Reaching the Gold Standard
Assessing Driving Ability among Expert and Student Drivers

Alyssa Davis
Statistics Senior Project
Advisor: Dr. Gary B. Hughes
California Polytechnic State University, San Luis Obispo
December 5, 2013

INTRODUCTION:

Mentor eData is a start-up company interested in enhancing driver safety with predicative analytics and seeking solutions for insurance telematics and dynamic risk assessment. According to their website (Mentor eData, 2012), their Mission is to: "Improve driver performance and safety with an automated system that provides routine and objective feedback and assessment of actual driving tendencies and risks." In short, their primary goal is to create safe drivers. They intend to do this via analyzing data collected from their DrivingBuddy application for smart phones. Mentor collects second-by second data with over thirty variables and 35 phases. My job as a Data Analyst was to create a Gold Methodology and a Scoring System that could help explain an Expert/Student classification.

DATA :

The DrivingBuddy application collects a large amount of data. For the simplification of this project, I chose to do the analysis based on the 35 phases that were created. I also communicated with Steve Lakoswke, (head of Mentor eData) to figure out the applicable variables that could be used at the current time. Some variables were eliminated because they do not make sense in context or are not being collected at this time (i.e. Steve plans to collect biometric data in the future). We came up with 21 applicable variables to be used in this initial analysis. **[Table 1 and Table 2]** contain a list of the 21 Applicable Variables used and the 35 phases collected.

Table 1: 21 Applicable Variables

	var	varname	units		var	varname	units
1	4	lat	degrees	12	27	obd_speed	mph
2	5	lon	degrees	13	29	rpm	rpm
3	6	alt	meters	14	35	gasPedal	%
4	7	gps_speed	mph	15	38	massAirFlow	units
5	8	gps_heading	degrees	16	45	userEvent	1=unsafe
6	18	laccelx	g's	17	46	rtEventsid	rtevent_detected
7	19	laccely	g's	18	52	mpg	mpg
8	20	laccelz	g's	19	53	distance	kilometers
9	21	gyrox	degrees/sec	20	54	deltaheading	degrees/sec
10	22	gyroy	degrees/sec	21	55	turnradius	ft
11	23	gyroz	degrees/sec				

Table 2: 35 Phases

	phasenums	phasetitle		phasenums	phase title
1	0	Moderate Straight	19	26	Fast Right Curve Accelerating
2	1	Slow Straight	20	32	Moderate Left Curve
3	2	Fast Straight	21	33	Slow Left Curve
4	3	Idle, Speed=0 mph	22	34	Fast Left Curve
5	4	Moderate Straight Decelerating	23	36	Moderate Left Curve Decelerating
6	5	Slow Straight Decelerating	24	37	Slow Left Curve Decelerating
7	6	Fast Straight Decelerating	25	38	Fast left Curve Decelerating
8	8	Moderate Straight Accelerating	26	40	Moderate Left Curve Accelerating
9	9	Slow Straight Accelerating	27	41	Slow Left Curve Accelerating
10	10	Fast and Accelerating	28	42	Fast Left Curve Accelerating
11	16	Moderate Right Curve	29	80	Moderate Right Turn
12	17	Slow Right Curve	30	81	Slow Right Turn
13	18	Fast Right Curve	31	82	Fast Right Turn
14	20	Moderate Right Curve Decelerating	32	160	Moderate Left Turn
15	21	Slow Right Curve Decelerating	33	161	Slow Left Turn
16	22	Fast Right Curve Decelerating	34	162	Fast Left Turn
17	24	Moderate Right Curve Accelerating	35	255	Idle
18	25	Slow Right Curve Decelerating			

GOLD STANDARD:

Preliminary analysis began by looking at a sample of ten gold driver trips and one student trip. Ultimately, we were looking for a process to see how the student driver compares to the gold drivers. Despite the fact that all eleven trips were made by one driver and is not a large enough sample size, these accumulated trips will be used to simulate the desired process and provide a starting point for creating a methodology.

The purpose of data analysis was to provide feedback about driving safety, based on data collected during a trip. In the longer term, we should be able to assess collision risk from trip data. Statistical methods can be used for estimating risks of rare events from voluminous associated categorical and numerical data. Methods available include mining transactional data (Berberidis *et al*., 2004; Weiss, 2004; Weiss and Hirsh, 1998), Bayesian Network modeling (Cheon *et al*., 2009; Meel and Seider, 2008; Meel and Seider, 2006), Logistic Regression (Maalouf and Trafalis, 2011) and other approaches. All of the cited methods require the occurrence of events within the data set in order to make likelihood estimates. As we began mining trip data to assess driving safety, there were no collision events, so none of the cited approaches could be used.

Unfortunately due to the nature of the data, we were unable to perform typical methods such as Linear Regression or Principal Components Analysis due to the lack of any real response (y) variable. For example, a simple binary variable of crash or no crash is not sufficient in explaining whether someone is a good or bad driver (i.e. a good driver may be involved in an accident but is not at fault; yet on the other hand, a bad driver may drive recklessly, but still avoid an accident). We were also unable to use a simple method like Step-wise Regression in the variable selection process because of this lack of response variable. Instead, we were obligated to initially analyze every variable that is reasonable in context and holds some potential predictive value, only excluding raw variables that are used to create more accurate derived variables (i.e. linear acceleration without the influence of gyros is not very useful).

We proposed to develop statistical models from "Gold Driver" trip data, and then assess student driver safety by comparing a student's trip data with the "Gold Driver" models. We

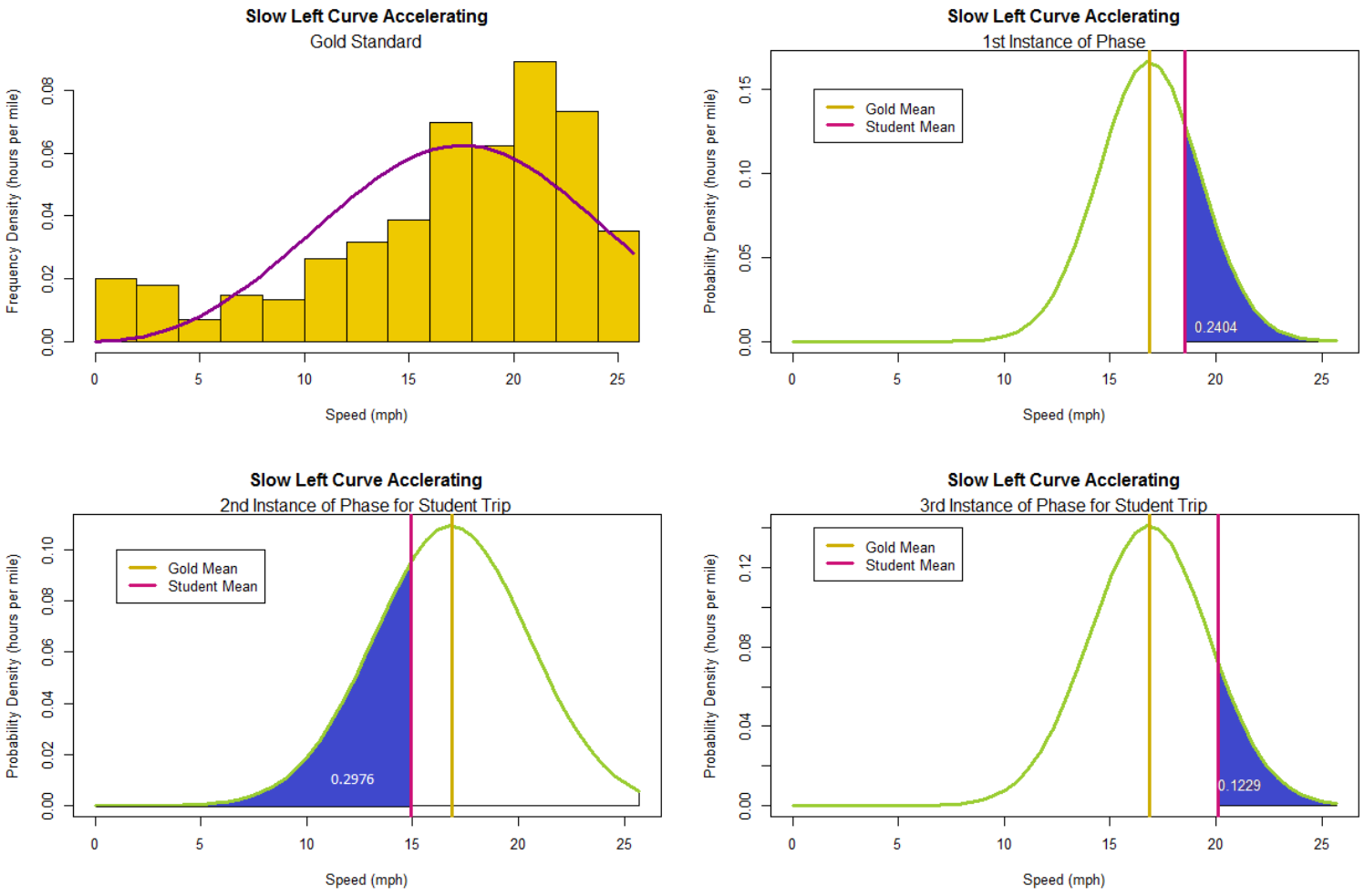
decided to look at phases and chose to use the idea of a phase with different variables during each phase as a base for the analysis.

Our first goal was to combine the gold driver trips and work to create a "Gold Standard" or "Gold Profile" that could be the criterion on which a given student could be compared to. This was done by creating a histogram for each variable of each phase and fitting a model to the data. **[see figure 1: plot 1]**

We viewed the Gold Profile as a population distribution, and a student's trip data as a sample from the population. We suggested that driving safety will diminish whenever the student performs maneuvers that lie in the fringes of the Gold population. For each phase of a student trip, we can determine the location of the phase within the Gold population model using the Central Limit Theorem. We then took data from a student driver's trip and created plots that contain the normal distribution with the gold mean and gold standard deviation divided by the square root of the duration of one instance of a given phase during the student trip [mean=gold mean; std. dev = gold sd/sqrt(n), where n is the duration]. We then plotted each instance of each phase and calculate a corresponding p-value. **[see figure 1: plot 2-4]**

For any sample mean above the Gold mean (as in the first and third instance below), we took 1 - pnorm (sample mean, Normal Mean, Normal St Dev/sqrt (n)) **[see plot 2 and 4].**
But for any sample mean below the Gold mean, we took pnorm (sample mean, Normal Mean, Normal St Dev/sqrt (n). This method applies to the second instance, where the p-value corresponds to the area to the left of the sample mean **[see plot 3].** Through this approach, then the maximum p-value is 0.5

Figure 1:



The next goal was to calculate a p-value for every phase of a student trip where there is a Gold Data model. We started by making a histogram of p-values in the trip, and calculated sample means, and sample st devs. We wanted to try to make a comparison between the sample mean and 0.5. and thought about what more we could do with the p-values for a trip in the future. In the short term, we planned to have the final results use the p-values to make a trip score that ranges from 0% to 100%. The process of implementing these ideas is shown below in **[Figure 2: Implementing Gold Methodology].** The R code is also provided in **Appendix A**.

Figure 2:

Implementing Gold Methodology

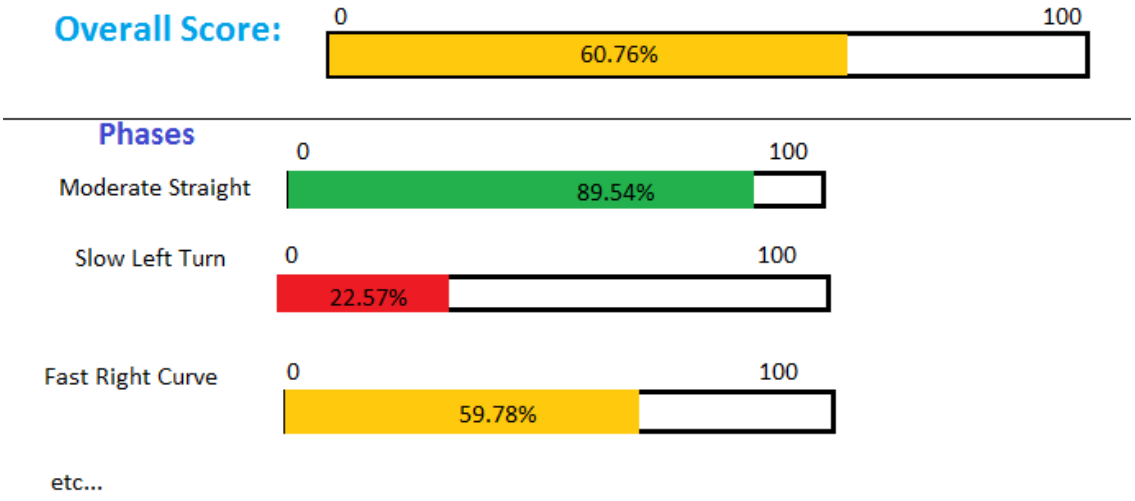
The Process (*in words*)

1. Gold Standard Data is read in
2. The following process is iterated for all phases and variable combinations
    a. corresponding goldmu and goldsd are found
b. Student mean is calculated
    c. n= length of student mean data/number of data points for given phase
    - **z-score** is calculated using: z=(studentmean-goldmu)/(goldsd/(sqrt(n)))
    - This z-score can then be used to find an appropriate p-value, based on the z-score table
        - http://en.wikipedia.org/wiki/Normal_cumulative_distribution_function#Numerical_approximations_for_the_normal_CDF
    - ideally, theses p-values would be stored in a 35x21 matrix
3. Row Statistics could then be performed on this matrix of p-values.
    a. find the mean of the p-values for each row/phase (outcome: 35 means)
    b. multiply each row mean by 2 for a 'phase score' (outcome: 35 scores)
    c. find the (column) mean of phase scores (outcome: 1 score)
    - This single score will equate to an overall score, between 0 and 1 [or a percentage between 0% and 100%].

This was our first shot at a scoring system. We did not do any substantial tests to see how accurate it assesses driving ability. However, a student score of 60.76%, or grade 'D' did not seem to be too far off. In the future, it would be useful to test other 'student driver trips' or even expert trips to see how their scores compare.

See **[Figure 3]** which shows one example (with made up numbers) of what we might provide to users and instructors via the app. That way they could see their overall score and which phases they did well, poorly on etc., to give them a better idea of where their score came from.

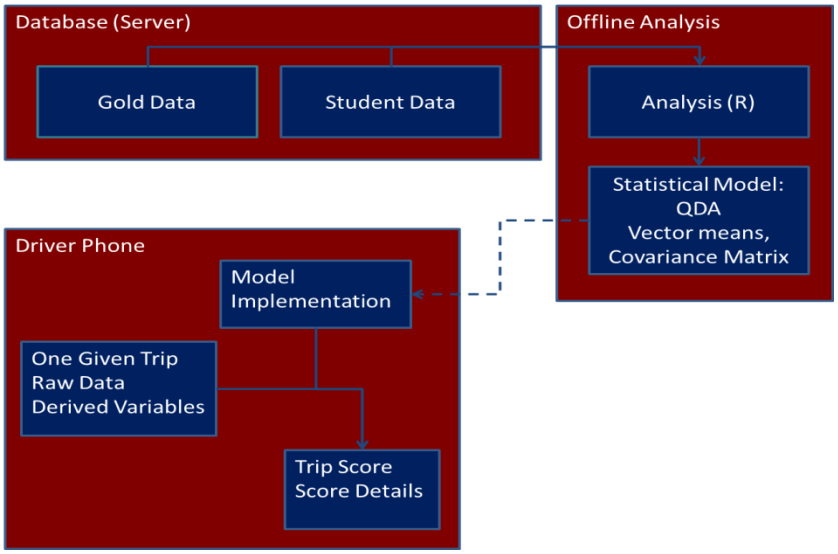
Figure 3:



QAUDRATIC DISCRIMINANT ANALYSIS:

In addition to using the Gold Standard Methodology to assess driver ability, we implemented Quadratic Discriminant Analysis in order to obtain a scoring system. Below, [**Figure 4]** gives an overview of how a score is obtained.

Figure 4:



The statistical model provides a method to answer the question: “Does a trip look more like a Gold Driver or a Student Driver?” That is, for any single trip, data from the trip will be input into the model, and the model will determine the likelihood that the trip was made by a Gold Driver or a Student Driver. Higher (better) trip scores will be produced when the data indicate that the trip is closer to a Gold Driver. The model is based on a method called Quadratic Discriminant Analysis (QDA). This method characterizes the pattern among many variables recorded during a trip; the trip’s pattern characterization is then compared to how the same variables are patterned in both Gold Driver and Student Driver trips. Since many variables are used to characterize trips, QDA provides a highly effective capability to discern a trip’s association with Gold or Student populations.

A QDA model is built from accumulated Gold Driver and Student Driver data. The model is built offline, using R and data stored in a large database. The model produces trip characterizations in the form of data vector Means and Standard Deviations for each population, and a Covariance Matrix inverse. The means, standard deviations and covariance matrix inverse are small, and are stored as part of the application on a driver’s phone/PDA. The driver’s phone/PDA application then takes data from a single trip, and assesses the trip using the stored QDA model values.

The assessment of a single trip will include an overall trip score, in the range 0% to 100%, representing the likelihood that the trip was made by a Gold Driver. In addition to the overall trip score, each phase within the trip can also be assessed individually. Using phase scores, also in the range of 0% to 100%, it will be possible to identify the reason(s) for a low overall trip score. Feedback about which aspects of a driver’s performance are good, and which

aren't so good, can be provided by ranking phase scores. A detailed trip report card can be assembled by showing all phase scores.

This process required six new derived variables based off of our current 21 applicable variables and 35 phases. These six new variables are Min, Max, Time at Min, Time at Max, Ratio of Area for Min, and Ratio of Area for Max. Below **[Figure 5]** shows the time warping process and explains how these derived variables are found.

Figure 5:

How to Time Warp:

given: a vector of values for one instance of phase
how to find the following values:

1. **Min Value:** min(vector)
2. **Max Value:** max(vector)
3. **MinWarpedTime:** $W_{min}=(t-t_0) / (t_n-t_0)$ (where W_{min} corresponds to the warped time (W_i) of the min value of the vector)
4. **MaxWarpedTime:** $W_{max}=(t-t_0) / (t_n-t_0)$
5. **MinRatio:** Ratio=(Area before tmin/ Total Area)
6. **MaxRatio:** Ratio=(Area before tmax/ Total Area)

example:
one instance of (phase 38 + variable 7) @ time t=7-12 = 15.347, 20.789, 17.269, 18.345, 21.975, 20.7
n=6, t=7-12, t_0=7, t_n=12

Real Time (t_i)	7	8	9	10	11	12
Warped Time (W_i)	0	.2	.4 W_{min}	.6	.8 W_{max}	1
Value @ time, w	17.269	20.789	15.347	18.345	21.975	20.7

Min=15.347
Max=21.975
MinWarpedTime=9
MaxWarpedTime=11
Minratio=(17.269+20.789+15.347)/ (114.425) =0.4667
Maxratio=(17.269+20.789+15.347+18.345) / (114.425)= .627

The analysis requires two datasets: one for the expert drivers, and one for the student drivers. Each dataset contains the six variables as well as the corresponding phase and variable. From this we would create a multi-dimensional dataset that is 35 X 126, or 35 x 21 with each cell containing the six derived variables. From this dataset we would extract thirty-five 1x26 vectors for each of the thirty-five phases. We would also have another multi-dimensional dataset for the student data that is identical in structure. Once created one time with accurate student and expert driver data, these datasets would not need to be repeatedly updated. Instead they would serve as the Gold Standard vs. Student Data that is stored on Mentor eData's Database (Server).

After the datasets are stored in the main database, offline analysis can begin. The offline Analysis was done in R. The programming language R is used to find the covariance matrix and the means to be used in the Quadratic Discriminant Function **[Figure 6]**. These covariance matrices and mean vectors only need to be computed once and stored so they can be used when called upon for the scoring process.

Figure 6:

$$g(x) = (\mathbf{x} - \bar{\mathbf{x}}_2)'\mathbf{S}_2^{-1}(\mathbf{x} - \bar{\mathbf{x}}_2) - (\mathbf{x} - \bar{\mathbf{x}}_1)'\mathbf{S}_1^{-1}(\mathbf{x} - \bar{\mathbf{x}}_1) - \ln(|\mathbf{S}_1|/|\mathbf{S}_2|)$$

S1 and **S2=** Gold and Student Covariance Matrices, respectively
(*note* $\boldsymbol{S^{-1}}$ *indicates the inverse covariance matrix*).
x-bar1 and **x-bar2**= mean vectors for Gold and Student, respectively.
x= the 1 x 26 mean vector for each instance of a given phase, for every phase

For any given trip we would obtain n number of g(x) values, (*where n=# of instances of a given phase, for every phases*). See **Figure 6** (Morrison, 2005) for explicit variable explanations. A mean (1x26) vector would be found for each instance of a given phase, for every phase. From this a g-value would be calculated of each instance of a given phase, for all phases. This g(x) value is a log(likelihood ratio). From this we would get a lambda value by taking the exp(g(x)),

so lambda =exp(g(x)). Finally, to obtain a score, we can calculate lambda/(lambda+1)=score. See **Appendix B** for more details.

CONCLUSION:

When applying these methods there are a few things to consider. Prior to my analysis, there was only one current method of scoring based on crash data which was not very accurate. My analysis is a first take on trying to come up with a reliable and explanatory model and scoring system to compare driving ability. Both the gold standard methodology and the scoring system based on the Quadratic Discriminant Analysis are based on phases. These phases were quickly created by Mentor's former programmer, Mike Maggee and may not accurately and fully describe driving behavior. Perhaps these phase definitions could be tested for accuracy and updated if necessary. A more in-depth method of describing driving behavior may be needed in order to provide an accurate base for the methods I created. More could be done with Events or other things that are now or will be collected by the app. The more accurate the data that is being collected the better predictive power of the models.

Future work with this data and trying to accurately assess drivers' ability could include exploring more derived variables. The six derived variables discussed above are initial ideas to work with the Quadratic Discriminant Function, but there are many more potential variables that could be used to replace or be in addition to the current six. Further analysis could be utilized to see which (if not all) variables are useful to the model. Perhaps some variables should have more weight than others. Finally, the validity of the models/methods should be tested to see how they compare to each other, as well as other existing methods.

REFERENCES

Berberidis, Christos, Lefteris Angelis, and Ioannis Vlahavas. "PREVENT: An algorithm for mining intertransactional patterns for the prediction of rare events." Proc. Second Starting AI Researchers' Symposium. Vol. 9. 2004.

Cheon, Seong-Pyo, *et al*. "Bayesian networks based rare event prediction with sensor data." Knowledge-Based Systems 22.5 (2009): 336-343.

Maalouf, Maher, and Theodore B. Trafalis. "Robust weighted kernel logistic regression in imbalanced and rare events data." Computational Statistics & Data Analysis 55.1 (2011): 168-183.

Meel, Anjana, and Warren D. Seider. "Real-time risk analysis of safety systems." Computers & Chemical Engineering 32.4 (2008): 827-840.

Meel, Anjana, and Warren D. Seider. "Plant-specific dynamic failure assessment using Bayesian theory." Chemical engineering science 61.21 (2006): 7036-7056.

Mentor eData.(2012). Enhancing Driver Safety and Predictive Analytics. *Mentor eData*. November 28, 2013.

Morrison, Donald F. (2005) Multivariate Statistical Methods. Belmont, CA: Thomas/Brooks/Cole.

Weiss, Gary M., and Haym Hirsh. "Learning to Predict Rare Events in Event Sequences." KDD. 1998.

Weiss, Gary M. "Mining with rarity: a unifying framework." ACM SIGKDD Explorations Newsletter 6.1 (2004): 7-19.

Appendix A:

```
##hypertables csv.files (named alyssa.zip) are read in;
#following corresponding line converts Unix time to real time;
#student driver;
student= read.table(unz("alyssa.zip", "b217.csv"), header=T, sep=",");
    student$realtime=as.POSIXct(c(student$datetime), origin="1970-01-01", tz="GMT");
#gold ("expert") driver;
gold1= read.table(unz("alyssa.zip", "g208.csv"), header=T, sep=",");
    gold1$realtime=as.POSIXct(c(gold1$datetime), origin="1970-01-01", tz="GMT");
gold2= read.table(unz("alyssa.zip", "g239.csv"), header=T, sep=",");
    gold2$realtime=as.POSIXct(c(gold2$datetime), origin="1970-01-01", tz="GMT");
gold3= read.table(unz("alyssa.zip", "g296.csv"), header=T, sep=",");
    gold3$realtime=as.POSIXct(c(gold3$datetime), origin="1970-01-01", tz="GMT");
gold4= read.table(unz("alyssa.zip", "g306.csv"), header=T, sep=",");
    gold4$realtime=as.POSIXct(c(gold4$datetime), origin="1970-01-01", tz="GMT");
gold5= read.table(unz("alyssa.zip", "g401.csv"), header=T, sep=",");
    gold5$realtime=as.POSIXct(c(gold5$datetime), origin="1970-01-01", tz="GMT");
gold6= read.table(unz("alyssa.zip", "g404.csv"), header=T, sep=",");
    gold6$realtime=as.POSIXct(c(gold6$datetime), origin="1970-01-01", tz="GMT");
gold7= read.table(unz("alyssa.zip", "g437.csv"), header=T, sep=",");
    gold7$realtime=as.POSIXct(c(gold7$datetime), origin="1970-01-01", tz="GMT");
gold8= read.table(unz("alyssa.zip", "g455.csv"), header=T, sep=",");
    gold8$realtime=as.POSIXct(c(gold8$datetime), origin="1970-01-01", tz="GMT");
gold9= read.table(unz("alyssa.zip", "g500.csv"), header=T, sep=",");
    gold9$realtime=as.POSIXct(c(gold9$datetime), origin="1970-01-01", tz="GMT");
gold10= read.table(unz("alyssa.zip", "g528.csv"), header=T, sep=",");
    gold10$realtime=as.POSIXct(c(gold10$datetime), origin="1970-01-01", tz="GMT");

###creating a matrix for VARIABLE info###
var=c(4,5,6,7,8,18,19,20,21,22,23,27,29,35,38,45,46,52,53,54,55)
applicable=as.data.frame(var)
phasenamesapp=gold1[c(4,5,6,7,8,18,19,20,21,22,23,27,29,35,38,45,46,52,53,54,55)]
applicablephasenames=names(phasenamesapp)
applicable[,"varname"]=applicablephasenames
unit=c("degrees","degrees","meters", "mph", "degrees", "g's", "g's","g's", "degrees/sec",
    "degrees/sec", "degrees/sec","mph","rpm","%", "units",
    "1=unsafe","rtevent_detected","mpg","kilometers", "degrees/sec", "ft")
applicable[,"units"]=unit
revunit=c("degrees","degrees","meters", "hours per mile", "degrees", "g's", "g's","g's",
    "sec/degrees", "sec/degrees", "sec/degrees","hours per mile","minutes per revolution","%",
    "units", "1=unsafe","rtevent_detected","gallons per mile","kilometers", "sec/degrees", "ft")
applicable[,"reverseunits"]=revunit

###creating a matrix for PHASE info###
```

```
phasenums=c(0,1,2,3,4,5,6,8,9,10,16,17,18,20,21,22,24,25,26,32,33,34,36,37,38,40,41,42,80,81,
    82,160,161,162,255)
PhaseNums=as.data.frame(phasenums)

for(itor in 1:35){
if (PhaseNums[itor,1]=="0"){
        PhaseNums[itor,2]="Moderate Straight"
    } else if (PhaseNums[itor,1]=="1"){
        PhaseNums[itor,2]="Slow Straight"
    } else if (PhaseNums[itor,1]=="2"){
        PhaseNums[itor,2]="Fast Straight"
    } else if (PhaseNums[itor,1]=="3"){
        PhaseNums[itor,2]="Idle, Speed=0 mph"
    } else if (PhaseNums[itor,1]=="4"){
        PhaseNums[itor,2]="Moderate Straight Decelerating"
    } else if (PhaseNums[itor,1]=="5"){
        PhaseNums[itor,2]="Slow Straight Decelerating"
    } else if (PhaseNums[itor,1]=="6"){
        PhaseNums[itor,2]="Fast Straight Decelerating"
    } else if (PhaseNums[itor,1]=="8"){
        PhaseNums[itor,2]="Moderate Straight Accelerating"
    } else if (PhaseNums[itor,1]=="9"){
        PhaseNums[itor,2]="Slow Straight Accelerating"
    } else if (PhaseNums[itor,1]=="10"){
        PhaseNums[itor,2]="Fast and Accelerating"
    } else if (PhaseNums[itor,1]=="16"){
        PhaseNums[itor,2]="Moderate Right Curve"
    } else if (PhaseNums[itor,1]=="17"){
        PhaseNums[itor,2]="Slow Right Curve"
    } else if (PhaseNums[itor,1]=="18"){
        PhaseNums[itor,2]="Fast Right Curve"
    } else if (PhaseNums[itor,1]=="20"){
        PhaseNums[itor,2]="Moderate Right Curve Decelerating"
    } else if (PhaseNums[itor,1]=="21"){
        PhaseNums[itor,2]="Slow Right Curve Decelerating"
    } else if (PhaseNums[itor,1]=="22"){
        PhaseNums[itor,2]="Fast Right Curve Decelerating"
    } else if (PhaseNums[itor,1]=="24"){
        PhaseNums[itor,2]="Moderate Right Curve Accelerating"
    } else if (PhaseNums[itor,1]=="25"){
        PhaseNums[itor,2]="Slow Right Curve Decelerating"
    } else if (PhaseNums[itor,1]=="26"){
        PhaseNums[itor,2]="Fast Right Curve Accelerating"
    } else if (PhaseNums[itor,1]=="32"){
        PhaseNums[itor,2]="Moderate Left Curve"
    } else if (PhaseNums[itor,1]=="33"){
```

```
        PhaseNums[itor,2]="Slow Left Curve"
    } else if (PhaseNums[itor,1]=="34"){
        PhaseNums[itor,2]="Fast Left Curve"
    } else if (PhaseNums[itor,1]=="36"){
        PhaseNums[itor,2]="Moderate Left Curve Decelerating"
    } else if (PhaseNums[itor,1]=="37"){
        PhaseNums[itor,2]="Slow Left Curve Decelerating"
    } else if (PhaseNums[itor,1]=="38"){
        PhaseNums[itor,2]="Fast left Curve Decelerating"
    } else if (PhaseNums[itor,1]=="40"){
        PhaseNums[itor,2]="Moderate Left Curve Accelerating"
    } else if (PhaseNums[itor,1]=="41"){
        PhaseNums[itor,2]="Slow Left Curve Accelerating"
    } else if (PhaseNums[itor,1]=="42"){
        PhaseNums[itor,2]="Fast Left Curve Accelerating"
    } else if (PhaseNums[itor,1]=="80"){
        PhaseNums[itor,2]="Moderate Right Turn"
    } else if (PhaseNums[itor,1]=="81"){
        PhaseNums[itor,2]="Slow Right Turn"
    } else if (PhaseNums[itor,1]=="82"){
        PhaseNums[itor,2]="Fast Right Turn"
    } else if (PhaseNums[itor,1]=="160"){
        PhaseNums[itor,2]="Moderate Left Turn"
    } else if (PhaseNums[itor,1]=="161"){
        PhaseNums[itor,2]="Slow Left Turn"
    } else if (PhaseNums[itor,1]=="162"){
        PhaseNums[itor,2]="Fast Left Turn"
    } else if (PhaseNums[itor,1]=="255"){
        PhaseNums[itor,2]="Idle"
    }
}

names(PhaseNums)[2]="phasetitle"


#creates a data frame with all means and then for the standard deviations;
means=matrix(nrow=35, ncol=21)
rownames(means)=PhaseNums$phasetitle
colnames(means)=applicable$varname
standard=matrix(nrow=35, ncol=21)
rownames(standard)=PhaseNums$phasetitle
colnames(standard)=applicable$varname

for(vitor in 1:21){
for(pitor in 1:35){
```

```
phasenum=PhaseNums[pitor,1]
var=applicable[vitor,1]
varname=applicable[vitor,2]
maintitle=PhaseNums[pitor,2]
labelx=applicable[vitor,3]
reverselablex=applicable[vitor,4]
labely=c(paste("Frequency Density (",reverselablex,")"))
labely2=c(paste("Probability Density (",reverselablex,")"))

#creating subsets for the phase for each gold driver trip;
phase.gold1=gold1[gold1$phase==phasenum,]
phase.gold2=gold2[gold2$phase==phasenum,]
phase.gold3=gold3[gold3$phase==phasenum,]
phase.gold4=gold4[gold4$phase==phasenum,]
phase.gold5=gold5[gold5$phase==phasenum,]
phase.gold6=gold6[gold6$phase==phasenum,]
phase.gold7=gold7[gold7$phase==phasenum,]
phase.gold8=gold8[gold8$phase==phasenum,]
phase.gold9=gold9[gold9$phase==phasenum,]
phase.gold10=gold10[gold10$phase==phasenum,]

#extracts gps_speed
one=phase.gold1[,var]
two=phase.gold2[,var]
three=phase.gold3[,var]
four=phase.gold4[,var]
five=phase.gold5[,var]
six=phase.gold6[,var]
seven=phase.gold7[,var]
eight=phase.gold8[,var]
nine=phase.gold9[,var]
ten=phase.gold10[,var]

#creates a vector that contains data from all gold driver trips
puregold=c(one, two, three, four, five, six, seven, eight, nine, ten)

#finds student mean;
#phasestud=student[student$phase==phasenum,]
#studentmean=mean(phasestud[,var])

goldensort=sort(puregold)
x=goldensort
mu=mean(goldensort)
n=length(goldensort)
#stddev=sd(goldensort)
stddev=(sd(goldensort))/(sqrt(n))
```

```
means[pitor,vitor]=mu;
standard[pitor,vitor]=stddev;
}
}

#creates a table for the means and standard deviations for the gold standard
write.csv(means, "gold.mean.csv")
write.csv(standard, "gold.stddev.csv")

write.csv(PhaseNums, "Phases.csv")
write.csv(applicable, "Variables.csv")
```

Appendix B:

```
setwd("G:/Mentor");

#reads in the two data sets for "good" and "bad" drivers;
baddrive=read.csv("baddrivers.csv")
bad=(baddrive[,-1])
badvector=factor(c(rep("bad",1606)))
bad[,"Driver"]=badvector
gooddrive=read.csv("gooddrivers.csv")
good=(gooddrive[,-1])
goodvector=factor(c(rep("good",1606)))
good[,"Driver"]=goodvector

#recombines good and bad driver data;
QDAta=rbind(bad,good)


Q.Data.split=split(QDAta[,1:6], QDAta$Driver)

meanbar=lapply(Q.Data.split, colMeans, na.rm=TRUE)
S=lapply(lapply(Q.Data.split, var, na.rm=TRUE),solve)

#student-gold
#student=x2
#gold=x1
#lambda/1+lambda

studentarray=array(c(20,17,.67,.348,1.547,3.9))
studentarray=array(c(40,39,.55,.56,1.8,4.5))
studentarray=array(c(34.52,31.985,.5908,.58365,1.6,3.21666))
studentarray=array(c(15,12,.3,.2,.6,.4))

studenttranspose=as.matrix(t(studentarray-meanbar$bad))
stud=as.matrix((studentarray-meanbar$bad))
Sstudinv=solve(S$bad)
S1=S$bad
goldtranspose=t(studentarray-meanbar$good)
gold=(studentarray-meanbar$good)
Sgoldinv=solve(S$good)
S2=S$good
gOFx=(studenttranspose%*%Sstudinv%*%stud)-(goldtranspose%*%Sgoldinv%*%gold)-
(log((det(S1))/(det(S2))))
lambda=exp(log(abs(gOFx)))
lambda/(1+lambda)
```