Horse Survival Prediction
Data Mining Project

Farid Ghorbani
Azad University of Tehran
Tehran, Iran
April 2020

1. Introduction

Data mining is the process of finding and extracting hidden information, patterns, and specific relationships within a vast amount of data with the aim of predicting future events and outcomes. Colic in horses is caused by improper attention to horses, which can lead to impaction, intestinal twisting, stomach ulcers, stomach rupture, and intestinal obstruction, resulting in colic. Colic usually occurs suddenly and unexpectedly, and it is important to identify the cause of this disease as quickly as possible in order to prevent its progression and control it.

In this project, we aim to analyze and model pre-collected data on horses previously affected by this disease, focusing on predicting mortality, survival, and hospitalization likelihood. We seek to develop a model capable of predicting outcomes for new data if provided with the required information, which includes factors such as experience with colic in horses, heart rate, etc. Various tools are available for data mining, some of which are widely used tools for data science, even for those with limited programming knowledge.

2. Dataset

Our dataset consists of two files. The first file contains information on 300 horses, which serves as our training data. The second file contains data on 68 additional horses, used for testing our model. The variable we aim to predict is labeled "V23," indicating the survival outcome of the horse.

Attribute Information:

1. **Surgery:** Binary (1 if the horse underwent surgery for colic, 0 otherwise)
2. **Age**: Categorical (1 for adult horses, 2 for young horses)
3. **Hospital Number:** Unique numerical identifier for each horse
4. **Rectal Temperature:** Continuous (in degrees Celsius)

5. **Pulse:** Continuous (heart rate in beats per minute)
6. **Respiratory Rate:** Continuous (breaths per minute)
7. **Temperature of Extremities:** Ordinal (1 = normal, 2 = warm, 3 = hot, 4 = cold)
8. **Peripheral Pulse:** Ordinal (1 = normal, 2 = increased, 3 = reduced, 4 = absent)
9. **Mucous Membranes:** Ordinal (1-6 representing different colors indicating blood circulation status)
10. **Capillary Refill Time:** Ordinal (1 for <3 seconds, 2 for >=3 seconds)
11. **Pain:** Ordinal (1-5 representing pain severity)
12. **Peristalsis:** Ordinal (1-4 representing intestinal activity)
13. **Abdominal Distension:** Ordinal (1-4 representing severity)
14. **Nasogastric Tube:** Ordinal (1-3 representing presence of gas in the tube)
15. **Nasogastric Reflux:** Ordinal (1 for none, 2 for >1 liter, 3 for <1 liter)
16. **Nasogastric Reflux PH:** Continuous (pH scale)
17. **Rectal Examination Feces:** Ordinal (1-4 representing consistency)
18. **Abdomen:** Ordinal (1-5 representing severity)
19. **Packed Cell Volume:** Continuous
20. **Total Protein:** Continuous
21. **Abdominocentesis Appearance:** Ordinal (1-3 representing clarity)
22. **Abdominocentesis Total Protein:** Continuous
23. **Outcome:** Categorical (1 for lived, 2 for died, 3 for euthanized)
24. **-26. Type of Lesion:** Categorical
27. **Surgical Lesion:** Binary
28. **CP Data:** Binary

3. Data Preprocessing

In the first stage, we load the data.

```
import csv
with open('horse_colic.csv') as f:
    reader = csv.reader(f)
    listOfRecords = []
    for row in reader:
        listOfRecords.append(row)
    for record in listOfRecords:
        print(record)
```

```
['1 1 530693 ? ? ? 3 3 3 3 5 3 3 3 2 ? 4 5 46.00 5.90 ? ? 2 1 03025 00000 00000 2 ']
['1 1 527463 37.50 96 18 1 3 6 2 3 4 2 2 3 5.00 ? 4 69.00 8.90 3 ? 1 1 02208 00000 00000 1 ']
['1 1 527518 36.40 98 35 3 3 4 1 4 3 2 ? ? ? 4 4 47.00 6.40 3 3.60 2 1 02205 00000 00000 1 ']
['1 1 534756 37.3 40 ? ? 3 1 1 2 3 2 3 1 ? 3 5 36 ? 3 2 1 1 03111 00000 00000 2 ']
['1 9 5290759 38.10 100 80 3 1 2 1 3 4 1 ? ? ? 1 ? 36.00 5.70 ? ? 1 1 03111 00000 00000 2 ']
['1 1 5279822 38.00 ? 24 3 3 6 2 5 ? 4 1 1 ? ? ? 68.00 7.80 ? ? 2 1 03205 00000 00000 2 ']
['1 1 529849 37.80 60 80 1 3 2 2 2 3 3 ? 2 5.50 4 ? 40.00 4.50 2 ? 1 1 05206 00000 00000 1 ']
['2 1 529304 38.00 54 30 2 3 3 3 3 1 2 2 2 ? ? 4 45.00 6.20 ? ? 1 2 00400 00000 00000 2 ']
['1 1 529388 ? 88 40 3 3 4 2 5 4 3 3 ? ? 4 5 50.00 7.70 3 1.40 2 1 04205 00000 00000 1 ']
['2 1 528006 ? 40 16 ? ? ? 1 ? ? ? ? ? ? ? ? 50.00 7.00 2 3.90 3 1 02208 00000 00000 1 ']
['2 1 529703 39.00 64 40 1 1 5 1 3 3 2 2 1 ? 3 3 42.00 7.50 2 2.30 1 2 05000 00000 00000 1 ']
['2 1 535381 38.3 42 10 1 1 1 1 1 1 1 ? ? ? ? ? 38 61 ? ? 1 2 00000 00000 00000 2 ']
['2 1 534197 38.0 52 16 ? ? ? ? 2 ? ? ? 3 1 1 1 53 86 ? ? 1 1 02322 00000 00000 2 ']
['2 1 529461 40.30 114 36 3 3 1 2 2 3 3 2 1 7.00 1 5 57.00 8.10 3 4.50 2 1 03205 00000 00000 1 ']
['2 1 530251 38.80 50 20 3 1 1 1 1 1 1 2 1 ? 3 1 42.00 6.20 ? ? 1 2 00000 00000 00000 2 ']
['2 1 530310 ? ? ? 3 3 1 1 5 3 3 1 1 ? 4 5 38.00 6.50 ? ? 2 1 03205 00000 00000 2 ']
['2 1 528729 37.50 48 30 4 1 3 1 ? 2 1 1 1 ? 1 1 48.00 8.60 ? ? 1 2 00000 00000 00000 2 ']
['1 1 535158 37.3 48 20 ? 1 2 1 3 3 3 2 1 ? 3 5 41 69 ? ? 1 1 03205 00000 00000 2 ']
['2 1 530319 ? 84 36 ? ? 3 1 ? 3 1 2 1 ? 3 2 44.00 8.50 ? ? 1 1 03111 00000 00000 2 ']
['1 1 534145 38.1 88 32 3 3 4 1 2 3 3 ? 3 1 4 5 55 60 ? ? 3 2 03205 00000 00000 2 ']
```

In the data, there are certain values '?' that represent our missing values.

There are various methods to address this issue, one of which is to delete these fields, which reduces the size of our data set and consequently decreases the accuracy of our model. Much better methods, such as classification with the (knn) nearest neighbor algorithm, exist. This method works by finding the nearest neighbors and filling in the missing feature with their value.

```python
import csv
with open('horse_colic.csv') as f:
    reader = csv.reader(f)
    listOfRecords = []
    for row in reader:
        listOfRecords.append(row)
    for i in range(0 ,len(listOfRecords)):
        listOfRecords[i] = listOfRecords[i][0].split()
    count_r = 0
    mean = [[],[],[]]
    for record in listOfRecords :
        label = record[22]
        countOfRecord = 0
        for attribute in record :
            mean[0].append(0)
            mean[1].append(0)
            mean[2].append(0)
            if label == '1' and attribute != '?':
                mean[0][countOfRecord] += float(attribute)
            elif label == '2' and attribute != '?':
                mean[1][countOfRecord] += float(attribute)
            elif label == '3' and attribute != '?':
                mean[2][countOfRecord] += float(attribute)
            countOfRecord += 1
```

```python
        count = 0
        for attribute in record :
            mean[0][count] = mean[0][count]/countOfRecord
            mean[1][count] = mean[1][count]/countOfRecord
            mean[2][count] = mean[2][count]/countOfRecord
            count += 0
        count = 0
        for attribute in record :
            if attribute == '?' :
                if label == '1':
                    listOfRecords[count_r][count] = str(mean[0][count])
                elif label == '2':
                    listOfRecords[count_r][count] = str(mean[1][count])
                elif label == '3':
                    listOfRecords[count_r][count] = str(mean[2][count])
            count += 1
        count_r += 1
```

In this project, we fill in these values using the mean of the features based on each label. We do this by first obtaining the mean of each feature based on its label. As you can see, we store this in mean[NumberOfLabel][attribute]. Then, if we encounter a feature with a value of '?', we replace it with the mean.

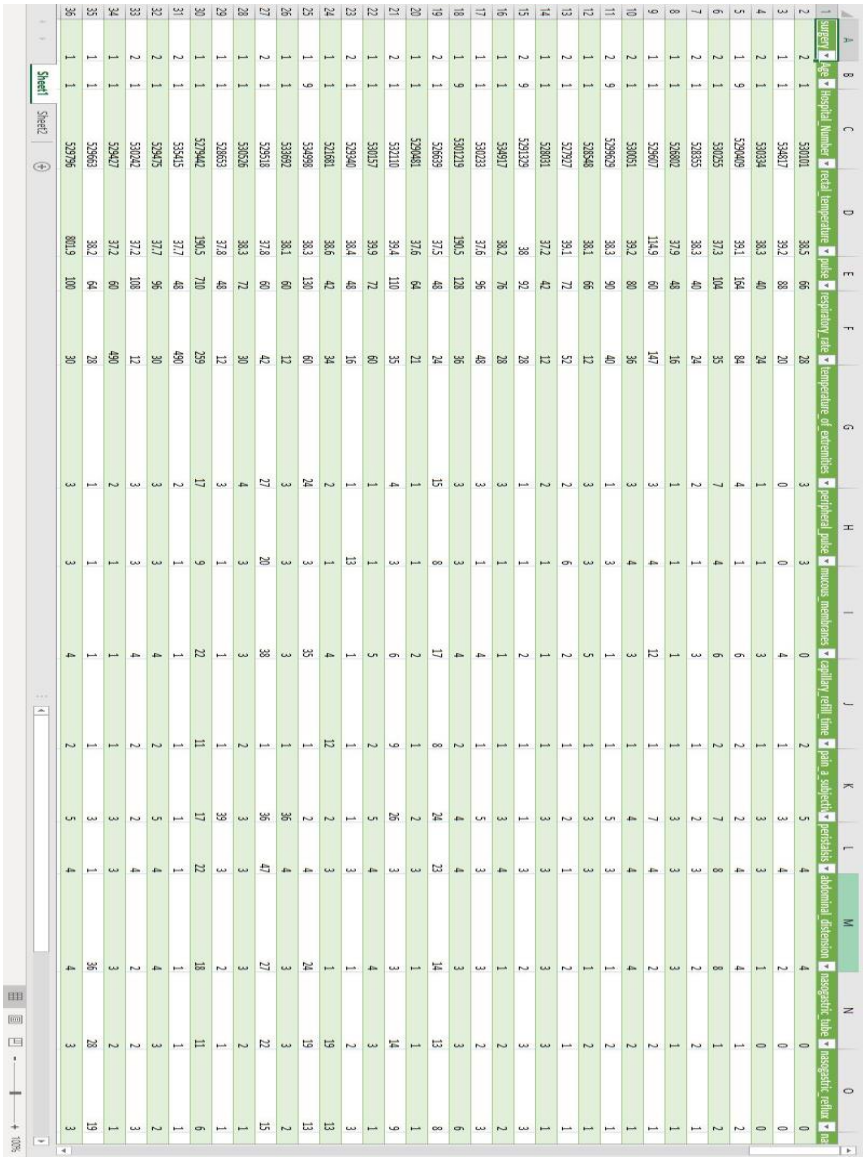
Next, we move on to data cleaning, fortunately, the data in this data warehouse is clean, and we do not need to perform any operations.

Then, we delete attributes 23 and onwards as these fields will not assist our model.

```
print(listOfRecords[0])
for i in range(0 , len(listOfRecords)) :
    listOfRecords[i] = listOfRecords[i][:23]
print(listOfRecords[0])
```

```
['2', '1', '530101', '38.50', '66', '28', '3', '3', '0', '2', '5', '4', '4', '0', '0', '0', '3', '5', '45.00', '8.40', '0', '0', '2', '2', '11300',
'00000', '00000', '2']
['2', '1', '530101', '38.50', '66', '28', '3', '3', '0', '2', '5', '4', '4', '0', '0', '0', '3', '5', '45.00', '8.40', '0', '0', '2']
```

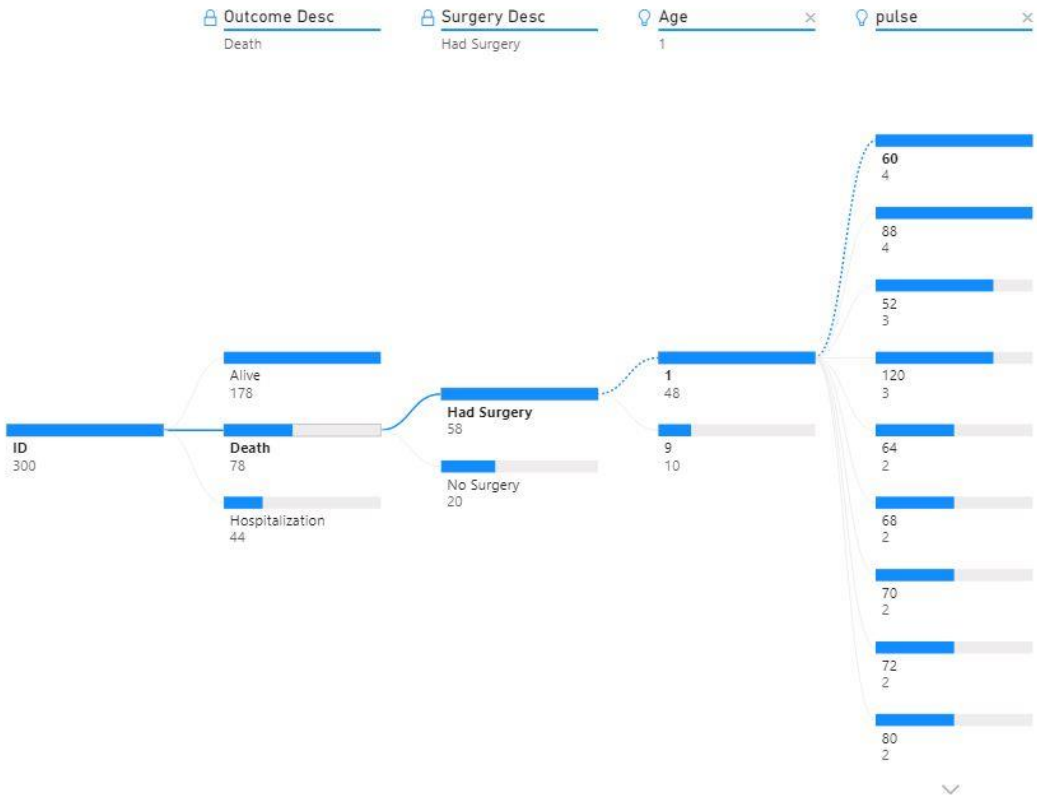
In the next step, we save the output CSV file along with the names of all attributes in an Excel file.



Our dataset is now in a usable format, with all missing values removed. The data values have been cleaned and are ready for the next stage.

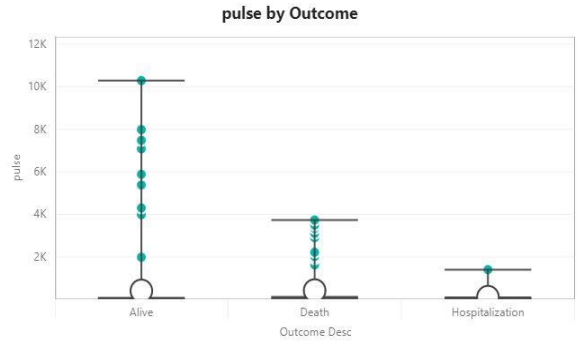
4. Data Analysis

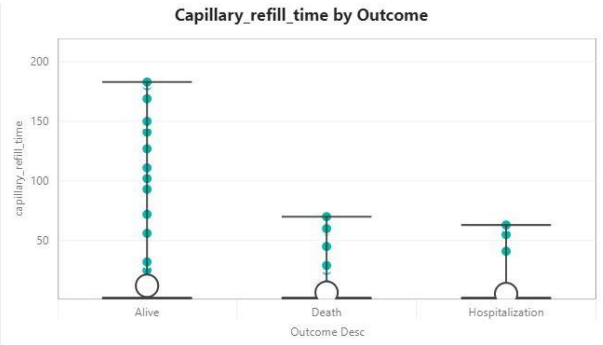
After completing the previous stage, we proceed to analyze the data using Power BI software. We create Boxplot, Histogram, and other visualizations for the features of the dataset, allowing us to evaluate the importance of each feature in the model.

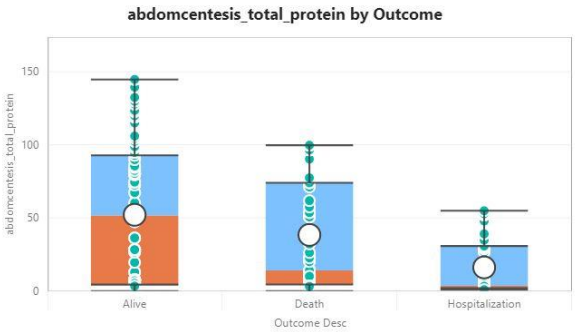


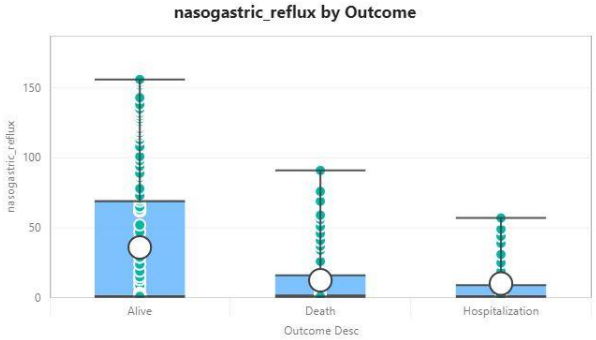
In this analytical chart, we observe the features of age, hospitalization history, and heart rate for samples that have been categorized based on whether they survived, died, or were hospitalized. Out of our 300 samples, 78 have resulted in death, with a higher percentage of hospitalization history among them. Moreover, this occurrence is more prevalent among older age groups. In these samples, the heart rate is measured significantly lower. The three charts below depict these same features for samples that survived and those that were hospitalized.

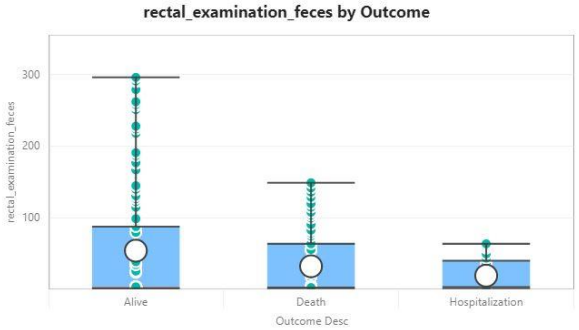
Below are some Boxplot charts displaying certain features:

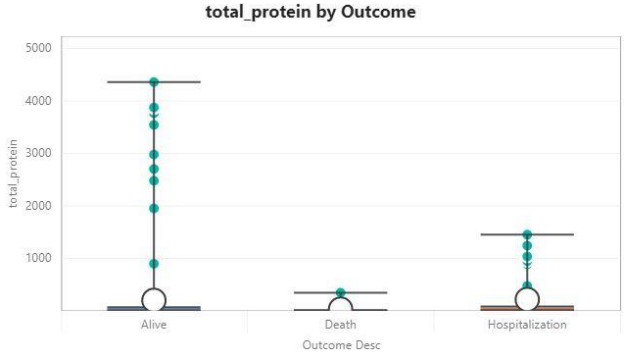


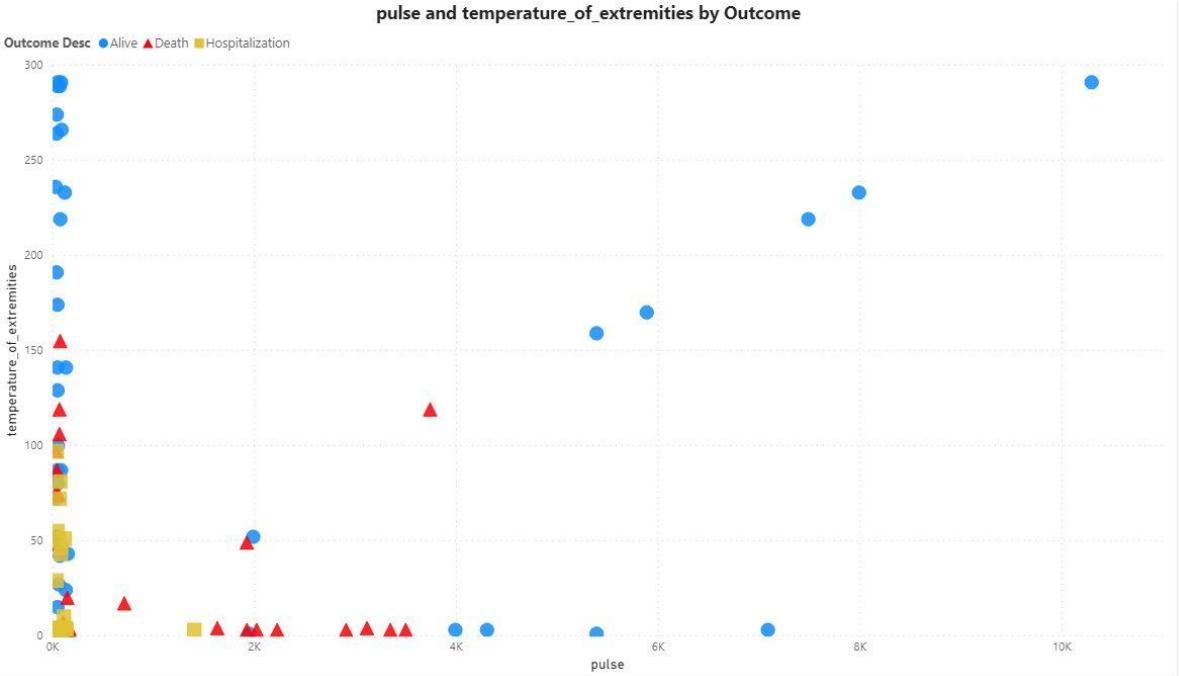






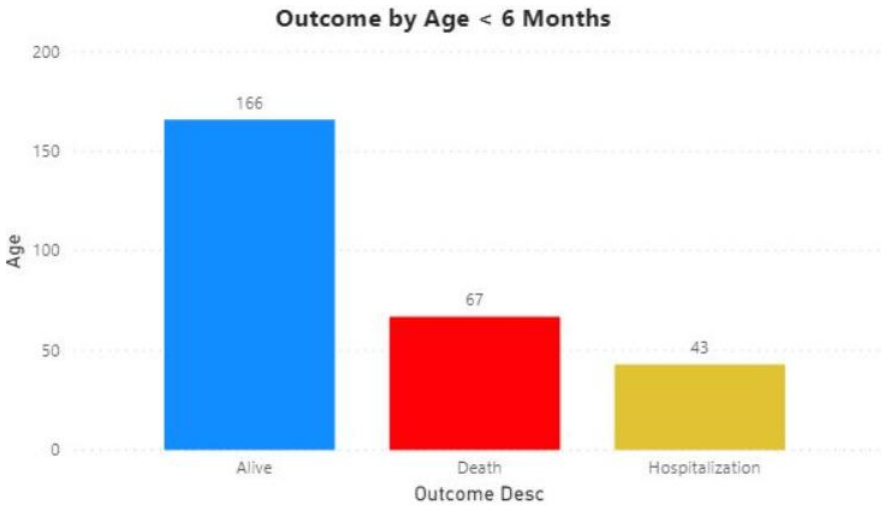




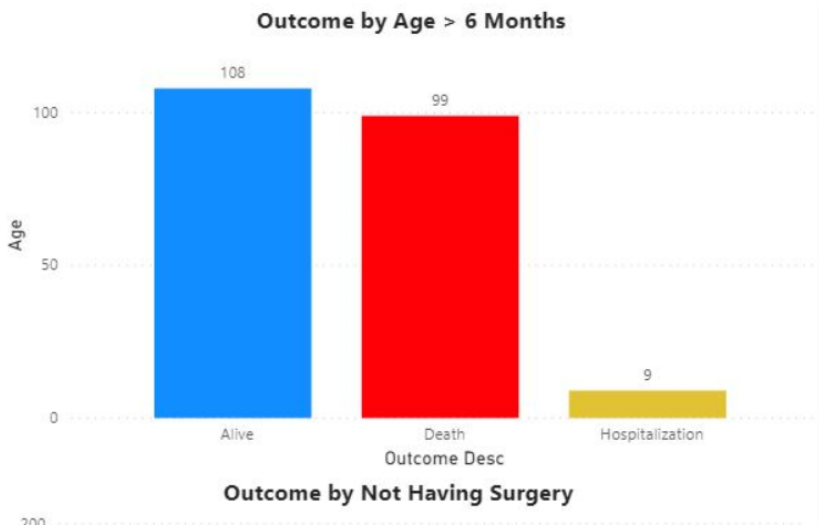


In this scatter plot, with heart rate on one axis and pressure on the other, you can observe blue circles representing surviving horses, triangles indicating deceased ones, and squares representing hospitalized ones.

As evident, all horses that died or were hospitalized have pressures below 150 and heart rates less than 4000, with this feature being more pronounced for hospitalized horses.



In this plot, as you can see, the number of horse deaths under 6 months is low, but in the next plot, this statistic is higher for adult horses.



Overall, the number of horses that died or were hospitalized is less than the number of surviving horses. However, the situation is different in contrast, with a significantly higher mortality rate among adult horses compared to young ones.

5. Decision Tree Model

Initially, we read the Excel file using the pandas library.

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

dataset_pressure = pd.read_excel(r'C:\Users\Farid\Desktop\DM project\New folder\horseColic2_out.xlsx')
```

Next, we fill in the training data using the features we want. We also specify the labels, which are the outcomes. If it is 1, the horse survives, if it is 2, it dies, and if it is 3, it is hospitalized, which is the feature our model is supposed to predict.

```
from sklearn.model_selection import train_test_split

X = dataset_pressure[['surgery','Age','Hospital_Number','rectal_temperature','pul
Y = dataset_pressure[['outcome']]
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3)
```

Then, using the sklearn library, we select the model features. Additionally, we allocate 30% of the data as the test set.

Next, we use the fit command to train our model. After that, it predicts the test data. Since we have the labels of the test data, we can calculate the accuracy of our model.

```
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier()
classifier.fit(X_train, Y_train)

Y_pred = classifier.predict(X_test)

from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(Y_test, Y_pred))
print(classification_report(Y_test, Y_pred))
```

We achieved a a test score of around 62.22%.

```
[[40  9  5]
 [10  7  0]
 [ 5  5  9]]
              precision    recall  f1-score   support

           1       0.73      0.74      0.73        54
           2       0.33      0.41      0.37        17
           3       0.64      0.47      0.55        19

    accuracy                           0.62        90
   macro avg       0.57      0.54      0.55        90
weighted avg       0.64      0.62      0.63        90
```

6. Acknowledgements

Source of the dataset:

- Mary McLeish & Matt Cecile
  Dept. of Computer Science
  U. of Guelph
  Guelph, Ontario, Canada N1G 2W1
  Email: mdmcleish@water.waterloo.edu

7. Reference

[1] https://archive.ics.uci.edu/ml/datasets/Horse+Colic

[2] https://dampezeshkan.com/cgi-sys/suspendedpage.cgi

[3] https://docs.microsoft.com/en-us/power-bi/fundamentals/desktop-what-is-desktop