In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib
import seaborn as sns
```

In [3]:
```
df = pd.read_csv("/Users/shubham/Downloads/in-vehicle-coupon-recommendation.csv")
df
```

Out[3]:

	destination	passanger	weather	temperature	time	coupon	expiration	gender	age	maritalStatus	...	CoffeeHouse	CarryAway	RestaurantL
0	No Urgent Place	Alone	Sunny	55	2PM	Restaurant(<20)	1d	Female	21	Unmarried partner	...	never	NaN	
1	No Urgent Place	Friend(s)	Sunny	80	10AM	Coffee House	2h	Female	21	Unmarried partner	...	never	NaN	
2	No Urgent Place	Friend(s)	Sunny	80	10AM	Carry out & Take away	2h	Female	21	Unmarried partner	...	never	NaN	
3	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	2h	Female	21	Unmarried partner	...	never	NaN	
4	No Urgent Place	Friend(s)	Sunny	80	2PM	Coffee House	1d	Female	21	Unmarried partner	...	never	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
12679	Home	Partner	Rainy	55	6PM	Carry out & Take away	1d	Male	26	Single	...	never	1~3	
12680	Work	Alone	Rainy	55	7AM	Carry out & Take away	1d	Male	26	Single	...	never	1~3	
12681	Work	Alone	Snowy	30	7AM	Coffee House	1d	Male	26	Single	...	never	1~3	
12682	Work	Alone	Snowy	30	7AM	Bar	1d	Male	26	Single	...	never	1~3	
12683	Work	Alone	Sunny	80	7AM	Restaurant(20-50)	2h	Male	26	Single	...	never	1~3	

12684 rows × 26 columns

In [18]:
```
print(df1[["weather", "age", "expiration", "income","destination","gender"]].apply(lambda x: x.unique()))
```

```
weather                                  [Sunny, Rainy, Snowy]
age                  [21, 46, 26, 31, 41, 50plus, 36, below21]
expiration                                          [1d, 2h]
income         [$37500 - $49999, $62500 - $74999, $12500 - $2...
destination                          [No Urgent Place, Home, Work]
gender                                         [Female, Male]
dtype: object
```

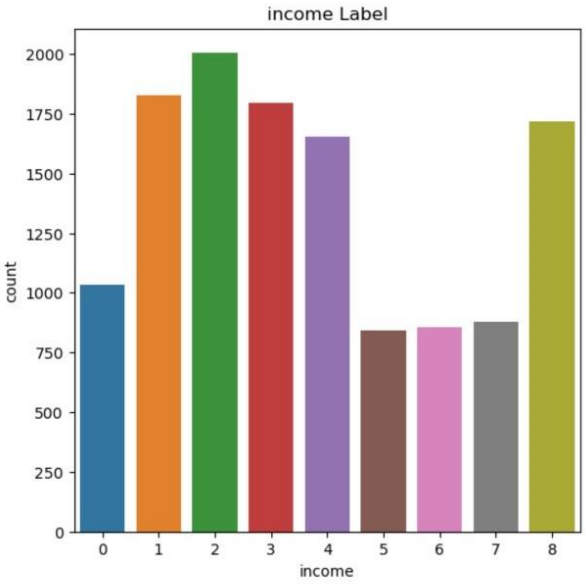
In [19]:
```
ion':{'2h': 0, '1d' : 1},
':{'Male': 0, 'Female' : 1},
r':{'Sunny': 2, 'Rainy' : 0 , 'Snowy':1},

'21': 0, '26': 1, '31': 2, '36': 3, '41': 4 , '46':5,'below21' :6 ,'50plus':7},
ature':{30: 0, 55: 1, 80: 2},
':{'Less than $12500':0, '$12500 - $24999':1, '$25000 - $37499':2, '$37500 - $49999':3, '$50000 - $62499':4, '$62500
tion' : {'No Urgent Place':3 , 'Home':1 ,'Work':2}})
```

In [20]:
```
print(df2[["weather", "age", "expiration", "income","destination","gender" , "Y"]])
```

```
       weather  age  expiration  income  destination  gender  Y
0            2    0           1       3            3       1  1
1            2    0           0       3            3       1  0
2            2    0           0       3            3       1  1
3            2    0           0       3            3       1  0
4            2    0           1       3            3       1  0
5            2    0           0       3            3       1  1
6            2    0           1       3            3       1  1
7            2    0           0       3            3       1  1
8            2    0           0       3            3       1  1
9            2    0           1       3            3       1  0
10           2    0           1       3            3       1  1
11           2    0           1       3            3       1  1
12           2    0           0       3            3       1  1
13           2    0           1       3            1       1  1
14           2    0           1       3            1       1  1
```

In [21]:
```
df2.income.value_counts()
plt.figure(figsize=(6,6))
sns.countplot(x='income', data=df2)
plt.title('income Label')
plt.show()
```



In [41]:
```
from sklearn.datasets import make_classification
X, y = make_classification(n_classes=2, class_sep=0.5,
weights=[0.05, 0.95], n_informative=6, n_redundant=0, flip_y=0,
n_features=6, n_clusters_per_class=1, n_samples=12000, random_state=10)
```

In [42]:
```
from sklearn.model_selection import train_test_split

# split into 70:30 ration
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 0)
```

In [43]:
```
from imblearn.over_sampling import SMOTE
sm = SMOTE(random_state=42)
X_res, y_res = sm.fit_resample(X_train, y_train)
```

In [44]:
```
# describes info about train and test set
print("Number transactions X_train dataset: ", X_train.shape)
print("Number transactions y_train dataset: ", y_train.shape)
print("Number transactions X_test dataset: ", X_test.shape)
print("Number transactions y_test dataset: ", y_test.shape)
```

```
Number transactions X_train dataset:  (8400, 6)
Number transactions y_train dataset:  (8400,)
Number transactions X_test dataset:  (3600, 6)
Number transactions y_test dataset:  (3600,)
```

In [27]:
```
x = df2[["weather", "age", "expiration", "income","destination","gender"]]
y = df2['Y']
```

In [28]:
```
#import classification algos and cross validation
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
from sklearn.model_selection import cross_val_score
```

In [ ]:
```
pip install --upgrade scikit-learn
```

In [29]:
```
cross_val_score(SVC(),x, y, cv = 5)
```

Out[29]:
```
array([0.59595559, 0.60586836, 0.60190325, 0.59397304, 0.6149881 ])
```

In [30]:
```
cross_val_score(DecisionTreeClassifier(), x, y, cv = 5)
```

Out[30]:
```
array([0.57890563, 0.56819984, 0.57176844, 0.55471848, 0.55233941])
```

In [31]:
```
cross_val_score(LogisticRegression(), x, y, cv = 5)
```

Out[31]:
```
array([0.60348929, 0.60348929, 0.60943695, 0.59714512, 0.5888184 ])
```

In [32]:
```
cross_val_score(KNeighborsClassifier(),x, y ,cv = 5)
```

Out[32]:
```
array([0.56066614, 0.55035686, 0.55511499, 0.55828707, 0.55352895])
```

In [35]:
```
cross_val_score(RandomForestClassifier(n_estimators=50), x, y, cv = 5)
```

Out[35]:
```
array([0.57652657, 0.5590801 , 0.57216495, 0.5630452 , 0.55828707])
```

In [33]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.2)
```

Out[31]:
```
array([0.60348929, 0.60348929, 0.60943695, 0.59714512, 0.5888184 ])
```

In [32]:
```
cross_val_score(KNeighborsClassifier(),x, y ,cv = 5)
```

Out[32]:
```
array([0.56066614, 0.55035686, 0.55511499, 0.55828707, 0.55352895])
```

In [35]:
```
cross_val_score(RandomForestClassifier(n_estimators=50), x, y, cv = 5)
```

Out[35]:
```
array([0.57652657, 0.5590801 , 0.57216495, 0.5630452 , 0.55828707])
```

In [33]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.2)
```

In [36]:
```
model = RandomForestClassifier()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

In [37]:
```
#Confusion matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
cm
```

Out[37]:
```
array([[ 461,  634],
       [ 386, 1041]])
```

In [38]:
```
print(accuracy_score(y_test, model.predict(X_test)))
```

```
0.5955590800951626
```

In [ ]:
```

```