statinfer

Data Visualization

statinfer.com

Contents

- Continuous variable distributions
  - Box Plots
  - Histograms
- Categorical Variables Summary
  - Bar Charts
- Continuous vs Continuous
  - Scatter Plots
- Continuous vs Categorical
  - Histogram overlay
  - Multiple category Box Plots
  - Violin plots
- Categorical vs Categorical
  - Bar Charts
  - Cross tables
- Pivot table visualization
  - Heatmaps

Data Visualizations

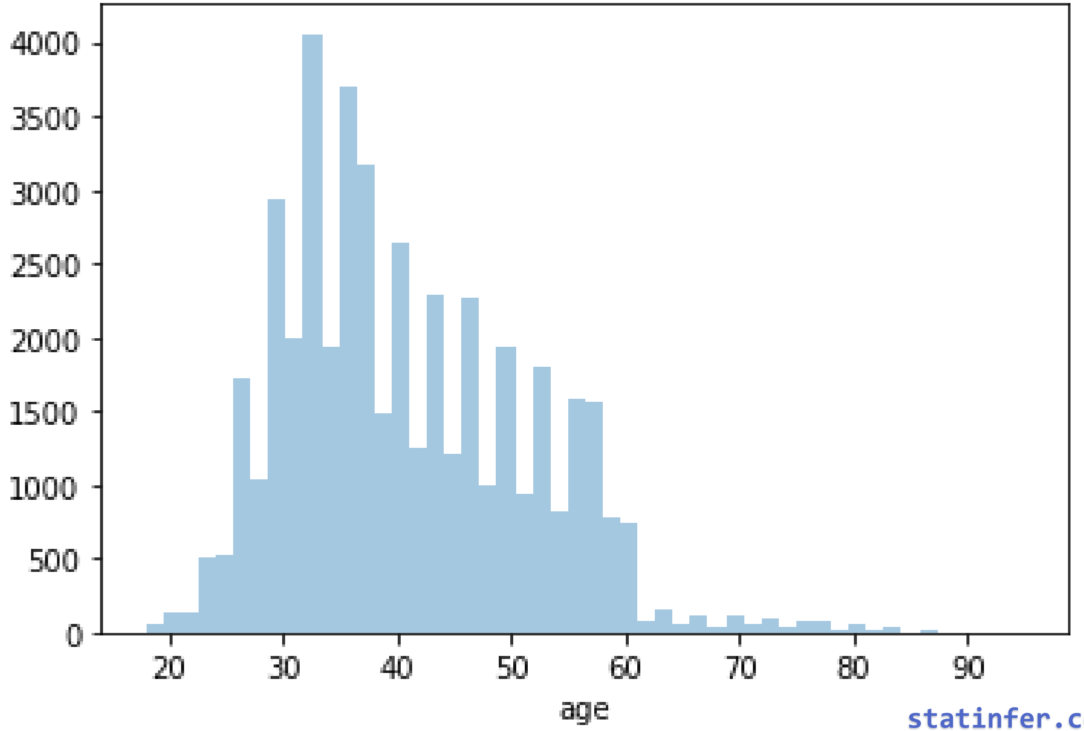
- Data Visualizations is NOT about beautiful charts and graphs
- It is about representing the data using right charts

statinfer

Continuous variable distributions

Histograms

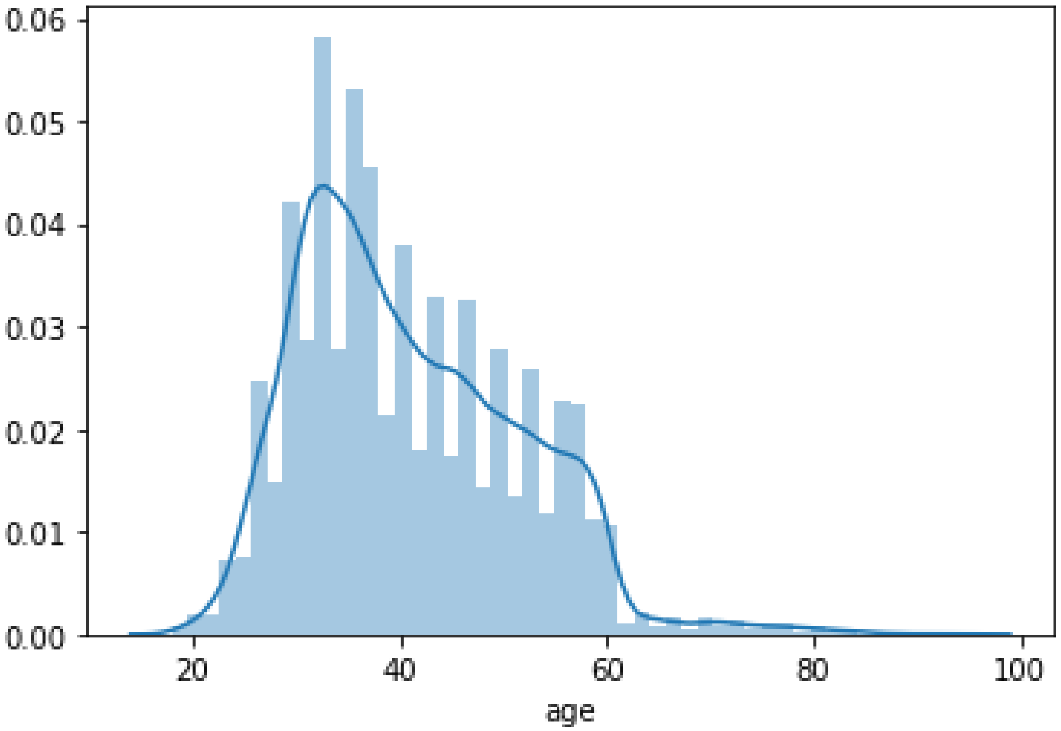
- Represents the frequency distribution of a numerical variable
- On X-axis, we have class intervals of the variable and on Y-axis we have corresponding frequencies.
- Gives an idea on the overall distribution of the variable.



statinfer.com

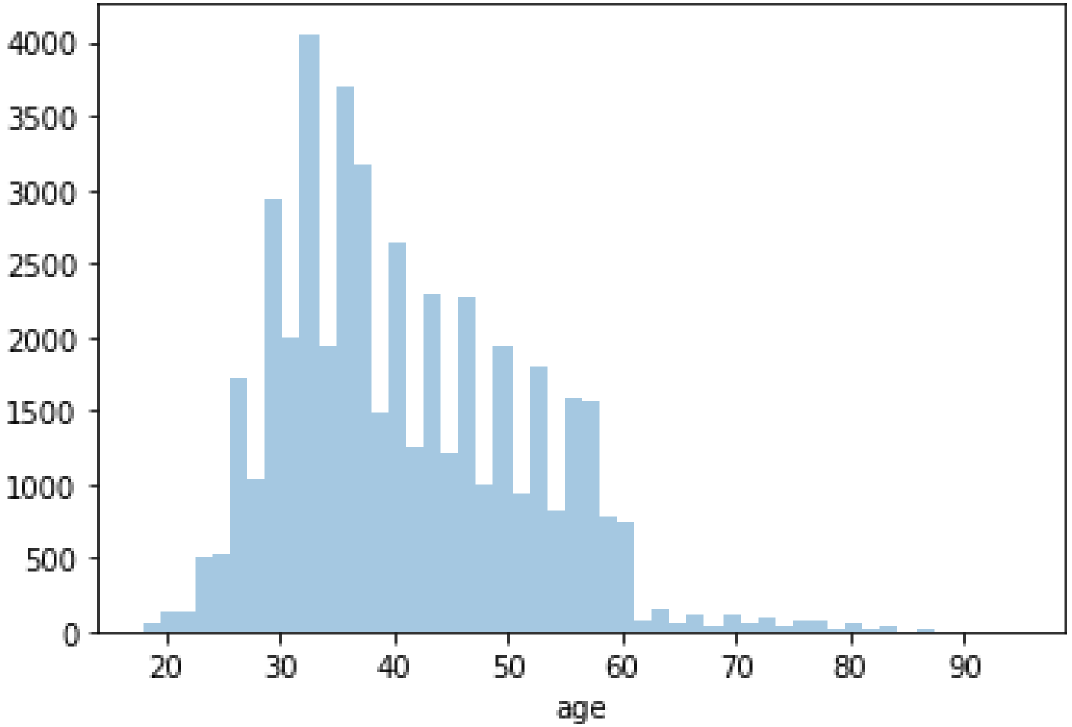
Code - Histogram Example

```
#Bank Marketing data
import seaborn as sns
sns.distplot(bank_data["age"])
```



Code - Histogram Example

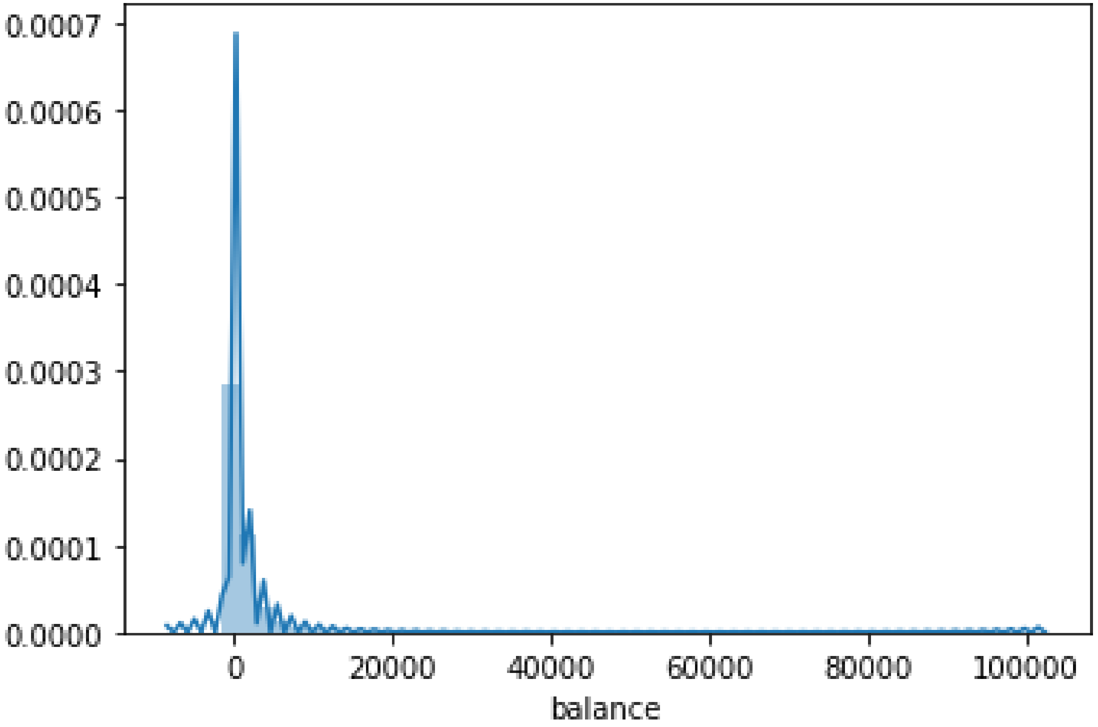
```
#Bank Marketing data
import seaborn as sns
sns.distplot(bank_data["age"] ,kde=False)
```



- **kde=False** Removes the smooth line from the diagram
- Removes the kernel density estimate.

Code - Histogram Example

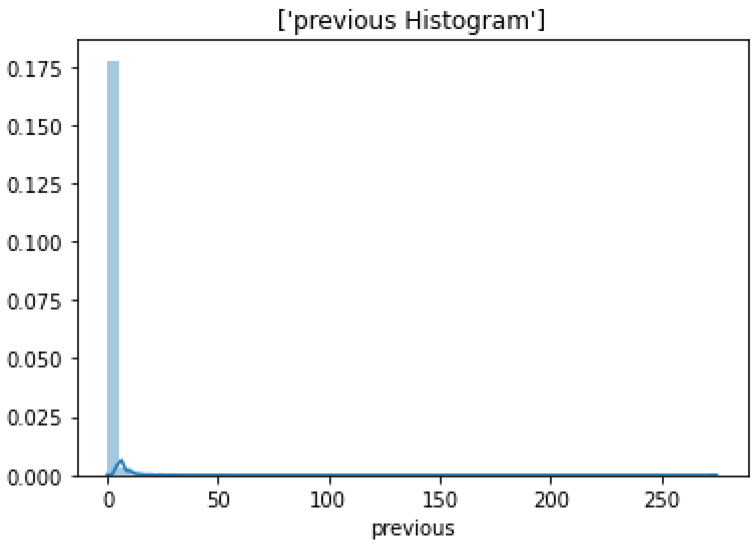
```
import seaborn as sns
sns.distplot(bank_data["balance"])
```

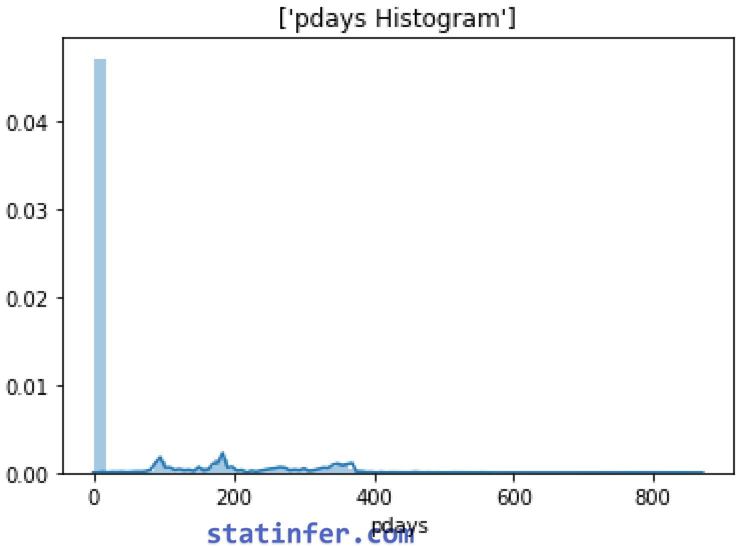


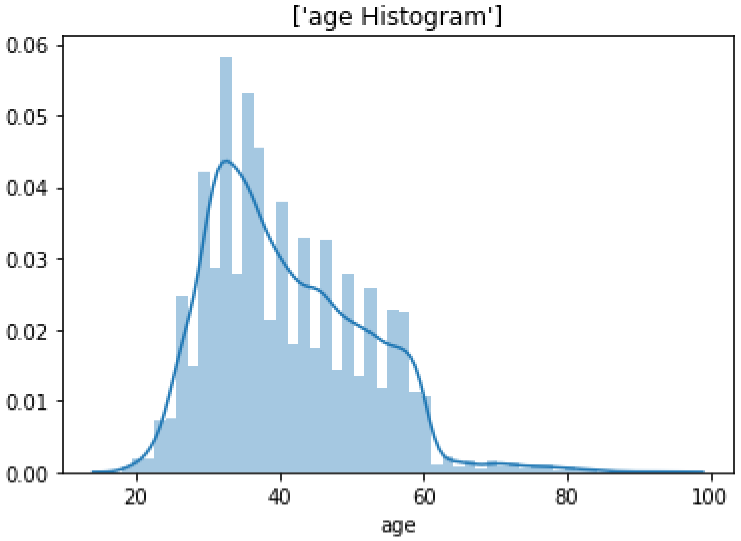
Histograms for all the numerical columns

```
numeric_cols=[col for col in bank_data.columns if bank_data[col].dtypes in
 ["int64","float64"]]
print(numeric_cols)

plt.figure()
for col in numeric_cols:
  sns.distplot(bank_data[col])
  plt.title([col + " Histogram"])
  plt.show()
```





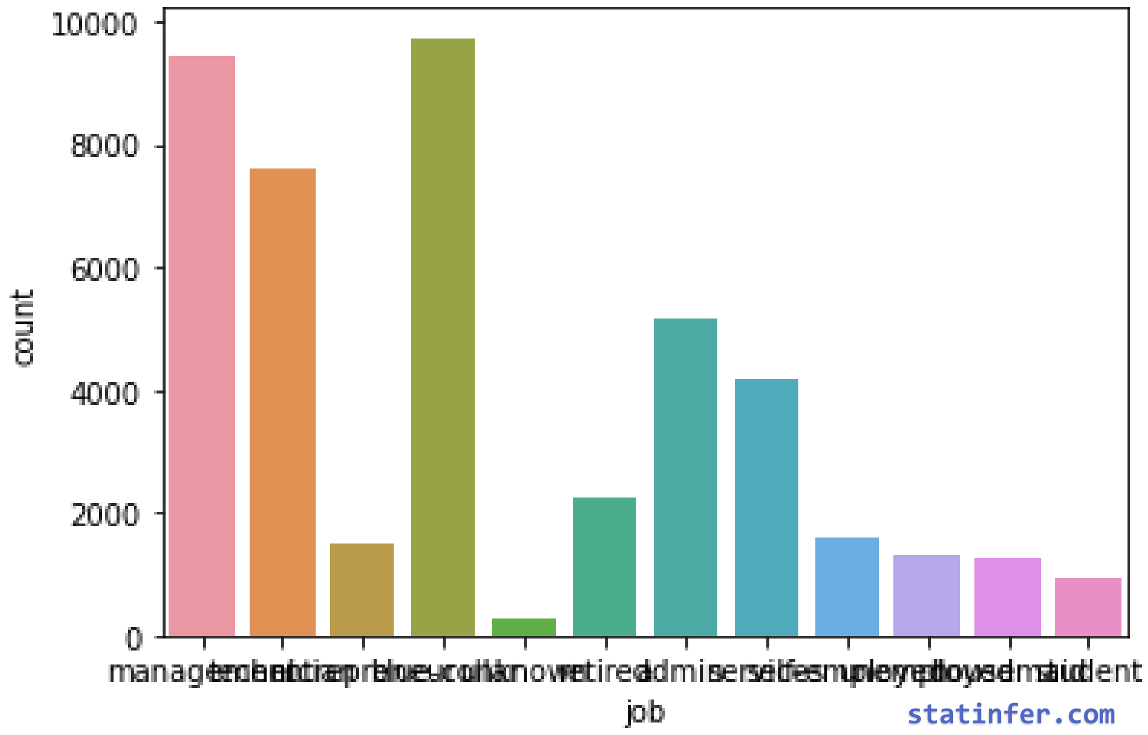


Categorical Variable Visualization

Bar charts

- Bar charts used to summarize the categorical variables

```
plt.figure()
sns.countplot(x="job",  data=bank_data)
```

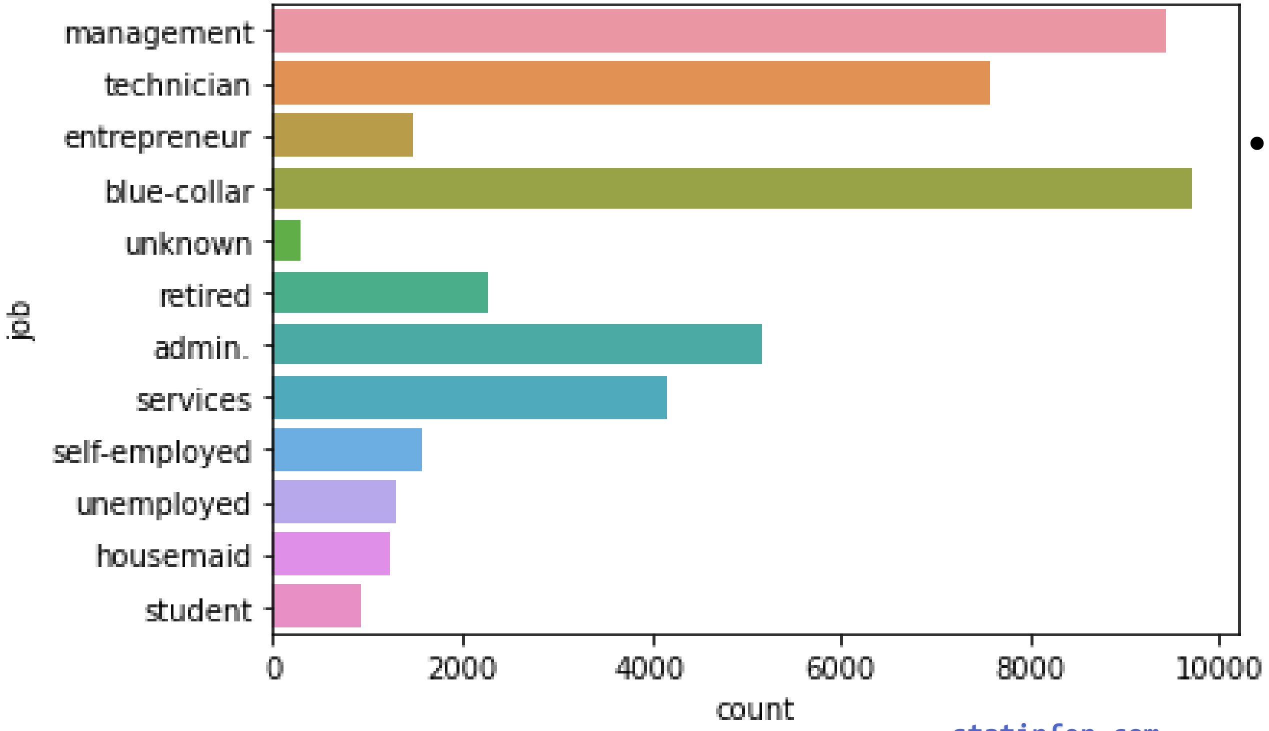


- Countplot() function is used to create box plots

Bar chart - Horizontal

- All the categories are shown in the chart

```
plt.figure()
sns.countplot(y="job",  data=bank_data)
```



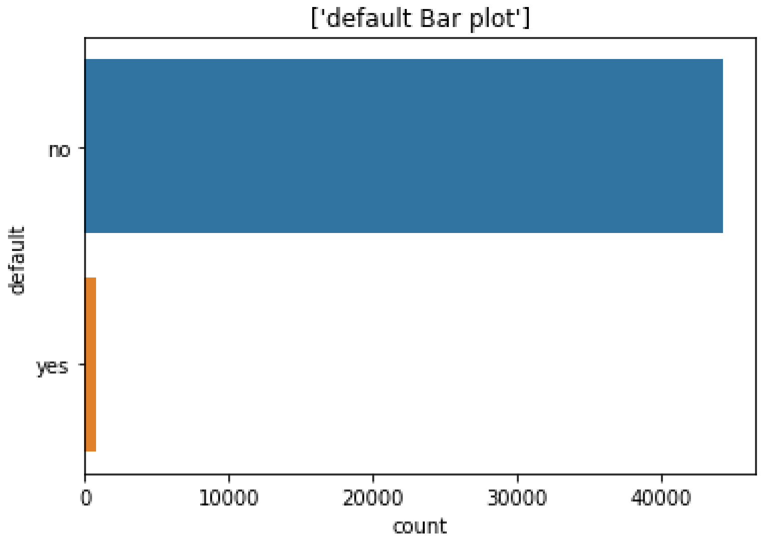
statinfer.com

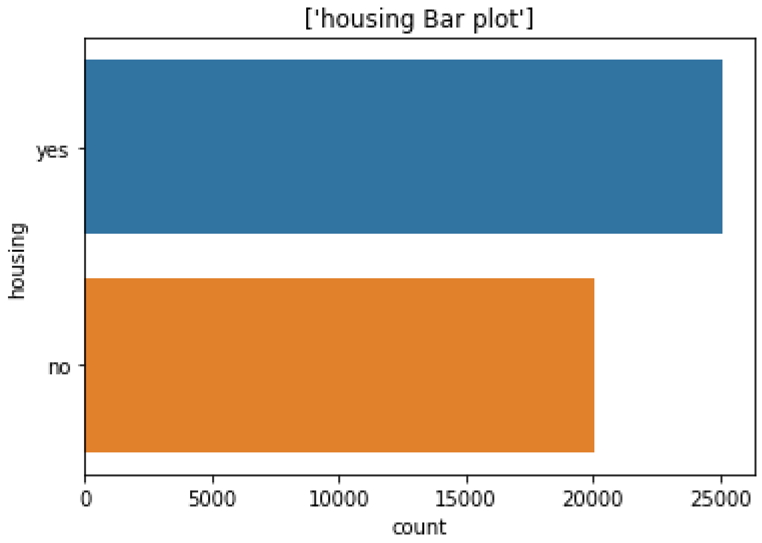
- Change the axis to display the category names

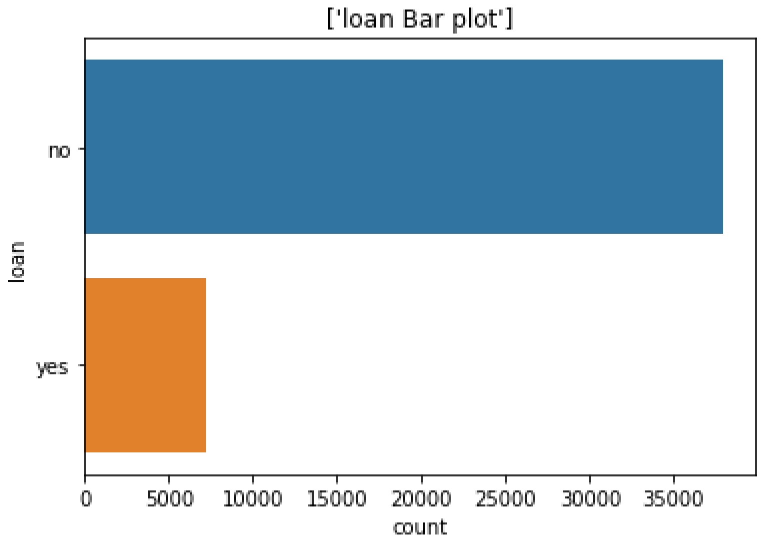
Bar chart for all the variables

```
categorical_cols=[col for col in bank_data.columns if bank_data[col].dtypes
 in ["object"]]
print(categorical_cols)

plt.figure()
for col in categorical_cols:
  sns.countplot(y=col,  data=bank_data)
  plt.title([col + " Bar plot"])
  plt.show()
```







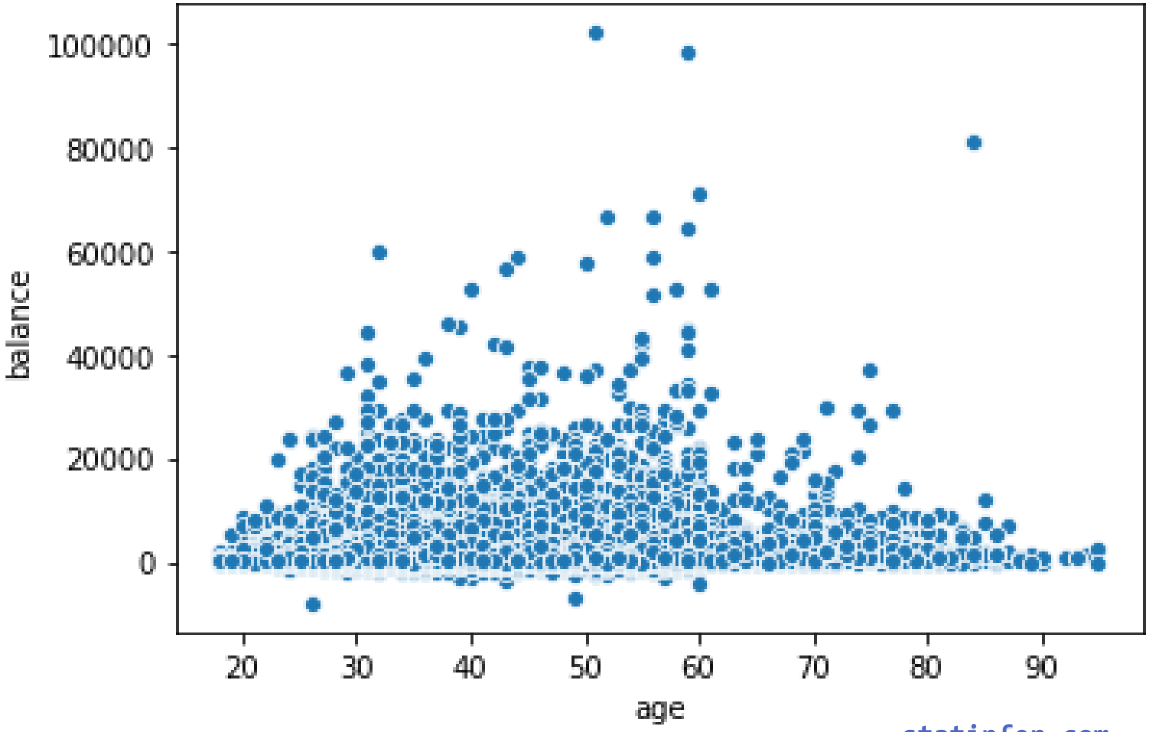
statinfer

Continuous vs Continuous

Continuous vs Continuous

- Scatter plot is for showing relation between Continuous Numerical Variables

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.scatterplot(x="age", y="balance", data=bank_data)
```

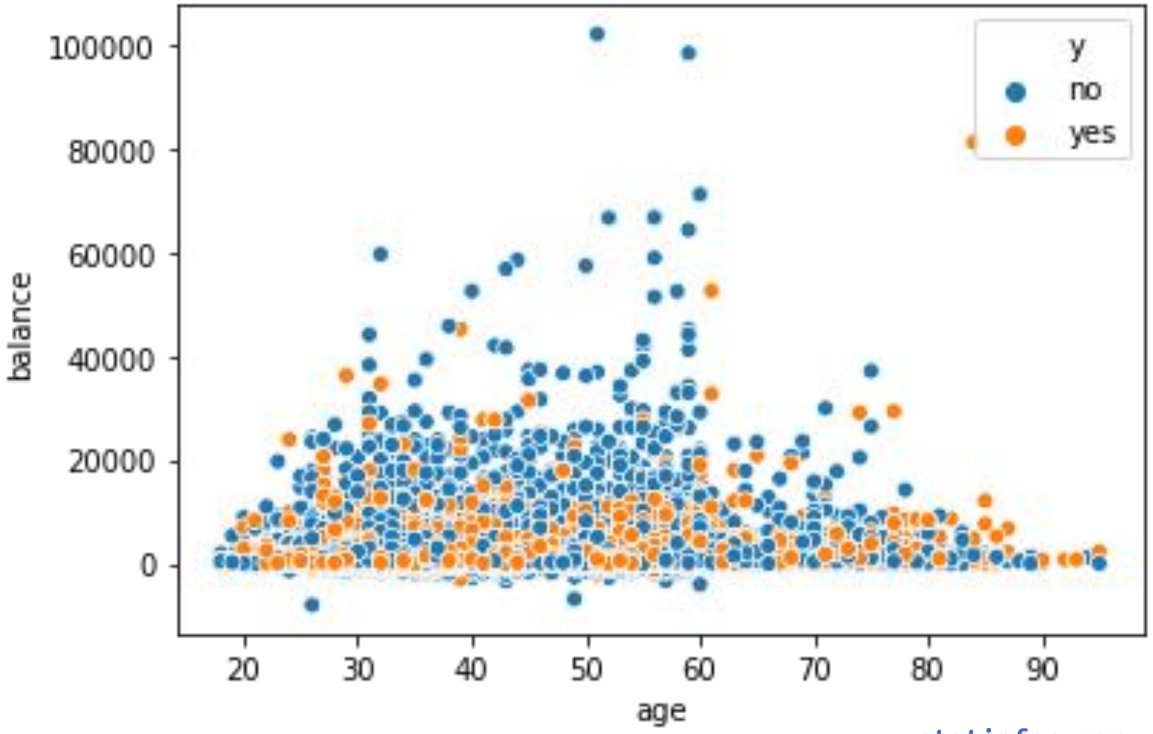


statinfer.com

Continuous vs Continuous

- Scatter plot is for showing relation between Continuous Numerical Variables

```
import seaborn as sns
import matplotlib.pyplot as plt
sns.scatterplot(x="age", y="balance", hue="y", data=bank_data)
```



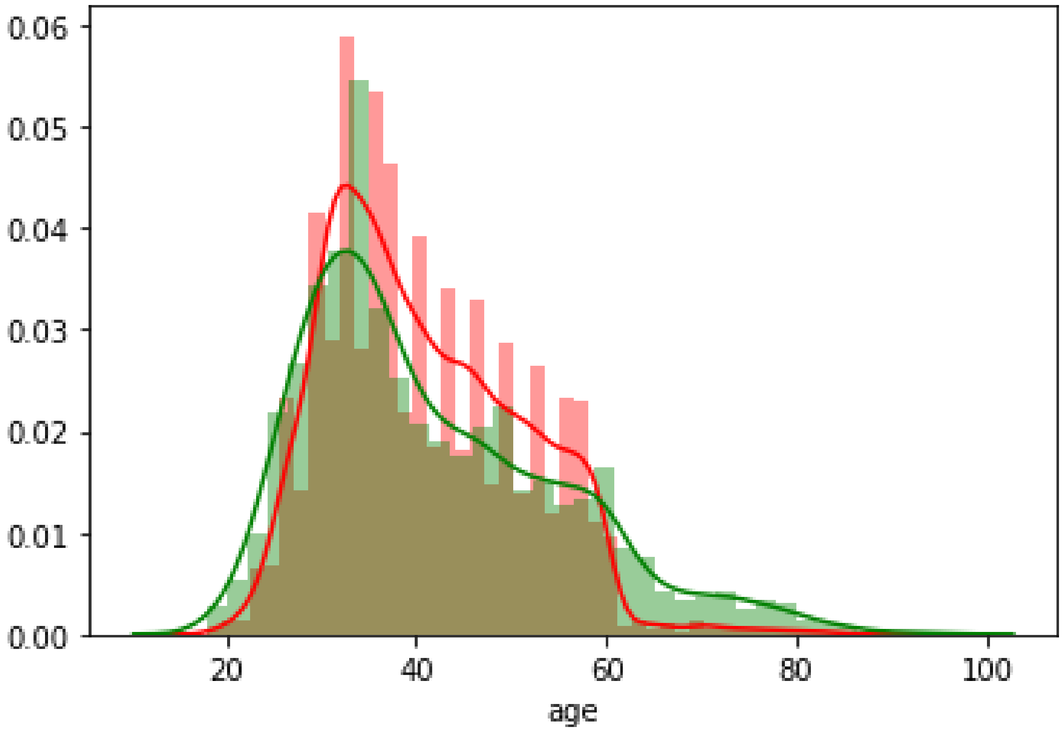
- Adding hue to the dots using a different variable

Continuous vs Categorical

Continuous vs Categorical

- Histograms for categories

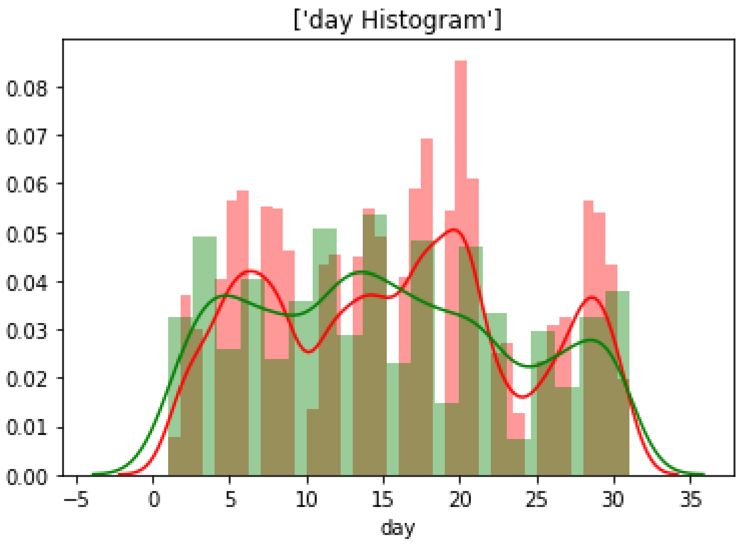
```
sns.distplot(bank_data[bank_data["y"]=="no"]["age"], color="red")
sns.distplot(bank_data[bank_data["y"]=="yes"]["age"], color="green")
```

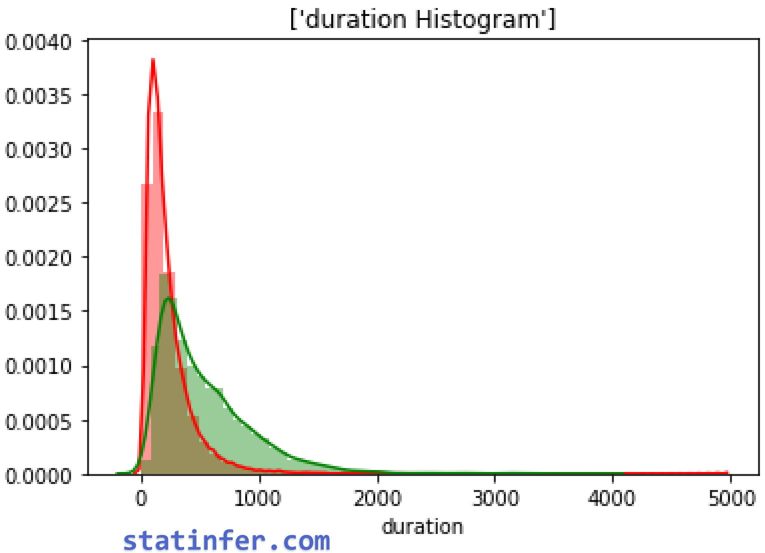


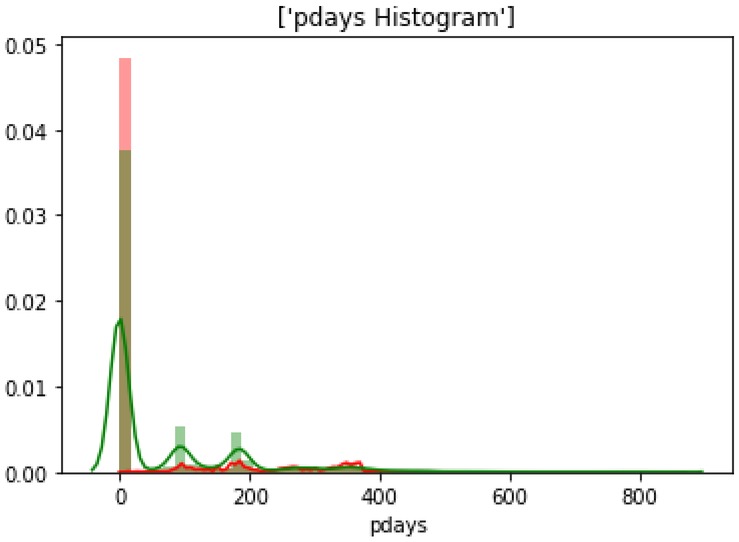
Continuous vs Categorical

- Histograms for categories

```
plt.figure()
for col in numeric_cols:
    sns.distplot(bank_data[bank_data["y"]=="no"][col], color="red")
    sns.distplot(bank_data[bank_data["y"]=="yes"][col], color="green")
    plt.title([col + " Histogram"])
    plt.show()
```

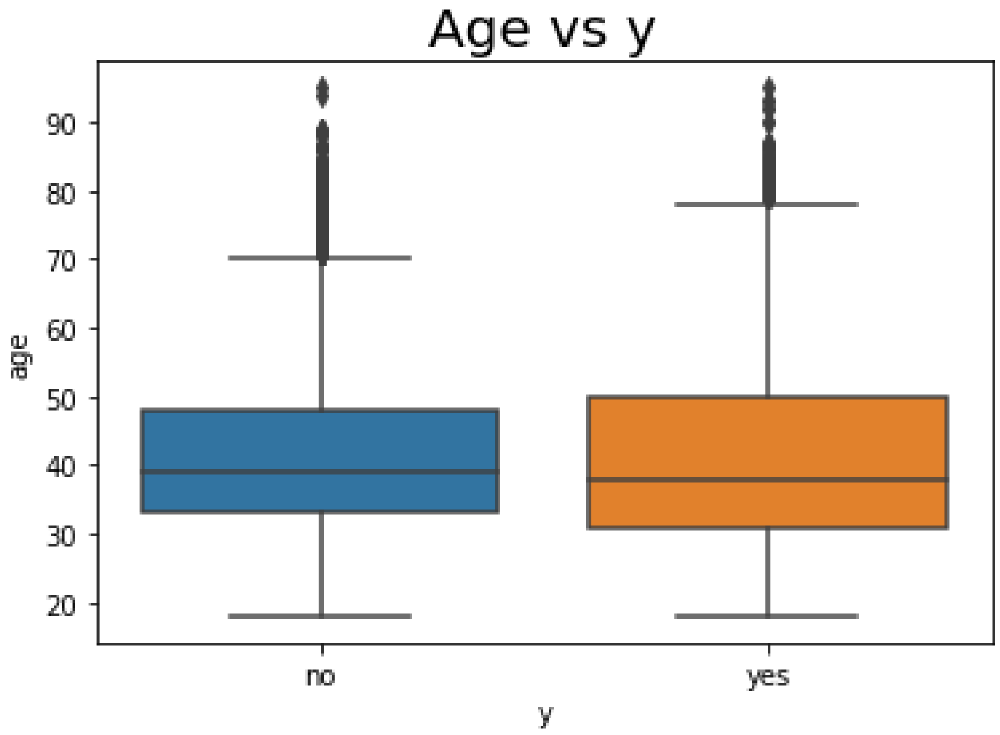






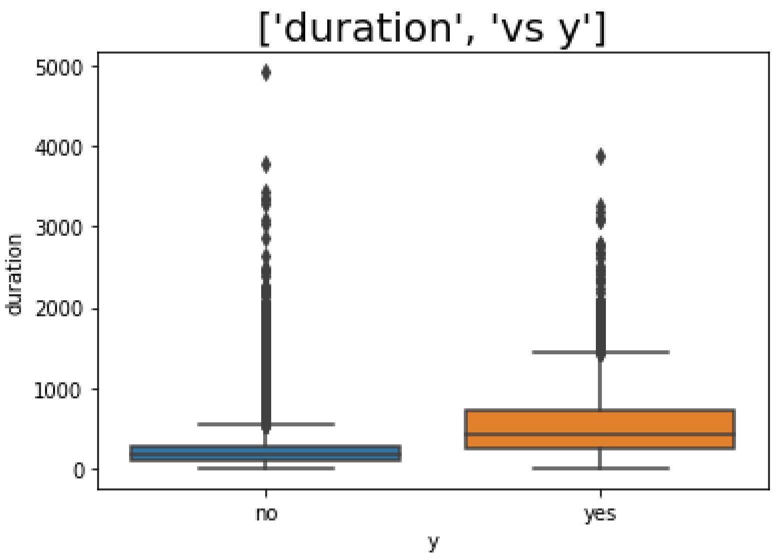
Continuous vs Categorical– Using Box Plots

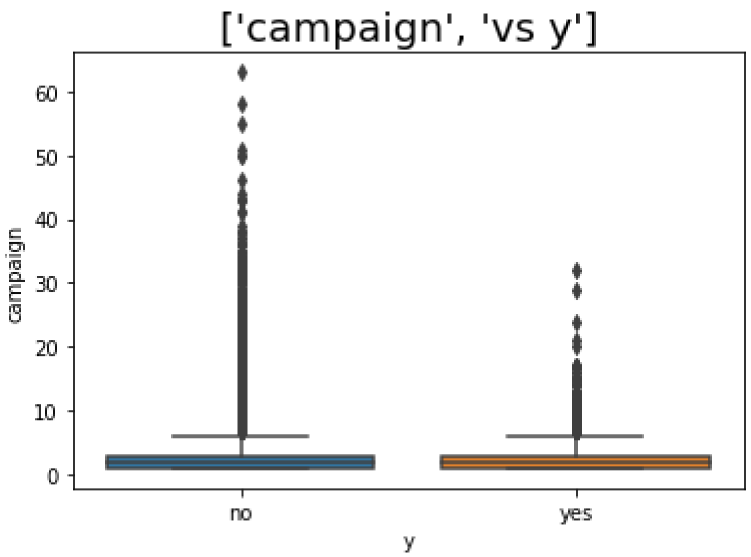
```
sns.boxplot( x=bank_data["y"], y=bank_data["age"])
plt.title('Age vs y', fontsize=20)
```

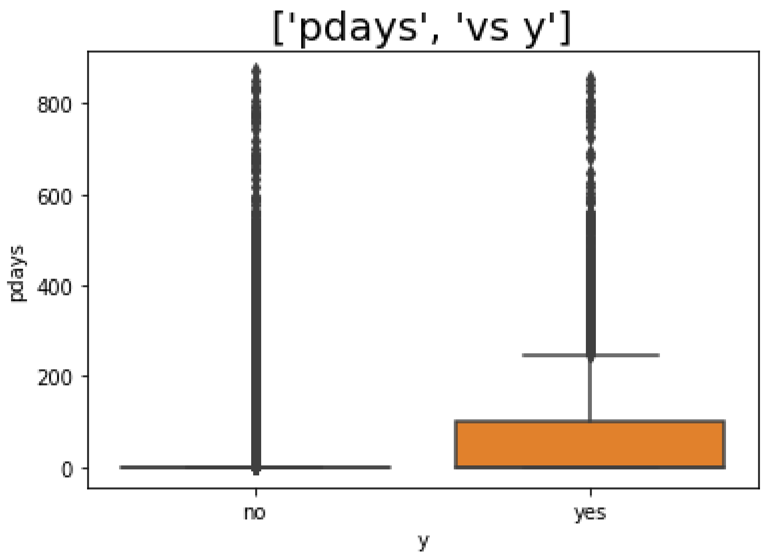


Continuous vs Categorical–Box Plots

```
for col in numeric_cols:
  sns.boxplot( x=bank_data["y"], y=bank_data[col])
  plt.title([col, "vs y"], fontsize=20)
  plt.show()
```

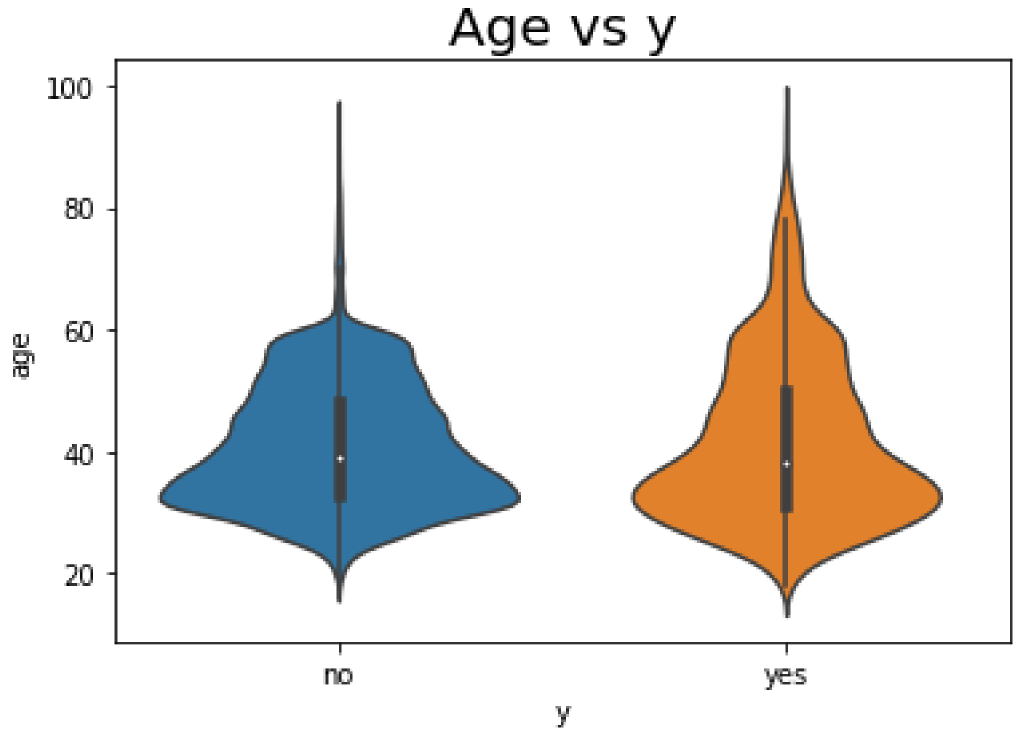






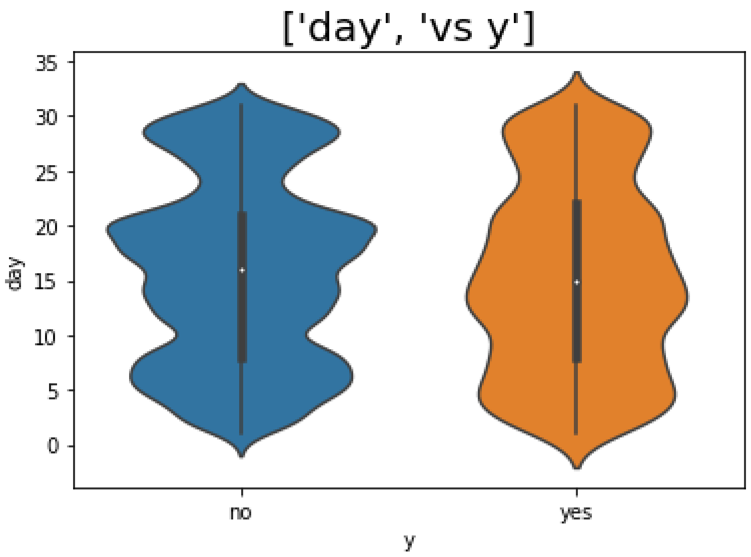
Continuous vs Categorical–Violin plots

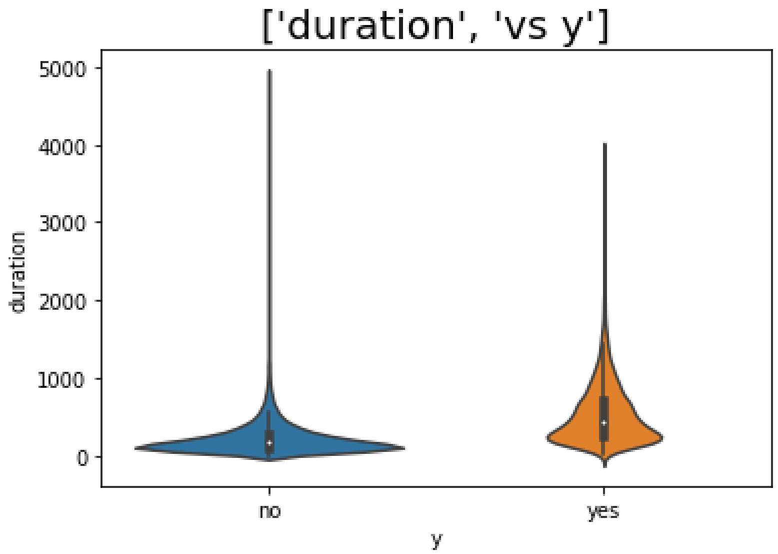
```
sns.violinplot( x=bank_data["y"], y=bank_data["age"])
plt.title('Age vs y', fontsize=20)
```

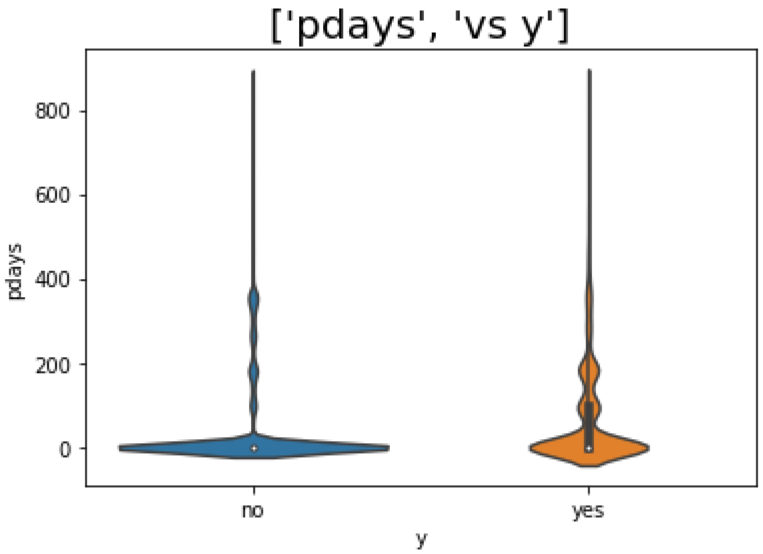


Continuous vs Categorical–Violin plots

```
for col in numeric_cols:
  sns.violinplot( x=bank_data["y"], y=bank_data[col])
  plt.title([col, "vs y"], fontsize=20)
  plt.show()
```


['day', 'vs y']


['duration', 'vs y']

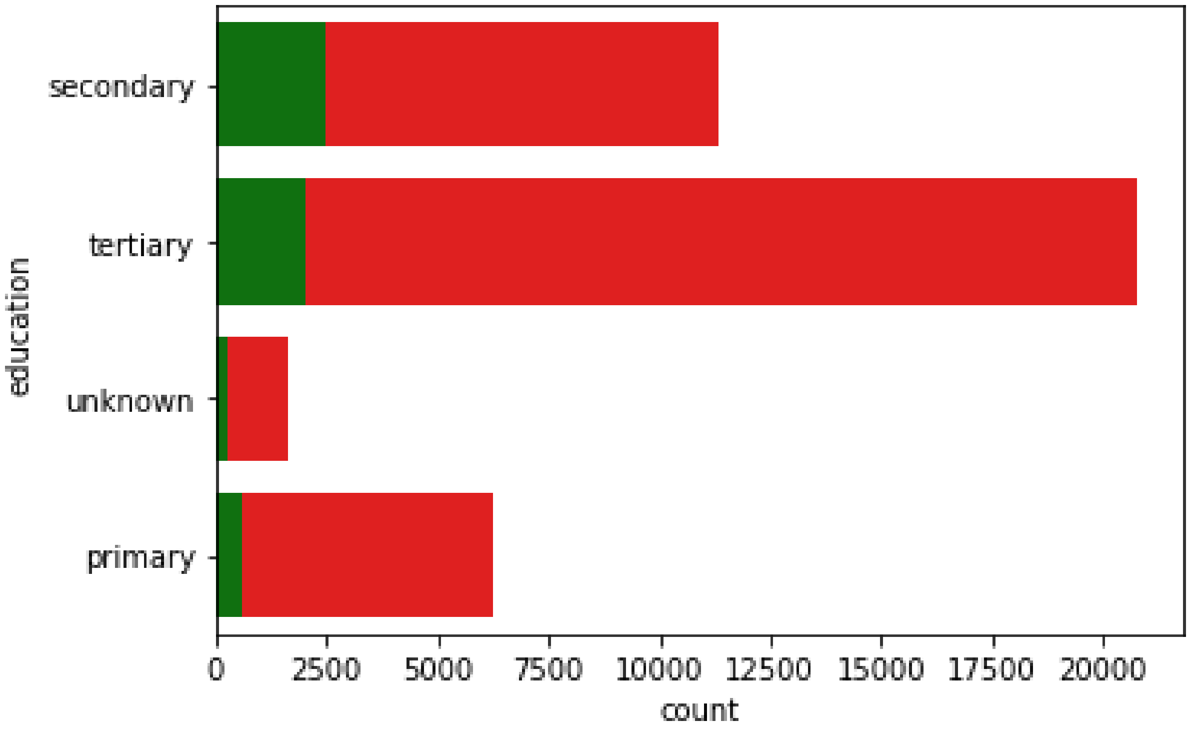

['pdays', 'vs y']

statinfer

Categorical vs Categorical

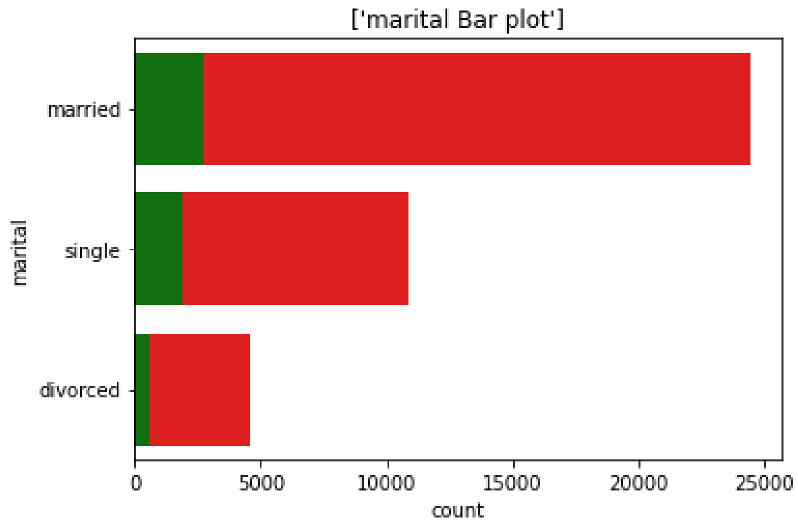
Categorical vs Categorical

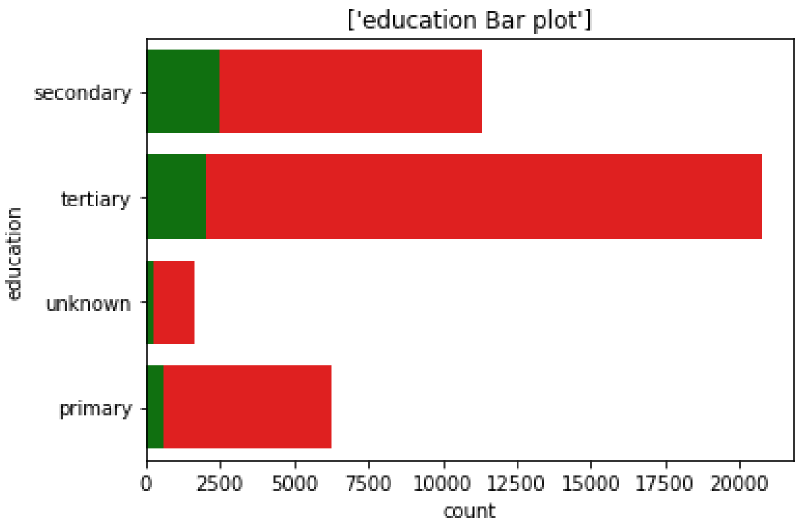
```
sns.countplot(y="education", data=bank_data[bank_data["y"]=="no"], color="red")
sns.countplot(y="education", data=bank_data[bank_data["y"]=="yes"], color="green")
```

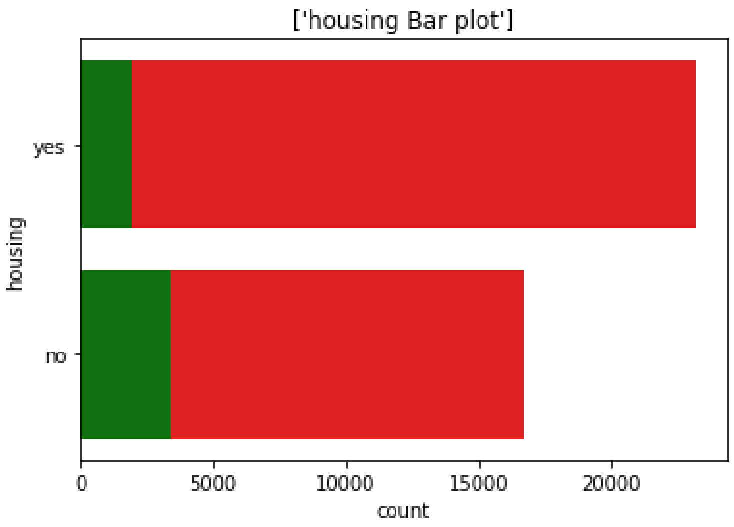


Categorical vs Categorical

```
plt.figure()
for col in categorical_cols:
  sns.countplot(y=col,  data=bank_data[bank_data["y"]=="no"], color="red")
  sns.countplot(y=col,  data=bank_data[bank_data["y"]=="yes"], color="green")
  plt.title([col + " Bar plot"])
  plt.show()
```

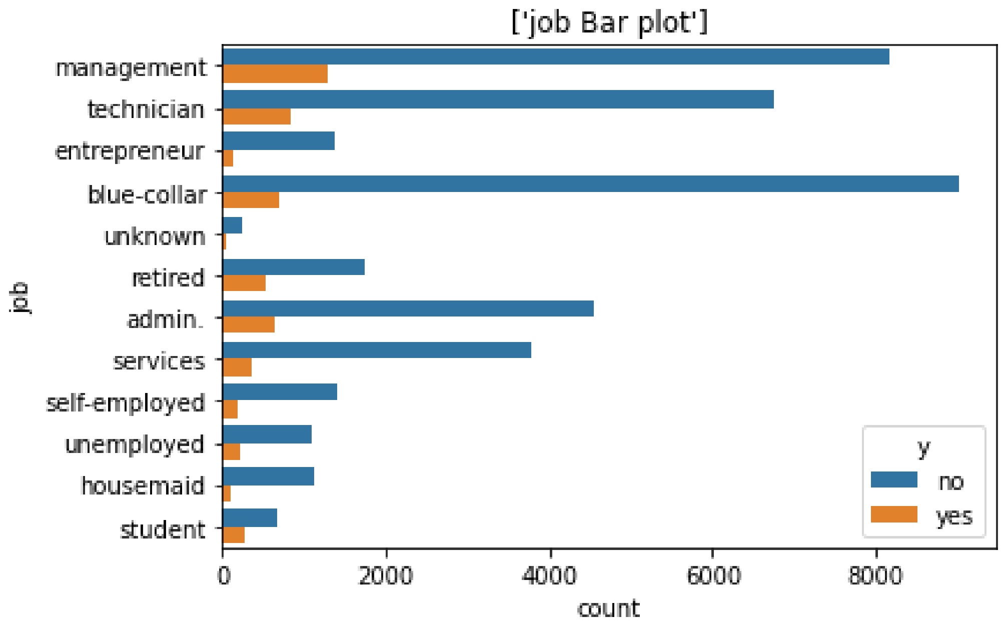


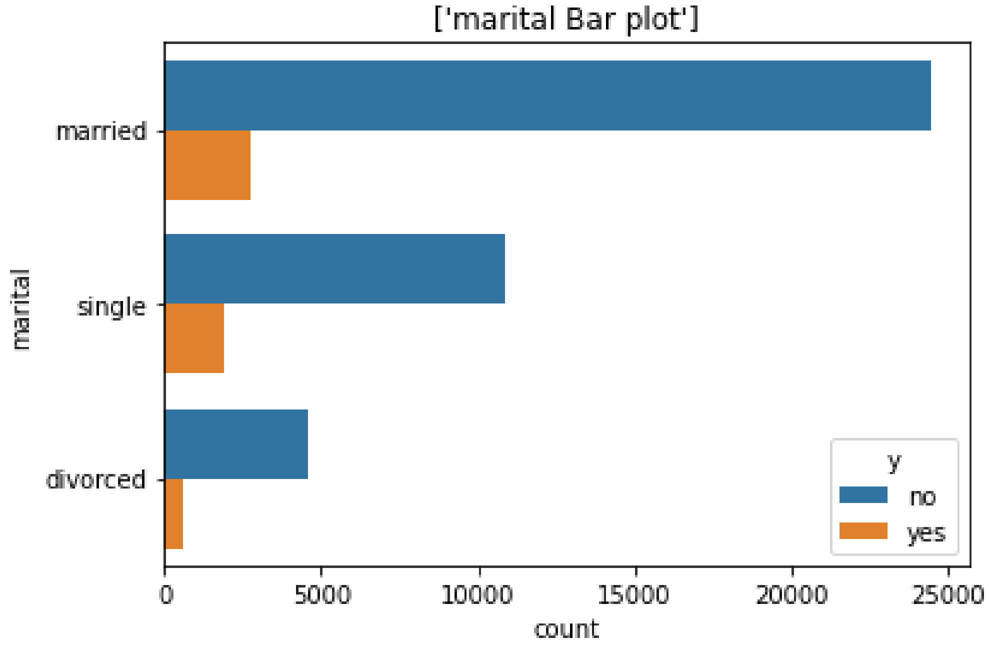




Categorical vs Categorical – Alternate method

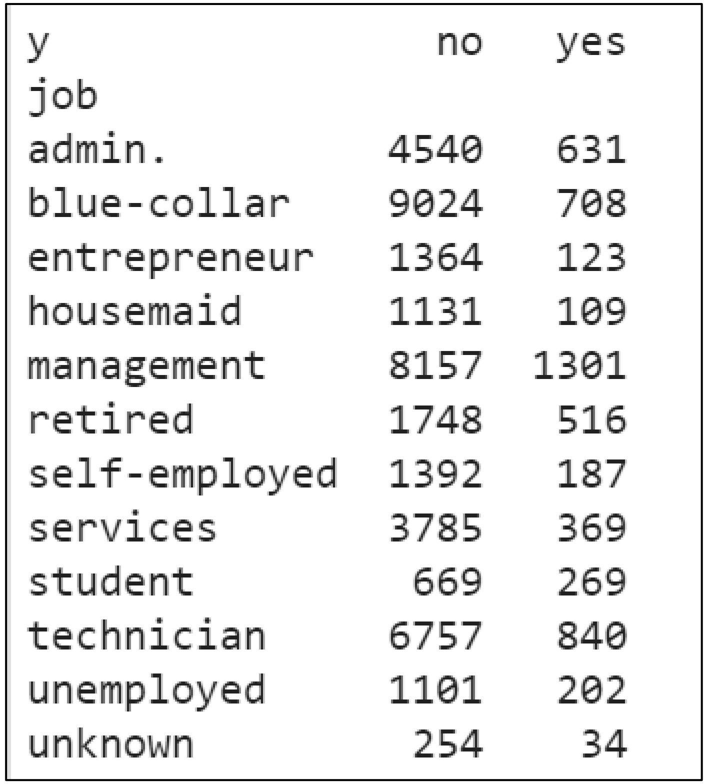
```
plt.figure()
for col in categorical_cols:
  sns.countplot(y=col,  data=bank_data,hue="y")
  plt.title([col + " Bar plot"])
  plt.show()
```



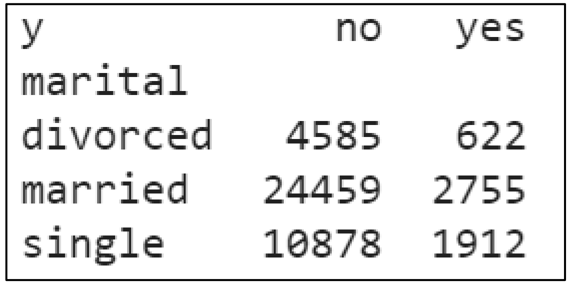


Cross tables

```
for col in categorical_cols:
  print(pd.crosstab(bank_data[col], bank_data['y']))
```



```
y                no   yes
job
admin.         4540   631
blue-collar    9024   708
entrepreneur   1364   123
housemaid      1131   109
management     8157  1301
retired        1748   516
self-employed  1392   187
services       3785   369
student         669   269
technician     6757   840
unemployed     1101   202
unknown         254    34
```



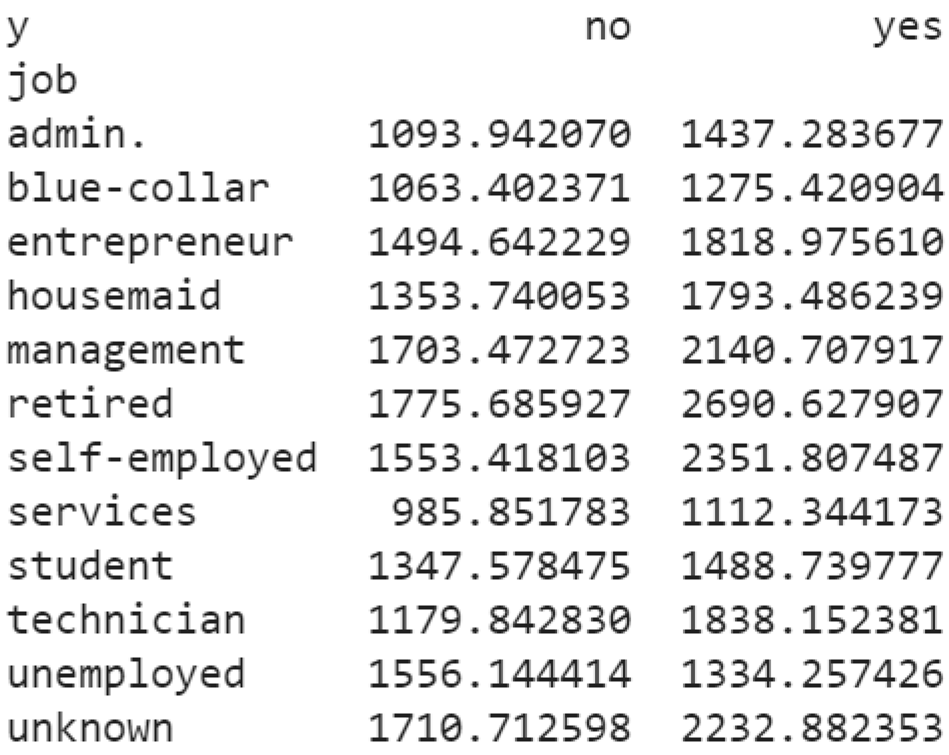
```
y            no   yes
marital
divorced   4585   622
married   24459  2755
single    10878  1912
```

statinfer

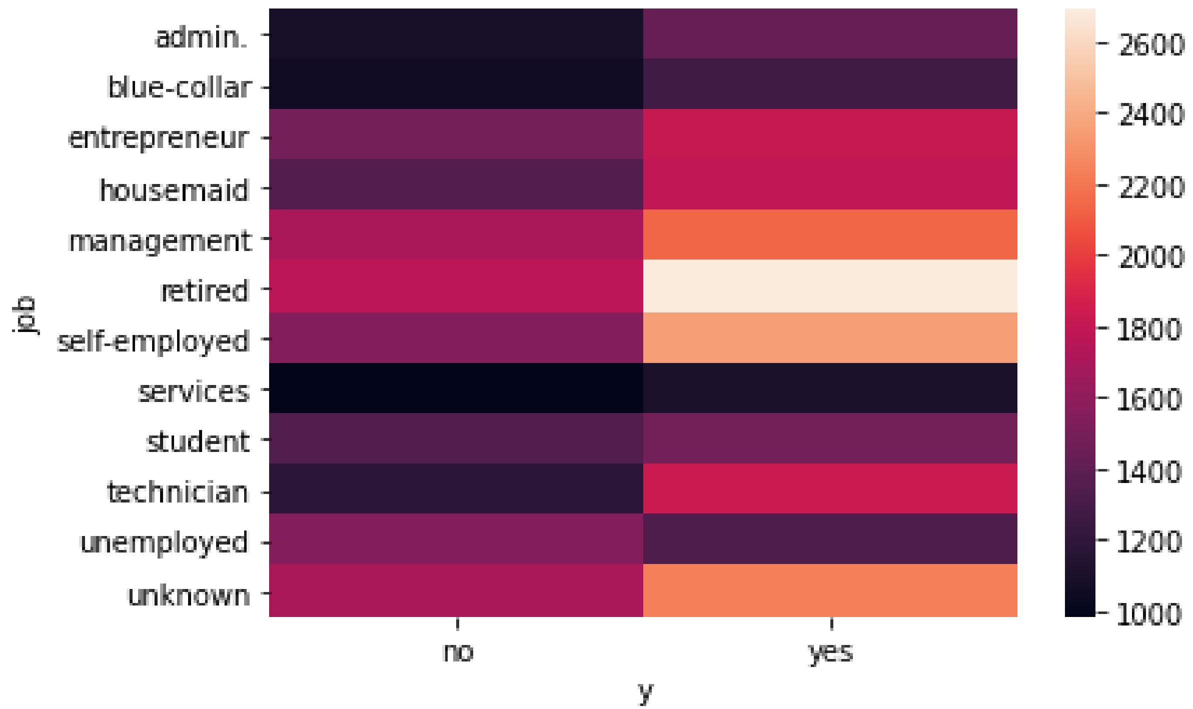
Pivot table visualization

Heat maps

```
pivot=pd.pivot_table(bank_data,values='balance', index=['job'], columns='y')
print(pivot)
sns.heatmap(pivot)
```

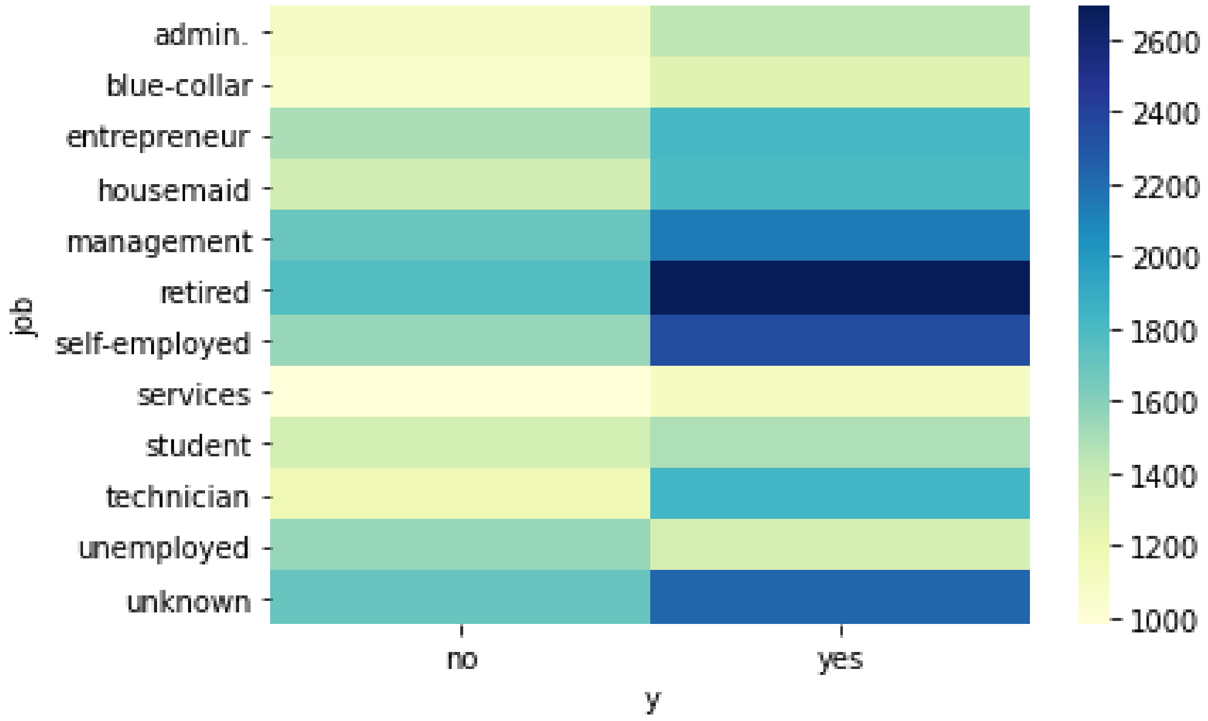


```
y                        no          yes
job
admin.          1093.942070  1437.283677
blue-collar     1063.402371  1275.420904
entrepreneur    1494.642229  1818.975610
housemaid       1353.740053  1793.486239
management      1703.472723  2140.707917
retired         1775.685927  2690.627907
self-employed   1553.418103  2351.807487
services         985.851783  1112.344173
student         1347.578475  1488.739777
technician      1179.842830  1838.152381
unemployed      1556.144414  1334.257426
unknown         1710.712598  2232.882353
```



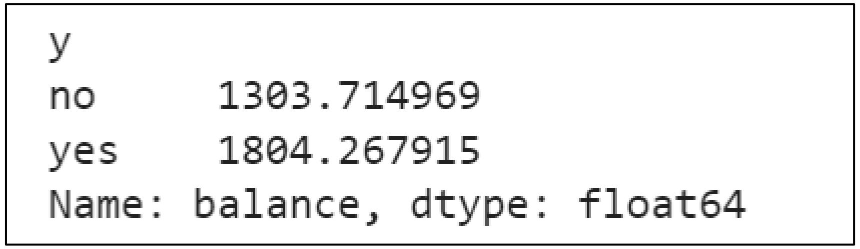
Heat maps

```
pivot=pd.pivot_table(bank_data,values='balance', index=['job'], columns='y')
sns.heatmap(pivot, cmap="YlGnBu")
```



Groupby

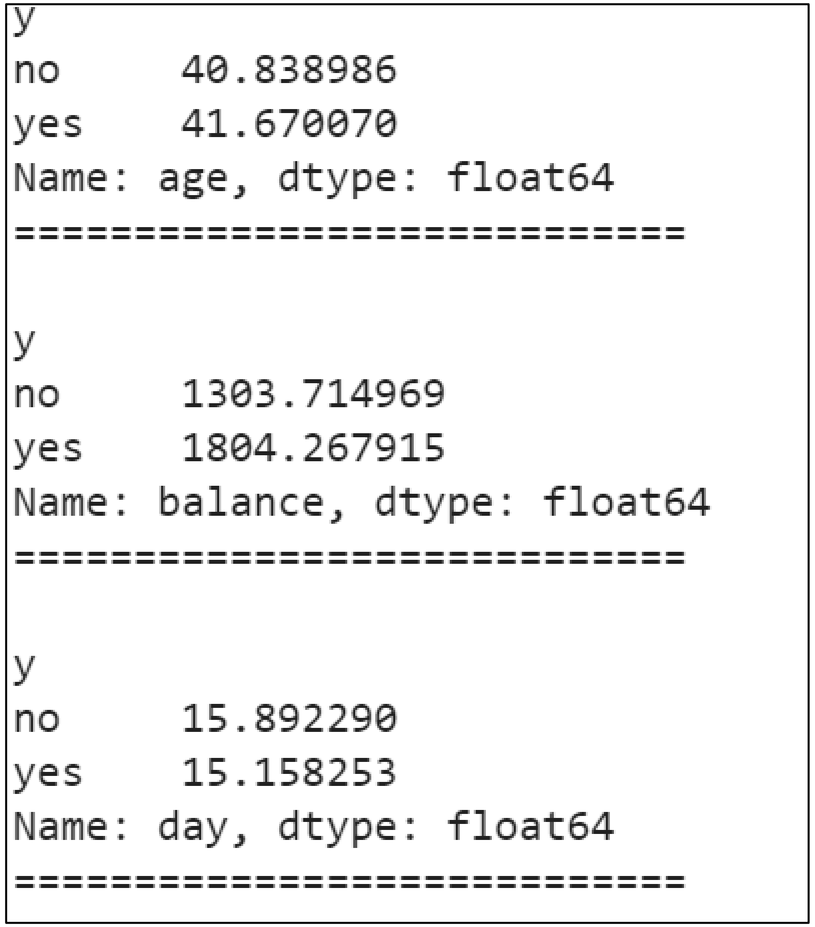
```
bank_data.groupby("y")['balance'].mean()
```



```
y
no      1303.714969
yes     1804.267915
Name: balance, dtype: float64
```

Groupby

```
for col in numeric_cols:
  print(bank_data.groupby("y")[col].mean())
  print("============================\n")
```



```
y
no      40.838986
yes     41.670070
Name: age, dtype: float64
============================

y
no      1303.714969
yes     1804.267915
Name: balance, dtype: float64
============================

y
no      15.892290
yes     15.158253
Name: day, dtype: float64
============================
```

statinfer

Thank you