In [163]:

```
from sklearn.ensemble import RandomForestClassifier
import pickle
import numpy as np
```

In [164]:

```
with open('X_train.pkl', 'rb') as input:
    X_train = pickle.load(input)
with open('y_train.pkl', 'rb') as input:
    y_train = pickle.load(input)
X_train = X_train[(y_train == 0) | (y_train == 1) | (y_train == 2)]
y_train = y_train[(y_train == 0) | (y_train == 1) | (y_train == 2)]
X_train.shape
```

Out[164]:

```
(155576, 32)
```

In [325]:

```
forest = RandomForestClassifier(n_estimators=50, criterion='entropy', ma
x_depth=13, min_samples_split=2,
                                min_samples_leaf=1, bootstrap=True, n_jo
bs=2, random_state=0, class_weight='balanced')
forest.fit(X_train,y_train)
```

Out[325]:

```
RandomForestClassifier(bootstrap=True, class_weight='balanced',
            criterion='entropy', max_depth=13, max_features='auto',
            max_leaf_nodes=None, min_impurity_decrease=0.0,
            min_impurity_split=None, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            n_estimators=50, n_jobs=2, oob_score=False, random_state=0,
            verbose=0, warm_start=False)
```

Accuracy Measure and HeatMap for Training Set

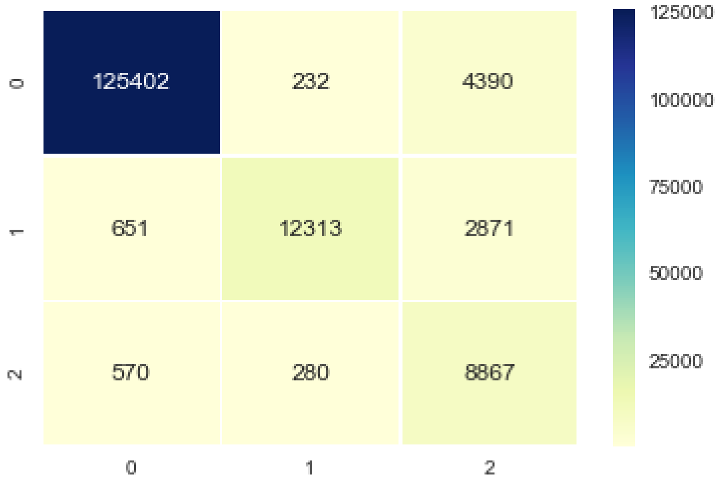
In [326]:

```
import seaborn as sns
from sklearn.metrics import accuracy_score,confusion_matrix
y_pred = forest.predict(X_train)
print(accuracy_score(y_train,y_pred))
cm = confusion_matrix(y_train,y_pred)
sns.heatmap(cm, annot=True, fmt="d", linewidths=.5, cmap='YlGnBu')
```

```
0.942189026585
```

Out[326]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a0f18cbe0>
```



Individual Label Accuracy

In [327]:

```
fix = (cm[0][0])/(cm[1][0]+cm[2][0]+cm[0][0])
sac = (cm[1][1])/(cm[0][1]+cm[2][1]+cm[1][1])
pso = (cm[2][2])/(cm[0][2]+cm[1][2]+cm[2][2])

print("fixation: %0.2f \nsaccades: %0.2f \npso: %0.2f" % (fix, sac, pso
))
```

```
fixation: 0.99
saccades: 0.96
pso: 0.55
```

In [328]:

```
'''
from sklearn.model_selection import cross_val_predict
from sklearn import metrics
pred = cross_val_predict(forest, X_train, y_train, cv=3)
metrics.accuracy_score(y_train, pred)
'''
```

Out[328]:

```
'\nfrom sklearn.model_selection import cross_val_predict\nfrom sklearn
import metrics\npred = cross_val_predict(forest, X_train, y_train, cv=
3)\nmetrics.accuracy_score(y_train, pred)\n'
```

Cross-Validation with Test Set

In [329]:

```
with open('X_test.pkl', 'rb') as input:
    X_val = pickle.load(input)
with open('y_test.pkl', 'rb') as input:
    y_val = pickle.load(input)
X_val = X_val[(y_val == 0) | (y_val == 1) | (y_val == 2)]
y_val = y_val[(y_val == 0) | (y_val == 1) | (y_val == 2)]
X_val.shape
```

Out[329]:

```
(20287, 32)
```

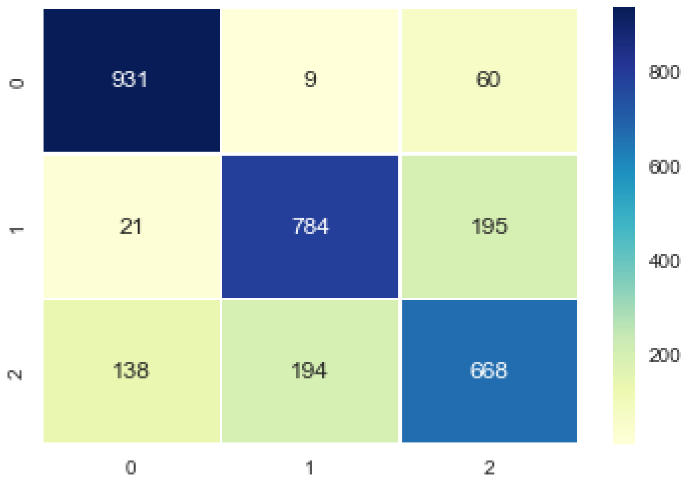
In [330]:

```
X_val_processed = []
y_val_processed = []
for i in [0,1,2]:
    X_tmp = X_val[y_val == i]
    for j in np.random.randint(len(X_tmp), size = 1000):
        X_val_processed.append(X_tmp[j])
        y_val_processed.append(i)
X_val_processed = np.array(X_val_processed)[:,:]
y_val_processed = np.array(y_val_processed)

y_pred = forest.predict(X_val_processed)
cm = confusion_matrix(y_val_processed,y_pred)
sns.heatmap(cm, annot=True, fmt="d", linewidths=.5, cmap='YlGnBu')
```

Out[330]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a0ec2a390>
```



In [331]:

```
fix = (cm[0][0])/(cm[1][0]+cm[2][0]+cm[0][0])
sac = (cm[1][1])/(cm[0][1]+cm[2][1]+cm[1][1])
pso = (cm[2][2])/(cm[0][2]+cm[1][2]+cm[2][2])

print("fixation: %0.2f \nsaccades: %0.2f \npso: %0.2f" % (fix, sac, pso
))
```

```
fixation: 0.85
saccades: 0.79
pso: 0.72
```

In [332]:

```
score = (accuracy_score(y_val_processed,y_pred))
print("Accuracy: %0.3f " % (score))
```

```
Accuracy: 0.794
```