IOWA STATE UNIVERSITY

Senior Design Team sdmay24-47

Accurate Cancer Prediction Using Artificial Intelligence

Advisor: Dr. Ashraf Gaffar

Team: Jack Sebahar, Nick Otto, Mason Wichman,
Helen Lau, Lal Siama & Isaiah Mundy

Introduction

- The current challenge in cancer diagnosis and prediction lies in the limited ability to accurately foresee its occurrence and recurrence, and give related diagnoses.
- Traditional diagnostic methods may not be as precise as desired, and human doctors, despite their expertise, may face challenges in providing consistently accurate predictions.
- Recent research suggests that artificial intelligence (AI) has the potential to significantly improve predictive accuracy, potentially surpassing the capabilities of human doctors.

Can we leverage machine learning and neural networking tools such as Tensorflow and Keras to enhance cancer prediction?

Goals

1. Use AI as a means to discover potential correlation in data
    a. does data suggest an output?
    b. investigative approach
        i. trial and error
2. Analyze different machine learning platforms
    a. Google Cloud Platform vs Amazon Web Services
    b. Compare metrics
        i. Cost
        ii. Performance
        iii. Usability

Requirements

Functional

- The model must be trained to accept data in the form of the sample data
- The application must retrieve prediction from cloud hosted model

Resource

- Cloud platform (Google Cloud and Amazon Web Services)
- Tensorflow machine learning library
- Web application framework (Python Flask)

UI

- The application shall present information in an organized manner
- Pages within the application shall be easily accessible by use of navigational menus and buttons

Requirements

Performance

- The application should run at a user-friendly speed, the user should not have to wait more than a minute for page navigation or model predictions

Legal

- Comply with data privacy and security regulations when handling medical data

Maintainability

- The application shall support the addition of new or changed information

Testing

- The application shall be tested thoroughly to ensure it has a high level of reliability, usability, and accuracy

Engineering Standards

- HIPAA - ensures the protection of patients' medical data.
- IEEE 1058 - provides guidelines for the preparation of software project management plans.
- IEEE 3051 - trustworthiness of Artificial Intelligence and Autonomous Systems
- IEEE 3123 - clear definitions for terminology utilized in artificial intelligence and machine learning.
- IEEE 12207 - defines the processes involved in the software development life cycle, ensuring the quality and reliability of software systems.
- ISO 13485:2016 - discusses quality management systems regarding medical devices.

Tools / Platforms

Model Development

- Tensorflow
- Keras
- Colab

Model Hosting

- Google Cloud Platform
- Amazon Web Services

Application

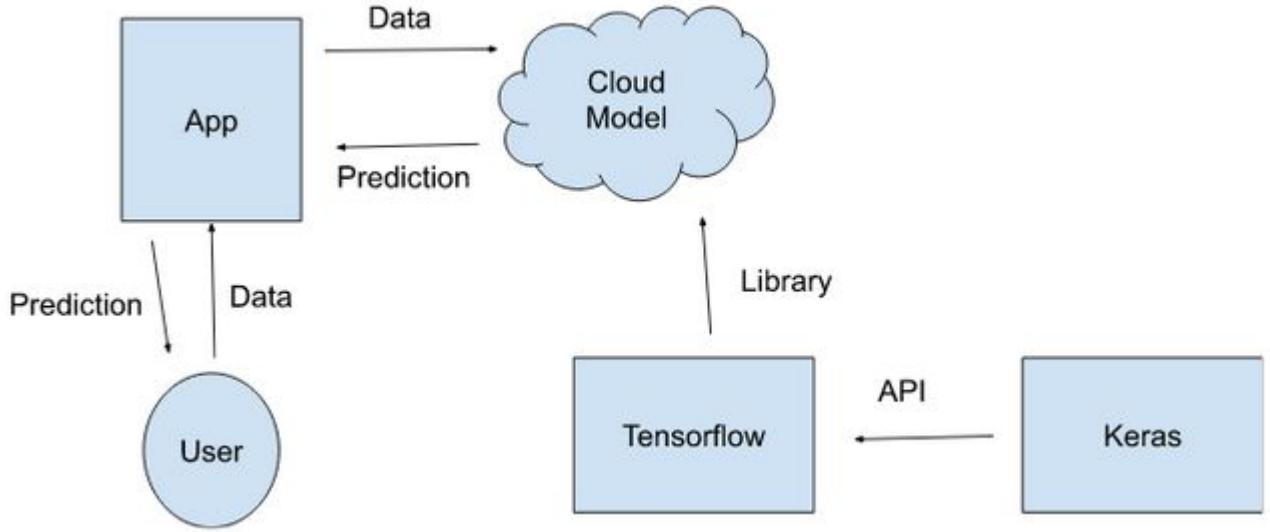
- Tkinter
- Python Flask
- OpenSSL

Users & Uses

- **Medical professionals** such as oncologists and **healthcare providers** seeking enhanced predictive tools for cancer outcomes.
- **Patients** or other users with access to pathology data that may want to inquire about their prognosis

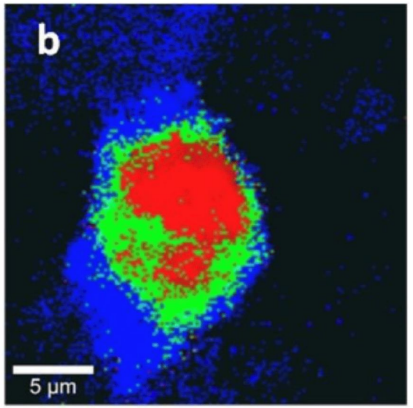
- Assisting medical professionals in making more **accurate** and timely **cancer-related** predictions.
- Serving as a supplementary tool for informed **decision-making** in cancer treatment.

Initial Design



Data Set

- Lab provided samples
  - encoded cell image of cancerous or benign structure
  - “data from some sensor”
  - extracts some sort of useful information on the cell structure
- Don’t necessarily need to understand their implications



Data Set

- One CSV file per patient
  - data represented via a vector
  - 2449 samples per patient
- One “master” CSV file
    - Displays a survival number in months

Is there a correlation between the pathology data and the survival data?

Sample 1	
2.919128473	1.24E-01
2.916994473	1.24E-01
2.914860473	1.24E-01
2.912726473	1.24E-01
2.910592473	1.24E-01
2.908458473	1.24E-01
2.906324473	1.24E-01
2.904190473	1.24E-01
2.902056473	1.24E-01
2.899922473	1.24E-01
2.897788473	1.24E-01
2.895654473	1.24E-01
2.893520473	1.24E-01
2.891386473	1.24E-01
2.889252473	1.24E-01
2.887118473	1.24E-01
2.884984473	1.24E-01
2.882850473	1.24E-01
2.880716473	1.24E-01
2.878582473	1.24E-01
2.876448473	1.23E-01
2.874314473	1.23E-01
2.872180473	1.23E-01
2.870046473	1.23E-01
2.867912473	1.23E-01
2.865778473	1.23E-01
2.863644473	1.23E-01
2.861510473	1.23E-01
2.859376473	1.23E-01
2.857242473	1.23E-01

Single Patient Data

num	sample	Survival
0	1	21
1	2	213
2	3	100
3	4	54
4	5	24
6	6	21
8	7	56
10	8	135
12	9	32
13	10	102
14	11	78
15	12	93
16	13	120
17	14	21
18	15	12
19	16	65
21	17	37
22	18	230
23	19	99
24	20	102

Master CSV

Data Pre-Processing

Challenges in Data Preparation

- Incomplete Master CSV
  - Not all samples are present in the master CSV file
- Training Data Set Discrepancies
  - Not all samples listed in the master CSV are included in the training dataset
- Corrupted Sample Files
  - Some sample files contain corrupted or inconsistent data

Data Cleansing Process

- Identify and Remove Corrupted Data
  - Use of parsing methods to flag corrupted samples
  - Manually remove corrupted sample (one-time task - small data set)

Data Pre-Processing

Data Integration and Sorting

- Match Samples with Master CSV
  - Filter data to retain only samples listed in the master CSV
- Align with Training Data Set
  - Further refine data to include only samples present in the training dataset

Outcome

- Clean dataset containing valid and relevant samples
- Ready for subsequent data analysis and modeling
- Total of 340 valid samples
- Each full sample is a list of 2449 pairs of data points
- shape = (1, 2449, 2)

Model Implementation

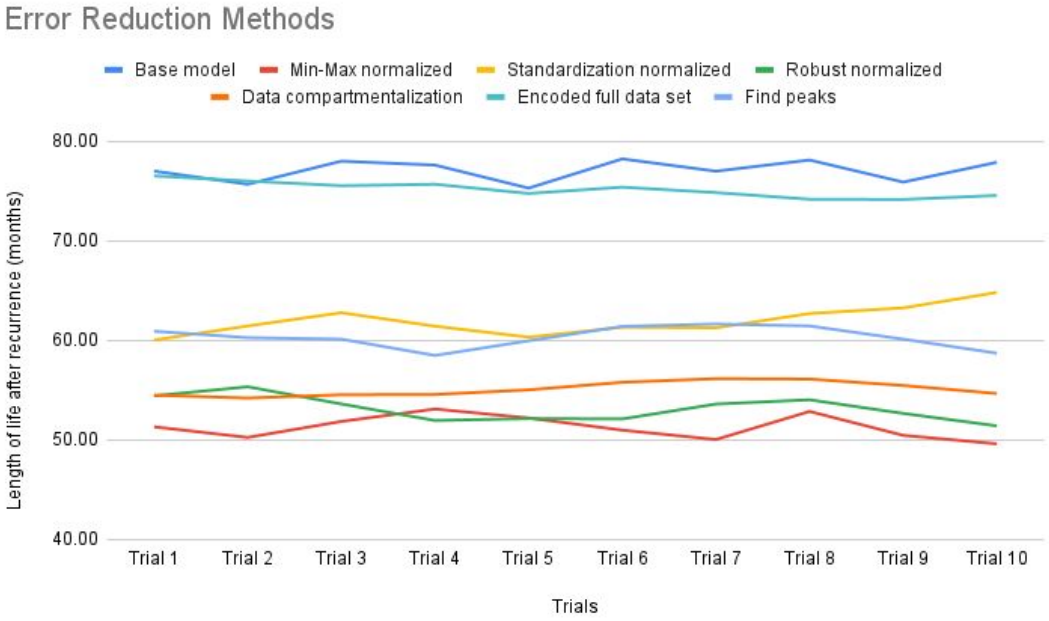
- Sequential Keras model
  - Straightforward to define
  - Easy prototyping and experimentation
  - Efficient in training
- “Base” model
  - 4 dense layers (64, 32, 16, 1)
  - Rapidly achieved accuracy convergence
  - Provided a relatively low error compared to other models

Error Reduction Methods

- Normalization
  - Ensures all data is on a similar scale
  - Min-Max, Standardization, Robust
- Identifying peaks
  - Treats each sample as a graph
  - Identifies highs and lows
- Compartmentalization of data points
  - Simplifies the data by focusing on the first 250 data points, first 500 data points, etc
- Unsupervised learning (autoencoding)
  - Compresses input data into a lower dimensional representation

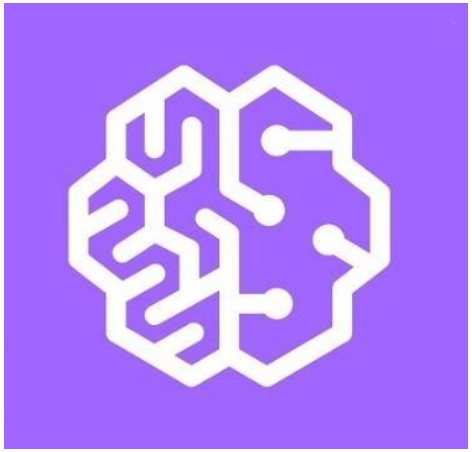
Model Testing

- Normalization (min-max) and Compartmentalization found most effective
- Find peaks found less effective
- Autoencoding found ineffective



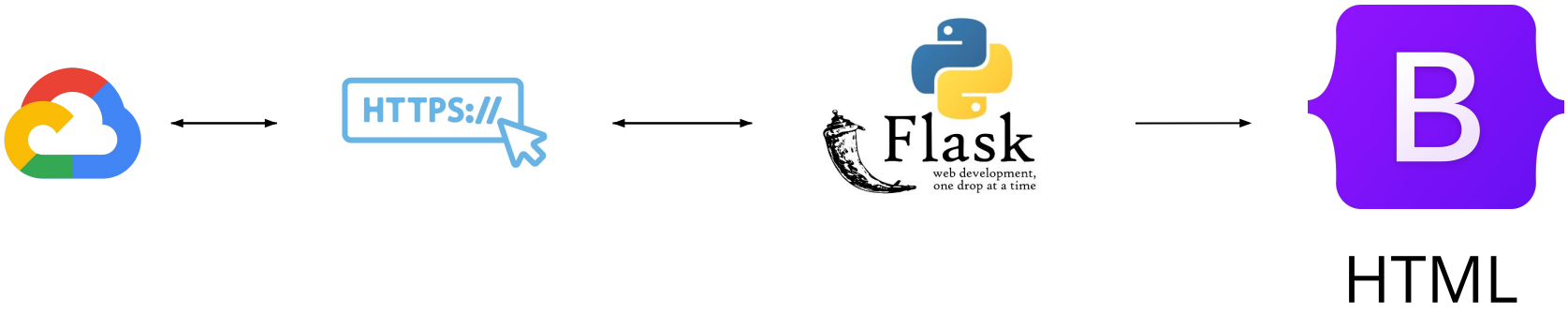
Model Deployment

GCP Vertex AI



AWS Sagemaker

Application Design



Application Testing

- End-to-end tests
  - Application prediction vs raw prediction
    - 15+ samples
  - Ensures data is transferred properly
- Stress testing
  - make simultaneous requests
  - view its effects on performance

Security

1. Model Development
    a. Data handling done physically
    b. Colab data stored on cloud (encrypted)
2. Application
    a. Implemented as HTTPS
        i. Transport Security Layer
    b. No data is ever stored
3. Deployment
    a. GCP and AWS require authentication
    b. Everything is encrypted

Cloud Comparison

GCP

Training

- Google Colab Jupyter Notebook
- Model trained in ~50 s

Cost

- Training free
- Deployment = $0.05 per hour with Vertex AI

AWS

Training

- Sagemaker Jupyter Notebook
- Model trained in ~50 s

Cost

- Training free
- Deployment = $0.05 per hour with Sagemaker

Conclusions

- Simplifying the data (normalization, compartmentalization) was most successful in reducing model prediction error
- Model consistently predicts ~150 months
  - average survival number in data set ~148 months
  - model guessing average?
- Is there even a correlation?
  - may require deeper understanding of data preprocessing and neural networks
- Better understanding of dataset and context would have helped

Thank you!

Any questions?