

Detecting Distracted Driving with Neural Nets

Joseph Blubaugh

09 November 2016

Overview

The first pass analysis used summarized statistical measures to fit both a random forest and neural network model to the data. The results primarily showed that aggregating the data over an entire event prevented the two algorithms from detecting texting events. Also, because the data was aggregated, there were very few data points to model between testing and training sets. The random forest algorithm was a better choice than the neural network because of the large number of predictor variables relative to the number of observations. Both models however, were guilty of overfitting. For the next phase of this analysis, I am focussing on looking at the data in its original form. I have chosen to focus on the neural network rather than the random forest mainly because of the data structure. In the previous analysis the aggregated data set contained 119 observations with 40 predictor variables. Since I am using the original data with no statistical measures the data set contains more than 1 million rows on 12 predictor variables.

Neural Nets applied to Distracted Driving

In order to standardize the results of my analysis, I have elected to use the R package caret to train all of my models (<http://topepo.github.io/caret/index.html>). Caret is a modeling framework that allows you to run many different types of models with different combinations of parameter sets at the same time. It can easily use models from other popular packages and offers a rich set of validation tests and diagnostic plots. Another benefit to using this package is enabling parallel computing of cross validation tasks.

I am using k=10 cross validation for all models as well as several combinations of paramters. Training and Testing sets are all approximately the same size 50/50 split. Only Models 1 and 2 have been fully trained.

Model Results: (DS: Data split at 365 Seconds, ES: Training Set taken from Entire Simulation)

Model	Data Processing	Data Split	Size	Decay	Training	Testing
Model 1:	Original	DS	15	.20	.783	.690
Model 2:	Original	ES	15	.05	.752	.752
Model 3:	Differencing	DS	15	.05	.594	.571
Model 4:	Differencing	ES	15	.05	.592	.591
Model 5:	Moving Avg	DS	30	.05	.525	.527
Model 6:	Moving Avg	ES	10	.00	.528	.528
Model 7:	30 Sec Cut	DS	50	.10	.822	.689
Model 8:	30 Sec Cut	ES	50	.00	.793	.782
Model 9:	30 Sec Diff	DS	50	.00	.631	.594
Model 10:	30 Sec Diff	ES	30	.05	.681	.623

```
#####
## Code for running Model 2
## Compute time: 12 hours over 6 CPU

## Set Cross Validation
fit.control = trainControl(method = "cv", number = 10)

## Create combination of model parameters to train on
search.grid = expand.grid(decay = c(0, .05, .1, .2),
                          size = c(1, 3, 5, 10, 15))

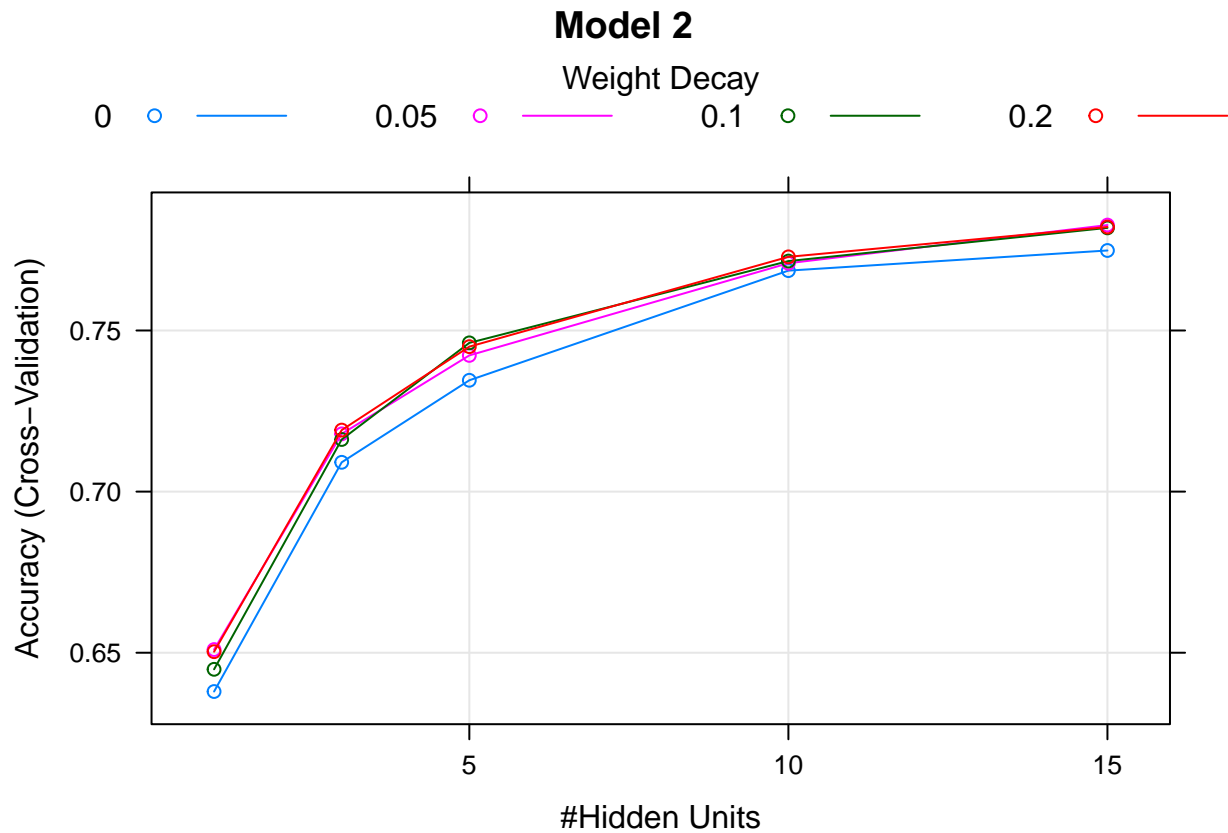
## Limit the iterations and weights each model can run
maxIt = 500; maxWt = 10000

fit = train(Texting ~ . - Time, mdl.02.train, method = "nnet",
            trControl = fit.control,
            tuneGrid = search.grid,
            MaxNWts = maxWt,
            maxit = maxIt)

542550 samples, 12 predictors
2 classes: '0', '1'

Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 488295, 488296, 488295, 488294, 488296, 488295, ...
Resampling results across tuning parameters:
```

decay	size	Accuracy	Kappa
0.00	1	0.6379560	0.2186901
0.00	3	0.7090977	0.3627473
0.00	5	0.7345479	0.4244812
0.00	10	0.7685799	0.5083622
0.00	15	0.7748299	0.5226515
0.05	1	0.6509813	0.2211743
0.05	3	0.7179190	0.3914567
0.05	5	0.7422744	0.4481362
0.05	10	0.7708690	0.5133975
0.05	15	0.7827131	0.5405813
0.10	1	0.6448476	0.2514474
0.10	3	0.7161829	0.3820061
0.10	5	0.7461580	0.4562864
0.10	10	0.7715399	0.5151954
0.10	15	0.7818469	0.5390179
0.20	1	0.6503197	0.2436473
0.20	3	0.7190932	0.3910539
0.20	5	0.7449894	0.4524052
0.20	10	0.7728504	0.5181559
0.20	15	0.7821215	0.5392592



Neural Network Confusion Matrix

	Predicted	
Actual	0	1
0	281242	39644
1	82515	139149

(Training Set) Overall Performance: 0.752101

	Predicted	
Actual	0	1
0	280817	39752
1	82577	139405

(Testing Set) Neural Net Overall Performance 0.7519984

Plots

The following 3 plots overlay the prediction for model 2 against the test sets for the first 3 subjects. Yellow dots represent predictions for Texting events and the black dots are no texting events. The blue curve is a loess curve for the probability prediction. It shows the overall trend for the predictions by time. It is encouraging that for Subject 001, the

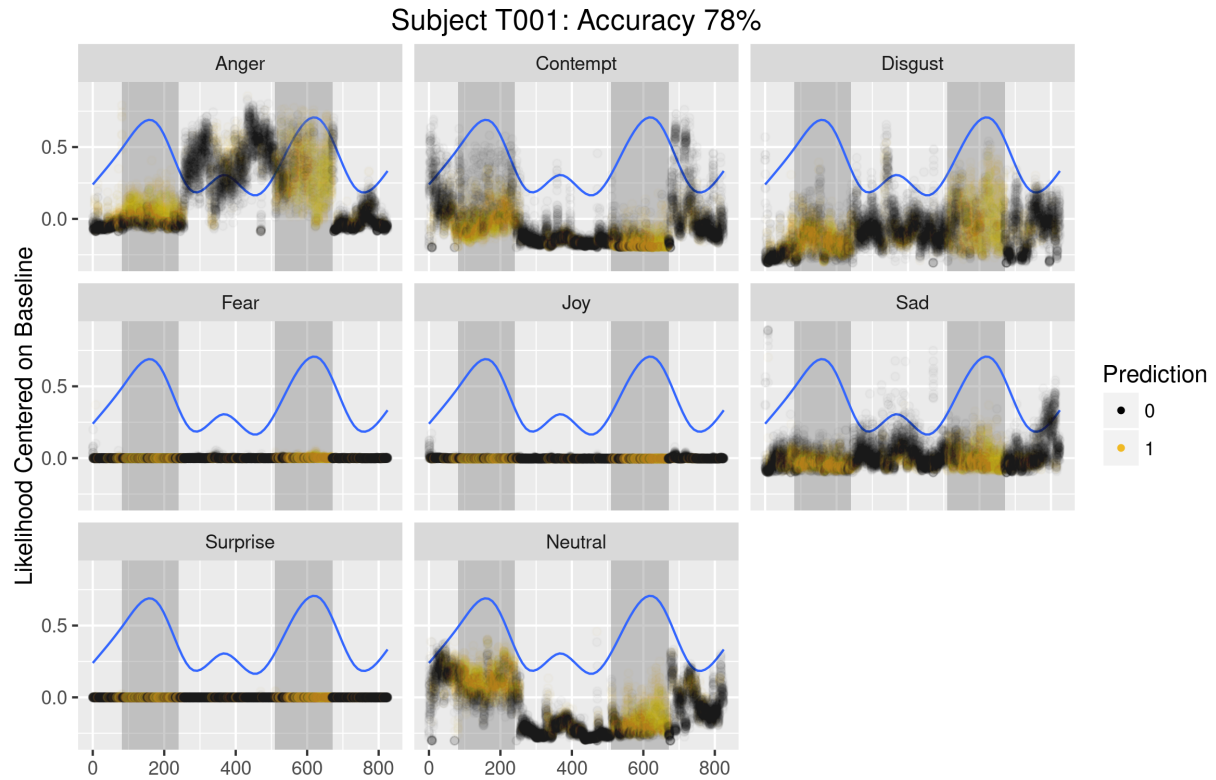
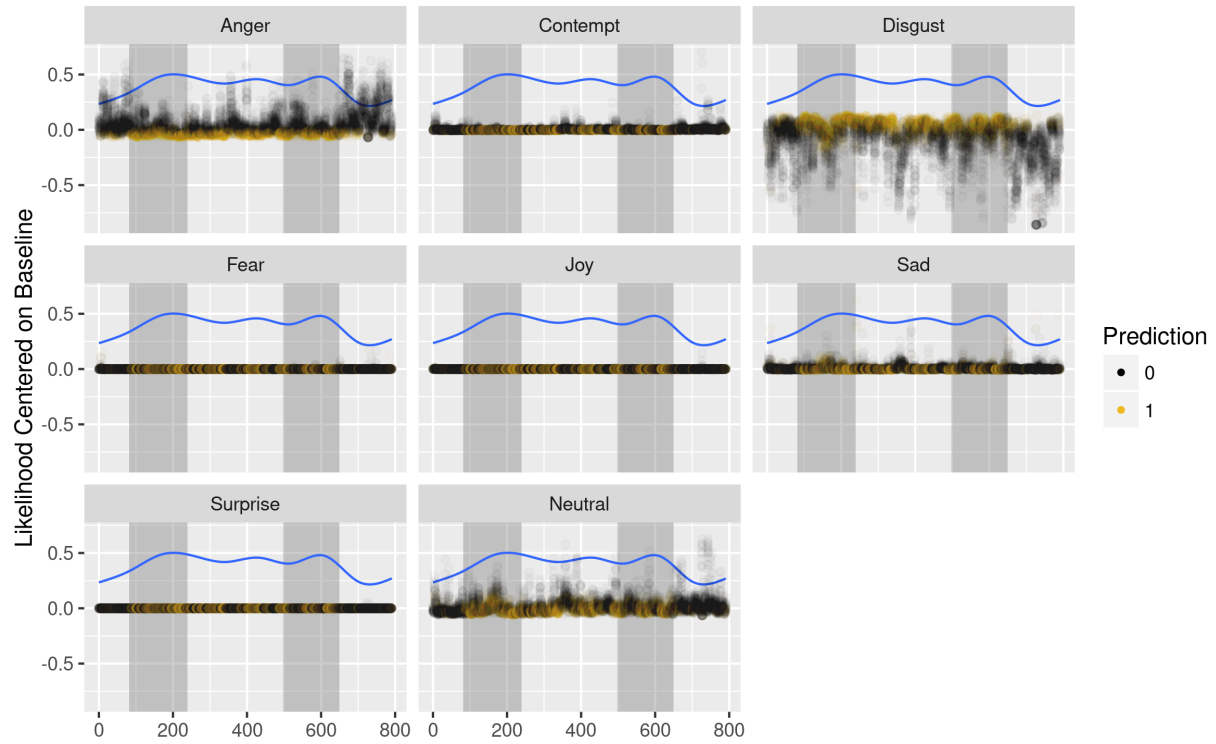


Figure 1: Model 2 Prediction

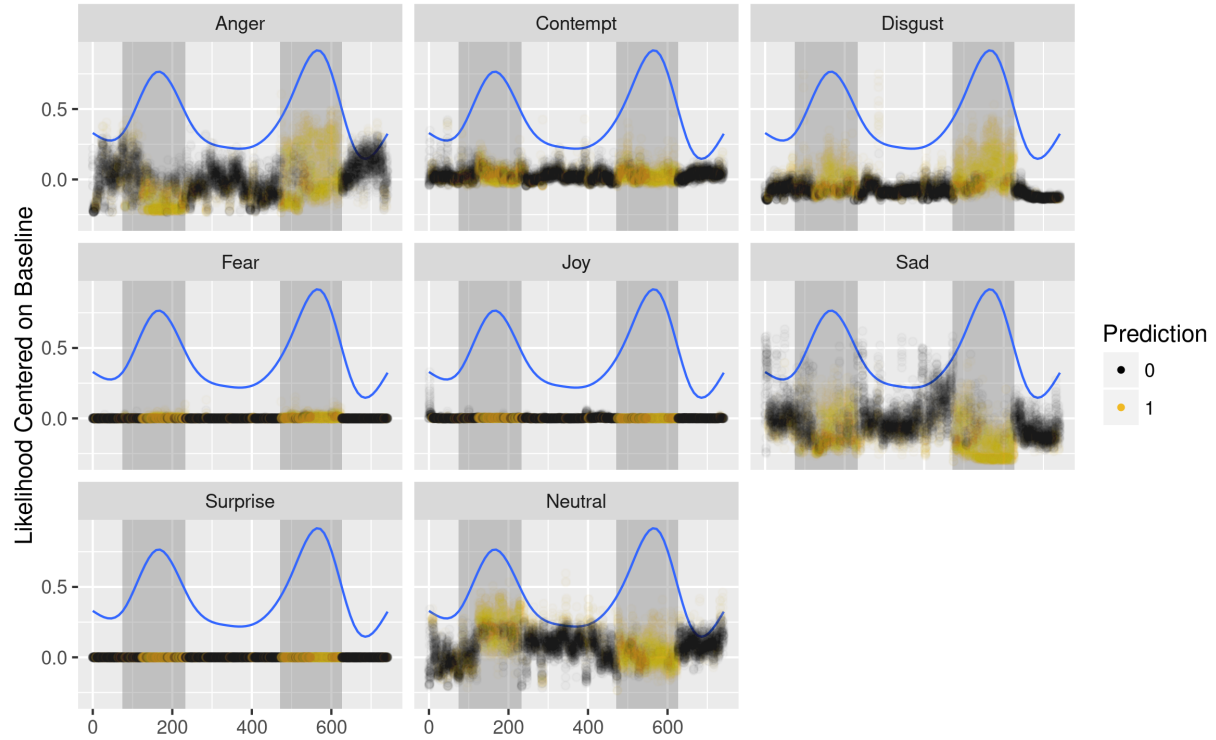
probability tends to peak during the periods of texting which are highlighted in gray in the background.

Just as interesting though is the plot for Subject 002 where the model has a horrible fit overall and only predicts texting correctly about 60% of the time. Subject 003 has an excellent fit overall with a large majority of positive predictions occurring during the texting event window.

Subject T002: Accuracy 60%



Subject T003: Accuracy 83%



Accuracy by Subject:

Table 2: Table continues below

	T022	T035	T018	T006	T083	T086	T076	T074	T032	T088	T009
Train	0.940	0.929	0.918	0.921	0.911	0.916	0.890	0.877	0.853	0.856	0.843
Test	0.943	0.930	0.926	0.920	0.920	0.903	0.894	0.878	0.862	0.861	0.837

Table 3: Table continues below

	T003	T020	T064	T008	T080	T013	T007	T081	T005	T011	T010
Train	0.833	0.827	0.834	0.832	0.824	0.804	0.812	0.815	0.815	0.804	0.803
Test	0.833	0.831	0.831	0.828	0.820	0.817	0.815	0.815	0.813	0.812	0.805

Table 4: Table continues below

	T044	T073	T079	T042	T001	T051	T016	T082	T024	T039	T066
Train	0.802	0.795	0.785	0.776	0.784	0.771	0.773	0.776	0.776	0.769	0.767
Test	0.802	0.801	0.787	0.786	0.785	0.779	0.779	0.776	0.773	0.768	0.765

Table 5: Table continues below

	T084	T061	T012	T029	T046	T015	T077	T031	T017	T033	T014
Train	0.754	0.753	0.742	0.750	0.736	0.731	0.745	0.738	0.726	0.709	0.704
Test	0.758	0.753	0.745	0.745	0.737	0.735	0.729	0.728	0.726	0.716	0.692

Table 6: Table continues below

	T060	T019	T036	T021	T041	T004	T047	T040	T054	T002	T034
Train	0.684	0.671	0.665	0.650	0.634	0.625	0.629	0.614	0.613	0.602	0.595
Test	0.681	0.669	0.664	0.656	0.634	0.629	0.618	0.616	0.609	0.597	0.594

	T025	T023	T027	T038
Train	0.563	0.512	0.506	0.502
Test	0.563	0.512	0.504	0.503

Conclusions and Next Steps

So far the neural net models have shown encouraging results in the ability to detect texting events for some subjects based on facial expressions alone. It clearly helps model performance to consider each individual separately. In my next few sessions I would like to finish building and testing the performance for the rest of the modeling scenarios I have proposed. I would also like to look into some additional neural net models that have recursive features. So far in the models I have trained I have ignored the time component which might be valuable.