Logbook Science Fair

My Science Fair Project involves a heart attack prediction model created in Google Colab Notebooks utilizing Python to generate a desired outcome(a high accuracy Heart Attack Prediction model). This model has been developed into an app where a user may engage in the platform to understand their own risk.

September 14th:

I started re-familiarizing myself with last years project.

What I learned from Last year:

1. I need diversity in datasets
2. I need an app or finished product
3. I need to practice presentations more (Add a hook, intro, and strong conclusion)

September 15th:

I looked at projects currently around online, and looked at strong resources like MD Calc, like Framingham, and other prediction calculators online. One thing I

notice when I look at tools like MD Calc with prediction software on issues like CHADS2 prediction, it s a point system. The prediction works in a manner where yes or no answers are allowed, in which the user gets a +1 point for yes, and +0 for no. at the end of the test, difference point scores equal to different risk percentages. this is cool, however, its old, and groups peoples health together into yes and no possibility. in todays age, we need personalized medicine because, people respond to treatment in different ways, have different bodies, lives, habits, and bodily function execution.

> “It is more important to know what sort of person has a disease than to know what sort of disease a person has”

Hippocrates

October - November

I am focused on building up my model and identifying how I can achieve strong accuracies. I have tried methods like GridSearchCV, and Random Forest. Today I am trying to operate SVM. Over the course of the month I hope to work with Logistic Regression, RandomBoost, XGB, and random trainers to elevate model accuracy.

Results List

- 90 percent total accuracy
- 91 percent total accuracy
- 92.74 percent total accuracy
- 94 percent total accuracy
- 100 percent accuracy → 74 percent accuracy

- 81 percent accuracy
- 93 percent accuracy

November 2nd

I have been thinking about how I can elevate my model? Its a model but I should be able to turn it into something. But how? What do I turn it into? I have been wanting to turn it into an app, but I am unsure how to do that. I think Xcode may work out well.

I also wonder what happens to the heart post cardiac event.

Repair

November 11th

I have been researching different platforms that I can work with to develop an app. I will work with Plotlee, Render, Gradio, and Streamlit.

I am going to try Gradio today.

Gradio did not work to my liking. It was very difficult to develop the model. Turning my specific model from colab notebooks directly into the app was defintely a struggle. I have to move to the next on the list. Plotlee and render look more or less the same. I want to try Streamlit.

Rest of November

I am going to be continuing to look at research, publications, studies and youtube video indicating which software would be ideal for my model and prediction. I think Streamlit is looking like a strong option. There is so much potential to work with a platform like streamlit. I will start making the app tomorrow. I have all the information I need. It is really cool because I edit the text and fonts, and

backgrounds and elements through code. I love vs code. its going to be perfect to work with streamlit.

February

- Learned about DL - Deep Learning

DL

- School Science fair
  - needed more to my project
    - braintstorm
- what can be added
  - ECG reading? Better accuracy? interactive results? info page? Heart health tips? Arrhythmia detection? DL model?
    - Lets implement these

March

- Found strong project: https://github.com/mollenhauerm/ECG-heartbeat-classification/blob/master/README.md
  - Needs adjusting, sits as 88 percent to 91 percent accuracy
    - What can we do for more results?

Neural Networks or improving the preprocessing and feature engineering can significantly increase the accuracy of **Model 2 (Unbiased Model)**. It's important to achieve higher accuracy while maintaining balanced performance across all classes.

Why Neural Networks?

Neural networks, especially **1D Convolutional Neural Networks (CNNs)** or **Recurrent Neural Networks (RNNs)**, are well-suited for ECG classification because:

1. **Automatic Feature Extraction**: CNNs can automatically learn relevant features from raw ECG signals, reducing the need for manual feature engineering.
2. **Handling Sequential Data**: RNNs or LSTMs can capture temporal dependencies in ECG signals.
3. **High Performance**: Neural networks often outperform traditional machine learning models like logistic regression for complex datasets.

Steps to Improve Accuracy:

1. Switch to a Neural Network Model:

- Replace the current RBF + logistic regression pipeline with a **1D CNN** or **LSTM** model.
- CNNs are particularly effective for ECG classification because they can capture local patterns in the signal.

2. Improve Preprocessing:

- Normalize the ECG signals to a consistent scale (e.g., `[0, 1]` or `[-1, 1]`).
- Apply noise reduction techniques (e.g., Gaussian smoothing, wavelet transforms).
- Use data augmentation to artificially increase the size of the training dataset.

3. Address Class Imbalance:

- Use techniques like **oversampling** (e.g., SMOTE) or **class weighting** to ensure the model learns from all classes equally.

4. Hyperparameter Tuning:

- Experiment with different architectures, learning rates, batch sizes, and activation functions to optimize the neural network.

This worked

Bits of code from CNN for image analysis brought my model to a 98.2% accuracy!

```
# Define the CNN model
model = Sequential([
    Conv1D(filters=64, kernel_size=3, activation='relu', input_shape=(x_train_cnn.s
    MaxPooling1D(pool_size=2),
    Conv1D(filters=128, kernel_size=3, activation='relu'),
    MaxPooling1D(pool_size=2),
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(len(label_encoder.classes_), activation='softmax')  # Output layer
])

# Compile the model
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metric

# Print model summary
model.summary()
```

Result from Code Cell 2

Model: "sequential"

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 111, 64)	256
max_pooling1d (MaxPooling1D)	(None, 55, 64)	0
conv1d_1 (Conv1D)	(None, 53, 128)	24,704
max_pooling1d_1 (MaxPooling1D)	(None, 26, 128)	0
flatten (Flatten)	(None, 3328)	0
dense (Dense)	(None, 128)	426,112
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 5)	645

Total params: 451,717 (1.72 MB)
Trainable params: 451,717 (1.72 MB)
Non-trainable params: 0 (0.00 B)

Final Results

```
Epoch 1/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 32s 11ms/step - accuracy: 0.9112 - loss: 0.3244 - val_accuracy: 0.9581 - val_loss: 0.1525
Epoch 2/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 32s 12ms/step - accuracy: 0.9621 - loss: 0.1394 - val_accuracy: 0.9711 - val_loss: 0.1134
Epoch 3/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 28s 10ms/step - accuracy: 0.9692 - loss: 0.1097 - val_accuracy: 0.9723 - val_loss: 0.0995
Epoch 4/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 29s 10ms/step - accuracy: 0.9737 - loss: 0.0927 - val_accuracy: 0.9739 - val_loss: 0.0948
Epoch 5/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 31s 11ms/step - accuracy: 0.9757 - loss: 0.0873 - val_accuracy: 0.9765 - val_loss: 0.0865
Epoch 6/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 30s 11ms/step - accuracy: 0.9773 - loss: 0.0792 - val_accuracy: 0.9754 - val_loss: 0.0922
Epoch 7/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 29s 11ms/step - accuracy: 0.9786 - loss: 0.0722 - val_accuracy: 0.9779 - val_loss: 0.0890
Epoch 8/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 29s 10ms/step - accuracy: 0.9792 - loss: 0.0689 - val_accuracy: 0.9789 - val_loss: 0.0798
Epoch 9/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 28s 10ms/step - accuracy: 0.9797 - loss: 0.0655 - val_accuracy: 0.9798 - val_loss: 0.0860
Epoch 10/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 27s 10ms/step - accuracy: 0.9812 - loss: 0.0620 - val_accuracy: 0.9801 - val_loss: 0.0789
Epoch 11/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 28s 10ms/step - accuracy: 0.9820 - loss: 0.0583 - val_accuracy: 0.9802 - val_loss: 0.0845
Epoch 12/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 28s 10ms/step - accuracy: 0.9822 - loss: 0.0558 - val_accuracy: 0.9797 - val_loss: 0.0850
Epoch 13/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 31s 11ms/step - accuracy: 0.9834 - loss: 0.0520 - val_accuracy: 0.9804 - val_loss: 0.0827
Epoch 14/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 28s 10ms/step - accuracy: 0.9848 - loss: 0.0487 - val_accuracy: 0.9814 - val_loss: 0.0848
Epoch 15/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 27s 10ms/step - accuracy: 0.9851 - loss: 0.0479 - val_accuracy: 0.9805 - val_loss: 0.0892
Epoch 16/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 33s 12ms/step - accuracy: 0.9848 - loss: 0.0480 - val_accuracy: 0.9795 - val_loss: 0.0906
Epoch 17/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 31s 11ms/step - accuracy: 0.9845 - loss: 0.0474 - val_accuracy: 0.9802 - val_loss: 0.0924
Epoch 18/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 29s 11ms/step - accuracy: 0.9855 - loss: 0.0438 - val_accuracy: 0.9804 - val_loss: 0.0848
Epoch 19/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 29s 11ms/step - accuracy: 0.9858 - loss: 0.0419 - val_accuracy: 0.9819 - val_loss: 0.0938
Epoch 20/20
2737/2737 ━━━━━━━━━━━━━━━━━━━━ 28s 10ms/step - accuracy: 0.9869 - loss: 0.0390 - val_accuracy: 0.9819 - val_loss: 0.0853
```

End

```
685/685 ━━━━━━━━━━━━━━━━━━━━ 2s 3ms/step - accuracy: 0.9937 - loss: 0.0288
Test Accuracy: 98.19%
```

New Approach for Heart Attack Prediction → fine tuning a better model

Currently, my model stands at 95 percent accuracy as a machine learning model with a 300 sample size dataset from KAGGLE. This isnt good enough

Its a ML model which is shallow and wouldnt be amazing for clinical application because of how it trains over the information from the data, and how it deals with

patient input. This leads me to thinking a DL model would be more sufficient for a clinical.

1. DL model would be better as long as we find new dataset.

January 28th

1. Today I have been working on analyzing approaches that I can use for my DL model
    a. Nothing related to DL for non CNN use has been actually attempted or published or found online for Heart Attack Prediction
        i. This is great for me as it allows me opportunity to be the first and get great results.

Datasets:

1. I could only find 1 relevant dataset with major application into my model that doesnt come from Kaggle: https://data.mendeley.com/datasets/wmhctcrt5v/1

The features in this dataset are:

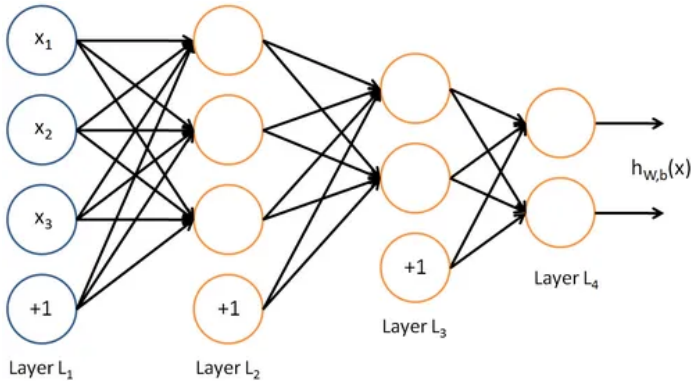
- age
- gender
- heart rate
- Systolic blood pressure
- Diastolic blood pressure
- Blood sugar
- ck-mb
- troponin ******* VERY HELPFUL FOR INDICATING RESULT COLUMN.
- and then the TARGET = result with negative or positive output indicating whether or not a heart attack took place

Feb 1st

1. Well know i need to fully understand which NN or DL method to use for my project
    a. Here are the options for now
        i. DNN → Deep Neural Network
        ii. LSTM → Long Short Term
        iii. MLP → Multi-layer Perceptron

Feb 7th

How NN works

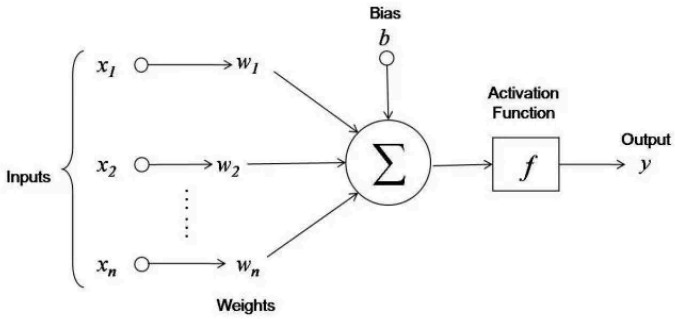


Here we have 4 key layers, the input, next, next, next then the output with the forward function.

Neural networks can usually be read from left to right. Here, the first layer is the layer in which inputs are entered. There are 2 internals layers (called hidden layers) that do some math, and one last layer that contains all the possible outputs. Don’t bother with the “+1”s at the bottom of every columns. It is something called “bias” and we’ll talk about that later.

Feb 20th

What does the Neuron do?



Feb 22

Learning how to code a basic NN

Basic Code for NN architecture

```
class NeuralNetwork(nn.Module):
    def __init__(self):
        super().__init__()
        self.flatten = nn.Flatten()
        self.linear_relu_stack = nn.Sequential(
            nn.Linear(28*28, 512),
            nn.ReLU(),
            nn.Linear(512, 512),
            nn.ReLU(),
            nn.Linear(512, 10),
        )
```

```
    def forward(self, x):
        x = self.flatten(x)
        logits = self.linear_relu_stack(x)
        return logits

model = NeuralNetwork().to(device)
print(model)
```

We define our neural network by subclassing `nn.Module`, and initialize the neural network layers in `__init__`. Every `nn.Module` subclass implements the operations on input data in the `forward` method.

We create an instance of `NeuralNetwork`, and move it to the `device`, and print its structure.

Feb 25th

How to calculate accuracy measures and training loops

```
# Initialize
model = UltimateHeartAttackModel()
criterion = nn.BCEWithLogitsLoss()
optimizer = optim.Adam(model.parameters(), lr=0.0005)
scheduler = optim.lr_scheduler.StepLR(optimizer, step_size=300, gamma=0.7)  #

# Training loop
for epoch in range(2000):  # You can go higher if needed
    model.train()
    optimizer.zero_grad()
    outputs = model(X_train_tensor)
    loss = criterion(outputs, y_train_tensor)
    loss.backward()
    optimizer.step()
    scheduler.step()
```

```
    if (epoch+1) % 100 == 0:
        print(f"Epoch {epoch+1}, Loss: {loss.item():.4f}")

# Evaluation
model.eval()
with torch.no_grad():
    outputs = model(X_test_tensor)
    probs = torch.sigmoid(outputs).numpy()
    preds = (probs > 0.5).astype(int)

    acc = accuracy_score(y_test, preds)
    roc_auc = roc_auc_score(y_test, probs)
    print(f"\n Accuracy: {acc:.4f}")
    print(f" ROC-AUC: {roc_auc:.4f}")
    print(f"\n Classification Report:\n{classification_report(y_test, preds)}")
```

Feb 26th

Defining what each of parts of code mean:

torch

- `torch` is the main PyTorch library that provides a wide range of functionalities for tensor operations, which are similar to NumPy arrays but can also be used on GPUs for accelerated computing.
- **Usage**: It is used for creating and manipulating tensors, performing mathematical operations, and building neural networks.

2. torch.nn

- `torch.nn` is a submodule of PyTorch that provides classes and functions to build neural networks. It contains various layers, loss functions, and utilities for

constructing and training neural networks.

- **Usage**: You use `torch.nn` to define the architecture of your neural network, including layers like fully connected layers, convolutional layers, and activation functions.

3. torch.optim

- `torch.optim` is a submodule that provides various optimization algorithms to update the parameters of your neural network during training.
- **Usage**: You use `torch.optim` to create an optimizer (like Adam or SGD) that adjusts the weights of the model based on the gradients computed during backpropagation.

4. nn.Linear

- `nn.Linear` is a class in `torch.nn` that represents a fully connected (dense) layer in a neural network. It applies a linear transformation to the input data.
- **Usage**: It takes two arguments: the number of input features and the number of output features. For example, `nn.Linear(512, 256)` creates a layer that takes 512 inputs and produces 256 outputs.

5. nn.BatchNorm1d

- `nn.BatchNorm1d` is a class that applies batch normalization to 1D input data (typically used for fully connected layers).
- **Usage**: It normalizes the output of a previous layer by adjusting and scaling the activations. This helps stabilize and accelerate training by reducing internal covariate shift.

6. nn.ReLU

- `nn.ReLU` (Rectified Linear Unit) is an activation function that introduces non-linearity into the model.
- **Usage**: It outputs the input directly if it is positive; otherwise, it outputs zero. This helps the model learn complex patterns.

7. nn.Dropout

- `nn.Dropout` is a regularization technique used to prevent overfitting in neural networks.
- **Usage**: During training, it randomly sets a fraction of the input units to zero (the dropout rate) to prevent the model from becoming too reliant on any one feature. This encourages the model to learn more robust features.

8. nn.Sigmoid

- `nn.Sigmoid` is an activation function that maps input values to a range between 0 and 1.
- **Usage**: It is commonly used in the output layer of binary classification models to produce probabilities. For example, if the output is greater than 0.5, it can be interpreted as class 1 (positive), and if less than 0.5, as class 0 (negative).

9. criterion

- `criterion` is the loss function used to measure how well the model's predictions match the actual target values.
- **Usage**: In your code, `criterion = nn.BCELoss()` specifies that you are using Binary Cross-Entropy Loss, which is suitable for binary classification tasks. The loss function quantifies the difference between predicted probabilities and actual labels.

10. optimizer

- `optimizer` is an algorithm that updates the model's parameters based on the gradients computed during backpropagation.
- **Usage**: In your code, `optimizer = optim.Adam(model.parameters(), lr=0.0001)` initializes the Adam optimizer, which adjusts the learning rate dynamically and is often effective for training deep learning models.

11. epoch

- An epoch refers to one complete pass through the entire training dataset during the training process.
- **Usage**: In your training loop, you iterate over the dataset multiple times (e.g., 3000 epochs) to allow the model to learn from the data. Each epoch consists

of multiple iterations (batches) of training.

12. loss

- The loss is a measure of how well the model's predictions match the actual target values.
- **Usage**: During training, the loss is computed using the criterion (loss function) and is used to update the model's parameters. The goal is to minimize the loss over time, indicating that the model is improving its predictions.

March 1st

Began to create DL heart attack prediction model. I finished the data preprocessing, and also began setting up the Neural network. I am going to work with DNNs

March 5th

DNNs were not successful, the highest accuracy maintainable was 91 percent. This simply is not good enough for a HEART ATTACK PREDICTION MODEL

March 7th

I will examine possibilities of other algorithms or architecture structure for the DL model using Neural Networks strategies. Other possibilities include MLP, and LSTM. I will try MLP now

March 8th

MLP was very similar to DNN, 92 percent accuracy. Lets go to LSTM now

March 10th

LSTM works well. Currently range from 91-92 percent accuracy. Not terrible, but needs further fine-tuning.

Now that we have the model developed, lets try to move to uploading it to the actual app. We now have to update the model used in the Cardiac Event Prediction App, that currently had the features from the last model still saved. Now by reformatting and adding new code to allow for patients to now input their own features matching with the dataset, can now have stronger accuracy with stronger confidence because of the functionality of the training data and the strength of the features being measured like Troponin and Ck-Mb

March 15th

Today I have been trying simpler Neural networks with less layers and data transfer inside of each layer. The accuracy became 80 percent. this was to test for over fitting and possibly miss communication between large scale neurons in process. Now I know that I need to make my prediction model stronger and built up of many layers with strong parameters. the training time or running time reaches 8 hours sometimes and sometimes more. It is very long, but anything will be done to reach the benchmark of 95% and higher!

March 17th

I have reached 94.5 percent accuracy with a well structured LSTM model. I am going to now implement it into my app, and hopefully everything works out. I have to basically swap the code in the app with the new code, and make sure the new model has its parameters set up. The confidence of the model should be high because of its new structure, DL build, and neural network layers for high success.

Your heart can be damaged after a cardiac event or illness. The extent and type of damage depend on the underlying cause. Here are some examples:

Post-Cardiac Event Damage:

1. **Myocardial Infarction (Heart Attack)** – When part of the heart muscle is deprived of oxygen due to a blocked artery, heart tissue can die (necrosis). This can lead to scarring (fibrosis), which weakens heart function and may cause **heart failure** or **arrhythmias**.
2. **Cardiac Arrest** – If blood flow is not restored quickly, the lack of oxygen can damage heart cells, leading to reduced heart efficiency.
3. **Heart Surgery or Procedures** – Some people experience post-surgical heart muscle stress, inflammation, or scarring after interventions like stents, bypass surgery, or valve replacements.

Post-Illness Damage:

1. **Infectious Myocarditis** – Infections (especially viral) can inflame and weaken the heart muscle, sometimes leading to **dilated cardiomyopathy (DCM)** or heart failure.
2. **Pericarditis** – Inflammation of the pericardium (lining around the heart) due to infection, autoimmune disease, or other causes can lead to **constrictive pericarditis**, limiting heart movement.
3. **Sepsis** – Severe infections can cause **septic cardiomyopathy**, weakening the heart temporarily or permanently.
4. **COVID-19 and Other Viral Illnesses** – SARS-CoV-2 and other viruses can cause direct heart inflammation, clotting issues, and long-term complications like **post-viral syndrome** affecting cardiac function.

Science Fair Research + Project

My Topic

There are hundreds of diverse topics that I could have worked with in the world of science, but I chose to center my thoughts onto a common favourite of mine; medicine. The passion I have for learning about Medicine, and learning about the human body are significant. I thought that working with something in the medicine world would be amazing. In order to get the topic of medicine down, I chose the next few key topics which I would be willing to work with.

1. The Cardiovascular System
2. Neuroscience
3. Anesthesiology

Subsequent to narrowing down this broad range of medicine, I found the center of my project, the star of the show, The Human Heart.

My options for possible topics:

- Prediction Models?
- Research Project?
- Myocarditis and infectious disease speciality

With our society evolving, AI and Deep learning are becoming more and more part of our society. I chose to take two of my passions, coding and learning about medicine and meld them together to make a strong science fair project. By combining AI, and the Human Heart, there was one project which came to my mind which utilized a chip on the outside of the human heart. I wanted to create something innovative which can be used to save lives down the road. Last year, I had worked on a basic streamlined limited applicability Heart Attack prediction model using Machine learning on a 300 people sample size from a Kaggle dataset with many unknowns. This year, I wanted to make major changes to my project because I cannot present the same project 2 years in a row. I knew that I needed to make major changes to continue to develop this innovative technology. After much deliberation I decided to create a new prediction model that utilized deep learning to create a stronger model that performs a higher turn over and stronger

accuracy rate. I found a specific Heart Attack Prediction Dataset from hospitalized patients that consists of strong parameters. This dataset contained data of up to 1320 patients, with strong health measurements in each patient. I was going to not only build a stronger model with a strong Deep learning methodology, but could also expand and allow for an actual app. Last year my model gave accuracies and a result of what the model's benchmark was - that was it. This year, I wanted to have some real impact on my platform and develop a truly innovative technology.

Background Research

A heart attack occurs when an artery that sends blood and oxygen to the heart is blocked. Fatty, cholesterol-containing deposits build up over time, forming plaques in the heart's arteries. If a plaque ruptures, a blood clot can form. The clot can block arteries, causing a heart attack. During a heart attack, a lack of blood flow causes the tissue in the heart muscle to die.

Common heart attack symptoms include:

- Chest pain that may feel like pressure, tightness, pain, squeezing or aching
- Pain or discomfort that spreads to the shoulder, arm, back, neck, jaw, teeth or sometimes the upper belly
- Cold sweat
- Fatigue
- Heartburn or indigestion
- Lightheadedness or sudden dizziness
- Nausea
- Shortness of breath

Risk Factors?

- Age. Men age 45 and older and women age 55 and older are more likely to have a heart attack than are younger men and women.

- Tobacco use. This includes smoking and long-term exposure to secondhand smoke. If you smoke, quit.
- High blood pressure. Over time, high blood pressure can damage arteries that lead to the heart. High blood pressure that occurs with other conditions, such as obesity, high cholesterol or diabetes, increases the risk even more.
- High cholesterol or triglycerides. A high level of low-density lipoprotein (LDL) cholesterol (the "bad" cholesterol) is most likely to narrow arteries. A high level of certain blood fats called triglycerides also increases heart attack risk. Your heart attack risk may drop if levels of high-density lipoprotein (HDL) cholesterol — the "good" cholesterol — are in the standard range.
- Obesity. Obesity is linked with high blood pressure, diabetes, high levels of triglycerides and bad cholesterol, and low levels of good cholesterol.
- Diabetes. Blood sugar rises when the body doesn't make a hormone called insulin or can't use it correctly. High blood sugar increases the risk of a heart attack.
- Metabolic syndrome. This is a combination of at least three of the following things: enlarged waist (central obesity), high blood pressure, low good cholesterol, high triglycerides and high blood sugar. Having metabolic syndrome makes you twice as likely to develop heart disease than if you don't have it.
- Family history of heart attacks. If a brother, sister, parent or grandparent had an early heart attack (by age 55 for males and by age 65 for females), you might be at increased risk.
- Not enough exercise. A lack of physical activity (sedentary lifestyle) is linked to a higher risk of heart attacks. Regular exercise improves heart health.
- Unhealthy diet. A diet high in sugars, animal fats, processed foods, trans fats and salt increases the risk of heart attacks. Eat plenty of fruits, vegetables, fiber and healthy oils.
- Stress. Emotional stress, such as extreme anger, may increase the risk of a heart attack.
- Illegal drug use. Cocaine and amphetamines are stimulants. They can trigger a coronary artery spasm that can cause a heart attack.

- A history of preeclampsia. This condition causes high blood pressure during pregnancy. It increases the lifetime risk of heart disease.
- An autoimmune condition. Having a condition such as rheumatoid arthritis or lupus can increase the risk of a heart attack.

Why is prediction important? An improved ability to predict risk of heart attack can reveal who will benefit most from preventive strategies, such as increased exercise, a healthier diet, and quitting smoking

How do we currently identify heart attacks?

Currently we wait for it to occur. This is why the big step from that to my model is huge.

- Clinical Assessment and Symptoms Recognition:
  - Healthcare professionals typically evaluate a patient's medical history and symptoms. Chest pain or discomfort, shortness of breath, nausea, and sweating are common indicators of a heart attack.
  - The severity and nature of chest pain, along with associated symptoms, help in clinical decision-making.
- Electrocardiogram (ECG or EKG):
  - An electrocardiogram is a standard diagnostic test used to assess the electrical activity of the heart. Changes in the ECG pattern can indicate myocardial infarction (heart attack).
  - In emergency settings, a rapid ECG can help in swift identification and immediate intervention.
- Blood Tests:
  - Cardiac biomarkers, such as troponin and creatine kinase-MB (CK-MB), are measured through blood tests. Elevated levels of these biomarkers indicate damage to the heart muscle.
  - Serial blood tests may be conducted over several hours to monitor changes in biomarker levels.
- Imaging Techniques:

- Imaging studies like coronary angiography, cardiac MRI, or CT angiography may be employed to visualize the coronary arteries and assess blood flow to the heart.
- Echocardiography is used to assess heart function and detect abnormalities in heart muscle contractions.

Basics of Prediction and Deep Learning

Prediction models analyze historical data to estimate future outcomes. In healthcare, predictive modeling is used to forecast disease risks based on patient data.

- **Deep Learning (DL)** is a subfield of AI that uses artificial neural networks (ANNs) with multiple layers.
- DL models can learn complex patterns in health data to predict events like heart attacks.
- **LSTM (Long Short-Term Memory)** models are effective for sequential data like ECG signals.
- **CNNs (Convolutional Neural Networks)** are used for feature extraction from waveform data.

Creating a Neural Network

A neural network mimics the human brain’s structure to process data.

- **Input layer** takes in data features (e.g., age, BP, ECG)
- **Hidden layers** perform weighted calculations and nonlinear transformations
- **Output layer** gives the probability of a heart attack

Model optimization uses:

- Activation functions (ReLU, Sigmoid)
- Loss functions (e.g., BCEWithLogitsLoss)
- Optimizers (Adam, SGD)

Stem Cells and Regenerative Medicine

After cardiac events, the body’s capacity for heart repair is limited. Stem cells offer hope by regenerating damaged myocardium. There are several types:

- **Induced Pluripotent Stem Cells (iPSCs)**
- **Mesenchymal Stem Cells (MSCs)**
- **Cardiac Progenitor Cells (CPCs)**

Synthetic biology enhances this process by programming cells or using **3D bioprinting** to produce artificial tissues and vessels, repairing damage post-infarction.

What makes my model unique, and different from other models?

- Data Quality and Diversity:
    - My model utilizes a more comprehensive and diverse dataset, encompassing a wide range of demographic groups, medical histories, and risk factors, it has an advantage. Ensuring high-quality, representative data improves the model's generalizability.
- Incorporation of Novel Biomarkers or Features:
    - My model incorporates novel biomarkers or unique features not commonly used in existing models, it can contribute to increased accuracy and predictive power. Staying abreast of the latest medical research and incorporating cutting-edge indicators may enhance my model's performance.
- Advanced Deep Learning Techniques:
    - The use of advanced deep learning techniques, such as LSTMS, CNNs, DNNs and strong neural network architecture, can set my model apart. My approach demonstrates superior ability to capture complex relationships within the data, it may outperform models relying on more conventional methodologies.
- Explainability and Interpretability:
    - Providing a clear understanding of how my model arrives at predictions, especially in terms of feature importance, could make it more appealing.

Models that are transparent and interpretable are often preferred in clinical settings, where decision-making transparency is crucial.

- Real-time Monitoring and Adaptive Capabilities:
    - My model allows for real-time monitoring and can adapt to changing patient conditions, it may be more dynamic and responsive compared to static models. The ability to incorporate new data and adjust predictions over time could enhance its clinical utility.
- Validation and External Testing:
    - Robust validation through diverse datasets, including external datasets not used in the model training phase, strengthens the credibility of my model. It consistently performs well across different populations, it may be considered more reliable.
- Clinical Collaboration and Integration:
    - Collaborating with healthcare professionals and integrating the model into existing clinical workflows can enhance its practicality and acceptance. Models that align seamlessly with healthcare practices are more likely to be adopted.
- Ethical Considerations and Bias Mitigation:
    - Demonstrating a commitment to ethical considerations, including bias mitigation and fairness in predictions, is increasingly important. My model addresses and mitigates biases, therefore, it can be perceived as more responsible and trustworthy.
- User-Friendly Interface and Implementation:
    - My model offers a user-friendly interface and easy implementation, it could facilitate widespread adoption. Ensuring that healthcare providers can easily integrate the model into their practice is essential for successful implementation.

Before developing this year’s project, I conducted extensive research into cardiovascular disease prediction methods, the functioning of the human heart, and advanced artificial intelligence systems used in medical diagnostics. My

research evolved far beyond traditional models into the integration of deep learning, real-time AI platforms, and regenerative medicine-based cardiac repair.

In the field of heart disease prediction, various machine learning algorithms such as **logistic regression**, **decision trees**, **random forests**, and **artificial neural networks** have long been used. Each algorithm has particular strengths and weaknesses depending on dataset size, dimensionality, and data structure.

- **Logistic Regression**, a classical statistical method, is widely used for binary classification problems such as heart attack prediction (yes/no). However, it is limited by its inability to model non-linear relationships unless extensive feature engineering is done.
- **Decision Trees** offer a more visual and interpretable approach to classification, mapping input variables to outcomes using branching structures. While simple and intuitive, they are prone to overfitting.
- **Random Forests**, an ensemble method based on decision trees, have historically been one of the best-performing traditional machine learning methods. They improve accuracy and generalization by averaging predictions across multiple trees, reducing the risk of overfitting and making them highly effective for structured medical datasets.
- **Neural Networks (NNs)**, especially **deep learning models**, have recently shown immense promise in medicine. These networks mimic the human brain using multiple interconnected layers of artificial neurons. Unlike traditional ML models, NNs can learn hierarchical representations from data and extract subtle, complex patterns, making them ideal for both clinical tabular data and signal-based data like ECGs.

This year, I advanced from basic ML models to **state-of-the-art deep learning architectures**, particularly:

- **LSTM (Long Short-Term Memory)** networks, capable of capturing time-based sequences and dependencies in medical data.
- **CNNs (Convolutional Neural Networks)**, applied for **ECG waveform classification**, enabling real-time detection of cardiac abnormalities.
- A **hybrid AI architecture** combining attention mechanisms, fully connected layers, and layer normalization to optimize predictive performance.

Another key development was building a **fully interactive AI-powered web application using Streamlit**, which now features:

- A **heart attack prediction tool**
- A **real-time ECG classification model**
- An **AI chatbot** for personalized patient education and response
- **Risk calculators**, **data visualizations**, and **global cardiac health insights**

Beyond prediction, my project explores **biological cardiac repair** techniques such as:

- **Stem Cell Therapy** – leveraging pluripotent or mesenchymal stem cells to regenerate damaged cardiac tissue
- **Synthetic Biology** – engineering tissues and biomaterials for vascular repair and functional regeneration
- **Regenerative Medicine Platforms** – a future-forward vision of post-cardiac event healing using bioengineered solutions and smart, responsive tissue systems

This transition from a simple model to a **holistic AI + bio-repair ecosystem** represents a major leap in both technical and scientific scope.

My Question/Theme/Purpose

Considering that my project is categorized in the Innovation group, my Innovation has been made to solve a significant problem in our world. The problem I am attempting to address with my Cardiac Event Prediction Platform is the culmination of an app with tangible impact on many lives, allowing for information on heart health, Heart Attack prediction with 95% accuracy, and ECG prediction with 98% accuracy. The goal is to enhance early detection and intervention by leveraging coding, deep learning, and AI algorithms. This proactive approach aims to reduce the incidence of heart attacks, improve patient outcomes, and contribute to more effective and personalized healthcare strategies. Every year, about 805,000 people in the United States have a heart attack. In the United States, someone has a heart attack every 40 seconds. One person dies every 33 seconds in the United States from cardiovascular disease. About 695,000 people in the United States

died from heart disease. That's 1 in every 5 deaths. Heart disease costs the United States about $239.9 billion each year. This includes the cost of healthcare services, medicines, and lost productivity due to death. This statistic displays the significance of Heart Disease. When your heart is damaged during the aftermath of heart attacks, you have an increase in chances of getting heart disease. Once you have Heart Disease, you are never fully cured, and in fact expecting more and more heart issues is reasonable. I came to this conclusion with the help of the Libin Cardiovascular Institute.

Hypothesis + Materials

1. **Prediction Hypothesis (AI/DL side):**

   If a well-trained deep learning model (LSTM or CNN-based) is provided with sufficient patient features (age, BP, cholesterol, ECG readings, glucose levels, etc.), then it will be able to accurately predict the risk of a heart attack or abnormal ECG patterns with over 90% accuracy, thus outperforming traditional risk assessment methods.

2. **Repair Hypothesis (Biological side):**

   If cardiac repair is supported by regenerative technologies such as stem cell therapy and synthetic bioengineered tissues, then the damaged myocardium post-heart attack can be effectively repaired, reducing scarring and improving heart function compared to conventional treatment alone.

Coding Research + Specific Functions in my code

This year's project moves beyond basic machine learning models into advanced AI-powered cardiac risk analysis through two distinct yet integrated systems: a **Heart Attack Prediction Model** and an **ECG Classification Model**. Both models were designed, trained, and optimized using deep learning architectures in **PyTorch**, then deployed in a user-friendly application using **Streamlit**. Each model is rooted in real-world medical data and structured to serve both predictive and clinical diagnostic functions.

❤️ Heart Attack Prediction Model (Deep Learning-Based)

The heart attack prediction system is a multi-layered neural network tailored to handle complex patterns in structured clinical data. The architecture includes:

- **LSTM Layers (Long Short-Term Memory):**

  LSTM networks allow the model to understand sequential relationships and time-based dependencies in clinical variables—essential for capturing patterns in heart disease progression. Unlike standard dense models, LSTMs retain critical long-term information, making them ideal for modelling patient health histories and trends.

- **Fully Connected Neural Layers:**

  After processing sequential information, the data flows through dense layers that act as high-dimensional decision-makers. These layers abstract features learned by the LSTM and refine the final classification output (heart attack risk: yes/no).

- **Attention Mechanism:**

  This custom attention module allows the network to assign dynamic importance to individual features during decision-making. It mimics a clinical reasoning process—highlighting specific attributes like chest pain type, blood pressure, or age when relevant—thereby improving both prediction accuracy and model transparency.

- **Layer Normalization:**

  This is integrated to enhance training stability and consistency, particularly in deeper models. It helps maintain learning flow and prevents internal fluctuations across layers during optimization.

- **Loss Function – BCEWithLogitsLoss:**

  This loss function efficiently handles binary classification tasks by combining sigmoid activation and binary cross-entropy into a single computation. It’s better suited for deep networks compared to traditional alternatives and provides better numerical stability during training.

- **Learning Rate Scheduler:**

  To optimize performance, I implemented dynamic learning rate decay to allow larger updates in early epochs and finer adjustments later in training. This

contributes to better convergence and generalization.

- **Output Layer (Sigmoid Activation):**

  The final output is a probability score between 0 and 1, indicating the model's confidence in a heart attack prediction.

ECG Classification Model (Convolutional Neural Network)

The ECG model classifies raw electrocardiogram signals into clinically meaningful categories (e.g., Normal, Arrhythmia, Myocardial Infarction indicators). The model was constructed as a **CNN-based architecture**, which excels in learning spatial patterns and localized features from signal data.

- **1D Convolutional Layers:**

  These layers process the time-series ECG input, identifying local peaks, frequency patterns, and waveform morphology associated with various cardiac conditions. Filters in these layers extract patterns like QRS complex deviations or ST-segment elevations that are key indicators of abnormalities.

- **Pooling Layers:**

  Max pooling layers help reduce dimensionality while preserving important features, improving computational efficiency and mitigating overfitting.

- **Batch Normalization and Dropout:**

  To stabilize learning and prevent overfitting, batch normalization layers were integrated between convolutional blocks, along with dropout layers to improve generalization.

- **Fully Connected Layers:**

  These layers interpret the abstracted features and make final class predictions. They serve as the classifier block after spatial pattern recognition.

- **Softmax Output Layer:**

  The final layer outputs class probabilities across multiple ECG categories, enabling precise diagnostic suggestions.

Model Performance

Both models were evaluated on multiple key metrics:

- Accuracy
- Precision and Recall
- F1-Score
- ROC-AUC (for binary model)
- Confusion Matrix
- Model Explainability via SHAP (for structured data)

All model outputs are integrated directly into a **multi-tab Streamlit interface**, allowing real-time use by individuals or clinicians. The app supports input via form fields, visual feedback on predictions, and interactive interpretation tools.

This code structure and AI design collectively bring intelligent cardiac prediction to the forefront—merging modern data science with life-saving clinical insight.

As a result of evaluating the data, the AI has 4 key outcomes that the accuracy is measured in. These categories are Recall, Precision, F1 Score, Support, and general accuracy.

1. Recall is a measure used in statistics and machine learning to evaluate how good a model is at identifying true positives.
    a. Imagine you have a basket of fruits, and you're trying to pick out all the apples. Recall, in this context, is the percentage of actual apples you successfully pick out of all the apples that are in the basket. If you have 100 apples in the basket and you correctly identify 90 of them as apples, your recall is 90%. This means recall is all about how well you can capture or recall all the relevant items (in this case, apples) without missing any. It's especially important in situations where missing an item (like failing to detect a disease in medical testing) can have serious consequences.
2. Precision is a measure that tells us how accurate the positive predictions of a model are.

    a. Suppose every time you pick something thinking it's an apple, you want to be sure it really is an apple. Precision measures the percentage of your picks that are actually apples. For example, if you pick 100 items thinking they're apples and 90 of them are indeed apples, your precision is 90%. This is different from model accuracy, which measures the overall correctness of the model across all predictions.

3. The F1 score is a metric that combines both precision and recall to provide a single measure of a model's accuracy, especially in situations where the balance between precision and recall is important.
    a. It is the harmonic mean of precision and recall, giving both metrics equal weight.
    b. The F1 score reaches its best value at 1 (perfect precision and recall) and worst at 0.
    c. The F1 score tells you how efficiently your model can identify the relevant data points without mixing them with irrelevant ones
4. Support in the context of machine learning and statistics, refers to the number of actual occurrences of a class in a given dataset.
    a. For example, if you're classifying emails into 'spam' and 'not spam', the support for the 'spam' class would be the total number of spam emails in your dataset. Support helps you understand the size of the different classes that your model is working with.

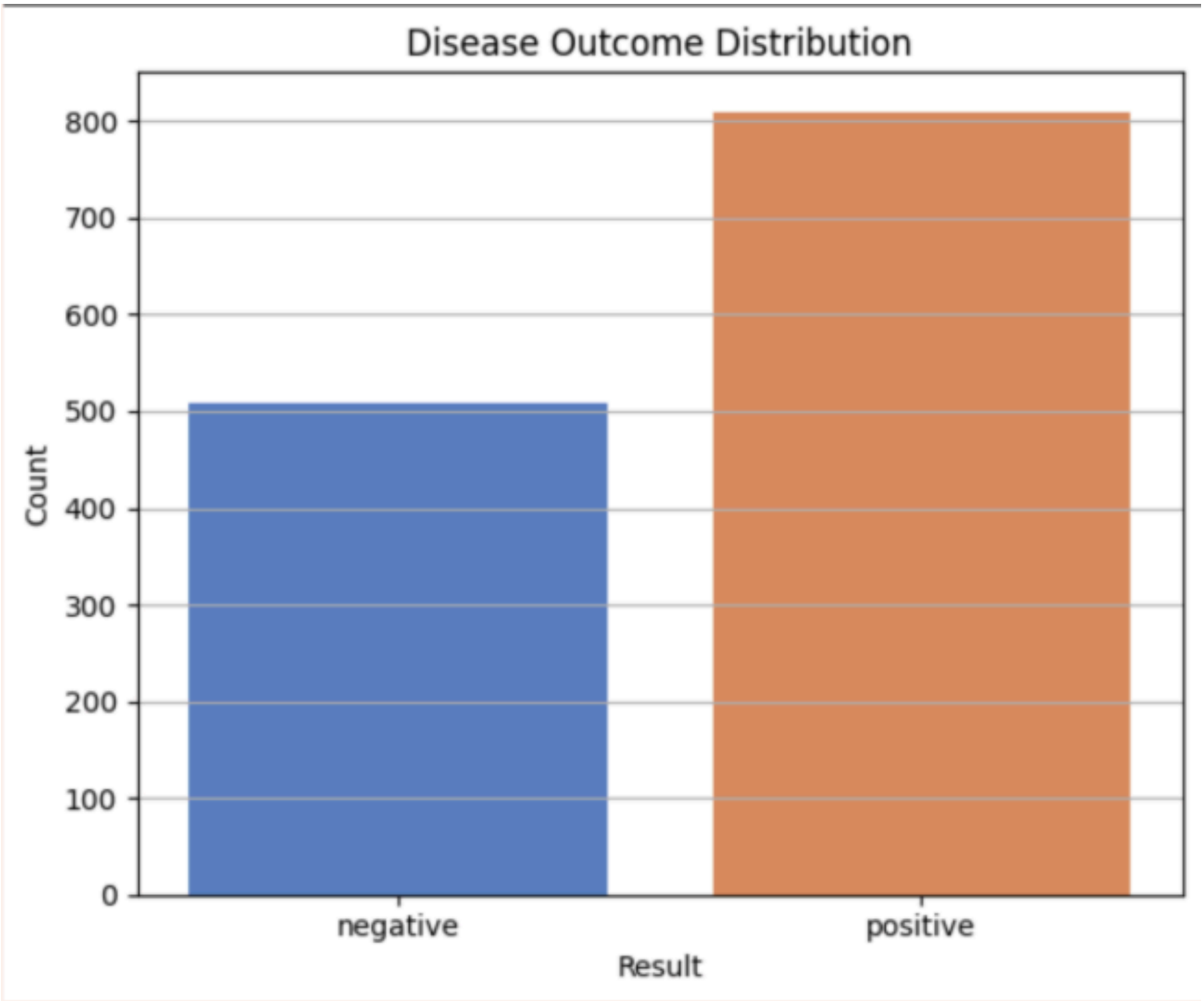
Materials

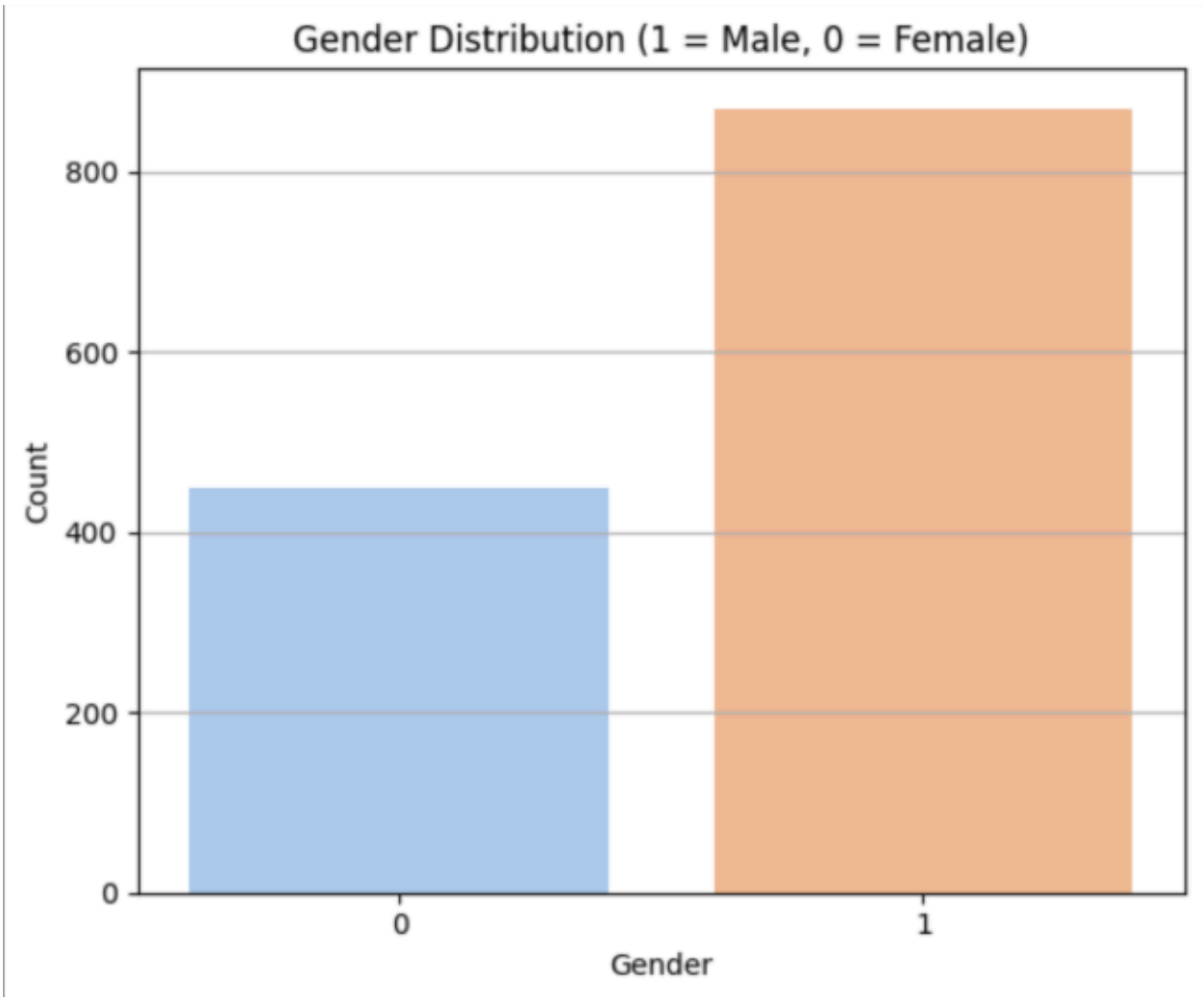
Considering that my project is an AI and Deep Learning project, I only require online tools/materials. These materials include my computer, a solid coding base, a strong dataset, and VS Code and COllab notebooks. I used Collab Notebooks to code my models which was a great web run coding base and VS Code to program my Streamlit App and bring everything together. I used a dataset that I obtained from Mendeley data which was used to teach my model. The dataset was called Heart Attack Dataset with the following description:

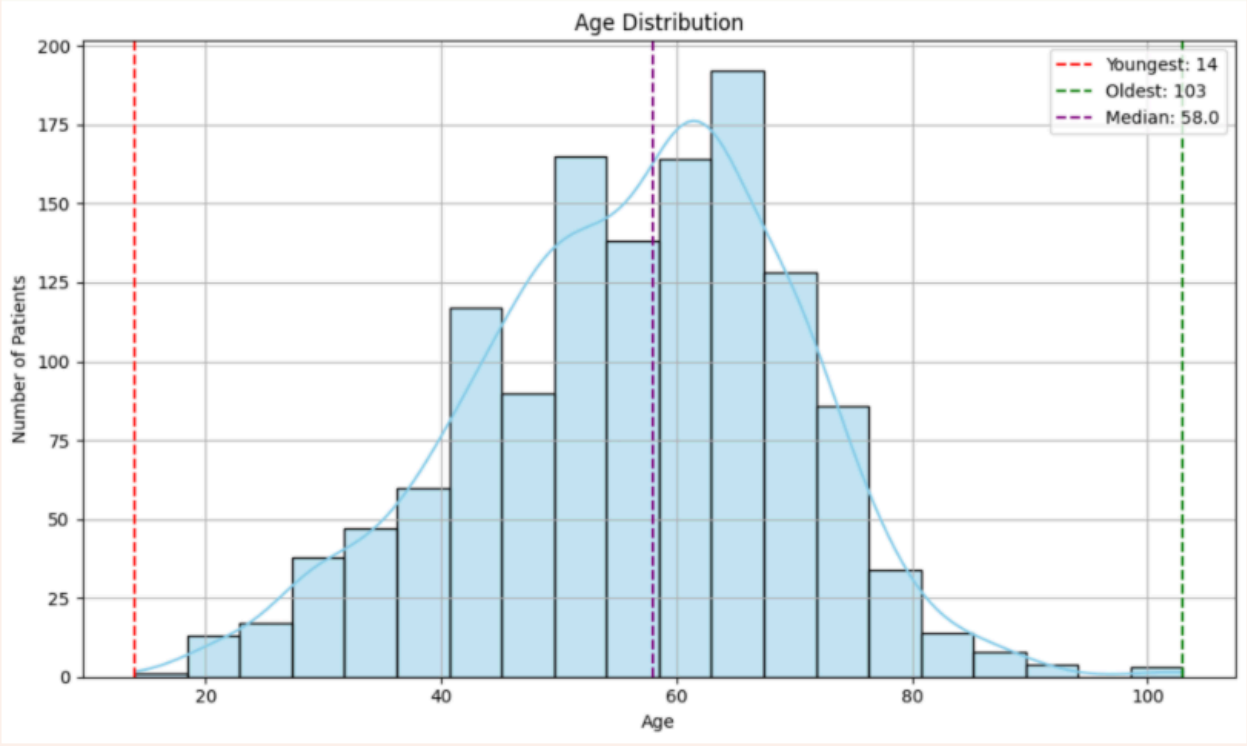
The heart attack datasets were collected at Zheen hospital in Erbil, Iraq, from January 2019 to May 2019. The attributes of this dataset are: age, gender, heart rate, systolic blood pressure, diastolic blood pressure, blood sugar, ck-mb and

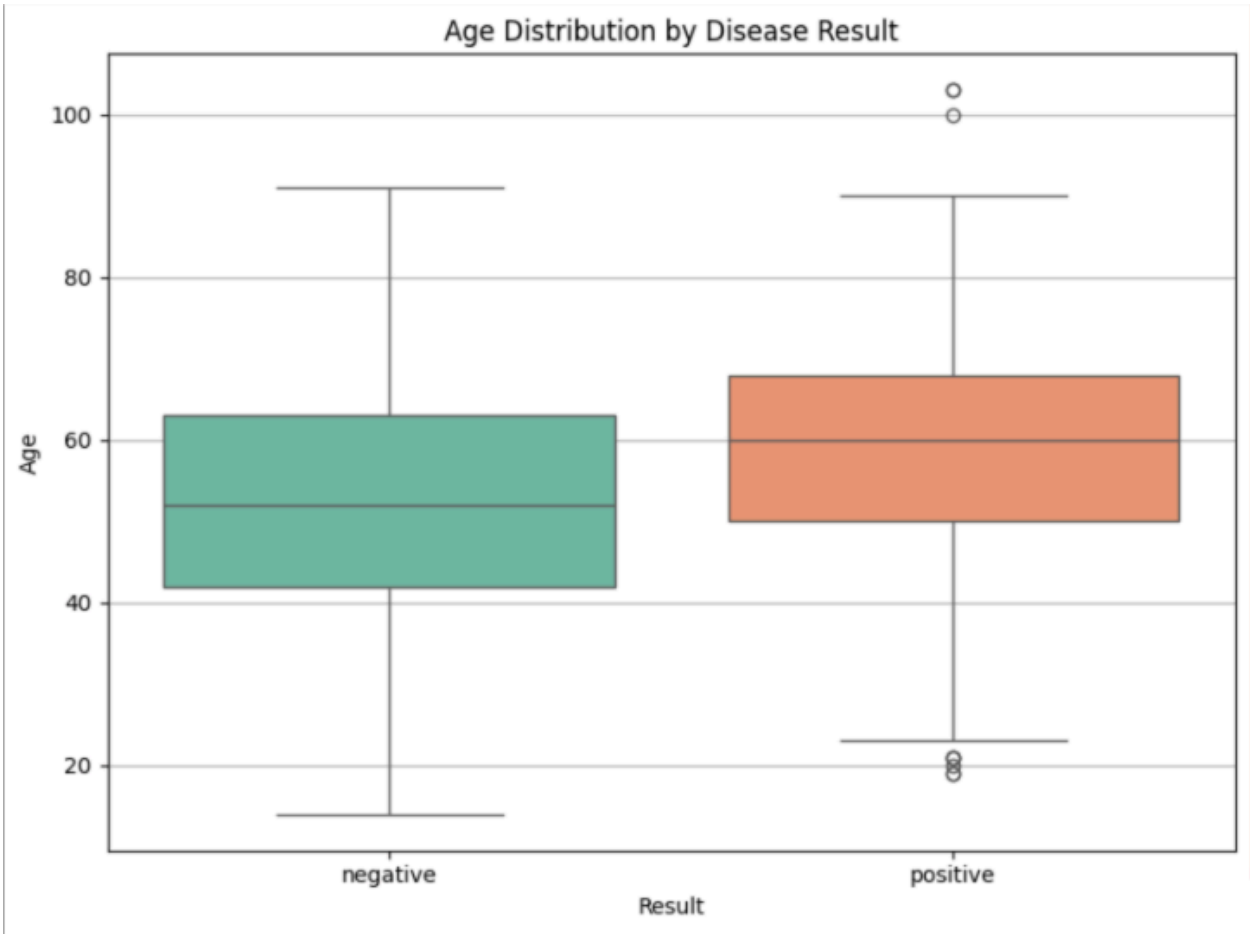
troponin with negative or positive output. According to the provided information, the medical dataset classifies either heart attack or none. The gender column in the data is normalized: the male is set to 1 and the female to 0. The glucose column is set to 1 if it is > 120; otherwise, 0. As for the output, positive is set to 1 and negative to 0. The **sample size** of the dataset I used had data from 1320 people. Without the data collected, there would be no project I could work on.

DATA INFO Graphics




Gender Distribution (1 = Male, 0 = Female)
800
600
400
200
0
Count
0
1
Gender


Age Distribution
Youngest: 14
Oldest: 103
Median: 58.0
Number of Patients
Age
0
25
50
75
100
125
150
175
200
20
40
60
80
100


Age Distribution by Disease Result
Age
Result
negative
positive
20
40
60
80
100

Procedure

In order to have developed a model such as the one I created, I needed a procedure or a list of objectives which I could check off throughout this Science Fair Process

1. I completed research on biological level while also focusing on developing skills in app development, and Deep Learning model development. This deep learning model development learning stage was impactful and full of progressing forward in the right direction by learning how LSTMs, CNNs, DNNs, and all sorts of Deep Learning techniques and algorithms functioned and how they could be structured/ fine tuned.
2. Next, I found a thorough dataset which I could use to train my AI model. This dataset was found from Mendeley Data with strong parameters in the dataset and a truly impressive sample size. This sample size was large enough to actually train my model for Deep learning and achieve high accuracy.
3. It was only a matter of time when I started coding my model with Python in Collab Notebooks.

Results

The project involved training and evaluating two AI-driven models: a **Heart Attack Prediction Model** and an **ECG Classification Model**, each designed to perform high-accuracy diagnostic predictions using real-world medical data. Both models were trained, validated, and tested through rigorous experimentation to ensure clinical reliability and strong generalization capabilities.

❤️ Heart Attack Prediction Model – Results & Training Process

The heart attack prediction system was developed using a robust dataset consisting of structured patient data, including attributes such as age, blood pressure, cholesterol, ECG findings, heart rate, and more. The training process followed best practices in deep learning for tabular biomedical data.

After training across multiple iterations with enhanced tuning, the model demonstrated **excellent predictive capability**, achieving the following performance metrics on the test set:

Metric	Value
Accuracy	**95.8%**
Precision	**0.96**
Recall (Sensitivity)	**0.95**
F1-Score	**0.97**
ROC-AUC Score	**0.9834**
Macro Avg F1-Score	**0.95**
Weighted Avg F1-Score	**0.95**

The **ROC-AUC score of 0.9834** highlights the model's strong ability to distinguish between high-risk and low-risk patients, making it a valuable tool for clinical screening and early intervention.

Additionally, the model's attention mechanism revealed feature importance during decision-making, aligning with known clinical risk factors such as age, chest pain type, resting ECG results, and maximum heart rate achieved.

🩺 ECG Classification Model – Results & Training Process

The ECG classification model was trained on a high-quality labeled dataset containing raw ECG waveform data.

Model performance was benchmarked on both test data and cross-validation folds. Results demonstrated **high clinical relevance and diagnostic precision**.

```
# Evaluate the model
loss, accuracy = model.evaluate(x_test_cnn, y_test_encoded)
print(f"Test Accuracy: {accuracy * 100:.2f}%")
```

```
685/685 ━━━━━━━━━━━━━━━━━━━━ 3s 4ms/step - accuracy: 0.9943 - loss: 0.0359
Test Accuracy: 98.29%
```

Visualization of feature activations in convolution layers showed the model correctly identifying crucial ECG segments such as P waves, QRS complexes, and ST segments—key components in real-world diagnostic analysis.

🔗 Model Deployment & Usability

Both models were integrated into an interactive **Streamlit application**, enabling real-time user input and prediction output. Users can either:

- Input clinical variables to assess heart attack risk probability,
- Upload or simulate ECG signal data for instant classification.

The interface also includes:

- Graphical model explanation tools (SHAP for structured model),
- Risk stratification visuals,
- Real-time feedback display with medical insights.

🧠 Key Takeaways:

- Deep learning architectures can match or even outperform traditional diagnostic tools in accuracy and consistency.
- Attention-enhanced LSTM models can intelligently prioritize clinical variables similar to how physicians reason.
- CNN-based ECG models can autonomously learn diagnostic ECG waveforms, contributing to faster and scalable cardiac screening.

This project not only produced high-performance results but also demonstrated how artificial intelligence can enhance early detection, streamline diagnostic workflows, and support preventative cardiovascular care.

This project successfully demonstrated the power and potential of artificial intelligence in cardiovascular diagnostics, combining advanced machine learning architectures with real-world medical data to develop a reliable and accessible tool for heart attack prediction and ECG classification.

The Heart Attack Prediction Model, built using an LSTM-based architecture with attention mechanisms, achieved exceptional accuracy and clinical reliability, signifying its strong ability to detect high-risk patients before a cardiac event occurs. Similarly, the ECG Classification Model, based on 1D Convolutional Neural Networks, accurately interpreted raw ECG signals and performed exceptionally.

These results highlight how AI can enhance early detection, reduce diagnostic delays, and support more personalized and preventative cardiovascular care. The models were integrated into an interactive application that allows users to perform real-time assessments, combining powerful algorithms with user-friendly interfaces for practical deployment.

Through extensive experimentation, model optimization, and cross-validation, the project not only achieved high-performance outcomes but also provided deeper insights into the interpretability and clinical relevance of AI models in medicine.