**机器学习作业1-PUBG排名预测**

**郭远帆，唐荣俊，罗乙然**

目录

[任务介绍 1](#_Toc528131802)

[任务说明 1](#_Toc528131803)

[数据集的使用 1](#_Toc528131804)

[任务目标与规划 2](#_Toc528131805)

[实现方案 2](#_Toc528131806)

[编程环境说明 2](#_Toc528131807)

[基本特征工程（线性模型） 2](#_Toc528131808)

[模型评估方法 3](#_Toc528131809)

[实验结果 3](#_Toc528131810)

[线性模型 4](#_Toc528131811)

[SVR支持向量机（回归） 4](#_Toc528131812)

[决策树模型 4](#_Toc528131813)

[决策树定义 4](#_Toc528131814)

[模型使用及结果 4](#_Toc528131815)

[结论与后续 5](#_Toc528131816)

[附录 5](#_Toc528131817)

# 任务介绍

## 任务说明

本次任务来源于Kaggle数据科学竞赛网站提供的比赛项目:**PUBG Finish Placement Prediction （绝地求生:大逃杀玩家排名预测）**

比赛说明网站: <https://www.kaggle.com/c/pubg-finish-placement-prediction> 网站提供了该任务的详细说明，输出要求以及数据（包括训练集和测试集）

这是一个非常有趣的项目，竞赛网站与PUBG制作商蓝洞公司合作，获取了超过65，000场游戏的玩家数据，要求竞赛参加者根据这些数据预测玩家在比赛中的排名。

## 数据集的使用

本项目原数据集相当庞大，训练集与测试集数据量一共有超过650万条数据，每一条数据拥有25个特征。在实际使用中,对于某些模型(如SVR支持向量机)我们需要对数据进行采样来测试模型的性能，以缩短训练时间，从而在短时间内了解多个模型的特点以及调参方法。

由于数据集庞大，提交作业时不一并提交，可于比赛说明网站上下载。

## 任务目标与规划

我们小组以此项目作为机器学习第一次作业，该问题属于回归问题。而由于该项目本身是数据科学竞赛项目，其中必然存在数据分析与特征提取的过程（但并非主要的），我们将任务划分为几个阶段:

第一阶段: 数据缺失处理，进行简单的数据可视化分析

第二阶段: 参考竞赛网站其它竞赛参加者提供的特征工程方案，提取简单特征

第三阶段: 构建不同的机器学习模型训练，并且进行模型的评估

第四阶段: 采用集成学习技术将数个模型整合，构成最终模型并进行分析比较

其中第三第四阶段为主要的工作。

通过本次任务，我们将学习到:

1. 利用pandas对数据集进行预处理
2. 利用seaborn和matplotlib对数据集进行简单分析
3. 如何调用sklearn中的机器学习模型，并理解其实现原理
4. 如何使用集成学习技术提升学习器的性能
5. 自己实现简单的模型并与sklearn库进行比较

# 实现方案

## 编程环境说明

编程语言: Python3.6.4

使用的工具包:

Numpy: 线性代数库，支持强大的矩阵运算

pandas: 数据处理库

Matplotlib: Matlab风格的画图工具

Seaborn: 统计画图工具

Sklearn: 最大的机器学习库，包含了一些常用的机器学习模型。

xgboost: 梯度提升库（集成学习方法）

## 基本特征工程（线性模型）

线性回归模型以模型简单、速度快而在工业界广泛应用。但是由于模型过于简单，最小二乘法的线性简单模型需要更多的特征提取工作，否则非常容易欠拟合而得到不满意的结果。在对训练集进行数据预处理以及相关性分析后，我们得到下图:

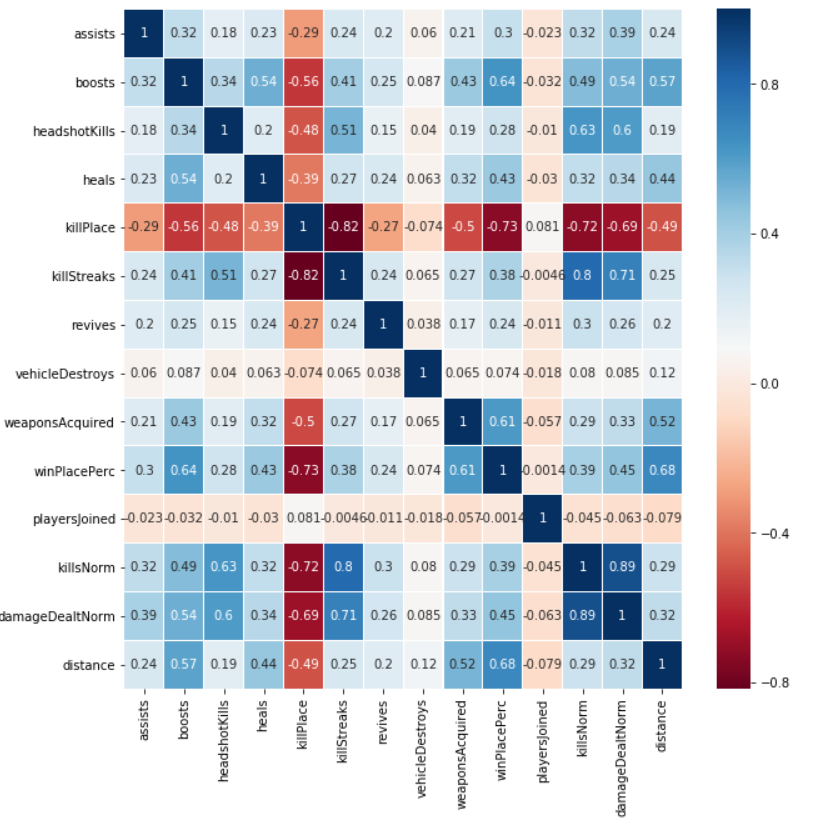


Figure1-特征相关性图

其中winPlacePerc为目标值(y)，从图中可以看到经过预处理后的特征的相关性，这些特征即线性模型中最终保留的特征。

## 模型评估方法

# 实验结果

## 线性模型

## 模型说明

sklearn库提供了多种线性模型，在本次实验中使用到其中三种: LinearRegression, Lasso, Ridge。

LinearRegression的原理十分简单，设**X**为输入数据集，**y**为预测结果，**w**为各特征权重，其优化目标为:

在sklearn库中，其训练复杂度为, 其中n为样本数量，p为特征量。根据[1]中对sklearn的 LinearRegression()源码解读，解线性回归方程参数时，首先判断训练集**X**是不是稀疏矩阵，如是，就用Golub & Kahan双对角线化过程方法来求解；否则就调用C库LAPACK中的用基于分治法的奇异值分解来求解。

sklearn中的Lasso和Ridge是引入了L1和L2正则化项的线性回归模型，即优化目标分别为:

和LinearRegression一样，在sklearn中采用双对角线化 或者 奇异值分解来进行优化，其计算复杂度也为

由于sklearn中的线性模型没有使用梯度下降法实现，为了实践课内学习到的理论，自己实现梯度下降法(GDLinear.py)并在下一部分与sklearn实现的模型进行比较。

## 实验结果

在完成一轮特征工程后，数据集中留下来的特征拥有较好的独立性，并且对于简单模型来说降低了数据的复杂度，因此也降低了拟合的难度。我们采用不同的线性模型进行拟合并观察实验结果，并尝试着对实验结果进行解释。

调用sklearn 中的几个线性模型进行训练以及预测（采用全局数据）并对结果进行比较:

**实验结果**:

|  |  |  |  |
| --- | --- | --- | --- |
| 模型: | 数据集 | 测试集比例 | M（平均绝对误差） |
| LinearRegression | train.csv(49M) | 0.25 | 0.095 |
| Lasso | train.csv(49M) | 0.25 | 0.095 |
| Ridge | Train.csv(49M) | 0.25 | 0.095 |

从表中可得，线性回归模型平均将引入9.5%的误差。正则化项对模型的影响不大

鉴于该模型的低成本性，9.5%误差的结果可以接受。

接下来考虑自己实现的梯度下降法的线性回归模型:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 学习速率( |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

## SVR支持向量机（回归）

## 决策树模型

## 决策树定义

决策树(Decision Tree）是在已知各种情况发生概率的基础上，通过构成决策树来求取净现值的期望值大于等于零的概率，评价项目风险，判断其可行性的决策分析方法，是直观运用概率分析的一种图解法。由于这种决策分支画成图形很像一棵树的枝干，故称决策树。[机器学习](https://baike.baidu.com/item/%E6%9C%BA%E5%99%A8%E5%AD%A6%E4%B9%A0" \t "_blank)中，决策树是一个预测模型，代表的是对象属性与对象值之间的一种映射关系。树中每个节点表示某个对象，而每个分叉路径则代表的某个可能的属性值，而每个叶结点则对应从根节点到该叶节点所经历的路径所表示的对象的值。

## 模型使用及结果

决策树模型在机器学习中既可以用作分类算法，也可以用作回归算法。这里我们采用它的回归形式来对结果做出预测，并使用scikit-learn中的DecisionTreeRegressor库进行回归预测。该库使用的是CART算法，是最优的决策树算法。

通过查阅资料，我们发现决策树回归模型有很多个参数需要调节。一些参数，如criterion、splitter等，采用默认的参数即可。而一些参数则需要根据我们数据的实际情况进行调节。由于我们的数据量大（共几百万条），学习的时间成本是必须考虑的问题。而且我们经过了数据预处理，忽略掉了一些显然无用的特征，例如groupId、winPoints，也按一定规则合并和归一化了一些特征，例如kills，实际上的特征量已经减少了很多，故而无需考虑最大特征数。所以经过考虑，我们最终确定了决策树最大深度max\_depth、内部节点再划分所需最小样本数min\_samples\_split、叶子节点最少样本数min\_samples\_leaf等三个参数进行调节。并且通过查阅资料，最终我们确定了它们的值分别是15、10、10。

模型在测试集上的表现为M=0.06420，M为平均绝对值误差。误差较小，可以尝试继续改进算法及参数降低误差。

# 结论与后续

# 附录

## 参考文献

[1]<https://blog.csdn.net/ybdesire/article/details/67701289> sklearn中的LinearRegression关键源码解读

[2]<http://scikit-learn.org/stable/user_guide.html> sklearn库官方文档

[3]<https://en.wikipedia.org/wiki/Singular_value_decomposition#Solving_homogeneous_linear_equations> 维基百科-奇异值分解