Assignment: Health Insurance Cost

The comments/sections provided are your cues to perform the assignment. You don't need to limit yourself to the number of rows/cells provided. You can add additional rows in each section to add more lines of code.

If at any point in time you need help on solving this assignment, view our demo video to understand the different steps of the code.

Happy coding!

Health Insurance Cost

DESCRIPTION

Health insurance has become an indispensable part of our lives in recent years, and people are paying for it so that they are covered in the event of an accident or other unpredicted factors. You are provided with medical costs dataset that has features such as Age, Cost, BMI.

Objective:

```
Determine the factors that contribute the most in the calculation of insurance
costs.
Predict the health Insurance Cost.
```

Actions to Perform:

```
Find the correlation of every pair of features (and the outcome variable).
Visualize the correlations using a heatmap.
Normalize your inputs.
Use the test data to find out the accuracy of the model.
Visualize how your model uses the different features and which features have a
greater effect.
```

In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

from sklearn.linear_model import LogisticRegression

import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
insuranceDF = pd.read_csv('insurance2.csv')
print(insuranceDF.head())
```

```
   age  sex     bmi  children  smoker  region      charges  insuranceclaim
0   19    0  27.900         0       1       3  16884.92400               1
1   18    1  33.770         1       0       2   1725.55230               1
2   28    1  33.000         3       0       2   4449.46200               0
3   33    1  22.705         0       0       1  21984.47061               0
4   32    1  28.880         0       0       1   3866.85520               1
```

Independent variables

1. age : age of policyholder
2. sex: gender of policy holder (female=0, male=1)
3. bmi: Body mass index, ideally 18.5 to 25
4. children: number of children / dependents of policyholder
5. smoker: smoking state of policyholder (non-smoke=0;smoker=1)
6. region: the residential area of policyholder in the US (northeast=0, northwest=1, southeast=2, southwest=3)
7. charges: individual medical costs billed by health insurance

Target variable

1. insuranceclaim - categorical variable (0,1)

In [3]:

```
insuranceDF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 8 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   age             1338 non-null   int64
 1   sex             1338 non-null   int64
 2   bmi             1338 non-null   float64
 3   children        1338 non-null   int64
 4   smoker          1338 non-null   int64
 5   region          1338 non-null   int64
 6   charges         1338 non-null   float64
 7   insuranceclaim  1338 non-null   int64
dtypes: float64(2), int64(6)
memory usage: 83.7 KB
```

Let's start by finding correlation of every pair of features (and the outcome variable), and visualizing the correlations using a heatmap.

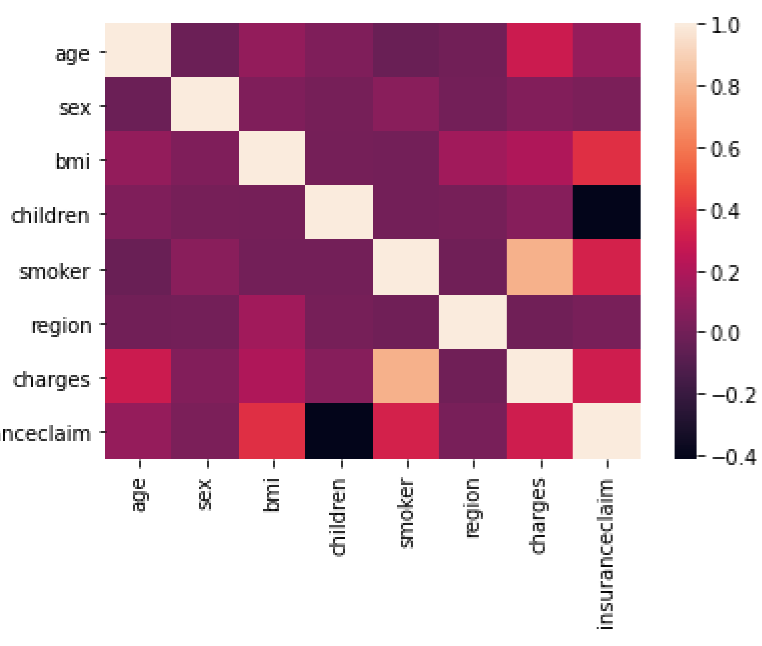
In [4]:

```
corr = insuranceDF.corr()
print(corr)
sns.heatmap(corr,
        xticklabels=corr.columns,
        yticklabels=corr.columns)
```

```
                     age       sex       bmi  children    smoker    region  \
age             1.000000 -0.020856  0.109272  0.042469 -0.025019  0.002127
sex            -0.020856  1.000000  0.046371  0.017163  0.076185  0.004588
bmi             0.109272  0.046371  1.000000  0.012759  0.003750  0.157566
children        0.042469  0.017163  0.012759  1.000000  0.007673  0.016569
smoker         -0.025019  0.076185  0.003750  0.007673  1.000000 -0.002181
region          0.002127  0.004588  0.157566  0.016569 -0.002181  1.000000
charges         0.299008  0.057292  0.198341  0.067998  0.787251 -0.006208
insuranceclaim  0.113723  0.031565  0.384198 -0.409526  0.333261  0.020891

                 charges  insuranceclaim
age             0.299008        0.113723
sex             0.057292        0.031565
bmi             0.198341        0.384198
children        0.067998       -0.409526
smoker          0.787251        0.333261
region         -0.006208        0.020891
charges         1.000000        0.309418
insuranceclaim  0.309418        1.000000
```

Out[4]: `<AxesSubplot:>`



The dataset consists the records of 1338 patients in total. Using 1000 records for training and 300 records for testing, and the last 38 records to cross check your model.

In [5]:

```
dfTrain = insuranceDF[:1000]
dfTest = insuranceDF[1000:1300]
dfCheck = insuranceDF[1300:]
```

In [6]:

```
trainLabel = np.asarray(dfTrain['insuranceclaim'])
trainData = np.asarray(dfTrain.drop('insuranceclaim',1))
testLabel = np.asarray(dfTest['insuranceclaim'])
testData = np.asarray(dfTest.drop('insuranceclaim',1))
```

Before using machine learning, normalize the inputs. Machine Learning models often benefit substantially from input normalization. It also makes it easier to understand the importance of each feature later, when looking at the model weights. Normalize the data such that each variable has 0 mean and standard deviation of 1.

In [7]:

```
means = np.mean(trainData, axis=0)
stds = np.std(trainData, axis=0)

trainData = (trainData - means)/stds
testData = (testData - means)/stds
```

In [8]:

```
insuranceCheck = LogisticRegression()
insuranceCheck.fit(trainData, trainLabel)
```

Out[8]: `LogisticRegression()`

Now, use test data to find out accuracy of the model.

In [9]:

```
accuracy = insuranceCheck.score(testData, testLabel)
print("accuracy = ", accuracy * 100, "%")
```

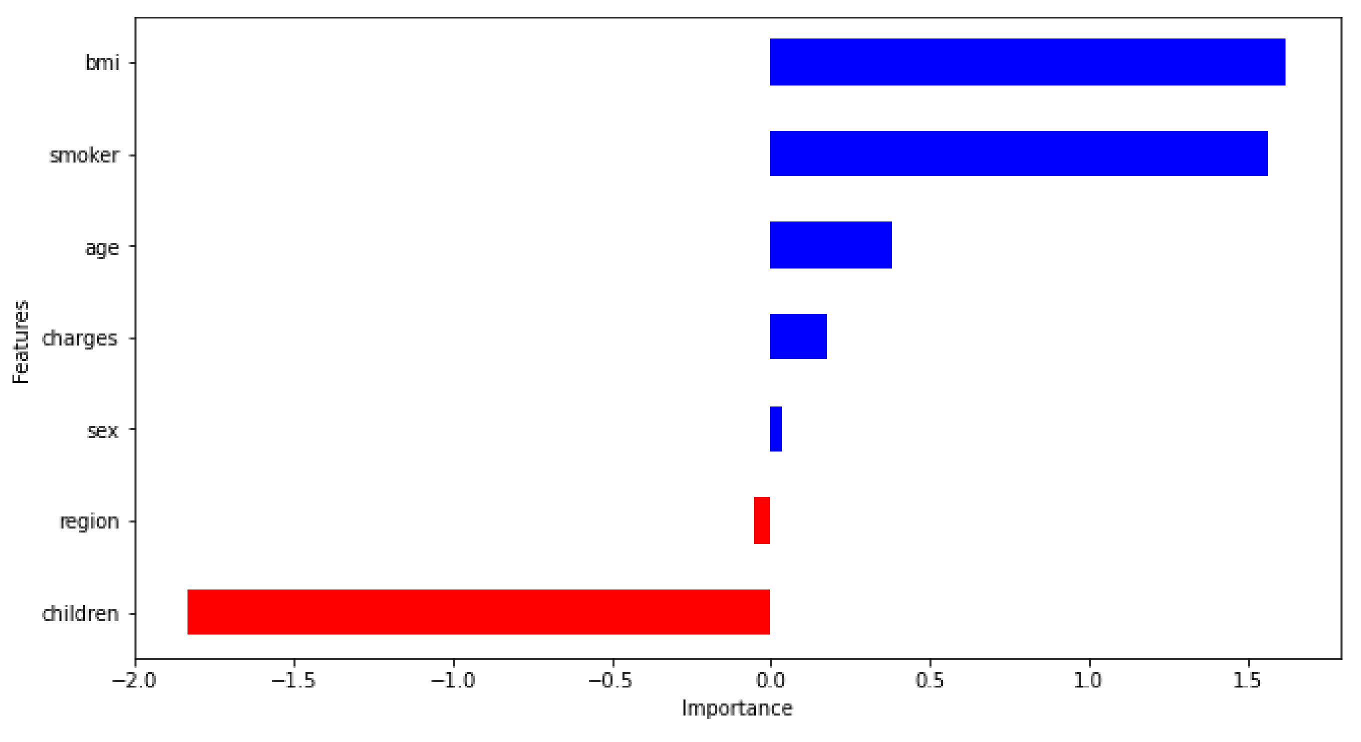
```
accuracy =  86.0 %
```

To get a better sense of what is going on inside the logistic regression model, visualize how your model uses the different features and which features have greater effect.

In [10]:

```
coeff = list(insuranceCheck.coef_[0])
labels = list(dfTrain.drop('insuranceclaim',1).columns)
features = pd.DataFrame()
features['Features'] = labels
features['importance'] = coeff
features.sort_values(by=['importance'], ascending=True, inplace=True)
features['positive'] = features['importance'] > 0
features.set_index('Features', inplace=True)
features.importance.plot(kind='barh', figsize=(11, 6),color = features.positive.map({'
plt.xlabel('Importance')
```

Out[10]: `Text(0.5, 0, 'Importance')`



From the above figure,

1. BMI, Smoker have significant influence on the model, specially BMI.

2. Children have a negative influence on the prediction, i.e. higher number children / dependents are correlated with a policy holder who has not taken insurance claim.

3. Although age was more correlated than BMI to the output variables, the model relies more on BMI. This can happen for several reasons, including the fact that the correlation captured by age is also captured by some other variable, whereas the information captured by BMI is not captured by other variables.

Note that this above interpretations require that your input data is normalized. Without that, you can't claim that importance is proportional to weights.