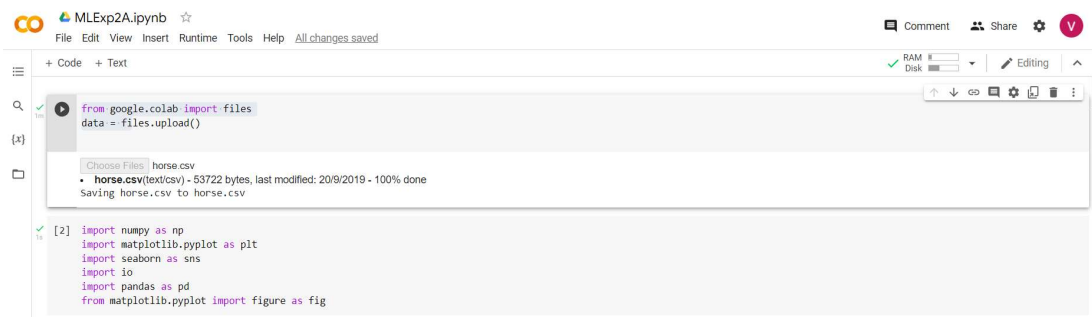
MINORS Lab 2A: Logistic Regression

Varun Kamath

2019110023

13/10/2022

1. Upload csv files and import libraries



2. Counted number of NaN values present for each attribute, and excluded attributes which have more than 100 NaN values or unimportant.

```
[7] df1 = df1.drop(columns=["hospital_number", "nasogastric_tube", "nasogastric_reflux", "nasogastric_reflux_ph", "rectal_exam_feces", "abdomen", "abdomo_appearance", "abdomo_protei

    df1 = df1.drop(columns=["lesion_1", "lesion_2", "lesion_3", "cp_data"])
```

```
df1.isnull().sum()
```

```
surgery                   0
age                       0
hospital_number           0
rectal_temp              60
pulse                    24
respiratory_rate         58
temp_of_extremities      56
peripheral_pulse         69
mucous_membrane          47
capillary_refill_time    32
pain                     55
peristalsis              44
abdominal_distention     56
nasogastric_tube        104
nasogastric_reflux      106
nasogastric_reflux_ph   246
rectal_exam_feces       102
abdomen                 118
packed_cell_volume       29
total_protein            33
abdomo_appearance       165
abdomo_protein          198
outcome                   0
surgical_lesion           0
lesion_1                  0
lesion_2                  0
lesion_3                  0
cp_data                   0
dtype: int64
```

3. Replaced the string values by their equivalent integer numbers

```
[10] df1['age'].replace(['young', 'adult'], [1, 2], inplace=True)
     df1['surgery'].replace(['yes', 'no'], [1, 2], inplace=True)
     df1['temp_of_extremities'].replace(['normal', 'warm', 'cool', 'cold'], [1, 2, 3, 4], inplace=True)
     df1['peripheral_pulse'].replace(['normal', 'increased', 'reduced', 'absent'], [1, 2, 3, 4], inplace=True)
     df1['mucous_membrane'].replace(['normal_pink', 'bright_pink', 'pale_pink', 'pale_cyanotic', 'bright_red', 'dark_cyanotic'], [1, 2, 3, 4, 5, 6], inplace=True)
     df1['capillary_refill_time'].replace(['less_3_sec', 'more_3_sec', 3], [1, 2, 2], inplace=True)
     df1['pain'].replace(['alert', 'depressed', 'mild_pain', 'severe_pain', 'extreme_pain'], [1, 2, 3, 4, 5], inplace=True)
     df1['peristalsis'].replace(['hypermotile', 'normal', 'hypomotile', 'absent'], [1, 2, 3, 4], inplace=True)
     df1['abdominal_distention'].replace(['none', 'slight', 'moderate', 'severe'], [1, 2, 3, 4], inplace=True)
     df1['outcome'].replace(['lived', 'died', 'euthanized'], [1, 2, 3], inplace=True)
     df1['surgical_lesion'].replace(['yes', 'no'], [1, 2], inplace=True)
```

4. Replace NaN by average values of respective column data

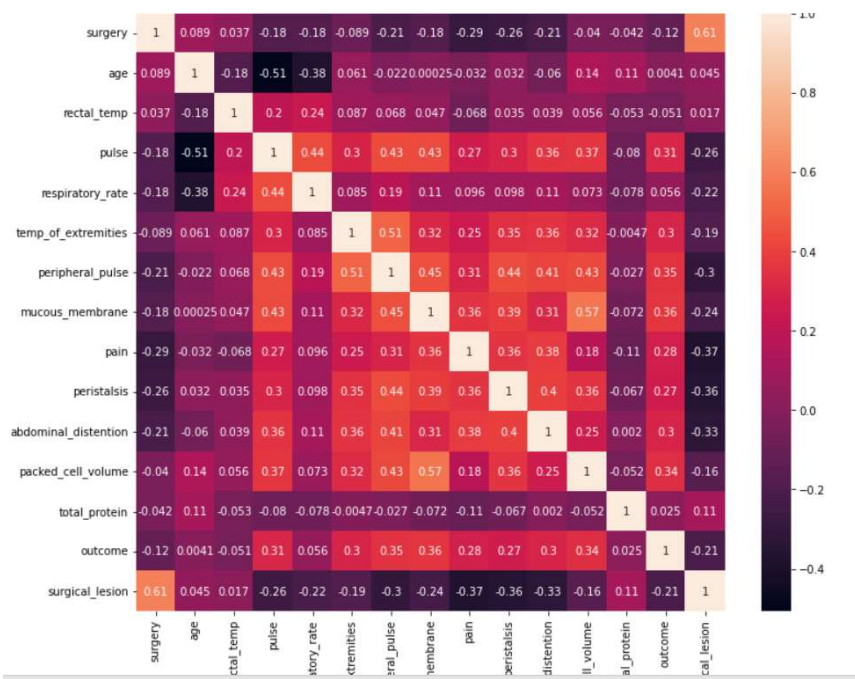
```
df1['rectal_temp'] = df1['rectal_temp'].fillna(df1['rectal_temp'].mean())
df1['pulse'] = df1['pulse'].fillna(round(df1['pulse'].mean()))
df1['respiratory_rate'] = df1['respiratory_rate'].fillna(round(df1['respiratory_rate'].mean()))
df1['temp_of_extremities'] = df1['temp_of_extremities'].fillna(round(df1['temp_of_extremities'].mean()))
df1['peripheral_pulse'] = df1['peripheral_pulse'].fillna(round(df1['peripheral_pulse'].mean()))
df1['mucous_membrane'] = df1['mucous_membrane'].fillna(round(df1['mucous_membrane'].mean()))
df1['capillary_refill_time'] = df1['capillary_refill_time'].fillna(1)
df1['pain'] = df1['pain'].fillna(round(df1['pain'].mean()))
df1['peristalsis'] = df1['peristalsis'].fillna(round(df1['peristalsis'].mean()))
df1['abdominal_distention'] = df1['abdominal_distention'].fillna(round(df1['abdominal_distention'].mean()))
df1['packed_cell_volume'] = df1['packed_cell_volume'].fillna(round(df1['packed_cell_volume'].mean()))
df1['total_protein'] = df1['total_protein'].fillna(round(df1['total_protein'].mean()))
```

5. Choose which data attributes you want as independent variables, segregate data into 80 percent train and 20 percent test data. Plot heat map of data frame.

```
[39] x = df1[['age', 'pulse', 'respiratory_rate', 'temp_of_extremities', 'peripheral_pulse', 'mucous_membrane',
              'capillary_refill_time', 'total_protein', 'pain', 'peristalsis', 'abdominal_distention', 'packed_cell_volume']]
     y = df1['outcome']
```

```
[40] from sklearn.model_selection import train_test_split
     from sklearn.metrics import accuracy_score
     x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)
     x_train.shape
     df1.outcome.describe()
     (df1.outcome == 0).sum()
     #sns.pairplot(df1[['surgery', 'age', 'rectal_temp', 'pulse', 'respiratory_rate', 'temp_of_extremities', 'peripheral_pulse', 'mucous_membrane',
     #         'capillary_refill_time', 'total_protein', 'pain', 'peristalsis', 'abdominal_distention', 'packed_cell_volume']].corr())

     plt.figure(figsize=(12,10))
     sns.heatmap(df1.corr(), annot=True)
     plt.show()
```



6. Evaluate the accuracy of model

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
clf = LogisticRegression(random_state=0, solver='lbfgs', multi_class='ovr', max_iter=10000)
clf = clf.fit(x_train, y_train)
clf.predict(x_test)
clf.score(x_test, y_test)
```

```
0.6333333333333333
```