In [132]:
```
# Dependencies
import tweepy
import numpy as np
import pandas as pd
from datetime import datetime
import matplotlib.pyplot as plt
from matplotlib import style
style.use('ggplot')

# Import and Initialize Sentiment Analyzer
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
analyzer = SentimentIntensityAnalyzer()

# Twitter API Keys
from config import (consumer_key,
                    consumer_secret,
                    access_token,
                    access_token_secret)

# Setup Tweepy API Authentication
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api = tweepy.API(auth, parser=tweepy.parsers.JSONParser())
```

In [263]:
```
# Target Account
target_users = ("@BBC")
# Variables for holding sentiments
sentiments = []
oldest_tweet = None
# Counter
counter = 1
# Loop through all target users
for target in target_users:

# Loop through 5 pages of tweets
    for x in range(5):
        public_tweets = api.user_timeline(target_users,
                                          count=20,
                                         max_id=oldest_tweet)


        for tweet in public_tweets:
        # Run Vader Analysis on each tweet
            results = analyzer.polarity_scores(tweet["text"])
            compound = results["compound"]
            pos = results["pos"]
            neu = results["neu"]
            neg = results["neg"]
            tweets_ago = counter


             # Add to counter

            counter = counter + 1

        # Add sentiments for each tweet into an array
            sentiments.append({"Name":target_users,
                               "Date": tweet["created_at"],
                               "Compound": compound,
                               "Positive": pos,
                               "Negative": neu,
                               "Neutral": neg,
                               "Tweets Ago": counter})


    oldest_tweet = int(tweet['id_str']) - 1

# Convert sentiments to DataFrame
sentiment_pd_bbc = pd.DataFrame.from_dict(sentiments)
```

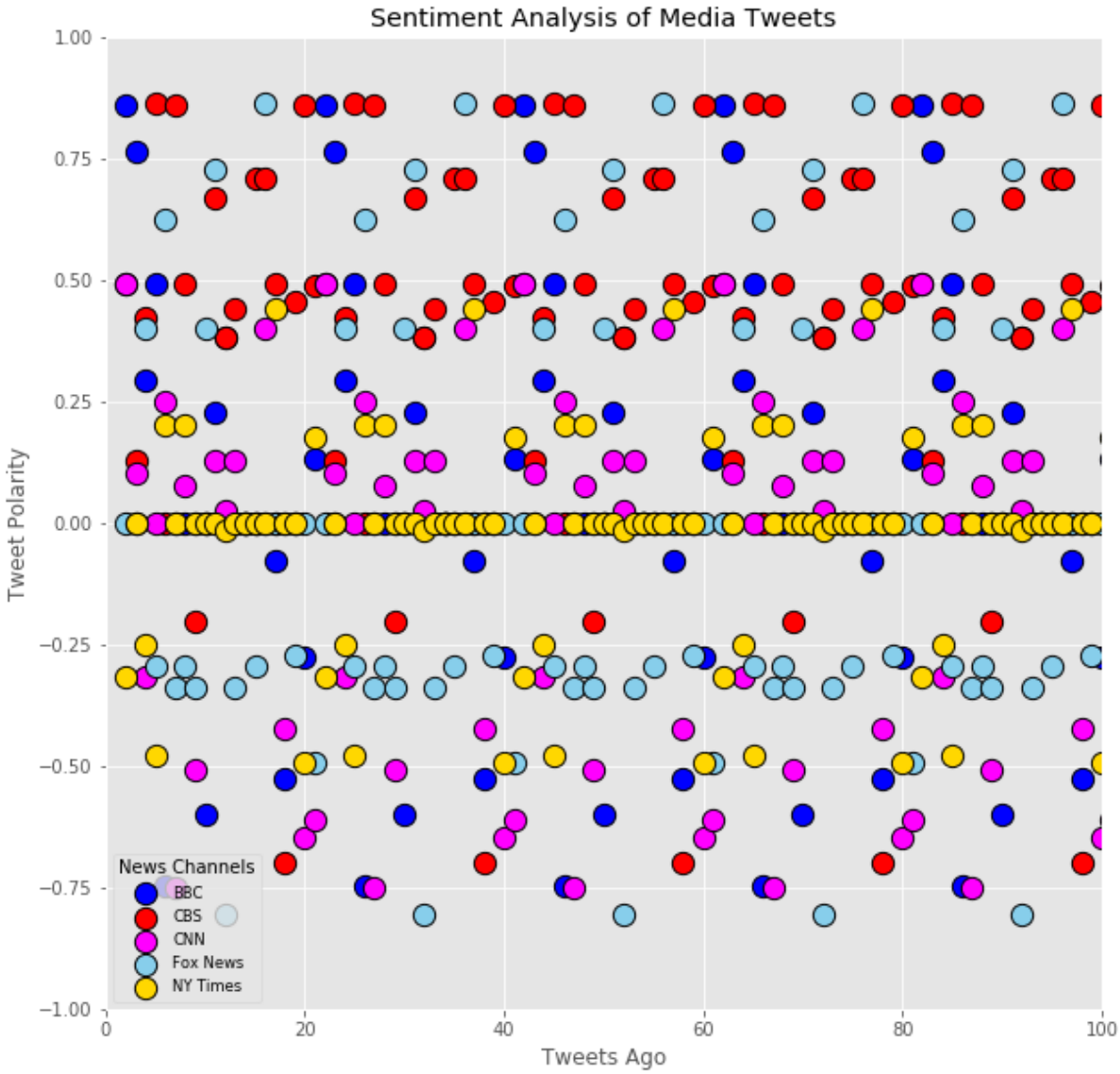
In [256]:
```
# Target Account
target_users = ("@CBS")
# Variables for holding sentiments
sentiments = []
oldest_tweet = None
# Counter
counter = 1
# Loop through all target users
for target in target_users:

# Loop through 5 pages of tweets
    for x in range(5):
        public_tweets = api.user_timeline(target_users,
                                          count=20,
                                         max_id=oldest_tweet)


        for tweet in public_tweets:
        # Run Vader Analysis on each tweet
            results = analyzer.polarity_scores(tweet["text"])
            compound = results["compound"]
            pos = results["pos"]
            neu = results["neu"]
            neg = results["neg"]
            tweets_ago = counter


             # Add to counter

            counter = counter + 1

        # Add sentiments for each tweet into an array
            sentiments.append({"Name":target_users,
                               "Date": tweet["created_at"],
                               "Compound": compound,
                               "Positive": pos,
                               "Negative": neu,
                               "Neutral": neg,
                               "Tweets Ago": counter})


    oldest_tweet = int(tweet['id_str']) - 1

# Convert sentiments to DataFrame
sentiment_pd_cbs = pd.DataFrame.from_dict(sentiments)
```

In [258]:
```
# Target Account
target_users = ("@CNN")
# Variables for holding sentiments
sentiments = []
oldest_tweet = None
# Counter
counter = 1
# Loop through all target users
for target in target_users:

# Loop through 5 pages of tweets
    for x in range(5):
        public_tweets = api.user_timeline(target_users,
                                          count=20,
                                         max_id=oldest_tweet)


        for tweet in public_tweets:
        # Run Vader Analysis on each tweet
            results = analyzer.polarity_scores(tweet["text"])
            compound = results["compound"]
            pos = results["pos"]
            neu = results["neu"]
            neg = results["neg"]
            tweets_ago = counter


             # Add to counter

            counter = counter + 1

        # Add sentiments for each tweet into an array
            sentiments.append({"Name":target_users,
                               "Date": tweet["created_at"],
                               "Compound": compound,
                               "Positive": pos,
                               "Negative": neu,
                               "Neutral": neg,
                               "Tweets Ago": counter})


    oldest_tweet = int(tweet['id_str']) - 1

# Convert sentiments to DataFrame
sentiment_pd_cnn = pd.DataFrame.from_dict(sentiments)
```

In [261]:
```
# Target Account
target_users = ("@FoxNews")
# Variables for holding sentiments
sentiments = []
oldest_tweet = None
# Counter
counter = 1
# Loop through all target users
for target in target_users:

# Loop through 5 pages of tweets
    for x in range(5):
        public_tweets = api.user_timeline(target_users,
                                          count=20,
                                         max_id=oldest_tweet)


        for tweet in public_tweets:
        # Run Vader Analysis on each tweet
            results = analyzer.polarity_scores(tweet["text"])
            compound = results["compound"]
            pos = results["pos"]
            neu = results["neu"]
            neg = results["neg"]
            tweets_ago = counter


             # Add to counter

            counter = counter + 1

        # Add sentiments for each tweet into an array
            sentiments.append({"Name":target_users,
                               "Date": tweet["created_at"],
                               "Compound": compound,
                               "Positive": pos,
                               "Negative": neu,
                               "Neutral": neg,
                               "Tweets Ago": counter})


    oldest_tweet = int(tweet['id_str']) - 1

# Convert sentiments to DataFrame
sentiment_pd_fox = pd.DataFrame.from_dict(sentiments)
```

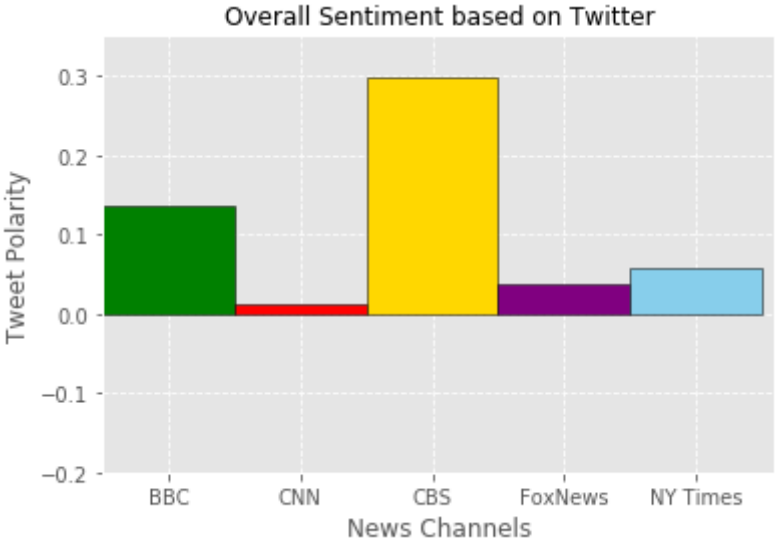
In [262]:
```
# Target Account
target_users = ("@nytimes")
# Variables for holding sentiments
sentiments = []
oldest_tweet = None
# Counter
counter = 1
# Loop through all target users
for target in target_users:

# Loop through 5 pages of tweets
    for x in range(5):
        public_tweets = api.user_timeline(target_users,
                                          count=20,
                                         max_id=oldest_tweet)


        for tweet in public_tweets:
        # Run Vader Analysis on each tweet
            results = analyzer.polarity_scores(tweet["text"])
            compound = results["compound"]
            pos = results["pos"]
            neu = results["neu"]
            neg = results["neg"]
            tweets_ago = counter


             # Add to counter

            counter = counter + 1

        # Add sentiments for each tweet into an array
            sentiments.append({"Name":target_users,
                               "Date": tweet["created_at"],
                               "Compound": compound,
                               "Positive": pos,
                               "Negative": neu,
                               "Neutral": neg,
                               "Tweets Ago": counter})


    oldest_tweet = int(tweet['id_str']) - 1

# Convert sentiments to DataFrame
sentiment_pd_nytimes = pd.DataFrame.from_dict(sentiments)
```

In [342]:
```
plt.figure(figsize=(10,10))
# Build the scatter plots
plt.scatter(sentiment_pd_bbc['Tweets Ago'],
            sentiment_pd_bbc['Compound'],
            s=150,color="blue",
            edgecolor="black", linewidths=1, marker="o",
            alpha=1.0, label="BBC")

plt.scatter(sentiment_pd_cbs['Tweets Ago'],
            sentiment_pd_cbs['Compound'],
            s=150,c="red",
            edgecolor="black", linewidths=1, marker="o",
            alpha=1.0, label="CBS")
plt.scatter(sentiment_pd_cnn['Tweets Ago'],
            sentiment_pd_cnn['Compound'],
            s=150,c="magenta",
            edgecolor="black", linewidths=1, marker="o",
            alpha=1.0, label="CNN")
plt.scatter(sentiment_pd_fox['Tweets Ago'],
            sentiment_pd_fox['Compound'],
            s=150,c="skyblue",
            edgecolor="black", linewidths=1, marker="o",
            alpha=1.0, label="Fox News")
plt.scatter(sentiment_pd_nytimes['Tweets Ago'],
            sentiment_pd_nytimes['Compound'],
            s=150,c="gold",
            edgecolor="black", linewidths=1, marker="o",
            alpha=1.0, label="NY Times")
lgnd = plt.legend(fontsize="small", mode="Expanded",
                  numpoints=1, scatterpoints=1,
                  loc="best", title="News Channels",
                  labelspacing=0.5)


# Incorporate the other graph properties
plt.title("Sentiment Analysis of Media Tweets")
plt.ylabel("Tweet Polarity")
plt.xlabel("Tweets Ago")
plt.xlim((0,100))
plt.ylim((-1,1))
plt.grid(True)

plt.savefig("scatterplot.png")
# Show plot
plt.show()
```



In [333]:
```
#Aggregating the values
bbc_mean = sentiment_pd_bbc["Compound"].mean()
cnn_mean = sentiment_pd_cnn["Compound"].mean()
cbs_mean = sentiment_pd_cbs["Compound"].mean()
fox_mean = sentiment_pd_fox["Compound"].mean()
nytimes_mean = sentiment_pd_nytimes["Compound"].mean()
```

In [343]:
```
#For all the news channels
bbc = plt.bar(0, bbc_mean, color='green', alpha=1, align="edge", ec="bla
ck", width=1)
cnn = plt.bar(1, cnn_mean, color='red', alpha=1, align="edge", ec="blac
k", width=1)
cbs = plt.bar(2, cbs_mean, color='gold', alpha=1, align="edge", ec="blac
k", width=1)
foxnews = plt.bar(3, fox_mean, color='purple', alpha=1, align="edge", ec
="black", width=1)
nytimes = plt.bar(4, nytimes_mean, color='skyblue', alpha=1, align="edg
e", ec="black", width=1)
def label(graphs):
 for graph in graphs:
    height = graph.get_height()
    plt.text(graph.get_x() + graph.get_width()/2., -8,'%d' % int(height)
 + "%", ha='center', va='bottom', color='white', fontsize=12)

def label(graphs):
 for graph in graphs:
    height = graph.get_height()
    plt.text(graph.get_x() + graph.get_width()/2., 2,'%d' % int(height)
+ "%",ha='center', va='bottom', color='white', fontsize=12)

#Axis assignment
xvalues=["BBC", "CNN", "CBS", "FoxNews", "NY Times"]
x_axis1 = np.arange(0,len(xvalues),1)
tick_locations = [value+0.5 for value in x_axis1]
plt.xticks(tick_locations, (xvalues))
plt.grid(linestyle="dashed")
#Axes Limits
plt.xlim(0, 5.1)
plt.ylim(-0.2,0.35)
#Axes and Chart Labels
plt.title("Overall Sentiment based on Twitter", fontsize=12)
plt.ylabel("Tweet Polarity")
plt.xlabel("News Channels")


plt.savefig("bargraph.png")
plt.show()
```


Overall Sentiment based on Twitter
Tweet Polarity
0.3
0.2
0.1
0.0
−0.1
−0.2
BBC
CNN
CBS
FoxNews
NY Times
News Channels