In [140]:
```
import pandas as pd
```

In [141]:
```
import numpy as np
```

In [142]:
```
import seaborn as sns
```

In [143]:
```
import matplotlib.pyplot as plt
```

In [144]:
```
%matplotlib inline
```

In [145]:
```
from sklearn.linear_model import LogisticRegression
```

In [146]:
```
from sklearn.externals import joblib
```

In [147]:
```
diabetesDF = pd.read_csv('DESKTOP/diabetes.csv')
```

In [148]:
```
print(diabetesDF.head())
```

```
   Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0            6      148             72             35        0  33.6
1            1       85             66             29        0  26.6
2            8      183             64              0        0  23.3
3            1       89             66             23       94  28.1
4            0      137             40             35      168  43.1

   DiabetesPedigreeFunction  Age  Outcome
0                     0.627   50        1
1                     0.351   31        0
2                     0.672   32        1
3                     0.167   21        0
4                     2.288   33        1
```

In [149]:
```
diabetesDF.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
Pregnancies                 768 non-null int64
Glucose                     768 non-null int64
BloodPressure               768 non-null int64
SkinThickness               768 non-null int64
Insulin                     768 non-null int64
BMI                         768 non-null float64
DiabetesPedigreeFunction    768 non-null float64
Age                         768 non-null int64
Outcome                     768 non-null int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

In [150]:
```
corr = diabetesDF.corr()
```

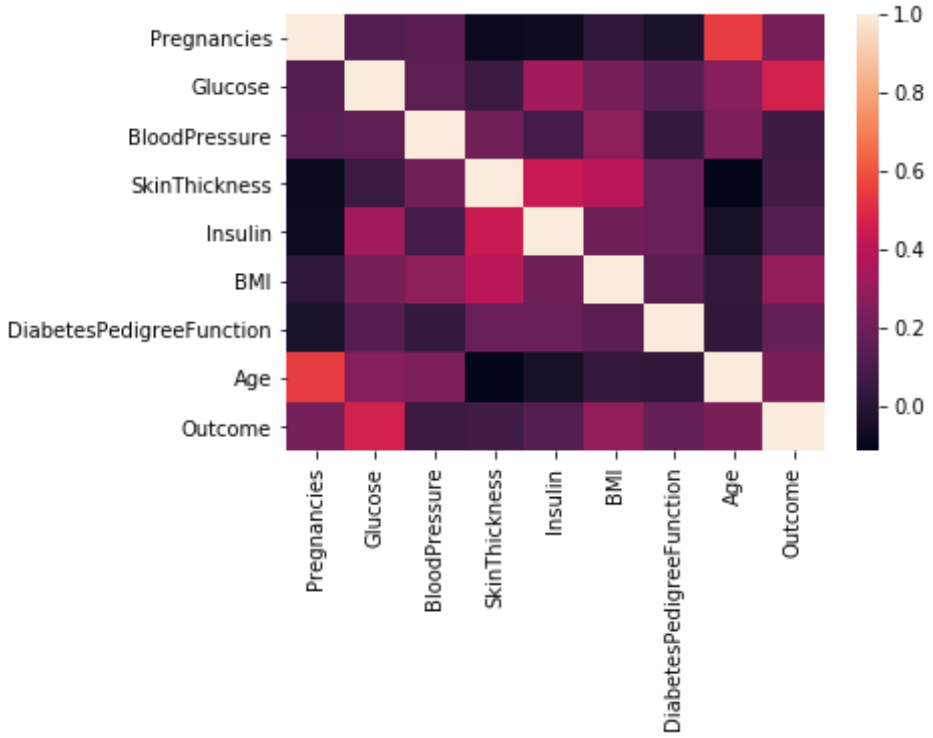
In [151]:
```
print(corr)
```

```
                          Pregnancies   Glucose  BloodPressure  SkinThi
ckness  \
Pregnancies                  1.000000  0.129459       0.141282      -0.
081672
Glucose                      0.129459  1.000000       0.152590       0.
057328
BloodPressure                0.141282  0.152590       1.000000       0.
207371
SkinThickness               -0.081672  0.057328       0.207371       1.
000000
Insulin                     -0.073535  0.331357       0.088933       0.
436783
BMI                          0.017683  0.221071       0.281805       0.
392573
DiabetesPedigreeFunction    -0.033523  0.137337       0.041265       0.
183928
Age                          0.544341  0.263514       0.239528      -0.
113970
```

```
Outcome                     0.221898  0.466581       0.065068       0.
074752

                            Insulin       BMI  DiabetesPedigreeFunction
\
Pregnancies               -0.073535  0.017683                 -0.033523

Glucose                    0.331357  0.221071                  0.137337

BloodPressure              0.088933  0.281805                  0.041265

SkinThickness              0.436783  0.392573                  0.183928

Insulin                    1.000000  0.197859                  0.185071

BMI                        0.197859  1.000000                  0.140647

DiabetesPedigreeFunction   0.185071  0.140647                  1.000000

Age                       -0.042163  0.036242                  0.033561

Outcome                    0.130548  0.292695                  0.173844


                                Age   Outcome
Pregnancies                0.544341  0.221898
Glucose                    0.263514  0.466581
BloodPressure              0.239528  0.065068
SkinThickness             -0.113970  0.074752
Insulin                   -0.042163  0.130548
BMI                        0.036242  0.292695
DiabetesPedigreeFunction   0.033561  0.173844
Age                        1.000000  0.238356
Outcome                    0.238356  1.000000
```

In [152]:
```
sns.heatmap(corr,xticklabels=corr.columns, yticklabels=corr.columns)
```

Out[152]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x18305bc7780>
```




In [153]:
```
dfTrain = diabetesDF[:650]
```

In [154]:
```
dfTest = diabetesDF[650:750]
```

In [155]:
```
dfCheck = diabetesDF[750:]
```

In [156]:
```
trainLabel = np.asarray(dfTrain['Outcome'])
```

In [157]:
```
trainData = np.asarray(dfTrain.drop('Outcome',1))
```

In [158]:
```
testLabel = np.asarray(dfTest['Outcome'])
```

In [159]:
```
testData = np.asarray(dfTest.drop('Outcome',1))
```

In [160]:
```
means = np.mean(trainData, axis=0)
```

In [161]:
```
stds = np.std(trainData, axis=0)
```

In [162]:
```
trainData = (trainData - means)/stds
```

In [163]:
```
testData = (testData - means)/stds
```

In [164]:
```
diabetesCheck = LogisticRegression()
```

In [165]:
```
diabetesCheck.fit(trainData, trainLabel)
```

```
C:\Users\admin\Anaconda3\lib\site-packages\sklearn\linear_model\logisti
c.py:433: FutureWarning: Default solver will be changed to 'lbfgs' in
0.22. Specify a solver to silence this warning.
  FutureWarning)
```

Out[165]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=
True,
          intercept_scaling=1, max_iter=100, multi_class='warn',
          n_jobs=None, penalty='l2', random_state=None, solver='warn',
          tol=0.0001, verbose=0, warm_start=False)
```

In [166]:
```
accuracy = diabetesCheck.score(testData, testLabel)
```

In [167]:
```
print("accuracy = ", accuracy * 100, "%")
```

```
accuracy =  78.0 %
```

In [168]:
```
coeff = list(diabetesCheck.coef_[0])
```

In [169]:
```
features = pd.DataFrame()
```

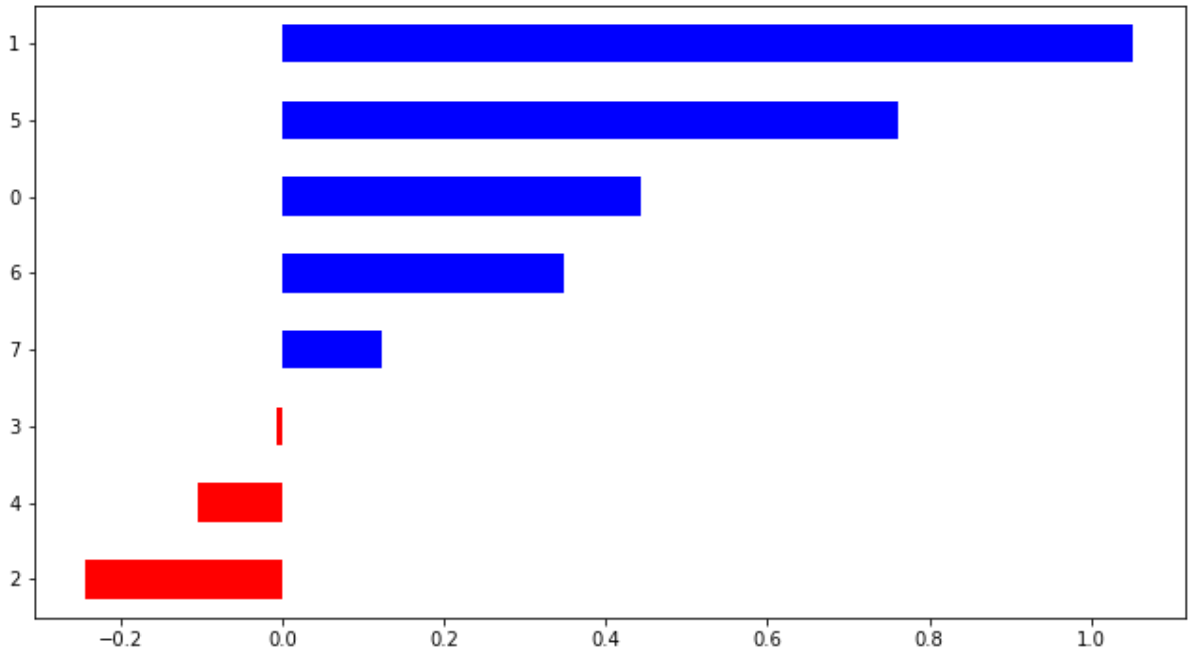
In [170]:
```
features['importance'] = coeff
```

In [171]:
```
features.sort_values(by=['importance'], ascending=True, inplace=True)
```

In [172]:
```
features['positive'] = features['importance'] > 0
```

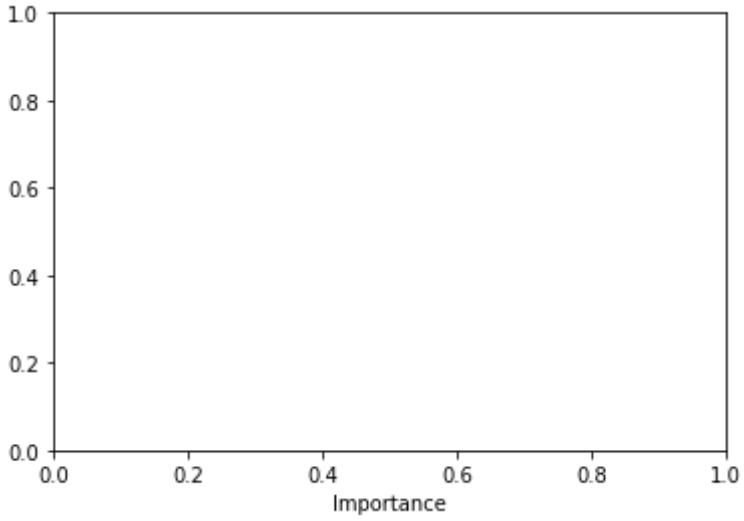
In [173]:
```
features.importance.plot(kind='barh', figsize=(11, 6),color = features.
positive.map({True: 'blue', False: 'red'}))
```

Out[173]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x18305d1d198>
```




In [174]:
```
plt.xlabel('Importance')
```

Out[174]:
```
Text(0.5, 0, 'Importance')
```




In [175]:
```
joblib.dump([diabetesCheck, means, stds], 'diabeteseModel.pkl')
```

Out[175]:
```
['diabeteseModel.pkl']
```

In [176]:
```
diabetesLoadedModel, means, stds = joblib.load('diabeteseModel.pkl')
```

In [177]:
```
accuracyModel = diabetesLoadedModel.score(testData, testLabel)
```

In [178]:
```
print("accuracy = ",accuracyModel * 100,"%")
```

```
accuracy =  78.0 %
```

In [179]:
```
print(dfCheck.head())
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI
\
750            4      136             70              0        0  31.2

751            1      121             78             39       74  39.0

752            3      108             62             24        0  26.0
```

```
753            0      181             88             44      510  43.3

754            8      154             78             32        0  32.4


     DiabetesPedigreeFunction  Age  Outcome
750                     1.182   22        1
751                     0.261   28        0
752                     0.223   25        0
753                     0.222   26        1
754                     0.443   45        1
```

In [180]:
```
sampleData = dfCheck[:1]
```

In [181]:
```
sampleDataFeatures = np.asarray(sampleData.drop('Outcome',1))
```

In [182]:
```
sampleDataFeatures = (sampleDataFeatures - means)/stds
```

In [183]:
```
predictionProbability = diabetesLoadedModel.predict_proba(sampleDataFea
tures)
```

In [184]:
```
prediction = diabetesLoadedModel.predict(sampleDataFeatures)
```

In [185]:
```
print('Probability:', predictionProbability)
```

```
Probability: [[0.4385153 0.5614847]]
```

In [186]:
```
print('prediction:', prediction)
```

```
prediction: [1]
```