1. （必填）自己提出的问题的理解（罗列全部）：
2. 提出的问题1：课后习题1.2,如何证明条件概率分布模型选择对数损失函数就可以通过经验风险最小化推导极大似然估计？

讨论后的理解： 模型是条件概率分布：Pθ(Y|X)，

损失函数是对数损失函数：L(Y,P(Y|X))=−logP(Y|X)，

经验风险为：

Remp(f)=

=

=

最小化经验风险，也就是最大化，也就是最大化就是极大似然估计。

提出的问题2：P23正则化项如何对应于模型的先验概率，为什么前边说模型越复杂正则化值越大，后面又说可以假设复杂的模型有较小的先验概率

讨论后的理解：正则化可以看作是对模型复杂度的量化指标，因为模型越复杂，正则化要限制的参数空间就越大。复杂模型对变量之间的关系建立了更多约束，相比于简单模型，总的来说更加难以满足，也就是先验概率低。两者并不矛盾，相反地，都反映了正则化对应于模型的先验分布。

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：采用结构化风险最小策略时，如何考虑模型的复杂度表示？

自己的理解：正则化等价于结构风险最小化，其是通过在经验风险项后加上表示模型复杂度的正则化项或惩罚项，达到选择经验风险和模型复杂度都较小的模型目的。

问题4：如何考虑权衡经验风险和模型复杂度的参数？

自己的理解： 正则化等价于带约束的目标函数中的约束项。以平方误差损失函数和L2范数为例，优化问题的数学模型为：

针对上述带约束条件的优化问题，采用拉格朗日乘积算子法可以转化为无约束优化问题，即：

由于参数C为常数，可以忽略，故上述公式和标准的正则化公式完全一致

至于的选择，大致有拟最优原则，L\_曲线法，广义交叉验证，偏差原理，误差极小化准则，无偏差预测风险估计等方法，不展开论述。

 提出的问题5：对泛化误差上界的理解？

讨论后的理解：泛化误差是检验训练后的模型具不具备代表性，用偏差（bias）和方差（variance）来描述。如果一种方法学习的模型比另外一种方法学习的模型拥有更小的泛化误差上界，那么这种方法就更有效。泛化误差上界是样本容量的函数，当样本容量增加时，泛化误差上界趋于0；泛化误差上界也是假设空间容量的函数，假设空间容量越大，模型越难学，泛化误差上界就越大。

1. （必填）读书计划

1、本周完成的内容章节：李航书第一章

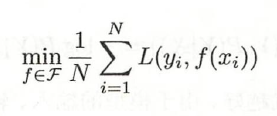
2、下周计划：李航书第二章

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）

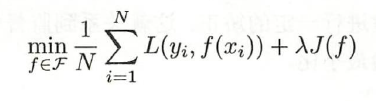
1、读书摘要及理解（选做）

### 【第1章】 统计学习方法概论

1. 统计学习方法包括**监督学习、无监督学习、半监督学习和强化学习。**
2. 统计学习方法三要素：**模型、策略、方法。**
3. 模型：所要学习的条件概率分布或决策函数。所有可能的条件概率分布或决策函数组成模型的假设空间。
4. 策略：  
   4.1 损失函数：能够度量模型一次预测的好坏，值越小（接近0）越好。包括**0-1损失函数、平方损失函数、绝对损失函数、对数损失函数。**  
   4.2 风险函数：能够度量平均意义下模型预测的好坏，包括**期望风险函数和经验风险函数。**  
   4.2.1 期望风险函数：损失函数的期望。但是联合分布P(X,Y)是未知的，因此监督学习就是一个病态问题。  
   4.2.2 经验风险函数：模型关于训练样本集的平均损失。根据大数定理，当样本容量趋于无穷时，Rexp(f) = Remp(f)。  
   4.3 经验风险最小化与结构风险最小化：



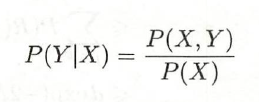
经验风险最小化



结构风险最小化

其中，J(f)为模型复杂度。结构风险最小化需要经验风险和模型复杂度同时小（λ趋于无穷时，后一项的系数收缩作用就增大了，很多系数趋于0，模型复杂度将变小；λ为0，则后一项没有起到收缩作用）。

1. 算法：求解最优模型的方法，找到全局最优解。
2. 使测试误差达到最小的两种方法：**正则化和交叉验证。**
3. 监督学习分为**生成方法和判别方法。**同时，监督学习也可以分为**分类问题（k近邻法、感知机、朴素贝叶斯、决策树、决策列表、logistic回归、SVM、Boosting、贝叶斯网络、神经网络、Winnow算法等）、标注问题（隐马尔科夫模型、条件随机场）、回归问题（数值型数据回归、树回归（CART，既可以分类，又可以回归））。**  
   7.1 生成方法



7.2 判别方法（未展开论述）

1. 分类问题的评价指标：**精确率（P）、召回率（R）、F1（P和R的调和均值：2/F = 1/P + 1/R）。**  
   8.1 **P是指被判别器预测为正类的样本（将正类预测为正类TP+将负类预测为正类FP）中正类正确分类（正类预测为正类TP）的个数，**如判别器预测有100个正类样本，但只有90个是真正预测正确（正类预测为正类），则 P=0.9。  
   8.2 **R是指所有正类样本（正类预测为正类TP+正类预测为负类FN）中，正类正确分类（正类预测为正类TP）的个数**，如样本中有100个正类，但只有80个正类正确分类，则 R=0.8。  
   8.3 **准确率/正确率（A）是指所有样本（正类预测为正类TP+正类预测为负类FN+负类预测为正类FP+负类预测为负类TN）中，判别器正确预测的样本数（正类预测为正类TP+负类预测为负类TN）。**  
   8.4 **希望P、R、F1、A的值越接近1越好。**

2.代码实现

import numpy as np

from scipy.optimize import leastsq

import matplotlib.pyplot as plt

# 目标函数

def real\_func(x):

return np.sin(2\*np.pi\*x)

# 多项式

def fit\_func(p, x):

f = np.poly1d(p)

return f(x)

# 残差

def residuals\_func(p, x, y):

ret = fit\_func(p, x) - y #注意此处没有平方

return ret

regularization = 0.0001

#正则化之后的残差

def residuals\_func\_regularization(p, x, y):

ret = fit\_func(p, x) - y

ret = np.append(ret, np.sqrt(0.5\*regularization\*np.square(p))) # L2范数作为正则化项

return ret

# 十个点'

x = np.linspace(0, 1, 10)

x\_points = np.linspace(0, 1, 1000)

# 加上正态分布噪音的目标函数的值

y\_ = real\_func(x)

y = [np.random.normal(0, 0.1) + y1 for y1 in y\_]

index = 0

plt.figure(figsize=(15, 8))

def fitting(M=0):

"""

M 为 多项式的次数

"""

# 随机初始化多项式参数

p\_init = np.random.rand(M + 1)

# 最小二乘法

p\_lsq = leastsq(residuals\_func, p\_init, args=(x, y))

#p\_lsq = leastsq(residuals\_func\_regularization, p\_init, args=(x, y)) #加入正则化

print('Fitting Parameters:', p\_lsq[0])

# 可视化

plt.subplot(141 + index)

plt.plot(x\_points, real\_func(x\_points), label='real')

plt.plot(x\_points, fit\_func(p\_lsq[0], x\_points), label='fitted curve')

plt.plot(x, y, 'bo', label='noise')

plt.legend()

return p\_lsq

for i in [0, 1, 3, 9]:

lsq\_0 = fitting(i)

index += 1

plt.subplots\_adjust(top=0.92, bottom=0.08, left=0.10, right=0.95, hspace=0.25,wspace=0.35) #调整子图间距

plt.savefig("demo.jpg")

plt.show()