Task 9 - EDA

We used translation for our columns, you can see them in translation folder.

We built a generic EDA (exploratory data analysis) for the common good.

Team 2 - Son and Gal

Tasks:

Task ID	Description	Status	Progress
2	Create a full Word file and Jupyter notebook	Written when everyone finishes their tasks.	Not Started
9	EDA of hospitalization1	**Finished**	100%
20	Connection between Doctor Type and rehospitalization	Created Classification model, considered adjustments to the data.	80%
26	Connection between age and gender to rehospitalization from 16-19	Pending for tasks 16-18	Not Started
38	Dimension Reduction for hospitalization2	Pending for task 20's pull request	60%
46	Editing conclusions chapter	Pending	Not Started
Unique	PowerPoint Presentation	Pending	Not Started

Conclusions:

```
**Task 9: **
Translations to hebrew, multiple diagnoses - top 10 dignoses:
        Admission: 1. 78609
                   2. 7865
                   3. 78060
                   4. 08889
                   5. 2859
                   6. 7895
                   7. 486
                   8. 4280
                   9. 42731
                   10. 7807
        Release: all of the above and in addition 5990 & 514

        No strong correlation between the different features.
        Most hospitalizations were urgent and most were short, mainly a few days.
```

```
        No drastic amount of patients in any specific unit.
        Most hospitalization patients were excorted from home.
**Task 20: **
- Based on Doctor rank data(Senior/Not Senior), the ranks are split almost equally to 4  cat
- Based on the PIE chart, They're split almost equally in amount of doctors.
- Based on Gradient Boosting Model that predicts rehospitalization based on Doctor rank, the
```

, 16-18 depends on: - [[Task 4]] - [[Task 16]]

#Task

Data cleaning and completion for table: general data

#Task

- [[Task 4]]
- [[Task 28]]

#Team

asdflkjqrgkajfdgqlekrjgqeropigjqegjqerpogijqe

#Task

Tasks:

Task 4: Data Cleaning

Task 14: erBeforeHospitalization2 EDA

Task 14: erBeforeHospitalization2 EDA:

Task 16: optimal distribution

Task 28: Find Connections

Task 32: Time Series Analysis

Comments:

For results and conclusions please download HTML file (not all plots can be seen in ipynb file)

#Team

Team 09

We have implemented a CLI that enables to perform EDA on the file `rehospitalization.xlsx`.
The file was originally uploaded to our course, in Moodle. The latest version we have worked with is in the directory `./assets`.

Our EDA consists of a couple of missions: - **task__06**: Missing values treatment in the sheet `erBeforeHospitalization2`.
The output is available in the file `team_09_task_06_erBeforeHospitalization.csv` (directory `assets`). - **task__15**: Parameter exploration in the `hospitalization2.csv`.
The results & conclusions are available in the document `team_09_task_15_hospitalization2_EDA` (directory `assets`). - **task__22**: Relationship exploration between the release day-of-week and rehospitalization. The output is available in the file `team_09_task_22_relationship_day_of_week_to_rehospitalization.md` (directory `assets`). - **task__31**: Timeseries analysis between 2nd admission date and rehospitalization occurrence. The output is available in the file `team_09_task_31_admission_date_timeseries_analysis.md` (directory `assets`).

How to use the CLI?

Fill in missing values for erBeforeHospitalization2

The full path for `erBeforeHospitalization2` includes a number of steps, each represented by a call to `main.py` below: - Transform sheet `erBeforeHospitalization2` to `ASCII` encoding only - Fill in missing values - Transform sheet `erBeforeHospitalization2` back to the original encoding - Create `erBeforeHospitalization2`

```
./main.py -v -i rehospitalization.xlsx -o rehospitalization.xlsx --ascii-encoded erBeforeHos
./main.py -v -i rehospitalization.xlsx -o rehospitalization.xlsx --missing-values erBeforeHo
./main.py -v -i rehospitalization.xlsx -o rehospitalization.xlsx --original-encoded erBefore
./main.py -v -i rehospitalization.xlsx -o team09_task06_erBeforeHospitalization.csv --sheet-
```

Relationship test between day of release and rehospitalization

```
./main.py -v -i rehospitalization.xlsx -o NA --relationship-test-release-date-rehospitalizat
```

Time series analysis between 2nd admission date and rehospitalization occurrence

```
./main.py -v -i rehospitalization.xlsx -o NA --time-series-analysis hospitalization2 Admissi
```

Note-worthy implementation details

task_06: Missing values treatment

erBeforeHospitalization2 sheet has many patients who were admitted to the 2nd hospitalization without going through the ER ( ).
These patients lack details about ER, which led us to supplement values that indicate that they did not visit the ER.
The parameters were chosen as following: - Medical_Record = 1000000 - ev_Admission_Date = 1900-01-01 - ev_Release_Time = 1900-01-01 - Transport_Means_to_ER ( ) = 'No Emergency Visit' - ER ( ) = 'No Emergency Visit' - urgencyLevelTime = 0 - Diagnoses_in_ER ( ) = 0 - codeDoctor = 0

Anyone who had a blank entry in the Transport_Means_to_ER ( ) column was updated with 'Not provided'.

For those in the ER ( ) column with the value ICU ( ) the missing values in the columns Diagnoses_in_ER ( ) and codeDoctor were updated with 1.

Non-ASCII chars

Hebrew characters belong to a broader encoding family, UTF-8.
While it is widely used, "best-practice" recommends to avoid its usage as it is impossible to know which 3rd party module will be used in the system as a whole. On the contrary, ASCII encoding is supported by virtually any 3rd party module. We have a dedicated mechanism, with module HebEngTranslator at its core, that transforms documents/tables to ASCII encoded only and back to the original format.

task_15: Timeseries analysis between 2nd admission date and rehospitalization occurrence

Load & Handle Missing Data
Start by loading the dataset from a CSV file. Then count and print the number of missing values (NaNs) per column. Handle missing data by dropping rows with any missing values.

Define Column Types
Dynamically identify numerical columns (int64 and float64) and categorical columns (object and category).

Descriptive Statistics
Print descriptive statistics for all columns, provide insights such as mean, median, count. Provide standard deviation insight for numerical data, count, unique values, and top values for categorical data.

Visualization of Numerical Data
* Histograms: Generate histograms for all numerical columns to visualize their

distributions. Use subplots to arrange these histograms in a grid that adapts to the number of numerical columns.
* Box Plots: Create box plots for these columns to further analyze the distribution of data and identify outliers. Visualization of Categorical Data: * Bar Charts: For categorical data, generate bar charts to visualize the frequency distribution of the top 10 most frequent categories in each categorical column. Adjust the figure size, rotate x-axis labels for readability, and ensure that the layout does not have overlapping elements. Correlation Analysis: * Compute and visualize a correlation matrix for numerical columns using a heatmap, which helps identify any significant correlations between variables. Clustering of Individual Variables: * Normalize the numerical data using StandardScaler to prepare for clustering. Perform KMeans clustering for each numerical column individually, visualizing the clustering result as a scatter plot of the scaled data against its index, colored by cluster label. This visualization is particularly useful for identifying patterns or groups within individual features. Key Details: * Matplotlib's subplots are used extensively to create grids of plots. The grid size is dynamically adjusted based on the number of columns. Scikit-learn's StandardScaler and KMeans are used for data scaling and clustering, respectively. Seaborn is used for enhanced visualization like box plots and heatmaps. DataFrame operations, such as selecting data types, handling missing values, and indexing, are effectively utilized to prepare and manipulate the data.

task_22: Relationship between release day of week and rehospitalization

None of the models should use this relationship.
There is a definitive bias towards finding predictive ability between day of week and rehospitalization, as *only patients who are rehospitalized are mentioned in sheet* ***hospitalization1***. We have no data regarding patients who are not rehospitilized.
This prevents the *mandatory establishment of relationship existence* between day of week and rehospitalization, which makes any predictive ability describe above invalid.

Here's the output of the relevant piece of logic to prove our statement:

```
(venv) maximc@Maxims-MacBook-Pro team_9 % ./main.py -i rehospitalization.xlsx -o NA --relati
Type of target variable: discrete.
        Possible target labels: "rehospitalized", "non-rehospitalized"
Type of feature variable: discrete.
        Possible target labels: "Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Fri
Conditions for statistical relationship test are not met, because of definitive bias:
        Number of rehospitilied patients: 7033 VS number of non-rehospitilized patients: 0
        We are unable to create "contingency table" that is a requirement for Chi-Squared or
(venv) maximc@Maxims-MacBook-Pro team_9 %
```

Research Question

Is there statistical relationship between `DayOfWeek` and `Rehospitalization`? ## Target variable * Type: discrete * Possible target labels: `"rehospitalized"`, `"non-rehospitalized"`

Feature variable

- Type: discrete
- Possible feature labels: `"Sunday"`, `"Monday"`, `"Tuesday"`, `"Wednesday"`, `"Thursday"`, `"Friday"`, `"Saturday"`

Analysis output

```
Conditions for statistical relationship test are not met, because of definitive bias:
  Number of rehospitilied patients: 7033 VS number of non-rehospitilized patients: 0
  We are unable to create "contingency table" that is a requirement for Chi-Squared or Fishe
```

Conclusion

It is impossible to state presence of statistical relationship between day-of-week and rehospitalization.

Timeseries analysis

We have performed timeseries analysis on sheet `hospitalization2`, column `Admission_Entry_Date`.

Our initial attempts were to produce a graph on a *daily* and *weekly* basis. However, the line was mostly "jumping" up-and-down without any consistency and clarity.

Our next attempt was a plot on an *yearly* basis. It produced a sort of reverse effect, as there is only 3 years worth of data.

The final plot that can be seeing below and is on a *monthly* basis: timeseries-analysis-on-monthly-intervals

We have used 2 lines: - Monthly number of rehospitalization occurrences - 3-month average of rehospitalization occurrences

The former is a bit "jumpy". However, it shows a consistent increase of the overall rehospitalizations year by year.

The later illustrates further that number of rehospitalizations increases on average.

In addition, we have identified that periods between `May` till `November` experience a sharp rise and sharp decrease compared to the rest of the months in a year. We have encircled such picks in black.

Type of day analysis

Unfortunately, as discussed in `team_09_task_22.md` - there is no way to establish presence of statistical relationship between type of day and rehospitalization.
Type of day (`workday`, `weekend`, `holiday`) and rehospitalization are both categorical variables.
Tests that define presence of statistical relationship (such as Chi-Square) require a "contingency table".
It can not be constructed because we have no data about patients who *did not rehospitalized* on a particular type of day.

Conclusion

A 100% bias towards rehospitalization exists and makes impossible to judge statistical relationship between type of day and rehospitalization.

There is a definitive increase, on average, in number of rehospitalizations per month.
Period of `May` till `November` experiences a sharp increase and decrease in number of rehospitalizations.