In [1]:

```
from glob import glob
import pandas as pd
import matplotlib.pyplot as plt

filepath = r"C:\\Users\\Pujachouhan\\OneDrive\\Desktop\\Activity Recognition from Single Chest-Mou
nted Accelerometer\\"
filesDir = glob(filepath + "/*.csv")
final_acc = pd.DataFrame()
```

In [3]:

```
#Reading all the files at once
pID = 0
for pID, filename in enumerate(filesDir):
    acc = pd.read_csv(filename, index_col = None, header=None)
    acc['User ID'] = pID + 1
    final_acc = final_acc.append(acc)

#Keeping only the required variables
del final_acc[0]
final_acc.columns = ['X-acceleration', 'Y-acceleration', 'Z-acceleration', 'Activity ID', 'User
ID']
```

In [4]:

```
#Basic information about the dataset
print("Dataser Info: ")
print(final_acc.info())
print("Dataset Description: ")
print(final_acc.iloc[:, 0:3].describe())
```

```
Dataser Info:
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1926896 entries, 0 to 166740
Data columns (total 5 columns):
 #   Column          Dtype
---  ------          -----
 0   X-acceleration  int64
 1   Y-acceleration  int64
 2   Z-acceleration  int64
 3   Activity ID     int64
 4   User ID         int64
dtypes: int64(5)
memory usage: 88.2 MB
None
Dataset Description:
       X-acceleration  Y-acceleration  Z-acceleration
count    1.926896e+06    1.926896e+06    1.926896e+06
mean     1.987652e+03    2.382523e+03    1.970596e+03
std      1.113578e+02    1.003151e+02    9.445893e+01
min      2.820000e+02    2.000000e+00    1.000000e+00
25%      1.904000e+03    2.337000e+03    1.918000e+03
50%      1.992000e+03    2.367000e+03    1.988000e+03
75%      2.076000e+03    2.413000e+03    2.032000e+03
max      3.828000e+03    4.095000e+03    4.095000e+03
```

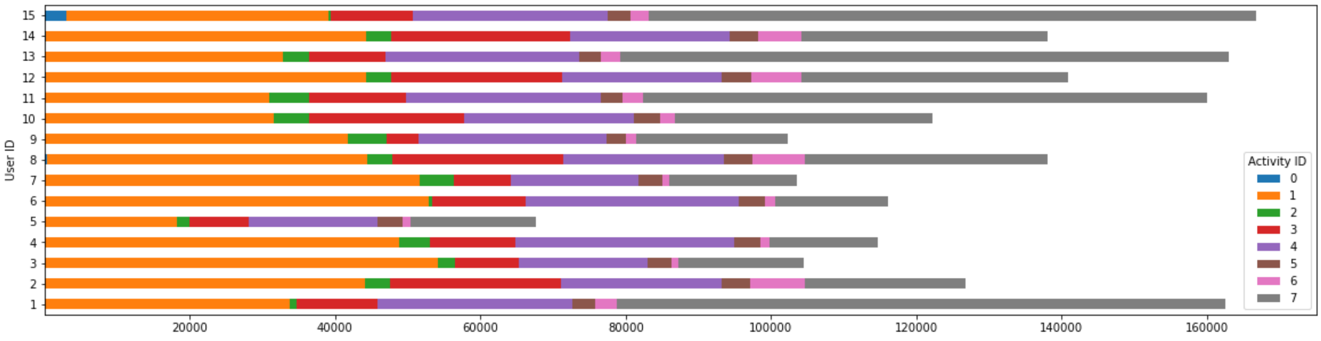
Data Exploration

In [5]:

```
test = pd.crosstab(index = final_acc.iloc[:,-1], columns = final_acc.iloc[:,-2])
test.plot(kind = 'barh', stacked = True, figsize = (20,5))
```

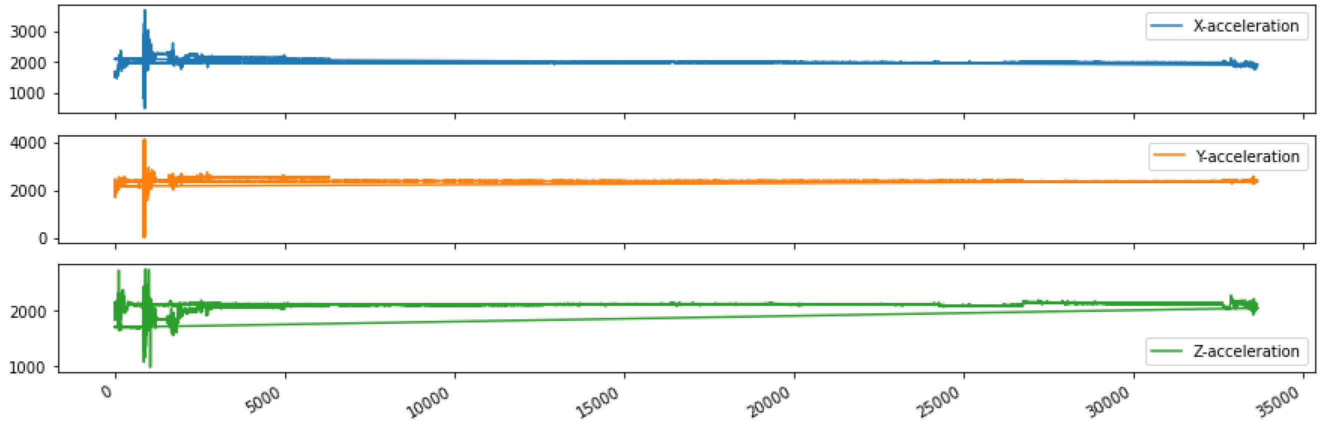
Out[5]:

```
<AxesSubplot:ylabel='User ID'>
```



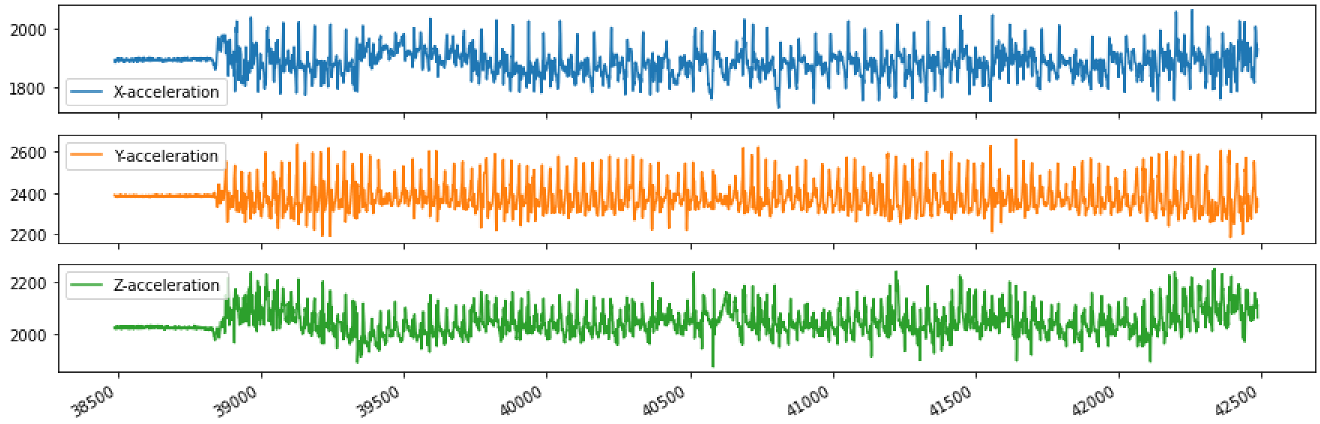
In [6]:

```
#Activity ID 1- Working at computers
expOne = final_acc[final_acc['Activity ID'] == 1]
expOne = expOne[['X-acceleration', 'Y-acceleration', 'Z-acceleration']]
expOne = expOne[:40000]
expOne = expOne.plot(subplots = True, figsize = (15, 5))
```



In [7]:

```
#Activity ID 4- Walking
expFour = final_acc[final_acc["Activity ID"]==4]
expFour = expFour[['X-acceleration', 'Y-acceleration', 'Z-acceleration']]
expFour = expFour[:4000]
expFour = expFour.plot(subplots = True, figsize = (15, 5))
```



Data Modelling

In [8]:

```
from sklearn.model_selection import train_test_split
x = final_acc.iloc[:, 0:3] #Features
y = final_acc.iloc[:, -2] #Target variable
```

```
#Sp;itiing the data into train and test, keeping 70% in train and rest in test
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.30)
print("x train\n", x_train.head(5), "\n")
print("y train\n", y_train.head(5), "\n")
print("x test\n", x_test.head(5), "\n")
print("y test\n", y_test.head(5), "\n")
```

```
x train
         X-acceleration  Y-acceleration  Z-acceleration
9278              2009            2386            2032
55648             1800            2339            1957
113851            1800            2332            1988
108642            2061            2470            1905
123339            1867            2378            2007

y train
 9278      1
55648     3
113851    7
108642    7
123339    7
Name: Activity ID, dtype: int64

x test
         X-acceleration  Y-acceleration  Z-acceleration
35666             1977            2272            1745
106957            2028            2388            1901
15505             1960            2340            1838
107039            2055            2383            2023
107132            2027            2391            1909

y test
 35666     1
106957    7
15505     1
107039    7
107132    7
Name: Activity ID, dtype: int64
```

Feature Selection using Hill Climbing

In [9]:

```
from sklearn.utils import shuffle
from sklearn.tree import DecisionTreeClassifier as dtc

new_Index = []
MaxScore = 0.0
column_no = 3
randomIndex = shuffle(range(0, column_no), random_state = 0)

for i in range(0, column_no):
    new_Index.append(randomIndex[i])
    newData = final_acc.iloc[:, new_Index]
    X_train, X_test, Y_train, Y_test = train_test_split(newData, y, test_size = 0.4, random_state =
0)
    classifier = dtc(criterion = 'gini', max_depth = 15)
    fit = classifier.fit(X_train, Y_train)
    cur_score = classifier.score(X_test, Y_test)
    if cur_score < MaxScore:
        new_index.remove(randomIndex[i])
    else:
        MaxScore = cur_score
        print("Score with " + str(len(new_Index)) + ' selected features' + str(cur_score))
```

```
Score with 1 selected features0.4744673756647668
Score with 2 selected features0.6429908700384945
Score with 3 selected features0.7381152863605874
```

Selecting the parameters for the Decision Tree Classifier (Tuning the parameters manually)

In [10]:

```
from sklearn import metrics
from sklearn.metrics import accuracy_score as acc

accur = []
accur2 = []
val = 0
val2 = 0

print("\n Criterion : Gini\n")
for i in range(1,20):
    k = i+1
    decisionTreeClassifier = dtc(criterion = 'gini', max_depth = k)
    decisionTreeClassifier.fit(x_train, y_train)
    y_predict = decisionTreeClassifier.predict(x_test)
    a = acc(y_test, y_predict)*100
    if a > val:
        val = a
        ind = k
    accur.append(a)
    print("Accuracy for criterion GINI and Max_depth = ", k, 'is', a, '%')

print("\n Criterion : Entropy\n")
for i in range(1,20):
    k = i+1
    decisionTreeClassifier = dtc(criterion = 'entropy', max_depth = k)
    decisionTreeClassifier.fit(x_train, y_train)
    y_predict = decisionTreeClassifier.predict(x_test)
    a = acc(y_test, y_predict)*100
    if a > val2:
        val2 = a
        ind = k
    accur2.append(a)
    print("Accuracy for criterion ENTROPY and Max_depth = ", k, 'is', a, '%')

print("\n")
if max(accur) > max(accur2):
    print('Criterion selecter as GINI and max depth', ind, ' will give us an accuracy score of ',
max(accur))
    plt.figure(figsize =(16,5))
    plt.title('Model Accuracy Score: \n')
    plt.ylabel("Accuracy Scores: (in %)")
    plt.ylim(40, 100)
    plt.xlim(0,25)
    plt.xlabel("Max_depth")
    plt.plot(range(1, 20), accur)
    plt.vlines(ind, plt.ylim()[0], plt.ylim()[1], linestyles = 'dashed', colors = 'red')
else:
    print('Criterion selecter as ENTROPY and max depth', ind, ' will give us an accuracy score of
', max(accur))
    plt.figure(figsize =(16,5))
    plt.title('Model Accuracy Score: \n')
    plt.ylabel("Accuracy Scores: (in %)")
    plt.ylim(40, 100)
    plt.xlim(0,25)
    plt.xlabel("Max_depth")
    plt.plot(range(1, 20), accur2)
    plt.vlines(ind, plt.ylim()[0], plt.ylim()[1], linestyles = 'dashed', colors = 'red')
```

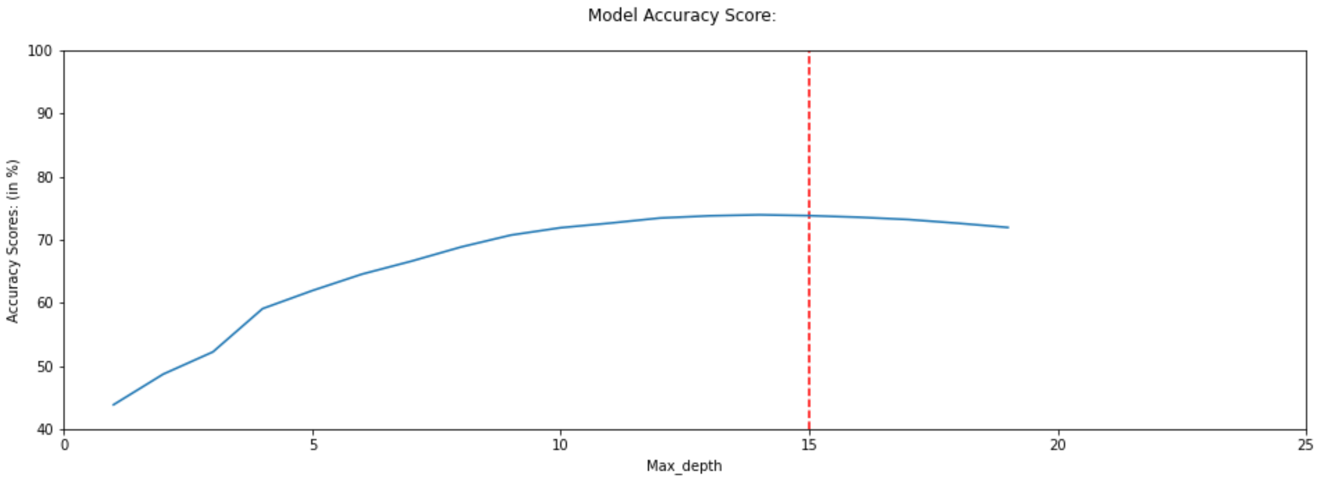
```
 Criterion : Gini

Accuracy for criterion GINI and Max_depth =  2 is 43.85428037137435 %
Accuracy for criterion GINI and Max_depth =  3 is 48.70179857421865 %
Accuracy for criterion GINI and Max_depth =  4 is 52.22629132508403 %
Accuracy for criterion GINI and Max_depth =  5 is 59.08516111398466 %
Accuracy for criterion GINI and Max_depth =  6 is 61.9424324777838 %
Accuracy for criterion GINI and Max_depth =  7 is 64.55405842555128 %
Accuracy for criterion GINI and Max_depth =  8 is 66.62197765318673 %
Accuracy for criterion GINI and Max_depth =  9 is 68.87067114825392 %
Accuracy for criterion GINI and Max_depth =  10 is 70.74899363224806 %
Accuracy for criterion GINI and Max_depth =  11 is 71.9154633789392 %
Accuracy for criterion GINI and Max_depth =  12 is 72.64945880163096 %
Accuracy for criterion GINI and Max_depth =  13 is 73.4505742394074 %
Accuracy for criterion GINI and Max_depth =  14 is 73.8038192672501 %
Accuracy for criterion GINI and Max_depth =  15 is 73.96020198280829 %
Accuracy for criterion GINI and Max_depth =  16 is 73.82717288074608 %
Accuracy for criterion GINI and Max_depth =  17 is 73.57547282417843 %
Accuracy for criterion GINI and Max_depth =  18 is 73.20908057688615 %
Accuracy for criterion GINI and Max_depth =  19 is 72.62247240381339 %
Accuracy for criterion GINI and Max_depth =  20 is 71.95507802701754 %

 Criterion : Entropy

Accuracy for criterion ENTROPY and Max_depth =  2 is 43.91240492051987 %
Accuracy for criterion ENTROPY and Max_depth =  3 is 48.647825778583524 %
Accuracy for criterion ENTROPY and Max_depth =  4 is 52.16643687864252 %
Accuracy for criterion ENTROPY and Max_depth =  5 is 59.04104873293673 %
Accuracy for criterion ENTROPY and Max_depth =  6 is 61.6547505574594 %
Accuracy for criterion ENTROPY and Max_depth =  7 is 65.0254554387106 %
Accuracy for criterion ENTROPY and Max_depth =  8 is 66.55676052512763 %
Accuracy for criterion ENTROPY and Max_depth =  9 is 68.90302022768908 %
Accuracy for criterion ENTROPY and Max_depth =  10 is 70.32551477418785 %
Accuracy for criterion ENTROPY and Max_depth =  11 is 71.96511143133432 %
Accuracy for criterion ENTROPY and Max_depth =  12 is 72.86362008687544 %
Accuracy for criterion ENTROPY and Max_depth =  13 is 73.60194025280718 %
Accuracy for criterion ENTROPY and Max_depth =  14 is 73.92249022175554 %
Accuracy for criterion ENTROPY and Max_depth =  15 is 73.9534553833539 %
Accuracy for criterion ENTROPY and Max_depth =  16 is 73.89204402934598 %
Accuracy for criterion ENTROPY and Max_depth =  17 is 73.55263817987127 %
Accuracy for criterion ENTROPY and Max_depth =  18 is 72.9643001095025 %
Accuracy for criterion ENTROPY and Max_depth =  19 is 72.24310592680112 %
Accuracy for criterion ENTROPY and Max_depth =  20 is 71.42745935173829 %


Criterion selecter as GINI and max depth 15  will give us an accuracy score of  73.96020198280829
```



```
In [ ]:
```