Team ID : PNT2022TMID27605

Project Report

Date	17th November 2022
Team ID	PNT2022TMID27605
Project Name	Project - Corporate Employee Attrition Analytics
Team Members	Team Lead - Sherin Sneha J Team Member 1 - Sharwin Xavier R Team Member 2 - Dini Shiba S Team Member 3 - Murugalakshmi V

INTRODUCTION

The two primary components that contribute to the development and growth of the nation are corporate businesses and industries. Manpower, often known as the workforce, is a crucial component of any corporation. Performance and expansion of the business depend on how long the employees stay in their jobs. The fundamental difference between attrition and retention is that each has a different aim, but they are fundamentally related since one clears the way for the other. Global marketplaces are getting more competitive over time, which has altered workplace culture. The existence of the labor force, the emerging imbalance between the supply and demand of competent workers, and the growing importance of work-life balance have made it difficult for the company's HR and management to find the right candidate for the right role. The two faces that reflect the approach to determine business employment trends, general business growth, motivation, and growth are attrition and retention. Because losing a valuable employee has a negative impact on knowledge value, uneasy coworkers lost capital, and the organization's reputation, it is observed that globally competitive organizations spend a significant amount of interest, time, and money on employee attrition. This ultimately results in the failure of the business or organization.

1.1 PROJECT OVERVIEW

The organization's success depends on its ability to draw in and keep outstanding personnel. Identifying the factors that retain employees at the organisation and those that cause others to quit is a crucial responsibility for HR. A number of data points about the personnel who are either still employed by the firm or have left it are included in the data. To stop the company from losing talented employees, it is important to recognise and address these problems.

1.2 PURPOSE

- To analyze the factors that causes the employee attrition through predictive analysis and to give suggestions by modelling techniques to reduce the cause of retention.
- Visualization Charts are prepared to highlight the insights for the given dataset
- Creating dashboard for the HR and managers for understanding the reasons for attrition and to take necessary measures in the organization.

2. LITERATURE SURVEY

Reduction in the number of employees in a company is referred to as employee attrition. Employee attrition has been a recognised problem for the corporate sector during the past twenty years. Employees depart from the company for a variety of reasons. Among the causes include the need for high pay, changes in technology or roles, obstacles in the workplace, etc. High attrition increases the cost of various company characteristics and functions. The overall cost to the employees is increased by recruitment, training, and development expenses.

2.1 EXISTING PROBLEM

Both the employer and the employee have recently lost faith in one another. The former believes that an employee can quit the company at any moment, whereas the latter believe that the former can dismiss the employee at any time. Regardless of who is at fault, a loss of workers is unavoidable. Attrition refers to this labor loss, regardless of the cause. Attrition is a prevalent issue in any organization, regardless of the kind of business or organizational structure, which not only hinders output and results in high long-term expenses and a loss of goodwill to the organisation. Therefore, it becomes necessary to investigate this complex issue and find workable answers.

2.2 REFERENCES

1. **TITLE**: From Big Data to Deep Data to Support People Analytics for Employee Attrition Prediction.

YEAR: 2021

AUTHORS: Nesrine Ben Yahia; Jihen Hlel; Ricardo Colomo-Palacios

DESCRIPTION: In the era of data science and big data analytics, firms and their HR managers can reduce attrition by using people analytics, which transforms how businesses and their human resources (HR) managers find and keep talent. Staff attrition is a big problem for businesses in this situation since it affects both production and the continuity of planning. The main contributions that this study has made in this situation are listed below. We start by proposing a people's The analytics approach to employee attrition prediction shifts from a large data environment to a deep data one by focusing on data quality rather than quantity.

2. **TITLE**: Towards Understanding Employee Attrition using a Decision Tree Approach

YEAR: 2019

AUTHORS: Saadat M Alhashmi

DESCRIPTION: The severe issue of employee attrition has been the subject of research for several decades. This problem has been approached using a variety of methods, including psychological studies and exit interviews. The goal is to prevent or minimise employees leaving a company before hiring a replacement. Recently, researchers in the field of artificial intelligence have also addressed this problem due to the amount of data. With the aid of publicly available data and a decision tree approach, this study tackled the problem of staff attrition. The results of this work-in-progress study are encouraging, and subsequent work-studies will add more factors and test the model using data from a nearby supermarket.

3. **TITLE**: Employee Attrition System Using Tree Based Ensemble Method

YEAR: 2022

AUTHORS: Vimoli Mehta; Shrey Modi

DESCRIPTION: Around the world, employee churn has grown to be a serious issue. The loss of the best personnel is one of the major problems that company owners deal with in their organisations. A competent employee is always a benefit to the company, and when they leave, it can cause a number of issues, including financial losses, performance declines, and knowledge loss. In addition, compared to recruiting new personnel, hiring new workers is far more expensive, time-consuming, and labor-intensive. It takes a long time to find a new employee because it takes him months to get trained and get used to the surroundings. Therefore, commercial organisations must take advantage of emerging trends and technology that uses machine learning algorithms. Companies can reduce this loss by knowing the cause of staff churn before it happens. Using the dataset "IBM HR Analytics Employee Attrition Performance" and the tree-based Ensemble Machine Learning Model, this article offers a thorough analysis of employee attrition. The decision of an employee to quit the company is connected to a number of statistically important factors. To acquire the best outcomes from the currently available tree approaches, the study assesses the tree-based ensemble.

4. **TITLE**: Early Prediction of Employee Attrition using Data Mining Techniques

YEAR: 2019

AUTHORS: Sandeep Yadav; Aman Jain; Deepti Singh

DESCRIPTION: Take away our best 20 employees, and we [Microsoft] become a mediocre firm, according to a comment attributed to Bill Gates. Bill Gates' comment brought our attention to one of the main issues with employee churn in the workplace. Any firm must pay a hefty price for employee attrition (turnover), which could ultimately affect how efficiently it operates as a whole. According to CompData

Surveys, total turnover climbed from 15.1 percent to 18.5 percent over the previous five years. Finding a qualified and experienced employee is a difficult endeavour for any firm, and replacing such workers is even more difficult. In addition to raising the major cost of human resources (HR), this has an effect on an organization's market worth. Despite these realities, the literature that has contributed to numerous misunderstandings between HR and employees receives little attention. As a result, the purpose of this study is to present a methodology for predicting employee churn by applying classification algorithms to analyse the particular behaviours and qualities of the employee.

5. **TITLE**: Prediction of Employee Attrition Using data mining

YEAR: 2018

AUTHORS: R. Shiva Shankar; J. Rajanikanth; V.V. Sivaramaraju; K.V.S.S.R. Murthy

DESCRIPTION: Employee attrition has recently grown to be a significant issue in enterprises. Employee attrition is a significant problem for firms, particularly when skilled, technical, and critical people leave for another company that offers greater opportunities. As a result, replacing a skilled person costs money. As a result, we examine the frequent causes of employee attrition using data on both present and historical employees. On the human resource data, we employed well-known classification algorithms, such as Decision tree, Logistic Regression, SVM, KNN, Random Forest, and Naive Bayes, in order to reduce employee attrition. To do this, we apply the feature selection approach to the data and analyse the outcomes to stop staff attrition. The ability to foresee employee turnover helps businesses expand economically by lowering the cost of their human resources

2.3 PROBLEM STATEMENT DEFINITION

Problem Statement

Employees are the most important part of an organization. Successful employees meet deadlines, make sales, and build the brand through positive customer interactions. Employee attrition is a major cost to an organization and predicting such attritions is the most important requirement of the Human Resources department in many organizations. In this problem, our task is to predict the attrition rate of employees in an organization. Among all employee-related problems, employee attrition is one of the key problems in today's scenario despite the changes in the external environment. Attrition is said to be a gradual reduction in the number of employees through resignation, death, and retirement. A high attrition rate indicates that the employees have a lot of issues with the organization. Consequently, they'll only spread the bad word about the company. This will pose a huge risk to the company's reputation and make it difficult for the employer to find the right replacements. Every organization wants its valuable employees to be a part of its organization for a long period. Still, when many employees start leaving, it will be a concern for the organization. The key to success for any organization is attracting and retaining top talent. One of the key tasks is to determine which factors keep employees at the company and which prompt others to leave. It’s more cost-effective to keep the employees a company already has. • A company needs to maintain a pleasant working atmosphere to make their employees stay in that company for a longer period. To reduce the cost of attrition, organizations need to ensure that employees’ aspirations are met.

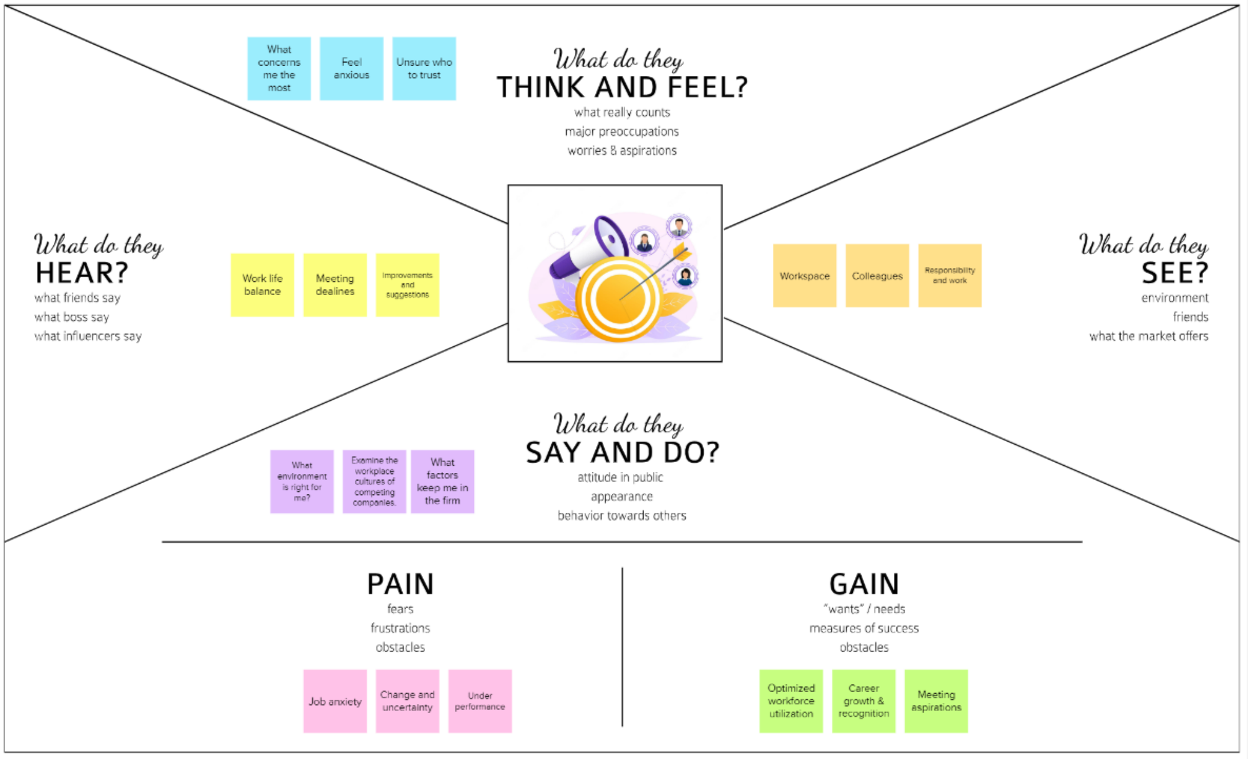
Business Model/Impact

• Organizations can use this tool to manage the team. • Reduction in Hiring Cost

3. IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

Team ID : PNT2022TMID27605



What do they
THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations

What concerns me the most

Feel anxious

Unsure who to trust

What do they
HEAR?
what friends say
what boss say
what influencers say

Work life balance

Meeting dealines

Improvements and suggestions

What do they
SEE?
environment
friends
what the market offers

Workspace

Colleagues

Responsibility and work

What do they
SAY AND DO?
attitude in public
appearance
behavior towards others

What environment is right for me?

Examine the workplace cultures of competing companies.

What factors keep me in the firm

PAIN
fears
frustrations
obstacles

Job anxiety

Change and uncertainty

Under performance

GAIN
"wants" / needs
measures of success
obstacles

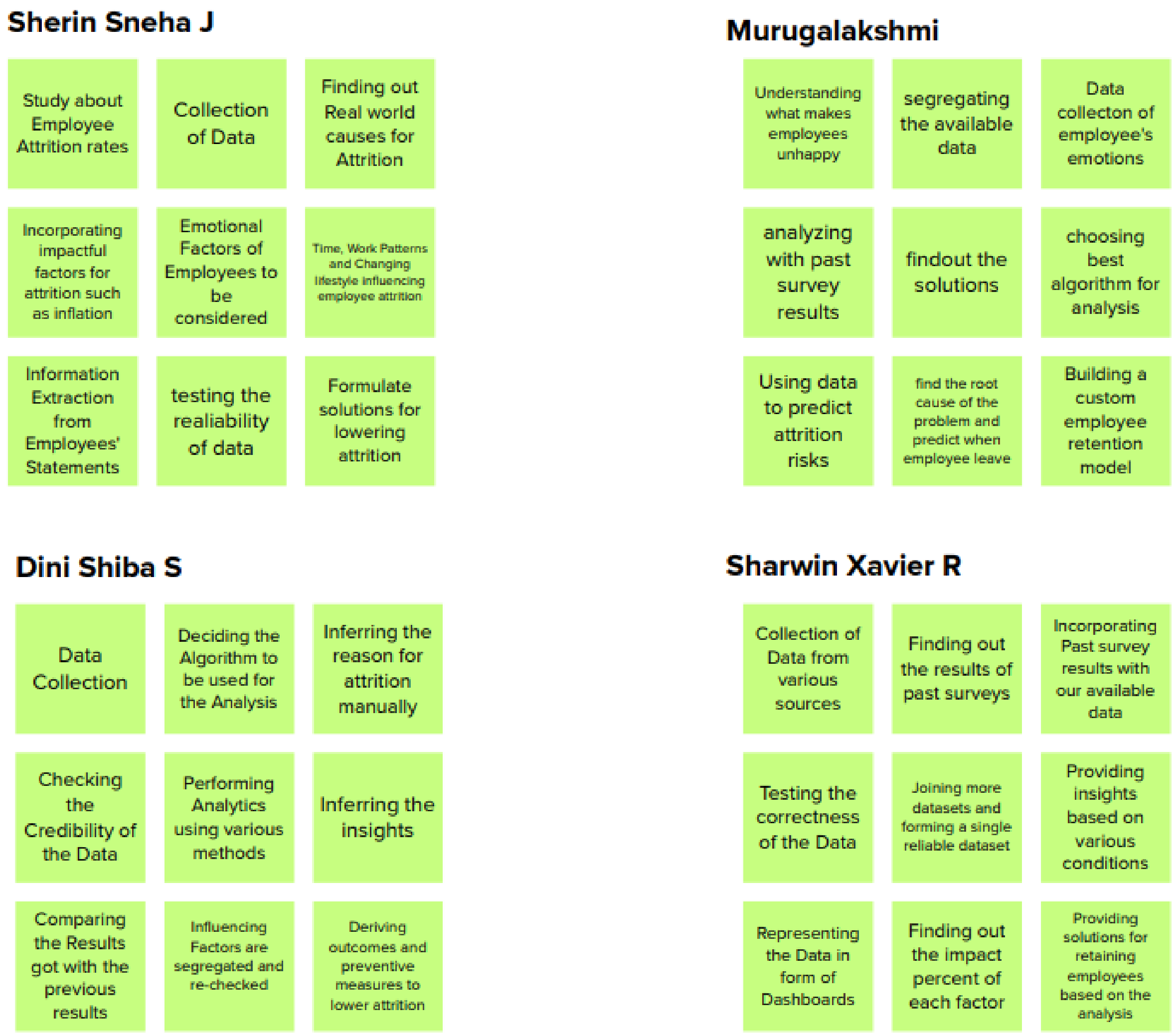
Optimized workforce utilization

Career growth & recognition

Meeting aspirations

3.2 Ideation and Brainstorming

1)Collection of Ideas



Sherin Sneha J

Study about Employee Attrition rates	Collection of Data	Finding out Real world causes for Attrition
Incorporating impactful factors for attrition such as inflation	Emotional Factors of Employees to be considered	Time, Work Patterns and Changing lifestyle influencing employee attrition
Information Extraction from Employees' Statements	testing the realiability of data	Formulate solutions for lowering attrition

Murugalakshmi

Understanding what makes employees unhappy	segregating the available data	Data collecton of employee's emotions
analyzing with past survey results	findout the solutions	choosing best algorithm for analysis
Using data to predict attrition risks	find the root cause of the problem and predict when employee leave	Building a custom employee retention model

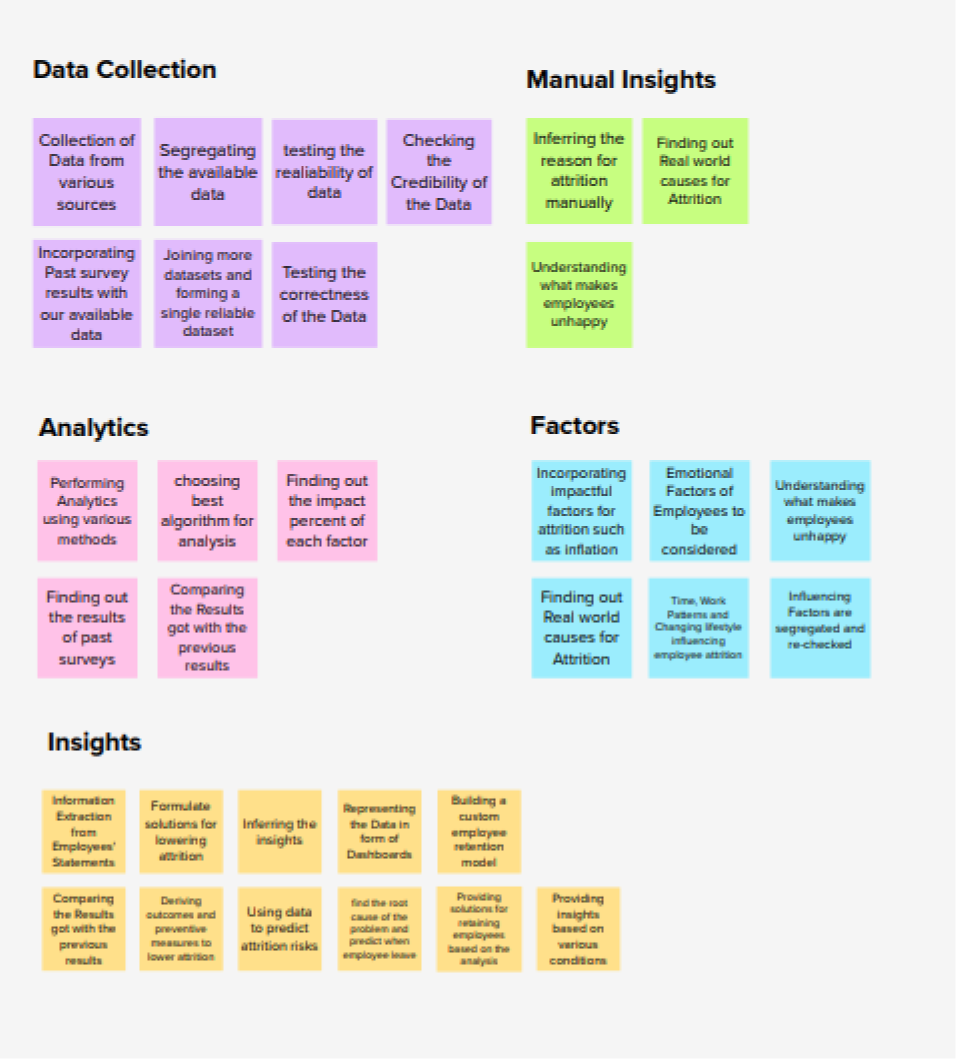
Dini Shiba S

Data Collection	Deciding the Algorithm to be used for the Analysis	Inferring the reason for attrition manually
Checking the Credibility of the Data	Performing Analytics using various methods	Inferring the insights
Comparing the Results got with the previous results	Influencing Factors are segregated and re-checked	Deriving outcomes and preventive measures to lower attrition

Sharwin Xavier R

Collection of Data from various sources	Finding out the results of past surveys	Incorporating Past survey results with our available data
Testing the correctness of the Data	Joining more datasets and forming a single reliable dataset	Providing insights based on various conditions
Representing the Data in form of Dashboards	Finding out the impact percent of each factor	Providing solutions for retaining employees based on the analysis

2) Grouping of ideas



Data Collection

- Collection of Data from various sources
- Segregating the available data
- testing the realiability of data
- Checking the Credibility of the Data
- Incorporating Past survey results with our available data
- Joining more datasets and forming a single reliable dataset
- Testing the correctness of the Data

Manual Insights

- Inferring the reason for attrition manually
- Finding out Real world causes for Attrition
- Understanding what makes employees unhappy

Analytics

- Performing Analytics using various methods
- choosing best algorithm for analysis
- Finding out the impact percent of each factor
- Finding out the results of past surveys
- Comparing the Results got with the previous results

Factors

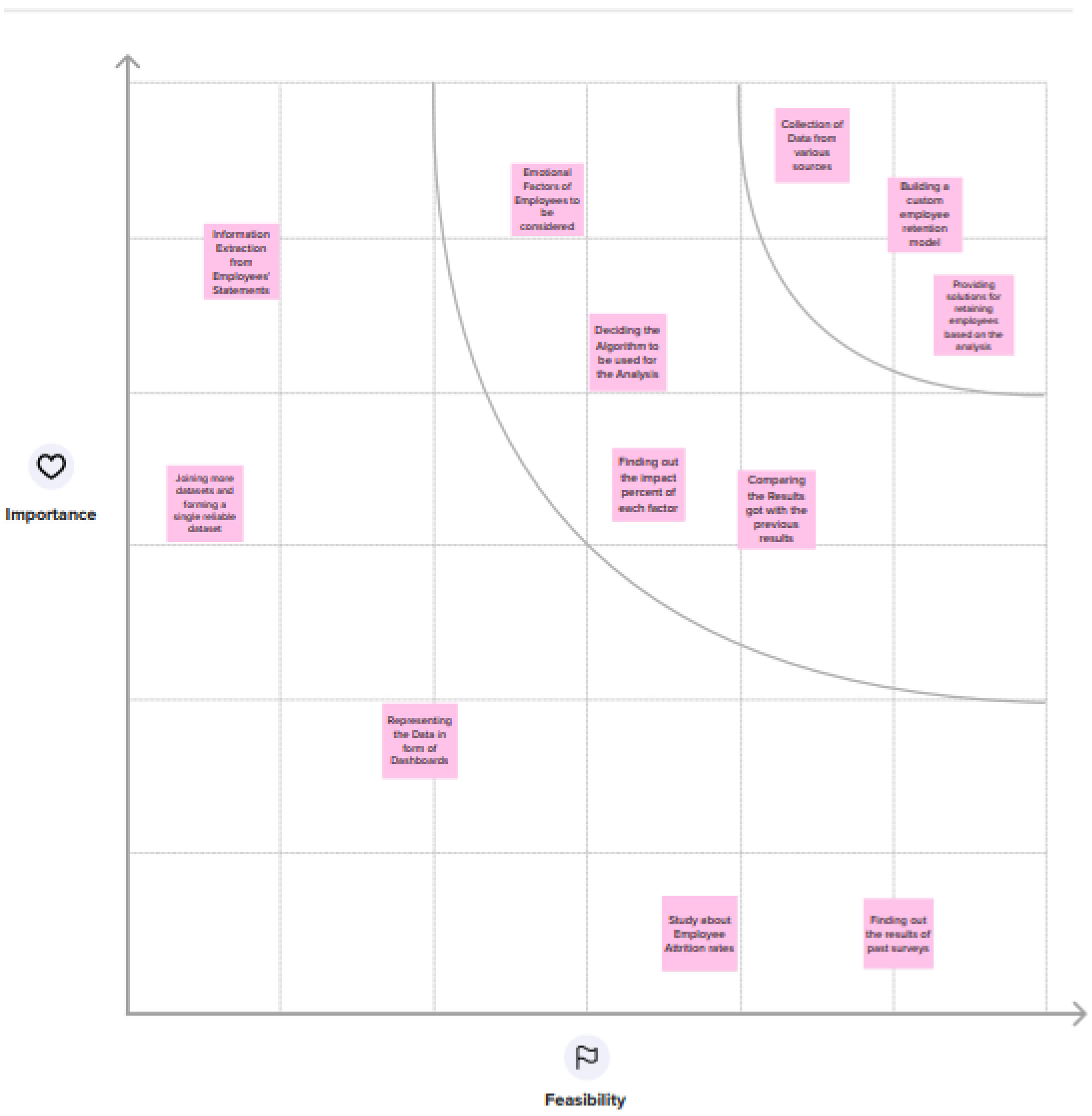
- Incorporating impactful factors for attrition such as inflation
- Emotional Factors of Employees to be considered
- Understanding what makes employees unhappy
- Finding out Real world causes for Attrition
- Time, Work Patterns and Changing lifestyle influencing employee attrition
- Influencing Factors are segregated and re-checked

Insights

- Information Extraction from Employees' Statements
- Formulate solutions for lowering attrition
- Inferring the insights
- Representing the Data in form of Dashboards
- Building a custom employee retention model
- Comparing the Results got with the previous results
- Deriving outcomes and preventive measures to lower attrition
- Using data to predict attrition risks
- find the root cause of the problem and predict when employee leave
- Providing solutions for retaining employees based on the analysis
- Providing insights based on various conditions

3) Prioritization

Prioritizing the Ideas

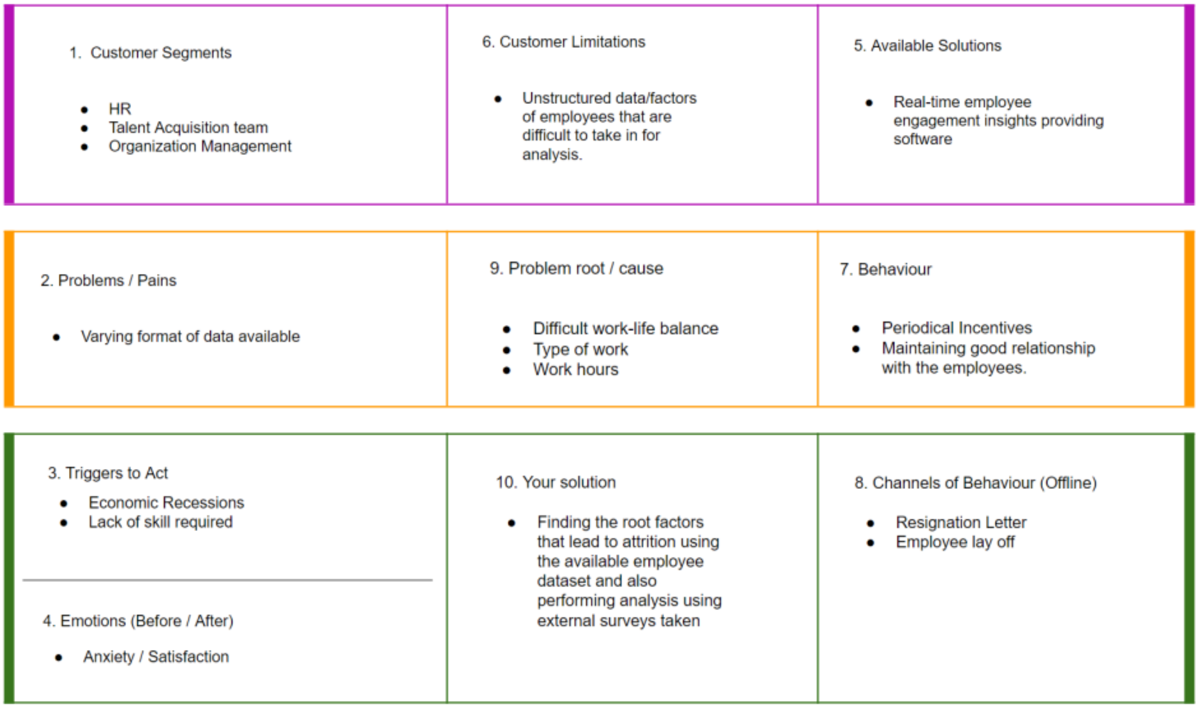

Information Extraction from Employees' Statements
Emotional Factors of Employees to be considered
Collection of Data from various sources
Building a custom employee retention model
Providing solutions for retaining employees based on the analysis
Deciding the Algorithm to be used for the Analysis
Importance
Joining more datasets and forming a single reliable dataset
Finding out the impact percent of each factor
Comparing the Results got with the previous results
Representing the Data in form of Dashboards
Study about Employee Attrition rates
Finding out the results of past surveys
Feasibility

3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Losing productive people would directly affect the growth of any organization. Given the data of employees working or resigned, the task is to analyse the data and find out the factors which lead the employees to leave the organization. This will help in retaining the employees and reduce the attrition rates.
2.	Idea / Solution description	Based on the results of the analysis of employee attrition, improving on the factors that lead the employees to leave the organization, maintaining good relationship with the employees and promoting personal career growth would have a positive impact on the retention of employees.
3.	Novelty / Uniqueness	Analysing the given data along with external survey results obtained from employees directly. This will help in improving the accuracy of the results.
4.	Social Impact / Customer Satisfaction	Reduction in the loss of valuable employees could be achieved. The Software directly benefits the customer by providing insights on the specific factors which need to be improved. The above factors subsequently lead to the growth of the company as well as customer satisfaction.
5.	Business Model (Revenue Model)	We plan to implement this application using a subscription-based model. Based on the number of employees, the subscription plans may differ.
6.	Scalability of the Solution	This software will be scalable for any organization as it runs only on the particular company’s employee dataset. Implementing this software with the help of cloud service providers helps in increasing the scalability.

Team ID : PNT2022TMID27605

3.4 Problem Solution Fit



1. Customer Segments

- HR
- Talent Acquisition team
- Organization Management

6. Customer Limitations

- Unstructured data/factors of employees that are difficult to take in for analysis.

5. Available Solutions

- Real-time employee engagement insights providing software

2. Problems / Pains

- Varying format of data available

9. Problem root / cause

- Difficult work-life balance
- Type of work
- Work hours

7. Behaviour

- Periodical Incentives
- Maintaining good relationship with the employees.

3. Triggers to Act

- Economic Recessions
- Lack of skill required

4. Emotions (Before / After)

- Anxiety / Satisfaction

10. Your solution

- Finding the root factors that lead to attrition using the available employee dataset and also performing analysis using external surveys taken

8. Channels of Behaviour (Offline)

- Resignation Letter
- Employee lay off

4 - Requirement Analysis

4.1 Functional Requirements

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form. Registration through Gmail. Registration through other accounts.
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Employee Data	Collection of Employee data through feedback forms and all recorded data about employees needs.

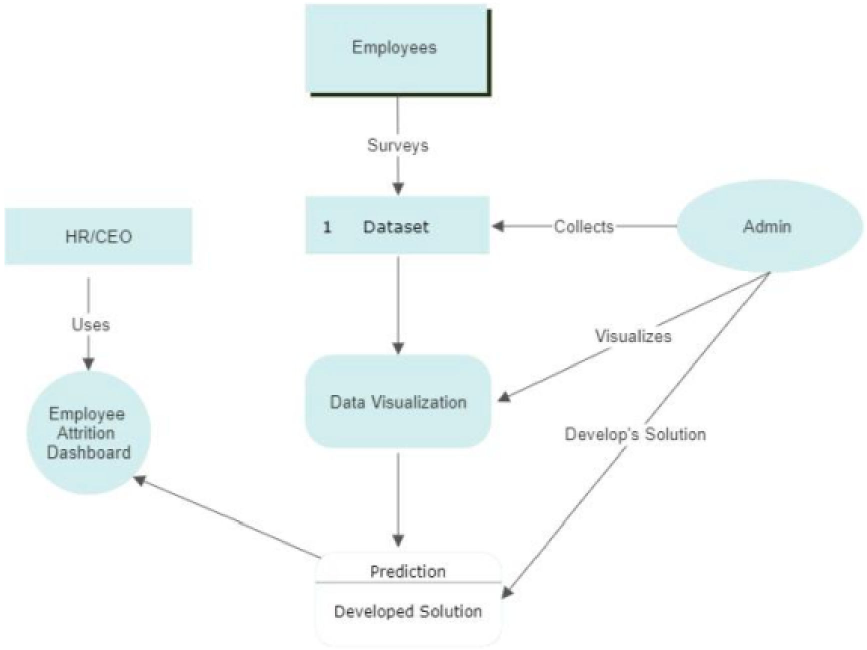
FR-4	Model	A model to train, test and predict the data.
FR-5	Storage	Data needs to be stored in an organized and Secured way.

4.2 Non Functional Requirements

FR No.	**Non-Functional Requirement**	**Description**
NFR-1	**Usability**	The organization should be able to use the application without any issues. The interface should be easy to operate and understand. The data processing speed should be effective.
NFR-2	**Security**	Privacy should be maintained over the employee and their feedback. It should be stored in a secure medium.
NFR-3	**Reliability**	The model accuracy should be high to identify the correct data. The Model should process the data without skipping any if single data for fair output
NFR-4	**Performance**	The model should perform effectively to give perfect output for growth of an organization.
NFR-5	**Availability**	The model should be available to users all time and should work efficiently. And at the same time gives good performance.
NFR-6	**Scalability**	The Model should stand out even a large dataset loaded to it and process that dataset in an effective way.

5 Project Design

5.1 Data Flow Diagrams



5.2 Solution & Technical Architecture

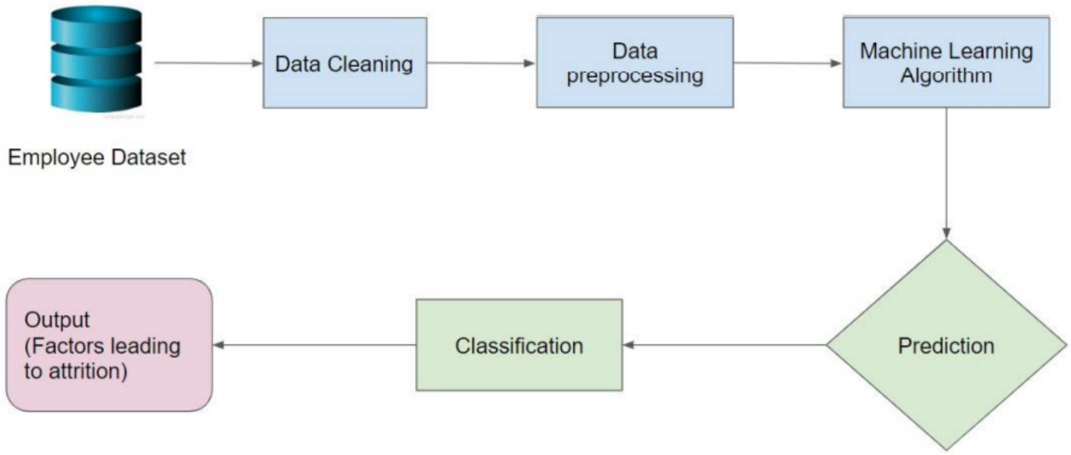
▪ The Process involves cleaning the input dataset first.

▪ Datasets are from various sources including the given one, plus the survey results obtained from employees

▪ Data preprocessing is then done to remove all unnecessary or unstructured data and also to make it structured ▪ After Pre-processing, using a machine learning algorithm (Supervised learning), we are classifying the common factors leading to attrition.

▪ Also, prediction of future attrition rates is projected with the available data ▪ Finally, the output is displayed to the user

Solution Architecture



5.3 User Stories

User Type	Functional Requirement (Epic)	User Number	User Story / Task Story	Acceptance criteria	Priority	Release
Customer (CEO)	Registration	USN-1	As a CEO, I can register for the application by entering my email, password, and confirming my password.	I can access my account /dashboard	High	Sprint-1
Customer (Employee)		USN-2	As an employee, I can register for the application by entering my mail, password,and confirming password.	I can access my account/dashboard	High	Sprint-1
		USN-3	As a user, I can register for the application	I can register & dashboard with login	Medium	Sprint-2
Customer (CEO)	Login	USN-4	As a user, I can log into the application by entering email & password	I can access my account/ dashboard	High	Sprint-3

Customer (Employee)		USN-5	As a user, I can log into the application by entering email and password.	I can access my account /dashboard	High	Sprint-3
CEO	Dashboard	USN-6	As a CEO, I can use the predict button to know which factor keeps the employee at the company and which prompts others to leave.	I can view the visual chart.	HIgh	Sprint-4
Employee		USN-7	As an employee of the organization, I can view, fill and submit the survey form that is displayed.	I can see the acknowledgement message for submitting the survey	High	Sprint-4

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Sprint-1	Dashboard	USN-1	As a user, I give the details of the employees working in our organization for the attrition detail.	5	High	Sherin Sneha J, Dini Shiba S, Murugalakshmi V, Sharwin Xavier R
Sprint-1		USN-2	As an Analyst, I will check the dataset and perform exploratory data analysis in Cognos Analytics	3	High	Sherin Sneha J, Dini Shiba S, Murugalakshmi V, Sharwin Xavier R
Sprint-2	Report	USN-3	As a user, I want Simpler limited number of visualizations that report a particular event	2	Low	Sherin Sneha J, Dini Shiba S, Murugalakshmi V, Sharwin Xavier R
Sprint-2		USN-4	As an Analyst, I will use Cognos Analytics to generate a report	3	Medium	Sherin Sneha J, Dini Shiba S, Murugalakshmi V, Sharwin Xavier R
Sprint-3	Story	USN-5	As a user, I can only understand the Analysis in animated presentation of	3	Medium	Sherin Sneha J, Dini Shiba S, Murugalakshmi V,

			dataset			Sharwin Xavier R
Sprint-3		USN-6	As an Analyst, I use Cognos Analytics to create an animated presentation (Story) of the dataset	3	Medium	Sherin Sneha J, Dini Shiba S, Murugalakshmi V, Sharwin Xavier R
Sprint-4	Predictive Analysis	USN-7	As a user, I want to predict the attrition rate of the company from the dataset	5	Medium	Sherin Sneha J, Dini Shiba S, S,Murugalakshmi V Sharwin Xavier R
Sprint-4		USN-8	As an Analyst, I will perform Prediction Analysis by utilizing various libraries in python	3	High	Sherin Sneha J, Dini Shiba S, Murugalakshmi V, Sharwin Xavier R

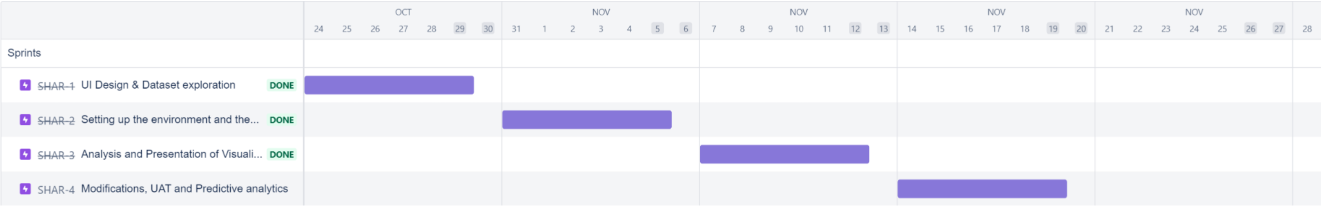
6.2 Sprint Delivery Schedule

`Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	5	6 Days	24 Oct 2022	29 Oct 2022	5	29 Oct 2022
Sprint-2	5	6 Days	31 Oct 2022	05 Nov 2022	5	05 Nov 2022
Sprint-3	5	6 Days	07 Nov 2022	12 Nov 2022	5	12 Nov 2022
Sprint-4	5	6 Days	14 Nov 2022	19 Nov 2022	5	19 Nov 2022

We have a 6-day sprint duration, and the velocity of the team is 5 (points per sprint). To calculate the team’s average velocity (AV) per iteration unit (story points per day)

AV = SPRINT DURATION/VELOCITY
= 6/5
= 1.2

6.3 Reports from JIRA



Sprints

~~SHAR-1~~ UI Design & Dataset exploration DONE

~~SHAR-2~~ Setting up the environment and the... DONE

~~SHAR-3~~ Analysis and Presentation of Visuali... DONE

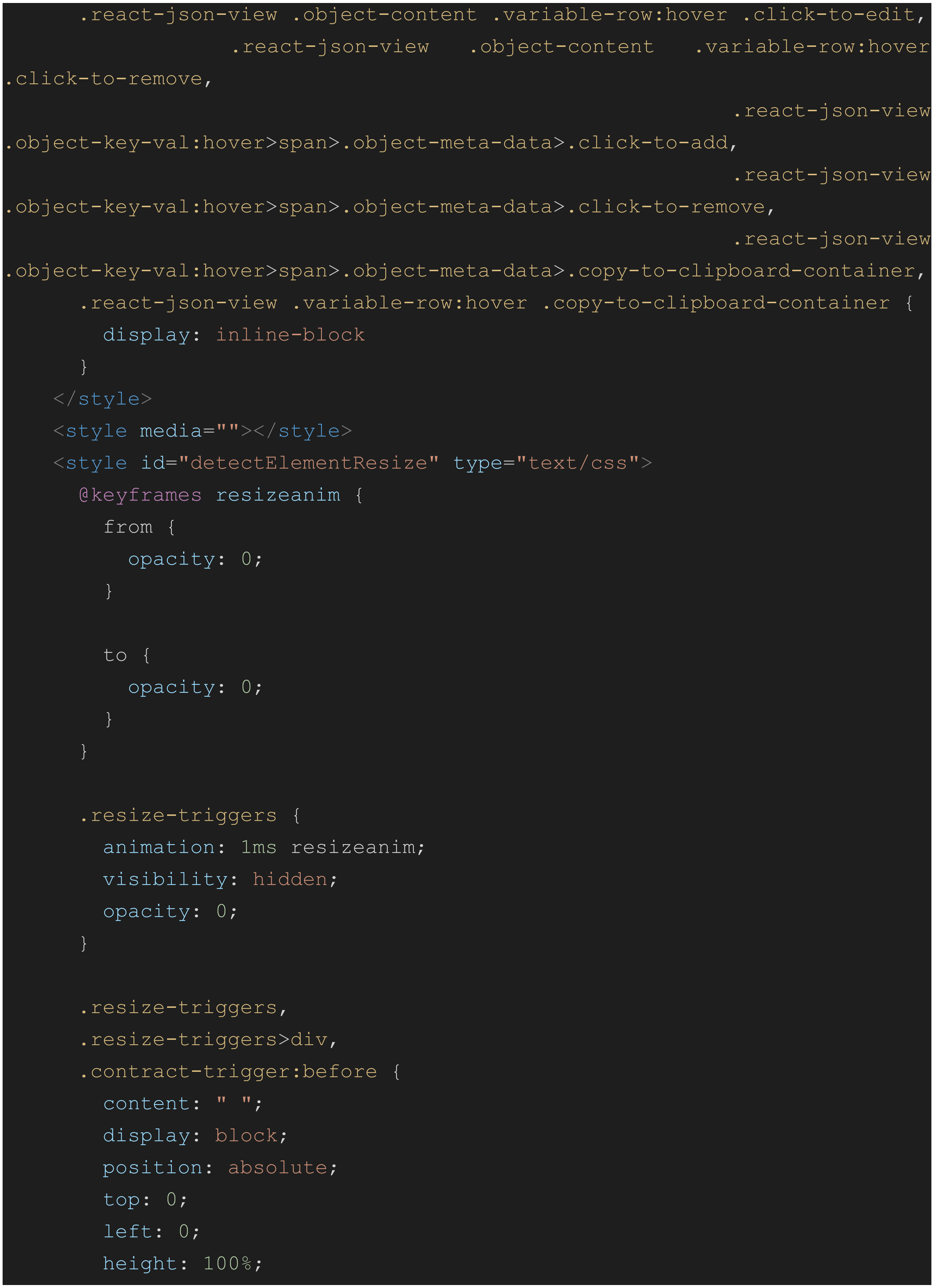
SHAR-4 Modifications, UAT and Predictive analytics

7. CODING & SOLUTION

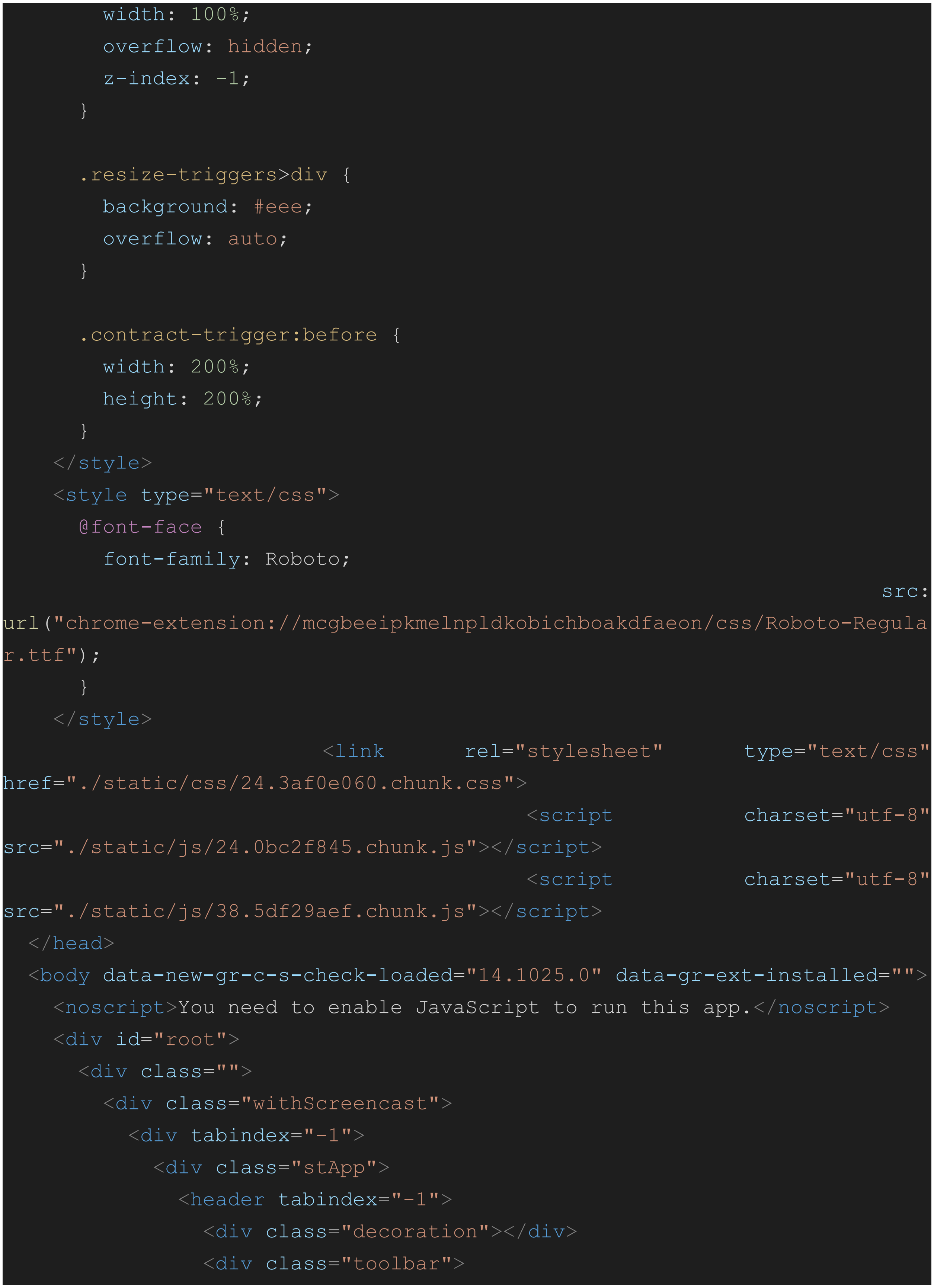
7.1 - Feature 1 - IBM Cognos Dashboard Representation in the web application and UI

Code

```
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
content="width=device-width,initial-scale=1,shrink-to-fit=no">
    <link rel="shortcut icon" href="./favicon.png">
    <title>app · Streamlit</title>
    <script type="text/javascript" async=""
src="https://cdn.segment.com/analytics.js/v1/iCkMy7ymtJ9qYzQRXkQpnAJEq7D4N
yMU/analytics.min.js"></script>
    <script src="./vendor/viz/viz-1.8.0.min.js"
type="javascript/worker"></script>
    <link href="./static/css/8.3bee90c5.chunk.css" rel="stylesheet">
    <link href="./static/css/main.739a4783.chunk.css" rel="stylesheet">
    <style type="text/css">
      .react-json-view .copy-to-clipboard-container {
        vertical-align: top;
        display: none
      }

      .react-json-view .click-to-add,
      .react-json-view .click-to-edit,
      .react-json-view .click-to-remove {
        display: none
      }
```



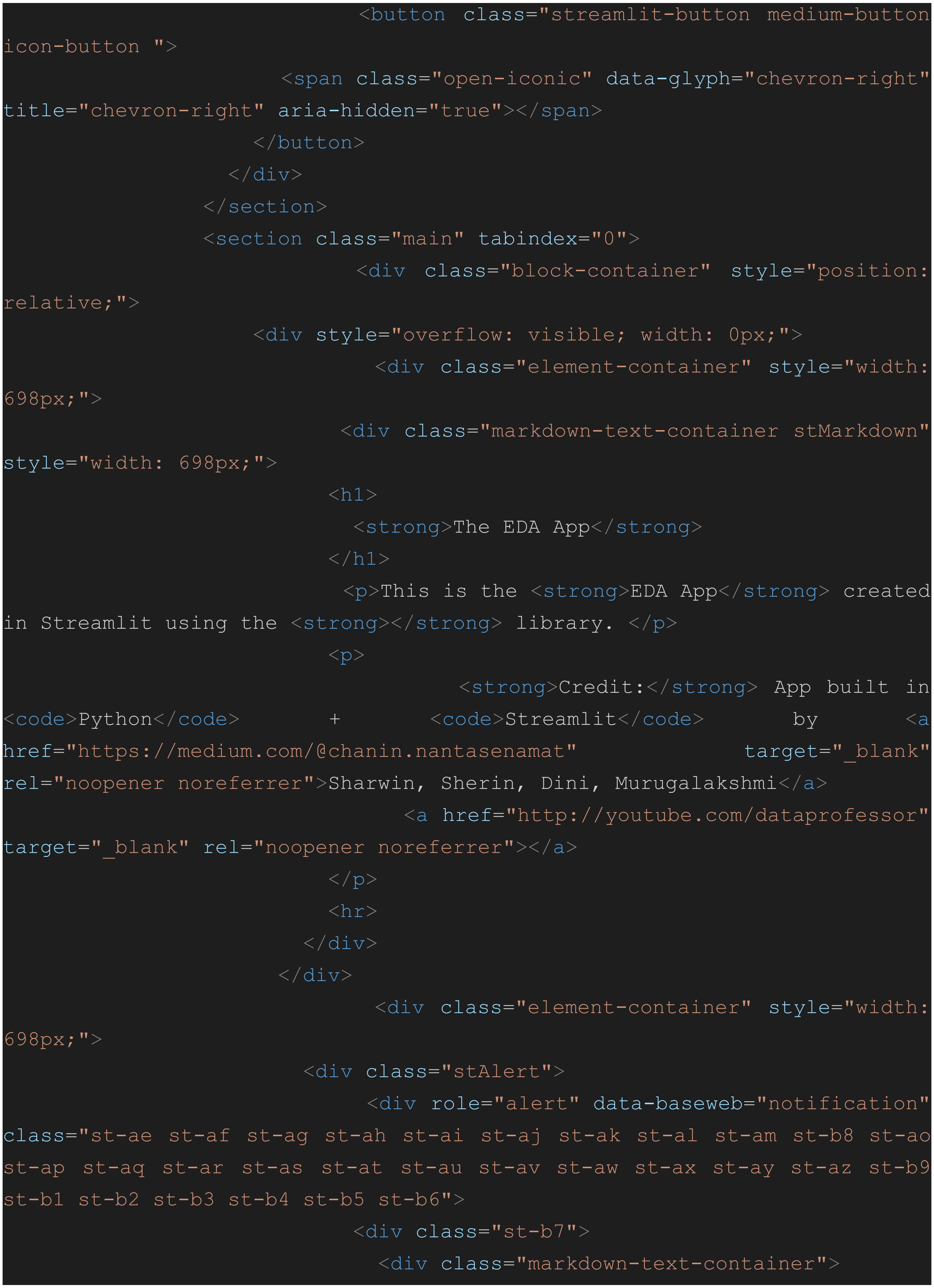
```
      .react-json-view .object-content .variable-row:hover .click-to-edit,
                  .react-json-view   .object-content   .variable-row:hover
.click-to-remove,
                                                           .react-json-view
.object-key-val:hover>span>.object-meta-data>.click-to-add,
                                                           .react-json-view
.object-key-val:hover>span>.object-meta-data>.click-to-remove,
                                                           .react-json-view
.object-key-val:hover>span>.object-meta-data>.copy-to-clipboard-container,
      .react-json-view .variable-row:hover .copy-to-clipboard-container {
        display: inline-block
      }
    </style>
    <style media=""></style>
    <style id="detectElementResize" type="text/css">
      @keyframes resizeanim {
        from {
          opacity: 0;
        }

        to {
          opacity: 0;
        }
      }

      .resize-triggers {
        animation: 1ms resizeanim;
        visibility: hidden;
        opacity: 0;
      }

      .resize-triggers,
      .resize-triggers>div,
      .contract-trigger:before {
        content: " ";
        display: block;
        position: absolute;
        top: 0;
        left: 0;
        height: 100%;
```



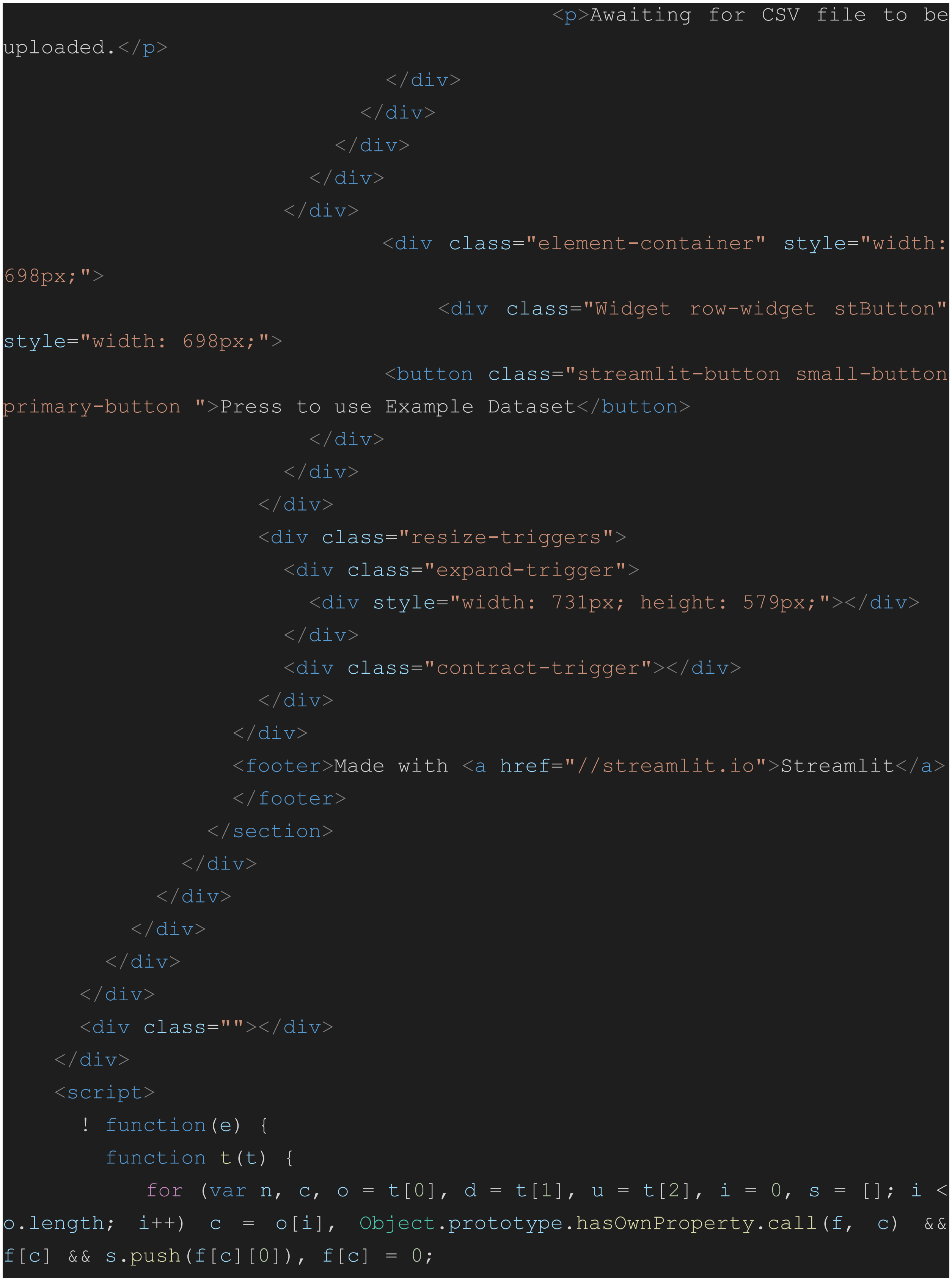
```
        width: 100%;
        overflow: hidden;
        z-index: -1;
      }

      .resize-triggers>div {
        background: #eee;
        overflow: auto;
      }

      .contract-trigger:before {
        width: 200%;
        height: 200%;
      }
    </style>
    <style type="text/css">
      @font-face {
        font-family: Roboto;
                                                                                 src:
url("chrome-extension://mcgbeeipkmelnpldkobichboakdfaeon/css/Roboto-Regula
r.ttf");
      }
    </style>
                          <link      rel="stylesheet"      type="text/css"
href="./static/css/24.3af0e060.chunk.css">
                                          <script          charset="utf-8"
src="./static/js/24.0bc2f845.chunk.js"></script>
                                          <script          charset="utf-8"
src="./static/js/38.5df29aef.chunk.js"></script>
  </head>
  <body data-new-gr-c-s-check-loaded="14.1025.0" data-gr-ext-installed="">
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root">
      <div class="">
        <div class="withScreencast">
          <div tabindex="-1">
            <div class="stApp">
              <header tabindex="-1">
                <div class="decoration"></div>
                <div class="toolbar">
```

```
                                 <span id="MainMenu" aria-haspopup="true"
aria-expanded="false">
                             <button class="streamlit-button medium-button
icon-button ">
                               <span class="open-iconic" data-glyph="menu"
title="menu" aria-hidden="true"></span>
                    </button>
                  </span>
                </div>
              </header>
              <div class="reportview-container">
                <section class="sidebar">
                  <div class="sidebar-content">
                    <div class="sidebar-close">
                             <button class="streamlit-button medium-button
icon-button ">
                        <span class="open-iconic" data-glyph="x" title="x"
aria-hidden="true"></span>
                      </button>
                    </div>
                            <div class="block-container" style="position:
relative;">
                      <div style="overflow: visible; width: 0px;">
                              <div class="element-container" style="width:
304px;">
                           <div class="markdown-text-container stMarkdown"
style="width: 304px;">
                            <h2>1. Upload your CSV data</h2>
                          </div>
                        </div>
                              <div class="element-container" style="width:
304px;">
                          <div class="Widget stFileUploader">
                            <label>Upload your input CSV file</label>
                                                    <section tabindex="0"
class="fileUploadDropzone st-bb st-bc st-bd st-be st-bf st-bg st-b4">
                                          <input accept=".csv" type="file"
autocomplete="off" tabindex="-1" style="display: none;">
                              <div class="st-bh st-bg st-b4">
```

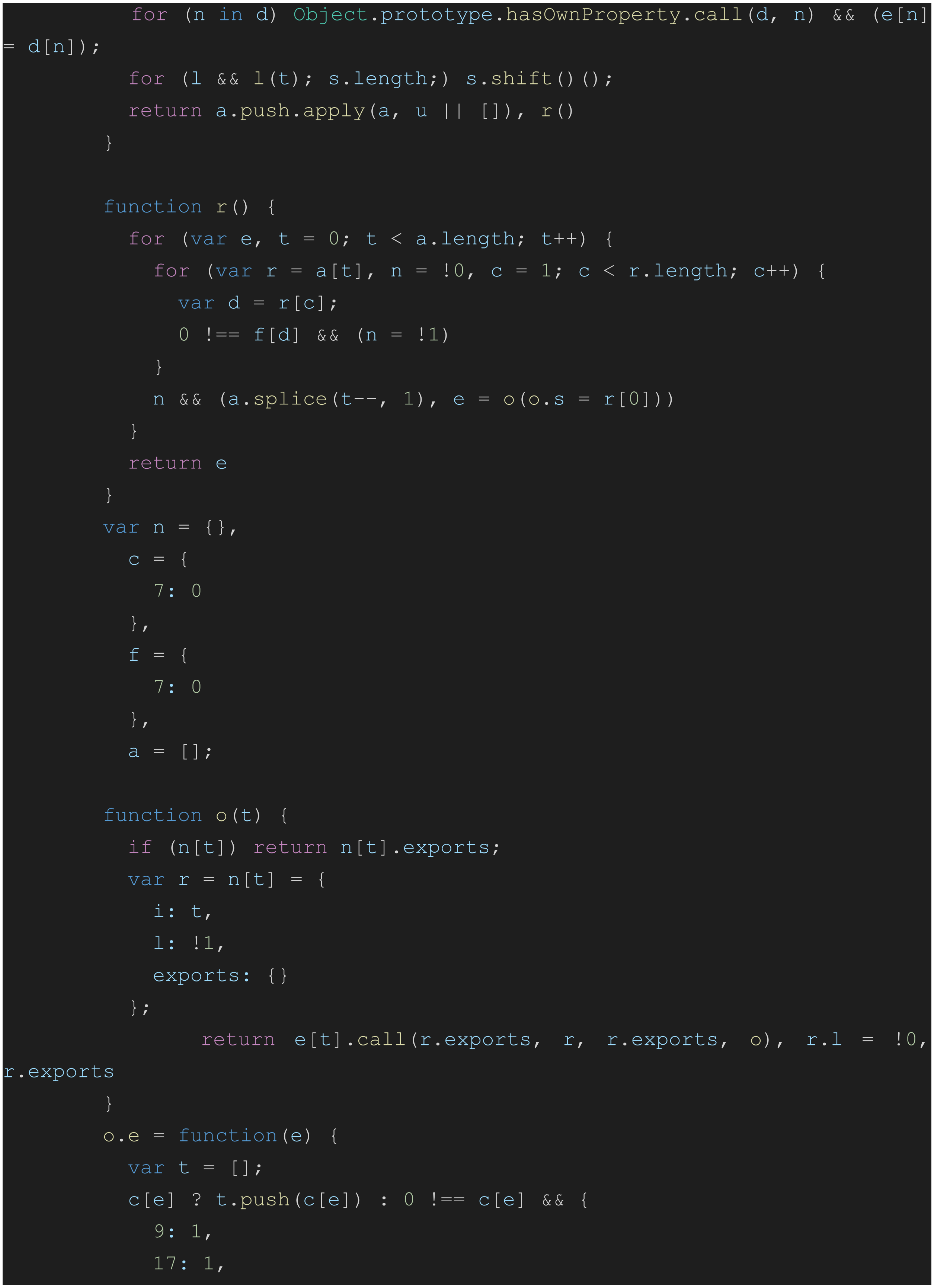
```
                                  <i class="material-outlined-icons icon-lg
fileUploaderIcon st-bi st-bj" aria-hidden="true">cloud_upload</i>
                                <div class="st-b4 st-bk">
                                     <span class="st-bl">Drag and drop file
here</span>
                                      <small class="st-bm st-bn st-bo st-bp
st-b4 st-bg">Limit 200MB per file • CSV</small>
                                </div>
                              </div>
                              <button class="streamlit-button small-button
primary-button ">Browse files</button>
                            </section>
                               <div class="uploadedFiles st-bq st-br st-bs
st-bt st-bu st-bv">
                              <ul></ul>
                            </div>
                          </div>
                        </div>
                              <div class="element-container" style="width:
304px;">
                            <div class="markdown-text-container stMarkdown"
style="width: 304px;">
                            <p>
                                                                         <a
href="https://www.kaggle.com/datasets/pavansubhasht/ibm-hr-analytics-attri
tion-dataset?select=WA_Fn-UseC_-HR-Employee-Attrition.csv" target="_blank"
rel="noopener noreferrer">Example CSV input file</a>
                            </p>
                          </div>
                        </div>
                      </div>
                      <div class="resize-triggers">
                        <div class="expand-trigger">
                          <div style="width: 305px; height: 262px;"></div>
                        </div>
                        <div class="contract-trigger"></div>
                      </div>
                    </div>
                  </div>
                  <div class="sidebar-collapse-control">
```



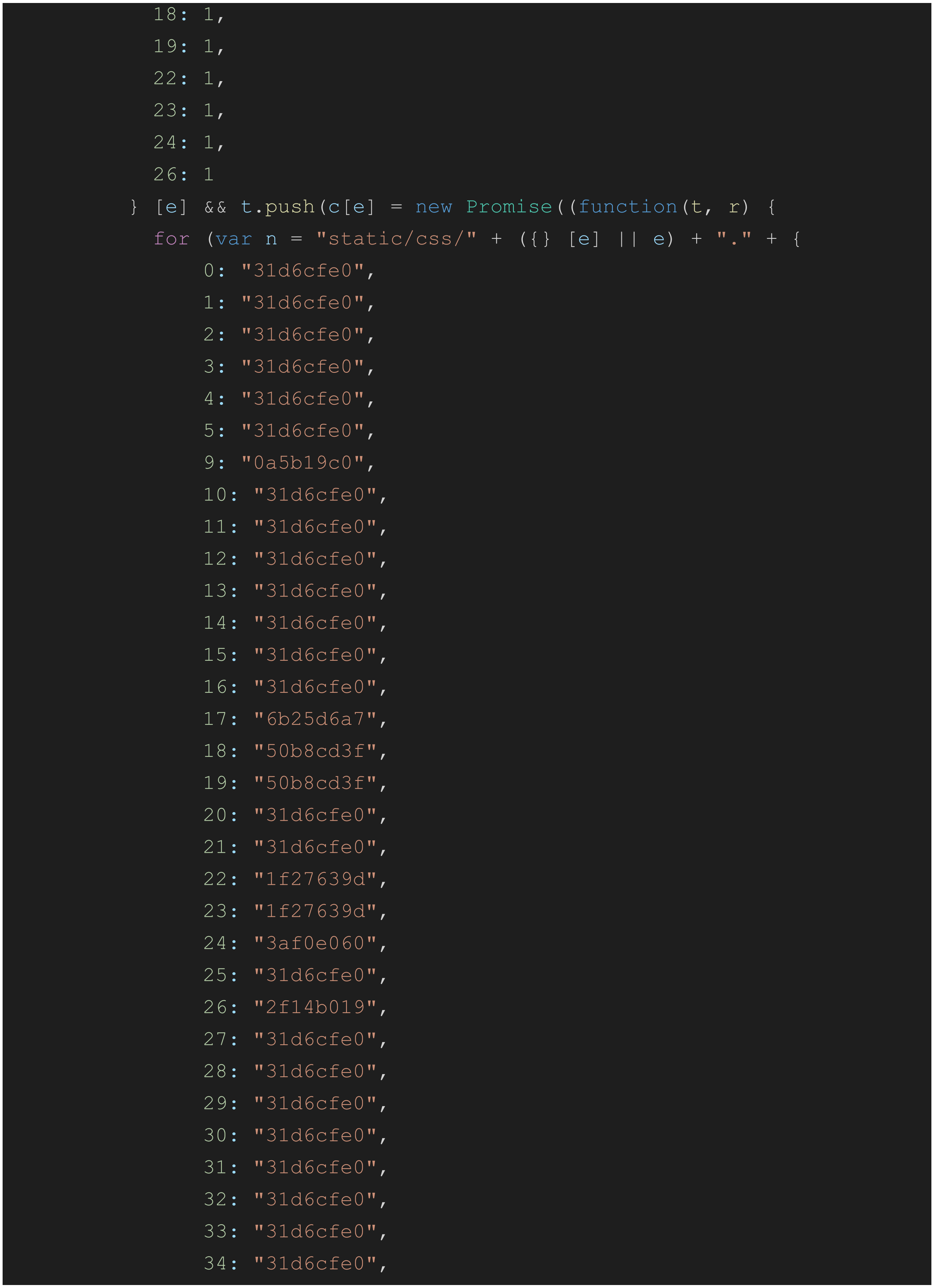
```
                            <button class="streamlit-button medium-button
icon-button ">
                      <span class="open-iconic" data-glyph="chevron-right"
title="chevron-right" aria-hidden="true"></span>
                    </button>
                  </div>
                </section>
                <section class="main" tabindex="0">
                            <div class="block-container" style="position:
relative;">
                    <div style="overflow: visible; width: 0px;">
                              <div class="element-container" style="width:
698px;">
                           <div class="markdown-text-container stMarkdown"
style="width: 698px;">
                          <h1>
                            <strong>The EDA App</strong>
                          </h1>
                           <p>This is the <strong>EDA App</strong> created
in Streamlit using the <strong></strong> library. </p>
                          <p>
                                    <strong>Credit:</strong> App built in
<code>Python</code> + <code>Streamlit</code> by <a
href="https://medium.com/@chanin.nantasenamat" target="_blank"
rel="noopener noreferrer">Sharwin, Sherin, Dini, Murugalakshmi</a>
                                <a href="http://youtube.com/dataprofessor"
target="_blank" rel="noopener noreferrer"></a>
                          </p>
                          <hr>
                        </div>
                      </div>
                              <div class="element-container" style="width:
698px;">
                        <div class="stAlert">
                             <div role="alert" data-baseweb="notification"
class="st-ae st-af st-ag st-ah st-ai st-aj st-ak st-al st-am st-b8 st-ao
st-ap st-aq st-ar st-as st-at st-au st-av st-aw st-ax st-ay st-az st-b9
st-b1 st-b2 st-b3 st-b4 st-b5 st-b6">
                            <div class="st-b7">
                              <div class="markdown-text-container">
```



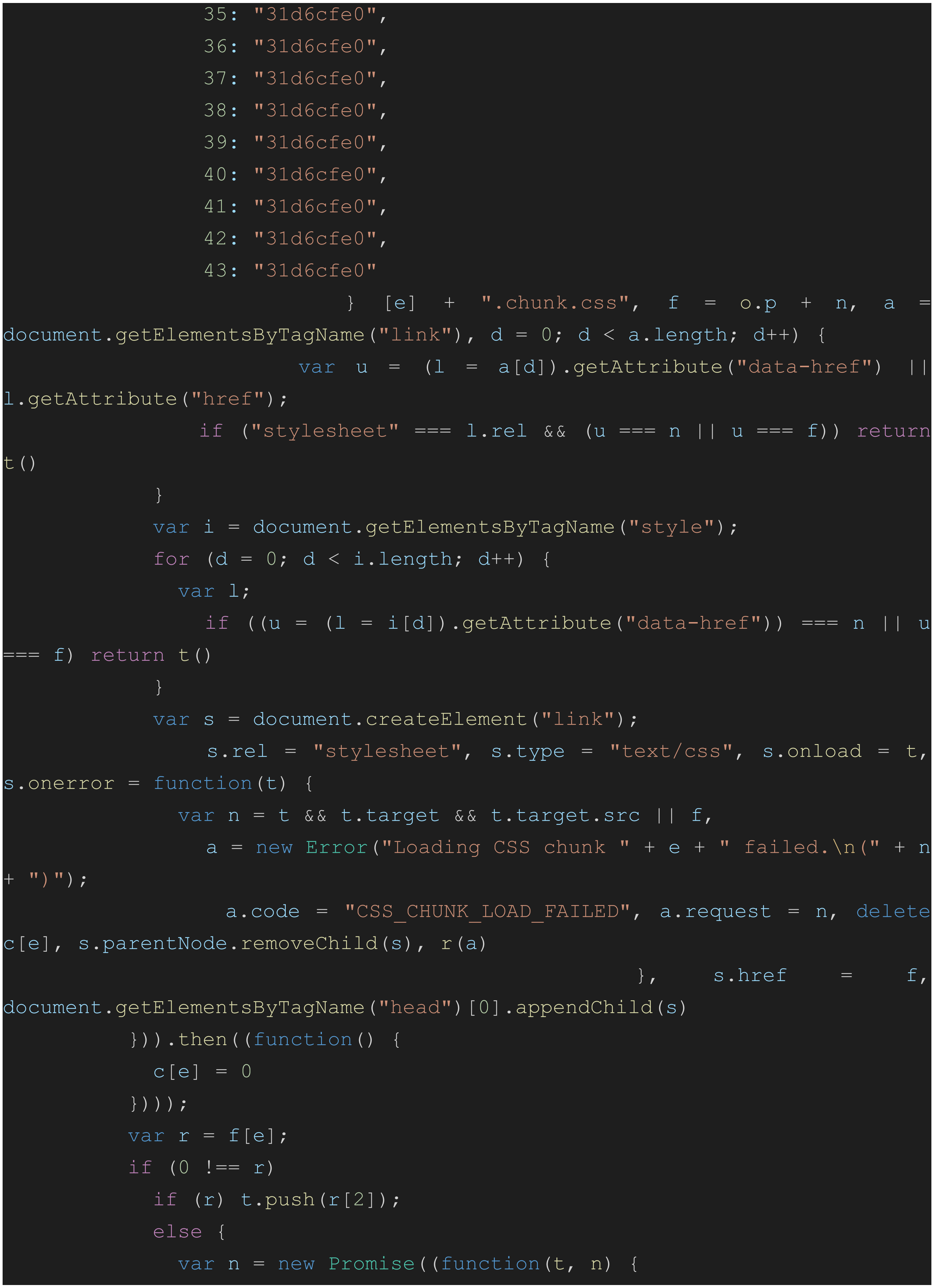
```
                                <p>Awaiting for CSV file to be
uploaded.</p>
                              </div>
                            </div>
                          </div>
                        </div>
                      </div>
                      <div class="element-container" style="width:
698px;">
                        <div class="Widget row-widget stButton"
style="width: 698px;">
                          <button class="streamlit-button small-button
primary-button ">Press to use Example Dataset</button>
                        </div>
                      </div>
                    </div>
                    <div class="resize-triggers">
                      <div class="expand-trigger">
                        <div style="width: 731px; height: 579px;"></div>
                      </div>
                      <div class="contract-trigger"></div>
                    </div>
                  </div>
                  <footer>Made with <a href="//streamlit.io">Streamlit</a>
                  </footer>
                </section>
              </div>
            </div>
          </div>
        </div>
      </div>
      <div class=""></div>
    </div>
    <script>
      ! function(e) {
        function t(t) {
          for (var n, c, o = t[0], d = t[1], u = t[2], i = 0, s = []; i <
o.length; i++) c = o[i], Object.prototype.hasOwnProperty.call(f, c) &&
f[c] && s.push(f[c][0]), f[c] = 0;
```



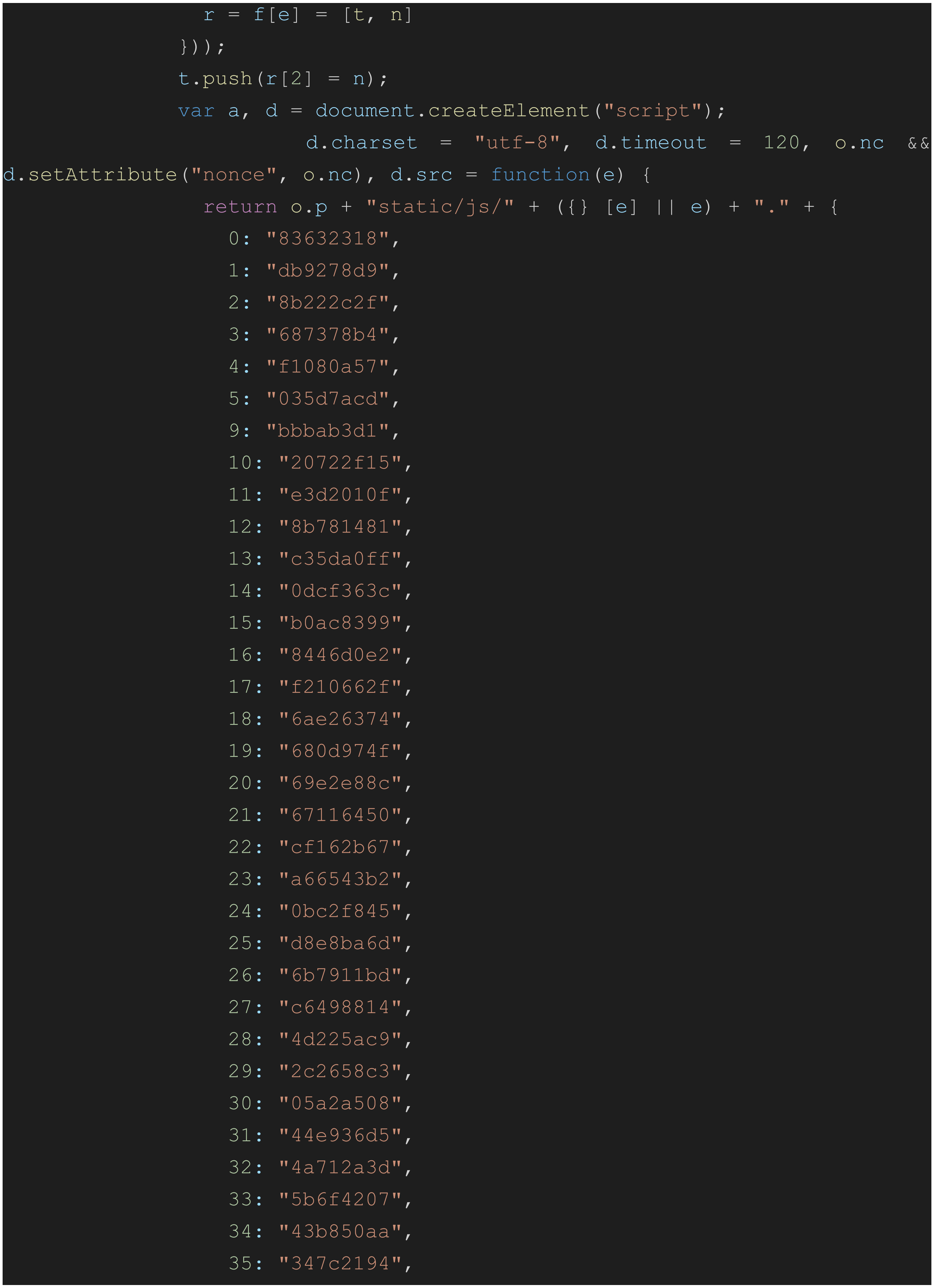
```
          for (n in d) Object.prototype.hasOwnProperty.call(d, n) && (e[n]
= d[n]);
          for (l && l(t); s.length;) s.shift()();
          return a.push.apply(a, u || []), r()
        }

        function r() {
          for (var e, t = 0; t < a.length; t++) {
            for (var r = a[t], n = !0, c = 1; c < r.length; c++) {
              var d = r[c];
              0 !== f[d] && (n = !1)
            }
            n && (a.splice(t--, 1), e = o(o.s = r[0]))
          }
          return e
        }
        var n = {},
          c = {
            7: 0
          },
          f = {
            7: 0
          },
          a = [];

        function o(t) {
          if (n[t]) return n[t].exports;
          var r = n[t] = {
            i: t,
            l: !1,
            exports: {}
          };
                return e[t].call(r.exports, r, r.exports, o), r.l = !0,
r.exports
        }
        o.e = function(e) {
          var t = [];
          c[e] ? t.push(c[e]) : 0 !== c[e] && {
            9: 1,
            17: 1,
```



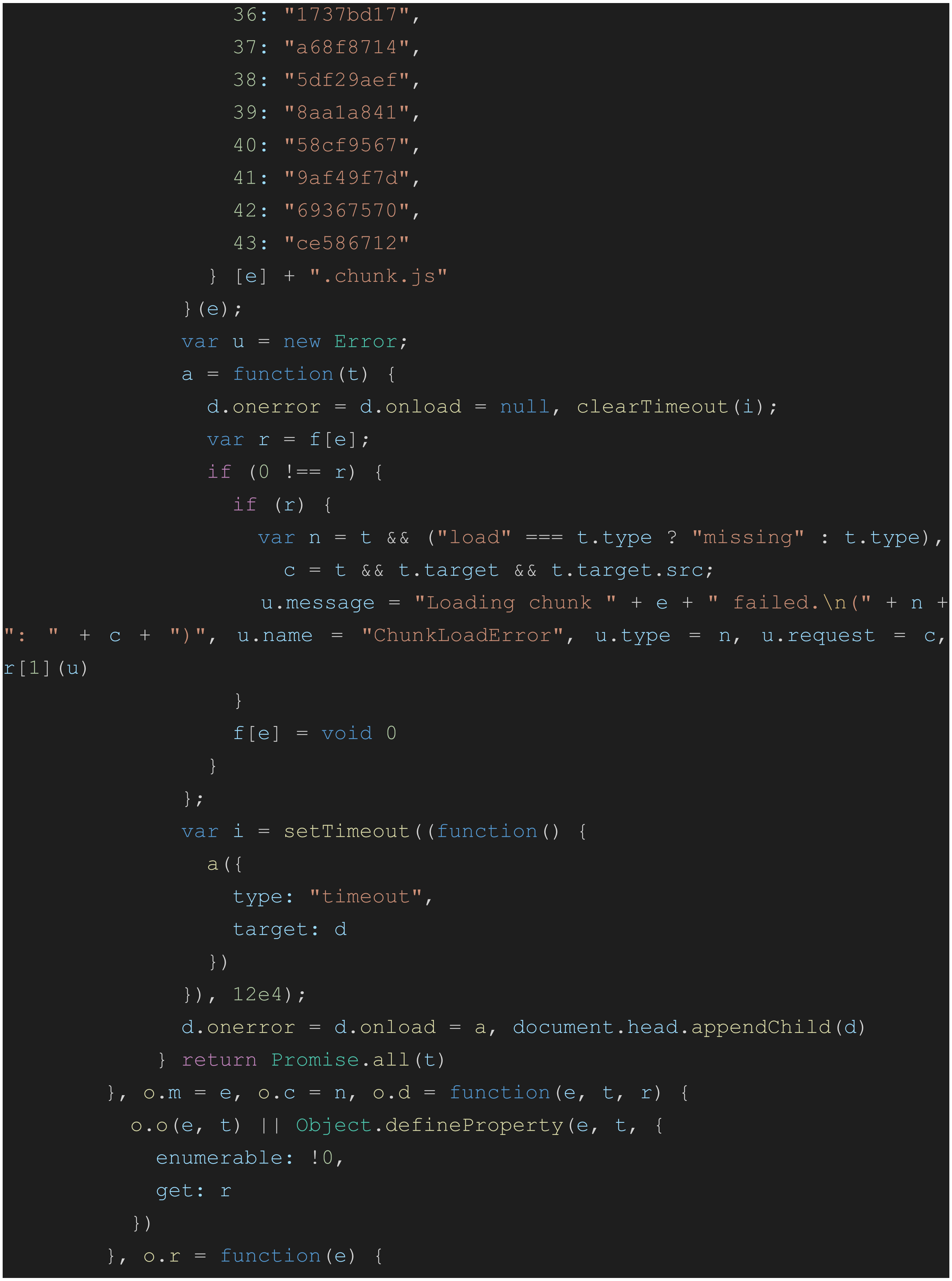
```
  18: 1,
  19: 1,
  22: 1,
  23: 1,
  24: 1,
  26: 1
} [e] && t.push(c[e] = new Promise((function(t, r) {
  for (var n = "static/css/" + ({} [e] || e) + "." + {
      0: "31d6cfe0",
      1: "31d6cfe0",
      2: "31d6cfe0",
      3: "31d6cfe0",
      4: "31d6cfe0",
      5: "31d6cfe0",
      9: "0a5b19c0",
      10: "31d6cfe0",
      11: "31d6cfe0",
      12: "31d6cfe0",
      13: "31d6cfe0",
      14: "31d6cfe0",
      15: "31d6cfe0",
      16: "31d6cfe0",
      17: "6b25d6a7",
      18: "50b8cd3f",
      19: "50b8cd3f",
      20: "31d6cfe0",
      21: "31d6cfe0",
      22: "1f27639d",
      23: "1f27639d",
      24: "3af0e060",
      25: "31d6cfe0",
      26: "2f14b019",
      27: "31d6cfe0",
      28: "31d6cfe0",
      29: "31d6cfe0",
      30: "31d6cfe0",
      31: "31d6cfe0",
      32: "31d6cfe0",
      33: "31d6cfe0",
      34: "31d6cfe0",
```



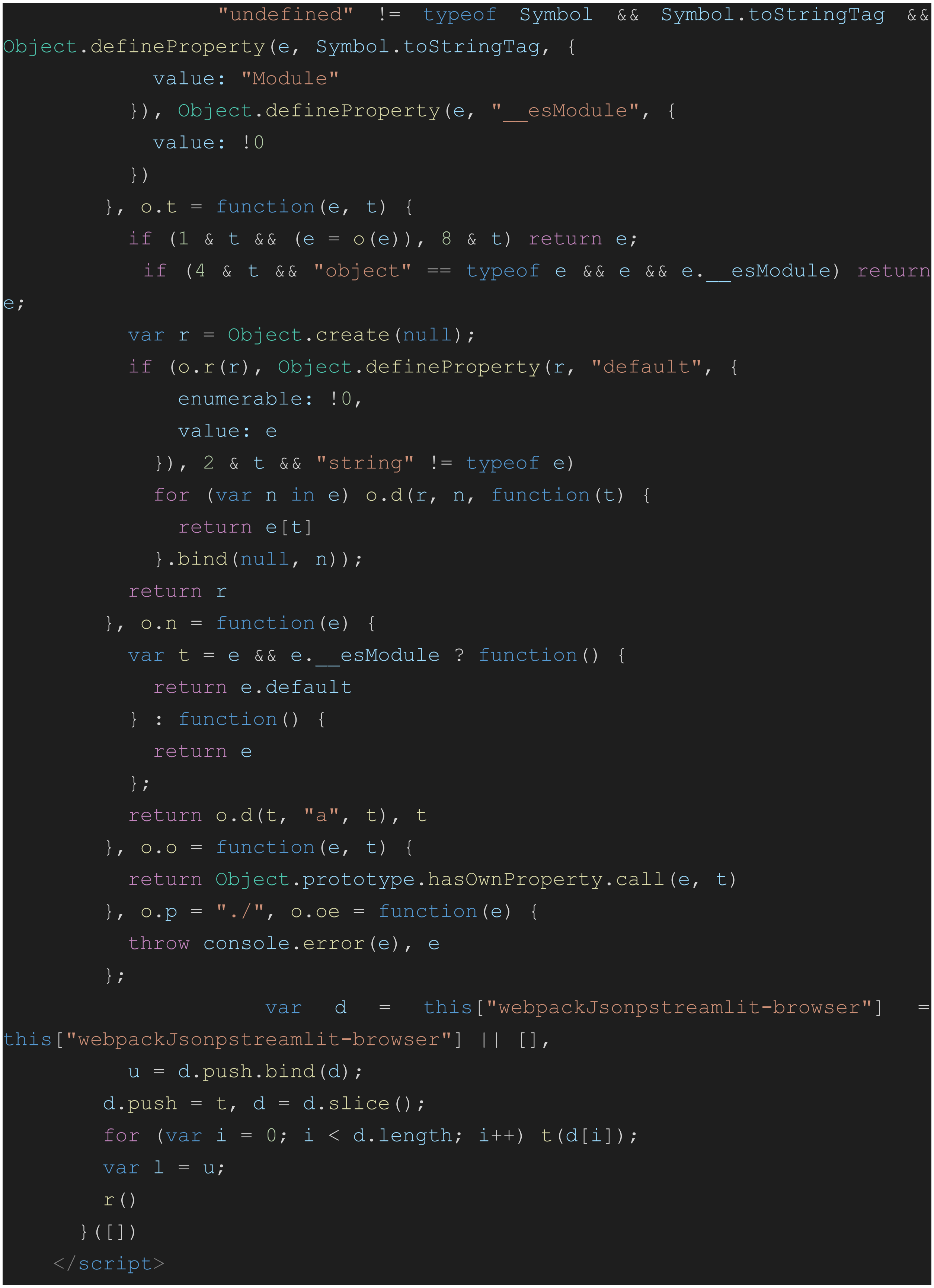
```
                35: "31d6cfe0",
                36: "31d6cfe0",
                37: "31d6cfe0",
                38: "31d6cfe0",
                39: "31d6cfe0",
                40: "31d6cfe0",
                41: "31d6cfe0",
                42: "31d6cfe0",
                43: "31d6cfe0"
                            } [e] + ".chunk.css", f = o.p + n, a =
document.getElementsByTagName("link"), d = 0; d < a.length; d++) {
                        var u = (l = a[d]).getAttribute("data-href") ||
l.getAttribute("href");
                if ("stylesheet" === l.rel && (u === n || u === f)) return
t()
            }
            var i = document.getElementsByTagName("style");
            for (d = 0; d < i.length; d++) {
              var l;
                if ((u = (l = i[d]).getAttribute("data-href")) === n || u
=== f) return t()
            }
            var s = document.createElement("link");
                s.rel = "stylesheet", s.type = "text/css", s.onload = t,
s.onerror = function(t) {
              var n = t && t.target && t.target.src || f,
                a = new Error("Loading CSS chunk " + e + " failed.\n(" + n
+ ")");
                  a.code = "CSS_CHUNK_LOAD_FAILED", a.request = n, delete
c[e], s.parentNode.removeChild(s), r(a)
                                                   }, s.href = f,
document.getElementsByTagName("head")[0].appendChild(s)
          })).then((function() {
            c[e] = 0
          })));
          var r = f[e];
          if (0 !== r)
            if (r) t.push(r[2]);
            else {
              var n = new Promise((function(t, n) {
```



```
                r = f[e] = [t, n]
              }));
              t.push(r[2] = n);
              var a, d = document.createElement("script");
                         d.charset = "utf-8", d.timeout = 120, o.nc &&
d.setAttribute("nonce", o.nc), d.src = function(e) {
                return o.p + "static/js/" + ({} [e] || e) + "." + {
                  0: "83632318",
                  1: "db9278d9",
                  2: "8b222c2f",
                  3: "687378b4",
                  4: "f1080a57",
                  5: "035d7acd",
                  9: "bbbab3d1",
                  10: "20722f15",
                  11: "e3d2010f",
                  12: "8b781481",
                  13: "c35da0ff",
                  14: "0dcf363c",
                  15: "b0ac8399",
                  16: "8446d0e2",
                  17: "f210662f",
                  18: "6ae26374",
                  19: "680d974f",
                  20: "69e2e88c",
                  21: "67116450",
                  22: "cf162b67",
                  23: "a66543b2",
                  24: "0bc2f845",
                  25: "d8e8ba6d",
                  26: "6b7911bd",
                  27: "c6498814",
                  28: "4d225ac9",
                  29: "2c2658c3",
                  30: "05a2a508",
                  31: "44e936d5",
                  32: "4a712a3d",
                  33: "5b6f4207",
                  34: "43b850aa",
                  35: "347c2194",
```

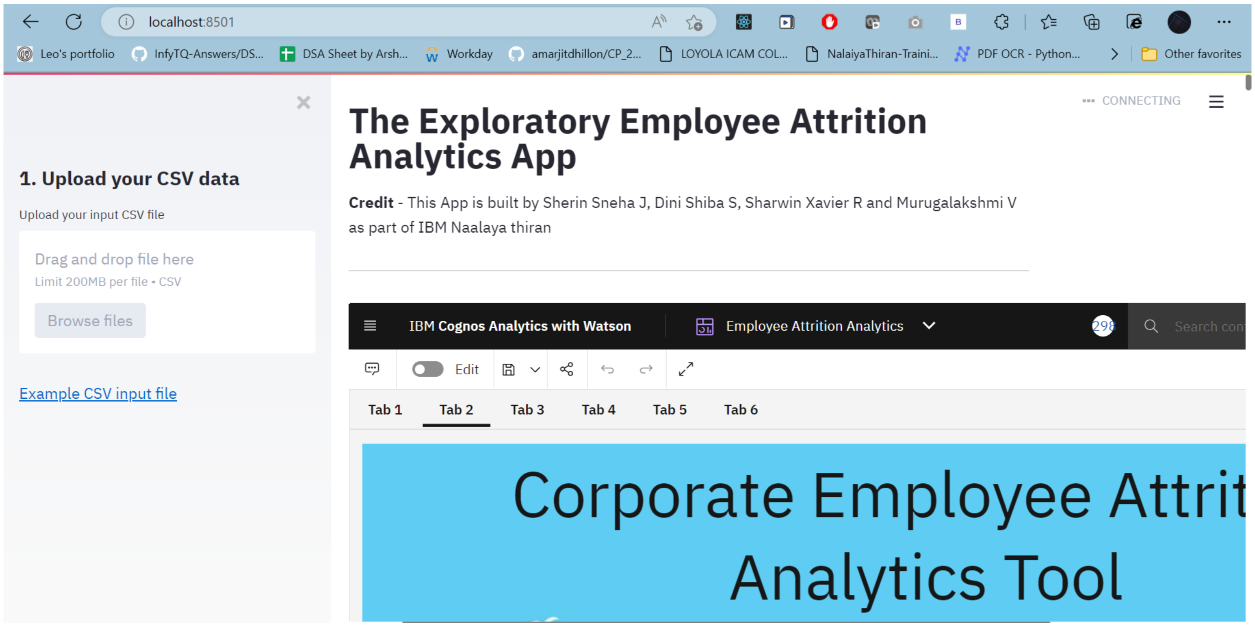


```
                  36: "1737bd17",
                  37: "a68f8714",
                  38: "5df29aef",
                  39: "8aa1a841",
                  40: "58cf9567",
                  41: "9af49f7d",
                  42: "69367570",
                  43: "ce586712"
                } [e] + ".chunk.js"
              }(e);
              var u = new Error;
              a = function(t) {
                d.onerror = d.onload = null, clearTimeout(i);
                var r = f[e];
                if (0 !== r) {
                  if (r) {
                    var n = t && ("load" === t.type ? "missing" : t.type),
                      c = t && t.target && t.target.src;
                    u.message = "Loading chunk " + e + " failed.\n(" + n +
": " + c + ")", u.name = "ChunkLoadError", u.type = n, u.request = c,
r[1](u)
                  }
                  f[e] = void 0
                }
              };
              var i = setTimeout((function() {
                a({
                  type: "timeout",
                  target: d
                })
              }), 12e4);
              d.onerror = d.onload = a, document.head.appendChild(d)
            } return Promise.all(t)
        }, o.m = e, o.c = n, o.d = function(e, t, r) {
          o.o(e, t) || Object.defineProperty(e, t, {
            enumerable: !0,
            get: r
          })
        }, o.r = function(e) {
```



```
                 "undefined" != typeof Symbol && Symbol.toStringTag &&
Object.defineProperty(e, Symbol.toStringTag, {
            value: "Module"
          }), Object.defineProperty(e, "__esModule", {
            value: !0
          })
        }, o.t = function(e, t) {
          if (1 & t && (e = o(e)), 8 & t) return e;
           if (4 & t && "object" == typeof e && e && e.__esModule) return
e;
          var r = Object.create(null);
          if (o.r(r), Object.defineProperty(r, "default", {
              enumerable: !0,
              value: e
            }), 2 & t && "string" != typeof e)
            for (var n in e) o.d(r, n, function(t) {
              return e[t]
            }.bind(null, n));
          return r
        }, o.n = function(e) {
          var t = e && e.__esModule ? function() {
            return e.default
          } : function() {
            return e
          };
          return o.d(t, "a", t), t
        }, o.o = function(e, t) {
          return Object.prototype.hasOwnProperty.call(e, t)
        }, o.p = "./", o.oe = function(e) {
          throw console.error(e), e
        };
                     var d = this["webpackJsonpstreamlit-browser"] =
this["webpackJsonpstreamlit-browser"] || [],
          u = d.push.bind(d);
        d.push = t, d = d.slice();
        for (var i = 0; i < d.length; i++) t(d[i]);
        var l = u;
        r()
      }([])
    </script>
    <script src="./static/js/8.0e5ebbf7.chunk.js"></script>
    <script src="./static/js/main.68bdddd4.chunk.js"></script>
  </body>
                                             <grammarly-desktop-integration
data-grammarly-shadow-root="true"></grammarly-desktop-integration>
</html>
```

Front Screen



1. Upload your CSV data

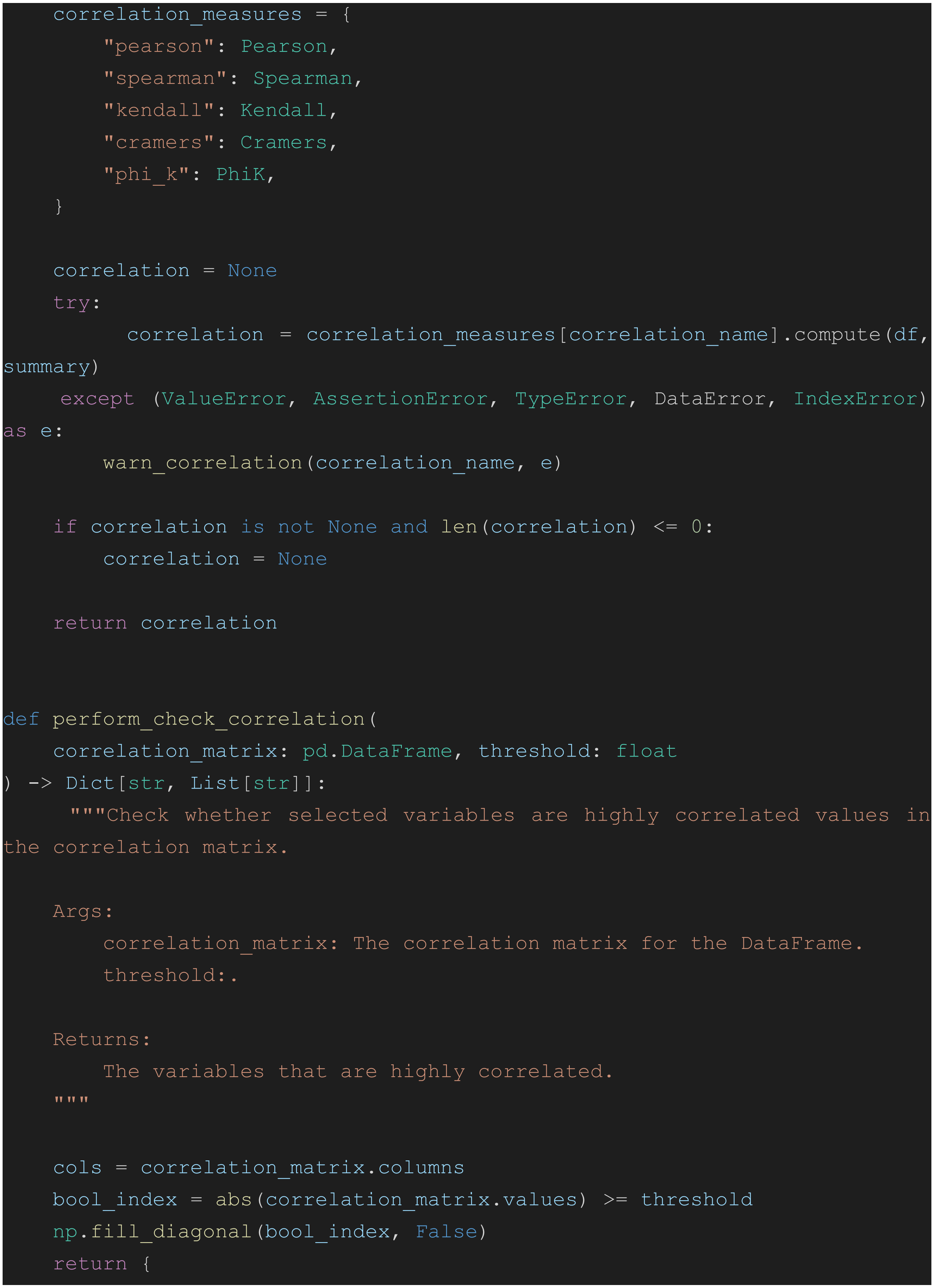
Upload your input CSV file

The Exploratory Employee Attrition Analytics App

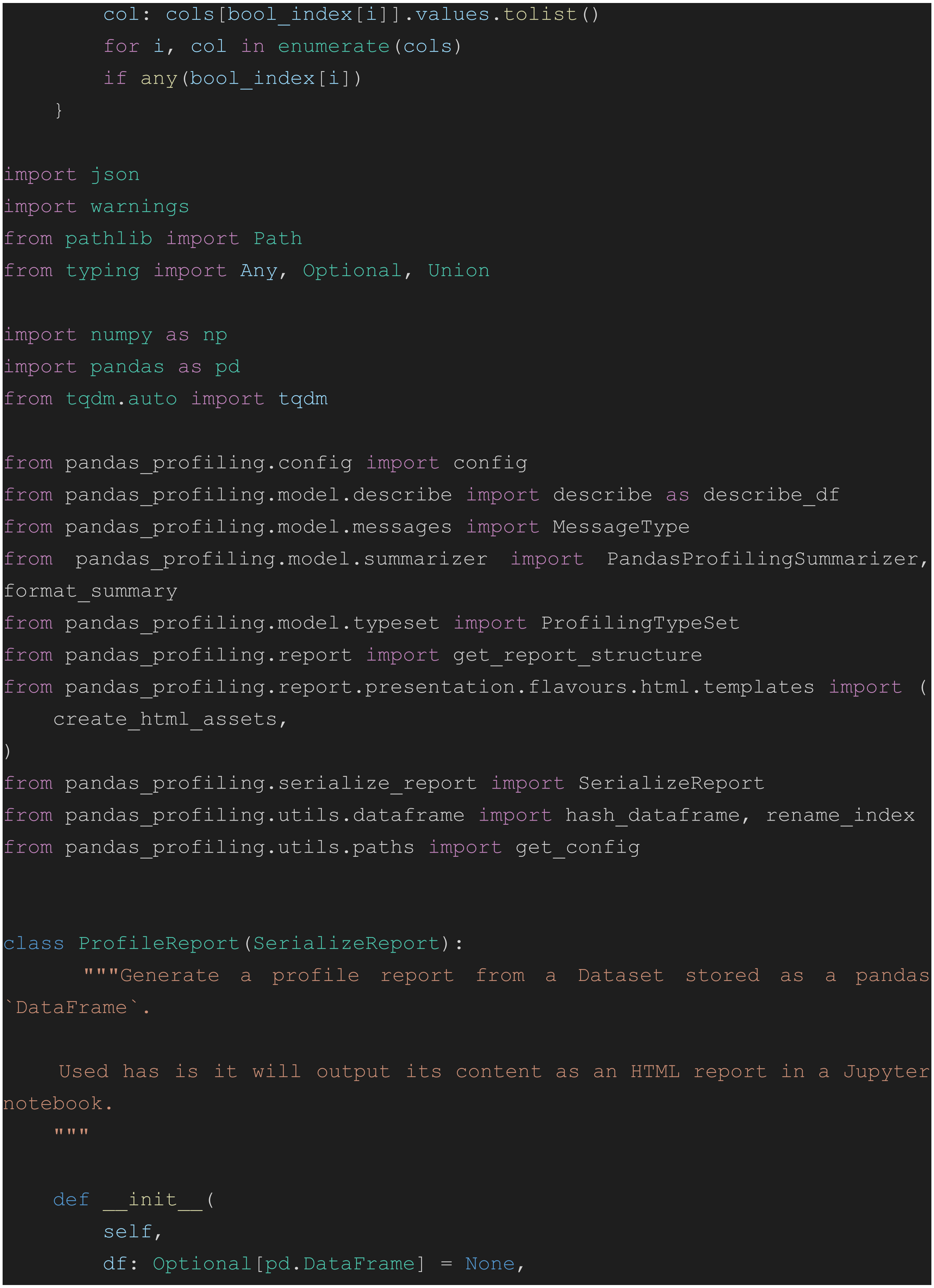
Credit - This App is built by Sherin Sneha J, Dini Shiba S, Sharwin Xavier R and Murugalakshmi V as part of IBM Naalaya thiran

7.2 - Feature 2 - Interrelations and Correlations between columns in the CSV File with respect to Attrition

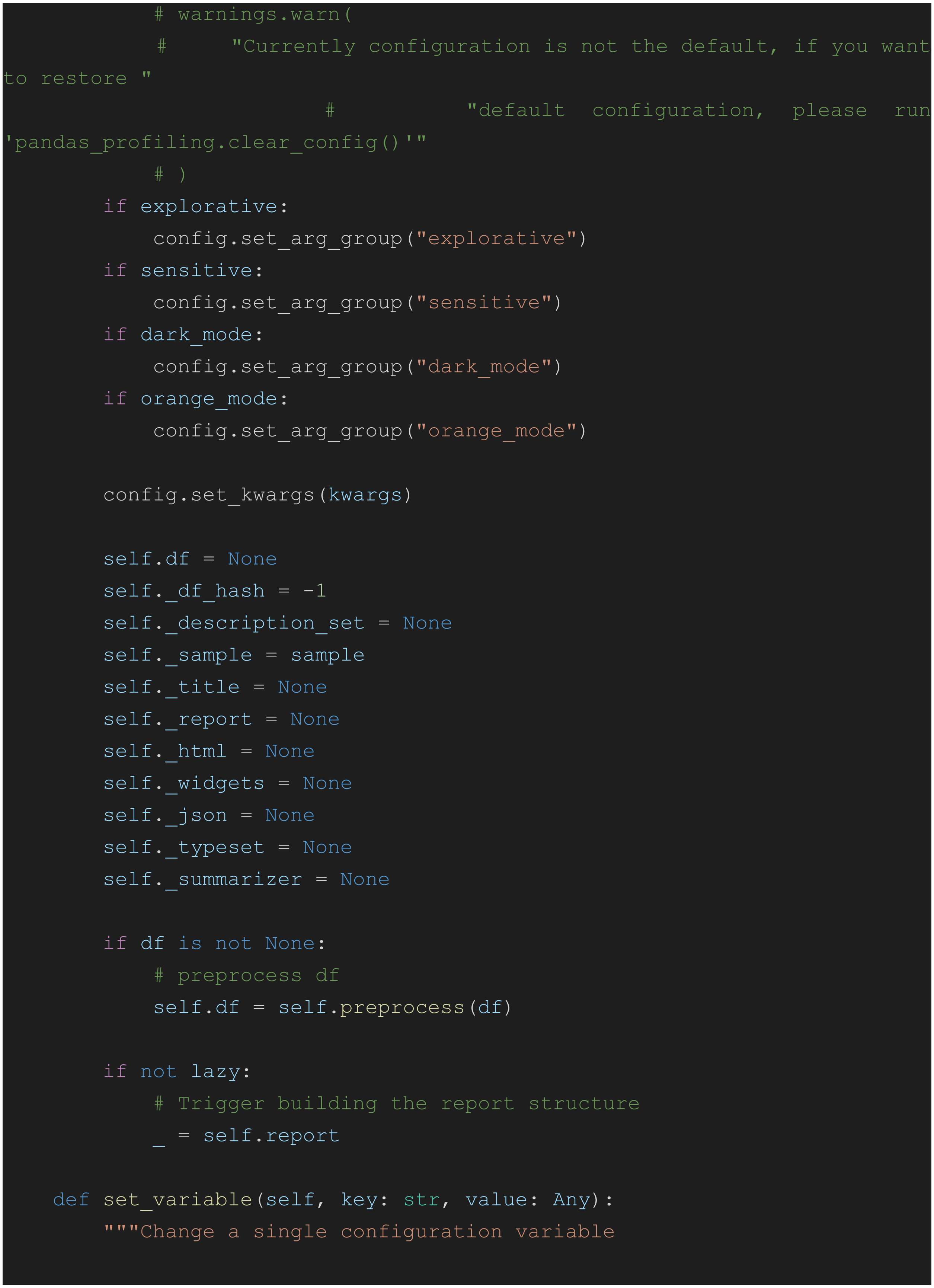
```
"""Correlations between variables."""
import itertools
import warnings
from typing import Dict, List, Optional

import numpy as np
import pandas as pd
from pandas.core.base import DataError
from scipy import stats

from pandas_profiling.config import config
from pandas_profiling.model.typeset import Boolean, Categorical, Numeric,
Unsupported


class Correlation:
    @staticmethod
    def compute(df, summary) -> Optional[pd.DataFrame]:
        return NotImplemented


class Spearman(Correlation):
    @staticmethod
    def compute(df, summary) -> Optional[pd.DataFrame]:
        return df.corr(method="spearman")


class Pearson(Correlation):
    @staticmethod
    def compute(df, summary) -> Optional[pd.DataFrame]:
        return df.corr(method="pearson")


class Kendall(Correlation):
    @staticmethod
    def compute(df, summary) -> Optional[pd.DataFrame]:
        return df.corr(method="kendall")


class Cramers(Correlation):
    @staticmethod
    def _cramers_corrected_stat(confusion_matrix, correction: bool) ->
float:
        """Calculate the Cramer's V corrected stat for two variables.

        Args:
            confusion_matrix: Crosstab between two variables.
            correction: Should the correction be applied?

        Returns:
            The Cramer's V corrected stat for the two variables.
        """
                        chi2    =   stats.chi2_contingency(confusion_matrix,
correction=correction)[0]
        n = confusion_matrix.sum().sum()
        phi2 = chi2 / n
        r = confusion_matrix.shape[0]
          k = confusion_matrix.shape[1] if len(confusion_matrix.shape) > 1
else 1

        # Deal with NaNs later on
        with np.errstate(divide="ignore", invalid="ignore"):
               phi2corr = max(0.0, phi2 - ((k - 1.0) * (r - 1.0)) / (n -
1.0))
            rcorr = r - ((r - 1.0) ** 2.0) / (n - 1.0)
            kcorr = k - ((k - 1.0) ** 2.0) / (n - 1.0)
            rkcorr = min((kcorr - 1.0), (rcorr - 1.0))
            if rkcorr == 0.0:
                corr = 1.0
            else:
                corr = np.sqrt(phi2corr / rkcorr)
        return corr

    @staticmethod
    def compute(df, summary) -> Optional[pd.DataFrame]:
                                                              threshold        =
config["categorical_maximum_correlation_distinct"].get(int)

        categoricals = {
            key
            for key, value in summary.items()
            if value["type"] in {Categorical, Boolean}
            and value["n_distinct"] <= threshold
        }

        if len(categoricals) <= 1:
            return None

        matrix = np.zeros((len(categoricals), len(categoricals)))
        np.fill_diagonal(matrix, 1.0)
        correlation_matrix = pd.DataFrame(
            matrix,
            index=categoricals,
            columns=categoricals,
        )

        for name1, name2 in itertools.combinations(categoricals, 2):
            confusion_matrix = pd.crosstab(df[name1], df[name2])
            correlation_matrix.loc[name2, name1] = Cramers._cramers_corrected_stat(
                confusion_matrix, correction=True
            )
            correlation_matrix.loc[name1, name2] = correlation_matrix.loc[name2, name1]
        return correlation_matrix


class PhiK(Correlation):
    @staticmethod
    def compute(df, summary) -> Optional[pd.DataFrame]:
        threshold = config["categorical_maximum_correlation_distinct"].get(int)
        intcols = {
            key
            for key, value in summary.items()
            # DateTime currently excluded
            # In some use cases, it makes sense to convert it to interval
            # See https://github.com/KaveIO/PhiK/issues/7
            if value["type"] == Numeric and 1 < value["n_distinct"]
        }

        selcols = {
            key
            for key, value in summary.items()
            if value["type"] != Unsupported and 1 < value["n_distinct"] <= threshold
        }
        selcols = selcols.union(intcols)

        if len(selcols) <= 1:
            return None
        with warnings.catch_warnings():
            warnings.simplefilter("ignore")
            import phik

            correlation = df[selcols].phik_matrix(interval_cols=intcols)

        return correlation


def warn_correlation(correlation_name: str, error):
    warnings.warn(
            f"""There was an attempt to calculate the {correlation_name}
correlation, but this failed.
To hide this warning, disable the calculation
(using `df.profile_report(correlations={{\"{correlation_name}\":
{{\"calculate\": False}}}})`
If this is problematic for your use case, please report this as an issue:
https://github.com/pandas-profiling/pandas-profiling/issues
(include the error message: '{error}')"""
    )


def calculate_correlation(
    df: pd.DataFrame, correlation_name: str, summary: dict
) -> Optional[pd.DataFrame]:
     """Calculate the correlation coefficients between variables for the
correlation types selected in the config
    (pearson, spearman, kendall, phi_k, cramers).

    Args:
        df: The DataFrame with variables.
        correlation_name:
        summary: summary dictionary

    Returns:
            The correlation matrices for the given correlation measures.
Return None if correlation is empty.
    """
```



```
    correlation_measures = {
        "pearson": Pearson,
        "spearman": Spearman,
        "kendall": Kendall,
        "cramers": Cramers,
        "phi_k": PhiK,
    }

    correlation = None
    try:
          correlation = correlation_measures[correlation_name].compute(df,
summary)
     except (ValueError, AssertionError, TypeError, DataError, IndexError)
as e:
        warn_correlation(correlation_name, e)

    if correlation is not None and len(correlation) <= 0:
        correlation = None

    return correlation


def perform_check_correlation(
    correlation_matrix: pd.DataFrame, threshold: float
) -> Dict[str, List[str]]:
     """Check whether selected variables are highly correlated values in
the correlation matrix.

    Args:
        correlation_matrix: The correlation matrix for the DataFrame.
        threshold:.

    Returns:
        The variables that are highly correlated.
    """

    cols = correlation_matrix.columns
    bool_index = abs(correlation_matrix.values) >= threshold
    np.fill_diagonal(bool_index, False)
    return {
```

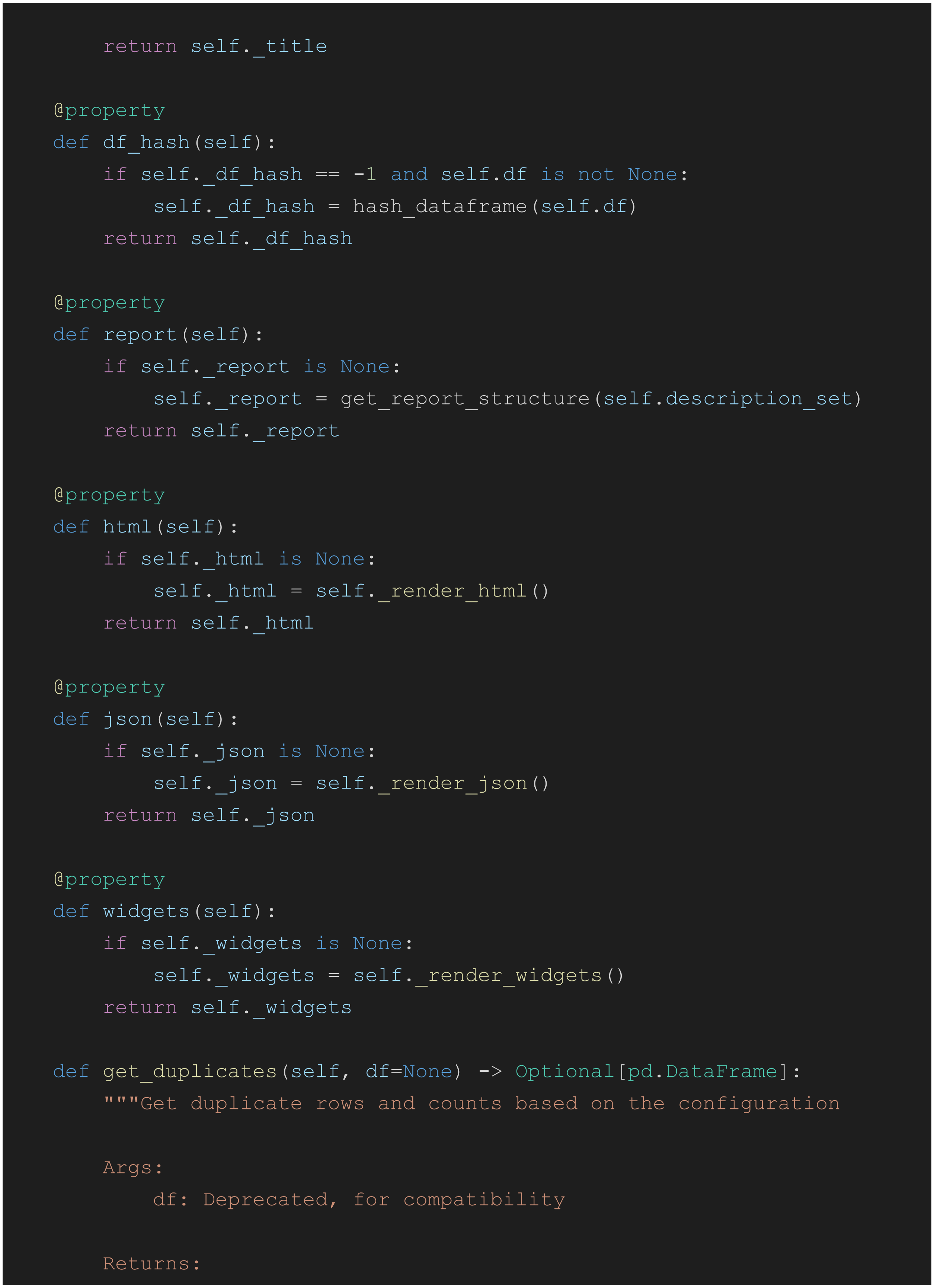


```
        col: cols[bool_index[i]].values.tolist()
        for i, col in enumerate(cols)
        if any(bool_index[i])
    }

import json
import warnings
from pathlib import Path
from typing import Any, Optional, Union

import numpy as np
import pandas as pd
from tqdm.auto import tqdm

from pandas_profiling.config import config
from pandas_profiling.model.describe import describe as describe_df
from pandas_profiling.model.messages import MessageType
from pandas_profiling.model.summarizer import PandasProfilingSummarizer,
format_summary
from pandas_profiling.model.typeset import ProfilingTypeSet
from pandas_profiling.report import get_report_structure
from pandas_profiling.report.presentation.flavours.html.templates import (
    create_html_assets,
)
from pandas_profiling.serialize_report import SerializeReport
from pandas_profiling.utils.dataframe import hash_dataframe, rename_index
from pandas_profiling.utils.paths import get_config


class ProfileReport(SerializeReport):
      """Generate a profile report from a Dataset stored as a pandas
`DataFrame`.

    Used has is it will output its content as an HTML report in a Jupyter
notebook.
    """

    def __init__(
        self,
        df: Optional[pd.DataFrame] = None,
        minimal: bool = False,
        explorative: bool = False,
        sensitive: bool = False,
        dark_mode: bool = False,
        orange_mode: bool = False,
        sample: Optional[dict] = None,
        config_file: Union[Path, str] = None,
        lazy: bool = True,
        **kwargs,
    ):
        """Generate a ProfileReport based on a pandas DataFrame

        Args:
            df: the pandas DataFrame
             minimal: minimal mode is a default configuration with minimal
computation
               config_file: a config file (.yml), mutually exclusive with
`minimal`
            lazy: compute when needed
             sample: optional dict(name="Sample title", caption="Caption",
data=pd.DataFrame())
                **kwargs: other arguments, for valid arguments, check the
default configuration file.
        """
        if config_file is not None and minimal:
            raise ValueError(
                      "Arguments `config_file` and `minimal` are mutually
exclusive."
            )

        if df is None and not lazy:
              raise ValueError("Can init a not-lazy ProfileReport with no
DataFrame")

        if config_file:
            config.set_file(config_file)
        elif minimal:
            config.set_file(get_config("config_minimal.yaml"))
        elif not config.is_default:
            pass
```

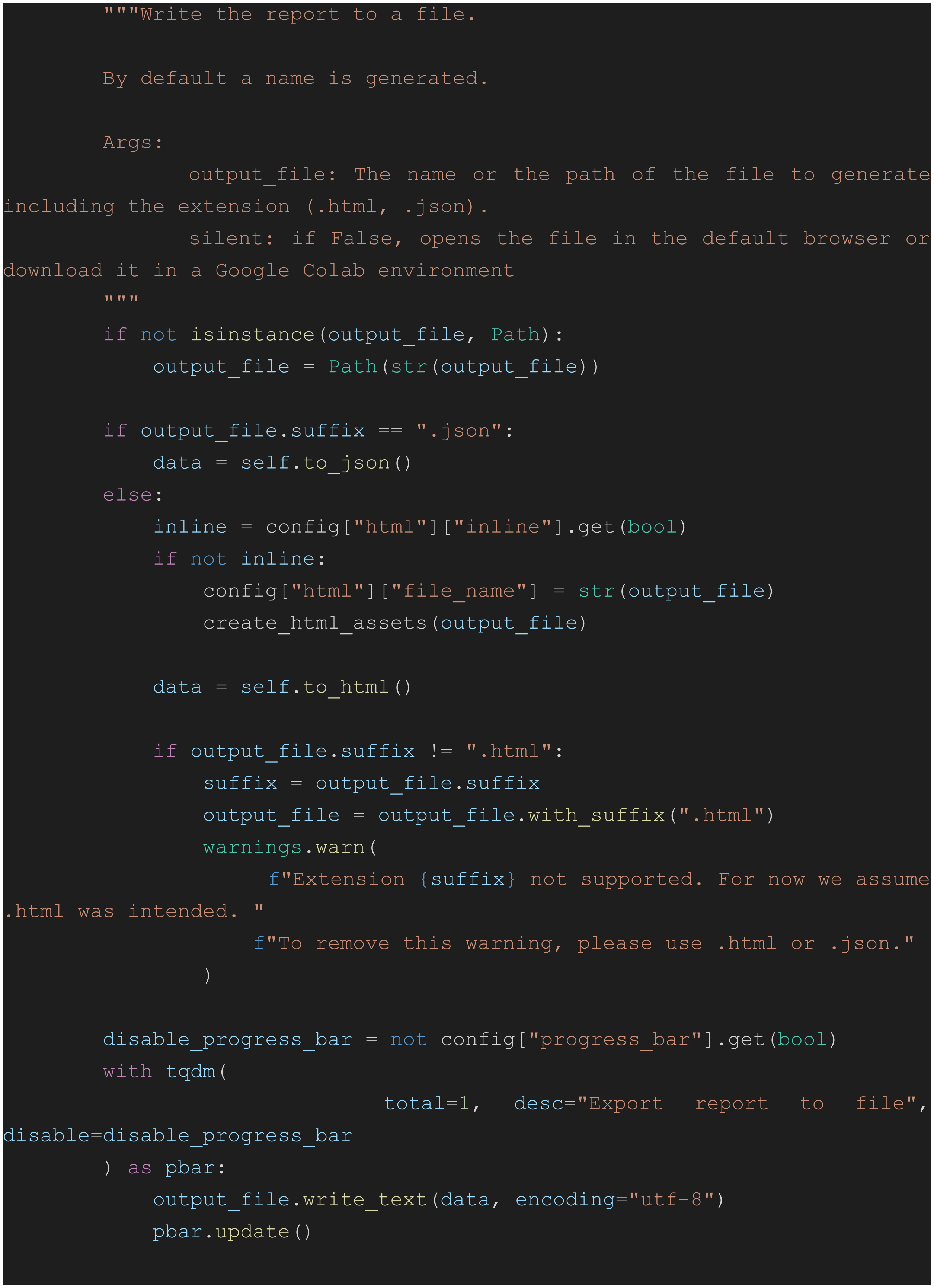


```
            # warnings.warn(
            #     "Currently configuration is not the default, if you want
to restore "
                          #          "default  configuration,  please  run
'pandas_profiling.clear_config()'"
            # )
        if explorative:
            config.set_arg_group("explorative")
        if sensitive:
            config.set_arg_group("sensitive")
        if dark_mode:
            config.set_arg_group("dark_mode")
        if orange_mode:
            config.set_arg_group("orange_mode")

        config.set_kwargs(kwargs)

        self.df = None
        self._df_hash = -1
        self._description_set = None
        self._sample = sample
        self._title = None
        self._report = None
        self._html = None
        self._widgets = None
        self._json = None
        self._typeset = None
        self._summarizer = None

        if df is not None:
            # preprocess df
            self.df = self.preprocess(df)

        if not lazy:
            # Trigger building the report structure
            _ = self.report

    def set_variable(self, key: str, value: Any):
        """Change a single configuration variable
        Args:
            key: configuration parameter name. Accepts nested syntax, e.g.
"html.minify_html"
            value: the new value

        Examples:
            >>> ProfileReport(df).set_variables("title", "NewTitle")
            >>> ProfileReport(df).set_variables("html", {"minify_html":
False})
            >>> ProfileReport(df).set_variables("html.minify_html", False)

        """
        keys = key.split(".")
        for e in reversed(keys[1:]):
            value = {e: value}

        self.set_variables(**{keys[0]: value})

    def set_variables(self, **vars):
        """Change configuration variables (invalidates caches where
necessary)

        Args:
            **vars: configuration parameters to change

        Examples:
            >>> ProfileReport(df).set_variables(title="NewTitle",
html={"minify_html": False})
        """
        changed = set(vars.keys())
        if len({"progress_bar", "pool_size"} & changed) > 0:
            # Cache can persist
            pass

        if len({"notebook"} & changed) > 0:
            self._widgets = None

        if len({"html", "title"} & changed) > 0:
            self._html = None
```

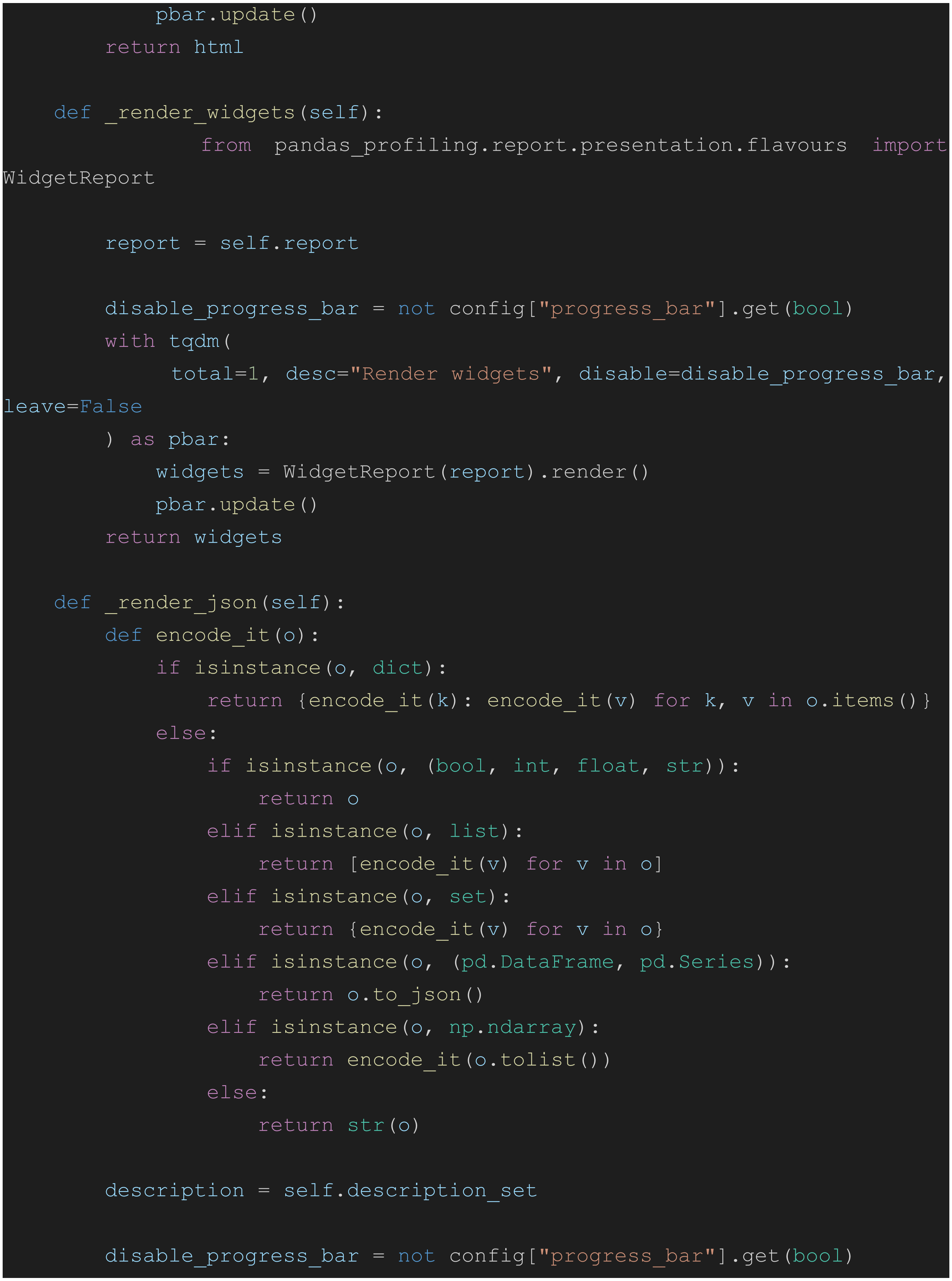
```
         if not {"progress_bar", "pool_size", "notebook", "html", "title"}
>= changed:
            # In all other cases, empty cache
            self._description_set = None
            self._title = None
            self._report = None
            self._html = None
            self._widgets = None
            self._json = None

        if len(vars) == 1:
            config[list(vars.keys())[0]] = list(vars.values())[0]
        else:
            config.set_kwargs(vars)

    @property
    def typeset(self):
        if self._typeset is None:
            self._typeset = ProfilingTypeSet()
        return self._typeset

    @property
    def summarizer(self):
        if self._summarizer is None:
            self._summarizer = PandasProfilingSummarizer(self.typeset)
        return self._summarizer

    @property
    def description_set(self):
        if self._description_set is None:
            self._description_set = describe_df(
                      self.title, self.df, self.summarizer, self.typeset,
self._sample
            )
        return self._description_set

    @property
    def title(self):
        if self._title is None:
            self._title = config["title"].get(str)
```



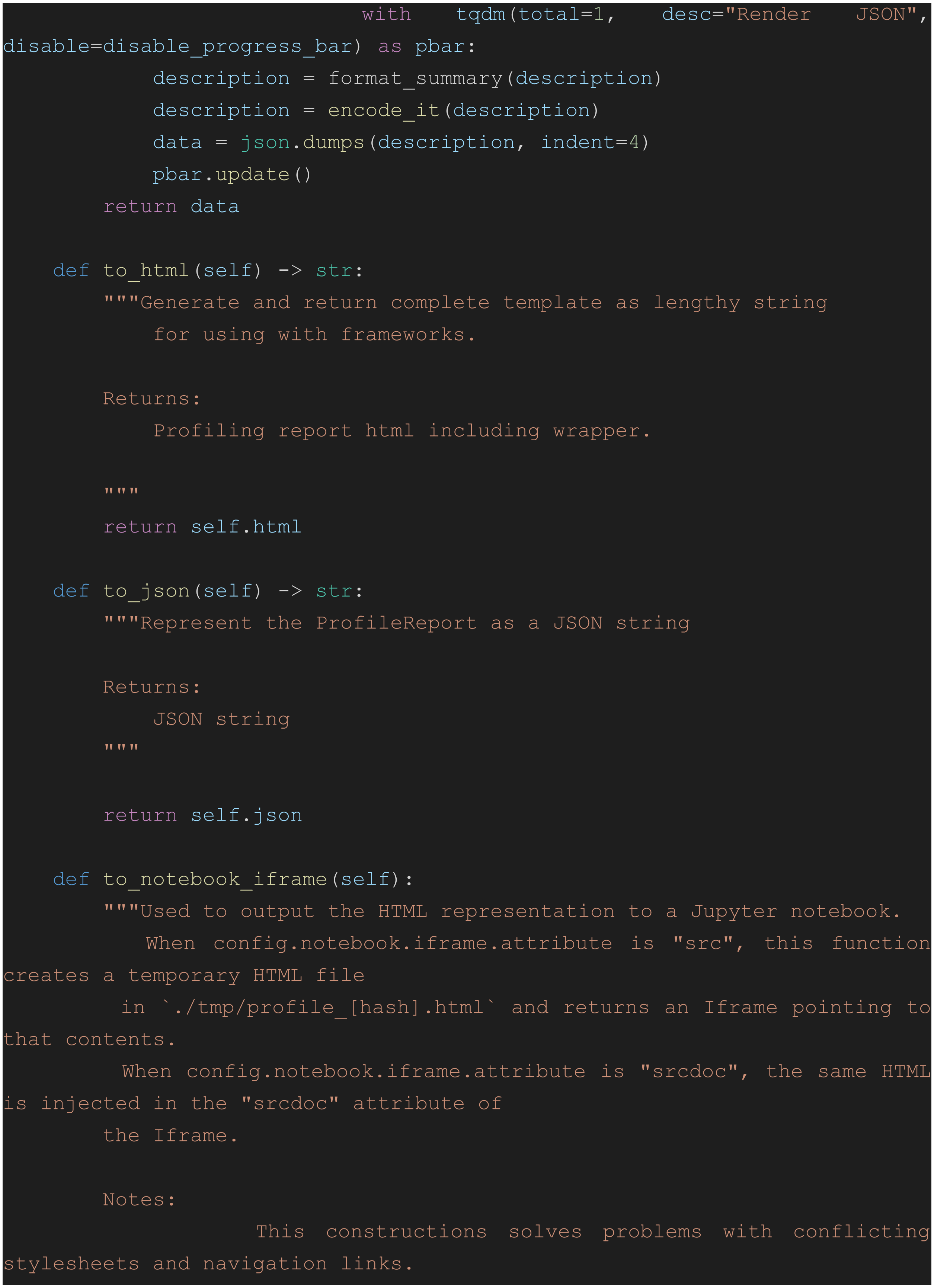
```
        return self._title

    @property
    def df_hash(self):
        if self._df_hash == -1 and self.df is not None:
            self._df_hash = hash_dataframe(self.df)
        return self._df_hash

    @property
    def report(self):
        if self._report is None:
            self._report = get_report_structure(self.description_set)
        return self._report

    @property
    def html(self):
        if self._html is None:
            self._html = self._render_html()
        return self._html

    @property
    def json(self):
        if self._json is None:
            self._json = self._render_json()
        return self._json

    @property
    def widgets(self):
        if self._widgets is None:
            self._widgets = self._render_widgets()
        return self._widgets

    def get_duplicates(self, df=None) -> Optional[pd.DataFrame]:
        """Get duplicate rows and counts based on the configuration

        Args:
            df: Deprecated, for compatibility

        Returns:
            A DataFrame with the duplicate rows and their counts.
        """
        return self.description_set["duplicates"]

    def get_sample(self, df=None) -> dict:
        """Get head/tail samples based on the configuration

        Args:
            df: Deprecated, for compatibility

        Returns:
            A dict with the head and tail samples.
        """
        return self.description_set["sample"]

    def get_description(self) -> dict:
           """Return the description (a raw statistical summary) of the
dataset.

        Returns:
                 Dict containing a description for each variable in the
DataFrame.
        """
        return self.description_set

    def get_rejected_variables(self) -> set:
          """Get variables that are rejected for analysis (e.g. constant,
mixed data types)

        Returns:
            a set of column names that are unsupported
        """
        return {
            message.column_name
            for message in self.description_set["messages"]
            if message.message_type == MessageType.REJECTED
        }

    def to_file(self, output_file: Union[str, Path], silent: bool = True)
-> None:
```



```
        """Write the report to a file.

        By default a name is generated.

        Args:
               output_file: The name or the path of the file to generate
including the extension (.html, .json).
               silent: if False, opens the file in the default browser or
download it in a Google Colab environment
        """
        if not isinstance(output_file, Path):
            output_file = Path(str(output_file))

        if output_file.suffix == ".json":
            data = self.to_json()
        else:
            inline = config["html"]["inline"].get(bool)
            if not inline:
                config["html"]["file_name"] = str(output_file)
                create_html_assets(output_file)

            data = self.to_html()

            if output_file.suffix != ".html":
                suffix = output_file.suffix
                output_file = output_file.with_suffix(".html")
                warnings.warn(
                     f"Extension {suffix} not supported. For now we assume
.html was intended. "
                    f"To remove this warning, please use .html or .json."
                )

        disable_progress_bar = not config["progress_bar"].get(bool)
        with tqdm(
                              total=1, desc="Export report to file",
disable=disable_progress_bar
        ) as pbar:
            output_file.write_text(data, encoding="utf-8")
            pbar.update()
        if not silent:
            try:
                from google.colab import files

                files.download(output_file.absolute().as_uri())
            except ModuleNotFoundError:
                import webbrowser

                webbrowser.open_new_tab(output_file.absolute().as_uri())

    def _render_html(self):
                from  pandas_profiling.report.presentation.flavours  import
HTMLReport

        report = self.report

        disable_progress_bar = not config["progress_bar"].get(bool)
                            with    tqdm(total=1,    desc="Render    HTML",
disable=disable_progress_bar) as pbar:
            html = HTMLReport(report).render(
                nav=config["html"]["navbar_show"].get(bool),
                offline=config["html"]["use_local_assets"].get(bool),
                inline=config["html"]["inline"].get(bool),
                file_name=Path(config["html"]["file_name"].get(str)).stem,

primary_color=config["html"]["style"]["primary_color"].get(str),
                logo=config["html"]["style"]["logo"].get(str),
                theme=config["html"]["style"]["theme"].get(str),
                title=self.description_set["analysis"]["title"],
                date=self.description_set["analysis"]["date_start"],

version=self.description_set["package"]["pandas_profiling_version"],
            )

            minify_html = config["html"]["minify_html"].get(bool)
            if minify_html:
                from htmlmin.main import minify

                         html  =  minify(html,  remove_all_empty_space=True,
remove_comments=True)
```

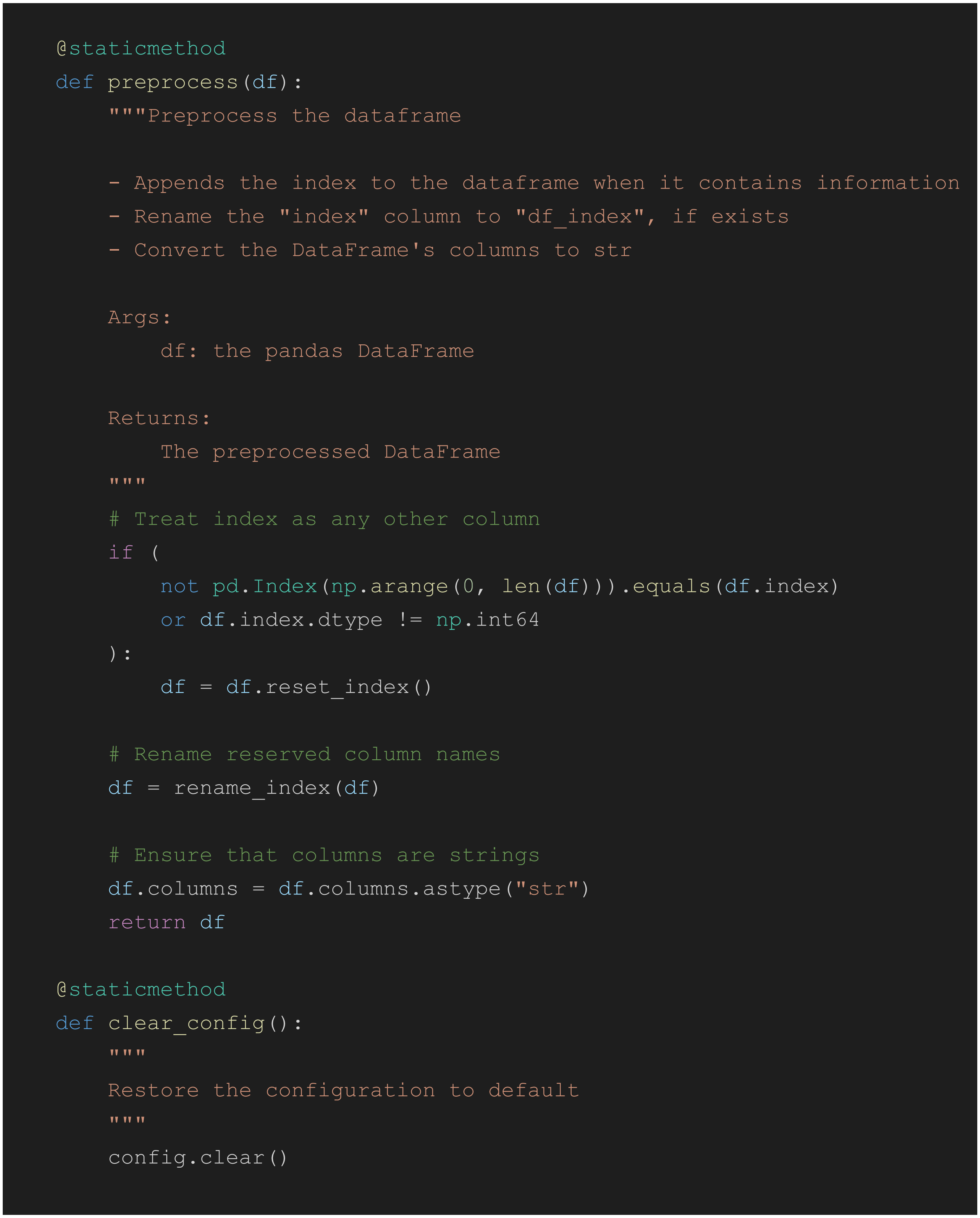


```
            pbar.update()
        return html

    def _render_widgets(self):
                from  pandas_profiling.report.presentation.flavours  import
WidgetReport

        report = self.report

        disable_progress_bar = not config["progress_bar"].get(bool)
        with tqdm(
             total=1, desc="Render widgets", disable=disable_progress_bar,
leave=False
        ) as pbar:
            widgets = WidgetReport(report).render()
            pbar.update()
        return widgets

    def _render_json(self):
        def encode_it(o):
            if isinstance(o, dict):
                return {encode_it(k): encode_it(v) for k, v in o.items()}
            else:
                if isinstance(o, (bool, int, float, str)):
                    return o
                elif isinstance(o, list):
                    return [encode_it(v) for v in o]
                elif isinstance(o, set):
                    return {encode_it(v) for v in o}
                elif isinstance(o, (pd.DataFrame, pd.Series)):
                    return o.to_json()
                elif isinstance(o, np.ndarray):
                    return encode_it(o.tolist())
                else:
                    return str(o)

        description = self.description_set

        disable_progress_bar = not config["progress_bar"].get(bool)
```



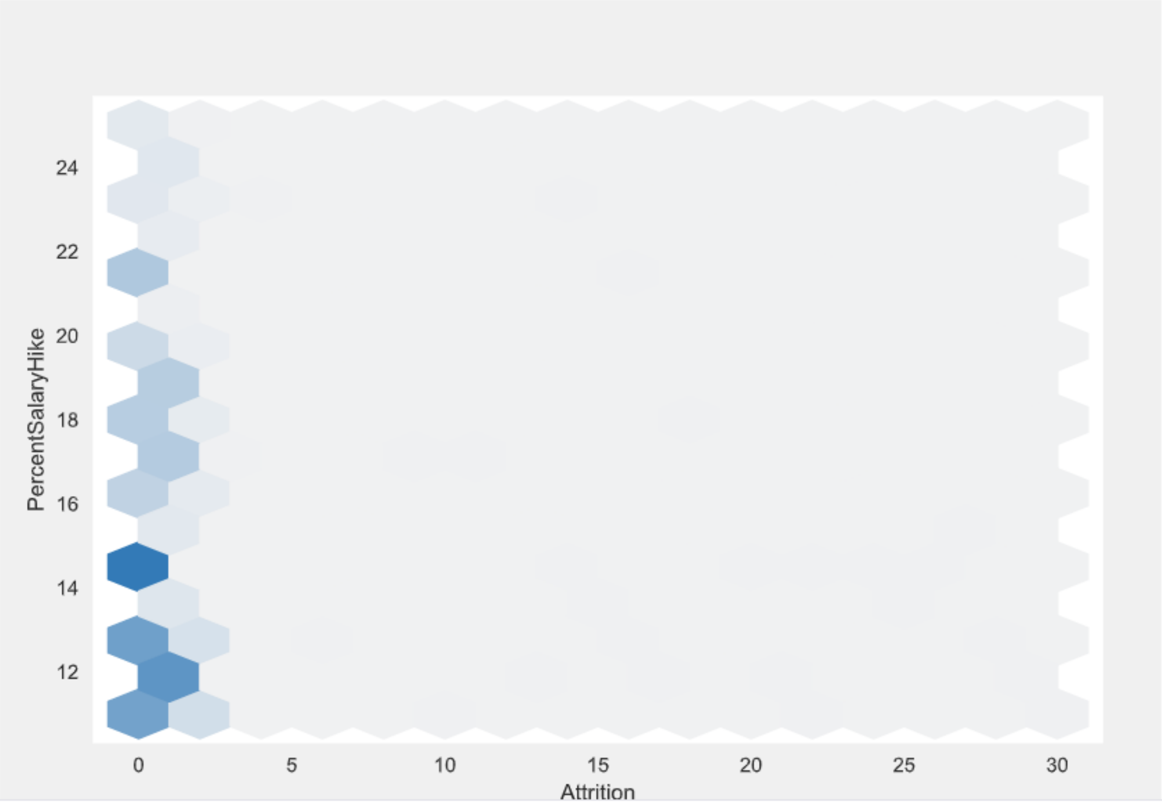
```
                            with    tqdm(total=1,    desc="Render    JSON",
disable=disable_progress_bar) as pbar:
            description = format_summary(description)
            description = encode_it(description)
            data = json.dumps(description, indent=4)
            pbar.update()
        return data

    def to_html(self) -> str:
        """Generate and return complete template as lengthy string
            for using with frameworks.

        Returns:
            Profiling report html including wrapper.

        """
        return self.html

    def to_json(self) -> str:
        """Represent the ProfileReport as a JSON string

        Returns:
            JSON string
        """

        return self.json

    def to_notebook_iframe(self):
        """Used to output the HTML representation to a Jupyter notebook.
           When  config.notebook.iframe.attribute  is  "src",  this  function
creates a temporary HTML file
         in `./tmp/profile_[hash].html` and returns an Iframe pointing to
that contents.
         When config.notebook.iframe.attribute is "srcdoc", the same HTML
is injected in the "srcdoc" attribute of
        the Iframe.

        Notes:
                    This  constructions  solves  problems  with  conflicting
stylesheets and navigation links.
```

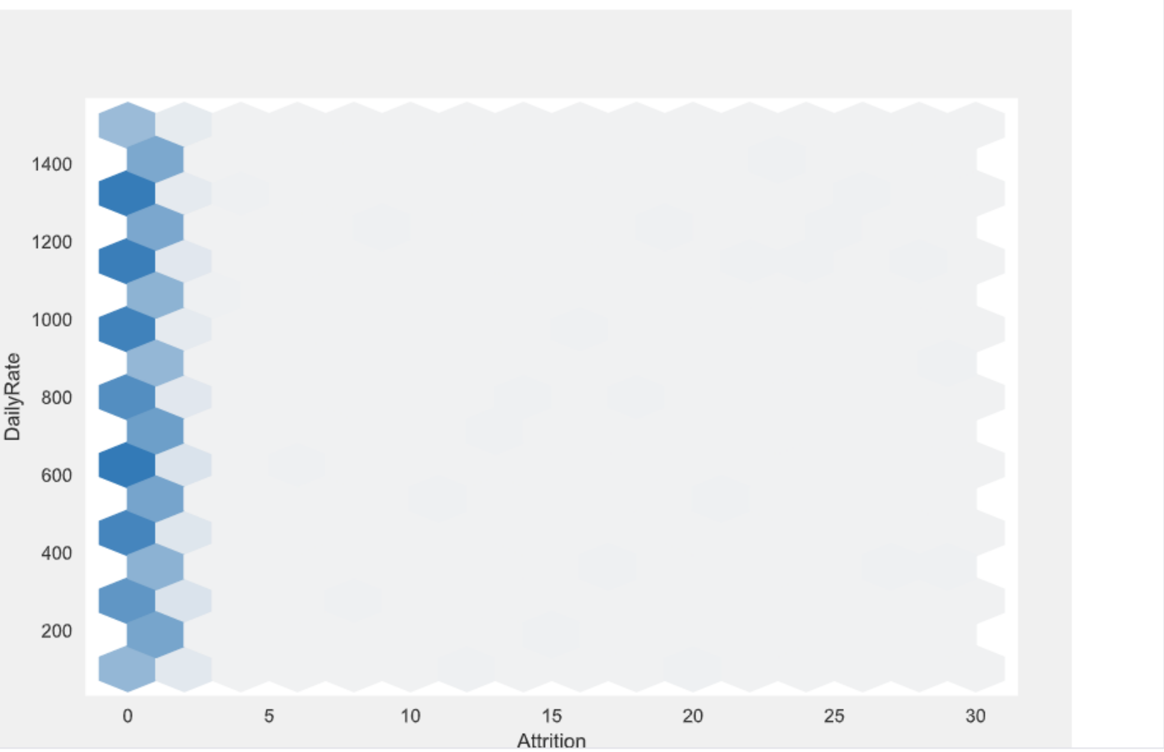
```
        """
        from IPython.core.display import display

        from pandas_profiling.report.presentation.flavours.widget.notebook
import (
            get_notebook_iframe,
        )

                                               #      Ignore      warning:
https://github.com/ipython/ipython/pull/11350/files
        with warnings.catch_warnings():
            warnings.simplefilter("ignore")
            display(get_notebook_iframe(self))

    def to_widgets(self):
        """The ipython notebook widgets user interface."""
        try:
            from google.colab import files

            warnings.warn(
                   "Ipywidgets is not yet fully supported on Google Colab
(https://github.com/googlecolab/colabtools/issues/60)."
                  "As an alternative, you can use the HTML report. See the
documentation for more information."
            )
        except ModuleNotFoundError:
            pass

        from IPython.core.display import display

        display(self.widgets)

    def _repr_html_(self):
         """The ipython notebook widgets user interface gets called by the
jupyter notebook."""
        self.to_notebook_iframe()

    def __repr__(self):
        """Override so that Jupyter Notebook does not print the object."""
        return ""
```

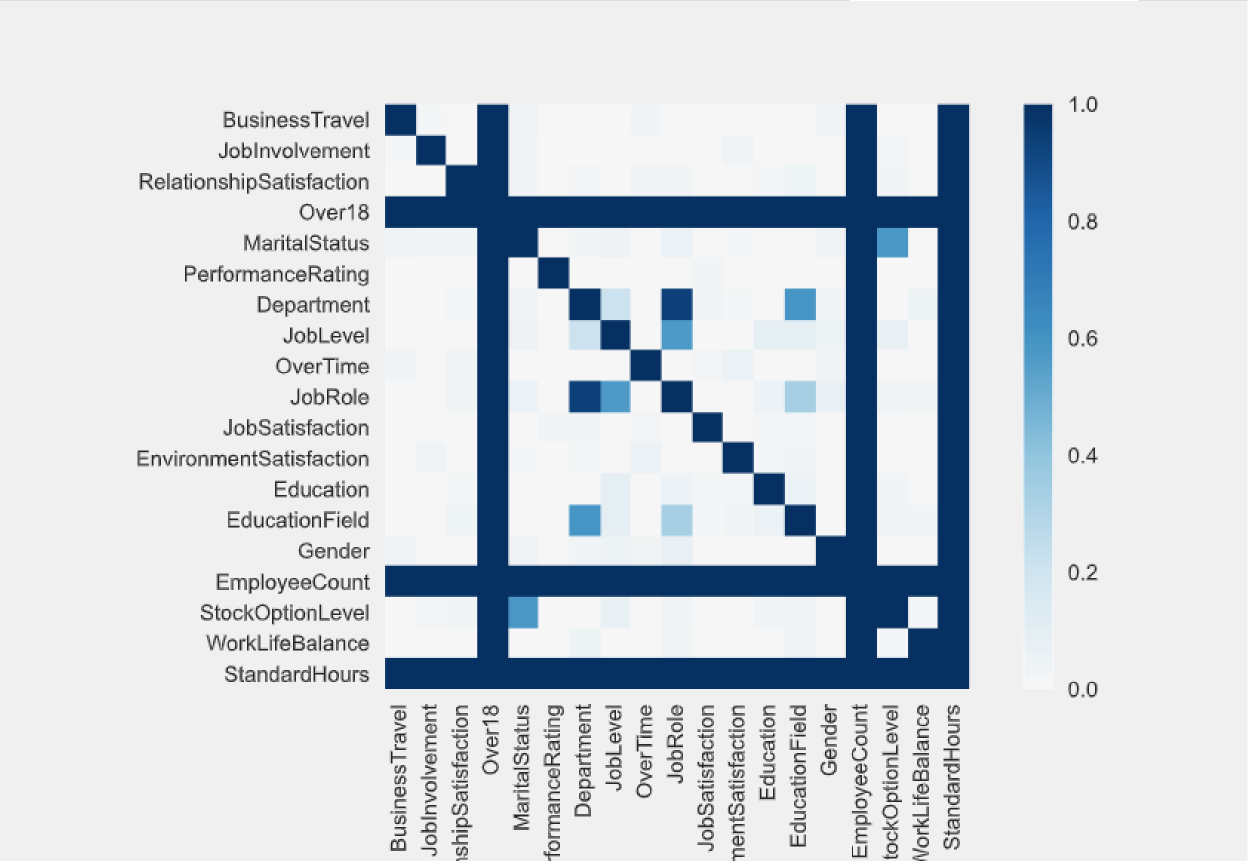


```
@staticmethod
def preprocess(df):
    """Preprocess the dataframe

    - Appends the index to the dataframe when it contains information
    - Rename the "index" column to "df_index", if exists
    - Convert the DataFrame's columns to str

    Args:
        df: the pandas DataFrame

    Returns:
        The preprocessed DataFrame
    """
    # Treat index as any other column
    if (
        not pd.Index(np.arange(0, len(df))).equals(df.index)
        or df.index.dtype != np.int64
    ):
        df = df.reset_index()

    # Rename reserved column names
    df = rename_index(df)

    # Ensure that columns are strings
    df.columns = df.columns.astype("str")
    return df

@staticmethod
def clear_config():
    """
    Restore the configuration to default
    """
    config.clear()
```

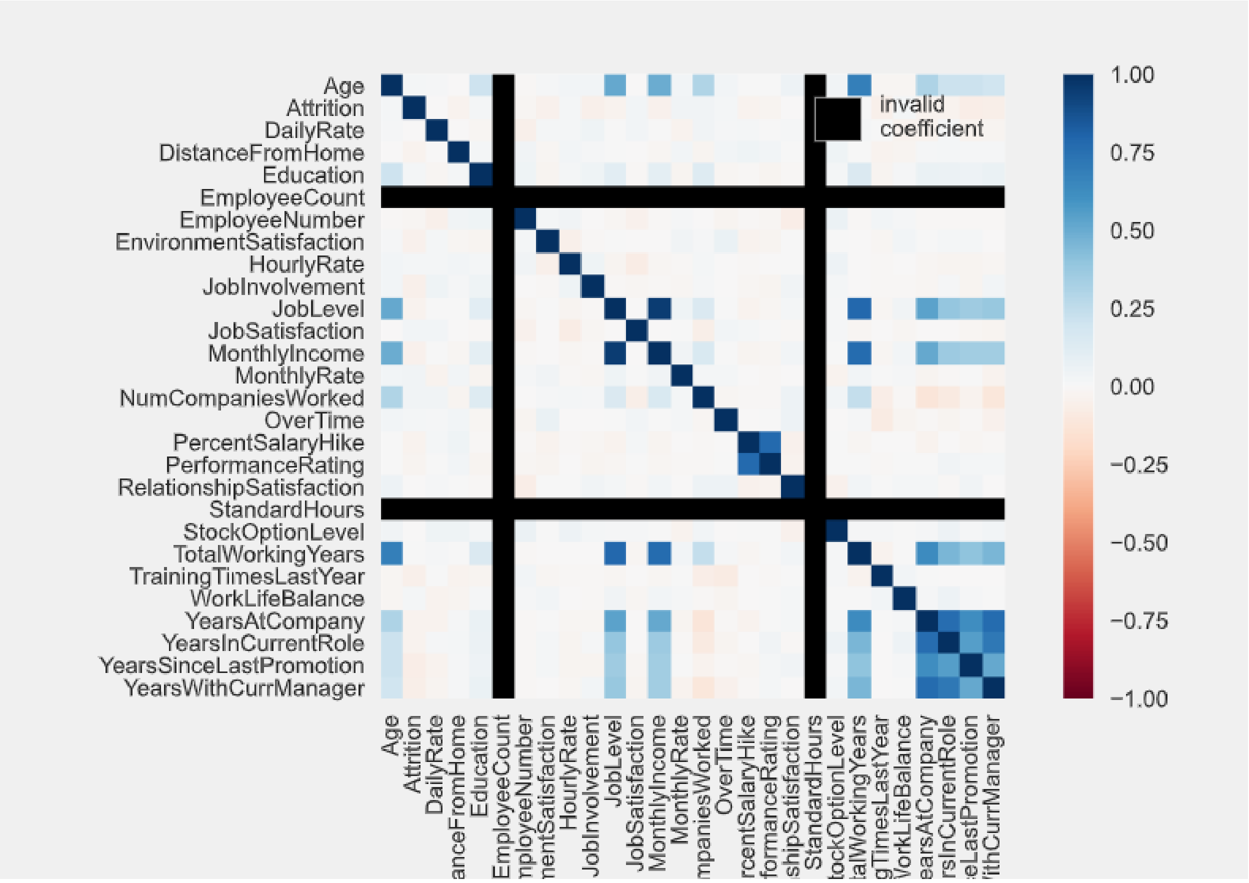
Interactions and Correlations

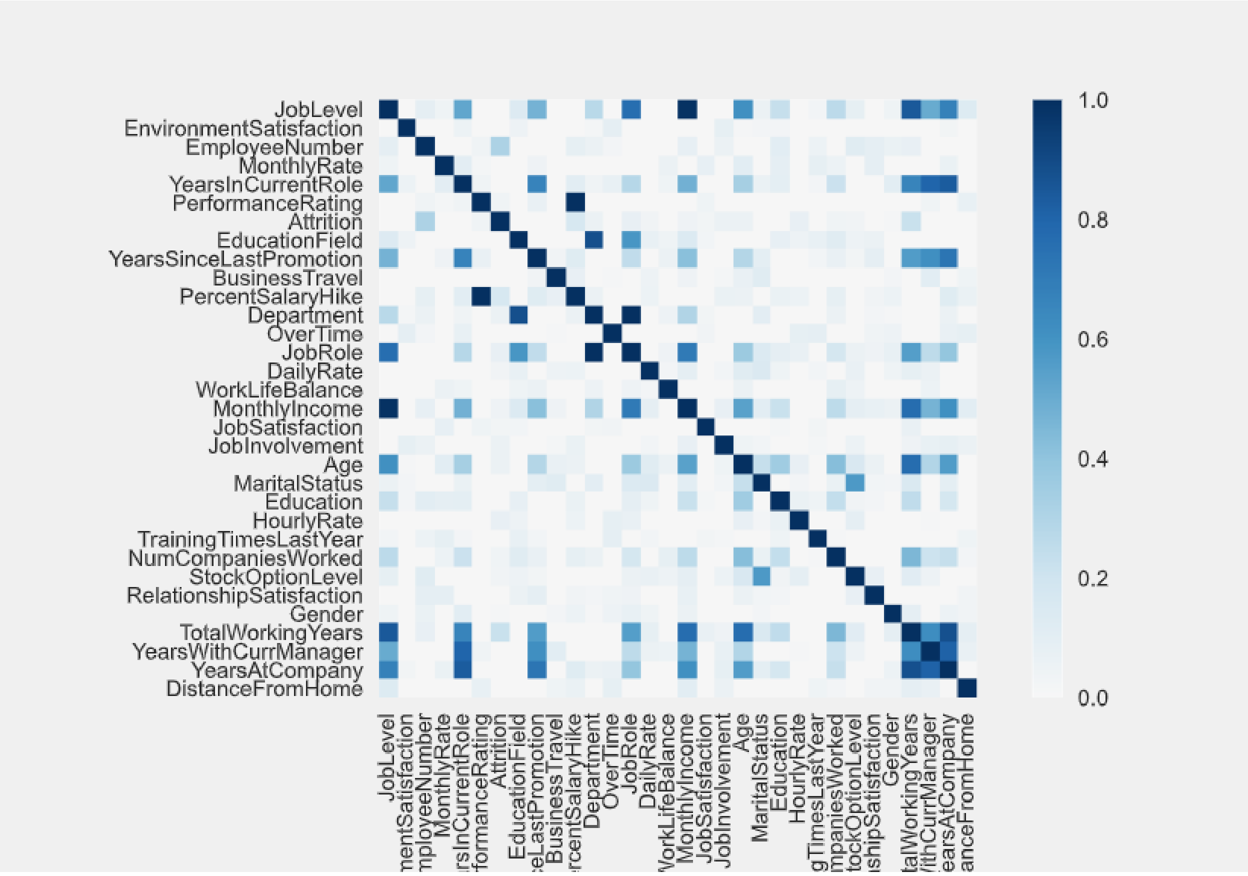



DailyRate
1400
1200
1000
800
600
400
200
0
5
10
15
20
25
30
Attrition


BusinessTravel
JobInvolvement
RelationshipSatisfaction
Over18
MaritalStatus
PerformanceRating
Department
JobLevel
OverTime
JobRole
JobSatisfaction
EnvironmentSatisfaction
Education
EducationField
Gender
EmployeeCount
StockOptionLevel
WorkLifeBalance
StandardHours
1.0
0.8
0.6
0.4
0.2
0.0

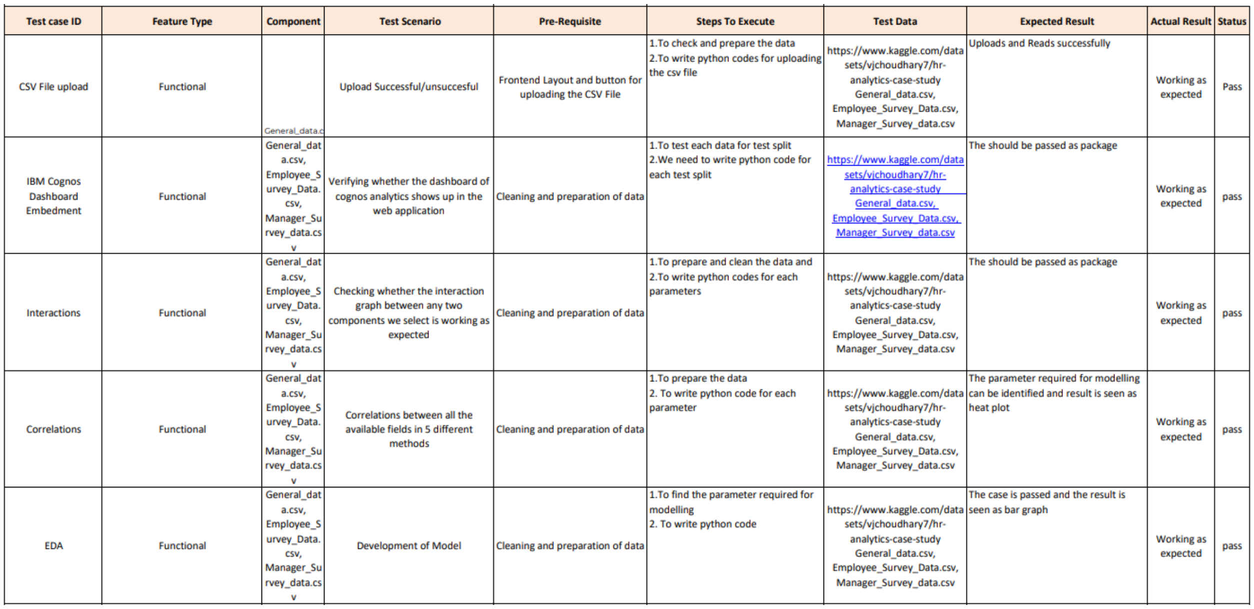
Team ID : PNT2022TMID27605





8. TESTING

8.1 Test Cases

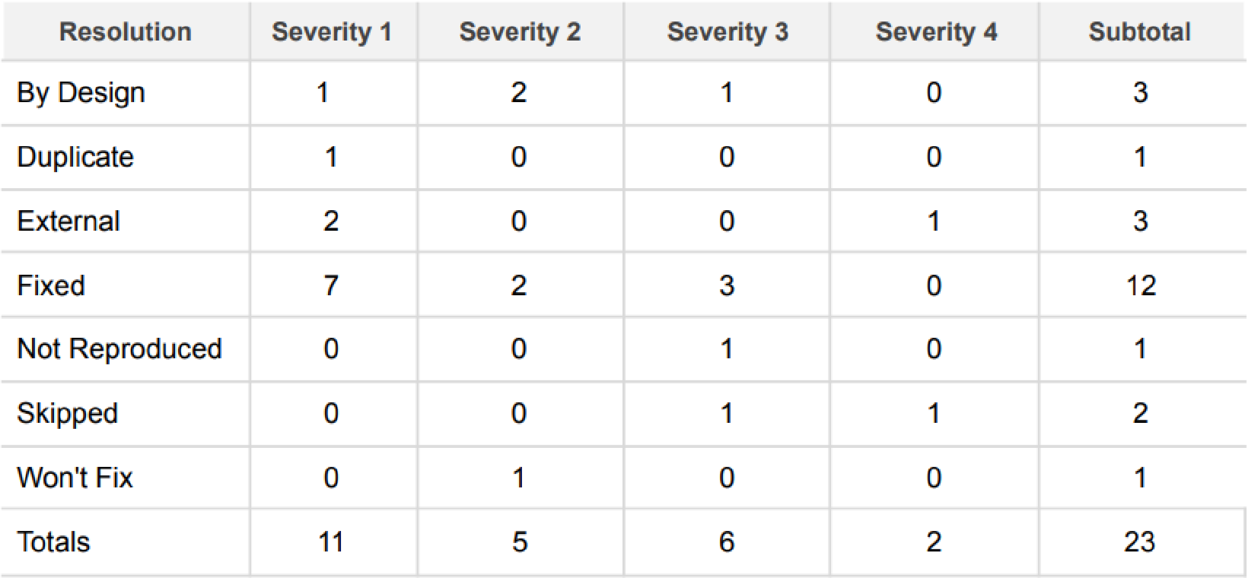


Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status
CSV File upload	Functional	General_data.c	Upload Successful/unsuccesful	Frontend Layout and button for uploading the CSV File	1.To check and prepare the data 2.To write python codes for uploading the csv file	https://www.kaggle.com/datasets/vjchoudhary7/hr-analytics-case-study General_data.csv, Employee_Survey_Data.csv, Manager_Survey_data.csv	Uploads and Reads successfully	Working as expected	Pass
IBM Cognos Dashboard Embedment	Functional	General_data.csv, Employee_Survey_Data.csv, Manager_Survey_data.csv	Verifying whether the dashboard of cognos analytics shows up in the web application	Cleaning and preparation of data	1.To test each data for test split 2.We need to write python code for each test split	https://www.kaggle.com/datasets/vjchoudhary7/hr-analytics-case-study General_data.csv, Employee_Survey_Data.csv, Manager_Survey_data.csv	The should be passed as package	Working as expected	pass
Interactions	Functional	General_data.csv, Employee_Survey_Data.csv, Manager_Survey_data.csv	Checking whether the interaction graph between any two components we select is working as expected	Cleaning and preparation of data	1.To prepare and clean the data and 2.To write python codes for each parameters	https://www.kaggle.com/datasets/vjchoudhary7/hr-analytics-case-study General_data.csv, Employee_Survey_Data.csv, Manager_Survey_data.csv	The should be passed as package	Working as expected	pass
Correlations	Functional	General_data.csv, Employee_Survey_Data.csv, Manager_Survey_data.csv	Correlations between all the available fields in 5 different methods	Cleaning and preparation of data	1.To prepare the data 2. To write python code for each parameter	https://www.kaggle.com/datasets/vjchoudhary7/hr-analytics-case-study General_data.csv, Employee_Survey_Data.csv, Manager_Survey_data.csv	The parameter required for modelling can be identified and result is seen as heat plot	Working as expected	pass
EDA	Functional	General_data.csv, Employee_Survey_Data.csv, Manager_Survey_data.csv	Development of Model	Cleaning and preparation of data	1.To find the parameter required for modelling 2. To write python code	https://www.kaggle.com/datasets/vjchoudhary7/hr-analytics-case-study General_data.csv, Employee_Survey_Data.csv, Manager_Survey_data.csv	The case is passed and the result is seen as bar graph	Working as expected	pass

8.2 User Acceptance Testing

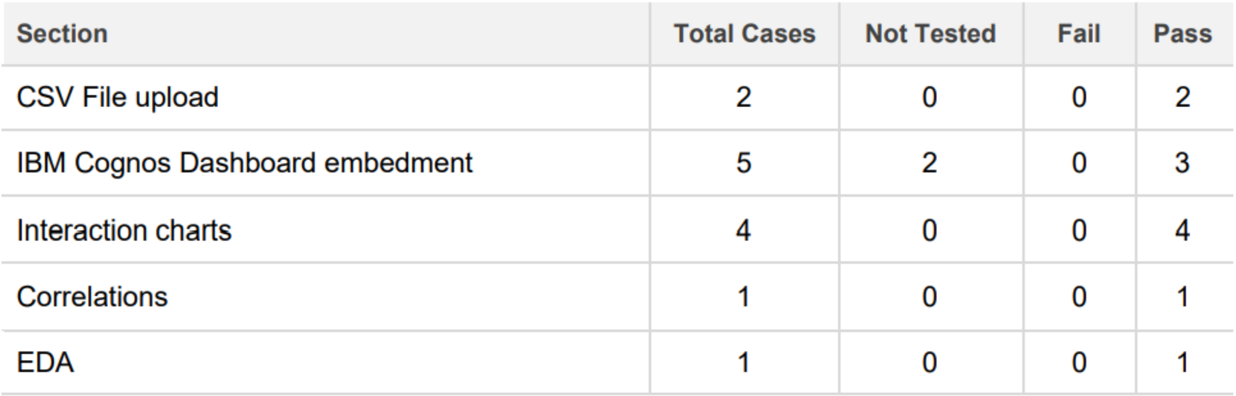
Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved



Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	1	2	1	0	3
Duplicate	1	0	0	0	1
External	2	0	0	1	3
Fixed	7	2	3	0	12
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	1	0	0	1
Totals	11	5	6	2	23

Test Case Analysis



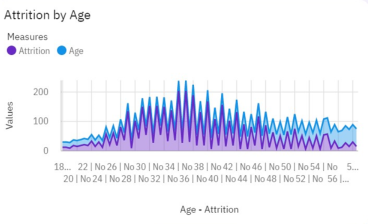
Section	Total Cases	Not Tested	Fail	Pass
CSV File upload	2	0	0	2
IBM Cognos Dashboard embedment	5	2	0	3
Interaction charts	4	0	0	4
Correlations	1	0	0	1
EDA	1	0	0	1

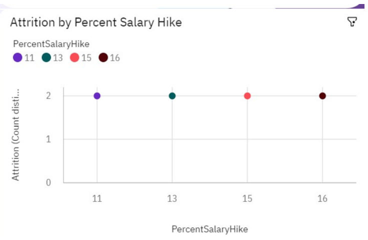
9. Results

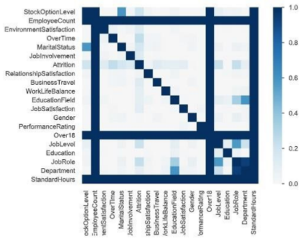
9.1 - Performance Testing

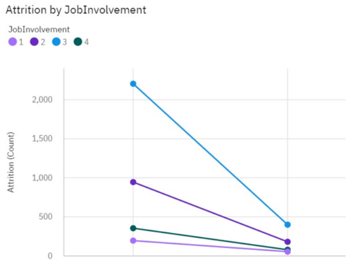
<table>
<tr><th>S.no</th><th>Parameter</th><th>Screenshot/ Values</th></tr>
<tr><td>1</td><td>Dashboard Design</td><td>

No. of Visualizations - 6

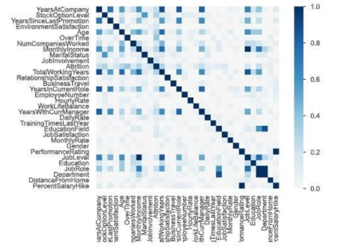
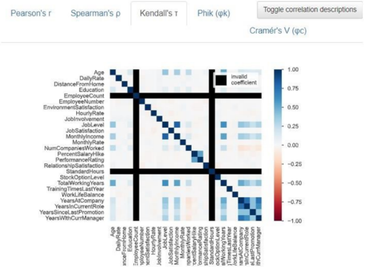


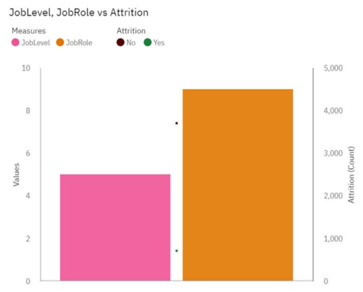


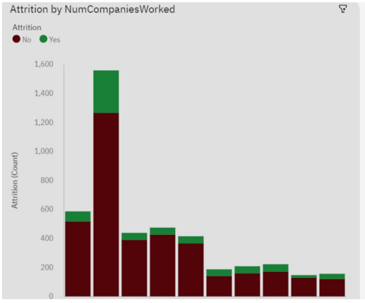


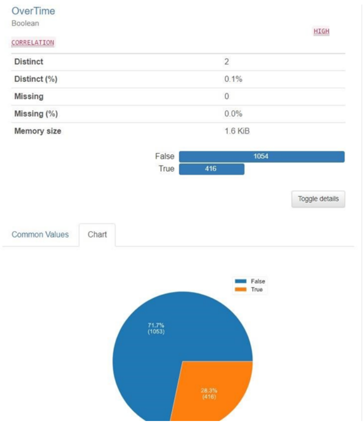


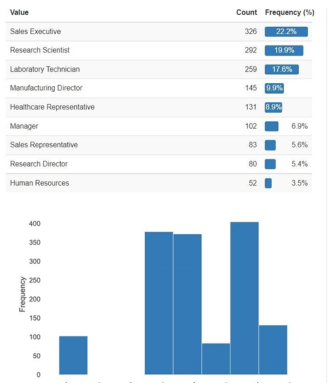
</td></tr>
</table>

		  Pearson's r Spearman's ρ Kendall's τ Phik (φk) Toggle correlation descriptions Cramér's V (φc) Age DailyRate DistanceFromHome Education EmployeeCount EmployeeNumber EnvironmentSatisfaction HourlyRate JobInvolvement JobLevel JobSatisfaction MonthlyIncome MonthlyRate NumCompaniesWorked PercentSalaryHike PerformanceRating RelationshipSatisfaction StandardHours StockOptionLevel TotalWorkingYears TrainingTimesLastYear WorkLifeBalance YearsAtCompany YearsInCurrentRole YearsSinceLastPromotion YearsWithCurrManager invalid coefficient 1.00 0.75 0.50 0.25 0.00 −0.25 −0.50 −0.75 −1.00
2	Data Responsiveness	Employee Attrition by Age Attrition by Business Travel Attrition by Department, Job Role, Education Level and Marital Status Attrition by Salary Hike Percent Attrition by No. of Companies Worked Attrition by Income Groups Attrition by Work Experience Groups Dashboard of Attrition of Employees based on Employment details
3	Amount Data to Rendered (DB2 Metrics)	General_data.csv, Employee_Survey_Data.csv , Manager_Survey_data.csv
4	Utilization of Data Filters	Grouping Sections

<table>
<tr><td></td><td></td><td>Auto
general</td></tr>
<tr><td>5</td><td>Effective User Story</td><td>No of Scene Added - 8</td></tr>
<tr><td>6</td><td>Descriptive Reports</td><td>No of Visualizations / Graphs - 6





OverTime
Boolean
HIGH
CORRELATION
<table>
<tr><td>Distinct</td><td>2</td></tr>
<tr><td>Distinct (%)</td><td>0.1%</td></tr>
<tr><td>Missing</td><td>0</td></tr>
<tr><td>Missing (%)</td><td>0.0%</td></tr>
<tr><td>Memory size</td><td>1.6 KiB</td></tr>
</table>
<table>
<tr><td>False</td><td>1054</td></tr>
<tr><td>True</td><td>416</td></tr>
</table>


<table>
<tr><th>Value</th><th>Count</th><th>Frequency (%)</th></tr>
<tr><td>Sales Executive</td><td>326</td><td>22.2%</td></tr>
<tr><td>Research Scientist</td><td>292</td><td>19.9%</td></tr>
<tr><td>Laboratory Technician</td><td>259</td><td>17.6%</td></tr>
<tr><td>Manufacturing Director</td><td>145</td><td>9.9%</td></tr>
<tr><td>Healthcare Representative</td><td>131</td><td>8.9%</td></tr>
<tr><td>Manager</td><td>102</td><td>6.9%</td></tr>
<tr><td>Sales Representative</td><td>83</td><td>5.6%</td></tr>
<tr><td>Research Director</td><td>80</td><td>5.4%</td></tr>
<tr><td>Human Resources</td><td>52</td><td>3.5%</td></tr>
</table>

</td></tr>
</table>

10. ADVANTAGES & DISADVANTAGES

Advantages:

1.Higher manpower cost

2.Stronger employee relationships

3.Setting a culture right

4.High performance

5.Improve employee satisfaction

6.Increased productivity

7.Increased Revenue

8.Morale improvement

Disadvantages:

1.Lack of knowledgeable people

2.Decreased overall performance

3.Poor work life balance

4.Create a negative image

5.Huge risk on company reputation

11.CONCLUSION

The following suggestion are given based on the analysis and modeling result:

CURRENT EMPLOYEES:

•Work life balance should be improved

•Work environment should be improved

•The manager of an employee should not be changed very often

•Employees should be provided relevant training regularly, especially for its younger employees

FUTURE EMPLOYEES (CHANGES IN HIRING PROCESS):

The company should follow either one of the strategies given below –

•Hire older people with decent work experience

•Hire young people and train them appropriately

12. FUTURE SCOPE

The future scope of the research is that these analysis and modeling helps in forecasting the cause of employee disengagement, enables HR managers develop long-term strategies to reduce attrition, Competitive measures to enhance company brand image, Develops and shapes drills that benefit both the management and the employees. The scope of this research can be extended to many numbers of samples and to other working fields other than corporations.

13. APPENDIX

Nowadays, employee attrition has become a serious issue regarding a company's competitive advantage. It's very expensive to find, hire and train new talents. It's more cost-effective to keep the employees a company already has. A company needs to maintain a pleasant working atmosphere to make their employees stay in that company for a longer period. A few years back it was done manually but it is an era of machine learning and data analytics. Now, a company's HR department uses some data analytics tool to identify which areas to be modified to make most of its employees stay.

Source code

profile_report.py

```
import json
import warnings
from pathlib import Path
from typing import Any, Optional, Union

import numpy as np
import pandas as pd
from tqdm.auto import tqdm

from pandas_profiling.config import config
from pandas_profiling.model.describe import describe as describe_df
from pandas_profiling.model.messages import MessageType
from pandas_profiling.model.summarizer import PandasProfilingSummarizer, format_summary
from pandas_profiling.model.typeset import ProfilingTypeSet
from pandas_profiling.report import get_report_structure
from pandas_profiling.report.presentation.flavours.html.templates import (
    create_html_assets,
)
from pandas_profiling.serialize_report import SerializeReport
from pandas_profiling.utils.dataframe import hash_dataframe, rename_index
from pandas_profiling.utils.paths import get_config


class ProfileReport(SerializeReport):
    """Generate a profile report from a Dataset stored as a pandas `DataFrame`.

    Used has is it will output its content as an HTML report in a Jupyter notebook.
    """

    def __init__(
        self,
        df: Optional[pd.DataFrame] = None,
        minimal: bool = False,
        explorative: bool = False,
        sensitive: bool = False,
        dark_mode: bool = False,
        orange_mode: bool = False,
        sample: Optional[dict] = None,
        config_file: Union[Path, str] = None,
        lazy: bool = True,
        **kwargs,
    ):
        """Generate a ProfileReport based on a pandas DataFrame

        Args:
            df: the pandas DataFrame
            minimal: minimal mode is a default configuration with minimal computation
            config_file: a config file (.yml), mutually exclusive with `minimal`
            lazy: compute when needed
            sample: optional dict(name="Sample title", caption="Caption", data=pd.DataFrame())
            **kwargs: other arguments, for valid arguments, check the default configuration file.
```

```
"""
if config_file is not None and minimal:
    raise ValueError(
        "Arguments `config_file` and `minimal` are mutually exclusive."
    )

if df is None and not lazy:
    raise ValueError("Can init a not-lazy ProfileReport with no DataFrame")

if config_file:
    config.set_file(config_file)
elif minimal:
    config.set_file(get_config("config_minimal.yaml"))
elif not config.is_default:
    pass
    # warnings.warn(
    #     "Currently configuration is not the default, if you want to restore "
    #     "default configuration, please run 'pandas_profiling.clear_config()'"
    # )
if explorative:
    config.set_arg_group("explorative")
if sensitive:
    config.set_arg_group("sensitive")
if dark_mode:
    config.set_arg_group("dark_mode")
if orange_mode:
    config.set_arg_group("orange_mode")

config.set_kwargs(kwargs)

self.df = None
self._df_hash = -1
self._description_set = None
self._sample = sample
```

```
    self._title = None
    self._report = None
    self._html = None
    self._widgets = None
    self._json = None
    self._typeset = None
    self._summarizer = None

    if df is not None:
        # preprocess df
        self.df = self.preprocess(df)

    if not lazy:
        # Trigger building the report structure
        _ = self.report

def set_variable(self, key: str, value: Any):
    """Change a single configuration variable

    Args:
        key: configuration parameter name. Accepts nested syntax, e.g. "html.minify_html"
        value: the new value

    Examples:
        >>> ProfileReport(df).set_variables("title", "NewTitle")
        >>> ProfileReport(df).set_variables("html", {"minify_html": False})
        >>> ProfileReport(df).set_variables("html.minify_html", False)

    """
    keys = key.split(".")
    for e in reversed(keys[1:]):
        value = {e: value}

    self.set_variables(**{keys[0]: value})
```

```
def set_variables(self, **vars):
    """Change configuration variables (invalidates caches where necessary)

    Args:
        **vars: configuration parameters to change

    Examples:
        >>> ProfileReport(df).set_variables(title="NewTitle", html={"minify_html": False})
    """
    changed = set(vars.keys())
    if len({"progress_bar", "pool_size"} & changed) > 0:
        # Cache can persist
        pass

    if len({"notebook"} & changed) > 0:
        self._widgets = None

    if len({"html", "title"} & changed) > 0:
        self._html = None

    if not {"progress_bar", "pool_size", "notebook", "html", "title"} >= changed:
        # In all other cases, empty cache
        self._description_set = None
        self._title = None
        self._report = None
        self._html = None
        self._widgets = None
        self._json = None

    if len(vars) == 1:
        config[list(vars.keys())[0]] = list(vars.values())[0]
    else:
        config.set_kwargs(vars)
```

```
@property
def typeset(self):
    if self._typeset is None:
        self._typeset = ProfilingTypeSet()
    return self._typeset

@property
def summarizer(self):
    if self._summarizer is None:
        self._summarizer = PandasProfilingSummarizer(self.typeset)
    return self._summarizer

@property
def description_set(self):
    if self._description_set is None:
        self._description_set = describe_df(
            self.title, self.df, self.summarizer, self.typeset, self._sample
        )
    return self._description_set

@property
def title(self):
    if self._title is None:
        self._title = config["title"].get(str)

    return self._title

@property
def df_hash(self):
    if self._df_hash == -1 and self.df is not None:
        self._df_hash = hash_dataframe(self.df)
    return self._df_hash
```

```
@property
def report(self):
    if self._report is None:
        self._report = get_report_structure(self.description_set)
    return self._report

@property
def html(self):
    if self._html is None:
        self._html = self._render_html()
    return self._html

@property
def json(self):
    if self._json is None:
        self._json = self._render_json()
    return self._json

@property
def widgets(self):
    if self._widgets is None:
        self._widgets = self._render_widgets()
    return self._widgets

def get_duplicates(self, df=None) -> Optional[pd.DataFrame]:
    """Get duplicate rows and counts based on the configuration

    Args:
        df: Deprecated, for compatibility

    Returns:
        A DataFrame with the duplicate rows and their counts.
    """
    return self.description_set["duplicates"]
```

```
def get_sample(self, df=None) -> dict:
    """Get head/tail samples based on the configuration

    Args:
        df: Deprecated, for compatibility

    Returns:
        A dict with the head and tail samples.
    """
    return self.description_set["sample"]

def get_description(self) -> dict:
    """Return the description (a raw statistical summary) of the dataset.

    Returns:
        Dict containing a description for each variable in the DataFrame.
    """
    return self.description_set

def get_rejected_variables(self) -> set:
    """Get variables that are rejected for analysis (e.g. constant, mixed data types)

    Returns:
        a set of column names that are unsupported
    """
    return {
        message.column_name
        for message in self.description_set["messages"]
        if message.message_type == MessageType.REJECTED
    }

def to_file(self, output_file: Union[str, Path], silent: bool = True) -> None:
    """Write the report to a file.
```

```
By default a name is generated.

Args:
    output_file: The name or the path of the file to generate including the extension (.html, .json).
          silent: if False, opens the file in the default browser or download it in a Google Colab
environment
"""
if not isinstance(output_file, Path):
    output_file = Path(str(output_file))

if output_file.suffix == ".json":
    data = self.to_json()
else:
    inline = config["html"]["inline"].get(bool)
    if not inline:
        config["html"]["file_name"] = str(output_file)
        create_html_assets(output_file)

    data = self.to_html()

    if output_file.suffix != ".html":
        suffix = output_file.suffix
        output_file = output_file.with_suffix(".html")
        warnings.warn(
            f"Extension {suffix} not supported. For now we assume .html was intended. "
            f"To remove this warning, please use .html or .json."
        )

disable_progress_bar = not config["progress_bar"].get(bool)
with tqdm(
    total=1, desc="Export report to file", disable=disable_progress_bar
) as pbar:
    output_file.write_text(data, encoding="utf-8")
```

```
            pbar.update()

        if not silent:
            try:
                from google.colab import files

                files.download(output_file.absolute().as_uri())
            except ModuleNotFoundError:
                import webbrowser

                webbrowser.open_new_tab(output_file.absolute().as_uri())

    def _render_html(self):
        from pandas_profiling.report.presentation.flavours import HTMLReport

        report = self.report

        disable_progress_bar = not config["progress_bar"].get(bool)
        with tqdm(total=1, desc="Render HTML", disable=disable_progress_bar) as pbar:
            html = HTMLReport(report).render(
                nav=config["html"]["navbar_show"].get(bool),
                offline=config["html"]["use_local_assets"].get(bool),
                inline=config["html"]["inline"].get(bool),
                file_name=Path(config["html"]["file_name"].get(str)).stem,
                primary_color=config["html"]["style"]["primary_color"].get(str),
                logo=config["html"]["style"]["logo"].get(str),
                theme=config["html"]["style"]["theme"].get(str),
                title=self.description_set["analysis"]["title"],
                date=self.description_set["analysis"]["date_start"],
                version=self.description_set["package"]["pandas_profiling_version"],
            )

            minify_html = config["html"]["minify_html"].get(bool)
            if minify_html:
```

```
            from htmlmin.main import minify

            html = minify(html, remove_all_empty_space=True, remove_comments=True)
        pbar.update()
    return html

def _render_widgets(self):
    from pandas_profiling.report.presentation.flavours import WidgetReport

    report = self.report

    disable_progress_bar = not config["progress_bar"].get(bool)
    with tqdm(
        total=1, desc="Render widgets", disable=disable_progress_bar, leave=False
    ) as pbar:
        widgets = WidgetReport(report).render()
        pbar.update()
    return widgets

def _render_json(self):
    def encode_it(o):
        if isinstance(o, dict):
            return {encode_it(k): encode_it(v) for k, v in o.items()}
        else:
            if isinstance(o, (bool, int, float, str)):
                return o
            elif isinstance(o, list):
                return [encode_it(v) for v in o]
            elif isinstance(o, set):
                return {encode_it(v) for v in o}
            elif isinstance(o, (pd.DataFrame, pd.Series)):
                return o.to_json()
            elif isinstance(o, np.ndarray):
                return encode_it(o.tolist())
            else:
                return str(o)

    description = self.description_set

    disable_progress_bar = not config["progress_bar"].get(bool)
    with tqdm(total=1, desc="Render JSON", disable=disable_progress_bar) as pbar:
        description = format_summary(description)
        description = encode_it(description)
        data = json.dumps(description, indent=4)
        pbar.update()
    return data

def to_html(self) -> str:
    """Generate and return complete template as lengthy string
        for using with frameworks.

    Returns:
        Profiling report html including wrapper.

    """
    return self.html

def to_json(self) -> str:
    """Represent the ProfileReport as a JSON string

    Returns:
        JSON string
    """

    return self.json

def to_notebook_iframe(self):
    """Used to output the HTML representation to a Jupyter notebook.
    When config.notebook.iframe.attribute is "src", this function creates a temporary HTML file
    in `./tmp/profile_[hash].html` and returns an Iframe pointing to that contents.
      When config.notebook.iframe.attribute is "srcdoc", the same HTML is injected in the "srcdoc"
attribute of
    the Iframe.

    Notes:
        This constructions solves problems with conflicting stylesheets and navigation links.
    """
    from IPython.core.display import display

    from pandas_profiling.report.presentation.flavours.widget.notebook import (
        get_notebook_iframe,
    )

    # Ignore warning: https://github.com/ipython/ipython/pull/11350/files
    with warnings.catch_warnings():
        warnings.simplefilter("ignore")
        display(get_notebook_iframe(self))

def to_widgets(self):
    """The ipython notebook widgets user interface."""
    try:
        from google.colab import files

        warnings.warn(
                                        "Ipywidgets is not yet fully supported on Google Colab
(https://github.com/googlecolab/colabtools/issues/60)."
                      "As an alternative, you can use the HTML report. See the documentation for more
information."
        )
    except ModuleNotFoundError:
        pass
    from IPython.core.display import display

    display(self.widgets)

def _repr_html_(self):
    """The ipython notebook widgets user interface gets called by the jupyter notebook."""
    self.to_notebook_iframe()

def __repr__(self):
    """Override so that Jupyter Notebook does not print the object."""
    return ""

@staticmethod
def preprocess(df):
    """Preprocess the dataframe

    - Appends the index to the dataframe when it contains information
    - Rename the "index" column to "df_index", if exists
    - Convert the DataFrame's columns to str

    Args:
        df: the pandas DataFrame

    Returns:
        The preprocessed DataFrame
    """
    # Treat index as any other column
    if (
        not pd.Index(np.arange(0, len(df))).equals(df.index)
        or df.index.dtype != np.int64
    ):
        df = df.reset_index()

    # Rename reserved column names
```

```
        df = rename_index(df)

        # Ensure that columns are strings
        df.columns = df.columns.astype("str")
        return df

    @staticmethod
    def clear_config():
        """
        Restore the configuration to default
        """
        config.clear()
```

correlations.py

```
"""Correlations between variables."""
import itertools
import warnings
from typing import Dict, List, Optional

import numpy as np
import pandas as pd
from pandas.core.base import DataError
from scipy import stats

from pandas_profiling.config import config
from pandas_profiling.model.typeset import Boolean, Categorical, Numeric, Unsupported


class Correlation:
    @staticmethod
    def compute(df, summary) -> Optional[pd.DataFrame]:
        return NotImplemented
```

```
class Spearman(Correlation):
    @staticmethod
    def compute(df, summary) -> Optional[pd.DataFrame]:
        return df.corr(method="spearman")


class Pearson(Correlation):
    @staticmethod
    def compute(df, summary) -> Optional[pd.DataFrame]:
        return df.corr(method="pearson")


class Kendall(Correlation):
    @staticmethod
    def compute(df, summary) -> Optional[pd.DataFrame]:
        return df.corr(method="kendall")


class Cramers(Correlation):
    @staticmethod
    def _cramers_corrected_stat(confusion_matrix, correction: bool) -> float:
        """Calculate the Cramer's V corrected stat for two variables.

        Args:
            confusion_matrix: Crosstab between two variables.
            correction: Should the correction be applied?

        Returns:
            The Cramer's V corrected stat for the two variables.
        """
        chi2 = stats.chi2_contingency(confusion_matrix, correction=correction)[0]
        n = confusion_matrix.sum().sum()
        phi2 = chi2 / n
        r = confusion_matrix.shape[0]
        k = confusion_matrix.shape[1] if len(confusion_matrix.shape) > 1 else 1

        # Deal with NaNs later on
        with np.errstate(divide="ignore", invalid="ignore"):
            phi2corr = max(0.0, phi2 - ((k - 1.0) * (r - 1.0)) / (n - 1.0))
            rcorr = r - ((r - 1.0) ** 2.0) / (n - 1.0)
            kcorr = k - ((k - 1.0) ** 2.0) / (n - 1.0)
            rkcorr = min((kcorr - 1.0), (rcorr - 1.0))
            if rkcorr == 0.0:
                corr = 1.0
            else:
                corr = np.sqrt(phi2corr / rkcorr)
        return corr

    @staticmethod
    def compute(df, summary) -> Optional[pd.DataFrame]:
        threshold = config["categorical_maximum_correlation_distinct"].get(int)

        categoricals = {
            key
            for key, value in summary.items()
            if value["type"] in {Categorical, Boolean}
            and value["n_distinct"] <= threshold
        }

        if len(categoricals) <= 1:
            return None

        matrix = np.zeros((len(categoricals), len(categoricals)))
        np.fill_diagonal(matrix, 1.0)
        correlation_matrix = pd.DataFrame(
            matrix,
            index=categoricals,
            columns=categoricals,
        )

        for name1, name2 in itertools.combinations(categoricals, 2):
            confusion_matrix = pd.crosstab(df[name1], df[name2])
            correlation_matrix.loc[name2, name1] = Cramers._cramers_corrected_stat(
                confusion_matrix, correction=True
            )
            correlation_matrix.loc[name1, name2] = correlation_matrix.loc[name2, name1]
        return correlation_matrix


class PhiK(Correlation):
    @staticmethod
    def compute(df, summary) -> Optional[pd.DataFrame]:
        threshold = config["categorical_maximum_correlation_distinct"].get(int)
        intcols = {
            key
            for key, value in summary.items()
            # DateTime currently excluded
            # In some use cases, it makes sense to convert it to interval
            # See https://github.com/KaveIO/PhiK/issues/7
            if value["type"] == Numeric and 1 < value["n_distinct"]
        }

        selcols = {
            key
            for key, value in summary.items()
            if value["type"] != Unsupported and 1 < value["n_distinct"] <= threshold
        }
        selcols = selcols.union(intcols)

        if len(selcols) <= 1:
            return None
        with warnings.catch_warnings():
            warnings.simplefilter("ignore")
            import phik

            correlation = df[selcols].phik_matrix(interval_cols=intcols)

        return correlation


def warn_correlation(correlation_name: str, error):
    warnings.warn(
        f"""There was an attempt to calculate the {correlation_name} correlation, but this failed.
To hide this warning, disable the calculation
(using `df.profile_report(correlations={{\"{correlation_name}\": {{\"calculate\": False}}}})`
If this is problematic for your use case, please report this as an issue:
https://github.com/pandas-profiling/pandas-profiling/issues
(include the error message: '{error}')"""
    )


def calculate_correlation(
    df: pd.DataFrame, correlation_name: str, summary: dict
) -> Optional[pd.DataFrame]:
     """Calculate the correlation coefficients between variables for the correlation types selected in the
config
    (pearson, spearman, kendall, phi_k, cramers).

    Args:
        df: The DataFrame with variables.
        correlation_name:
        summary: summary dictionary

    Returns:
        The correlation matrices for the given correlation measures. Return None if correlation is empty.
    """

    correlation_measures = {
        "pearson": Pearson,
        "spearman": Spearman,
        "kendall": Kendall,
        "cramers": Cramers,
        "phi_k": PhiK,
    }

    correlation = None
    try:
        correlation = correlation_measures[correlation_name].compute(df, summary)
    except (ValueError, AssertionError, TypeError, DataError, IndexError) as e:
        warn_correlation(correlation_name, e)

    if correlation is not None and len(correlation) <= 0:
        correlation = None

    return correlation


def perform_check_correlation(
    correlation_matrix: pd.DataFrame, threshold: float
) -> Dict[str, List[str]]:
    """Check whether selected variables are highly correlated values in the correlation matrix.

    Args:
        correlation_matrix: The correlation matrix for the DataFrame.
        threshold:.

    Returns:
        The variables that are highly correlated.
    """
```

```
cols = correlation_matrix.columns
bool_index = abs(correlation_matrix.values) >= threshold
np.fill_diagonal(bool_index, False)
return {
    col: cols[bool_index[i]].values.tolist()
    for i, col in enumerate(cols)
    if any(bool_index[i])
}
```

Github - IBM-EPBL/SI-GuidedProject-37547-1665844488: Corporate Employee Attrition Analytics (github.com)

Demo - https://drive.google.com/file/d/1QgrSjEVU-0U7tlLpop72OKbo5xA5wN-G/view?usp=sharing