Binary Road Wheel Alignment Classification: Final Report and Experiment Summary

Joshua McCready, M.S.E. ECE Student and Core Steering Mechatronics Engineer at Ford Motor Co.

***Abstract*—Improper tire wear due to wheel misalignment in passenger vehicles results in negative side effects for customers and the OEM. This paper proposes the beginning of passenger vehicle wheel misalignment prognostic strategy using internal vehicle signal for detection of degraded conditions in the absence of DTCs and without the need dynamic or kinematic models of the passenger vehicle chassis and road wheels. Identifying degraded conditions before they result in negative effects reduces waste, customer headache, and warranty cost to the automaker. The data in this study was collected in a longitudinal study of a 2017 fusion instrumented with a data collection system. The feasibility of implementing prognostics via machine learning classification of internal vehicle data using over the air data collection for offline classification shall be studied.**

I. Experiment Primer

THE original dataset required additional processing to be tractable. Originally the dataset was 24.9 hours of time-series data sampled at 100 Hz. To enable work without the aid of additional computational resources, the dataset was down sampled to 0.2 Hz. Then, the data was divided into several Pandas dataframes one each for the numerical features, categorical features, and response variables. Those dataframes were then pickled such that they could be easily accessed without rerunning code too often. This was accomplished in the `DataProcessing.ipynb` Jupyter notebook.

A. Feature Selection with Filtering Methods

From the feature to feature correlation study also performed in `DataProcessing.ipynb` it was observed that all of the wheel speed measurements were exactly correlated (or at least to the precision printed in the heat-map). There was a two fold amelioration to correct this issue. First some feature engineering such that two new features were created differencing the right and left wheel speed features for the front and rear. The names of the new features are:

1) "Front Wheel Left Right Difference"
2) "Rear Wheel Left Right Difference"

The second step of amelioration of exact feature to feature correlation was to remove all but one the offending features ("Vehicle Speed (Engine or Brakes)", and all four wheel speed features). Because it was not used in creating the two additional features, "Vehicle Speed (Engine or Brakes)" was retained. It is important to note that the two new features created from the wheel speeds did not have exact correlations with any other features.

Doing that preliminary examination of feature to feature correlation sets up an initial filter method feature selection step removing the following numeric features:

- "Front Right Wheel Speed"
- "Front Left Wheel Speed"
- "Rear Right Wheel Speed"
- "Rear Left Wheel Speed"

The mutual information of the entire dataset was studied in `DataProcessing.ipynb`, and the ranks were used to eliminate feature with very low probability of predicting the output. Perhaps this was in error because allows information leakage, and should have been calculated with only the training set. Nonetheless, the following features were removed due to low rank:

- Feature ranked 29 is "Steering Control Module Current" with value 0.000009
- Feature ranked 30 is "Accelerator Pedal Position Percent Rate" with value 0.000004
- Feature ranked 31 is "Vehicle Lateral Acceleration" with value 0.000002
- Feature ranked 32 is "Total Brake Torque" with value 0.000000

Intuitively, variables related to acceleration and braking have very little relation to the road wheel alignment which primarily impacts steering. One can also argue that lateral acceleration depends more on the vehicle maneuver (recall $A_{lateral} = v^2/R$ from kinematics) than any of its degraded state.

B. Classification of Output

At the time of this report, only numerical features were considered as model inputs. Categorical features were excluded solely because of lack of time. With the filter methods applied, 24 features remained. The classification problem was broken down into a binary choice of "Nominal" (0) or "Misaligned" (1). The train test split is set at 0.2 for all experiments.

Training	*Initial (ave)*	*Final (ave)*	**Holdout**
Accuracy:	0.75	0.82	0.82
F1 Macro:	0.78	0.86	0.87
Recall Macro:	0.67	0.91	0.83
Precision Macro:	0.91	0.83	0.9
Time training:	0.139	0.038	N/A

TABLE I: Cross validation macro performance averages for training from initial and experiment compared with performance from the holdout set

II. Naive Bayes

A. Experiment Description

We begin with Naive Bayes, the simplest classifier method, after filter methods of feature selection have been applied. The experiment is carried out in `P3_Bayes_Classifiers.ipynb`. The experiment consists of applying recursive feature elimination using mutual information scores to successively remove k features with cross validation of a pipeline performing standardization and the Naive Bayes classifier. An outline of the procedure is included to ease understanding of `P3_Bayes_Classifiers.ipynb`

1) Apply filter methods of feature elimination
2) Create train vs. test split
3) Create Pipeline of standardization and Naive Bayes
4) Choose recursive experiment parameters:
   a) Feature indices to be used
   b) Number of folds for *cross_validate*
   c) Number of features to eliminate at each step
5) *RFE_MI()* to perform recursive feature elimination:
   a) Calculate split with *Kfold* separately from *cross_validate*
   b) Perform cross validation using *cross_validate* with pipeline
   c) Select best model using ”f1_train” scoring
   d) Calculate mutual information score for all features
   e) Remove k lowest rank feature indices
   f) Call *RFE_MI()* until stopping criteria reached
6) Performance for the training, validation, and holdout set are compared
7) The best model is selected using a heuristic of low number of features, high f1 and accuracy score.

B. Experiment Performance

The performance for different Naive Bayes classifiers that were developed can be seen in Table I and Figure 4. Slight improvements were had between the initial and final best model fits. These improvements were the result of removing features that had low mutual information ranks. We can see that the test, validation, and holdout data-sets performed similarly for sets of features. Each trial with 12 or less features performed similarly. It does not appear that over-fitting was a problem as validation and holdout sets performed similarly to training. The use of cross validation is likely to thank. Without further feature engineering or expansion of the features by some basis function it appears that a Naive Bays model underfits the data as it has been presented.

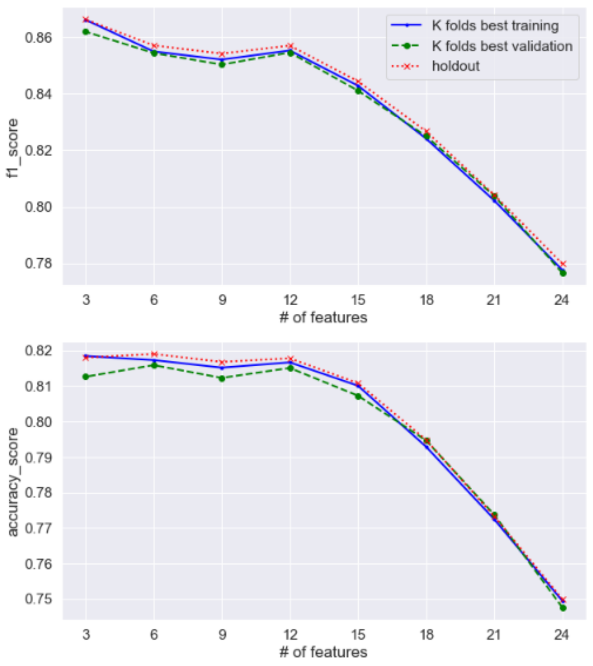

Fig. 1: Performance of Naive Bayes cross validation and holdout data over recursive feature elimination judged by F1 and accuracy scores. Initially, there are 24 features yielding minimal performance, but as some are eliminated performance improves somewhat.

Given the lack of benefit in including more than three features we choose the simplest model with the fewest features as performance benchmark. The final features included in the model are:

- ”Vehicle Speed Condition”
- ”CVC Control Error Angle Estimate”
- ”Average Assist Torque Long Term”

These features were all at the top of the mutual information scores for the whole population. ”CVC Control Error Angle Estimate” and ”Average Assist Torque Long Term” intuitively explain the error state that the model is trying to predict. ”Average Assist Torque Long Term” explains there is some change to how much torque the driver has to put in given a misalignment. Basically, this is the ghost of extra steering pull. The ”CVC Control Error Angle Estimate” is an internal signal modeling the offset angle that the steering wheel. Again, given the knowledge of toe and how it displaces the steering

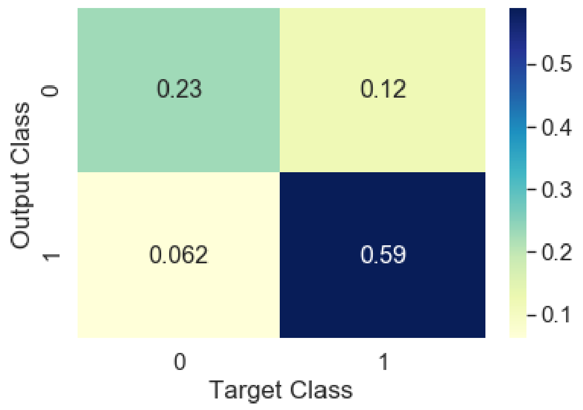

Fig. 2: Confusion Matrix for the holdout dataset. Entries are in decimal percentages rather than number of samples. This figure demonstrates that there is a skew in the holdout set, with only 29% of examples being "Nominal" and 71% being "Misaligned". The model incorrectly predicts 18% of the time, with more false negatives than false positives. This is likely due to the many noise factors applied in this dataset.

wheel, it is not surprising that this signal predicts a misalignment condition.

III. Logistic Regression

A. Experiment Description

The next most complicated model used was logistic regression. The experiment was carried out in `P3_Logistic_Classifiers.ipynb`. After the application of the filter feature selection, the experiment consists of grid search of hyperparameters regularization choice, α, and λ with elastic net recursive feature elimination with cross validation. An outline of the procedure is included to ease understanding of `P3_Logistic_Classifiers.ipynb`

1) Apply filter methods of feature elimination
2) Create train vs. test split
3) Create Pipeline of standardization and Logistic Regression and max iterations of 500.
4) Setup hyper parameters:
    a) Regularization: {'none', 'l1','l2', 'elasticnet'}
    b) λ: {0.01, 0.25, 0.5, 1, 5}
    c) α: {0.1, 0.5, 0.9}
5) Choose recursive experiment parameters:
    a) Feature indices to be used
    b) Number of folds for *cross_validate*
6) *RFE()* to perform recursive feature elimination:
    a) Calculate split with *Kfold* separately from *cross_validate*
    b) Perform cross validation using *cross_validate* with pipeline according to grid search with given hyperparameterss
    c) Select best model using "f1_train" scoring
    d) Perform feature selection with elastic net
        i) Find hyperparameter configurations for 'elasticnet'
        ii) Determine ranks of 'elasticnet' configurations
        iii) Select top ranked pipeline object and inspect weights
        iv) Remove feature indices that have weights near 0
    e) Call *RFE()* until stopping criteria reached
7) Performance for the training, validation, and holdout set are compared
8) The best model is selected using a heuristic of low number of features, high f1 and accuracy score.

It is important to note the way that elastic net recursive feature elimination was attempted. Because of the grid search, the best model was not always going to use elastic net regularization. To solve this, the best ranked model with elastic net was found in order to eliminate features.

Training	Initial	Final	Holdout
Accuracy (Avg):	0.74	0.73	0.74
F1 Macro (Avg):	0.81	0.81	0.81
Recall Macro (Avg):	0.88	0.88	0.76
Precision Macro (Avg):	0.76	0.75	0.88
Time training (Avg):	2.28	1.34	N/A

TABLE II: Logistic Regression: Cross validation macro performance averages for training from initial and experiment compared with performance from the holdout set. Performance was not altered greatly using grid search and there were not significant differences in performance among the holdout set.

Iteration	# Features	Regularization	λ	α
1	24	l1	0.5	N/A
2	20	l1	0.25	N/A
3	18	l1	0.5	N/A
4	17	l1	0.5	N/A

TABLE III: Logistic regression grid search results for best fit model according to F1 score on training sets. 'l1' models seemed popular, but the 'l1' choice wasn't consistent among lower ranks, with 'elasticnet', 'none', or 'l2' sometimes breaking into the top 5.

B. Experiment Performance

The performance for different Logistic Regression classifiers that were developed can be seen in Table I and Figure 4, and the best pick models according to F1 training score can be seen in Table III.

It does not appear that logistic regression improves at all over a simple Naive Bayes model with three inputs.

The grid search that was performed appeared to result in no performance gains as many of the models had similar performance despite their changed parameters. Additionally, applying recursive feature elimination with elastic nets did not appear to work correctly. The weight matrix never had elements that were equal to 0, so to accomplish any feature reduction I simply eliminated weights below 0.04, which was not picked scientifically.

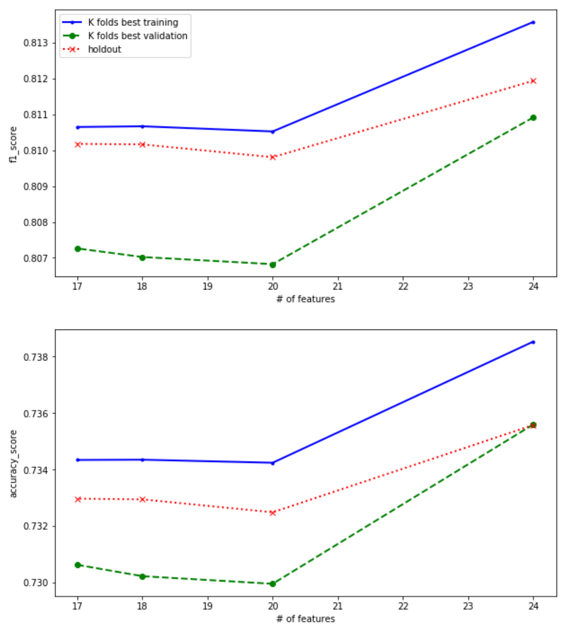

Fig. 3: Performance of Logistic Regression cross validation of best model pick from grid search and holdout data over recursive feature elimination judged by F1 and accuracy scores. It doesn't appear that feature elimination affects performance as the y range is small.

No over-fitting occurred, but the data seemed to be underfit in all test, validation, and holdout sets. Based on my experience trying grid search and feature elimination across all regularization types, I believe logistic regression is not useful in predicting better than simple models in this case. In the future, it might be worth evaluating this again with polynomial expansion to test if improvements can be had. I am also concerned with the fact that my weights never became space under 'elasticnet' or even under 'l1' regularization, this should be investigated.

IV. Discussion

Regretfully, neither Naive Bayes or any of the Logistic Regression forms that were tested appear to predict the dataset well. Though, the simplest Naive Bayes model performed pretty well considering it used only 3 features. This is not unexpected given that little feature

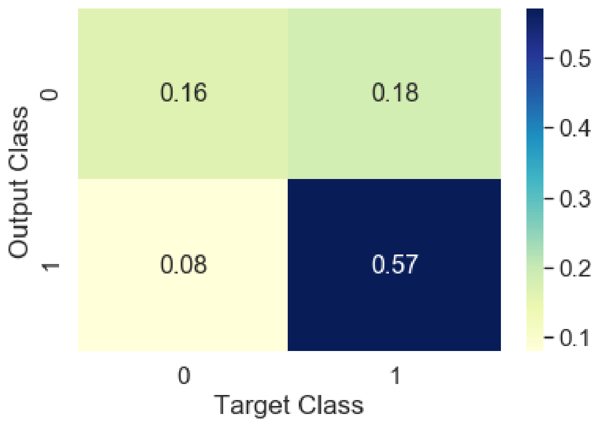

Fig. 4: Logistic Regression Confusion Matrix for the holdout dataset. Entries are in decimal percentages rather than number of samples. This figure demonstrates that there is a skew in the holdout set, with only 24% of examples being "Nominal" and 76% being "Misaligned". The model incorrectly predicts 24% of the time, with more false negatives than false positives.

engineering was done to prepare the dataset which might have facilitated analysis. However, I was able to develop a somewhat robust way of doing wrapper methods of feature selection over grid search with cross validation. The code framework that I have developed with the help of this class can be used to evaluate the real world problem of alignment detection in passenger vehicles.

A. *Limited Time For Model Coverage*

Time constraints stemming from the creation of the dataset from raw files created a bottleneck around the different kinds of models I could evaluate. As a result I was unable to try any MLP models. MLPs and other neural nets should be evaluated. I also did not try to do feature expansion with any basis functions. Nor was I able to include the categorical features in the dataset that might be of use. Given that there is still potential machine learning and pattern recognition methods to be used to effectively identify, I propose some next steps.

B. *Next Steps*

The next steps for this research are listed below:

1) Continue the work of study of the raw time series numeric features
    a) Evaluate MLP based classifiers in their effectiveness at alignment classification
    b) Record "best" models from Naive Bayes, Logistic Regression (after investigating my concerns), and MLP

   c) Put ”best” models together to try to predict a holdout set with greater accuracy by implimenting a majority voting scheme on a sample by sample basis
2) Investigate what models can absorb categorical and numeric features and see if any improvements can be made the first item.
3) Perform feature engineering, starting simply with creating moving window histograms of the features.
4) Study the different models that are available for classification and see if improvements can be made over time series features that are directly used for classification
5) Study the feasibility of data collection according to planned capabilities for in-vehicle networks and telematics.

These steps will support Ford’s success at delivering improved customer convenience and satisfaction, environmental stewardship, and warranty reduction.