COMP 551

ASSIGNMENT 3

AHMET AKGUL – 260 624 068

2. Yelp Dataset

Random Classifier

Training F1-Measure: 0.180419612228
Confusion Matrix:
```
[[104 102 114  97 105]
 [115 134 136 124 132]
 [225 197 183 195 197]
 [490 496 481 487 514]
 [459 486 461 479 487]]
```

Validation F1-Measure: 0.169737750897
Confusion Matrix:
```
[[15 16 15 22 16]
 [12 21 22 21 20]
 [47 33 24 28 32]
 [71 74 78 67 66]
 [69 72 47 55 57]]
```

Test F1-Measure: 0.186163692511
Confusion Matrix:
```
[[ 38  31  28  25  21]
 [ 41  36  34  34  45]
 [ 60  48  58  76  58]
 [148 157 134 124 139]
 [119 149 124 125 148]]
```

Most Frequent Classifier Score

Training F1-Measure: 0.104267004647
Confusion Matrix:
```
[[   0    0    0  522    0]
 [   0    0    0  641    0]
 [   0    0    0  997    0]
 [   0    0    0 2468    0]
 [   0    0    0 2372    0]]
```

Validation F1-Measure: 0.105014749263
Confusion Matrix:
```
[[  0   0   0  84   0]
 [  0   0   0  96   0]
 [  0   0   0 164   0]
 [  0   0   0 356   0]
 [  0   0   0 300   0]]
```

Test F1-Measure: 0.103923019985
Confusion Matrix:
```
[[  0   0   0 143   0]
 [  0   0   0 190   0]
 [  0   0   0 300   0]
 [  0   0   0 702   0]
 [  0   0   0 665   0]]
```

Bernoulli Naive Bayes

Training F1-Measure: 0.577269725617
Confusion Matrix:
```
 [[ 275    8   15   22  202]
 [  10  323   23   85  200]
 [   9   25  471  153  339]
 [  36  155   86 1211  980]
 [  53  124   73  248 1874]]
```

Validation F1-Measure: 0.330091223067
Confusion Matrix:
```
 [[ 29   9   5   7  34]
 [  9  16  12  26  33]
 [  2  23  28  54  57]
 [  8  33  27 123 165]
 [ 13  25   9  69 184]]
```

Test F1-Measure: 0.341137874213
Confusion Matrix:
```
 [[ 42  24   5  18  54]
 [ 22  46  32  31  59]
 [  9  36  44  99 112]
 [ 10  58  61 246 327]
 [ 10  37  27 150 441]]
```

Decision Tree

Training F1-Measure: 1.0
Confusion Matrix:
```
[[ 522    0    0    0    0]
 [   0  641    0    0    0]
 [   0    0  997    0    0]
 [   0    0    0 2468    0]
 [   0    0    0    0 2372]]
```

Validation F1-Measure: 0.261495450896
Confusion Matrix:
```
[[ 16  15  15  14  24]
 [ 14  10  24  22  26]
 [ 12  18  40  57  37]
 [ 19  35  56 135 111]
 [ 20  15  33 115 117]]
```

Test F1-Measure: 0.279169541182
Confusion Matrix:
```
[[ 29  21  31  34  28]
 [ 28  23  45  59  35]
 [ 17  30  59 127  67]
 [ 35  39  97 298 233]
 [ 22  38  60 251 294]]
```

Linear SVC

Training F1-Measure: 0.997870801179
Confusion Matrix:
```
 [[ 520    0    0    0    2]
 [   0  640    0    0    1]
 [   0    0  996    1    0]
 [   0    0    0 2454   14]
 [   0    0    0    2 2370]]
```

Validation F1-Measure: 0.413976371409
Confusion Matrix:
```
 [[ 36  23   8   9   8]
 [ 14  25  16  27  14]
 [  7  23  50  65  19]
 [ 10  11  31 171 133]
 [  4   9  18 102 167]]
```

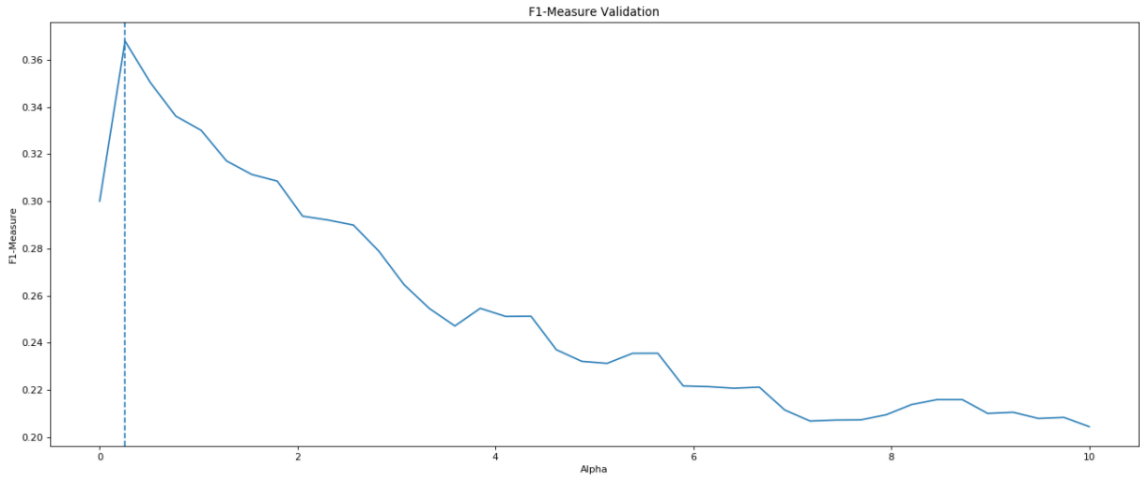
Test F1-Measure: 0.400687403274
Confusion Matrix:
```
 [[ 55  33  15  20  20]
 [ 33  58  41  36  22]
 [ 12  42  77 118  51]
 [ 12  31  83 341 235]
 [ 15  12  41 236 361]]
```

Yelp

Bernoulli Naive Bayes Hyper-Parameter Tuning (Bag of Words)

For the Bernoulli Naïve Bayes training model, there is only one parameter to tune, alpha. This alpha is the additive smoothing parameter. A value of 0 means no smoothing. We will first vary this between 0 and 20.

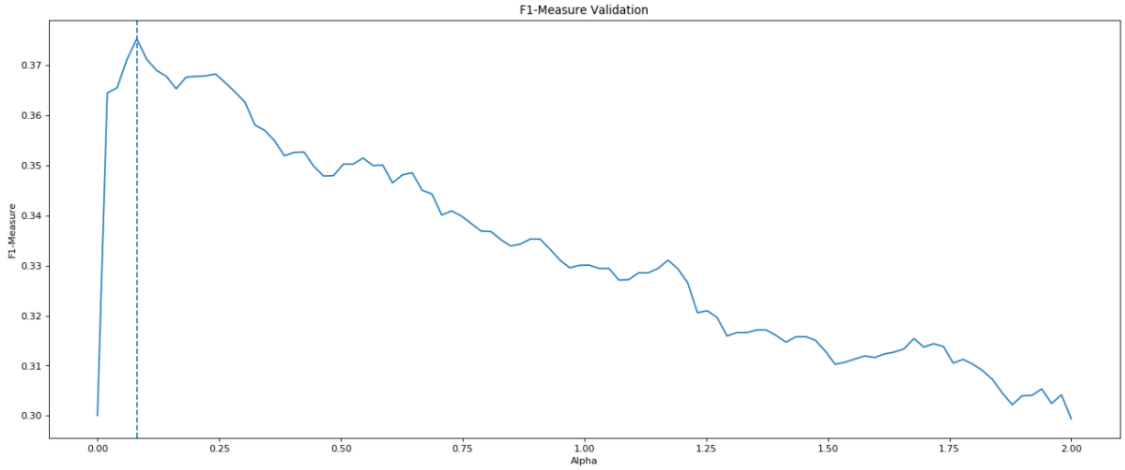
Iteration 1



MAX Alpha 0.256411230769
MAX F1-Measure 0.367900535598

The higher F1-measures occur on the validation set for alpha values between 0 and 2. Now we update the iteration loop and re-run the program for values of alpha between 0 and 2.

Iteration 2

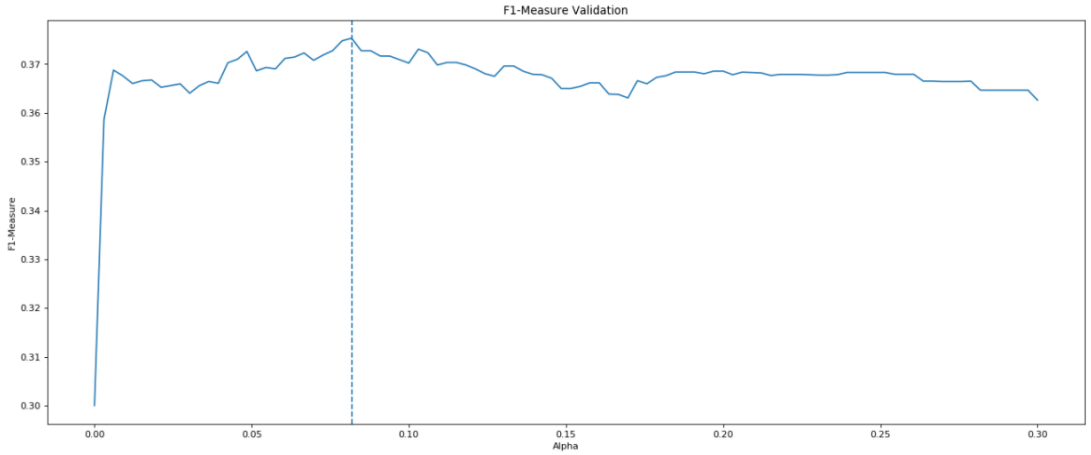



```
Best Alpha for F1-Measure  0.080809040404
MAX F1-Measure  0.375289839258
F1 Score for Testing at Alpha = 0.08080904040404041 is 0.38725580913826974
```

Testing F1-Measure: 0.3873

The testing F1-measure at alpha = 0.081 is 0.3873. We run through once more for values of alpha between 0 and 0.3.

Iteration 3




```
Best Alpha for F1-Measure  0.0818189090909
MAX F1-Measure  0.375289839258
F1 Score for Testing at Alpha = 0.0818189090909091 is 0.38725580913826974
```

Testing F1-Measure: 0.3872

At the value of alpha = 0.0818, the validation F1-measure is the highest—0.3753. At this value, the testing F1-Measure is 0.3872. This is the best tuning we can do for this model given the yelp dataset and a Bag of Words representation.

```
{'alpha': 0.0818, 'binarize': 0.0, 'class_prior': None, 'fit_prior':
True}

Training F1-Measure: 0.722816991255
Confusion Matrix:
 [[ 380    3    4    4  131]
 [  15  410   24   36  156]
 [   6   11  660   62  258]
 [  25   45   46 1489  863]
 [  24   50   55  238 2005]]

Validation F1-Measure: 0.375289839258
Confusion Matrix:
 [[ 36  12   5   4  27]
 [ 12  19  18  20  27]
 [  5  17  41  48  53]
 [  8  24  27 136 161]
 [ 11  14  12  84 179]]

Test F1-Measure: 0.387255809138
Confusion Matrix:
 [[ 59  23  14  10  37]
 [ 30  47  38  27  48]
 [ 12  38  61  86 103]
 [  8  33  68 283 310]
 [ 10  18  30 179 428]]
```

Yelp

Linear SVC Hyper-Parameter Tuning (Bag of Words)

For Linear SVC, there were more parameters to consider when tuning the model. These parameters were C, the penalty parameter of the error term, Dual vs Primal, which specifies which optimization problem to solve, loss, which specifies the loss function, max iterations, which are the maximum number of iterations the model will run while training the data, and tol, which is the tolerance for stopping criteria.

For C, I considered values from the array [0.01, 0.1, 1.0, 10, 100, 1000].

For tolerance, I considered values from the array [1e-6, 1e-5, 1e-4, 1e-3,1e-2,1e-1, 1, 10]

Because in this dataset the number of training samples (7000) is less than the number of features (10,000), I am setting Dual to true and solving the dual problem.

For maximum iterations I am considering values between the range 1000 and 10,000. The default value was 1000, this will essentially show whether increasing the iterations allows for accuracy improvements.

For loss, I am considering squared-hinge and hinged loss.

Iteration 1

```
Model with rank: 1
Mean validation score: 0.486 (std: 0.003)
Parameters: {'C': 0.1, 'dual': True, 'loss': 'squared_hinge', 'max_i
ter': 2466, 'tol': 0.001}

Model with rank: 2
Mean validation score: 0.485 (std: 0.006)
Parameters: {'C': 0.1, 'dual': True, 'loss': 'hinge', 'max_iter': 57
09, 'tol': 0.001}

Model with rank: 3
Mean validation score: 0.456 (std: 0.002)
Parameters: {'C': 1.0, 'dual': True, 'loss': 'squared_hinge', 'max_i
ter': 9370, 'tol': 0.01}
```

This first iteration shows that the model prefers iterations that are above 1000. For the next iter ation I will decrease the range. The model also seems to prefer lower tolerance values, hence w e can remove some of the larger tolerance values, such as 1 and 10.
The best F1-Measure for validation is 0.486.

Iteration 2

```
Model with rank: 1
Mean validation score: 0.514 (std: 0.005)
Parameters: {'C': 0.01, 'dual': True, 'loss': 'squared_hinge', 'max_
iter': 4710, 'tol': 0.001}

Model with rank: 1
Mean validation score: 0.514 (std: 0.005)
Parameters: {'C': 0.01, 'dual': True, 'loss': 'squared_hinge', 'max_
iter': 4066, 'tol': 0.0001}

Model with rank: 1
Mean validation score: 0.514 (std: 0.005)
Parameters: {'C': 0.01, 'dual': True, 'loss': 'squared_hinge', 'max_
iter': 9689, 'tol': 0.001}
```

This iteration shows that the model generally prefers Squared Hinge loss. Hence, I will only cons ider that loss for the next iteration. Again, the model prefers lower tolerance values, so I will de crease the array some more.
This iteration resulted in a validation F1-Measure of 0.514.

Iteration 3

```
Model with rank: 1
Mean validation score: 0.514 (std: 0.005)
Parameters: {'C': 0.01, 'max_iter': 7449, 'tol': 1e-06}

Model with rank: 1
Mean validation score: 0.514 (std: 0.005)
Parameters: {'C': 0.01, 'max_iter': 6358, 'tol': 0.0001}
```

```
Model with rank: 1
Mean validation score: 0.514 (std: 0.005)
Parameters: {'C': 0.01, 'max_iter': 3775, 'tol': 0.0001}

{'C': 0.01, 'class_weight': None, 'dual': True, 'fit_intercept': Tru
e, 'intercept_scaling': 1, 'loss': 'squared_hinge', 'max_iter': 7449
, 'multi_class': 'ovr', 'penalty': 'l2', 'random_state': None, 'tol'
: 1e-06, 'verbose': 0}

Training F1-Measure: 0.85867638506
Confusion Matrix:
 [[ 458    2    7   18   37]
 [  18  509   30   47   37]
 [   8   10  743  152   84]
 [   4    0   17 2084  363]
 [   1    2    9  213 2147]]

Validation F1-Measure: 0.467650693044
Confusion Matrix:
 [[ 41  18   7   9   9]
 [ 14  23  24  22  13]
 [  0  13  50  81  20]
 [  4   7  27 194 124]
 [  0   2   5  93 200]]

Test F1-Measure: 0.456741680687
Confusion Matrix:
 [[ 74  27   9  16  17]
 [ 33  50  46  39  22]
 [ 12  26  77 138  47]
 [  9   9  57 375 252]
 [  7   2  11 208 437]]
```

Yelp

Decision Tree Hyper Parameter Tuning (Bag of Words)

There are many parameters to tune for the decision tree. Not all these values were considered. The first parameter considered is max depth, which is the maximum depth of the tree. The default for this is infinity. I first considered values between 0 and 50.

The second parameter I considered is max features, which is the maximum number of features to consider when looking for the best split. The default for this is the number of features in the model, which is 10,000. I considered the options None, which considers n_features, 'sqrt', which considers sqrt(n_features), and 'log2', which considers log2(n_features).

The third parameter I considered is max leaf nodes, which defines the maximum number of leaf nodes the model will have. The default for this is an unlimited number of leaf nodes. I first considered between 0 and 8,000 leaf nodes. This is because 8000 is essentially the maximum

number of leaf nodes I could get with the given yelp training and validation data (used K-folds cross validation).

The fourth parameter I considered is min impurity decrease. This decides whether a node will be split. The split happens if the decrease in impurity is greater than or equal to this value. The default for this parameter is 0. I considered values between 0 and 0.5.

The fifth parameter I considered is min samples leaf. This is the minimum number of samples required to be at a leaf node. The default is 1. I considered values between 1 and 20.

The sixth and final parameter I considered is min samples split. This is the minimum number of samples required to split an internal node. The default is 2. I considered values between 2 and 20.

Iteration 1

```
Model with rank: 1
Mean validation score: 0.308 (std: 0.002)
Parameters: {'max_depth': 23, 'max_features': None, 'max_leaf_nodes'
: 10000, 'min_impurity_decrease': 0, 'min_samples_leaf': 4, 'min_sam
ples_split': 6}

Model with rank: 2
Mean validation score: 0.303 (std: 0.007)
Parameters: {'max_depth': 23, 'max_features': None, 'max_leaf_nodes'
: None, 'min_impurity_decrease': 0, 'min_samples_leaf': 5, 'min_samp
les_split': 8}

Model with rank: 3
Mean validation score: 0.301 (std: 0.001)
Parameters: {'max_depth': 26, 'max_features': None, 'max_leaf_nodes'
: None, 'min_impurity_decrease': 0, 'min_samples_leaf': 1, 'min_samp
les_split': 9}
```

The best validation F1-measure is 0.308. It is clear the maximum leaf nodes is optimum between 20 and 30. For the next iteration, I want to introduce min impurity decrease as a uniform variable between 0 and 0.5.

Iteration 2

```
Model with rank: 1
Mean validation score: 0.239 (std: 0.014)
Parameters: {'max_depth': 32, 'max_features': None, 'max_leaf_nodes'
: 1000, 'min_impurity_decrease': 0.0017847551550062902, 'min_samples
_leaf': 8, 'min_samples_split': 4}

Model with rank: 2
Mean validation score: 0.223 (std: 0.026)
Parameters: {'max_depth': 17, 'max_features': None, 'max_leaf_nodes'
: 15000, 'min_impurity_decrease': 0.0021257207973072734, 'min_sample
s_leaf': 1, 'min_samples_split': 2}

Model with rank: 3
```

```
Mean validation score: 0.191 (std: 0.024)
Parameters: {'max_depth': 10, 'max_features': 'sqrt', 'max_leaf_node
s': 10000, 'min_impurity_decrease': 0.00048700757764565952, 'min_sam
ples_leaf': 9, 'min_samples_split': 7}
```

Minimum impurity should stay at as close as possible to 0. The best validation F1-measure decreased to 0.239.

Iteration 3

```
Model with rank: 1
Mean validation score: 0.313 (std: 0.005)
Parameters: {'max_depth': 24, 'max_features': None, 'max_leaf_nodes'
: None, 'min_impurity_decrease': 0.000438451445691912, 'min_samples_
leaf': 6, 'min_samples_split': 2}

Model with rank: 2
Mean validation score: 0.305 (std: 0.004)
Parameters: {'max_depth': 25, 'max_features': None, 'max_leaf_nodes'
: 15000, 'min_impurity_decrease': 0.00024863477331978133, 'min_sampl
es_leaf': 3, 'min_samples_split': 9}

Model with rank: 3
Mean validation score: 0.301 (std: 0.008)
Parameters: {'max_depth': 39, 'max_features': None, 'max_leaf_nodes'
: None, 'min_impurity_decrease': 9.8586025656947074e-05, 'min_sample
s_leaf': 6, 'min_samples_split': 5}
```

The best validation F1-measure is 0.313. For the next iteration I will increase the range for min samples split between 2 and 200.

Iteration 4

```
Model with rank: 1
Mean validation score: 0.320 (std: 0.010)
Parameters: {'max_depth': 25, 'max_features': None, 'max_leaf_nodes'
: 26406, 'min_impurity_decrease': 0.00051830527754645494, 'min_sampl
es_leaf': 3, 'min_samples_split': 2}

Model with rank: 2
Mean validation score: 0.320 (std: 0.001)
Parameters: {'max_depth': 28, 'max_features': None, 'max_leaf_nodes'
: 61954, 'min_impurity_decrease': 0.00052115656459021263, 'min_sampl
es_leaf': 7, 'min_samples_split': 16}

Model with rank: 3
Mean validation score: 0.319 (std: 0.001)
Parameters: {'max_depth': 28, 'max_features': None, 'max_leaf_nodes'
: 117155, 'min_impurity_decrease': 0.00054116549571744141, 'min_samp
les_leaf': 7, 'min_samples_split': 9}
```

The best validation F1-measure is 0.32. It occurs at a depth of 25, 10000 features, 26406 max_leaf_nodes, 0.000518 minimum impurity decrease, 3 minimum samples per leaf, and 2 minimum samples per split.

{'class_weight': None, 'criterion': 'gini', 'max_depth': 25, 'max_features': None, 'max_leaf_nodes': 26406, 'min_impurity_decrease': 0.000518305277546454 9, 'min_impurity_split': None, 'min_samples_leaf': 3, 'min_samples_split': 2, 'min_weight_fraction_leaf': 0.0, 'presort': False, 'random_state': None, 'splitter': 'best'}

Training F1-Measure: 0.456494980587
Confusion Matrix:

```
[[ 220   44   11  169   78]
 [  68  131   60  292   90]
 [  42   35  243  560  117]
 [  62   39   71 1889  407]
 [  56   16   47  936 1317]]
```

Validation F1-Measure: 0.309005474262
Confusion Matrix:

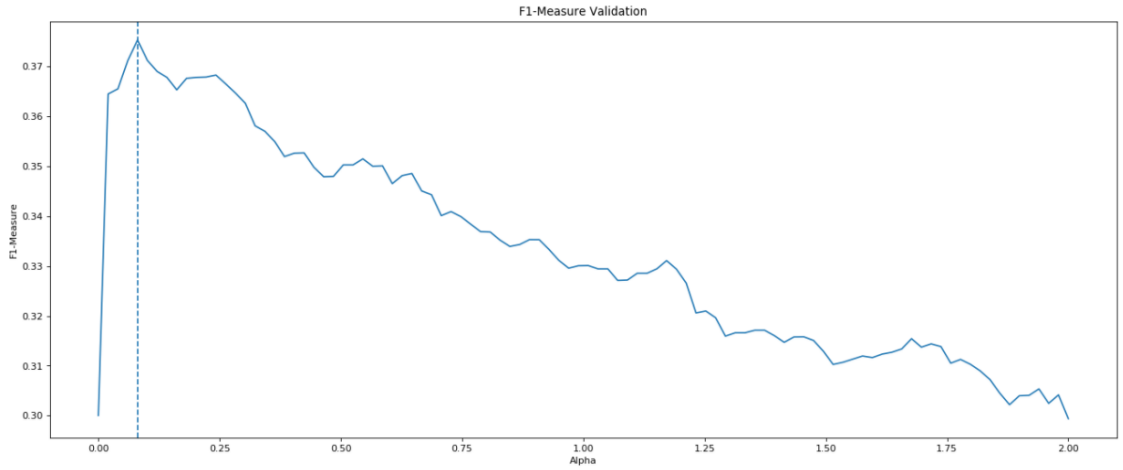
```
[[ 26  14   6  26  12]
 [ 10   8  11  46  21]
 [  9  13  23  95  24]
 [  9  10  25 221  91]
 [  8   5   8 167 112]]
```

Test F1-Measure: 0.299015102134
Confusion Matrix:

```
[[ 41  17   9  57  19]
 [ 26  19  26  86  33]
 [ 14  16  29 195  46]
 [ 21  11  39 438 193]
 [ 19   7  23 358 258]]
```

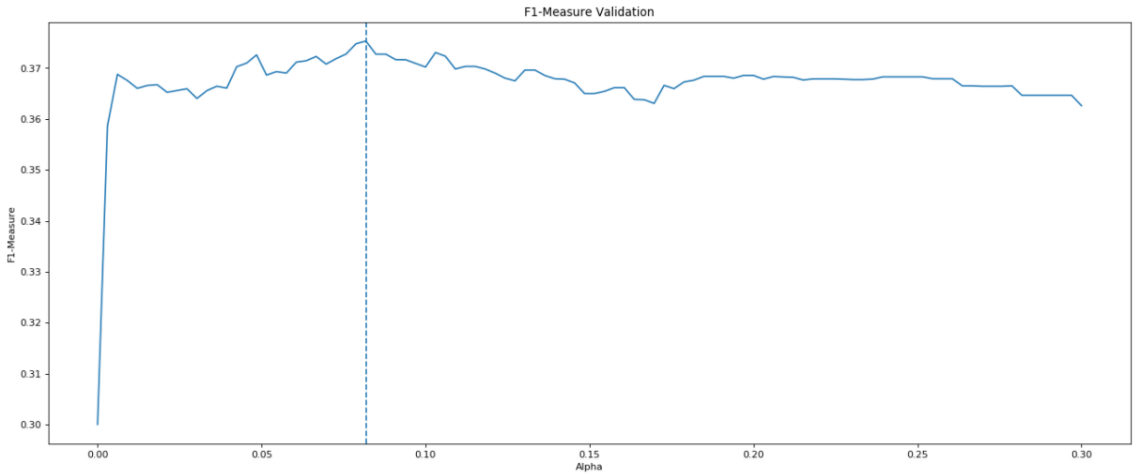
Yelp

Gaussian Naïve-Bayes Hyper-Parameter Tuning (Frequency)




```
Best Alpha for F1-Measure  0.080809040404
MAX F1-Measure  0.375289839258
F1 Score for Testing at Alpha = 0.08080904040404041 is 0.38725580913826974
```

Best F1-measure occurs between 0 and 0.25. Re-iterate to find best value




```
Best Alpha for F1-Measure  0.0818189090909
MAX F1-Measure  0.375289839258
F1 Score for Testing at Alpha = 0.08181890909090911 is 0.38725580913826974
```

Best F1-measure occurs at alpha = 0.0818 and the testing F1-measure is 0.387.

Note that the best F1-measure on the validation set occurs at the same alpha value, and the testing F1-measure at this point is only slightly better.

```
{'alpha': 0.0818, 'binarize': 0.0, 'class_prior': None, 'fit_prior':
True}

Training F1-Measure: 0.722816991255
Confusion Matrix:
 [[ 380    3    4    4  131]
 [  15  410   24   36  156]
 [   6   11  660   62  258]
 [  25   45   46 1489  863]
 [  24   50   55  238 2005]]

Validation F1-Measure: 0.375289839258
Confusion Matrix:
 [[ 36  12   5   4  27]
 [ 12  19  18  20  27]
 [  5  17  41  48  53]
 [  8  24  27 136 161]
 [ 11  14  12  84 179]]

Test F1-Measure: 0.387255809138
Confusion Matrix:
 [[ 59  23  14  10  37]
 [ 30  47  38  27  48]
 [ 12  38  61  86 103]
 [  8  33  68 283 310]
 [ 10  18  30 179 428]]
```

Yelp

Linear SVC Hyper-Parameter Tuning (Frequency)

```
Model with rank: 1
Mean validation score: 0.447 (std: 0.008)
Parameters: {'C': 100, 'max_iter': 2450, 'tol': 1e-06}

Model with rank: 2
Mean validation score: 0.447 (std: 0.006)
Parameters: {'C': 100, 'max_iter': 9022, 'tol': 0.01}

Model with rank: 3
Mean validation score: 0.446 (std: 0.007)
Parameters: {'C': 100, 'max_iter': 2290, 'tol': 0.01}
```

The frequency representation on the Yelp dataset did not show any improvements. Note, this could be because the random classifier did not run through the same number of iterations. In Bag of Words representation, the model ran through 3 times as many iterations, as I had more time to train. Now that training is taking long, and the assignment is due very soon, I am unable to train the model for as long as I would like.

```
{'C': 100, 'class_weight': None, 'dual': True, 'fit_intercept': True
, 'intercept_scaling': 1, 'loss': 'squared_hinge', 'max_iter': 2450,
'multi_class': 'ovr', 'penalty': 'l2', 'random_state': None, 'tol':
1e-06, 'verbose': 0}
```

Training F1-Measure: 0.891559550272
Confusion Matrix:

```
 [[ 493    8    5   12    4]
 [  18  545   16   53    9]
 [   6   16  784  158   33]
 [   3    1   25 2227  212]
 [   1    1    8  235 2127]]
```

Validation F1-Measure: 0.45823811089
Confusion Matrix:

```
 [[ 43  19   5   9   8]
 [ 18  23  19  25  11]
 [  4  14  48  82  16]
 [  6   5  30 195 120]
 [  5   2  10  88 195]]
```

Test F1-Measure: 0.453660238328
Confusion Matrix:

```
 [[ 77  19   8  22  17]
 [ 32  50  47  41  20]
 [ 17  30  74 134  45]
 [  8  15  63 383 233]
 [  2   9  21 216 417]]
```

Yelp

Decision Tree Hyper-Parameter Tuning (Frequency)

```
Model with rank: 1
Mean validation score: 0.312 (std: 0.013)
Parameters: {'max_depth': 25, 'max_features': None, 'max_leaf_nodes'
: 116, 'min_impurity_decrease': 0.00034614266116079795, 'min_samples
_leaf': 2, 'min_samples_split': 195}

Model with rank: 2
Mean validation score: 0.310 (std: 0.020)
Parameters: {'max_depth': 20, 'max_features': None, 'max_leaf_nodes'
: 927, 'min_impurity_decrease': 0.00033034519106199799, 'min_samples
_leaf': 2, 'min_samples_split': 75}

Model with rank: 3
Mean validation score: 0.310 (std: 0.017)
Parameters: {'max_depth': 20, 'max_features': None, 'max_leaf_nodes'
: 360, 'min_impurity_decrease': 0.00039506476060957921, 'min_samples
_leaf': 1, 'min_samples_split': 95}
```

The Decision tree model for the Frequency representation of the Yelp dataset does slightly wors e that the Bag of Words representation. Note: The difference is only 0.01 F-measure for the vali dation set.

```
{'class_weight': None, 'criterion': 'gini', 'max_depth': 25, 'max_fe
atures': None, 'max_leaf_nodes': 116, 'min_impurity_decrease': 0.000
34614266116079795, 'min_impurity_split': None, 'min_samples_leaf': 2
, 'min_samples_split': 195, 'min_weight_fraction_leaf': 0.0, 'presor
t': False, 'random_state': None, 'splitter': 'best'}
```

Training F1-Measure: 0.451092729318
Confusion Matrix:

```
 [[ 203   53   33  112  121]
 [  74  120   84  193  170]
 [  40   61  279  351  266]
 [  49   41  126 1501  751]
 [  59   27   63  511 1712]]
```

Validation F1-Measure: 0.309577715889
Confusion Matrix:

```
 [[ 26  12   3  26  17]
 [ 12  11  18  29  26]
 [  6  10  30  66  52]
 [ 14  10  32 144 156]
 [  9  10   8 121 152]]
```

Test F1-Measure: 0.302908462128
Confusion Matrix:

```
 [[ 33  16  10  40  44]
 [ 25   9  39  76  41]
 [ 16  12  52 135  85]
 [ 17  21  61 326 277]
 [ 17  10  19 218 401]]
```

Comments:

For the Yelp dataset, the best classifier is clearly the Linear SVC. For this, the F1-measure was above 0.5. This occurred with the Bag of Words representation. The other two classifiers performed much worse, around 0.3 F-measure. However, looking at the confusion matrix, it is easy to see why this would happen. There are a lot of numbers along the diagonal, and squares close to the diagonal, but when the problem is 5 class it is easy to have lots of false positive and false negatives.

3. IMDB

Random Classifier

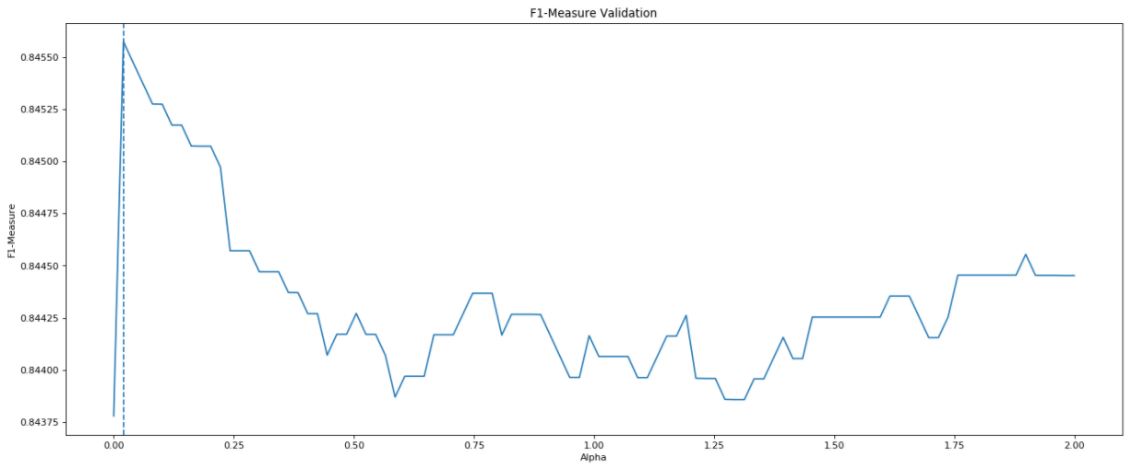
Training F1-Measure: 0.505866666667
Confusion Matrix:
[[3794 3706]
[3706 3794]]

Validation F1-Measure: 0.49864043347
Confusion Matrix:
[[2439 2561]
[2452 2548]]

Test F1-Measure: 0.500239996802
Confusion Matrix:
[[6254 6246]
[6248 6252]]

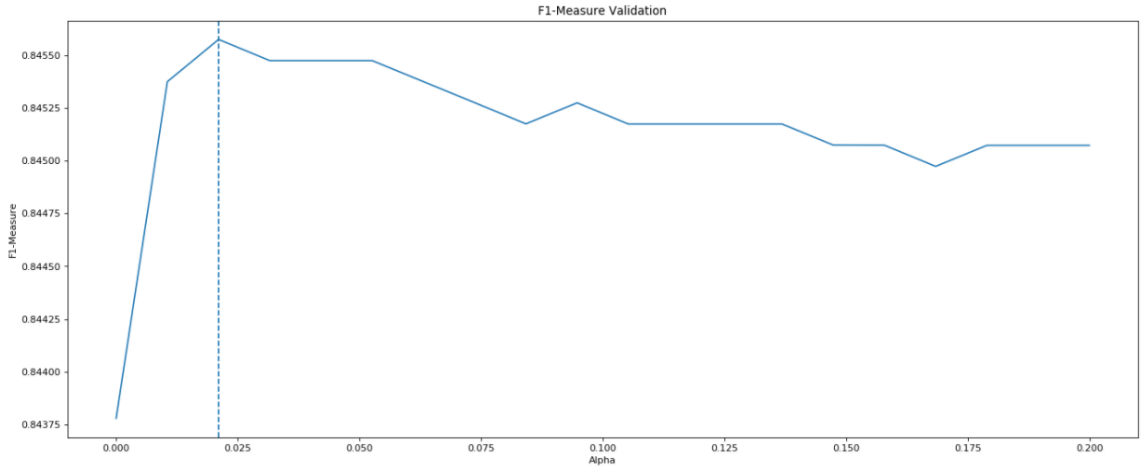
IMDB

Bernoulli Naïve Bayes Hyper-Parameter Tuning (Bag of Words)



Best Alpha for F1-Measure 0.020203010101
MAX F1-Measure 0.845574698959
F1 Score for Testing at Alpha = 0.020203010101010103 is 0.834172342808849

The best F1-Measures occur between 0 and 0.2 for values of alpha. Re-iterating after updating loop:




```
Best Alpha for F1-Measure  0.0210535263158
MAX F1-Measure  0.845574698959
F1 Score for Testing at Alpha = 0.021053526315789477 is 0.834172342808849
```

The best validation F1-measure occurs at alpha = 0.02105, and the testing F1-measure at this value is 0.8342.

```
{'alpha': 0.02105, 'binarize': 0.0, 'class_prior': None, 'fit_prior'
: True}

Training F1-Measure: 0.873838532099
Confusion Matrix:
 [[6666  834]
 [1058 6442]]

Validation F1-Measure: 0.845574698959
Confusion Matrix:
 [[4292  708]
 [ 836 4164]]

Test F1-Measure: 0.834172342809
Confusion Matrix:
 [[10802  1698]
 [ 2444 10056]]
```

IMDB

Linear SVC Hyper-Parameter Tuning (Bag of Words)

```
Model with rank: 1
Mean validation score: 0.883 (std: 0.004)
```

```
Parameters: {'C': 0.01, 'dual': False, 'max_iter': 4056, 'tol': 0.01
}

Model with rank: 1
Mean validation score: 0.883 (std: 0.004)
Parameters: {'C': 0.01, 'dual': False, 'max_iter': 8331, 'tol': 0.01
}

Model with rank: 3
Mean validation score: 0.868 (std: 0.006)
Parameters: {'C': 0.1, 'dual': False, 'max_iter': 8678, 'tol': 0.1}


{'C': 0.01, 'class_weight': None, 'dual': False, 'fit_intercept': Tr
ue, 'intercept_scaling': 1, 'loss': 'squared_hinge', 'max_iter': 405
6, 'multi_class': 'ovr', 'penalty': 'l2', 'random_state': None, 'tol
': 0.01, 'verbose': 0}
```

Training F1-Measure: 0.965333229179
Confusion Matrix:
```
 [[7227  273]
 [ 247 7253]]
```

Validation F1-Measure: 0.881093663481
Confusion Matrix:
```
 [[4369  631]
 [ 558 4442]]
```

Test F1-Measure: 0.874518263534
Confusion Matrix:
```
 [[10885  1615]
 [ 1522 10978]]
```

IMDB

Decision Tree Hyper-Parameter Tuning (Bag of Words)

```
Model with rank: 1
Mean validation score: 0.743 (std: 0.006)
Parameters: {'max_depth': 27, 'max_features': None, 'max_leaf_nodes'
: 1053, 'min_impurity_decrease': 0.00026333022487043378, 'min_sample
s_leaf': 17, 'min_samples_split': 169}

Model with rank: 2
Mean validation score: 0.741 (std: 0.005)
Parameters: {'max_depth': 22, 'max_features': None, 'max_leaf_nodes'
: 1802, 'min_impurity_decrease': 0.00041782087619818666, 'min_sample
s_leaf': 13, 'min_samples_split': 47}

Model with rank: 3
Mean validation score: 0.740 (std: 0.004)
```

```
Parameters: {'max_depth': 21, 'max_features': None, 'max_leaf_nodes'
: 1538, 'min_impurity_decrease': 0.0002483325133484494, 'min_samples
_leaf': 16, 'min_samples_split': 109}


{'class_weight': None, 'criterion': 'gini', 'max_depth': 27, 'max_fe
atures': None, 'max_leaf_nodes': 1053, 'min_impurity_decrease': 0.00
02633302248704338, 'min_impurity_split': None, 'min_samples_leaf': 1
7, 'min_samples_split': 169, 'min_weight_fraction_leaf': 0.0, 'preso
rt': False, 'random_state': None, 'splitter': 'best'}
```

Training F1-Measure: 0.784689544034
Confusion Matrix:
[[5548 1952]
[1271 6229]]

Validation F1-Measure: 0.746888924834
Confusion Matrix:
[[3535 1465]
[1062 3938]]

Test F1-Measure: 0.749895462554
Confusion Matrix:
[[8864 3636]
[2606 9894]]

IMDB

Gaussian Naïve Bayes Hyper-Parameter Tuning (Frequency)

Best Alpha for F1-Measure 0.025641974359
MAX F1-Measure 0.845474285376
F1 Score for Testing at Alpha = 0.025641974358974362 is 0.8341731341759987

```
{'alpha': 0.0256419, 'binarize': 0.0, 'class_prior': None, 'fit_prio
r': True}

Training F1-Measure: 0.873838532099
Confusion Matrix:
 [[6666  834]
 [1058 6442]]

Validation F1-Measure: 0.845474285376
Confusion Matrix:
 [[4292  708]
 [ 837 4163]]

Test F1-Measure: 0.834173134176
Confusion Matrix:
 [[10801  1699]
 [ 2443 10057]]
```

IMDB

Linear SVC Hyper-Parameter Tuning (Frequency)

```
Model with rank: 1
Mean validation score: 0.887 (std: 0.004)
Parameters: {'C': 100, 'dual': False, 'max_iter': 5758, 'tol': 0.01}

Model with rank: 2
Mean validation score: 0.887 (std: 0.004)
Parameters: {'C': 100, 'dual': False, 'max_iter': 4945, 'tol': 1e-05
}

Model with rank: 3
Mean validation score: 0.880 (std: 0.008)
Parameters: {'C': 1000, 'dual': False, 'max_iter': 7093, 'tol': 0.1}

{'C': 100, 'class_weight': None, 'dual': False, 'fit_intercept': Tru
e, 'intercept_scaling': 1, 'loss': 'squared_hinge', 'max_iter': 5758
, 'multi_class': 'ovr', 'penalty': 'l2', 'random_state': None, 'tol'
: 0.01, 'verbose': 0}

Training F1-Measure: 0.948599537396
Confusion Matrix:
 [[7092  408]
 [ 363 7137]]

Validation F1-Measure: 0.882395479282
Confusion Matrix:
 [[4381  619]
 [ 557 4443]]

Test F1-Measure: 0.877359038495
```

```
Confusion Matrix:
 [[10932  1568]
 [ 1498 11002]]
```

IMDB

Decision Tree Hyper-Parameter Tuning (Frequency)

```
Model with rank: 1
Mean validation score: 0.735 (std: 0.008)
Parameters: {'max_depth': 23, 'max_features': None, 'max_leaf_nodes'
: 351, 'min_impurity_decrease': 0.00036711903528788959, 'min_samples
_leaf': 16, 'min_samples_split': 195}

Model with rank: 2
Mean validation score: 0.733 (std: 0.007)
Parameters: {'max_depth': 29, 'max_features': None, 'max_leaf_nodes'
: 726, 'min_impurity_decrease': 0.00014745050473358057, 'min_samples
_leaf': 19, 'min_samples_split': 151}

Model with rank: 3
Mean validation score: 0.731 (std: 0.009)
Parameters: {'max_depth': 24, 'max_features': None, 'max_leaf_nodes'
: 387, 'min_impurity_decrease': 0.00065437416382644633, 'min_samples
_leaf': 14, 'min_samples_split': 31}

{'class_weight': None, 'criterion': 'gini', 'max_depth': 23, 'max_fe
atures': None, 'max_leaf_nodes': 351, 'min_impurity_decrease': 0.000
3671190352878896, 'min_impurity_split': None, 'min_samples_leaf': 16
, 'min_samples_split': 195, 'min_weight_fraction_leaf': 0.0, 'presor
t': False, 'random_state': None, 'splitter': 'best'}
```

Training F1-Measure: 0.791918478527
```
Confusion Matrix:
 [[5700 1800]
 [1318 6182]]
```

Validation F1-Measure: 0.726264209885
```
Confusion Matrix:
 [[3434 1566]
 [1167 3833]]
```

Test F1-Measure: 0.740742472922
```
Confusion Matrix:
 [[8787 3713]
 [2759 9741]]
```

Comments:

For the IMDB dataset, the best classifier is between the Bernoulli Naïve Bayes model and the Linear SVC model. For these, the F1-measures are above 0.85. This is great!

4.

Clearly, the IMDB dataset is easier for the model to predict which class. This is because it is only a 2-class problem, as compared with the 5-class yelp dataset. With yelp, the F1-measures perform better than the random and majority classifiers, but only around 0.2-0.4 F1-measure points better. Whereas with the IMDB dataset, the classifiers perform significantly better than the random classifier.

NOTE: All the hyper parameter tuning was done with a RandomizedSearchCV, and ran with 60 iterations, which mathematically guarantees that you will find within 5% of the best value 95% of the time. Each model was trained with 5-Fold cross validation.