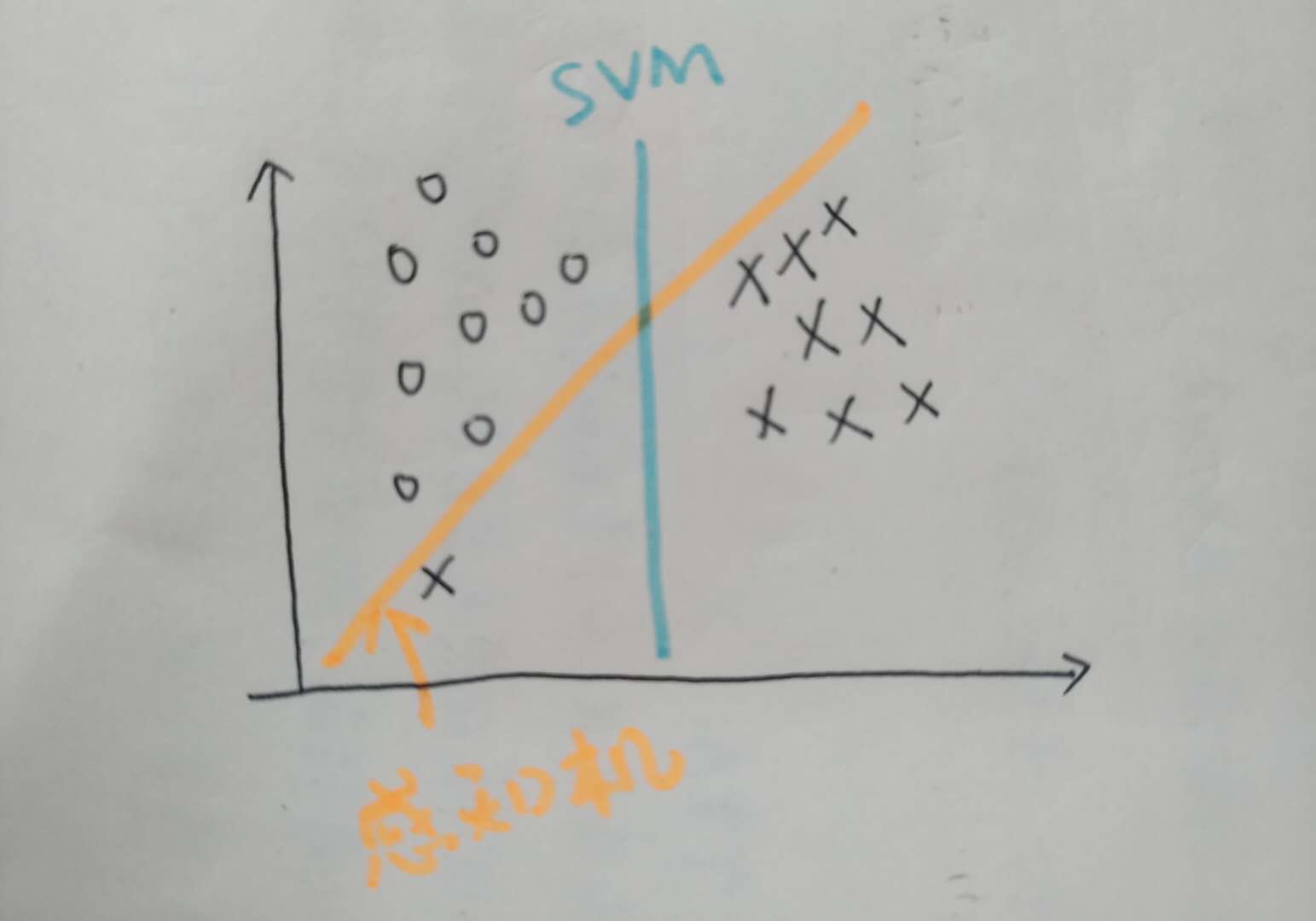
1. （必填）自己提出的问题的理解（罗列全部）：
2. **提出的问题1：**

感知机与SVM的区别是什么？讨论后的理解：

讨论后的理解：首先，感知机和SVM都是属于监督学习的一种分类器。

区别：

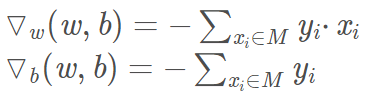
* 感知机追求最大程度的正确划分，同时最小化误差，效果例如下图橙色线，其中蓝色线与橙色线之间的数据很有可能是极端数据或脏数据，因此感知机很容易造成过拟合。



* 支持向量机追求大致正确分类的同时，一定程度上避免过拟合，效果类似下图中的蓝色线。
* 感知机使用的学习策略是梯度下降算法通过不断迭代求解，感知机的目标为：



所谓的梯度下降就是让下面两个式子的绝对值的数值不断降低，降低为接近0为止，其实就是在模仿求导的效果，因为下面两个式子接近0的时候，也就是求导为0的时候。



而SVM采用的是由不等式约束条件构造拉格朗日函数，然后求偏导令其为0，根据一大堆的ai参数（一直迭代到满足kkt条件为止，kkt条件是用来满足不等式约束下的拉格朗日乘子法的泛化），来最终求得w和b。

一般来说拉格朗日函数是符合凸函数的，因此对于凸函数一定存在极值点，也是唯一的最优解。而一般的非凸函数，只好采用梯度下降法一步一步的求得极值点，如果非凸函数还是采用求导令为0，可能找不到极值点！因为鞍点也是导数为0的点，但却不是极值点的特例，如y = x^3函数。导数为0是函数极值点的必要条件。

1. **提出的问题2：**

感知机学习中，在定义损失函数时，为什么可以直接忽略w的L2范数的倒数从而得到该损失函数？

讨论后的理解：

首先，感知机的目标是找到一个超平面将正例负例样本完全分开，根据算法使用场景可以知道数据本身是线性可分的，所以算法最终一定会找到一个这样的平面将数据分开，即最终的损失函数一定会为0。而||w||表示距离，事实上在感知机中不关心具体的距离，而是关注误分类样本的数量，既然最终损失函数一定会下降到0，所以可以在计算过程中忽略w。

1. （必填）别人提出的问题的理解（选择几个问题罗列，并给出理解）：
2. **问题3**：

P44页中为什么说更新次数越多，对学习结果造成的影响越大？

自己的理解：

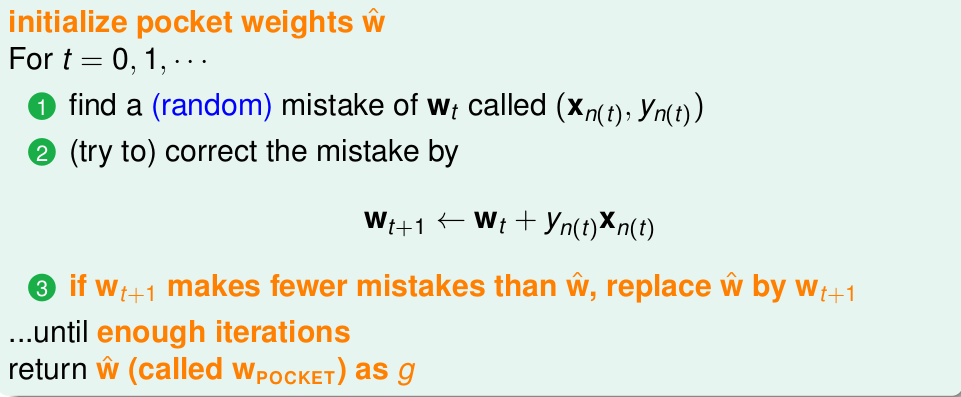
首先，对算法结果影响的点只会是那些被误分类的点，这是由算法目的和损失函数决定的。因此，在讨论这个问题时应该关注那些被误分类的点，之所以数据会被误分类，是因为距离分类平面太近，导致无法被正确分类。而距离超平面越近，就越有可能被误分类。因此对那些更新次数越多，即被误分类次数越多的样本点，几乎每次都会对最终的超平面产生影响，因此说这些样本点对学习结果造成的影响最大。

1. **问题4：**

普通感知机只能处理线性可分问题，那线性不可分的模型怎么处理

自己的理解：

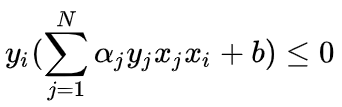
感知机的前提是要求数据必须线性可分，所以当数据不可分时，感知机算法便不可使用。此时可以使用Pocket Algorithm（口袋算法）来解决这个问题，口袋算法基于贪心的思想，总是让遇到的最好的线（或者超平面）保存。首先，假设有一条分割线wt，将数据实例不断带入，发现数据点(xn, yn)再上面出现错误分类，那么就纠正分割线得到w(t+1)，然后我们让wt和w(t+1)遍历所有的数据，看一下哪条线犯的错误少，那么就让w(t+1)代替wt，否则wt不变。



1. **问题5：**

感知机对偶形式相对于普通形式的优点？

对偶形式的目的是在特征空间的维度很高时降低运算量。不妨设特征空间是很大，一共有N个训练数据，N相对n很小。考虑原始形式的感知机学习算法，每一轮迭代中至少都要判断某个输入实例是不是误判点。这里的运算量主要集中在求输入实例x和权值向量w的内积上，O(n)的时间复杂度，由于特征空间维度很高，所以这很慢。而在对偶形式的感知机学习算法中，对于输入实例 (x, y)是否误判的条件变换为



注意到这里所有输入实例都仅仅以内积的形式出现，所以可以预先计算输入实例两两之间的内积，得到Gram矩阵。这样一来每次做误判检测的时候可以在Gram矩阵里查表就能拿到内积，所以这个误判检测的时间复杂度是O(N)。也就是说，对偶形式的感知机，把每轮迭代的时间复杂度的数据规模从特征空间维度 n转移到了训练集大小N上，那么对于维度非常高的空间，自然就能提升性能了。

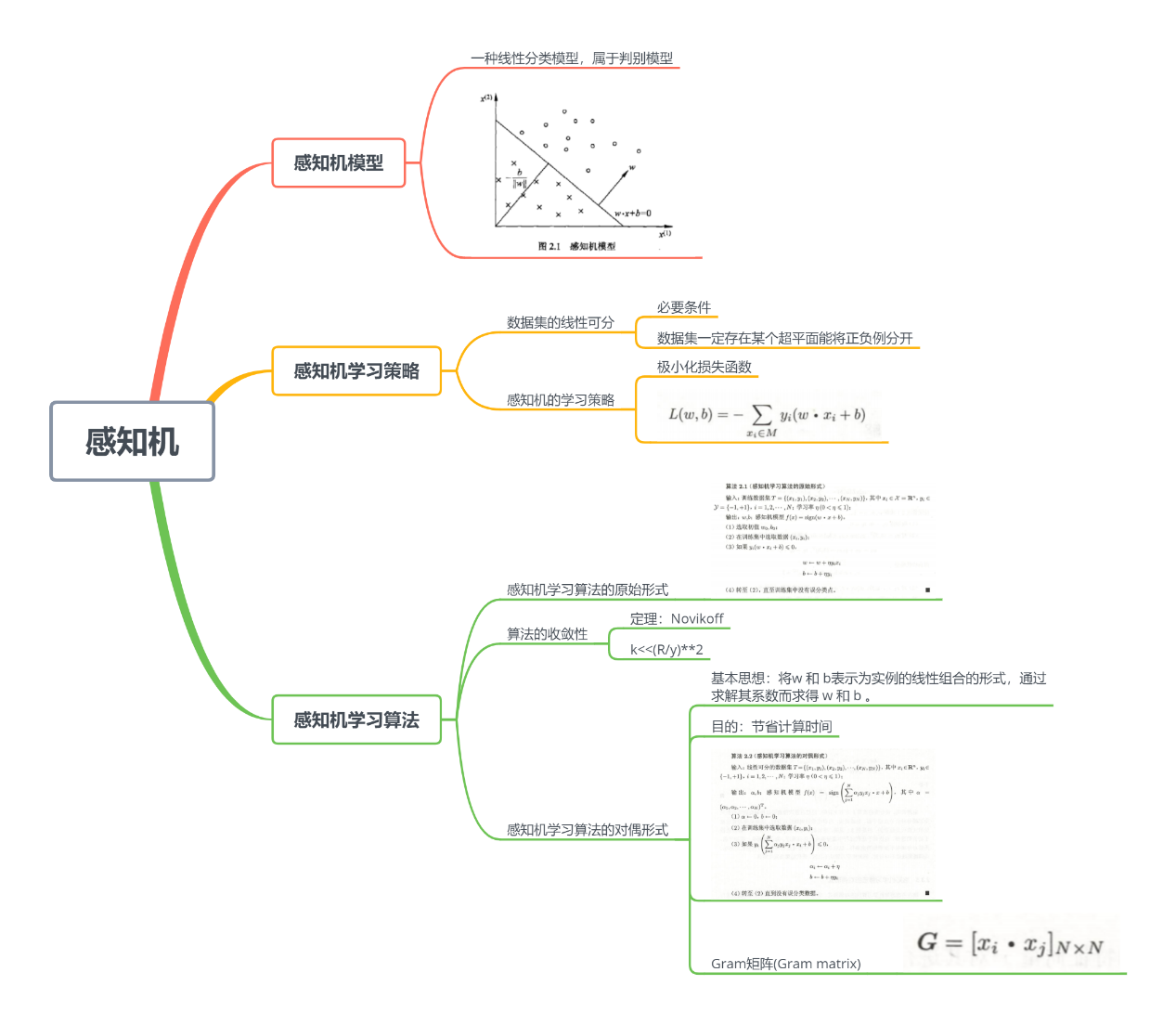
1. （必填）读书计划

1、本周完成的内容章节：统计学习方法（第二章）

2、下周计划：统计学习方法（第三章）

四、（选做）读书摘要及理解或伪代码的具体实现（读书摘要、伪代码的具体实现代码等可以写到这个部分）

1、读书摘要及理解（选做）



2、伪代码的具体实现(选做)