UP TO SPEED ON ACTIVE LEARNING

Data & Benchmarks

- Data table is made up of output from running both CodeSonar and Clang tools on the Juliet Test Suite.
- 59875 rows total.
- Features are: Severity, CWE, Clang Alert Flag, CodeSonar Alert Flag, Clang Rule, CodeSonar Rule, Line, True Positive
- Random Forest: 94.6% Accuracy with 5 folds
- Lasso Regression: 87.9% Accuracy with 5 folds

Severity	CWE	Clang Alert	CodeSonar Alert	Clang Rule	CodeSonar Rule	Line	True Positive
High	119	1	1	Buffer Operation	Buffer Overrun	45	1
Low	561	1	0	Dead Code	N/A	50	0
High	465	1	0	Invalid Pointer	N/A	45	0
High	465	1	0	Invalid Pointer	N/A	104	0
High	465	1	0	Invalid Pointer	N/A	79	0
Low	561	1	0	Dead Code	N/A	104	0
Low	561	1	0	Dead Code	N/A	129	0
Low	561	1	0	Dead Code	N/A	30	0
Low	561	1	0	Dead Code	N/A	71	0
Low	561	1	0	Dead Code	N/A	61	0

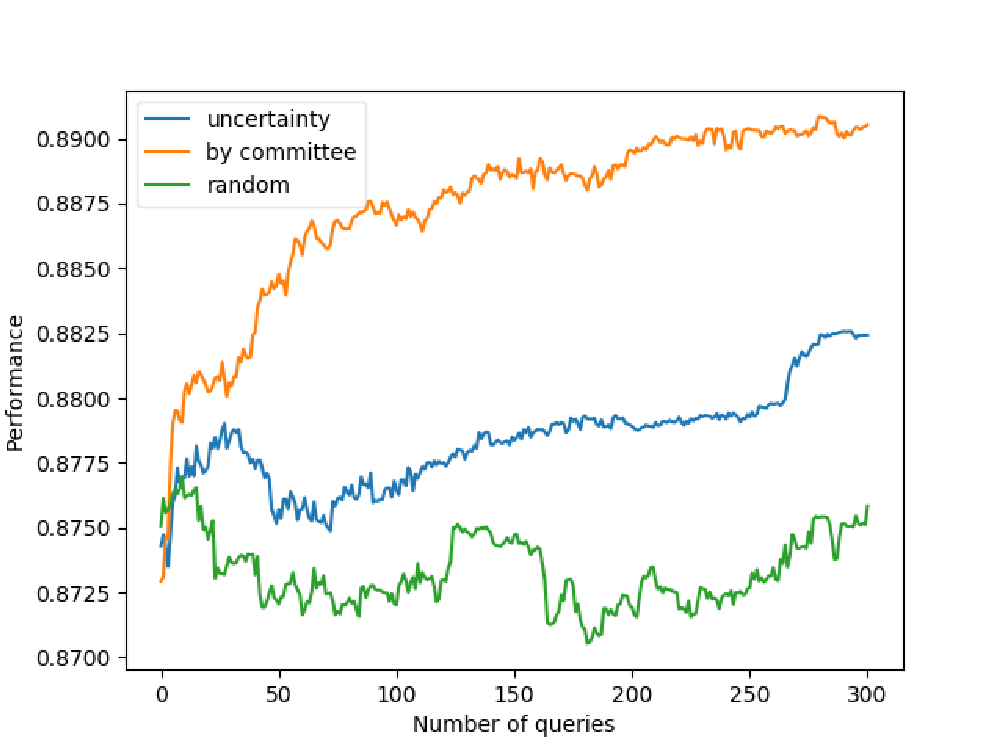
Alipy Library

- Prebuilt library to handle active learning loops
- Primarily used their AlExperiment object. Initialization of object takes 7 parameters- independent and dependents variables, the predictive model object from sci-kit learn, performance metric, any stopping criteria, the stopping value, and batch size
- Data is then split with split_AL method which takes 4 parameters- test ratio, initial label rate, split count, and whether or not the initially labeled data should include all classes (if possible)
- The query strategy is then set with set_query_strategy method. There are a number of choices for simplicity's sake and the limits of my own understanding I tested three different strategies- Uncertainty, Query by Committee, and Random.
- After the loop is start and the experiment is started. Theoretically offers multi-threaded capability but I've found it to be buggy.

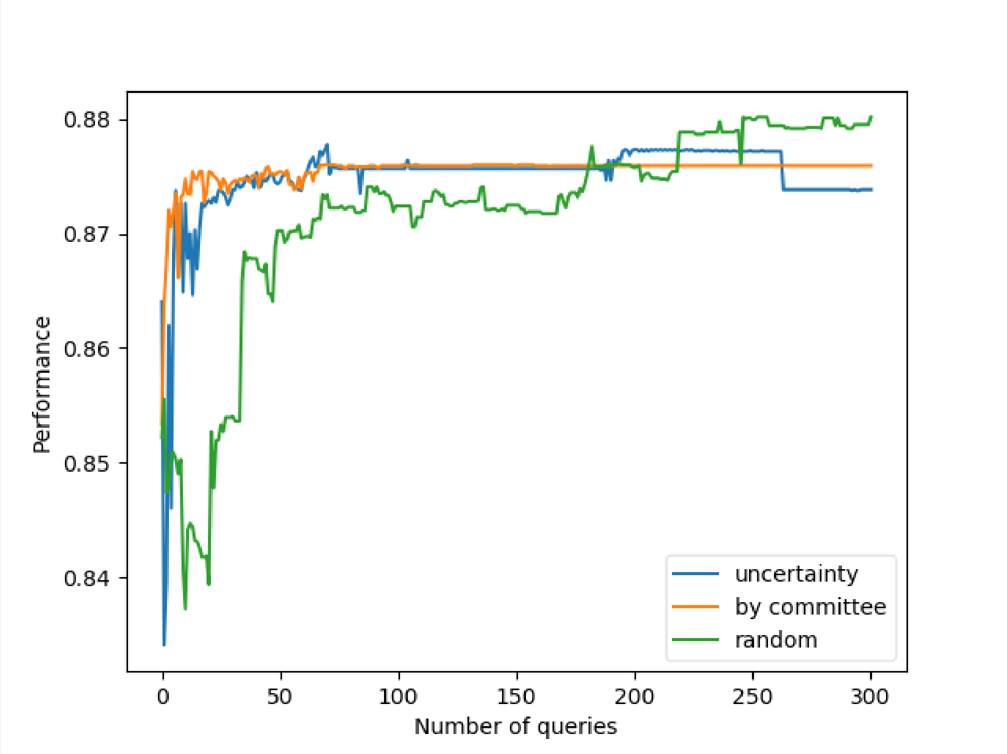
Query Strategies

- Uncertainty- query the row which the model was previously least certain of it’s true value.
- Query by Committee- build multiple models on subsets of the data, vote on the value of each row. Query the row which previously had the most split vote.
- Random- query a random row each loop.

Lasso Logistic Regression Results

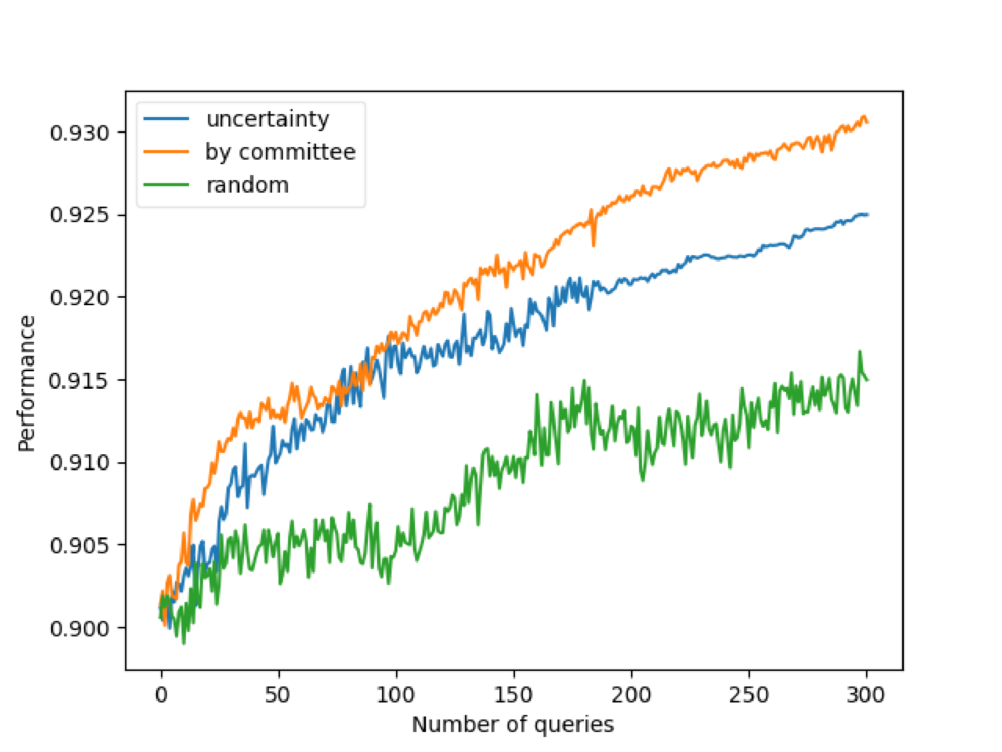


300 Initially Labeled, 300 Queries, 5 Folds

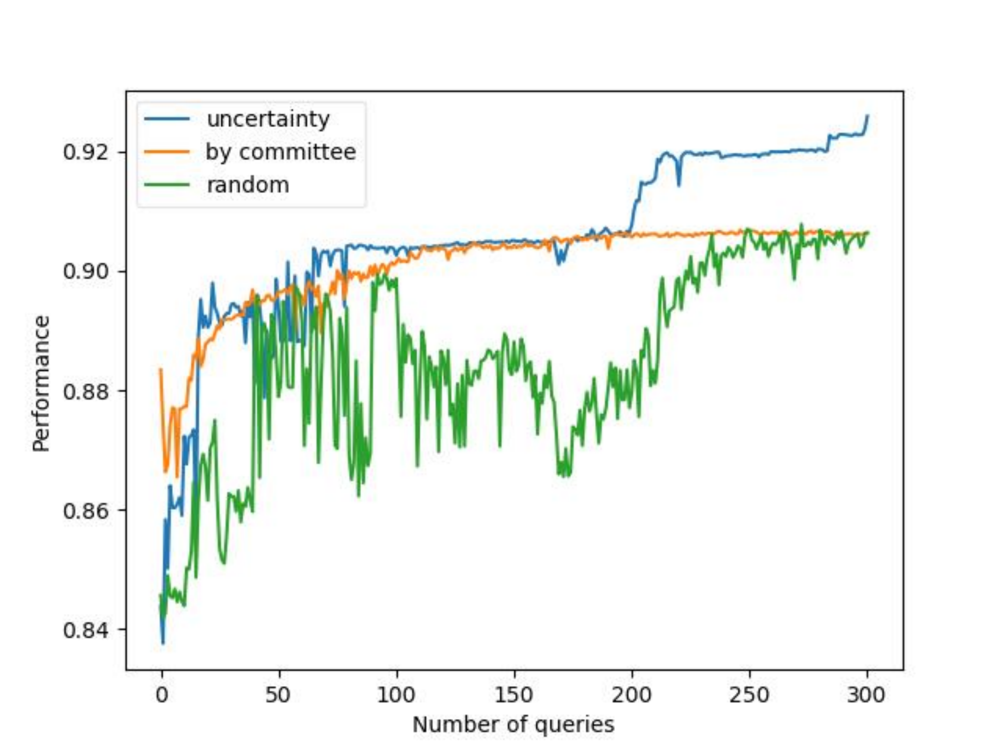


25 Initially Labeled, 300 Queries

Random Forest Classifier Results



300 Initially Labeled, 300 Queries, 5 Folds



25 Initially Labeled, 300 Queries

Concerns and Limitations

- Active Learning Lasso Regression surpasses the static model when given 600 entries. Given the size of the data set, it seems unlikely that this is due to overfitting.
- Lasso Regression on the tiny set of 25 initially labelled rows+300 queries approximates the static model for Uncertainty and Committee querying strategy and actually *surpasses* it with a random querying strategy. Seems extremely strange to me.
- Potential reasons for this- overfitting, problems with the alipy library, some sort of flaw in our data.
- Data does not come from a true code database, but rather a test suite.
- Alipy library does not allow me to change the random seed, so I'm unable to test different seeds using this framework.
- Documentation for Alipy is rather spotty, and it's clear that English is not the author's first language.

Suggested avenues of exploration

- Expand our existing data table with output from the other two tools and potentially cyclomatic complexity measures, see how this then affects our results.
- Code the active learning loop by hand, see if I get different results from the Alipy Library.