CSE343/CSE543/ECE363/ECE563: Machine Learning (PG)

Monsoon 2022
Assignment-1

Sehban Fazili
MTech CSE
MT21143

Q2. Linear Regression:

(d) Steps involved in implementing the Linear Regression model are as follows:

1. Importing Libraries:

   import numpy as np
   import pandas as pd
   import matplotlib.pyplot as plt
   import seaborn as sns

2. Reading the dataset:
   As the given data had no column names, the column names from the documentation were taken in the same order and added to the data.
   colnames = ['Frequency','Angle of attack','Chord length','Free-stream velocity','Suction side displacement thickness','Scaled sound pressure level']
   The data was read using the command:
   data = pd.read_csv('/home/sehbanfazili/Downloads/airfoil_self_noise.dat', sep = '\t',names=colnames,index_col=False)

3. Dataset Information & Statistics:

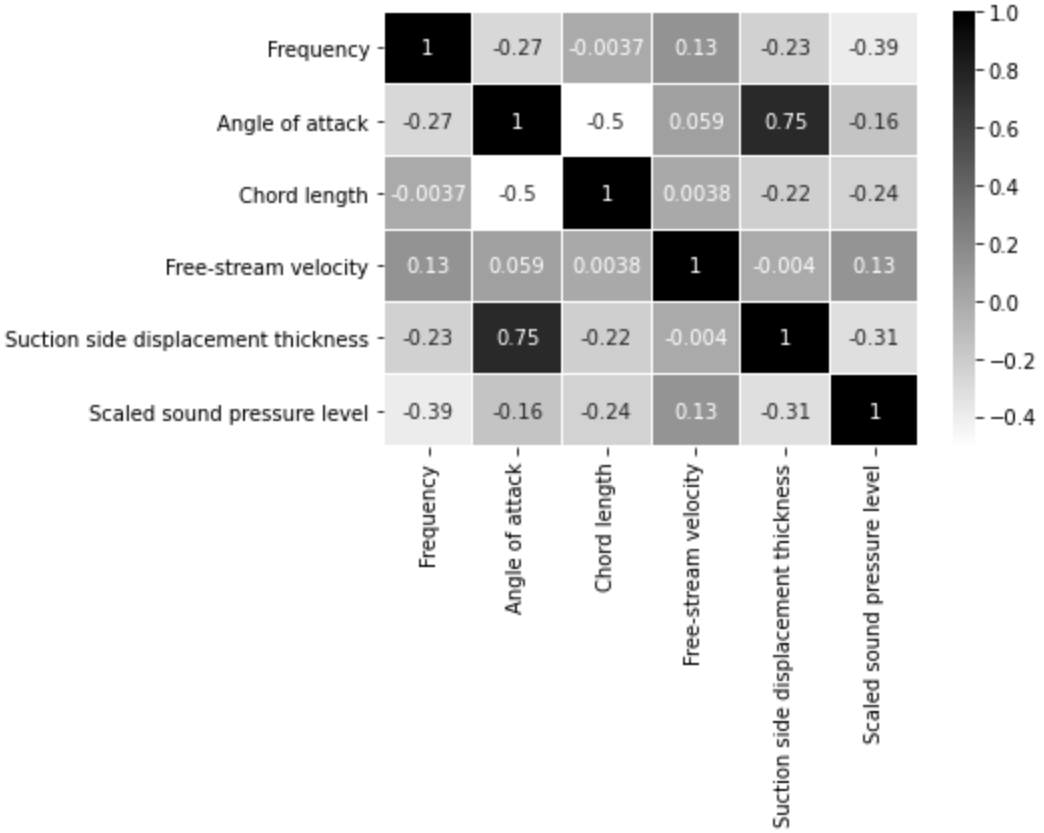
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1503 entries, 0 to 1502
Data columns (total 6 columns):
 #   Column                                Non-Null Count  Dtype
---  ------                                --------------  -----
 0   Frequency                             1503 non-null   int64
 1   Angle of attack                       1503 non-null   float64
 2   Chord length                          1503 non-null   float64
 3   Free-stream velocity                  1503 non-null   float64
 4   Suction side displacement thickness   1503 non-null   float64
 5   Scaled sound pressure level           1503 non-null   float64
dtypes: float64(5), int64(1)
memory usage: 70.6 KB
```

	Frequency	Angle of attack	Chord length	Free-stream velocity	Suction side displacement thickness	Scaled sound pressure level
count	1503.000000	1503.000000	1503.000000	1503.000000	1503.000000	1503.000000
mean	2886.380572	6.782302	0.136548	50.860745	0.011140	124.835943
std	3152.573137	5.918128	0.093541	15.572784	0.013150	6.898657
min	200.000000	0.000000	0.025400	31.700000	0.000401	103.380000
25%	800.000000	2.000000	0.050800	39.600000	0.002535	120.191000
50%	1600.000000	5.400000	0.101600	39.600000	0.004957	125.721000
75%	4000.000000	9.900000	0.228600	71.300000	0.015576	129.995500
max	20000.000000	22.200000	0.304800	71.300000	0.058411	140.987000

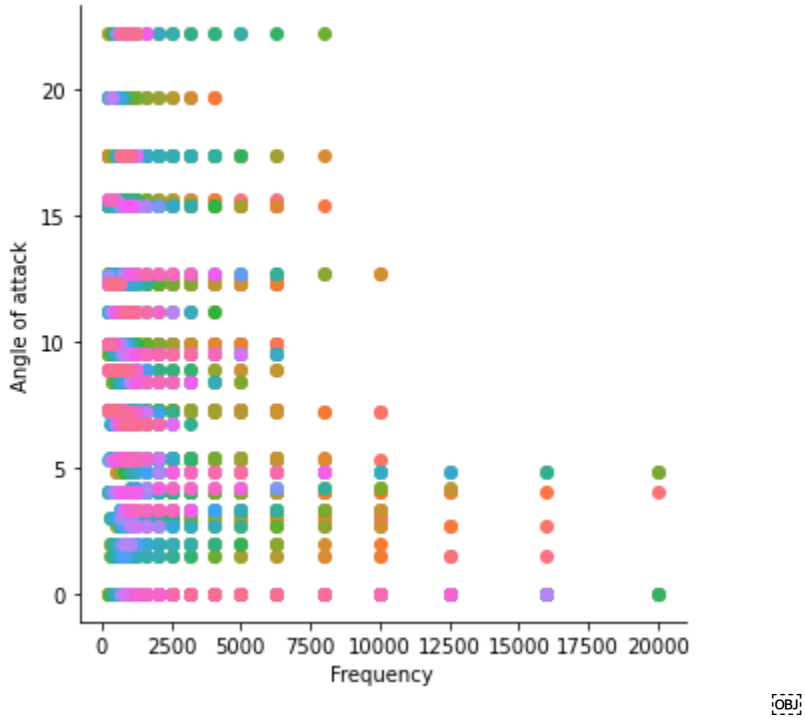
4. Data Preprocessing:
    - Null values:
      No null value was found in the dataset.

```
Frequency                                0
Angle of attack                          0
Chord length                             0
Free-stream velocity                     0
Suction side displacement thickness      0
Scaled sound pressure level              0
dtype: int64
```

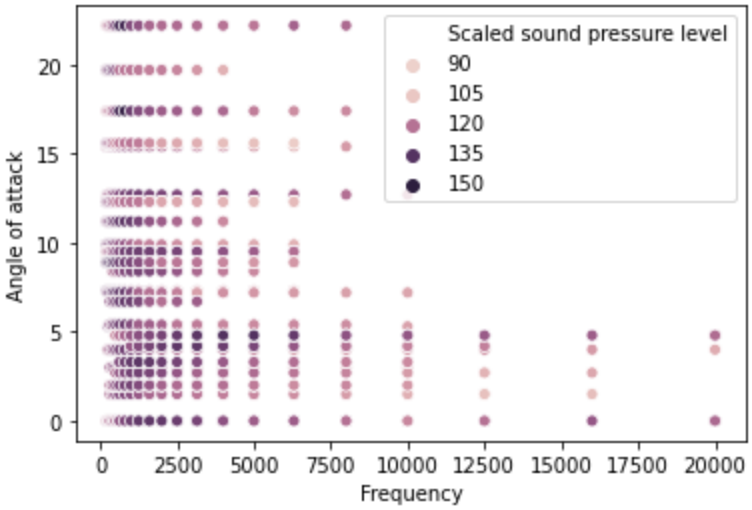
5. Data Visualization:
    - Heatmap:



- Facetgrid:



- Scatterplot:



- Pair plot:
  It can be seen in the ipynb file.

6. Splitting the data into test and train:
   - First, we take **y** as the label i.e., Scaled sound pressure level column, and **x** as the rest data.
   - Then we normalize the x data.
   - Then we split the data using the command:
     x_train, x_test, y_train, y_test = train_test_split(data_norm, y, test_size=0.2, random_state=42)
7. Fitting the Linear Regression model:
   - Import the model from sklearn using command:
     from sklearn.linear_model import LinearRegression
   - Fit the modal on train data usin command:
     reg = LinearRegression().fit(x_train, y_train)
8. Results:

MSE	RMSE	MAE
22.128643318247278	4.704109194974887	3.6724145641788013

	Train R2score	Test R2score
With scaling data	0.5034475371198581	0.5582979754897288
Without scaling data	0.5034475371198581	0.5582979754897284
Dropping a correlated column	0.4763134758997891	0.5126998147396657

9. Loss Function from scratch gave the same result as the value using sklearn library which is 22.128643318247278.

Q3. Classification/ Logistic Regression:

(a) Steps involved are as follows:

1. Import libraries:

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

2. Reading the dataset:
   Using command:

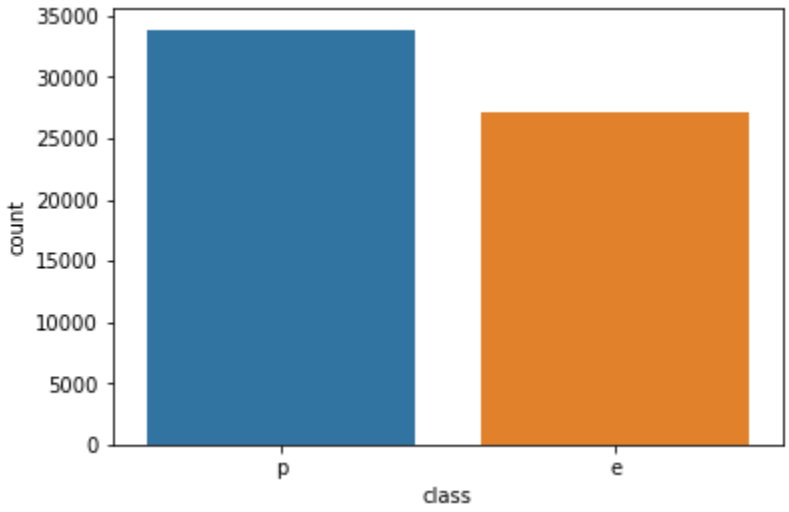
```
df = pd.read_csv('MushroomDataset/secondary_data.csv',sep=';')
```

3. Dataset information & Statistics:

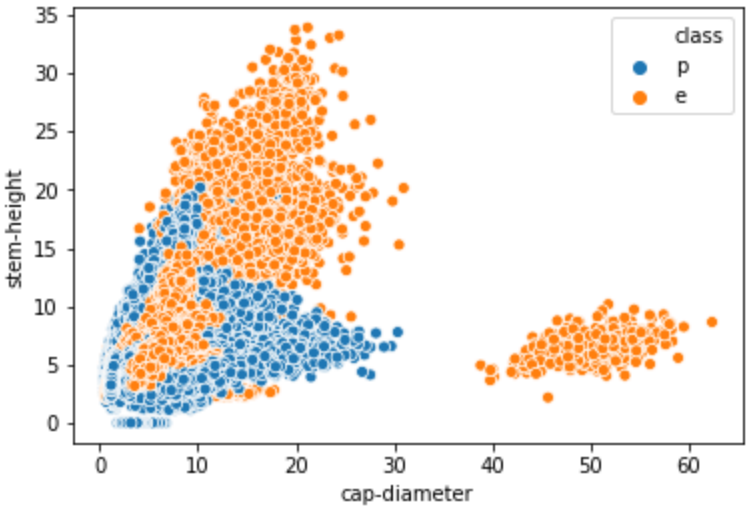
	cap-diameter	stem-height	stem-width
count	61069.000000	61069.000000	61069.000000
mean	6.733854	6.581538	12.149410
std	5.264845	3.370017	10.035955
min	0.380000	0.000000	0.000000
25%	3.480000	4.640000	5.210000
50%	5.860000	5.950000	10.190000
75%	8.540000	7.740000	16.570000
max	62.340000	33.920000	103.910000

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 61069 entries, 0 to 61068
Data columns (total 21 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   class                  61069 non-null  object
 1   cap-diameter           61069 non-null  float64
 2   cap-shape              61069 non-null  object
 3   cap-surface            46949 non-null  object
 4   cap-color              61069 non-null  object
 5   does-bruise-or-bleed   61069 non-null  object
 6   gill-attachment        51185 non-null  object
 7   gill-spacing           36006 non-null  object
 8   gill-color             61069 non-null  object
 9   stem-height            61069 non-null  float64
 10  stem-width             61069 non-null  float64
 11  stem-root              9531 non-null   object
 12  stem-surface           22945 non-null  object
 13  stem-color             61069 non-null  object
 14  veil-type              3177 non-null   object
 15  veil-color             7413 non-null   object
 16  has-ring               61069 non-null  object
 17  ring-type              58598 non-null  object
 18  spore-print-color      6354 non-null   object
 19  habitat                61069 non-null  object
 20  season                 61069 non-null  object
dtypes: float64(3), object(18)
memory usage: 9.8+ MB
```

4. Data Visualization:
    - Count plot:



    - Scatter plot:



5. Imputing Null values:

```
class                       0
cap-diameter                0
cap-shape                   0
cap-surface             14120
cap-color                   0
does-bruise-or-bleed        0
gill-attachment          9884
gill-spacing            25063
gill-color                  0
stem-height                 0
stem-width                  0
stem-root               51538
stem-surface            38124
stem-color                  0
veil-type               57892
veil-color              53656
has-ring                    0
ring-type                2471
spore-print-color       54715
habitat                     0
season                      0
dtype: int64
```

   Many null values were found. **Null values > 15000** columns were dropped and others were filled with there forward rows value using **ffill**.

6. Correlation heatmap and handlin of categorical variables usin dummy encodin can be seen in ipynb file.

(b) Steps involved in implementing the Logistic Regression model are as follows:

1. Data Preprocessing:
   - No null values were found.
   - Data was also in a normal range so no need to normalize.
2. Logistic Regression model

Command used:

```
clf = LogisticRegression(random_state=0)
clf.fit(x_train, y_train)
```

3. Results:

Accuracy	Precision	Recall	F1 Score
0.72	0.639	0.72	0.664