Accurate Cancer Prediction Using AI

SDMAY24-47

Jack Sebahar, Nicholas Otto, Mason Wichman, Lal Siama, Helen Lau, Isaiah Mundy

Client & Faculty Advisor: Ashraf Gaffar

Problem

Cancer is one of the biggest medical mysteries we have yet to solve. We are limited in our ability to predict its occurrence and recurrence.

Solution

Our project is to build and train an AI model to provide accurate predictions of cancer given a set of data.

Intended Users & Uses

Our primary users will be medical professionals with the model being used to aid in medical diagnosis.

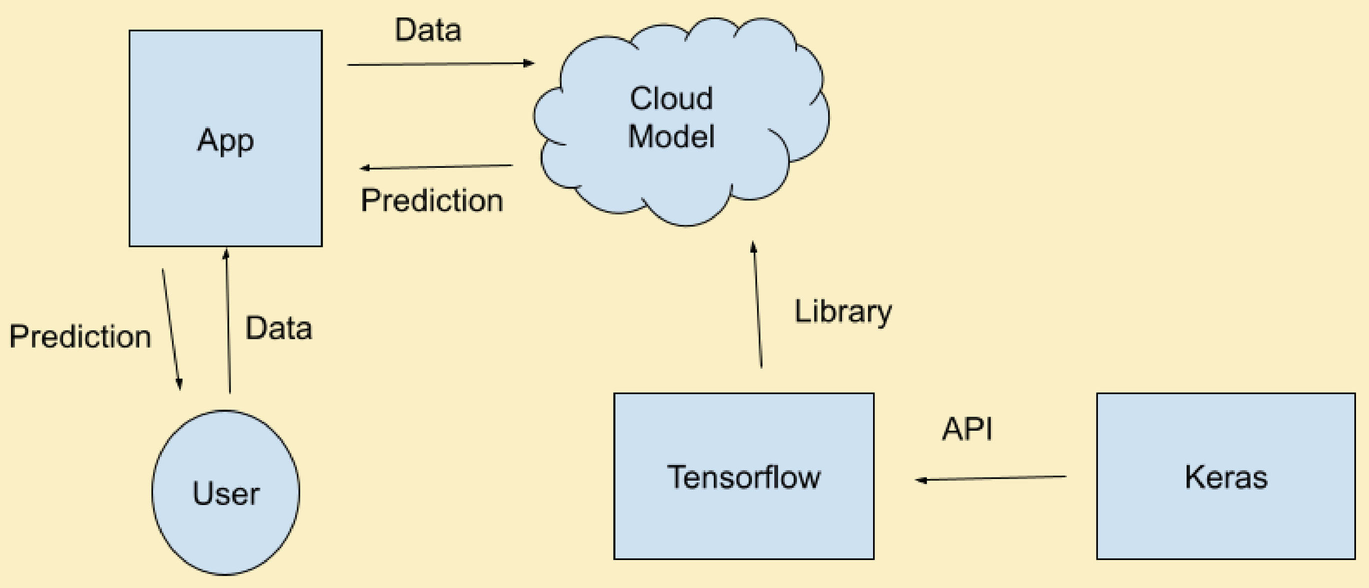
Design Requirements

- The model must be capable of making predictions based on user input data
- The model must be deployed on a cloud platform
- The application must allow the user to upload their own data
- The application must retrieve prediction from cloud-hosted model
- The application must display the prediction to the user

Design

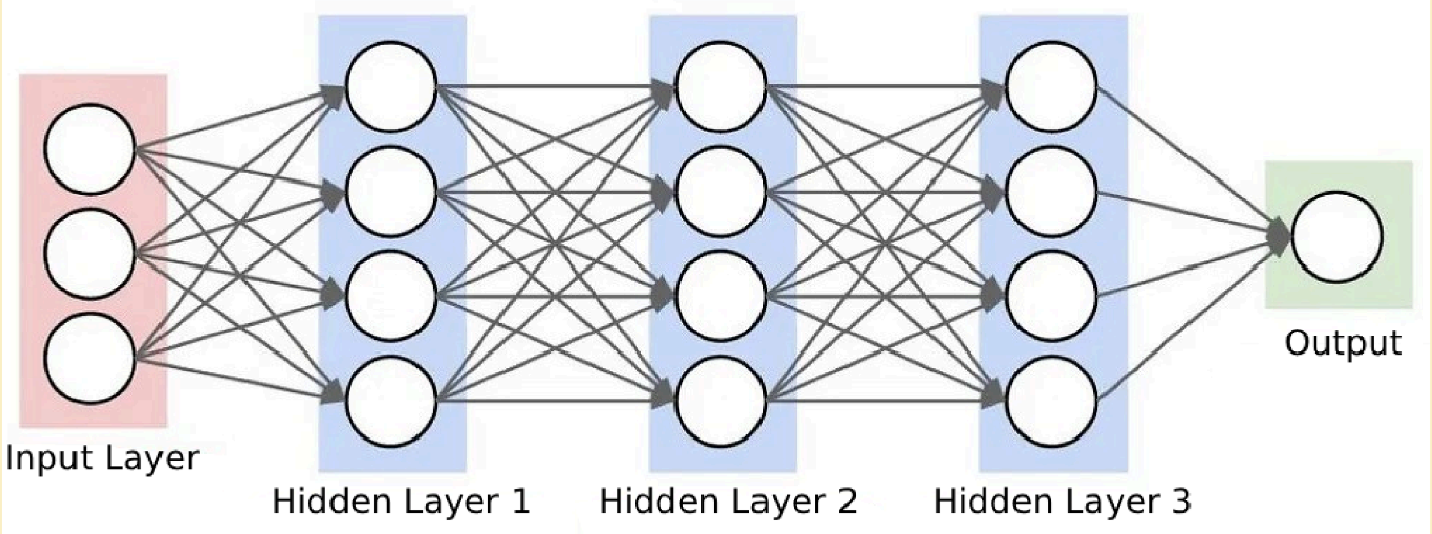
Overall Design

- User inputs data
- Data is sent to a cloud-hosted model
- Survival prediction is returned to the user in the UI



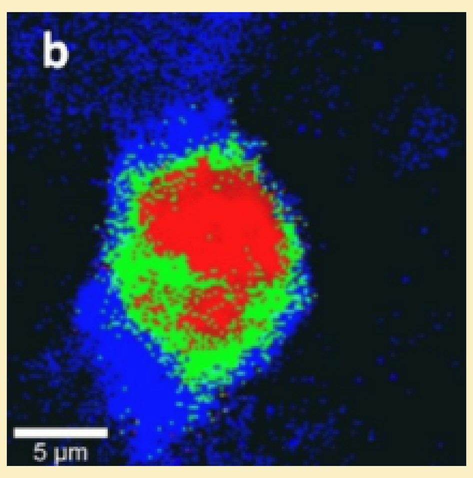
Model Design

- Base model is a Sequential Keras model with 4 dense layers
  - Layers represent different stages of computation in our model that data flows through
- We chose this particular model because they are:
  - Very straightforward to design and prototype
  - Very efficient in training



Data

- Data gathered from cancer recurrent patients
- Samples represent an encoded cell image of a cancerous or benign structure
- Samples each have a corresponding number that represents survival duration (months) after cancer recurrence



Model Deployment

- Utilized Vertex AI on Google Cloud Platform and Sagemaker on AWS
- Imported the model under the Tensorflow framework
- Deployed to an endpoint with 1 compute node
- Once deployed, the model is able to receive requests from the application

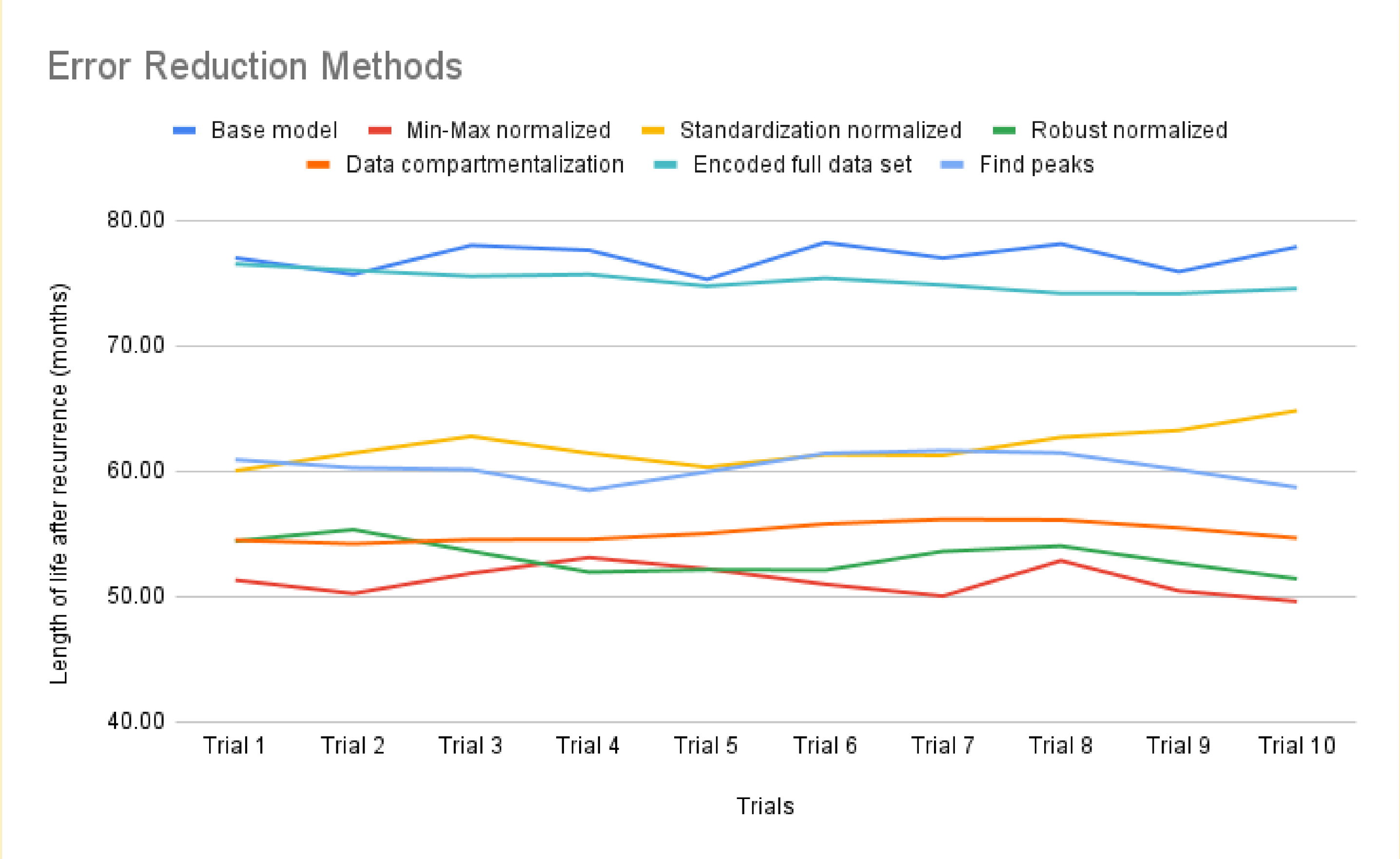
Application

- Designed as a web application using the Python Flask framework
- UI written in HTML
- The application is mostly in charge of preprocessing the data into a form that the model expects
- Once the data is ready, the application sends a request with the data to the model hosted in the cloud and retrieves a prediction

Error Reduction

One main focus of our project was to reduce the error of the model's predictions. The four main methods we tested to achieve this were:

- Normalization
- Unsupervised Learning
- Data compartmentalization
- Find peaks



Technical Challenges

Handling Malformed or Corrupt Data

- Wrote a Python script to help preprocess our data by filtering out corrupted or incomplete samples

Data Representation

- Utilized Pandas Dataframes to store patient data
- experimented with different ways of representing the data in the model training

Reducing Error

- Tried a number of approaches to reduce error such as data normalization, using a condensed representation of our data, adding more layers to our model

IOWA STATE UNIVERSITY College of Engineering