Spotify Data Analysis

```
> import matplotlib
  import numpy as np
  import matplotlib.pyplot as plt
  # %matplotlib inline
  import time
  import pandas as pd

/databricks/python/local/lib/python2.7/site-packages/matplotlib/font_manager.py:
273: UserWarning: Matplotlib is building the font cache using fc-list. This may
 take a moment.
  warnings.warn('Matplotlib is building the font cache using fc-list. This may t
ake a moment.')
```

- Loading the "user_data_sample.csv" in to a pandas dataframe

```
> users_df = pd.read_csv("https://s3-us-west-
  1.amazonaws.com/vamsinallabothubucket/user_data_sample.csv")
  users_df.describe()

Out[2]:
       acct_age_weeks
count     9565.000000
mean        74.094093
std         76.810872
min         -1.000000
25%         15.000000
50%         49.000000
75%        113.000000
max        363.000000
```

```
> users_df.head()

Out[3]:
   gender age_range country  acct_age_weeks                          user_id
0    male   25 - 29      FR             329  97f47c9fba714ca68320b8a80e010a1a
1  female   45 - 54      US             178  d615ca85849d458e9a5d755ec4727e8f
```

```
2  female   18 - 24      DE              68  6c83a5bf63b74f85b106ac7e7e015a1b
3  female   30 - 34      US               8  530fcedb3f244e6f91ecb326740005eb
4  female   30 - 34      FR              42  d2ed6a815eda4f61aa346b7936d03ef7
```

- Loading the end_song_sample.csv from AWS into another pandas dataframe.

```
> songs_df = pd.read_csv("https://s3-us-west-
  1.amazonaws.com/vamsinallabothubucket/end_song_sample.csv")
```

```
> songs_df.head()
```

Out[5]:

```
   ms_played     context                          track_id product  \
0     330962       album  2ab4f3b3a6c34fbaba95c2451b65efbd    open
1       7476       album  0f5f2acbcf244490948ac2e63adade73    open
2     227280  collection  0f4a2173eb1f4aa9b8693ad7a92fab73    open
3        325    playlist  affc7467b68e4dfab9d1d7b9ec8d4673    open
4     204196  collection  427fd37cbfe640a8a78179477c9f33d3    open

   end_timestamp                           user_id
0   1.444790e+09  a9abbb14c8544898a0e06feb94f8051e
1   1.444790e+09  a9abbb14c8544898a0e06feb94f8051e
2   1.444797e+09  a9abbb14c8544898a0e06feb94f8051e
3   1.444796e+09  a9abbb14c8544898a0e06feb94f8051e
4   1.444799e+09  a9abbb14c8544898a0e06feb94f8051e
```

- Merging both the song and user dataframes into a single dataframe where the user_id matches in both the data drames

```
> combined_df = pd.merge(users_df,songs_df,on='user_id')
  combined_df.head()
```

Out[6]:

```
  gender age_range country  acct_age_weeks                           user_id  \
0   male   25 - 29      FR             329  97f47c9fba714ca68320b8a80e010a1a
1   male   25 - 29      FR             329  97f47c9fba714ca68320b8a80e010a1a
2   male   25 - 29      FR             329  97f47c9fba714ca68320b8a80e010a1a
```

```
3   male   25 - 29      FR             329  97f47c9fba714ca68320b8a80e010a1a
4   male   25 - 29      FR             329  97f47c9fba714ca68320b8a80e010a1a

   ms_played   context                          track_id product  \
0     408000  playlist  f9105d43bf1940caa82802c97b59684f    free
1     292429  playlist  558bef60e515435c9c2e64aab10c83a6    free
2     359769  playlist  e8b1cd3e2956436a982a97dd76490a8d    free
3     329085  playlist  f017dad7ef8e40e682523b75c07ea145    free
4     337425  playlist  8d48d9cd55074b529a8cdd63ea90bce1    free

   end_timestamp
0   1.443822e+09
1   1.443822e+09
2   1.443829e+09
3   1.443816e+09
4   1.443826e+09
```

- The combined dataframe contains a total of 1342891 rows and 10 columns

```
> combined_df.shape # number of rows and columns
  #combined_df.drop(['column1', 'column2', 'column3'], axis=1,
  inplace=True) <- for removing the rows and columns from the  dataframe
  #axis = 1 for columns and axis = 0 for rows

Out[7]: (1342891, 10)
```

```
> combined_df.age_range.value_counts()

Out[8]:
18 - 24    490827
25 - 29    256564
0 - 17     171452
30 - 34    154528
35 - 44    146816
45 - 54     72892
55+         48972
dtype: int64
```

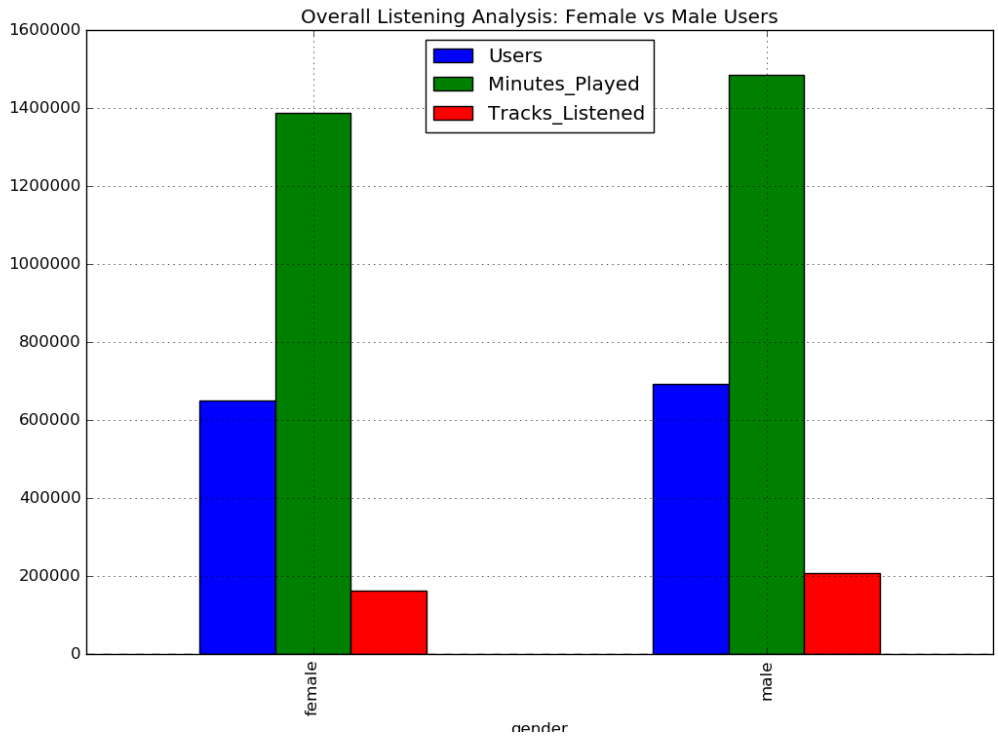
- finding the total number of tracks played and minutes(60000 millisec/min) listened and number of users under each gender

```
> # finding the total number of tracks played and minutes(60000
  millisec/min) listened and number of users under each gender
  usage_diff = combined_df.groupby('gender').aggregate({'gender':'count',
  'ms_played':lambda x: sum(x)/60000, 'track_id':lambda x:
  len(x.unique()),})
  usage_diff.rename(columns={'gender':'Users',
  'ms_played':'Minutes_Played', 'track_id':'Tracks_Listened'},
  inplace=True)
  usage_diff.head(2).plot(kind='bar',figsize=(12, 8),title="Overall
  Listening Analysis: Female vs Male Users")
  usage_diff
```

Out[9]:

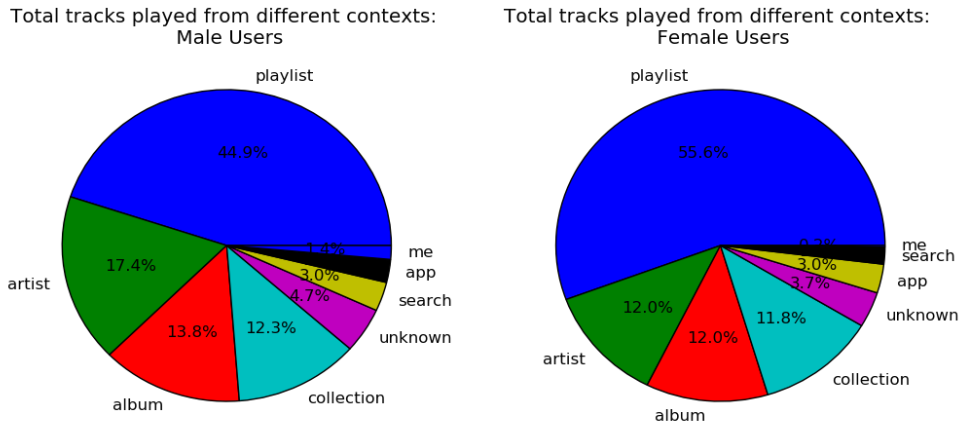
gender	Users	Minutes_Played	Tracks_Listened
female	649178	1388677	161590
male	691479	1486112	207131
unknown	2234	5978	1409

```
> display()
```



- Finding the differences in listening contexts of both the Male and Female users. The below graph shows the difference in which male and female users listen to tracks from different contexts

```
> fig, axs = plt.subplots(ncols=2, figsize=(12, 5))
  male_context =
  combined_df[combined_df['gender']=='male'].groupby(['context']).size().o
  rder(ascending=False)
  female_context =
  combined_df[combined_df.gender=='female'].groupby(['context']).size().or
  der(ascending=False)
  male_context.plot(kind='pie', ax=axs[0],autopct='%1.1f%%', title='Total
  tracks played from different contexts: \n Male Users')
  female_context.plot(kind='pie', autopct='%1.1f%%', ax=axs[1],
  title='Total tracks played from different contexts: \n Female Users')
  display()
```



- Capturing the number of male and female users in the specified age_range in to male_ageRange and female_ageRange pandas series

```
> female_ageRange = combined_df.loc[combined_df.gender=='male',
  'age_range'].value_counts()
  male_ageRange = combined_df.loc[combined_df.gender=='female',
  'age_range'].value_counts()
```

```
> female_ageRange
```

```
Out[13]:
18 - 24    240384
25 - 29    146195
30 - 34     86096
35 - 44     76952
0 - 17      65372
45 - 54     42004
55+         34416
dtype: int64
```

```
> male_ageRange
```

```
Out[14]:
18 - 24    250243
25 - 29    110222
0 - 17     106080
35 - 44     69485
30 - 34     68276
45 - 54     30792
55+         14080
dtype: int64
```

- Converting the male_ageRange and female_ageRange series in to mdf and fmdf DataFrames

```
> mdf = pd.DataFrame(male_ageRange)

  fmdf = pd.DataFrame(female_ageRange)

  mdf['age_range'] = mdf.index

  fmdf['age_range'] = fmdf.index

  mdf.reset_index(drop=True, inplace=True)

  fmdf.reset_index(drop=True, inplace=True)
```

```
> mdf
```

Out[16]:

	0	age_range
0	250243	18 - 24
1	110222	25 - 29
2	106080	0 - 17
3	69485	35 - 44
4	68276	30 - 34
5	30792	45 - 54
6	14080	55+

```
> fmdf
```

Out[17]:

	0	age_range
0	240384	18 - 24
1	146195	25 - 29
2	86096	30 - 34
3	76952	35 - 44
4	65372	0 - 17
5	42004	45 - 54
6	34416	55+

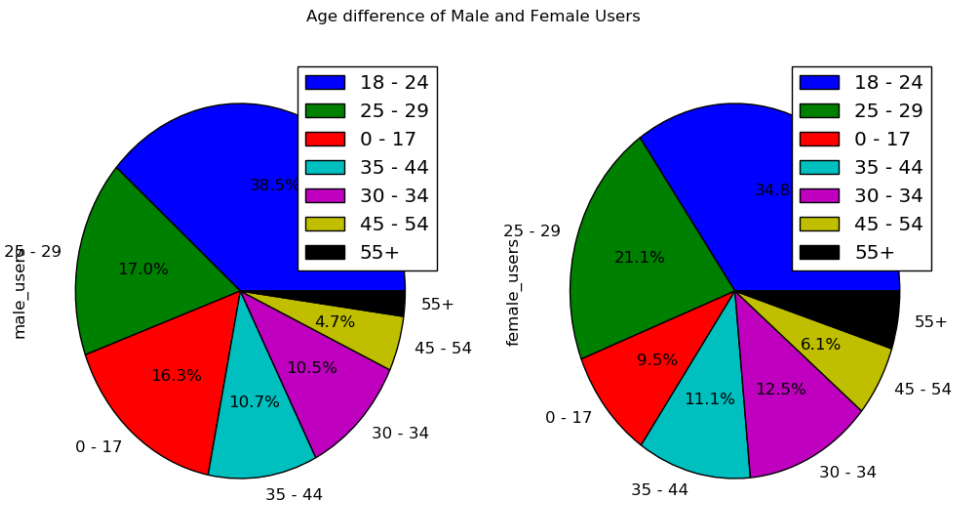
- Merging the DataFrames on the age_range value

```
> ageRange_df = pd.merge(mdf,fmdf, on='age_range')

  ageRange_df.columns = ['male_users', 'age_range', 'female_users']

  ageRange_df.set_index('age_range', inplace = True)
```

- Comparing the age range of Male and Female users listening to the songs on the Spotify

```
> ageRange_df.plot(kind='pie',figsize=(12,6), autopct='%1.1f%%',
  title="Age difference of Male and Female Users", subplots = True)
```

```
display()
```



- A function to classify the session based on hour of the day

```
> def get_session(x):
      end_time = time.localtime(x).tm_hour
      if 0 < end_time <= 6:
          return 'Mid Night'
      elif 6 < end_time <= 12:
          return 'Morning'
      elif 12 < end_time <= 18:
          return 'Noon'
      else:
          return 'Evening'
```

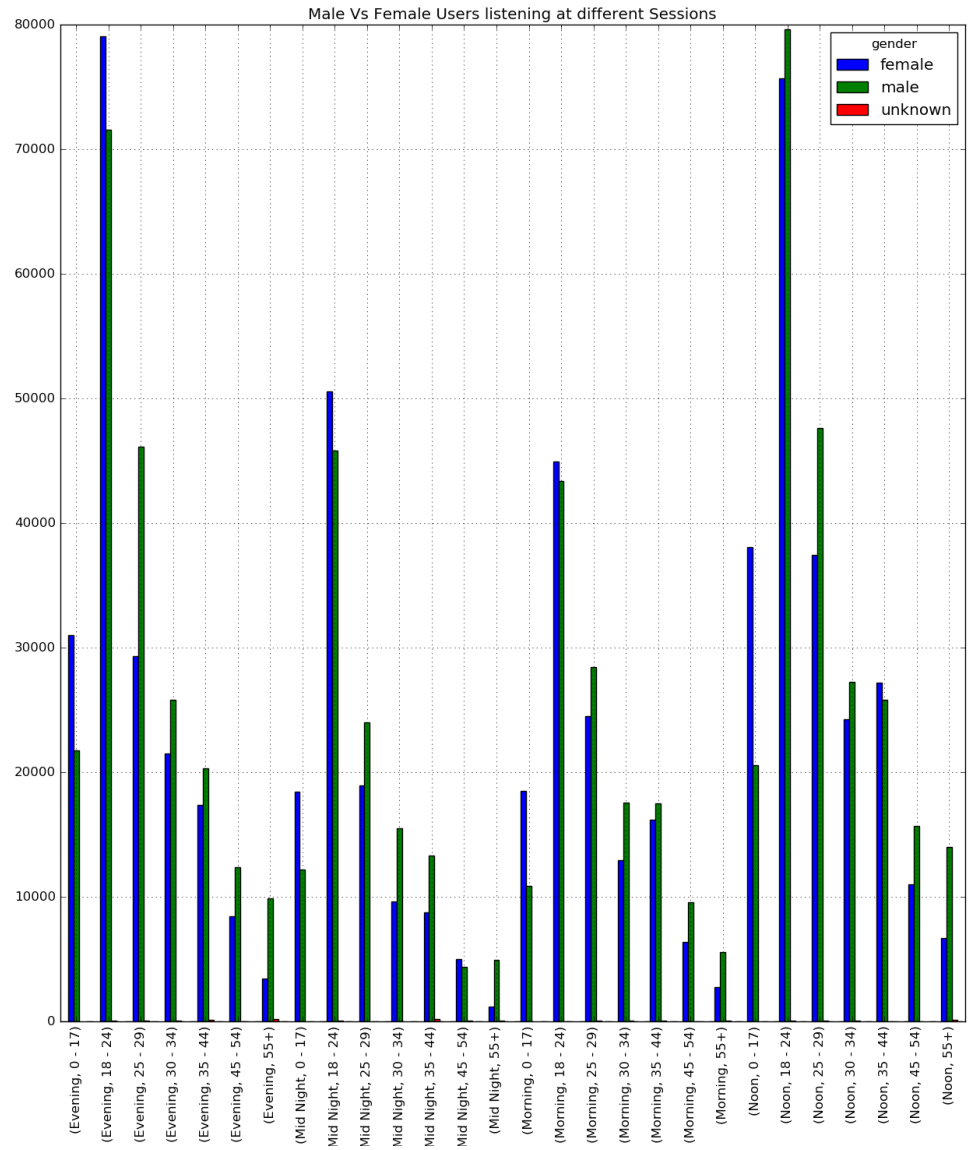
- using the get_session function, set the session for each user based on the end_timestamp and add the "session" series to the combined_df DataFrame

```
> combined_df['session'] = combined_df['end_timestamp'].apply(get_session)
  combined_df.head()

Out[21]:
  gender age_range country  acct_age_weeks                          user_id  \
```

```
0   male   25 - 29      FR             329  97f47c9fba714ca68320b8a80e010a1a
1   male   25 - 29      FR             329  97f47c9fba714ca68320b8a80e010a1a
2   male   25 - 29      FR             329  97f47c9fba714ca68320b8a80e010a1a
3   male   25 - 29      FR             329  97f47c9fba714ca68320b8a80e010a1a
4   male   25 - 29      FR             329  97f47c9fba714ca68320b8a80e010a1a

   ms_played   context                          track_id product  \
0     408000  playlist  f9105d43bf1940caa82802c97b59684f    free
1     292429  playlist  558bef60e515435c9c2e64aab10c83a6    free
2     359769  playlist  e8b1cd3e2956436a982a97dd76490a8d    free
3     329085  playlist  f017dad7ef8e40e682523b75c07ea145    free
4     337425  playlist  8d48d9cd55074b529a8cdd63ea90bce1    free

   end_timestamp  session
0   1.443822e+09  Evening
1   1.443822e+09  Evening
2   1.443829e+09  Evening
3   1.443816e+09  Evening
4   1.443826e+09  Evening
```

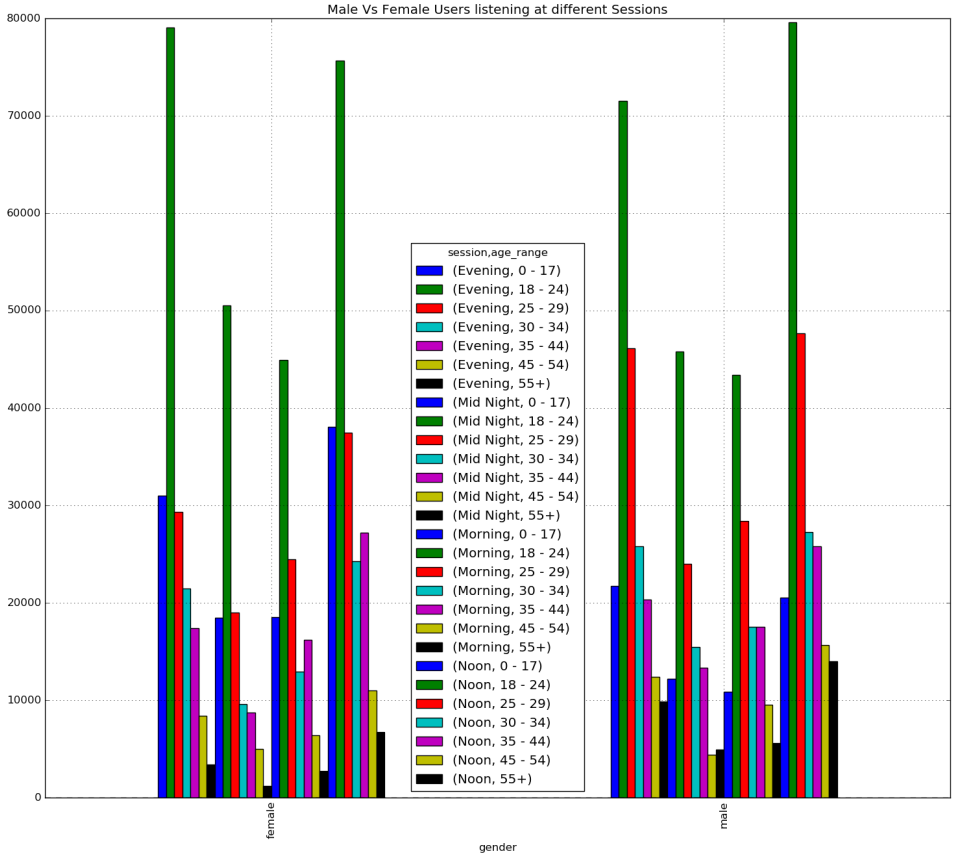
- Plotting the graph showing the difference in number male and female users listening at different sessions

```
> df = combined_df.groupby(['session','age_range', 'gender']).size()
  df.unstack().plot(kind='bar', figsize=(15,16),title="Male Vs Female
  Users listening at different Sessions")
  display()
```



- Plotting the graph showing the difference in number male and female users listening at different sessions

```
> df.unstack(level=[0,1]).drop(['unknown'], axis=0).plot(kind="bar",
  figsize=(18, 15), title="Male Vs Female Users listening at different
  Sessions")
  display()
```



- categorizing the total users based on the session and age_Range for each session. Finding the average amount of stream per each session

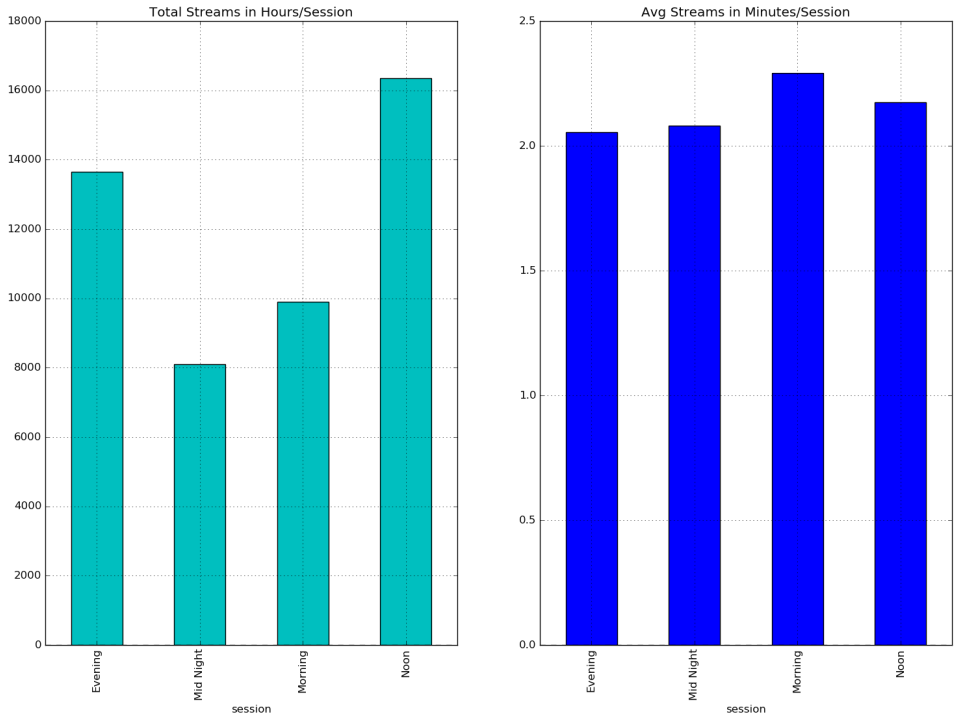
```
> s = combined_df.groupby(['session','age_range']).size()
  s_top = s.order(ascending=False).head(10)
  s_avg = combined_df.groupby(['session'])['ms_played'].agg({'avg':lambda
  x: np.mean(x)/60000,'total':lambda x: np.sum(x)/3600000})
```

```
> s.head()

Out[25]:
session  age_range
Evening  0 - 17        52776
         18 - 24      150724
```

```
          25 - 29       75490
          30 - 34       47316
          35 - 44       37825
dtype: int64
```

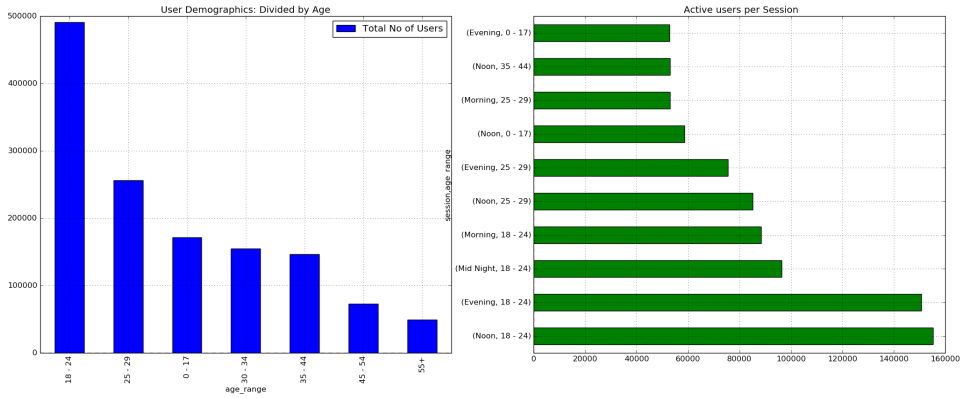
- Total stream in hours per session and the Average time streaming of song in each session

```
> fig, axs = plt.subplots(ncols=2,figsize=(18,12))
  s_avg['total'].plot(kind='bar',title='Total Streams in
  Hours/Session',ax=axs[0],color='c')
  s_avg['avg'].plot(kind='bar',title='Avg Streams in
  Minutes/Session',ax=axs[1])
  display()
```



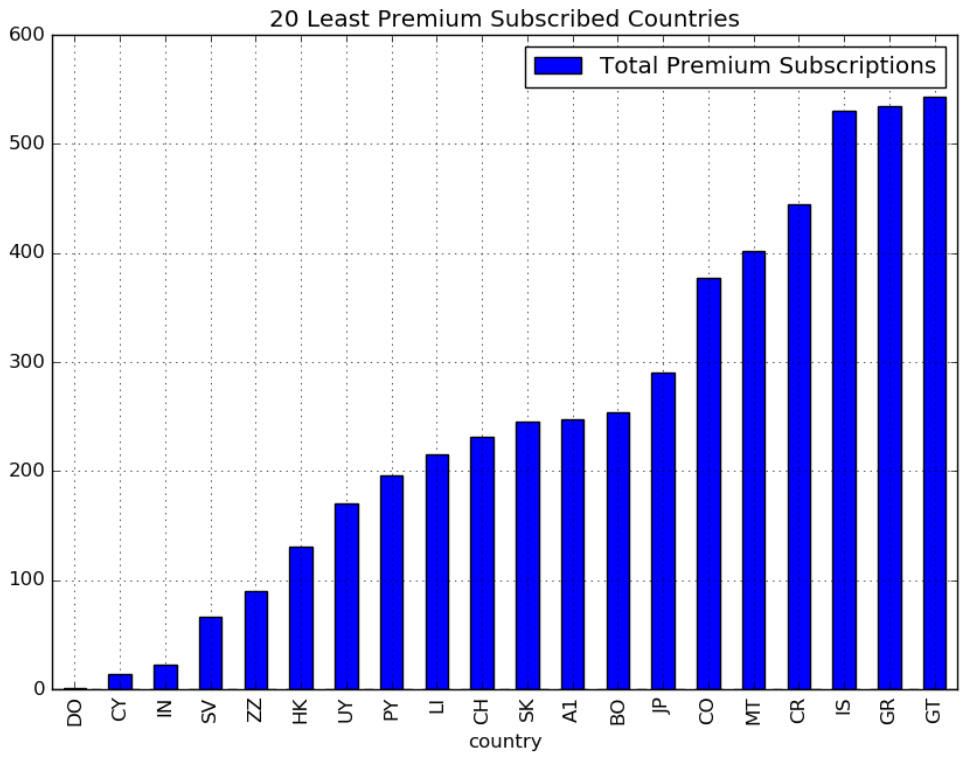
- No of users active per session and number of users of certain age range

```
> fig, axs = plt.subplots(ncols=2,figsize=(25,9))
  combined_df.groupby('age_range')['age_range'].agg({'Total No of
  Users':np.size}).sort('Total No of
  Users',ascending=False).plot(kind='bar',ax=axs[0],title="User
  Demographics: Divided by Age")
  s_top.plot(kind='barh',color=['g'],legend=False,ax=axs[1],title=" Active
  users per Session")
  display()
```



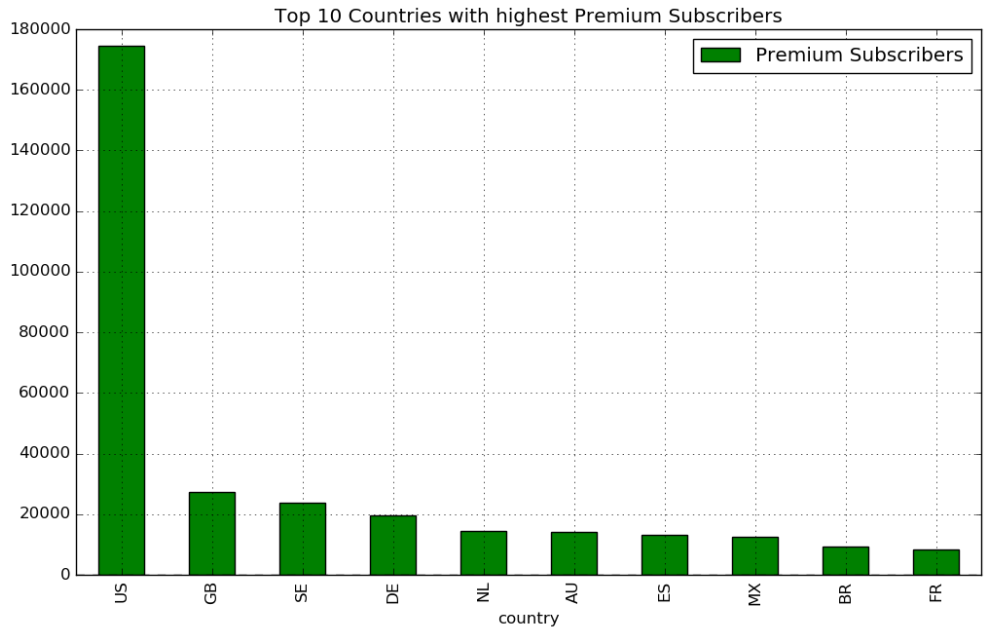
- Top 10 countries with least premium subscribers

```
> subs=combined_df.groupby(['country','product']).size().reset_index()
  subs[subs['product']=='premium'].rename(columns={0:'Total Premium
  Subscriptions'}).sort('Total Premium
  Subscriptions',ascending=True).set_index(['country']).head(20).plot(kind
  ='bar',figsize=(10,7),title="20 Least Premium Subscribed
  Countries",stacked=True)
  display()
```



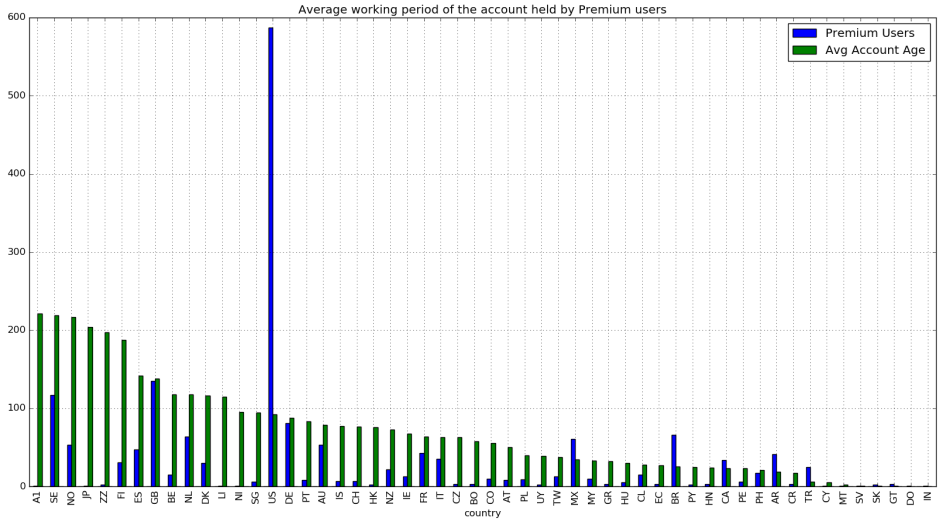
- Top 10 Countries with more premium subscirbers

```
> subs[subs['product']=='premium'].rename(columns={0:'Premium
  Subscribers'}).sort('Premium
  Subscribers',ascending=False).set_index(['country']).head(10).plot(kind=
  'bar',figsize=(12,7),title="Top 10 Countries with highest Premium
  Subscribers",color='g', stacked=True)
  display()
```

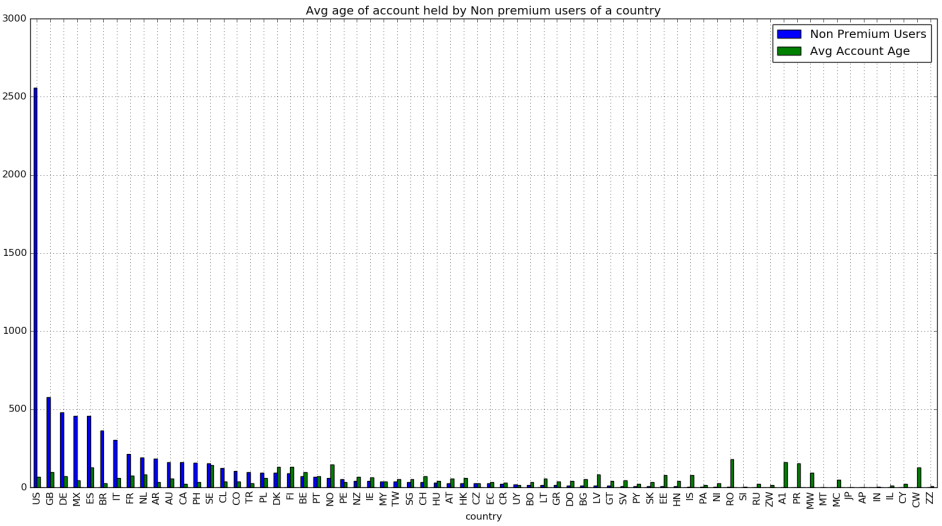


Longevity of Premium subscriber accounts in each country

```
> usr=combined_df.groupby(['user_id','country','age_range','acct_age_weeks
  '])
  premium_usr=usr.aggregate({'product':lambda x: list(set(x))})
  premium_usr['Premium'] = premium_usr['product'].apply(lambda x:
  x.__contains__('premium'))
  premium_usr[premium_usr['Premium']==True].reset_index().groupby('country
  ').aggregate({'country':'count','acct_age_weeks':np.mean}).sort(['acct_a
  ge_weeks'],ascending=False).rename(columns={'country':'Premium
  Users','acct_age_weeks':'Avg Account Age'}).plot(kind='bar',figsize=
  (20,10),title='Average working period of the account held by Premium
  users')
  display()
```




```
> premium_usr[premium_usr['Premium'] ==
  False].reset_index().groupby('country').aggregate({'country':'count','ac
  ct_age_weeks':np.mean}).sort(['country'],ascending=False).rename(columns
  ={'country':'Non Premium Users','acct_age_weeks':'Avg Account
  Age'}).plot(kind='bar',figsize=(20,10),title='Avg age of account held by
  Non premium users of a country')
  display()
```



Graph showing number of Premium users and Free users

```
>
fig, axs = plt.subplots(ncols=2, figsize=(15, 6))
combined_df[combined_df['product']=='premium'].groupby(['context']).size
().order(ascending=False).plot(kind='pie',ax=axs[0],title='Premium
Content Users: Songs Context', autopct='%1.1f%%')
combined_df[combined_df['product']!='premium'].groupby(['context']).size
().order(ascending=False).plot(kind='pie',ax=axs[1],title='Free Content
Users: Songs Context', autopct='%1.1f%%')
display()
```