Store planner software project report

Author: Gleb Samokhvalov 211-2
Submission date: 12.06.2022
Suprevisor: Shershakov Sergey Andreevich

Problem statement:

I had to create a QT application which will be helpful in finding the closest store with one parametr as an input. CSV file with the list of his competitors' store.

Individual problem:

1) Add rows 2) Delete rows 3) Implement analog of fillna method from python package "Pandas" but using some magic: you should not just fill the cell with the data from upper or lower row, but try finding the closest value to this one by other left store characteristics (the importance of values is Area -> Customers -> Sales -> Items) so, for example if you are lacking the Sales cell, you first look for 50 closest stores on Area, after that you look for 25 closest on Customers value (across 50 closest on Area) and so on. This numbers(50, 25) you can choose by yourself. I hope the main idea is clear

Moreover, author presented the way how UI should be performed. Details can be found here.

Implementation secrion

As a programmer, I faced two major problems. The first one was connected with the csv file parsing. However, I came up with an idea of using RapidCSV library, documentation of which can be found here. I found this library very convinient and it makes the process of working with CSV files much easier. The second problem was about kind of predictions based on the given dataset. In case one adds new store with empty cell, by pressing "Fill blank" button the cell should be filled by closest value by other store charasteristic. For solving this problem I decided to write a structure to sort the "Store" object. Here I decided to slightly deviate from the specification(it is said to start with 50 closest for "Area" => 25 closest for "Customers" and etc.). As I have 4 columns, lets name them parameters. To end up with only one the best suitable option, I have chosen 4! = 24 options after the first dropout, because after 24 options there will be 4 parameters => 24/4 = 6 => 6/3 = 2 => 2/2 = 1. This is what we needed. Still there is one button(which was optional) which I have not done. It is about choosing the priorities of sorting.

Results and discussion

The requriements were fullfilled with some changes in algorithms and except one optional button. Author asked to follow the UI strictly and I did it.

Conclusion

Further imporvements could be connected with an integration of machine learning to optimize and reduce the chances of mistakes in finding the closest one.