jupyter Random Forest Tutorial Last Checkpoint: Last Sunday at 1:40 PM (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

In [7]:
```
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
```

In [8]:
```
bank = pd.read_csv("C:/Users/amandeep/Downloads/bank.csv")
```

In [9]:
```
bank.head()
```

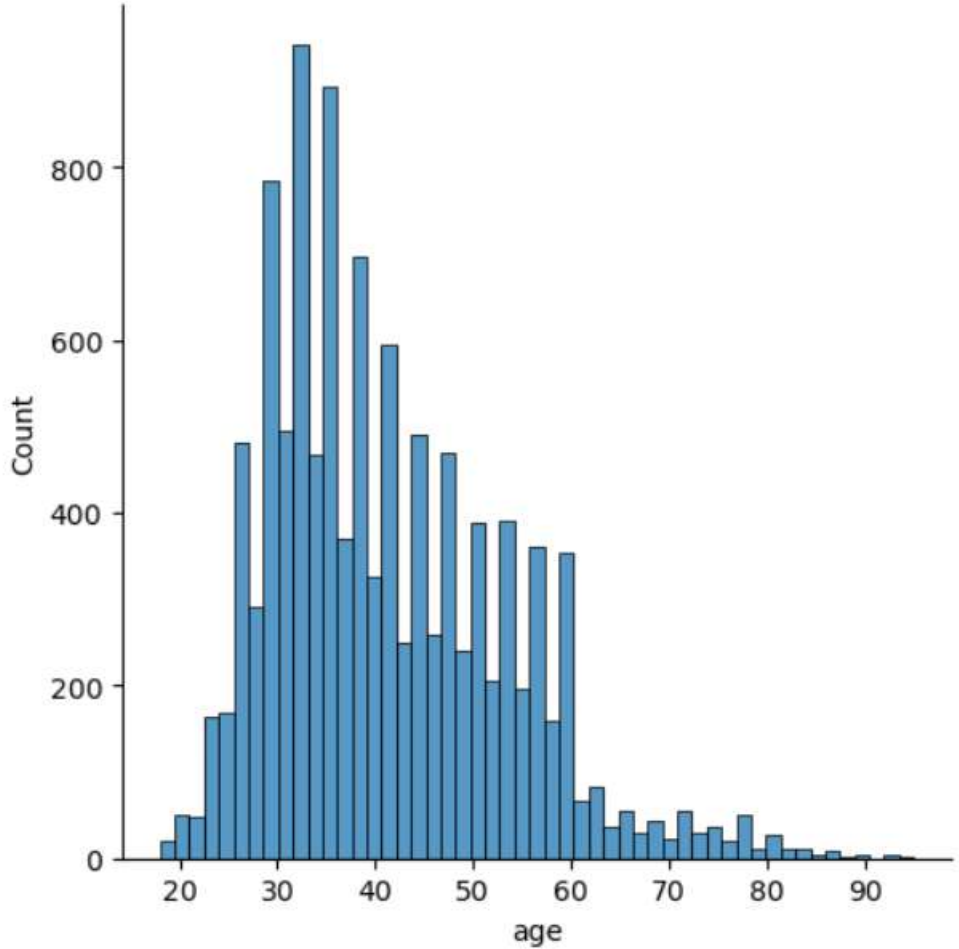
Out[9]:

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	deposit
0	59	admin.	married	secondary	no	2343	yes	no	unknown	5	may	1042	1	-1	0	unknown	yes
1	56	admin.	married	secondary	no	45	no	no	unknown	5	may	1467	1	-1	0	unknown	yes
2	41	technician	married	secondary	no	1270	yes	no	unknown	5	may	1389	1	-1	0	unknown	yes
3	55	services	married	secondary	no	2476	yes	no	unknown	5	may	579	1	-1	0	unknown	yes
4	54	admin.	married	tertiary	no	184	no	no	unknown	5	may	673	2	-1	0	unknown	yes

In [10]:
```
bank.shape
```

Out[10]: (11162, 17)

In [13]:
```
sns.displot(bank['age'])
plt.show()
```



In [14]:
```
bank.describe()
```

Out[14]:

	age	balance	day	duration	campaign	pdays	previous
count	11162.000000	11162.000000	11162.000000	11162.000000	11162.000000	11162.000000	11162.000000
mean	41.231948	1528.538524	15.658036	371.993818	2.508421	51.330407	0.832557
std	11.913369	3225.413326	8.420740	347.128386	2.722077	108.758282	2.292007
min	18.000000	-6847.000000	1.000000	2.000000	1.000000	-1.000000	0.000000
25%	32.000000	122.000000	8.000000	138.000000	1.000000	-1.000000	0.000000
50%	39.000000	550.000000	15.000000	255.000000	2.000000	-1.000000	0.000000
75%	49.000000	1708.000000	22.000000	496.000000	3.000000	20.750000	1.000000
max	95.000000	81204.000000	31.000000	3881.000000	63.000000	854.000000	58.000000

In [16]:
```
bank['job'].value_counts(),bank.shape
```

Out[16]:
```
(job
management       2566
blue-collar      1944
technician       1823
admin.           1334
services          923
retired           778
self-employed     405
student           360
unemployed        357
entrepreneur      328
housemaid         274
unknown            70
Name: count, dtype: int64,
(11162, 17))
```

In [17]:
```
bank['marital'].value_counts()
```

Out[17]:
```
marital
married     6351
single      3518
divorced    1293
Name: count, dtype: int64
```

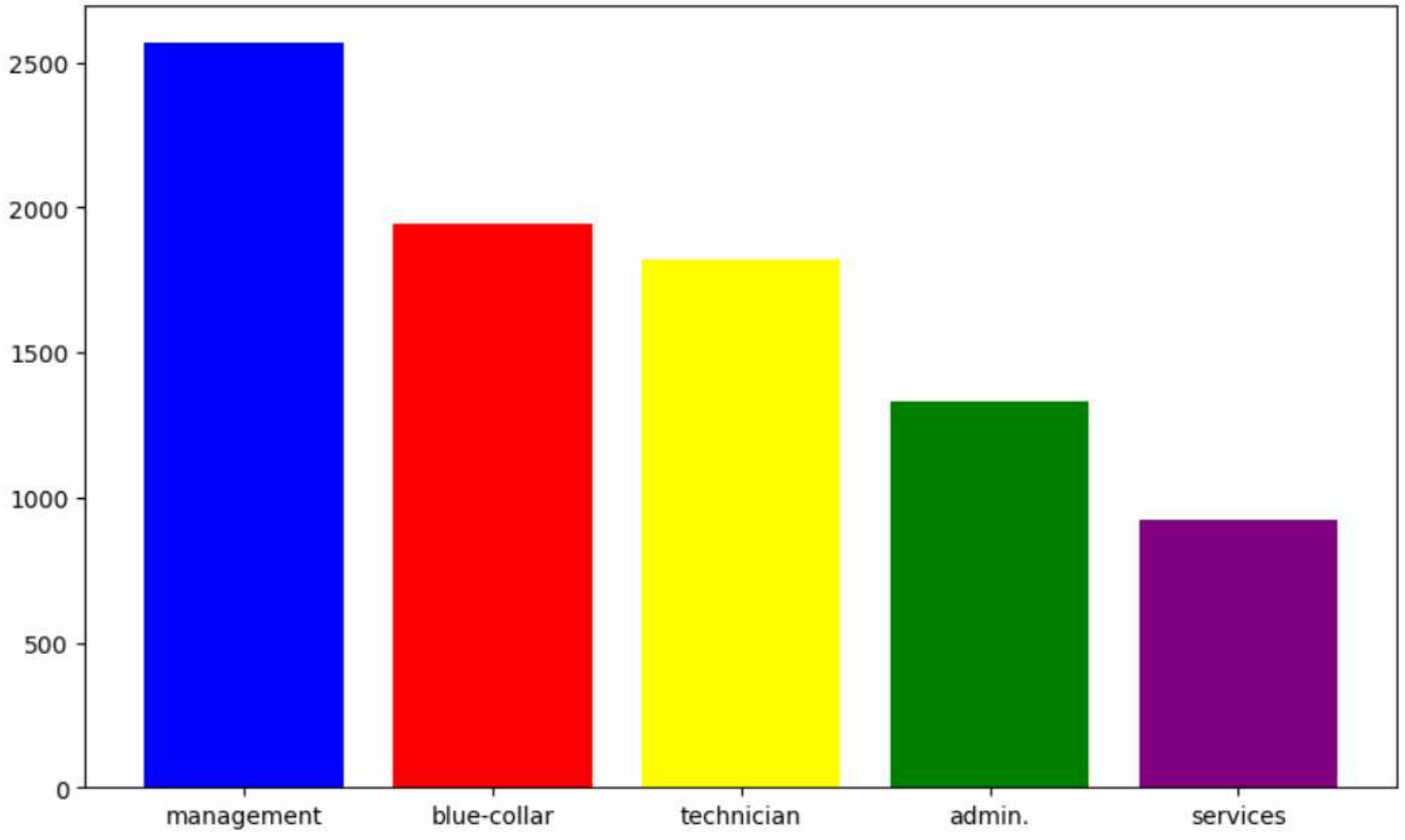
In [18]:
```
bank['job'].value_counts().keys()
```

Out[18]:
```
Index(['management', 'blue-collar', 'technician', 'admin.', 'services',
       'retired', 'self-employed', 'student', 'unemployed', 'entrepreneur',
       'housemaid', 'unknown'],
      dtype='object', name='job')
```

In [19]:
```
bank['job'].value_counts().values
```

Out[19]:
```
array([2566, 1944, 1823, 1334,  923,  778,  405,  360,  357,  328,  274,
         70], dtype=int64)
```

In [31]:
```
plt.figure(figsize=(10,6))
plt.bar(list(bank['job'].value_counts().keys()[0:5]),list(bank['job'].value_counts())[0:5],color=['blue','red','yellow','green','
plt.show()
```



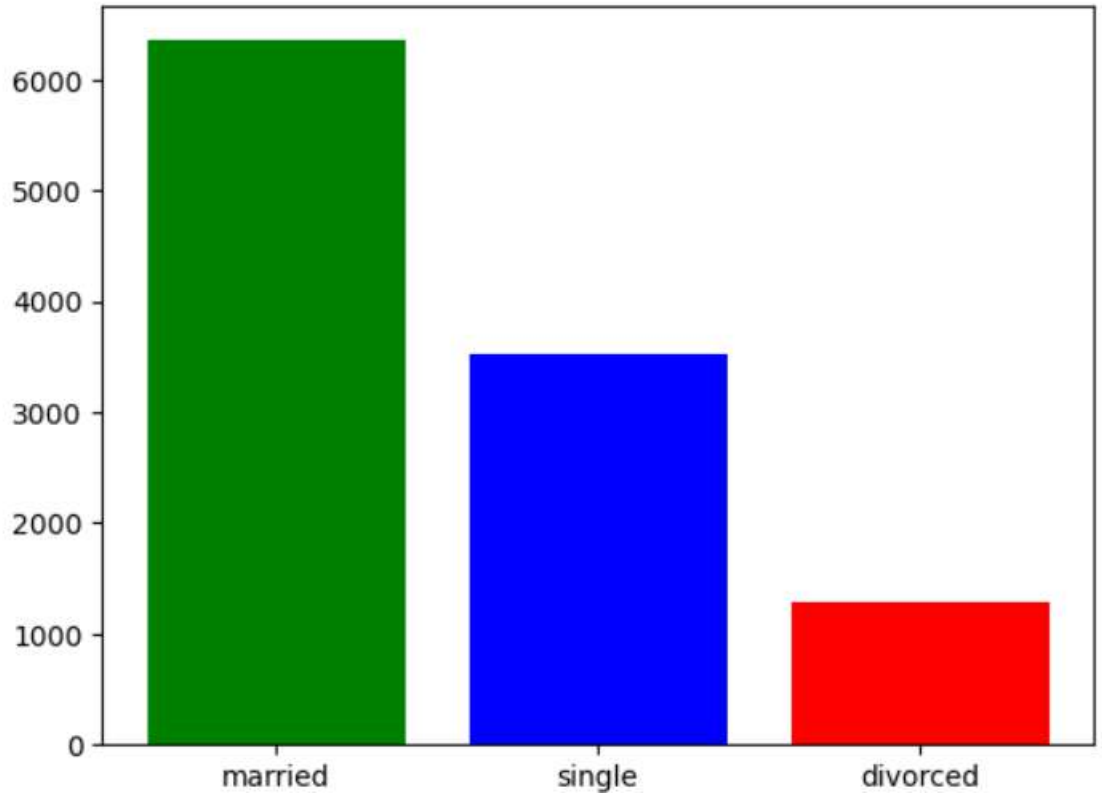
In [35]:
```
bank['marital'].value_counts().keys()
```

Out[35]:
```
Index(['married', 'single', 'divorced'], dtype='object', name='marital')
```

In [36]:
```
bank['marital'].value_counts().values
```

Out[36]:
```
array([6351, 3518, 1293], dtype=int64)
```

In [38]:
```
plt.bar(list(bank['marital'].value_counts().keys()),list(bank['marital'].value_counts()),color=['green','blue','red'])
plt.show()
```



In [39]:
```
bank.head()
```

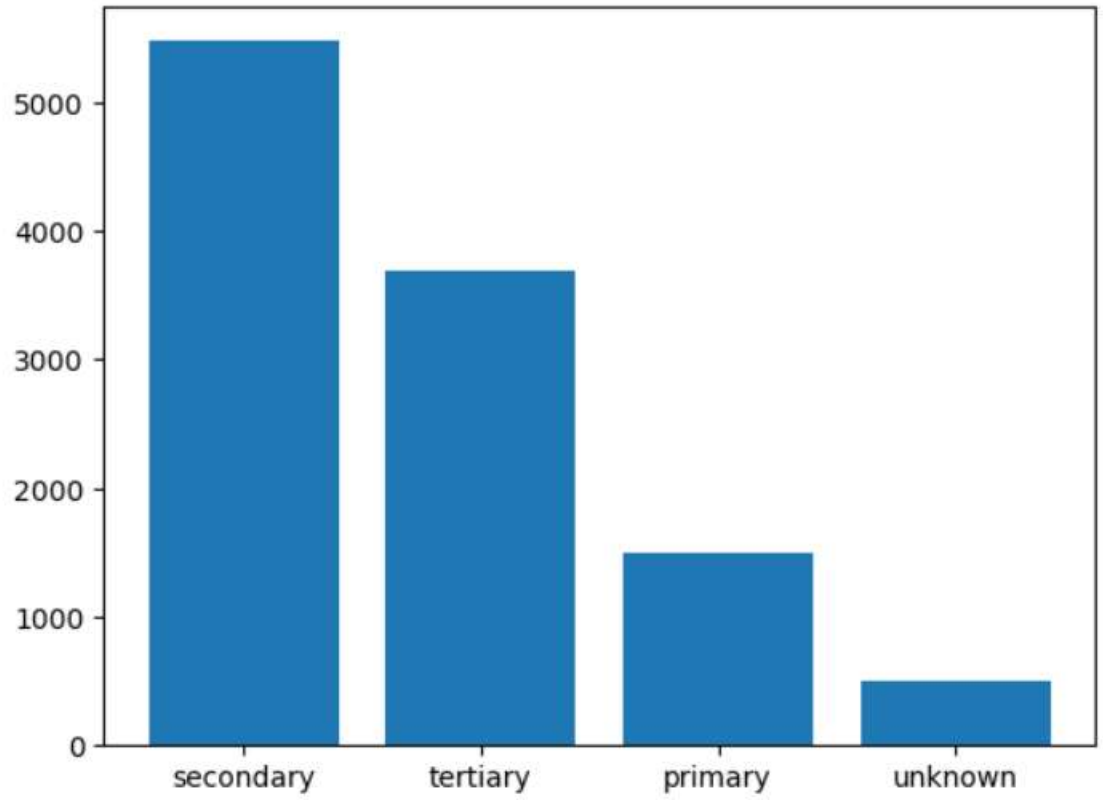
Out[39]:

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	deposit
0	59	admin.	married	secondary	no	2343	yes	no	unknown	5	may	1042	1	-1	0	unknown	yes
1	56	admin.	married	secondary	no	45	no	no	unknown	5	may	1467	1	-1	0	unknown	yes
2	41	technician	married	secondary	no	1270	yes	no	unknown	5	may	1389	1	-1	0	unknown	yes
3	55	services	married	secondary	no	2476	yes	no	unknown	5	may	579	1	-1	0	unknown	yes
4	54	admin.	married	tertiary	no	184	no	no	unknown	5	may	673	2	-1	0	unknown	yes

In [40]:
```
bank['education'].value_counts()
```

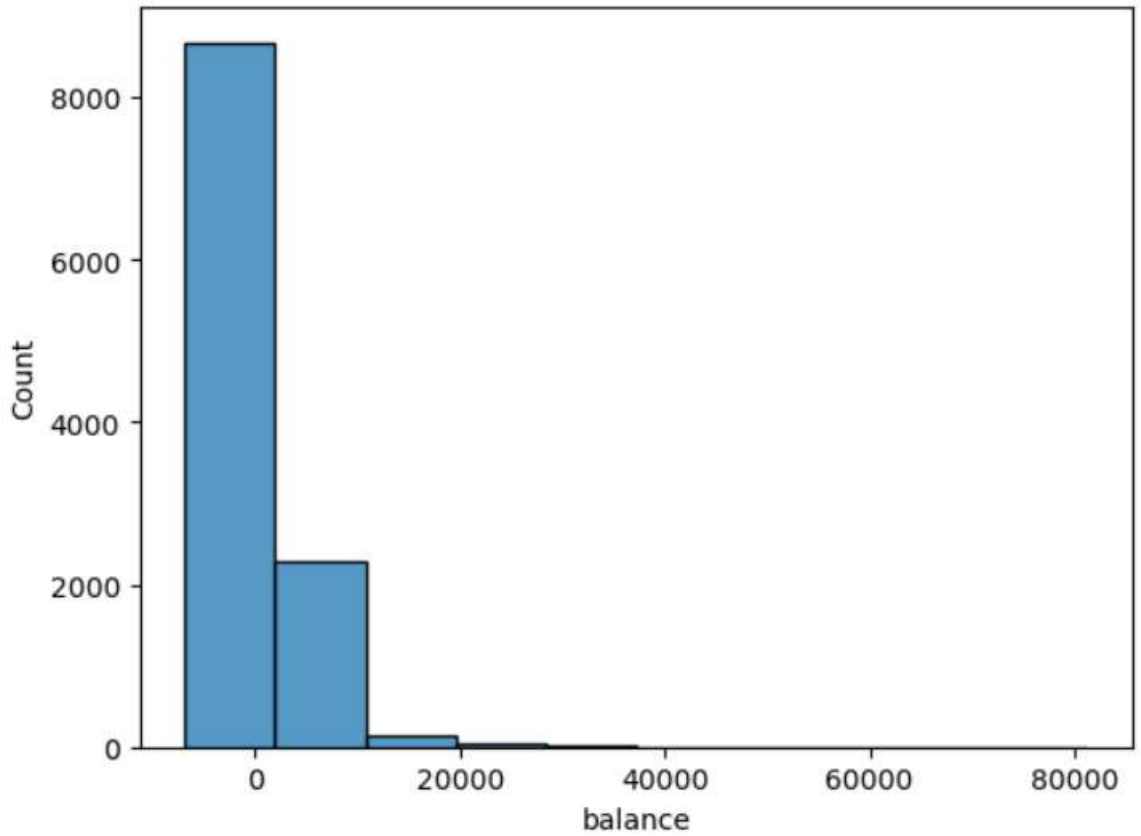
Out[40]:
```
education
secondary    5476
tertiary     3689
primary      1500
unknown       497
Name: count, dtype: int64
```

In [41]:
```
plt.bar(list(bank['education'].value_counts().keys()),list(bank['education'].value_counts()))
plt.show()
```



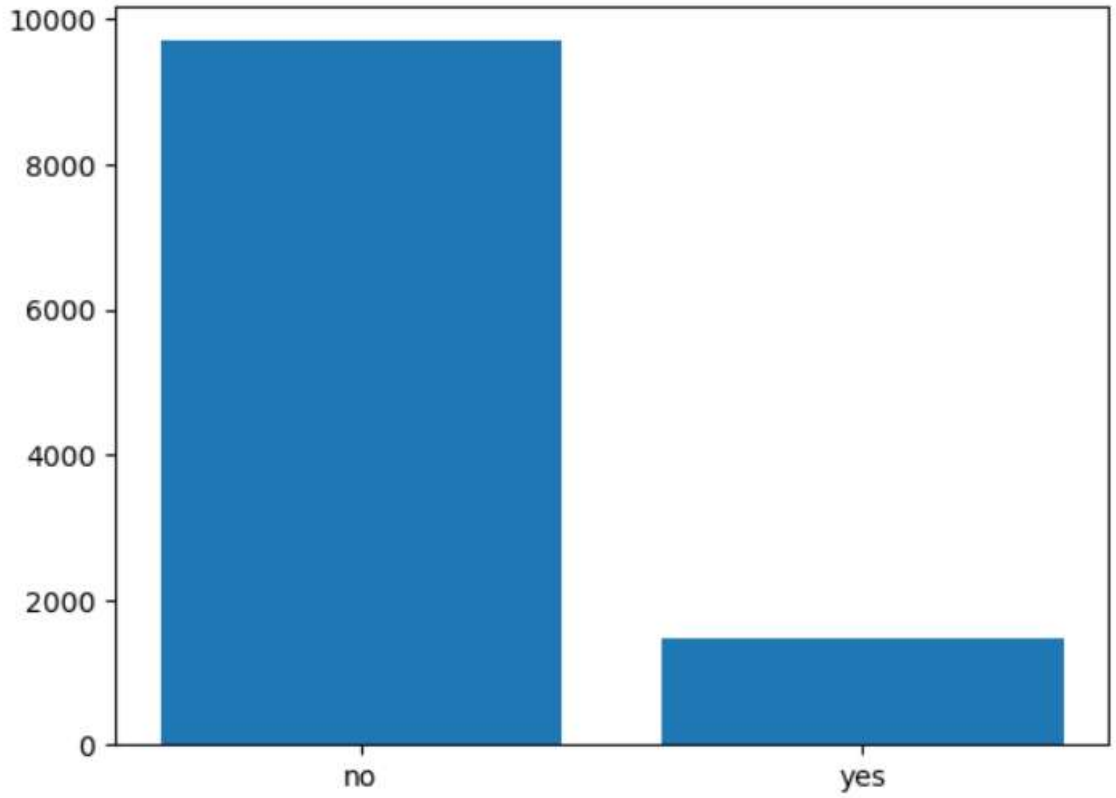
In [47]:
```
sns.histplot(bank['balance'],bins=10)
```

Out[47]: <Axes: xlabel='balance', ylabel='Count'>



In [51]:
```
plt.bar(list(bank['loan'].value_counts().keys()),list(bank['loan'].value_counts()))
```

Out[51]: <BarContainer object of 2 artists>



In [76]:
```
x = bank['age']
y = bank['balance']
```

In [80]:
```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor

# Perform train-test split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3)

# Initialize RandomForestRegressor
rfg = RandomForestRegressor()

# Fit the model
rfg.fit(x_train.values.reshape(-1, 1), y_train)
```

Out[80]:

▾ RandomForestRegressor

RandomForestRegressor()

In [83]:
```
x_test = np.array(x_test)

# Reshape the input array if it is 1D
if x_test.ndim == 1:
    x_test = x_test.reshape(-1, 1)

# Now, predict using the RandomForestRegressor
y_pred = rfg.predict(x_test)
```

In [84]:
```
y_test.head(),y_pred[0:5]
```

Out[84]:
```
(1003     6993
 1425    12039
 9887     4545
 1446     1819
 5450      493
 Name: balance, dtype: int64,
 array([1664.70068805, 2297.07928629,  983.14347157, 1120.94984494,
        937.95351084]))
```

In [85]:
```
from sklearn.metrics import mean_squared_error
```

In [86]:
```
mean_squared_error(y_test,y_pred)
```

Out[86]:
```
9967591.783513248
```

In [ ]:
```
#Thank you
```