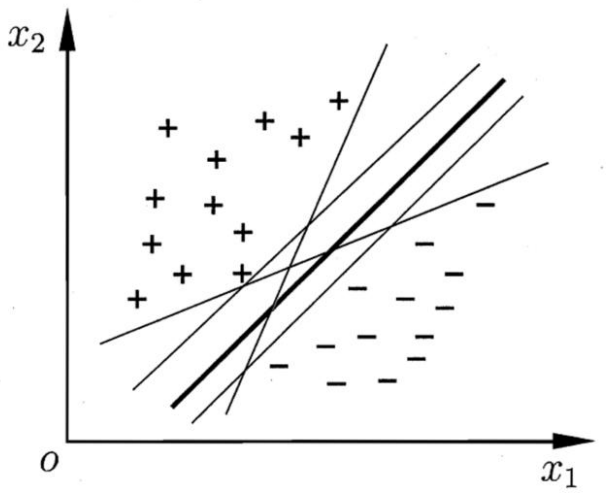


图 1 二维空间分类直线。有多种直线可以将数据区分开。

划分超平面的方程我们可以设为

$$W^TX+b=0 \qquad (1)$$

为了判断样本点所属的类别，令 $f(x)= W^TX+b$，当样本点满足 $f(x)<0$ 时，认为样本为阴性，若样本点满足 $f(x)>0$ 则认为样本点为阳性。为了使得超平面的划分效果最好（面对训练样本局部扰动的“容忍性”最好），需要使得距离超平面最近的这几个训练样本点对划分超平面的间隔尽量大。这些距离超平面最近的这几个训练样本点则称为支持向量。

4.2 寻找最大间隔

4.2.1 点到超平面的距离公式

有了超平面的表达式之后之后，我们就可以计算样本点到平面的距离了。假设 P（X1,X2,X3...Xn）为样本的中的一个点，其中 Xi 表示为第 i 个特征变量。那么该点到超平面的距离 d 就可以用如下公式进行计算：

$$d=\frac{|w1*x1+w2*x2+...wn*xn+b|}{\sqrt{w1^2+w2^2+...+wn^2}}=\frac{|W^T*X+b|}{\|W\|} \qquad (2)$$

其中||W||为超平面的范数，常数 b 类似于直线方程中的截距。

4.2.2 最大间隔的优化模型

在超平面确定的情况下，我们就能够找出所有支持向量，然后计算出间隔 margin。每一个超平面都对应着一个 margin，我们的目标就是找出所有 margin 中最大的那个值对应的超平面。因此用数学语言描述就是确定 w、b 使得 margin 最大。这是一个优化问题其目标函数可以写成：

$$\arg\max_{w,b}\{\min(\ y(w^Tx+b))\cdot\frac{1}{||w||}\} \qquad (3)$$

除上||W||之后，W 和 b 缩放时几何距离是不会改变的，而只随着超平面的

变动而变动，因此几何距离更适合作为优化的计算对象。由此，目标函数相当于定义为：

$$\max \frac{1}{||W||} \qquad (4)$$

为了计算方面，我们将目标函数等价替换为：

$$min \ \frac{1}{2}||W||^2 \qquad (5)$$

通过解决这个优化问题，我们就可以求得“容忍性”最好，即最大间隔的超平面。

4.3 对偶问题

可以通过拉格朗日对偶性转变为其对偶问题：

$$L(w,b,\alpha)= \frac{1}{2}||W||^2-\sum_{(i=1,n)}\alpha_i(y_i(w^Tx_i+b)-1) \qquad (6)$$

这里每一个样本变量都被加上了一个拉格朗日乘子式，将约束条件和目标函数融合在一起。然后我们令：

$$\Theta(W) = max(\alpha i>=0)\ L(W，b，\alpha) \qquad (7)$$

容易验证，当约束条件不满足时，有 $\Theta(W)=\infty$，当所有约束条件满足时有 $\Theta(W)= \frac{1}{2}||W||^2$，因此对偶问题的目标函数可以写成：

$$\min_{w,b} \theta(w) = \min_{w,b} \max_{\alpha_i \geq 0} L(w, b, \alpha) = p^* \qquad (8)$$

将最小化和最大化的进行调换，有：

$$\max_{\alpha_i \geq 0} \min_{w,b} L(w, b, \alpha) = d^* \qquad (9)$$

接着求 L 关于 W 和 b 的最小化，分别求偏导令梯度为 0，可以得到：

$$\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{n} \alpha_i y_i x_i \qquad (10)$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^{n} \alpha_i y_i = 0 \qquad (11)$$

代回 L 得到：

$$\begin{aligned} L(w, b, \alpha) &= \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j - \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j - b \sum_{i=1}^{n} \alpha_i y_i + \\ &= \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \end{aligned} \qquad (14)$$

于是最终我们得到其对偶的优化问题：

$$\begin{aligned} &\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \\ &s.t., \ \alpha_i \geq 0, i = 1, \ldots, n \\ &\sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned} \qquad (15)$$

4.4 松弛变量

如果数据集中存在噪点的话，那么在求超平的时候就会出现很大问题。因此引入一个松弛变量 ξ 来允许一些数据可以处于分隔面错误的一侧。这时新的约束条件变为:

$$y_i(w^T x_i + b) \geq 1 - \xi_i \; i = 1,2...n \qquad (16)$$

目标函数变为：

$$\begin{aligned} &\min \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi_i \\ &\text{s.t.,} \quad \xi_i \geq 0, i = 1,2...n \\ &\qquad y_i(w^T x_i + b) \geq 1 - \xi_i, i = 1,2...n \end{aligned} \qquad (17)$$

后续处理与上述相同，最终得到：

$$\begin{aligned} &\mathrm{W} = \sum_i \alpha_i y_i \mathrm{X}_i \\ &\sum_i \alpha_i y_i = 0 \;\; 0 \leq \alpha_i \leq C \\ &\alpha_i[1 - \xi_i - y_i(\mathrm{W} \cdot \mathrm{X}_i - b)] = 0 \\ &\pi_i \geq 0 \;\; \pi_i \xi_i = 0 \\ &C - \alpha_i - \pi_i = 0 \end{aligned} \qquad (18)$$

只要确定 α，便可解出 W、b

4.5 核函数

把（14）式代入 f(x)表达式，可以得到：

$$\begin{aligned} f(x) &= \left(\sum_{i=1}^{n} \alpha_i y_i x_i\right) x + b \\ &= \sum_{i=1}^{n} \alpha_i y_i \langle x_i, x \rangle + b \end{aligned} \qquad (19)$$

可以发现，经过变换之后表达式成为一个内积的形式，往往我们判断一个样本点的类别时，只要计算其相应的内积即可。

对于简单的分类问题，（18）式形式的超平面可以达到理想的分类效果，但当遇到较为复杂、在当前维度下线性不可分的情况时，则不会有很好的分类效果。此时，我们就需要将样本点通过适当的函数投射到更高维度的空间中，使得样本达到较好的分类效果。设投射函数为 Φ(x)，则有：

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i \langle \phi(x_i), \phi(x) \rangle + b \qquad (20)$$

通过投射函数进行内积计算时在维数极高的情况下是难以实现的，因此就设计了核函数来等效于投射函数的内积计算。

在 SVM 中一般利用三种核函数，即线性核函数（linear）、多项式核函数（poly）、高斯核函数（rbf），对应的公式如下[5]：

$$\text{linear: } K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j.$$

$$\text{polynomial: } K(\mathbf{x}_i, \mathbf{x}_j) = (\gamma \mathbf{x}_i{}^T \mathbf{x}_j + r)^d, \gamma > 0.$$

$$\text{radial basis function (RBF): } K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2), \gamma > 0.$$

1）对于线性核函数，没有专门需要设置的参数

2）对于多项式核函数，有三个参数。-d 用来设置多项式核函数的最高此项次数，也就是公式中的 d，默认值是 3。-g 用来设置核函数中的 gamma 参数设置，也就是公式中的第一个 γ (gamma)，默认值是 1/k（k 是类别数）。-r 用来设置核函数中的 coef0，也就是公式中的第二个 r，默认值是 0。

3）对于 RBF 核函数，有一个参数。-g 用来设置核函数中的 gamma 参数设置，也就是公式中的第一个 γ (gamma)，默认值是 1/k（k 是类别数）

5. 算法实现过程与结果

5.1 数据处理

在本文的数据处理中，仍然选用 donor 位点前 3 个碱基与后 4 个碱基，共 9 个碱基的序列，并对其进行编码实现特征提取，编码的原则为：

A = [1,0,0,0],C = [0,1,0,0],T = [0,0,1,0],G = [0,0,0,1]

所以每个碱基都变成了一个 4 维向量，由于序列共 9 个碱基，所以一个特征共有 4*9=36 维向量。

同时，还对一个 36 维向量进行归一化处理，归一化是指对特征的每一维度分别做归一化[6]，虽然由于 SVM 是线性分类器，貌似不对特征做归一化并不会对最终的实验结果产生较大影响。在机器学习中对特征做归一化目的有:

①避免训练得到的模型权重过小，引起数值计算不稳定；

②使参数优化时能以较快的速度收敛。

5.2 SVM 模型训练

由于在 sklearn 的库中有 SVM 模型可以调用，所以在本次训练中我直接调用了 sklearn.svm，我们需要做的就是对模型进行参数设定、选择合适的核函数，利用测试集对其分类性能进行计算。

训练所用的训练集的阳性数据共 2381 条序列，阴性数据所有第四位是 G、第五位是 T 的序列共有 283607 条序列，但由于阳性数据与阴性数据极度不匹配，相差了 100 多倍，由于阴性数据太多，导致模型训练时间过长，甚至超过了 24 小时还没有运行完，因此我们首先平衡阳性数据与阴性数据，在 283607 条阴性数据中随机选择 2381 条，以达到阳性数据与阴性数据 1:1 平衡的关系，这样模型运行速度会大大加快，后续我们会再讨论平衡数据与性能之间的关

系。

模型训练的主要代码为：

```
model_linear = svm.SVC(C = 0.1, kernel='linear',probability=True)
model_linear.fit(x, y)
model_poly = svm.SVC(C = 0.1, kernel='poly',probability=True,degree = 3)
model_poly.fit(x, y)
model_rbf = svm.SVC(C=0.1, kernel='rbf',probability=True)
model_rbf.fit(x, y)
```

在本次上机中，对于模型参数的训练中，我只选取了参数 C 进行改变训练，而另外的参数 γ 、r、d 则采取默认值的方式。在本文中，r=0, γ = 1/36, d=3。

参数 C：松弛系数的惩罚项系数[7]。如果 C 值设定比较大，那 SVC 可能会选择边际较小的，能够更好地分类所有训练点的决策边界，不过模型的训练时间也会更长。如果 C 的设定值较小，那 SVC 会尽量最大化边界，决策功能会更简单，但代价是训练的准确度。换句话说，C 在 SVM 中的影响就像正则化参数对逻辑回归的影响。

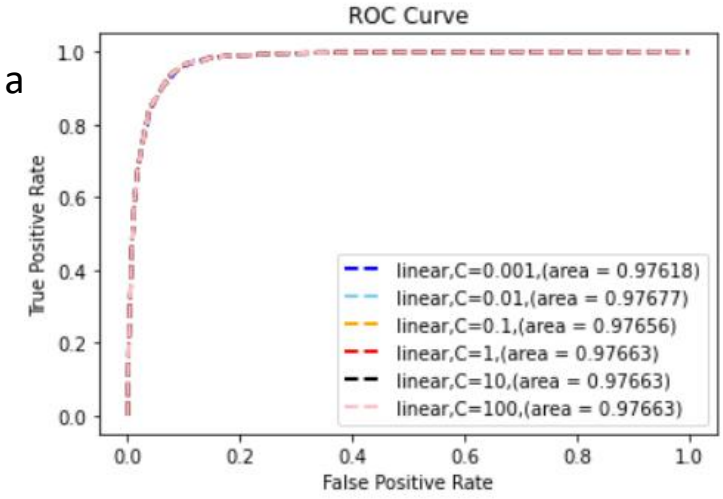
5.3 SVM 模型预测性能

对于测试集，我们仍然选用测试集中的阳性数据 2079 条，所有第四位是 G，第五位是 T 的阴性数据 149206 条。

5.3.1 线性核函数（linear）

在训练线性核函数时，对于模型参数 C，我选取了（0.001，0.01，0.1，1，10，100）这以 10 为倍数的六个备选参数，对其训练后利用测试集进行性能测试。

a



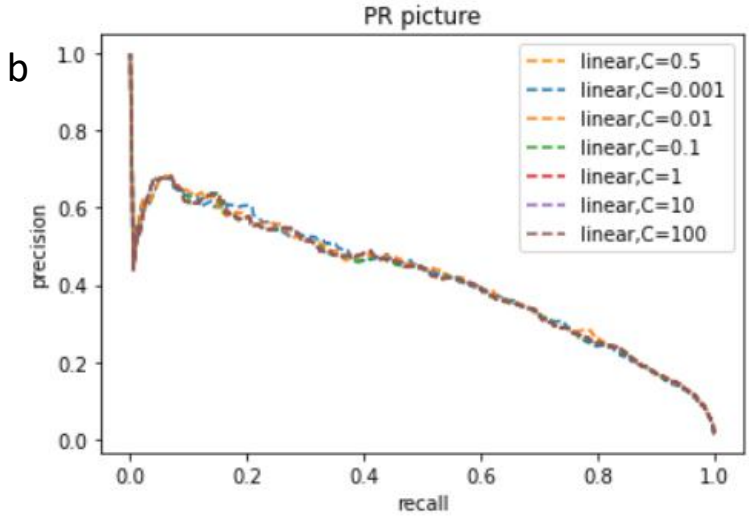


图 2 线性核函数预测性能。a，ROC 图。对于线性核函数的 ROC 图，发现设定不同数量级的 C 值，ROC 图基本完全重合，而 AUC 的值基本都是在 0.976 左右，并没有很大区别。**b，**PR 图。对于不同参数的 PR 图也基本上完全重合，且随着 recall 的增加，precision 也在下降，符合我们的预期。

在线性核函数中，各个参数对预测性能改变并不大，而且预测性能都接近 0.98，预测性能良好。

5.3.2 多项式核函数（poly）

训练多项式核函数时，维度 degree 选择 3，即多项式项数为 3，选取（0.001，0.01，0.1，1，10，100）这以 10 为倍数的六个备选参数 C，对其训练后利用测试集进行性能测试。

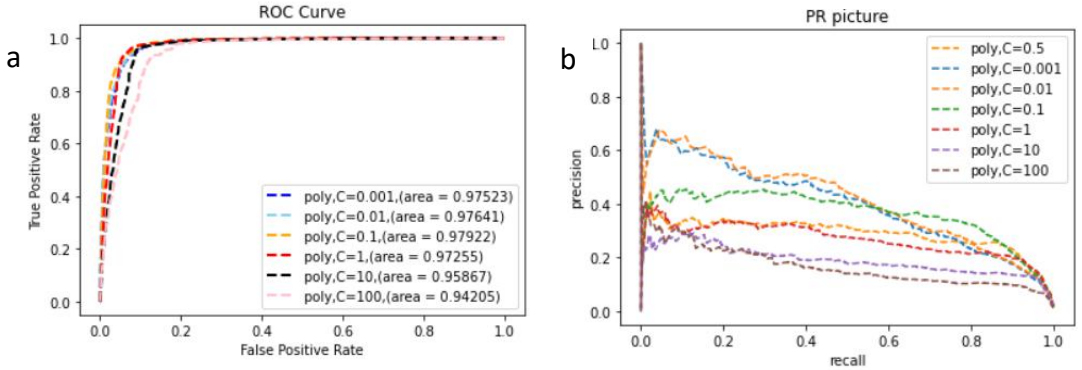


图 3，多项式核函数预测性能。a，ROC 图。在多项式核函数中，不同的数值的参数 C 对预测性能有影响，基本上呈现 C 值越大，预测性能越差的趋势。**b，**PR 图，不同的 C 值对 PR 图也有很大影响，也是基本上呈现 C 值越大，预测性能越差的表现。

在多项式核函数中，参数 C 值对预测性能有比较大的影响，但最好的参数应该在 0.1 附近，由于在 0.1 附近的参数预测性能都在 0.975 以上，且相差不大，没有做进一步的在 0.01 到 1 之间寻找 AUC 值最大的参数，这样做意义不大，因为预测性能几乎相同，且都表现比较好。

5.3.3 高斯核函数（rbf）

训练高斯核函数时，参数 γ 选取默认值 1/k，在本文中 k=36，所以 γ 默认为 1/36，选取（0.001，0.01，0.1，1，10，100）这以 10 为倍数的六个备选参数 C，对其训练后利用测试集进行性能测试。

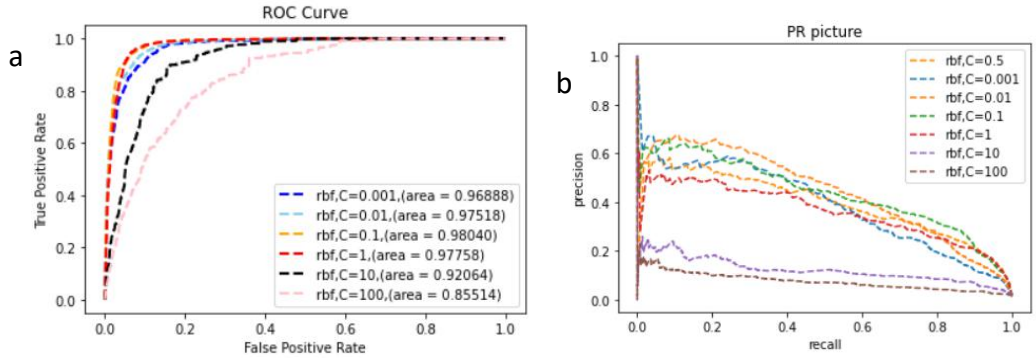


图 4，高斯核函数预测性能。a，ROC 图。在高斯核函数中，不同数值的 C 对预测性能影响更大，最低的预测性能 AUC 达到了 0.855，而最好的预测性能 AUC 达到了 0.98，基本上可以认为最佳参数在 0.1 附近。**b，**PR 图。C 的值对 PR 图影响也比较明显，当 C 在 10 以上是，PR 图表现很差，几乎成为直角。

在高斯核函数中，参数 C 值对预测性能有很大的影响，但最好的参数应该在 0.1 附近，由于在 0.1 附近的参数预测性能都在 0.975 以上，且相差不大，没有做进一步的在 0.01 到 1 之间寻找 AUC 值最大的参数，这样做意义不大，因为预测性能几乎相同，且都表现比较好。

5.3.4 核函数预测性能比较

在上述三种核函数中，我们分别选取了各个核函数中最优的 C 值，即选择每种核函数最优模型放在一起作图比较，由图 2、3、4 可知，每个核函数的最佳参数 C 的值都为 0.1。

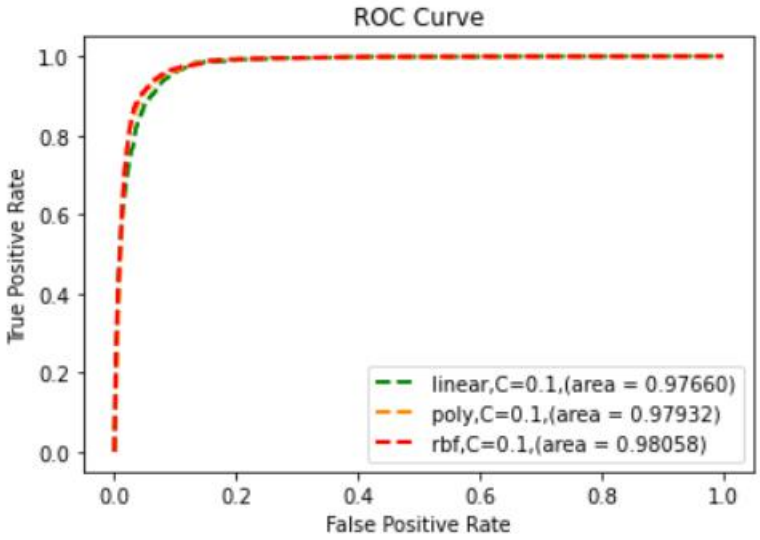


图 5，三种核函数最优模型 ROC 图。当三种核函数都选取最优参数时，表现最好的是高斯核函数 rbf，其实是多项式核函数，最后是线性核函数。但三者的 ROC 曲线几乎重合，AUC

的值也相差在 0.03 左右，AUC 值基本上达到了 0.98，说明三者的预测性能都比较好，当选择的参数比较好时，三者并没有太大的区别。

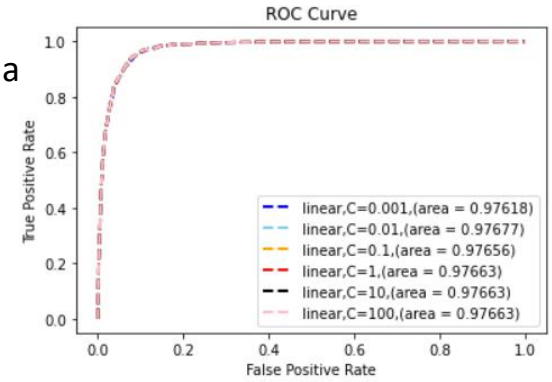
6. 平衡数据对模型影响

前面我们提到训练模型时可选的阳性数据有 2381 条，阴性数据有 283607 条，相差过大，上述实现的算法选择的阴性数据为 2381 条，即阳性阴性数据为 1：1，现在我们讨论当训练数据不平衡时会对算法造成什么样的影响。

考虑到算法的运行时间，阳性阴性数据相同外，我们选择了阴性数据:阳性数据比值为 2，5，10 的倍数的数据来进行预测模型比较。

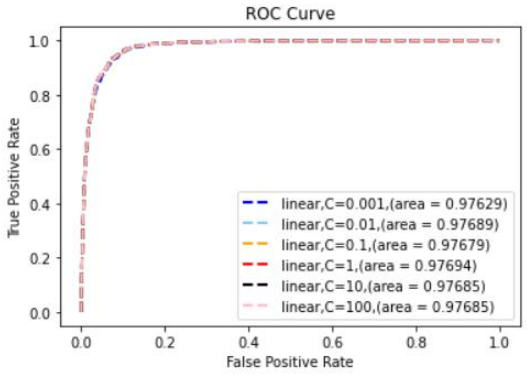
6.1 线性核函数

a



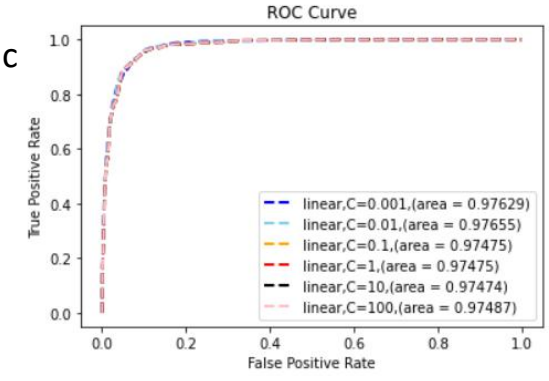

```
maxscore,maxc #最大的ROC面积与对应的C值
(0.9767719604656457, 0.01)
```

b



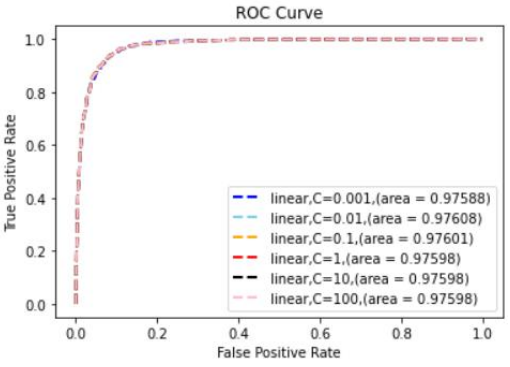

```
maxscore,maxc #最大的ROC面积与对应的C值
(0.9769352232590975, 1)
```

c




```
maxscore,maxc #最大的ROC面积与对应的C值
(0.9765515924450552, 0.01)
```

d

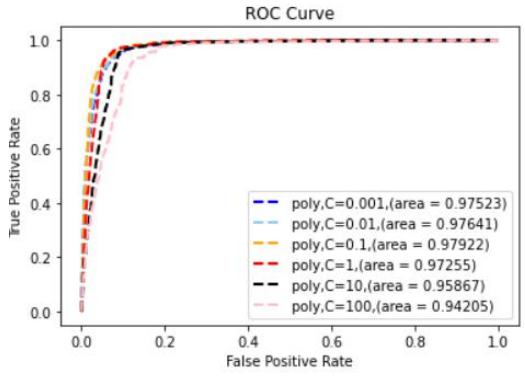



```
maxscore,maxc #最大的ROC面积与对应的C值
(0.9760843089529603, 0.01)
```

图 6，不同的训练数据比对线性核函数预测性能的影响。a-d，阴性数据与阳性数据比值分别为 1，2，5，10。可以看到除了比值为 2 时 C 的最佳参数值为 1，其他的 C 最佳参数值仍然为 0.1，且预测性能都在 0.976 左右，预测性能几乎没有差别。

6.2 多项式核函数

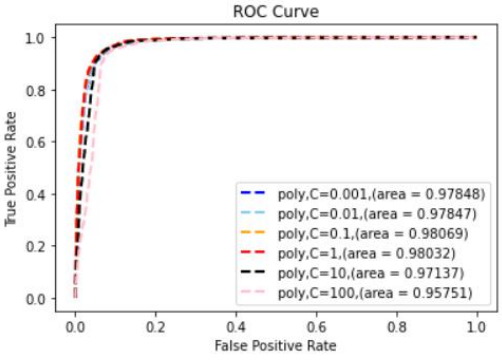
a




```
maxscore, maxc  #最大的ROC面积与对应的C值
```

```
(0.9792205832177414, 0.1)
```

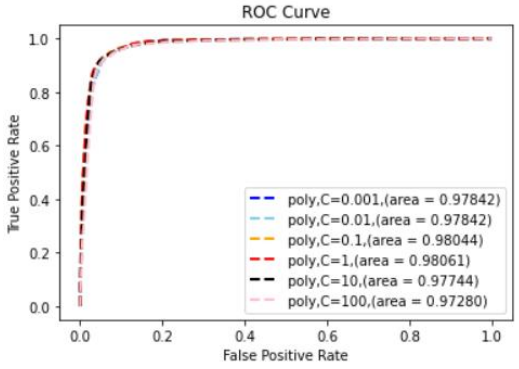
b




```
maxscore, maxc  #最大的ROC面积与对应的C值
```

```
(0.9806943100711447, 0.1)
```

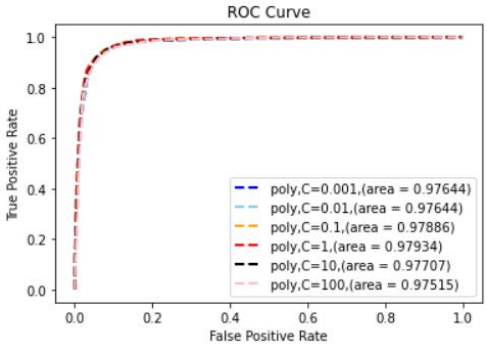
c




```
maxscore, maxc  #最大的ROC面积与对应的C值
```

```
(0.9806122015617612, 1)
```

d

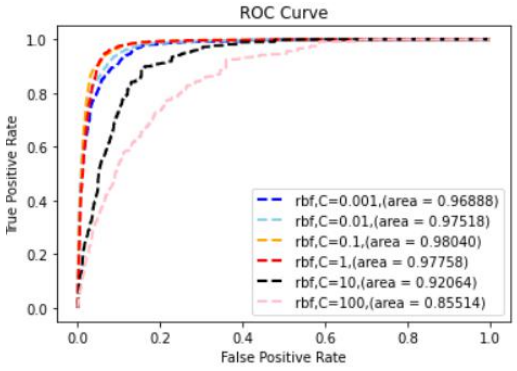



```
maxscore, maxc  #最大的ROC面积与对应的C值
```

```
(0.979344829156499, 1)
```

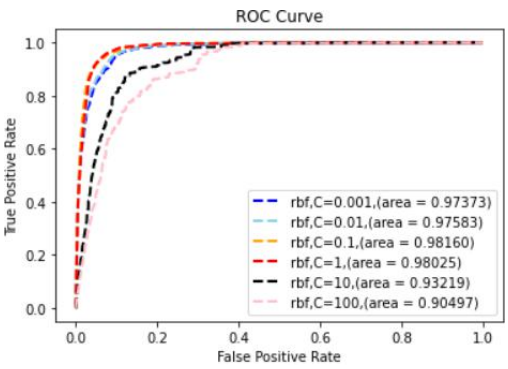
图 7，不同的训练数据比对多项式核函数预测性能的影响。a-d，阴性数据与阳性数据比值分别为 1，2，5，10。可以看到除了比值为 5 和 10 时 C 的最佳参数值为 1，其他的 C 最佳参数值仍然为 0.1，且预测性能都在 0.98 左右，预测性能几乎没有差别。但当阴性数据增多时，当 C 值增大，预测性能并不会明显变化。

6.3 高斯核函数



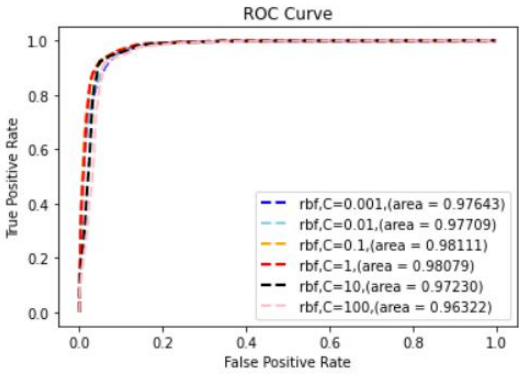

```
maxscore, maxc  #最大的ROC面积与对应的C值
```

```
(0.9803991417465407, 0.1)
```

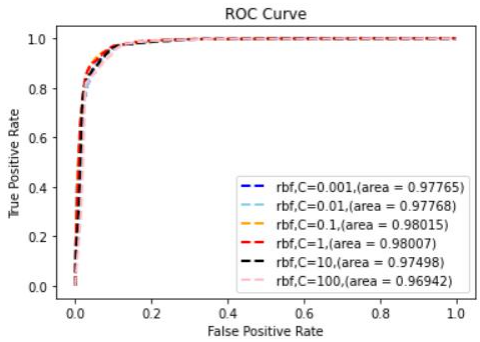



```
maxscore, maxc  #最大的ROC面积与对应的C值
```

```
(0.9816007338560051, 0.1)
```




```
maxscore,maxc #最大的ROC面积与对应的C值
(0.981108704980399, 0.1)
```




```
maxscore,maxc #最大的ROC面积与对应的C值
(0.9801516653452902, 0.1)
```

图 8，不同的训练数据比对高斯核函数预测性能的影响。a-d，阴性数据与阳性数据比值分别为 1，2，5，10。可以看到 C 最佳参数值仍然都为 0.1，且预测性能都在 0.98 以上，预测性能几乎没有差别。但当阴性数据增多时，当 C 值增大，预测性能并不会明显变化。当 C 为 10 或 100 时，不同参数的 ROC 曲线几乎重合。

从图 6、7、8 可以看出，当阴性数据量增多时，对 SVM 模型的预测性能并不会明显增加，由于程序运行时间的问题，当阴性数据过多时，导致程序运行非常慢，可以预测到的是即使阴性数据很多，对 SVM 模型的预测性能也不会有太大的影响。

同时，可以看到当训练数据量增多时，C 值的变化对模型预测性能的改变的影响正在逐渐减小，这也符合 C 值的定义，即松弛量的惩罚程度很大时，模型会更注重准确性。

7. 讨论与结论

在本文的 SVM 模型中，首先对三种核函数都分别取了 6 中参数 C 的值，来分别比较每种核函数的最佳 C 值。其次当选取了三种不同的核函数最合适的参数时，其预测性能都在 0.98 左右，说明这三种核函数都适用与本文所选取的特征用于预测真核生物的剪接位点。而当训练数据集中阳性数据和阴性数据平衡时，模型训练和预测的速度也很快。

对于数据的平衡，在本文中所选取的阴性数据与阳性数据比值为 1，2，5，10 时，并没有发现明显的预测性能的变化，但程序运行的时间却在逐渐增加。在后续的工作中，还可以对训练数据比值为 20，50，100 等进行作图比较预测性能，但可以预见的是运行时间会增加非常多，但预测性能不会有太大的变化。

同时，将 SVM 算法与 WAM 算法的预测性能做比较：

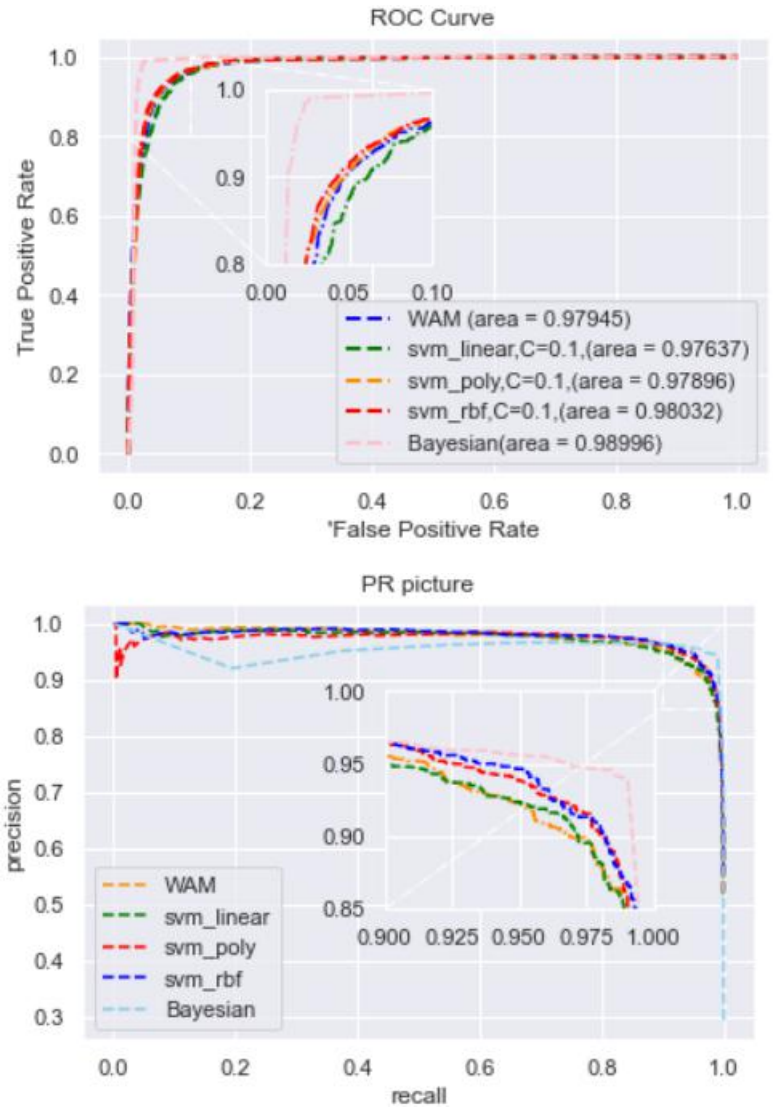


图 9 SVM 算法与其他算法预测性能比较。三种模型的预测性能都在 0.98 左右，ROC 曲线基本重合。

SVM 算法运行速度比贝叶斯网络快，比 WAM 慢，但性能也介于二者之间，且性能都在 0.98 左右。可以说 SVM 的性能和速度综合性价比最高。此外，由于 SVM 算法的复杂度只取决于支持向量的个数，且在训练之前对数据进行了平衡，只用了共 4000 多条训练集序列，故 SVM 的效率理论上要更高。在整个运行过程中，SVM 的训练效率要远远大于贝叶斯网络。因此对于 donor 位点的预测这个问题上，SVM 的性价比要更好。对于训练好模型之后的应用，SVM 预测很快，但贝叶斯网络预测则需要不断的数乘运算，速度非常慢。但性能且没有提升太多。

8. 数据获取

本文算法实现的完整代码与产出图片可在 github 上获取。获取链接如下：
https://github.com/xuweialan/Splicing-site-prediction/tree/main/SVM

9. 参考资料

1.分子生物学书本 Molecular+Biology+of+the+Gene+(7th+Edition) P505
2.分子生物学书本 Molecular+Biology+of+the+Gene+(7th+Edition) P469
3. https://zhuanlan.zhihu.com/p/31886934
4. https://blog.csdn.net/d__760/article/details/80387432
5. https://blog.csdn.net/lqhbupt/article/details/8610443
6. https://blog.csdn.net/breeze5428/article/details/40147839?utm_source=itdadao&utm_medium=referral
7. https://blog.csdn.net/weixin_44507435/article/details/105286110