After you complete the program, please write a report. The report should include the followings:
(1) The contribution of each member in your group.
(2) Brief summary of how you implemented the program.
(3) The error rate of the test data.

Each group member was responsible for creating their own interpretation of the Naive Bayesian classifier model before we then combined the best components of each model into a final product. Joe and Nikita worked to combine the three models while Colten provided documentation for each and wrote the final report. All models including each member's and the final combined model are included in the deliverable.

Colten's model works by modeling the probability of distribution of each provided feature including eye color, mode, and skill given a specific class label or either good or bad and uses these probabilities to make predictions for new data. This implementation first reads in training data from a file and parses each line into a structured Record that contains the three features and the associated class label. Throughout the training process a count tracking the appearances of each class and feature is tracked with class_couts and feature_counts respectively. The classifier in this case also uses Laplace smoothing for conditional probabilities as a way to help prevent zero-probability issues for unseen feature values. For predictions the classifier calculates the posterior probability of each class for a given input record by adding the logarithms of the prior probability of the class and conditional probability of the features. The class with the highest resulting score is chosen as the prediction. For evaluation, the test() function reads test data from a file, predicts the label for each row, prints both the predicted and actual class labels, and computes the overall accuracy of prediction. Generally speaking, this method follows a structured approach that provides clear distinction between parsing, training, prediction, and evaluation.

Joe's model provides a more abstract and generalized version of the NB classifier in comparison to Colten's. Each data point is represented as a Sample structure that contains a vector of data features and only a single class label. Instead of hardcoding the specific feature types of eye color, skill, and alternate mode, this model accepts any number of categorical features. Counts for each class label are tracked during training with classCounts and a feature map is built of each feature value for each class with featureCounts. This is accomplished with a vector of maps for each class in which each map is associated with a feature position and records the counts of observed values at that position. The classifier calculates log-probability of each class when predicting a label for new input by using Bayes' theorem. The log of the class's prior probability and the log of the smoothed conditional probabilities of each feature value occurring in that class are added together and the label with the highest overall log probability is returned as the prediction. Performance evaluation is carried out with evaluateAccuracy() and different training proportions can be created with splitAndEvaluate(). This method provides a more adaptable approach to the Naive Bayesian classifier that could be applied to other datasets.

Nikita's implementation combines elements of the former two to create a structured model with a hardcoded NB classifier that works for the provided robot dataset. A Robot struct contains categorical attributes and the class label, and class_counts and feature_counts are used to track the names they represent. A variable named total_samples is used to track the overall dataset size. Similar to the other two, posterior probabilities for each class are calculated by multiplying class prior probability with smoothed likelihood of observed features and Laplace smoothing is applied to avoid zero probabilities. The prediction is selected as the class with the highest probability. Additional evaluation and accuracy is also provided.

The combined model implementation attempts to combine the best parts of the three separate models to provide a more generalized yet effective Naive Bayes classifier. A Robot struct provides a structured approach to representing each data point with feature fields and the class label. Nested maps are used to store the frequencies of each feature value for each class and the classifier to compute class priors and conditional probabilities while also using Laplace smoothing for unseen value handling. The main benefit of this implementation is that it can be adapted to additional features without altering the core logic. Training and Testing sets are again loaded from their respective files and predictions and actual labels are displayed together to compare. Both classification accuracy and error rate are also shown. The mode here generally provides better readability and utility.

Every implemented model achieved an accuracy score of 81.2% and an error rate of 18.8%. Testing smaller sets of training data only resulted in less accurate results and a higher error rate that is ultimately expected. More training data would in theory provide more accurate results.