```
import numpy as np
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
# !pip install missingno
import missingno as msno
from datetime import date
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.neighbors import LocalOutlierFactor
from sklearn.preprocessing import MinMaxScaler, LabelEncoder, StandardScaler, RobustScaler
```

```
from google.colab import files
```

```
uploaded = files.upload()
```

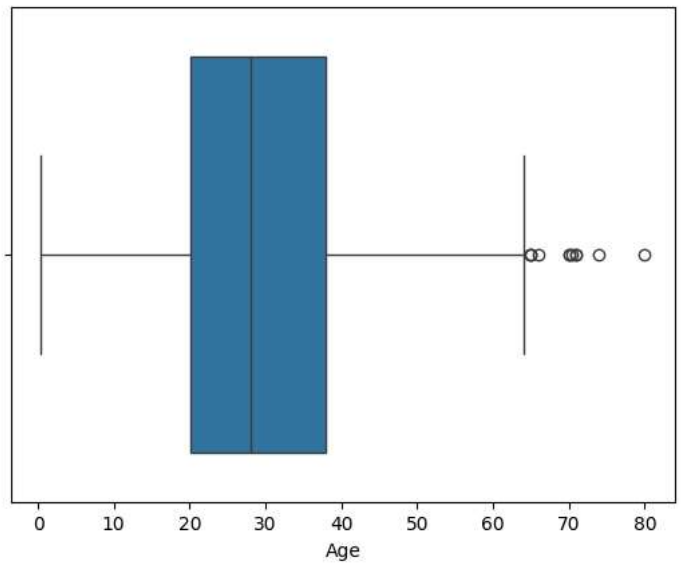
```
Choose Files titanic.csv
  • titanic.csv(text/csv) - 61194 bytes, last modified: 6/28/2024 - 100% done
Saving titanic.csv to titanic.csv
```

```
def load():
    data = pd.read_csv("titanic.csv")
    return data
```

```
#See the shape of smaller dataset
df = load()
print(df.shape)
```

```
(891, 12)
```

```
sns.boxplot(x=df["Age"])
plt.show()
```




```
q1 = df["Age"].quantile(0.25)
q3 = df["Age"].quantile(0.75)
iqr = q3 - q1
up = q3 + 1.5 * iqr
low = q1 - 1.5 * iqr
```

```
#Now see outliers
print(df[(df["Age"] < low) | (df["Age"] > up)])
```

```
     PassengerId  Survived  Pclass                                     Name  \
33            34         0       2                    Wheadon, Mr. Edward H
54            55         0       1           Ostby, Mr. Engelhart Cornelius
96            97         0       1                Goldschmidt, Mr. George B
116          117         0       3                     Connors, Mr. Patrick
```

```
      280          281         0       3                     Duane, Mr. Frank
      456          457         0       1             Millet, Mr. Francis Davis
      493          494         0       1               Artagaveytia, Mr. Ramon
      630          631         1       1  Barkworth, Mr. Algernon Henry Wilson
      672          673         0       2           Mitchell, Mr. Henry Michael
      745          746         0       1          Crosby, Capt. Edward Gifford
      851          852         0       3                   Svensson, Mr. Johan

            Sex   Age  SibSp  Parch      Ticket     Fare Cabin Embarked
      33   male  66.0      0      0  C.A. 24579  10.5000   NaN        S
      54   male  65.0      0      1      113509  61.9792   B30        C
      96   male  71.0      0      0    PC 17754  34.6542    A5        C
      116  male  70.5      0      0      370369   7.7500   NaN        Q
      280  male  65.0      0      0      336439   7.7500   NaN        Q
      456  male  65.0      0      0       13509  26.5500   E38        S
      493  male  71.0      0      0    PC 17609  49.5042   NaN        C
      630  male  80.0      0      0       27042  30.0000   A23        S
      672  male  70.0      0      0  C.A. 24580  10.5000   NaN        S
      745  male  70.0      1      1   WE/P 5735  71.0000   B22        S
      851  male  74.0      0      0      347060   7.7750   NaN        S
```

```
print(df[(df["Age"] < low) | (df["Age"] > up)].index)
print(df[(df["Age"] < low) | (df["Age"] > up)].any(axis=None))  #True
print(df[(df["Age"] < low)].any(axis=None))  #  False
```

```
Index([33, 54, 96, 116, 280, 456, 493, 630, 672, 745, 851], dtype='int64')
True
False
```

```
def outlier_thresholds(dataframe, col_name, q1=0.25, q3=0.75):
    quartile1 = dataframe[col_name].quantile(q1)
    quartile3 = dataframe[col_name].quantile(q3)
    interquantile_range = quartile3 - quartile1
    up_limit = quartile3 + 1.5 * interquantile_range
    low_limit = quartile1 - 1.5 * interquantile_range
    return low_limit, up_limit
print(outlier_thresholds(df, "Age")) # (-6.6875, 64.8125)
low, up = outlier_thresholds(df, "Fare")
print(df[(df["Fare"] < low) | (df["Fare"] > up)].head())
```

```
(-6.6875, 64.8125)
    PassengerId  Survived  Pclass  \
1             2         1       1
27           28         0       1
31           32         1       1
34           35         0       1
52           53         1       1

                                                 Name     Sex   Age  SibSp  \
1   Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1
27                      Fortune, Mr. Charles Alexander    male  19.0      3
31     Spencer, Mrs. William Augustus (Marie Eugenie)  female   NaN      1
34                              Meyer, Mr. Edgar Joseph    male  28.0      1
52           Harper, Mrs. Henry Sleeper (Myna Haxtun)  female  49.0      1

    Parch    Ticket      Fare        Cabin Embarked
1       0  PC 17599   71.2833          C85        C
27      2     19950  263.0000  C23 C25 C27        S
31      0  PC 17569  146.5208          B78        C
34      0  PC 17604   82.1708          NaN        C
52      0  PC 17572   76.7292          D33        C
```

```
def check_outlier(dataframe, col_name):
    low_limit, up_limit = outlier_thresholds(dataframe, col_name)
    if dataframe[(dataframe[col_name] > up_limit) | (dataframe[col_name] < low_limit)].any(axis=None):
        return True
    else:
        return False
print(check_outlier(df, "Age"))
print(check_outlier(df, "Fare"))
```

```
True
True
```

```
def grab_col_names(dataframe, cat_th=10, car_th=20):

    cat_cols = [col for col in dataframe.columns if dataframe[col].dtypes == "O"]
    num_but_cat = [col for col in dataframe.columns if dataframe[col].nunique() < cat_th and dataframe[col].dtypes != "O"]
    cat_but_car = [col for col in dataframe.columns if dataframe[col].nunique() > car_th and dataframe[col].dtypes == "O"]
    cat_cols = cat_cols + num_but_cat
    cat_cols = [col for col in cat_cols if col not in cat_but_car]
    num_cols = [col for col in dataframe.columns if dataframe[col].dtypes != "O" and col not in num_but_cat]

    print(f"Observations: {dataframe.shape[0]}")
    print(f"Variables: {dataframe.shape[1]}")
    print(f"cat_cols: {len(cat_cols)}")
    print(f"num_cols: {len(num_cols)}")
    print(f"cat_but_car: {len(cat_but_car)}")
    print(f"num_but_cat: {len(num_but_cat)}")

    return cat_cols, num_cols, cat_but_car



cat_cols, num_cols, cat_but_car = grab_col_names(df)
num_cols = [col for col in num_cols if col not in "PassengerId"]
print(num_cols)  #  ['Age', 'Fare']

for col in num_cols:
    print(col, check_outlier(df, col))
```

```
Observations: 891
Variables: 12
cat_cols: 6
num_cols: 3
cat_but_car: 3
num_but_cat: 4
['Age', 'Fare']
Age True
Fare True
```

```
df = load()
low, up = outlier_thresholds(df, "Fare")
print(df.shape) # (891, 12)
print(df[~((df["Fare"] < low) | (df["Fare"] > up))].shape) #(775,12)

def remove_outlier(dataframe, col_name):
    low_limit, up_limit = outlier_thresholds(dataframe, col_name)
    df_without_outliers = dataframe[~((dataframe[col_name] < low_limit) | (dataframe[col_name] > up_limit))]
    return df_without_outliers

cat_cols, num_cols, cat_but_car = grab_col_names(df)
```

```
(891, 12)
(775, 12)
Observations: 891
Variables: 12
cat_cols: 6
num_cols: 3
cat_but_car: 3
num_but_cat: 4
```

```
num_cols.remove('PassengerId')
for col in num_cols:
    df = remove_outlier(df,col)
print(df.shape) # (765,12)
```

```
(765, 12)
```

```
def replace_with_thresholds(dataframe, variable):
    low_limit, up_limit = outlier_thresholds(dataframe, variable)
    dataframe.loc[(dataframe[variable] < low_limit), variable] = low_limit
    dataframe.loc[(dataframe[variable] > up_limit), variable] = up_limit
df = load()

cat_cols, num_cols, cat_but_car = grab_col_names(df)
```

```
Observations: 891
Variables: 12
cat_cols: 6
num_cols: 3
cat_but_car: 3
num_but_cat: 4
```

```
num_cols.remove('PassengerId')
for col in num_cols:
    print(col, check_outlier(df, col))
```

```
Age True
Fare True
```

```
for col in num_cols:
    replace_with_thresholds(df, col)
for col in num_cols:
    print(col, check_outlier(df, col))
```

```
Age False
Fare False
```

```
df = sns.load_dataset('diamonds')
print(df.shape)  #  (53940, 10)
print(df.head())
```

```
(53940, 10)
   carat      cut color clarity  depth  table  price     x     y     z
0   0.23    Ideal     E     SI2   61.5   55.0    326  3.95  3.98  2.43
1   0.21  Premium     E     SI1   59.8   61.0    326  3.89  3.84  2.31
2   0.23     Good     E     VS1   56.9   65.0    327  4.05  4.07  2.31
3   0.29  Premium     I     VS2   62.4   58.0    334  4.20  4.23  2.63
4   0.31     Good     J     SI2   63.3   58.0    335  4.34  4.35  2.75
```

```
df = df.select_dtypes(include=['float64', 'int64'])
df = df.dropna()
print(df.shape)   #  (53940, 7)
print(df.head())
```

```
(53940, 7)
   carat  depth  table  price     x     y     z
0   0.23   61.5   55.0    326  3.95  3.98  2.43
1   0.21   59.8   61.0    326  3.89  3.84  2.31
2   0.23   56.9   65.0    327  4.05  4.07  2.31
3   0.29   62.4   58.0    334  4.20  4.23  2.63
4   0.31   63.3   58.0    335  4.34  4.35  2.75
```

```
for col in df.columns:
    print(col, check_outlier(df, col))
```

```
carat True
depth True
table True
price True
x True
y True
z True
```

```
low, up = outlier_thresholds(df, "carat")
print(df[((df["carat"] < low) | (df["carat"] > up))].shape)  # (1889, 7)

low, up = outlier_thresholds(df, "depth")
print(df[((df["depth"] < low) | (df["depth"] > up))].shape)  # (2545, 7)

clf = LocalOutlierFactor(n_neighbors=20)
clf.fit_predict(df)
df_scores = clf.negative_outlier_factor_
print(df_scores)

print(np.sort(df_scores)[0:5])
scores = pd.DataFrame(np.sort(df_scores))
scores.plot(stacked=True, xlim=[0, 20], style='.-')
plt.show()
```

```
(1889, 7)
(2545, 7)
[-1.58352526 -1.59732899 -1.62278873 ... -1.06721815 -1.00809552
 -1.00849038]
[-8.60430658 -8.20889984 -5.86084355 -4.98415175 -4.81502092]
```