期末汇报

张亦佳

处理文本特征

- 合并所有文本特征

```
demo["all_text"] = demo["房屋优势"] + " " + demo["核心卖点"] + " " +
```

- 使用jieba进行分词并绘制词云图

```
tokens_list = []
for text in demo["all_text"]:
    if isinstance(text, str):
        words = jieba.lcut(text)
        tokens = [word for word in words if len(word) > 1]  # 只保留长
        tokens_list.append(tokens)
    else:
        tokens_list.append([])  # 处理空文本
```

```
wc=WordCloud(font_path=font_path,
             background_color='#FFEEEE',     # 背景色（浅红色）
             colormap='hsv'     # 词语颜色方案（彩虹色）
            )
wc.fit_words(dict(all_tokens))

fig=plt.figure(figsize=(10,5))
plt.imshow(wc)
plt.axis('off')
plt.show()
```

- 应用Word2Vec训练出词向量

```
model = Word2Vec(
    sentences=tokens_list,
    vector_size=20,    #
    window=3,          # 前后5个词作为上下文
    min_count=3,       # 整体中出现次数少于5的词删掉
    workers=4,     # 并行计算加速训练过程
    epochs=5     # 模型看5遍所有数据
)
# 可以参数优化吗?
# 语义相近的词在向量空间中相近
```

处理文本特征

- 用训练好的Word2Vec得到每条数据的文本的词向量

```
doc_vectors = []

# 遍历每个房屋的分词结果
for tokens in tokens_list:
    # 初始化一个50维的零向量
    vector = np.zeros(50)
    count = 0    # 记录文档中有多少在Word2Vec模型中用到的有效词

    for word in tokens:
        # 检查词是否在Word2Vec模型的词汇表中
        if word in model.wv:
            # 累加该词的向量到文档向量
            vector += model.wv[word]

            count += 1    # 统计文档中有多少个词被包含在词向量模型中

    # 计算平均向量  保证不同长度的文本有可比性
    if count > 0:
        vector /= count  # 向量除以词数

    # 将当前文档向量添加到结果列表
    doc_vectors.append(vector)
```

- 随机森林模型预测