# 复旦大学数学科学学院

# 2021 ～2022 学年第 1 学期期末考试试卷

课程名称：_机器学习与神经网络导论___ 课程代码：_MATH620165__

开课院系：_数学科学学院_____ __ 考试形式：闭卷

姓 名：_刘威_____ 学 号：_21210180060_ 专 业：_应用统计_____

| 题 号 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 总 分 |
|---|---|---|---|---|---|---|---|---|---|
| 得 分 | | | | | | | | | |

（共 8 题，可选 6 题完成，1-5 题为必选题，6-8 题三选一，总分 100 分）

一、 （15 分）叙述至少三种深度卷积神经网络训练的正则化方法。

二、 （15 分）叙述三种以上集成学习方法，并逐个详细阐述。

三、 （15 分）完整叙述并推导贝叶斯估计的最小最大原理。

四、 （15 分）完整叙述并推导梯度提升树（GBDT）的原理与推导。

五、 （20 分)完成叙述并推导结构风险最小化(Structural Risk Minimization）准则与实现方法。

（以下为编程题目，每题 20 分，三选一，解答包含一个简要报告：1.数据处理与准备；2.模型构建；3.训练过程；4.测试集上结果汇报）

六、 通过预测用户对电影评级。实验数据库：http://grouplens.org/datasets/movielens/
建议挑小的数据库。所有作业需要交叉验证来说明你的模型的效率。

七、 深证 B 股指数有大约 50 多家样本股的价格加权平均而得，请通过一段时间的历史数据，挑出尽量少的样本股精确估计该指数。数据来源：http://www.szse.cn

八、 设计神经网络模型完成附件 1 中的回归问题:最后一列是需要回归的数值型输出（表中 sheet1 用于训练； sheet2 用于测试）；预测效果用在测试集的平方和相对误差来刻画。请报告误差的均值及其方差。

（装订线内不要答题）

# 机器学习与神经网络导论

## November 10, 2021

刘威
21210180060

**问题 1.** （15 分）叙述至少三种深度卷积神经网络训练的正则化方法。

**Solution.**

1. 参数范数惩罚项。对目标函数 J 添加一个参数范数的惩罚项 $\omega(\theta)$，限制模型的学习能力。新的目标函数为

$$\hat{J}(\theta; X, y) = J(\theta; X, y) + \alpha\omega(\theta)$$

   其中 $\alpha \in [0, +\infty)$ 为 0 时，没有正则化，$\alpha$ 越大正则化惩罚越大。

2. 提前终止。在每次验证集误差有所改善后，存储模型参数。当训练算法终止时，返回改善后的参数而非最后一次训练时产生的参数。当验证集上的误差在事先指定的循环次数内没有进一步改善时，提前终止训练。

3. dropout。假设一个掩码向量 $\mu$ 指定被包括的单元，$J(\theta, \mu)$ 是由参数 $\theta$ 和掩码 $\mu$ 定义的模型代价函数。dropout 训练的目标是最小化 $E_\mu J(\theta, \mu)$。

4. 数据增强。根据已有数据构造新的数据并添加到训练集中。例如图像识别中，镜面反射、图像像素级移动、缩放等。

■

**问题 2.** （15 分）叙述三种以上集成学习方法，并逐个详细阐述。

**Solution.**

1. Bagging

   在 Bagging 方法中，利用 bootstrap 方法从整体数据集中采取有放回抽样得到 N 个数

据集，在每个数据集上学习出一个模型，最后的预测结果利用 N 个模型的输出得到。具体地：分类问题采用 N 个模型预测投票的方式，回归问题采用 N 个模型预测平均的方式。

2. AdaBoost

   是一种可以用来减小监督学习中偏差的机器学习算法。主要也是学习一系列弱分类器，并将其组合为一个强分类器。刚开始训练时对每一个训练例赋相等的权重，然后对训练集训练 t 轮，每次训练后，对错分类的训练例权重增大，对正确分类的训练样例权重减小。也就是让学习算法在每次学习以后更注意学错的样本，从而得到多个预测函数。

3. Stacking

   Stacking 方法是指训练一个模型用于组合其他各个模型。首先我们先训练多个不同的模型，然后把之前训练的各个模型的输出为输入来训练一个模型，以得到一个最终的输出。理论上，Stacking 可以表示上面提到的两种 Ensemble 方法，只要我们采用合适的模型组合策略即可

4. GBDT

   GBDT 也是一种 Boosting 方法，每个子模型是根据已训练出的学习器的性能（残差）训练出来的，子模型是串行训练获得，不易并行化。GBDT 基于残差学习的算，没有 AdaBoost 中的样本权重的概念。GBDT 结合了梯度迭代和回归树，准确率非常高，但是也有过拟合的风险。GBDT 中迭代的残差的梯度，残差就是目前结合所有得到的训练器预测的结果与实际值的差值。

■

**问题 3.** （15 分）完整叙述并推导贝叶斯估计的最小最大原理。

**Solution.**

**贝叶斯估计的最小最大原理：**

设 $\delta^*$ 为先验分布 $H(\theta)$ 下的贝叶斯估解，且 $\delta^*$ 的风险函数为常数 $c$，即对任意 $\theta \in \Theta$ 有 $R(\delta^*, \theta) = c$，则 $\delta^*$ 为一个最小最大解。其中 $R(\delta, \theta)$ 为风险函数。

**证明：**

反证法，若不然，$\delta^*$ 不是最小最大解，则存在估计量 $\delta$ 使得

$$\sup_{\theta\in\Theta} R(\delta,\theta) < \sup_{\theta\in\Theta} R(\delta^*,\theta) = c$$

故 $R(\delta,\theta) < c$ 对一切 $\theta \in \Theta$ 成立，此时将两边关于 $\theta$ 的先验分布 $H(\theta)$ 求平均可得到

$$R_H(\theta) = \int_\Theta R(\delta,\theta)dH(\theta) < c\int_\Theta dH(\theta) = \int_\Theta R(\delta^*,\theta)dH(\theta) = R_H(\delta^*)$$

即 $R_H(\delta) < R_H(\delta^*)$，其中 $R_H(\delta)$ 为 $\delta$ 的 bayes 风险，$R_H(\delta^*)$ 为 $\delta^*$ 的 bayes 风险，这与 $\delta^*$ 是贝叶斯解矛盾。

■

**问题 4.** （15 分）完整叙述并推导梯度提升树（GBDT）的原理与推导。

**Solution.**

GBDT 是使用了前向分布的迭代算法，弱学习器限定使用 CART 回归树模型。

输入：训练数据集 $D = (x_1, y1), (x_2, y2), ...., (x_n, y_n)$

输出：$f(x) = \sum_{m=1}^{M} \hat{c}_m I(x \in R_m)$

每个树都是输入空间的一个划分，假设已经将输入空间划分为 M 个子集 $R_1, R_2, R_3....R_m$，并且每个子集都有固定的输出值 $c_m$，其中 $I$ 为示性函数。

预测误差：$loss = \sum_{x_i \in R_m} (y - f(x_i))^2$

假设我们前一轮迭代得到的强学习器是 $f_{t-1}(x)$, 损失函数是 $L(y, f_{t-1}(x))$, 我们本轮迭代的目标是找到一个 CART 回归树模型的弱学习器 $h_t(x)$，让本轮的损失函数 $L(y, f_t(x)) = L(y, f_{t-1}(x) + h_t(x))$ 最小。也就是说，本轮迭代找到决策树，要让样本的损失尽量变得更小。

推导：第 t 轮的第 i 个样本的损失函数的负梯度表示为

$$r_{ti} = -[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}]_{f(x)=f_{t-1}(x)}$$

利用 $(x_i, r_{ti})(i = 1, 2, ...m)$，可以你和 CART 回归树，得到第 t 棵回归树，其对应的叶节点

区域 $R_{tj}, j = 1, 2, ..., J$，$J$ 为叶子节点个数。针对每一个叶子节点的样本，得到你和叶子节点最好的输出值 $c_{tj}$，

$$c_{tj} = argmin \sum_{x_i \in R_{tj}} L(y_i, f_{t-1}(x_i) + c)$$

本轮决策树拟合函数：

$$h_t(x) = \sum_{j=1}^{J} c_{tj} I(x \in R_{tj})$$

从而本轮的强学习器的表达式：

$$f_t(x) = f_{t-1}(x) + \sum_{j=1}^{J} c_{tj} I(x \in R_{tj})$$

■

**问题 5.** （20 分）完成叙述并推导结构风险最小化（Structural Risk Minimization）准则与实现方法。

**Solution. 结构风险最小化准则：**

把函数集构造为一个函数子集序列, 使各个子集按照 VC 维的大小排列; 在每个子集中寻找最小经验风险, 在子集间折衷考虑经验风险和置信范围, 取得实际风险的最小。这种思想称作结构风险最小化 (Structural Risk Minimization)，即 SRM 准则。

Vapnik 给予了如下定理：

**Theorem:**

SRM 方法提供了一种风险 $R(\alpha_l^{n(l)})$ 的近似 $Q(z, \alpha_l^{n(l)})$，能够收敛至最小风险 $R(\alpha_0) = \inf_\alpha \int Q(z, \alpha) p(z) dz$，渐近收敛速率 $V(l) = r_{n(l)} + T_{n(l)} \sqrt{\frac{h_{n(l)} \ln l}{l}}$，$r_{n(l)} = R(\alpha_0^{n(l)} - R(\alpha_0)$。若 $n(l)$ 满足 $\lim_{l \to \infty} \frac{T_{n(l)}^2 h_{n(l)} \ln l}{l} = 0$ ，(a) 对于有界结构函数，$T_n = B_n$,(b) 对于一个非负结构函数，$T_n = \tau_n$

**推导：** 考虑 $T_n = B_n$ 即损失函数有界，此时对于 $S_k$ 有：$R(\alpha_l^k) \leq R_{emp}(\alpha_l^k) + B_k \epsilon_k ...$ 其中 $\epsilon_k = \sqrt{\frac{h_k(1 + \ln(2l/h_k)) + \ln \delta/4}{l}}$, 对于 $S_k$ 中的参数 $\alpha_0^k$ 有如下不等式：

$$R(\alpha_0^k) \leq R_{emp}(\alpha_0^k) + B_k \sqrt{\frac{-\ln \delta}{2l}}$$

因此有：

$$\Delta(\alpha_l^{)} = R(\alpha_l^k) - R(\alpha_0^k) \leq B_k(\sqrt{\frac{-\ln\delta}{2l} + \epsilon_k})$$

从而有

$$R_{emp}(\alpha_l^k) \leq R_{emp}(\alpha_0^k)$$

令 $k = n(l)$ 且 $\delta = 1/l^2$ 则有

$$R(\alpha_l^{n(l)}) - R(\alpha_0) \leq r_{n(l)} + B_{n(l)}(\sqrt{\frac{2\ln l}{2l}} + \epsilon_{n(l)})$$

其中 $r_{n(l)} = R(\alpha_0^{n(l)} - R(\alpha_0)$,

从而有 $\lim_{l\to\infty} r_{n(l)} = 0$, 定义

$$V(l) = r_{n(l)} + B_{n(l)}(\sqrt{\frac{2\ln l}{2l}} + \sqrt{\frac{h_{n(l)}(1 + \ln(2l/h_{n(l)}) + 2\ln 4l}{l}})$$

若 $\lim_{l\to\infty} \frac{B_{n(l)}^2 h_{n(l)} \ln l}{l} = 0$，则 $\lim_{l\to\infty} V(l) = 0$ 改写下式

$$R(\alpha_l^{n(l)}) - R(\alpha_0) \leq r_{n(l)} + B_{n(l)}(\sqrt{\frac{2\ln l}{2l}} + \epsilon_{n(l)})$$

为

$$Pr\{V^{-1}(l)(R(\alpha_l^{n(l)}) - R(\alpha_0)) > 1\} < \frac{2}{l^2}, forl > l0$$

从而有：

$$\sum_{l=1}^{\infty} Pr\{V^{-1}(l)(R(\alpha_l^{n(l)}) - R(\alpha_0)) > 1\} < l_0 + \sum_{l=l_0+1}^{\infty} \frac{2}{l^2} < \infty$$

由 Borel-Cantelli 引理有：

$$\overline{\lim_{l\to\infty}} V^{-1}(l)(R(\alpha_l^{n(l)}) - R(\alpha_0)) \leq 1$$

是一个有效的概率。

**实现方法：**

结构风险最小化准则认为经验风险最小的模型是最优模型。求最优模型就是找到一组 $\theta^*$ 使

结构风险损失函数取得最小值

$$\theta^* = \arg\min_{\theta} R_{srm}(\theta)$$

这时结构风险就是最优化的目标函数，监督学习优化问题转化为结构风险函数的最优化问题。

结构风险最小化也可以通过正则化方式实现。结构风险 = 经验风险 + 正则化项。在假设空间、损失函数以及训练集确定的情况下，结构风险的定义如下

$$R_{srm}(\theta) = R_{emp} + \lambda J(\theta) = \frac{1}{N}\sum_{n=1}^{N} L(y, f(x,\theta)) + \lambda J(\theta)$$

式中 $J(\theta)$ 为模型的复杂度，是定义在假设空间 $F$ 上的泛函，常用的有 $L_1$ 范数和 $L_2$ 范数。$J(\theta)$ 可以理解为对模型复杂度的惩罚项。$\lambda > 0$ 用来控制正则化强度，以权衡经验风险和模型复杂度。 ■

**问题 6.** （20 分）设计神经网络模型完成附件 1 中的回归问题：最后一列是需要回归的数值型输出（表中 sheet1 用于训练；sheet2 用于测试）；预测效果用在测试集的平方和相对误差来刻画。请报告误差的均值及其方差。

1. **训练集、验证集、测试集划分**
   数据集总共包含 550 条训练数据 137 条测试集数据，在原本的训练数据上，我将之按照 4:1 划分为用于训练参数的测试集和用于超参选择的验证集。最终各个集合元素个数比为：
   训练集：验证集：测试集 = 440:110:137（约为 4:1:0.8）。

2. **数据分析与标准化**
   该问题训练集样本中包含两类解释变量：30 个数值型（第 0 列至第 29 列）、分类型（第 30 列）。

   (a) **数值型变量**
   我们已知数值型变量是指一些指标分数，可以推断出分数很可能是有界的，观察到这些解释变量 $x_i$ 都是 1 到 7 之间的整数；
   在表2a中，我们列出了数值型变量的数字特征，如均值、方差、最大最小值、分位数等。
   结合实际意义和数字特征两方面，我采用了 MinMax 方法进行归一化处理。
   公式如下，其中 $X = (x^{(1)}, x^{(2)}, ..., x^{(n)})$，$x^{(i)} = (x_1^{(i)}, x_2^{(i)}, ..., x_m^{(i)})^T$，$n$ 为训练集样本数，$m$ 为特征数。

   $$x_{m:scaled}^{(i)} = \frac{x_m^{(i)} - \min\limits_k(x_m^{(k)})}{\max\limits_k(x_m^{(k)}) - \min\limits_k(x_m^{(k)})}.$$

   后续为了记号的简便，用同样的记号表示归一化后的数据，$x_m^{(i)} = x_{m:scaled}^{(i)}$.

   (b) **分类型变量**
   我们已经知道了第 30 列为所服药物，但是由于缺乏药物之间是否有相似作用或想反作用等先验知识，我们无法对该分类变量做有效的 Embidding，因此，为后续处理方便，采用传统的 one-hot 编码形式对分类型变量处理。
   将药物按照是否是（利培酮、喹硫平、奋乃静、奥氮平、氟哌啶醇、阿立哌唑、齐拉西酮）的顺序分别记为 $x_{c1}, x_{c2}, ..., x_{c7}$，$x_{ci} = 1$ 表示处方为是第 $i$ 个药物，否则 $x_{ci} = 0$。

   (c) **数值型被解释变量**
   根据训练集上的数值形式来看，被解释变量似乎不是有界的。因此使用最大最小的方法进行归一化或许会出现些许问题。这里我使用了 Z-score 方法，将被解释变量进行一个可逆的线性变换，这样保证了使用标准化后的模型的输出值可以唯

表 1: 训练集上数值型变量的数字特征

| variance | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|
| 0 | 5.163636 | 0.959568 | 2 | 5 | 5 | 6 | 7 |
| 1 | 3.6 | 1.404192 | 1 | 3 | 4 | 5 | 7 |
| 2 | 4.052273 | 1.772864 | 1 | 3 | 4 | 5 | 7 |
| 3 | 2.890909 | 1.476285 | 1 | 1 | 3 | 4 | 7 |
| 4 | 1.672727 | 1.175754 | 1 | 1 | 1 | 2 | 7 |
| 5 | 4.838636 | 1.094138 | 1 | 4 | 5 | 6 | 7 |
| 6 | 3.722727 | 1.394892 | 1 | 3 | 4 | 4 | 7 |
| 7 | 3.206818 | 1.375162 | 1 | 2 | 3 | 4 | 7 |
| 8 | 3.247727 | 1.391479 | 1 | 2 | 3 | 4 | 7 |
| 9 | 3.615909 | 1.301907 | 1 | 3 | 4 | 4 | 7 |
| 10 | 3.347727 | 1.379807 | 1 | 2 | 3 | 4 | 7 |
| 11 | 2.652273 | 1.44591 | 1 | 1 | 3 | 4 | 7 |
| 12 | 3.281818 | 1.405371 | 1 | 2 | 3 | 4 | 7 |
| 13 | 2.447727 | 1.440385 | 1 | 1 | 2 | 4 | 7 |
| 14 | 2.163636 | 1.303769 | 1 | 1 | 2 | 3 | 7 |
| 15 | 2.443182 | 1.246937 | 1 | 1 | 2 | 3 | 6 |
| 16 | 1.381818 | 0.818152 | 1 | 1 | 1 | 1 | 5 |
| 17 | 2.518182 | 1.310423 | 1 | 1 | 3 | 3 | 6 |
| 18 | 1.856818 | 1.235374 | 1 | 1 | 1 | 3 | 7 |
| 19 | 1.661364 | 0.940779 | 1 | 1 | 1 | 2 | 5 |
| 20 | 2.1 | 1.223908 | 1 | 1 | 2 | 3 | 6 |
| 21 | 3.434091 | 1.472673 | 1 | 2 | 3 | 4 | 7 |
| 22 | 3.934091 | 1.585159 | 1 | 3 | 4 | 5 | 7 |
| 23 | 1.295455 | 0.741836 | 1 | 1 | 1 | 1 | 5 |
| 24 | 2.763636 | 1.356179 | 1 | 1 | 3 | 4 | 7 |
| 25 | 5.529545 | 0.9101 | 1 | 5 | 6 | 6 | 7 |
| 26 | 3.072727 | 1.46615 | 1 | 2 | 3 | 4 | 7 |
| 27 | 2.879545 | 1.557271 | 1 | 1 | 3 | 4 | 7 |
| 28 | 2.856818 | 1.617092 | 1 | 1 | 3 | 4 | 7 |
| 29 | 3.386364 | 1.432075 | 1 | 3 | 4 | 4 | 7 |

一地回到对真实值的预测。公式如下，其中：$\mu$ 和 $\sigma$ 分别表示训练集中，所有样本的被解释变量 $y$ 的均值和方差。

$$y_{zs} = \frac{y - \mu}{\sigma}$$

表 2: 训练集上被解释变量的数字特征

| variance | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|
| y | 57.640909 | 19.238419 | 31 | 42 | 54 | 69 | 157 |

3. **网络模型**

注意到实际意义中，医生结合前 30 个数值型变量开出处方药（第 31 列），再观察治疗后的综合指标。因此我将指标分数（数值型变量）和处方药（分类型变量）分开设计，通过两个不同阶段的模型来达到模拟医生开处方治疗的过程。

**Step1**，神经网络 Network-A 作用在 $x = (x_1, ...x_{30})^T$ 三十个变量上，得到一个向量 $z = (z_1, ..., z_m)^T$ 把它叫做诊断向量，其中 $m$ 是一个超参数，表示 Network-A 的输出层维数。

**Step2**，对于分类变量 $x_{c0}, x_{c1}, ..., x_{c6}$，我们注意到不同的药物在不同的诊断向量 $z$ 上的表现是不全相同的，这符合人的生活认知：不同的药物对生理指标起到的作用是不同的。因此，我们给出了如下假设：

**不同药物之间对判断指标 $z$ 的效果是独立的**。

从而设计出神经网络 Network-B。他的作用是根据分类变量的某个满足 $x_{ci} = 1, x_{cj} = 0, \forall j \neq i$ 的分量 $x_{ci}$，得到一个系数向量 $T \in \mathbb{M}(p, m)$，$p$ 为正整数是一个超参，$m$ 和诊断向量 $z$ 的维数一致。

**Step3**，根据用药情况，诊断向量 $z$ 会根据用药 $x_{ci}$ 的不同受到不同的影响。我们假设这种影响是线性的。因此，一个新的维数 $p$ 的向量 $v$。根据上述理念就有了如下表达式：

$$v = z + T$$

（这是因为：对 $v$ 的每个分量 $v_r = T_r + z_r$，$r \in \{1, 2, ..., p\}$）

**Step4**，将 $v$ 通过网络 Network-C 得到最后的预测结果输出 $\hat{f} \in \mathbb{R}^1$。

4. **损失函数**

该模型是一个回归问题，我们采用了最常用的均方误差（MSELoss）作损失函数，并且予以一定正则化措施。

$$loss_\theta(\hat{F}, Y_{zs}) = \frac{1}{n}\|(\hat{F} - Y_{zs})\|_2^2 + \lambda\|\theta\|_2^2 = \frac{1}{n}\sum_{i=1}^{n}(\hat{f}^{(i)} - y_{zs}^{(i)})^2 + \lambda\|\theta\|_2^2$$

其中，$\hat{F} = (\hat{f}^{(1)}, \hat{f}^{(2)}, ..., \hat{f}^{(n)})^T$，$Y_{zs} = (y_{zs}^{(1)}, y_{zs}^{(2)}, ..., y_{zs}^{(n)})^T$，$n$ 为测试集样本数。

5. **优化器选择**

在优化器选择上，我们选择了 Adam。Adam 是一种替代传统随机梯度下降过程的一阶优化算法，它能基于训练数据迭代地更新神经网络权重。此外，adam 的收敛速度更快。

6. **模型结构**

模型分别对应 Network1, Network2, Network3.

```
MY_MODEL(
  (Network1): ModuleList(
    (0): Sequential(
      (0): ResidualBlock(
        (ln1): Linear(in_features=30, out_features=32, bias=True)
        (dropout1): Dropout(p=0.1, inplace=False)
        (ln2): Linear(in_features=32, out_features=30, bias=True)
      )
      (1): Linear(in_features=30, out_features=16, bias=True)
      (2): ResidualBlock(
        (ln1): Linear(in_features=16, out_features=10, bias=True)
        (dropout1): Dropout(p=0.1, inplace=False)
        (ln2): Linear(in_features=10, out_features=16, bias=True)
      )
    )
  )
  (Network2): ModuleList(
    (0): Sequential(
      (0): Linear(in_features=7, out_features=16, bias=True)
      (1): ReLU()
      (2): Dropout(p=0.1, inplace=False)
    )
  )
  (Network3): Sequential(
    (0): Linear(in_features=16, out_features=1, bias=True)
  )
)
```

图 1: 模型结构与参数

7. **超参选择**
   正则化时超参数 $\lambda$ 的选择可以改善模型的效果，根据验证集展现出的结果，我们取 $\lambda = 0.10125$。

8. **训练过程**
   训练时 loss function 在训练集与验证集上的结果见图2，由于使用了 early stop 的方式，因此并非采取的最后一次的结果。

9. **结果分析**
   将模型输出的预测值 $\hat{f}$ 做 Z-score 的逆映射并四舍五入取整，得到最终的预测值 $\hat{y}$，

$$\hat{y} = \left[\sigma \hat{f} + \mu\right]$$

令 $e_i = \hat{y}_i - y_i$，可以得到 $e$ 相关信息：

- 均值 $\mu = -0.364964$;

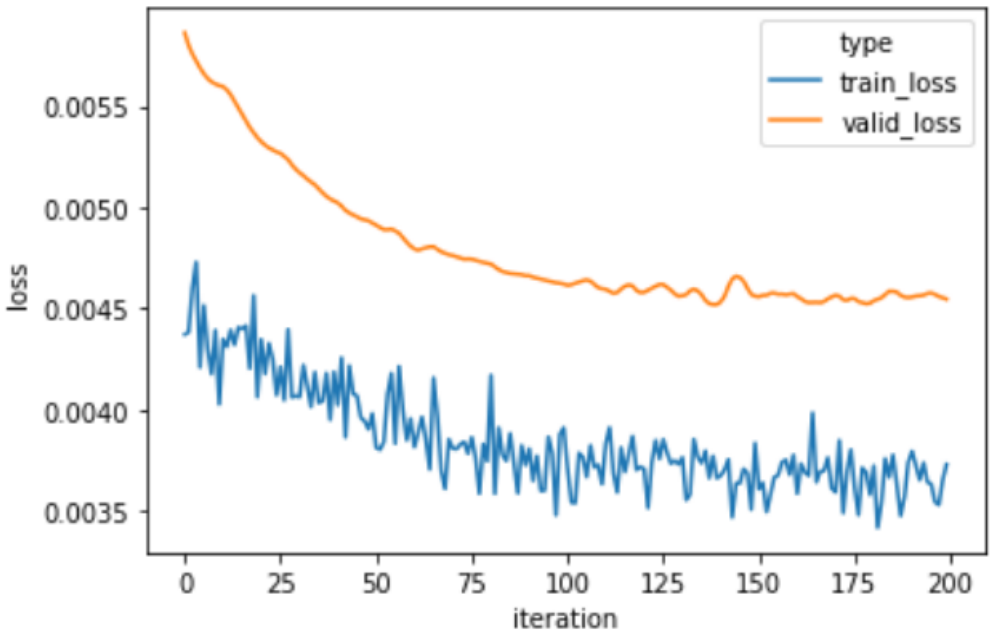


图 2: 训练过程：Loss

- 标准差 $std = 16.514645$；
- 最小值为 $-54$，最大值 32；
- 下四分位数 $Q_1 = -8$，中位数 $Q_2 = 1$，上四分位数 $Q_3 = 11$；
- $|e|$ 的最大值为 54，此时 $\hat{y} - y = -54$，即 $\hat{y}$ 低估了此时的样本。

在全体测试集上有：

- 均方误差 $MSE = 270.8759$；
- 平方和相对误差 $\sum \delta_i^2 = 12.08567$；
- 均方和相对误差 $\sum \delta_i^2/n = 0.08821$；
- 残差分布图见图3；
- 密度估计见图4。

这里给出了 $\hat{y}$ 的残差图以及基于核密度方法的残差密度图像：

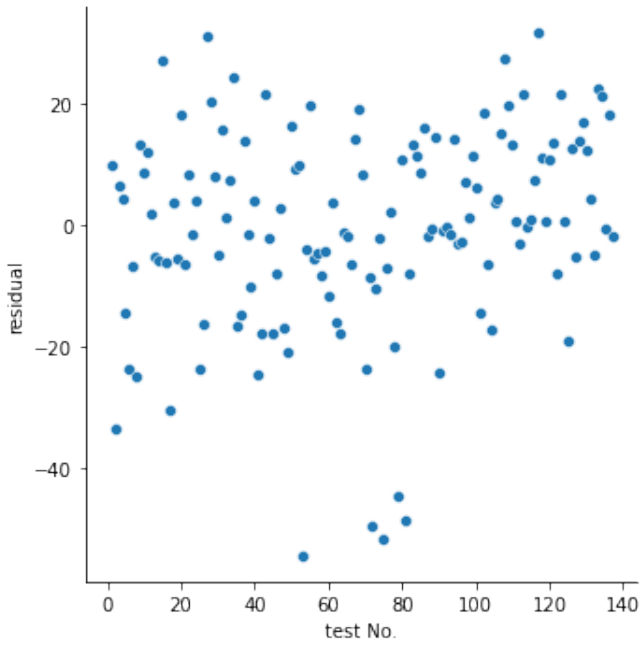


图 3: 残差图

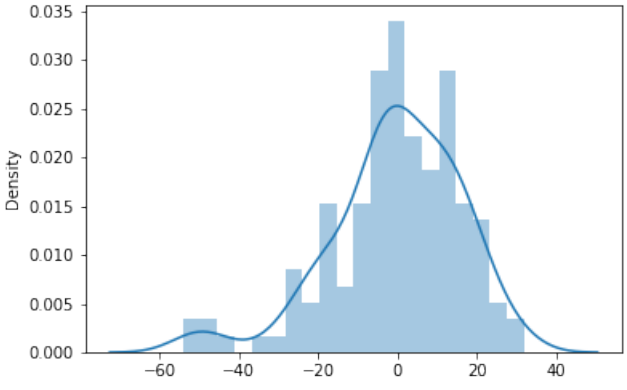


图 4: 残差分布图与核密度估计