UDACITY

PROJECT

Build a Sign Language Recognizer

A part of the Artificial Intelligence Program

PROJECT REVIEW

CODE REVIEW 13

NOTES

SHARE YOUR ACCOMPLISHMENT!

Meets Specifications

Congratulations! Great project!

You are a born coder 😉

Just apply the cosmetic change to BIC and that's it.

We're waiting for your quick resubmission 😉

PART 1: Data

1. **Student provides correct alternate feature sets: delta, polar, normalized, and custom.**
2. **Student passes unit tests.**
3. **Student provides a reasonable explanation for what custom set was chosen and why (Q1).**

Nice idea of taking left and right hand differences.

PART 2: Model Selection

1. **Student correctly implements CV, BIC, and DIC model selection techniques in “my_model_selectors.py”.**
2. **Student code runs error-free in notebook, passes unit tests and code review of the algorithms.**
3. **Student provides a brief but thoughtful comparison of the selectors (Q2).**

👏 Superb model implementations!

There is a slight difference in BIC, please refer to code review.

About your comments: BIC measures model complexity with the number of features and the number of data points. A complex model will have loads of features and few data points. Secondly, we can't really say that DIC has the highest accuracy of the 3 selectors. All of them have their advantages and disadvantages.

You may also want to know that SelectorCV is the only selector that tests unseen data. BIC tries to avoid *overfitting* by penalizing a large number of features (and states). DIC is useful because evaluates the performance of competing words, it chooses models that get a low score on competing words. This is useful because penalizes the model when it confuses with other words.

PART 3: Recognizer

1. **Student implements a recognizer in “my_recognizer.py” which runs error-free in the notebook and passes all unit tests**
2. **Student provides three examples of feature/selector combinations in the submission cells of the notebook.**
3. **Student code provides the correct words within <60% WER for at least one of the three examples student provided.**
4. **Student provides a summary of results and speculates on how to improve the WER.**

👏 Very nice WER score under 55%! This is really good!

Optionally, you may rerun BIC related cases after updating the implementation.

Nice answer, one of the key takeaway for improving WER is that words are influenced by their **context** (words before and after). For example is highly likely to see the word *Union* after the word *Soviet*.