1. Algorithm

Problem Statement:

Loop would like to offer their employees an allowance to live a healthier lifestyle, and improve the physical health of their employees. However, the organization cannot afford to give this allowance to all employees and have requested the Data Science team to help identify the employees that will benefit most from this.

The Data Science team collected a dataset relevant to the problem statement, that they believe can help them predict the likelihood of an employee being at risk for heart disease, which they believe will be a good indicator of employees that will benefit from this health-allowance .

The dataset was collected by asking the following questions to a certain group of employees:

- **EmployeeID:** Unique identifier of an employee.
- **HeartDisease:** Respondents that have ever reported having a heart disease.
- **BMI:** Body Mass Index (BMI)
- **Smoking:** Have you smoked at least 100 cigarettes in your entire life?
- **AlcoholDrinking:** Are you a heavy drinker? (more than 14 drinks per week and adult women having more than 7 drinks per week)
- **Stroke:** (Ever told) (you had) a stroke?
- **PhysicalHealth:** How many days during the past 30 days was your physical health not good? (0-30 days)
- **MentalHealth:** How many days during the past 30 days was your mental health not good? (0-30 days)
- **DiffWalking:** Do you have serious difficulty walking or climbing stairs?
- **Sex:** Are you male or female?
- **AgeCategory:** Fourteen-level age category
- **Diabetic:** Ever told you had diabetes?
- **PhysicalActivity:** Doing physical activity or exercise during the past 30 days.
- **SleepTime:** On average, how many hours of sleep do you get in a 24-hour period?
- **Asthma:** Ever told you had asthma?
- **KidneyDisease:** Were you ever told you had kidney disease?
- **SkinCancer:** Ever told you had skin cancer?

GOAL: *Build a Machine Learning model to predict the PROBABILITY of an employee being at risk of a heart disease.*

Please follow the following steps to help Loop solve the problem:

1. Is this problem suitable for supervised or unsupervised learning, and why?.
2. Identify the target column and assign it to y.
3. The target column is provided in a string format, convert this to numeric format and motivate why this step is important for the classifier model.
4. Is there any class imbalance in the data? If yes, showcase how you handled this. If no, please explain how you would handle it.
5. Identify columns from the data set that you will use as features for the model, please motivate columns that you REMOVE, if any.
6. Perform feature engineering from the existing data that can potentially improve the performance of the classifier model. This can include creating interaction features or statistical features.
7. Using your own discretion, apply any transformation(s) to the dataset so that it can be used for the classifier model and motivate each transformation in the comments.
8. Create visualizations that will help you determine if there are correlations between the features you created for this model. Also motivate why you removed (if any) or kept some features for the classifier model.
9. Create test and training data sets.
10. Evaluate the model's performance using accuracy, precision, recall and F-1 score in predicting a case of HeartDisease.
11. Apply any steps (please explain the steps in the comments) you see fit to improve the initial model's output.
12. Motivate if the model in its current form will be sufficient for its intended purpose or not.
13. Briefly discuss the issue of overfitting and how to mitigate it.
14. Besides using Machine Learning to solve Loop's problem statement, what alternative solutions can you suggest to Loop to solve this problem?.

Feel free to make any assumptions about the data while coming up with your solution.

Tech Stack:

- Please use Python to solve this problem and save your code with answers in this format loop_assessment_answers.py

2. SQL

2.1

Given a table named orders in BigQuery (or any SQL data warehouse) with the following schema:

- client_id: STRING (eg. 123XPH)
- branch_name: STRING (e.g CPT)
- order_date: TIMESTAMP (eg. 2023-05-01 15:23:15)

Write a SQL query to calculate the number of days since the last order from the current date for each client at each branch. Categorize the number of days into different week ranges, for example 2-3 weeks, 3-4 weeks e.t.c. The result should include the following columns: **client_id, branch_name, last_order_date, n_days_since_last_order, and days_since_last_order_category**. Sort the result in descending order of n_days_since_last_order.

You can use any SQL service of your choice to test the output of the query.

2.2

- The SQL script below aims to compare the actual vs predicted hourly number of orders per branch per weekday.
- The script compares the predicted orders with the actual number of orders received in the past 7 days per hour per branch and weekday. It then marks the rest as next 7 days predictions.
- There are some **INTENTIONAL ERRORS present** in the script.
- Please **find and fix the errors** to ensure that the script runs properly and produces the desired output.
- The expected resulting columns are: **hour, branch, weekday, predicted_orders, actual_orders, prediction_error and prediction_period_indicator**
- You are welcome to make any assumptions regarding the data.

```sql
SELECT *,
 CASE
    WHEN
        DATETIME(Hour) BETWEEN DATETIME(CURRENT_TIMESTAMP())
        AND TIMESTAMP_ADD(DATETIME(CURRENT_TIMESTAMP()), INTERVAL 169 HOUR) THEN 'next_7_days'
    ELSE 'previous_7_days'
 END AS prediction_period_indicator
FROM (
    SELECT
       predicted_orders.weekday,
       predicted_orders.Hour,
       predicted_orders.branch,
       predicted_orders.predicted_orders,
       actual_orders.actual_orders,
       ABS((actual_orders.actual_orders -
predicted_orders.predicted_orders)/actual_orders.actual_orders) AS prediction_error
    FROM (
        SELECT
           DATETIME_TRUNC(DATETIME(order_date, 'Africa/Johannesburg'), HOUR) AS Hour,
           branch,
           FORMAT_DATETIME('%A', DATETIME(order_date, 'Africa/Johannesburg')) AS weekday
        FROM predicted_orders
    ) predicted_orders
    LEFT JOIN (
        SELECT
           DATETIME_TRUNC(DATETIME(order_date, 'Africa/Johannesburg'), HOUR) AS Hour,
           branch,
           FORMAT_DATETIME('%A', DATETIME(order_date, 'Africa/Johannesburg')) AS weekday,
           COUNT(DISTINCT order_id) AS actual_orders
        FROM actual_orders
        GROUP BY 1, 2, 3
    ) actual_orders
    ON predicted_orders.branch = actual_orders.branch
    AND predicted_orders.Hour = actual_orders.Hour
    AND predicted_orders.weekday = actual_orders.weekday
) resulting_table
WHERE
    DATETIME(Hour) BETWEEN DATETIME(CURRENT_TIMESTAMP())
    AND TIMESTAMP_ADD(DATETIME(CURRENT_TIMESTAMP()), INTERVAL 169 HOUR)
    AND TIME(Hour) >= '10:00:00' AND TIME(Hour) < '23:00:00'
    OR DATETIME(Hour) >= TIMESTAMP_ADD(DATETIME(CURRENT_TIMESTAMP()), INTERVAL -169 HOUR)
    AND TIME(Hour) >= '10:00:00' AND TIME(Hour) < '23:00:00';
```