DATA SCIENCE PORTFOLIO

Previous Research and Programming Application Examples

Jessica Gisela Lieblich

2019

DATA SCIENCE PORTFOLIO TABLE OF CONTENTS

Previous Research Projects

Additional Programing Application Examples

Previous Research: Predicting Price and Overall Satisfaction of Airbnb Listings in Los Angeles County In SAS Enterprise Miner

ABSTRACT

The goals of these analyses are to predict the price per night and overall satisfaction of each Airbnb listing in the sample of data taken from http://insideairbnb.com/get-the-data.html. We analyze Airbnb data from Los Angeles County, with variables including the following: *room id, host id, room type, neighborhood, overall satisfaction, number of accommodates, number of bedrooms, year, and price*. The method of this paper is to use SAS Enterprise Miner to optimize and interpret Decision Tree, Neural Network, and Regression models. The first step of this method is to effectively create and clean the dataset, and the final step is to compare the optimal models with one another in order to draw conclusions, as well as analyzing the Ensemble Tool. There are two target variables, price and overall satisfaction; therefore, we complete the process twice with models that predict each target variable separately. Additionally, we draw from the conclusions combined for predicting each target variable, as they are each important factors in modeling one another. The models predicting price could assist Airbnb in raising prices, while the models predicting overall satisfaction could help Airbnb increase its customer satisfaction. We further discuss these real-world possibilities in the "CONCLUSIONS" section.

For the Decision Trees that model overall satisfaction, the optimal model is a CART-like Decision Tree that only leads to variable importance and is not easily interpretable. Variable importance from the Decision Tree predicting overall satisfaction is price, neighborhood, accommodates, and then room type, Year, and bedrooms. The Neural Network predicting overall satisfaction does not overfit the data, and thus it confirms the variable importance found from the overfitting Decision Tree also predicting overall satisfaction. The accuracy of the Neural Network modeling overall satisfaction is about 63.5%. Recall that variable importance for the Decision Tree modeling overall satisfaction is price, neighborhood, accommodates, room type, year and bedrooms. We draw further conclusions about the input parameters and their relationships from the Regression equation for predicting overall satisfaction. We see that the year with the highest overall satisfaction is 2014. We compare this with the conclusion drawn that the year with the highest price is 2015, with the knowledge that the most important variable from the decision tree for predicting overall satisfaction rating is price. We also see from the regression equation modeling overall satisfaction that, generally, the higher the price, the higher the overall satisfaction. We see that entire homes and apartments increase overall satisfaction roughly the same as do private rooms. We see that the number of bedrooms is a factor that increases overall satisfaction, while number of accommodates increasing, decreases overall satisfaction rating. This may be because there are more people to have the opportunity to give bad reviews, or because staying in an Airbnb/room with another person is generally less satisfying (hypothetically more annoying, etc.)

For the Decision Trees that model price, the best model is a ProbF Decision Tree that both leads to variable importance conclusions, and is easily interpretable. Variable importance from the Decision Tree predicting price is bedrooms, room type, year, number of accommodates neighborhood, and overall satisfaction. Some basic interpretations from the node rules from the Decision Tree modeling price include the following:

If room type IS ONE OF: ENTIRE HOME/APT AND bedrooms =1 then Predicted: price = $155.90956364 per night, if room type IS ONE OF: PRIVATE ROOM, SHARED ROOM AND bedrooms =1 then Predicted: price = $85.671898734 per night.

From contrasting these two nodes, we conclude that entire homes/apartments are generally more expensive than are private and shared rooms (single rooms).

If bedrooms = 2 AND Year IS ONE OF: 2014, 2016 then Predicted: price = $232.37313433 per night, if bedrooms = 2 AND Year IS 2015 then Predicted: price = $318.461855 per night.

We conclude from these two node rules that 2015 listings generally have higher prices than do 2014 and 2016.

If bedrooms = 2 or 3 then Predicted: price = $386.87704918 per night, if bedrooms = 4 Predicted: price =$659.10625 per night, if bedrooms >= 5 then Predicted: price = $1851.09523 per night. Finally, we see that when number of bedrooms increases, predicted price also increases. We learn about input variable relationships of variables that factor into the price through these node rules. We then compare with the Regression equation for predicting price and make further conclusions. We expect price to increase by $25.91 if year is 2015, which confirms the decision tree interpretation. We see that we expect price to increase about $21 per one-person increase in number of accommodates. When later combining this knowledge with the fact that increasing the number of accommodates decreases overall satisfaction, this makes sense because the customer is paying more for the same Airbnb listing, simply because more people are staying there. Important findings are that with each additional bedroom we expect price to increase about $85, with each additional unit in overall satisfaction rating we expect price to increase about $31, and that price is generally higher when room type is entire home/apartment.

INTRODUCTION

There are two motivations for these analyses. These research goals include:

1) **Predict price**: price per night of each listing
2) **Predict overall_satisfaction:** The average rating (out of five) that the listing has received from those visitors who left a review

We collect the data from the public Airbnb website, www.insideairbnb.com/get-thedata. We choose to analyze Airbnb data from Los Angeles County. There are numerous .csv files on the Airbnb website, each representing a single set of data. The data collection dates include (*each a separate .csv file)*: 2014-09-01 (10027 observations), 2014-09-18 (11317 observations), 2014-10-25 (11780 observations), 2015-04-06 (13899 observations), 2015-08-03 (15790 observations), 2015-08-27 (16649 observations), 2015-09-09 (16968 observations), 2015-10-13 (17420 observations), 2015-11-16 (18665 observations), 2015-12-10 (19709 observations), 2016-01-11 (18863 observations), 2016-02-10 (19311 observations), 2016-03-12 (19717 observations), 2016-04-12 (20951 observations), 2016-05-14 (21986 observations), 2016-06-14
(21996 observations), 2016-07-13 (23071 observations), 2016-08-15 (23793 observations), 2016-09-13 (24744 observations), 2016-10-14 (25347 observations), 2016-11-17 (24633 observations), 2016-12-17 (30735 observations).

-The variables (and their descriptions) include:

1) room_id: unique ID of an Airbnb listing. Each listing has a URL at http://airbnb.com/rooms/room_id,
2) host_id: unique ID of an Airbnb host. Each host's page has a URL at http://airbnb.com/users/show/host_id,
3) room_type: Either "Entire home/apt", "Private room", or "Shared room,"
4) neighborhood: a subregion of the city or search area
5) reviews: The number of reviews that a listing has received. The website states that 70% of visits end up with a review, so the number of reviews can be used to estimate the number of visits. (Note that estimating this way will not be reliable for individual observations, but if analyze by grouping by borough, it might be a useful metric)
6) overall_satisfaction: The average rating (out of five) that the listing has received from those visitors who left a review,
7) accommodates: The number of guests a listing can accommodate,
8) bedrooms: The number of bedrooms a listing offers,
9) year: 2014, 2015, or 2016
10) price: The price for a night stay

DATA CLEANING

We initially randomly select (see SAS code in Appendix) 6,000 observations from each year (of three years, 2014-2016), randomly choosing three surveys from each year, as 2014 only consists of three datasets. We add an extra variable of year (2014, 2015, or 2016). We randomly select 2,000 observations from each survey, and thus have 18,000 observations in the final dataset. Because the initial datasets (listed in Introduction) are so large, we were able to delete observations with any missing values, and then take a random sample 2,000 from each. Therefore, we have no missing data. We then merge all of these datasets into the final dataset. There are no imputations, nor illogical observations found when exploring the final dataset; therefore our data cleaning process is complete.

see APPENDIX on final pages for SAS CODE of data cleaning

METHODS FOR DECISION TREES

Predicting Overall Satisfaction

C4.5 vs. CART Split-Search Algorithms

"The split worth for the Entropy (C4.5) and Gini (CART) options are calculated as shown below. Let a set of cases S be partitioned into p subsets S1, ..., Sp such that…

$$S = \bigcup_{i=1}^{p} S_i$$

Let the number of cases in S equal N and the number of cases in each subset S_i equal n_i. Then the worth of a particular partition of S is given by the following:

$$worth = I(S) - \sum_{i=1}^{p} w_i I(S_i)$$

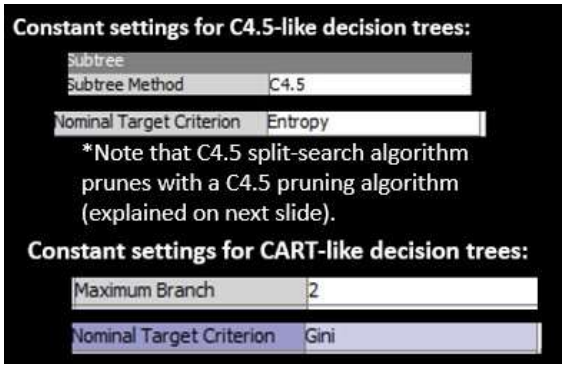
where $w_i = n_i/N$ (the proportion of cases in subset S_i), and for the specified split worth measure, $I(\cdot)$ has the following value:

$$I(\cdot) = \sum_{classes} p_{class} \cdot \log_2 p_{class} \qquad \text{Entropy}$$

$$I(\cdot) = 1 - \sum_{classes} p_{class}^2 \qquad \text{Gini}$$

Because Gini reduction and entropy reduction criteria tend to grow enormous trees, pruning and selecting a tree complexity based on validation do limit this problem to some extent. The predictive model sequence consists of subtrees obtained from the maximal tree by removing splits. The first batch of subtrees is obtained by Pruning (that is, removing) one split from the maximal tree. This process results in several models with one less leaf than the maximal

tree. The subtrees are compared using a model-rating statistic calculated on validation data. The subtree with the best validation assessment statistics is selected to represent a particular level of model complexity. A similar process is followed to obtain the next batch of subtrees. Each time the subtrees are formed by removing two splits, then three splits, four splits, etc. from the maximal tree. The process is repeated until subtrees represent all levels of complexity. C4.5 split-search algorithm prunes with a C4.5 pruning algorithm. The CART-like trees use the Misclassification pruning algorithm, as the target variable is set to a variable of categorical-type." (1)



Validation Assessment

The general principle of addressing model complexity optimization is creating a sequence of related models of increasing model complexity from training data, and using validation data to select the optimal model. The 'best' decision tree is the model with the lowest model complexity, with the highest validation performance. High validation performance assessment for these Decision Trees is average square error…

Suppose the target variable has L outcomes given by $(C_1, C_2, \ldots C_L)$, then

$$Average\ square\ error = \frac{1}{N \cdot L} \sum_{i=1}^{N} \sum_{j=1}^{L} \left(I\left(y_i = C_j\right) - \hat{p}_{ij} \right)^2$$

where, for the i^{th} case, y_i is the actual target value, $\hat{p}_{ij}$ is the predicted target value for the j^{th} class, and the following:

$$I\left(y_i = C_j\right) = \begin{cases} 1 & y_i = C_j \\ 0 & y_i \neq C_j \end{cases}$$

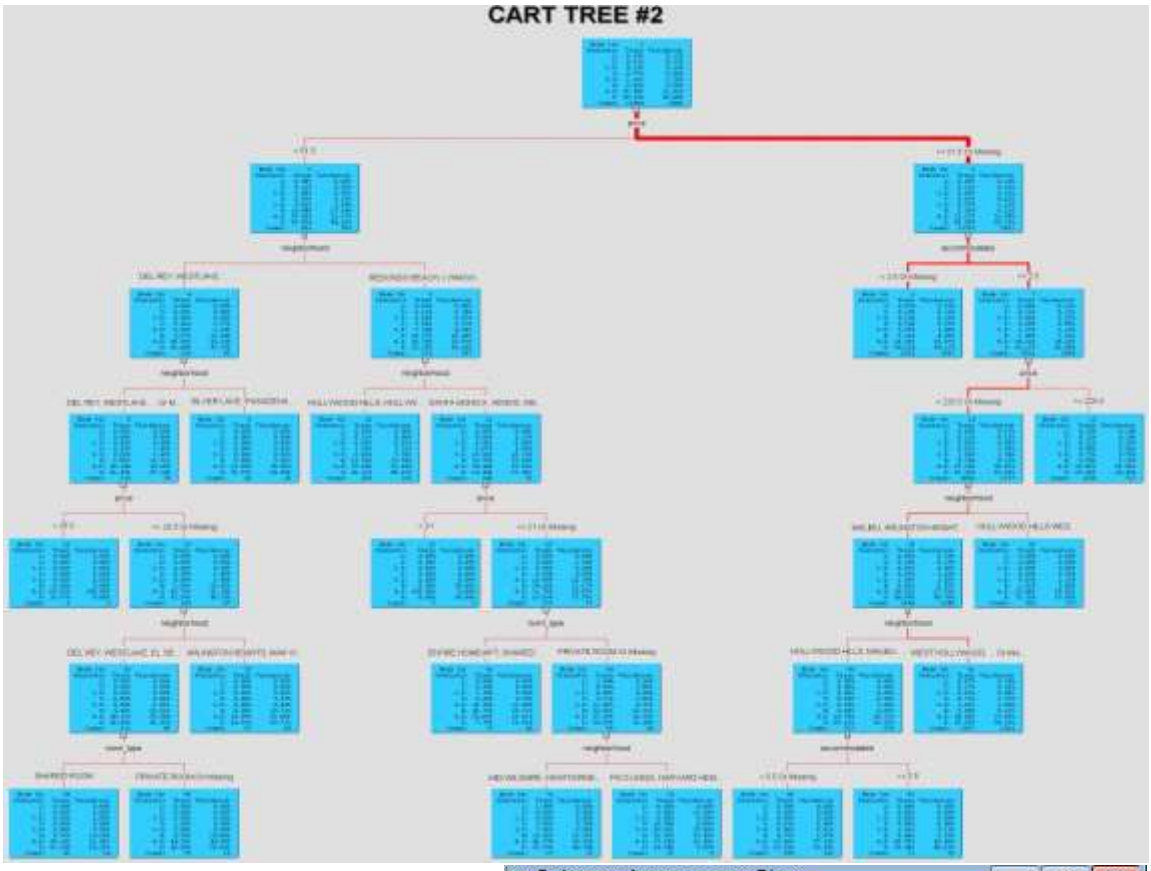
Constant Settings for Basic Decision Tree Predicting Price

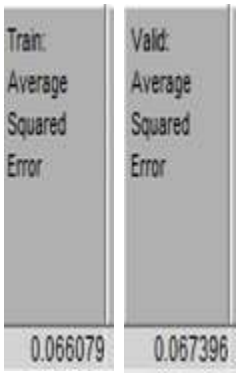
Splitting Rule	
Interval Target Criterion	ProbF
Nominal Target Criterion	Gini
Ordinal Target Criterion	Entropy
Significance Level	0.2
Missing Values	Use in search
Use Input Once	No
Maximum Branch	2
Maximum Depth	3
Minimum Categorical Size	2
Node	
Leaf Size	5
Number of Rules	5

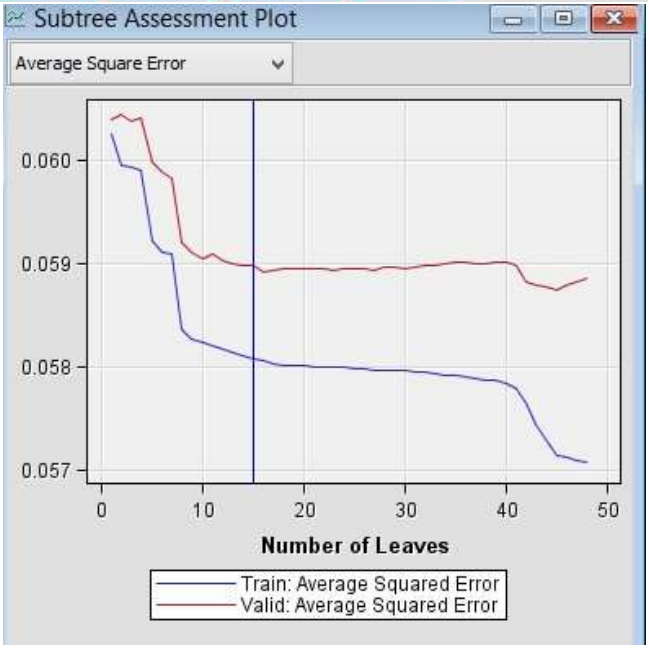
RESULTS FOR DECISION TREES Predicting Overall Satisfaction

The first research goal is to predict overall_satisfaction, or the average rating (from one to five) that each Airbnb listing has received. We use a nominal target, instead of an ordinal target, because pruning with a C4.5 algorithm is only available for nominal targets. We use the same target level type for each model predicting overall_satisfaction in order to accurately compare all models. The following settings turn out to result in the optimal model.

Nominal Target Criterion	Gini
Ordinal Target Criterion	Gini
Significance Level	0.2
Missing Values	Use in search
Use Input Once	No
Maximum Branch	2
Maximum Depth	6
Minimum Categorical Size	5
Node	
Leaf Size	5
Number of Rules	2
Number of Surrogate Rules	0
Split Size	.







The subtree assessment plot shows the Average Square Error corresponding to each subtree as the data is sequentially split. This plot confirms suspicions about the optimality of the a more complex tree. The performance on the training sample becomes monotonically better as the tree becomes more complex. However, the performance on

the validation sample only improves up to a tree of, approximately, seven or eight leaves, and then diminishes as model complexity increases. (The leaf size setting is set to 5 for the optimal model). Over the range of one to approximately seven leaves, the precision of the model improves with the increase in complexity. A marginal increase in complexity over this range results in better accommodation of the systematic variation or signal in data. “The validation performance shows evidence of model overfitting. Precision diminishes as complexity increases past this range; the additional complexity accommodates idiosyncrasies in the training sample, and the model extrapolates less well.” (1)

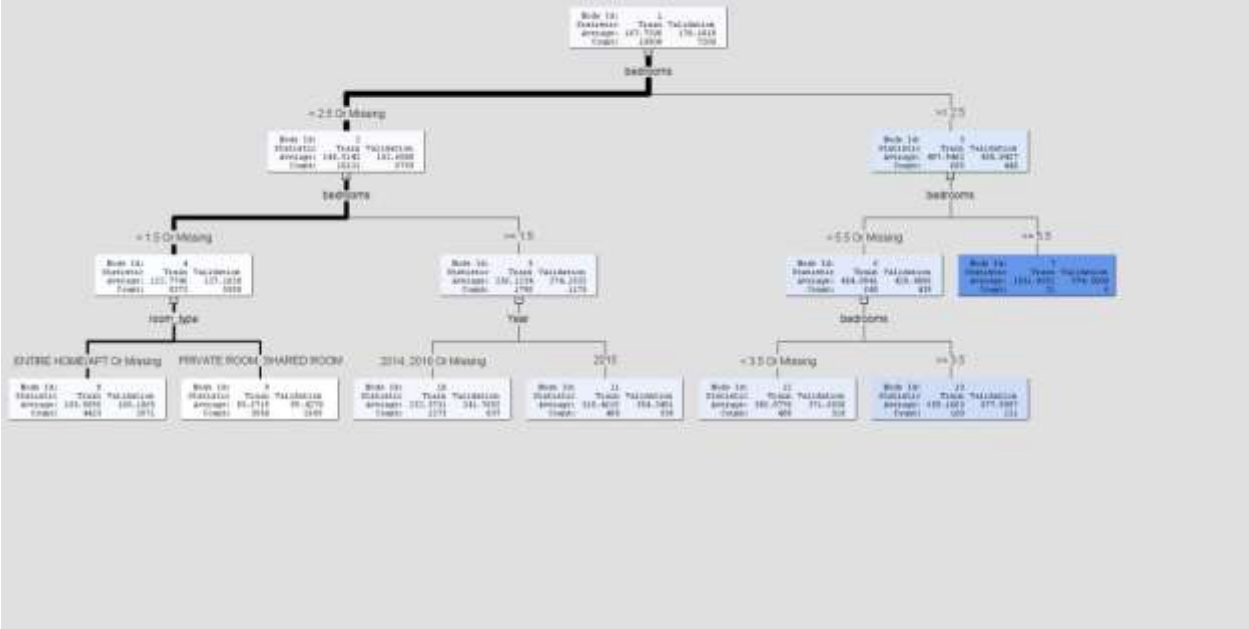
Var Name:	Number of Splitting Rules:	Importance:	Validation Importance:	Ratio of Validation to Training Importance:
Price	4	1	1	1
Neigborhood	7	.9751	.7508	.779
Accommodates	2	.8704	.7504	1.193
Room_type	2	.247	.2316	.8375
Year	0	0	0	0
Bedrooms	0	0	0	0

It seems that variable importance for this tree (predicting overall_satisfaction) is price, then neighborhood, accommodates, room_type, Year, and bedrooms (with the main three being price, neighborhood, and accommodates: Nodes 9 and 29 have the *highest proportion of predicted overall_satisfaction score of ‘5’* of all nodes. We conclude from node 9 that if price is less than $51.50 per night and neighborhood is one of those listed below, then the decision tree predicts that (of the 43 observations that meet these criterion) 79% will have the highest possible value for overall_satisfaction variable. We conclude from node 29 that if price is greater than or equal to $51.50 per night and less than 220.5 per night, and neighborhood is one of those listed below, and accommodates is greater than 2, then the decision tree predicts that (of the 362 observations that meet these criterion) 74% will have the highest possible value (optimal) for the overall_satisfaction variable. Therefore, *when analyzing the variable importance from this decision tree, combined with the nodes with the highest proportion of the optimal response*, we notice that price, neighborhood, and accommodates are the most important factors when aiming for the optimal overall_satisfaction value. The CART-like decision tree is difficult to interpret beyond these variable importance measures because a main input variable is neighborhood and there are so many in Los Angeles County.

Because the validation performance shows evidence of model overfitting, for future research, in this case the succeeding models of Neural Networks and Regression, will benefit from learning about the variable importance when predicting overall_satisfaction. Reducing the model complexity of future models with this knowledge may reduce risk of overfitting with these models.

Predicting Price

Research Goal is to predict price: price per night of each listing.

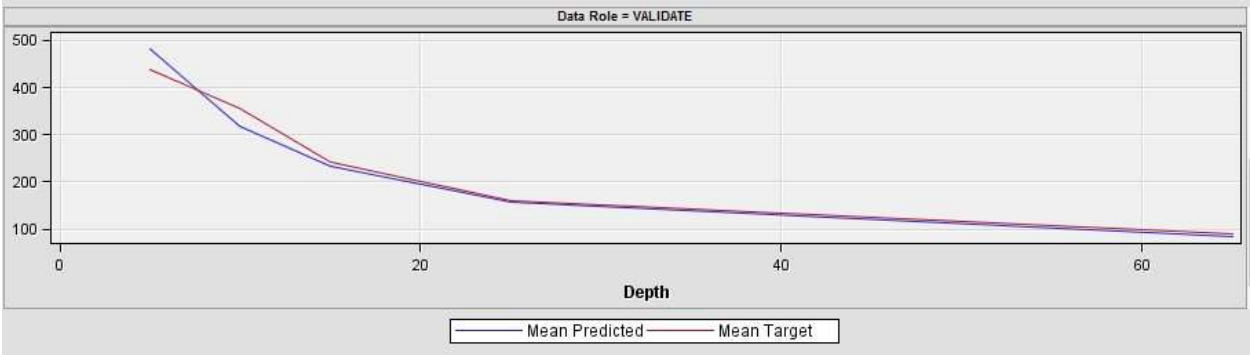


Unlike the Decision Tree for predicting overall_satisfaction, the Decision Tree for predicting price is easily interpretable…

Node = 7: **if bedrooms >= 5 then Number of Observations = 21 & Predicted: price = $1851.09523 per night** Node = 8: **if room_type IS ONE OF: ENTIRE HOME/APT AND bedrooms < 2 then Number of Observations = 4423 & Predicted: price = $155.90956364 per night** Node = 9: **if room_type IS ONE OF: PRIVATE ROOM, SHARED ROOM AND bedrooms < 2 then Number of Observations = 3950 & Predicted: price = $85.671898734 per night** Node = 10**: if bedrooms = 2 AND Year IS ONE OF: 2014, 2016 then Number of Observations = 1273 & Predicted: price = $232.37313433 per night** Node = 11 **if bedrooms = 2 AND Year IS 2015 then Number of Observations = 485 & Predicted: price = $318.461855 per night**
Node = 12: **if bedrooms < 4 AND bedrooms >= 2 then Number of Observations = 488 & Predicted: price = $386.87704918 per night** Node = 13: **if bedrooms < 5 AND bedrooms >= 4 then Number of Observations = 160 & Predicted: price =$659.10625 per night**

Fit Statistics

Target	Target Label	Fit Statistics	Statistics Label	Train	Validation
price		_NOBS_	Sum of Frequencies	10800	7200
price		_MAX_	Maximum Absolute Err...	14681.54	9914.328
price		_SSE_	Sum of Squared Errors	9.7296E8	7.7254E8
price		_ASE_	Average Squared Error	90061.31	107296.6
price		_RASE_	Root Average Squared...	300.1022	327.5617
price		_DIV_	Divisor for ASE	10800	7200
price		_DFT_	Total Degrees of Free...	10800	.



The root average square error is about $300, so the model is not very accurate. However, it still yields useful results which lead to variable importance conclusions that will assist in future analyses of the data (Neural Networks and Regression).

Variable Importance

Variable Name	Label	Number of Splitting Rules	Importance	Validation Importance	Ratio of Validation to Training Importance
bedrooms		4	1.0000	1.0000	1.0000
room_type		1	0.2600	0.3545	1.3633
Year		1	0.1308	0.2296	1.7563
accommod...		0	0.0000	0.0000	.
neighborho...		0	0.0000	0.0000	.
overall_sati...		0	0.0000	0.0000	.

Number of bedrooms, room type, year, number of accommodates, neighborhood, and overall satisfaction ranking are important in the prediction of price. This knowledge can assist in creating a much more accurate model, that is not a Decision Tree, and can lead to more conclusions.

METHODS FOR NEURAL NETWORKS Predicting Overall Satisfaction

Due to the variable importance knowledge gained from the Decision Trees, we now only include the important input variables into the Neural Network in order to reduce risk of overfitting. Overfitting is a major obstacle in the optimization of any Neural Network that's main goal is prediction accuracy.

Name	Role	Level	Report	Order	Drop
Year	**Input**	**Nominal**	**No**		**No**
accommodates	**Input**	**Interval**	**No**		**No**
bedrooms	**Input**	**Interval**	**No**		**No**
host_id	**ID**	**Nominal**	**No**		**Yes**
latitude	**Input**	**Interval**	**No**		**Yes**
longitude	**Input**	**Interval**	**No**		**Yes**
neighborhood	**Input**	**Nominal**	**No**		**No**
overall_satisfact	**Target**	**Nominal**	**No**		**No**
price	**Input**	**Interval**	**No**		**No**
reviews	**Input**	**Interval**	**No**		**Yes**
room_id	**ID**	**Nominal**	**No**		**Yes**
room_type	**Input**	**Nominal**	**No**		**No**

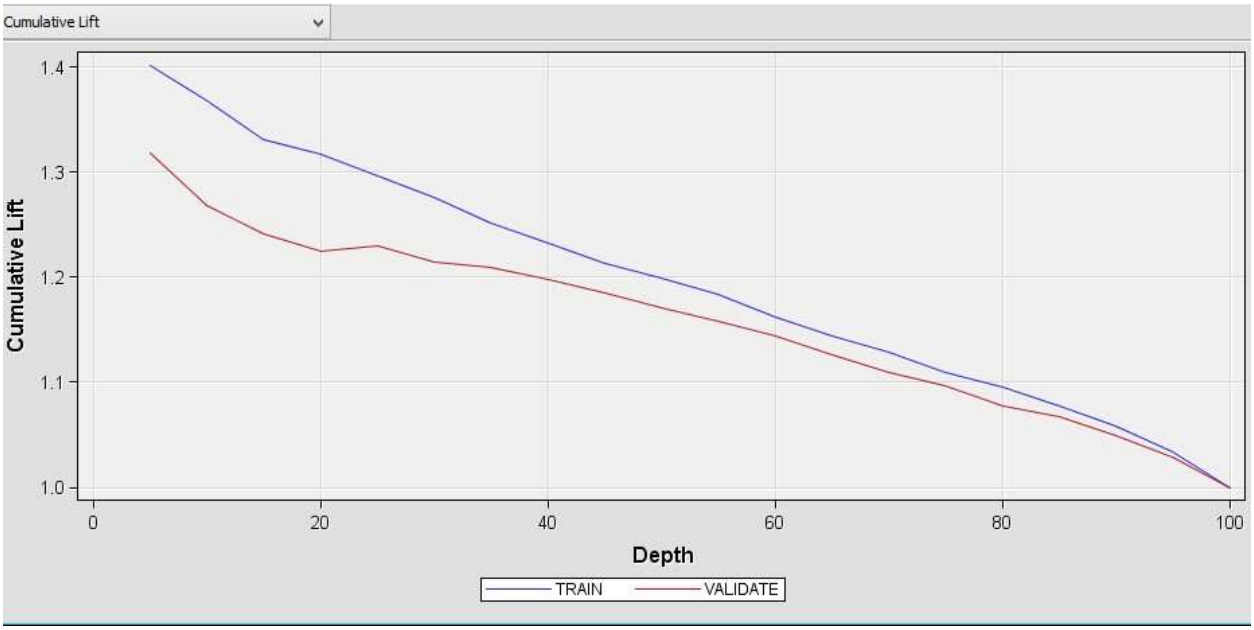
We determine that DBLDog (Double-Dogleg) is the best optimization algorithm for the Neural Network. The Double-Dogleg optimization method combines the ideas of quasi-Newton and trust region methods. The double dogleg algorithm computes in each iteration the step $s^{(k)}$ as the linear combination of the steepest descent or ascent search direction $s_1^{(k)}$ and a quasi-Newton search direction s_2 ,

(k)

$$s^{(k)} = \alpha_1 s_1^{(k)} + \alpha_2 s_2^{(k)}$$

Selected Model	Predecessor Node	Model Node	Model Description	Target Variable	Target Label	Selection Criterion: Valid: Misclassification Rate
Y	Neural7	Neural7	NN DBLDog	overall_sati...		0.395003
	Neural3	Neural3	NN Quasi ...	overall_sati...		0.395559
	Neural9	Neural9	NN QProp	overall_sati...		0.397641
	Neural2	Neural2	NN default	overall_sati...		0.400833
	Neural6	Neural6	NN Conjug...	overall_sati...		0.400833
	Neural	Neural	NN BackProp	overall_sati...		0.403609
	Neural8	Neural8	NN RProp	overall_sati...		0.405135

There is evidence of overfitting. Up to about 15 training iterations, the precision of the Neural Network improves. These first fifteen iterations show better accommodation to noise. We will therefore only use the important variables (discovered in Decision Tree for Overall Satisfaction), in the final model.



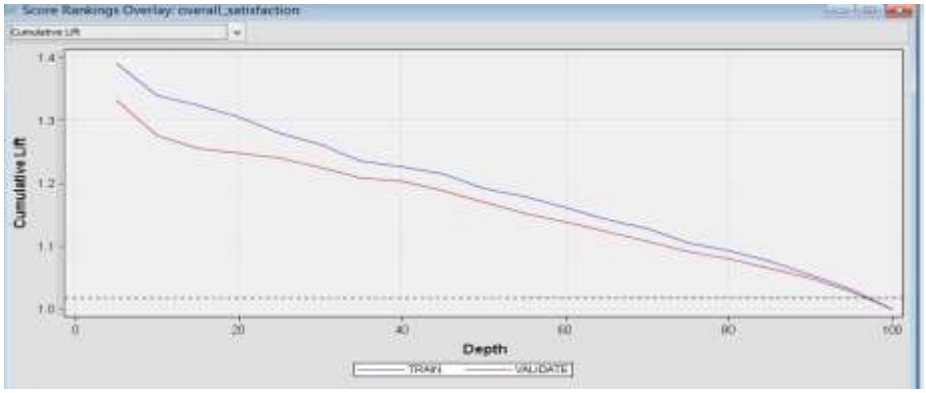
Now, we experiment with the Architecture of the Neural Networks in order to determine one that is optimal. We decide Multilayer Perceptron is the best architecture, and must then chose the number of units in the hidden layer. It shows that five units is optimal.

Selected Model	Predecessor Node	Model Node	Model Description	Target Variable	Target Label	Selection Criterion: Valid: Misclassification Rate
Y	Neural	Neural	NN Percept...	overall_sati...		0.395003
	Neural2	Neural2	NN Ordinar...	overall_sati...		0.401666
	Neural3	Neural3	NN Ord. Ra...	overall_sati...		0.406246

Fit Statistics

Selected Model	Predecessor Node	Model Node	Model Description	Target Variable	Target Label	Selection Criterion: Valid: Misclassification Rate	Train: Total Degrees of Freedom	Train: Degrees of Freedom for Error	Train: Model Degrees of Freedom	Train: Number of Estimated Weights	Train: Akaike's Information Criterion	Train: Schwarz's Bayesian Criterion	Train: Average Squared Error	Train: Maximum Absolute Error	Train: Divisor for ASE	Train: Sum of Frequencies	Train: Root Average Squared Error	Train: Sum of Squared Errors	Train: Sum of Case Weights Times Freq	Train: Final Prediction Error	Train: Mean Squared Error	Train: Root Final Prediction Error	Train: Root Mean Squared Error	Train: Average Error Function
Y	Neural2	Neural2	NN 5 units	overall_sati...		0.391572	96368	85282	1078	1078	22025.73	32122.58	0.058619	0.998808	97155	10795	0.237847	5690.807	87155	0.05885	0.057335	0.242638	0.239446	0.204519
	Neural	Neural	NN 3 units	overall_sati...		0.395003	96368	85716	650	650	21968.5	27146.59	0.058464	0.998838	97155	10795	0.237621	5685.747	87155	0.05732	0.056882	0.239417	0.238521	0.203391
	Neural3	Neural3	NN 8 units	overall_sati...		0.39868	96368	84640	1720	1720	23762.88	39872.88	0.05814	0.998815	97155	10795	0.241123	5648.697	87155	0.060503	0.056322	0.245974	0.24359	0.206172

We see that there is no longer clear evidence of model overfitting after changing the architecture of the Neural Network with the optimal algorithm.



We repeat this process, but standardize the input variables, in order to increase the predictive power even more. This does not improve our model.

Default Methods	
Interval Inputs	Standardize
Interval Targets	None
Class Inputs	None
Class Targets	None
Treat Missing as Level	No

Selected Model	Predecessor Node	Model Node	Model Description	Target Variable	Target Label	Selection Criterion: Valid: Misclassification Rate
	Neural2	Neural2	NN 5 units	overall_sati...		0.391672
	Neural	Neural	NN 3 units	overall_sati...		0.395003
	Neural3	Neural3	NN 8 units	overall_sati...		0.39889

Predicting Price

Again, due to the variable importance knowledge gained from the Decision Trees, we now only include the important input variables into the Neural Network, in order to reduce risk of overfitting. Overfitting is a major obstacle in the optimization of a Neural Network.

Recall that number of bedrooms, room type, year, number of accommodates, neighborhood, and overall satisfaction were important variables in predicting price, concluded from the decision tree model.

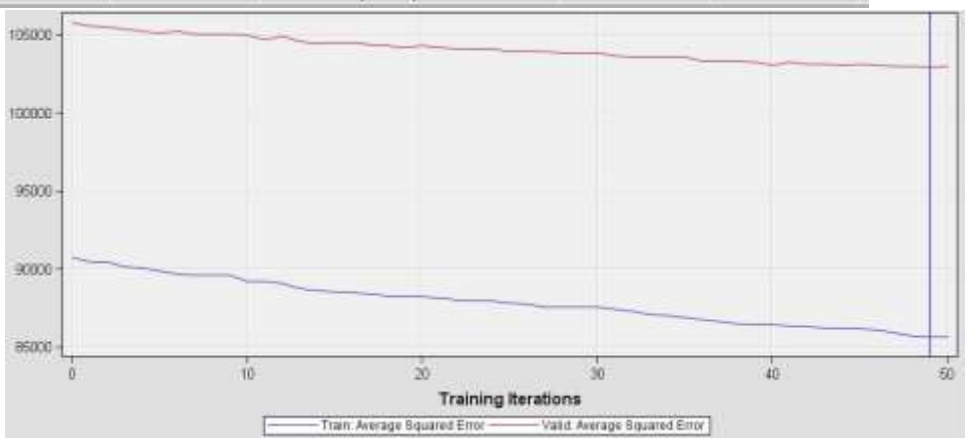
Name	Role	Level	Report	Order	Drop
accommodates	Input	Interval	No		No
bedrooms	Input	Interval	No		No
host_id	ID	Nominal	No		No
latitude	Input	Interval	No		Yes
longitude	Input	Interval	No		Yes
neighborhood	Input	Nominal	No		No
overall_satisfact	Input	Nominal	No		No
price	Target	Interval	No		No
reviews	Input	Interval	No		Yes
room_id	ID	Nominal	No		Yes
room_type	Input	Nominal	No		No
Year	Input	Nominal	No		No

We, again, determine that DBLDog (Double-Dogleg) is the best optimization algorithm for the Neural Network. The Double-Dogleg optimization method combines the ideas of quasiNewton and trust region methods. The double dogleg algorithm

computes in each iteration the step $s^{(k)}$ as the linear combination of the steepest descent or ascent search direction $s_1^{(k)}$ and a quasi-Newton search direction $s_2^{(k)}$,

$$s^{(k)} = \alpha_1 s_1^{(k)} + \alpha_2 s_2^{(k)}$$

Selected Model	Predecessor Node	Model Node	Model Description	Target	Target Label	Selection Criterion: Valid: Average Squared Error
Y	Neural5	Neural5	NN DBLDog	price		102936.5
	Neural	Neural	NN default	price		102969.1
	Neural4	Neural4	NN Conjug...	price		102969.1
	Neural3	Neural3	NN Quasi ...	price		103043.1
	Neural6	Neural6	NN RProp	price		103304.6
	Neural2	Neural2	NN BackProp	price		105795.8
	Neural7	Neural7	NN QProp	price		105795.8



The Neural Network for predicting price does not have a significant decrease in error than does the Decision Tree for predicting price.

Target	Target Label	Fit Statistics	Statistics Label	Train	Validation
price		_DFT_	Total Degrees of Free...	10800	.
price		_DFE_	Degrees of Freedom f...	10157	.
price		_DFM_	Model Degrees of Fre...	643	.
price		_NW_	Number of Estimated ...	643	.
price		_AIC_	Akaike's Information C...	123952.6	.
price		_SBC_	Schwarz's Bayesian C...	128638.4	.
price		_ASE_	Average Squared Error	85649.67	102936.5
price		_MAX_	Maximum Absolute Err...	14262.41	9934.724
price		_DIV_	Divisor for ASE	10800	7200
price		_NOBS_	Sum of Frequencies	10800	7200
price		_RASE_	Root Average Squared...	292.6597	320.8372
price		_SSE_	Sum of Squared Errors	9.2502E8	7.4114E8
price		_SUMW_	Sum of Case Weights ...	10800	7200
price		_FPE_	Final Prediction Error	96493.97	.
price		_MSE_	Mean Squared Error	91071.82	102936.5
price		_RFPE_	Root Final Prediction ...	310.6348	.
price		_RMSE_	Root Mean Squared E...	301.7811	320.8372
price		_AVERR_	Average Error Function	85649.67	102936.5
price		_ERR_	Error Function	9.2502E8	7.4114E8

Therefore, we must manipulate the Architecture of the Neural Networks in order to determine one that is optimal. We decide Perceptron is the best architecture and must then chose the number of units in the hidden layer. It shows that eight units is optimal.

Selected Model	Predecessor Node	Model Node	Model Description	Target	Target Label	Selection Criterion: Valid: Average Squared Error
	Neural11	Neural11	NN Percept...	price		102936.5
	Neural10	Neural10	NN Ord. Ra...	price		105702.3
	Neural9	Neural9	NN Ordinar...	price		107564.1

Selected Model	Predecessor Node	Model Node	Model Description	Target	Target Label	Selection Criterion: Train: Average Squared Error
Y	Neural14	Neural14	NN 8 units	price		91104.78
	Neural12	Neural12	NN 3 units	price		92462.33
	Neural13	Neural13	NN 5 units	price		93233.15

We then attempt to increase the predictive power of the Neural Network, by standardizing input variables. This slightly improves the model.

Default Methods	
Interval Inputs	Standardize
Interval Targets	None
Class Inputs	None
Class Targets	None
Treat Missing as Level	No

Selected Model	Predecessor Node	Model Node	Model Description	Target	Target Label	Selection Criterion: Valid: Average Squared Error
Y	Neural13	Neural13	NN 5 units	price		102868.4
	Neural14	Neural14	NN 8 units	price		102873.8
	Neural12	Neural12	NN 3 units	price		103096.2

RESULTS FOR NEURAL NETWORKS Predicting Overall Satisfaction

Average Squared Error of Training set is about .0566 and of Validation set is about .0578 *(these are lower/better than for the decision tree)*

Misclassification Rate of Training set is about .385 and of Validation set is about .392

The accuracy for the predictions, from the model on the training set, against the original dataset is 63.6%

The accuracy for the predictions, from the model on the validation set, against the original dataset is 63.1%

Event Classification Table

Data Role=TRAIN Target=overall_satisfaction Target Label=' '

False Negative	True Negative	False Positive	True Positive
411	854	3517	6013

Data Role=VALIDATE Target=overall_satisfaction Target Label=' '

False Negative	True Negative	False Positive	True Positive
269	527	2393	4016

Target Profile

Ordered Value	overall_satisfaction	Total Frequency
1	5	6424
2	4.5	3389
3	4	744
4	3.5	128
5	3	77
6	2.5	9
7	2	13
8	1.5	2
9	1	9

Target	Target Label	Fit Statistics	Statistics Label	Train	Validation
overall_satisfaction		_MISC_	Misclassification Rate	0.3849	0.391672
overall_satisfaction		_ASE_	Average Squared Error	0.056619	0.057844

Predicting Price

The Neural Network for modeling price is not *significantly* more accurate than is the decision tree for predicting price (as the average squared error are almost exactly the same for both models). We cannot draw further conclusions than with the decision tree. There is a relatively high average square error, and differences between the training and validation set results.

Target	Target Label	Fit Statistics	Statistics Label	Train
price		_DFT_	Total Degrees of Free...	18000
price		_DFE_	Degrees of Freedom f...	16239
price		_DFM_	Model Degrees of Fre...	1761
price		_NW_	Number of Estimated ...	1761
price		_AIC_	Akaike's Information C...	209077.8
price		_SBC_	Schwarz's Bayesian C...	222810.3
price		_ASE_	Average Squared Error	91104.78
price		_MAX_	Maximum Absolute Err...	14139.95
price		_DIV_	Divisor for ASE	18000
price		_NOBS_	Sum of Frequencies	18000
price		_RASE_	Root Average Squared...	301.8357
price		_SSE_	Sum of Squared Errors	1.6399E9
price		_SUMW_	Sum of Case Weights ...	18000
price		_FPE_	Final Prediction Error	110864.1
price		_MSE_	Mean Squared Error	100984.4
price		_RFPE_	Root Final Prediction ...	332.9626
price		_RMSE_	Root Mean Squared E...	317.7805
price		_AVERR_	Average Error Function	91104.78
price		_ERR_	Error Function	1.6399E9

METHODS FOR REGRESSION Predicting Overall Satisfaction

Model Fit Statistics

R-Square	0.0152	Adj R-Sq	0.0142
AIC	-19514.2079	BIC	-19512.1812
SBC	-19426.7603	C(p)	12.0000

Type 3 Analysis of Effects

Effect	DF	Sum of Squares	F Value	Pr > F
Year	2	1.3312	4.06	0.0173
accommodates	1	10.8256	66.02	<.0001
bedrooms	1	4.4969	27.42	<.0001
price	1	3.1622	19.28	<.0001
room_type	2	8.3920	25.59	<.0001
Year*room_type	4	2.5243	3.85	0.0040

Analysis of Maximum Likelihood Estimates

Parameter			DF	Estimate	Standard Error	t Value	Pr > |t|
Intercept			1	4.7107	0.00988	476.79	<.0001
Year	2014		1	0.0277	0.0108	2.58	0.0100
Year	2015		1	-0.00549	0.0112	-0.49	0.6227
accommodates			1	-0.0241	0.00296	-8.13	<.0001
bedrooms			1	0.0369	0.00704	5.24	<.0001
price			1	0.000056	0.000013	4.39	<.0001
room_type	Entire home/apt		1	0.0424	0.00843	5.03	<.0001
room_type	Private room		1	0.0577	0.00862	6.70	<.0001
Year*room_type	2014	Entire home/apt	1	-0.0275	0.0115	-2.40	0.0166
Year*room_type	2014	Private room	1	-0.0170	0.0122	-1.39	0.1634
Year*room_type	2015	Entire home/apt	1	-0.00512	0.0119	-0.43	0.6662
Year*room_type	2015	Private room	1	-0.0193	0.0124	-1.56	0.1183

Predicted Overall Satisfaction Average Rating from One to Five = 4.71 + .0277*(1 if Year = 2014) - .0241*(# of accommodates) + .0369*(# of bedrooms) + .000056*(Price) + .0424*(1 if room type = Entire home/apartment) + .0577*(1 if room type = private room)

The R squared value is extremely low, at only about 1.5% of the variation of the response of overall satisfaction that the regression model accounts for. When we attempt to transform the variables using the Transform Variables node, as well as using polynomial terms in SAS Enterprise Miner, the R squared value does improve, but not significantly.

Model Fit Statistics

R-Square	0.0154	Adj R-Sq	0.0144
AIC	-22957.6348	BIC	-22955.6036
SBC	-22853.4544	C(p)	14.0000

Type 3 Analysis of Effects

Effect	DF	Sum of Squares	F Value	Pr > F
Year	2	1.8593	5.76	0.0032
accommodates	1	7.5718	46.88	<.0001
bedrooms	1	5.1289	31.75	<.0001
room_type	2	9.9994	30.95	<.0001
Year*room_type	4	3.4022	5.27	0.0003
accommodates*accommodates	1	2.4578	15.22	<.0001
accommodates*bedrooms	1	0.3782	2.34	0.1260
bedrooms*bedrooms	1	0.0000	0.00	0.9902

Predicting Price

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	7	141325160	20189309	218.66	<.0001
Error	10792	996467814	92334		
Corrected Total	10799	1137792974			

Model Fit Statistics

R-Square	0.1242	Adj R-Sq	0.1236
AIC	123486.2006	BIC	123488.2125
SBC	123544.4991	C(p)	8.0000

Type 3 Analysis of Effects

Effect	DF	Sum of Squares	F Value	Pr > F
Year	2	5319500.87	28.81	<.0001
accommodates	1	8425986.87	91.26	<.0001
bedrooms	1	24720376.2	267.73	<.0001
overall_satisfaction	1	1788985.34	19.38	<.0001
room_type	2	9651080.97	52.26	<.0001

Analysis of Maximum Likelihood Estimates

Parameter		DF	Estimate	Standard Error	t Value	Pr > |t|
Intercept		1	-180.3	34.7120	-5.19	<.0001
Year	2014	1	3.1720	4.1532	0.76	0.4450
Year	2015	1	25.9115	4.2157	6.15	<.0001
accommodates		1	21.0961	2.2084	9.55	<.0001
bedrooms		1	85.2490	5.2101	16.36	<.0001
overall_satisfaction		1	31.7491	7.2129	4.40	<.0001
room_type	Entire home/apt	1	52.8140	6.2344	8.47	<.0001
room_type	Private room	1	-11.7042	6.3972	-1.83	0.0673

Predicted Price per Night of Airbnb Listing = -$180.30 + $25.9115*(1 if Year=2015) + $21.0961*(# of accommodates) + $85.249*(# of bedrooms) + $31.7491*(overall satisfaction average rating from one to five) + $52.814*(1 if room type= Entire Home/Apartment)

There are limitations to the interpretation of this regression equation, as the price could turn out to be negative. This is a common issue of a regression model with such a small R squared value. When we attempt to transform the variables in SAS, and use a polynomial model, and we do achieve a *slightly* higher R squared value.

Model Fit Statistics

R-Square	0.1269	Adj R-Sq	0.1257
AIC	145870.8501	BIC	145872.9016
SBC	146004.7962	C(p)	18.0000

Type 3 Analysis of Effects

Effect	DF	Sum of Squares	F Value	Pr > F
Year	2	917121.258	4.31	0.0135
accommodates	1	1103332.02	10.36	0.0013
bedrooms	1	3350.6782	0.03	0.8592
overall_satisfaction	1	582537.299	5.47	0.0193
room_type	2	8811318.86	41.38	<.0001
Year*room_type	4	1815423.29	4.26	0.0019
accommodates*accommodates	1	200354.907	1.88	0.1702
accommodates*bedrooms	1	6252.4189	0.06	0.8085
accommodates*overall_satisfaction	1	1304284.66	12.25	0.0005
bedrooms*bedrooms	1	7272028.71	68.30	<.0001
bedrooms*overall_satisfaction	1	43295.9262	0.41	0.5237
overall_satisfaction*overall_satisfaction	1	467044.480	4.39	0.0362

RESULTS FOR REGRESSION Predicting Overall Satisfaction

The regression model that predicts overall satisfaction reveals a little more about the variable importance discovered by the decision trees (through interpretations of parameters), and this is discussed in "CONCLUSIONS."

Predicted Overall Satisfaction Average Rating from One to Five = 4.71 + .0277*(1 if Year = 2014) - .0241*(# of accommodates) + .0369*(# of bedrooms) + .000056*(Price) + .0424*(1 if room type = Entire home/apartment) + .0577*(1 if room type = private room) - .0275*(1 if Year = 2014*1 if room type= Entire home/apartment)

A limitation of both of the optimal Regression models (predicting each target variable) is that many of the variables do not meet the assumption of Linear Regression models that they do not contain multi-collinearity. Input variables that do not meet this assumption include 1) number of bedrooms and room type, 2) price and number of bedrooms (for predicting overall satisfaction), 3) overall satisfaction and number of accommodates (for predicting price), etc.

Predicting Price

The regression model that predicts price reveals a little more about the variable importance discovered by the decision trees (through interpretations of parameters), but has about the same predictive power of both the Decision Tree and the Neural Network modeling price. All models have a relatively high average error of about $300. The R-squared value for the optimal regression model is only about 12.3%, so the model does not account for a large amount of variation in the response. Another limitation of this model is that we had to exclude neighborhood as an important variable, even though we know from the decision tree that it is important, as it did not increase the validity of our model *enough to make up for the impossible parameter effect-interpretations that including the input variable requires.*

Predicted Price per Night of Airbnb Listing = -$180.30 + $25.9115*(1 if Year=2015) + $21.0961*(# of accommodates) + $85.249*(# of bedrooms) + $31.7491*(overall satisfaction average rating from one to five) + $52.814*(1 if room type= Entire Home/Apartment)

There also are limitations to the interpretation of this regression equation, as the price could turn out to be negative. This is a common issue of a regression model with such a small R squared value. When we attempt to transform the variables in SAS, and use a polynomial model, and we do achieve a *slightly* higher R squared value.

METHODS FOR ENSEMBLE

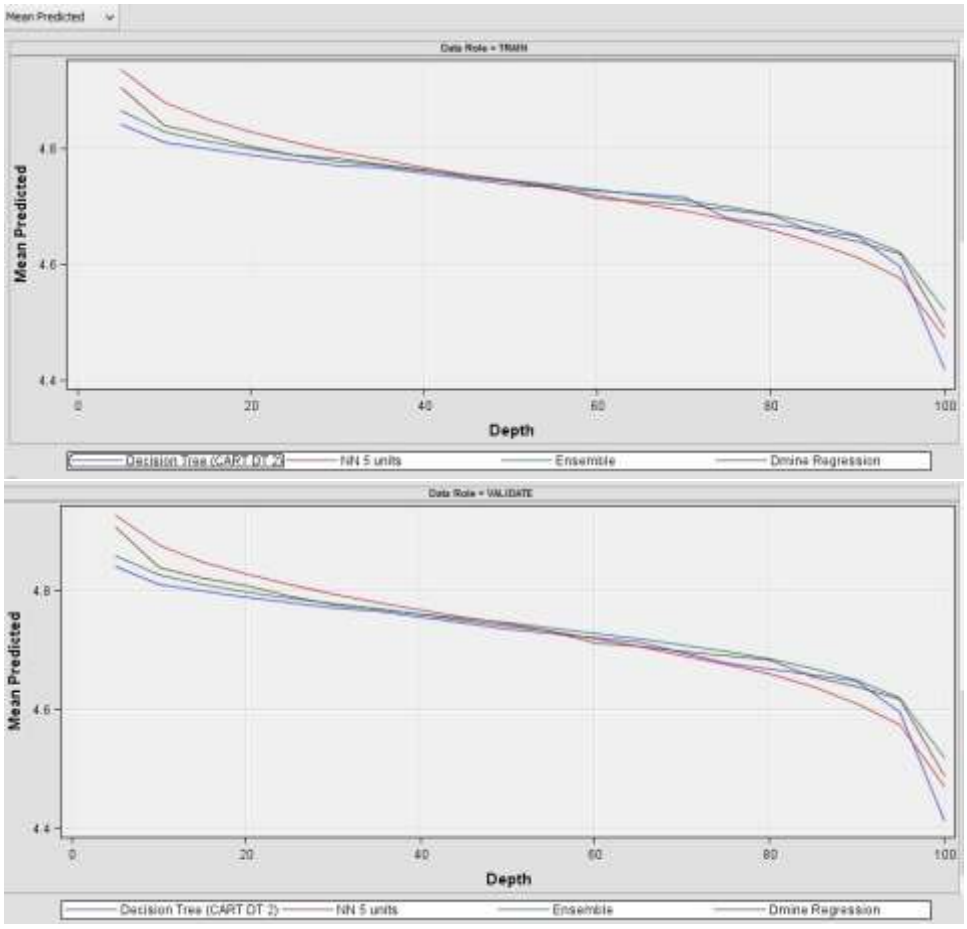
"The Ensemble node is not itself a model. It merely combines model predictions… Run the Model Comparison node (in SAS Enterprise Miner) and view the results. This enables you to see how the ensemble model compares to the individual models." (1)

RESULTS FOR ENSEMBLE Predicting Overall Satisfaction

Model Selection based on Valid: Average Squared Error (_VASE_)

Selected Model	Model Node	Model Description	Valid: Average Squared Error	Train: Average Squared Error
Y	Ensmbl	Ensemble	0.15581	0.15212
	Neural	NN 5 units	0.15709	0.15300
	Tree	Decision Tree (CART DT 2)	0.15770	0.15673
	DmineReg	Dmine Regression	0.16149	0.15535

The ensemble node shows that it is similar to other models that model overall satisfaction. This also confirms that the Neural Network modeling overall satisfaction is more accurate than is the Decision Tree, and that the Regression model has a very low R squared value.

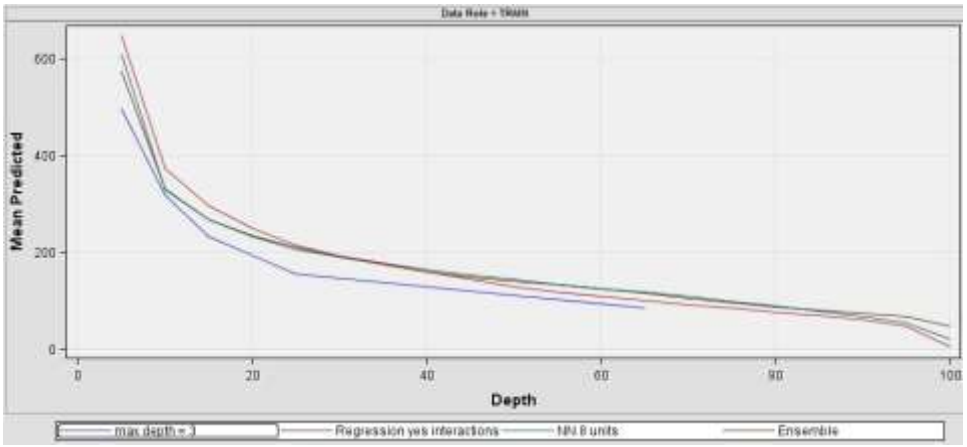


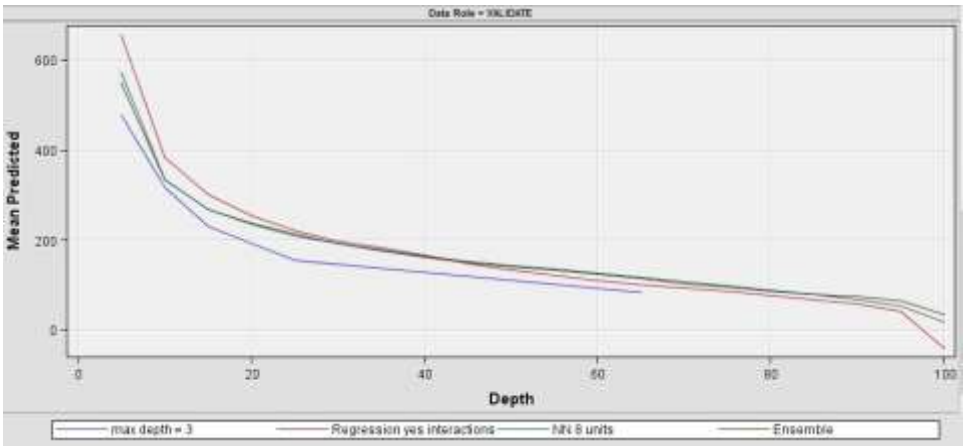
Predicting Price

Model Selection based on Valid: Average Squared Error (_VASE_)

Selected Model	Model Node	Model Description	Valid: Average Squared Error	Train: Average Squared Error
Y	Ensmbl	Ensemble	103441.28	82820.70
	Neural	NN 8 units	103689.32	83318.37
	Tree	max depth = 3	107296.65	90061.31
	Reg	Regression yes interactions	107844.32	81906.52

The ensemble node shows that the ensemble "model" is slightly better, yet still similar, to other models that model price. It also again establishes that the standardized Neural Network modeling price is a better model than is the Decision Tree, and that the Linear Regression model is very weak.





CONCLUSIONS

Model Comparisons & Conclusions Drawn for Modeling Overall Satisfaction

Variable importance from the Decision Tree for predicting overall satisfaction is price, then neighborhood, number accommodates, room type, year, and number bedrooms. A limitation of the regression model predicting overall satisfaction is that we cannot include the important input variable of neighborhood, because if we did include it then the regression model cannot provide more about variable importance *in an interpretable way* as it currently does. The R squared value is extremely low, with only about 1.5% of the variation of the response of overall satisfaction that the regression model accounts for. Another limitation is that there are many input variables with multicollinearity (for example, number of bedrooms and room type, number of accommodates and price, number of bedrooms and price, etc.) Yet, the model provides information about parameter relationships between the input variables, that are not analyzed with the Neural Network as it provides only "black box" predictions. Note that the Neural Network does not overfit, as does the Decision Tree that provides the variable importance.

Predicted Overall Satisfaction Average Rating from One to Five = 4.71 + .0277*(1 if Year = 2014) - .0241*(# of accommodates) + .0369*(# of bedrooms) + .000056*(Price) + .0424*(1 if room type = Entire home/apartment) + .0577*(1 if room type = private room)

This equation provides us with further information about the relationship between input variables that are important in predicting overall satisfaction. A main additional conclusion is that entire home/apartments and private rooms are better contributors to a high overall satisfaction rating than are shared rooms. It also reveals that people generally gave better reviews in year 2014, probably because Airbnb was new (four years since founded, gaining popularity, no "horror stories" yet.) We see that the year with the highest overall satisfaction is 2014. We compare this with the conclusion drawn that the year with the highest price is 2015, with the knowledge that the most important variable from the decision tree for a predicting overall satisfaction rating is price. We also see from the regression equation modeling overall satisfaction that, generally, the higher the price, the higher the overall satisfaction. We see that entire homes and apartments increase overall satisfaction roughly the same as do private rooms.
We see that the number of bedrooms increasing is a factor that increases overall satisfaction, Finally, we see that the number of accommodates increasing, decreases overall satisfaction rating. This may be because there are more people who have the opportunity to give bad reviews, or because staying in an Airbnb/room with another person is generally less satisfying (hypothetically more annoying. messier, less private, less room, etc.)

Model Comparisons & Conclusions Drawn for Modeling Price

The root average square error of the decision tree modeling price is about $300, so the model is not very accurate. However, we expect this of the data on such a simple decision tree model, and it still yields useful results. It leads to variable importance conclusions that assist in the Neural Network and Regression models. Number of bedrooms, room type, year, number of accommodates, neighborhood, and overall satisfaction ranking are important in the prediction of price. This knowledge can assist in creating a much more accurate model that is not a Decision Tree and can lead to more conclusions. Because overall satisfaction is an important factor when predicting price, and vice versa, we combine the knowledge from the variables that predict overall satisfaction and that predict price for the Neural Network and Regression models.

We know the Neural Network is better than the Decision Tree model, due to a lack of evidence of overfitting; overfitting that variable importance knowledge from the decision tree assisted in avoiding overfitting in the Neural Network. Average Squared Error of Training set is about .0566 and of Validation set is about .0578 (these are lower/better than for the decision tree), & Misclassification Rate of Training set is about .385 and of Validation set is about .392. The accuracy for the predictions, from the model on the training set, against the original dataset is 63.6%. The accuracy for the predictions, from the model on the validation set, against the original dataset is 63.1%

As previously stated, the regression model that predicts price reveals more about the variable importance discovered by the decision trees (through interpretations of parameters), but note that all models have a relatively high average error of about $300.

Predicted Price Per Night of Airbnb Listing = -$180.30 + $25.9115*(1 if Year=2015) + $21.0961*(# of accommodates) + $85.249*(# of bedrooms) + $31.7491*(overall_satisfaction average rating from one to five) + $52.814*(1 if room_type= Entire Home/Apartment)

Some basic interpretations from the node rules from the Decision Tree modeling price include the following: **If room type IS ONE OF: ENTIRE HOME/APT AND bedrooms =1 then Predicted: price = $155.90956364 per night, if room type IS ONE OF: PRIVATE ROOM, SHARED ROOM AND bedrooms =1 then Predicted: price = $85.671898734 per night**. From contrasting these two nodes we conclude that entire homes/apartments are generally more expensive then are private and shared rooms (single rooms). **If bedrooms = 2 AND Year IS ONE OF: 2014, 2016 then Predicted: price = $232.37313433 per night, if bedrooms = 2 AND Year IS 2015 then Predicted: price = $318.461855 per night**. We conclude from these two node rules that 2015 listings generally have higher prices than do 2014 and 2016. **If bedrooms = 2 or 3 then Predicted: price = $386.87704918 per night, if bedrooms = 4 Predicted: price =$659.10625 per night, if bedrooms >= 5 then Predicted: price = $1851.09523 per night**. Finally, we see that when number of bedrooms increases, predicted price also increases. We learn about input variable relationships of variables

that factor into the price through these node rules. We then compare with the Regression equation for predicting price and make further conclusions. We expect price to increase by $25.91 if year is 2015, which confirms the decision tree interpretation. We see that we expect price to increase about $21 per one person increase in number of accommodates. When combining this knowledge with the fact that increasing the number of accommodates decreases overall satisfaction, this makes sense because the customer is paying more for the same Airbnb listing, simply because more people are staying there. We see that with each additional bedroom, price is expected to increase about $85. We see that with each additional unit in overall satisfaction rating, price is expected to increase about $31, and that price is generally higher when room type is entire home/apartment (expected because more bedrooms increases price). Again, a limitation of this regression model is that there is a lot of multicollinearity between input variables.

Conclusions from Combination of Predicting Overall Satisfaction and Price

Because overall satisfaction is an important factor when predicting price, and vice versa, we combine the knowledge from the variable importance and conclusions drawn from models that predict both target variables. From the Decision Tree modeling price, we see that when number of bedrooms increases, predicted price also increases. We compare with the Regression equation for predicting price, and make further conclusions. Variable importance for the Decision Tree modeling overall satisfaction is price, neighborhood, accommodates, room type, year and bedrooms. We draw further conclusions about the input parameters and their relationships from the Regression equation for predicting overall satisfaction. We see that the year with the highest overall satisfaction is 2014. We compare this with the conclusion drawn that the year with the highest price is 2015, with the knowledge that the most important variable from the decision tree for predicting overall satisfaction rating is price. We see that the number of bedrooms is a factor that increases overall satisfaction, while number of accommodates increasing, decreases overall satisfaction rating. This may be because there are more people to have the opportunity to give bad reviews, or because staying in an Airbnb/room with another person is generally less satisfying due to less privacy, space, etc. From the regression equation predicting price, we expect price to increase about $21 per one-person increase in number of accommodates. When combining this knowledge with the fact that increasing the number of accommodates decreases overall satisfaction, this makes sense because the customer is paying more for the same Airbnb listing, simply because more people are staying there. An increase in overall satisfaction results in an increase in price, while simultaneously, an increase in price increases overall satisfaction. We make further conclusions by contrasting and comparing the regression equations for predicting price and overall satisfaction. We expect overall satisfaction to increase .000056 units per dollar increase in price per night, while we expect price to increase by about $32 per one unit increase in overall satisfaction. An interesting tradeoff between we observe between the two target variables is that **a one unit increase in number of accommodates is expected to *decrease overall satisfaction about .0241 units*, while simultaneously expecting to *increase price about $21.*** For future research, these analyses may assist Airbnb in adjusting the number of accommodates in a way such that this tradeoff is at its lowest point, keeping in mind that price is the most influential factor on overall satisfaction. This is real-world research, as Marriot has done studies and concluded that a one-level increase in a hotel's TripAdvisor rating results in a certain (do not recall exact amount) dollar increase in per night additional room rate. TripAdvisor has always allowed guests to rank their experiences at hotels, restaurants, etc. based on a five-star system with written reviews, just as overall satisfaction is an average rating from one to five with written reviews as another (deleted) related variable in the Airbnb dataset.

A final limitation of this paper is that we could have grouped Neighborhoods somehow (for example, relating to average income), or that we only analyze Los Angeles County, because the level of influence of each factor could vary between places and could affect the relationship between overall satisfaction and price. This

possible limitation is reinforced in the article "TrustYou Study with AccorHotels Shows Effect of TripAdvisor Reviews on Bookings"…

“A Cornell study found that a one-point increase in reputation (based on a five-point scale) may result in a hotel’s ability to raise room rates up to 11.2%…(While) TripAdvisor hotel rankings are heavily influenced by a hotel’s percentage of five-bubble reviews. These effects are stronger in Europe: if the share of 5-bubble reviews increases by 10%, the ranking improves by 11.3%. A hotel’s percentage of 5-bubble reviews has a positive influence on the number of bookings. However, the level of influence of the factor varies between Europe and Asia-Pacific. For instance, the effect of the percentage of 5-bubble reviews is greater in Europe compared to AsiaPacific...If the share of 5-bubble reviews increases by 10%, the number of bookings increases by 10.2% in Europe and 7.8% in Asia-Pacific."

**Note that we provide LIMITATIONS and FUTURE RESEARCH in “RESULTS” and “CONCLUSIONS” sections*

*These algorithms require clean and valid data, correct algorithm settings and assumptions, and implementation on training data and test data in order hold any statistical value.

* Data scientists experiment with a multitude of predictive modeling and machine-learning algorithms in order to find the one(s) that work best for their specific problem. Automated modeling tournaments where miners can experiment to identify the winning modeling strategy quickly function for scoring inside Hadoop, etc. for very seamless integration with business applications and fast operational results. In addition to generating score code in different languages and formats, SAS Enterprise Miner also generates many assets that enable easy deployment, management, and monitoring of predictive models as part of operational business processes. Metadata supports all of these assets.

BIBLIOGRAPHY

Applied Analytics Using SAS Enterprise Miner. Copyright 2011, SAS Institute Inc., Cary, North Carolina, USA. Peter Christie, Jim Georges, Jeff Thompson, and Chip Wells. Curriculum Development and Support Department. (1)

Ady, Margaret. TrustYou Study with AccorHotels Shows Effect of TripAdvisor Reviews on Bookings. September 30, 2015. https://www.trustyou.com/press/trustyou-studyaccorhotels-shows-effect-tripadvisor-reviews-bookings-2 (2)

Data source: http://insideairbnb.com/get-the-data.html

APPENDIX

SAS CODE FOR CLEANING DATASET FOR ANALYSES:

```
*BELOW: first dataset from 2014 (full); filename csvone '/home/jgl12/airbnb1of2014.csv';
proc import datafile=csvone
		DBMS=CSV
		OUT=work.firstoffourteen;
		GETNAMES=YES;
run;
*BELOW: second dataset from 2014 (full); filename csvtwo '/home/jgl12/airbnb2of2014.csv';
proc import datafile=csvtwo
		DBMS=CSV
		OUT=work.secondoffourteen;
		GETNAMES=YES;
run;
*BELOW: third/final dataset from 2014 (full); filename csvthree '/home/jgl12/airbnb3of2014.csv';
proc import datafile=csvthree
		DBMS=CSV
		OUT=work.thirdoffourteen;
		GETNAMES=YES;
run;
*BELOW: first dataset from 2015 (full); filename csvfour '/home/jgl12/airbnb1of2015.csv';
proc import datafile=csvfour
		DBMS=CSV
		OUT=work.firstoffifteen;
		GETNAMES=YES;
run;
*BELOW: second dataset from 2015 (full); filename csvfive '/home/jgl12/airbnb2of2015.csv';
proc import datafile=csvfive
		DBMS=CSV
		OUT=work.secondoffifteen;
		GETNAMES=YES;
run;
*BELOW: third/final dataset from 2016 (full); filename csvsix '/home/jgl12/airbnb3of2015.csv';
proc import datafile=csvsix
		DBMS=CSV
		OUT=work.thirdoffifteen;
		GETNAMES=YES;
run;
*BELOW: first dataset from 2016 (full); filename csvseven '/home/jgl12/airbnb1of2016.csv'; proc import datafile=csvseven
		DBMS=CSV
		OUT=work.firstofsixteen;
		GETNAMES=YES;
run;
*BELOW: second dataset from 2016 (full); filename csveight '/home/jgl12/airbnb2of2016.csv';
proc import datafile=csveight
		DBMS=CSV
		OUT=work.secondofsixteen;
		GETNAMES=YES;
run;
*BELOW: third/final dataset from 2016 (full); filename csvnine '/home/jgl12/airbnb3of2016.csv';
proc import datafile=csvnine
		DBMS=CSV
		OUT=work.thirdofsixteen;
		GETNAMES=YES;
run;
*BELOW- deleted observations with numeric, variable values missing for each dataset, before taking random sample from each .csv
file;
data firstoffourteen;
set firstoffourteen; if cmiss(of _numeric_) then delete;
run; data secondoffourteen; set secondoffourteen; if cmiss(of _numeric_) then delete;
run; data secondoffourteen; set secondoffourteen; if cmiss(of _numeric_) then delete;
run; data thirdoffourteen; set thirdoffourteen; if cmiss(of _numeric_) then delete;
run; data firstoffifteen; set firstoffifteen; if cmiss(of _numeric_) then delete;
run; data secondoffifteen; set secondoffifteen; if cmiss(of _numeric_) then delete;
run; data thirdoffifteen; set thirdoffifteen; if cmiss(of _numeric_) then delete;
```

```
run; data firstofsixteen; set firstofsixteen; if cmiss(of _numeric_) then delete;
run; data secondofsixteen; set secondofsixteen; if cmiss(of _numeric_) then delete;
run; data thirdofsixteen; set thirdofsixteen; if cmiss(of _numeric_) then delete;
run;
*BELOW: SIMPLE RANDOM SAMPLING of 2,000 observations from each dataset; proc sql outobs=2000; create table sql_2000_sample as
select * from firstoffourteen order by ranuni(0);
quit;

proc sql outobs=2000; create table sql_2of2014_sample as
select * from secondoffourteen order by ranuni(0);
quit;

proc sql outobs=2000; create table sql_3of2014_sample as
select * from thirdoffourteen order by ranuni(0);
quit;

proc sql outobs=2000; create table sql_1of2015_sample as
select * from firstoffifteen order by ranuni(0);
quit;

proc sql outobs=2000; create table sql_2of2015_sample as select * from secondoffifteen order by ranuni(0);
quit;

proc sql outobs=2000; create table sql_3of2015_sample as
select * from thirdoffifteen order by ranuni(0); quit;

proc sql outobs=2000; create table sql_1of2016_sample as
select * from firstofsixteen order by ranuni(0);
quit;

proc sql outobs=2000; create table sql_2of2016_sample as
select * from secondofsixteen order by ranuni(0);
quit;

proc sql outobs=2000; create table sql_3of2016_sample as
select * from thirdofsixteen order by ranuni(0);
quit;
*BELOW, add year variable to each random sample dataset; data sample1from2014; set work.sql_2000_sample; drop minstay; Year='2014'; run; data sample2from2014; set work.sql_2of2014_sample; drop minstay; Year='2014'; run; data sample3from2014; set work.sql_3of2014_sample; drop minstay; Year='2014'; run; data sample1from2015; set work.sql_1of2015_sample; drop minstay; Year='2015'; run; data sample2from2015; set work.sql_2of2015_sample; Year='2015'; drop minstay; run; data sample3from2015; set work.sql_3of2015_sample; Year='2015'; drop minstay; run; data sample1from2016; set work.sql_1of2016_sample; Year='2016'; drop minstay; run; data sample2from2016; set work.sql_2of2016_sample; drop minstay; Year='2016'; run;
data sample3from2016; set work.sql_3of2016_sample;
drop minstay; Year='2016'; run;
*BELOW, create final dataset for analyses; data created;
set sample1from2014 sample1from2015 sample1from2016 sample2from2014 sample2from2015 sample2from2016 sample3from2014 sample3from2015 sample3from2016; drop borough last_modified survey_id country city bathrooms location; run;
*there are now 10 variables: room_id, host_id, room_type, neighborhood reviews, overall_satisfaction, accommodates, bedrooms, price, and year (plus latitude and longitude variables for Tableau); proc contents data=created; run;
proc means data=created NMISS; run;
*no missing numeric variable values; proc freq data=created; tables _CHAR_ / missing; run;
*no missing character variables, note that in the original dataset- there were four variables with a relatively low number of missing values, therefore I was able to delete these incomplete observations, prior to taking each random sample;

data sasuser.readyforanalyses;
set created; run;
```

Property	Value
Table Name	SASUSER.READYFORANALYSES
Description	
Member Type	DATA
Data Set Type	DATA
Engine	V9
Number of Variables	12
Number of Observations	18000
Created Date	March 23, 2018 11:47:02 AM PDT
Modified Date	March 23, 2018 11:47:02 AM PDT

Deep Learning in Python and in SAS Enterprise Miner: Prediction of Type II Diabetes in Female Pima Indians

ABSTRACT

Obesity is a major problem in the United States, and results in the development of Type II Diabetes. Our goal is to produce a deep learning model that most accurately predicts the development of type II diabetes. Our population is female patients of Pima Indian heritage of Arizona, whom have the highest reported prevalence of diabetes of any population in the world. All patients in this sample are females of at least 21 years of age. This population appears to have subjects with exclusively Type II Diabetes. We utilize both Python and Enterprise Miner in order to achieve two separate (and optimal) Deep Learning models. We predict whether a specific observation has diabetes or does not (yes=1). We then make conclusions based on these results, and articulate the limitations of the models/data.

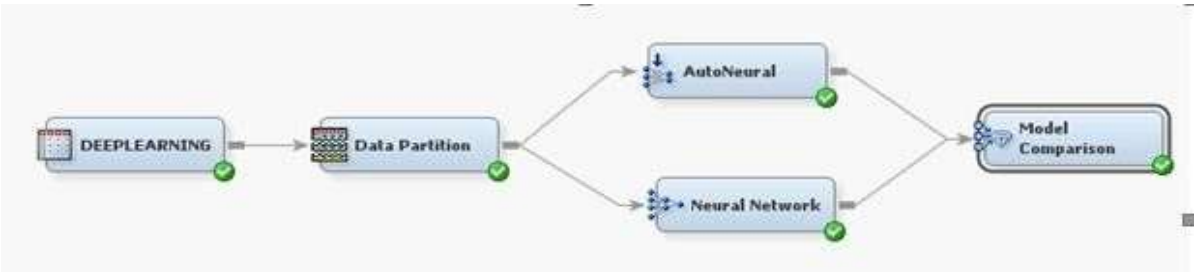
INTRODUCTION

There are eight predictor variables, and a binary response variable:

1) Pregnancies: Number of times pregnant
2) Glucose: Plasma glucose concentration 2 hours in an oral glucose tolerance test
3) BloodPressure: Diastolic blood pressure (mm Hg)
4) SkinThickness: Triceps skinfold thickness (mm)
5) Insulin: 2-Hour serum insulin (mu U/ml)
6) BMI: Body mass index (weight in kg/(height in m)^2)
7) DiabetesPedigreeFunction: *description below list
8) Age: In years
9) Outcome: Class variable (0 or 1) (diabetes=1)

*“The Diabetes Pedigree Function (DPF) was developed by Smith, et al. to provide a synthesis of the diabetes mellitus history in relatives and the genetic relationship of those relatives to the subject. The DPF uses information from parents, grandparents, siblings, aunts and uncles, and first cousins. It provides a measure of the expected genetic influence of affected and unaffected relatives on the subject’s eventual diabetes risk.” (1)

DEEP LEARNING IN SAS ENTERPRISE MINER



Number of Hidden Units 8

Target	Target Label	Fit Statistics	Statistics Label	Train	Validation
Outcome		_DFT_	Total Degrees of ...	306	.
Outcome		_DFE_	Degrees of Freed...	297	.
Outcome		_DFM_	Model Degrees of...	9	.
Outcome		_NW_	Number of Estim...	9	.
Outcome		_AIC_	Akaike's Informati...	297.8727	.
Outcome		_SBC_	Schwarz's Bayesi...	331.385	.
Outcome		_ASE_	Average Squared ...	0.148815	0.157943
Outcome		_MAX_	Maximum Absolut...	0.96009	0.994789
Outcome		_DIV_	Divisor for ASE	612	924
Outcome		_NOBS_	Sum of Frequenci...	306	462
Outcome		_RASE_	Root Average Squ...	0.385766	0.39742
Outcome		_SSE_	Sum of Squared ...	91.07505	145.9389
Outcome		_SUMW_	Sum of Case Wei...	612	924
Outcome		_FPE_	Final Prediction E...	0.157835	.
Outcome		_MSE_	Mean Squared Er...	0.153325	0.157943
Outcome		_RFPE_	Root Final Predict...	0.397284	.
Outcome		_RMSE_	Root Mean Squar...	0.391567	0.39742
Outcome		_AVERR_	Average Error Fun...	0.457308	0.495165
Outcome		_ERR_	Error Function	279.8727	457.532
Outcome		_MISC_	Misclassification ...	0.218954	0.225108
Outcome		_WRONG_	Number of Wrong...	67	104

Average Squared Error of Training set is about .1488 and of Validation set is about .158, & Misclassification Rate of Training set is about .21895 and of Validation set is about .22511

Event Classification Table

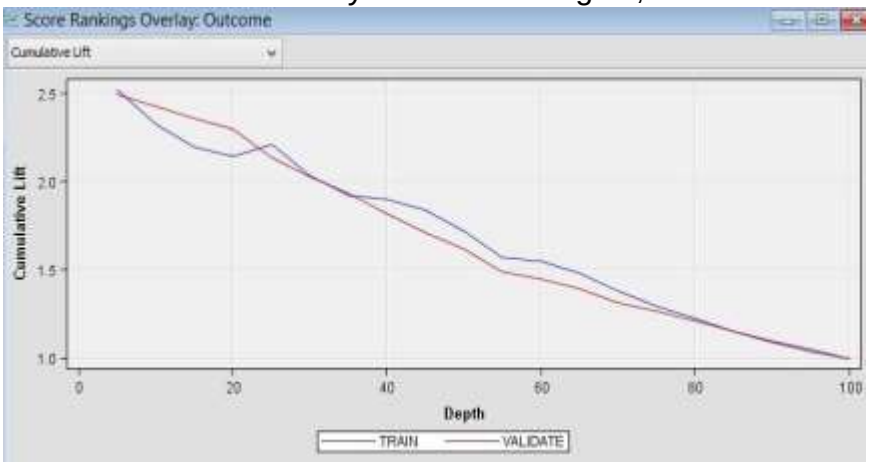
Data Role=TRAIN Target=Outcome Target Label=' '

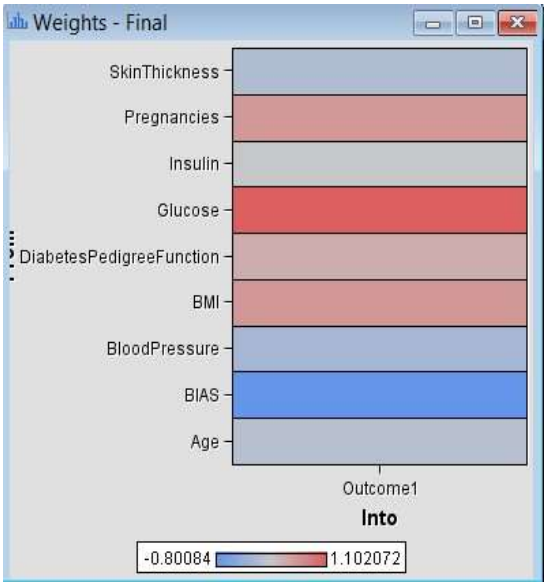
False Negative	True Negative	False Positive	True Positive
42	175	25	64

Data Role=VALIDATE Target=Outcome Target Label=' '

False Negative	True Negative	False Positive	True Positive
68	264	36	94

Note that the model only contains 9 weights, so termination criterion set at 'Overfitting'





The iteration plot shows us the average squared error vs. optimization iteration. A large divergence in training and validation average squared error occurs near iteration 5. This indicates a decent model.

DEEP LEARNING IN PYTHON

Importing Deep Learning Packages into Python

```
from keras.models import Sequential
from keras.layers import Dense
import numpy
# fix random seed for reproducibility
numpy.random.seed(7)
```

```
Using TensorFlow backend.
```

Creating three-layer model. First two layers use rectifier activation function, last uses sigmoid. Number of neurons: 12, 8, and 1.

Create model

```
model = Sequential()
model.add(Dense(12, input_dim=8, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

Compiling the model. Logarithmic loss, our metric is prediction accuracy. Fit over 150 epochs**.**

Compile model

```
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
```

Fit the model

```
model.fit(X, Y, epochs=150, batch_size=10)
Epoch 142/150
768/768 [==============================] - 0s 396us/step - loss: 0.4826 - acc: 0.7695
Epoch 143/150
```

Evaluate the model

```
scores = model.evaluate(X, Y)
print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
```

```
768/768 [==============================] - 0s 142us/step

acc: 77.86%
```

Calculate predictions

```
predictions = model.predict(X)
```

Round predictions

```
rounded = [round(x[0]) for x in predictions]
print(rounded)
```

```
[1.0, 0.0, 1.0, 0.0, 1.0, 0.0, 0.0, 1.0, 1.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 1.
0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0, 0.0, 1.0, 1.0, 1.0, 0.0, 1.0, 1.0, 1.0, 1.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 0.0, 1.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 1.0, 1.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0
.0, 0.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 1.0, 0.0, 0.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0,
0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1.0, 1.0, 0.0, 0.0, 1.0
```

RESULTS

Deep Learning and Neural Networks are known as “black box” predictions, where the underlying parameters that lead to the predictions are neither relevant nor possible to comprehend. The main point is to be accurate on future predictions as possible. Thus, we compare the accuracies of both models, and get very similar results.

IN ENTERPRISE MINER DEEP LEARNING MODEL,
The accuracy for the predictions, from the model on the training set, against the original dataset is 78%. The accuracy for the predictions, from the model on the validation set, against the original dataset, is 77.49%.

IN PYTHON DEEP LEARNING MODEL,
The accuracy, which automatically computes model predictions against the original dataset, is 77.86%.

As previously mentioned, the iteration plot for model 1 (done in Enterprise Miner) shows us the average squared error vs. optimization iteration. A large divergence in training and validation average squared error occurs near iteration 5. This indicates a decent model. Although, there are suspicions of overfitting.

LIMITATIONS

Both the EM model and the Python model have a small number of layers for a Deep Learning model. When changing the maximum number of hidden layers in EM from 3 to 8, the results of the model do not change. We suspect overfitting of our predictions. As previously mentioned, these are “black box” predictions, which is another limitation. This paper is simply provided in order to exemplify application of Deep Learning in both SAS Enterprise Miner and Python experience

BIBLIOGRAPHY

Estimating Probabilities of Diabetes Mellitus Using Neural Networks. Murali Shanker. M.Y. Hu. M. S. Hung. November 13, 1999.

Data Source:
https://data.world/data-society/pima-indians-diabetes-database

SURVIVAL ANALYSIS ON WORCESTER HEART ATTACK STUDY SUBSET

AN EMPIRICAL COMPARISON

INTRODUCTION

Survival analysis is concerned with studying the time between entry to a study and a subsequent event and becomes one of the most important fields in statistics. The Cox proportional hazard model is the most widely used for the analysis of survival data in the presence of covariates or prognostic factors because of its simplicity, and not being based on any assumptions about the survival distribution. However, the Cox proportional hazard model may not be appropriate in many situations and other modifications such as stratified Cox model, Interaction Model, or Cox model with time dependent variables can be used for the analysis of survival data.

In this paper, we will be analyzing a subset of data from the original Worcester Heart Attack Study using Cox regression, Stratified Cox regression, Interaction Model, and time-dependent Cox regression model. This paper examined several covariates, such as age, gender and peak cardiac enzyme, that may influence survival time after heart attack. The outcome variable is the follow-up time. Follow-up time for all participants begins at the time of hospital admission after heart attack and ends with death or loss to follow up (censoring). The main objective is to build a model that describes the survival data as adequately as possible yet contains as few variables as possible. The effect of covariates on hazard rate will be examined for the most adequate model.

DATA

The data set used for this study is obtained by taking a 10% random sample within 6 of the cohort years from the original Worcester Heart Attack Study (WHAS). Only a small subset of variables is included in this data set. The data set includes 481 observations with 249 censored observation. Out of Eight covariates, six covariates are binary or nominal. The two continuous variables are age and CPK. FSTAT is the censoring variable, which is equal to 0 if the patient is still alive but the actual survival time is unknown, and 1 if the patient dies. We delete the 6 observations with MITYPE = 3 since the MITYPE value is indeterminate (unknown).

The covariates used in this dataset are displayed in Table1.

Table1. Description of the covariates obtained from WHAS

Covariate	Description	Codes/Units
Age	Age	Years
Sex	Gender	0= Male, 1= Female
CPK	Peak Cardiac Enzyme	International Units(iu)
SHO	Cardiogenic Shock Complications	0= No, 1= Yes
CHF	Left Heart Failure Complications	0= No, 1= Yes
MIORD	MI Order	0 = First, 1= Recurrent
MITYPE	MI Type	1= Q - wave, 2 = Not Q - wave
YRGRP	Grouped Cohort Year	1 = 1975 & 1978, 2 = 1981 & 1984, 3 = 1986 & 1988

METHODS

After fitting a reduced model, named the “WHAS” model, from the original/full model… Using the forward selection approach to the partial log-likelihood ratio test,

$$H_0 : \beta_i = 0 \qquad -2\log L\,(\text{reduced model}) - (-2\log L\,(\text{full model})) =$$

$$H_1 : \beta_i \neq 0 \qquad -2\log\left(L\,(\text{reduced model}) \,/\, L\,(\text{full model})\right)$$

$$\sim \chi^2_{\alpha}\,(\text{df})$$

where df = # of parameters (covariates) dropped from full to reduced model

*H0 and H1 test specific/singular β_i, not global/all

...It is assumed that the hazard of an individual with a certain value, or stratum, for each of the included covariates is <u>constantly</u> proportional to the hazard of an individual with another possible value for that particular covariate (HR is independent of time t.)

$$\hat{HR} = \frac{\hat{h}(t, X^*)}{\hat{h}(t, X)} \cdot = \frac{\lambda_0(t) e^{\beta_1 x_{1i} + \ldots + \beta_k x_{ki}}}{\lambda_0(t) e^{\beta_1 x_{1j} + \ldots + \beta_k x_{kj}}} = \exp\left[\sum_{i=1}^{p} \hat{\beta}_i \left(X_i^* - X_i\right)\right]$$

This confirms that each covariate included in the WHAS model acts additively on the log hazard ratio, as all additive models assume that the effect of each covariate is independent of time, and thus multiplicatively on the hazard ratio.

$$\lambda_i(t|\mathbf{x}_i) = \lambda_0(t)\exp\{\mathbf{x}_i'\boldsymbol{\beta}\}$$

Note, $\lambda_0(t)$ must be positive but is left unspecified

$$\log(\lambda(t, \boldsymbol{x}_1)) = \log(\lambda(t, \boldsymbol{x}_2)) + \boldsymbol{\beta}'(\boldsymbol{x}_1 - \boldsymbol{x}_2)$$

$$\lambda(t, \boldsymbol{x}_1) = e^{\boldsymbol{\beta}'(\boldsymbol{x}_1 - \boldsymbol{x}_2)}\ \lambda(t, \boldsymbol{x}_2)$$

A main limitation that the partial Worcester dataset presents is that YRGRP does not meet the PH assumption. Though this variable is not included in the reduced WHAS model, it is included in the full model that the WHAS model is fitted from and thus it is necessary to investigate further. This means that the effect of YRGRP on the hazard ratio is not constant across time, and thus YRGRP acts directly on survival time, and not additively with the other covariates on the log hazard as it should. It is determined to fit a stratified model, by YRGRP, with the four covariates included in the WHAS model. If it is found that there are noticeable differences in the four main effects across the YRGRP cohort groups, then it is possible that the effect of the covariates in the WHAS model are not constant across the YRGRP cohorts. Therefore, interactions between YRGRP and each of the four covariates are investigated univariately, in order to see if the interaction model better accounts for the differences of the covariates across the YRGRP cohort groups. An additional method (in the attempt to account for this difference in long-term survival in the YRGRP cohort groups) with this dataset, is finally implemented to investigate the effectiveness of a time-dependent variable of patient “discharge” status that depends on the different lengths of stay in the hospital.

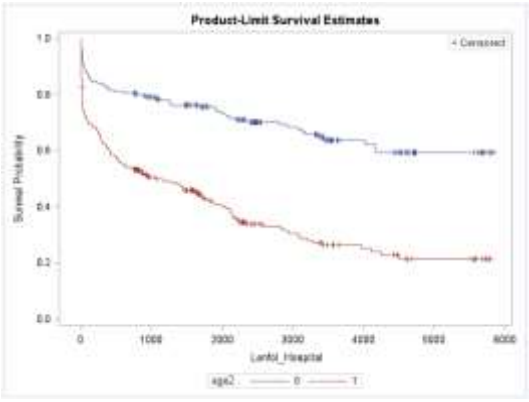
RESULTS

Covariate Prescreening

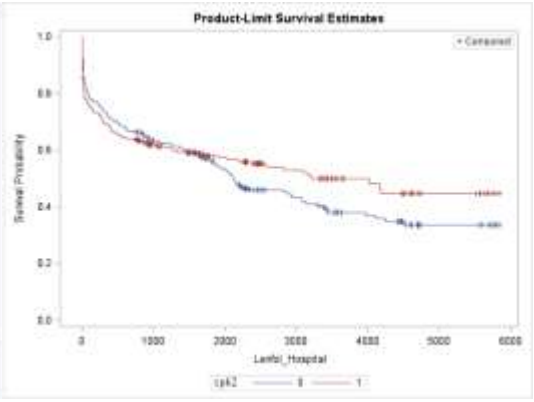
First, we will conduct a univariate analysis to screen out potentially significant variables for consideration in the multivariate model. The following figure shows that Age, Sex, SHO, CHF and MIORD might be important while CPK, MITYPE and YRGRP may not be significant.

Figure 1. Univariate KM plots of Survival curves

(Continuous variables have been dichotomized)

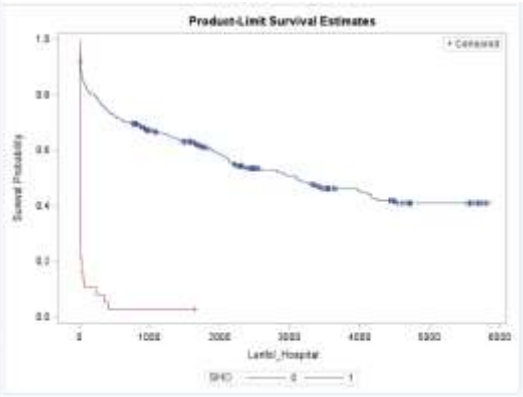


Age

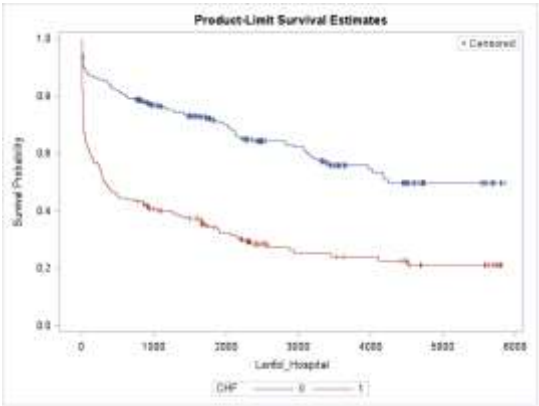


CPK

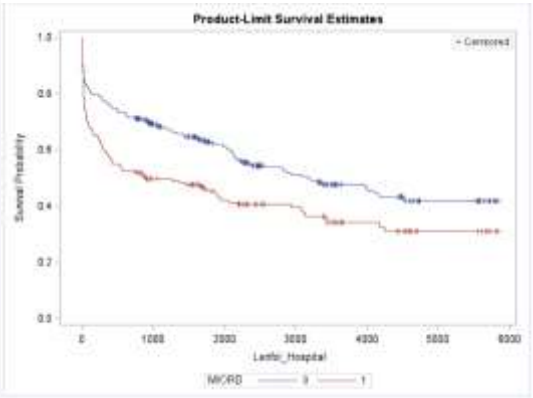
Sex



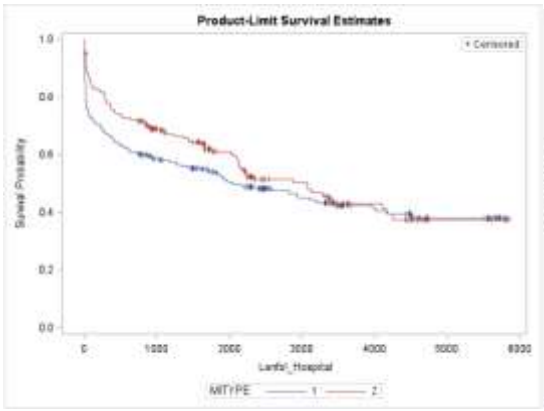
SHO

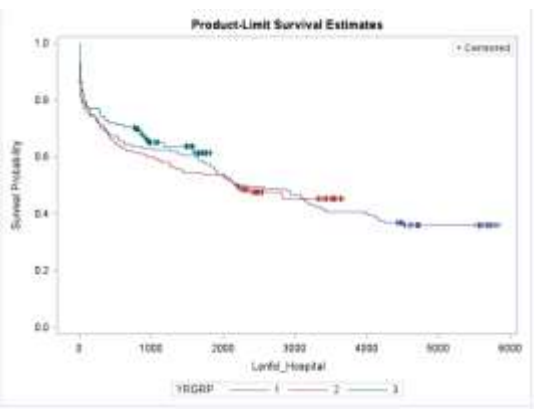


CHF



MIORD





MITYPE* YRGRP

*6 observations with MITYPE = 3(indeterminate) are deleted

After variable prescreening step, we decide to keep age, SEX, SHO, CHF, MIORD into our multivariate model selection step.

MODEL DEVELOPMENT

For use in practice, a cox proportional hazards model needs to contain as few variables as possible, yet must still describe the data as adequately as possible. We used forward selection, backward and stepwise selection method to include or exclude variables in a Cox model. We start from the variables age, Sex, SHO, CHF, MIORD. Table 2 shows the forward selection process. In forward method, variables are added to the model one at a time. At each stage in the process, the variable added is the one that gives the largest decrease in the value of -2 Log on its inclusion. The process ends when the next candidate for inclusion in the model does not reduce the value of -2logL by more than a prespecified amount.

Table 2 Value of -2logL for model fitted to data following forward selection

Variables in Model	**-2 LOG L**	**Forward Selection**
none	2778.526	
SHO	**2683.394**	*add SHO
age	2712.631	
CHF	2714.318	

MIORD	2765.359	
sex	2768.979	
SHO + age	2633.653	* add age
SHO + CHF	2648.357	
SHO + MIORD	2677.386	
SHO + sex	2679.086	
SHO + age + CHF	2616.211	
SHO + age + MIORD	2627.662	
SHO + age + sex	2633.544	*add CHF
SHO + age + CHF + MIORD	2611.673	
SHO + age + CHF + sex	2616.167	*add MIORD
SHO + age + CHF + MIORD + sex	2611.663	*terminate process

The first step is to fit the null model and models that contain each of the five explanatory variables on their own. Of these variables, SHO leads to the largest reduction in -2LogL, reducing the value of the statistic from 2778.526 to 2683.394. This reduction is significant at the 5% level when compared with percentage points of the chi-square distribution with 1 d.f. We include SHO in the model for the first step.

The next step is to fit the model that contains SHO. The effect of including each of the four remaining variables to this model is shown in table**.** When age is entered, given that SHO is already in the model, the decrease in -2LogL is the largest. This reduction is significant at 5% level, indicating that this variable will be considered for inclusion for this step.

We continue the process for the third and fourth step, adding variables CHF and MIORD into the model. Finally, we decide if sex should be included in the model that contains SHO, age, CHF and MIORD. Table 2 shows that when sex is added, the reduction in -2Log L is less than 3.84- the 95% percentile of chi-square distribution with d.f.1, so sex is excluded from the model. The most satisfactory model is that containing SHO, age, CHF and MIORD.

We also confirm this subset of variables by following the general strategy for model selection. Table 3 shows the process.

Table 3. Value of -2logL for model fitted to data following general strategy of model selection

Variables in Model	**-2 Log**	General Selection
none	2778.526	
SHO	2683.394	
age	2712.631	
CHF	2714.318	
MIORD	2765.359	
sex	2768.979	* ALL five covariates entered
SHO + age + CHF + MIORD + sex	2611.663	
SHO + age + CHF + MIORD	2611.673	*final model(we checked that no term in the model can be omitted)
SHO + age + CHF + sex	2616.167	
age + CHF + MIORD+SEX	2669.132	
SHO+CHF+MIORD+SEX	2641.58	
SHO+AGE +MIORD+SEX	2627.613	* sex omitted
SHO+ age+CHF	2616.211	
SHO+ age+MIORD	2627.662	
SHO+CHF+MIORD	2643.655	
AGE + CHF+MIORD	2669.522	

MODEL ADEQUACY

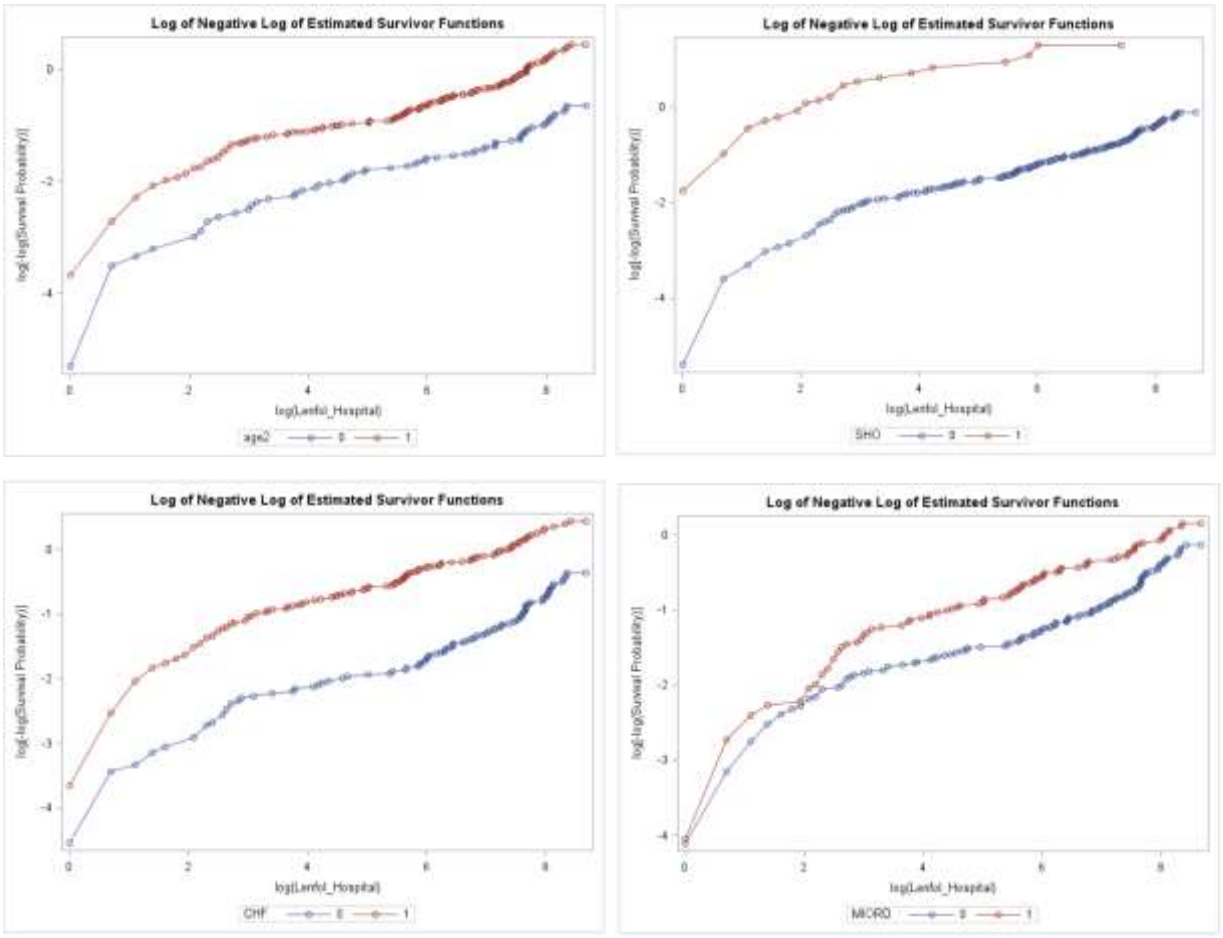
The proportional hazards assumption is vital to the interpretation and use of a fitted proportional hazards model. The estimated hazard ratios do not depend on time. We use Graphical method and formal test to evaluate the proportional hazard assumption.

Method 1. Graphical Approach

Assessing the proportional hazards assumption is an examination of the extent to which the two plotted log hazard functions are equidistant from each other over time. For each of the categorical covariates SHO, CHF and MIORD, we plot the log-log survival curves based on the estimated Kaplan-Meier survival functions for two groups. Then we see if the two curves are parallel to each other. For continuous covariate age, we break it into two groups and apply the method for categorical covariates.

The proportional hazards assumption implies that log-log survival curves should be parallel. If the covariate satisfies the proportional hazard assumption then the shapes of the curves should be basically the same, and the separation between the curves should remain proportional across analysis time. From the figure below we can see, for AGE, SHO, CHF and MIORD, the proportional hazards assumptions are acceptable.

Figure 2. Log-Log survival curves for different groups (continuous variables are dichotomized)



Method 2. Including Time Dependent Covariates in the Cox Model

The proportional hazards assumption is that the ratio of hazards is a constant that does not depend on time. When this assumption fails, it is because the hazard ratio changes over time.

To test this, we generate the time dependent covariates by creating interactions of the covariates and a function of survival time and include the interactions in the model. If any of the time dependent covariates are significant then those corresponding covariates are not proportional. Again, for AGE, SHO, CHF and MIORD, none of the interaction terms are highly significant(Table 4), thus we have no evidence against the proportional hazard assumptions.

Table 4. Test for Proportional Hazard Assumption								
Parameter	**DF**	**Parameter Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**	**Hazard Ratio**	**95% Hazard Ratio Confidence Limits**	
AGE	1	0.02409	0.01222	3.8850	0.0487	1.024	1.000	1.049
SHO	1	2.32278	0.36454	40.5993	<.0001	10.204	4.994	20.848
CHF	1	0.72738	0.33697	4.6595	0.0309	2.070	1.069	4.006
MIORD	1	0.29903	0.26962	1.2301	0.2674	1.349	0.795	2.288
aget	1	0.00181	0.00225	0.6448	0.4220	1.002	0.997	1.006
SHOt	1	-0.20472	0.12468	2.6962	0.1006	0.815	0.638	1.040
CHFt	1	-0.02662	0.05947	0.2004	0.6544	0.974	0.867	1.094
MIORDt	1	-0.00490	0.05066	0.0094	0.9229	0.995	0.901	1.099

After assessing the Proportional Hazard Assumption, we keep all the four covariates in our main model. We fit a proportional hazards model that estimates the effects of Age, cardiogenic shock complication, left heart failure complications (yes/no) and MI Order on long-term survival following hospitalization for an acute myocardial infarction in the WHAS dataset. **Table 5** shows the hazard ratio and 95% confidence intervals for each covariate.

Table 5. Parameter Estimates for Main Effects Model(Non-Stratified)								
Parameter	**DF**	**Parameter Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**	**Hazard Ratio**	**95% Hazard Ratio Confidence Limits**	
SHO	1	1.80964	0.20910	74.9017	<.0001	6.108	4.054	9.202
AGE	1	0.03254	0.00580	31.4913	<.0001	1.033	1.021	1.045

CHF	1	0.58086	0.14465	16.1245	<.0001	1.788	1.346	2.374
MIORD	1	0.28366	0.13216	4.6069	0.0318	1.328	1.025	1.721
-2 Log L : 2611.673								

STRATIFIED COX REGRESSION MODEL

Based on the non-stratified model developed in the previous section (call it “WHAS” model), we treat the year group variable as a stratification variable and fit the WHAS model. The reason for stratifying on YRGRP is that this covariate does not satisfy the Proportional Hazard Assumption. Visual check of this assumption shows that the log-log curve crossed for different year group (Figure 3).

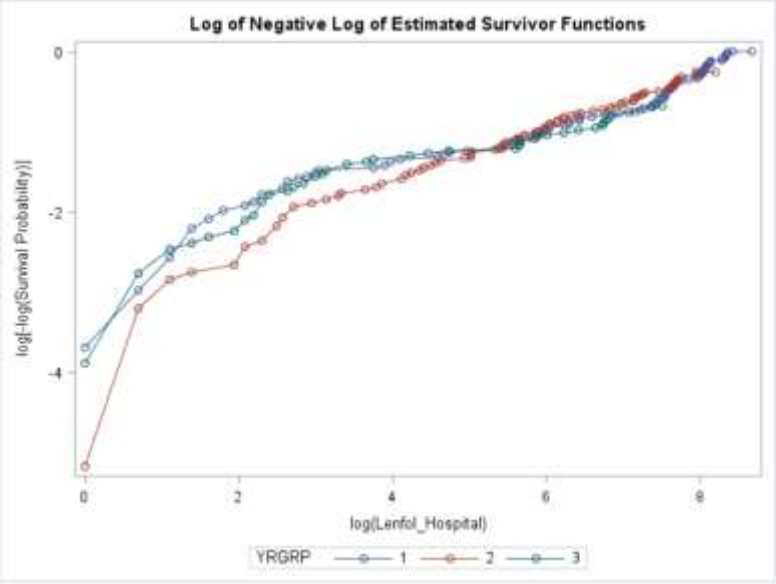


Figure 3. Log-Log Survival curves by YRGRP

If the covariates do not satisfy the Proportional Hazards Assumption, we put them into strata. The stratified Cox Regression model assumes no interaction between stratification variable and covariates. Table 6 shows the hazard ratios and 95% confidence intervals for stratified main effects Cox Proportional hazards model.

Table 6. Parameter Estimates for Main Effects Model(Stratified)								
Parameter	**DF**	**Parameter Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**	**Hazard Ratio**	**95% Hazard Ratio Confidence Limits**	
SHO	1	2.04689	0.22361	83.7890	<.0001	7.744	4.996	12.003
AGE	1	0.03329	0.00581	32.8062	<.0001	1.034	1.022	1.046
CHF	1	0.54822	0.14449	14.3955	0.0001	1.730	1.303	2.297
MIORD	1	0.28781	0.13196	4.7569	0.0292	1.333	1.030	1.727
-2 Log L: 2131.389								

As we compare the estimated coefficients between stratified and non-stratified model, there are no important differences between the results. This means that including the stratification variable does not change the coefficient estimates. The baseline survival functions for three different cohort year groups provides a visual tool to check this effect (Figure 4). It also suggests no significant strata effect (p-value = 0.7959), though the mean survival time in different groups is different.

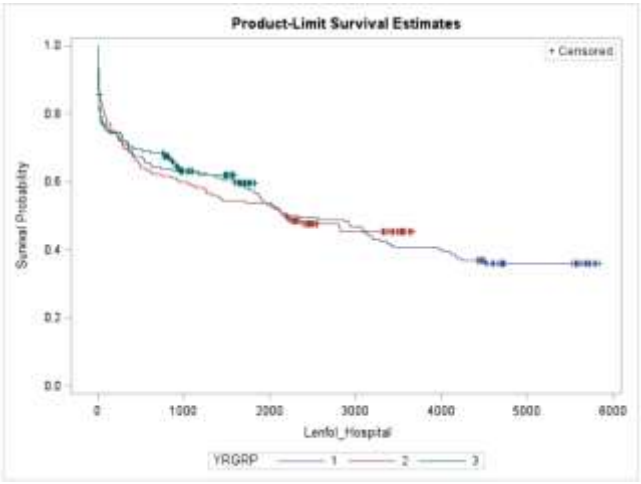


Figure 4. Survival curves by YRGRP

The previous stratified model assumes no interaction between stratification and covariate variables. It is possible that the effects of the covariate in **table (table for stratified WHAS model)** are not constant across cohorts defined by YRGRP.

To test the possible interaction effect, we add to the model the interaction terms between each covariate and YRGRP, and at the same time, stratify on YRGRP. We compare this extended model with the no interaction model (all models are stratified on YRGRP) to see which model fits the data better. A summary of the values of -2 LogL for the models that are to be considered is given in the following Table 7.

Table 7. Value of -2logL for models with interaction fitted to data

Variables in Model（all stratified by YRGRP)	**-2 LOG L**
Baseline: SHO + age + CHF + MIORD	2131.389
+age * YRGRP	2130.978
+SHO * YRGRP	2127.145
+CHF * YRGRP	2127.24
+MIORD * YRGRP	2130.361

Of these interaction terms, SHO * YRGRP leads to the largest reduction in -2LogL, reducing the value of the statistic from 2131.389to 2127.145. However, this reduction is not significant at the 5% level when compared with percentage points of the chi-square distribution with 2 d.f. Thus, we terminate the interaction selection process and conclude that the four-covariate WHAS model with YRGRP as stratification variable fits the data adequately.

TIME-DEPENDENT COX REGRESSION MODEL

It is possible in the WHAS model that differences in long-term survival in the grouped cohorts, YRGRP, could be due to different lengths of stay in the hospital. We examine this by creating a dichotomous time-varying covariate called in_hosp, by comparing LENSTAY and LENFOL and adding it to the WHAS model.

Table 8. Parameter Estimates for Time Dependent Cox Regression Model								
Parameter	**DF**	**Parameter Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**	**Hazard Ratio**	**95% Hazard Ratio Confidence Limits**	
AGE	1	0.03243	0.00581	31.1534	<.0001	1.033	1.021	1.045
SHO	1	1.71861	0.20874	67.7891	<.0001	5.577	3.704	8.396
CHF	1	0.55183	0.14499	14.4862	0.0001	1.736	1.307	2.307
MIORD	1	0.30594	0.13173	5.3936	0.0202	1.358	1.049	1.758
in_hosp	1	1.33699	0.39964	11.1923	0.0008	3.808	1.740	8.333
-2 LOG L: 2599.449								

As we can see from the result that there is a high hazard ratio associated with in_hosp, a significant variable in the model. The interpretation is that patients discharged from the hospital would likely to have higher survival rates than patients not discharged from the hospital. This is consistent with the effect of hospitalization: the longer the patient stay in the hospital, the more serious the condition is and the lower the survival rate is.

The non-linear regression models were fitted using 475 observations (6 observation was removed as their MITYPE is unknown) and the results are presented in Table 9. It shows that the four covariates namely AGE, SHO, CHF and MIORD are significantly associated with the survival time under all the model. Among the models, Stratified Cox model has the lowest level of deviance (-2 LOG L) compared to all other models. The other two models have significantly higher deviance compared to Stratified Cox model. Our final model for the dataset is the stratified cox regression model with cohort group(YRGRP) as the stratification.

Table 9. Parameter Estimates under Three Models

Estimates and 95% for Cox Regression Model(Non-Stratified)								
Parameter	**DF**	**Parameter Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**	**Hazard Ratio**	**95% Hazard Ratio Confidence Limits**	
SHO	1	1.80964	0.2091	74.9017	<.0001	6.108	4.054	9.202
AGE	1	0.03254	0.0058	31.4913	<.0001	1.033	1.021	1.045
CHF	1	0.58086	0.14465	16.1245	<.0001	1.788	1.346	2.374
MIORD	1	0.28366	0.13216	4.6069	0.0318	1.328	1.025	1.721
							-2 Log L : 2611.673	
Estimates and 95% for Extended Cox Regression Model(Stratified)								
Parameter	**DF**	**Parameter Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**	**Hazard Ratio**	**95% Hazard Ratio Confidence Limits**	
SHO	1	2.04689	0.22361	83.789	<.0001	7.744	4.996	12.003
AGE	1	0.03329	0.00581	32.8062	<.0001	1.034	1.022	1.046
CHF	1	0.54822	0.14449	14.3955	0.0001	1.73	1.303	2.297
MIORD	1	0.28781	0.13196	4.7569	0.0292	1.333	1.03	1.727
							-2 Log L: 2131.389	
Estimates and 95% CI for Time Dependent Cox Regression Model								
Parameter	**DF**	**Parameter Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**	**Hazard Ratio**	**95% Hazard Ratio Confidence Limits**	
AGE	1	0.03243	0.00581	31.1534	<.0001	1.033	1.021	1.045
SHO	1	1.71861	0.20874	67.7891	<.0001	5.577	3.704	8.396
CHF	1	0.55183	0.14499	14.4862	0.0001	1.736	1.307	2.307
MIORD	1	0.30594	0.13173	5.3936	0.0202	1.358	1.049	1.758
in_hosp	1	1.33699	0.39964	11.1923	0.0008	3.808	1.74	8.333
							-2 LOG L: 2599.449	

From Table 9, it is found that all four covariates are significant to the survivorship. Among the models, Stratified Cox model has the lowest level of deviance compared to other two models. Controlling for other covariates, the risk of death increases by 3.4 % as a patient's age increases by 1 year. Controlling for other covariates, the risk of death of patients with recurrent heart attack is 1.33 times the risk of patients with first heart attack. Controlling for other covariates, the risk of death of patients with left heart failure complication is 1.73 times the risk of patients with no such complication. Finally, risk of death for patients with Cardiogenic Shock Complications is 6.7 times greater than the risk of death for patients with no such condition.

APPENDIX

```
data WHAS;
input id age sex CPK SHO CHF MIORD MITYPE YEAR YRGRP LENSTAY DSTAT
LENFOL FSTAT;
datalines;
      1       62       1       485       1       1       0       1       1       1       1
1       1       1
      2       78       1       910       0       1       1       1       1       1       1
1       1       1
      3       81       1       320       1       1       0       1       1       1       1
2       6       3       8       0       284       1
9       1
    446       62       0       2460       0       0       0       1       6       3       9
0     1026       0  1
……….
    471       71       0       723       0       0       0       2       6       3       15
0      869       0
    480       84       1       579       0       1       0       3       6       3       39
1       39       1
    481       74       1       2203       1       1       0       1       6       3       68
1       68       1
    ;
    run;


proc means data=whas nmiss;
run;


proc contents data=whas;
run;
```

```
*PART 1- MODEL DEVELOPMENT (FULL VS REDUCED COX REGRESSION MODELS

on vars:  age, sex, CPK, SHO, CHF, MIORD, and MITYPE;


proc phreg data=WHAS;

model LENFOL*FSTAT(0)=age sex CPK SHO CHF MIORD MITYPE/RL selection=forward;

run;


*Confirm with partial log likelihood ratio tests- MAKE TABLE IN REPORT;


*no variables:  -2 LOG L =  2841.217;

proc phreg data=WHAS;

model LENFOL*FSTAT(0)=;

run;


*one variable- VARIABLE ADDED AT EACH STAGE IS ONE THAT GIVES LARGEST DESCREASE

IN VALUE OF -2 LOG L ON ITS INLCUSION-so ADD SHO IN STEP 1 is confirmed;


*-2 LOG L = 2832.089;

proc phreg data=WHAS;

model LENFOL*FSTAT(0)=sex/RL;

run;


*-2 LOG L = 2771.152;

proc phreg data=WHAS;

model LENFOL*FSTAT(0)=age/RL;

run;
```

```
*-2 LOG L = 2840.905;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=CPK;
run;

*-2 LOG L = 2743.869;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=SHO;
run;

*-2 LOG L = 2775.131;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=CHF;
run;

*-2 LOG L = 2829.460;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=MIORD;
run;

*-2 LOG L = 2840.683;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=MITYPE;
run;

*two-covariate models (add to SHO)- sho & age, sho & sex, sho & cpk, sho & chf,
sho & mitype, sho & miord)
-STEP 2 confirmed- age is added to the model;
```

```
*-2 LOG L = 2690.873;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=SHO age;
run;

*-2 LOG L = 2707.631;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=SHO CHF;
run;

*-2 LOG L = 2743.745;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=SHO CPK;
run;

*-2 LOG L = 2739.750;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=SHO sex;
run;


*-2 LOG L = 2738.826;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=SHO MIORD;
run;

*-2 LOG L = 2743.406;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=SHO MITYPE;
```

```
run;

*three-covariate models- add to SHO and age:
age&sex&SHO, age&CPK&SHO, age&SHO&CHF, age&SHO&MIORD,
age&SHO&MITYPE
add CHF to model confirmed (step 3);

*-2 LOG L = 2673.068;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO CHF;
run;

*-2 LOG L = 2685.691;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO MIORD;
run;

*-2 LOG L = 2689.554;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO MITYPE;
run;

*-2 LOG L = 2690.194;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age CPK SHO;
run;

*-2 LOG L = 2690.811;
proc phreg data=WHAS;
```

```
model LENFOL*FSTAT(0)=age sex SHO;
run;

*four covariate models:  add to SHO, age, and CHF
- confirmed adding MIORD (step 4);

*-2 LOG L = 2669.216;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO CHF MIORD;
run;

*-2 LOG L = 2673.054;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO CHF sex;
run;

*-2 LOG L = 2672.015;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO CHF CPK;
run;

*-2 LOG L = 2671.983;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO CHF MITYPE;
run;


*5 covariates-add to age, SHO, CHF, and MIORD
note, process ends when next candidate for inlcusion in model does not
```

```
reduce value of -2 LOG L by more than 2, confirm stoppage of fitting;

*-2 LOG L = 2669.216;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO CHF MIORD sex;
run;

*-2 LOG L = 2667.473;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO CHF MIORD CPK;
run;

*-2 LOG L = 2667.479;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO CHF MIORD MITYPE;
run;

*PART 2- MODEL ADEQUACY;

*PH ASSUMPTION MET FOR SEX (not in model);
proc lifetest data=whas plot=(s, lls);
time LENFOL*FSTAT(0);
strata sex;
run;

*CATEGORIZE AGE INTO 2 GROUPS and check PH assumption
NOTE:  The US Census bureau defines elderly as 65+, and most other US Government
programs use this definitition;
data project;
```

```
set whas;
age2=(age ge 65 & age le 98);
run;
*PH ASSUMPTION MET FOR AGE (in model);
proc lifetest data=project plot=(s, lls);
time LENFOL*FSTAT(0);
strata age2;
run;

*PH ASSUMPTION MET FOR SHO (in model);
proc lifetest data=project plot=(s, lls);
time LENFOL*FSTAT(0);
strata SHO;
run;

*PH ASSUMPTION MET FOR CHF (in model);
proc lifetest data=project plot=(s, lls);
time LENFOL*FSTAT(0);
strata CHF;
run;

*PH ASSUMPTION MET FOR MIORD (in model);
proc lifetest data=project plot=(s, lls);
time LENFOL*FSTAT(0);
strata MIORD;
run;

*PH ASSUMPTION MET FOR MITYPE (not in model);
proc lifetest data=project plot=(s, lls);
```

```
time LENFOL*FSTAT(0);
strata MITYPE;
run;
```

*PART 3: STRATIFIED COX MODEL (BY YRGRP)
Compare the estimated coefficients of the stratified model with those
from the fit of the WHAS model. Are there anyimportant differences?
•If the WHAS model was not proportional in any covariates,
examine the effec-tiveness of using these covariates to
define stratification variables.•It is possible that the
effect of the covariates in the WHAS model are not constant across
cohorts defined by YRGRP. Examine this via inclusion of interactions
between YRGRP and model covariates in an extended WHAS
model with YRGRP as a stratification variable;

```
*PH ASSUMPTION NOT MET FOR YRGRP, so STRATIFY BY YRGRP;
proc lifetest data=project plot=(s, lls);
time LENFOL*FSTAT(0);
strata YRGRP;
run;

*ORIGINIAL MODEL VS. STRATIFIED MODEL;
*Original Model
-2 LOG L = 2669.216;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO CHF MIORD;
run;
*Stratified Model
```

```
-2 LOG L = 2177.608;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO CHF MIORD;
strata YRGRP;
run;

*PH ASSUMPTION MET FOR ALL COVARIATES (given in previous parts)
-also, there are no noticeable differences between the new estimated
coefficients- asked in WHAS;

*Fitting an Interaction Model;
data interaction;
set WHAS;
sho_yrgrp=sho*yrgrp;
chf_yrgrp=chf*yrgrp;
age_yrgrp=age*yrgrp;
miord_yrgrp=miord*yrgrp;
run;
*original stratified -2LOGL = 2177.608;


*-2LOGL = 2173.104 SO STEP 1 FORWARD SELECTION FOR ADDITION OF
INTERACTION TERMS
add SHO_YRGRP;
proc phreg data=interaction;
model LENFOL*FSTAT(0)=age SHO CHF MIORD sho_yrgrp;
strata YRGRP;
run;
```

```
*-2LOGL = 2176.825;
proc phreg data=interaction;
model LENFOL*FSTAT(0)=age SHO CHF MIORD chf_yrgrp;
strata YRGRP;
run;

*-2LOGL = 2176.942;
proc phreg data=interaction;
model LENFOL*FSTAT(0)=age SHO CHF MIORD age_yrgrp;
strata YRGRP;
run;

*-2LOGL = 2177.474;
proc phreg data=interaction;
model LENFOL*FSTAT(0)=age SHO CHF MIORD miord_yrgrp;
strata YRGRP;
run;

*step 2- forward selection;

*-2LOGL = 2170.935;
proc phreg data=interaction;
model LENFOL*FSTAT(0)=age SHO CHF MIORD sho_yrgrp chf_yrgrp;
strata YRGRP;
run;

*-2LOGL = 2172.131;
proc phreg data=interaction;
model LENFOL*FSTAT(0)=age SHO CHF MIORD sho_yrgrp age_yrgrp;
```

```
strata YRGRP;
run;

*-2LOGL = 2173.098;
proc phreg data=interaction;
model LENFOL*FSTAT(0)=age SHO CHF MIORD sho_yrgrp miord_yrgrp;
strata YRGRP;
run;

*so, step 2- ADD CHF_YRGRP interaction term to model;

*step 3- forward selection
-2LOGL not reduced by more than 2, so stop selection process here;
*-2LOGL = 2170.920;
proc phreg data=interaction;
model LENFOL*FSTAT(0)=age SHO CHF MIORD sho_yrgrp chf_yrgrp miord_yrgrp;
strata YRGRP;
run;

*-2LOGL = 2170.701;
proc phreg data=interaction;
model LENFOL*FSTAT(0)=age SHO CHF MIORD sho_yrgrp chf_yrgrp age_yrgrp;
strata YRGRP;
run;

*CHECK SEPARATE MODELS FOR EACH STRATA TO SEE DIFFERENCE IN MAIN
EFFECTS
OF COVARIATES ACROSS STRATA (below)
```

-We see that mostly constant across strata for age and MIORD (nothing beyond normal assumed variance)

-so no interaction for age or MIORD covariates confirmed;

;

```
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO CHF MIORD;
strata YRGRP;
run;
proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO CHF MIORD;
where YRGRP=1;
run;

proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO CHF MIORD;
where YRGRP=2;
run;

proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO CHF MIORD;
where YRGRP=3;
run;
```

*NOTICE CHF has stronger relationship for more modern/recent years, and SHO has weaker relationship for more modern/recent years;

```
proc phreg data=interaction;
model LENFOL*FSTAT(0)=age SHO CHF MIORD sho_yrgrp chf_yrgrp age_yrgrp
miord_yrgrp/RL selection=forward;
```

```
strata YRGRP;
run;

*Interaction Model;
*-2LOGL = 2168.803;
proc phreg data=interaction;
model LENFOL*FSTAT(0)=age SHO CHF MIORD sho_yrgrp chf_yrgrp;
strata YRGRP;
run;

proc phreg data=interaction;
model LENFOL*FSTAT(0)=age SHO MIORD sho_yrgrp;
strata YRGRP;
run;

proc phreg data=WHAS;
model LENFOL*FSTAT(0)=age SHO CHF MIORD;
strata YRGRP;
run;
*Note, Original Stratified Model re-coded above (from earlier section)
-2 LOG L = 2176.015, so we conclude that the interaction model isappropriate
COMPUTE ALL CHI SQUARE REJECTION REGIONS FOR REPORT;



*PART 4:   TIME-DEPENDENT COX REGRESSION MODEL
Again, based on the final model you selected, call it the ”WHAS” model,
answer the following question. It is quite possible that differences
in long-term survivalin the grouped cohorts, YRGRP, could be due to different
```

```
lengths of stay in the hospital.
Examine this by creating a dichotomous time-varying covariate comparing
   LENSTAY and LENFOL and adding it to the WHAS model;

*DIFFERENCES IN LONG TERM SURVIVAL PER YRGRP SEEN IN PLOT OF
SURVIVAL CURVES
FROM PART 3 (they are different lengths, etc.) so, we examine this further
show comparative survival plot in this section, and log log survival plots
in stratified model section of results, along with anything else relevant
such as comparative hazard ratio CI's, etc);

data time;
set WHAS;
if (dstat=1 or lenfol < lenstay) then discharge=0;
else discharge=1;
run;

*PH ASSUMPTION MET FOR DISCHARGE;
proc lifetest data=time plot=(s, lls);
time LENFOL*FSTAT(0);
strata discharge;
run;

proc phreg data=time;
model LENFOL*FSTAT(0)=age SHO CHF MIORD discharge/RL;
run;

proc phreg data=time;
model LENFOL*FSTAT(0)=age SHO CHF MIORD discharge/RL;
```

```
where yrgrp=1;
run;

proc phreg data=time;
model LENFOL*FSTAT(0)=age SHO CHF MIORD discharge/RL;
where yrgrp=2;
run;

proc phreg data=time;
model LENFOL*FSTAT(0)=age SHO CHF MIORD discharge/RL;
where yrgrp=3;
run;
```

Discrete Choice Analysis Survey Statistics:

Physical Activity Importance for College Students

GOAL

Along with a healthy diet, regular exercise is key to leading a healthy lifestyle. Unfortunately, regular exercise is usually inconvenient for the typical college student, especially when one must rush from class to class. The goal of this assignment was to discover the importance of physical activity for college students.

VARIABLES

We used four variables, each with two levels (level 1 representing the lower value, level 2 the higher). The most important variable was time the student was willing to walk, which is primarily used to measure how important physical exercise is to the student.

Time spent finding parking was included to help determine if students did not like walking because it took too much time, or because they hated walking. Parking permit cost was included to determine if the student was willing to pay extra to forego walking. Finally, class time was included to determine if the student would choose an earlier time of the day instead of walking an extra few minutes.

	1	**2**
Walking (time)	5 minutes	20 minutes
Parking (time to find)	5 minutes	30 minutes
Parking permit cost	$160	$300
Class time	8am	10am

DESIGN

Our design consisted of a full 2^4factorial design. This means that we have 4 different levels of analysis with 2 levels, a high and low level as described above. In order to determine the selection of the design, we utilized the %choiceeff, %mktruns, %mktdups,

and %mktex macros as outlined by Warren F. Kuhfield in his *Discrete Choice* paper. Below are the optimal designs:

Saturated = 5
Full Factorial = 16

Some Reasonable Design Sizes		Violations	Cannot Be Divided By
8	*	0	
12	*	0	
16	*	0	
6		6	4
10		6	4
14		6	4
5	S	10	2 4
7		10	2 4
9		10	2 4
11		10	2 4

* - 100% Efficient design can be made with the MktEx macro.
S - Saturated Design - The smallest design that can be made.

We can see that we could have used a design size of 8 and 12, however, in order to keep the thoroughness of the questionnaire, we opted for a full design with all 16 options in the results. We believe that 16 options was not overwhelming for the survey and we could afford to ask all combinations.

Next, we identify the optimal way to present the questions. The macro concluded that the best way to represent the choices are in 8 questions with 2 alternatives per question. Below is the final results of the %choiceeff macro along with the variance-covariance matrix of the four factors.

Final Results

Design	4
Choice Sets	8
Alternatives	2
Parameters	4
Maximum Parameters	8
D-Efficiency	8.0000
Relative D-Eff	100.0000
D-Error	0.1250
1 / Choice Sets	0.1250

'Variance-Covariance Matrix'

'00'x	x1 1	x2 1	x3 1	x4 1
x1 1	0.125	-0.000	-0.000	0.000
x2 1	-0.000	0.125	0.000	0.000
x3 1	-0.000	0.000	0.125	0.000
x4 1	0.000	0.000	0.000	0.125

We can see that we have a D-efficiency of 8, which is equal to our number of our choice sets, or in other words, the number of our questions. Also, our D-error and the variances of each factor is equal to 1 over the number of our choice sets, all covariates are zero, and the relative D-efficiency is equal to 100. These are all indicators of an optimal design as explained in *Discrete Choice.* Finally, we ensure that there is no duplication of the choices per set and proceed with the questionnaire.

DEMOGRAPHICS

We were interested in lifestyle choices between male and female students. Our survey collected 72% of responses from females and 28% from males. The disproportion in our population further motivated the analysis between males and females to understand the differences, if any, of student's health choices.

RESULTS

Analysis of Maximum Likelihood Estimates

Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio	Label
Parking1	1	0.66445	0.07711	74.2479	<.0001	1.943	1
Parking2	0	0	.	.	.	.	2
Walk1	1	0.54166	0.07586	50.9894	<.0001	1.719	1
Walk2	0	0	.	.	.	.	2
Permit1	1	1.26730	0.08121	243.5538	<.0001	3.551	1
Permit2	0	0	.	.	.	.	2
ClassTime1	1	-0.64407	0.07687	70.1951	<.0001	0.525	1
ClassTime2	0	0	.	.	.	.	2

We first notice that, based on the p-values, each of the variables is significant. Overall students prefer to park closer, walk less, pay less for a permit, and have a later class time. This is expected.

However, it's interesting that students place a somewhat higher premium on saving time on parking, as opposed to saving time on walking (Parameter estimate of 0.66445 for the former, only .54166 on the latter). Also, students were more willing to save money on a permit (parameter estimate 1.26730) than to save time on either walking or parking. Finally, students were more willing to begin their class at 10am instead of 8am than to walk less.

Next, we look at the results blocked on gender.

Female

Analysis of Maximum Likelihood Estimates							
Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio	Label
Parking1	1	0.49333	0.09237	28.5232	<.0001	1.638	1
Parking2	0	0	.	.	.	.	2
Walk1	1	0.60232	0.09377	41.2629	<.0001	1.826	1
Walk2	0	0	.	.	.	.	2
Permit1	1	1.44754	0.10159	203.0191	<.0001	4.253	1
Permit2	0	0	.	.	.	.	2
ClassTime1	1	-0.72739	0.09602	57.3888	<.0001	0.483	1
ClassTime2	0	0	.	.	.	.	2

Male

Analysis of Maximum Likelihood Estimates							
Parameter	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio	Label
Parking1	1	1.06167	0.14695	52.1938	<.0001	2.891	1
Parking2	0	0	.	.	.	.	2
Walk1	1	0.46481	0.13752	11.4242	0.0007	1.592	1
Walk2	0	0	.	.	.	.	2
Permit1	1	0.92494	0.14563	40.3418	<.0001	2.522	1
Permit2	0	0	.	.	.	.	2
ClassTime1	1	-0.53355	0.13832	14.8793	0.0001	0.587	1
ClassTime2	0	0	.	.	.	.	2

Based on the parameter estimates for both, males and females, it looks like males are less averse to walking for 20 minutes instead of 5 minutes, and also slightly more willing to wake up early for an 8am class. Males are also more willing to pay a premium for a better parking spot. However, females are more willing to spend extra time to find parking.

The data suggests that males are more likely to spend extra time walking, although the results are not conclusive.

CONCLUSION

Overall, the results show that students are not eager to walk more. Health concerns are not important enough motivators. However, they would prefer to walk more than to spend extra time finding parking, paying more for a permit, or waking up earlier to get to class. While this is not the ideal situation, it is encouraging that students are, if not enthusiastic, at least willing to walk extra.

CODE

```
%mktex(2 ** 4, n=2**4, seed=238)
proc print; run;

%mktruns(2 2 2 2)

%choiceff(data=design,    /* candidate set of alternatives            */
model=class(x1-x4 / sta), /* model with stdz orthogonal coding    */
nsets=8,                  /* number of choice sets                */
flags=2,                  /* 2 alternatives, generic candidates   */
seed=289,                 /* random number seed                   */
maxiter=60,               /* maximum number of designs to make    */
options=relative,         /* display relative D-efficiency        */
beta=zero)                /* assumed beta vector, Ho: b=0         */
proc print; var x1-x4; id set; by set; run;
proc print data=bestcov label;
title 'Variance-Covariance Matrix';
id __label;
label __label = '00'x;
var x:;
run;
Title;
```

```
%mktdups(generic, data=best, factors=x1-x4, nalts=2)


data car_res;
input subj gender q1 q2 q3 q4 q5 q6 q7 q8;
datalines;
1 1 1 1 2 2 2 2 1 2
2 1 1 1 2 1 1 1 2 2
3 2 1 2 2 2 2 2 1 2
4 1 1 1 2 1 1 2 2 2
5 1 1 2 2 2 1 1 2 2
6 2 1 1 2 1 2 2 2 1
7 1 1 2 2 2 2 2 2 2
8 1 1 1 2 2 2 1 2 2
9 1 1 1 2 2 1 2 2 2
10 1 1 1 2 2 1 1 2 2
11 1 1 1 2 2 1 1 2 2
12 1 1 2 2 2 1 2 2 1
13 1 1 2 2 2 2 2 2 2
14 1 1 1 2 1 2 2 2 2
15 2 1 2 2 2 2 2 1 2
16 1 1 1 2 2 1 1 2 2
17 1 1 2 2 2 2 2 2 2
18 1 1 1 2 2 1 2 1 2
19 1 1 2 2 1 1 2 2 2
20 1 1 1 2 1 1 1 2 2
21 1 1 2 2 2 1 2 2 2
22 2 1 1 2 1 1 1 2 2
23 1 1 1 2 1 2 2 2 2
24 1 1 1 2 2 2 1 2 2
25 1 2 2 2 2 1 1 1 2
26 2 1 2 2 2 2 2 2 2
27 1 1 1 2 1 1 1 2 2
```

28 1 1 1 2 1 1 1 2 2

29 1 1 1 2 2 1 1 2 2

30 2 1 1 2 2 1 1 2 2

31 2 1 2 2 2 2 2 1 2

32 2 1 1 2 1 1 1 2 2

33 2 1 1 2 1 2 2 2 1

34 1 1 1 2 2 1 1 2 2

35 2 1 2 2 2 2 1 2 2

36 1 1 2 2 2 2 2 1 2

37 2 1 2 2 2 2 2 2 2

38 1 1 2 2 2 1 2 2 2

39 2 1 2 2 2 2 2 1 2

40 2 1 1 2 1 1 2 2 2

41 1 1 1 2 1 1 1 2 2

42 2 1 1 2 1 2 2 2 2

43 1 1 1 2 1 2 1 2 1

44 1 1 1 1 2 2 1 1 1

45 1 1 1 2 1 1 2 2 2

46 1 1 1 2 1 1 2 2 2

47 1 1 1 2 2 1 2 2 2

48 1 1 1 2 2 1 1 2 2

49 1 1 1 2 2 1 2 2 2

50 1 1 1 2 1 2 2 2 2

51 1 1 2 2 1 1 1 2 1

52 1 1 1 2 1 1 1 2 2

53 1 1 1 2 1 1 1 2 2

54 1 1 1 2 2 1 1 2 2

55 2 1 2 2 1 1 2 2 2

56 1 1 1 2 1 1 1 2 2

57 1 1 1 2 1 1 1 2 2

58 1 1 1 2 1 2 2 2 1

59 1 1 2 2 1 2 2 2 2

60 1 1 1 2 1 2 2 2 1

61 2 1 1 2 1 2 2 2 2

62 2 1 2 2 2 2 2 1 2

63 2 1 2 2 2 2 2 2 2

64 2 1 1 2 2 1 1 2 2

65 2 1 2 2 2 2 2 1 2

66 2 1 1 2 2 1 1 2 2

67 2 1 2 2 2 2 2 1 2

68 2 1 2 2 2 1 2 2 2

69 1 1 1 2 1 1 2 2 2

70 1 1 1 2 2 2 2 1 2

71 1 1 1 2 1 1 1 2 2

72 1 1 2 2 2 2 2 1 2

73 1 2 2 2 2 2 2 1 2

74 1 1 1 2 2 1 1 2 2

75 1 1 1 2 1 1 1 2 2

76 1 1 1 2 2 1 1 2 2

77 1 1 1 2 1 1 1 2 2

78 1 1 1 2 2 1 1 2 2

79 1 1 1 2 2 1 1 2 2

80 1 1 1 2 1 2 2 2 2

81 1 1 2 2 1 1 2 2 2

82 2 1 2 2 1 2 2 2 2

83 2 1 2 2 1 1 1 2 2

84 2 1 2 2 2 2 2 1 2

85 1 1 1 2 2 1 2 2 2

86 1 1 1 2 2 2 2 1 2

87 1 1 1 2 2 1 1 2 2

88 1 1 2 2 1 2 2 2 1

89 2 1 2 2 2 2 2 1 2

90 1 1 1 2 1 1 1 2 2

91 1 1 2 2 2 2 1 2 2

92 1 2 2 2 1 1 1 2 2

93 1 1 1 2 1 1 2 2 2

94 1 1 2 2 1 2 2 2 2

95 2 1 1 2 2 1 1 2 2

96 1 2 2 2 1 1 2 2 2

97 2 1 1 1 1 2 2 1 1

98 1 1 2 2 2 1 1 1 2

99 2 1 2 1 2 2 2 1 1

100 1 1 1 2 1 1 2 1 2

101 2 1 1 2 2 1 1 2 2

102 2 1 1 2 1 2 2 2 2

103 2 1 2 2 1 2 2 2 2

104 2 1 1 2 2 1 1 2 2

105 1 1 1 2 1 1 2 2 2

106 2 1 1 2 1 1 1 2 2

107 1 1 1 2 1 1 1 2 2

108 1 1 1 2 1 2 2 2 1

109 2 1 2 2 2 1 2 2 2

110 1 1 1 2 1 2 2 2 2

111 1 1 2 2 2 1 1 2 2

112 1 1 1 2 1 1 2 2 2

113 2 2 2 2 1 1 2 2 2

114 1 1 2 2 2 1 1 2 2

115 1 1 2 2 2 1 2 2 2

116 1 1 1 2 2 2 1 2 2

117 1 1 1 2 1 2 2 2 1

118 1 1 1 2 1 1 2 1 2

119 1 1 1 2 1 1 1 2 2

120 1 1 1 2 1 1 1 2 2

121 2 1 1 2 2 1 1 2 2

122 1 1 1 2 1 1 1 2 2

123 1 1 1 2 2 1 2 2 2

```
124 1 1 1 2 1 1 1 2 2
125 1 1 2 2 2 1 2 2 2
126 1 1 1 2 1 1 1 2 2
127 1 1 1 2 2 1 1 2 2
128 2 1 1 2 2 1 2 2 2
129 1 1 1 2 2 1 1 2 2
130 1 1 1 2 1 1 1 2 2
131 1 1 2 1 2 2 1 2 1
132 1 1 1 2 2 1 1 2 2
133 1 1 1 2 2 1 1 2 2
134 1 1 2 2 2 2 2 2 2
135 1 1 1 2 2 1 1 2 2
136 1 1 1 2 2 1 1 2 2
137 1 1 2 2 1 2 2 2 2
138 2 1 2 2 2 2 2 1 2
139 2 2 2 2 2 1 1 1 2
140 1 1 2 2 2 2 2 1 2
141 1 1 1 2 2 1 1 2 2
142 1 1 2 2 1 1 2 2 2
143 1 1 2 2 2 2 2 2 2
144 1 1 2 2 2 2 2 2 1
run;

/*design*/
data car_des;
input set Parking Walk Permit ClassTime;
datalines;
1 1 2 1 2
1 2 1 2 1
2 2 2 1 2
2 1 1 2 1
3 2 2 2 1
```

```
3 1 1 1 2
4 2 1 2 2
4 1 2 1 1
5 2 1 1 1
5 1 2 2 2
6 2 2 1 1
6 1 1 2 2
7 1 2 2 1
7 2 1 1 2
8 2 2 2 2
8 1 1 1 1
run;

proc format;
value parking 1='Find parking in 5 Min' 2='Find parking in 30 min';
value walk 1='5 min walk to class' 2='20 min walk to class';
value permit 1='$160 parking permit' 2='$300 parking permit';
value classtime 1='8am class time' 2='10am class time';
run;

%mktmerge(design=car_des, data=car_res, out=c_res2,nsets=8, nalts=2, setvars=q1-q8);

proc transreg design=5000 data=c_res2 nozeroconstant norestoremissing;
model class (parking walk permit classtime/ zero=none order=formatted) /lprefix=0;
output out=c_coded(drop=_type_ _name_ intercept);
id subj gender set c;
run;

proc phreg data=c_coded brief;
model c*c(2) = &_trgind / ties=breslow;
strata subj set;
run;
```

SURVEY

1. What is your gender? (Circle one)
   Male or Female
2. What is your height? (Circle one)
   Below 4'11" 5'0"-5'3" 5'4"-5"7" 5'8"-5'11" 6'0"-6'3" Over 6'3"
3. Do you live on campus? (Circle one)
   Yes No
4. Do you drive to school? (Circle One)
   Yes No
5. Would you rather:
   a. Take 5 minutes to find parking, walk 20 minutes to class, pay $160 for a parking permit, and have a 10 am class, or
   b. Take 30 minutes to find parking, walk 5 minutes to class, pay $300 for a parking pass, and have an 8 am class
6. Would you rather:
   a. Take 30 minutes to find parking, walk 20 minutes to class, pay $160 for a parking permit, and have a 10 am class, or
   b. Take 5 minutes to find parking, walk 5 minutes to class, pay $300 for a parking pass, and have an 8 am class
7. Would you rather:
   a. Take 30 minutes to find parking, walk 20 minutes to class, pay $300 for a parking permit, and have a 10 am class, or
   b. Take 5 minutes to find parking, walk 5 minutes to class, pay $160 for a parking pass, and have an 10 am class
8. Would you rather:
   a. Take 30 minutes to find parking, walk 5 minutes to class, pay $160 for a parking permit, and have a 10 am class, or
   b. Take 5 minutes to find parking, walk 20 minutes to class, pay $160 for a parking pass, and have an 8 am class
9. Would you rather:
   a. Take 30 minutes to find parking, walk 5 minutes to class, pay $160 for a parking permit, and have an 8 am class, or
   b. Take 5 minutes to find parking, walk 20 minutes to class, pay $300 for a parking pass, and have an 10 am class
10. Would you rather:
    a. Take 30 minutes to find parking, walk 20 minutes to class, pay $160 for a parking permit, and have an 8 am class, or
    b. Take 5 minutes to find parking, walk 5 minutes to class, pay $300 for a parking pass, and have an 10 am class
11. Would you rather:
    a. Take 5 minutes to find parking, walk 20 minutes to class, pay $300 for a parking permit, and have an 8 am class, or
    b. Take 30 minutes to find parking, walk 5 minutes to class, pay $160 for a parking pass, and have an 10 am class

PROC SQL (IN SAS) EXPERIENCE

PROC SQL EXPERIENCE EXAMPLE 1

```
proc sql;
alter table prev
modify year format = 6.0, yearrecordbegan format = 6., yearrecordends
format = 6.;
select *
from prev;
quit;
```

```
proc sql;
select count(*)as countrec
from prev;
Quit;
```

countrec
18302

```
proc contents data = prev short;
Run;
```

The CONTENTS Procedure

Alphabetic List of Variables for WORK.PARTONE

ELEVATIONINFEET LATITUDE LONGITUDE MaxSWEinches REGION STATIONID
STATIONNAME YEAR YEARRECORDBEGAN YEARRECORDENDS

```
select count(distinct stationid) label 'Number of Stations'
from prev;
Quit;
```

Number of Stations
262

```
proc sql;
select distinct stationname, yearrecordbegan, yearrecordends,
(yearrecordends - yearrecordbegan + 1) as yearsrecorded label 'Years Recorded'
from prev;
Quit;
```

STATION NAME	YEAR RECORD BEGAN	YEAR RECORD ENDS	Years Recorded
ABBEY	1963	2012	50
ADIN MOUTAIN	1930	2012	83
AGNEW PASS	1930	2012	83
ALPHA	1965	2012	48
ANTELOPE RIDGE	1963	2012	50
WILMA LAKE	1946	2012	67
WOLFORD CABIN	1949	2012	64
WOODCHUCK MEADOW	1930	2012	83
WRIGHTS LAKE	1956	2012	57
YUBA PASS	1937	2012	76

```
proc sql;
select distinct year, count(stationid) as stationsperyears label 'Stations Recorded for
That Year'
from prev
group by year;
order by year;
Quit;
```

YEAR	Stations Recorded for That Year
1910	3
1911	3
1912	3
1913	6
2009	262
2010	262
2011	262
2012	262

```
proc sql;
select stationname, max(maxsweinches) as maxsweperstation label 'Maximum Annual Snow Water Equivalent'
from problem3
group by stationname;
order by stationname;
Quit;
```

STATION NAME	Maximum Annual Snow Water Equivalent
ABBEY	65.4
ADIN MOUTAIN	33.6
AGNEW PASS	65.6
ALPHA	84.1
WOLFORD CABIN	92.8
WOODCHUCK MEADOW	81.3
WRIGHTS LAKE	83.5
YUBA PASS	76.7

```
Data prev;
modify prev;
if stationname = 'ADIN MOUTAIN' then stationname = 'ADIN MOUNTAIN';
Run;
```

```
proc sql;
select count(distinct region) as numofregions label 'Number of Regions'
from prev;
Quit;
```

Number of Regions
12

```
proc sql;
select distinct region, count(distinct stationid) label 'Number of Stations'
from prev
group by region;
Quit;
```

REGION	Number of Stations
AMERI_MOKEL	31
FEATHER	25
KAWEAH_TULE_KERN	24
KINGS	22
NORTH COAST	20
NORTH LAHONTAN	13
SACRAMENTO	19
SAN JOAQUIN	25
SOUTH LAHONTAN	27
STANISLAUS	15
TUOLU_MERCED	22
YUBA	19

```
proc sql;
```

```
select distinct stationid, stationname, max(maxsweinches) as totalmaxswe label
'Highest Maximum SWE Value', region, elevationinfeet, latitude, longitude

from prev

where year ge 1963

group by stationid;

Quit;
```

STATION ID	STATION NAME	Highest Maximum SWE Value	REGION	ELEVATION(IN FEET)	LATITUDE	LONGITUDE
3LK	THREE LAKES	83.4	FEATHER	6250.0	39.97300	121.21300
ABN	LAKE AUDRAIN	73.8	AMERI_MOKEL	7300.0	38.82000	120.03700
ABY	ABBEY	65.4	FEATHER	5650.0	39.95500	120.53800
...						
WRG	WRIGHTS LAKE	83.5	AMERI_MOKEL	6900.0	38.84700	120.23300
WRN	WARNER CREEK	39.6	FEATHER	5100.0	40.38700	121.31000
YBP	YUBA PASS	62.3	YUBA	6700.0	39.61700	120.49600

PROC SQL EXPERIENCE EXAMPLE 2

```
input prodnum prodname $ 6-27 manunum prodtype$ 33-43 rtlcost dollar7.;
cards;
5009 Dream Machine          500 Workstation $3,200
4506 Business Machine       450 Workstation $3,345
2101 Travel Laptop          400 Laptop      $2,760
2212 Analog Cell Phone      230 Phone       $35
4509 Digital Cell Phone     245 Phone       $175
5003 Office Phone           560 Phone       $145
1110 Spreadsheet Software   134 Software    $300
1200 Database Software      113 Software    $799
3409 Statistical Software   243 Software    $1,899
2102 Wordprocessor Software 245 Software    $345
2200 Graphics Software      246 Software    $599
;
run;

proc sql;
insert into prev2
set prodnum = 3480, prodname = 'Desktop Computer', manunum = 780,
        prodtype = 'Workstation', rtlcost = 1799;
        select*
        from prev2;
quit;
```

prodnum	prodname	manunum	prodtype	rtlcost
5009	Dream Machine	500	Workstation	2560
4506	Business Machine	450	Workstation	2676
2101	Travel Laptop	400	Laptop	2208
2212	Analog Cell Phone	230	Phone	28
4509	Digital Cell Phone	245	Phone	140
5003	Office Phone	560	Phone	116
1110	Spreadsheet Software	134	Software	360
1200	Database Software	113	Software	958.8
3409	Statistical Software	243	Software	2278.8
2102	Wordprocessor Software	245	Software	414
2200	Graphics Software	246	Software	718.8
3480	Desktop Computer	780	Workstation	1439.2
3480	Desktop Computer	780	Workstation	1799

```
proc sql;
update prev2
set rtlcost =
case
        when prodtype = 'Software    ' then rtlcost*1.20
        else rtlcost*.80
        end;
        select*
        from prev2;
Quit;
```

prodnum	prodname	manunum	prodtype	rtlcost
5009	Dream Machine	500	Workstation	2048
4506	Business Machine	450	Workstation	2140.8
2101	Travel Laptop	400	Laptop	1766.4
2212	Analog Cell Phone	230	Phone	22.4
4509	Digital Cell Phone	245	Phone	112
5003	Office Phone	560	Phone	92.8
1110	Spreadsheet Software	134	Software	432
1200	Database Software	113	Software	1150.56
3409	Statistical Software	243	Software	2734.56
2102	Wordprocessor Software	245	Software	496.8
2200	Graphics Software	246	Software	862.56
3480	Desktop Computer	780	Workstation	1151.36
3480	Desktop Computer	780	Workstation	1439.2

```
proc sql;
title 'Product Information';
footnote "Updated on &sysdate";
select prodnum label 'Product Number', prodname label 'Product Name', manunum
label 'Manufacturer Number',
        prodtype label 'Product Type', rtlcost format = dollar8.2 label 'Retail Unit Cost'
from prev2;
quit;
```

Product Information

Product Number	Product Name	Manufacturer Number	Product Type	Retail Unit Cost
5009	Dream Machine	500	Workstation	$2048.00
4506	Business Machine	450	Workstation	$2140.80
2101	Travel Laptop	400	Laptop	$1766.40
2212	Analog Cell Phone	230	Phone	$22.40
4509	Digital Cell Phone	245	Phone	$112.00
5003	Office Phone	560	Phone	$92.80
1110	Spreadsheet Software	134	Software	$432.00
1200	Database Software	113	Software	$1150.56
3409	Statistical Software	243	Software	$2734.56
2102	Wordprocessor Software	245	Software	$496.80
2200	Graphics Software	246	Software	$862.56
3480	Desktop Computer	780	Workstation	$1151.36
3480	Desktop Computer	780	Workstation	$1439.20

Updated on 11MAY17

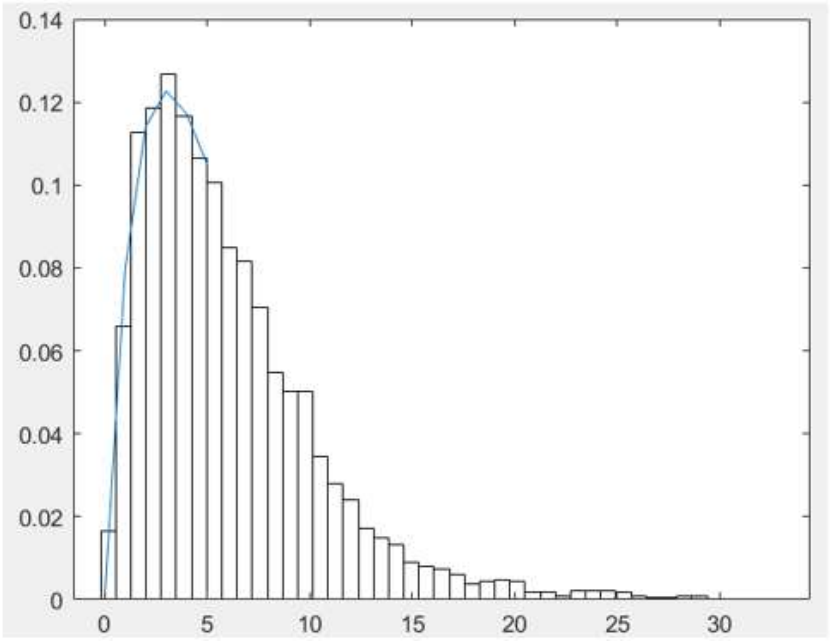
Simulation and Density Estimation in MATLAB

Monte Carlo Markov Chain (MCMC) Metropolis-Hasting Sampler, Random Walk Sampler, and Monte Carlo Simulation with Bootstrap and Jackknife Resampling, PART I in MATLAB

EXAMPLE 1 OF EXPERIENCE: Monte Carlo Markov Chain (MCMC) Metropolis-Hasting Sampler for Gamma (2,3) with Normal proposal density

```
strg = '(x.*(exp(1).^(-x/3))./9';
gamma = inline(strg,'x');
strg = '1/sig*exp(-0.5*((x-mu)/sig).^2)';
norm = inline(strg,'x','mu','sig');
n = 10000;
sig = 2;
x = zeros(1,n);
x(1) = randn(1);
for i = 2:n
   y = normrnd(x(i-1),sig);
   u = rand(1);
   alpha = min([1, gamma(y)*norm(x(i-1),y,sig)/...
     (gamma(x(i-1))*norm(y,x(i-1),sig))]);
   if u <= alpha
      x(i) = y;
   else
      x(i) = x(i-1);
   end
end
n1=.1*n;
x=x(n1+1:n);
n = length(x);
h = 3.5*std(x)*n^(-1/3);
t0 = min(x) - 1;
tm = max(x) + 1;
```

OUTPUT:



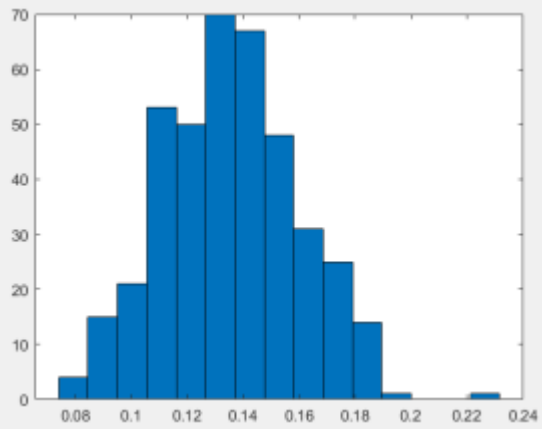
```
rng = tm - t0;
nbin = ceil(rng/h);
bins = t0:h:(nbin*h + t0);
vk = histc(x,bins);
fhat = vk/(n*h);
fhat(end) = [];
tm = max(bins);
bc = (t0+h/2):h:(tm-h/2);
bar(bc,fhat,1,'w')
hold on
xx=0:1:5;
trueden=gamma(xx);
plot(xx,trueden)
hold off
```

The MCMC Metro Hasting Sampler accepted an accurate posterior distribution for Gamma (2,3)

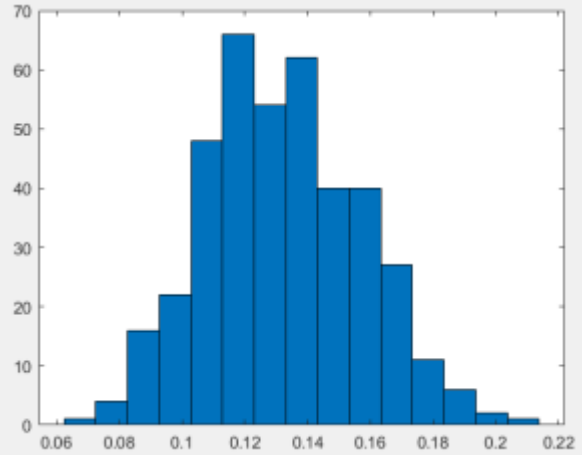
EXAMPLE 2: We use Monte Carlo (MC) Simulation to compare performance of bootstrap and jackknife methods for estimation of standard error and bias of the sample second central moment. For every MC trial, we generate 100 standard normal random variables and calculate the bootstrap and jackknife estimates.

```
n=400;
jack_SE =zeros(n,1);
boot_SE = zeros(n,1);
jack_bias = zeros(n,1);
boot_bias = zeros(n,1);
for i = 1:n
Y = normrnd(0, 1, 100, 1);
[jackB, jackSE, jv] = csjack(Y, 'mom');
[bootreps, bootsam] = bootstrp(100, 'mom', Y);
SEboot = std(bootreps, 1);
jack_SE(i) = jackSE; boot_SE(i) = SEboot;
jack_bias(i)= jackB;
```

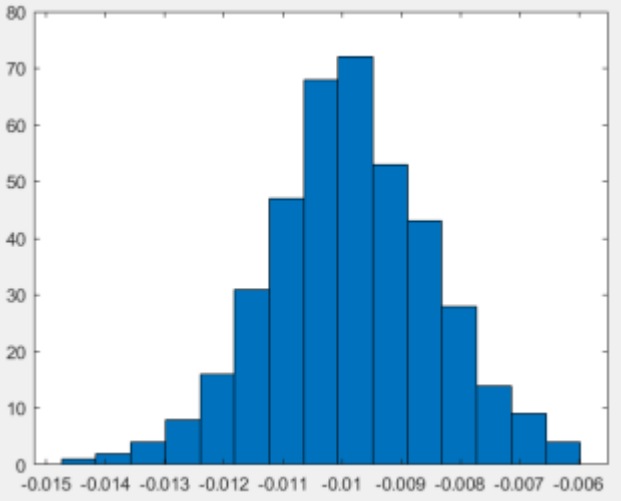
```
boot_bias(i) = mean(bootreps)-1;
end
%Histogram of jackknife SE
 [fr_jackSE,edge_jackSE] = hist(jack_SE, linspace(min(jack_SE), max(jack_SE),15));
bar(edge_jackSE, fr_jackSE, 1)
```



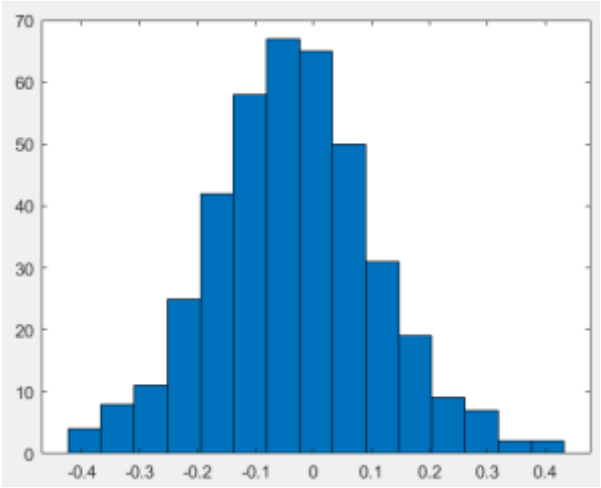

```
%Histogram of bootstrap SE
 [fr_bootSE, edge_bootSE] = hist(boot_SE, linspace(min(boot_SE), max(boot_SE),15));
bar(edge_bootSE, fr_bootSE, 1)
```




```
%Histogram of jackknife bias
 [fr_jackbi, edge_jackbi] = hist(jack_bias, linspace(min(jack_bias),max(jack_bias), 15));
bar(edge_jackbi, fr_jackbi, 1)
```




```
%Histogram of bootstrap bias

 [fr_bootbi, edge_bootbi] = hist(boot_bias, linspace(min(boot_bias),max(boot_bias), 15));

bar(edge_bootbi, fr_bootbi,1)
```



Judging by the histograms, the Bootstrap and Jackknife standard errors and biases are very similar. This makes sense, as Jackknife is approximation to Bootstrap, and Monte Carlo Simulations approximate well/similarly

EXAMPLE 3:

We generate Monte Carlo for Beta(alpha, beta) with support 0,1 using,
q(.|xsubt) iid Unif(xsubt - .5, xsubt+ .5) and for,
a) alpha=beta=3
b) alpha=beta=.5
for each case,
i) we generate a random sample of size 100 using MH (Burn in 5%)

ii) kernel density estimation for random sample in 1

iii) plot histogram and kernel density of random sample with true density superimposed

iv) do MC simulation to estimate ISE of estimation in 2 and MSE at xsub0= .2, .5, and .8

MATLAB CODE A)

```
n=1000;
x = zeros(1,n);
x(1) = randn(1);

alpha = 0.5;
Beta = 0.5;

for i = 2:n
   y=unifrnd(x(i-1)-.5,x(i-1)+.5);
   u=rand(1);
   if y <= 0 || y>=1
      x(i)=x(i-1);
   else

      alpha0=min([1 betapdf(y,alpha,Beta)/betapdf(x(i-1),alpha,Beta)]);
      if u <= alpha0
         x(i)=y;
      else
         x(i) = x(i-1);
      end
   end
```

```
end

xx = x(0.05*n+1:n);

x=0.01:0.01:0.99;
fhat = zeros(size(x));
n=length(xx);
h = 1.06*std(xx)*n^(-1/5);
for kl=1:n
    f=exp(-(1/(2*h^2))*(x-xx(kl)).^2)/sqrt(2*pi)/h;
    fhat = fhat+f/(n);
end

tm = 1;
t0 =0;
bins = t0:h:tm;
vk = histc(xx,bins);
vk(end) = [];
fhat1 = vk/(n*h);
tm = max(bins);
bc = (t0+h/2):h:(tm-h/2);
bar(bc,fhat1,1,'w')

hold on
plot(x,fhat);

plot(x,betapdf(x,alpha,Beta),'-r')
hold off

ISE = sum((fhat-betapdf(x,alpha,Beta)).^2);

M=100;
ind=1;
```

```
MSE=zeros(3,1);
format long
for x0=[.2 .5 .8]
    true_f = betapdf(x0,alpha,Beta);
    fhats = zeros(M,1);

    for j = 1:M
        n=1000;
x = zeros(1,n);
x(1) = randn(1);
        for i = 2:n
            y=unifrnd(x(i-1)-.5,x(i-1)+.5);
            u=rand(1);
            if y <= 0 || y>=1
                x(i)=x(i-1);
            else

                alpha0=min([1 betapdf(y,alpha,Beta)/betapdf(x(i-1),alpha,Beta)]);
                if u <= alpha0
                    x(i)=y;

                else
                    x(i) = x(i-1);
                end

            end
        end
        xx = x(0.05*n+1:n);
        x=0.01:0.01:0.99;
        fhat = zeros(size(x));
        n=length(xx);
        for kl=1:n
            f=exp(-(1/(2*h^2))*(x-xx(kl)).^2)/sqrt(2*pi)/h;
```

```
        fhat = fhat+f/(n);
    end
    index = find(x==x0);
    fhats(j)=fhat(index);

  end
  MSE(ind) = 1/M*sum((fhats-true_f).^2);
  ind=ind+1;
end
```

B)

```
n=1000;
x = zeros(1,n);
x(1) = randn(1);

alpha = 3;
Beta = 3;

for i = 2:n
  y=unifrnd(x(i-1)-.5,x(i-1)+.5);
  u=rand(1);
  if y <= 0 || y>=1
    x(i)=x(i-1);
  else

    alpha0=min([1 betapdf(y,alpha,Beta)/betapdf(x(i-1),alpha,Beta)]);
    if u <= alpha0
      x(i)=y;
    else
      x(i) = x(i-1);
    end
  end
```

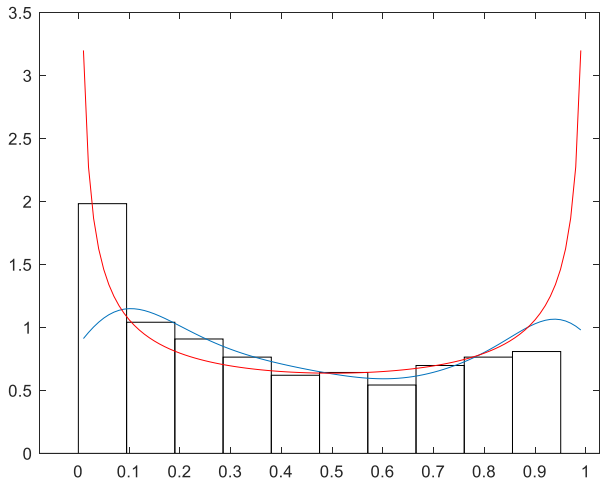
```
end

xx = x(0.05*n+1:n);

x=0.01:0.01:0.99;
fhat = zeros(size(x));
n=length(xx);
h = 1.06*std(xx)*n^(-1/5);
for kl=1:n
    f=exp(-(1/(2*h^2))*(x-xx(kl)).^2)/sqrt(2*pi)/h;
    fhat = fhat+f/(n);
end

tm = 1;
t0 =0;
bins = t0:h:tm;
vk = histc(xx,bins);
vk(end) = [];
fhat1 = vk/(n*h);
tm = max(bins);
bc = (t0+h/2):h:(tm-h/2);
bar(bc,fhat1,1,'w')

hold on
plot(x,fhat);

plot(x,betapdf(x,alpha,Beta),'-r')
hold off

ISE = sum((fhat-betapdf(x,alpha,Beta)).^2);

M=100;
ind=1;
```

```
MSE=zeros(3,1);
format long
for x0=[.2 .5 .8]
    true_f = betapdf(x0,alpha,Beta);
    fhats = zeros(M,1);

    for j = 1:M
        n=1000;
x = zeros(1,n);
x(1) = randn(1);
        for i = 2:n
            y=unifrnd(x(i-1)-.5,x(i-1)+.5);
            u=rand(1);
            if y <= 0 || y>=1
                x(i)=x(i-1);
            else

                alpha0=min([1 betapdf(y,alpha,Beta)/betapdf(x(i-1),alpha,Beta)]);
                if u <= alpha0
                    x(i)=y;

                else
                    x(i) = x(i-1);
                end

            end
        end
        xx = x(0.05*n+1:n);
        x=0.01:0.01:0.99;
        fhat = zeros(size(x));
        n=length(xx);
        for kl=1:n
            f=exp(-(1/(2*h^2))*(x-xx(kl)).^2)/sqrt(2*pi)/h;
```

```
        fhat = fhat+f/(n);
    end
    index = find(x==x0);
    fhats(j)=fhat(index);

end
MSE(ind) = 1/M*sum((fhats-true_f).^2);
ind=ind+1;
end
```

MATLAB OUTPUT:

A) Note, blue line is kernel density estimation, red is true density, white box is histogram

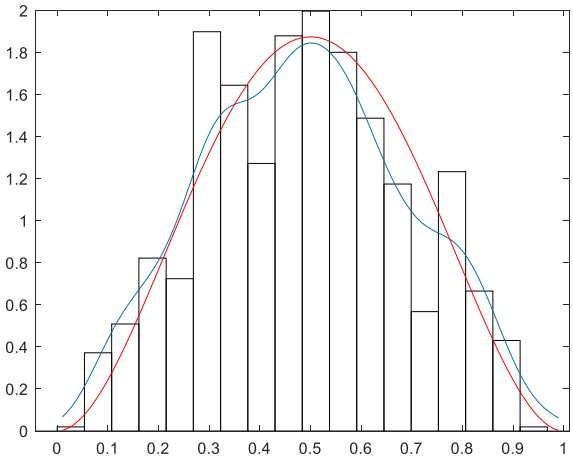



```
ISE =

  16.981861336578259
```

*Note, MSE below for xsub0= .2, .5, and .8

```
MSE =

   0.263102121883454
   0.152985577404734
   0.239289083999211
```

B)




```
ISE =

    0.475094118358671
```

MSE for xsub0= .2, .5, and .8

```
0.255544375751707
1.072529782376164
0.227362170229618
```

DISCUSSION:

Part A Comments:

MC simulation of estimate of ISE of kernel estimation is around equal to 17. We see from the plot of the histogram and kernel density with true density superimposed that this is not a very good estimation. MSE at xsub0= .2, .5, and .8 is about .2631, .153, .2393, which is poor to decent, given that alpha and beta = .5

Part B Comments:

MC simulation of estimate of ISE of kernel estimation is around equal to 0.475. We see from the plot of the histogram and kernel density with true density superimposed that this is a good estimation. MSE at xsub0= .2, .5, and .8 is about .2555, 1.07253, and .2274, which is decent given alpha and beta = 3

EXAMPLE 4: Additionally uses Random Walk

Parameter theta= proportion of defective items

Prior theta iid Uniform(theta,1)

X1…Xn iid Bernouli(theta=p) (x = 1 with probability of theta, x=0 else)

f(x|theta) = theta^y*(1-theta)^n-y

Likelihood = theta^SUM(Xsubi)) * (1-theta)^(n-SUM(Xsubi))

First, we generate a hypothesized sample of size 100 from Bernoulli(0,2); x = 1 with probability of .2, x = 0 with probability of .8. Initial thetanot(theta0) iid Uniform(0,1). Also, we plot the generated MC and calculate the mixrate. Then we use a M-H sampler to generate MC of size 2000, where invariant distribution is given by the posterior distribution of theta|x. Candidate Y iid Uniform(c,d) where c = max(0, thetasubt - .5) and d = min(1, thetasubt + .5). Y iid Uniform(thetasubt - .5, thetasubt +.5). We use the later 1500 MC to calculate mean and variance of theta. In fact, we know that the posterior distribution, theta|x iid Beta(y+1, n-y+1), y = SUM(xi). Now we generate MC from Beta(y+1,n-y+1) using “Random-walk Sampler” where a= -.5 and b= .5, and calculate the mean and variance

Zsubt= a + (b-a)*u, where u iid Uniform(0,1)

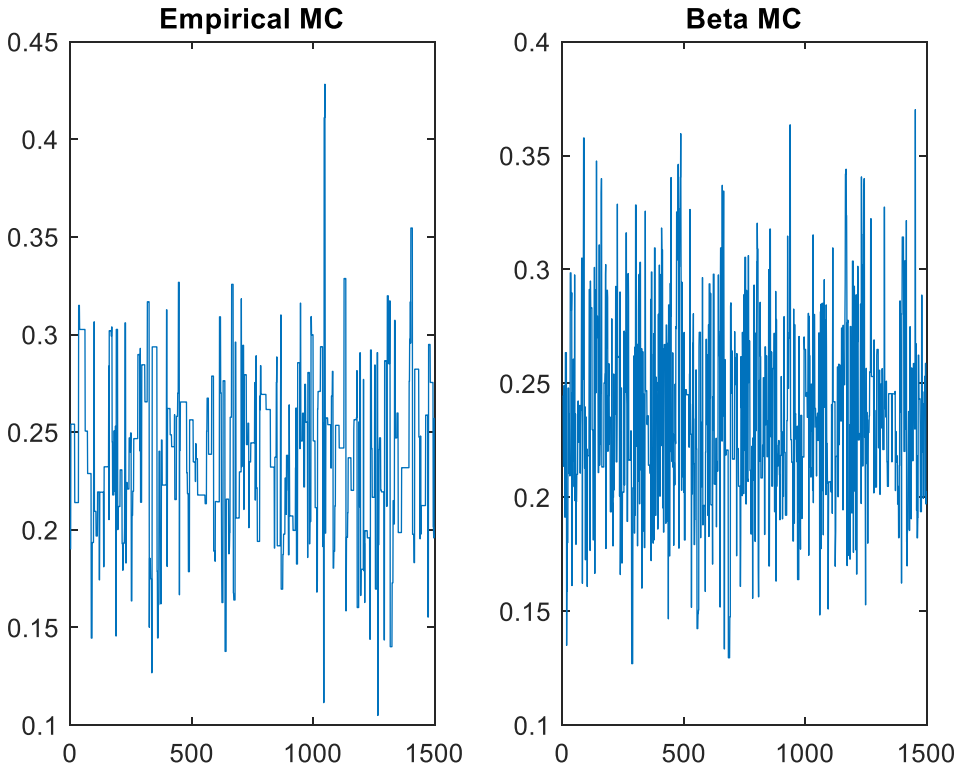
Y= Xsubt + Zsubt/sqrt(12)

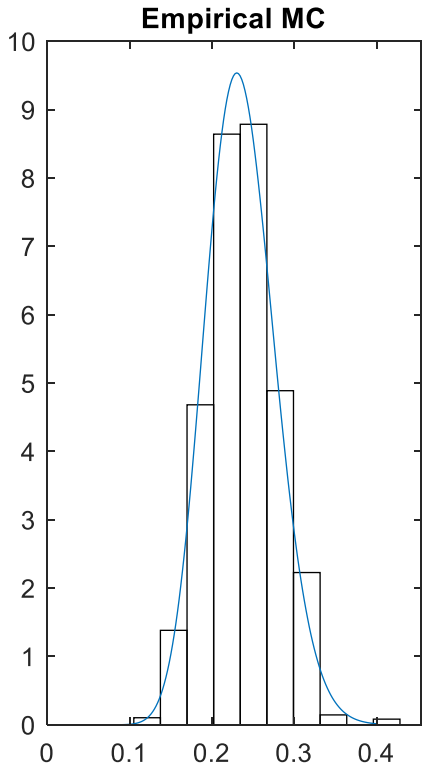
```
%Generating the Bernoulli sample
x=zeros(1,100);
for i=1:100
   u=rand;
   if u <= .2
      x(i)=1;
   else
      x(i)=0;
   end
end
```

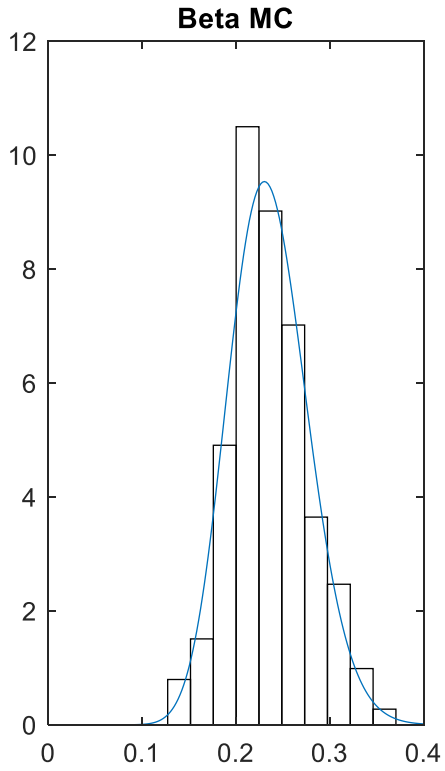
```
%Generating MC with the sample
theta=zeros(1,2000);
theta(1)=rand;
iacp=0; %COUNTING ACCEPTED NUMBERS
for i = 2:2000
    %uniform candidate distribution
    c= max(0,theta(i-1)-.5);
    d= min(1,theta(i-1)+.5);
    y=unifrnd(c,d);
    %generating a uniform for comparison
    u=rand(1);
    alpha= min([1, ((y^(sum(x)).*(1-y).^(100-sum(x)))./((theta(i-1).^(sum(x)).*(1-theta(i-1)).^(100-
(sum(x)))))]);
    if u <= alpha
        theta(i)=y;
        iacp = iacp + 1;
    else
        theta(i)=theta(i-1);
    end
end
theta= theta(501:2000);
mixrate= iacp./2000;
mean0=mean(theta);
variance0= var(theta);
theta1(i)=rand;
iacp1=0;

for i = 2:2000
    %"Random Walk"
    y1=theta1(i-1) + (rand -.5)./sqrt(12);
      u1=rand(1);
  alpha1=min([1, betapdf(y1,sum(x)+1,101-sum(x))...
      ./betapdf(theta1(i-1),sum(x)+1,101-sum(x))]);
```

```
  if u1 <= alpha1
     theta1(i)=y1;
     iacp1= iacp1 + 1;
  else
     theta1(i)= theta1(i-1);
  end
end
theta1=theta1(501:2000);
mixrate1= iacp1/2000;
mean1= mean(theta1);
variance1= var(theta1);

%Plot MCs
figure(1)
subplot(121)
l=linspace(1,1500,1500);
plot(l,theta)
title('Empirical MC')
subplot(122)
plot(l,theta1)
title('Beta MC')

%plot both MCs, with true densities superimposed
figure(2)
xx= 0:.002:.4;
yy= betapdf(xx,sum(x)+1,101-sum(x));
[M,H]=hist(theta);
M=M./(H(2)-H(1))./1500;
subplot(121)
bar(H,M,1,'w')
hold on
plot(xx,yy)
hold off
```

```
title('Empirical MC')

[H1,M1]=hist(theta1);
H1=H1./(M1(2)-M1(1))./1500;
subplot(122)
bar(M1,H1,1,'w')
hold on
plot(xx,yy)
hold off
title('Beta MC')
```








```
mixrate =

    0.1810
```

(plot above)

MH BELOW

```
mean0 =

    0.2381

variance0 =

    0.0018
```

RANDOM WALK BELOW

```
mean1 =

    0.2360

variance1 =

    0.0017
```

MH and Random Walk have similar means and variances.

FINAL EXAMPLE OF PART 1:

A sample of size 200 has been taken iid from the estimated mixture normal distribution below….

$$\delta N\left(\mu_1, \sigma_1^2\right) + (1-\delta) N\left(\mu_2, \sigma_2^2\right)$$

-We are interested in Monte Carlo inference on δ

Assuming a Uniform(0,1) prior distribution for δ, we use MCMC techniques to construct a Markov chain of size 1000 whose stationary distribution equals the posterior density of δ.

-For μ's and σ's, we use the estimates from the finite mixture.

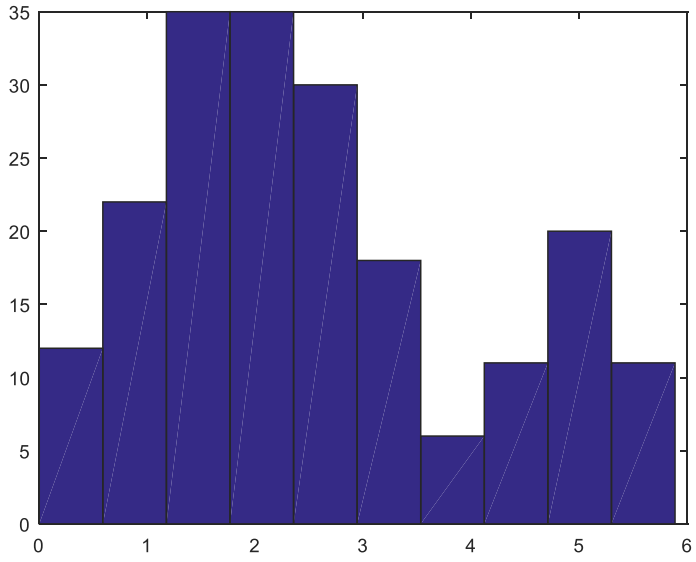
-After 10% burn-in, we provide the histogram of the chain, Monte Carlo estimate of the mean E(δ) and the variance Var(δ). We construct an approximate 90% CI for E(δ) and give the mix rate.

```
muin = [2 5];

varin = [1./3 2./3];

piesin = [1./2 1./2];

maxiter = 100;

tol = 0.001;


[pies mus vars] = csfinmix(data, muin, varin, piesin, maxiter, tol);

n = 4000;

x = zeros(n,1);

r = rand(1,n);

ind = length(find(r <= pies(1)));
```

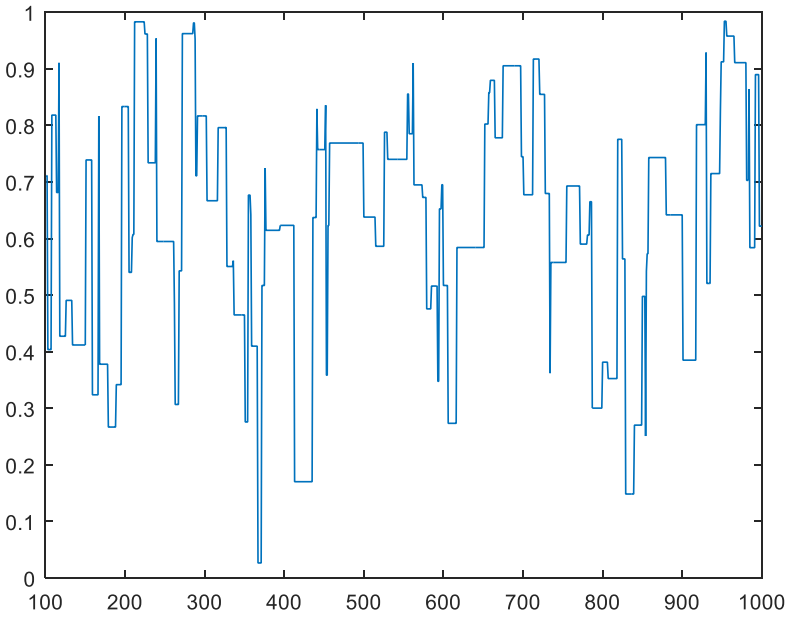
```
x(1:ind)= randn(ind,1)*sqrt(vars(1)) + mus(1);
x(ind+1:n)= randn(n-ind,1)*sqrt(vars(2)) + mus(2);

figure(1)
hist(data)

n = 1000;
theta = zeros(1,n);
theta(1) = rand(1);
iacc = 0;

strg = 'pies(1)*normpdf(x, mus(1),sqrt(sig(1)))+pies(2)*normpdf(x,mus(2),sqrt(sig(2)))'
dist = inline(strg, 'x');

for i = 2:n
   y = unifrnd(max([0,theta(i-1)-.5]), min([1,theta(i-1) + .5]) );
   u = rand;

   alpha = min([1 , pies(1)*normpdf(y, mus(1),sqrt(vars(1)))+pies(2)*normpdf(y,mus(2),sqrt(vars(2)))...
      /pies(1)*normpdf(theta(i-1), mus(1),sqrt(vars(1)))+pies(2)*normpdf(theta(i-1),mus(2),sqrt(vars(2)))]);
   if u < alpha
      theta(i) = y;
      iacc = iacc + 1;
   else
      theta(i) = theta(i-1);
   end
end
mixrate = iacc/n

burnin = 0.1.*n;
theta = theta(burnin+1:n);
mean_theta = mean(theta)
```

```
var_theta = var(theta)

lin = linspace(burnin+1,n,n-burnin);
figure(2)
plot(lin, theta);

figure(3)
[start, stop] = hist(theta,20);
start = start./(stop(2)-stop(1))./length(theta);
bar(stop,start,1,'r');

cilow=mean_theta -norminv(.975).*sqrt(var_theta);
cihigh=mean_theta +norminv(.975).*sqrt(var_theta);
```

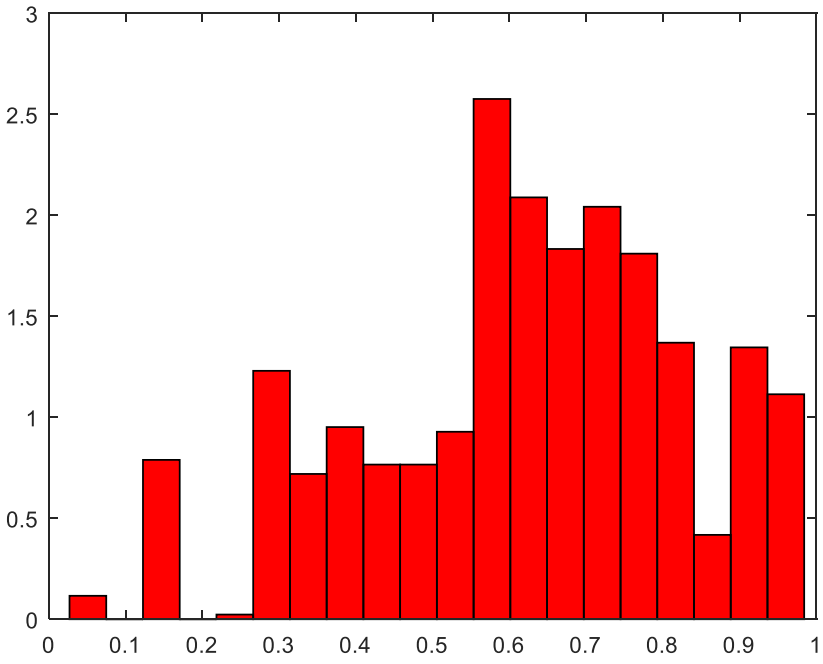
EDA- provides estimate mean for the model (2 and 5)



Markov chain



Accepted data



```
mixrate =

    0.125000000000000

mean_theta =

   0.622323226807151


var_theta =

   0.043946724114360

cilow =

   0.211446686968575

cihigh =

   1.033199766645728
```

We are 95% confident that the true value of δ is between .21145 and 1.03320,

This mixrate is higher than that in part b (Beta(2,3)) density as prior).

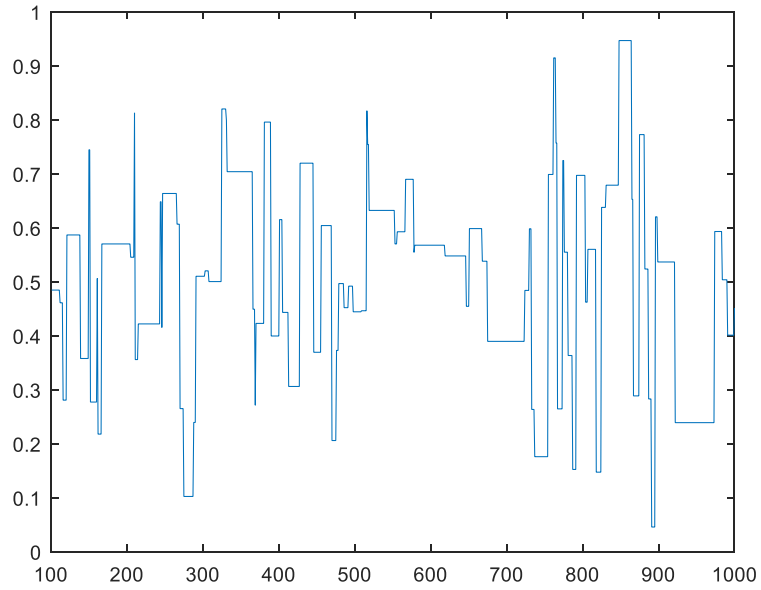
We now repeat this process using a Beta density as the prior distribution. We choose parameters of the proposal density carefully so that the chain moves quickly away from the starting value and converges to it stationary distribution.

```
muint = [2 5];

varint = [1./3 2./3];

piesint = [1./2 1./2];

maxiter = 100;

tol = 0.001;


[pies mus vars] = csfinmix(data, muint, varint, piesint, maxiter, tol);


n = 4000;

x = zeros(n,1);

r = rand(1,n);
```

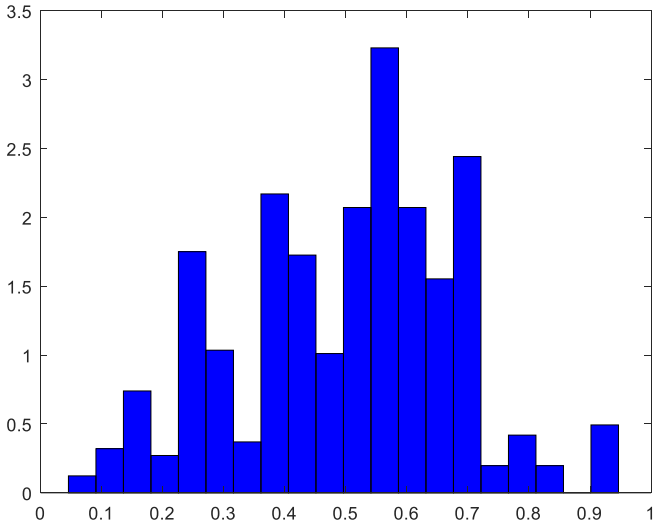
```
ind = length(find(r <= pies(1)));

x(1:ind)= randn(ind,1)*sqrt(vars(1)) + mus(1);
x(ind+1:n)= randn(n-ind,1)*sqrt(vars(2)) + mus(2);

figure(1)
hist(data)

n = 1000;
theta = zeros(1,n);
theta(1) = rand(1);
iacc = 0;

strg = 'pies(1)*normpdf(x, mus(1),sqrt(sig(1)))+pies(2)*normpdf(x,mus(2),sqrt(sig(2)))'
dist = inline(strg, 'x');

for i = 2:n
   y = betarnd(2,3);
   u = rand;

   alpha = min([1 , pies(1)*normpdf(y, mus(1),sqrt(vars(1)))+pies(2)*normpdf(y,mus(2),sqrt(vars(2)))...
      /pies(1)*normpdf(theta(i-1), mus(1),sqrt(vars(1)))+pies(2)*normpdf(theta(i-1),mus(2),sqrt(vars(2)))]);
   if u < alpha
      theta(i) = y;
      iacc = iacc + 1;
   else
      theta(i) = theta(i-1);
   end
end
mixrate = iacc/n

burnin = 0.1.*n;
theta = theta(burnin+1:n);
```

```
mean_theta = mean(theta)
var_theta = var(theta)

lin = linspace(burnin+1,n,n-burnin);
figure(2)
plot(lin, theta);

figure(3)
[start, stop] = hist(theta,20);
start = start./(stop(2)-stop(1))./length(theta);
bar(stop,start,1,'b');

cilow=mean_theta -norminv(.975).*sqrt(var_theta);
cihigh=mean_theta +norminv(.975).*sqrt(var_theta);
```

Markov chain



Accepted data




```
mixrate =

    0.1000

mean_theta =

    0.4990


var_theta =

    0.0321

cilow =

    0.1476

>> cihigh

cihigh =

    0.8504
```

We are 95% confident that the true value of δ is between .1476 and .8504,

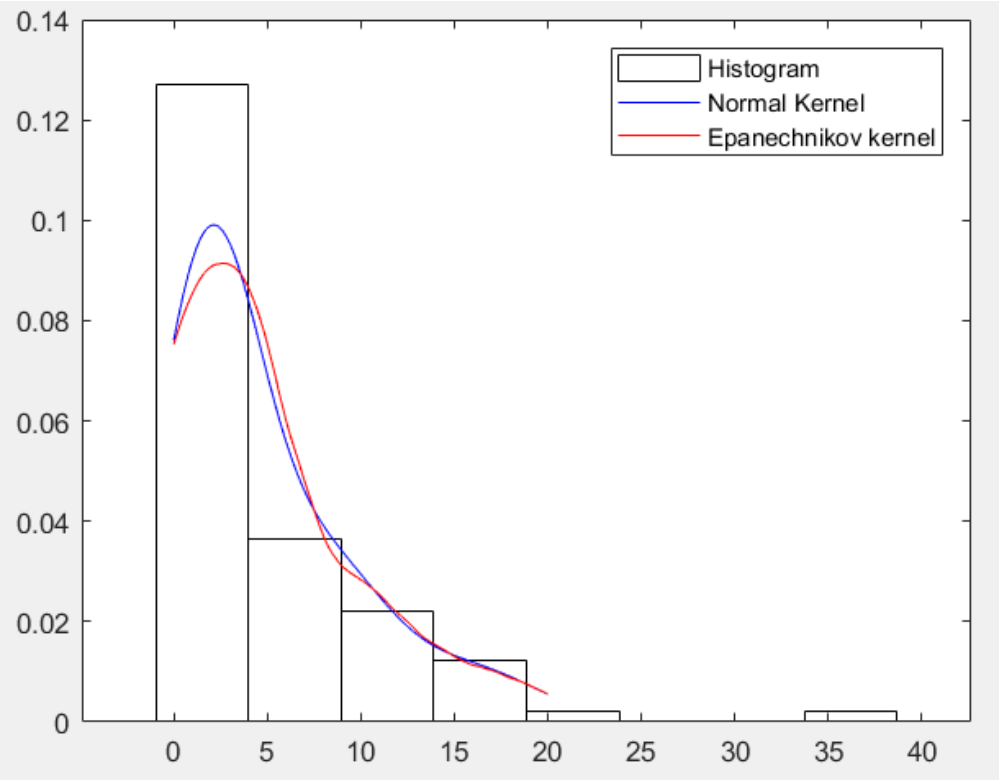
The mixrate is lower than that in part a (Uniform(0,1) density as prior. The mean_theta is lower, and the var_theta is higher, but about the same. The confidence interval is about the same, with part b being a little lower (as expected due to differences in mean_theta)

Some More Density Estimation in MATLAB

FIRST EXAMPLE OF EXPERIENCE: We generate random sample of size 100 from Exponential (5) to get Density Estimation using Normal kernel and Density estimation using Epenchinikev kernel.

```
data=exprnd(5,100,1)
n = length(data);

h = 2.15*sqrt(var(data))*n^(-1/5);
bins = 0:h:20;
vk = histc(data,bins);
vk(end) = [];
fhat = vk/(n*h);

h = 2.15*sqrt(var(data))*n^(-1/5);
t0 = min(data)-1;
tm = max(data)+1;
bins = t0:h:tm;
vk = histc(data,bins);
vk(end) = [];
fhat = vk/(n*h);
bc2=(t0-h/2):h:(tm+h/2);
binh = [0 fhat' 0];
xinterp = linspace(min(bc2),max(bc2));
fp = interp1(bc2, binh, xinterp);
tm = max(bins);
bc = (t0+h/2):h:(tm-h/2);
bar(bc,fhat,1,'w')
hold on
area = trapz(xinterp,fp);

x = linspace(0,20,n); len=length(x);
fhatN = zeros(size(x));
h1 = 1.06*n^(-1/5)*std(data);
for i=1:n
   % Normal kernel
   f=exp(-(1/(2*h1^2))*(x-data(i)).^2)/sqrt(2*pi)/h1;
   fhatN = fhatN+f/(n);
end

fhatE = zeros(size(x));
h2=h1*(30*sqrt(pi))^0.2;

for i=1:n
 % Epanechnikov kernal
   dom=(x-data(i))/h2;
   for j=1:len
       if abs(dom(j))<=1
       f(j)=3*(1-((x(j)-data(i))/h2).^2)/(4*h2);
   else
   f(j)=0;
       end
   end
   fhatE = fhatE+f/(n);
```

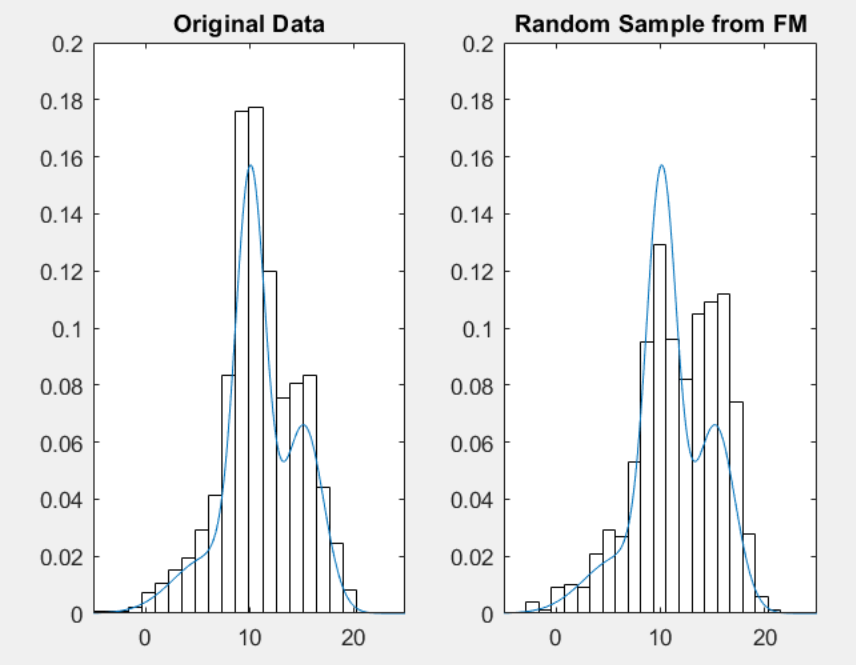
```
end

plot(x,fhatN,'b',x,fhatE,'r');
legend('Histogram','Normal Kernel','Epanechnikov kernel')
hold off
```



We develop and implement algorithm for generation of random variables based on a kernel density estimate, using method for Finite Mixture

```
n=2000;
nx = [200,800,500];
mu=[5 10 15];
sigma=[3 1.5 2];
data = zeros(n,1);
data(1:nx(1)) = normrnd(mu(1),sigma(1),nx(1),1);
data(nx(1)+1:nx(1)+nx(2)) = normrnd(mu(2),sigma(2),nx(2),1);
data(nx(1)+nx(2)+1:n) = normrnd(mu(3),sigma(3),nx(3),1);
muin = [2 8 20];
piesin = [.333 .333 .333];
varin = [1 1 1];
max_it = 100;
tol = 0.00001;
[pies,mus,vars]=...
    csfinmix(data,muin,varin,piesin,max_it,tol);

xx=-5:.1:25;
```

```
fm = zeros(size(xx));
for i=1:3
   fm = fm+pies(i)*normpdf(xx,mus(i),sqrt(vars(i)));
end
N=2000;
x = zeros(N,1);.
r = rand(N,1);
ind1 = length(find(r <= pies(1)));
ind2 = length(find(r <= pies(2)));
x(1:ind1) = normrnd(mus(1),sqrt(vars(1)),ind1,1);
x(ind1+1:ind2) = normrnd(mus(2),sqrt(vars(2)),ind2-ind1,1);
x(ind2+1:N) = normrnd(mus(3),sqrt(vars(3)),N-ind2,1);

figure(1)
subplot(121)
[cnt,data]=hist(data,20);
bar(data,cnt/n,1,'w')
title('Original Data')
axis([-5 25 0 .2])
hold on
plot(xx,fm)
hold off

subplot(122)
[cnt,x]=hist(x,20);
bar(x,cnt/N,1,'w')
title('Random Sample from Finite Mixture')
axis([-5 25 0 .2])
hold on
plot(xx,fm)
hold off
```

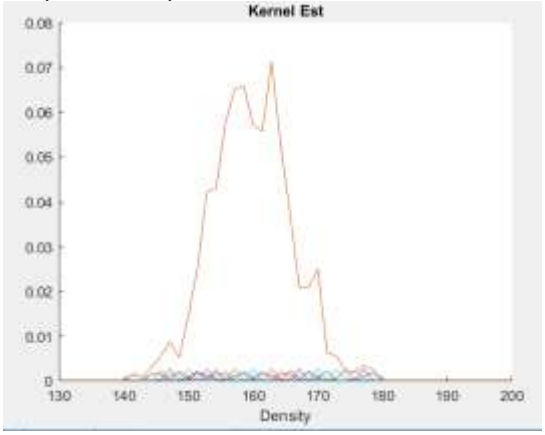


The Finite mixture estimate based on the normal kernel is very similar in distribution shape.

THIRD EXAMPLE: Kernel Density Estimation for height measurements of 351 elderly females [Hand, et al., 1994]

```
load elderly
data = heights;
n = length(data);

h = 2.15*sqrt(var(data))*n^(-1/5);
t0 = min(data) -1; tm = max(data) + 1;
bins = t0:h:tm;

repres = histc(data, bins);
repres(end) = [];
fhat = repres/(n*h);

bc2 = (t0-h/2):h:(tm+h/2);
binh = [0 fhat 0];
xinterp = linspace(min(bc2), max(bc2), 20);
fp = interp1(bc2, binh, xinterp);

x = linspace(130,200,50);
fhatk = zeros( size(x) );
hk = 2.12*n^(-1/5);
figure
hold on
for i = 1:n
f = exp( -(1/(2*hk^2))*(x- data(i)).^2)/sqrt(2*pi)/hk;
plot(x, f/(n*hk));
fhatk = fhatk+f/(n);
end
plot(x, fhatk);
hold off
xlabel('Density')
```

```
title('Kernel Est')
```



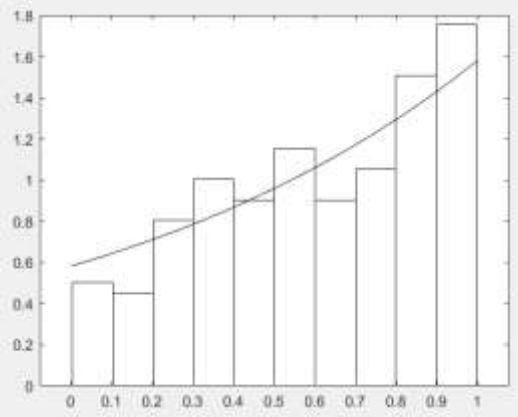
There is evidence of bumps and nodes.

FINAL EXAMPLE:

We use the inverse CDF method to generate a random sample of size 200 with density function: e^x/e-1 with x going from 0 to 1 (continuous)

```
n=200
Y=rand(n,1);
rv=zeros(n,1);
for i = 1:n
    rv(i) = log(Y(i)*exp(1)-Y(i)+1);
end

x= [0:.01:1]
z= (exp(x))./(exp(1)-1);

[N,h]=hist(rv);
N=N/ (h(2)-h(1))/n;
bar(h,N,1,'w'), hold on
```

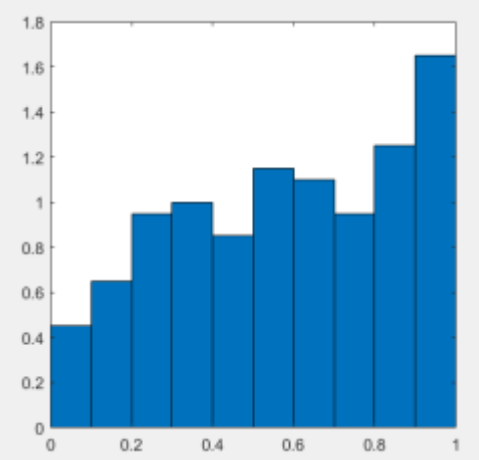
```
plot(x,z,'k'), hold off
```

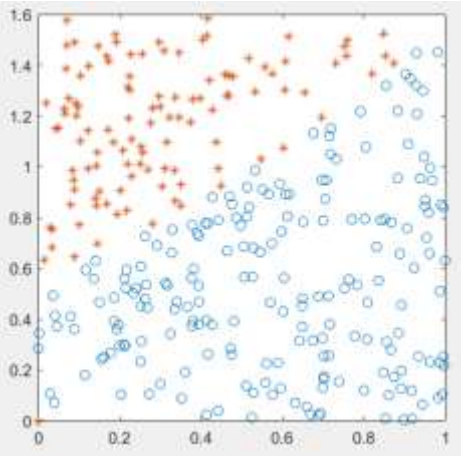


Part 2) ACCEPTANCE/REJECTION METHOD with Uniform(0,1) candidate distribution

```
n = 200;
c = exp(1)/(exp(1)-1)
x = zeros(1,n);
xy = zeros(1,n);
irv = 1;
irej = 1;
rej = zeros(1,n);
rejy = zeros(1,n);

while irv <= n
    u = unifrnd(0,1);
    y = unifrnd(0,1);
    if u <= exp(y)/(exp(1)-1)/(c)
        x(irv) = y;
        xy(irv) = c*u;
        irv = irv + 1;
    else
        rej(irej) = y;
```

```
        rejy(irej) = c*u;
        irej = irej + 1;
    end
end

x= [0:.01:1]
z= (exp(x))./(exp(1)-1);

plot(x,xy,'o',rej,rejy,'*')
axis square
hold on
hold off

[fr,x] =hist(x);
h=x(2)-x(1)
bar(x,fr/(n*h),1)
axis square
hold on
hold off
```





The histogram of random sample also appears to fit true density, as with Inverse CDF method, so acceptance/rejection plot also appears to fit well.

Now we estimate the median using both methods (inverse cdf and accept/reject) and compare it to the true value

CODE: x= [0:.01:1]

z= (exp(x))./(exp(1)-1);

median(z)

OUTPUT: 0.9595

The median of the random sample is .9595 and the true value is 1.60943791 (integral from 0 to t of density = .5 and solve for t). We conclude that the median is close to the true value, as both are roughly between one and two.

Running a Wordcount on Hadoop Using R Script

This dataset contains over 80,000 reports of UFO sightings over the last century. There are two versions of this dataset: scrubbed and complete. The complete data includes entries where the location of the sighting was not found or blank (0.8146%) or have an erroneous or blank time (8.0237%). Since the reports date back to the 20th century, some older data might be obscured. Data contains city, state, time, description, and duration of each sighting. This dataset was scraped, geolocated, and time standardized from NUFORC data by Sigmond Axel. Variables: City in which UFO was sighted, State in which UFO was sighted, Country of Sighting, Shape of the UFO, Duration of the Sighting in seconds, Duration of the sighting in hours and min, Sighting description, Posted date of the sighting, Latitude coordinate of the sighting, and Longitude coordinate of the sighting.

```
# HadoopR Example
# Running a WordCount on Hadoop Using R Script

#################################################################################
#! /usr/bin/env Rscript

# mapper.R - Wordcount program in R
# script for Mapper (R-Hadoop integration)

trimWhiteSpace <- function(line) gsub("(^ +)|( +$)", "", line)
splitIntoWords <- function(line) unlist(strsplit(line, "[[:space:]]+"))

## **** could wo with a single readLines or in blocks
con <- file("stdin", open = "r")
while (length(line <- readLines(con, n = 1, warn = FALSE)) > 0) {
  line <- trimWhiteSpace(line)
  words <- splitIntoWords(line)
  ## **** can be done as cat(paste(words, "\t1\n", sep=""), sep="")
  for (w in words)
    cat(w, "\t1\n", sep="")
}
close(con)

#################################################################################
#! /usr/bin/env Rscript

# reducer.R - Wordcount program in R
# script for Reducer (R-Hadoop integration)

trimWhiteSpace <- function(line) gsub("(^ +)|( +$)", "", line)

splitLine <- function(line) {
  val <- unlist(strsplit(line, "\t"))
  list(word = val[1], count = as.integer(val[2]))
}

env <- new.env(hash = TRUE)

con <- file("stdin", open = "r")
while (length(line <- readLines(con, n = 1, warn = FALSE)) > 0) {
  line <- trimWhiteSpace(line)
  split <- splitLine(line)
  word <- split$word
  count <- split$count
  if (exists(word, envir = env, inherits = FALSE)) {
    oldcount <- get(word, envir = env)
    assign(word, oldcount + count, envir = env)
  }
  else assign(word, count, envir = env)
}
close(con)

for (w in ls(env, all = TRUE))
  cat(w, "\t", get(w, envir = env), "\n", sep = "")
```

BIBLIOGRAPHY

Data source: https://www.kaggle.com/NUFORC/ufo-sightings