

```
In [ ]: !pip install pandas
!pip install -U scikit-learn matplotlib seaborn
```

```
In [ ]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import preprocessing
import seaborn as sns
```

```
In [ ]: dataset = pd.read_csv('Dataset_spine.csv')
```

```
In [ ]: dataset.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 310 entries, 0 to 309
Data columns (total 14 columns):
#   Column          Non-Null Count  Dtype
---  ---
0   Col1             310 non-null   float64
1   Col2             310 non-null   float64
2   Col3             310 non-null   float64
3   Col4             310 non-null   float64
4   Col5             310 non-null   float64
5   Col6             310 non-null   float64
6   Col7             310 non-null   float64
7   Col8             310 non-null   float64
8   Col9             310 non-null   float64
9   Col10            310 non-null   float64
10  Col11            310 non-null   float64
11  Col12            310 non-null   float64
12  Class_att        310 non-null   object
13  Unnamed: 13      14 non-null    object
dtypes: float64(12), object(2)
memory usage: 34.0+ KB
```

We have one unnamed column which has 14 non-null values

```
In [ ]: dataset.isna().sum()
```

```
Out[ ]: Col1             0
Col2             0
Col3             0
Col4             0
Col5             0
Col6             0
Col7             0
Col8             0
Col9             0
Col10            0
Col11            0
Col12            0
Class_att        0
Unnamed: 13      296
dtype: int64
```

```
In [ ]: dataset['Unnamed: 13'].info
```

```

Out[ ]: <bound method Series.info of 0
1
2      Prediction is done by using binary classificat...
3
4
...
305
306
307
308
309
Name: Unnamed: 13, Length: 310, dtype: object>

```

We will drop the Unnamed: 13 column

```
In [ ]: dataset = dataset.drop(['Unnamed: 13'], axis=1)
```

The columns name are not very informative, so using the information present at Kaggle , we will rename our columns

```
In [ ]: dataset.columns = [
    'pelvic_incidence',
    'pelvic_tilt',
    'lumbar_lordosis_angle',
    'sacral_slope',
    'pelvic_radius',
    'degree_spondylolisthesis',
    'pelvic_slop',
    'Direct_tilt',
    'thoracic_slope',
    'cervical_tilt',
    'sacrum_angle',
    'scoliosis_slope',
    'label'
]
```

```
In [ ]: dataset.describe
```

```
Out[ ]: <bound method NDFrame.describe of
_angle  sacral_slope  \
0      63.027817    22.552586      39.609117    40.475232
1      39.056951    10.060991    25.015378    28.995960
2      68.832021    22.218482    50.092194    46.613539
3      69.297008    24.652878    44.311238    44.644130
4      49.712859     9.652075    28.317406    40.060784
..      ...      ...      ...      ...
305     47.903565    13.616688    36.000000    34.286877
306     53.936748    20.721496    29.220534    33.215251
307     61.446597    22.694968    46.170347    38.751628
308     45.252792     8.693157    41.583126    36.559635
309     33.841641     5.073991    36.641233    28.767649

      pelvic_radius  degree_spondylolisthesis  pelvic_slop  Direct_tilt  \
0      98.672917      -0.254400      0.744503      12.5661
1     114.405425       4.564259      0.415186      12.8874
2     105.985135     -3.530317      0.474889      26.8343
3     101.868495     11.211523      0.369345      23.5603
4     108.168725      7.918501      0.543360      35.4940
..      ...      ...      ...      ...
305    117.449062     -4.245395      0.129744       7.8433
306    114.365845     -0.421010      0.047913      19.1986
307    125.670725     -2.707880      0.081070      16.2059
308    118.545842      0.214750      0.159251      14.7334
309    123.945244     -0.199249      0.674504      19.3825

      thoracic_slope  cervical_tilt  sacrum_angle  scoliosis_slope  label
0      14.5386      15.30468    -28.658501      43.5123  Abnormal
1      17.5323      16.78486    -25.530607      16.1102  Abnormal
2      17.4861      16.65897    -29.031888      19.2221  Abnormal
3      12.7074      11.42447    -30.470246      18.8329  Abnormal
4      15.9546       8.87237    -16.378376      24.9171  Abnormal
..      ...      ...      ...      ...      ...
305     14.7484       8.51707    -15.728927      11.5472  Normal
306     18.1972       7.08745     6.013843      43.8693  Normal
307     13.5565       8.89572     3.564463      18.4151  Normal
308     16.0928       9.75922     5.767308      33.7192  Normal
309     17.6963      13.72929     1.783007      40.6049  Normal
```

[310 rows x 13 columns]>

```
In [ ]: dataset.info()
```

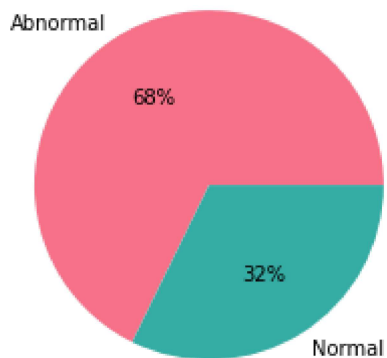
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 310 entries, 0 to 309
Data columns (total 13 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   pelvic_incidence                      310 non-null    float64
1   pelvic_tilt                           310 non-null    float64
2   lumbar_lordosis_angle                 310 non-null    float64
3   sacral_slope                          310 non-null    float64
4   pelvic_radius                         310 non-null    float64
5   degree_spondylolisthesis             310 non-null    float64
6   pelvic_slop                           310 non-null    float64
7   Direct_tilt                           310 non-null    float64
8   thoracic_slope                       310 non-null    float64
9   cervical_tilt                        310 non-null    float64
10  sacrum_angle                         310 non-null    float64
11  scoliosis_slope                      310 non-null    float64
12  label                                310 non-null    object
dtypes: float64(12), object(1)
memory usage: 31.6+ KB
```

```
In [ ]: dataset.describe(include='all')
```

```
Out[ ]:
```

	pelvic_incidence	pelvic_tilt	lumbar_lordosis_angle	sacral_slope	pelvic_radius	degree_spond
count	310.000000	310.000000	310.000000	310.000000	310.000000	
unique	NaN	NaN	NaN	NaN	NaN	
top	NaN	NaN	NaN	NaN	NaN	
freq	NaN	NaN	NaN	NaN	NaN	
mean	60.496653	17.542822	51.930930	42.953831	117.920655	
std	17.236520	10.008330	18.554064	13.423102	13.317377	
min	26.147921	-6.554948	14.000000	13.366931	70.082575	
25%	46.430294	10.667069	37.000000	33.347122	110.709196	
50%	58.691038	16.357689	49.562398	42.404912	118.268178	
75%	72.877696	22.120395	63.000000	52.695888	125.467674	
max	129.834041	49.431864	125.742385	121.429566	163.071041	

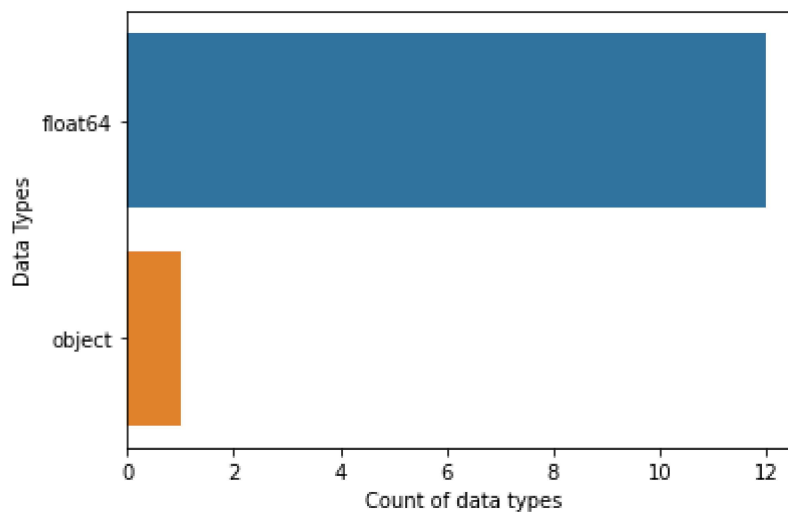
```
In [ ]: plt.pie(list(dataset.label.value_counts().to_dict().values()),
               labels=list(dataset.label.value_counts().to_dict().keys()),
               colors=sns.color_palette('husl',2),
               autopct='%.0f%%')
plt.show()
```



Our dataset it's unbalanced. We expect that this may affect our model perform

Let's verify the data types present in our dataset. In order to run our MLP model we need to have only float numbers

```
In [ ]: sns.countplot(y=dataset.dtypes, data=dataset)
plt.xlabel("Count of data types")
plt.ylabel("Data Types")
plt.show()
```



And now we will replace the object values by float values

```
In [ ]: dataset['label'].head
```

```
Out[ ]: <bound method NDFrame.head of 0      Abnormal
1      Abnormal
2      Abnormal
3      Abnormal
4      Abnormal
...
305     Normal
306     Normal
307     Normal
308     Normal
309     Normal
Name: label, Length: 310, dtype: object>
```

```
In [ ]: dataset['label_val'] = preprocessing.LabelEncoder().fit_transform(
        dataset['label']
    )
```

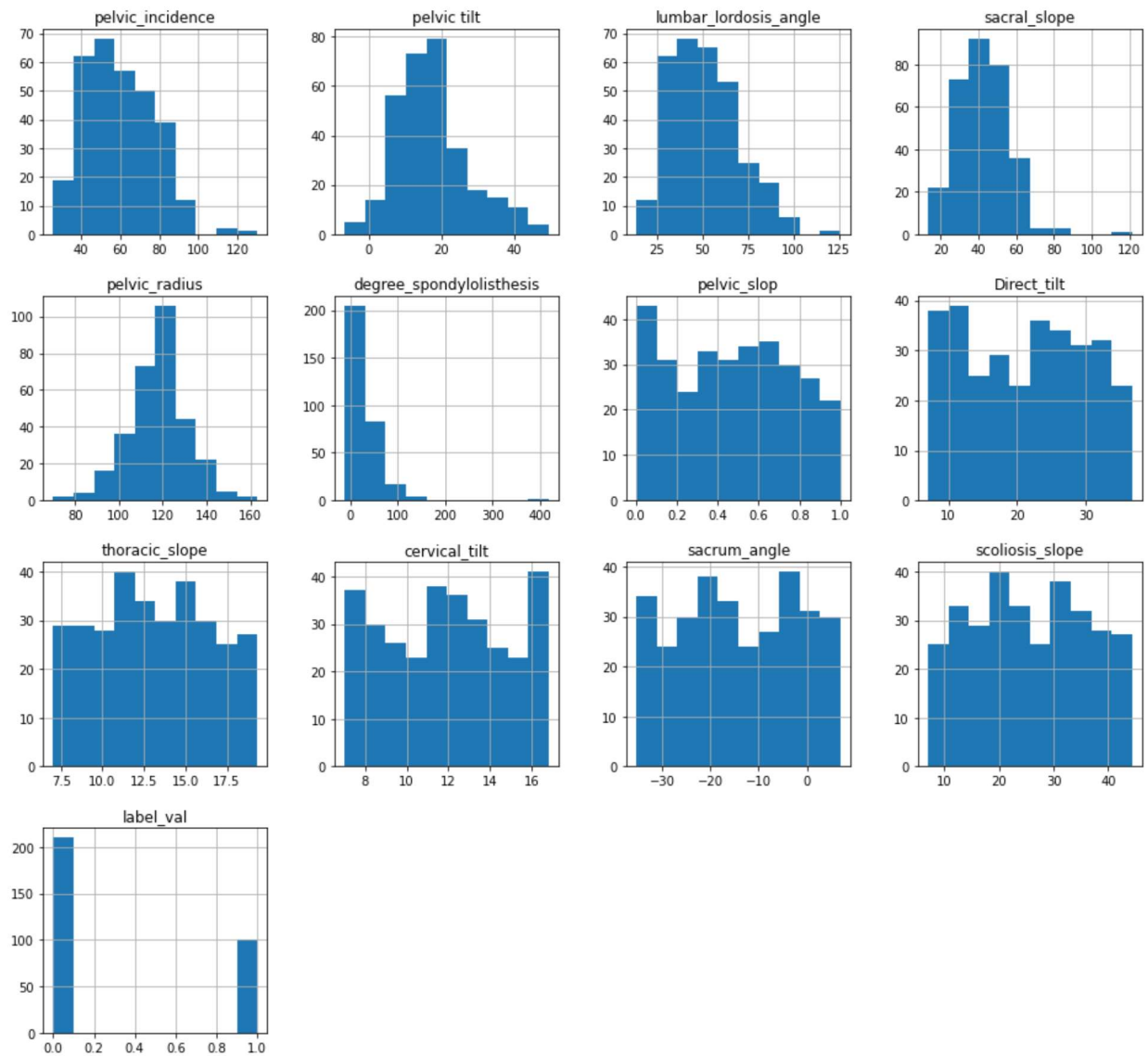
```
In [ ]: dataset['label_val'].head
```

```
Out[ ]: <bound method NDFrame.head of 0      0
1      0
2      0
3      0
4      0
..
305    1
306    1
307    1
308    1
309    1
Name: label_val, Length: 310, dtype: int32>
```

So in the `label_val` column we have 1 for normal and 0 for abnormal

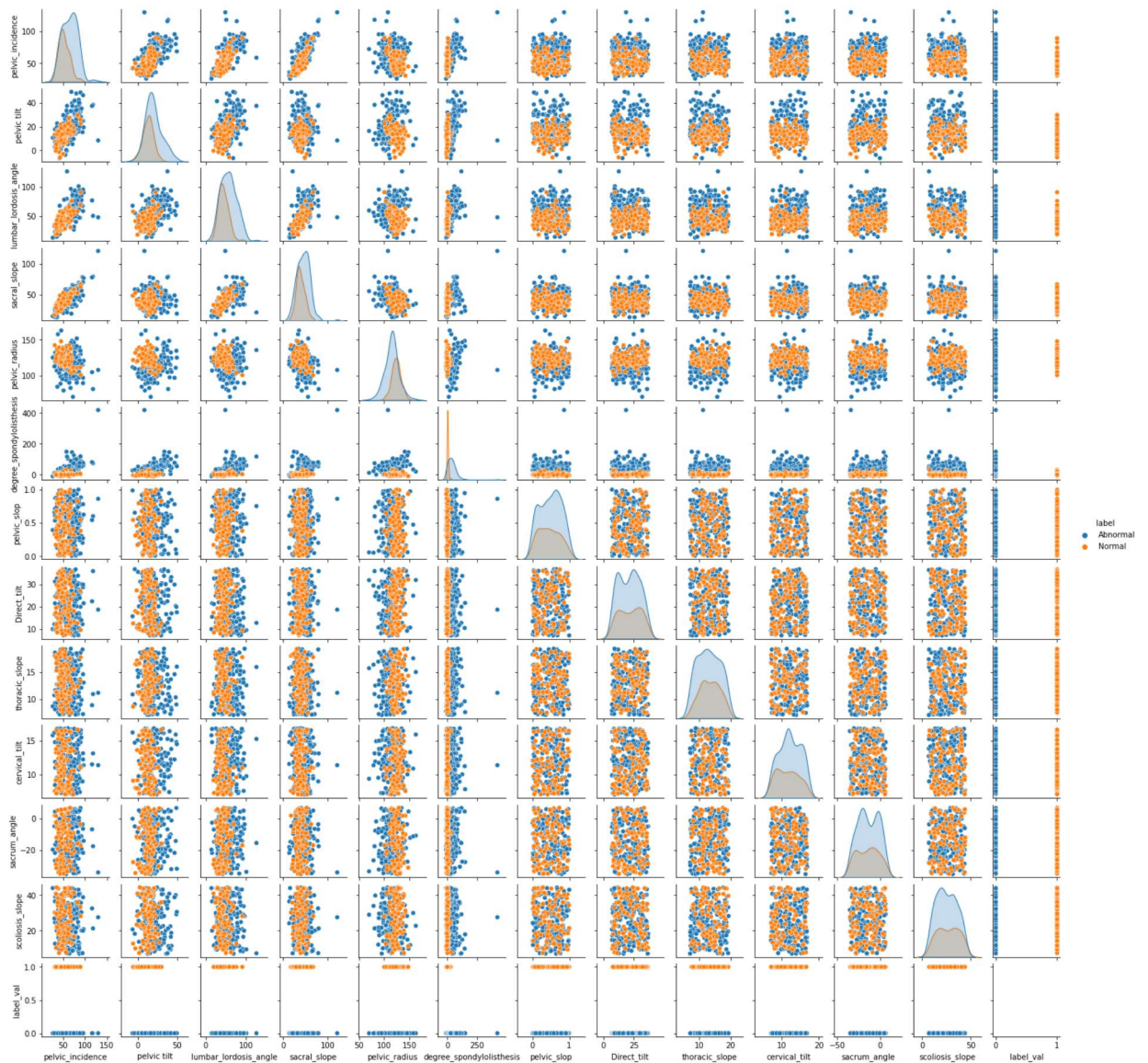
Now we will check the distribution of the features

```
In [ ]: dataset.hist(figsize=(16,15))
plt.title("Features distribution")
plt.show()
```



```
In [ ]: sns.pairplot(dataset,height=1.5, hue='label')
```

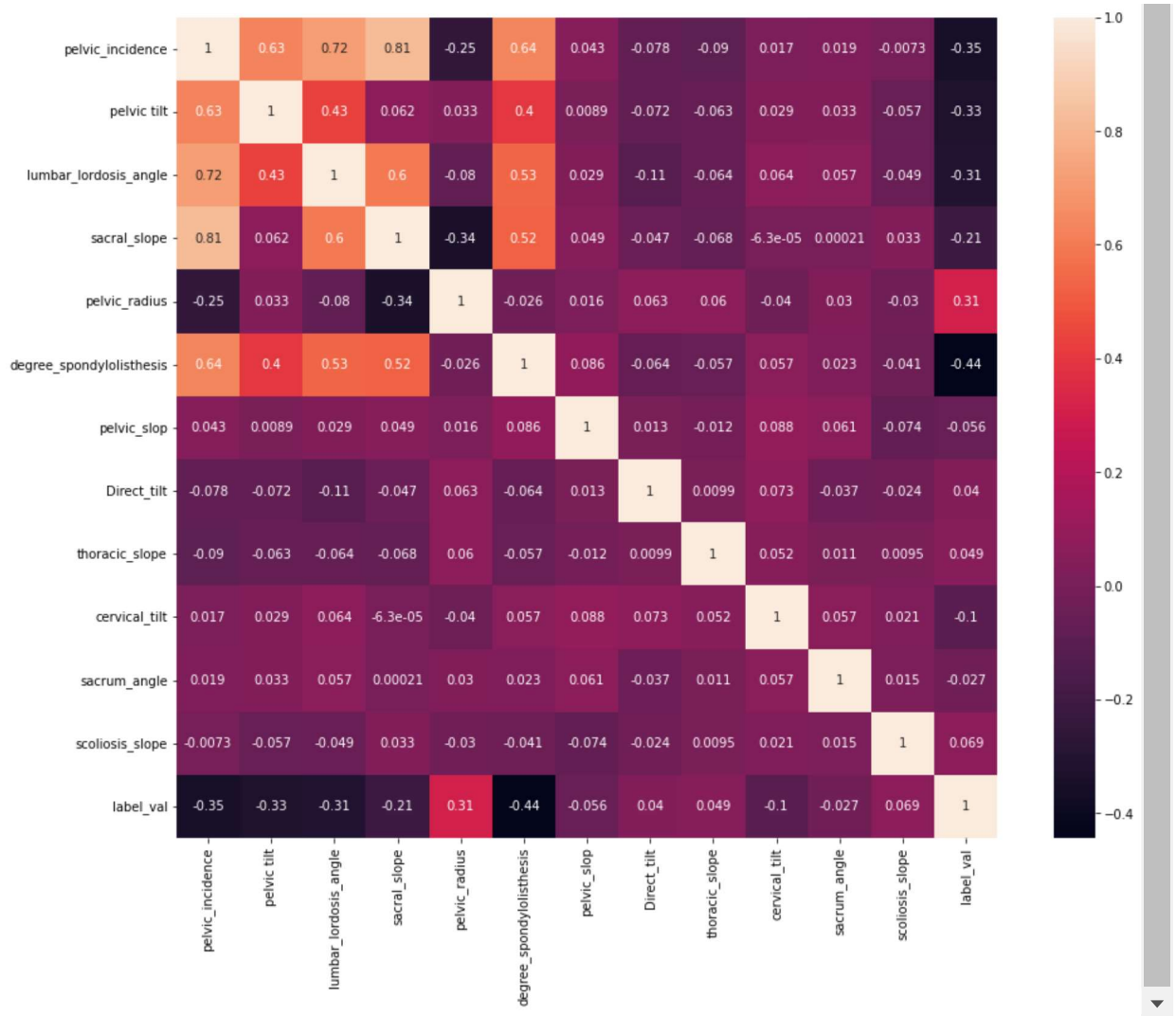
```
Out[ ]: <seaborn.axisgrid.PairGrid at 0x193c5066310>
```



Now we can verify the relation between features

```
In [ ]: plt.subplots(figsize=(16,12))
sns.heatmap(
    dataset.corr(),
    annot=True,
    square=True,
    cbar=True
)
```

```
Out[ ]: <AxesSubplot:>
```

```
In [ ]: dataset.isna().sum()
```

```
Out[ ]: pelvic_incidence      0
pelvic_tilt                  0
lumbar_lordosis_angle       0
sacral_slope                 0
pelvic_radius                0
degree_spondylolisthesis    0
pelvic_slop                  0
Direct_tilt                  0
thoracic_slope               0
cervical_tilt                0
sacrum_angle                 0
scoliosis_slope              0
label                        0
label_val                    0
dtype: int64
```

Now we can export our clean dataset

```
In [ ]: dataset.to_csv('dataset_spine_clean.csv')
```

```
In [ ]:
```