Lab 01

Data Exploration & Visualization

Pandas ⇒ Most popular python library for data analysis & manipulation
Matplotlib ⇒ graphical representation of data using matlab
Seaborn ⇒ used to create attractive charts

---

1. Import libraries

2. Read data
   * pd.read_csv( ) ⇒ X
     (pd ↓ Pandas)
   * get 1st n rows X.head(n)
   * get shape of data X.shape
   * get information on data (type, number, non null, Count, etc)
     X.info()
   * get some Statistics of data (count, mean, std, min, 25%., 50%., 75%., max)
     X.describe() (of numerical Columns)
   * to include non numerical data
     X.describe(include = ['object'])
   * check the frequency of each value in a column
     X['Pclass'].value_counts()
   * to check unique values of each variable
     X.nunique()
   * to check how many null values
     X.isnull().sum()

3. Analyse the variables of data (individual variable analysis - Univariate Analysis)
   For Continuous Variables
   * sns.distplot(X["Age"])
     (sns ↓ Seaborn, distplot ↓ distribution plot)
   * to plot more than one data
     sns.FacetGrid(X, hue = "sex").map(sns.distplot, "Age").addlegend()
     (hue ↓ to plot different subset of data with different color)

* use boxplot to identify skewness & outliers

```
Sns.boxplot(x='Age', data=X)
plt.title('Boxplot of Age')
plt.show()
```

Categorical Variables

* to count the Pclass (Bar plot)

```
Sns.countplot('Pclass, data=X)
plt.title('Bar plot for Pclass')
Plt.Show()
```

* in each class how many people survived

```
Survived = (X [(X['Age'].notnull()) & (X['Survived']==1)])
Class_Series = Survived.groupby(['pclass'])['Survived'].Sum()
```

* get the count of survivors vs Pclass

```
bar_chart = plt.bar(class_series.index, class_series)
```

* if we want to show more than one observation in a plot number of people in Pclass & how many of them are survived (stacked bar plot)

Bivariate Analysis ⟹ Find the relationship between two variables (Scatter plot)

* load a dataset in Seaborn

```
O=Sns.load_dataset('name')
```

* use Scatterplot to see relation between two variables

```
sns.Scatterplot(X=V1 , Y=V2 , data=O)
plt.show()
```

* add a regression line to see best fit of the data

```
Sns.regplot(X=V1, Y=V2, data=O)
plt.show()
```

* We can also add the gender to see its effect

```
Sns.lmplot(X=V1, Y=V2, data=O, hue='sex', palette='set1')
plt.Show()
```

MultiVariate Analysis ⟹ relationship between more than two variables

* `Sns.pairplot(O)`

be able to deal with the provided data, the 1st step is to explore the data and visualize it.

Exploratory Data Analysis

1. Read Data

* Use Pandas Library
* Read data
  X = pd.read_csv('directory of csv file')
* show group of the data
  X.head(no. in rows)
* get size of the data
  X.shape = (rows, columns)
* check the data information
  X.info()
* get some statistics of data
  X.describe()
  (get the mean, std, min, 25%, 50%, 75%, max)
* get statistics that are not numerical
  X.describe(include=['object'])
* get the frequency of a certain variable
  X['variable'].value_counts()
* how many unique values
  X.nunique()
* how many null values
  X.isnull().sum()

2. Visualize Data

* use MatPlotLib Library
* use Seaborn Library

a. Univariate Analysis

Continuous Variables

* plot the histogram of values

sns.distplot(X['var'])

* plot the value with two categories

sns.FacetGrid(X, hue="var1").map(sns.distplot, "Var 2").add_legend()

multiplot with conditional relationship

3rd dimension sub

* plot boxplot

Sns.boxplot(x="Var", data=X)

We have to take into consideration min, max, Q_1, Q_2, Q_3 → 25% 50% 75%

Categorical Variables

* plot the Categories of a value

sns.countplot(data=X, 'var')

* get number of survivals in each class

.groupby(...)

* plot bar plot

plt.bar(index, data)

* add another category to the bar chart

ax.bar(---, alpha=0.5, color='g')

color density

.groupby(---).count()
.sort_values(ascending=False)[:10]
last 10 years

Sns.lineplot(x, y)
Plot lines

b. Bivariate Analysis

* load dataset

sns.load_dataset('tips')

* plot scatterplot

sns.scatterplot(x='var1', y='var2', data=tips)

* best fit of the scatter plot using linear regression

sns.regplot(x='var1', y='var2', data=tips)

* add another category to the scatter plot

sns.lmplot(x='var1', y='var2', data=tips)

c. Multivarite Analysis

sns.pairplot(tips)
Plot pairwise relationships in a dataset

Lab02:

Visualization cont. using Numpy which can be used to perform wide variety mathematical operations on arrays & matrices

Distribution

(frequency distribution, probability distribution, Spread of data with respect to central values (mean, median))

[1]- Histogram

(frequency of occurance of a certain variable)

```
ax.hist(X['var'], width=no., bins=no.)
```

[2]- Kernel Density Plot

(used for probability distribution gives probability of occurance of the given data points

```
Sns.kdeplot(X['var'])
```

[1,5,5,5,8,8,8,10,12,12]

Q_1 8 Q_3

IQR = 10 - 5 = 5

Min = 1

Max = 12

[3]- Boxplot

(how variable is spread with respect to central tendancy)

```
Sns.boxplot(X['var'])
```

→ Median ⇒ Middle value

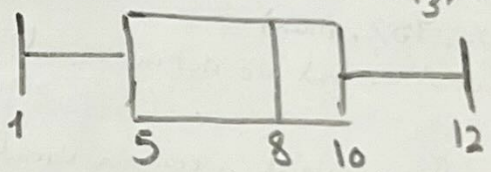
→ Q_1 ⇒ 25%.

→ Q_3 ⇒ 75%.

→ IQR = $Q_3 - Q_1$

→ Min ⇒ $Q_1 - 1.5$ IQR

→ Max ⇒ $Q_3 + 1.5$ IQR



Relationships

(get relationship between more than one variable)

[1]- Scatter plot

(how much one variable is affected by another)

```
ax.scatter(index, count)
Sns.regplot(index, count)
```

[2] Joint Plot

(Combination of Scatter plot with density plot (histogram) for both features)

```
Sns.jointplot(x='var1', y='var2', data=X, kind='reg')
```

[3]- Pair Plot

(show the relationship between variable & every other variable)

```
Sns.pairplot(X)
```

[4] - Heatmap

(a way to see the correlation between variables)

```
Sns.heatmap(X.corr())
```

Comparison

[1]- Line Chart

(commonly used in time series analysis where the temporal evolution of two sets)

```
Plt.plot(index, X['var1'], label=, marker=, markeredgecolor= )
```

// // (another plot of different variable) [2]

or use `Sns.lineplot(data=X)` to plot all data

[2]- Bar chart

(Simplest and old to visualize data)

```
Sns.barplot(x='var1', y='var2', data=X)
```

Composition

(to show the composition of one or more variables in absolute & relative terms (%))

1. Stacked Bar chart

```
ax.bar(index, data)
ax.bar(index, data2)
```