Problem 1:

Bag-of-Words Design Decision Description

Our Bag-of-Words design used the sklearn CountVectorizer class rather than implementing our own to determine our vocabulary set. Specifically, we cleaned the data set by converting each word to lowercase, since we decided that there was likely no correlation between case and complexity. Second, we chose to keep stop words like 'the', 'and', and 'is' because we decided that they would help the model differentiate sentences of different structure (a more difficult text may use more stop words to break up complex sentences). Then, we removed occurrences of words in only 1 document or in more than 90% of documents. This was in part to keep the vocabulary size down (most tests ran around 50k items in vocabulary) and because very rare or very common words have less impact on reading level (super common stop words or made up words likely won't contribute much to a document's rating). Lastly, we used the ngram parameter from CountVectorizer to account for both single words and bigrams that may give away difficulty such as repetitions of pairs of words like 'not only' or 'such that'. We chose to ignore new words encountered in the test for the same reason we ignored words that appeared in only 1 test document. We used a word count instead of binary values because we believe that the number of time a word is seen in a document is correlated to its complexity.

Cross Validation Design Description

We again used sklearn's StratifiedKFold and GridSearchCV to perform cross validation on our train set. We used 5 folds which landed us with 4446 in test and 1111 validation entries per fold. Folds are split at random chosen with shuffled indices in the data set so that there was no bias carried over from the ordering of the train set. We stored the optimal parameters based on each model's performance over AUROC, searching through 7 logarithmically spaced regularization strengths and both L1 and L2 regularization for a total of 14 hyperparameter combinations. AUROC was chosen because it presents a better assessment of classifier performance over other metrics like accuracy. The optimal parameters were assigned to a model using the best_estimator_ attribute.

Hyperparameter Selection for Logistic Regression Classifier

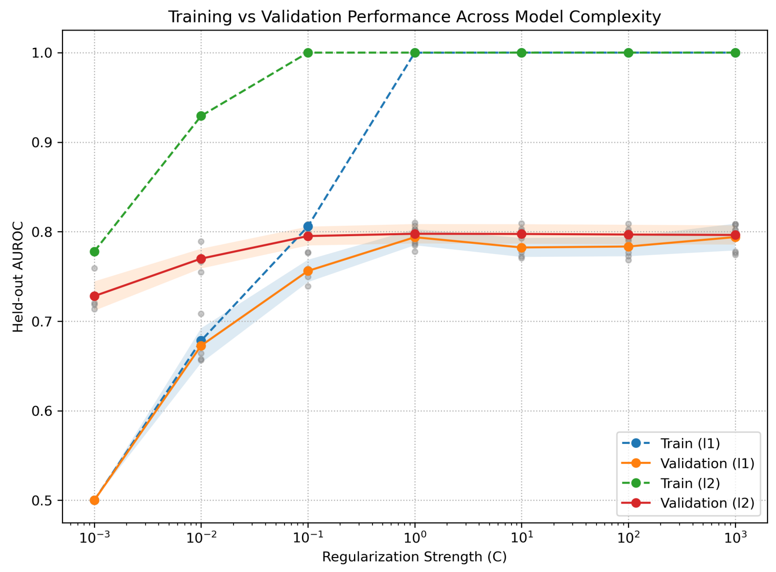
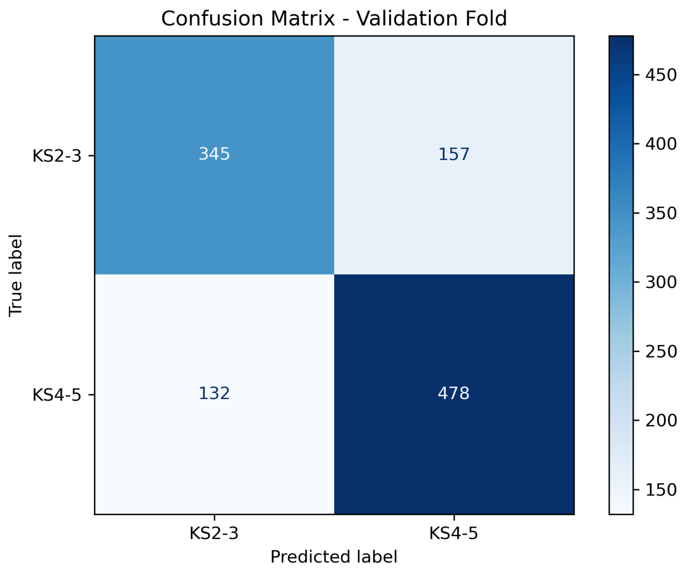


Figure: AUROC on training and validation sets as a function of regularization strength (C). Each point shows average performance across 5 folds, with shaded regions indicating ±1 SD. Low C values (<0.0.1) show underfitting (both scores low), while high C values (>10) show overfitting (train high, validation drops).

Logistic Regression is a strong choice for determining reading level, as it deals well with sparse features and high dimensions that come with the bag of words data representation. We explored both L1 and L2 regularization as well as 7 regularization strengths spaced over the domain $10^{-}3$ to 10^3. The type of regularization affects whether weights will end up as zero (feature gets ignored) or not. We found that L2 performed slightly better, which makes sense, as we wanted words with little weight to still affect the model's prediction and not simply be ignored. For C, a higher value will penalize the weights less than a low value, and it is used to prevent overfitting. The optimal C value fell at 1, where validation AUROC just started to plateau. The model was trained with the liblinear solver from sklearn. No convergence issues were had, as we set the max iterations to 1000. Our final model's parameters were L2 regularization with a C value of 1.0, found via searching over all 14 combinations of C values and both regularization patterns.

Analysis of Predictions for the Best Classifier



Our model tended to over-predict readings as higher level, as shown in the confusion matrix. Looking at the average length of the misclassified texts, they appear to be the same (around 79 words). We did notice, however, that the most misclassified authors wrote lower level texts in much older english, such that the model perceived the old words to mean the text was more difficult than reality. Authors like M. G. Lewis and Robert Louis Stevenson were classified incorrectly often and happened to write around the 18th century. The model tends to perform poorly on documents with a low score but historical language.

Report of Performance

Problem 2: