暑期学校实验项目：高考志愿填报助手

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 小组名称 | AI算法A组 | | | | | | |
| 姓 名 | 陶特 | 专业 | 人工智能 | 班级 | 091182 | 学号 | 09118206 |
| 实验时间 | 2020.8.31-2020.9.23 | | 指导教师 | 孔祥龙 | | 成绩 |  |
| 1. **实验背景和目的**   高考志愿填报对每一年的考生来说都是一项意义重大的选择。我们总体的项目：高考志愿填报助手建立了一个高考志愿系统，旨在根据考生高考的分数和相关省份、科类、偏好信息，为考生提供合理有效、满足需求的高校及相关专业推荐。在长学期，我们收集并且初步处理了过去三年的部分985高校专业录取情况和各省市一分一段数据表。在短学期，我们的项目目标是基于数据实现高考志愿推荐系统，帮助考生预测可选大学，确定其冲一冲，稳一稳，保一保的若干所大学。 | | | | | | | |
| **二、小组任务和个人任务**  小组任务：  用AI算法实现学校推荐。根据学生的高考分数排名，以及所采集大学的录取分数线和录取人数，可以训练一个能够预测学生报考学校的神经网络分类器。具体任务为整合相关信息，根据一分一段表和高校专业录取信息随机模拟出一些考生排名数据，然后基于预训练模拟数据用神经网络进行分类预测，并对结果进行后续输出和筛选，实现功能扩展。  个人任务：  负责第1～15所高校的录取考生的随机模拟排名生成（信息包括录取学校、所在省份、省份排名和科类），并针对一些由数据不完善导致的问题进行合理的调整，确保后续工作合理、平稳进行。 | | | | | | | |
| **三、个人任务需求分析**  我们组内的数据组任务主要是模拟生成考生省份、科类、省内排名、录取学校的数据。就个人任务而言，是将数据任务的88所高校分别分配到个人，其中我负责第1～15所高校。  我们拥有的数据信息是在长学期收集的高校各专业最低录取分数线（有残缺），以及各省份的一分一段表。总体来说，我们编程处理的基本思路是针对每所大学、每一年份、每个省份、不同科类，分别找出该大学所有专业中最低录取专业分数线和最高录取专业分数线，再最高专业向上给一个合理的录取分数浮动范围（如全国卷可能浮动30分，而江苏卷则浮动10分），这样得到一个总的合理的录取分数范围。例如，2018年江苏的东南大学最高的建筑类397分，最低的专业388分，那么我们就设想获得388～407分的同学可能都会选择并被东南大学录取。然而，该分数区间的考生并不是一定会去该所学校，只是有一定的概率选择该高校。于是我们再从分数范围中，根据设定一个代表拥有某特定分数的学生可能会有多少所大学的选择数的参数k（如获得670分的同学可能只会考虑10所左右的高校，而630分的同学可能会考虑相近的20所高校），对分数区间进行概率上的抽样和估算，对应一分一段表，随机在区间n内挑选n/k名考生，则假设这些考生考上了该大学（例如某个获得670分考生大概有十所学校可以选择，那么在这个大学的这个区分数间的考生就大概有十分之一会去该所大学）。最终模拟出考生分数排名和录取情况信息。  经过不断的运行和调试，最终我们特殊处理了：   1. 江苏省考生的情况，具体表现在将录取分数浮动范围设定为+10，而其他设为+30。 2. 考虑到某些大学并没有所有专业的录取线，甚至只有一个全校总录取线。我们对这类情况采取最低分直接+30处理。而对待正常的数据，为了保证严谨，最终我们也仅在最高专业分上浮动10分。 3. 鉴于某些艺术类/体育类专业的录取分数较低，且情况复杂，我们在处理数据时不考虑，只考虑正常的文化课科目专业分数线。具体表现为通过调整确保每一学校的最高最低专业分数线相差不超过50分。 4. 在我负责的15所高校中，清华北大不仅出现了（2）（3）中的问题，且涉及到每个省的考生最高分数问题（常理中，每个省的前十几被清北录取的概率非常大），于是我们人工查询并适当模拟了所有的清华北大录取人数的数据，在此基础上再做数据模拟。 5. 对于json文件中残缺的一分一段表，我们对超出的分数设置为nan，不存在的设为null。最后人工对null和nan进行近似。 | | | | | | | |
| 1. **实验过程（需附上关键代码及相关说明）**   总体思路为针对每所大学、每一年份、每个省份、不同科类（for循环），分别找出该大学所有专业中最低录取专业分数线和最高录取专业分数线，再最高专业向上给一个合理的录取分数浮动范围，这样得到一个总的合理的录取分数范围。再从分数范围中给定一个恰当的超参数k表示概率，概率合理地根据一分一段表生成恰当人数的排名、学校、省份录取信息。  针对初版本的输出情况，我们又针对艺体专业分数线低的情况、清华北大的特殊情况、专业录取信息残缺情况、一分一段表残缺及不对应情况、江苏等省份的特殊情况进行了处理。并经过多次尝试，调整了k和addition等超参数，最终得到了较为合理的模拟数据。详见代码及个人任务需求分析部分。  以下展示相关代码，详细相关说明在代码注释中呈现：  college\_data = pd.read\_csv('大学信息.csv') # 读取大学数据 print(college\_data[0:15])  college\_batch\_data = college\_data[0:15] # 取前15条数据  '''分别是北京大学 中国人民大学 清华大学 北京交通大学 北京航空航天大学 北京理工大学 北京科技大学北方工业大学 北京化工大学 北京工商大学 北京邮电大学 北京林业大学 北京师范大学 首都师范大学 北京外国语大学'''  Year = [2017,2018,2019] Province = ['北京','天津','河北','山西','内蒙古','辽宁','吉林','黑龙江','上海','江苏','浙江',  '安徽','福建','江西','山东','河南','湖北','湖南','广东','广西','海南','重庆','四川',  '贵州','云南','西藏','陕西','甘肃','青海','宁夏','新疆','香港','澳门','台湾'] category = ["文科","理科", "all"]  result = open("09118206陶特.csv","w",encoding="utf-8-sig",newline='') writer=csv.writer(result) writer.writerow(["rank","province","category","college"])  addition = 10 #向上浮动的分值  for u in university: # 针对每所大学、每一年份、每个省份、不同科类  for y in Year:  for p in Province:  for c in category:  temp\_data = csv\_data.loc[(csv\_data["College"] == u) & (csv\_data["Province"] == p) &  (csv\_data["Year"] == y) & (csv\_data["category"] == c),   ["College", "Year" , "Province", "category","score"]]   if not temp\_data.empty: #存在dataframe  score\_list = temp\_data["score"].values.tolist()  score\_list.sort()  low\_score = score\_list[0] #录取最低专业分  high\_score = score\_list[-1] #最高专业分  for ii in range(20):  if high\_score-low\_score > 50:#保证分数差在50以内，避免艺体专业的影响  low\_score = score\_list[ii+1] #否则就放弃目前的最低分（大概率是艺体专业的分数线  else:  break  if high\_score != low\_score or p == "江苏":#单独处理江苏情况和正常多专业的情况 +10  high\_score = score\_list[-1] + addition  else: #没有专业区分，最低分与最高分相同，且避开江苏（江苏分段小）  high\_score = score\_list[-1] + 30  if c == "不分文理": #没有文理区分，整体处理  temp = "json/" + str(y) + p + "all" + ".json"   else:  temp = "json/" + str(y) + p + c + ".json"    if os.path.exists(temp):  with open(temp,'r',encoding='UTF-8') as load\_f:  load\_dict = json.load(load\_f)  if (str(low\_score) in load\_dict[str(y)][p][c]) & (str(high\_score) in load\_dict[str(y)][p][c]):  low\_rank = load\_dict[str(y)][p][c][str(low\_score)]  high\_rank = load\_dict[str(y)][p][c][str(high\_score)]  number = (low\_rank-high\_rank) // 15 + 1  #k=15，整除15代表近似该分数段有1/15的学生去该大学  for \_ in range(number):  random\_score = random.randint(low\_score, high\_score) #产生随机分数  if (str(random\_score) in load\_dict[str(y)][p][c]):  random\_rank = load\_dict[str(y)][p][c][str(random\_score)]  else:   random\_rank = 1 #说明random\_score超过了一分一段表的最高分  writer.writerow([random\_rank, p, c, u]) | | | | | | | |
| 1. **实验结果与分析**   实验结果： 09118206陶特.csv  rank,province,category,college 3,北京,文科,北京大学 15,北京,文科,北京大学  ...  2145,湖南,文科,北京化工大学 1465,湖南,文科,北京化工大学  ...  795,四川,理科,北京外国语大学 1164,四川,理科,北京外国语大学 1053,四川,理科,北京外国语大学 3687,四川,理科,北京外国语大学  分析：  输出数据为csv，每一条数据分别是学生分数排名、所在省份、文理类别、录取大学。如北京的文科第三名考生被北京大学录取。其中我们忽略了年份，因为排名，甚至归一化后的排名并不会受到年份的影响。csv文件中共240630条数据，其中参数k设定为15。其实也可以对大学进行恰当的编码，根据编码和算式计算出对应的k，最终数据会更加合理。目前，由于缺乏途径和时间，我们只能在一个粗略的范围里用k=15来概括地近似。模拟完成的数据还要经过包装整合、预处理归一化（省份排名/省考试人数）等操作后，再作为训练数据，进入神经网络进行训练。 | | | | | | | |
| 1. **实验总结与心得体会**   本次软件实践的项目：高考志愿填报推荐系统，旨在根据考生高考的分数和相关省份、科类、偏好信息，为考生提供合理有效、满足需求的高校及相关专业推荐。我们小组的任务是用AI算法实现学校推荐，具体来说是根据各省的一分一段表和高校录取成绩模拟若干考生数据，在此基础上训练神经网络分类器，实现预测录取学校的功能。其中，我的工作主要是针对第1～15所高校的历年录取情况和各省市的一分一段表随机模拟出基于考生个人的录取信息和成绩排名情况。  本次项目实验不仅锻炼我本身的python编程能力，更是增强了我的团队意识。这是我首次参与并体验到一个一百多人共同完成的项目。就小组内部而言，我们在组内进行了多次线上线下的讨论，分工合作，探讨方案，一次次修改分工计划，协调分工与进度，一起发现问题、解决问题，通过每一个人的努力把模型完善的更好。而就小组之间而言，我们模拟数据部分需要多次和组长以及第一组的同学对接，也要和ui组、数据库组对接。同学们相互协调，共同努力，给了我很特别的体会。  作为一个实验总结，每个组都基本完成了各自的任务。我们数据模拟小组由于数据的缺失和局限，做了一些自己的调整，所幸最终提供了基本合理的模拟数据。我们也在对接和小组项目展示时看到了其他组的成果，都完成的很不错。总之，在老师和同学们共同的努力下，这个项目从计划到需求分析，再到系统设计，实现完善，维护修改。尽管最终推荐系统网页并不会上线提供使用，这个项目本身也给我们所有人带来了很有裨益的体验。从中我也体会到了AI算法的在项目实现中的魅力，感受到了团队合作的重要性以及在项目进程中创新能力的可贵。 | | | | | | | |

2020年9月制