Bharatiya Vidya Bhavan's
Sardar Patel Institute of Technology
Bhavan's Campus, Munshi Nagar, Andheri (West), Mumbai-400058-India
(Autonomous College Affiliated to University of Mumbai)

BE-ETRX **UID:**2019110039 **Sub-Minor ML**
NAME: Devansh Palliyath

Exp 2A

Horse Colic DataSet

Logistic Regression

Aim: To determine the output of the horse colic dataset by performing Logistic Regression and bringing the accuracy as high as possible.

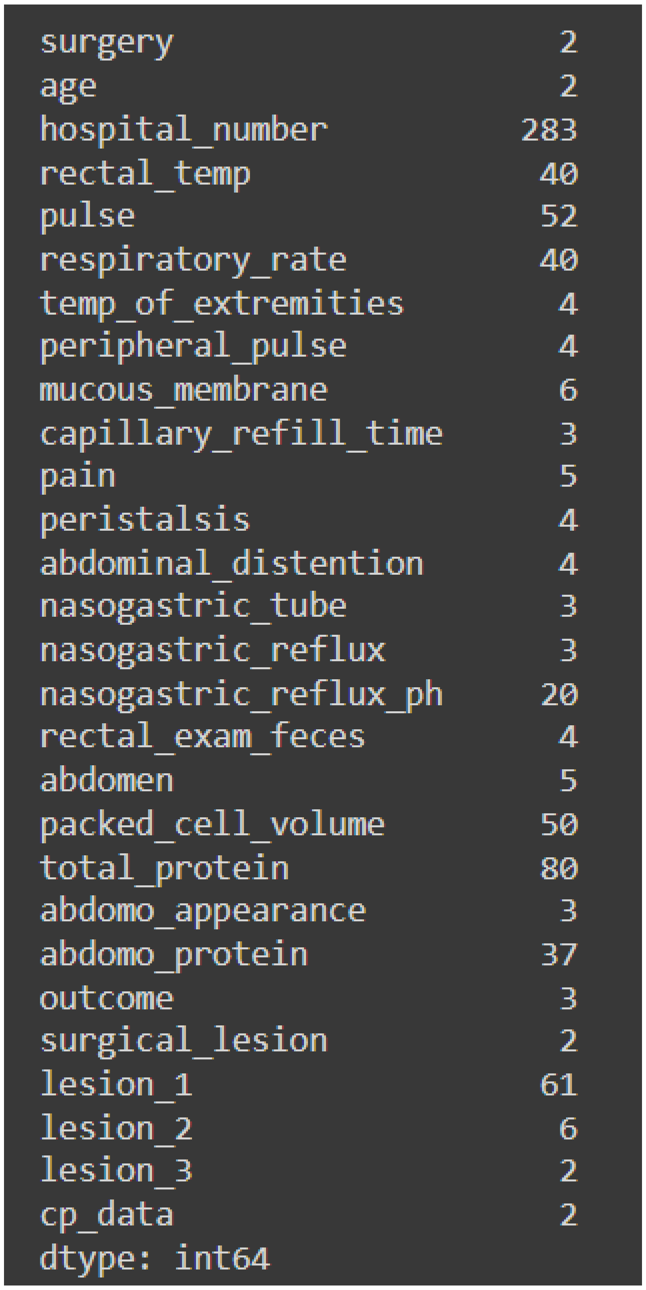
Objective:

1. Perform logistic regression
2. Bringing high accuracy.

Dataset Link: https://www.kaggle.com/datasets/uciml/horse-colic

Dataset Info:

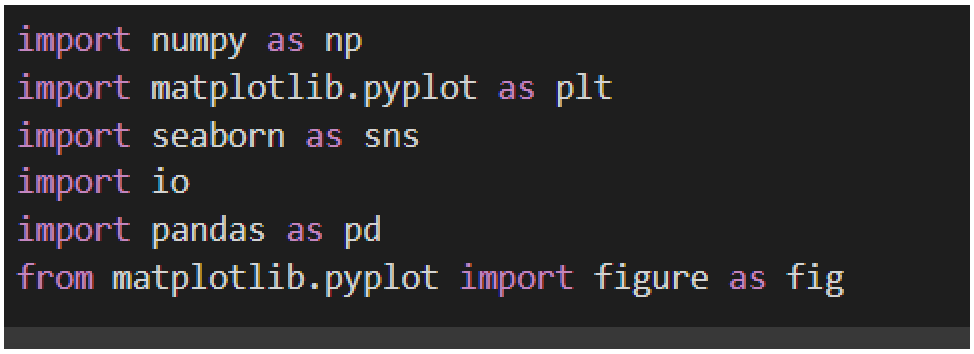
299 rows and 28 columns



```
surgery                    2
age                        2
hospital_number          283
rectal_temp               40
pulse                     52
respiratory_rate          40
temp_of_extremities        4
peripheral_pulse           4
mucous_membrane            6
capillary_refill_time      3
pain                       5
peristalsis                4
abdominal_distention       4
nasogastric_tube           3
nasogastric_reflux         3
nasogastric_reflux_ph     20
rectal_exam_feces          4
abdomen                    5
packed_cell_volume        50
total_protein             80
abdomo_appearance          3
abdomo_protein            37
outcome                    3
surgical_lesion            2
lesion_1                  61
lesion_2                   6
lesion_3                   2
cp_data                    2
dtype: int64
```

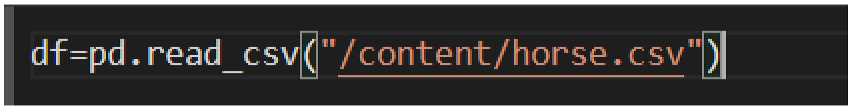
Code:

1. Importing required libraries:



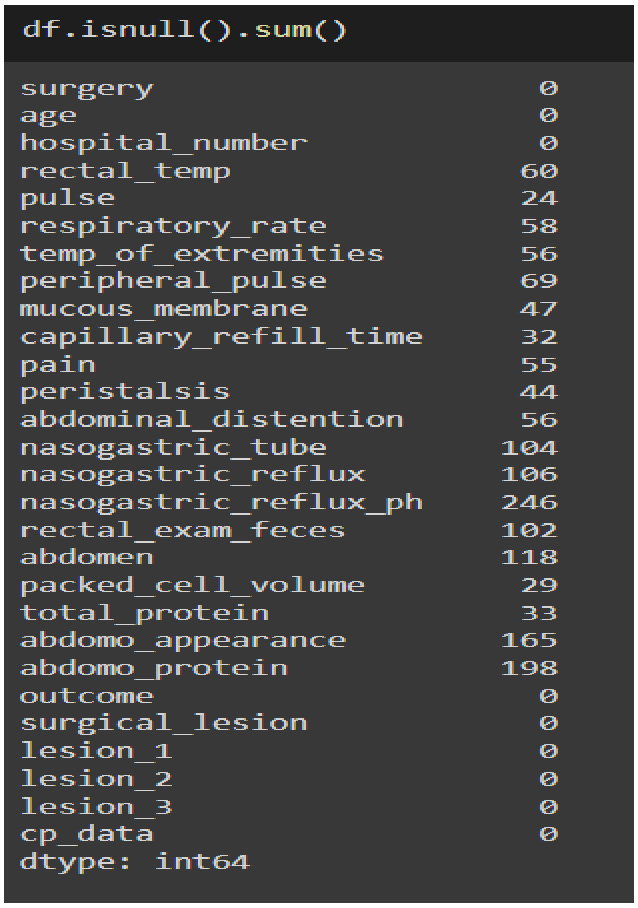
```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import io
import pandas as pd
from matplotlib.pyplot import figure as fig
```

2. Uploading and reading the dataset:



```
df=pd.read_csv("/content/horse.csv")
```

3. Checking the null values:



```
df.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
```

4. Copying the dataset in another variable df1:

```
df1 = df.copy()
```

5. Checking the data types of all the values:

```
df.dtypes
```

```
surgery                   object
age                       object
hospital_number            int64
rectal_temp              float64
pulse                    float64
respiratory_rate         float64
temp_of_extremities       object
peripheral_pulse          object
mucous_membrane           object
capillary_refill_time     object
pain                      object
peristalsis               object
abdominal_distention      object
nasogastric_tube          object
nasogastric_reflux        object
nasogastric_reflux_ph    float64
rectal_exam_feces         object
abdomen                   object
packed_cell_volume       float64
total_protein            float64
abdomo_appearance         object
abdomo_protein           float64
outcome                   object
surgical_lesion           object
lesion_1                   int64
lesion_2                   int64
lesion_3                   int64
cp_data                   object
dtype: object
```

6. Dropping the columns where null values are more than 100:

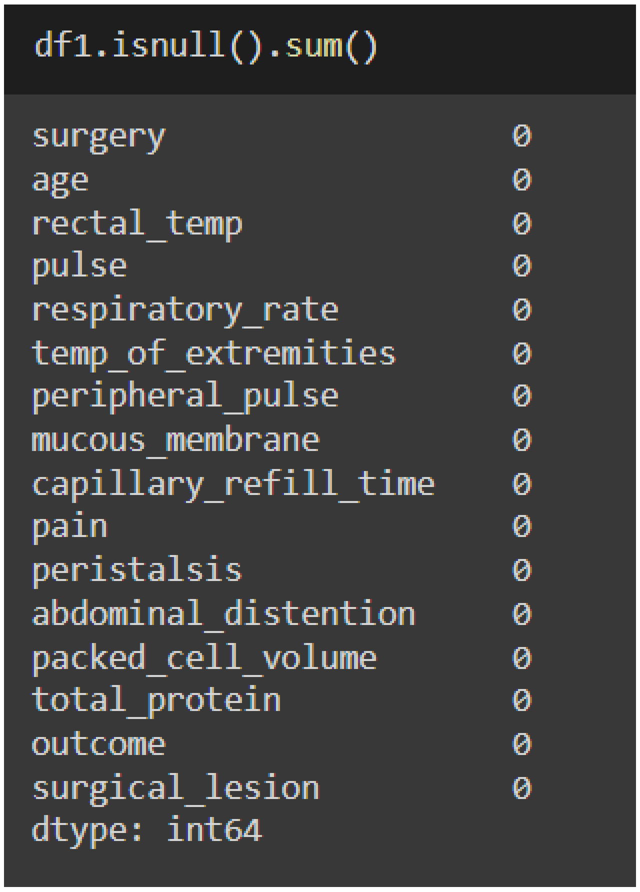
```
df1 = df1.drop(columns=["hospital_number", "nasogastric_tube",
"nasogastric_reflux", "nasogastric_reflux_ph", "rectal_exam_feces",
"abdomen", "abdomo_appearance", "abdomo_protein"])


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

7. To replace the null values we will need to replace it by average values in the respective columns replace with the median values is mostly preffered:

```
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)
```

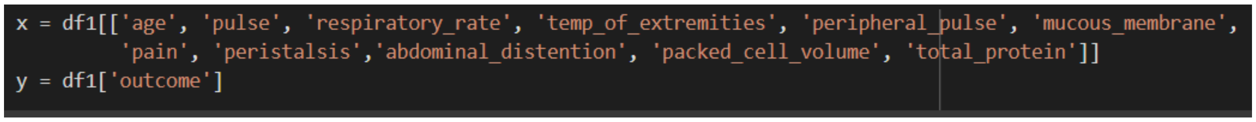
```python
df1['rectal_temp'] =
df1['rectal_temp'].fillna(df1['rectal_temp'].median())

df1['pulse'] = df1['pulse'].fillna(round(df1['pulse'].median()))

df1['respiratory_rate'] =
df1['respiratory_rate'].fillna(round(df1['respiratory_rate'].median(
)))

df1['temp_of_extremities'] =
df1['temp_of_extremities'].fillna(round(df1['temp_of_extremities'].m
edian()))

df1['peripheral_pulse'] =
df1['peripheral_pulse'].fillna(round(df1['peripheral_pulse'].median(
)))

df1['mucous_membrane'] =
df1['mucous_membrane'].fillna(round(df1['mucous_membrane'].median())
)

df1['capillary_refill_time'] =
df1['capillary_refill_time'].fillna(1)

df1['pain'] = df1['pain'].fillna(round(df1['pain'].median()))

df1['peristalsis'] =
df1['peristalsis'].fillna(round(df1['peristalsis'].median()))

df1['abdominal_distention'] =
df1['abdominal_distention'].fillna(round(df1['abdominal_distention']
.median()))

df1['packed_cell_volume'] =
df1['packed_cell_volume'].fillna(round(df1['packed_cell_volume'].med
ian()))

df1['total_protein'] =
df1['total_protein'].fillna(round(df1['total_protein'].median()))
```

8. Now again checking the null values:



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

surgery                  0
age                      0
rectal_temp              0
pulse                    0
respiratory_rate         0
temp_of_extremities      0
peripheral_pulse         0
mucous_membrane          0
capillary_refill_time    0
pain                     0
peristalsis              0
abdominal_distention     0
packed_cell_volume       0
total_protein            0
outcome                  0
surgical_lesion          0
dtype: int64
```

9. Defining the x and y values that is the input and output labels:

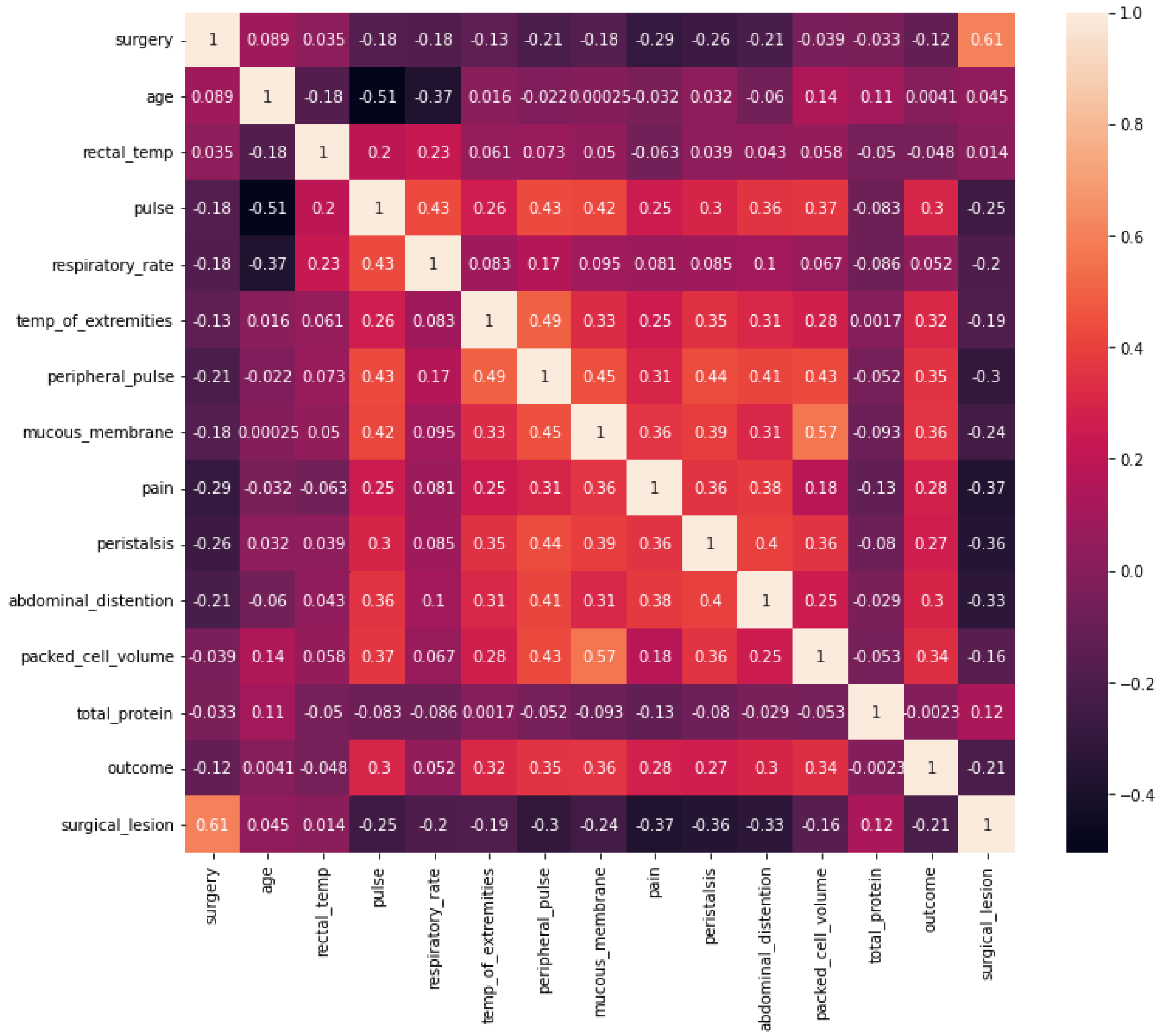


```
x = df1[['age', 'pulse', 'respiratory_rate', 'temp_of_extremities', 'peripheral_pulse', 'mucous_membrane',
        'pain', 'peristalsis','abdominal_distention', 'packed_cell_volume', 'total_protein']]
y = df1['outcome']
```

10. Performing logistic regression by splitting the dataset into 30% testing and 70% training:

```
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=0.3, random_state=0)

x_train.shape
```

```
df1.outcome.describe()

plt.figure(figsize=(12,10))

sns.heatmap(df1.corr(), annot=True)

plt.show()
```



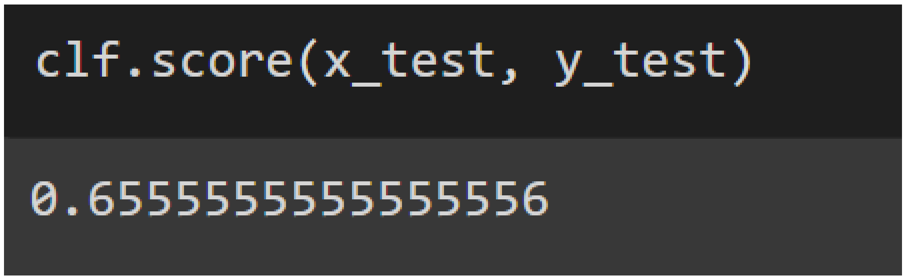
	surgery	age	rectal_temp	pulse	respiratory_rate	temp_of_extremities	peripheral_pulse	mucous_membrane	pain	peristalsis	abdominal_distention	packed_cell_volume	total_protein	outcome	surgical_lesion
surgery	1	0.089	0.035	-0.18	-0.18	-0.13	-0.21	-0.18	-0.29	-0.26	-0.21	-0.039	-0.033	-0.12	0.61
age	0.089	1	-0.18	-0.51	-0.37	0.016	-0.022	0.00025	-0.032	0.032	-0.06	0.14	0.11	0.0041	0.045
rectal_temp	0.035	-0.18	1	0.2	0.23	0.061	0.073	0.05	-0.063	0.039	0.043	0.058	-0.05	-0.048	0.014
pulse	-0.18	-0.51	0.2	1	0.43	0.26	0.43	0.42	0.25	0.3	0.36	0.37	-0.083	0.3	-0.25
respiratory_rate	-0.18	-0.37	0.23	0.43	1	0.083	0.17	0.095	0.081	0.085	0.1	0.067	-0.086	0.052	-0.2
temp_of_extremities	-0.13	0.016	0.061	0.26	0.083	1	0.49	0.33	0.25	0.35	0.31	0.28	0.0017	0.32	-0.19
peripheral_pulse	-0.21	-0.022	0.073	0.43	0.17	0.49	1	0.45	0.31	0.44	0.41	0.43	-0.052	0.35	-0.3
mucous_membrane	-0.18	0.00025	0.05	0.42	0.095	0.33	0.45	1	0.36	0.39	0.31	0.57	-0.093	0.36	-0.24
pain	-0.29	-0.032	-0.063	0.25	0.081	0.25	0.31	0.36	1	0.36	0.38	0.18	-0.13	0.28	-0.37
peristalsis	-0.26	0.032	0.039	0.3	0.085	0.35	0.44	0.39	0.36	1	0.4	0.36	-0.08	0.27	-0.36
abdominal_distention	-0.21	-0.06	0.043	0.36	0.1	0.31	0.41	0.31	0.38	0.4	1	0.25	-0.029	0.3	-0.33
packed_cell_volume	-0.039	0.14	0.058	0.37	0.067	0.28	0.43	0.57	0.18	0.36	0.25	1	-0.053	0.34	-0.16
total_protein	-0.033	0.11	-0.05	-0.083	-0.086	0.0017	-0.052	-0.093	-0.13	-0.08	-0.029	-0.053	1	-0.0023	0.12
outcome	-0.12	0.0041	-0.048	0.3	0.052	0.32	0.35	0.36	0.28	0.27	0.3	0.34	-0.0023	1	-0.21
surgical_lesion	0.61	0.045	0.014	-0.25	-0.2	-0.19	-0.3	-0.24	-0.37	-0.36	-0.33	-0.16	0.12	-0.21	1

11. Performing Logistic regression on the dataset:

```
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)
```

12. Checking out the final score or the accuracy:



```
clf.score(x_test, y_test)
```

```
0.6555555555555556
```

Conclusion:

- We got the final accuracy as 65.5 percent.
- We successfully performed the experiment and did the logistic regression.
- After splitting the dataset into test:30 and train:70 we got a bit higher accuracy as compared to test:20 and train:80.
- We removed the columns or so called features from the input label because they were having -ve correlation with the required outcome feature.
- Heatmap becomes quite helpful in determining the features that should be considered.