Accurate Cancer Prediction

Team 47

Introduction

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, however with the recent developments of AI we may be able to predict cancer with higher accuracy than previously. Our project is to build and train a simple AI model to provide accurate predictions of cancer given a set of data. Our primary users will be Medical Professionals with the model being used to aid in medical diagnosis. This is important because any progress in being able to predict cancer’s movements would greatly aid in not only deshrouding the mystery around cancer but also potentially decreasing the number of fatalities in the not just the United States but worldwide.

Work Progress (491)

1. Understand the domain and problem
   a. 100% complete
2. Investigate AI as a solution
   a. 100% complete
3. Model selection
   a. 100% complete
4. Train basic models
   a. 100% complete

Work Progress (492)

- UI Development
    - 75% complete
- Data processing and model training
    - 50% complete
    - Need to increase accuracy
    - Better methods to process data
- Performance / port to other cloud
    - 70% complete
    - Began AWS set up
    - Have not trained on AWS

Detailed Design

We are currently using Google Colab on the Google Cloud to securely store the medical data and to train and test our models. In the future we plan to test if other cloud platforms, like AWS, has an effect on model accuracy.

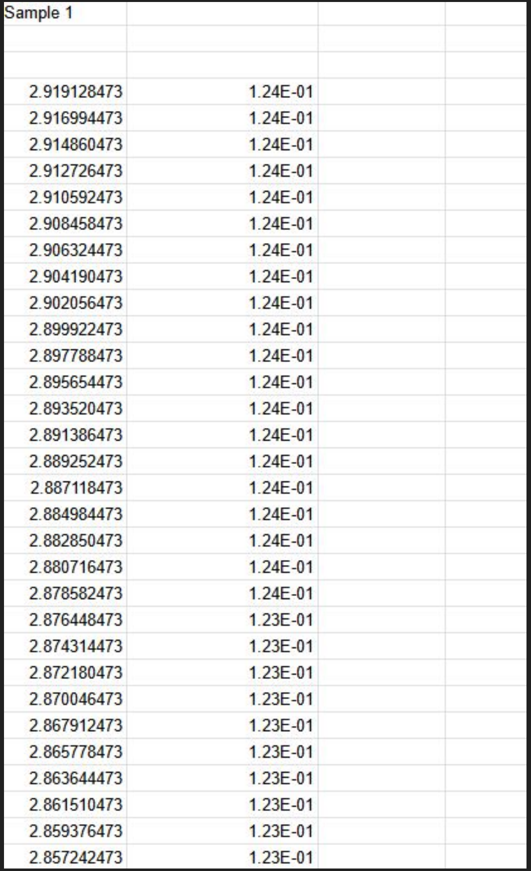
On the cloud we preprocess the data using Pandas. Once it is processed, we use Keras on TensorFlow to train dense neural network models.

Then, using Tkinter for front end, we host our model as part of an application.

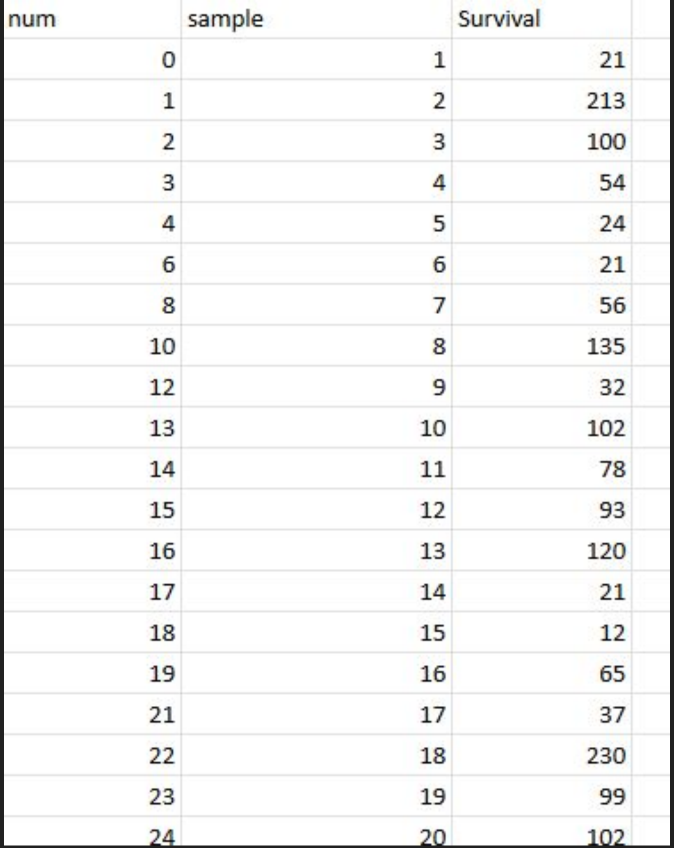
Data

- Data
  - One CSV file per patient
    - Pathology data represented via a vector
    - Thousands of samples per patient
  - One “master” CSV file
    - Displays the number of months the patient survived
  - Extracting useful criteria (find peaks)

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



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

Demonstrate Current Design

1. Will use fabricated data with no medical significance to show client experience
2. Data preprocessing and model loaded on clients end
   a. Plans to migrate to cloud (AWS, Google Cloud) - Centralizes model updates/changes - Privacy concerns
3. User provides csv file as input
4. User will be given a predicted survival value
   a. Again, this value holds no medical significance
   b. For demo purposes only

Challenges and Solutions

1. Handling malformed or corrupt data
   - Wrote a python script to help preprocess our data by filtering out corrupted or incomplete samples
2. How to best represent our data in the training of the model
   - Utilized Pandas Dataframes to store patient data and experimented with different ways of representing the data in the model training
3. 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, and trying different types of models.

Conclusion