Naive Bayes:
https://scikit-learn.org/stable/modules/naive_bayes.html

```
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
```

Name the Headers:

col_names = ["age", "workclass", "fnlwgt", "education", "education-num", "marital-status", "occupation", "relationship", "race", "sex", "capital-gain", "capital-loss", "hours-per-week", "native-country", "result"]
data = pd.read_csv("XXXXXXX.csv", names=col_names)

Import Data and Read CSV:

adult = pd.read_csv(directory address)

Clean Data:

adult_clean=adult.replace(regex=[r'\?|\.|\$'],value=np.nan)
#replace all the stupid wrong data with np.nan
adult_clean.isnull().any()
#see the null data
adult=adult_clean.dropna(how='any')
#drop all wrong data

```
adult=adult.drop(['fnlwgt'],axis=1)
adult.info()

#Make result 0 for <=50k and 1 for >50k
le = preprocessing.LabelEncoder()
data_clean['result'] = le.fit_transform(data_clean['result'])
```

Split data into Train and Test sets:

from sklearn.cross_validation import train_test_split

col_names = ["age", "workclass", "education", "education-num", "marital-status", "occupation", "relationship", "race", "sex", "capital-gain", "capital-loss", "hours-per-week", "native-country", "result"]

X_train,X_test,y_train,y_test=train_test_split(adult[col_names[1:13]],adult[col_names[13]],test_size=0.25,random_state=33)

Categorical to Numerical
from sklearn import preprocessing
le = preprocessing.LabelEncoder()

```
data_clean['workclass'] = le.fit_transform(data_clean['workclass'])
data_clean['marital-status'] = le.fit_transform(data_clean['marital-status'])
data_clean['occupation'] = le.fit_transform(data_clean['occupation'])
data_clean['relationship'] = le.fit_transform(data_clean['race'])
data_clean['race'] = le.fit_transform(data_clean['race'])
data_clean['sex'] = le.fit_transform(data_clean['sex'])
data_clean['native-country'] = le.fit_transform(data_clean['native-country'])
data_clean['result'] = le.fit_transform(data_clean['result'])
#better way:
--------------------------
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
for col in adult:
  adult[col] = le.fit_transform(adult[col])
----------------------------
```

Data and Target sets

```
cols = [col for col in adult.columns if col not in ['result']]
adult = data_clean[cols]
target = data_clean['result']
adult.head()
```

Split into training and Testing

```
from sklearn.model_selection import train_test_split

data_train,data_test,target_train,target_test=train_test_split(adult,target, test_size = .3,
random_state = 10)
```

```
import pandas as pd
import numpy as np
col_names = ["age", "workclass", "fnlwgt", "education", "education-num", "marital-status",
"occupation",  "relationship", "race", "sex", "capital-gain", "capital-loss", "hours-per-week",
"native-country", "result"]
data =
pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data',
names = col_names)
data = data.drop(['fnlwgt'], axis=1)
data = data.drop(['education'], axis=1)


# In[124]:


data_clean = data.replace(regex = [r'\?'], value = np.nan)
data_clean.isnull().any()
data_clean=data_clean.dropna(how='any')
data_clean.head(n = 15)


# In[125]:


import seaborn as sns
import matplotlib.pyplot as plt

sns.set(style="whitegrid", color_codes=True)
sns.set(rc={'figure.figsize':(11.7,8.27)})
#sns.countplot('age',data=data_clean,hue = 'hours')
sns.scatterplot('age', 'hours-per-week', data = data_clean)
sns.despine(offset=10, trim=True)

plt.show()


# In[143]:


from sklearn import preprocessing
le = preprocessing.LabelEncoder()

data_clean['workclass'] = le.fit_transform(data_clean['workclass'])
data_clean['marital-status'] = le.fit_transform(data_clean['marital-status'])
data_clean['occupation'] = le.fit_transform(data_clean['occupation'])
data_clean['relationship'] = le.fit_transform(data_clean['race'])
```

```
data_clean['race'] = le.fit_transform(data_clean['race'])
data_clean['sex'] = le.fit_transform(data_clean['sex'])
data_clean['native-country'] = le.fit_transform(data_clean['native-country'])
data_clean['result'] = le.fit_transform(data_clean['result'])
data_clean.head(n=15)
--------------------------
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
for col in adult:
  adult[col] = le.fit_transform(adult[col])
---------------------------


# In[164]:


cols = [col for col in data_clean.columns if col not in ['result']]
adult = data_clean[cols]
target = data_clean['result']
adult.head()


# In[165]:


from sklearn.model_selection import train_test_split

data_train,data_test,target_train,target_test=train_test_split(adult,target, test_size = .3,
random_state = 10)


# In[166]:


from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score

gnb = GaussianNB()

pred = gnb.fit(data_train, target_train).predict(data_test)
print(accuracy_score(target_test, pred, normalize = True))
```

Random Forest

http://bluewhale.cc/2016-12-23/use-python-to-train-random-forest-model-to-identify-suspicious-traffic.html

```
from sklearn.ensemble import RandomForestClassifier
clf = RandomForestClassifier()
clf = clf.fit(X_train, y_train)
clf.score(X_test, y_test)
```