OVERVIEW

• In-depth look at Part 3 (Classification)
  • (last week you saw in-depth look at Parts 1 and 2)
• Breakdown of sample data
• Q&A
CLASSIFICATION

Four parts:

• Compare classifiers
• Experiment with the amount of training data used
• Select the best features for classification
• Do cross-fold validation
CLASSIFICATION 1: COMPARE CLASSIFIERS

- Randomly split data into 80% training, 20% testing.

- We have 5 classification methods, which you can consider to be ‘black boxes’ (input goes in, classes come out).
  1. Support vector machine with linear kernel
  2. Gaussian naïve Bayes classifier.
  3. Random forest classifier
  4. Neural network
  5. Adaboost (with decision tree)
CLASSIFICATION 1: COMPARE CLASSIFIERS

Results for SGDClassifier:
   Accuracy: 0.XXXX
   Recall: [0.XXXX, 0.XXXX, 0.XXXX, 0.XXXX]
   Precision: [0.XXXX, 0.XXXX, 0.XXXX, 0.XXXX]
   Confusion Matrix:

   [[ XXX XXX XXX XXX]
    [ XXX XXX XXX XXX]
    [ XXX XXX XXX XXX]
    [ XXX XXX XXX XXX]]

Results for GaussianNB:
... results for the rest of classifiers

my optional written analysis goes here :(
CLASSIFICATION I: COMPARE CLASSIFIERS

Results for SGDClassifier:
Accuracy: 0.XXXX
Recall: 0.XXXX, 0.XXXX, 0.XXXX, 0.XXXX
Precision: [0.XXXX, 0.XXXX, 0.XXXX, 0.XXXX]
Confusion Matrix:
[[ XXX XXX XXX XXX ]
 [ XXX XXX XXX XXX ]
 [ XXX XXX XXX XXX ]
 [ XXX XXX XXX XXX ]]

Results for GaussianNB:
... results for the rest of classifiers

my optional written analysis goes here :)

Accuracy over all classes
Precision and recall per class (e.g. index 0 corresponds to class Left)
CLASSIFICATION 1: COMPARE CLASSIFIERS

Results for SGDClassifier:
Accuracy: 0.0000
Recall: [0.0000, 0.0000, 0.0000, 0.0000]
Precision: [0.0000, 0.0000, 0.0000, 0.0000]
Confusion Matrix:

Examples labeled Left in the training data

Examples the model classified as Left

Results for GaussianNB:
... results for the rest of classifiers

my optional written analysis goes here :(
You previously used a random $0.8 \cdot 40K = 32K$ comments to train.

Using the classifier with the highest accuracy from Sec3.1, retrain the system using an arbitrary $1K, 5K, 10K, 15K, 20K$ samples from the original $32K$. 
CLASSIFICATION 2:
AMOUNT OF DATA

1000: 0.XXXX
5000: 0.XXXX
10000: 0.XXXX
15000: 0.XXXX
20000: 0.XXXX

Here is my insightful comment. It's 2+ sentences and explains much.
CLASSIFICATION 2: AMOUNT OF DATA

<table>
<thead>
<tr>
<th>Number of training examples</th>
<th>Accuracy on test set when training on the corresponding number of training examples.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000: 0.XXX</td>
<td></td>
</tr>
<tr>
<td>5000: 0.XXX</td>
<td></td>
</tr>
<tr>
<td>10000: 0.XXX</td>
<td></td>
</tr>
<tr>
<td>15000: 0.XXX</td>
<td></td>
</tr>
<tr>
<td>20000: 0.XXX</td>
<td></td>
</tr>
</tbody>
</table>

deep insight is needed. It's 2+ sentences and explains much.
Certain features may be more or less useful for classification, and too many can lead to various problems.

Here, you will select the best $k$ features for classification for $k = \{5, 50\}$.

Train the best classifier from Sec3.1 on just $k = 5$ features on both $1K$ and $32K$ training samples.

Are some features always useful? Are they useful to the same degree ($p$-value)? Why are certain features chosen and not others?
CLASSIFICATION 3: FEATURE ANALYSIS

5 p-values: [0.XXXX, 0.XXXX, ... p-values for all 173 feats]
50 p-values: [0.XXXX, 0.XXXX, ... p-values for all 173 feats]
Accuracy for 1k: 0.XXXX
Accuracy for full dataset: 0.XXXX
Chosen feature intersection: {XX, XXX, XX, XXX} # should be 5 or fewer
Top-5 at higher: {XXX, XX, XXX, XXX, XX} # should be 5

My answers to questions go here:
(a) answer
(b) goes
(c) here :)
CLASSIFICATION 3: FEATURE ANALYSIS

5 p-values: 0.XXXX, 0.XXXX, … p-values for all 173 feats]
50 p-values: [0.XXXX, 0.XXXX, … p-values for all 173 feats]
Accuracy for 1k: 0.XXXX
Accuracy for full dataset: 0.XXXX
Chosen feature intersection: {XX, XXX, XX, XXX} # should be 5 or fewer
Top-5 at higher: {XXX, XX, XXX, XXX, XX} # should be 5

My answers to questions go here:
(a) answer
(b) goes
(c) here :)

p-values for the 173 features when we set k=5 and k=50 for SelectKBest, using the full dataset
CLASSIFICATION 3: FEATURE ANALYSIS

5 p-values: [0.XXXX, 0.XXXX, ... p-values for all 173 feats]
50 p-values: [0.XXXX, 0.XXXX, ... p-values for all 173 feats]

Accuracy for 1k: 0.XXXX
Accuracy for full dataset: 0.XXXX

Chosen feature intersection: {XX, XXX, XX, XXX} # should be 5 or fewer
Top-5 at higher: {XXX, XX, XXX, XXX, XX} # should be 5

My answers to questions go here:
(a) answer
(b) goes
(c) here :)

Accuracy for the best model from 3.1, trained on the 5 best features from the 1K dataset and the full dataset
CLASSIFICATION 3: FEATURE ANALYSIS

5 p-values: [0.XXXX, 0.XXXX, ... p-values for all 173 feats]
50 p-values: [0.XXXX, 0.XXXX, ... p-values for all 173 feats]
Accuracy for 1k: 0.XXXX
Accuracy for full dataset: 0.XXXX

Chosen feature intersection: \{XX, XXX, XX, XXX\}  # should be 5 or fewer
Top-5 at higher: \{XXX, XX, XXX, XXX, XX\}  # should be 5

My answers to questions go here:
(a) answer
(b) goes
(c) here :)

Indices of the best features (in range 0-172)
• "Chosen feature intersection" means intersection of the top $k=5$ features selected for 1K and the full dataset
• "Top-5 at higher" means the top $k=5$ features for the full dataset.
### Classification 4: Cross-fold Validation

- What if the ‘best’ classifier from Sec3.1 only appeared to be the best because of a random accident of sampling?
- Test your claims more rigorously.

<table>
<thead>
<tr>
<th>Part 1</th>
<th>Part 2</th>
<th>Part 3</th>
<th>Part 4</th>
<th>Part 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iteration 1</td>
<td></td>
<td></td>
<td></td>
<td>: Err1 %</td>
</tr>
<tr>
<td>Iteration 2</td>
<td></td>
<td></td>
<td>: Err2 %</td>
<td></td>
</tr>
<tr>
<td>Iteration 3</td>
<td></td>
<td>: Err3 %</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iteration 4</td>
<td>: Err4 %</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iteration 5</td>
<td>: Err5 %</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Testing Set**

**Training Set**
### Classification 4: Cross-Fold Validation

<table>
<thead>
<tr>
<th>Kfold Accuracies:</th>
<th>[0.0000, 0.0000, 0.0000, 0.0000, 0.0000]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kfold Accuracies:</td>
<td>[0.0000, 0.0000, 0.0000, 0.0000, 0.0000]</td>
</tr>
<tr>
<td>Kfold Accuracies:</td>
<td>[0.0000, 0.0000, 0.0000, 0.0000, 0.0000]</td>
</tr>
<tr>
<td>Kfold Accuracies:</td>
<td>[0.0000, 0.0000, 0.0000, 0.0000, 0.0000]</td>
</tr>
<tr>
<td>Kfold Accuracies:</td>
<td>[0.0000, 0.0000, 0.0000, 0.0000, 0.0000]</td>
</tr>
<tr>
<td>p-values:</td>
<td>[0.0000, 0.0000, 0.0000, 0.0000]</td>
</tr>
</tbody>
</table>
CLASSIFICATION 4: CROSS-FOLD VALIDATION

Kfold Accuracies: [0.0000, 0.0000, 0.0000, 0.0000, 0.0000]
Kfold Accuracies: [0.0000, 0.0000, 0.0000, 0.0000, 0.0000]
Kfold Accuracies: [0.0000, 0.0000, 0.0000, 0.0000, 0.0000]
Kfold Accuracies: [0.0000, 0.0000, 0.0000, 0.0000, 0.0000]
p-values: [0.0000, 0.0000, 0.0000, 0.0000]

First classifier from Part 3.1 (SGDClassifier)

First fold of the data
p-values from t-tests comparing the accuracies across folds between the best classifier from Part 3.1 and the other classifiers
CLASSIFICATION 4: CROSS-FOLD VALIDATION

Kfold Accuracies: [0.XXXX, 0.XXXX, 0.XXXX, 0.XXXX, 0.XXXX]
Kfold Accuracies: [0.XXXX, 0.XXXX, 0.XXXX, 0.XXXX, 0.XXXX]
Kfold Accuracies: [0.XXXX, 0.XXXX, 0.XXXX, 0.XXXX, 0.XXXX]
Kfold Accuracies: [0.XXXX, 0.XXXX, 0.XXXX, 0.XXXX, 0.XXXX]
p-values: [0.XXXX, 0.XXXX, 0.XXXX, 0.XXXX]

p-values from t-tests comparing the accuracies across folds between the best classifier from Part 3.1 and the other classifiers.

Ex: If the best classifier from 3.1 was the RandomForestClassifier (the 3rd classifier), then the p-values should be reported in the order: [1 vs 3, 2 vs. 3, 4 vs. 3, 5 vs. 3].

What do these p-values tell us?
OVERVIEW

• In-depth look at Part 3
  • (last week you saw in-depth look at Parts 1 and 2)
• Breakdown of sample data
• Piazza recap
• Q&A
SAMPLE DATA

- 3 files:
  - `sample_in.json` -- input to `a1_preproc.py`
  - `sample_out.json` -- output of `a1_preproc.py` (to be fed to `a1_extractFeatures.py`)
  - `sample.npz` -- output of `a1_extractFeatures.py`
sample_in.json -- input to al_preproc.py

{"archived":True, 'downs': 0, 'distinguished': None, 'edited': False, 'link_id': 't3_9rgob', 'name': 't1_c0e4p0h', 'body': 'Hehe, I second this. I ADORE him.', 'score_hidden': False, 'id': 'c0e4p0h', 'retrieved_on': 1426066769, 'author_flair_css_class': None, 'subreddit': 'conspiracy', 'subreddit_id': 't5_2qh4r', 'ups': 0, 'created_utc': '1254938129', 'author_flair_text': None, 'controversiality': 0, 'parent_id': 't3_9rgob', 'gilded': 0, 'author': '[deleted]', 'score': 0}

"body": "Hehe, I second this. I ADORE him."
SAMPLE DATA

sample_out.json -- output of a l_preproc.py

{"id': 'c0e4p0h', 'body': 'hehe/UH .,/ I/PRP second/VBD this/DT ./\nl/PRP ADORE/VBP him/PRP ./\n', 'cat': 'Alt'}

'hehe/UH .,/ I/PRP second/VBD this/DT ./\nl/PRP ADORE/VBP him/PRP ./\n'
sample.npz -- output of a l_extractFeatures.py

```
In [1]: import numpy as np

In [2]: sample_feats = np.load("sample.npz")["arr_0"]

In [3]: sample_feats[4]
Out[3]:
array([ 0.00000000e+00,  2.00000000e+00,  0.00000000e+00,  0.00000000e+00,
       0.00000000e+00,  0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
       0.00000000e+00,  0.00000000e+00,  0.00000000e+00,  0.00000000e+00,  1.00000000e+00])
```

Tip: compare your output to the numbers in sample.npz using the method np.allclose()
**GENERAL QUESTIONS**

- **Global variables**
  - You may define them outside of main, as we have done for some wordlists.
  - OR in the main() function, using the global keyword.
- **Versions** – Use Python 3.8 on cdf (where spaCy 2.2.3 is installed).
- **Runtime** – Parts 1, 2, and 3 should each take around 10-15 minutes or less. Up to 30 minutes is acceptable for the autotester.
- Don't change **function headers** or string **output formats** for Part 3.
• **SpaCy version matters**
  • Tagging varies between spaCy versions. Use version 2.2.3.
  • There was some confusion about future tense. Use the word/phrase pattern in the handout -- this is necessary and sufficient.
What counts as **future tense**?

- 'll, will, gonna, going+to+VB
- Note that "going" --> "go/VBG" after preprocessing.
- Don't need to worry about:
  - Other obvious cases (e.g. "I shall")
  - Non-standard contractions (e.g. "I'ma")

In general, we will consider whatever spaCy outputs to be the “correct” output, and we will not autotest unusual edge cases.
QUESTIONS ON PART 2

• What counts as **multiple punctuation**?
  • All characters in the token must be punctuation.

• What about **spaces in lemmas**?
  • (e.g. "N.Y." -> "New York/NNP")
  • Use your judgement, we will not test these cases.

• Comments with [deleted]/[removed] etc.
  • Use your judgement.
QUESTIONS ON PART 3

• What are we supposed to do with train and test data in 3.4, when we're doing k-fold validation?
  • Re-combine these into one dataset, then use that.
OVERVIEW

• In-depth look at Part 3
  • (last week you saw in-depth look at Parts 1 and 2)
• Breakdown of sample data
• Piazza recap
• Q&A