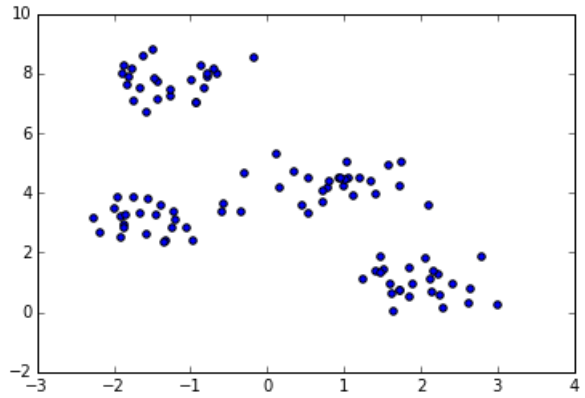
In [1]:

```
import matplotlib.pyplot as plt
plt.jet() # set the color map. When your colors are lost, re-run this.
import sklearn.datasets as datasets
X, y = datasets.make_blobs(centers=4, cluster_std=0.5, random_state=0)
```

```
<matplotlib.figure.Figure at 0x547b850>
```

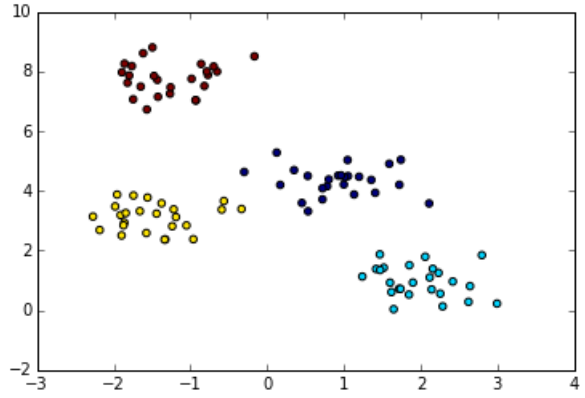
In [2]:

```
%matplotlib inline
#Remember you don't need the previous line in Spyder
import pylab as plt
plt.scatter(X[:,0], X[:,1]);
plt.show()
```



In [3]:

```
plt.scatter(X[:,0], X[:,1], c=y);
```

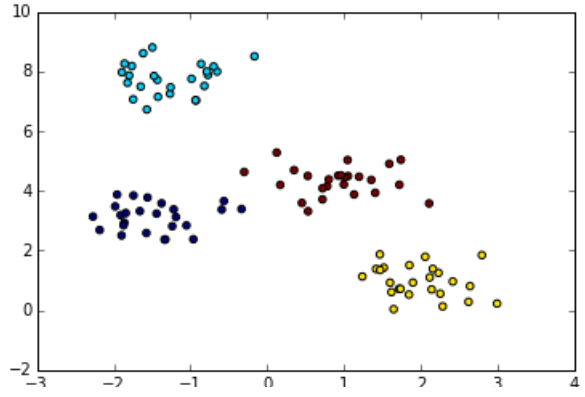


In [4]:

```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=4, random_state=8) #You guess there are 4 groups in the data
y_hat = kmeans.fit(X).labels_
```

In [5]:

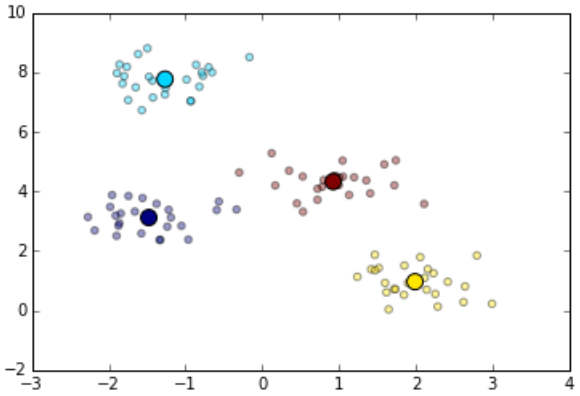
```
plt.scatter(X[:,0], X[:,1], c=y_hat);
```



In [6]:

```
import numpy as np
plt.scatter(X[:,0], X[:,1], c=y_hat, alpha=0.4)
mu = kmeans.cluster_centers_
plt.scatter(mu[:,0], mu[:,1], s=100, c=np.unique(y_hat))
print mu
```

```
[[-1.47935679  3.11716896]
 [-1.26811733  7.76378266]
 [ 1.99186903  0.96561071]
 [ 0.92578447  4.32475792]]
```



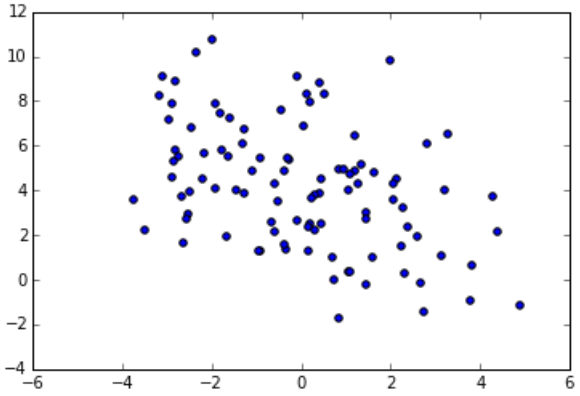
In [11]:

```
import matplotlib.pyplot as plt
plt.jet() # set the color map. When your colors are lost, re-run this.
import sklearn.datasets as datasets
X, y = datasets.make_blobs(centers=4, cluster_std=1.5, random_state=0)
```

```
<matplotlib.figure.Figure at 0xd378350>
```

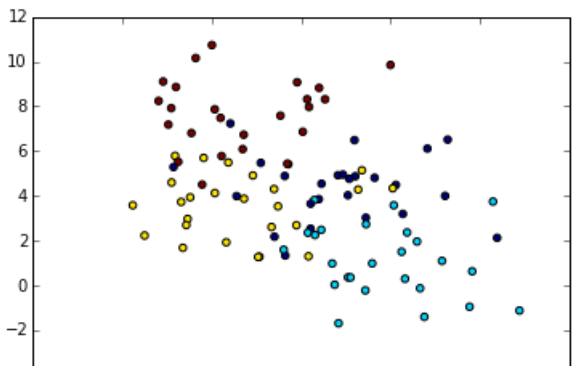
In [12]:

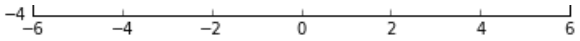
```
import pylab as plt
plt.scatter(X[:,0], X[:,1]);
plt.show()
```



In [13]:

```
plt.scatter(X[:,0], X[:,1], c=y);
```



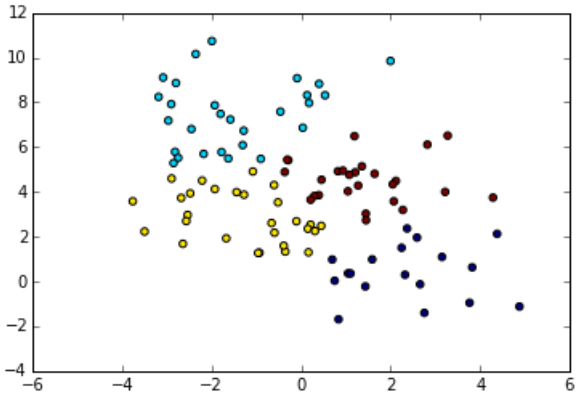


In [14]:

```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=4, random_state=8) #You guess there are 4 groups in the data
y_hat = kmeans.fit(X).labels_ #y_hat contains the estimated group belonging of each data point
```

In [15]:

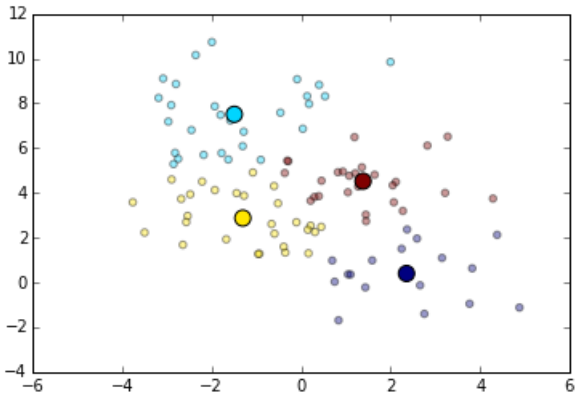
```
plt.scatter(X[:,0], X[:,1], c=y_hat);
```



In [16]:

```
import numpy as np
plt.scatter(X[:,0], X[:,1], c=y_hat, alpha=0.4)
mu = kmeans.cluster_centers_
plt.scatter(mu[:,0], mu[:,1], s=100, c=np.unique(y_hat))
print mu
```

```
[[ 2.35482022  0.39586806]
 [-1.49467058  7.5040995 ]
 [-1.30509451  2.87059874]
 [ 1.38273928  4.51892998]]
```

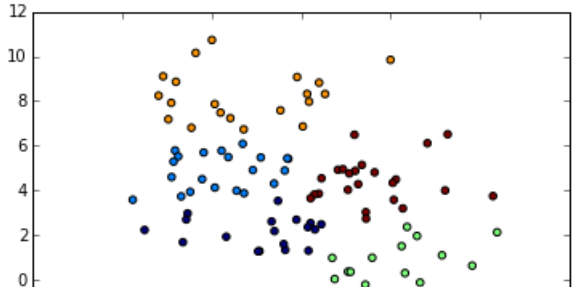


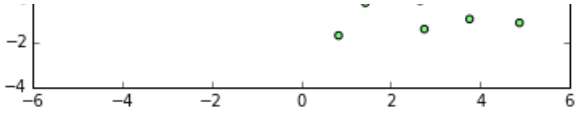
In [17]:

```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=5, random_state=8) #You guess there are 4 groups in the data
y_hat = kmeans.fit(X).labels_ #y_hat contains the estimated group belonging of each data point
```

In [18]:

```
plt.scatter(X[:,0], X[:,1], c=y_hat);
```

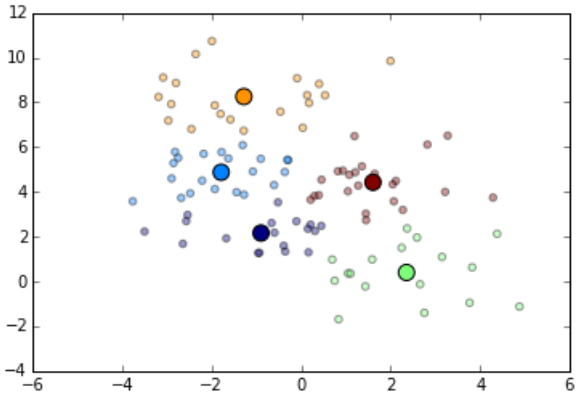




In [19]:

```
import numpy as np
plt.scatter(X[:,0], X[:,1], c=y_hat, alpha=0.4)
mu = kmeans.cluster_centers_
plt.scatter(mu[:,0], mu[:,1], s=100, c=np.unique(y_hat))
print mu
```

```
[[-0.89838181  2.15896767]
 [-1.7883789   4.87302073]
 [ 2.35482022  0.39586806]
 [-1.2815995   8.25116189]
 [ 1.60507036  4.424593  ]]
```



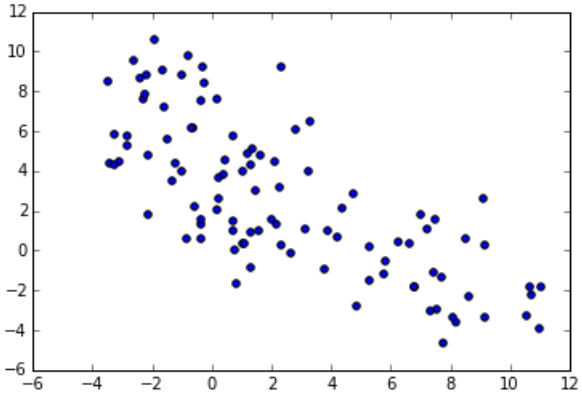
In [20]:

```
import matplotlib.pyplot as plt
plt.jet() # set the color map. When your colors are lost, re-run this.
import sklearn.datasets as datasets
X, y = datasets.make_blobs(centers=6, cluster_std=1.5, random_state=0)
```

```
<matplotlib.figure.Figure at 0xd60aa70>
```

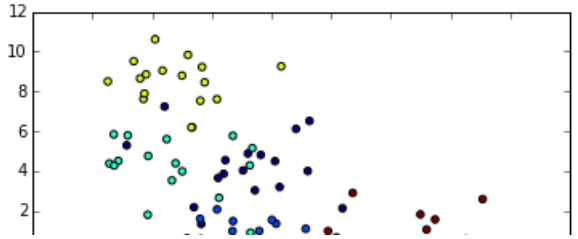
In [21]:

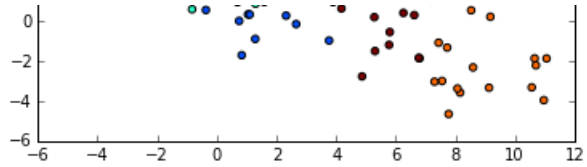
```
import pylab as plt
plt.scatter(X[:,0], X[:,1]);
plt.show()
```



In [22]:

```
plt.scatter(X[:,0], X[:,1], c=y);
```



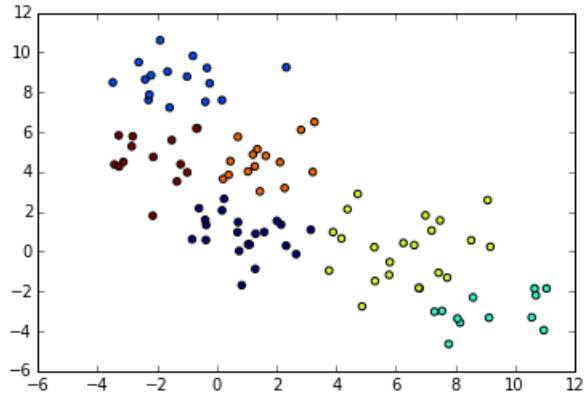


In [23]:

```
kmeans = KMeans(n_clusters=6, random_state=8) #choose 6 clusters for 6 centers
y_hat = kmeans.fit(X).labels_ #y_hat contains the estimated group belonging of each data point
```

In [24]:

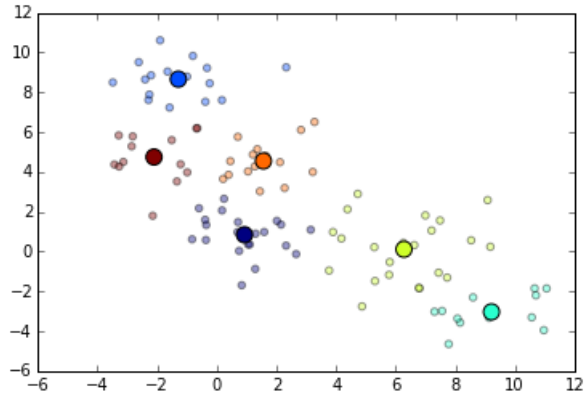
```
plt.scatter(X[:,0], X[:,1], c=y_hat);
```



In [25]:

```
import numpy as np
plt.scatter(X[:,0], X[:,1], c=y_hat, alpha=0.4)
mu = kmeans.cluster_centers_
plt.scatter(mu[:,0], mu[:,1], s=100, c=np.unique(y_hat))
print mu
```

```
[[ 0.92831758  0.84086908]
 [-1.29015068  8.65807783]
 [ 9.20775959 -3.03290697]
 [ 6.27366861  0.11299419]
 [ 1.56304744  4.54811313]
 [-2.10501718  4.74641298]]
```



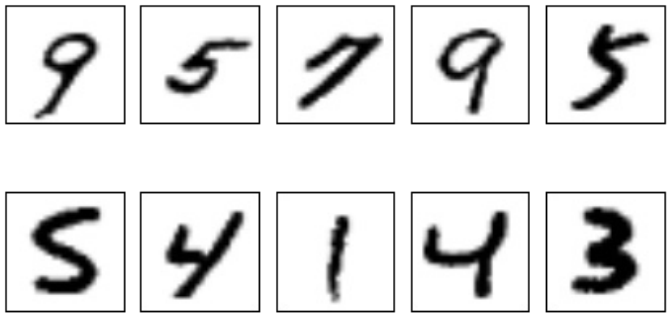
In [26]:

```
from sklearn.datasets import fetch_mldata
from sklearn.cluster import KMeans
from sklearn.utils import shuffle
X_digits, _,_, Y_digits = fetch_mldata("MNIST Original").values() # fetch dataset from internet
X_digits, Y_digits = shuffle(X_digits,Y_digits) # shuffle dataset (which is ordered!)
X_digits = X_digits[-5000:]       # take only the last instances, to shorten runtime of KMeans
```

In [27]:

```
plt.rc("image", cmap="binary") # use black/white palette for plotting
for i in xrange(10):
    plt.subplot(2,5,i+1)
```

```
    plt.imshow(X_digits[i].reshape(28,28))
    plt.xticks(())
    plt.yticks(())
plt.tight_layout()
```

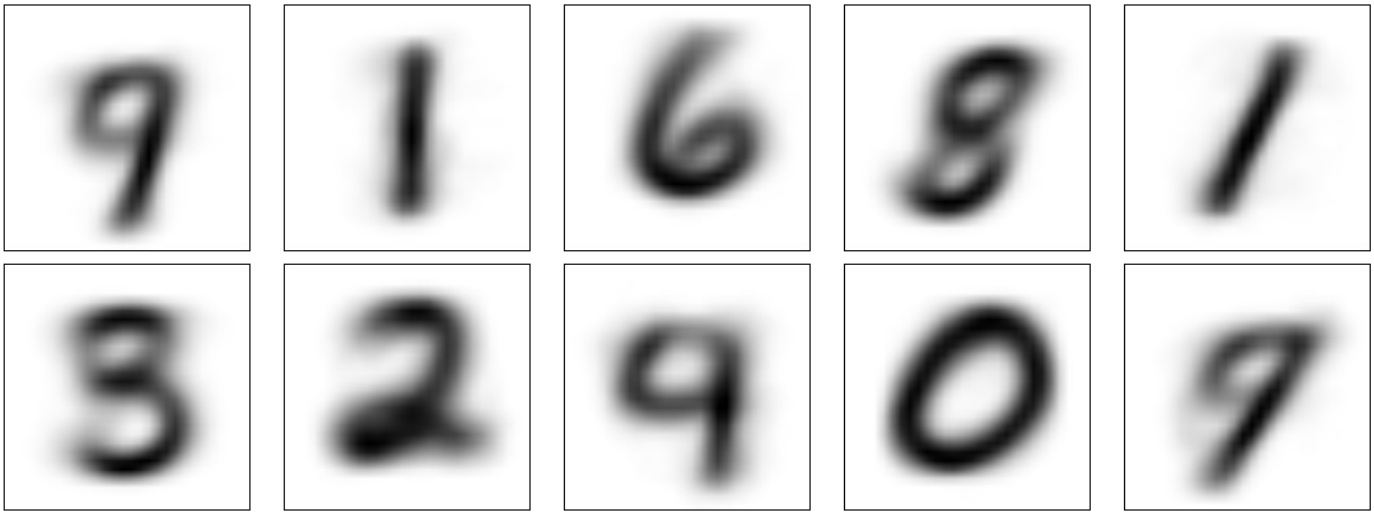


In [28]:

```
kmeans = KMeans(n_clusters=10)
mu_digits = kmeans.fit(X_digits).cluster_centers_ #mu_digits contains the centroids for each one of
the n_clusters
digits_prediction = kmeans.fit(X_digits).labels_ #digits_prediction contains a predictions of to which
cluster each instance belongs
```

In [29]:

```
plt.figure(figsize=(16,6))
for i in xrange(2*(mu_digits.shape[0]/2)): # loop over all means
    plt.subplot(2,mu_digits.shape[0]/2,i+1)
    plt.imshow(mu_digits[i].reshape(28,28))
    plt.xticks(())
    plt.yticks(())
plt.tight_layout()
```

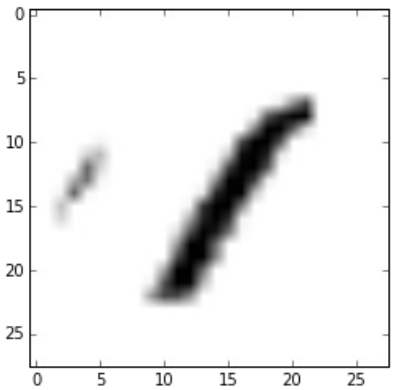


In [30]:

```
plt.imshow(X_digits[318].reshape(28,28))
```

Out[30]:

```
<matplotlib.image.AxesImage at 0xd363850>
```



In [56]:

```
kmeans = KMeans(n_clusters=10)
mu_digits = kmeans.fit(X_digits).cluster_centers_ #mu_digits contains the centroids for each one of
the n_clusters
digits_prediction = kmeans.fit(X_digits).labels_
```

In [146]:

```
X_digits, _,_, Y_digits = fetch_mldata("MNIST Original").values() # fetch dataset from internet
X_digits, Y_digits = shuffle(X_digits,Y_digits) # shuffle dataset (which is ordered!)
X_digits = X_digits[-5000:]
```

In [147]:

```
print "Unique entries of y_digits:", np.unique(Y_digits)
```

```
Unique entries of y_digits: [ 0.  1.  2.  3.  4.  5.  6.  7.  8.  9.]
```

In [148]:

```
X_digits, _,_, Y_digits = fetch_mldata("MNIST Original").values() # fetch dataset from internet
```

In [149]:

```
group5X=X_digits[Y_digits==5]
```

In [150]:

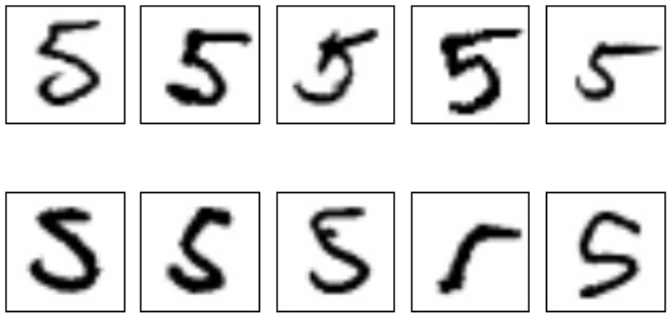
```
group5Y=Y_digits[Y_digits==5]
```

In [151]:

```
group5X, group5Y = shuffle(group5X,group5Y) # shuffle dataset (which is ordered!)
group5X= group5X[-5000:]
```

In [152]:

```
plt.rc("image", cmap="binary") # use black/white palette for plotting
for i in xrange(10):
    plt.subplot(2,5,i+1)
    plt.imshow(group5X[i].reshape(28,28))
    plt.xticks(())
    plt.yticks(())
plt.tight_layout()
```

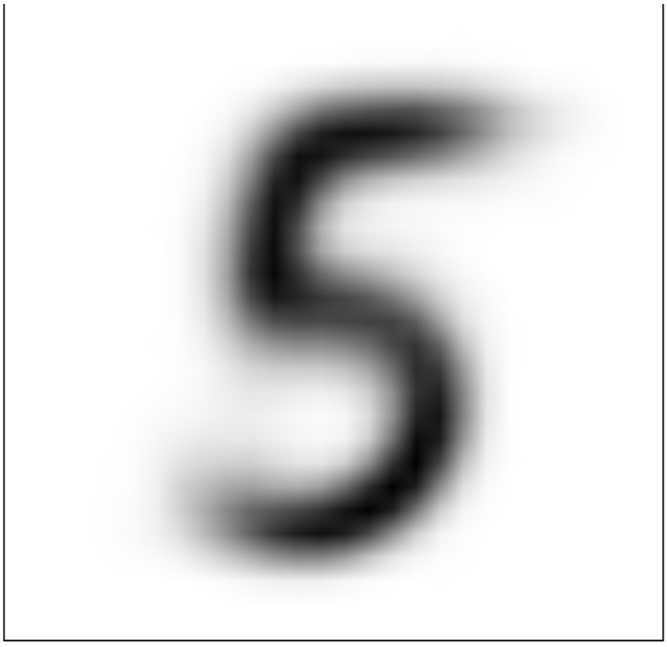


In [155]:

```
kmeans = KMeans(n_clusters=1)
mu_digits = kmeans.fit(group5X).cluster_centers_ #mu_digits contains the centroids for each one of the
n_clusters
digits_prediction = kmeans.fit(group5X).labels_
```

In [159]:

```
plt.figure(figsize=(16,6))
for i in xrange(mu_digits.shape[0]/10): # choose 1 centroid for 5 group
    #plt.subplot(2,mu_digits.shape[0]/2,i+1)
    plt.imshow(mu_digits[i].reshape(28,28))
    plt.xticks(())
    plt.yticks(())
plt.tight_layout()
```

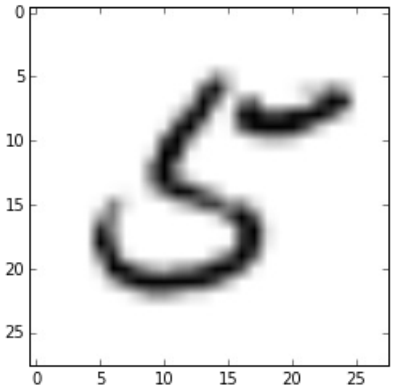


In [160]:

```
plt.imshow(group5X[318].reshape(28,28))
```

Out[160]:

<matplotlib.image.AxesImage at 0x106c3450>



In [ ]:

```

```