Data Management Algorithm for Decision Making
02360003

Final Project Report

Code repository - https://github.com/ShangolMangol/AlgoForDecisions/tree/main/project

Ido Tausi - 214008997
Afek Nahum - 214392706

Project Abstract:

The goal of our project is to find interesting visualizations of subsets with high correlations, or subsets that are different from overall correlation.
The project consists of generic scripts that can run on any normalized and binned dataset with enough data. We ran the scripts on 2 different datasets and provide results and analysis in this report. We selected the following intriguing datasets from Kaggle for our project:

- Student Depression Dataset, that includes different attributes about the students' mental well being, work/academic situation etc'.
- Heart Attack Risk Dataset, that includes many different attributes about patients' health.

Note: originally our second dataset was a billionaires dataset. However, we decided to switch to a different one after further examining it.

Pre-processing:

The first step in order find interesting visualizations in our data, is to perform a few pre-processing steps. These include filling $Null$ values with real ones (according to other values in the database), transforming continuous attributes to categorical ones by creating bins for a range of continuous values, and switch non-numerical values to numerical values.
These pre-processing steps are crucial in order to find subsets more easily for interesting visualizations results, since they allow to measure the correlations between various attributes to the target attribute numerically and reliably.

Execution:

After the pre-processing of the data, we arrive to the main execution of the script. We firstly find different subsets in the database, in order to do that we iterate on all of the columns and all of their unique values, and making a subset for each of this unique value, only if it would result in a subset size that is bigger the a threshold (for example, at least 10 students in a subset).
The subsets would be organized in a dictionary that is built like the example below, where the keys are the column name, and the data is another dictionary where the keys are the unique values that the subset was selected from. For each key in the inner dictionary, is a data frame that contains the rows for the dataset, and it would be able to use intersection from other subsets and create multiple constraints subsets.

```
subsets = {
    "Age": {
        (18, 20]: dataframe,
        (20, 22]: dataframe,
        ...
    },
    "CGPA": {
        (6.5, 7.0]: dataframe,
        (7.0, 7.5]: dataframe,
        ...
    },
    ...
}
```

Main functionality and correlation computation:

The core functionality of this project is implemented in two key functions, each designed to identify the most insightful visualizations based on different criteria:

1. **Highest Correlations Within Subsets** – Which determines the strongest correlation between a specific attribute and the target variable within a specific subset of the data.
2. **Greatest Difference in Correlations** – Which identifies attributes where the correlation within a subset differs the most from the overall correlation in the entire dataset.

Why we chose Spearman's correlation:

We chose to use Spearman's correlation in our project, this correlation is used to find a connection between 2 categorical variables, while also taking into account their natural order

(also known as ordinal variables). Each one of the functions returns a list of tuples of the subset group, the effecting attribute, and the correlations results, as demonstrated below.

```
Highest Correlations Within Subsets:
[(('Degree', 'B.Ed', 'Have you ever had suicidal thoughts ?'),
  0.6175350534452649),

 (('Degree', 'B.Arch', 'Have you ever had suicidal thoughts ?'),
  0.6112021600544345),
  ...
]

Greatest Difference in Correlations:
[(('Degree', 'MA', 'Dietary Habits'),
  {'correlation': 0.09300349820254594,
   'overall_correlation': 0.20752016720151734,
   'difference': 0.1145166689989714}),

 (('Academic Pressure', 1, 'Study Satisfaction'),
  {'correlation': -0.061594465270066914,
   'overall_correlation': -0.16823010856067103,
   'difference': 0.10663564329060411}),
  ...
]
```

We use these dictionaries to generate insightful visual results to learn about the correlation and interesting findings in the data, on which we'll expand later.

Computational cost:

In our pre-processing of the datasets, we made sure that there are at most 10 different values in each columns, including numerical columns that were distributed to bins.

Assuming there are d columns in a dataset (which are not the target column) and p rows, our algorithm run on all of the different combinations of values from each column, and selecting its subset, at the worst case in which each column would have exactly 10 values, it would cost $10d \cdot p = O(d \cdot p)$ operations.
Now, each subset would calculate its intersection with each value of another column, therefore it would cost another $10d \cdot (d-1) \cdot p = O(d^2 \cdot p)$ operations. We assumed that

the distribution of values in each column are uniform, meaning each subset is $1/10$ of the number of rows, so $O(d^2 \cdot p) \cdot O\left(\left(1/10\right) \cdot p\right) = O(d^2 \cdot p^2)$ operations for subset intersections.

(To calculate the correlation of each subset to the target column, we assume that the calculation of correlation is $O(n \log n)$ when n is the number of rows in the table)
In addition, we calculate the correlation of each column in the overall dataset to the target column, so there is another $O(d \cdot p \log p)$ operations.

At the end we want the top-k correlations, so we need to sort all of the $O(d^2 \cdot p)$ values, so there is another $O\left(d^2 p \cdot \log\left(d^2 \cdot p\right)\right) = O\left(d^2 p \cdot \log(d \cdot p)\right)$.

In conclusion, the total cost of the algorithm is:

$$O(d \cdot p) + O\left(d^2 \cdot p^2\right) + O\left(d^2 p \cdot \log(d \cdot p)\right) \Longrightarrow O\left(d^2 \cdot p^2\right)$$

Optimizations:

Highest correlations mode, optimized:

- Firstly, we can calculate Entropy score for each column, and take the top-N to make the subsets from.
- From those columns, we take the top-M columns with highest correlation to the target column.
- Now calculate subset correlation between each of top-M columns.

Biggest difference correlations mode, optimized:

- As before, we can calculate Entropy score for each column, and take the top-N to make the subsets from.
- Now, we would check 3 different methods:
    1. From those columns, we take the top-M columns with highest absolute correlation to the target column.
    2. From those columns, we take the top-M columns with lowest correlation to the target column.
    3. We mix the top-$m/2$ columns with high correlation, and the rest with the lowest correlation, trying to make a balance between them.
- Now calculate subset correlation between each of top-M columns.

Why we chose Entropy?:

- Speed up our algorithm by reducing the number of columns we need to consider for subset creation.
- Increase the likelihood of finding interesting results, by focusing on columns that are more diverse and potentially more informative for distinguishing subsets with different correlation patterns.

Computational cost:

In our pre-processing of the datasets, we made sure that there are at most 10 different values in each columns, including numerical columns that were distributed to bins.

Assuming there are d columns in a dataset (which are not the target column) and p rows.
Now we choose the top-N columns with highest entropy, so we need to calculate entropy for each column for all of the p rows and choose the top $N \Rightarrow O(d \cdot p + d \log d)$.
After that, our algorithm run on all of the different combinations of values from each of N columns, and selecting its subset, at the worst case in which each column would have exactly 10 values, it would cost $10N \cdot p = O(N \cdot p)$ operations.
Now, each subset would calculate its intersection with each value of one of the M columns

that were chosen for this part (for example, top-M highest correlation to the target column). The computation cost for the top-M column correlations would cost $O(d \cdot p \log p + d \log(d))$ because we need to calculate the correlation for each column which is $O(p \log p)$, and then choose its top-M. Therefore it would be $10N \cdot M \cdot p = O(N \cdot M \cdot p)$ subsets.
To calculate the correlation of each subset to the target column, we assume that the calculation of correlation is $O(n \log n)$ when n is the number of rows in the table, and we the distribution of values in each column are uniform, meaning each subset is $1/10$ of the number of rows, so $O(N \cdot M \cdot p) \cdot O\left(\left(1/10\right) \cdot p\right) = O\left(N \cdot M \cdot p^2\right)$ operations.

At the end we want the top-k correlations, so we need to sort all of the $O(N \cdot M \cdot p)$ values, so there is another $O(N \cdot M \cdot p \cdot \log(N \cdot M \cdot p))$.

In conclusion, the total cost of the algorithm is:

$$O(d \cdot p + d \log d) + O\left(N \cdot M \cdot p^2\right) + O(N \cdot M \cdot p \cdot \log(N \cdot M \cdot p))$$

$$\Longrightarrow O(d \cdot p + d \log d + N \cdot M \cdot p \cdot \log(N \cdot M \cdot p))$$

We can see that when we choose $N, M \ll d$ we get better complexity than the naive method.

Real examples from our work:

On the depression dataset, we have $d = 13,\ p = 27,901$.
When we ran the **Highest correlations mode optimized**, we got roughly the same result, and it started to deteriorate and got worse results when $N, M \approx 5$**,** we believe it is due to the nature of the dataset and the connections between the columns, but overall it ran faster and the optimizations were reducing the computational load.

Non optimized - 4.5 seconds to run:

```
get_k_top_different_from_overall_correlations(binned_dataset, target_column, k=10)
```

✓ 4.5s

```
[(('Heart_Rate', 109, 'Alcohol_Consumption'),
  {'correlation': 0.0927077709862501,
   'overall_correlation': 0.004998848518908901,
```

Optimized - 0.9 seconds to run: <u>80% faster!</u>

```
    get_top_k_subsets_different_corr_optimized(binned_dataset, target_column, mode=ChosenColumnMode.LOWEST_CORR, m=5, n=5)
5]  ✓ 0.9s

[(('Heart_Rate', 89, 'Physical_Activity_Level'),
  {'correlation': 0.08523305237482913,
   'overall_correlation': -0.0004178313143681622 7,
   'difference': 0.0856508836891973}),
```

Note: the results here represent the trade-off between the exhaustive search naive algorithm, and the optimized algorithm. In the optimized algorithm we get results that are starting from 0.05 difference from the real solution, and missing more solutions in between, as $M = 5,\ N = 5$.

Results:

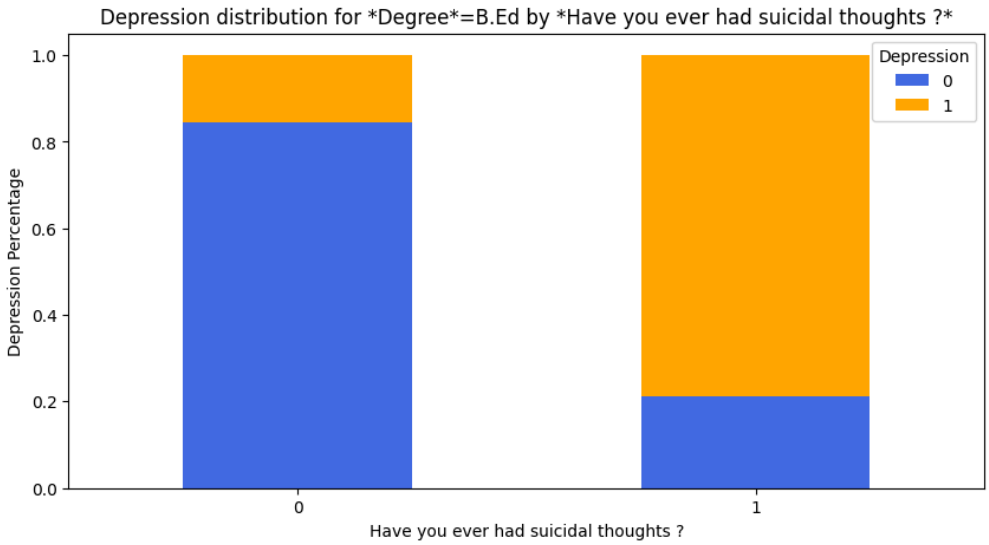
We'll go over a few interesting visualizations results from our chosen data sets.

Students Depression dataset:

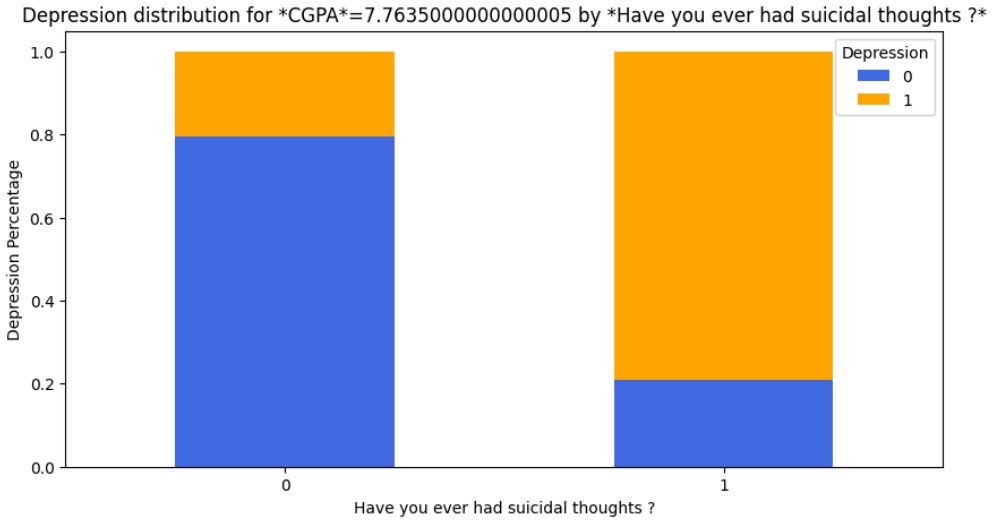
Depression: 1 = yes, 0 = no.

Subsets with high correlations:

- For the subset of people that learn education, we can see that there is a high correlation between having suicidal thoughts and having depression. The Spearman correlation between Having suicidal thoughts and having depression for this subset is 0.617 (and $p\_value < 0.05$).

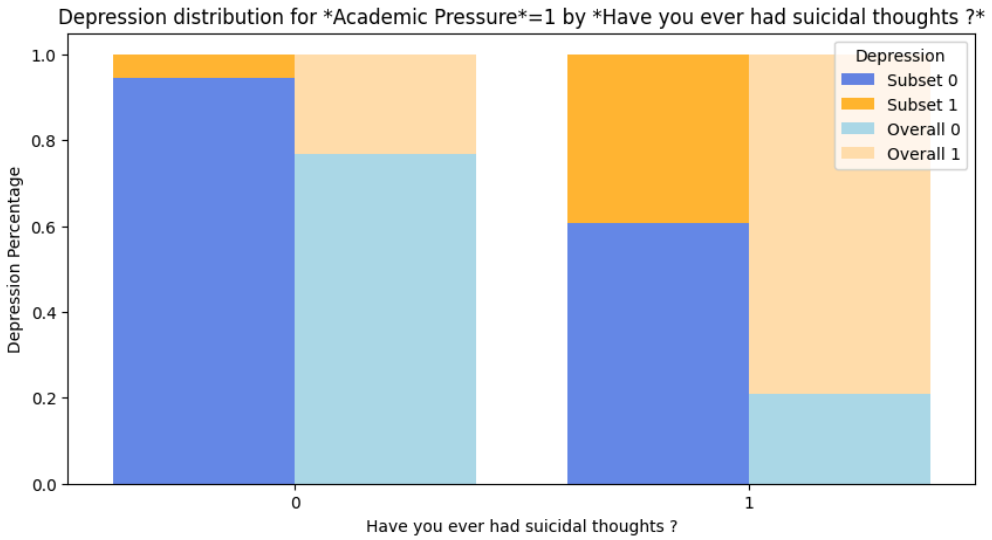


- For the subset of which have a CGPA of approximately 7.77 (since we used bins, and this is its median), we can see that there is a high correlation between having suicidal thoughts and having depression. The Spearman correlation between Having suicidal thoughts and having depression for this subset is 0.576 (and $p\_value < 0.05$).

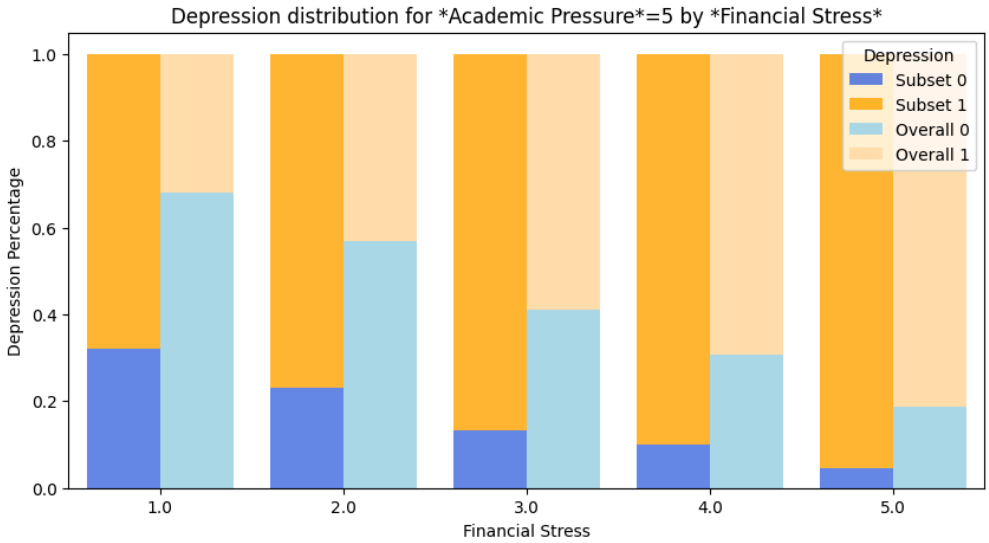


Subsets that are different from overall correlation:

- For the subset of people which have a very low academic pressure (=1), we can see that if the they had suicidal thoughts before, the depression rate is rising. The Spearman correlation between having suicidal thoughts and having depression for this subset is 0.42 (and $p\_value < 0.05$). Meanwhile, the overall correlation of the whole dataset between having suicidal thoughts and having depression is 0.546.
  The meaning of this result, is that people with a very low academic pressure, are less likely to experience depression, even if they had suicidal thoughts before.



- For the subset of people which have a very high academic pressure (=5), we can see that a rise in financial stress causes the depression rate to rise as well. The Spearman correlation between financial stress and having depression for this subset is 0.256 (and $p\_value < 0.05$). Meanwhile, the overall correlation of the whole dataset between financial stress and having depression is 0.362.
  The meaning of this result, is that people with a very high academic pressure, are less much more likely to experience depression, regardless of their financial stress.

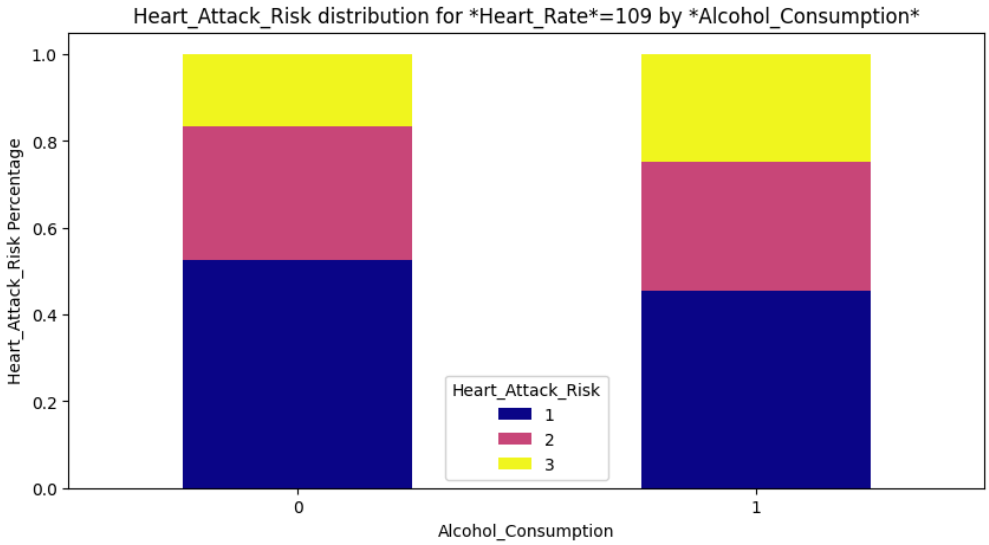


<u>Heart Attack Risk dataset:</u>

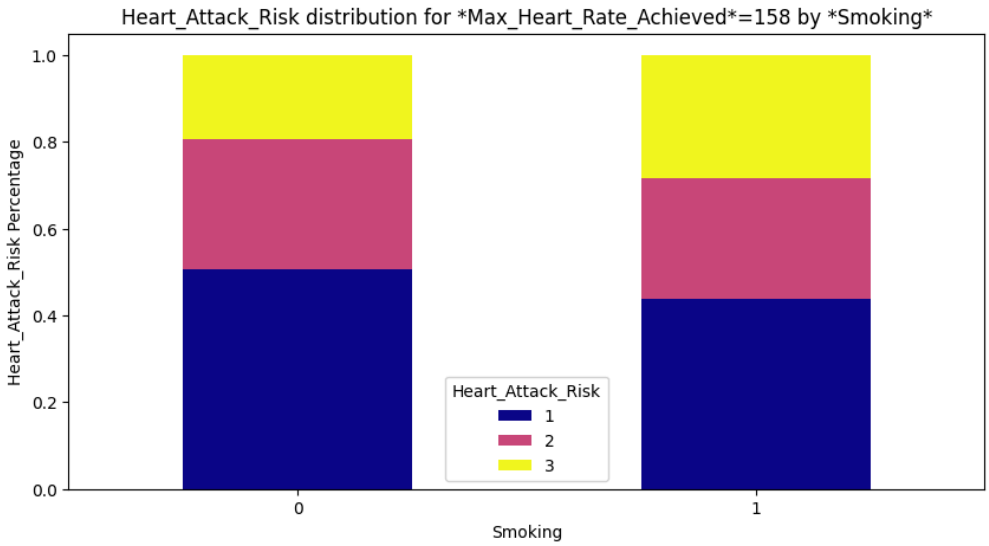
Heart_Attack_Risk: Low = 1, Moderate = 2, High = 3.

Subsets with high correlations:

- For the subset of people which have a resting heart rate of approximately 109 (since we used bins, and this is its median), we can see that if the patient is consuming alcohol (=1), the risk is rising. The Spearman correlation between alcohol consumption and the risk of heart attack for this subset is 0.0927 (and $p\_value < 0.05$).



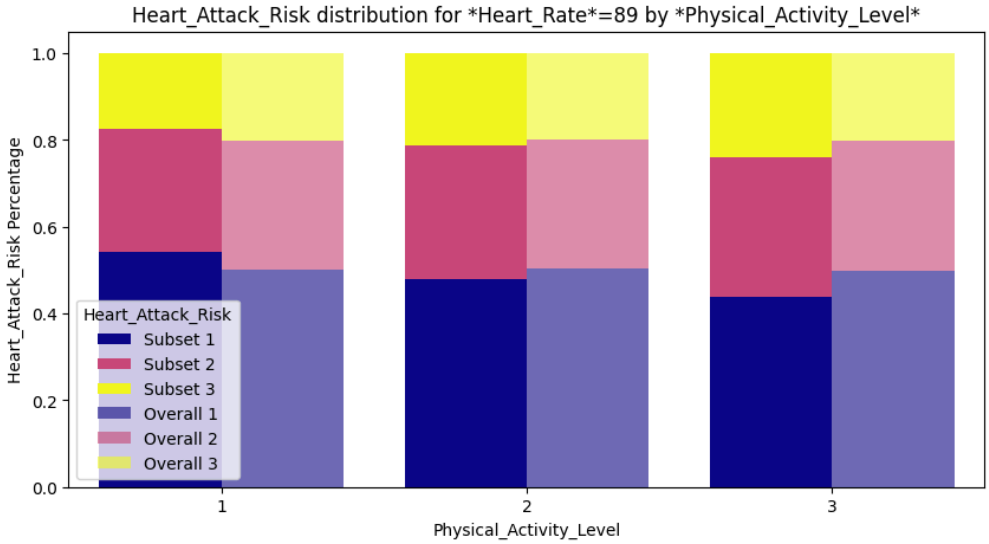
- For the subset of people which have a max heart rate of approximately 158 (median of its bin), we can see that if the patient is smoking (=1), the risk is rising. The calculated Spearman correlation between smoking and the risk of heart attack for this subset is 0.0857 (and $p\_value < 0.05$).



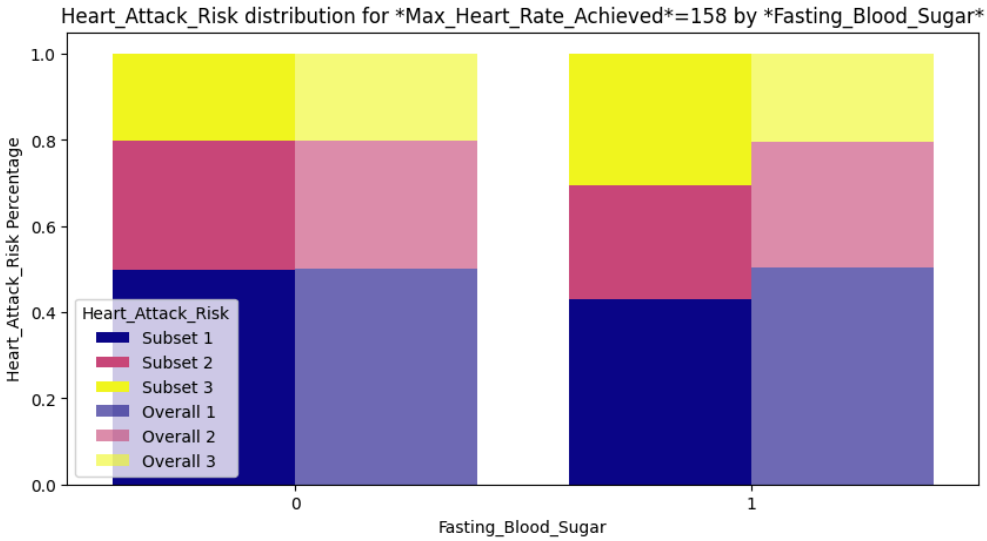
Subsets that are different from overall correlation:

- For the subset of people which have a resting heart rate of approximately 89 (the median of the bin), we can see that if the patient is doing physical activity (3 is the highest amount of activity), the risk is rising. The Spearman correlation between physical activity level and the risk of heart attack for this subset is 0.0852 (and

$p\_value < 0.05$). Meanwhile, the overall correlation of the whole dataset between physical activity level and the risk of heart attack is -0.0004.
The meaning of this result, is that people with resting heart rate of approximately 89, should not be doing high level of physical activity because it increases their risk.



- For the subset of people which that have max heart rate of 158, we can see that if a patient also have fasting blood level of 1 ($0 \implies < 120\ mg/dL$, and $1 \implies \geqslant 120\ mg/dL$), the risk is rising significantly. The Spearman correlation between physical activity level and the risk of heart attack for this subset is 0.073 (and $p\_value < 0.05$). Meanwhile, the overall correlation of the whole dataset between fasting blood level and the risk of heart attack is -0.0008.
  The meaning of this result, is that people with max heart rate of approximately 158, should be careful about their level of sugar in their blood, because the higher it is, the higher is their risk.


Heart_Attack_Risk distribution for *Max_Heart_Rate_Achieved*=158 by *Fasting_Blood_Sugar*
Heart_Attack_Risk Percentage
1.0
0.8
0.6
0.4
0.2
0.0
0
1
Fasting_Blood_Sugar
Heart_Attack_Risk
Subset 1
Subset 2
Subset 3
Overall 1
Overall 2
Overall 3