```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline

df = pd.read_csv("bank-additional-full.csv",delimiter=';')
df.rename(columns={'y':'deposit'}, inplace=True)

cat_cols = df.select_dtypes(include='object').columns
print(cat_cols)

num_cols = df.select_dtypes(exclude='object').columns
print(num_cols)
```

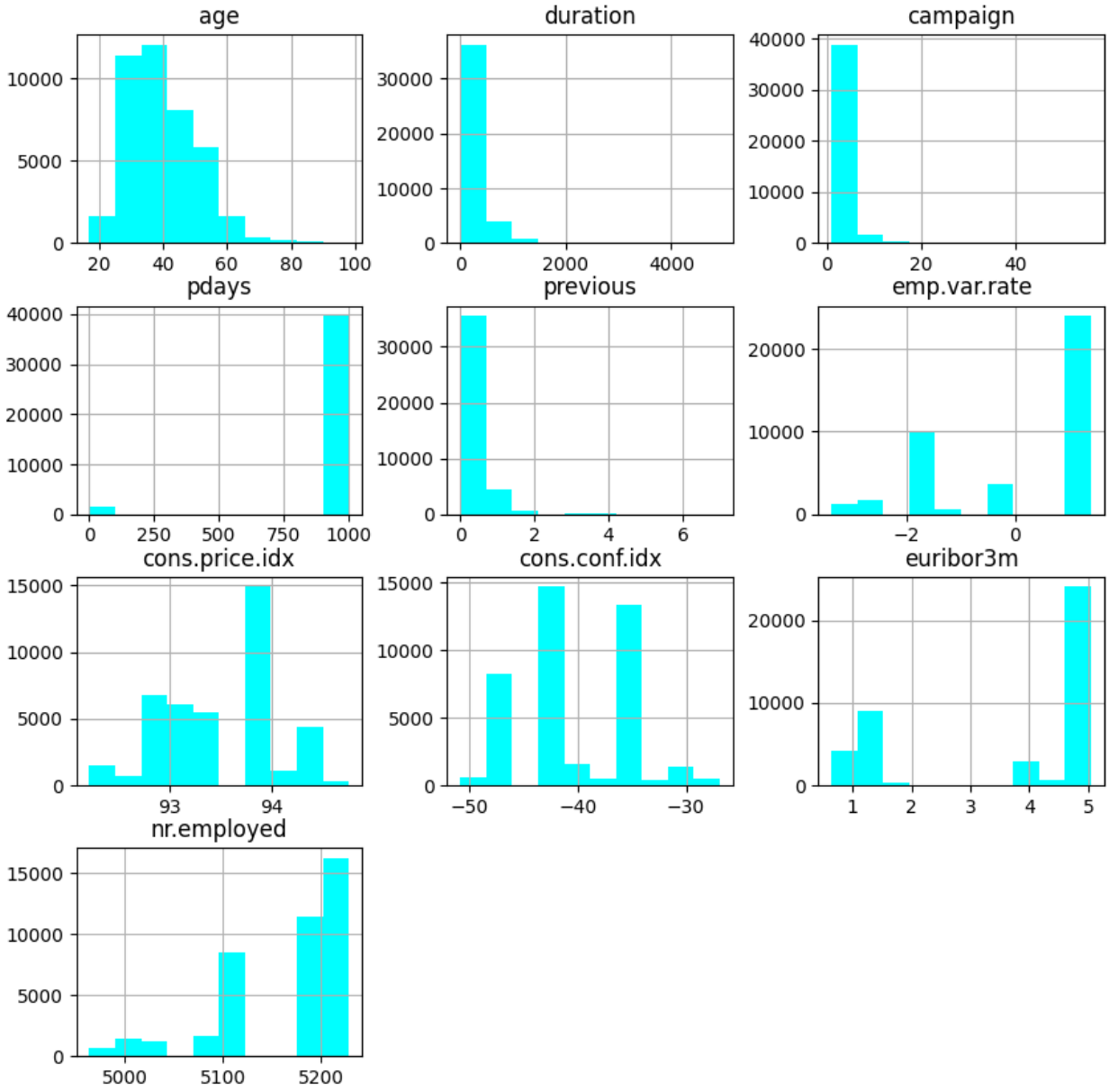
```
Index(['job', 'marital', 'education', 'default', 'housing', 'loan',
'contact',
       'month', 'day_of_week', 'poutcome', 'deposit'],
      dtype='object')
Index(['age', 'duration', 'campaign', 'pdays', 'previous',
'emp.var.rate',
       'cons.price.idx', 'cons.conf.idx', 'euribor3m', 'nr.employed'],
      dtype='object')
```

```
df.describe()
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 8,\n  \"fields\": [\n
{\n      \"column\": \"age\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 14547.915389437558,\n
\"min\": 10.421249980934048,\n        \"max\": 41188.0,\n
\"num_unique_values\": 8,\n        \"samples\": [\n
40.02406040594348,\n          38.0,\n          41188.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"duration\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
14353.429834977824,\n        \"min\": 0.0,\n        \"max\": 41188.0,\
n        \"num_unique_values\": 8,\n        \"samples\": [\n
258.2850101971448,\n          180.0,\n          41188.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"campaign\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
14558.717737799894,\n        \"min\": 1.0,\n        \"max\": 41188.0,\
n        \"num_unique_values\": 7,\n        \"samples\": [\n
41188.0,\n          2.567592502670681,\n          3.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"pdays\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 14308.123597767704,\
n        \"min\": 0.0,\n        \"max\": 41188.0,\n
```

\"num_unique_values\": 5,\n   \"samples\": [\n
962.4754540157328,\n   999.0,\n   186.9109073447418\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"previous\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
14561.769966637723,\n   \"min\": 0.0,\n   \"max\": 41188.0,\
n   \"num_unique_values\": 5,\n   \"samples\": [\n
0.17296299893172767,\n   7.0,\n   0.4949010798392897\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"emp.var.rate\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
14562.139340189238,\n   \"min\": -3.4,\n   \"max\":
41188.0,\n   \"num_unique_values\": 7,\n   \"samples\": [\n
41188.0,\n   0.08188550063125165,\n   1.1\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"cons.price.idx\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
14533.81120862695,\n   \"min\": 0.5788400489541262,\n
\"max\": 41188.0,\n   \"num_unique_values\": 8,\n
\"samples\": [\n   93.57566436826262,\n   93.749,\n
41188.0\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"cons.conf.idx\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 14574.009636834653,\n   \"min\": -
50.8,\n   \"max\": 41188.0,\n   \"num_unique_values\": 8,\n
\"samples\": [\n   -40.50260027192386,\n   -41.8,\n
41188.0\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"euribor3m\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 14561.036053130754,\n   \"min\":
0.634,\n   \"max\": 41188.0,\n   \"num_unique_values\": 8,\n
\"samples\": [\n   3.621290812858114,\n   4.857,\n
41188.0\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"nr.employed\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 13119.9618027212,\n   \"min\":
72.25152766826108,\n   \"max\": 41188.0,\n
\"num_unique_values\": 7,\n   \"samples\": [\n   41188.0,\
n   5167.035910944936,\n   5191.0\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe"}

```
df.hist(figsize=(10,10),color='#00FFFF')
plt.show()
```



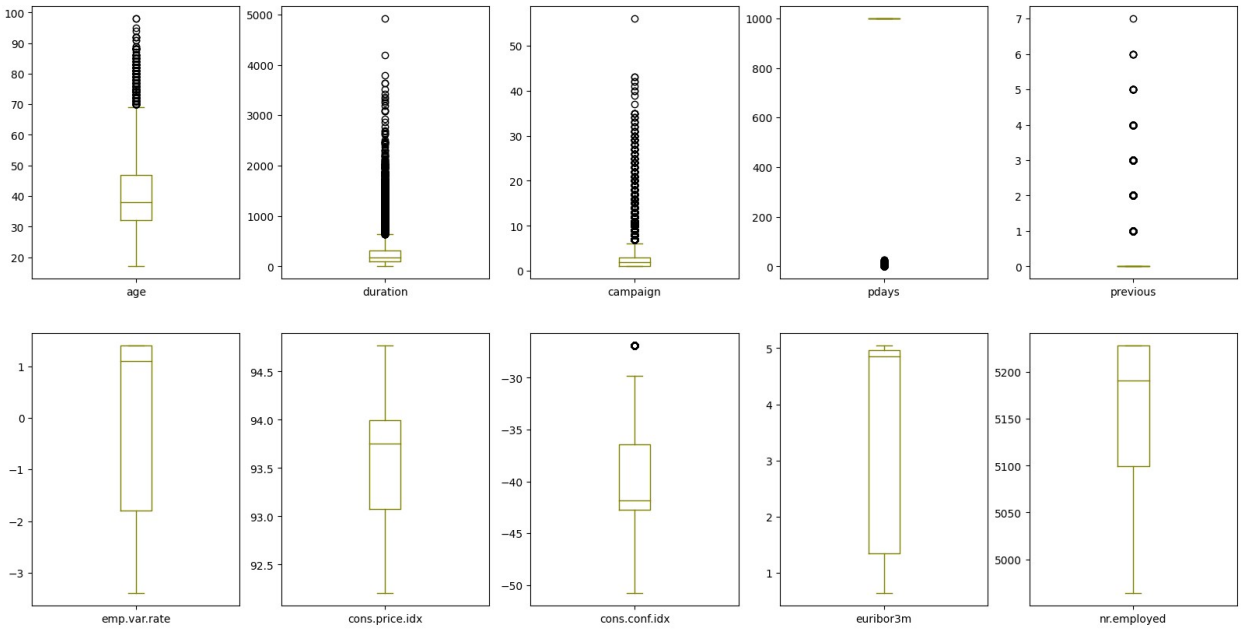

```
df.head()
```

```
{"type":"dataframe","variable_name":"df"}
```

```
df.tail()
```

```
{"type":"dataframe"}
```

```
df.plot(kind='box', subplots=True,
layout=(2,5),figsize=(20,10),color='#808000')
plt.show()
```

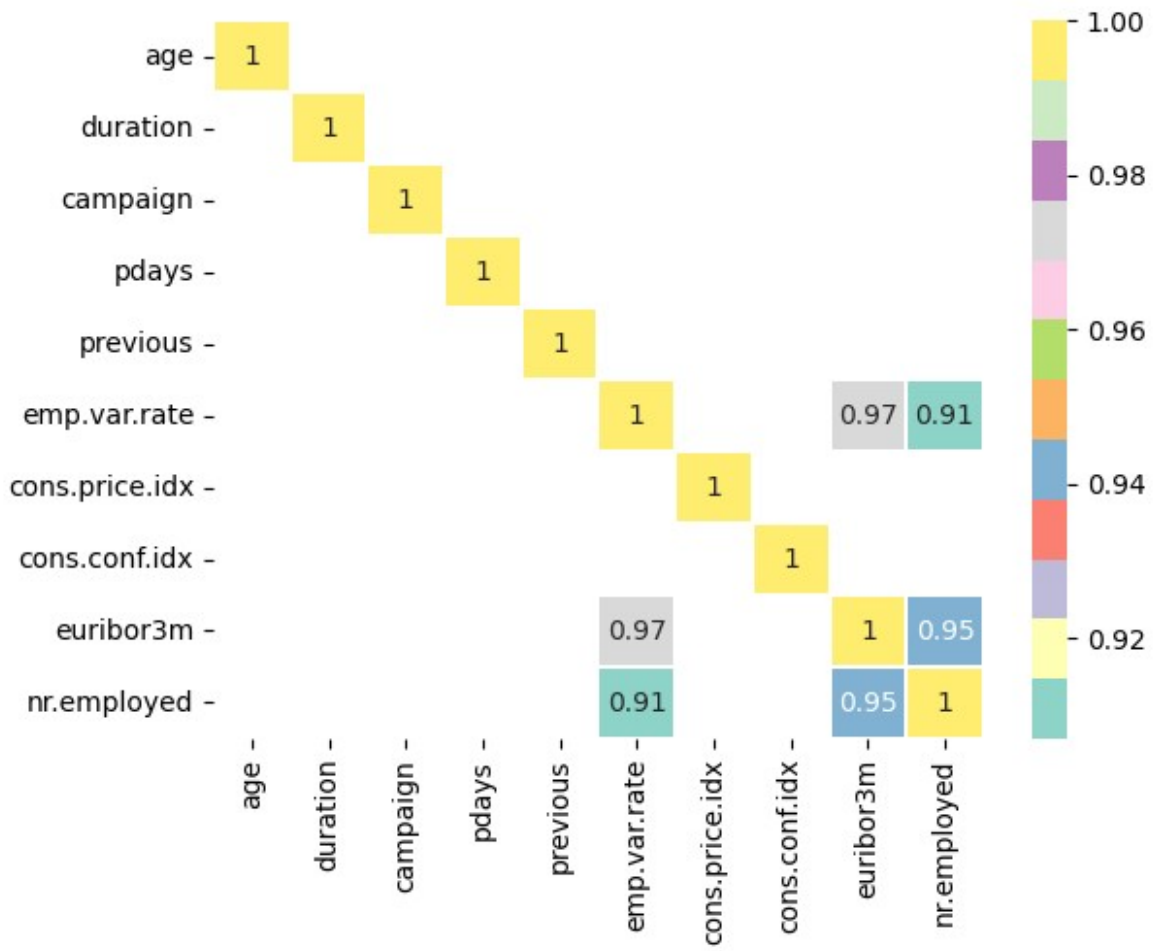



```
# Exclude non-numeric columns
numeric_df = df.drop(columns=cat_cols)

# Compute the correlation matrix
corr = numeric_df.corr()

# Print the correlation matrix
print(corr)

# Filter correlations with absolute value >= 0.90
corr = corr[abs(corr) >= 0.90]

sns.heatmap(corr,annot=True,cmap='Set3',linewidths=0.2)
plt.show()
```

```
                      age  duration  campaign     pdays  previous  \
age              1.000000 -0.000866  0.004594 -0.034369  0.024365
duration        -0.000866  1.000000 -0.071699 -0.047577  0.020640
campaign         0.004594 -0.071699  1.000000  0.052584 -0.079141
pdays           -0.034369 -0.047577  0.052584  1.000000 -0.587514
previous         0.024365  0.020640 -0.079141 -0.587514  1.000000
emp.var.rate    -0.000371 -0.027968  0.150754  0.271004 -0.420489
cons.price.idx   0.000857  0.005312  0.127836  0.078889 -0.203130
cons.conf.idx    0.129372 -0.008173 -0.013733 -0.091342 -0.050936
euribor3m        0.010767 -0.032897  0.135133  0.296899 -0.454494
nr.employed     -0.017725 -0.044703  0.144095  0.372605 -0.501333

                 emp.var.rate  cons.price.idx  cons.conf.idx  euribor3m
\
age                -0.000371        0.000857       0.129372   0.010767
```

```
duration            -0.027968        0.005312      -0.008173  -0.032897

campaign             0.150754        0.127836      -0.013733   0.135133

pdays                0.271004        0.078889      -0.091342   0.296899

previous            -0.420489       -0.203130      -0.050936  -0.454494

emp.var.rate         1.000000        0.775334       0.196041   0.972245

cons.price.idx       0.775334        1.000000       0.058986   0.688230

cons.conf.idx        0.196041        0.058986       1.000000   0.277686

euribor3m            0.972245        0.688230       0.277686   1.000000

nr.employed          0.906970        0.522034       0.100513   0.945154
```

```
                nr.employed
age               -0.017725
duration          -0.044703
campaign           0.144095
pdays              0.372605
previous          -0.501333
emp.var.rate       0.906970
cons.price.idx     0.522034
cons.conf.idx      0.100513
euribor3m          0.945154
nr.employed        1.000000
```




```
df1 = df.copy()

from sklearn.preprocessing import LabelEncoder
lb = LabelEncoder()
df_encoded = df1.apply(lb.fit_transform)
df_encoded
```

{"type":"dataframe","variable_name":"df_encoded"}

```
X = df_encoded.drop('deposit',axis=1)  # independent variable
y = df_encoded['deposit']              # dependent variable
print(X.shape)
print(y.shape)
print(type(X))
print(type(y))
```

```
(41188, 20)
(41188,)
<class 'pandas.core.frame.DataFrame'>
<class 'pandas.core.series.Series'>
```

```
from sklearn.model_selection import train_test_split

X_train,X_test,y_train,y_test =
train_test_split(X,y,test_size=0.25,random_state=1)
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(30891, 20)
(10297, 20)
(30891,)
(10297,)
```

```
from sklearn.tree import DecisionTreeClassifier

dt =
DecisionTreeClassifier(criterion='gini',max_depth=5,min_samples_split=
10)
dt.fit(X_train,y_train)
```

```
DecisionTreeClassifier(max_depth=5, min_samples_split=10)
```

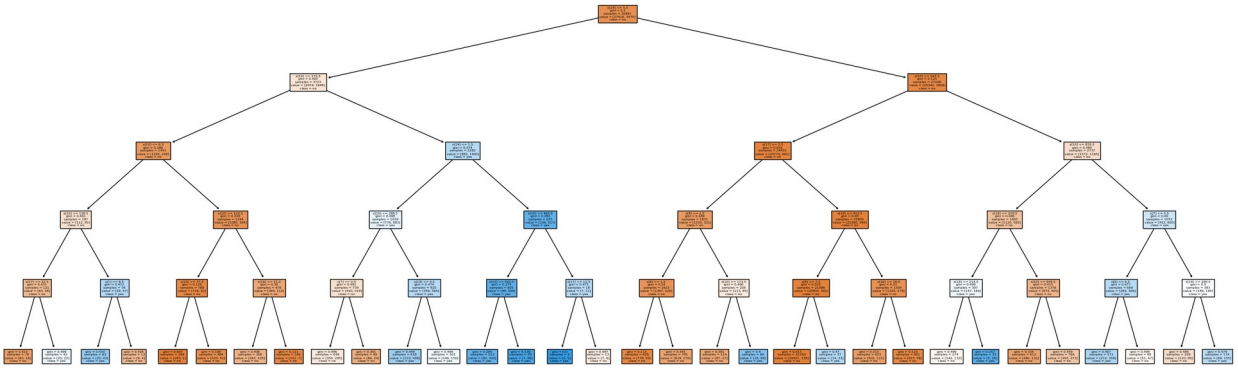
```
ypred_dt = dt.predict(X_test)
print(ypred_dt)
```

```
[0 0 0 ... 0 0 1]
```

```
from sklearn.tree import plot_tree

cn = ['no','yes']
fn = X_train.columns
print(fn)
print(cn)
```

```
Index(['age', 'job', 'marital', 'education', 'default', 'housing',
'loan',
       'contact', 'month', 'day_of_week', 'duration', 'campaign',
'pdays',
       'previous', 'poutcome', 'emp.var.rate', 'cons.price.idx',
       'cons.conf.idx', 'euribor3m', 'nr.employed'],
      dtype='object')
['no', 'yes']
```

```
plt.figure(figsize=(30,10))
plot_tree(dt,class_names=cn,filled=True)
plt.show()
```



```
dt1 =
DecisionTreeClassifier(criterion='entropy',max_depth=4,min_samples_spl
it=15)
dt1.fit(X_train,y_train)
```

```
DecisionTreeClassifier(criterion='entropy', max_depth=4,
min_samples_split=15)
```

```
plt.figure(figsize=(40,20))
plot_tree(dt1,class_names=cn,filled=True)
plt.show()
```