The Seattle Public Library

KNN, RandomForest, and XGBoost Classifiers are doing the best job at the moment.

XGBoost Classifier

In [266]:
```
import xgboost as xgb
```

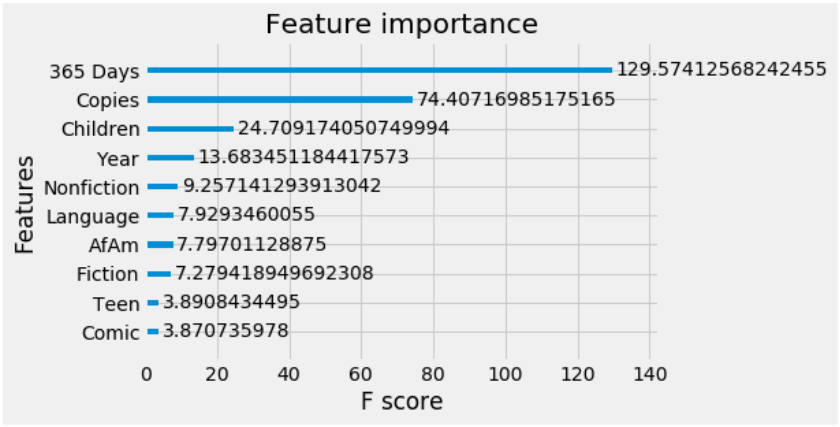
In [267]:
```
gbm = xgb.XGBClassifier(
                        n_estimators=30000,
                        max_depth=4,
                        objective='binary:logistic', #new objective
                        learning_rate=.05,
                        subsample=.8,
                        min_child_weight=3,
                        colsample_bytree=.8
                       )

eval_set=[(X_train,y_train),(X_val,y_val)]
fit_model = gbm.fit(
                    X_train, y_train,
                    eval_set=eval_set,
                    eval_metric='error', #new evaluation metric: classification error (could also use AUC, e.g.)
                    early_stopping_rounds=50,
                    verbose=False
                   )

# accuracy_score(y_test, gbm.predict(X_test, ntree_limit=gbm.best_ntree_limit))
```

In [268]:
```
y_val_predict = fit_model.predict(X_val)
```

In [269]:
```
print("Accuracy: ", accuracy_score(y_val, y_val_predict ))
print("Recall: ",recall_score(y_val, y_val_predict))          # Recall
print("Precision: ",precision_score(y_val, y_val_predict))    # Precision
print("f1: ",f1_score(y_val, y_val_predict))
```

```
Accuracy:  0.8200159277940006
Recall:  0.8055391868002357
Precision:  0.7970845481049562
f1:  0.8012895662368112
```

These are the feature importances sp far from XGBoost



My next step is to consider adding in more features