课程设计报告

**基于往年数据使用随机森林模型对未来天气状况进行预测**

Random forest model is used to predict future weather conditions based on previous data

|  |  |  |  |
| --- | --- | --- | --- |
| 课程名称 | 大数据概论 | 学院 | 软件学院 软件工程专业 |
| 小组成员 | 贺思超、韩熔、解世超、陈杰 | | |

**北京交通大学**

2022年6月

项目成员姓名：贺思超（20301037）、韩熔（20301036）、解世超（20301044）、陈杰（20301033）

Project Members: He Sichao (20301037), Han Rong (20301036), Xie Shichao (20301044), Chen Jie (20301033)

摘要：基于天气预测的广泛应用和重要意义，以及其与大数据极强的关联性，本小组选择气温预测进行课程研究，旨在运用大数据分析与挖掘的理论与技术通过随机森林模型解决气温预测的相关问题。

Abstract: Based on the extensive application and significance of weather prediction, as well as its strong correlation with big data, our group chose temperature prediction for course research, aiming to solve the problems related to temperature prediction by using the theory and technology of big data analysis and mining through random forest model.

# 研究内容

由于大气运动的不确定性等因素，天气预测需要收集大量的数据用以分析处理，大数据分析与挖掘的相关理论与技术对天气预测有着重要的作用。新的全国平均气温序列与以往的研究结果比较给出了更高的增温趋势估计值,这主要与采用新的月平均气温统计方法改善了原序列的均一性有关。[1]

由于可能会持续的增温，所以就近选取往年数据作为数据集，再通过对预设时间段的天气气温分析，由最高气温、最低气温、降雨量、气压、风力等特征值的分析进行从当前日期开始的后七天时间段的天气情况进行预测。本实验拟采用Python语言，从www.meteomanz.com网页爬取气温数据并存储为CSV文件，使用panda读取CSV文件并进行数据预处理，并采用Sklearn框架进行机器学习处理。最后使用随机数森林模型进行建模预测。

# 二、设计方案

## 基本构架（整体设计思路）：

**数据的获取**

训练数据来源于[http://www.meteomanz.com/](http://www.meteomanz.com/" \t "dlt)

**数据的获得与处理**

http://www.meteomanz.com/sy2?l=1&cou=2250&ind=59287&d1=02&m1=02&y1=2019&d2=13&m2=12&y2=2020指从2019/2/2到2020/12/13的北京的每日天气数据，是我们要取到的数据的地址。按F12打开开发者工具，我们可以发现tbody里面的tr里面的每个td里是数据。

**数据的预处理**

使用ProcessData()进行数据的预处理，返回训练数据集以及训练数据集的验证集，和预测数据集。

**模型建立与训练：数据的读取与模型训练**

构建随机树森林模型训练模型

预测模型，用上个星期的数据

用MAE评估

保存模型到本地

返回MAE

## 回归随机森林模型

### 1.原理

（1）随机森林是bagging方法的一种具体实现。它会训练多棵决策树，然后将这些结果融合在一起就是最终的结果。随机森林可以用于分裂，也可以用于回归。主要在于决策树类型的选取，根据具体的任务选择具体类别的决策树。

对于分类问题，一个测试样本会送到每一颗决策树中进行预测，然后投票，得票最多的类为最终的分类结果；

对于回归问题，随机森林的预测结果是所有决策树输出的均值。

**我们这里利用随机森林进行天气状态的简单回归预测，满足大部分科研需求。**

（2）**关于随机森林回归器：**

随机森林模型是许多决策树的集合，其中决策树被称为弱学习器。它可以应用于分类和回归问题。使用随机森林的回归过程可以通过以下步骤完成：

**数据拆分**：该过程通过特征拆分，每一行负责创建决策树。

**决策**：每棵树都根据数据做出自己的决策。

**决策聚合**：在这一步中，树的平均值预测成为最终结果。

这种来自树的平均决策使得随机森林回归比任何其他算法都强。

（3）建立多个决策树并将他们融合起来得到一个更加准确和稳定的模型是bagging思想和随机选择特征的结合。随机森林构造了多个决策树，当需要对某个样本进行预测时.统计森林中的每棵树对该样本的预测结果，然后通过投票法从这些预测结果中选出最后的结果。[2]

随机体现在两个方面，一个是随机取特征，另一个是随机取样本，让森林中的每棵树既有相似性又有差异性。森林中每棵树按照奶下方式生长：

1.如果过训练样本中有N个样本.那么从这N个样本中有放回的抽样N次，将得到的样本用于建树。

2.设M为输入样本的特征数，对于每个节点分裂时，我们先从这M个特征中选择m(m<<M)个特征，然后再在这m个特征中选择最佳的分裂点进行分裂。

3.每棵树都尽可能的生长，没有剪枝m的值越大，上述1中的相关性越高，2中的分类能力也越强，所以m在随机森林中是个非常重要的参数。

### 2．特点

1、优点

随机森林善于处理高维数据，特征遗失数据，和不平衡数据

（1）训练可以并行化，速度快

（2）对高维数据集的处理能力强，它可以处理成千上万的输入变量，并确定最重要的变量，因此被认为是一个不错的降维方法。既能处理离散性数据，也能处理连续型数据，数据集无需规范化；

（3）在训练集缺失数据时依旧能保持较好的精度（原因：随机森林随机选取样本和特征；随机森林可以继承决策树对缺失数据的处理方式）

（4）泛化能力强，因为随机

（5）在数据集上表现良好，两个随机性的引入，使得随机森林不容易陷入过拟合，但是对于小数据集还是有可能过拟合，所以还是要注意

（6）两个随机性的引入，使得随机森林具有很好的抗噪能力

（7）训练速度快，可以得到变量重要性排序

（8）在训练过程中，能够检测到feature间的互影响

2、缺点

（1）由于随机森林的两个随机性，导致运行结果不稳定。

（2）对特征数比较少的数据，它的随机性不好发挥，效果不太好,对于小数据集和低维的数据效果可能不是很好。

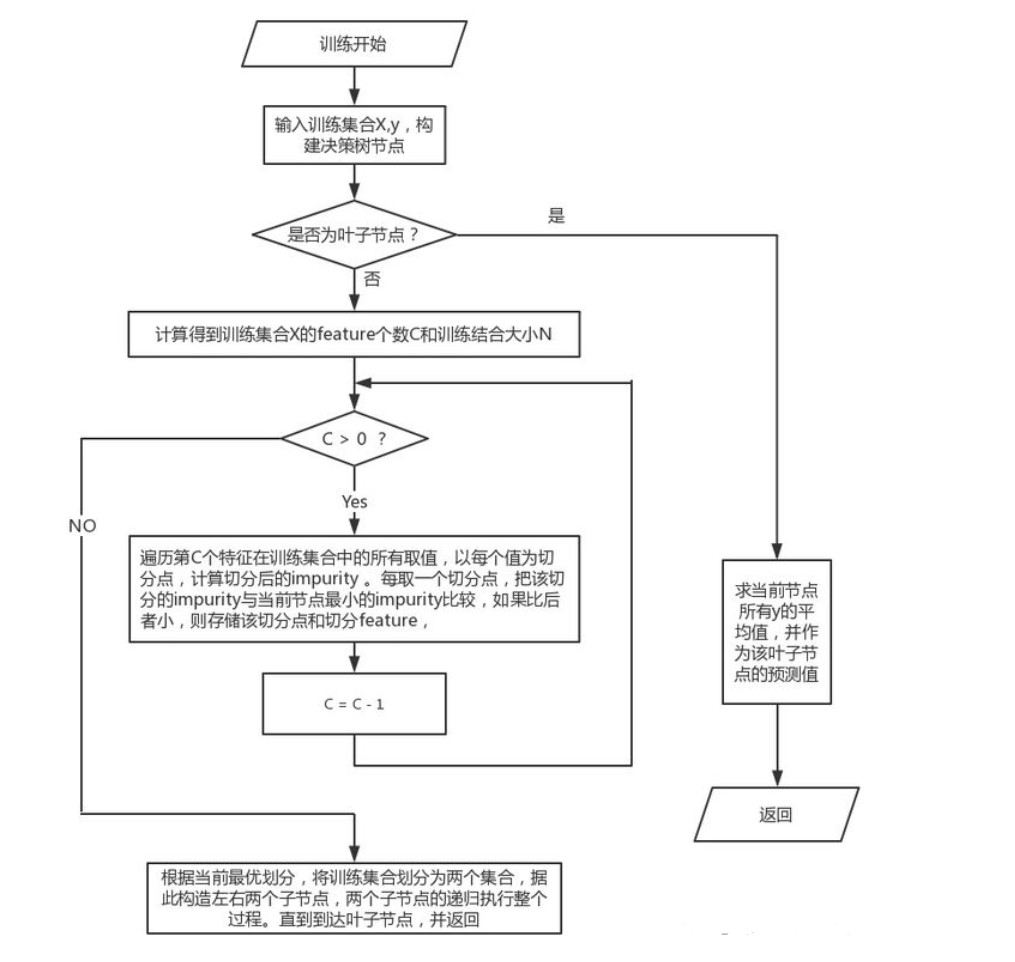
（3）整个模型为黑盒，没有很强的解释性。

## 技术难点与解决方案：

技术难点：用随机森林实现简单回归预测

对于回归问题，随机森林的预测结果是所有决策树输出的均值。

1.训练过程

下图为n个回归决策树的训练过程  


2.模型预测

RFR的预测结果是由内部所有二叉决策树的预测结果取平均值得到的。二叉决策树的预测过程主要分为以下步骤：

（1）针对某一输入样本，从二叉决策树的根节点起，判断当前节点是否为叶子节点，如果是则返回叶子节点的预测值(即当前叶子中样本目标变量的平均值），如果不是则进入下一步；

（2）根据当前节点的切分变量的和切分值，将样本中对应变量的值与节点的切分值对比。如果样本变量值小于等于当前节点切分值，则访问当前节点的左子节点；如果样本变量值大于当前节点切分值，则访问当前节点的右子节点；

（3）循环步骤（2），直到访问到叶子节点，并返回叶子节点的预测值。

## 总体流程（代码架构）：

使用GetData.py编写相关的爬虫程序爬取数据。

通过Write.py进行爬取数据的分析、清洗；之后写到创建好的csv文件中进行下一步的数据预处理。

之后通过ProcessData.py调用Write中的方法write进行数据的读取与csv文件的写入，并通过这些文件进行数据集的处理与划分。

最后模型构建在GetModel.py中，使用随机森林模型，填入处理好的数据集，进行模型的训练与保存。

之后通过main.py进行已经训练好的模型的调用与预测。

# 三、设计实现

**1.获取数据**

# 功能: 爬取数据

import urllib3

class GetData:

url = ""

headers = ""

def \_\_init\_\_(self, url, header=""):

"""

:param url: 获取的网址

:param header: 请求头，默认已内置

"""

self.url = url

if header == "":

self.headers = {

'Connection': 'Keep-Alive',

'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,'

'\*/\*;q=0.8,application/signed-exchange;v=b3;q=0.9',

'Accept-Language': 'zh-CN,zh;q=0.9,en;q=0.8',

'Accept-Encoding': 'gzip, deflate',

'User-Agent': 'Mozilla/5.0 (Linux; Android 6.0; Nexus 5 Build/MRA58N) AppleWebKit/537.36 (KHTML, '

'like Gecko) Chrome/87.0.4280.66 Mobile Safari/537.36 ',

'Host': 'www.meteomanz.com'

}

else:

self.headers = header

def Get(self):

"""

:return: 网址对应的网页内容

"""

http = urllib3.PoolManager()

return http.request('GET', self.url, headers=self.headers).data

再通过write.py将数据写入csv文件

**2.数据预处理**

# 功能: 数据预处理

def ProcessData():

"""

:return:

[X\_train X训练数据集,

X\_valid X训练数据集的验证集,

y\_train Y训练数据集,

y\_valid Y训练数据集的验证集,

imputed\_X\_test 预测数据集]

"""

# 用近几年的数据做训练集

# 如 [1,1], [20, 0]就是用2019年的今天的20天前到2019年的今天数据做训练集

# 写入csv

write([1, 1], [15, 0], "weather\_train\_train.csv")

write([1, 1], [0, 15], "weather\_train\_valid.csv")

write([0, 0], [15, 0], "weather\_test.csv")

X\_test = pd.read\_csv("weather\_test.csv", index\_col="Time", parse\_dates=True)

# 读取测试集和验证集

X = pd.read\_csv("weather\_train\_train.csv", index\_col="Time", parse\_dates=True)

y = pd.read\_csv("weather\_train\_valid.csv", index\_col="Time", parse\_dates=True)

# 填充缺少的数值用方差，不清楚效果如何

my\_imputer = SimpleImputer()

X\_train, X\_valid, y\_train, y\_valid = train\_test\_split(X, y, train\_size=0.8, test\_size=0.2, random\_state=0)

imputed\_X\_train = pd.DataFrame(my\_imputer.fit\_transform(X\_train))

imputed\_X\_valid = pd.DataFrame(my\_imputer.transform(X\_valid))

imputed\_X\_train.columns = X\_train.columns

imputed\_X\_valid.columns = X\_valid.columns

imputed\_y\_train = pd.DataFrame(my\_imputer.fit\_transform(y\_train))

imputed\_y\_valid = pd.DataFrame(my\_imputer.transform(y\_valid))

imputed\_y\_train.columns = y\_train.columns

imputed\_y\_valid.columns = y\_valid.columns

imputed\_X\_test = pd.DataFrame(my\_imputer.fit\_transform(X\_test))

return [imputed\_X\_train, imputed\_X\_valid, imputed\_y\_train, imputed\_y\_valid, imputed\_X\_test]

**3.训练模型**

# 训练并保存模型

def GetModel(a="Model.pkl"):

"""

:param a: 模型文件名

:return:

[socre: MAE评估结果,

X\_test: 预测数据集]

"""

# 取到数据

[X\_train, X\_valid, y\_train, y\_valid, X\_test] = ProcessData()

# 随机树森林模型

model = RandomForestRegressor(random\_state=0, n\_estimators=1001)

# 训练模型

model.fit(X\_train, y\_train)

# 预测模型，用上个星期的数据

preds = model.predict(X\_valid)

# 用MAE评估

score = mean\_absolute\_error(y\_valid, preds)

# 保存模型到本地

joblib.dump(model, a)

# 返回MAE

return [score, X\_test]

**4.预测结果**

# 最终预测结果

preds = model.predict(r[1])

# 打印结果到控制台

print("未来7天预测")

all\_ave\_t = []

all\_high\_t = []

all\_low\_t = []

all\_rainfall = []

all\_pressure = []

all\_wind = []

all\_windSpeed = []

for a in range(1, 7):

    today = DT.datetime.now()

    time = (today + DT.timedelta(days=a)).date()

    print(time.year, '/', time.month, '/', time.day,

          ': 平均气温',  round(preds[a][0], 2), '℃ ',

          '最高气温', round(preds[a][1], 2), '℃ ',

          '最低气温', round(preds[a][2], 2), '℃ ',

          "降雨量", round(preds[a][3], 2), "mm ",

          '气压', round(preds[a][4], 2), 'hPa ', end=''

          )

    if preds[a][5] <90:

        print('东北风', round(preds[a][5], 2), '° ', end='')

    elif preds[a][5] <180:

        print('东南风', round(preds[a][5], 2), '° ', end='')

    elif preds[a][5] <270:

        print('西南风', round(preds[a][5], 2), '° ', end='')

    elif preds[a][5] <360:

        print('西北风', round(preds[a][5], 2), '° ', end='')

    print('风速', round(preds[a][6], 2), 'km/h',)

# 四、结果与可视化

表格

描述已自动生成图表, 折线图

描述已自动生成图表, 折线图

描述已自动生成图表, 折线图

描述已自动生成图片包含 形状

描述已自动生成

# 五、总结与展望

**项目总结：**

通过一次简单的大数据项目，在总体对于大数据相关知识有了一个较为系统的认识。从数据的获取到数据处理，再到建立模型和训练，我们完整的走完了一遍对于数据的利用路线。

在技术方面，我们使用了随机森林模型作为学习模型解决了在特定噪声的数据进行建模时出现的过度拟合问题，完整的使用了MAE回归模型评估。最后在所有人的通力合作之下，我们成功的完成了既定的目标，既运用大数据分析与挖掘的理论与技术解决气温预测的相关问题。

**项目展望：**

天气预测不仅仅是对于气温的预测。若要建立完整的气象预测模型，是非常困难的一件事情，直至今日，最优秀的天气预测系统也不能达到100%的预测准确性，仍需更多更好的算法以及特定气象模型的重新构建。

# 六、参考文献

[1]唐国利, 任国玉. 近百年中国地表气温变化趋势的再分析[J]. 气候与环境研究, 2005, 10(4):8.

[2]Scornet, Erwan, Biau, et al. A random forest guided tour[J]. Test: An Official Journal of the Spanish Society of Statistics and Operations Research, 2016.