# Reading Data

In [1]:
```
import pandas as pd
import numpy as np
import gc
import time
import warnings

#stats
from scipy.misc import imread
from scipy import sparse
import scipy.stats as ss

#viz
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec
import seaborn as sns
from wordcloud import WordCloud ,STOPWORDS
from PIL import Image
#import matplotlib_venn as venn

#nlp
import string
import re    #for regex
import nltk
from nltk.corpus import stopwords
import spacy
from nltk import pos_tag
from nltk.stem.wordnet import WordNetLemmatizer
from nltk.tokenize import word_tokenize
# Tweet tokenizer does not split at apostophes which is what we want
from nltk.tokenize import TweetTokenizer

color = sns.color_palette()
sns.set_style("dark")
eng_stopwords = set(stopwords.words("english"))
warnings.filterwarnings("ignore")

lem = WordNetLemmatizer()
tokenizer=TweetTokenizer()
```

In [2]:
```
train = pd.read_csv('./data/train.csv')
```

In [3]:
```
import sys
sys.executable
```

Out[3]:
```
'/usr/bin/python2.7'
```

# Analysis

## Looking at the data

In [35]:
```
train.head()
train=train[ (train.toxic!=0) | (train.severe_toxic!=0) | (train.obscene!=0) | (t
```

Here's a couple of examples of comments, one toxic, and one with no labels.

In [36]:
```
train
```

Out[36]:

| | id | comment_text | toxic | severe_toxic | obscene | threat | in |
|---|---|---|---|---|---|---|---|
| **6** | 0002bcb3da6cb337 | COCKSUCKER BEFORE YOU PISS AROUND ON MY WORK | 1 | 1 | 1 | 0 | |
| **12** | 0005c987bdfc9d4b | Hey... what is it..\n@ \| talk .\nWhat is it..... | 1 | 0 | 0 | 0 | |
| **16** | 0007e25b2121310b | Bye! \n\nDon't look, come or think of comming ... | 1 | 0 | 0 | 0 | |
| **42** | 001810bf8c45bf5f | You are gay or antisemmitian? \n\nArchangel WH... | 1 | 0 | 1 | 0 | |
| **43** | 00190820581d90ce | FUCK YOUR FILTHY MOTHER IN THE ASS, DRY! | 1 | 0 | 1 | 0 | |
| **44** | 001956c382006abd | I'm Sorry \n\nI'm sorry I screwed around with ... | 1 | 0 | 0 | 0 | |
| **51** | 001dc38a83d420cf | GET FUCKED UP. GET FUCKEEED UP. GOT A DRINK T... | 1 | 0 | 1 | 0 | |
| **55** | 0020e7119b96eeeb | Stupid peace of shit stop deleting my stuff as... | 1 | 1 | 1 | 0 | |
| **56** | 0020fd96ed3b8c8b | =Tony Sidaway is obviously a fistfuckee. He lo... | 1 | 0 | 1 | 0 | |
| **58** | 0021fe88bc4da3e6 | My Band Page's deletion. You thought I was gon... | 1 | 0 | 1 | 0 | |
| **59** | 002264ea4d5f2887 | Why can't you believe how fat Artie is? Did yo... | 1 | 0 | 0 | 0 | |
| **65** | 0028d62e8a5629aa | All of my edits are good. Cunts like you who ... | 1 | 0 | 1 | 0 | |
| **79** | 003217c3eb469ba9 | Hi! I am back again!\nLast warning!\nStop undo... | 1 | 0 | 0 | 1 | |
| **86** | 0036621e4c7e10b5 | Would you both shut up, you don't run wikipedi... | 1 | 0 | 0 | 0 | |
| **105** | 00472b8e2d38d1ea | A pair of jew-hating weiner nazi schmucks. | 1 | 0 | 1 | 0 | |
| **151** | 005f59485fcddeb0 | "\n\nSORRY PUCK BUT NO ONE EVER SAID DICK WAS ... | 1 | 0 | 0 | 0 | |
| **159** | 00637960a7ec3436 | "\n\nUNBLOCK ME OR I'LL GET MY LAWYERS ON TO Y... | 1 | 0 | 0 | 0 | |
| **168** | 00686325bcc16080 | You should be fired, you're a moronic wimp who... | 1 | 0 | 0 | 0 | |
| **176** | 006b94add72ed61c | I think that your a Fagget get a oife and burn... | 1 | 0 | 1 | 1 | |
| **181** | 006e87872c8b370c | you are a stupid fuck \n\nand your mother's cu... | 1 | 1 | 1 | 0 | |
| **201** | 007f1839ada915e6 | Your blatant POV pushing \n\nNeither of you gu... | 1 | 0 | 1 | 0 | |
| **206** | 0082b5a7b4a67da2 | Give me a permanat block raseac....!!! remembe... | 1 | 0 | 0 | 0 | |

| | id | comment_text | toxic | severe_toxic | obscene | threat | in |
|---|---|---|---|---|---|---|---|
| **211** | 0086998b34865f93 | Fuck you, block me, you faggot pussy! | 1 | 0 | 1 | 0 | |
| **218** | 008e0818dde894fb | Kill all niggers. \n\nI have hard, that others... | 1 | 0 | 1 | 0 | |
| **231** | 009371b0ef213487 | Burn Deck \n\nIf that'd guy's burn deck is lik... | 1 | 0 | 1 | 0 | |
| **238** | 0097dd5c29bf7a15 | u r a tw@ fuck off u gay boy.U r smelly.Fuck u... | 1 | 0 | 1 | 0 | |
| **268** | 00ab65775c601cf9 | Atheism is full of bias shit | 1 | 0 | 0 | 0 | |
| **278** | 00afb4dec99a231f | Hey why you are spreading misconceptions and t... | 1 | 0 | 0 | 0 | |
| **286** | 00b77cb600c897b4 | "\n\nAnd you are? Let me know when you've craw... | 1 | 0 | 0 | 0 | |
| **295** | 00be7dcac98dc95d | this user is such a worthless goddamn faggot f... | 1 | 0 | 1 | 0 | |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **159253** | fae97a014c011e3a | what do you mean \n\nwhy don't you keep your n... | 1 | 0 | 1 | 0 | |
| **159268** | fb32b002bc46b830 | Hi Bading \nPutang ina mong bakla ka. Fuck you... | 1 | 0 | 1 | 0 | |
| **159274** | fb4cbf4eeabe23d4 | "\n\nStudy some linguistics before you say som... | 1 | 0 | 0 | 0 | |
| **159281** | fb726deec64157bd | LoL!! \n\nyou're GAY!! you will never know how... | 1 | 1 | 1 | 0 | |
| **159290** | fb91faebc0197bd1 | Hey alabamoy boy why dont you stick your head ... | 1 | 0 | 1 | 0 | |
| **159298** | fbb37645ecf5e403 | , are you dumber than you look? asshole. | 1 | 0 | 1 | 0 | |
| **159312** | fbf20e312cd4a78d | Walter Mercado \n\nAntonio, quite frankly, you... | 1 | 1 | 1 | 0 | |
| **159315** | fbf8672ea3b4ddf7 | http://www.nysun.com/article/23698 - public in... | 1 | 0 | 0 | 0 | |
| **159334** | fc3a75b57f1f6923 | Horse's ass \n\nSeriously, dude, what's that h... | 1 | 0 | 1 | 0 | |
| **159336** | fc3efa2f6f025f6d | Oh, fuck off. The pansy Jew would just whine a... | 1 | 0 | 1 | 0 | |
| **159342** | fc4a76f9f0ecd189 | Fuck off turd. Don't ever ban me again you cun... | 1 | 0 | 1 | 0 | |
| **159358** | fc6d45d108129fc8 | Goethean and me\n\nI would like you to know I ... | 1 | 0 | 0 | 0 | |
| **159368** | fc8f351add0fd065 | "\n\n Palin/Satan 2012 \n\nWow, what a surpris... | 1 | 0 | 1 | 0 | |
| **159378** | fcb09a6d428bdb74 | GO AHEAD AND FUCKING BAN ME ~ LIKE THAT WILL H... | 1 | 0 | 1 | 0 | |
| **159382** | fcbc0c74d75584c2 | shut up you goddamn assclown. | 1 | 0 | 1 | 0 | |

| | id | comment_text | toxic | severe_toxic | obscene | threat | in |
|---|---|---|---|---|---|---|---|
| **159386** | fccf0939631ab7c8 | Stop telling lies and trying to promote your p... | 1 | 0 | 0 | 0 | |
| **159394** | fcf5a6ad5918f164 | your boring \n\nand retarded two | 1 | 0 | 0 | 0 | |
| **159398** | fd0129fde97321cb | Why did that idiot revert the reversion I made... | 1 | 0 | 0 | 0 | |
| **159400** | fd052883fa6a8697 | Shalom \n\nSemite, get the fuck out of here. I... | 1 | 1 | 1 | 1 | |
| **159411** | fd2f53aafe8eefcc | Fat piece of shit \n\nyou obese piece of shit.... | 1 | 0 | 1 | 0 | |
| **159423** | fd68ef478b3dfd05 | PS: you're all middle-aged losers at home in ... | 1 | 0 | 0 | 0 | |
| **159448** | fdc92e571d39e7e1 | Yeah i no it sucks. | 1 | 0 | 0 | 0 | |
| **159449** | fdce660ddcd6d7ca | I think he is a gay fag!!! | 1 | 0 | 0 | 0 | |
| **159478** | feb5637c531f933d | "\nThank you. Given the misuse of tools here a... | 1 | 0 | 0 | 0 | |
| **159493** | fef142420a215b90 | FUCKING FAGGOT \n\nLOLWAT. | 1 | 0 | 1 | 0 | |
| **159494** | fef4cf7ba0012866 | "\n\n our previous conversation \n\nyou fuckin... | 1 | 0 | 1 | 0 | |
| **159514** | ff39a2895fc3b40e | YOU ARE A MISCHIEVIOUS PUBIC HAIR | 1 | 0 | 0 | 0 | |
| **159541** | ffa33d3122b599d6 | Your absurd edits \n\nYour absurd edits on gre... | 1 | 0 | 1 | 0 | |
| **159546** | ffb47123b2d82762 | "\n\nHey listen don't you ever!!!! Delete my e... | 1 | 0 | 0 | 0 | |
| **159554** | ffbdbb0483ed0841 | and i'm going to keep posting the stuff u dele... | 1 | 0 | 1 | 0 | |

15294 rows × 8 columns

In [43]:
```
train['comment_text'][12]
```

Out[43]:
```
'Hey... what is it..\n@ | talk .\nWhat is it... an exclusive group of some WP T
ALIBANS...who are good at destroying, self-appointed purist who GANG UP any one
who asks them questions abt their ANTI-SOCIAL and DESTRUCTIVE (non)-contributio
n at WP?\n\nAsk Sityush to clean up his behavior than issue me nonsensical warn
ings...'
```

In [44]:
```
train['comment_text'][6]
```

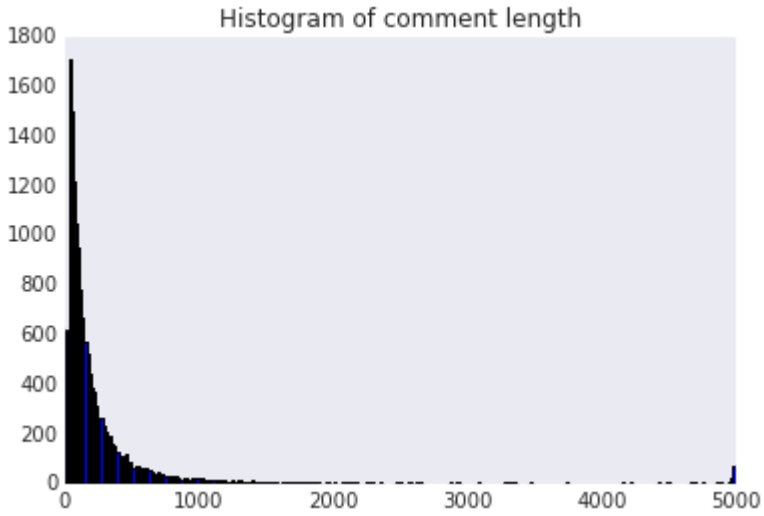
Out[44]:
```
'COCKSUCKER BEFORE YOU PISS AROUND ON MY WORK'
```

The length of the comments varies a lot.

In [45]:
```
lens = train.comment_text.str.len()
#print(lens)
lens.mean(), lens.std(), lens.max()
```

Out[45]:
```
(295.46953053485026, 617.75265941816826, 5000)
```

In [48]:
```
%matplotlib inline
import matplotlib.pyplot as plt
plt.hist(lens.values, bins="auto")
plt.title("Histogram of comment length")
plt.show()
```




We'll create a list of all the labels to predict, and we'll also create a 'none' label so we can see how many comments have no labels. We can then summarize the dataset.

**Now lets analyse statistics of the overall dataset**

In [49]:
```
len(train)
```

Out[49]:
```
15294
```

There are a few empty comments that we need to get rid of, otherwise sklearn will complain.

# Frequent words from dataset

We will plot wordclouds. The detail on how the cloud was created is from
https://www.kaggle.com/arthurtok/spooky-nlp-and-topic-modelling-tutorial
(https://www.kaggle.com/arthurtok/spooky-nlp-and-topic-modelling-tutorial)

In [50]:
```
stopword=set(STOPWORDS)
toxic_mask=np.array(Image.open("./data/safe-zone.png"))

toxic_mask=toxic_mask[:,:,1]
#wordcloud for clean comments
subset=train[train.toxic==1]
text=subset.comment_text.values
wc= WordCloud(background_color="black",max_words=4000,mask=toxic_mask,stopwords=s
wc.generate(" ".join(text))
plt.figure(figsize=(20,20))
plt.subplot(221)
plt.axis("off")
plt.title("Words frequented in negative Comments", fontsize=20)
plt.imshow(wc.recolor(colormap= 'gist_earth' , random_state=244), alpha=0.98)
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-50-47da2de0ab22> in <module>()
      2 toxic_mask=np.array(Image.open("./data/safe-zone.png"))
      3
----> 4 toxic_mask=toxic_mask[:,:,1]
      5 #wordcloud for clean comments
      6 subset=train[train.toxic==1]

IndexError: too many indices for array
```

## Analysis

In [51]:
```
# main_col="toxic"
# corr_mats=[]
# for other_col in temp_df.columns[1:]:
#     confusion_matrix = pd.crosstab(temp_df[main_col], temp_df[other_col])
#     corr_mats.append(confusion_matrix)
# out = pd.concat(corr_mats,axis=1,keys=temp_df.columns[1:])

# #cell highlighting
# #out = out.style.apply(highlight_min,axis=0)
# out
```

In [52]:
```
# COMMENT = 'comment_text'
# #print(train.isnull())
# train[COMMENT].fillna("unknown", inplace=True)
# test[COMMENT].fillna("unknown", inplace=True)
# train_text=train[COMMENT]
# test_text = test[COMMENT]

# all_text = pd.concat([train_text, test_text]) # for feature extraction
```

## Feature Selection/ Feature Engg

We will use multiple simple features and some complex features as shown below

**Simple features**

- count of sentences
- count of words
- count of unique words
- count of letters
- count of punctuations
- count of uppercase words/letters
- count of stop words
- Avg length of each word

**Complex features**

- TFIDF
- Count vectors
- Word2vec

*In the next two cells I am creating a table of simple features for each text,please look carefully*

In [53]:
```
merge=pd.concat([train.iloc[:,0:2]]) #please see what iloc does
df=merge.reset_index(drop=True)
```

In [54]:
```
df['count_sent']=df["comment_text"].apply(lambda x: len(re.findall("\n",str(x)))+
#Word count in each comment:
df['count_word']=df["comment_text"].apply(lambda x: len(str(x).split()))
#Unique word count
df['count_unique_word']=df["comment_text"].apply(lambda x: len(set(str(x).split()
#Letter count
df['count_letters']=df["comment_text"].apply(lambda x: len(str(x)))
#punctuation count
df["count_punctuations"] =df["comment_text"].apply(lambda x: len([c for c in str(
#upper case words count
df["count_words_upper"] = df["comment_text"].apply(lambda x: len([w for w in str(
#title case words count
df["count_words_title"] = df["comment_text"].apply(lambda x: len([w for w in str(
#Number of stopwords
df["count_stopwords"] = df["comment_text"].apply(lambda x: len([w for w in str(x)
#Average length of the words
df["mean_word_len"] = df["comment_text"].apply(lambda x: np.mean([len(w) for w in
```

In [55]:
```
train_feats=df.iloc[0:len(train),]
 # compare this with original train set

train_feats
```

Out[55]:

| | id | comment_text | count_sent | count_word | count_unique |
|---|---|---|---|---|---|
| **0** | 0002bcb3da6cb337 | COCKSUCKER BEFORE YOU PISS AROUND ON MY WORK | 1 | 8 | |
| **1** | 0005c987bdfc9d4b | Hey... what is it..\n@ \| talk .\nWhat is it..... | 5 | 53 | |
| **2** | 0007e25b2121310b | Bye! \n\nDon't look, come or think of comming ... | 3 | 10 | |
| **3** | 001810bf8c45bf5f | You are gay or antisemmitian? \n\nArchangel WH... | 15 | 117 | |
| **4** | 00190820581d90ce | FUCK YOUR FILTHY MOTHER IN THE ASS, DRY! | 1 | 8 | |
| **5** | 001956c382006abd | I'm Sorry \n\nI'm sorry I screwed around with ... | 3 | 56 | |
| **6** | 001dc38a83d420cf | GET FUCKED UP. GET FUCKEEED UP. GOT A DRINK T... | 1 | 25 | |

In [56]:
```
train_tags=train.iloc[:,1:]# getting class labels
train_tags
```

Out[56]:

| | comment_text | toxic | severe_toxic | obscene | threat | insult | identity_hate |
|---|---|---|---|---|---|---|---|
| **6** | COCKSUCKER BEFORE YOU PISS AROUND ON MY WORK | 1 | 1 | 1 | 0 | 1 | 0 |
| **12** | Hey... what is it..\n@ \| talk .\nWhat is it..... | 1 | 0 | 0 | 0 | 0 | 0 |
| **16** | Bye! \n\nDon't look, come or think of comming ... | 1 | 0 | 0 | 0 | 0 | 0 |
| **42** | You are gay or antisemmitian? \n\nArchangel WH... | 1 | 0 | 1 | 0 | 1 | 1 |
| **43** | FUCK YOUR FILTHY MOTHER IN THE ASS, DRY! | 1 | 0 | 1 | 0 | 1 | 0 |
| **44** | I'm Sorry \n\nI'm sorry I screwed around with ... | 1 | 0 | 0 | 0 | 0 | 0 |
| **51** | GET FUCKED UP. GET FUCKEEED UP. GOT A DRINK T... | 1 | 0 | 1 | 0 | 0 | 0 |
| **55** | Stupid peace of shit stop deleting my stuff as... | 1 | 1 | 1 | 0 | 1 | 0 |
| **56** | =Tony Sidaway is obviously a fistfuckee. He lo... | 1 | 0 | 1 | 0 | 1 | 0 |
| **58** | My Band Page's deletion. You thought I was gon... | 1 | 0 | 1 | 0 | 0 | 0 |
| **59** | Why can't you believe how fat Artie is? Did yo... | 1 | 0 | 0 | 0 | 0 | 0 |
| **65** | All of my edits are good. Cunts like you who ... | 1 | 0 | 1 | 0 | 1 | 0 |
| **79** | Hi! I am back again!\nLast warning!\nStop undo... | 1 | 0 | 0 | 1 | 0 | 0 |
| **86** | Would you both shut up, you don't run wikipedi... | 1 | 0 | 0 | 0 | 1 | 0 |
| **105** | A pair of jew-hating weiner nazi schmucks. | 1 | 0 | 1 | 0 | 1 | 1 |
| **151** | "\n\nSORRY PUCK BUT NO ONE EVER SAID DICK WAS ... | 1 | 0 | 0 | 0 | 0 | 0 |
| **159** | "\n\nUNBLOCK ME OR I'LL GET MY LAWYERS ON TO Y... | 1 | 0 | 0 | 0 | 0 | 0 |
| **168** | You should be fired, you're a moronic wimp who... | 1 | 0 | 0 | 0 | 1 | 0 |
| **176** | I think that your a Fagget get a oife and burn... | 1 | 0 | 1 | 1 | 1 | 1 |
| **181** | you are a stupid fuck \n\nand your mother's cu... | 1 | 1 | 1 | 0 | 1 | 0 |
| **201** | Your blatant POV pushing \n\nNeither of you gu... | 1 | 0 | 1 | 0 | 0 | 0 |
| **206** | Give me a permanat block raseac....!!! remembe... | 1 | 0 | 0 | 0 | 0 | 0 |

| | comment_text | toxic | severe_toxic | obscene | threat | insult | identity_hate |
|---|---|---|---|---|---|---|---|
| **211** | Fuck you, block me, you faggot pussy! | 1 | 0 | 1 | 0 | 1 | 0 |
| **218** | Kill all niggers. \n\nI have hard, that others... | 1 | 0 | 1 | 0 | 1 | 1 |
| **231** | Burn Deck \n\nIf that'd guy's burn deck is lik... | 1 | 0 | 1 | 0 | 1 | 0 |
| **238** | u r a tw@ fuck off u gay boy.U r smelly.Fuck u... | 1 | 0 | 1 | 0 | 1 | 1 |
| **268** | Atheism is full of bias shit | 1 | 0 | 0 | 0 | 0 | 0 |
| **278** | Hey why you are spreading misconceptions and t... | 1 | 0 | 0 | 0 | 0 | 0 |
| **286** | "\n\nAnd you are? Let me know when you've craw... | 1 | 0 | 0 | 0 | 0 | 0 |
| **295** | this user is such a worthless goddamn faggot f... | 1 | 0 | 1 | 0 | 1 | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **159253** | what do you mean \n\nwhy don't you keep your n... | 1 | 0 | 1 | 0 | 0 | 0 |
| **159268** | Hi Bading \nPutang ina mong bakla ka. Fuck you... | 1 | 0 | 1 | 0 | 1 | 0 |
| **159274** | "\n\nStudy some linguistics before you say som... | 1 | 0 | 0 | 0 | 0 | 0 |
| **159281** | LoL!! \n\nyou're GAY!! you will never know how... | 1 | 1 | 1 | 0 | 1 | 1 |
| **159290** | Hey alabamoy boy why dont you stick your head ... | 1 | 0 | 1 | 0 | 1 | 0 |
| **159298** | , are you dumber than you look? asshole. | 1 | 0 | 1 | 0 | 1 | 0 |
| **159312** | Walter Mercado \n\nAntonio, quite frankly, you... | 1 | 1 | 1 | 0 | 1 | 0 |
| **159315** | http://www.nysun.com/article/23698 - public in... | 1 | 0 | 0 | 0 | 0 | 0 |
| **159334** | Horse's ass \n\nSeriously, dude, what's that h... | 1 | 0 | 1 | 0 | 0 | 0 |
| **159336** | Oh, fuck off. The pansy Jew would just whine a... | 1 | 0 | 1 | 0 | 1 | 1 |
| **159342** | Fuck off turd. Don't ever ban me again you cun... | 1 | 0 | 1 | 0 | 1 | 0 |
| **159358** | Goethean and me\n\nI would like you to know I ... | 1 | 0 | 0 | 0 | 0 | 0 |
| **159368** | "\n\n Palin/Satan 2012 \n\nWow, what a surpris... | 1 | 0 | 1 | 0 | 1 | 0 |
| **159378** | GO AHEAD AND FUCKING BAN ME ~ LIKE THAT WILL H... | 1 | 0 | 1 | 0 | 1 | 0 |
| **159382** | shut up you goddamn assclown. | 1 | 0 | 1 | 0 | 1 | 0 |

| | comment_text | toxic | severe_toxic | obscene | threat | insult | identity_hate |
|---|---|---|---|---|---|---|---|
| **159386** | Stop telling lies and trying to promote your p... | 1 | 0 | 0 | 0 | 0 | 0 |
| **159394** | your boring \n\nand retarded two | 1 | 0 | 0 | 0 | 0 | 0 |
| **159398** | Why did that idiot revert the reversion I made... | 1 | 0 | 0 | 0 | 0 | 0 |
| **159400** | Shalom \n\nSemite, get the fuck out of here. I... | 1 | 1 | 1 | 1 | 1 | 1 |
| **159411** | Fat piece of shit \n\nyou obese piece of shit.... | 1 | 0 | 1 | 0 | 1 | 0 |
| **159423** | PS: you're all middle-aged losers at home in ... | 1 | 0 | 0 | 0 | 0 | 0 |
| **159448** | Yeah i no it sucks. | 1 | 0 | 0 | 0 | 0 | 0 |
| **159449** | I think he is a gay fag!!! | 1 | 0 | 0 | 0 | 0 | 1 |
| **159478** | "\nThank you. Given the misuse of tools here a... | 1 | 0 | 0 | 0 | 0 | 0 |
| **159493** | FUCKING FAGGOT \n\nLOLWAT. | 1 | 0 | 1 | 0 | 1 | 0 |
| **159494** | "\n\n our previous conversation \n\nyou fuckin... | 1 | 0 | 1 | 0 | 1 | 1 |
| **159514** | YOU ARE A MISCHIEVIOUS PUBIC HAIR | 1 | 0 | 0 | 0 | 1 | 0 |
| **159541** | Your absurd edits \n\nYour absurd edits on gre... | 1 | 0 | 1 | 0 | 1 | 0 |
| **159546** | "\n\nHey listen don't you ever!!!! Delete my e... | 1 | 0 | 0 | 0 | 1 | 0 |
| **159554** | and i'm going to keep posting the stuff u dele... | 1 | 0 | 1 | 0 | 1 | 0 |

15294 rows × 7 columns

In [57]:
```
train_feats=pd.concat([train_feats,train_tags],axis=1)
```

## Cleaning corpus

Clean before generating count vectors

In [58]:
```
APPO = {
"aren't" : "are not",
"can't" : "cannot",
"couldn't" : "could not",
"didn't" : "did not",
"doesn't" : "does not",
"don't" : "do not",
"hadn't" : "had not",
"hasn't" : "has not",
"haven't" : "have not",
"he'd" : "he would",
"he'll" : "he will",
"he's" : "he is",
"i'd" : "I would",
"i'd" : "I had",
"i'll" : "I will",
"i'm" : "I am",
"isn't" : "is not",
"it's" : "it is",
"it'll":"it will",
"i've" : "I have",
"let's" : "let us",
"mightn't" : "might not",
"mustn't" : "must not",
"shan't" : "shall not",
"she'd" : "she would",
"she'll" : "she will",
"she's" : "she is",
"shouldn't" : "should not",
"that's" : "that is",
"there's" : "there is",
"they'd" : "they would",
"they'll" : "they will",
"they're" : "they are",
"they've" : "they have",
"we'd" : "we would",
"we're" : "we are",
"weren't" : "were not",
"we've" : "we have",
"what'll" : "what will",
"what're" : "what are",
"what's" : "what is",
"what've" : "what have",
"where's" : "where is",
"who'd" : "who would",
"who'll" : "who will",
"who're" : "who are",
"who's" : "who is",
"who've" : "who have",
"won't" : "will not",
"wouldn't" : "would not",
"you'd" : "you would",
"you'll" : "you will",
"you're" : "you are",
"you've" : "you have",
"'re": " are",
```

```
"wasn't": "was not",
"we'll":" will",
"didn't": "did not"
}

def clean(comment):
    """
    This function receives comments and returns clean word-list
    """
    #Convert to lower case , so that Hi and hi are the same
    comment=comment.lower()
    #remove \n
    comment=re.sub("\\n","",comment)
    # remove leaky elements like ip,user
    comment=re.sub("\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}","",comment)
    #removing usernames
    comment=re.sub("\[\[.*\]","",comment)

    #Split the sentences into words
    words=tokenizer.tokenize(comment)

    # (')aphostophe  replacement (ie)   you're --> you are
    # ( basic dictionary lookup : master dictionary present in a hidden block of
    words=[APPO[word] if word in APPO else word for word in words]
    words=[lem.lemmatize(word, "v") for word in words]
    words = [w for w in words if not w in eng_stopwords]

    clean_sent=" ".join(words)
    # remove any non alphanum,digit character
    #clean_sent=re.sub("\W+"," ",clean_sent)
    #clean_sent=re.sub("  "," ",clean_sent)
    return(clean_sent)
```

In [59]:
```
corpus=merge.comment_text

corpus.iloc[12235]
```

Out[59]:
```
'Reverting entries/User Zippity Doo Dah \n\nUser Chadbryant is vandalising arti
cles and user pages on Wikipedia with baseless and unsubstantiated sockpuppet a
ccusations.  He has been warned on this by at least one Wikipedia admin, and th
ere is an ongoing dispute between he and I regarding this immature and baseless
behavior; as such, I will NOT allow him to do so, especially on user talk pages
(user pages are bad enough).  Before you go jumping the gun next time, please c
heck to see who the weapon belongs to.  For more information, see the user talk
page of .  And yes, this IS a sockpuppet of , but only because you blocked me w
ithout giving me a chance to explain MY side of the story  an action I have att
empted to do multiple times, only to be stopped by idiots like yourself.'
```

In [62]:
```
clean(corpus.iloc[12235])
clean_corpus=corpus
print(clean_corpus)
```

```
6                COCKSUCKER BEFORE YOU PISS AROUND ON MY WORK
12          Hey... what is it..\n@ | talk .\nWhat is it......
16          Bye! \n\nDon't look, come or think of comming ...
42          You are gay or antisemmitian? \n\nArchangel WH...
43                   FUCK YOUR FILTHY MOTHER IN THE ASS, DRY!
44          I'm Sorry \n\nI'm sorry I screwed around with ...
51          GET FUCKED UP. GET FUCKEEED UP.  GOT A DRINK T...
55          Stupid peace of shit stop deleting my stuff as...
56          =Tony Sidaway is obviously a fistfuckee. He lo...
58          My Band Page's deletion. You thought I was gon...
59          Why can't you believe how fat Artie is? Did yo...
65          All of my edits are good.  Cunts like you who ...
79          Hi! I am back again!\nLast warning!\nStop undo...
86          Would you both shut up, you don't run wikipedi...
105               A pair of jew-hating weiner nazi schmucks.
151         "\n\nSORRY PUCK BUT NO ONE EVER SAID DICK WAS ...
159         "\n\nUNBLOCK ME OR I'LL GET MY LAWYERS ON TO Y...
168         You should be fired, you're a moronic wimp who...
176         I think that your a Fagget get a oife and burn...
181         you are a stupid fuck \n\nand your mother's cu...
201         Your blatant POV pushing \n\nNeither of you gu...
206         Give me a permanat block raseac....!!! remembe...
211                     Fuck you, block me, you faggot pussy!
218         Kill all niggers. \n\nI have hard, that others...
231         Burn Deck \n\nIf that'd guy's burn deck is lik...
238         u r a tw@ fuck off u gay boy.U r smelly.Fuck u...
268                            Atheism is full of bias shit
278         Hey why you are spreading misconceptions and t...
286         "\n\nAnd you are? Let me know when you've craw...
295         this user is such a worthless goddamn faggot f...
                                  ...
159253      what do you mean \n\nwhy don't you keep your n...
159268      Hi Bading \nPutang ina mong bakla ka. Fuck you...
159274      "\n\nStudy some linguistics before you say som...
159281      LoL!! \n\nyou're GAY!! you will never know how...
159290      Hey alabamoy boy why dont you stick your head ...
159298               , are you dumber than you look? asshole.
159312      Walter Mercado \n\nAntonio, quite frankly, you...
159315      http://www.nysun.com/article/23698 (http://www.nysun.com/article/2369
8) - public in...
159334      Horse's ass \n\nSeriously, dude, what's that h...
159336      Oh, fuck off. The pansy Jew would just whine a...
159342      Fuck off turd. Don't ever ban me again you cun...
159358      Goethean and me\n\nI would like you to know I ...
159368      "\n\n Palin/Satan 2012 \n\nWow, what a surpris...
159378      GO AHEAD AND FUCKING BAN ME ~ LIKE THAT WILL H...
159382                          shut up you goddamn assclown.
159386      Stop telling lies and trying to promote your p...
159394                        your boring \n\nand retarded two
159398      Why did that idiot revert the reversion I made...
159400      Shalom \n\nSemite, get the fuck out of here. I...
159411      Fat piece of shit \n\nyou obese piece of shit....
```

```
159423    PS:  you're all middle-aged losers at home in ...
159448                                 Yeah i no it sucks.
159449                           I think he is a gay fag!!!
159478    "\nThank you. Given the misuse of tools here a...
159493                           FUCKING FAGGOT \n\nLOLWAT.
159494    "\n\n our previous conversation \n\nyou fuckin...
159514                   YOU ARE A MISCHIEVIOUS PUBIC HAIR
159541    Your absurd edits \n\nYour absurd edits on gre...
159546    "\n\nHey listen don't you ever!!!! Delete my e...
159554    and i'm going to keep posting the stuff u dele...
Name: comment_text, Length: 15294, dtype: object
```

In [ ]:

## Complex features

Lets create some features based on frequency distribution of the words. Initially lets consider taking words one at a time (ie) Unigrams

Python's SKlearn provides 3 ways of creating count features.All three of them first create a vocabulary(dictionary) of words and then create a sparse matrix of word counts for the words in the sentence that are present in the dictionary. A brief description of them:

- CountVectorizer
  - Creates a matrix with frequency counts of each word in the text corpus
- TF-IDF Vectorizer
  - TF - Term Frequency -- Count of the words(Terms) in the text corpus (same of Count Vect)
  - IDF - Inverse Document Frequency -- Penalizes words that are too frequent. We can think of this as regularization
- HashingVectorizer
  - Creates a hashmap(word to number mapping based on hashing technique) instead of a dictionary for vocabulary
  - This enables it to be more scalable and faster for larger text coprus
  - Can be parallelized across multiple threads

Using TF-IDF here. Note: Using the concatenated dataframe "merge" which contains both text from train and test dataset to ensure that the vocabulary that we create does not missout on the words that are unique to testset.

In [162]:

```
def top_tfidf_feats(row, features, top_n=25):
    ''' Get top n tfidf values in row and return them with their corresponding fe
    topn_ids = np.argsort(row)[::-1][:top_n]
    top_feats = [(features[i], row[i]) for i in topn_ids]
    df = pd.DataFrame(top_feats)
    df.columns = ['feature', 'tfidf']
    return df

def top_feats_in_doc(Xtr, features, row_id, top_n=25):
    ''' Top tfidf features in specific document (matrix row) '''
    row = np.squeeze(Xtr[row_id].toarray())
    return top_tfidf_feats(row, features, top_n)

def top_mean_feats(Xtr, features, grp_ids, min_tfidf=0.1, top_n=25):
    ''' Return the top n features that on average are most important amongst docu
        indentified by indices in grp_ids. '''

    D = Xtr[grp_ids].toarray()

    D[D < min_tfidf] = 0
    tfidf_means = np.mean(D, axis=0)
    return top_tfidf_feats(tfidf_means, features, top_n)

# modified for multilabel milticlass
def top_feats_by_class(Xtr, features, min_tfidf=0.1, top_n=20):
    ''' Return a list of dfs, where each df holds top_n features and their mean t
        calculated across documents with the same class label. '''
    dfs = []
    cols=train_tags.columns
    for col in cols:
        ids = train_tags.index[train_tags[col]==1]
        feats_df = top_mean_feats(Xtr, features, ids, min_tfidf=min_tfidf, top_n=
        feats_df.label = labels
        dfs.append(feats_df)
    return dfs
```

In [156]:
```
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer, Has
from sklearn.decomposition import TruncatedSVD
from sklearn.base import BaseEstimator, ClassifierMixin
from sklearn.utils.validation import check_X_y, check_is_fitted
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
from sklearn.metrics import log_loss
from sklearn.model_selection import StratifiedKFold
from sklearn.model_selection import train_test_split


### Unigrams -- TF-IDF
# using settings recommended here for TF-IDF -- https://www.kaggle.com/abhishek/a

start_unigrams=time.time()
tfv = TfidfVectorizer(min_df=200,  max_features=10000,
            strip_accents='unicode', analyzer='word',ngram_range=(1,1),
            use_idf=1,smooth_idf=1,sublinear_tf=1,
            stop_words = 'english')
tfv.fit(clean_corpus)
features = np.array(tfv.get_feature_names())

train_unigrams =  tfv.transform(clean_corpus.iloc[:train.shape[0]])
```

In [157]:
```
# This will take time
start_time=time.time()
tfv = TfidfVectorizer(min_df=100,  max_features=30000,
            strip_accents='unicode', analyzer='char',ngram_range=(1,4),
            use_idf=1,smooth_idf=1,sublinear_tf=1,
            stop_words = 'english')

tfv.fit(clean_corpus)
features = np.array(tfv.get_feature_names())
train_charngrams =  tfv.transform(clean_corpus.iloc[:train.shape[0]])
end_time=time.time()
print("total time till charngrams",end_time-start_time)
```

```
('total time till charngrams', 16.94825506210327)
```

In [158]:
```
tfv = TfidfVectorizer(min_df=150,  max_features=30000,
            strip_accents='unicode', analyzer='word',ngram_range=(2,2),
            use_idf=1,smooth_idf=1,sublinear_tf=1,
            stop_words = 'english')

tfv.fit(clean_corpus)
features = np.array(tfv.get_feature_names())
train_bigrams =  tfv.transform(clean_corpus.iloc[:train.shape[0]])
#get top n for bigrams
#tfidf_top_n_per_lass=top_feats_by_class(train_bigrams,features)
```

In [163]:
```
tfidf_top_n_per_lass=top_feats_by_class(train_bigrams,features)
plt.figure(figsize=(16,22))
plt.suptitle("TF_IDF Top words per class(unigrams)",fontsize=20)
gridspec.GridSpec(4,2)
plt.subplot2grid((4,2),(0,0))
sns.barplot(tfidf_top_n_per_lass[0].feature.iloc[0:9],tfidf_top_n_per_lass[0].tfi
plt.title("class : Toxic",fontsize=15)
plt.xlabel('Word', fontsize=12)
plt.ylabel('TF-IDF score', fontsize=12)

plt.subplot2grid((4,2),(0,1))
sns.barplot(tfidf_top_n_per_lass[1].feature.iloc[0:9],tfidf_top_n_per_lass[1].tfi
plt.title("class : Severe toxic",fontsize=15)
plt.xlabel('Word', fontsize=12)
plt.ylabel('TF-IDF score', fontsize=12)


plt.subplot2grid((4,2),(1,0))
sns.barplot(tfidf_top_n_per_lass[2].feature.iloc[0:9],tfidf_top_n_per_lass[2].tfi
plt.title("class : Obscene",fontsize=15)
plt.xlabel('Word', fontsize=12)
plt.ylabel('TF-IDF score', fontsize=12)


plt.subplot2grid((4,2),(1,1))
sns.barplot(tfidf_top_n_per_lass[3].feature.iloc[0:9],tfidf_top_n_per_lass[3].tfi
plt.title("class : Threat",fontsize=15)
plt.xlabel('Word', fontsize=12)
plt.ylabel('TF-IDF score', fontsize=12)


plt.subplot2grid((4,2),(2,0))
sns.barplot(tfidf_top_n_per_lass[4].feature.iloc[0:9],tfidf_top_n_per_lass[4].tfi
plt.title("class : Insult",fontsize=15)
plt.xlabel('Word', fontsize=12)
plt.ylabel('TF-IDF score', fontsize=12)


plt.subplot2grid((4,2),(2,1))
sns.barplot(tfidf_top_n_per_lass[5].feature.iloc[0:9],tfidf_top_n_per_lass[5].tfi
plt.title("class : Identity hate",fontsize=15)
plt.xlabel('Word', fontsize=12)
plt.ylabel('TF-IDF score', fontsize=12)


plt.subplot2grid((4,2),(3,0),colspan=2)
sns.barplot(tfidf_top_n_per_lass[6].feature.iloc[0:19],tfidf_top_n_per_lass[6].tf
plt.title("class : Clean",fontsize=15)
plt.xlabel('Word', fontsize=12)
plt.ylabel('TF-IDF score', fontsize=12)

plt.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-163-55ad03bcb1f8> in <module>()
```

```
----> 1 tfidf_top_n_per_lass=top_feats_by_class(train_bigrams,features)
      2 plt.figure(figsize=(16,22))
      3 plt.suptitle("TF_IDF Top words per class(unigrams)",fontsize=20)
      4 gridspec.GridSpec(4,2)
      5 plt.subplot2grid((4,2),(0,0))

<ipython-input-162-e7c0595b2c2f> in top_feats_by_class(Xtr, features, min_tfid
f, top_n)
     31         ids = train_tags.index[train_tags[col]==1]
     32         feats_df = top_mean_feats(Xtr, features, ids, min_tfidf=min_tfi
df, top_n=top_n)
---> 33         feats_df.label = labels
     34         dfs.append(feats_df)
     35     return dfs

NameError: global name 'labels' is not defined
```

In [66]:
```
# #save tdidf to avoid recomputation
# import cPickle as pickle
# # with open("unigram.pkl", 'wb') as handle:
# #                     pickle.dump(train_unigrams, handle)

# # with open("charngram.pkl", 'wb') as handle:
# #                     pickle.dump(train_charngrams, handle)

# # with open("bigram.pkl", 'wb') as handle:
# #                     pickle.dump(train_bigrams, handle)

# #load the content

# train_unigrams = pickle.load(open("unigram.pkl", "rb" ))
# train_charngrams = pickle.load(open("charngram.pkl", "rb" ) )
# train_bigrams = pickle.load(open("bigram.pkl", "rb" ) )
```

## Word Vector Features

In [122]:

```
with open("./glove.6B.50d.txt", "rb") as lines:
    w2v = {line.split()[0]: np.array(map(float, line.split()[1:]))
           for line in lines}
import gensim
# # # let X be a list of tokenized texts (i.e. list of lists of tokens)
# from gensim.models import word2vec
# tokentext=[i.split() for i in clean_corpus]
# model = word2vec.Word2Vec(tokentext, size=100, window=5, min_count=5, workers=4,
# w2v = dict(zip(model.wv.index2word, model.wv.syn0))


# start_time=time.time()
# clean_corpus=corpus.apply(lambda x :clean(x))

# end_time=time.time()
# print("total time till Cleaning",end_time-start_time)


class MeanEmbeddingVectorizer(object):
    def __init__(self, word2vec):
        self.word2vec = word2vec
        # if a text is empty we should return a vector of zeros
        # with the same dimensionality as all the other vectors
        self.dim = len(word2vec.itervalues().next())

    def fit(self, X):
        return self

    def transform(self, X):
        return np.array([
            np.mean([self.word2vec[w] for w in words if w in self.word2vec]
                    or [np.zeros(self.dim)], axis=0)
            for words in X
        ])

class TfidfEmbeddingVectorizer(object):
    def __init__(self, word2vec):
        self.word2vec = word2vec
        self.word2weight = None
        self.dim = len(word2vec.itervalues().next())

    def fit(self, X):
        tfidf = TfidfVectorizer(analyzer=lambda x: x)
        tfidf.fit(X)
        # if a word was never seen - it must be at least as infrequent
        # as any of the known words - so the default idf is the max of
        # known idf's
        max_idf = max(tfidf.idf_)
        self.word2weight = defaultdict(
            lambda: max_idf,
            [(w, tfidf.idf_[i]) for w, i in tfidf.vocabulary_.items()])

        return self

    def transform(self, X):
        return np.array([
```

```
                np.mean([self.word2vec[w] * self.word2weight[w]
                         for w in words if w in self.word2vec] or
                        [np.zeros(self.dim)], axis=0)
                for words in X
            ])
```

In [123]:
```
from collections import defaultdict
wordvect = TfidfEmbeddingVectorizer(w2v)
wordvect.fit(clean_corpus)
wordvectors =  wordvect.transform(clean_corpus.iloc[:train.shape[0]])
```

In [ ]:
```

```

## Model Building

In [124]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report

SELECTED_COLS=['count_sent', 'count_word', 'count_unique_word',
       'count_letters', 'count_punctuations', 'count_words_upper',
       'count_words_title', 'count_stopwords']
target_x=train_feats[SELECTED_COLS]
# target_x

TARGET_COLS=['severe_toxic', 'obscene', 'threat', 'insult', 'identity_hate']
target_y=train_tags[TARGET_COLS]
```

In [125]:
```
from scipy.sparse import csr_matrix, hstack
from imblearn.over_sampling import SMOTE
#Using all direct features
print("Using all features")
#target_x = hstack((train_bigrams,train_charngrams,train_unigrams,train_feats[SEL
target_x = hstack((train_bigrams,train_unigrams)).tocsr()
#target_x= wordvectors
#target_x = hstack((wordvectors)).tocsr()
end_time=time.time()
print("total time till Sparse mat creation",end_time-start_time)
```

```
Using all features
('total time till Sparse mat creation', 40.43341016769409)
```

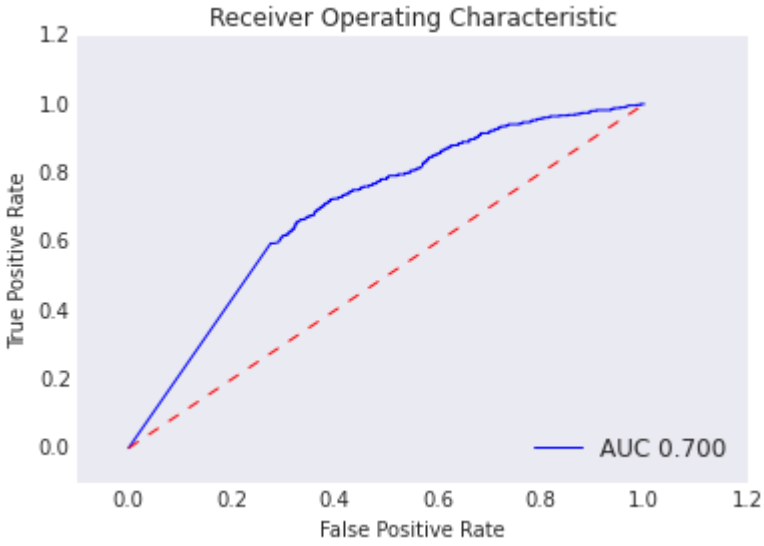
In [136]:
```
from sklearn.model_selection import cross_val_score
start_time=time.time()
from sklearn.metrics import roc_curve, auc
from sklearn.metrics import roc_auc_score
from sklearn.naive_bayes import GaussianNB
from imblearn.under_sampling import CondensedNearestNeighbour
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.gaussian_process import GaussianProcessClassifier
from sklearn.gaussian_process.kernels import RBF
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import roc_curve, auc

X_train, X_valid, y_train, y_valid = train_test_split(target_x, target_y, test_si
```

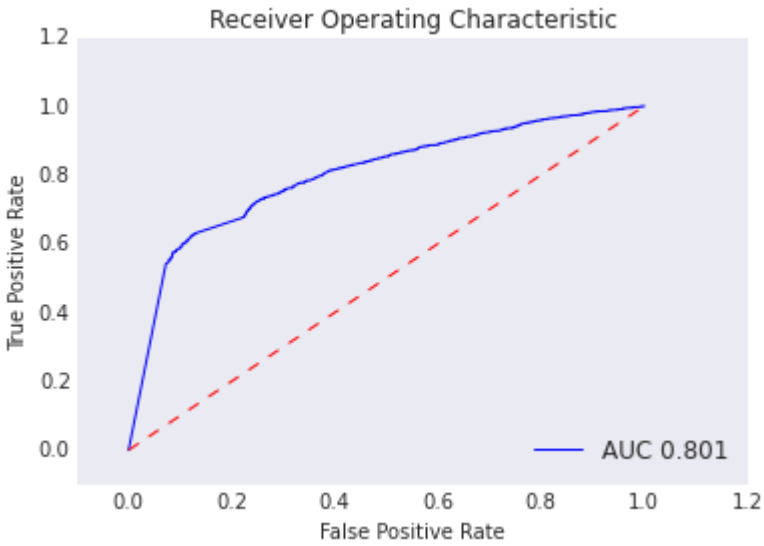
## Gaussian Naive Bayes

In [149]:
```
model=GaussianNB()
for i, j in enumerate(TARGET_COLS):
    print(cross_val_score(model,X_train.toarray(),y_train[j],cv=5,scoring="roc_au
    model.fit(X_train.toarray(),y_train[j])
    bb = model.predict_proba(X_valid.toarray())[:, 1]
    false_positive_rate, true_positive_rate, thresholds = roc_curve(y_valid[j], b
    score=(roc_auc_score(y_valid[j],bb))
    roc_curve(y_valid[j].values, bb)
    plt.title('Receiver Operating Characteristic')
    plt.plot(false_positive_rate, true_positive_rate, 'b',label='AUC %0.3f' % sco
    plt.legend(loc='lower right')
    plt.plot([0,1],[0,1],'r--')
    plt.xlim([-0.1,1.2])
    plt.ylim([-0.1,1.2])
    plt.ylabel('True Positive Rate')
    plt.xlabel('False Positive Rate')
    plt.show()
```

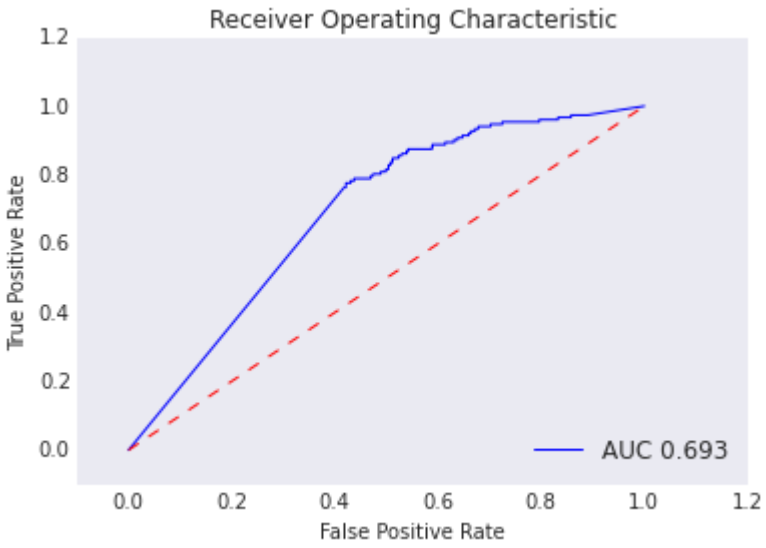
```
[ 0.68076449  0.70531312  0.69939931  0.68779556  0.73912951]
```




```
[ 0.819002    0.80227637  0.80469397  0.80605229  0.83082716]
```

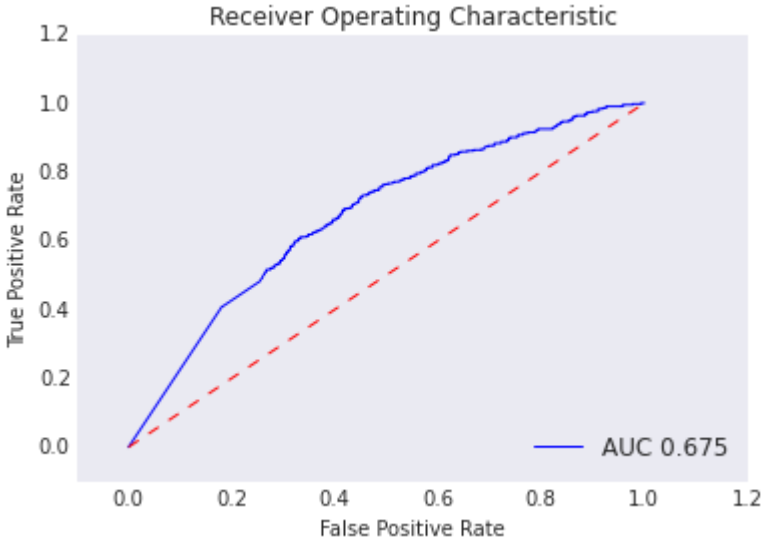


```
[ 0.62832915  0.6163191   0.56677455  0.68348096  0.65030975]
```




```
[ 0.70171356  0.71577508  0.70985936  0.70565723  0.71635158]
```

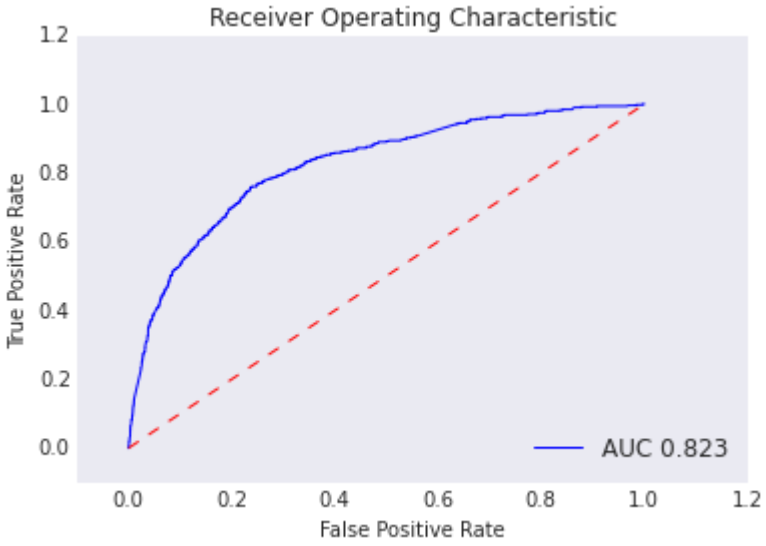

```
[ 0.6790979   0.67218005  0.67326401  0.65737844  0.6475177 ]
```



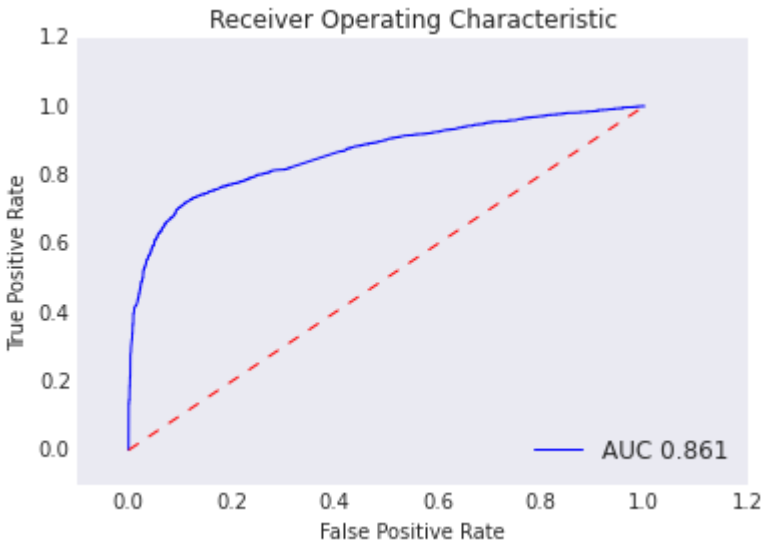
# Logistic Regression

In [150]:
```
model=LogisticRegression()
for i, j in enumerate(TARGET_COLS):
    print(cross_val_score(model,X_train.toarray(),y_train[j],cv=5,scoring="roc_au
    model.fit(X_train.toarray(),y_train[j])
    bb = model.predict_proba(X_valid.toarray())[:, 1]
    false_positive_rate, true_positive_rate, thresholds = roc_curve(y_valid[j], b
    score=(roc_auc_score(y_valid[j],bb))
    roc_curve(y_valid[j].values, bb)
    plt.title('Receiver Operating Characteristic')
    plt.plot(false_positive_rate, true_positive_rate, 'b',label='AUC %0.3f' % sco
    plt.legend(loc='lower right')
    plt.plot([0,1],[0,1],'r--')
    plt.xlim([-0.1,1.2])
    plt.ylim([-0.1,1.2])
    plt.ylabel('True Positive Rate')
    plt.xlabel('False Positive Rate')
    plt.show()
```

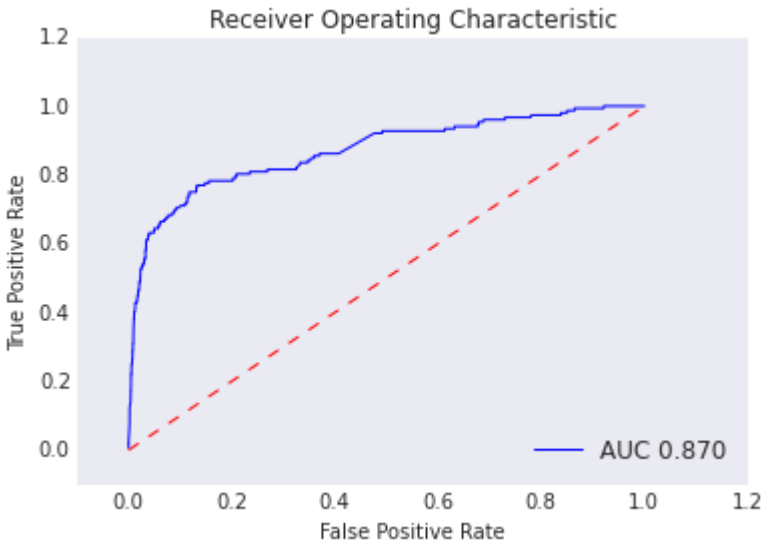
```
[ 0.8352868   0.82484077  0.81508965  0.81539905  0.83707212]
```



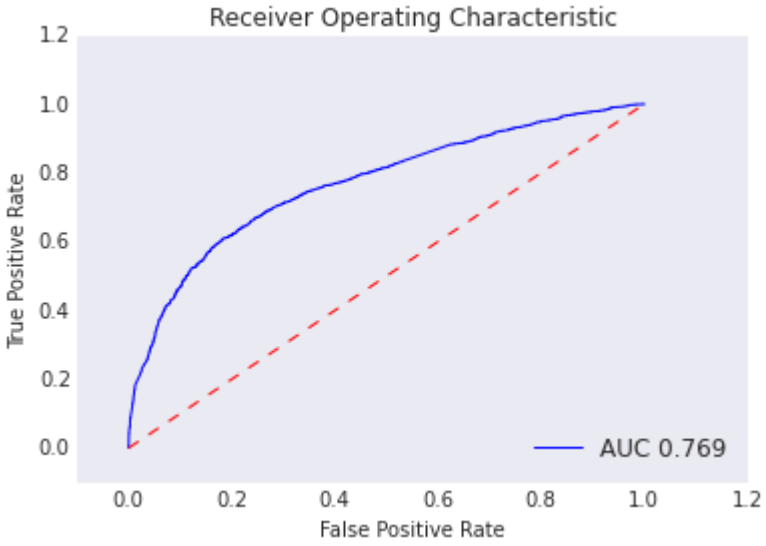

```
[ 0.86190683  0.85799876  0.86411962  0.8769487   0.87671408]
```



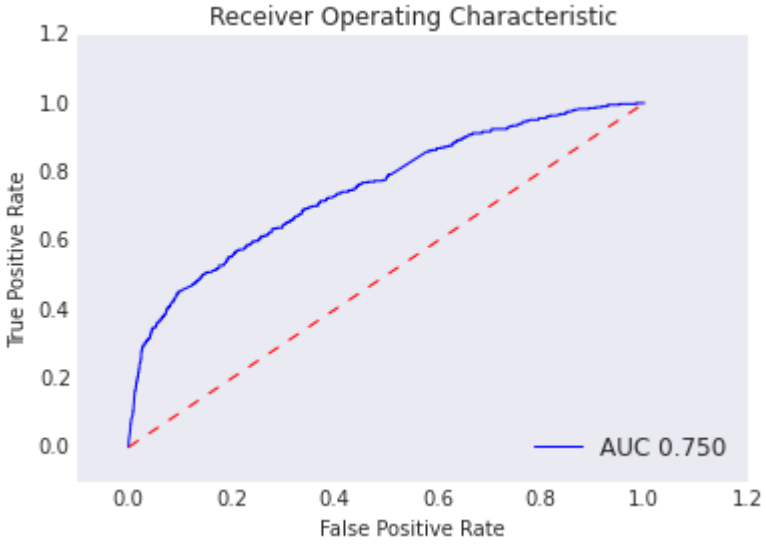
```
[ 0.89278894  0.83067839  0.8264245   0.85081765  0.86557848]
```




```
[ 0.77695348  0.77527109  0.77818404  0.76537012  0.77031197]
```



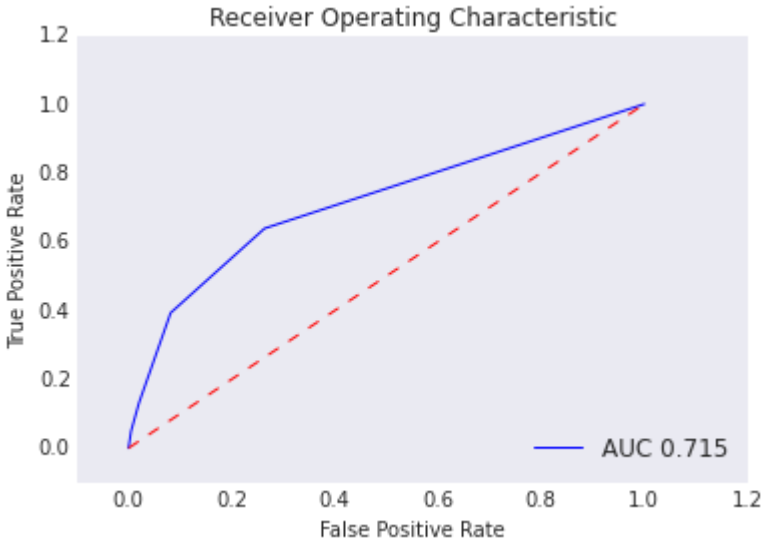

```
[ 0.75769388  0.72255118  0.73681682  0.74432647  0.78407869]
```



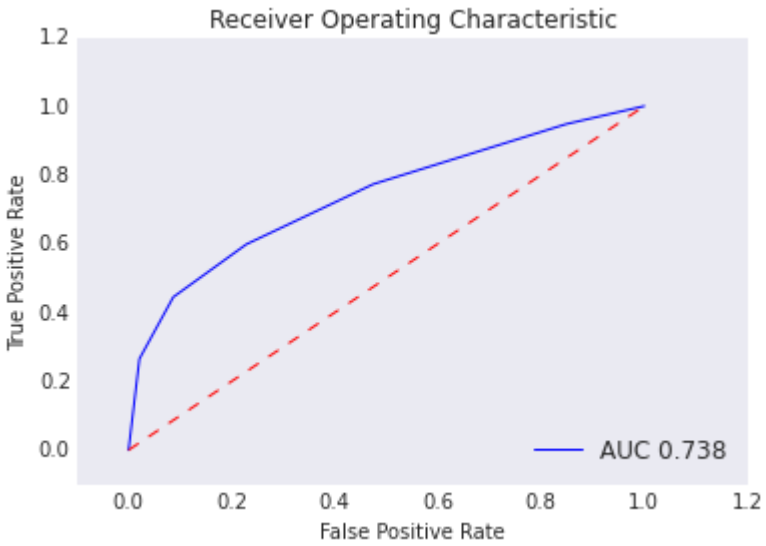
# KNN CLassifier

In [144]:
```
model=KNeighborsClassifier()
for i, j in enumerate(TARGET_COLS):
    print(cross_val_score(model,X_train.toarray(),y_train[j],cv=5,scoring="roc_au
    model.fit(X_train.toarray(),y_train[j])
    bb = model.predict_proba(X_valid.toarray())[:, 1]
    false_positive_rate, true_positive_rate, thresholds = roc_curve(y_valid[j], b
    score=(roc_auc_score(y_valid[j],bb))
    roc_curve(y_valid[j].values, bb)
    plt.title('Receiver Operating Characteristic')
    plt.plot(false_positive_rate, true_positive_rate, 'b',label='AUC %0.3f' % sco
    plt.legend(loc='lower right')
    plt.plot([0,1],[0,1],'r--')
    plt.xlim([-0.1,1.2])
    plt.ylim([-0.1,1.2])
    plt.ylabel('True Positive Rate')
    plt.xlabel('False Positive Rate')
    plt.show()
```

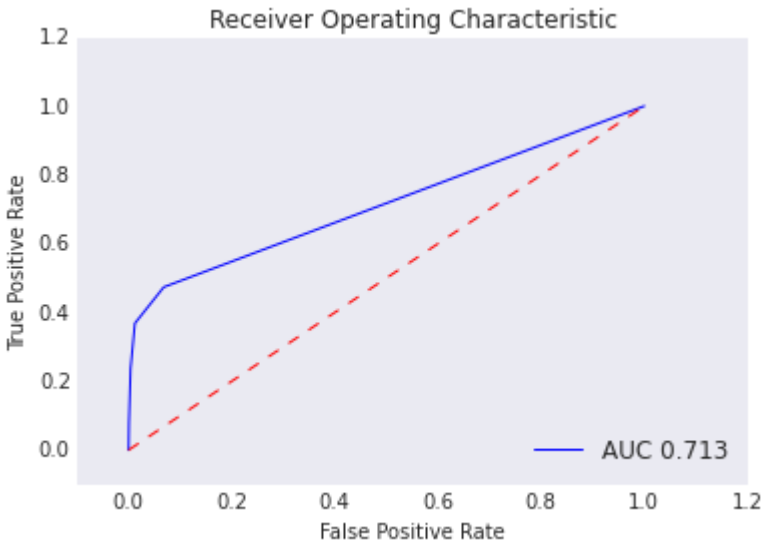
```
[ 0.88097561  0.8999512   0.88433382  0.88140556  0.88189361]
```



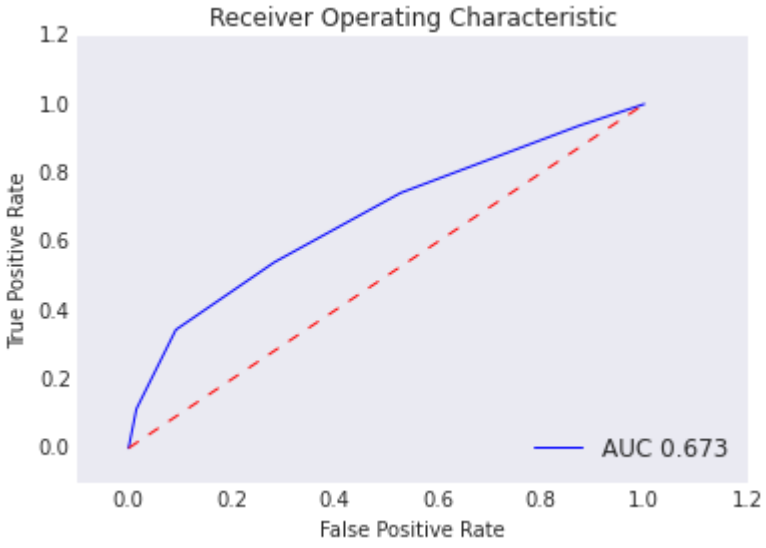

```
[ 0.66390244  0.66097561  0.63396779  0.67447535  0.69140625]
```



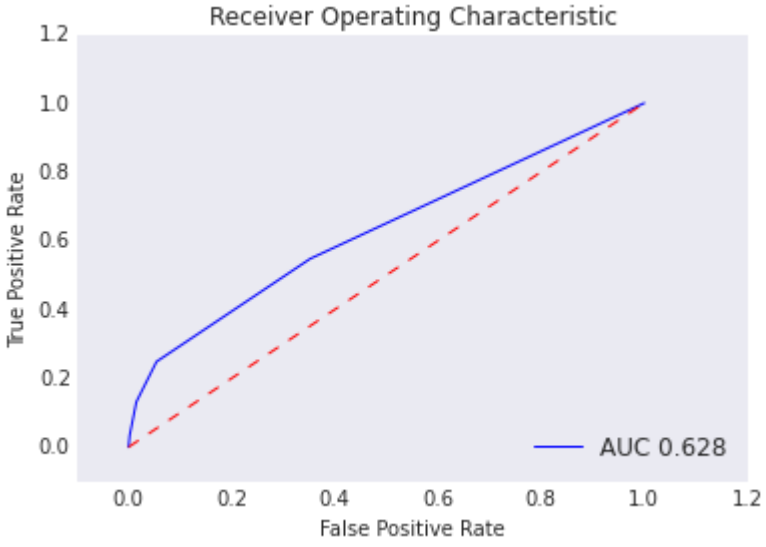
```
[ 0.97073171  0.97414634  0.97120547  0.97169351  0.97167969]
```




```
[ 0.64634146  0.6297561   0.6193265   0.64714495  0.63574219]
```



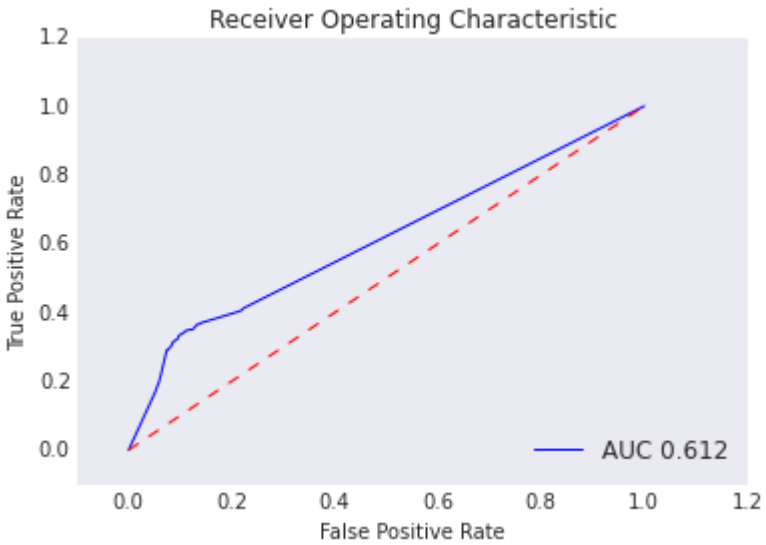

```
[ 0.90634146  0.90922401  0.90873597  0.91166423  0.90873597]
```



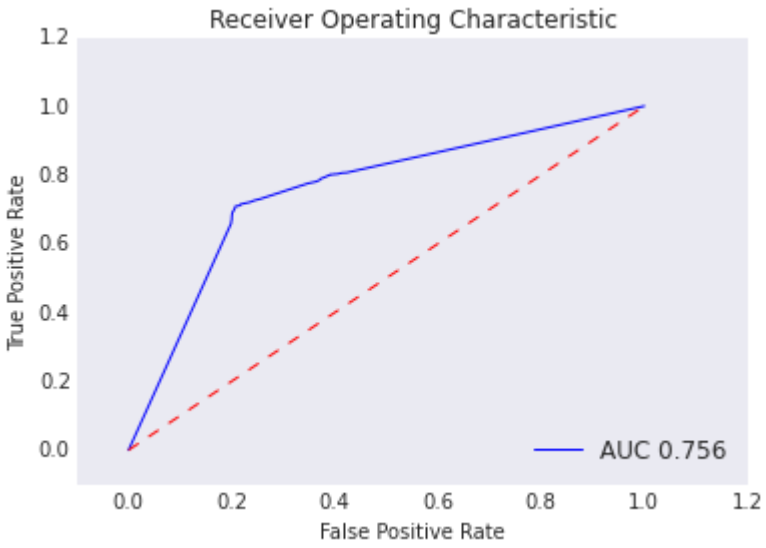
# Decision Tree Classifier

In [151]:
```
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
model=DecisionTreeClassifier()
for i, j in enumerate(TARGET_COLS):
    print(cross_val_score(model,X_train.toarray(),y_train[j],cv=5,scoring="roc_au
    model.fit(X_train.toarray(),y_train[j])
    bb = model.predict_proba(X_valid.toarray())[:, 1]
    false_positive_rate, true_positive_rate, thresholds = roc_curve(y_valid[j], b
    score=(roc_auc_score(y_valid[j],bb))
    roc_curve(y_valid[j].values, bb)
    plt.title('Receiver Operating Characteristic')
    plt.plot(false_positive_rate, true_positive_rate, 'b',label='AUC %0.3f' % sco
    plt.legend(loc='lower right')
    plt.plot([0,1],[0,1],'r--')
    plt.xlim([-0.1,1.2])
    plt.ylim([-0.1,1.2])
    plt.ylabel('True Positive Rate')
    plt.xlabel('False Positive Rate')
    plt.show()
```

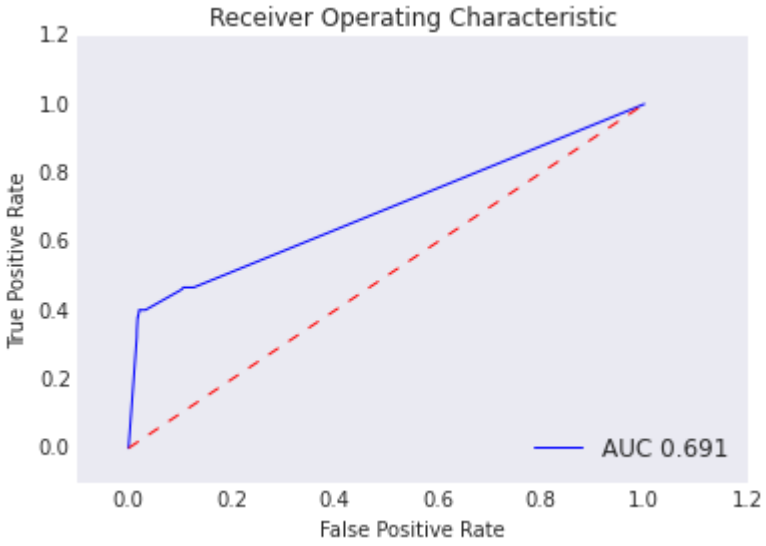
```
[ 0.62934052  0.62452584  0.62179532  0.62051999  0.60386901]
```



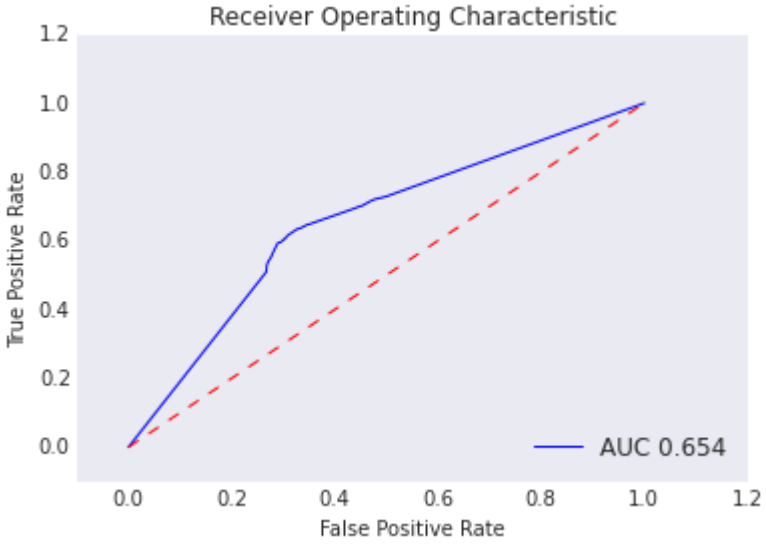

```
[ 0.74049065  0.76480259  0.76440169  0.75719297  0.76731569]
```



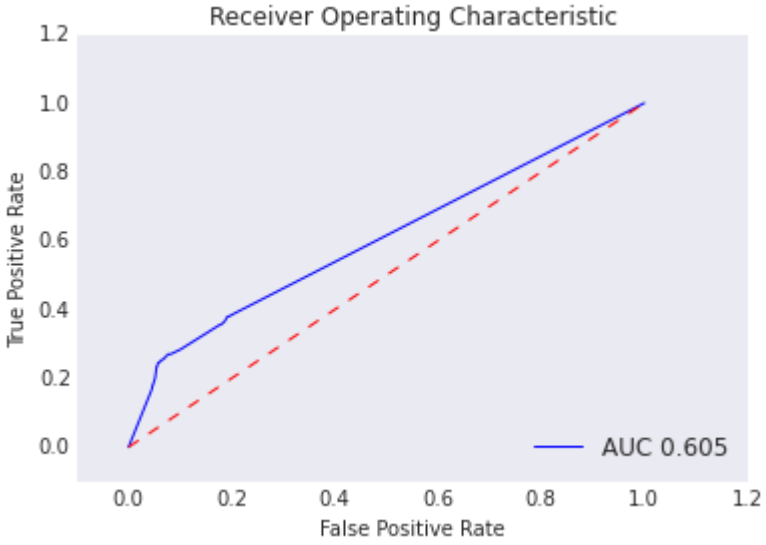
```
[ 0.66195561  0.68365997  0.6462269   0.63699855  0.60781331]
```




```
[ 0.66565565  0.65138823  0.66498321  0.66197521  0.66292458]
```



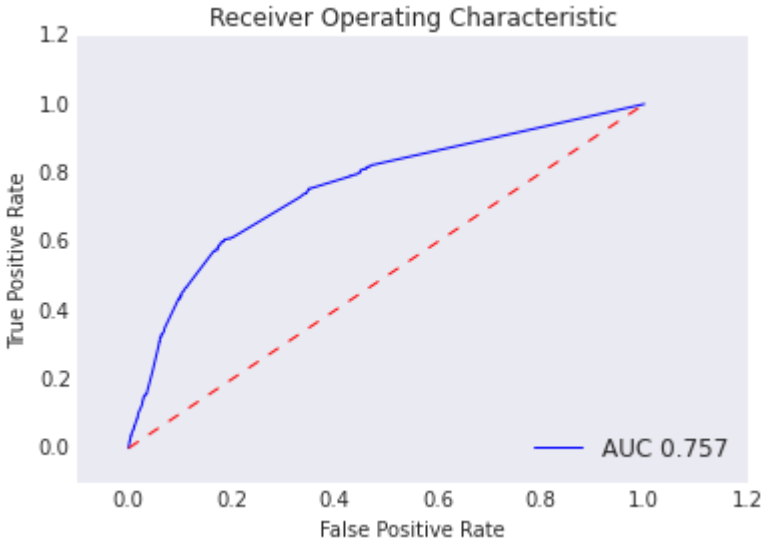

```
[ 0.61987137  0.60923378  0.60887947  0.59794228  0.60390706]
```



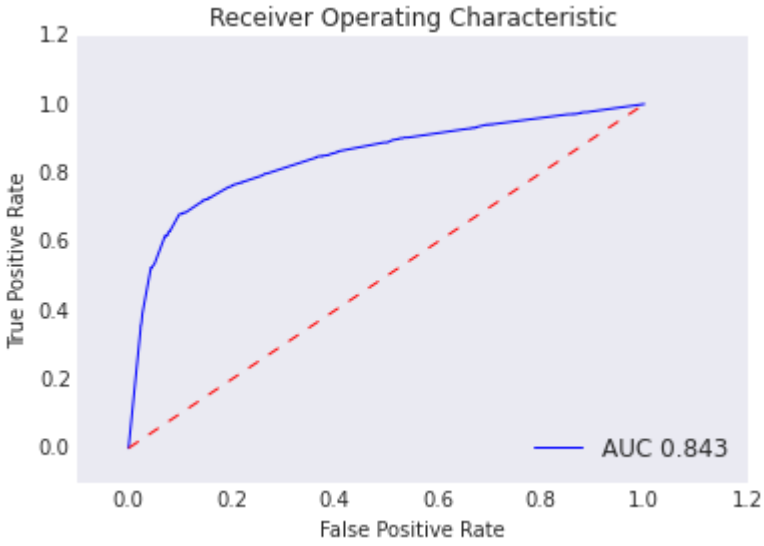
# Random Forest Classifier

In [152]:
```
model=RandomForestClassifier()
for i, j in enumerate(TARGET_COLS):
    print(cross_val_score(model,X_train.toarray(),y_train[j],cv=5,scoring="roc_au
    model.fit(X_train.toarray(),y_train[j])
    bb = model.predict_proba(X_valid.toarray())[:, 1]
    false_positive_rate, true_positive_rate, thresholds = roc_curve(y_valid[j], b
    score=(roc_auc_score(y_valid[j],bb))
    roc_curve(y_valid[j].values, bb)
    plt.title('Receiver Operating Characteristic')
    plt.plot(false_positive_rate, true_positive_rate, 'b',label='AUC %0.3f' % sco
    plt.legend(loc='lower right')
    plt.plot([0,1],[0,1],'r--')
    plt.xlim([-0.1,1.2])
    plt.ylim([-0.1,1.2])
    plt.ylabel('True Positive Rate')
    plt.xlabel('False Positive Rate')
    plt.show()
```

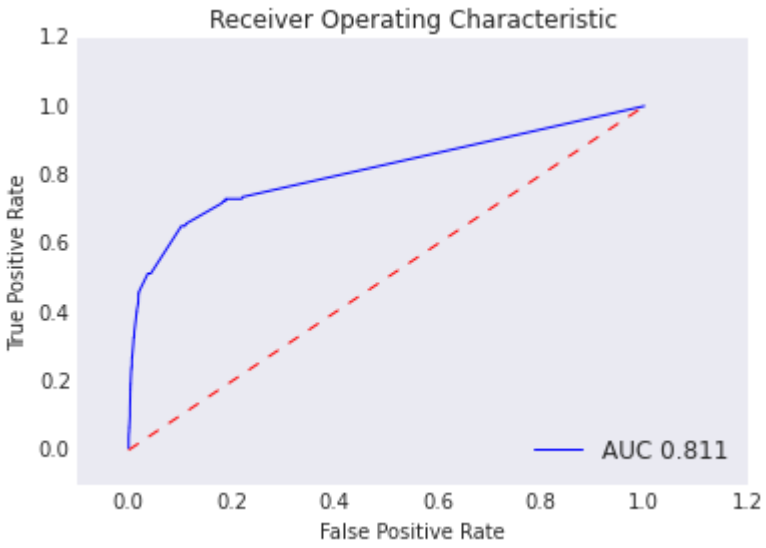
```
[ 0.73978657  0.74388118  0.73809063  0.76400977  0.74355291]
```



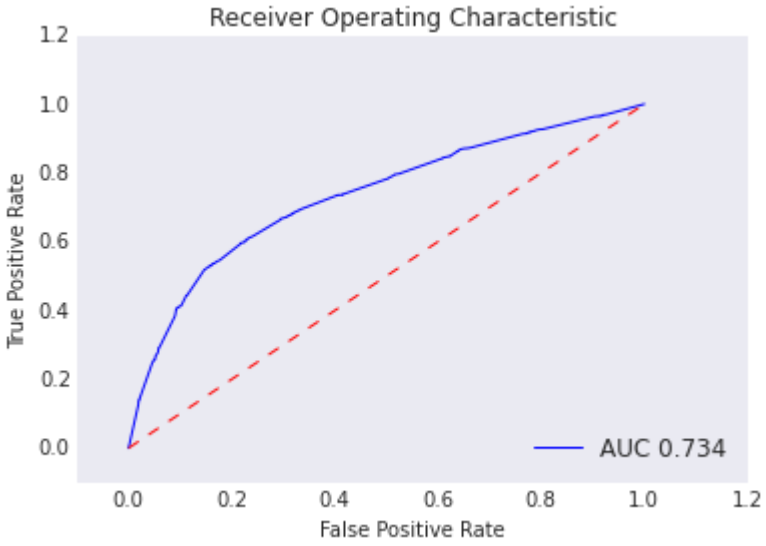

```
[ 0.84955035  0.84251669  0.84279176  0.86167456  0.85942252]
```



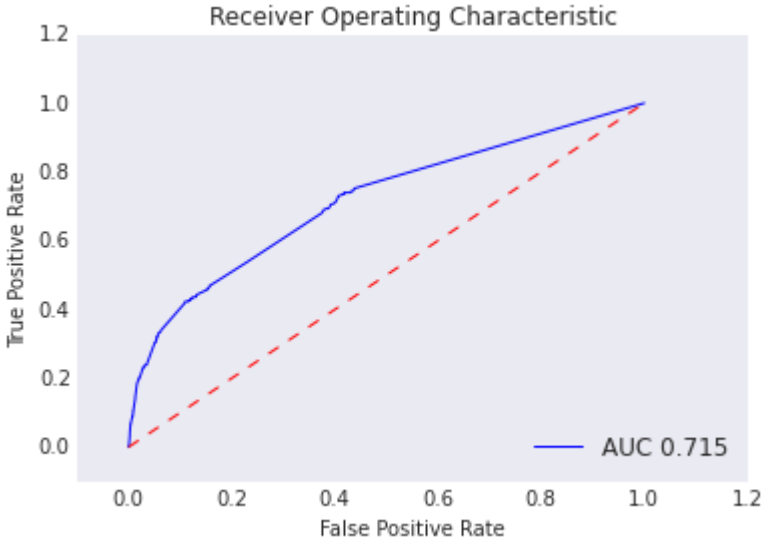
```
[ 0.81757538  0.80905779  0.78661528  0.73865514  0.76167651]
```




```
[ 0.74895339  0.73567968  0.74109671  0.73887876  0.74931737]
```



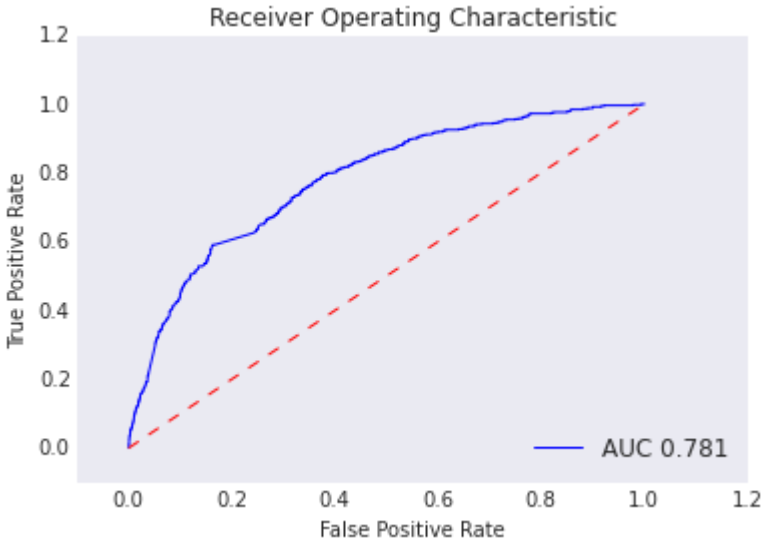

```
[ 0.73242582  0.70230424  0.65118545  0.67623064  0.71594082]
```



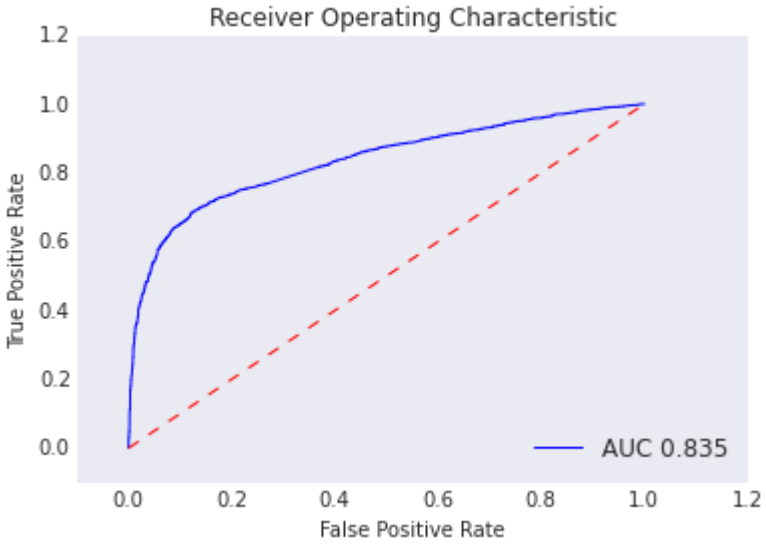
# Multilayer perceptron

In [153]:
```
model=MLPClassifier()
for i, j in enumerate(TARGET_COLS):
    print(cross_val_score(model,X_train.toarray(),y_train[j],cv=5))
    model.fit(X_train.toarray(),y_train[j])
    bb = model.predict_proba(X_valid.toarray())[:, 1]
    false_positive_rate, true_positive_rate, thresholds = roc_curve(y_valid[j], b
    score=(roc_auc_score(y_valid[j],bb))
    roc_curve(y_valid[j].values, bb)
    plt.title('Receiver Operating Characteristic')
    plt.plot(false_positive_rate, true_positive_rate, 'b',label='AUC %0.3f' % sco
    plt.legend(loc='lower right')
    plt.plot([0,1],[0,1],'r--')
    plt.xlim([-0.1,1.2])
    plt.ylim([-0.1,1.2])
    plt.ylabel('True Positive Rate')
    plt.xlabel('False Positive Rate')
    plt.show()
```

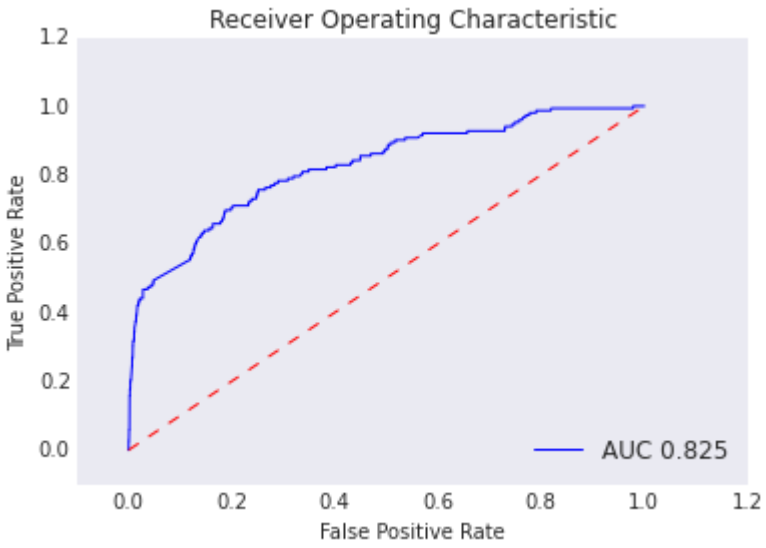
```
[ 0.88731707  0.88677404  0.88042948  0.88433382  0.88677404]
```



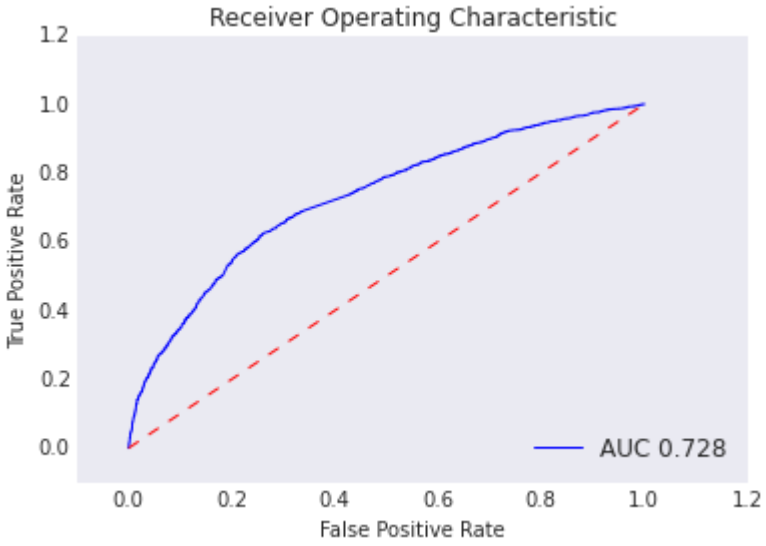

```
[ 0.77707317  0.77414634  0.7750122   0.80917521  0.79052734]
```



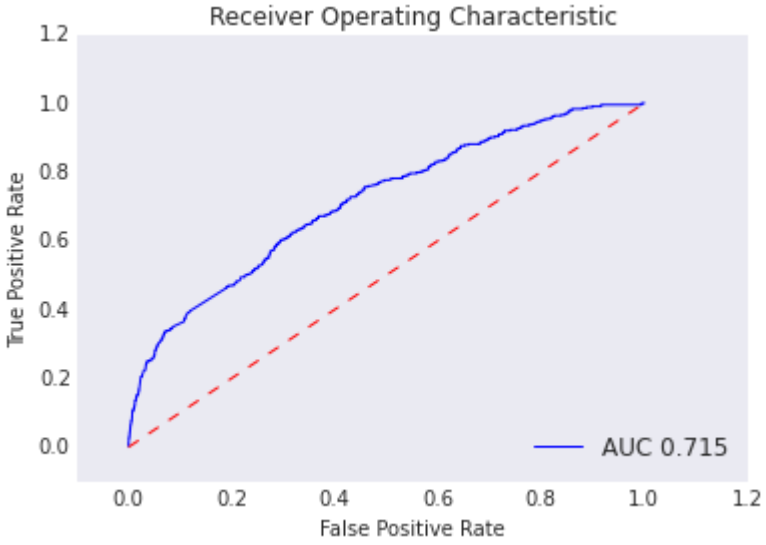
```
[ 0.97463415  0.97463415  0.96827721  0.97266959  0.97460938]
```




```
[ 0.68634146  0.67902439  0.68911664  0.68570034  0.67919922]
```



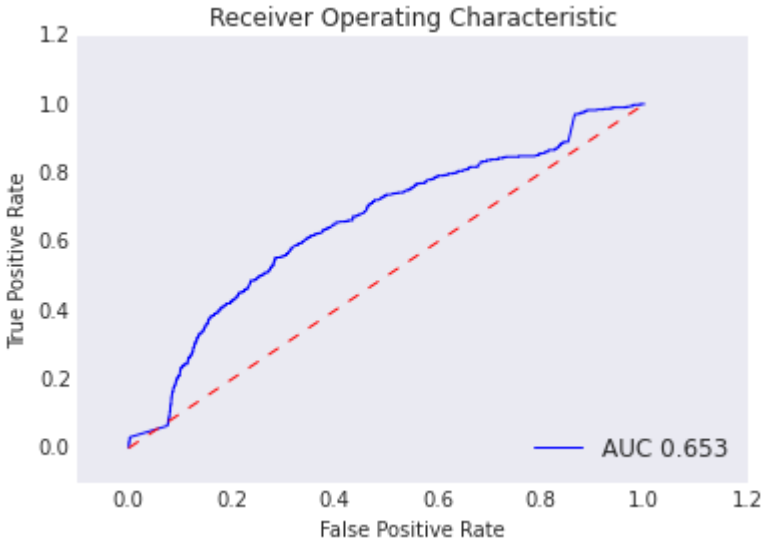

```
[ 0.90390244  0.90775988  0.91068814  0.91117618  0.9136164 ]
```



# SVM Classifier

In [148]:
```
model=SVC(probability=True)
for i, j in enumerate(TARGET_COLS):
    print(cross_val_score(model,X_train.toarray(),y_train[j],cv=5))
    model.fit(X_train.toarray(),y_train[j])
    bb = model.predict_proba(X_valid.toarray())[:, 1]
    false_positive_rate, true_positive_rate, thresholds = roc_curve(y_valid[j], b
    score=(roc_auc_score(y_valid[j],bb))
    roc_curve(y_valid[j].values, bb)
    plt.title('Receiver Operating Characteristic')
    plt.plot(false_positive_rate, true_positive_rate, 'b',label='AUC %0.3f' % sco
    plt.legend(loc='lower right')
    plt.plot([0,1],[0,1],'r--')
    plt.xlim([-0.1,1.2])
    plt.ylim([-0.1,1.2])
    plt.ylabel('True Positive Rate')
    plt.xlabel('False Positive Rate')
    plt.show()
```

```
[ 0.89365854  0.89409468  0.89409468  0.89409468  0.89409468]
```




```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
<ipython-input-148-7c3d0ce200e9> in <module>()
      1 model=SVC(probability=True)
      2 for i, j in enumerate(TARGET_COLS):
----> 3     print(cross_val_score(model,X_train.toarray(),y_train[j],cv=5))
      4     model.fit(X_train.toarray(),y_train[j])
      5     bb = model.predict_proba(X_valid.toarray())[:, 1]

/usr/local/lib/python2.7/dist-packages/sklearn/model_selection/_validation.pyc
 in cross_val_score(estimator, X, y, groups, scoring, cv, n_jobs, verbose, fit_
params, pre_dispatch)
    138                                                 train, test, verbose, Non
e,
    139                                                 fit_params)
--> 140                         for train, test in cv_iter)
    141     return np.array(scores)[:, 0]
    142
```

```
/usr/local/lib/python2.7/dist-packages/sklearn/externals/joblib/parallel.pyc in
 __call__(self, iterable)
    756             # was dispatched. In particular this covers the edge
    757             # case of Parallel used with an exhausted iterator.
--> 758             while self.dispatch_one_batch(iterator):
    759                 self._iterating = True
    760             else:

/usr/local/lib/python2.7/dist-packages/sklearn/externals/joblib/parallel.pyc in
 dispatch_one_batch(self, iterator)
    606                 return False
    607             else:
--> 608                 self._dispatch(tasks)
    609                 return True
    610

/usr/local/lib/python2.7/dist-packages/sklearn/externals/joblib/parallel.pyc in
 _dispatch(self, batch)
    569         dispatch_timestamp = time.time()
    570         cb = BatchCompletionCallBack(dispatch_timestamp, len(batch), se
lf)
--> 571         job = self._backend.apply_async(batch, callback=cb)
    572         self._jobs.append(job)
    573

/usr/local/lib/python2.7/dist-packages/sklearn/externals/joblib/_parallel_backe
nds.pyc in apply_async(self, func, callback)
    107     def apply_async(self, func, callback=None):
    108         """Schedule a func to be run"""
--> 109         result = ImmediateResult(func)
    110         if callback:
    111             callback(result)

/usr/local/lib/python2.7/dist-packages/sklearn/externals/joblib/_parallel_backe
nds.pyc in __init__(self, batch)
    324         # Don't delay the application, to avoid keeping the input
    325         # arguments in memory
--> 326         self.results = batch()
    327
    328     def get(self):

/usr/local/lib/python2.7/dist-packages/sklearn/externals/joblib/parallel.pyc in
 __call__(self)
    129
    130     def __call__(self):
--> 131         return [func(*args, **kwargs) for func, args, kwargs in self.it
ems]
    132
    133     def __len__(self):

/usr/local/lib/python2.7/dist-packages/sklearn/model_selection/_validation.pyc
 in _fit_and_score(estimator, X, y, scorer, train, test, verbose, parameters, f
it_params, return_train_score, return_parameters, return_n_test_samples, return
_times, error_score)
    236             estimator.fit(X_train, **fit_params)
    237         else:
--> 238             estimator.fit(X_train, y_train, **fit_params)
```

```
    239
    240     except Exception as e:

/usr/local/lib/python2.7/dist-packages/sklearn/svm/base.pyc in fit(self, X, y,
 sample_weight)
    187
    188         seed = rnd.randint(np.iinfo('i').max)
--> 189         fit(X, y, sample_weight, solver_type, kernel, random_seed=seed)
    190         # see comment on the other call to np.iinfo in this file
    191

/usr/local/lib/python2.7/dist-packages/sklearn/svm/base.pyc in _dense_fit(self,
 X, y, sample_weight, solver_type, kernel, random_seed)
    254                 cache_size=self.cache_size, coef0=self.coef0,
    255                 gamma=self._gamma, epsilon=self.epsilon,
--> 256                 max_iter=self.max_iter, random_seed=random_seed)
    257
    258         self._warn_from_fit_status()

KeyboardInterrupt:
```

In [ ]: