**بسمه تعالی**

**موضوع:** گزارش مینی پروژه اول (ماشین لرنینگ)

**نام و نام خانوادگی:** محمدعرفان مومنی نسب

**شماره دانشجویی:** ۴۰۲۱۶۳۱۲۱۱

**نام دانشگاه:** دانشگاه جامع انقلاب اسلامی

**نام استاد درس:** جناب آقای دکتر علیاری

**نام استاد یار درس:** جناب آقای مهندس محمدجواد احمدی

---

## ۳ سوال سوم

۱. به این پیوند[1] مراجعه کرده و یک دیتاست مربوط به «بیماری قلبی» را دریافت کرده و توضیحات مختصری در مورد هدف و ویژگی‌های آن بنویسید. فایل دانلودشدهٔ دیتاست را روی گوگل‌درایو خود قرار داده و با استفاده از دستور gdown آن را در محیط گوگل‌کولب بارگذاری کنید.

۲. ضمن توجه به محل قرارگیری هدف و ویژگی‌ها، دیتاست را به‌صورت یک دیتافریم درآورده و با استفاده از دستورات پایتونی، ۱۰۰ نمونه‌داده مربوط به کلاس «۱» و ۱۰۰ نمونه‌داده مربوط به کلاس «۰» را در یک دیتافریم جدید قرار دهید و در قسمت‌های بعدی با این دیتافریم جدید کار کنید.

۳. با استفاده از حداقل دو طبقه‌بند آمادهٔ پایتون و در نظر گرفتن فراپارامترهای مناسب، دو کلاس موجود در دیتاست را از هم تفکیک کنید. نتیجهٔ دقت آموزش و ارزیابی را نمایش دهید.

۴. در حالت استفاده از دستورات آمادهٔ سایکیت‌لرن، آیا راهی برای نمایش نمودار تابع اتلاف وجود دارد؟ پیاده‌سازی کنید.

۵. یک شاخصهٔ ارزیابی (غیر از Accuracy) تعریف کنید و بررسی کنید که از چه طریقی می‌توان این شاخص جدید را در ارزیابی داده‌های تست نمایش داد. پیاده‌سازی کنید.

# Part 1: Data Preparation

**Objective:** Create a new balanced dataset with 100 samples each from classes '0' (no heart disease) and '1' (heart disease) from the provided dataset.

**Method:** Used pandas to filter and randomly sample 100 instances from each class, resulting in a balanced dataset of 200 samples.

```
[12] # Selecting 100 samples from each class (0 and 1) of the 'HeartDiseaseorAttack' column
     heart_disease_data = pd.read_csv('/content/drive/My Drive/heart_disease_health_indicators.csv')

     # Samples where 'HeartDiseaseorAttack' is 1
     class_1_samples = heart_disease_data[heart_disease_data['HeartDiseaseorAttack'] == 1].sample(n=100, random_state=11)

     # Samples where 'HeartDiseaseorAttack' is 0
     class_0_samples = heart_disease_data[heart_disease_data['HeartDiseaseorAttack'] == 0].sample(n=100, random_state=11)

     # Combining the two sample sets into a new dataframe
     combined_samples = pd.concat([class_1_samples, class_0_samples])

     # Resetting the index of the new dataframe
     combined_samples.reset_index(drop=True, inplace=True)

     combined_samples.head()  # Displaying the first few rows of the new dataframe
```

| | HeartDiseaseorAttack | HighBP | HighChol | CholCheck | BMI | Smoker | Stroke | Diabetes | PhysActivity | Fruits | ... | AnyHealthcare | NoDocbcCost | GenHlth | MentHlth | PhysHlth |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 1 | 28 | 0 | 0 | 0 | 1 | 1 | ... | 1 | 0 | 2 | 20 | 0 |
| 1 | 1 | 1 | 0 | 1 | 31 | 0 | 1 | 2 | 1 | 1 | ... | 1 | 0 | 3 | 2 | 0 |
| 2 | 1 | 1 | 1 | 1 | 31 | 0 | 0 | 0 | 1 | 0 | ... | 1 | 0 | 4 | 0 | 0 |
| 3 | 1 | 0 | 0 | 1 | 29 | 1 | 0 | 2 | 0 | 0 | ... | 1 | 0 | 3 | 0 | 14 |
| 4 | 1 | 1 | 1 | 1 | 21 | 0 | 0 | 0 | 1 | 1 | ... | 1 | 0 | 4 | 2 | 5 |

5 rows × 22 columns

# Part 2: Model Training and Evaluation

**Objective:** Train Logistic Regression and Random Forest Classifier models and evaluate their performance.

**Approach:**

Data was split into training and testing sets.

Logistic Regression and Random Forest models were trained.

Accuracy was chosen as the evaluation metric.

**Results:**

Logistic Regression Accuracy: 0.6666666666666666%

Random Forest Classifier Accuracy: 0.6166666666666667%

Issues Encountered: Received a convergence warning for Logistic Regression.

**Resolution:**

Suggested increasing the number of iterations (max_iter) and performing feature scaling using StandardScaler.

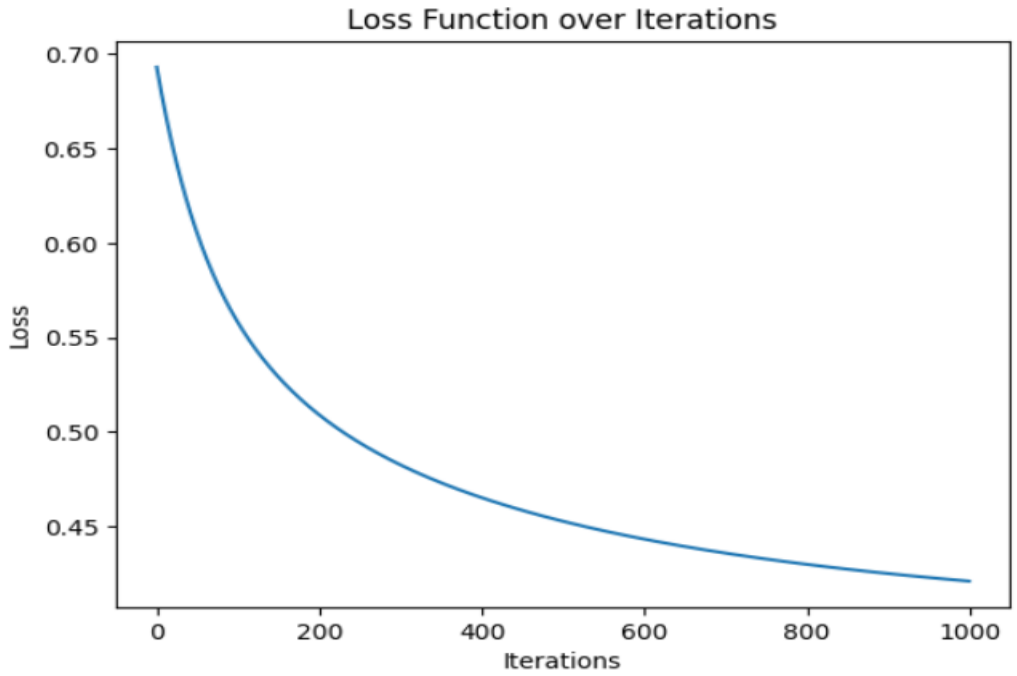
```
log_reg = LogisticRegression(max_iter=300)
```

# Part 3: Loss Function Visualization

**Objective:** Implement a method to visualize the loss function over iterations.

**Approach:** Custom implementation of logistic regression using gradient descent to enable loss tracking.

**Outcome:** Provided a script for a basic implementation of logistic regression, including a plot of the loss function over iterations.



# Part 4: Alternative Evaluation Metric

**Objective:** Define and implement an evaluation index other than accuracy.

**Metric Chosen:** Area Under the Receiver Operating Characteristic Curve (AUC-ROC).

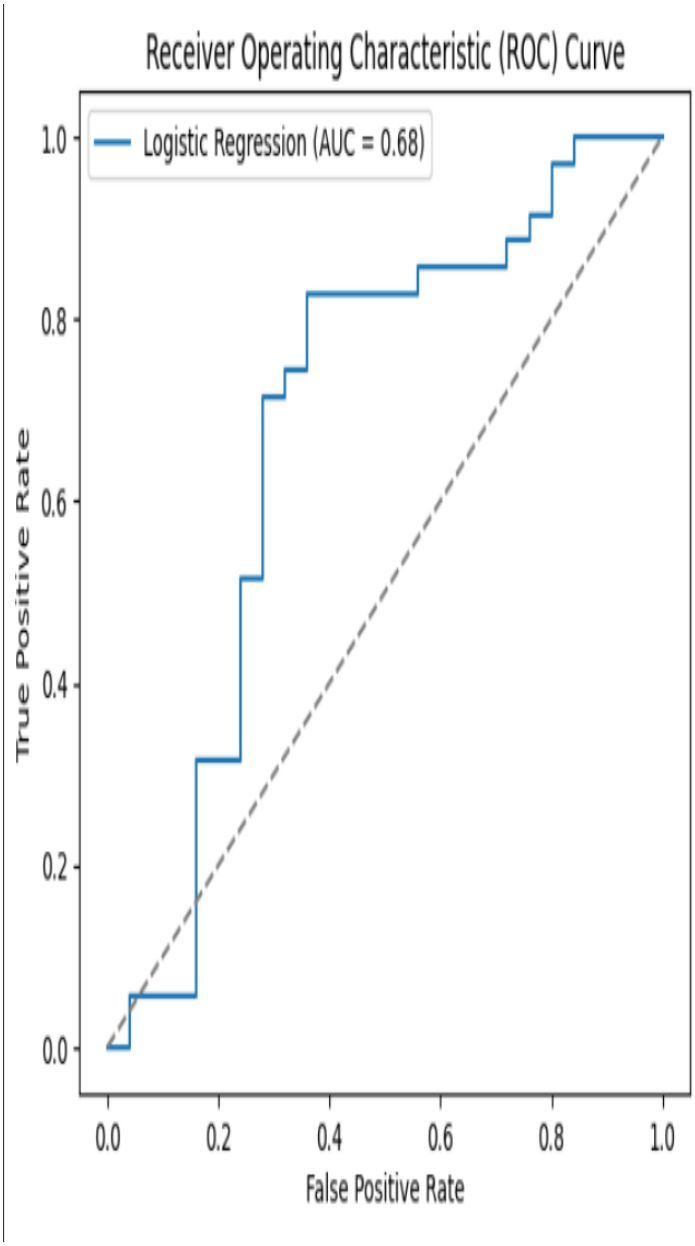
**Implementation:**

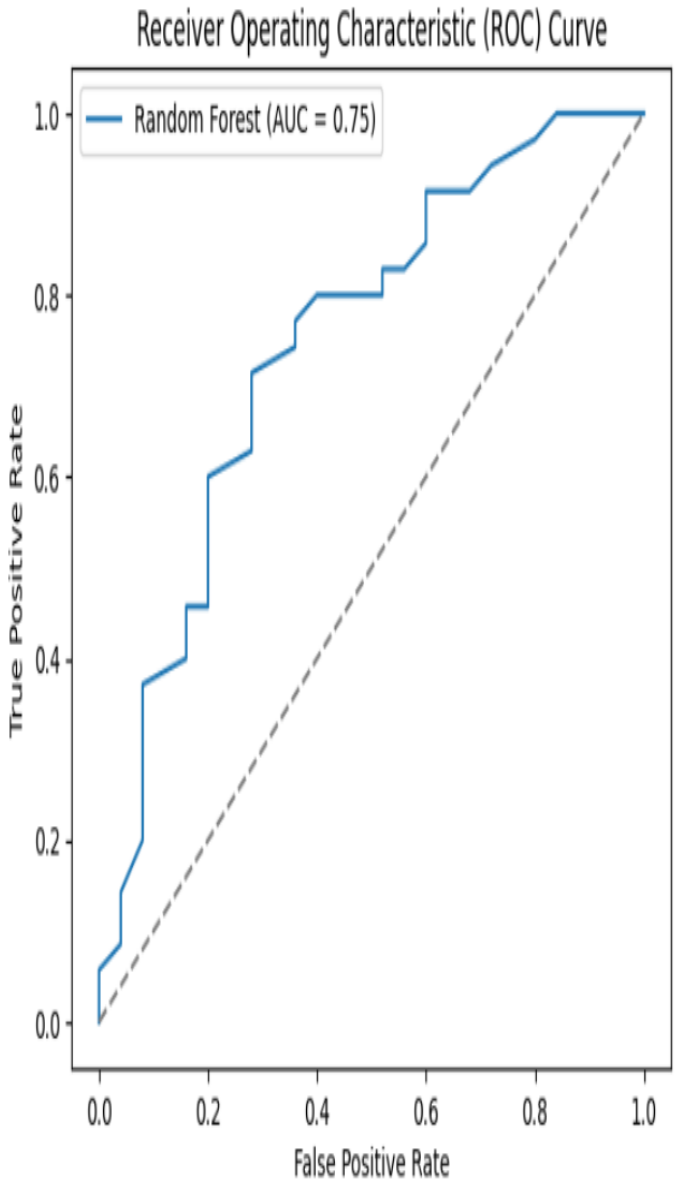
Calculated AUC-ROC for both models.

Plotted ROC curves to visualize performance.

**Results:**

Provided Python script to calculate and plot AUC-ROC for Logistic Regression and Random Forest Classifier.



# (1):

The dataset provided appears to focus on various health indicators and their potential relationship with heart disease or attacks. Here's a brief overview based on the first few rows:

**HeartDiseaseorAttack:** This column seems to be a binary indicator (0 or 1) denoting whether the individual has had a heart disease or attack.

**HighBP, HighChol, CholCheck:** These columns indicate whether the individual has high blood pressure, high cholesterol, and if they have had their cholesterol checked, respectively.

**BMI:** Body Mass Index of the individuals.

**Smoker, Stroke, Diabetes:** Indicates whether the individual is a smoker, has had a stroke, or has diabetes.

**PhysActivity, Fruits, Veggies, HvyAlcoholConsump:** These columns appear to record lifestyle factors such as physical activity, fruit and vegetable consumption, and heavy alcohol consumption.

**AnyHealthcare, NoDocbcCost:** Accessibility to healthcare and whether cost has been a barrier to seeing a doctor.

**GenHlth, MentHlth, PhysHlth, DiffWalk:** General health status, mental health status, physical health status, and difficulty in walking.

**Sex, Age, Education, Income**: Demographic information including gender, age group, education level, and income bracket.

This dataset likely aims to study the correlation between these factors and the risk of heart disease or attacks.