logistische Regression

In [1]:
```
# Load the Pandas libraries, matplotlib and numpy
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# Read data
data = pd.read_csv("Umfragedaten_v1_an.csv")

# Preview the first 5 lines of the loaded data
data.head()
```

Out[1]:

	ID	GESCHL	GEBJAHR	BERUFSTAETIG	ARBEITSSTD	ARZTBES	RAUCH	GRO
0	1359	WEIBLICH	1967.0	NICHT ERWERBSTAETIG	NaN	1.0	JA	162.
1	2455	WEIBLICH	1964.0	HAUPTBERUFL.HALBTAGS	30.0	1.0	NEIN	165.
2	200	MAENNLICH	1980.0	NICHT ERWERBSTAETIG	NaN	3.0	NEIN	166.
3	1280	MAENNLICH	1968.0	HAUPTBERUFL.GANZTAGS	50.0	0.0	NEIN	180.
4	2384	WEIBLICH	1995.0	NICHT ERWERBSTAETIG	NaN	1.0	NEIN	161.

In [2]:
```
#load libraries for regression
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
import seaborn as sn
```

Data often have NaN or other bad values, which means data need to be cleaned first.

In [4]:
```
#need to dummy code first
x  = data['NETTO'].values.reshape(-1,1)
dummies = pd.get_dummies(data, columns=['RAUCH'])
y = dummies['RAUCH_JA'].values.reshape(-1,1)
dummies.head()
```

Out[4]:

	ID	GESCHL	GEBJAHR	BERUFSTAETIG	ARBEITSSTD	ARZTBES	GRO	GEW	
0	1359	WEIBLICH	1967.0	NICHT ERWERBSTAETIG	NaN	1.0	162.0	79.0	
1	2455	WEIBLICH	1964.0	HAUPTBERUFL.HALBTAGS	30.0	1.0	165.0	59.0	
2	200	MAENNLICH	1980.0	NICHT ERWERBSTAETIG	NaN	3.0	166.0	86.0	
3	1280	MAENNLICH	1968.0	HAUPTBERUFL.GANZTAGS	50.0	0.0	180.0	95.0	
4	2384	WEIBLICH	1995.0	NICHT ERWERBSTAETIG	NaN	1.0	161.0	46.0	

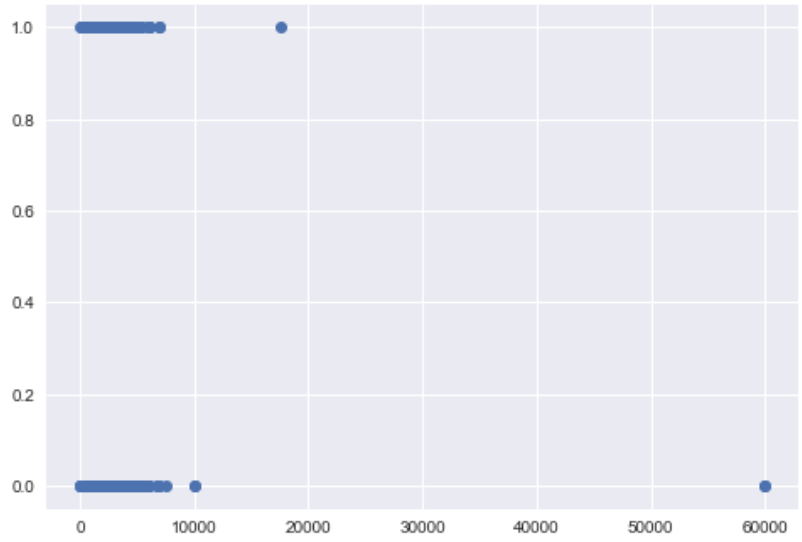
In [5]:
```
#not sure why pandas creates 2D arrays, but need 1D otherwise logistic regression
x=x[:,0]
y=y[:,0]

#need to clean data, there seem to be some NaN in x
print(np.any(np.isnan(x)))
print(np.any(np.isnan(y)))
```

```
True
False
```

In [6]:
```
#print(np.where(np.isnan(x) == True))
print(x.ndim)
print(len(x))
index = np.where(~np.isnan(x))  #index of good data entries
#print(len(index))

xnew = x[index]  #only keep good entries
ynew = y[index]
print(xnew)
```

```
1
3471
[  475.   780.  4000. ...,  1500.  1300.  1100.]
```

In [7]:
```
#show a scatter plot
plt.plot(xnew,ynew,'o')
plt.show()

#1=raucher, 0=nichtraucher
```



Now we train the model by splitting it into a training set and then the test set is used to validate it.

In [8]:
```
#train the model

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(xnew.reshape(-1,1), ynew, test
```

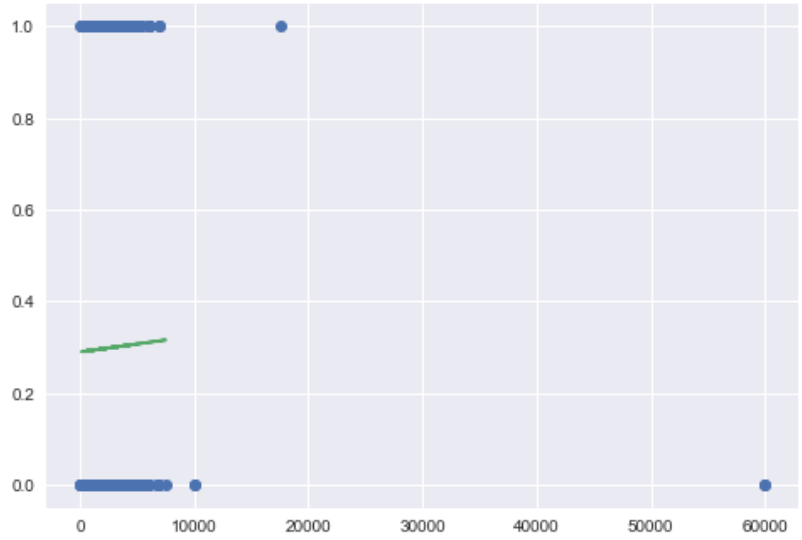
In [9]:
```
# logistic regression using the default parameters)
logreg = LogisticRegression()

# fit the model with data
logreg.fit(X_train,y_train)

#
y_pred=logreg.predict(X_test)
y_pred2 = logreg.predict_proba(X_test)[:,1]

print(logreg.coef_, logreg.intercept_)
```

```
(array([[  1.63918095e-05]]), array([-0.89435454]))
```

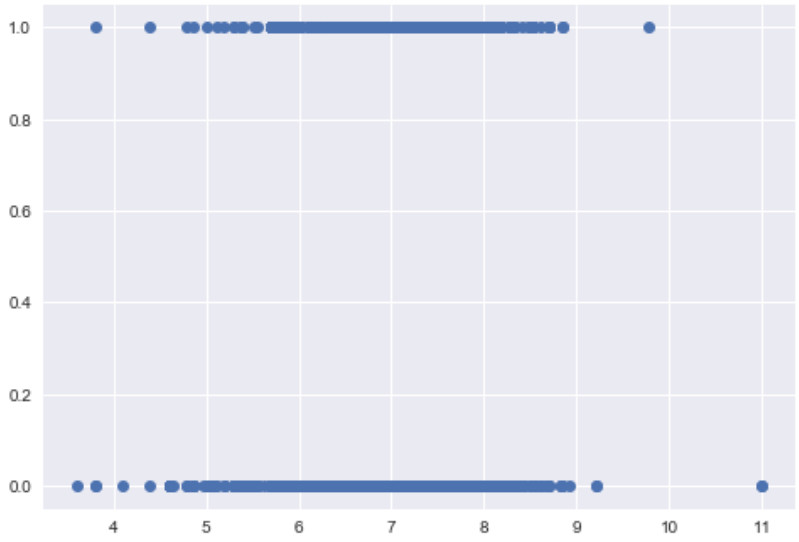
In [10]:
```
#confusion matrix
from sklearn import metrics
cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
cnf_matrix

#the matrix shows that 37 data points were predicted wrongly, so this fit was not
```

Out[10]:
```
array([[99,  0],
       [37,  0]])
```

In [11]:
```
#plotting
#show a scatter plot
plt.plot(xnew,ynew,'o')
plt.plot(X_test,y_pred2)
plt.show()

#this also shows a bad fit.
```



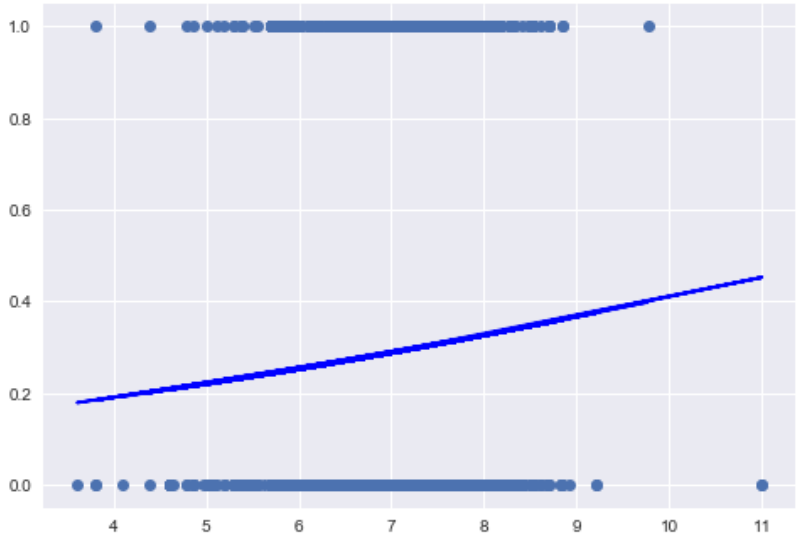
In [12]:
```
#let's try changing the model by taking the (natural) log of the income
plt.plot(np.log(xnew),ynew,'o')
plt.show()
#one can see that there is actually not a very clear separation between non-smoker
#We fit it anyway.
```



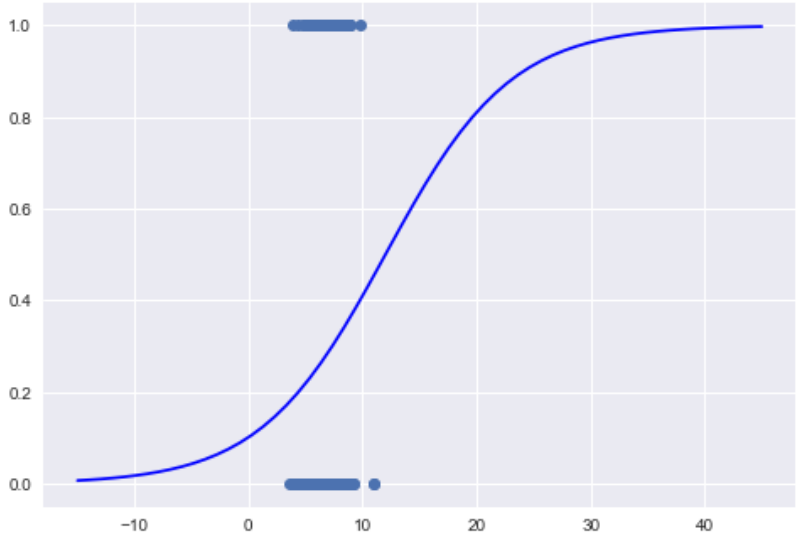
In [13]:
```
xnew2 = np.log(xnew)
X_train, X_test, y_train, y_test = train_test_split(xnew2.reshape(-1,1), ynew, tes

# logistic regression using the default parameters)
logreg = LogisticRegression(C=9)

# fit the model with data
logreg.fit(X_train,y_train)

#
y_pred=logreg.predict(X_test)
y_pred2 = logreg.predict_proba(X_test)[:,1]

print(logreg.coef_, logreg.intercept_)
```

```
(array([[ 0.1815504]]), array([-2.17652385]))
```

In [14]:
```
#create the model manually to verify the prediction of python works (yes, it's the
pi = np.exp(-2.17+.18*xnew2) / (1+np.exp(-2.17+.18*xnew2))

#plotting
#show a scatter plot
plt.plot(xnew2,ynew,'o')
plt.plot(X_test,y_pred2)
plt.plot(xnew2,pi,color='b')
plt.show()
```



In [15]:
```
#plot a larger x-axis to see the full model
xlarge = np.linspace(-15,45,60).reshape((-1,1))
ylarge = logreg.predict_proba(xlarge)[:,1]

plt.plot(xnew2,ynew,'o')
plt.plot(xlarge,ylarge,color='b')
plt.show()

#yes, the model is a logit model, but the data are not clearly separated, which me
```



In [16]:
```
#wahrscheinlichkeit bei 2000 gehalt [%]
print(logreg.predict_proba(np.log(2000))[:,1])*100.
```

```
[ 31.07582063]
```

In [ ]: