Predicting Plasmids from Kmer Frequencies with SVM

- Plasmids can be difficult to identify in large sequencing data sets with fragmented assemblies
- Generate model to predict plasmid/chromosome from complete genomes/plasmids in NCBI for *Bacteroides* isolates
- Will apply to ~600 short-read ~300 Nanopore assemblies of *Bacteroides* isolates from my research

Get References

GCA_005706655.1	●	5.24	43.12	chromosome: NZ_CP036555.1 /CP036555.1 plasmid pBF9343: NZ_CP036556.1 /CP036556.1
GCA_000724815.2	●	5.50	43.60	chromosome: NZ_CP037440.1 /CP037440.1
GCA_000724805.2	●	5.48	43.80	chromosome: NZ_CP036553.1 /CP036553.1 plasmid pBFO67_1: NZ_CP036554.1 /CP036554.1

Label DNA

```
GCF_000009925_chromosome.fasta
GCF_000009925_plasmid.fasta
GCF_000011065_chromosome.fasta
GCF_000011065_plasmid.fasta
```

Fragment

```
Split500kb-GCF_000009925_chromosome.fasta
Split500kb-GCF_000009925_plasmid.fasta
Split500kb-GCF_000011065_chromosome.fasta
Split500kb-GCF_000011065_plasmid.fasta
```

Calc. kmer frequencies

Export CSV


```
1 import pandas as pd
2 import numpy as np
3 df = pd.read_csv("Split500kb-kmers.csv",index_col=0)
```

	Acc	Type	AAAAA	AAAAC	AAAAG	AAAAT
1	AP_022661.1	plasmid	0.005667	0.001980	0.004301	0.004392
2	AP_022662.1	plasmid	0.007103	0.004653	0.003306	0.004164
3	AP_022663.1	plasmid	0.005566	0.002551	0.002087	0.003711
4	AP_022664.1	plasmid	0.003239	0.002159	0.001080	0.003598
5	CP_050955.1	plasmid	0.008267	0.004313	0.004313	0.004313

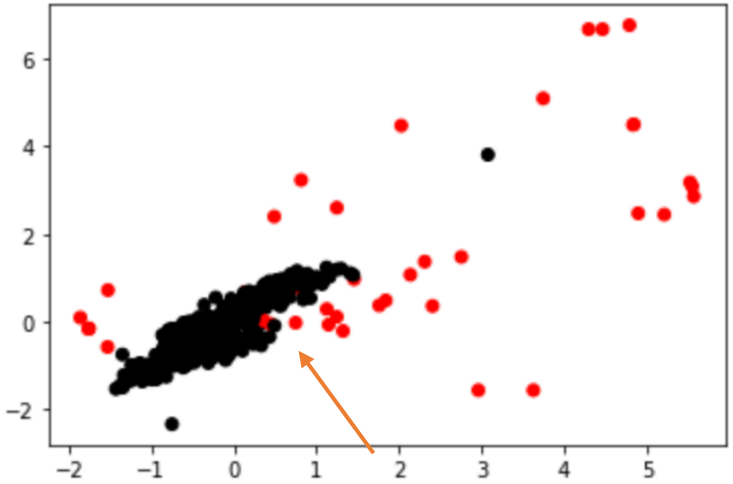
5 rows × 1026 columns

```python
#Create sub-df excluding Y variable
df2=df.iloc[:,2::1]
X=df2
y=df['Type']
```

```python
from sklearn.preprocessing import StandardScaler
#StandardScaler is the appopriate method to transform data for SVM
#assumption is that data features are centered around 0 and
#have variance in the same order.
scaler = StandardScaler()
X_std = scaler.fit_transform(X)
```

Generated scatter plot of features after transformation (red = plasmids)

Hope to figure out how to properly plot hyperplane from model ….
(especially if not linear)




```python
X_train, X_test, y_train, y_test= train_test_split(X_std, y, test_size=0.33)
```

```python
from sklearn.model_selection import GridSearchCV
param_grid = {'C': [0.1,1,10,100], 'gamma': [1,0.1,0.01,0.001],'kernel':['rbf','poly','linear',
                                                                          'sigmoid']}
```

```python
grid = GridSearchCV(SVC(),param_grid,refit=True,verbose=2)
grid.fit(X_train,y_train)
```

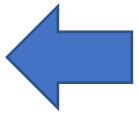
```
[CV] C=0.1, gamma=1, kernel=sigmoid ....................................
[CV] ..................... C=0.1, gamma=1, kernel=sigmoid, total=   0.0s
```

```
|: 1 print(grid.best_estimator_)
```

```
SVC(C=1, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.001, kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

```
|: 1 grid_predictions = grid.predict(X_test)
2 print(confusion_matrix(y_test,grid_predictions))
3 print(classification_report(y_test,grid_predictions))
```

```
[[154   2]
 [  1  12]]
              precision    recall  f1-score   support

  chromosome       0.99      0.99      0.99       156
     plasmid       0.86      0.92      0.89        13

    accuracy                           0.98       169
   macro avg       0.93      0.96      0.94       169
weighted avg       0.98      0.98      0.98       169
```



Unbalanced classes

Export model to test on unfragmented data

```
1 from joblib import dump, load
2 dump(trainedmodelWD, '500kbmodel.joblib')
```

```
['500kbmodel.joblib']
```

Model performs poorly on whole chromosomes/plasmids

```
clf = load('500kbmodel.joblib')
```

```
pred=clf.predict(X_std)
print(confusion_matrix(y,pred))
print(classification_report(y,pred))
```

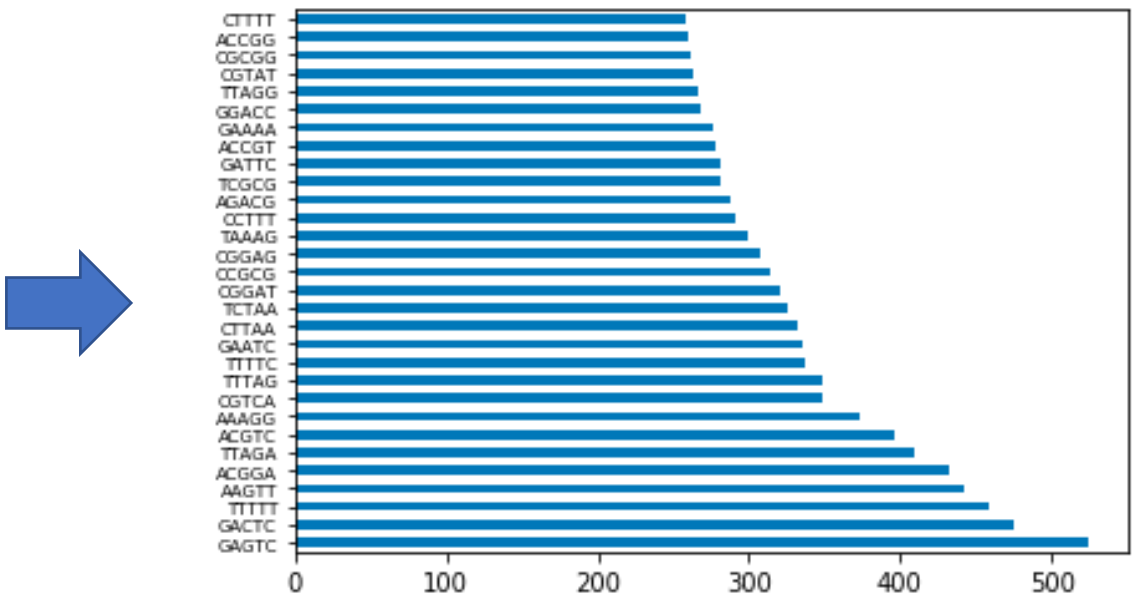
```
[[42  0]
 [40  1]]
              precision    recall  f1-score   support

  chromosome       0.51      1.00      0.68        42
     plasmid       1.00      0.02      0.05        41

    accuracy                           0.52        83
   macro avg       0.76      0.51      0.36        83
weighted avg       0.75      0.52      0.37        83
```

Feature Reduction: SelectKBest

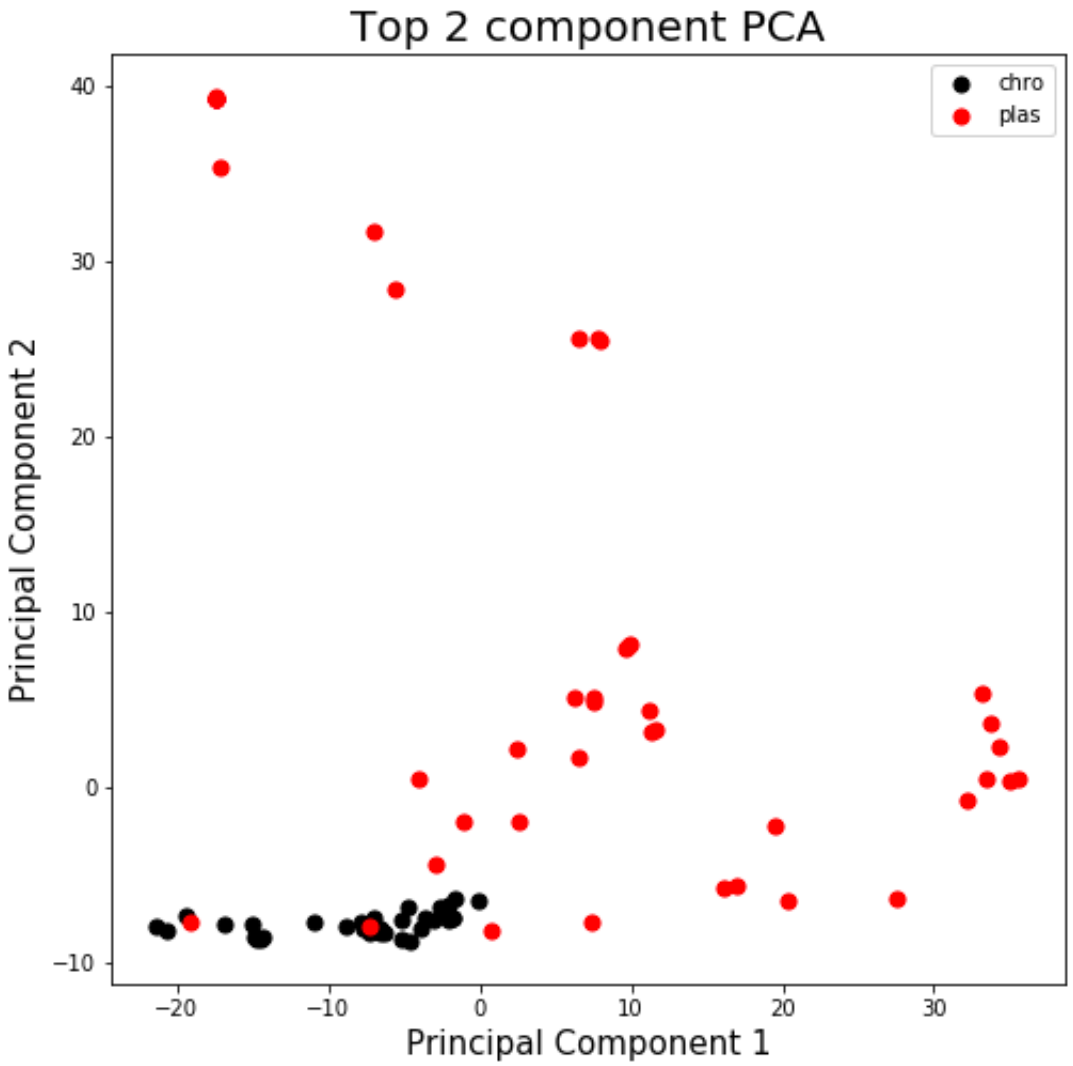
```
import pandas as pd
import numpy as np
from sklearn.feature_selection import SelectKBest

bestfeatures = SelectKBest(k=30)
fit = bestfeatures.fit(X_std,y)
dfscores = pd.DataFrame(fit.scores_)
dfpval= pd.DataFrame(fit.pvalues_)
dfcolumns = pd.DataFrame(X.columns)
featureScores = pd.concat([dfcolumns,dfscores,dfpval],axis=1)
featureScores.columns = ['Specs','Score','Pval']
print(featureScores.nlargest(20,'Score'))
```

Possible solutions:

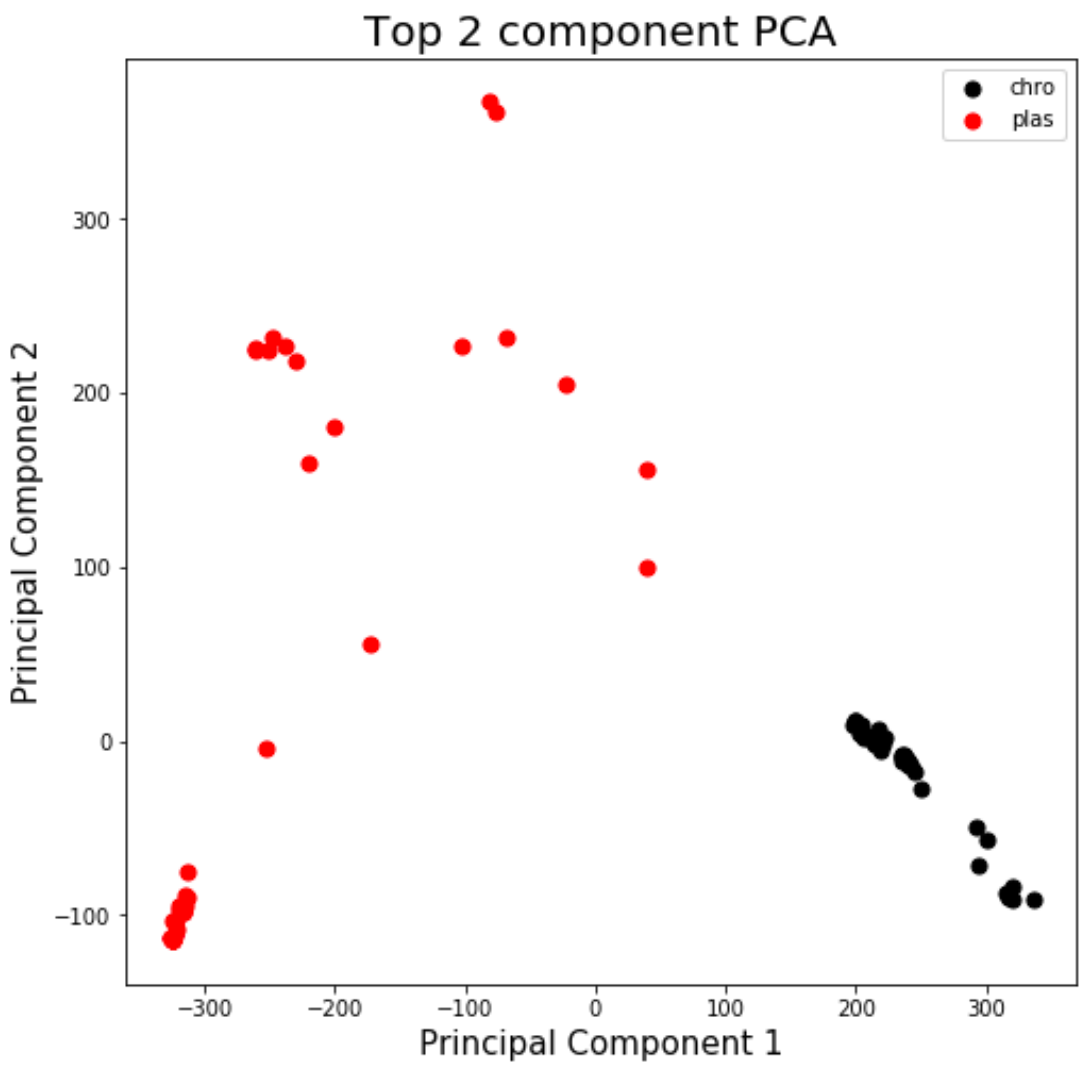
- Train on whole data (better class balance)
- Use larger kmer size (5 = 1024 features, 9 = 262,144 features)
- Reduce number of features to be in better scale with number of samples



5-mers for unfragmented data



9-mers for unfragmented data



Model trained with 9-mers on whole chromosomes and plasmids

In [415]:
```
grid_predictions = grid.predict(X_test)
print(confusion_matrix(y_test,grid_predictions))
print(classification_report(y_test,grid_predictions))
```

```
[[13  0]
 [ 2 13]]
              precision    recall  f1-score   support

  chromosome       0.87      1.00      0.93        13
     plasmid       1.00      0.87      0.93        15

    accuracy                           0.93        28
   macro avg       0.93      0.93      0.93        28
weighted avg       0.94      0.93      0.93        28
```

In [416]:
```
modelwholeData = SVC(C=0.1, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=1, kernel='sigmoid',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

In [417]:
```
trainedmodelWD = modelwholeData.fit(X_std,y)
from joblib import dump, load
dump(trainedmodelWD, 'wholexsomes.joblib')
```

Model applied to fragmented dataset

```
from joblib import dump, load
clf = load('wholexsomes.joblib')
```

```
clf_predictions = clf.predict(X_test)
print(confusion_matrix(y_test,clf_predictions))
print(classification_report(y_test,clf_predictions))
```

```
[[417   3]
 [  0  39]]
              precision    recall  f1-score   support

  chromosome       1.00      0.99      1.00       420
     plasmid       0.93      1.00      0.96        39

    accuracy                           0.99       459
   macro avg       0.96      1.00      0.98       459
weighted avg       0.99      0.99      0.99       459
```