Contents

[Data Visualisation 2](#_Toc176337462)

[Plot Types 2](#_Toc176337463)

[*Bar Plots* 2](#_Toc176337464)

[Histogram Plots 3](#_Toc176337465)

[Boxplots 5](#_Toc176337466)

[SCATTER PLOTS 6](#_Toc176337467)

[LMPLOT(regplot) 7](#_Toc176337468)

[*Pairplot* 8](#_Toc176337469)

[HeatMaps 10](#_Toc176337470)

[Line Plots: 11](#_Toc176337471)

[Artificial Neural Networks 12](#_Toc176337472)

[Regularization 12](#_Toc176337473)

[Bias & Variance 12](#_Toc176337474)

[Dropout 13](#_Toc176337475)

[Noise Addition 13](#_Toc176337476)

[Overfitting & Underfitting 13](#_Toc176337477)

[Activation Funcitons 14](#_Toc176337478)

# Data Visualisation

*Ordinal categoric variables should be defined.*

*from pandas.api.types import CategoricalDtype*

*cut\_rank = [] -> rank your variables*

*dataCat["cut"] = dataCat["cut"].astype(CategoricalDtype(ordered= True, categories= cut\_rank))*

## Plot Types

### *Bar Plots*

*Bar plots are usually used to visualise categorycal variables.*

*dataCat is a dataframe, “cut” is a column that have 5 different values*;

cut\_frequencies = dataCat.cut.value\_counts().reset\_index()

sbn.barplot(x= xx["cut"], y=xx["count"])

output:

A graph of blue bars

Description automatically generated with medium confidence

Catplot:

sbn.catplot(x="cut", y="price", data= data)

A graph of blue lines

Description automatically generated

Hue:

sbn.barplot(x= "cut", y="price", data= data, hue= "color")

A graph of different colored lines

Description automatically generated

### Histogram Plots

Histogram plots are used to visualise continuous numerical variables. Shows distributions of variables.

|  |  |
| --- | --- |
| Histogram-> sns.histplot | Density Plot-> sns.pkeplot |
| A graph of a number of numbers  Description automatically generated |  |

|  |  |
| --- | --- |
|  | (sbn.FacetGrid(df, hue= "cut", height= 5, xlim= (0,10000))).map(sbn.kdeplot, "price", shade= True).add\_legend() |

|  |  |
| --- | --- |
| import seaborn as sns  import matplotlib.pyplot as plt  # Örnek veri seti  tips = sns.load\_dataset("tips")  # FacetGrid oluşturma ve histogram çizme  g = sns.FacetGrid(tips, row= "sex" ,hue= "smoker")  g.map(sns.kdeplot, "total\_bill", shade= True).add\_legend()  plt.show() |  |

### Boxplots

|  |  |
| --- | --- |
| x= sbn.boxplot(x= df.day, y= df.total\_bill, data=df) |  |
| sbn.boxplot(x = df["total\_bill"],color= "#C14953" , orient="y") |  |
|  |  |

|  |
| --- |
| SCATTER PLOTS |
|  |
| fig, ax = plt.subplots(2,2 , figsize=(10,10), gridspec\_kw={'width\_ratios': [1, 1], 'height\_ratios': [1, 2]})  ax[1,0]  graph = sns.scatterplot(x="tip", y="total\_bill", data=df, ax=ax[0,0], hue="time" ,style="time")  ax[0,0].set\_title("SCATTER PLOT time")  grafik = sns.scatterplot(x="tip", y="total\_bill", data=df, ax=ax[0,1], hue="day", style="time")  ax[0,1].set\_title("SCATTER PLOT- time/day")  bask = sns.scatterplot(x="tip", y="total\_bill",size="size" ,data=df, ax=ax[1,0], hue="day", style="time")  ax[1,0].set\_title("SCATTER PLOT- time/day + size")  ax[1,0].legend(ncol=3,loc=4)  bas1k = sns.scatterplot(x="tip", y="total\_bill",size="size" ,data=df, ax=ax[1,1], hue="size")  ax[1,1].set\_title("SCATTER PLOT- size/color")  ax[1,1].legend(loc=4, ncol=2) |

### LMPLOT(regplot)

|  |  |
| --- | --- |
| sns.lmplot(x= "total\_bill", y="tip", data=df, hue="smoker", col="sex") |  |
| sns.lmplot(x= "total\_bill", y="tip", data=df, hue="smoker", row="time") |  |

#### Pairplot:

sns.pairplot(df, hue= "species",  markers=["o","s","d"])

A group of graphs showing different sizes of dots

Description automatically generated with medium confidence

#### Pairplot-reg

y = sns.pairplot(df, kind="reg", hue= "species")

A collage of graphs

Description automatically generated

### HeatMaps

|  |
| --- |
| sns.heatmap(df, cmap= plt.get\_cmap("viridis")) |
|  |
| sns.heatmap(df, cmap= plt.get\_cmap("viridis").reversed(), annot= True, fmt="d", linewidths=.1) |
|  |

### Line Plots:

Tablo : Line Plots

|  |  |
| --- | --- |
| sns.lineplot(x= "timepoint", y= "signal", data=df, color=sns.color\_palette("Set3")[9]) |  |
| sns.lineplot(x= "timepoint", y= "signal", data=df, hue= "event") |  |
| sns.lineplot(x= "timepoint", y= "signal", data=df, hue= "event", style="event", markers=True, dashes= False) |  |
| #ci -> confidence interval(default= 95)  sns.lineplot(x= "timepoint", y= "signal", data=df, hue= "region", style="event", ci= 50, palette= sns.color\_palette("Set1", n\_colors=4)) |  |

# Artificial Neural Networks

## Regularization

### Bias & Variance

A diagram with red and blue lines

Description automatically generated

|  |  |
| --- | --- |
| High Bias | High Variance |
| Simple model  Low requirements | High accuracy |
| Low accuracy | High requirements  Complicated model |

A diagram of a diagram of a person's body

Description automatically generated

### Dropout

* Randomly dropping some neurons in the network is a simple way to maket he model more robust and prevent overfitting.
* keep\_prob is the hyperparameter that specifies the probablity of each neuron remaining active during training.

### Noise Addition

* Adding some noise data to make model more generalizable

### Overfitting & Underfitting

* Overfitting is when the model learns the data too well (memorizes it), while underfitting is when the model doesn't learn the data well enough
* Early stopping is a good option to prevent overfitting

A diagram of a curve

Description automatically generated

## Activation Funcitons

A graph of different colored lines

Description automatically generated