Main

December 5, 2018

1 Program overview

- Look at relationships between income & different attributes.
- Extract important features from a Random Forest Model.

```
In [1]: import os
        os.chdir(r"C:\Users\Arnold\OneDrive\R_Python_working_directory")
        import pandas as pd
        import numpy as np
        from plotnine import *
        %matplotlib inline
        (theme_update(plot_background = element_rect(fill = "gold"),panel_background = element_
        colour = "blue",size = 1.99,linetype = "solid"),plot_title = element_text(hjust = 0.5)
```

2 Read in data

```
In [2]: df=pd.read_csv('cencus income.csv',na_values='?')
```

3 Data preview

```
In [3]: df.head()
```

```
Out[3]:    age  workclass  fnlwgt     education  educational-num      marital-status  \
        0   25    Private  226802          11th                7       Never-married
        1   38    Private   89814       HS-grad                9  Married-civ-spouse
        2   28  Local-gov  336951    Assoc-acdm               12  Married-civ-spouse
        3   44    Private  160323  Some-college               10  Married-civ-spouse
        4   18        NaN  103497  Some-college               10       Never-married

                  occupation relationship   race  gender  capital-gain  capital-loss  \
        0  Machine-op-inspct    Own-child  Black    Male             0             0
        1    Farming-fishing      Husband  White    Male             0             0
        2    Protective-serv      Husband  White    Male             0             0
        3  Machine-op-inspct      Husband  Black    Male          7688             0
        4                NaN    Own-child  White  Female             0             0
```

```
   hours-per-week native-country income
0              40  United-States  <=50K
1              50  United-States  <=50K
2              40  United-States   >50K
3              40  United-States   >50K
4              30  United-States  <=50K
```

4 Data summary

In [4]:
```
df.describe().T
```

Out[4]:
```
                  count           mean            std      min       25%  \
age             48842.0      38.643585      13.710510     17.0      28.0
fnlwgt          48842.0  189664.134597  105604.025423  12285.0  117550.5
educational-num 48842.0      10.078089       2.570973      1.0       9.0
capital-gain    48842.0    1079.067626    7452.019058      0.0       0.0
capital-loss    48842.0      87.502314     403.004552      0.0       0.0
hours-per-week  48842.0      40.422382      12.391444      1.0      40.0

                      50%       75%        max
age                  37.0      48.0       90.0
fnlwgt           178144.5  237642.0  1490400.0
educational-num      10.0      12.0       16.0
capital-gain          0.0       0.0    99999.0
capital-loss          0.0       0.0     4356.0
hours-per-week       40.0      45.0       99.0
```

5 Check the shape

In [5]:
```
df.shape
```

Out[5]:
```
(48842, 15)
```

6 Check % of rows with missing values

In [6]:
```
np.mean(pd.isnull(df).any(axis=1))*100
```

Out[6]:
```
7.411653904426519
```

7% is low, so I'll drop the rows with missing values

7 Drop rows with missing values

In [7]:
```
df.dropna(inplace=1==1)
```

8 Age binning

```
In [8]: df.age=pd.cut(df.age,bins=[16,20,30,40,50,60,70,100])
```

9 Remove columns that I'm not going to use

```
In [9]: df.drop(columns=['relationship','capital-gain','capital-loss','hours-per-week'],
                inplace=1==1)
```

9.1 Function to look at the relationships between different attributes & income

9.1.1 This function only works with categorical variables.

```
In [10]: def relation(attr):
             tdf=df.pivot_table(values='fnlwgt',columns=['income'],index=attr,aggfunc=len,
                                margins=1==1,fill_value=0)
             tdf.iloc[:-1,0]=tdf.iloc[:-1,0].values/(tdf.All[:-1]).values
             tdf.iloc[:-1,1]=1-tdf.iloc[:-1,0].values
             tdf.sort_values(by=['>50K'],inplace=1==1,ascending=1==0)
             tdf.iloc[1:,:2]=tdf.iloc[:,:2].applymap(lambda x: round(x*100))
             tdf=tdf.iloc[1:,:-1].stack().reset_index()
             tdf.rename(columns={0:'%'},inplace=1==1)
             return tdf
         def rggplot(attr,size=8):
             graph=ggplot(tdf,aes(x=attr,y='%',fill=attr))+geom_bar(stat='identity',
                                    show_legend=1==0)+geom_text(aes(label='%',y=5),
             size=size,format_string='{}%')+coord_flip()+ggtitle('% of income > 50K')
             return graph
```

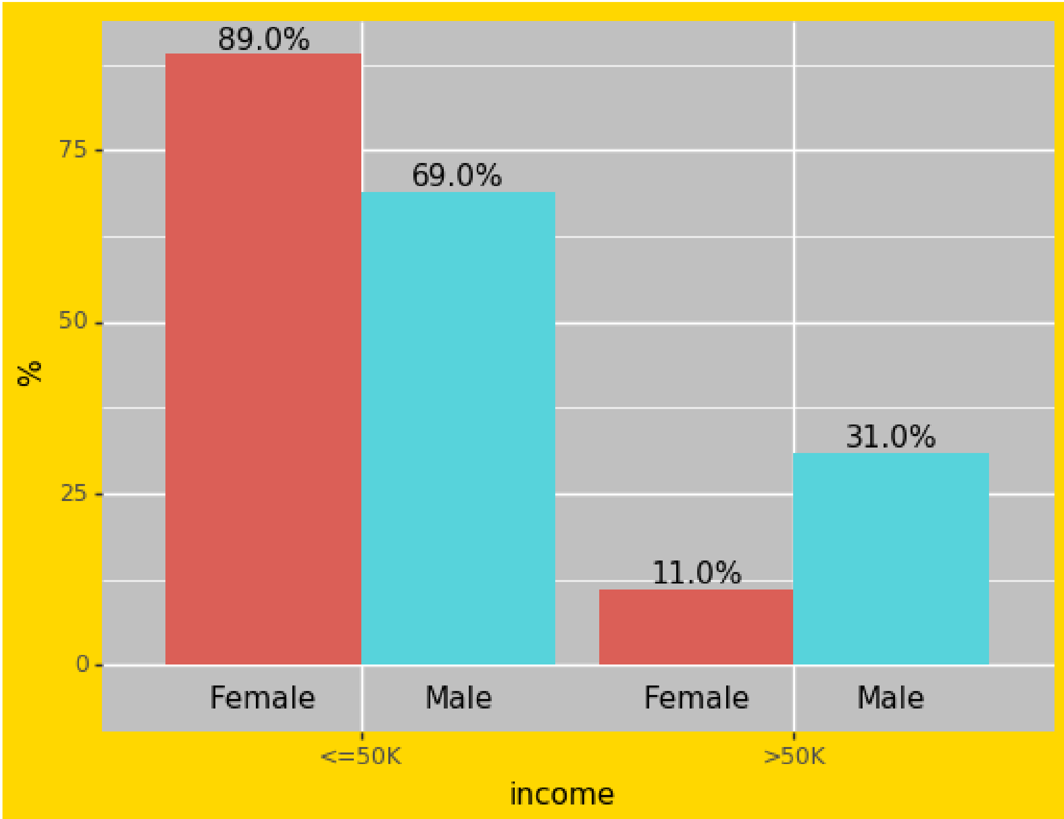
10 Gender vs Income

```
In [11]: tdf=relation('gender')
         tdf

Out[11]:    gender income     %
         0    Male  <=50K  69.0
         1    Male   >50K  31.0
         2  Female  <=50K  89.0
         3  Female   >50K  11.0
```

```
In [12]: (ggplot(tdf,aes(x='income',y='%',fill='gender'))+geom_bar(stat='identity',position=
         'dodge',show_legend=1==0)+geom_text(aes(y=-5,label='gender'),position=
         position_dodge(width=0.9))+geom_text(aes(label='%'),position=
         position_dodge(width=0.9),va='bottom', format_string='{}%'))
```

```
C:\Anocanda\lib\site-packages\plotnine\layer.py:517: MatplotlibDeprecationWarning: isinstance(
  return not cbook.iterable(value) and (cbook.is_numlike(value) or
```



Out[12]: <ggplot: (149970748634)>

We can see that males are more likely to have high income than females.

11 Race VS Income

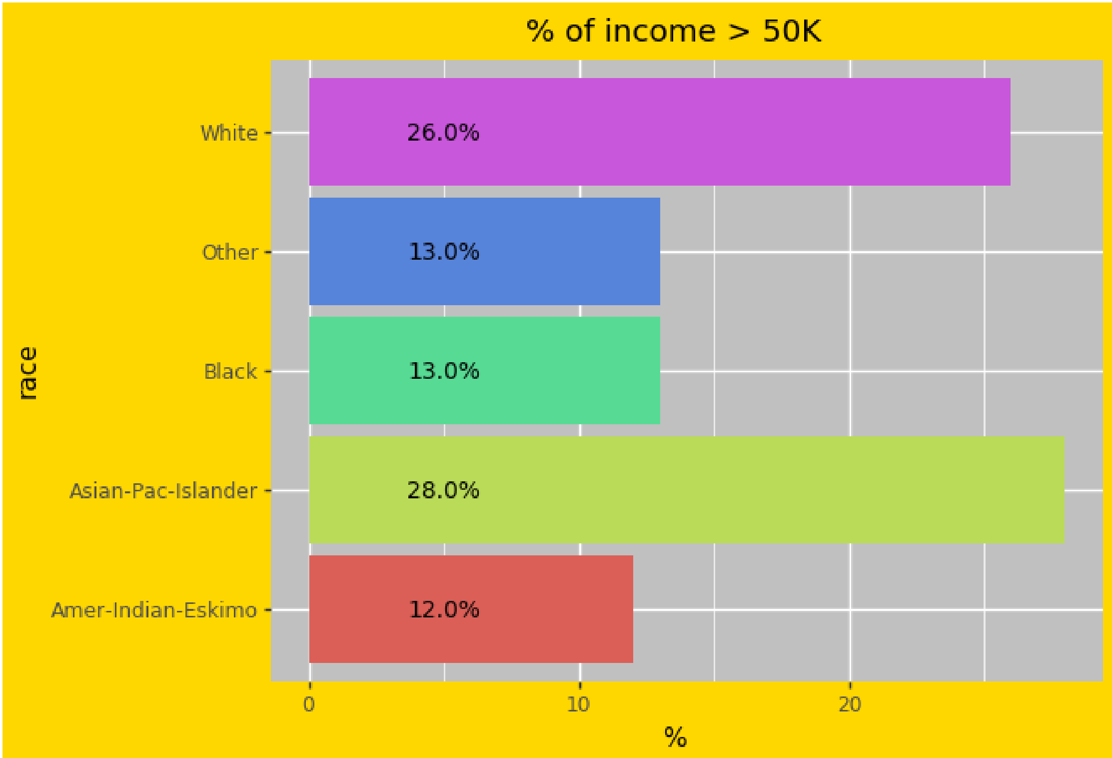
In [13]:
```
tdf=relation('race')
tdf=tdf[tdf.income=='>50K']
tdf
```

Out[13]:

	race	income	%
1	Asian-Pac-Islander	>50K	28.0
3	White	>50K	26.0
5	Other	>50K	13.0
7	Black	>50K	13.0
9	Amer-Indian-Eskimo	>50K	12.0

In [14]:
```
rggplot('race',10)
```

```
C:\Anocanda\lib\site-packages\plotnine\layer.py:517: MatplotlibDeprecationWarning: isinstance(
  return not cbook.iterable(value) and (cbook.is_numlike(value) or
```




```
Out[14]: <ggplot: (149970995709)>
```

We can see that Asian-Pac-Islander & White races are more likely to have high income than other races.

12 Occupation VS Income

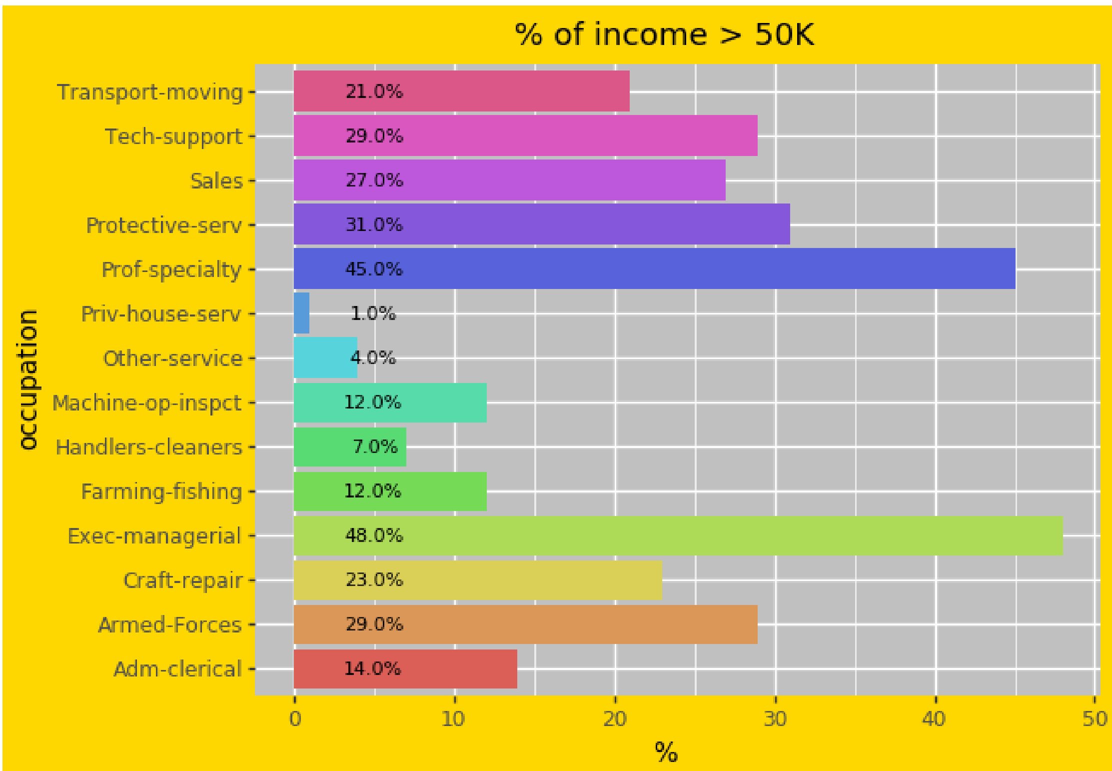
```
In [15]: tdf=relation('occupation')
         tdf=tdf[tdf.income=='>50K']
         tdf
```

```
Out[15]:          occupation income     %
         1   Exec-managerial   >50K  48.0
         3    Prof-specialty   >50K  45.0
         5   Protective-serv   >50K  31.0
         7      Tech-support   >50K  29.0
         9      Armed-Forces   >50K  29.0
         11            Sales   >50K  27.0
         13     Craft-repair   >50K  23.0
         15 Transport-moving   >50K  21.0
         17     Adm-clerical   >50K  14.0
         19 Machine-op-inspct  >50K  12.0
         21  Farming-fishing   >50K  12.0
```

```
23  Handlers-cleaners   >50K   7.0
25      Other-service   >50K   4.0
27     Priv-house-serv   >50K   1.0
```

In [16]:
```
rggplot('occupation')
```

```
C:\Anocanda\lib\site-packages\plotnine\layer.py:517: MatplotlibDeprecationWarning: isinstance(
  return not cbook.iterable(value) and (cbook.is_numlike(value) or
```



Out[16]: <ggplot: (-9223371886883740174)>

We see that Exec-managerial, Prof-specialty, & Protective-serv are more likely to have high income than other occupations.

13 Workclass VS Income

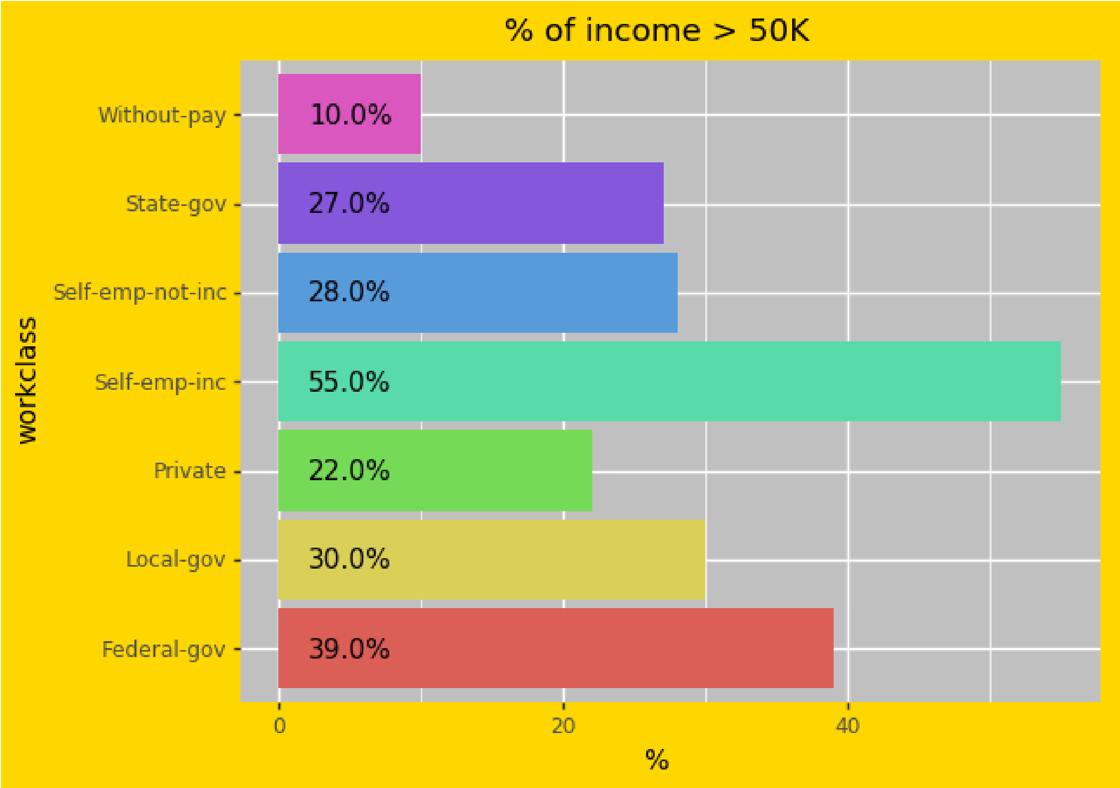
In [17]:
```
tdf=relation('workclass')
tdf=tdf[tdf.income=='>50K']
tdf
```

Out[17]:
```
          workclass income     %
1      Self-emp-inc   >50K  55.0
```

```
3          Federal-gov   >50K  39.0
5            Local-gov   >50K  30.0
7     Self-emp-not-inc   >50K  28.0
9            State-gov   >50K  27.0
11             Private   >50K  22.0
13         Without-pay   >50K  10.0
```

```
In [18]: (ggplot(tdf,aes('workclass','%',fill='workclass'))+geom_bar(stat='identity',show_legen
         1==0)+geom_text(aes(label='%',y=5),format_string='{}%')+coord_flip()+ggtitle('% of inc
```

```
C:\Anocanda\lib\site-packages\plotnine\layer.py:517: MatplotlibDeprecationWarning: isinstance(
  return not cbook.iterable(value) and (cbook.is_numlike(value) or
```




```
Out[18]: <ggplot: (-9223371886883613303)>
```

We see that Self-emp-inc, Federal-gov, Local-gov are more likely to have high income than other workclass.

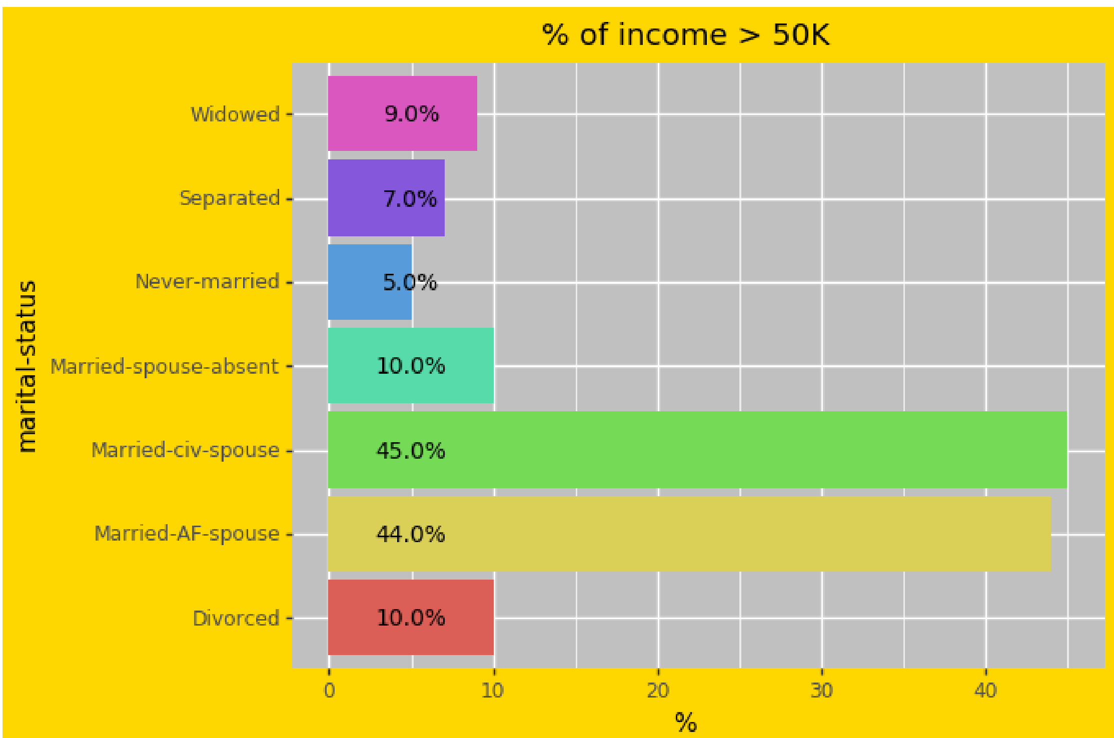
14 Marital-status VS Income

```
In [19]: tdf=relation('marital-status')
         tdf=tdf[tdf.income=='>50K']
         tdf
```

```
Out[19]:           marital-status income     %
        1       Married-civ-spouse   >50K  45.0
        3        Married-AF-spouse   >50K  44.0
        5                 Divorced   >50K  10.0
        7    Married-spouse-absent   >50K  10.0
        9                  Widowed   >50K   9.0
        11               Separated   >50K   7.0
        13           Never-married   >50K   5.0
```

```
In [20]: rggplot('marital-status',10)
```

```
C:\Anocanda\lib\site-packages\plotnine\layer.py:517: MatplotlibDeprecationWarning: isinstance(
  return not cbook.iterable(value) and (cbook.is_numlike(value) or
```




```
Out[20]: <ggplot: (-9223371886883712368)>
```

We see people who has the marital status of Married-civ-spouse, or Married-AF-spouse are much more likely to have high income than others.

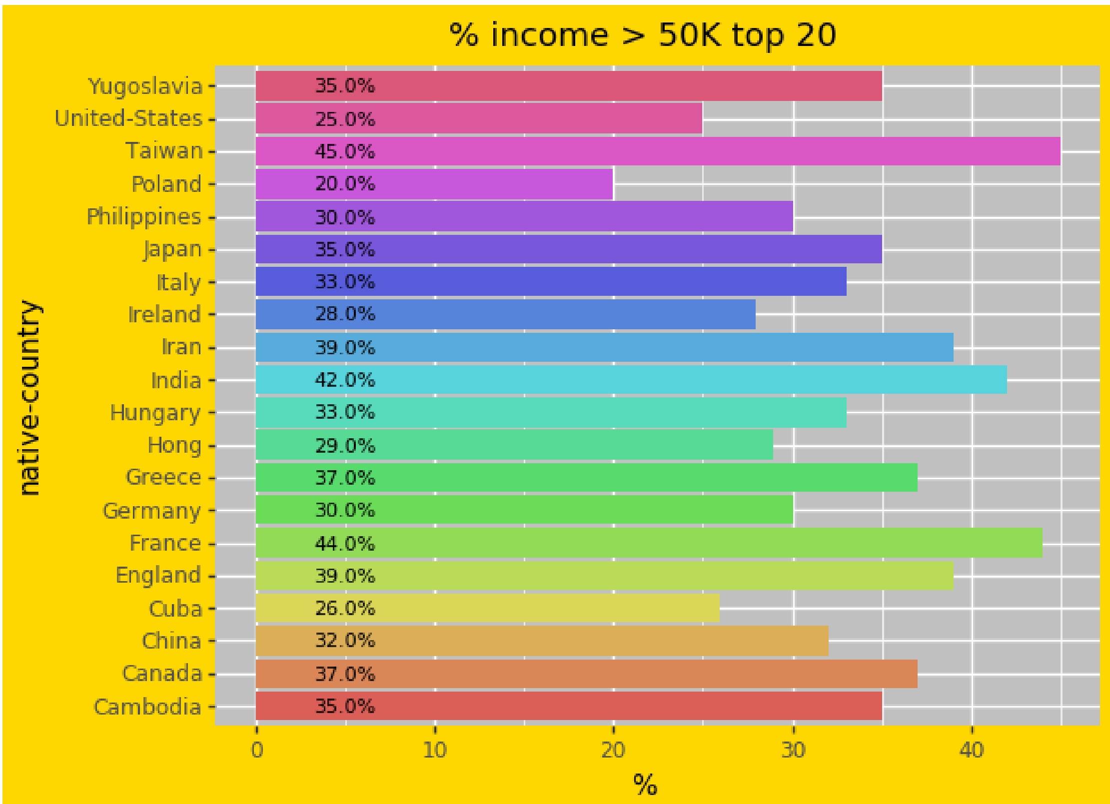
15 Native-country VS Income

```
In [21]: tdf=relation('native-country')
         tdf=tdf[tdf.income=='>50K']
```

```
tdf
tdf=tdf.iloc[:20,:]
```

```
In [22]: rggplot('native-country')+ggtitle('% income > 50K top 20')
```

```
C:\Anocanda\lib\site-packages\plotnine\layer.py:517: MatplotlibDeprecationWarning: isinstance(
  return not cbook.iterable(value) and (cbook.is_numlike(value) or
```




```
Out[22]: <ggplot: (-9223371886883736004)>
```

- We see that top 3 native country that have higher % of high income individuals are Taiwan, France, & India.
- U.S.A. as native country is ranked in the middle with 25% of people with high income.

16 Random Forest model

```
In [23]: from custom import tts
         from sklearn.ensemble import RandomForestClassifier as rfc
```

16.1 Getting data ready for modeling

```
In [24]: from sklearn.preprocessing import LabelEncoder as le
         from pandas import get_dummies as getd
```

```
In [25]: tdf=df.loc[:,['educational-num','workclass','marital-status','occupation','race',
                       'gender','native-country','income']]
```

```
In [26]: tdf.loc[:,'age']=le().fit_transform(df.age)
         clm=['workclass','marital-status','occupation','race','gender','native-country','incom
         tdf=getd(tdf,columns=clm,prefix=clm,drop_first=1==1)
```

16.2 Split data into training and testing group

```
In [27]: xtr,xte,ytr,yte=tts(tdf,dep='income_>50K')
```

```
C:\Anocanda\lib\site-packages\sklearn\model_selection\_split.py:2069: FutureWarning: From versi
  FutureWarning)
```

16.3 Model fitting

```
In [28]: mod=rfc(n_estimators=100).fit(xtr,ytr)
```

16.4 Look at feature importance

```
In [29]: imp=pd.DataFrame({'Attribute':tdf.columns[:-1],'Score':mod.feature_importances_.round
```

16.5 Clean up

```
In [30]: def clean(string):
             if '_' in string:
                 return string[:string.find('_')]
             return string
```

```
In [31]: imp.Attribute=imp.Attribute.map(clean)
```

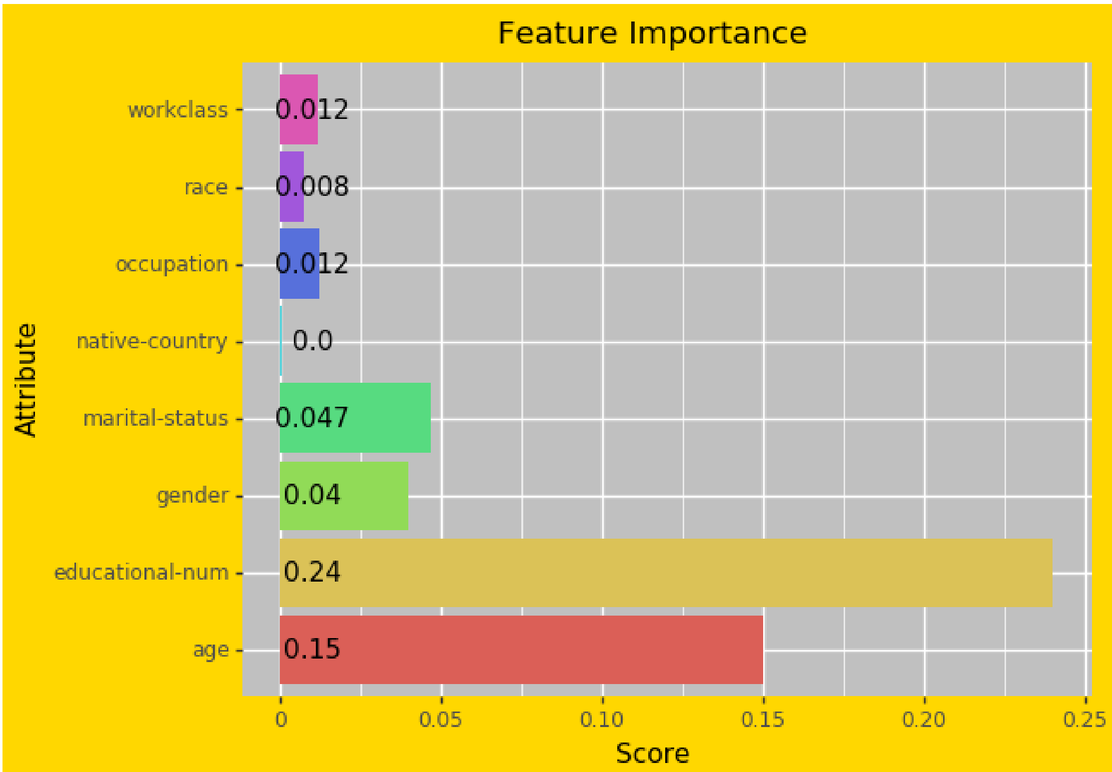
```
In [32]: tdf=imp.groupby('Attribute').mean().reset_index().sort_values('Score',ascending=1==0)
         tdf
```

Out[32]:

	Attribute	Score
1	educational-num	0.240000
0	age	0.150000
3	marital-status	0.046667
2	gender	0.040000
5	occupation	0.012308
7	workclass	0.011667
6	race	0.007500
4	native-country	0.000250

In [34]:
```
ggplot(tdf,aes('Attribute','Score',fill='Attribute'))+geom_bar(stat='identity',
show_legend=1==0)+geom_text(aes(label='round(Score,3)',y=.01))+coord_flip()+\
labs(title='Feature Importance',caption='Random Forest')
```

```
C:\Anocanda\lib\site-packages\plotnine\layer.py:517: MatplotlibDeprecationWarning: isinstance(
  return not cbook.iterable(value) and (cbook.is_numlike(value) or
```



Out[34]: <ggplot: (149972913721)>

We see that education & age are by far the most important features according to this model.

16.6 Accuracy of the model

In [35]:
```
mod.score(xte,yte)
```

Out[35]: 0.8219339622641509