

“天池大赛-工业蒸汽量预测”最终报告

成员

- 组长：产子健 (3220190776)
- 组员：苏向迎 (3220190864)
- 组员：王元杰 (3220190889)
- 组员：孙天柠 (3220180737)
- 组员：宋迎新 (3220190975)

1. 简介

本项目为天池大赛赛题——工业蒸汽量预测。主要内容是根据脱敏后的锅炉传感器采集的数据，预测产生的蒸汽量。我们分别基于随机森林方法和深度学习方法构建了两个预测模型，之后进行参数调优等优化，最后对比分析两个模型的预测性能。

2. 问题描述

2.1 问题场景

火力发电的基本原理是：燃料在燃烧时加热水生成蒸汽，蒸汽压力推动汽轮机旋转，然后汽轮机带动发电机旋转，产生电能。在这一系列的能量转化中，影响发电效率的核心是锅炉的燃烧效率，即燃料燃烧加热水产生高温高压蒸汽。锅炉的燃烧效率的影响因素很多，包括锅炉的可调参数，如燃烧给量，一二次风，引风，返料风，给水水量；以及锅炉的工况，比如锅炉床温、床压，炉膛温度、压力，过热器的温度等。

本项目希望可以使用经脱敏后的锅炉传感器采集的数据（采集频率是分钟级别），根据锅炉的工况，预测产生的蒸汽量，从而帮助相关从业者分析火力发电的效能。

2.2 使用数据

赛题网站提供了相应的训练数据 (train.txt) 和测试数据 (test.txt)，其中字段“V0”-“V37”，这38个字段是作为特征变量，“target”作为目标变量。需要利用训练数据训练出模型，预测测试数据的目标变量。

2.3 评价标准

本项目模型预测性能以均方误差 (mean square error) 作为评判标准，希望最终可以取得较低的预测结果均方误差。

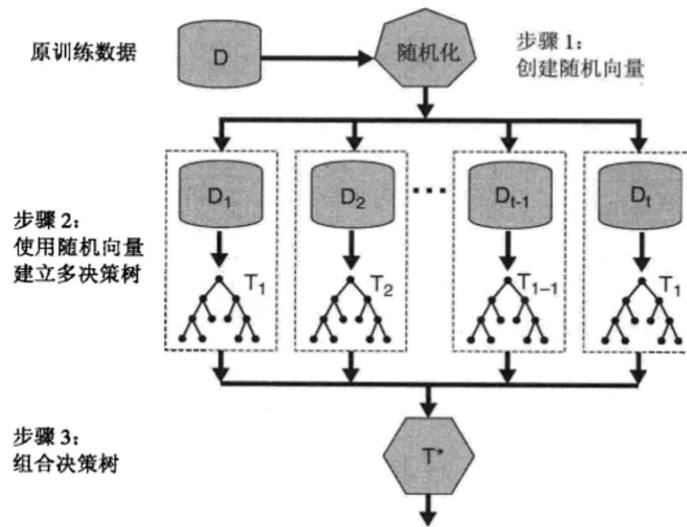
2.4 预期结果

本项目计划利用训练数据训练模型，预测测试数据的目标变量，并尽可能降低预测结果的均方误差。

3. 背景知识

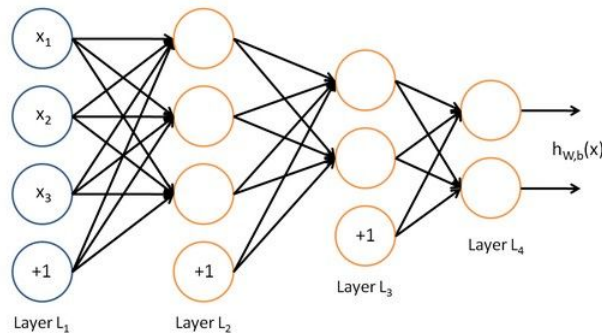
3.1 随机森林

随机森林是由很多决策树融合在一起的算法，它属于Bagging框架的一种算法。每一颗决策树模型在构建的时候训练样本和特征变量是随机抽取的。这种随机性的引入使得随机森林模型不容易陷入过拟合，具有很好的抗噪能力。



3.2 深度学习

深度学习是机器学习的一种，它通过组合低层特征形成更加抽象的高层表示属性类别或特征，以发现数据的分布式特征表示。深度学习具备很强的学习能力，它的神经网络层数很多，宽度很广，理论上可以映射到任意函数，可以解决很复杂的问题。



4. 实现方案

4.1 数据预处理及可视化

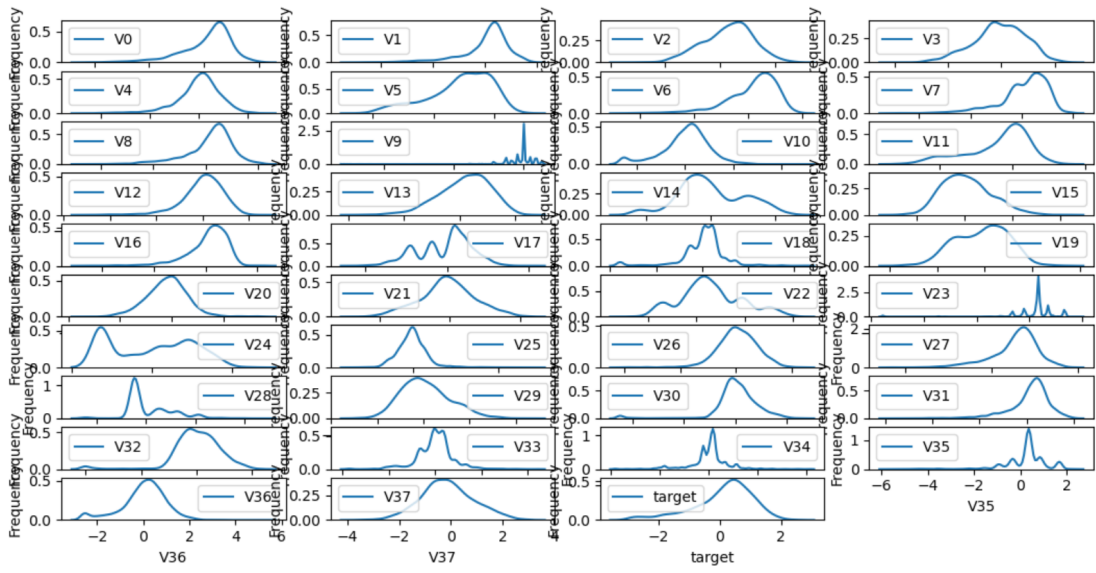
4.1.1 数据预处理

根据对数据集的分析，我们进行如下数据预处理操作：

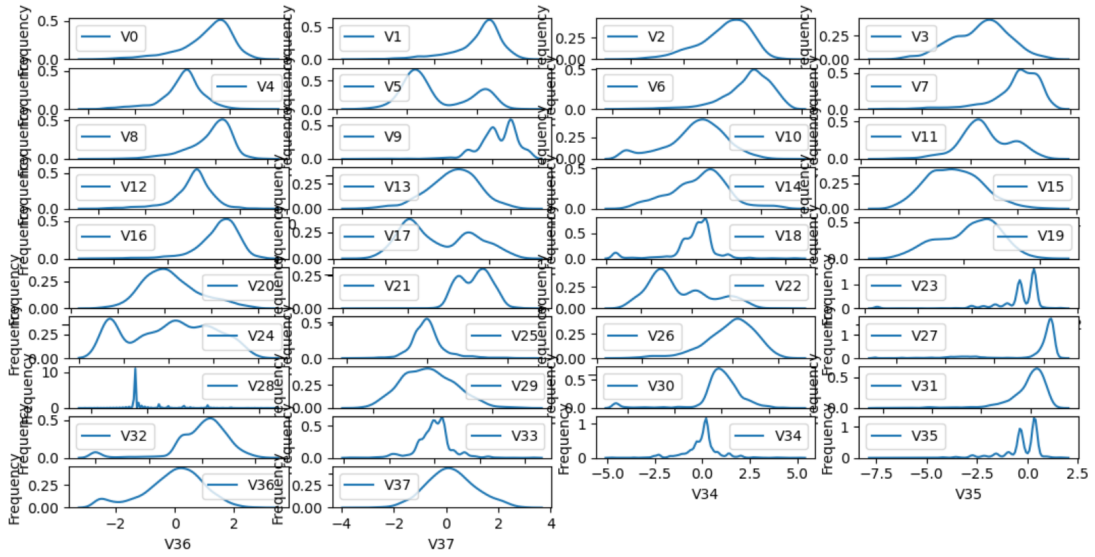
- 官方提供的数据较为清晰和完整，并不存在缺失值的情况，因此不进行缺失值处理；
- 由于各个字段的含义暂不清晰，在此不进行异常值处理；
- 根据4.1.2数据可视化部分展示的特征相关性，我们选取与目标变量相关性绝对值0.1以上的特征变量，同时剔除存在高度相关性的特征变量；
- 数据中个别特征波动较大可能会造成不同特征权重系数变化过大。为了降低这种情况的影响，我们将数据进行z-score标准化。

4.1.2 数据可视化

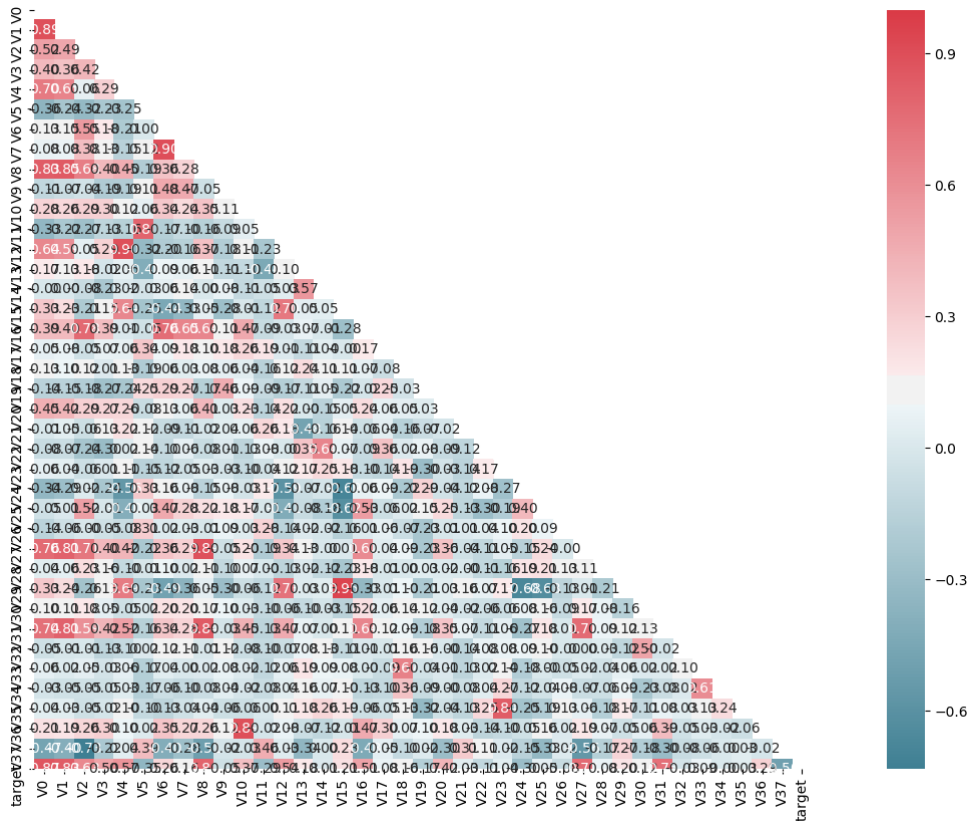
训练数据各字段分布如下图所示：



测试数据各字段分布如下图所示：



特征相关性如下图所示：



特征变量与目标变量相关性绝对值0.1以上的有：

V0,V1,V2,V3,V4,V5,V6,V7,V8,V10,V11,V12,V13,V15,V16,V18,V19,V20,V22,V23,V24,V27,V29,V30,V31,V36,V37;

特征变量相关性0.8以上的有：

V0: V1 V8 ; V1: V8 V27 V31; V4: V12; V5: V11; V6: V7; V8: V27 V31 ; V10: V36; V15: V29; V23: V35;

选取与目标变量相关性绝对值0.1以上的特征变量，同时剔除存在高度相关性的特征变量。剩余变量为：

['V1','V2','V3','V4','V5','V6','V10','V13','V15','V16','V18','V19','V20','V22','V23','V24','V30','V37','target']

4.2 基于随机森林方法的预测模型

4.2.1 模型构建

随机森林的构建过程大致如下：

- 1、从原始训练集中使用Bootstrapping方法随机有放回采样选出m个样本，共进行n_tree次采样，生成n_tree个训练集；
- 2、对于n_tree个训练集，我们分别训练n_tree个决策树模型；
- 3、对于单个决策树模型，假设训练样本特征的个数为n，那么每次分裂时根据信息增益/信息增益比/基尼指数选择最好的特征进行分裂；
- 4、每棵树都一直这样分裂下去，直到该节点的所有训练样例都属于同一类。在决策树的分裂过程中不需要剪枝；
- 5、将生成的多棵决策树组成随机森林。对于分类问题，按多棵树分类器投票决定最终分类结果；对于回归问题，由多棵树预测值的均值决定最终预测结果。

随机森林的实现主要借助机器学习库sklearn，具体用到的工具如下：

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestRegressor #随机森林
from sklearn.model_selection import train_test_split #随机划分训练子集与测试子集
from statsmodels.stats.outliers_influence import variance_inflation_factor #多重共线性方差膨胀因子
```

下面是实现过程：

```
train_data, test_data, train_target, test_target = train_test_split(train, target, test_size=0.2, random_state=0)
m = RandomForestRegressor()
m.fit(train_data, train_target)
score = mean_squared_error(test_target, m.predict(test_data))
print(score)
```

4.2.2 重要参数、属性和接口

衡量指标选用均方误差mean squared error(MSE)，父节点和叶子节点之间的均方误差的差额将被用来作为特征选择的标准，这种方法通过使用叶子节点的均值来最小化L2损失；

接口含有fit,predict，其中fit函数的参数为data，target指定输入集合和标签集合，用来输入训练数据，predict函数用来预测训练，并返回score。随机森林回归并没有predict_proba这个接口，因为对于回归来说，并不存在一个样本要被分到某个类别的概率问题，因此没有predict_proba这个接口；

决策树参数有criterion: "gini" or "entropy"(default="gini")是计算属性的gini(基尼不纯度)还是entropy(信息增益)，来选择最合适的节点。max_features: 选择最适属性时划分的特征不能超过此值。剩下的参数取默认值，包括splitter、max_depth、min_samples_split、min_samples_leaf、max_leaf_nodes、min_weight_fraction_leaf和verbose等；

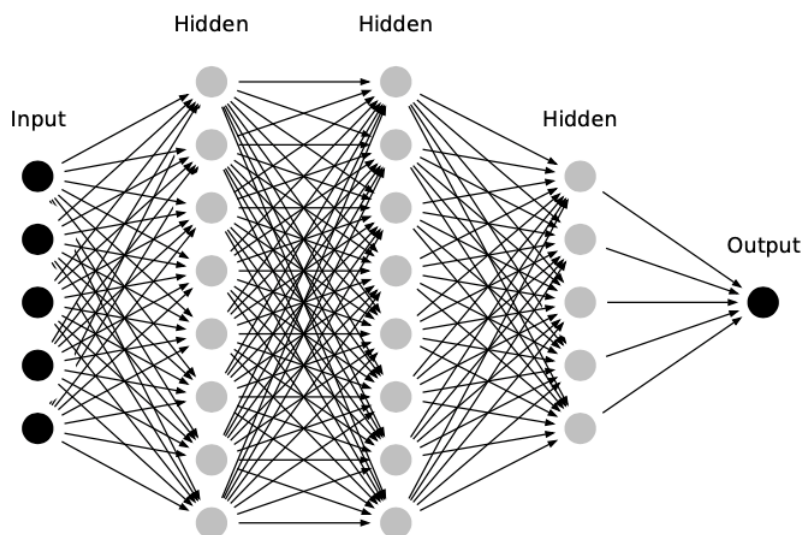
随机森林RandomForestRegressor参数包含n_estimator (决策树个数) max_features(每次选取的特征个数)等。

4.3 基于深度学习方法的预测模型

观察数据可以发现，数据的38个特征之间没有给出明显的相关信息，特征之间没有时间与空间的联系，所以使用全连接网络做预测。

4.3.1 网络结构

使用四层全连接网络，中间隐藏层单元个数分别为50，50，30，在第一层的输出加入dropout防止过拟合。



4.3.2 数据特征选择

对比4.1.2数据可视化部分训练数据和测试数据的特征分布，发现部分特征（V5、V9、V11、V14、V17、V19、V21和V22）分布相差较大，对于预测可能产生干扰，因此只考虑使用训练数据和测试数据分布相近的特征，并按照4.1.2数据可视化部分展示的方法根据特征相关性选取特征变量。

4.3.3 损失函数调整

官方的评估标准为MSE（均方误差），因此在神经网络中使用MSE作为损失函数，同时加入L2正则化，防止过拟合：

$$Loss = MSE + r \times \sum L_2(W)$$

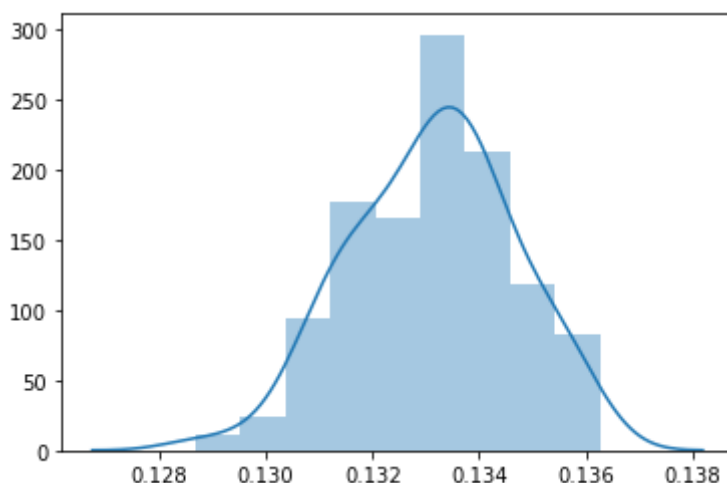
5. 实验

5.1 基于随机森林方法的预测模型

5.1.1 预测模型探索

使用sklearn库的RandomForestRegressor方法在测试集上探索预测模型。进行多次重复，查看模型预测结果的稳定性。

预测结果分布如下：



5.1.2 模型参数调优

使用GridSearchCV对建立的随机森林预测模型进行参数调优, n_estimators分别取[1,5,10,25,50,100], max_features分别取('auto','sqrt','log2')。得到最优的树的数目和特征选取方法。

```
# 模型参数调优
```

```
param_grid = {'n_estimators':[1,5,10,25,50,100],  
'max_features':('auto','sqrt','log2')}  
m = GridSearchCV(RandomForestRegressor(),param_grid)  
m=m.fit(train_data,train_target)  
score=mean_squared_error(test_target,m.predict(test_data))  
print(score)  
print(m.best_score_)  
print(m.best_params_)
```

```
0.13449637250034602  
0.865835600480889  
{'max_features': 'log2', 'n_estimators': 100}
```

使用参数调优后最终建立的预测模型对测试集进行预测, 并保存结果。

```
m = RandomForestRegressor(n_estimators=100,max_features='log2')  
m.fit(train_data, train_target)  
predict = m.predict(test)  
np.savetxt('predict.txt',predict)
```

5.2 基于深度学习方法的预测模型

5.2.1 模型训练

使用pyTorch库进行训练, 训练结果如下:

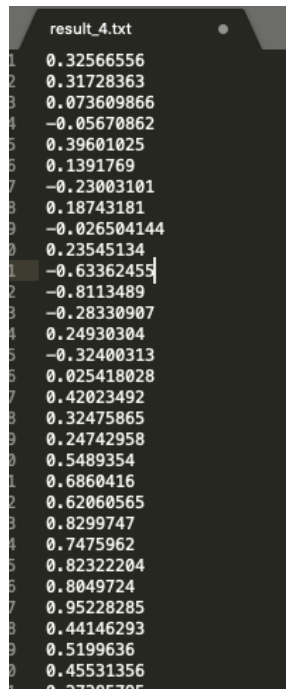
```
epoch:0, loss:3.5377554893493652  
validation loss: 1.3427892923355103  
-----  
epoch:1, loss:3.126810073852539  
validation loss: 1.3047785758972168  
-----  
epoch:2, loss:3.0270791053771973  
validation loss: 1.1794116497039795  
-----  
epoch:3, loss:2.828667402267456  
validation loss: 1.1555625200271606  
-----  
epoch:4, loss:2.721574306488037  
validation loss: 1.1917815208435059  
-----  
epoch:5, loss:2.689497709274292  
validation loss: 1.1775553226470947  
-----  
epoch:6, loss:2.6289291381835938  
validation loss: 1.1275001475147514
```



```
-----
epoch:2994, loss:1.039137840270996
validation loss: 0.3246290683746338
-----
epoch:2995, loss:1.0432790517807007
validation loss: 0.3246290683746338
-----
epoch:2996, loss:1.039317011833191
validation loss: 0.3246290683746338
-----
epoch:2997, loss:1.0424728393554688
validation loss: 0.3246290683746338
-----
epoch:2998, loss:1.040894627571106
validation loss: 0.3246290683746338
-----
epoch:2999, loss:1.0380526781082153
validation loss: 0.3246290683746338
-----
```

5.2.2 模型预测

部分数据集预测结果如下图所示：



```
result_4.txt
1 0.32566556
2 0.31728363
3 0.073609866
4 -0.05670862
5 0.39601025
6 0.1391769
7 -0.23003101
8 0.18743181
9 -0.026504144
10 0.23545134
11 -0.63362455
12 -0.8113489
13 -0.28330907
14 0.24930304
15 -0.32400313
16 0.025418028
17 0.42023492
18 0.32475865
19 0.24742958
20 0.5489354
0.6860416
0.62060565
0.8299747
0.7475962
0.82322204
0.8049724
0.95228285
0.44146293
0.5199636
0.45531356
0.77285705
```

5.2.3 线上评测

将优化好的预测模型放在线上平台评测，预测结果均方差为：0.1823

6. 结果分析

从以上实验结果可以看出，基于随机森林方法的模型预测结果均方误差0.1345要比基于深度学习的模型预测结果均方误差0.1823低一些，这种情况也在天池大赛的讨论区里出现过。针对于上述随机森林预测效果更好的情况，我们分析后认为可能有模型设计方法不同、数据量较小、过拟合等原因。模型设计方式以及算法原理的不同会导致不同的预测效果；深度学习预测适用于数据量较大的场景，本项目官网提供的数据集较小，可能难以满足深度学习训练要求；另外，随机森林本身不易产生过拟合，这可能也是预测效果不同的一个原因。

我们的项目也存在一些不足，例如我们的模型相比较天池大赛讨论区里比较优秀的模型均方误差还是高了一些，仍有一定的提升空间。但总体来说，我们的模型还是实现了预期效果。

