Group 45

By: Bao, Arul, Taylor

Goal

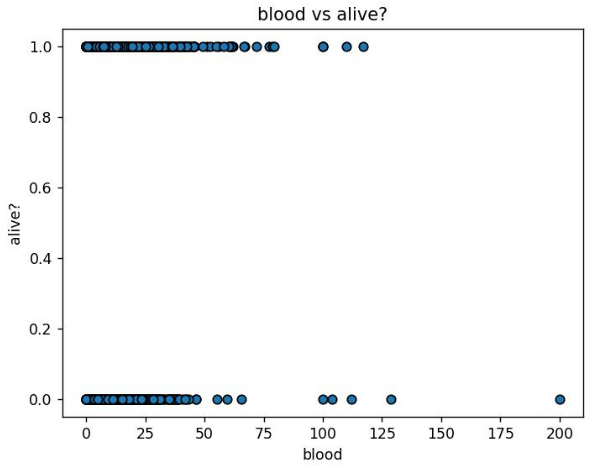
- The goal that we are trying to do is train an AI model that can accurately predict which patient will survive based on medical information provided.

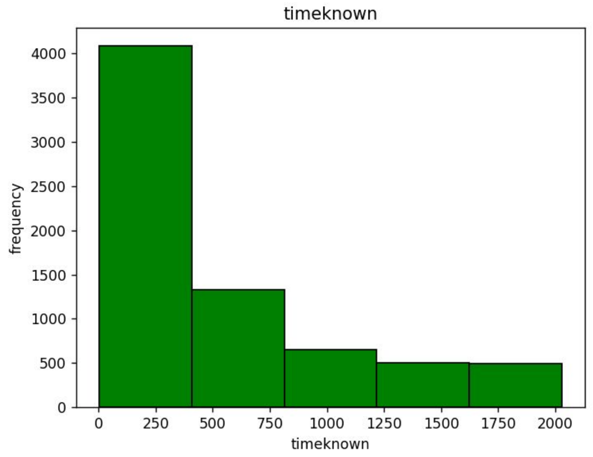
What our plan was

Our plan was to analyze the data and see how we can clean it up before implementing our model. We wanted to find which columns to use and which to exclude, as well as which data points to keep. To do this, we tried to find the relationship between each category and death to see the correlation and see how much each feature affected the output. Then, we attempted to get rid of outliers, unnecessary columns, empty data, and duplicates.

How we trimmed down the data

- Initially, we created histogram graphs of each data type given (sex, blood, bloodchem1..etc.) in order to look for any outliers. Then we used scatter plot on those data with outliers to narrow down the scope of data that we want our model to test on.

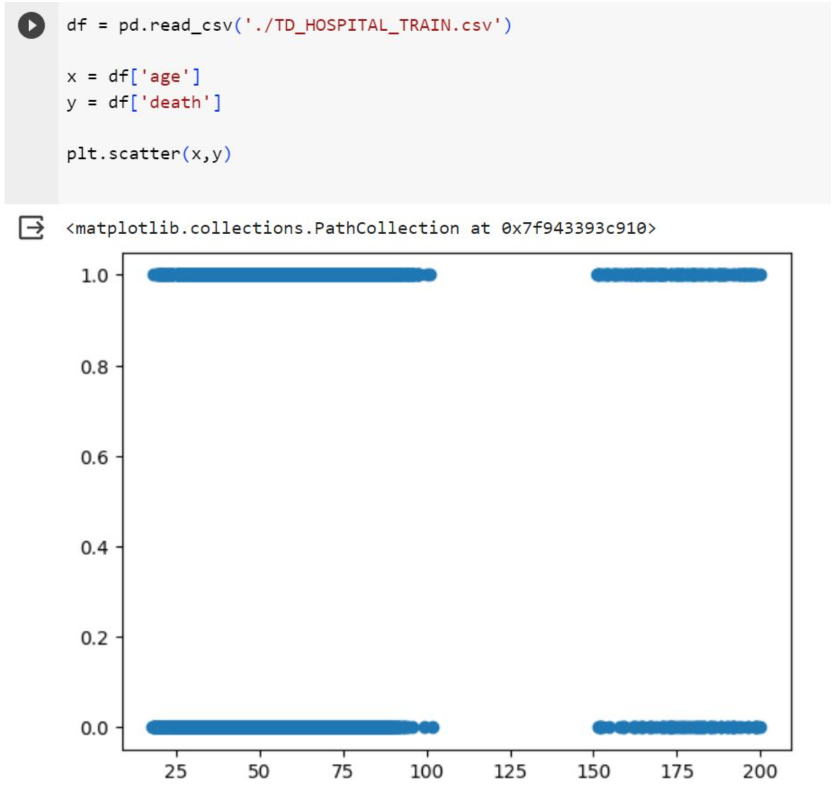




Issues Along the Way

When graphing the relationships between each feature and death, we saw a lot of unreliable and weird data. For example, age had values from 150 to 200. However, when dropping so much data, we accidentally overfitted our training model, as we would get 0.97 or 1.0 accuracy with the training data but 0.28 when submitting to attorney with test data.

We also faced a lot of issue submitting the code, as we tried to drop test data, which is not allowed.

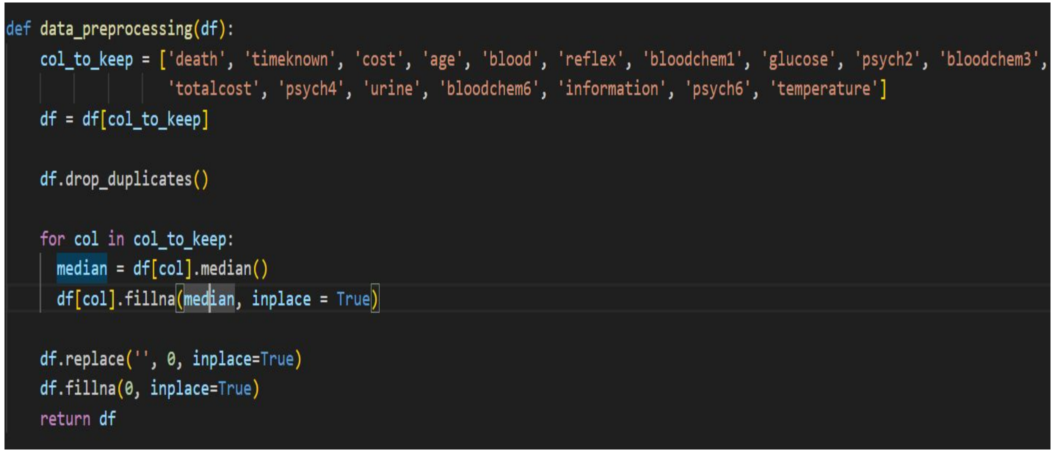


```
df = pd.read_csv('./TD_HOSPITAL_TRAIN.csv')

x = df['age']
y = df['death']

plt.scatter(x,y)
```

```
<matplotlib.collections.PathCollection at 0x7f943393c910>
```

Fixing the Issue and Implementing the Solution

Instead of dropping values, we kept the outliers and we instantly saw better results. However, there were still empty cells so we replaced them with the median of that column. We also removed duplicate values so there was no unnecessary, extra data for the model to go through. We used pandas to alter the data in our code.



```python
def data_preprocessing(df):
    col_to_keep = ['death', 'timeknown', 'cost', 'age', 'blood', 'reflex', 'bloodchem1', 'glucose', 'psych2', 'bloodchem3',
                   'totalcost', 'psych4', 'urine', 'bloodchem6', 'information', 'psych6', 'temperature']
    df = df[col_to_keep]

    df.drop_duplicates()

    for col in col_to_keep:
        median = df[col].median()
        df[col].fillna(median, inplace = True)

    df.replace('', 0, inplace=True)
    df.fillna(0, inplace=True)
    return df
```

Conclusion/Reflection

We finally got our model to 64 percent accuracy on the test data. It is obviously still not the best and can definitely improve but it is a huge improvement from 28. If we had more time, we could make our model more accurate by filtering and cleaning the data a little better. With more time we could have take into account race, disability, primary, and dnr as we were not able to develop a solution to converting subjective data into quantifiable data that we can train our model on. Also, we could experiment with different model types, for example a Decision Tree Regressor, to see which would give the most accurate results.