Name: Heramb Pawar

Roll No: 67

Prac: 7

Time

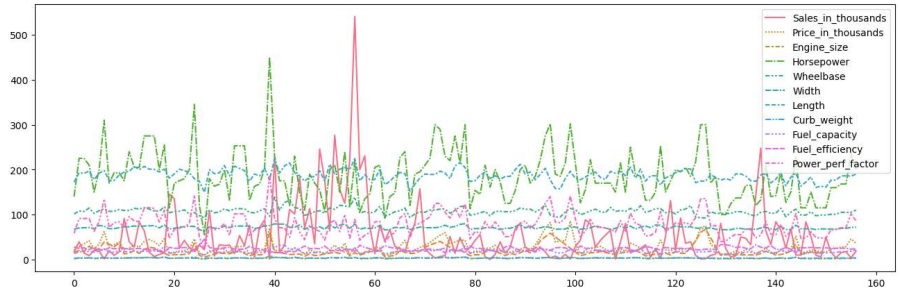
Line Chart

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


sales = "Car_sales.csv"
sales_data = pd.read_csv(sales,encoding='ISO-8859-1')


plt.figure(figsize=(16,5))
sns.lineplot(data=sales_data)
```

```
<Axes: >
```



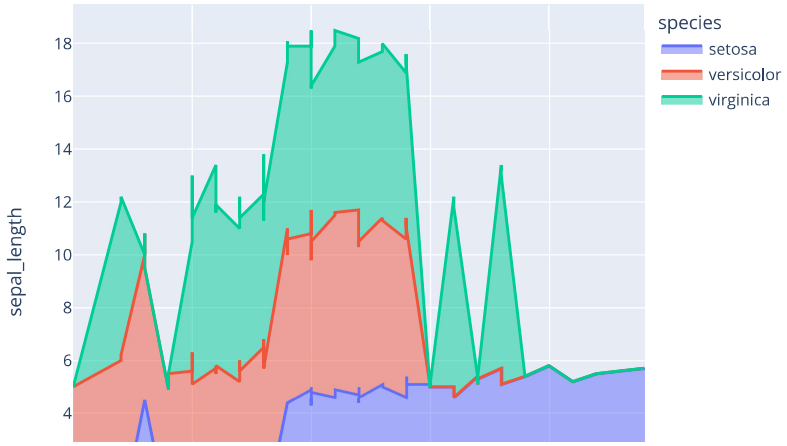
Stacked Area Chart

```
import plotly.express as px

df = px.data.iris()

fig = px.area(df, x="sepal_width",y="sepal_length",
              color="species",
              hover_data=['petal_width'],)

fig.show()
```



Relationship

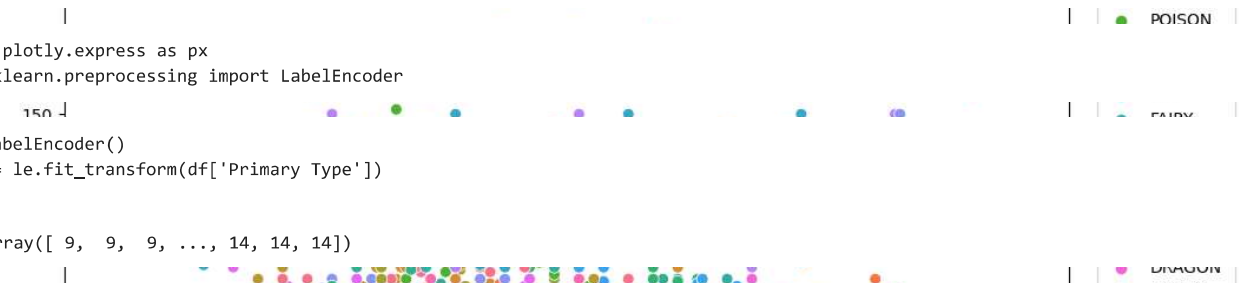
Scatter Plot

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


file="dataset.csv"
data = pd.read_csv(file,encoding='ISO-8859-1')


plt.figure(figsize=(11,7))
sns.scatterplot(x = 'Attack', y = 'Defense', s = 50, hue = 'Primary Type', data=data)
plt.legend(bbox_to_anchor=(1.02, 1)) # move legend to outside of the chart
plt.title('Defense vs Attack for All Pokemons', fontsize=16)
plt.xlabel('Attack', fontsize=12)
plt.ylabel('Defense', fontsize=12)
plt.show()
```

Parallel Coordinates Plot




```
import plotly.express as px
from sklearn.preprocessing import LabelEncoder
```

```
le = LabelEncoder()
label = le.fit_transform(df['Primary Type'])
label
```

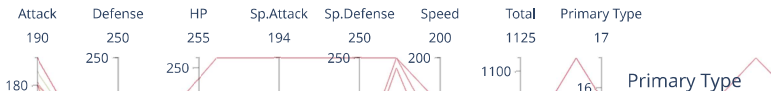
```
array([ 9,  9,  9, ..., 14, 14, 14])
```

```
df.drop("Purchased", axis=1, inplace=True)
df["Primary Type"] = label
df
```

	Name	Name2	Secondary type	Attack	Defense	HP	Sp.Attack	Sp.Defense	Speed	Total	Primary Type
0	Bulbasaur	NaN	POISON	49	49	45	65	65	45	318	9
1	Ivysaur	NaN	POISON	62	63	60	80	80	60	405	9
2	Venusaur	NaN	POISON	82	83	80	100	100	80	525	9
3	Venusaur	Mega Venusaur	POISON	100	123	80	122	120	80	625	9
4	Charmander	NaN	NaN	52	43	39	60	50	65	309	6
...	...	...	...	...	...	...	...	...	...	...	...
1040	Glastrier	NaN	NaN	145	130	100	65	110	30	580	11
1041	Spectrier	NaN	NaN	65	60	100	145	80	130	580	8
1042	Calyrex	NaN	GRASS	80	80	100	80	80	80	500	14
1043	Calyrex	Ice Rider	ICE	165	150	100	85	130	50	680	14
1044	Calyrex	Shadow Rider	GHOST	85	80	100	165	100	150	680	14

1045 rows × 11 columns

```
fig = px.parallel_coordinates(df, color="Primary Type" ,labels={"Attack": "Attack",
                "Defense": "Defense", "Health": "HP",
                "Special Attack": "Sp.Attack", "Speed": "Speed", },
                            color_continuous_scale=px.colors.diverging.Tealrose,
                            color_continuous_midpoint=2)
fig.show()
```



Part to whole

Tree Map

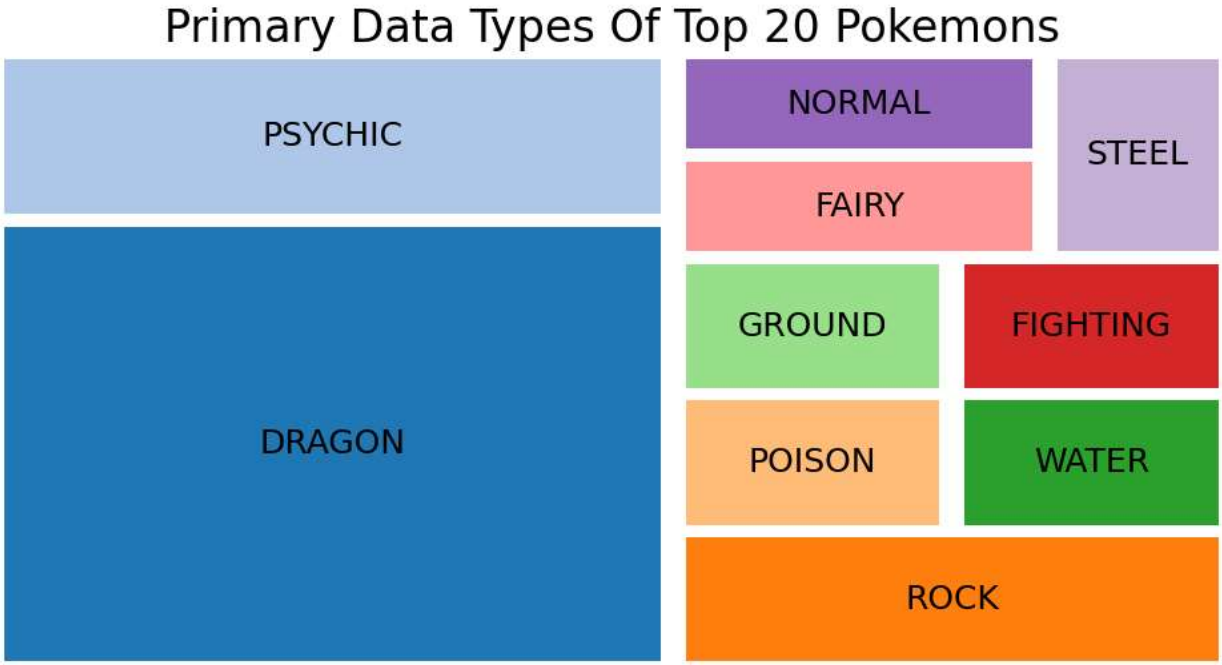
```
pip install squarify
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting squarify
  Downloading squarify-0.4.3-py3-none-any.whl (4.3 kB)
Installing collected packages: squarify
Successfully installed squarify-0.4.3
```

```
file="dataset.csv"
```

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

```
Text(0.5, 1.0, 'Primary Data Types Of Top 20 Pokemons')
```



Pie Chart

```
dataset = pd.read_csv(file,encoding='ISO-8859-1')
df = pd.DataFrame(dataset)
```

```
top20_pokemon = df.loc[:, ["Name", "Total",
                           'Primary Type']].sort_values(
    by="Total", ascending=False)[:20]
```

```
plt.figure(figsize=(12, 6))

plt.axis("off")
axis = squarify.plot(top20_pokemon['Primary Type'].value_counts(),
                     label=top20_pokemon['Primary Type'].value_counts().index,
                     color=sb.color_palette("tab20", len(
                         top20_pokemon['Primary Type'].value_counts())),
                     pad=1,
                     text_kwargs={'fontsize': 18})
axis.set_title("Primary Data Types Of Top 20 Pokemons", fontsize=24)
```



```
df = pd.DataFrame(dataset)
count=df['Primary Type'].value_counts()
print(count)
```

```
WATER       134
NORMAL      115
GRASS        91
BUG          81
PSYCHIC      79
FIRE         65
ELECTRIC     62
ROCK         60
DARK         46
GHOST        42
FIGHTING     42
POISON       41
GROUND       41
DRAGON       41
ICE          39
STEEL        36
FAIRY        22
FLYING        8
Name: Primary Type, dtype: int64
```

```
print(type(count))
count1=count.to_dict()
print(count1)
```

```
<class 'pandas.core.series.Series'>
{'WATER': 134, 'NORMAL': 115, 'GRASS': 91, 'BUG': 81, 'PSYCHIC': 79, 'FIRE': 65, 'ELECTRIC': 62, 'ROCK': 60, 'DARK': 46, 'GHOST': 42, 'F
```

```
fig = plt.figure(figsize =(10, 7))
plt.pie(count1.values(),labels=count1.keys())

plt.show()
```

GRASS

NORMAL

Comparing Category

Heat Map

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

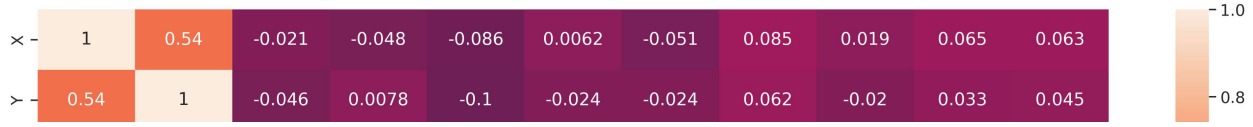
```
df=pd.read_csv('forestfires.csv')
df.head(10)
```

	X	Y	month	day	FFMC	DMC	DC	ISI	temp	RH	wind	rain	area
0	7	5	mar	fri	86.2	26.2	94.3	5.1	8.2	51	6.7	0.0	0.0
1	7	4	oct	tue	90.6	35.4	669.1	6.7	18.0	33	0.9	0.0	0.0
2	7	4	oct	sat	90.6	43.7	686.9	6.7	14.6	33	1.3	0.0	0.0
3	8	6	mar	fri	91.7	33.3	77.5	9.0	8.3	97	4.0	0.2	0.0
4	8	6	mar	sun	89.3	51.3	102.2	9.6	11.4	99	1.8	0.0	0.0
5	8	6	aug	sun	92.3	85.3	488.0	14.7	22.2	29	5.4	0.0	0.0
6	8	6	aug	mon	92.3	88.9	495.6	8.5	24.1	27	3.1	0.0	0.0
7	8	6	aug	mon	91.5	145.4	608.2	10.7	8.0	86	2.2	0.0	0.0
8	8	6	sep	tue	91.0	129.5	692.6	7.0	13.1	63	5.4	0.0	0.0
9	7	5	sep	sat	92.5	88.0	698.6	7.1	22.8	40	4.0	0.0	0.0

```
data=df.corr()
```

```
plt.figure(figsize=(8,9),dpi=300)
sns.heatmap(data,annot=True)
plt.plot()
```

[]



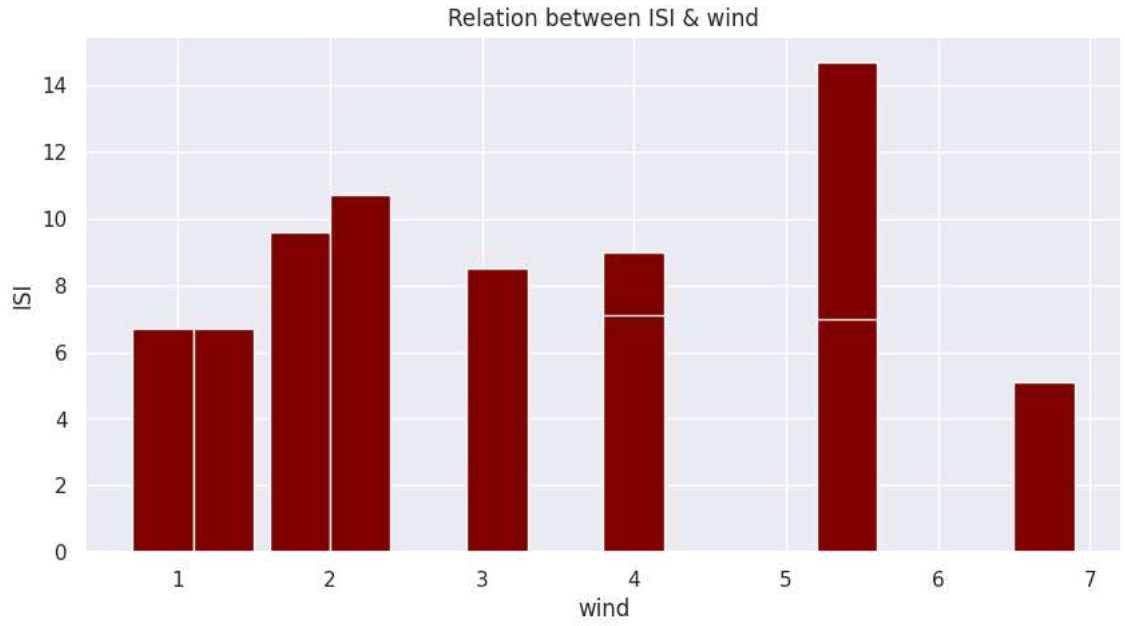
Bar Charts


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

df=pd.read_csv('forestfires.csv',nrows=10)
df.head(10)

fig = plt.figure(figsize = (10, 5))
plt.bar(df['wind'],df['ISI'], color ='maroon',
        width = 0.4)

plt.xlabel("wind")
plt.ylabel("ISI")
plt.title("Relation between ISI & wind")
plt.show()
```



Qualitative

Word Cloud

```
import pandas as pd
import matplotlib.pyplot as plt
from wordcloud import WordCloud


df = pd.read_csv("android-games.csv")
df.head()

text2 = " ".join(title for title in df.title)
word_cloud2 = WordCloud(collocations = False, background_color = 'white').generate(text2)

plt.imshow(word_cloud2, interpolation='bilinear')
plt.axis("off")
plt.show()
```