MP3: Pattern Recognition

Abhi Kamboj
Hockenmaier
4/1/2019

Table of Contents:

Contents

Section I:

Part 1.1: Naïve Bayes Model

Results

Overview Screenshot:

```
C:\Users\Abhi Kamboj\ECE-448-Intro-to-AI\MP3 pattern recognition\mp3-code\part1>image_main.py
highest log probabilities: [-2129.09086194 -1139.03812197 -2267.58209501 -1250.77229853
 -2020.27436206  -819.3329071  -2207.7717257  -1141.78450757
 -2003.99323831 -1612.71877855]
lowest log probabilities: [-4062.29499162 -2335.64414529 -4081.80869228 -2948.25022535
 -3592.91335072 -3669.03271733 -5010.60099152 -2401.16866074
 -4543.87070131 -3954.27894561]
classification rates: [0.762 0.894 0.455 0.842 0.634 0.765 0.445 0.905 0.885 0.883]
accuracy: 0.747
Normalized confusion matrix
[[0.762 0.001 0.01  0.11  0.016 0.    0.083 0.    0.018 0.   ]
 [0.006 0.894 0.014 0.045 0.023 0.    0.018 0.    0.    0.   ]
 [0.006 0.    0.455 0.012 0.292 0.    0.221 0.    0.014 0.   ]
 [0.049 0.006 0.    0.842 0.038 0.    0.065 0.    0.    0.   ]
 [0.001 0.001 0.104 0.096 0.634 0.001 0.153 0.    0.01  0.   ]
 [0.    0.    0.    0.    0.    0.765 0.004 0.184 0.005 0.042]
 [0.158 0.    0.098 0.059 0.213 0.001 0.445 0.    0.026 0.   ]
 [0.    0.    0.    0.    0.    0.03  0.    0.905 0.    0.065]
 [0.    0.001 0.002 0.031 0.01  0.022 0.025 0.023 0.885 0.001]
 [0.    0.    0.    0.006 0.    0.033 0.    0.075 0.003 0.883]]

C:\Users\Abhi Kamboj\ECE-448-Intro-to-AI\MP3 pattern recognition\mp3-code\part1>
```

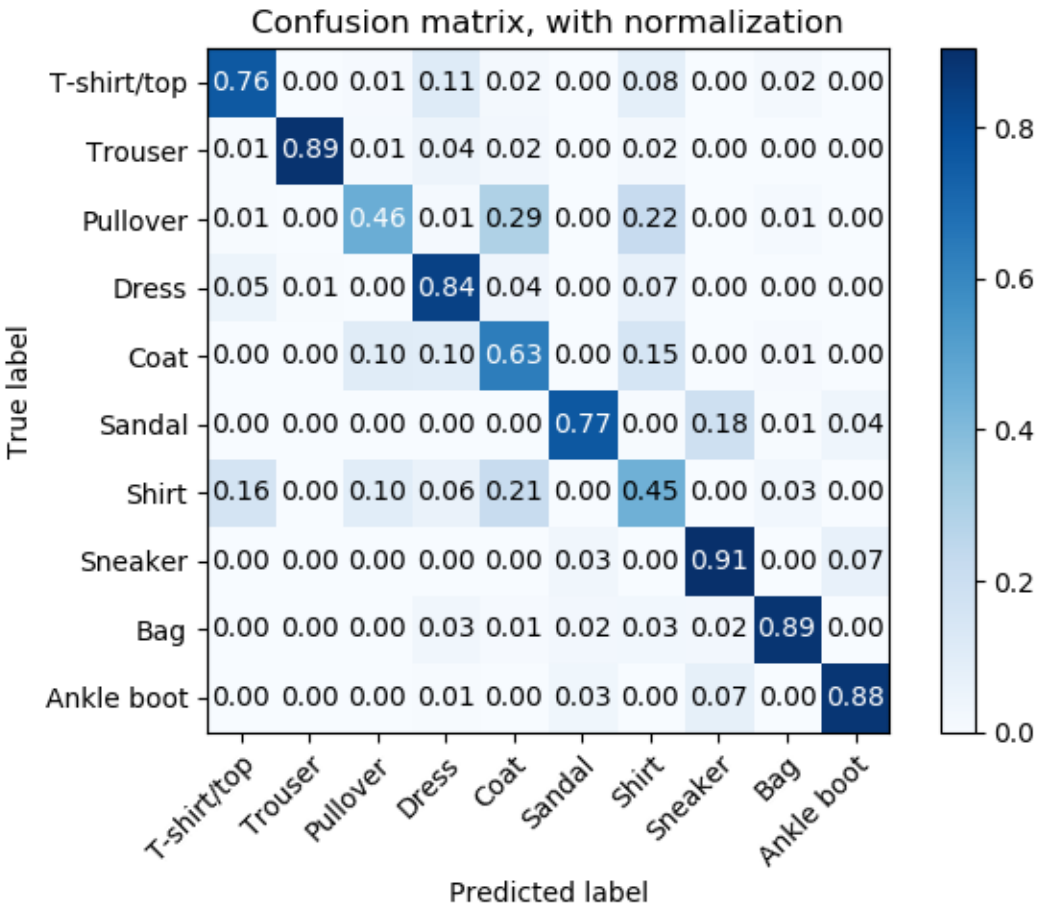
Average Classification Rate:

Average Classification Rate: 0.747

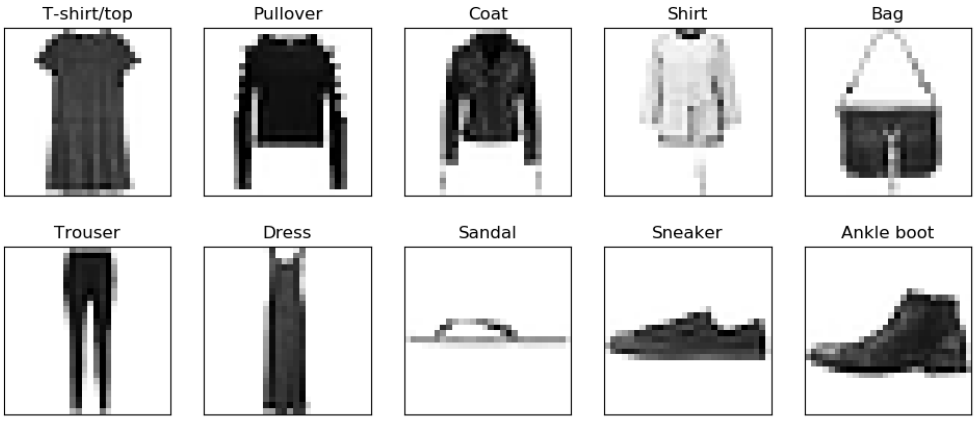
Classification Rate for Each Class:

classification rates: [0.762 0.894 0.455 0.842 0.634 0.765 0.445 0.905 0.885 0.883]
Indices in the array shown above correspond to classes

Confusion Matrix:

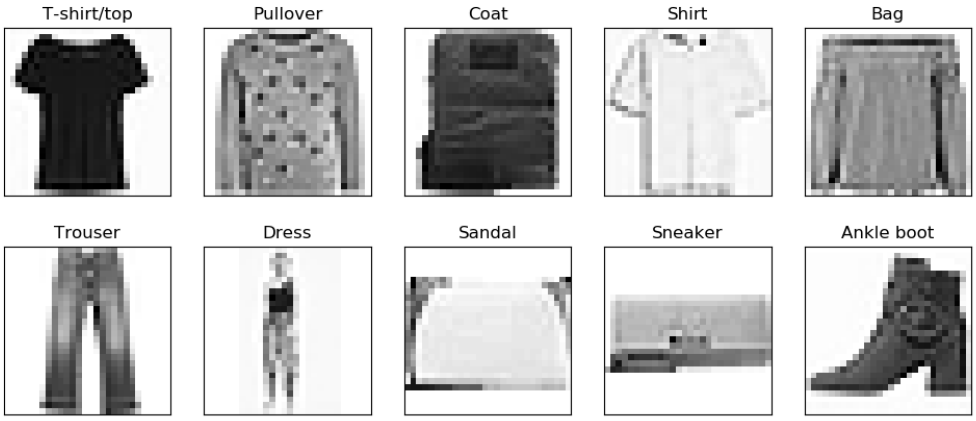


Highest Posterior Probabilities:



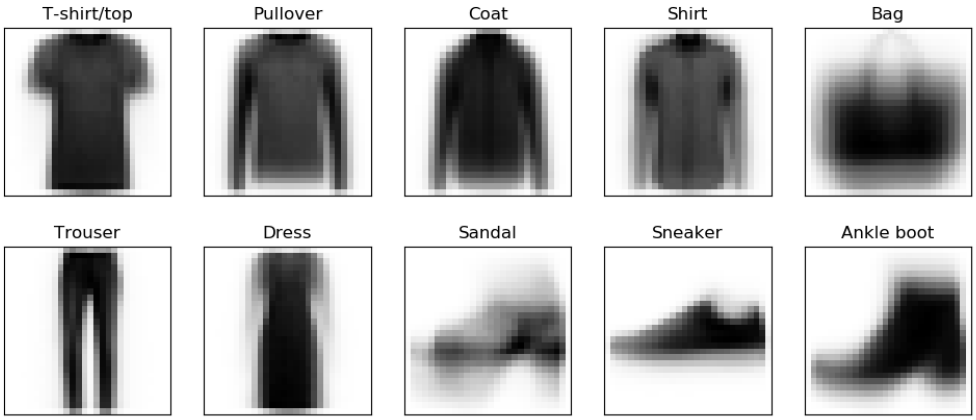
highest log probabilities: [-2129.09086194 -1139.03812197 -2267.58209501 -1250.77229853 -2020.27436206 -819.3329071 -2207.7717257 -1141.78450757 -2003.99323831 -1612.71877855]

Lowest Posterior Probabilities:



lowest log probabilities: [-4062.29499162 -2335.64414529 -4081.80869228 -2948.25022535 -3592.91335072 -3669.03271733 -5010.60099152 -2401.16866074 -4543.87070131 -3954.27894561]

Visualization Plots for Feature Likelihood:



Optimizations

A program was written testing K values (k used in Laplace smoothing) from .1 through 10 to determine which one produced the highest accuracy and it was determined that a K value of .1 gives the highest accuracy.

Part 1.2: Perceptron Model

Results

Overview Screenshot:

```
C:\Users\Abhi Kamboj\ECE-448-Intro-to-AI\MP3 pattern recognition\mp3-code\part1>image_main.py
highest perceptron scores: [34838585.65912683 30045408.56150796 21326682.09801592 30363093.39365072
 36533512.42142881 41763258.75873019 39561245.86150788 29116503.5349206
 42694675.96031752 48832292.84047616]
lowest perceptron scores: [3025541.93293648 3809036.85912699 3851418.87658732 4729209.13650789
 4624707.40793655 2902097.65793652 3150409.59166664 3713386.76071428
 3317798.31071429 3549814.20079362]
classification rates: [0.322 0.933 0.018 0.782 0.8   0.894 0.83  0.942 0.882 0.818]
accuracy: 0.7221
Normalized confusion matrix
[[0.322 0.005 0.003 0.046 0.026 0.    0.587 0.    0.009 0.002]
 [0.    0.933 0.    0.036 0.011 0.    0.02  0.    0.    0.   ]
 [0.    0.001 0.018 0.007 0.292 0.    0.678 0.    0.004 0.   ]
 [0.008 0.011 0.001 0.782 0.087 0.    0.108 0.    0.003 0.   ]
 [0.    0.    0.    0.012 0.8   0.    0.185 0.    0.003 0.   ]
 [0.001 0.001 0.    0.    0.001 0.894 0.005 0.048 0.019 0.031]
 [0.014 0.004 0.005 0.017 0.121 0.    0.83  0.    0.009 0.   ]
 [0.    0.    0.    0.    0.    0.043 0.    0.942 0.006 0.009]
 [0.005 0.002 0.    0.004 0.026 0.011 0.065 0.004 0.882 0.001]
 [0.    0.    0.    0.001 0.    0.065 0.    0.115 0.001 0.818]]

C:\Users\Abhi Kamboj\ECE-448-Intro-to-AI\MP3 pattern recognition\mp3-code\part1>
```

Average Classification Rate:

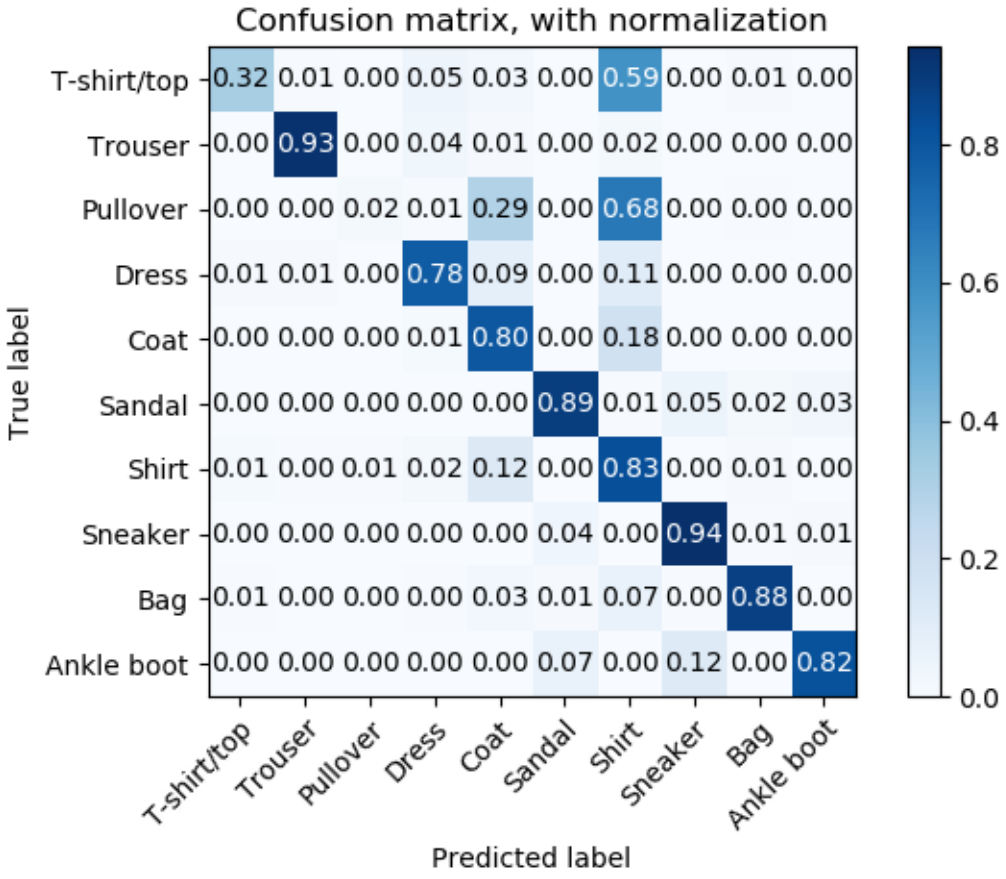
Average Classification Rate: 0.7221

Classification Rate for Each Class:

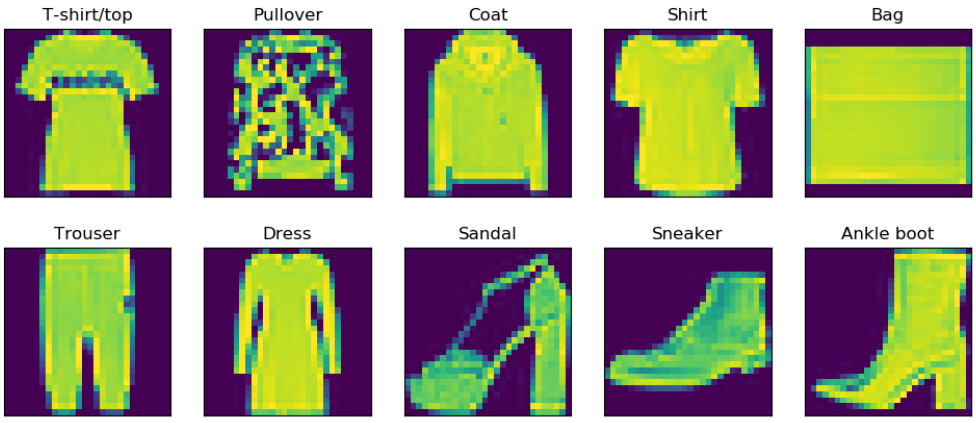
classification rates: [0.322 0.933 0.018 0.782 0.8 0.894 0.83 0.942 0.882 0.818]

Indices in the array shown above correspond to classes

Confusion Matrix:

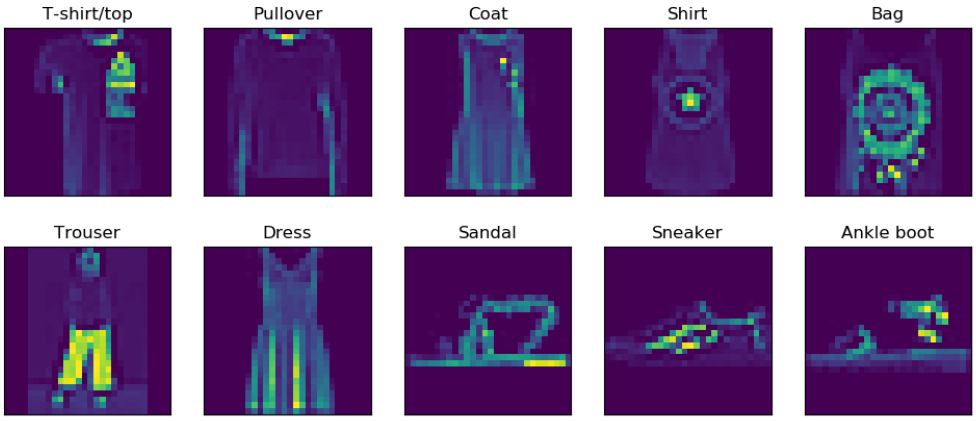


Highest Perceptron Scores:



highest perceptron scores: [34838585.65912683 30045408.56150796 21326682.09801592 30363093.39365072 36533512.42142881 41763258.75873019 39561245.86150788 29116503.5349206 42694675.96031752 48832292.84047616]

Lowest Perceptron Scores:



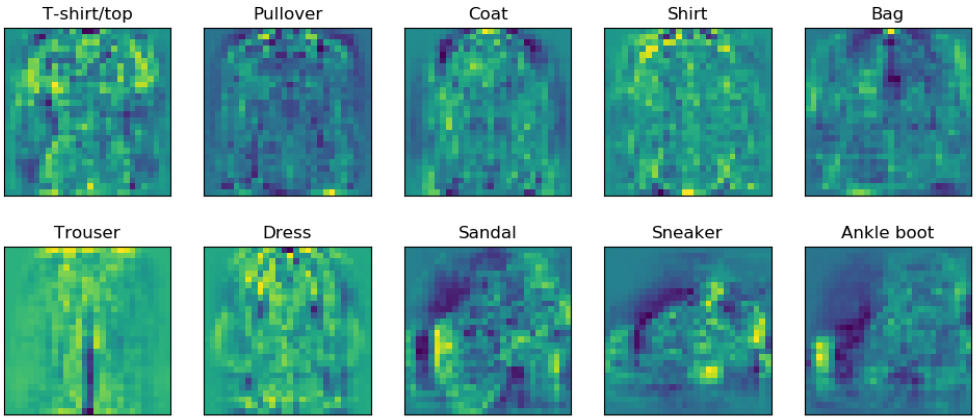
lowest perceptron scores: [3025541.93293648 3809036.85912699 3851418.87658732 4729209.13650789 4624707.40793655 2902097.65793652 3150409.59166664 3713386.76071428 3317798.31071429 3549814.20079362]

Visualization Plots for Perceptron Weights:

After running through data once:

Average classification rate is .6881

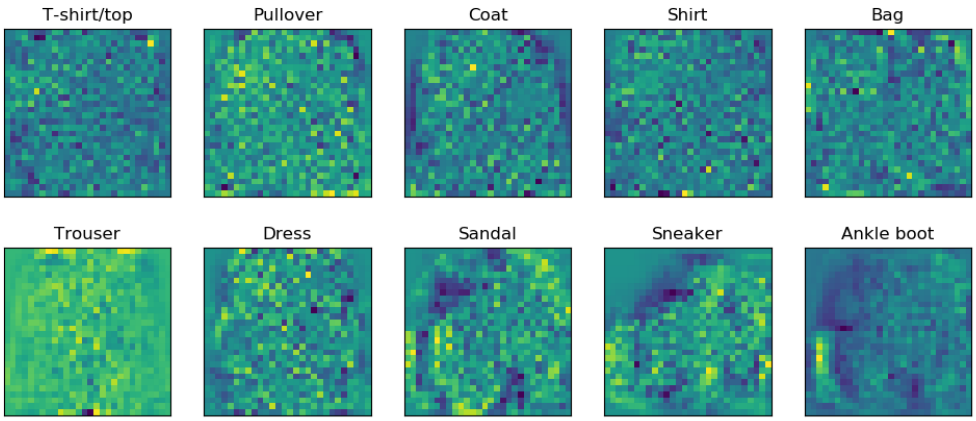
Perceptron weights:



After running through training data 10 times:

Average classification rate is .7221

Perceptron weights:



Optimizations

Initially the perceptron weights were trained by running through the training dataset once and updating the weights accordingly. The perceptron visualizations looked accurate, however the average classification rate was below 70%. The perceptron weights were then trained by running through the data set 10 times each time with a decreasing learning 1/n applied to the weight updates, where n is the trail run the set is on. This increased the accuracy to about 72% however made the perceptron weights a little less visually clear (see Visualization Plots above). This could be because the perceptron weights captured higher dimensional specific features that aren't as visual for us to see.

Section II:

Unigram Model:

Results:

Confusion matrix:

True Label x Predicted Label

Class Labels	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	0	0.02439	0	0	0.04878	0.17073	0.04878	0.04878	0.02439	0.02439	0.04878	0.17073	0.07317	0.31707
2	0	0.8913	0	0	0.02174	0	0.04348	0	0	0	0.02174	0	0	0.02174
3	0	0	0.42857	0.09524	0.04762	0	0.04762	0	0	0	0	0.2381	0.04762	0.09524
4	0	0	0	1	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0.95455	0	0	0	0	0	0	0.04545	0	0
6	0	0	0	0	0.02083	0.77083	0.02083	0.0625	0	0	0	0.08333	0.02083	0.02083
7	0	0.03333	0	0.03333	0	0	0.76667	0.06667	0	0	0	0.06667	0	0.03333
8	0	0	0	0	0	0.05882	0.02941	0.82353	0	0.02941	0	0.05882	0	0

9	0	0	0	0	0	0	0	0.125	0.75	0	0	0.125	0	0
10	0	0	0	0	0	0	0	0.02	0	0.84	0.14	0	0	0
11	0	0	0	0	0	0	0	0	0	0.04444	0.95556	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0.97619	0.02381	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0.97368	0.02632
14	0	0	0	0	0	0	0	0	0	0	0	0.05714	0	0.94286

Accuracy:

Accuracy: 0.7909812877026547

Classification Rate, Recall, Precession, F1:

Class:	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
Classification Rate:	0	0.891304	0.428571	1	0.954545	0.770833	0.766667	0.823529	0.75	0.84	0.955556	0.97619	0.973684	0.942857
Precision:	0	0.953488	1	0.884615	0.807692	0.804348	0.766667	0.756757	0.857143	0.913043	0.811321	0.630769	0.860465	0.634615
Recall:	0	0.891304	0.428571	1	0.954545	0.770833	0.766667	0.823529	0.75	0.84	0.955556	0.97619	0.973684	0.942857
F1 Score:	0	0.921348	0.6	0.938776	0.875	0.787234	0.766667	0.788732	0.8	0.875	0.877551	0.766355	0.91358	0.758621

Top Feature words:

Each entry is: “word”|probability

Class 0		Class 1		Class 2		Class 3		Class 4	
"company"	0.007325	"school"	0.040849	"born"	0.01215	"born"	0.01835	"born"	0.011535
"based"	0.002634	"high"	0.016811	"american	0.006537	"football"	0.011548	"member"	0.010757
"business"	0.002374	"located"	0.009387	"known"	0.005089	"played"	0.010366	"district"	0.0078
"founded"	0.002113	"university	0.006736	"new"	0.004003	"league"	0.009479	"politician	0.007644
"bergen"	0.001853	"college"	0.006383	"band"	0.003098	"footballe	0.005339	"state"	0.005309
"record"	0.001853	"schools"	0.005675	"best"	0.002917	"plays"	0.005339	"senate"	0.004998
"records"	0.001853	"public"	0.005675	"writer"	0.002917	"player"	0.005339	"democrat	0.004998
"services"	0.001592	"students"	0.004261	"rock"	0.002736	"professio	0.005339	"house"	0.004998
"systems"	0.001592	"educatior	0.003908	"work"	0.002554	"former"	0.005043	"party"	0.004842
"products"	0.001332	"county"	0.003378	"musician"	0.002554	"national"	0.0046	"served"	0.004687
"office"	0.001332	"district"	0.003378	"music"	0.002554	"american	0.004156	"former"	0.004531
"college"	0.001071	"new"	0.002671	"singer"	0.002192	"hockey"	0.003565	"county"	0.004064
"distributic	0.001071	"one"	0.002671	"also"	0.002192	"currently"	0.003565	"since"	0.003908
"buses"	0.001071	"founded"	0.002671	"york"	0.002011	"also"	0.003565	"represent	0.003753
"pierce"	0.001071	"independ	0.002494	"author"	0.00183	"rugby"	0.003417	"republica	0.003286
"life"	0.001071	"establishe	0.002494	"books"	0.00183	"team"	0.003269	"elected"	0.00313
"establishe	0.001071	"united"	0.002494	"album"	0.00183	"australiar	0.003121	"united"	0.00313
"including"	0.001071	"city"	0.002494	"member"	0.001649	"novembe	0.002973	"american	0.002974
"national"	0.001071	"part"	0.002494	"series"	0.001649	"world"	0.002825	"represent	0.002819
"located"	0.001071	"private"	0.002317	"united"	0.001649	"new"	0.002678	"national"	0.002819

Class 5		Class 6		Class 7		Class 8		Class 9	
"navy"	0.012553	"historic"	0.017234	"river"	0.025204	"village"	0.023521	"family"	0.03054
"built"	0.010856	"house"	0.01337	"lake"	0.015084	"district"	0.008188	"species"	0.02653
"war"	0.007744	"built"	0.012807	"mountain	0.009809	"populatio	0.005715	"found"	0.015503
"ship"	0.006706	"located"	0.010392	"located"	0.008624	"province"	0.005468	"genus"	0.014956
"uss"	0.006046	"church"	0.010231	"south"	0.006687	"located"	0.005221	"moth"	0.008941
"united"	0.005574	"building"	0.009426	"km"	0.00561	"census"	0.004973	"gastropo	0.00566
"aircraft"	0.00548	"national"	0.009346	"north"	0.005395	"nepal"	0.003984	"sea"	0.005478
"class"	0.00548	"register"	0.007736	"county"	0.004211	"municipal	0.003984	"known"	0.005296
"world"	0.005386	"places"	0.006931	"near"	0.004103	"state"	0.003489	"marine"	0.004931
"states"	0.005197	"listed"	0.005563	"tributary"	0.003995	"india"	0.003489	"described	0.00484
"launched"	0.005103	"street"	0.004999	"range"	0.003888	"county"	0.002995	"tropical"	0.004567
"service"	0.003877	"county"	0.004999	"west"	0.003888	"km"	0.002748	"snail"	0.004202
"designed"	0.003783	"united"	0.004194	"lies"	0.00378	"people"	0.002748	"mollusk"	0.004111
"named"	0.003783	"known"	0.004033	"creek"	0.003672	"within"	0.002253	"endemic"	0.00402
"first"	0.003783	"museum"	0.003872	"crater"	0.003565	"1991"	0.002006	"subtropic	0.003747
"royal"	0.003594	"also"	0.003872	"east"	0.003565	"2010"	0.002006	"habitat"	0.003564
"commissi	0.003311	"states"	0.00355	"state"	0.003457	"southern"	0.001758	"forests"	0.0032
"american	0.003217	"designed"	0.003229	"ft"	0.003457	"kerala"	0.001758	"natural"	0.0032
"ii"	0.003123	"hospital"	0.003148	"flows"	0.003242	"road"	0.001758	"snails"	0.003018
"company"	0.002934	"added"	0.003148	"pass"	0.003134	"time"	0.001758	"moist"	0.002926

Class 10		Class 11		Class 12		Class 13	
"species"	0.029372	"album"	0.050413	"film"	0.05129	"published	0.016849
"family"	0.019659	"released"	0.028552	"directed"	0.020873	"book"	0.013071
"plant"	0.017268	"band"	0.009291	"starring"	0.0082	"novel"	0.012913
"genus"	0.016446	"records"	0.009185	"american	0.007728	"first"	0.008926
"native"	0.0089	"first"	0.007965	"stars"	0.006078	"journal"	0.008401
"endemic"	0.00875	"studio"	0.007646	"released"	0.00596	"written"	0.006932
"flowering	0.008676	"american	0.005683	"written"	0.005547	"series"	0.00562
"known"	0.008451	"songs"	0.005418	"based"	0.005135	"newspape	0.004203
"found"	0.006509	"music"	0.005365	"drama"	0.005017	"story"	0.004098
"common"	0.005612	"second"	0.005259	"comedy"	0.004899	"american	0.004098
"leaves"	0.005164	"release"	0.0051	"also"	0.004781	"author"	0.003521
"plants"	0.005164	"recorded"	0.004834	"produced	0.004781	"new"	0.003101
"habitat"	0.00494	"rock"	0.004728	"films"	0.003366	"magazine	0.002839
"tree"	0.004417	"live"	0.004251	"silent"	0.003072	"fiction"	0.002787
"grows"	0.003894	"debut"	0.004251	"first"	0.003072	"books"	0.002734
"name"	0.003894	"tracks"	0.004091	"movie"	0.003013	"peerrevie	0.002629
"orchid"	0.003669	"label"	0.003932	"roles"	0.002541	"also"	0.002524
"south"	0.003595	"albums"	0.003773	"novel"	0.002482	"publicatic	0.002419
"bulbophyl	0.003296	"new"	0.003667	"name"	0.002482	"science"	0.002419
"perennial	0.003146	"ep"	0.003455	"documen	0.002423	"life"	0.002314

Class Prior Manipulation:

Results without class prior:

Classification Rates: [0.0, 0.9130434782608695, 0.5714285714285714, 1.0, 1.0, 0.7708333333333334, 0.7666666666666667, 0.8235294117647058, 0.75, 0.84, 0.9555555555555556, 0.9285714285714286, 0.9736842105263158, 0.9428571428571428]

Accuracy: 0.8025835570688992

Results with uniform class priors: (Class prior=1/14)

Classification Rates: [0.0, 0.9130434782608695, 0.5714285714285714, 1.0, 1.0, 0.7708333333333334, 0.7666666666666667, 0.8235294117647058, 0.75, 0.84, 0.9555555555555556, 0.9285714285714286, 0.9736842105263158, 0.9428571428571428]

Accuracy: 0.8025835570688992

Without the class priors or with uniform class priors the results increased by about a percent. Priors don't necessarily help the in testing. Priors help alleviate for when one class is seen more often than others in the training set. They don't make a difference in the case that training doesn't depend on how many examples of each set you saw. For example, you could have seen less examples of class 2 than class 3, but maybe class 2 had much more distinctive words than class 3, so it's easier to classify. In this case adding priors might throw it off and make it seem like class 3 is more likely than it should be, leading to more misclassifications.

Extra Credit:

Bigram Model:

```
C:\Users\Abhi Kamboj\ECE-448-Intro-to-AI\MP3 pattern recognition\mp3-code\part2>text_main.py
Confusion Matrix:
0.19512 0.00000 0.04878 0.00000 0.00000 0.00000 0.00000 0.00000 0.73171 0.00000 0.00000 0.00000 0.00000 0.02439
0.15217 0.02174 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.82609 0.00000 0.00000 0.00000 0.00000 0.00000
0.14286 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.85714 0.00000 0.00000 0.00000 0.00000 0.00000
0.34783 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.65217 0.00000 0.00000 0.00000 0.00000 0.00000
0.09091 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.90909 0.00000 0.00000 0.00000 0.00000 0.00000
0.22917 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.77083 0.00000 0.00000 0.00000 0.00000 0.00000
0.13333 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.86667 0.00000 0.00000 0.00000 0.00000 0.00000
0.20588 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.79412 0.00000 0.00000 0.00000 0.00000 0.00000
0.12500 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.87500 0.00000 0.00000 0.00000 0.00000 0.00000
0.18000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.82000 0.00000 0.00000 0.00000 0.00000 0.00000
0.15556 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.84444 0.00000 0.00000 0.00000 0.00000 0.00000
0.23810 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.76190 0.00000 0.00000 0.00000 0.00000 0.00000
0.31579 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.68421 0.00000 0.00000 0.00000 0.00000 0.00000
0.11429 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.85714 0.00000 0.00000 0.00000 0.00000 0.02857
classification rates: [0.1951219512195122, 0.021739130434782608, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.875, 0.0, 0.0, 0.0, 0.0
, 0.02857142857142857]
Precision for all classes : [0.08602150537634409, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.01818181818181818, 0.0, 0.0, 0.0,
 0.0, 0.5]
Recall for all classes: [0.1951219512195122, 0.021739130434782608, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.875, 0.0, 0.0, 0.0, 0
.0, 0.02857142857142857]
F1 Score for all classes: [0.11940298507462689, 0.042553191489361694, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.035623409669211195
, 0.0, 0.0, 0.0, 0.0, 0.05405405405405405]
Accuracy 0.0800

C:\Users\Abhi Kamboj\ECE-448-Intro-to-AI\MP3 pattern recognition\mp3-code\part2>
```

Notice the much lower accuracy of 8% in the fully bigram model. Discussed further in the Results.

Mixture Model:

Lambda	Accuracy		Lambda	Accuracy
0.01	0.790981		0.51	0.082962
0.02	0.55122		0.52	0.081409
0.03	0.355584		0.53	0.081409
0.04	0.257825		0.54	0.081409
0.05	0.209096		0.55	0.081409
0.06	0.181575		0.56	0.081409
0.07	0.163364		0.57	0.081409
0.08	0.147556		0.58	0.081409
0.09	0.144028		0.59	0.081409
0.1	0.13807		0.6	0.081409
0.11	0.134669		0.61	0.081409
0.12	0.134669		0.62	0.081409
0.13	0.131415		0.63	0.081409
0.14	0.121779		0.64	0.081409
0.15	0.120226		0.65	0.081409
0.16	0.120226		0.66	0.081409
0.17	0.112462		0.67	0.078162
0.18	0.106614		0.68	0.078162
0.19	0.106614		0.69	0.078162
0.2	0.103509		0.7	0.076575
0.21	0.103509		0.71	0.076575
0.22	0.103509		0.72	0.076575
0.23	0.103509		0.73	0.076575
0.24	0.098709		0.74	0.076575
0.25	0.098709		0.75	0.076575
0.26	0.095242		0.76	0.076575
0.27	0.095242		0.77	0.076575
0.28	0.092137		0.78	0.076575
0.29	0.092137		0.79	0.076575
0.3	0.089031		0.8	0.076575
0.31	0.089031		0.81	0.076575
0.32	0.089031		0.82	0.076575
0.33	0.089031		0.83	0.076575
0.34	0.089031		0.84	0.076575
0.35	0.089031		0.85	0.076575
0.36	0.089031		0.86	0.076575
0.37	0.089031		0.87	0.076575
0.38	0.089031		0.88	0.076575
0.39	0.08693		0.89	0.076575
0.4	0.085343		0.9	0.076575
0.41	0.085343		0.91	0.076575
0.42	0.085343		0.92	0.078317
0.43	0.085343		0.93	0.078317
0.44	0.085343		0.94	0.078317
0.45	0.085343		0.95	0.080059
0.46	0.085343		0.96	0.078507
0.47	0.085343		0.97	0.080249
0.48	0.085343		0.98	0.083733
0.49	0.082962		0.99	0.076547
0.5	0.082962		1	0.078289

Results

The bigram model decreased the accuracy of the results.

1. Running naive Bayes on the bigram model relaxes the naive assumption of the model a bit. However, is this always a good thing? Why or why not?

   Although it does relax the naïve assumption of the model, occasionally that can be a bad thing like in this case since there were not enough training examples. For the bigram model to be successful, it would probably need a robust data set with more possible combinations of 2-word pairs.

2. What would happen if we did an N-gram model where N was a really large number?

   The model would fail for very large N, because then you would be looking at almost the entire text sample. In this case, you would be overfitting the data to the text sample, not picking up unique features for the example to work.