中国研究生创新实践系列大赛

“华为杯”第十六届中国研究生

数学建模竞赛

学　　校		福州大学
参赛队号		**19103860033**
队员姓名	**1.**	聂宽
	2.	潘锡意
	3.	乔锦浩

中国研究生创新实践系列大赛

“华为杯”第十六届中国研究生

数学建模竞赛

题 目 无线智能传播模型

摘 要：

随着 5G 技术的发展，运营商在部署 5G 网络过程中，合理选择覆盖区域内基站站址、预测各种工程参数、地理环境等因素与用户平均接收信号功率(RSRP)之间的关系显得尤为重要，为此我们建立了基于神经网络的预测模型。在特征选择方面，根据 Cost231-Hata 模型设计 4 个特征参数并基于实际工程背景及理论知识，进一步设计了 15 个特征参数。为了衡量不同特征和目标之间的相关性，我们采用随机森林方法进行特征重要性度量。之后选择最为重要的 12 个特征作为神经网络的输入，并采用 dropout 来防止模型过拟合。最后总结了在本次建模过程中存在的不足和未来改进的方向。

一、 问题重述

随着 5GNR 技术的发展， 5G 在全球范围内的应用也在不断地扩大。运营商在部署 5G 网络的过程中，需要合理地选择覆盖区域内的基站站址，进而通过部署基站来满足用户的通信需求。由于无线电波传播环境复杂，会受到传播路径上各种因素的影响，如平原、山体、建筑物、湖泊、海洋、森林、大气、地球自身曲率等，使电磁波不再以单一的方式和路径传播而产生复杂的透射、绕射、散射、反射、折射等，所以建立一个准确的模型是一项非常艰巨而必要的任务。现阶段根据无线传播的研究，学者们提出了三种类型的模型：经验模型、 理论模型和改进型经验模型。但这些模型在实际使用中往往不够精确，所以仍然需要通过采集大量的工程参数以及实际平均信号接收功率（Reference Signal Receiving Power， RSRP）测量值进行经验模型公式的修正。现请建立一个数学模型通过寻找工程参数、地理环境等因素与平均信号接收功率（RSRP）之间的映射关系，从而能够在新的环境中快速预测特定地理位置的 RSRP 值。

从模型设计出发，提出以下三个问题。

问题一：

根据 Cost 231-Hata 模型以及下述数据集信息设计合适的特征。

问题二：

根据数据集设计多个合适的特征，计算这些特征和目标的相关性。并将结果量化排序。

问题三;

建立数学模型并进行 RSRP 的预测。

二、 基本假设和数据集分析

基本假设：

1）相邻小区在 RSRP 测量时不会相互干扰，同一个位置对两个小区进行测量两个小区 RSRP 值不会叠加。

2）用户测量时在地面测量。

3）发射机时定向天线，信号线时信号强度最强的方向

数据集分析：

该数据集包含多个文件，每个文件代表一个小区内数据。其中将 train_set 的 4000 个文件划分为两部分。选取前 3500 个文件作为训练集，用于模型的训练。余下 500 成为验证集，来检验模型的有效性。

文件的每一行代表小区内固定大小的测试区域的相关数据，行数不定(根据小区大小不同， 面积越大的小区行数越多，反之亦然)，列数则固定为 18 列，其中前 9 列为站点的工程参数数据；中间 8 列为地图数据；最后 1 列是用于训练的 **RSRP** 标签数据。每个字段的具体含义如下表所示：

字段名称	含义	单位
Cell Index	小区唯一标识	-
Cell X	小区所属站点的栅格位置，X 坐标	-
Cell Y	小区所属站点的栅格位置，Y 坐标	-
Height	小区发射机相对地面的高度	m
Azimuth	小区发射机水平方向角	Deg
Electrical Downtilt	小区发射机垂直电下倾角	Deg
Mechanical Downtilt	小区发射机垂直机械下倾角	Deg
Frequency Band	小区发射机中心频率	MHz
RS Power	小区发射机发射功率	dBm
Cell Building Height	小区站点所在栅格	m
Cell Altitude	小区站点所在栅格(Cell X, Cell Y)的海拔高度	m
Cell Clutter Index	小区站点所在栅格(Cell X, Cell Y)的地物类型索引	-
X	栅格位置， X 坐标	-
Y	栅格位置， Y 坐标	-
Building Height	栅格(X,Y)上的建筑物高度，若该栅格没有建筑物，则为 0	m
Altitude	栅格(X,Y)上的海拔高度	m
Clutter Index	栅格(X,Y)上的地物类型索引	-
RSRP	栅格(X, Y)的平均信号接收功率，标签列	dBm

三、 符号说明

序号	变量名	含义	单位
1	Height	小区发射机相对地面的高度	m
2	Azimuth	小区发射机水平方向角	Deg
3	Electrical Downtilt	小区发射机垂直电下倾角	Deg
4	Mechanical Downtilt	小区发射机垂直机械下倾角	Deg
5	Frequency Band	小区发射机中心频率	MHz
6	RS Power	小区发射机发射功率	dBm
7	Cell Building Height	小区站点所在栅格的建筑物高度，	m
8	Cell Altitude	小区站点所在栅格海拔高度	m
9	Cell Clutter Index	小区站点所在栅格的地物类型索引	-
10	Building Height	栅格(X,Y)上的建筑物高度，	m
11	Altitude	栅格(X,Y)上的海拔高度	m
12	Clutter Index	栅格(X,Y)上的地物类型索引	-
13	Level_relative_degree	水平相对角	Deg
14	Y_distance	Y 坐标相对距离	m
15	X_diatance	X 坐标相对距离	m
16	Downtilt	天线下倾角	Deg
17	relative_height	基站和栅格的相对高度	M
18	Distance	栅格与发射机距离	M
19	Fa_height	发射机有效高度	m

四、 问题一分析与求解

根据 Cost 231-Hata 模型以及下述数据集信息设计合适的特征。

4.1 问题分析

无线信号从天线发出，以直射、反射、透射、绕射和散射等形式在空间传播，直至移动终端接收。其中，反射、衍射和散射是影响无线信号传播的最关键因素，无线设备接收到的信号主要是直射波、反射波、绕射波和散射波的叠加。

直射：通常发生在视距传播情况下，即基站和移动终端之间无障碍物遮挡，可以认为是自由空间传播。直射造成的信号衰减最小。

反射：当无线信号在传播时遇到远大于其波长的障碍物时，会发生反射，在实际无线环境中发生反射必定会有能量损失。

绕射：当无线信号在传播路径上遇到与入射波波长相近时的障碍物时，信号会绕过障碍物并继续传播，这种现象也称阴影效应。虽然信号经过障碍物后迅速衰落，但是足以能够被移动终端接收。绕射损耗受地形的影响非常大。

散射：当无线信号在传播路径上遇到小于信号波长或表面粗糙障碍物使信号的传播方向发生不确定性改变的现象。在实际传播环境中，树叶、指示牌、路灯柱等都会引起信号反射。

透射：是无线信号穿透介质后继续传播的现象，穿透过程必然伴随能量损耗，损耗的大小与信号频率、物体的物理属性都相关。

基于此经典模型 Cost231-Hata，路径损耗公式如下：

$$\mathrm{PL} = 46.3 + 33.9\log_{10} f - 13.82\log_{10} h_b - \alpha + (44.9 - 6.55\log_{10} h_{ue})\log_{10} d + C_m,$$

其中 PL 定义为传播路径损耗（dB）、f为载波频率（MHz）、h_b为基站天线有效高度（m）、h_{ue}为用户天线有效高度（m）、 α为用户天线高度纠正项(dB)、 d 为链路距离（km） 以及C_m为场景纠正常数（dB）。 RSRP 与 PL 的关系为：

$$\mathrm{RSRP} = P_t - PL$$

其中 Pt 是小区发射机发射功率（dBm）。

无线信号从天线发出，以直射、反射、透射、绕射和散射等形式在空间传播，直至移动终端接收。其中，反射、衍射和散射是影响无线信号传播的最关键因素。因此本文从影响直射反射，衍射，散射的因素出发，根据 Cost231-Hata 设计特征。

4.2 问题求解

根据 Cost231-Hata 设计的特征如下

特征一：小区发射机发射频率（Frequency Band）

选择依据：无线信号在自由空间传播过程中因能量分散而造成传播路径损耗，是最简单的传播模型，能量损耗只和传播距离和电磁波频率有关。表现在模型中为符号 f。

数据获取：由数据集中 Frequency Band 字段直接获得

特征二：栅格与发射机距离 （distance）

选择依据：无线信号在自由空间传播过程中因能量分散而造成传播路径损耗，是最简单的传播模型，能量损耗只和传播距离和电磁波频率有关。即为模型中的 distance

计算方法：由数据集中 cell X ,Cell Y.字段获取基站坐标。根据数据集中 X,Y 字段获取接收点坐标。由勾股转换得到。具体公式如下：

$$\mathrm{distance} = \sqrt{(5(cell\_x - x))^2 + (5(cell\_y - y))^2}$$

特征三：基站和栅格的相对高度（fa_height）

选择依据：对于一个基站站点来说，如果站点相对于接收点相对高度较高，则更容易覆盖到较大的范围。

计算方法：计算基站发射天线的海拔和接收栅格的海拔高度差具体公式如下

$$\text{Relative}_{\text{height}} = \text{height} + \text{cell\_altitude} - \text{altitude}$$

其中 height 为小区站点发射机相对地面高度，cell_altitude 为小区站点所在栅格的海拔高度，altitude 为栅格的海拔高度。

特征四：地物类型（Clutter Index）

选择依据：无线信号在基站至移动终端之间的传播路径上会遇到各种错综复杂的地物和不同的地形，不同的地物和地形对无线信号反射、透射、绕射和散射的影响不尽相同。

计算方法：地物类型可以直接从数据集中 c 中获取。其中不同的地物类型对应的含义如下表所示：

表 4.1 地物类型名称的编码含义

Clutter Index	含义	**Clutter Index**	含义
1	海洋	**11**	城区高层建筑（40m~60m）
2	内陆湖泊	**12**	城区中高层建筑（20m~40m）
3	湿地	**13**	城区<20m 高密度建筑群
4	城郊开阔区域	**14**	城区<20m 多层建筑
5	市区开阔区域	**15**	低密度工业建筑区域
6	道路开阔区域	**16**	高密度工业建筑区域
7	植被区	**17**	城郊
8	灌木植被	**18**	发达城郊区域
9	森林植被	**19**	农村
10	城区超高层建筑（>60m）	**20**	CBD 商务圈

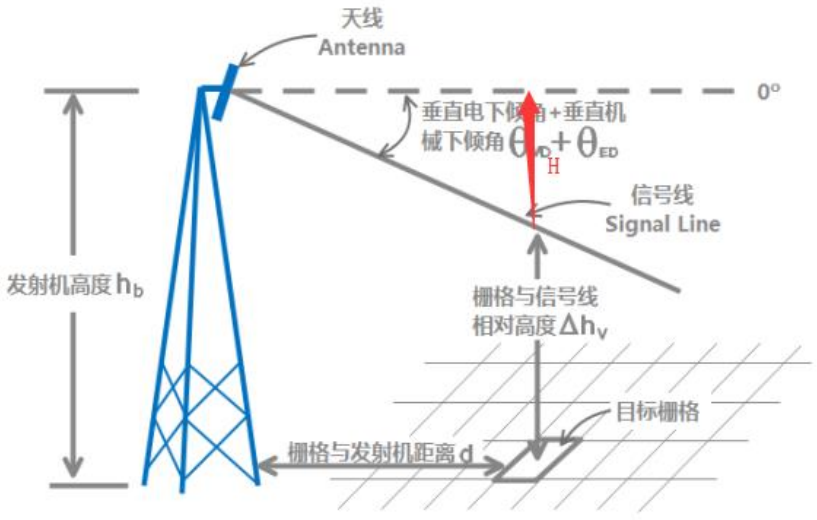


图 4.1 目标栅格和发射机地理位置关系示意图

五、 问题二分析与求解

根据数据集设计多个合适的特征，计算这些特征和目标的相关性，并将结果量化排序。

5.1 问题分析

根据专业理论知识，结合本题实例本文进一步设计了 4 个特征分别如下：小区发射功率（RS Power）、天线下倾角 Downtilt、栅格与信号线相对高度 relative_height、发射机和栅格的相对水平方向角level_Relative_azimuth、发射机和栅格相对水平距离（X-distance）、发射机和栅格相对垂直距离（Y_distance）结合数据集中所给信息，选择 11 个字段字段作为特征小区发射机相对地面的高度（Height）、小区发射机水平方向角（Azimuth）、小区发射机垂直电下倾角（Electrical Downtilt）、小区发射机垂直机械下倾角（Mechanical Downtilt）、小区发射机发射功率（RS Power）、小区站点所在栅格海拔高度（Cell Altitude）、栅格(X,Y)上的建筑物高度（Building Height）、栅格(X,Y)上的海拔高度（Altitude）、

小区站点所在栅格的建筑物高度（Cell Building Height），同时结合问题一中给出的 4 个特征小区发射机和栅格之间的相对距离（distance）、小区发射机中心频率（Frequency Band）、基站和栅格的相对高度 （relative_height）、地物类型（Clutter Index）。共 19 个特征。采用随机森林进行特征重要性度量，最后选择 12 个有效特征作为建模的关键特征。

5.2 问题求解

5. 2. 1 选取特征

特征五：小区发射机功率（RS Power）

选择依据：无线信号在传播过程中因能量分散而造成传播路径损耗， 由 RSRP 与损耗 PL 的关系。

$$\text{RSRP} = P_t - PL$$

可知当发射功率大使，相同损耗下，接收到的功率也大。

计算方法：由数据集中 RS Power 字段直接获得

特征六：天线下倾角 downtilt

选择依据：发射机发射的信号线相对于其他方向上功率较大，天线下倾角能够影响到信号线下

计算方法：天线下倾角由垂直电下倾角和垂直机械下倾角叠加组成，具体计算公式为：

$$\text{downtilt} = \text{electrical\_downtilt} + \text{mechanical\_downtilt}$$

特征七：栅格与信号线相对高度 (relative_height)

选择依据：发射机发射的信号线相对于其他方向上功率较大，因此接收栅格的位置距离信号线的垂直高度也会影响到接收栅格的接收功率。示意如图 5.1 所示。

计算方法：

$$\text{relative height} = \text{fa height} - \text{H}$$

$$\text{H} = \text{distance} \times \tan(\text{downtilt})$$

其中 H 如图 4.1 所示，downtilt 为天线下倾角

特征八： 发射机和栅格的相对水平方向角Relative_azimuth

选择依据：如图所示为基站天线辐射方向图，接收栅格和发射机的不同方位会影响到接收强度。发射机发射的信号线相对于其他方向上功率较大，因此接收栅格的位置偏离信号线水平方向的程度也会影响到接收栅格的接收功率。

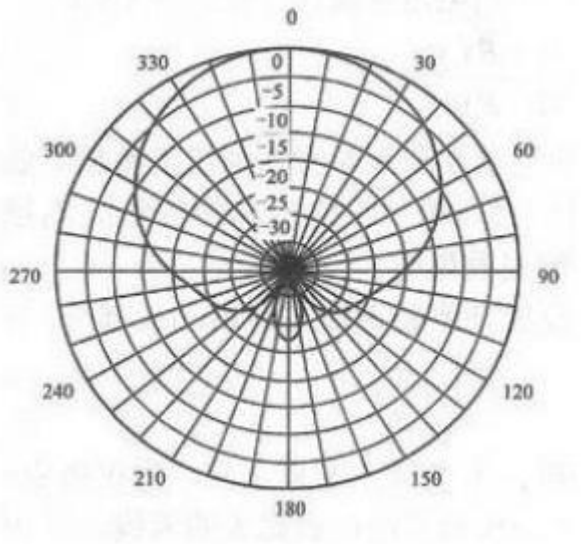


图 5.1　天线辐射方向图

计算方法：可以根据基站所在栅格和接收栅格的坐标计算出方位角，根据数据集中发射机水平方向角字段。计算出发射机和栅格的相对水平方向角。具体计算公式如下：

$$\alpha = \arctan\left(\frac{y - cell_y}{x - cell_x}\right) + \gamma.$$

$$\beta = |\alpha - \theta|$$

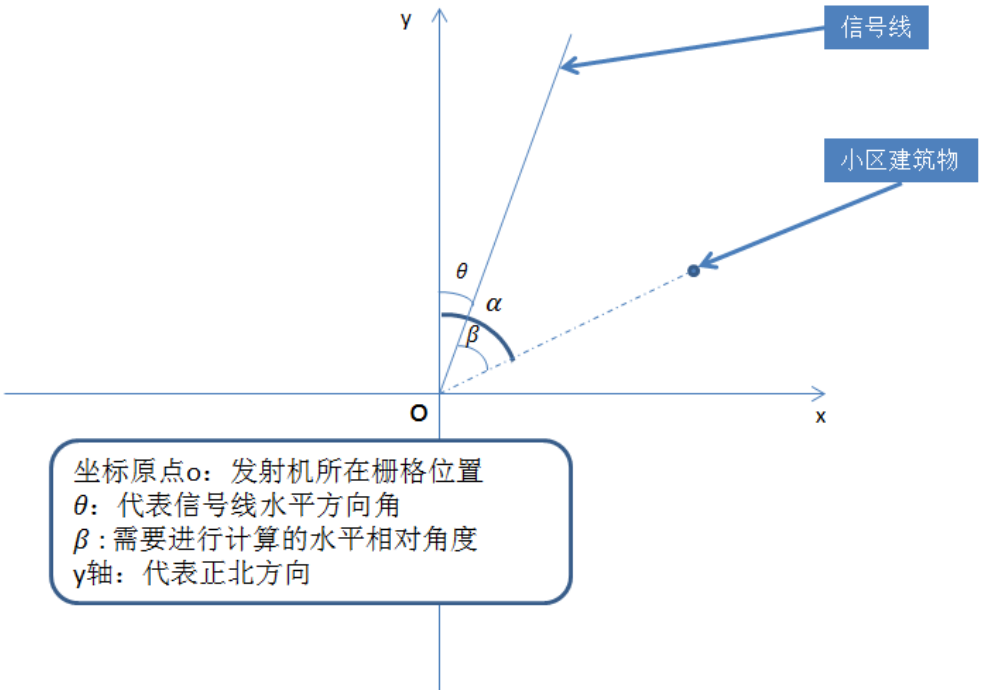


图 5.2 水平方向角图示

其中∝为接收栅格相对发射即的水平方向角，γ为方向角修正量，若接收栅格在发射机的第一象限内γ为 0，第二象限内γ为 270 °第三象限γ为 180 °，第四象限内γ为 90 °。β为发射机和接收栅格相对水平方向角即为所求 Relative_azimuth。θ为发射机水平方向角即为 Azimuth 字段。

特征九：水平相对位置（X_distance）

选择依据：反映了发射机和栅格之间的水平距离

计算方法：

$$X\_distance = |X - cell\_X|$$

其中 X 表示小区栅格水平坐标，可由数据中直接获得，cell_X 表示小区站点所在栅格水平坐标，可由数据中直接获得。

特征十：垂直相对位置（Y_distance）

选择依据：反映了发射机和栅格之间的垂直距离

计算方法：

$$Y\_distance = |Y - cell\_Y|$$

其中 Y 表示小区栅格垂直坐标，可由数据中直接获得，cell_Y 表示小区站点所在栅格垂直坐标，可由数据中直接获得。

5.2.2 重要性度量

完成特征之后通常需要选择有意义的输入到机器学习的模型中进行训练。对于不同方法得到的特征主要从特征是否发散（即此特征对于样本的区分有无作用）和特征与目标的相关性。

计算特征和目标的相关性的指标由很多如信息增益，信息增益率，基尼指数等。信息增益意味着用某一属性来进行划分所获得的集合纯度的提高，但信息增益对取值数目较多的属性有所偏好。著名的 C4.5 决策树算法不直接采用信息增益而是采用信息增益率，但信息增益率准则对可取值数目较少的属性有所偏好。CART 树则采用信息增益率来选择属性。基尼指数表示集合 D 的不确定性，基尼指数 Gini(D,A)表示经 A=a 分割后集合 D 的不确定性，基尼指数越大，样本的不确定性也就越大。本文采用基于基尼指数的随机森林进行特征重要性预测。

随机森林是于 2001 年 由 CutlerAdele 和 LeoBreiman 研究发现的一种数据挖掘方法，这是组合型的自学习与现代回归与分类结合的一种技术。随机森林既可用作分类与回归，也可用于聚类和生存分析。相比较于其它算法，它的优势在于对数据集的适应能力强，具有很好的抗噪性能和极强的拟合能力但是不会产生过拟合现象。随机森林通过自助法随机选择部分向量生长为分类树，每个树都会完整生长而不会修剪。其树的生成过程中，在随机挑出的少量变量中确定其节点的变量。该方式生成的庞大的 树便用于回归和分类，因而称作随机森林。随机森林模型有两个重要的参数：分类树的数量 k 和分割节点的随机变量m，在随机森林应用中区别是分类还是回归，取决于随机森林的每棵 cart 树是分类树还是回归树。

随机森林回归模型的优点在于：①可以处理大规模数据集，可以预测上千个解释变量的作用；②对多元线性公式不敏感，缺失数据和非平衡数据的预测结果也比较稳健；③可以给出变量的重要性估计；④训练速度快。它可以分析变量的重要性及模型的预测效果。随机森林回归算法的核心是确定随机森林决策树的数量 ntree 和分割节点的随机变量数目 mtree，mtree 代表着决策树每次分枝时选择的变量个数，在随机森林回归模型中该参数大小一般为变量个数的 1 / 3 ，同时也需要根据具体情况分析，另外为保证样本的预测次数，ntree 值不宜设定过小，否则将无法构建森林。

随机森林的一个重要特色是能够对特征的重要性进行度量， 因此，可用于特征选择。

通常采用袋外数据精度进行特征重要性度量。袋外(out of bag， OOB) 数据是指每棵决策树训练时没有被抽样到的训练样本，它们没有参与此决策树的建立，因此，可用来评估决策树的性能。其基本原理是，重新排列特征在袋外数据上的取值(即袋外数据样本间交换在该特征上的取值) ，利用重排前后的袋外数据精度差来衡量特征重要性。计算某特征的重要性具体步骤是: 首先，对于每一棵决策树预测其袋外数据精度，然后将袋外数据所有样本该特征上的取值进行重新排列后再一次对袋外数据精度进行预测，最后将每棵决策树中该特征值重新排列前后的精度差求平均，即得该特征的重要性。本研究中，为了更直观地评价特征重要性，将所有特征的重要性值的和设为 1，计算出每个特征占所有特征的重要性的比例，用于特征选择。

Sklearn 为此提供了一个很好的工具，可以很容易地测量每个特征对预测的相对重要性，它在训练后自动计算每个特征的得分，并对结果进行标准化。本节实验基于 SKlearn 框架，在数据集中给定的特征中选取 12 个特征，然后加上设计的 7 个特征共 19 个特征进行计算。

综合问题一和问题二特征，设计特征集如表 5.1 所示

表 5.1 全部设计特征

序号	变量名	含义	单位
1	Height	小区发射机相对地面的高度	m
2	Azimuth	小区发射机水平方向角	Deg
3	Electrical Downtilt	小区发射机垂直电下倾角	Deg
4	Mechanical Downtilt	小区发射机垂直机械下倾角	Deg
5	Frequency Band	小区发射机中心频率	MHz
6	RS Power	小区发射机发射功率	dBm
7	Cell Building Height	小区站点所在栅格的建筑物高度，	m
8	Cell Altitude	小区站点所在栅格海拔高度	m
9	Cell Clutter Index	小区站点所在栅格(Cell X, Cell Y)的地物类型索引	-
10	Building Height	栅格(X,Y)上的建筑物高度，	m
11	Altitude	栅格(X,Y)上的海拔高度	m
12	Clutter Index	栅格(X,Y)上的地物类型索引	-
13	Level_relative_degree	水平相对角	Deg
14	Y_distance	Y 坐标相对距离	m
15	X_diatance	X 坐标相对距离	m
16	Downtilt	天线下倾角	Deg

17	relative_height	基站和栅格的相对高度	M
18	Distance	栅格与发射机距离	M
19	Fa_height	发射机有效高度	m

综合以上讨论，随机森林在特征重要性度量方面展现出强大优势，因此本文选择随机森林方法作为特征重要性度量方法所得实验结果如图 5.3 所示，在图 5.4 的饼状图直观显示了 19 个特征各个分量所占比例。另一方面，为了验证随机森林方法的正确性，我们还采用 GradientBoostingRegressor 用于特征重要性度量作为对照。具体的实验结果如图 5.5 所示。图 5.4 中直观显示栅格与发射台的距离和，栅格和发射台的水平相对夹角对信号的影响最大，栅格与信号线相对高度，和天线水平角对栅格 RSRP 的影响次之。而在 5.5 中 GradientBoostingRegressor 方法做特征重要性度量显示，发射机与栅格的距离的重要性最高达到 0.59，其他特征的重要性远低于该特征，也就是说距离特征起到了决定性作用。但是和实际经验想去甚远。这从另一个侧面也反映出本文采用随机森林的方法进行特征重要性度量的正确性。按照特征重要的大小，本文选择了 12 特征进行后续模型的训练。具体如表 5.2 所示。

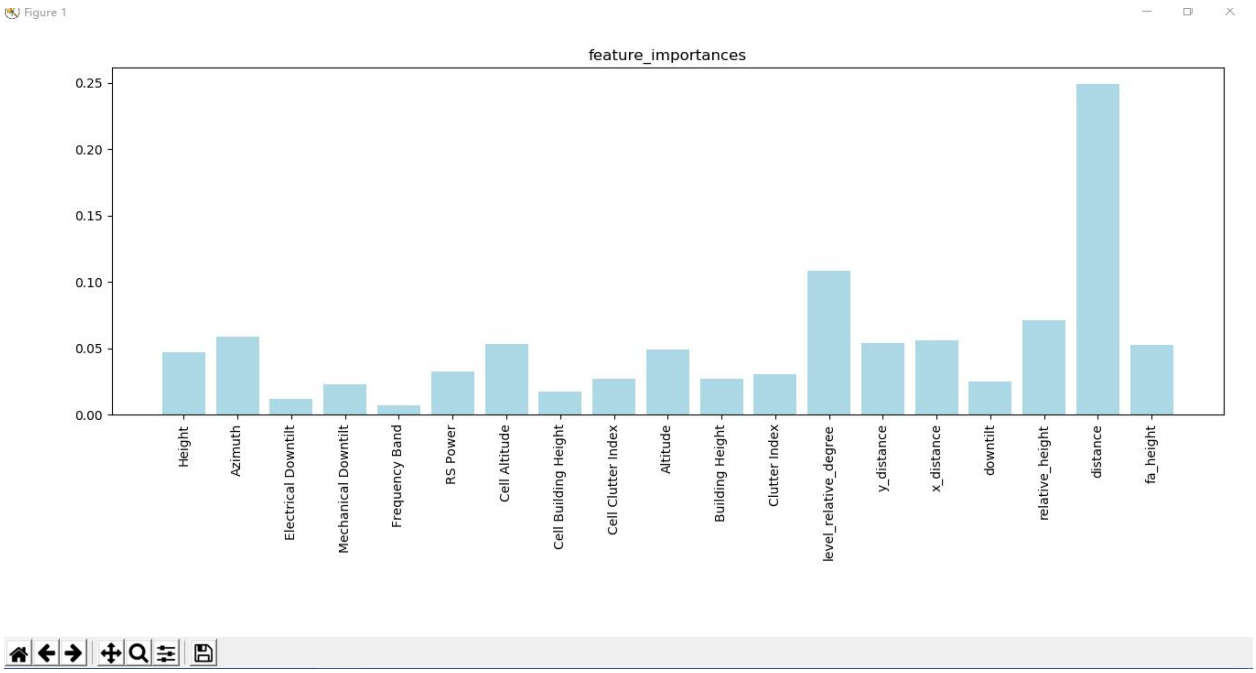


图 5.3 随机森林方法特征重要性度量结果

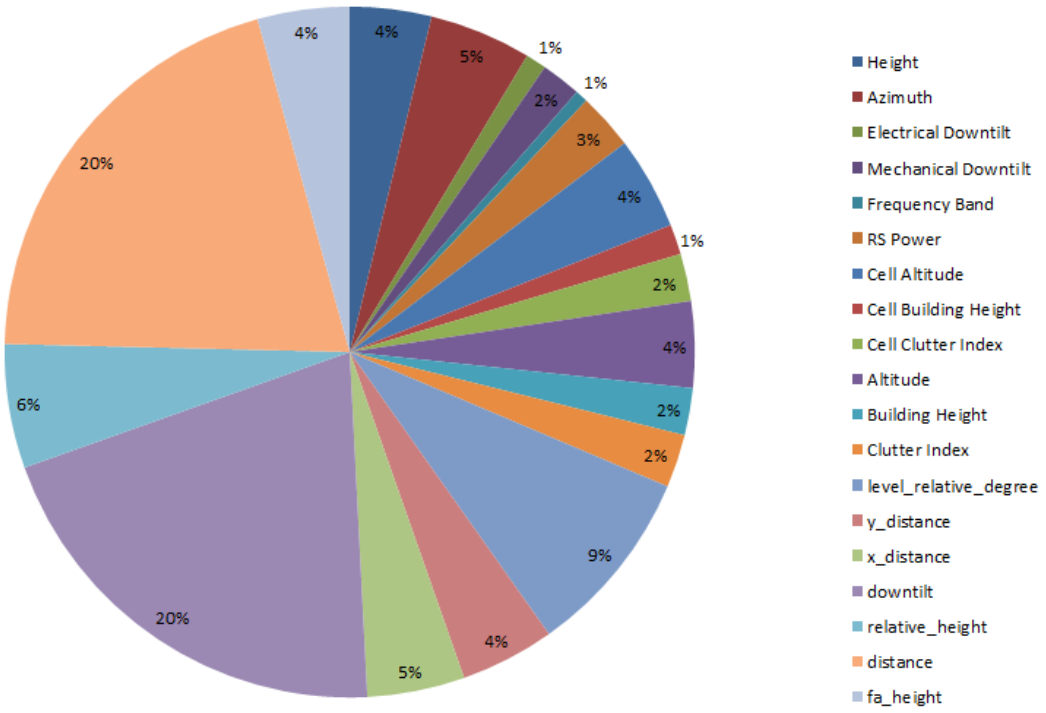


图 5.4 随机森林方法特征重要性度量结果（饼状图）

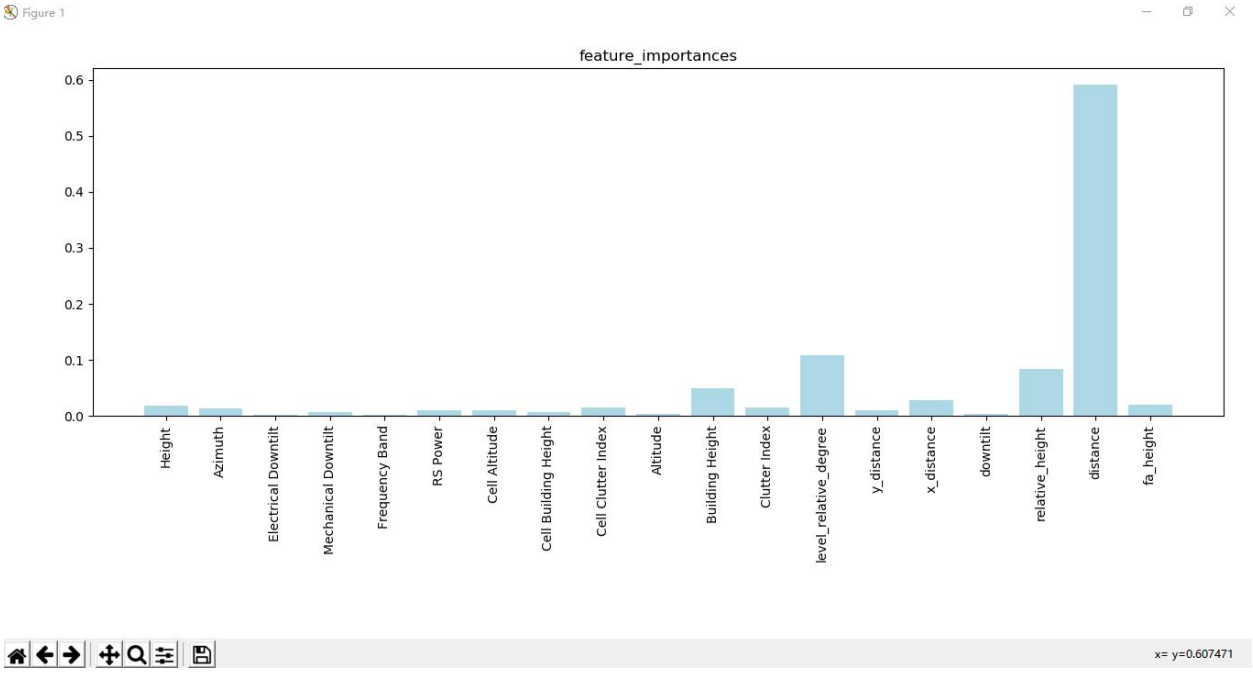


图 5.5 GradientBoostingRegressor 方法特征重要性度量结果

表 5.2 本文所选 12 特征

序号	变量名	含义	特征与目标相关性
1	Height	小区发射机相对地面的高度	0.046793
2	Azimuth	小区发射机水平方向角	0.058567
3	RS Power	小区发射机发射功率	0.032401
4	Cell Altitude	小区站点所在栅格海拔高度	0.053162
5	Altitude	栅格(X,Y)上的海拔高度	0.049391
6	Level_relative_degree	水平相对角	0.108216
7	Y_distance	Y 坐标相对距离	0.054103
8	X_diatance	X 坐标相对距离	0.055919
9	Downtilt	天线下倾角	0.024959
10	relative_height	基站和栅格的相对高度	0.071081
11	Distance	栅格与发射机距离	0.248898
12	Fa_height	发射机有效高度	0.052721

六、 问题三、分析与求解

建立数学模型并进行 RSRP 的预测。

6.1问题分析

题目中要求利用建立的模型对 RSRP 值进行预测，因此需事先训练出一个模型来表征特征和 RSRP 值之间的关系。神经网络回归是一种非参数非线性方法，结合了神经网络和回归两方面的优势，具有强大的功能，不仅可以揭示响应变量的整个条件分布，还可以非线性特征。介于此本文采用神经网络的方法进行模型构建。对所给数据集的 4000 个文件，划分前 3500 个文件作为训练集来对模型进行训练，后 500 的文件用作验证集来验证模型的正确性。

6.2 问题求解

6.2.1 BP神经网络

BP (Back Propagation)神经网络是一种多层前馈神经网络。其由输入层、中间层、输出层组成的阶层型神经网络，中间层可扩展为多层。相邻层之间各神经元进行全连接，而每层各神经元之间无连接，网络按反向传播的方式进行学习，当一对学习模式提供给网络后，各神经元获得网络的输入响应产生连接权值（Weight），隐含层神经元将传递过来的信息进行整合，通常还会在整合过程添加一个阈值，这是模仿生物学中神经元必须达到一定阈值才会触发的原理。输入层神经元数量由输入样本的维数决定，输出层神经元数量由输出样本的维数决定，隐藏层神经元合理选择。然后按减小希望输出与实际输出误差的方向，从输出层经各中间层逐层修正各连接权，回到输入层。此过程反复交替进行，直至网络的全局误差趋向给定的极小值，即完成学习的过程。BP 网络的核心是数学中的“负梯度下降”理论，即 BP 的误差调整方向总是沿着误差下降最快的方向进行。

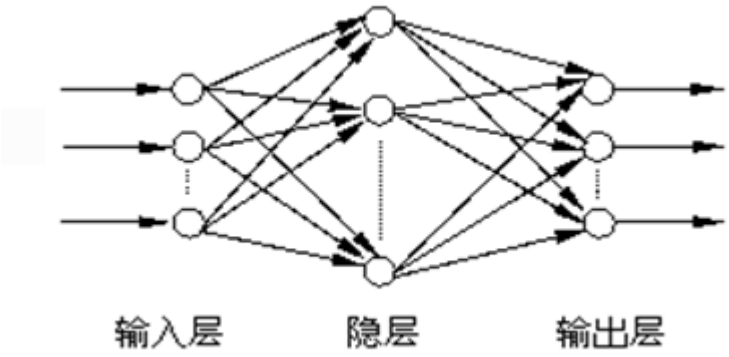


图 6.1 神经网络模型图

相关传递函数（或激励函数）：

阈值型（一般只用于简单分类的 MP 模型中）

$$f(x)=\begin{cases}0, x<0\\ 1, x>=0\end{cases}$$

线性型（一般只用于输入和输出神经元）

$$f(x)=x$$

S 型(常用于隐含层神经元)

$$f(x)=\frac{1}{1+e^{-x}} \quad 或 \quad f(x)=\frac{1-e^{-x}}{1+e^{-x}}$$

通常还会在激励函数的变量中耦合一个常数以调整激励网络的输出幅度。

输入特征在网络前面的层之间进行前向传播，很多输入层和输出层同隐层的每个神经元相连接，每个输入数据 x 将会乘上一个相应的权重 w。这些乘积之和再加上一个偏置，被送到一个 sigmoid 激活函数。对于每个隐层神经元，sigmoid 输入一个激活值 a,在网络的输出节点,计算这些激活值之和作为最后的输出。也就是说，利用神经网络来做回归将有一个输出节点，而且这个节点仅是对前面节点的激活值进行相加得到的。ŷ 就是由你所有的 x 映射得到的独立变量。由于 BP 神经网络强大的函数拟合能力本文选择神经网络来进行函数拟合来预测 RSRP 的值。

6.2.2 数据准备

将数据集中的 4000 个文件划分为 3500 个训练集，和 500 个验证集。每个文件包含某个小区内各个栅格内的参数和接收到信号强度 RSRP 值。按照第二问中所得到的特征重要性，选择 12 个重要性较大的特征，作为输入。12 个特征的重要性之和达到了 85.6%。训练神经网络利用这 12 个特征来进行 RSRP 的预测。

归一化是一种简化计算的方式，即将有量纲的表达式，经过变换，化为无量纲的表达式，成为纯量。归一化是能够加快训练网络的收敛性，因此本文对输入的 12 个特征都进行归一化操作。

6.2.3 网络初始参数

表 6.1 网络初始参数

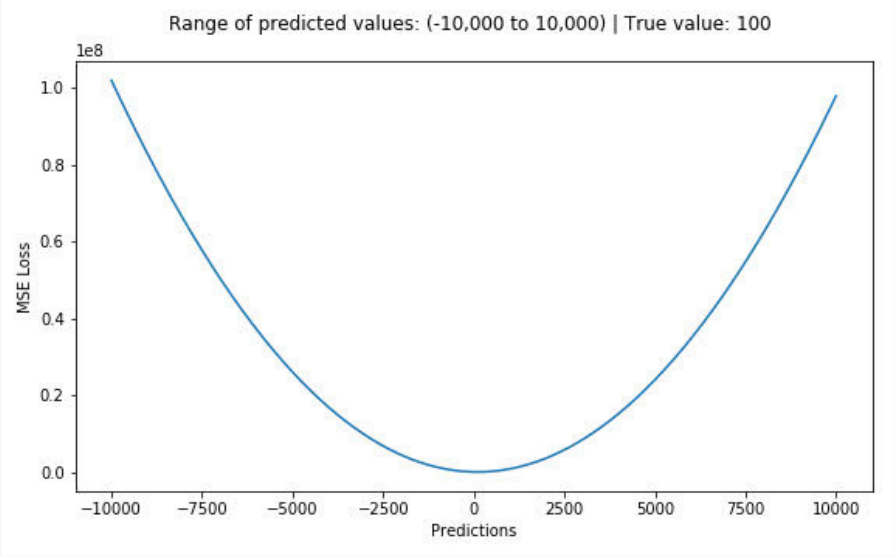
网络参数	数值
网络隐藏层数	3
输入特征数	12
学习率	0.01
激活函数	Sigmoid
隐层神经元个数	2000
优化器	AdamOptimizer

6.2.4 损失函数

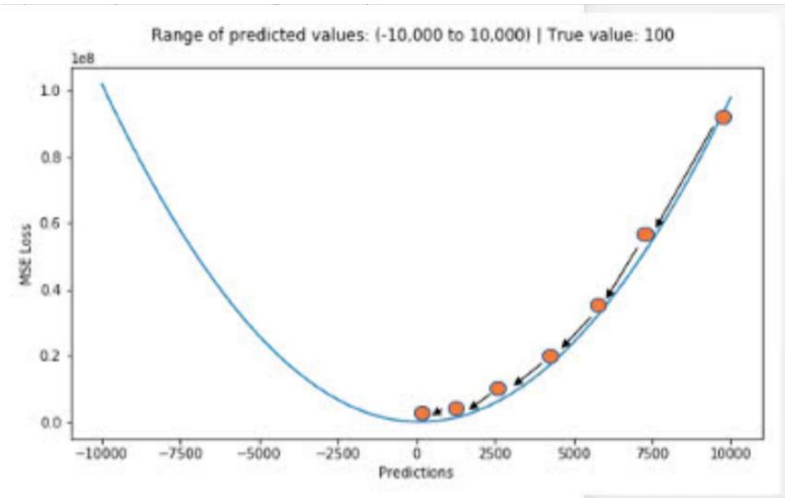
均方误差（MSE）是回归损失函数中最常用的误差，它是预测值与目标值之间差值的平方和，其公式如下所示：

$$\mathrm{MSE}(\mathrm{y},\mathrm{y}')=\frac{\sum_{i=1}^{n}(y_i-y_i')^2}{n}$$

下图是均方根误差值的曲线分布，其中最小值为预测值为目标值的位置。我们可以看到随着误差的增加损失函数增加的更为迅猛。



利用均方差更容易求解能够优化则我们会得到所有观测的均值。MSE 在极值点却有着良好的特性，及时在固定学习率下也能收敛。MSE 的梯度随着损失函数的减小而减小，这一特性使得它在最后的训练过程中能得到更精确的结果（如下图）。



综合以上均方误差容易训练，和较为精确的有点。我们选择均方误差作为本文的损失函数，具体公式为

$$\text{loss} = \frac{\sum_{I=1}^{N}(Y_{\text{true}} - \text{prediction})}{\text{N}}$$

6.2.4 模型优化

在机器学习的模型中，如果模型的参数太多，而训练样本又太少，训练出来的模型很容易产生过拟合的现象。在训练神经网络的时候经常会遇到过拟合的问题，过拟合具体表现在：模型在训练数据上损失函数较小，预测准确率较高；但是在测试数据上损失函数比较大，预测准确率较低。

过拟合是很多机器学习的通病。如果模型过拟合，那么得到的模型几乎不能用。为了解决过拟合问题，一般会采用模型集成的方法，即训练多个模型进行组合。此时，训练模型费时就成为一个很大的问题，不仅训练多个模型费时，测试多个模型也是很费时。

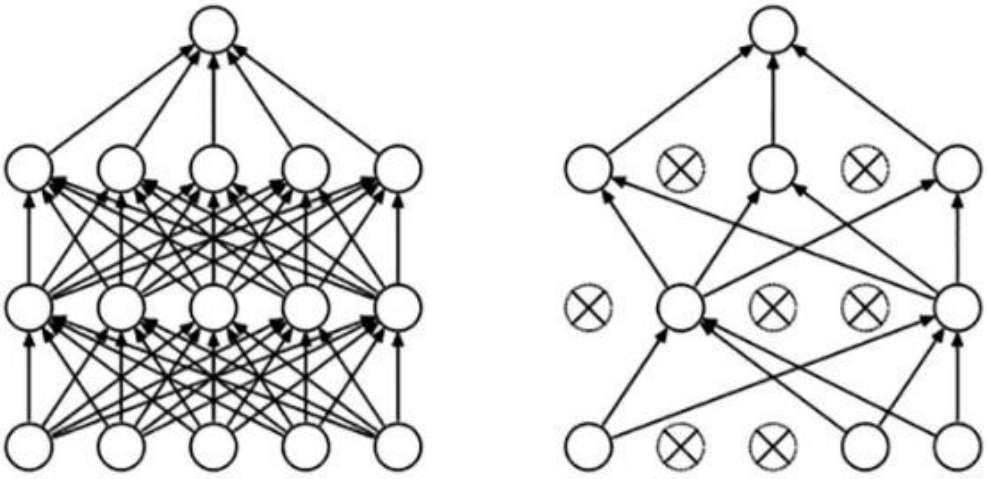
综上所述，训练深度神经网络的时候，总是会遇到两大缺点：

（1）容易过拟合

（2）费时

Dropout 可以比较有效的缓解过拟合的发生，在一定程度上达到正则化的效果。Dropout 可以作为训练深度神经网络的一种技巧。在每个训练批次中，通过忽略一部分的特征检测器（让一部分的隐层节点值为 0），可以明显地减少过拟合现象。这种方式可以减少特征检测器（隐层节点）间的相互作用，检测器相互作用是指某些检测器依赖其他检测器才能发挥作用。

Dropout 说的简单一点就是：我们在前向传播的时候，让某个神经元的激活值以一定的概率 p 停止工作，这样可以使模型泛化性更强，因为它不会太依赖某些局部的特征，如图 1 所示。



因为我们训练的时候会随机的丢弃一些神经元，但是预测的时候不能随机丢弃。如果丢弃一些神经元，这会带来结果不稳定的问题。在本文训练中擦用 dropout 值为 0.7。

6.2.5实验结果

本文实验使用 Python3.6 解释器，使用 TensorFlow1.8 版本。使用 GTX970GPU 加速训练过程。损失函数具体变化趋势如下图所示，可以直观看出模型较快收敛，loss 稳定在 0.03 左右。

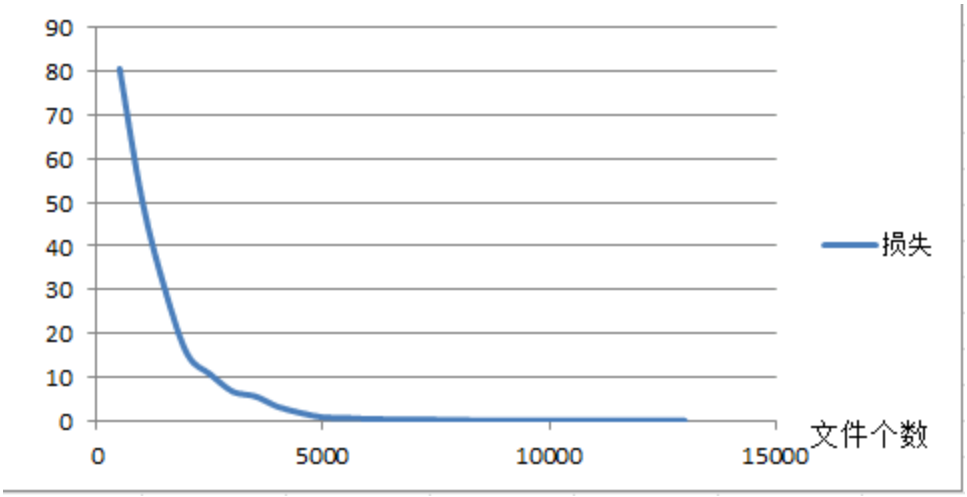


图 6.1 损失下降图

七、 模型的不足和改进

在本次建模中采用神经网络来寻找特征和 RSPR 之间映射的关系。我们结合 Cost231-Hata 模型提出 4 个特征，并运用相关领域知识，进一步提出 15 个特征。随后采用随机森林的方法对设计的 19 个特征和目标之间相关性进行衡量。最终选择 12 个作为模型训练的特征。

但是受限于专业知识的欠缺，在设计特征的过程中，难免会存在特征设计不合理的地方。同时神经网络可以直接学习某些高阶特征。受限于时间因素，在本次建模中未有神经网络作为特征提取器的使用。因此在接下来的研究中可以使用神经网络来进行特征提取。进一步把人工设计的特征和神经网络提取出的高阶特征进行特征融合然后再进行预测。

八、 参考文献

[1]包姣.基于深度神经网络的回归模型及其应用研究[D].四川:电子科技大学,2017.

[2]张雷,王琳琳,张旭东, 等.随机森林算法基本思想及其在生态学中的应用——以云南松分布模拟为例[J].生态学报,2014,34(3):650-659. DOI:10.5846/stxb201306031292.

[3]张雯,刘爱利,齐威, 等.基于随机森林的月貌面向对象分类[J].遥感信息,2018,33(1):93-98. DOI:10.3969/j.issn.1000-3177.2018.01.014.

[4]刘振波,邹娴,葛云健, 等.基于高分一号 WFV 影像的随机森林算法反演水稻 LAI[J].遥感技术与应用,2018,33(3):458-464. DOI:10.11873/j.issn.1004-0323.2018.3.0458.

[5]兰馨.基于 BP 神经网络的电动汽车负荷预测[J].电气工程,2019,7(01):7-18. DOI:10.12677/JEE.2019.71002.

[6]王蕾蕾,成金勇.基于卷积神经网络和 Softmax 的蛋白质二级结构预测[J].计算机科学与应用,2019,9(02):450-457. DOI:10.12677/CSA.2019.92051.

[7]张煌,梁朋,肖琨武, 等.基于小波去噪和神经网络的期货预测模型[J].长春理工大学学报（自然科学版）,2019,42(4):129-132,142.

[8]黄童毅,何林,郭庆, 等.基于 BP 神经网络的空调性能预测研究[J].环境技术,2019,37(4):100-103,114.

[9]胡浩儒,闫江,张永华.基于 BP 神经网络的刻蚀偏差预测模型[J].贵州大学学报（自然科学版）,2019,36(4):88-92. DOI:10.15958/j.cnki.gdxbzrb.2019.04.16.

[10]程航.基于 BP 神经网络的短期负荷预测[J].福建质量管理,2019,(16):256-258.

[11]许倩倩,许佳佳,凤鹏飞.基于 LM 神经网络的交叉口短时交通量预测[J].德州学院学报,2019,(4):40-43.

[12]罗序荣,任彬.基于 RBF 神经网络的膝关节步态预测方法研究[J].工业控制计算机,2019,32(8):90-91,94.

九、 附录

代码：

```
from __future__ import print_function
import tensorflow as tf
from sklearn.preprocessing import Normalizer
import matplotlib.pyplot as plt
import numpy as np
import math
import sklearn

### 从文件获取数据
# my_matrix = np.loadtxt(open("data2.csv", "rb"), dtype=np.float, delimiter=",", skiprows=1)
import pandas as pd
import os

# os.chdir('C:\\Users\\Administrator\\Desktop\\TensorDemo-master\\train_set')
os.chdir(

'C:\\Users\\Administrator\\Desktop\\TensorDemo-master\\train_set_process_by_column_add_xy_distance_lev
el_relative_degree')
file_chdir = os.getcwd()
filecsv_list = []
for root, dirs, files in os.walk(file_chdir):
    for file in files:
        if os.path.splitext(file)[1] == '.csv':
            filecsv_list.append(file)
FileSize = len(filecsv_list)


def get_distance(data):
    data['distance'] = np.sqrt(pow(data['Cell X'] - data['X'], 2) + pow(data["Cell Y"] - data['Y'], 2)) * 5
    # data['distance']    = pow(data['Cell X'] - data['X'], 2) + pow(data["Cell Y"] - data['Y'], 2)
    return data


def get_relative_height(data):
    # data['distance']    = math.sqrt(pow(data['Cell X'] - data['X'], 2) + pow(data["Cell Y"] - data['Y'], 2)) * 5
    data['relative_height'] = np.fabs(data['Cell Altitude'] + data["Height"] - (
                    math.tan(math.radians(data['Electrical Downtilt'] + data['Mechanical Downtilt'])) *
data['distance'] +
                    data['Altitude']))
    return data
```

```
def get_x_distance(data):
    data['x_distance'] = np.sqrt(pow(data['Cell X'] - data['X'], 2)) * 5
    return data


def get_y_distance(data):
    data['y_distance'] = np.sqrt(pow(data["Cell Y"] - data['Y'], 2)) * 5
    return data


def count_level_relative_degree(cellx, celly, x, y):
    degree = math.degrees(math.atan((celly - y) / (cellx - x)))
    if degree < 0:
        degree = 180 - math.fabs(degree)
    return degree


def get_level_relative_degree(data):
    # 计算水平相对角度
    # 先判断是否为特殊的点   在 x 轴上或者在 y 轴上
    cellx = data['Cell X']
    celly = data['Cell Y']
    x = data['X']
    y = data['Y']
    Azimuth = data['Azimuth']
    jiaodu = 0
    if cellx == x:
        if celly > 0:
            jiaodu = math.fabs(90 - Azimuth)
        else:
            jiaodu = math.fabs(Azimuth - 270)
    if celly == y:
        if x > 0:
            jiaodu = math.fabs(90 - Azimuth)
        else:
            jiaodu = math.fabs(Azimuth - 270)
    # 在象限内
    if cellx < x and y > celly:    # 1
        jiaodu = 90 - count_level_relative_degree(cellx, celly, x, y) - Azimuth
    elif cellx > x and y > celly:    # 2
        jiaodu = 360 - count_level_relative_degree(cellx, celly, x, y) + 90 - Azimuth
    elif cellx > x and y < celly:    # 3
        jiaodu = 90 - count_level_relative_degree(cellx, celly, x, y) + 180 - Azimuth
```

```
    elif cellx < x and y < celly:    # 4
        jiaodu = 180 - count_level_relative_degree(cellx, celly, x, y) + 90 - Azimuth
    jiaodu = math.fabs(jiaodu)
    if jiaodu <= 180:
        data['level_relative_degree'] = math.fabs(jiaodu)
    else:
        data['level_relative_degree'] = math.fabs(jiaodu - 180)
    return data


def getFileFuture(FileName):
    data = pd.DataFrame()
    data = pd.read_csv(FileName, header=0, sep=None, encoding='gb18030')
    one_data = data['Altitude']
    data.insert(17, 'fa_height', one_data)
    data.insert(17, 'distance', one_data)
    data.insert(17, 'relative_height', one_data)
    data.insert(17, 'downtilt', one_data)
    data.insert(17, 'x_distance', one_data)
    data.insert(17, 'y_distance', one_data)
    data.insert(17, 'level_relative_degree', one_data)
    # 多列一同计算
    data['downtilt'] = data.apply(lambda x: x['Electrical Downtilt'] + x['Mechanical Downtilt'], axis=1)
    data['fa_height'] = data.apply(
        lambda x: x['Height'] + x['Cell Building Height'] + x['Cell Altitude'] - x['Altitude'], axis=1)

    data = data.apply(get_distance, axis=1)    #
    data = data.apply(get_relative_height, axis=1)    #

    # x y distance
    data = data.apply(get_x_distance, axis=1)
    data = data.apply(get_y_distance, axis=1)

    data = data.apply(get_level_relative_degree, axis=1)
    return data


def getFileData(batch):
    x_input_data = []
    y_input_data = []

    num = 10 * batch
    for file_content in filecsv_list:
        print("第" + str(num) + "个文件！")
```

```
            pb_data = getFileFuture(file_content)
            input_x_data = np.array(pb_data.get_values()[:, 0:24], dtype=np.float32)
            input_y_data = np.array(pb_data.get_values()[:, 24:25], dtype=np.float32)
            x_input_data.extend(input_x_data)
            y_input_data.extend(input_y_data)
            num = num + 1
            if num >= 10 * batch + 10:
                break

    x_input_data = np.array(x_input_data)
    y_input_data = np.array(y_input_data)
    x_input_data = np.delete(x_input_data, [0, 1, 2, 5, 6, 7, 10, 11, 12, 13, 15, 16, 20], axis=1)
    tezheng = x_input_data
    sale = y_input_data
    X_train = Normalizer().fit_transform(tezheng)   ## 多维特征
    # 假设最大值为-70 最小值为-110   归一化操作
    MaxValue = -70
    MinValue = -110
    rang = MaxValue - MinValue
    sale = np.array(sale)
    sale = (sale - MinValue) / rang
    y_train = sale.reshape((-1, 1))   ## 结果，从一维转为二维
    return X_train, y_train


def get_Batch(X_train, y_train, i):
    return X_train[(i - 1) * 1000:i * 1000, :], y_train[(i - 1) * 1000:i * 1000]


#### 开始进行图的构建

## 特征与结果的替代符，声明类型，维度 ，name 是用来生成模型之后，使用模型的时候调用用的
inputX = tf.placeholder(shape=[None, 11], dtype=tf.float32, name="myInput")
y_true = tf.placeholder(shape=[None, 1], dtype=tf.float32, name="y_true")

### 第一层，一个隐藏层 开始
## shape 的第一维就是特征的数量，第二维是给下一层的输出个数， 底下的矩阵相乘实现的该转
Weights1 = tf.Variable(tf.random_normal(shape=[11, 500]), name="weights1")   ## 权重
biases1 = tf.Variable(tf.zeros(shape=[1, 500]) + 0.1, name="biases1")   ## 偏置

## matmul 矩阵相乘，nn.dropout 丢弃部分不靠谱数据
Wx_plus_b1 = tf.matmul(inputX, Weights1)
Wx_plus_b1 = tf.add(Wx_plus_b1, biases1)
```

```
Wx_plus_b1 = tf.nn.dropout(Wx_plus_b1, keep_prob=1)

## 将结果曲线化，通常说非线性化
l1 = tf.nn.sigmoid(Wx_plus_b1, name="l1")

### 第一层结束

### 第二层开始，即输出层
## 上一层的 10，转为 1，即输出销售量
Weights2 = tf.Variable(tf.random_normal(shape=[500, 1]), name="weights2")  ## 权重
biases2 = tf.Variable(tf.zeros(shape=[1, 1]) + 0.1, name="biases2")  ## 偏置

## matmul 矩阵相乘 ,l1 为上一层的结果
Wx_plus_b2 = tf.matmul(l1, Weights2)
prediction = tf.add(Wx_plus_b2, biases2, name="myOutput")  ## pred 用于之后使用 model 时进行恢复

## 这里使用的这个方法还做了一个第一维结果行差别的求和，reduction_indices=1，实际这个例子每行只有一个结果,使用 loss = tf.reduce_sum(tf.square(y_true - prediction)) 即可
loss = tf.reduce_mean(tf.reduce_sum(tf.square(y_true - prediction), reduction_indices=[1]))

## 训练的 operator，AdamOptimizer 反正说是最好的训练器，训练速率 0.01
train_op = tf.train.AdamOptimizer(learning_rate=0.001).minimize(loss)

batchsize = 1000


def TrueOrFalse(x):
    return x < -103


def compute_pcrr(y_pred, y_true):
    t = -103
    y_pred = np.array(y_pred).reshape(-1)
    y_true = np.array(y_true).reshape(-1)
    y_pred = np.array(y_pred) * (110 - 70) + (-110)
    y_true = np.array(y_true) * (110 - 70) + (-110)
    num = 0
    tp = 0
    fp = 0
    fn = 0
    for item in y_pred:
        if y_true[num] < t and y_pred[num] < t:
            tp = tp + 1
        elif y_true[num] >= t and y_pred[num] < t:
```

```
                fp = fp + 1
            elif y_true[num] < t and y_pred[num] >= t:
                fn = fn + 1
            num = num + 1
        if (tp + fp) == 0:
            return -1
        if (tp + fn) == 0:
            return -1
        precision = tp / (tp + fp)
        recall = tp / (tp + fn)
        if (precision + recall) == 0:
            return -1
        pcrr = 2 * (precision * recall) / (precision + recall)
        return pcrr


### 开始执行
with tf.Session() as sess:
    saver = tf.train.Saver(tf.global_variables(), max_to_keep=15)  # 初始化 saver，用于保存模型

    init = tf.global_variables_initializer()  # 初始化全部变量
    sess.run(init)  # 初始化全部变量

    ## 要给模型进行训练的数据，只有 placeholder 类型的需要传进去数据

    for j in range(3):
        print("EPOCH", str(j))
        for i in range(int(FileSize / 1000)):
            X_train, y_train = getFileData(i)
            for k in range(int(len(X_train) / 1000)):
                new_x_batch, new_y_batch = get_Batch(X_train, y_train, k)
                feed_dict_train = {inputX: new_x_batch, y_true: new_y_batch}
                _loss, _ = sess.run([loss, train_op],
                                    feed_dict=feed_dict_train)  # 训练，注：loss 没有训练，只是走到 loss，返回值，走到 train_op 才会训练
                feed_dict_trains = {inputX: new_x_batch}
                print("acc", compute_pcrr(sess.run([prediction], feed_dict=feed_dict_trains), new_y_batch))
                print("Epoch%d:步数:%d:loss:%.5f" % (j, k, _loss))

    tf.saved_model.simple_save(sess,"./model", inputs={"myInput": inputX}, outputs={"myOutput": prediction})
```