Financial Inclusion in Africa complete

June 10, 2020

1 Imorting the Necessary Library

[25]:
```
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn import model_selection, preprocessing
from sklearn.metrics import classification_report
```

2 Imorting our Data

[26]:
```
train = pd.read_csv('Train_v2.csv')
```

[27]:
```
train.head()
```

[27]:
```
  country  year    uniqueid bank_account location_type cellphone_access  \
0   Kenya  2018  uniqueid_1          Yes         Rural              Yes
1   Kenya  2018  uniqueid_2           No         Rural               No
2   Kenya  2018  uniqueid_3          Yes         Urban              Yes
3   Kenya  2018  uniqueid_4           No         Rural              Yes
4   Kenya  2018  uniqueid_5           No         Urban               No

   household_size  age_of_respondent gender_of_respondent  \
0               3                 24               Female
1               5                 70               Female
2               5                 26                 Male
3               5                 34               Female
4               8                 26                 Male

  relationship_with_head           marital_status  \
0                 Spouse  Married/Living together
1      Head of Household                  Widowed
2         Other relative     Single/Never Married
3      Head of Household  Married/Living together
4                  Child     Single/Never Married
```

```
                education_level                  job_type
0           Secondary education             Self employed
1            No formal education      Government Dependent
2  Vocational/Specialised training           Self employed
3             Primary education  Formally employed Private
4             Primary education       Informally employed
```

3 Exploratory Data Analysis

[28]:
```
train.columns
```

[28]:
```
Index(['country', 'year', 'uniqueid', 'bank_account', 'location_type',
       'cellphone_access', 'household_size', 'age_of_respondent',
       'gender_of_respondent', 'relationship_with_head', 'marital_status',
       'education_level', 'job_type'],
      dtype='object')
```

[29]:
```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23524 entries, 0 to 23523
Data columns (total 13 columns):
country                   23524 non-null object
year                      23524 non-null int64
uniqueid                  23524 non-null object
bank_account              23524 non-null object
location_type             23524 non-null object
cellphone_access          23524 non-null object
household_size            23524 non-null int64
age_of_respondent         23524 non-null int64
gender_of_respondent      23524 non-null object
relationship_with_head    23524 non-null object
marital_status            23524 non-null object
education_level           23524 non-null object
job_type                  23524 non-null object
dtypes: int64(3), object(10)
memory usage: 2.3+ MB
```

[30]:
```
train = train.drop_duplicates()
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 23524 entries, 0 to 23523
Data columns (total 13 columns):
country                   23524 non-null object
```

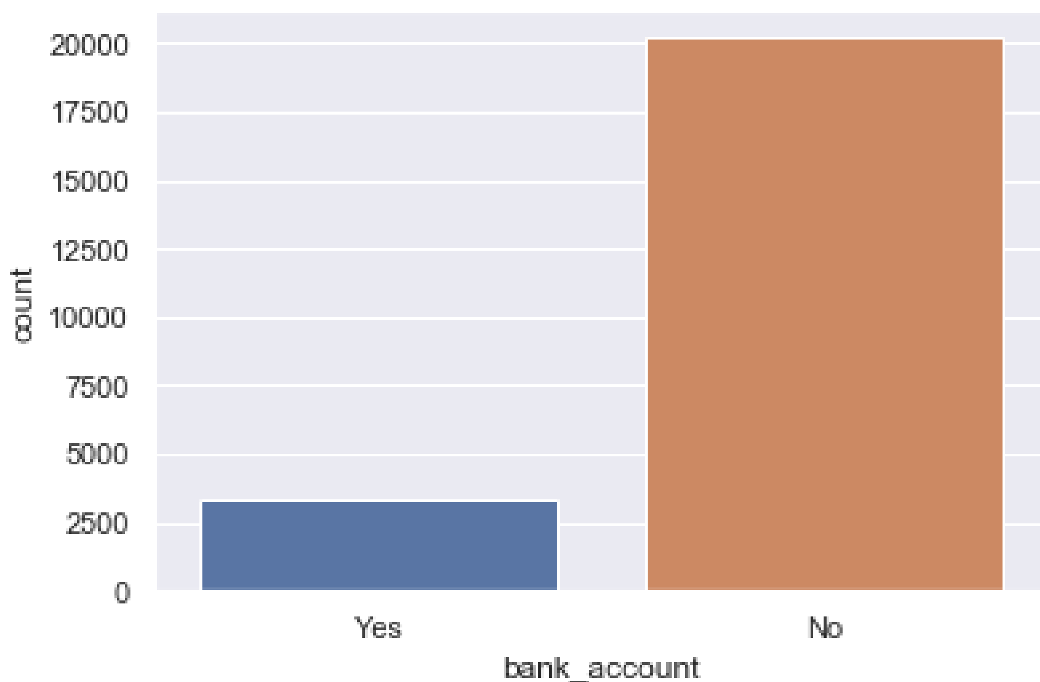
```
year                       23524 non-null int64
uniqueid                   23524 non-null object
bank_account               23524 non-null object
location_type              23524 non-null object
cellphone_access           23524 non-null object
household_size             23524 non-null int64
age_of_respondent          23524 non-null int64
gender_of_respondent       23524 non-null object
relationship_with_head     23524 non-null object
marital_status             23524 non-null object
education_level            23524 non-null object
job_type                   23524 non-null object
dtypes: int64(3), object(10)
memory usage: 2.5+ MB
```

4 Data Visualization

Let's use seaborn to explore the data!

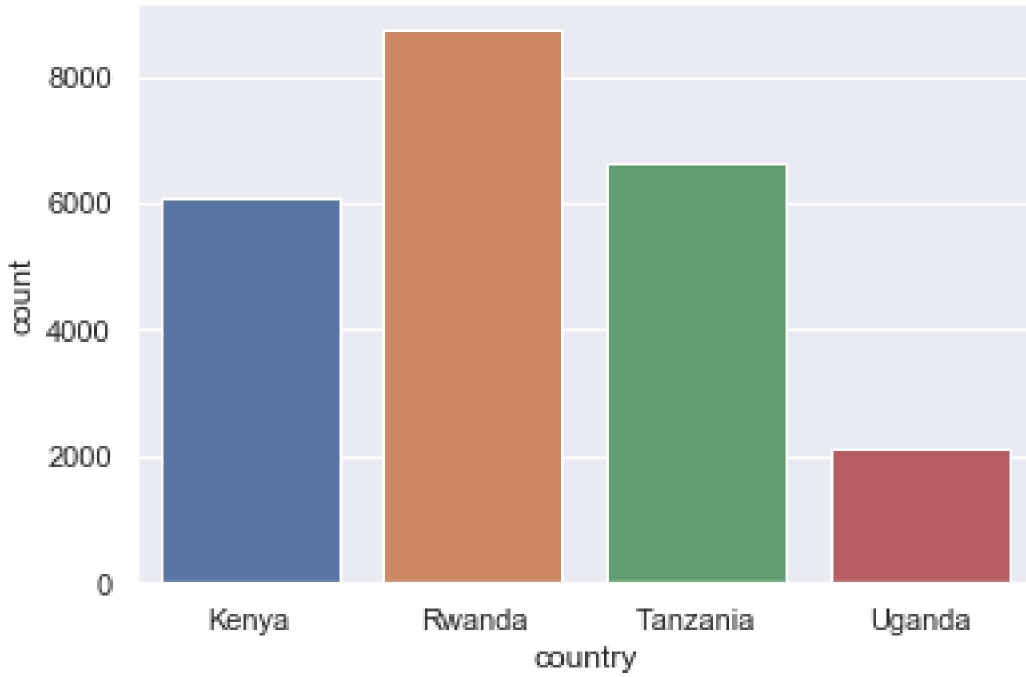
[31]:
```
sns.countplot(x = 'bank_account', data = train)
```

[31]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c547ea3e88>
```



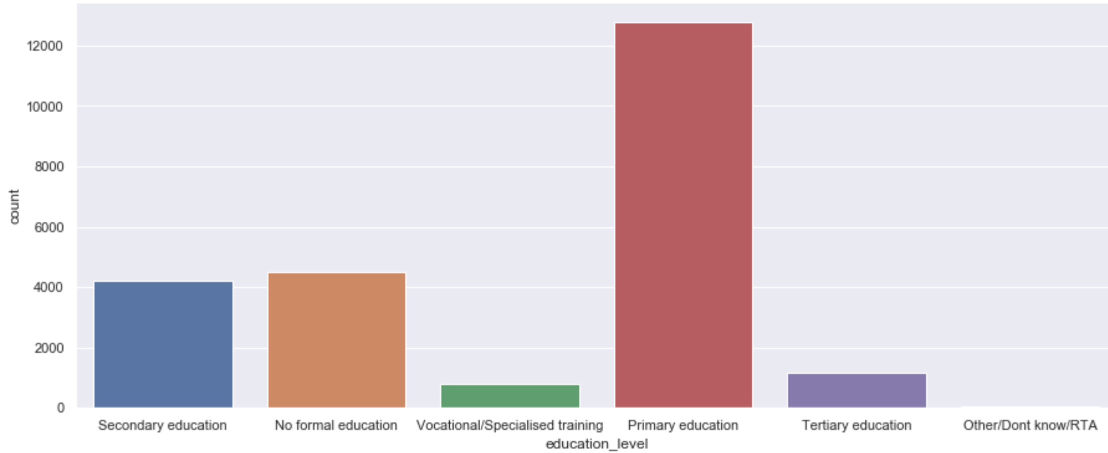
[32]:
```
sns.countplot(x = 'country', data = train)
```

[32]: <matplotlib.axes._subplots.AxesSubplot at 0x1c547fe1c48>



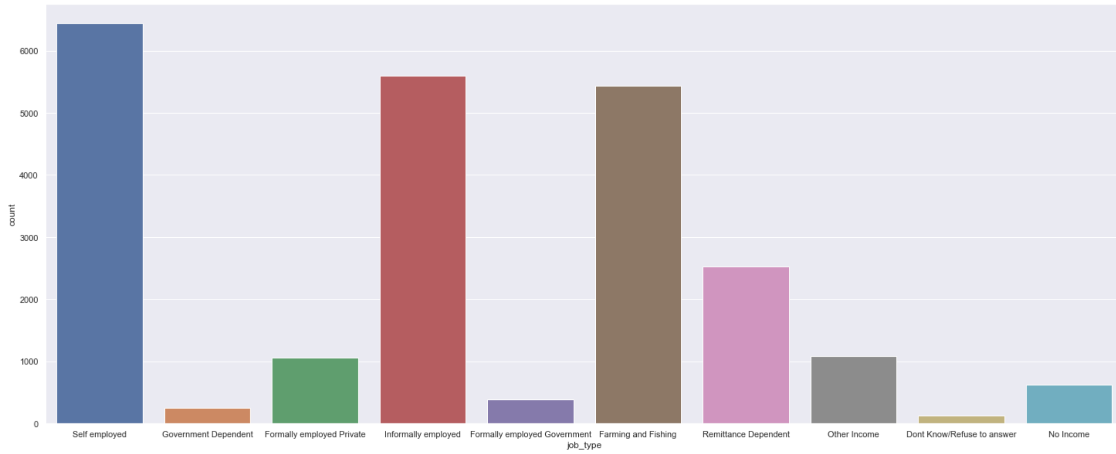
[33]:
```
plt.figure(figsize=[15,6])
sns.countplot(x = 'education_level', data = train)
```

[33]: <matplotlib.axes._subplots.AxesSubplot at 0x1c547fada88>



[34]:
```
plt.figure(figsize=[25,10])
sns.countplot(x = 'job_type', data = train)
```

[34]: <matplotlib.axes._subplots.AxesSubplot at 0x1c548b57f88>



5 Logistic Regression

Now it's time to do a train test split, and train our model!

[35]:
```
train_lr = pd.read_csv('Train_v2.csv')
```

[36]:
```
train_lr.head()
```

[36]:
```
  country  year    uniqueid bank_account location_type cellphone_access  \
0   Kenya  2018  uniqueid_1          Yes         Rural              Yes
1   Kenya  2018  uniqueid_2           No         Rural               No
2   Kenya  2018  uniqueid_3          Yes         Urban              Yes
3   Kenya  2018  uniqueid_4           No         Rural              Yes
4   Kenya  2018  uniqueid_5           No         Urban               No

   household_size  age_of_respondent gender_of_respondent  \
0               3                 24               Female
1               5                 70               Female
2               5                 26                 Male
3               5                 34               Female
4               8                 26                 Male

  relationship_with_head           marital_status  \
0                 Spouse  Married/Living together
1      Head of Household                  Widowed
2         Other relative     Single/Never Married
3      Head of Household  Married/Living together
4                  Child     Single/Never Married
```

```
                        education_level                     job_type
0                   Secondary education                Self employed
1                   No formal education         Government Dependent
2  Vocational/Specialised training                Self employed
3                     Primary education  Formally employed Private
4                     Primary education         Informally employed
```

[37]:
```python
for e in train_lr.columns:
    if train_lr[e].dtype == 'object':
        lbl = preprocessing.LabelEncoder()
        lbl.fit(list(train_lr[e].values))
        train_lr[e] = lbl.transform(list(train_lr[e].values))
```

[39]:
```python
train_lr.head()
```

[39]:
```
   country  year  uniqueid  bank_account  location_type  cellphone_access  \
0        0  2018         0             1              0                 1
1        0  2018      1111             0              0                 0
2        0  2018      2222             1              1                 1
3        0  2018      3333             0              0                 1
4        0  2018      4444             0              1                 0

   household_size  age_of_respondent  gender_of_respondent  \
0               3                 24                     0
1               5                 70                     0
2               5                 26                     1
3               5                 34                     0
4               8                 26                     1

   relationship_with_head  marital_status  education_level  job_type
0                       5               2                3         9
1                       1               4                0         4
2                       3               3                5         9
3                       1               2                2         3
4                       0               3                2         5
```

[49]:
```python
X = train_lr[['country', 'year', 'uniqueid','location_type',
       'cellphone_access', 'household_size', 'age_of_respondent',
       'gender_of_respondent', 'relationship_with_head', 'marital_status',
       'education_level', 'job_type']]
y = train['bank_account']
```

[50]:
```python
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,␣
 ↪random_state=42)
```

[51]:
```python
logmodel = LogisticRegression()
logmodel.fit(X_train,y_train)
```

```
C:\Users\Bona\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:432:
FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify a
solver to silence this warning.
  FutureWarning)
```

[51]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='warn', n_jobs=None, penalty='l2',
                   random_state=None, solver='warn', tol=0.0001, verbose=0,
                   warm_start=False)
```

6 Predictions and Evaluations

[52]:
```python
predictions = logmodel.predict(X_test)
```

[53]:
```python
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

           0       0.89      0.98      0.93      6678
           1       0.67      0.22      0.33      1085

    accuracy                           0.88      7763
   macro avg       0.78      0.60      0.63      7763
weighted avg       0.86      0.88      0.85      7763
```

[ ]:
```python

```