Machine learning to estimate the effects of road proximity on preterm birth

by

Brian O'Connell

BS, Rochester Institute of Technology, 2014

Submitted to the Graduate Faculty of the

Department of Biostatistics

School of Public Health in partial fulfillment

of the requirements for the degree of

Master of Science

University of Pittsburgh

2022

UNIVERSITY OF PITTSBURGH

SCHOOL OF PUBLIC HEALTH

This thesis was presented

by

Brian O'Connell

It was defended on

April 25, 2022

and approved by

Thesis Advisor: Ada Youk, PhD, Associate Professor, Biostatistics
School of Public Health, University of Pittsburgh

Jeanine M. Buchanich, PhD, Research Associate Professor, Biostatistics
School of Public Health, University of Pittsburgh

Jenna C. Carlson, PhD, Assistant Professor, Biostatistics
School of Public Health, University of Pittsburgh

James Fabisiak, PhD, Associate Professor, Biostatistics
School of Public Health, University of Pittsburgh

Machine Learning Analysis of Preterm Birth in Southwestern Pennsylvania

Brian O'Connell, MS

University of Pittsburgh, 2022

Background: Preterm birth is a critical public health issue because babies born before 37 weeks of gestation have much higher risks of having chronic health issues. Infants born preterm also experience more immediate health complications due to the lack of development of critical organs such as the lungs.

Methods: Logistic regression and the machine learning methods of neural networks, random forests, and extreme gradient boosted trees were implemented to create models that predict whether a baby will be born preterm. These models were trained using the mothers' age, ethnicity, socioeconomic status, education level, smoking status, pre-pregnancy weight, date of giving birth, county of birth, and number of prenatal visits. Adjustments for father's race and ethnicity were accommodated in the model. These demographic variables of the parents were used in conjunction with an environmental measure of proximity to the nearest major road or train tracks as an air quality metric to determine the most important variables for predicting preterm birth.

Results: The random forest model performed the best with an area under the curve (AUC) of 0.731 for the receiver operating characteristic (ROC) plot and had the distance to the nearest road or train tracks as the most important variable to the model. All models performed similarly well with the lowest being logistic regression with an AUC of 0.657. All models besides the random forest method identified the number of prenatal visits as the most important variable.

Conclusion: Using machine learning methods, the number of prenatal visits and the distance to the nearest major road or train tracks were the most important variables in predicating

preterm birth. Monitoring these variables can help public health officials and medical professionals give the best recommendations for expecting mothers.

Public Health Impact: This thesis will help discover nonlinear associations between demographic or environmental and preterm birth. Identifying these variables can help guide public health officials know where to focus their efforts in putting forth legislation and implementing policies. Additionally, providing this information to doctors can also help them better inform and guide their patients as they navigate their pregnancy.

Table of Contents

List of Tables

List of Figures

1.0 Introduction

1.1 Background

Preterm birth is a critical public health issue to explore as approximately 10% of all births worldwide are preterm [4]. Preterm births are defined as babies born before 37 weeks of gestation and are the leading cause of neonatal mortality [8]. Babies that are born preterm have much higher risks of having chronic health issues as well as more immediately having complications due to lack of development of critical organs such as the lungs [5]. In addition to the multitude of adverse health risks of being born preterm, there is also substantial economic strain associated with these births as preterm births on average cost almost 10 times as much in medical care [8]. Further understanding what most influences and causes preterm birth can lead to saving many lives as well as improving the quality of life of infants.

1.2 Previous Research

Previous research by Miranda et al. (2013) using proximity to roads as an air quality measure and explored the relationship between this metric and a preterm birth outcome. Multiple linear regression models were implemented for predicting preterm birth. Exposure to air pollution during pregnancy has been shown by Ritz et al. (2000) to be associated with preterm birth [7]. In this thesis, proximity to the nearest major roadway or train tracks is deployed as an air quality indicator for the mother's residence. The World Health Organization found in 2013 that living in

close proximity to roadways was associated with increased levels of carbon monoxide, nitrogen dioxide, black carbon, and polycyclic aromatic hydrocarbons [6]. Proximity to the nearest major roadway or train tracks operates as a measure of contact to numerous potentially harmful environmental exposures.

This study seeks to extend the study Miranda et al. developed and employ a regression model as well as the machine learning techniques of neural networks, random forest, and extreme gradient boosted trees to observe the patterns between the demographic variables, proximity to roads, and preterm birth. Due to the sparse nature of preterm births and the assumption of equality between the event and nonevent in regression models as well as the machine learning methods, undersampling was used to circumvent this violation of class imbalance.

1.3 Public health impact

Identifying which demographic and environmental factors that can influence whether a baby will be born preterm can help public health officials focus on the most important policies to prevent preterm birth. Understanding the relationships of these variables can also aid physicians in the treatment of expecting mothers and their recommendations of behavior while pregnant.

2.0 Methods

2.1 Data

2.1.1 Data Source

These data came from the Health Effects of Hydraulic Fracturing Studies conducted at the University of Pittsburgh. The study area includes mothers who gave birth and had permanent residency from eight counties in western Pennsylvania including Allegheny, Armstrong, Beaver, Butler, Fayette, Greene, Washington, and Westmoreland counties. The observations for the study came from 2009-2020 and 256,681 mothers are recorded to have given birth in this time.

2.1.2 Covariate Selection

The variables used as input for the models were the mothers' age, ethnicity, socioeconomic status, education level, smoking status, pre-pregnancy weight, date of giving birth, county of birth, and number of prenatal visits. The fathers' race and ethnicity were also accounted for in the model. These demographic variables of the parents were used in conjunction with an environmental measure of proximity to the nearest major road or train tracks as an air quality metric.

2.1.3 Data Cleaning

Observations that had missing values were dropped from the dataset for this analysis. Some variables were also encoded with flags for missingness, observations with these flags were also

removed from the study. After missing values were removed the resulting dataset had 140,244 observations.

Preterm birth was calculated from the listed gestational age in weeks and defined as any birth delivered as less than 37 weeks gestational age labeled “preterm” and births of 37 weeks or above as “term”. Births with a recorded gestational age of 21 weeks or less were removed from the dataset as 22 weeks of gestational age is the lower limit of viability for life [1]. Mothers’ education was separated into a categorical variable in accordance with the fracking study; less than high school, GED, and at least come college. Smoking was quantified in this dataset by the number of cigarettes smoked during each trimester, these were then combined into a variable describing the total number of cigarettes over the course of the whole pregnancy to control for smoking. Hispanic origin in this dataset was categorized by the area of south America an individual’s ancestors hailed from, but for the purposes of model building this variable was dichotomized into has “Hispanic” and “Not Hispanic”. All categorical variables were transformed into dummy variables where the variable was broken down into n-1 new variables where n is the number of levels in the original categorical variable. These new variables were all assigned a 1 for the level of the variable that the observation did have and 0 for all of the other variables for that original variable.

The ArcGIS and the simple features (sf) packages in R were used to calculate the distance to the nearest primary road, secondary road, or rail feature using each mother’s residence latitude and longitude as well as shapefiles for roads obtained from the U.S. Census Bureau. Definitions for the rail features and two road classifications were also obtained from the U.S. Census Bureau. Shapefiles are used by ArcGIS to map the roads based their location and shape which was then integrated to calculate the nearest major road or train tracks to each mothers’ residence.

Rail features in Pennsylvania are defined as a fixed rail line, generally visible from the surface, which carries any type of rail vehicle including railroad, off-street transit and mountain rail systems.

Primary roads are limited-access highways that connect to other roads only at interchanges and not at at-grade intersections. This category includes Interstate highways, as well as all other highways with limited access (some of which are toll roads). Limited-access highways with only one lane in each direction, as well as those that are undivided.

Secondary roads are main arteries that are not limited access, usually in the U.S. highway, state highway, or county highway systems. These roads have one or more lanes of traffic in each direction, may or may not be divided, and usually have at-grade intersections with many other roads and driveways. They often have both a local name and a route number.

2.2 Statistical analysis

2.2.1 Test Set Creation

Once the variables were selected and the resulting dataset was cleaned, the data were split in a 70/30 ratio. The 30% of the dataset was set aside as the test set to test model performance on data not used in training. The 70% portion of the data was used to do model construction. Using 30% of the data for test helps ensure that the predictions are not biased to the training set as the techniques to overcome class imbalance can cause this. Due to the sample size being so big 70% of the data for training is still sufficient to capture trends in the variables. In the training set there

were 7,099 preterm and 91,073 term birth observations. The model building was then used on the 70% partition which further splits the data into training and testing sets. The training set applies the model building technique to identify trends and important variables for predicting preterm birth by increasing or decreasing the coefficients of the variables in the model. This model then applies predictions to the testing set and accuracy of predictions can be measured.

2.2.2 Technique Selection to Overcome a Sparse Outcome

The outcome of interest, preterm birth, occurred in 7,099 (7.2%) out of the 98,172 observations in the training set. A key assumption for logistic regression and the three machine learning methods when doing classification problems is that the event and non-event are observed equally within the dataset (will add reference and rephrase). In classification problems with an imbalanced outcome the model will only predict the majority class still report a high accuracy despite failing to correctly identify any of the minority class.

Two strategies were attempted in this study to overcome the class imbalance present with preterm birth, Synthetic Minority Over-Sampling Technique (SMOTE) and random undersampling [9]. SMOTE was implemented using the smotefamily package in R and created synthetic observations where preterm was observed until equality of preterm and term were achieved in the dataset. This is a computationally expensive task given the sample size in the dataset being so large and was discarded in favor of an undersampling technique.

Random undersampling of the majority class was the second class imbalance technique applied to the dataset. The term births were randomly sampled using the Random Over-Sampling Examples (ROSE) package in R such that there were an equal amount of term and preterm births,

in this case 7,099 observations for both term and preterm births, giving a total sample size of 14,198 to construct the models.

2.2.3 Classification Model Building

For all models the caret package in R was used for construction. Five-fold cross validation was implemented, and accuracy was specified as the metric to maximize. Centering and scaling were also applied to the logistic regression and neural network models to provide better interpretability of the results [3]. All models were constructed using the 70% undersampled split using the "train" function in caret with the appropriate method for the model specified. These models were evaluated for their accuracy on the training set using confusion matrices. A confusion matrix is a table that aids in summarizing classification of the outcome by comparing the model prediction for preterm birth and comparing that to whether preterm birth or term birth was observed in the data. Each model was then applied to the 30% test set using the "predict" function to predict whether the observations would have preterm birth or not. This prediction was checked against what was observed in the test set and the metrics of sensitivity, specificity, accuracy, and area under the curve (AUC) of the receiver operating characteristic curve (ROC) were used to compare model performance. Variable importance was also derived from these models to identify the driving variables in classifying preterm birth. The sensitivity calculation can be found in Equation 1 and is the proportion of true positives out of all preterm births, in this case the number of preterm births correctly predicted by the model over all preterm births in the test set. Specificity can be found in Equation 2 and represents how well each model can accurately predict term births. Accuracy can be found in Equation 3 and is a measure of how well each model can both identify preterm births and term births in the test set.

$$Sensitivity = \frac{TP}{TP + FN} \quad \textbf{Equation 1}$$

$$Specificity = \frac{TN}{TN + FP} \quad \textbf{Equation 2}$$

$$Accuracy = \frac{TN + TP}{TN + FN + TP + FP} \quad \textbf{Equation 3}$$

In Equations 1-3 above, TP refers to total count of true positive cases where the model predicts preterm birth and preterm birth is observed in those cases. TN is the total count of true negative cases where the model predicts term birth and term birth is observed in those observations. FP refers to total count of false positive cases where the model predicts preterm birth but a term birth is observed for those cases. FN refers to total count of false negative cases where the model predicts term birth but a preterm birth is observed for those cases.

2.2.3.1 Logistic Regression

When examining a classification problem, it is intuitive to use a logistic regression model as a baseline comparison to machine learning methods. Caret uses the "glm" function to fit the logistic regression model and this can be represented by Equation 4 below

$$ln\left(\frac{\mu}{1-\mu}\right) = \boldsymbol{X}\beta \quad \textbf{Equation 4}$$

Where μ is the probability of the baby being born preterm, X is the covariate matrix, and β is the vector of parameter coefficients. This model was implemented on the undersampling training set and then using the "predict" function tested against the 30% test set to predict whether the observations would have preterm birth or not.

2.2.4 Machine learning models

Machine learning models apply different techniques to achieve the same goal of minimizing the loss function, in this case mean square error is the loss function that was minimized. All of the implemented machine learning methods use tuning parameters which are variables that aid in how the models are constructed. Regularization parameters are a type of tuning parameter that penalizes models based on the number of variables as a way to mitigate overfitting. Overfitting occurs when the model becomes too attuned to the training set that was used to create it and performs poorly on any testing or validation partitions. Shrinkage parameters are another type of tuning parameter implemented in some machine learning models that direct the model how quickly coefficients of the covariates are reduced towards zero. Combinations of tuning parameters are iterated through by caret and the best permutation of these parameters is identified for each model.

2.2.4.1 Neural Network

The neural network serves as a statistical model that performs a series of functional transformations using the observed data, a hidden unit layer consisting of one or more perceptron's that operate as the "neurons", and the output layer that makes the prediction of preterm or term. An example of a single layer neural network can be found in Figure 1.

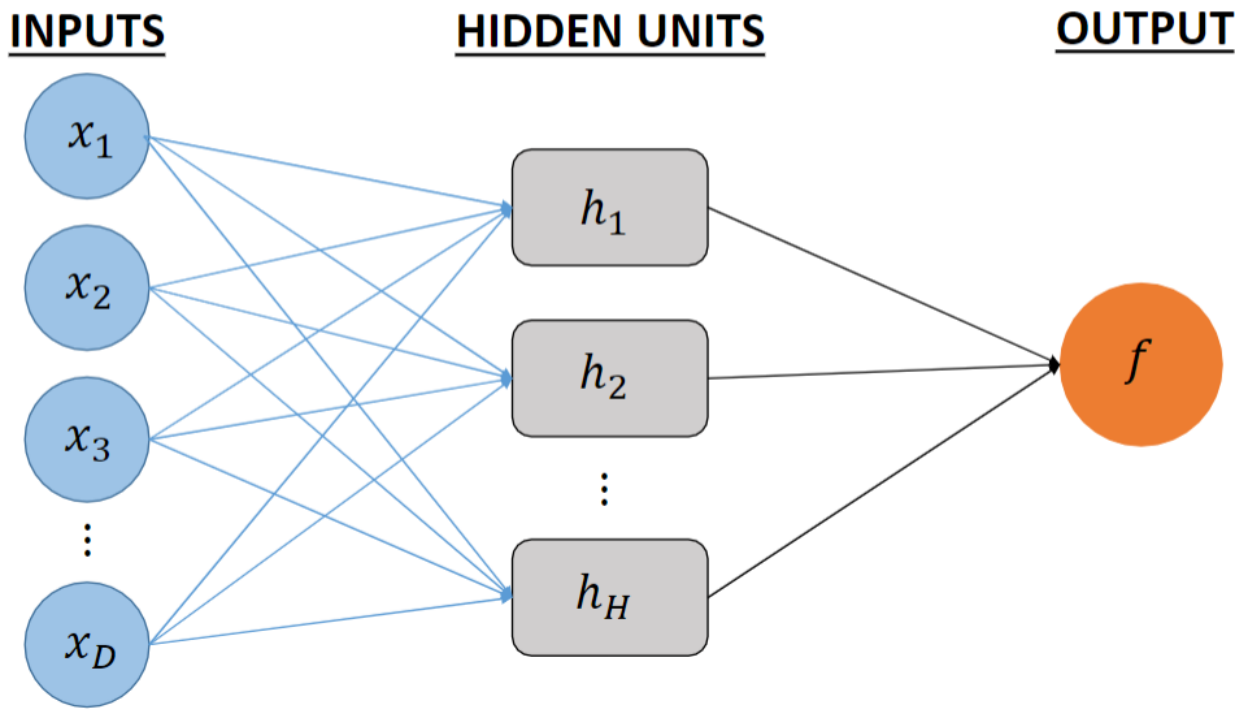

Figure 1 Shallow neural network

Caret was used to implement the neural network for this dataset and a feedforward method was adopted which indicates the model only moves in one direction, from left to right as visualized in Figure 1. The input layer contains all of the covariates in the model, each of these are fed into the hidden unit layer. The input variables are put into a linear combination and each neuron in the hidden unit layer applies a non-linear transformation. The hidden unit layer then passes to the output layer which has similar functional form to a linear model. The output layer can then be used to predict preterm birth using the parameters it has weighted and retained in the model. Caret by default tunes the neural networks with either 1, 3, or 5 neurons in the hidden unit layer.

2.2.4.2 Tree based models

Tree based models are widely used in statistical learning and machine learning applications due to their flexibility and intuitive structure [9]. Both random forest and extreme gradient boosted

trees were implemented in this study and are from the tree-based model family. Both models utilize the creation of many decision trees which are controlled by different tuning parameters explained in the sections below. Each individual decision tree specifies a cutoff point for one of the variables in the model to guide prediction. For a continuous variable, the split will be some integer value where if it is less than that number go one way and if not go the other. This can either be to another decision point or finally to a prediction. In these two models many trees are fit in this manner and then the average prediction is calculated across all of the trees.

2.2.4.3 Random Forest

Random forest is a tree-based model that randomly selects covariates to use for each specific tree. Caret randomly selects variables for each iteration of the tree and is controlled by the tuning parameter “mtry”. Bootstrapping was performed five times, as indicated in the cross-validation setting specified for Caret, and then 500 trees were created using the randomly selected variables. Each of the 500 trees creates a decision tree using the randomly selected variables from mtry. The prediction from each tree on the test set is averaged for each cross-validation sample and then output. Figure 2 visualizes how an example random forest model is created [3].

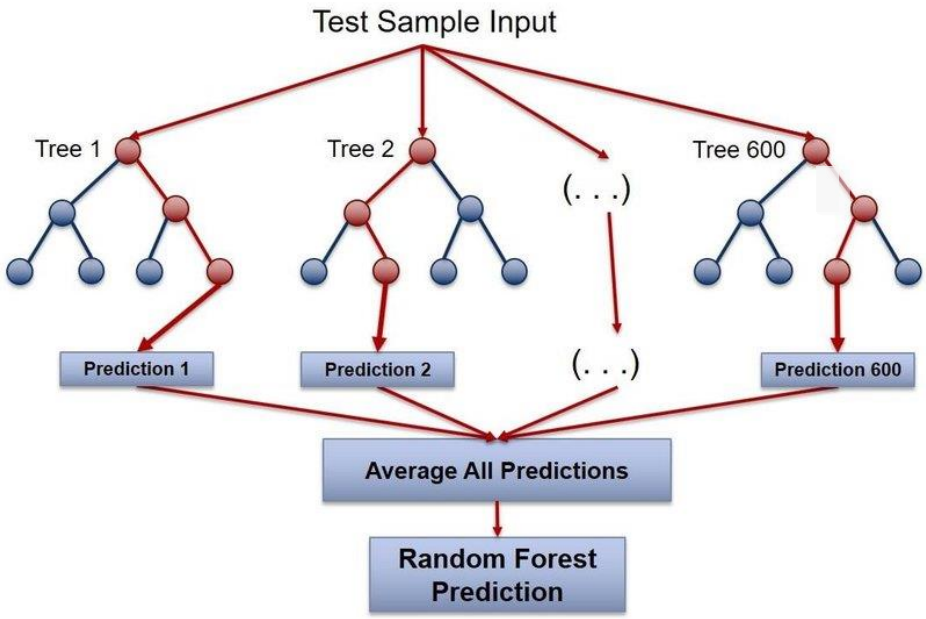

Figure 2 Random forest visualizaion

2.2.4.4 Extreme Gradient Boosted Trees

Extreme gradient boosted trees (XGBoost) are another tree-based method for modeling classification problems. It performs similarly to random forest in that there is a tuning parameter similar to “mtry” that randomly selects variables for each tree that is grown. XGBoost also randomly subsamples the data using bootstrapping for each tree to randomly create an internal training and testing split. The major difference between random forest and XGBoost is the addition of regularization parameters such as the max depth of each tree and a learning or shrinkage rate which controls how quickly parameters coefficients are reduced towards zero based on the model.

3.0 Results

3.1 Summary Statistics

After filtering out missing values from the variables selected for this study 140,244 observations were used in the analysis. The summary of demographic variables for these parents and their newborns can be found in Table 1 below. Of the 140,244 number of infants in the study 10,141 of them were born preterm. The average age for mother in this study was 29.42 and the average distance the mother's residence from a major road was 674 meters. Of the mothers in this study, 25% were enrolled in the WIC supplemental food program. As stated previously, all of the categorical variables were transformed into dummy variables and thus have a mean and standard deviation associated with them in table of based on how many observations had 0's and how many had 1's.

Table 1 Summary Statistics

	Overall	
Total Sample size	140,244	
Month of first prenatal visit (N, %)		
January	13049	9.30%
February	11772	8.40%
March	11707	8.30%
April	11103	7.90%
May	10864	7.70%
June	11218	8.00%
July	10993	7.80%
August	12019	8.60%
September	11967	8.50%
October	12604	9.00%
November	11131	7.90%

December	11817	8.40%
Year (N, %)		
2009	7790	5.60%
2010	12856	9.20%
2011	11328	8.10%
2012	11782	8.40%
2013	14016	10%
2014	13352	9.50%
2015	13037	9.30%
2016	13083	9.30%
2017	13374	9.50%
2018	12313	8.80%
2019	11365	8.10%
2020	5948	4.20%
Number of Prenatal visits (mean, SD)	11.97	4.08
Mother Pre-pregnancy weight (mean, SD)	156.73	40.58
Number of cigarettes smoked during whole pregnancy (mean, SD)	3.62	11.37
Distance to nearest major road or train tracks (mean, SD)	674.33	716.41
Percentage of babies born to term (%)	92.80%	
Sex of baby male (mean, SD)	0.51	0.5
Mother receives WIC Food support (mean, SD)	0.25	0.44
Mother Education		
Highschool or GED (mean, SD)	0.19	0.39
At least some college or more (mean, SD)	0.76	0.43
County of Mother's Residence (mean, SD)		
Cambria	0.00	0.06
Clarion	0.00	0.02
Clearfield	0.00	0.02
Allegheny	0.67	0.47
Fayette	0.05	0.21
Armstrong	0.03	0.16
Indiana	0.00	0.06
Lawrence	0.00	0.04
Lehigh	0.00	0.01
Beaver	0.05	0.21
Mercer	0.00	0.05
Montgomery	0.00	0.01
Philadelphia	0.00	0.01
Somerset	0.00	0.02
Venango	0.00	0.03
Washington	0.06	0.24
Westmoreland	0.09	0.28

Mother's Race (mean, SD)		
Other Asian	0.00	0.06
Other Pacific Islander	0.00	0.02
Other	0.01	0.1
Black or African American	0.07	0.26
American Indian or Alaska Native	0.00	0.04
Asian Indian	0.01	0.1
Chinese	0.00	0.07
Filipino	0.00	0.05
Japanese	0.00	0.02
Korean	0.00	0.04
Vietnamese	0.00	0.04
Father's Race (mean, SD)		
Other Asian	0.00	0.06
Other Pacific Islander	0.00	0.02
Other	0.02	0.12
Black or African American	0.10	0.3
American Indian or Alaska Native	0.00	0.04
Asian Indian	0.01	0.1
Chinese	0.00	0.06
Filipino	0.00	0.03
Japanese	0.00	0.02
Korean	0.00	0.04
Vietnamese	0.00	0.04
Parents Hispanic Ethnicity (mean, SD)		
Mother of Hispanic origin	0.02	0.12
Father of Hispanic origin	0.02	0.13

3.2 Model construction

Figure 3 shows the average accuracy and 95% confidence interval of each model on the training set. The highest accuracy to predict preterm correctly on the training set was the random forest model with an accuracy of 0.6676. For the Neural network the best tuning parameter was a neural network with 1 hidden unit and a weight decay of 0.1 as seen in Figure 4. For the random forest model, the best number of starting variables was 37 randomly selected variables which yielded an accuracy of 0.6676 which can be seen in Figure 5. The best tuning parameters for the extreme gradient boosted tree method is shown in Figure 6 and was 50 trees, a max depth of each tree of 3 splits, and a learning or shrinkage rate of 0.3.

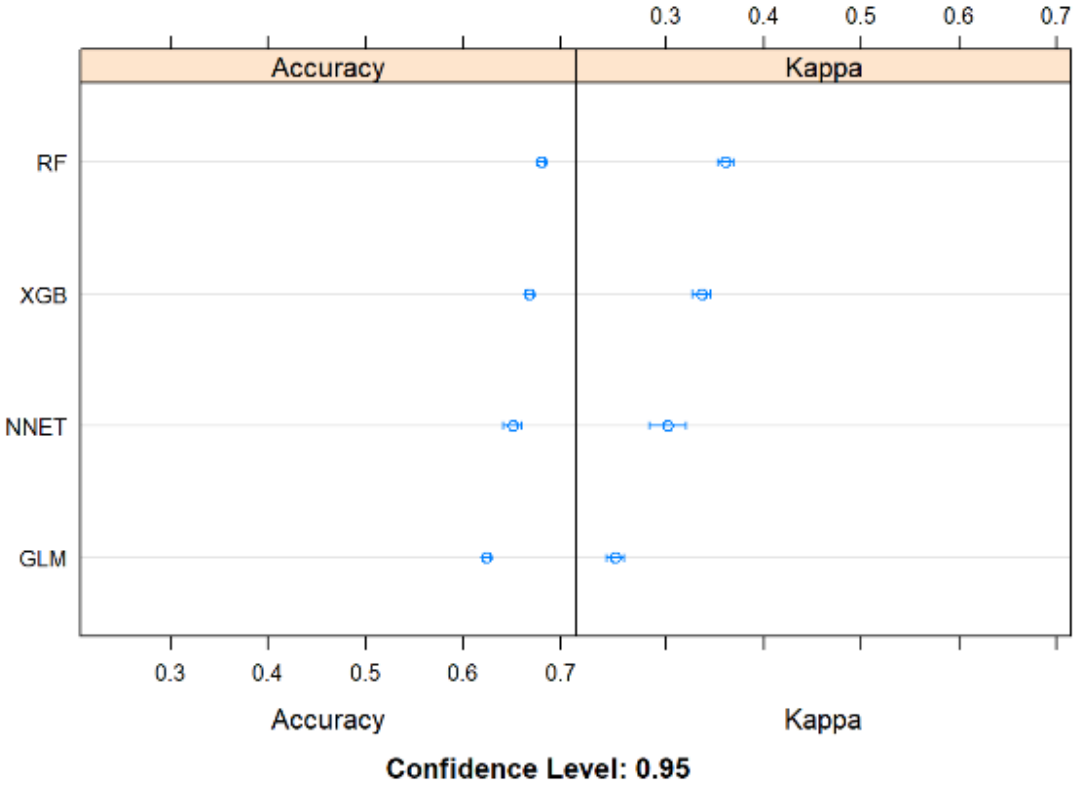

Figure 3 Accuracy comparison

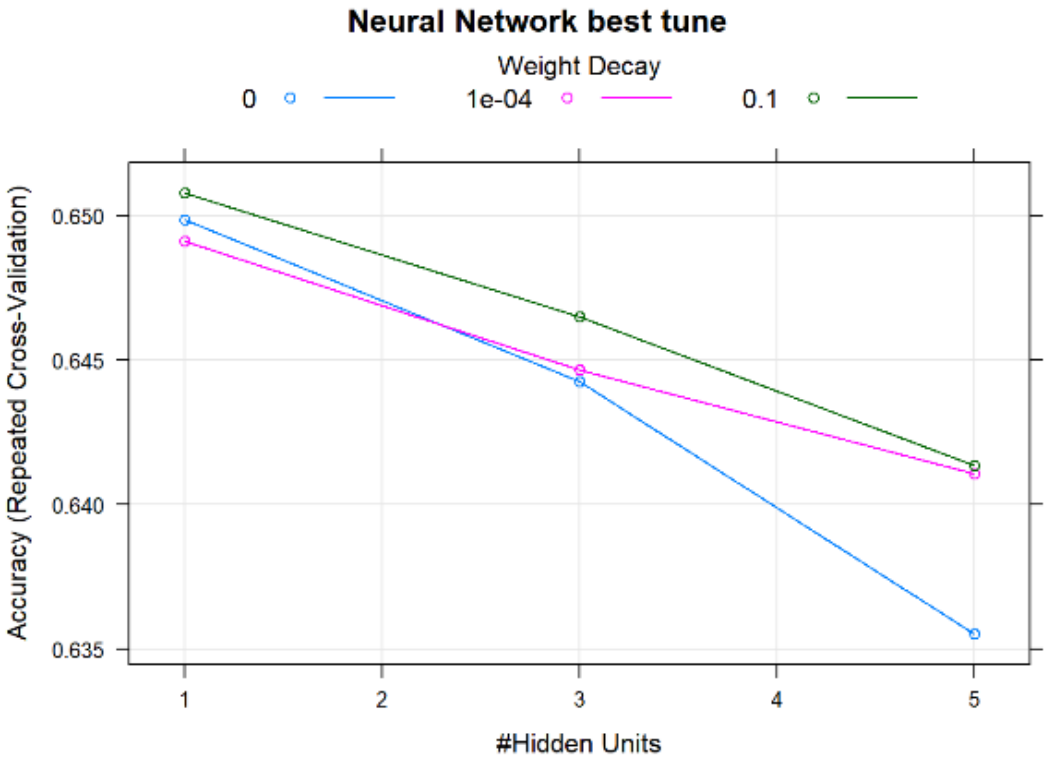

Figure 4 Neural network best tuning parameters

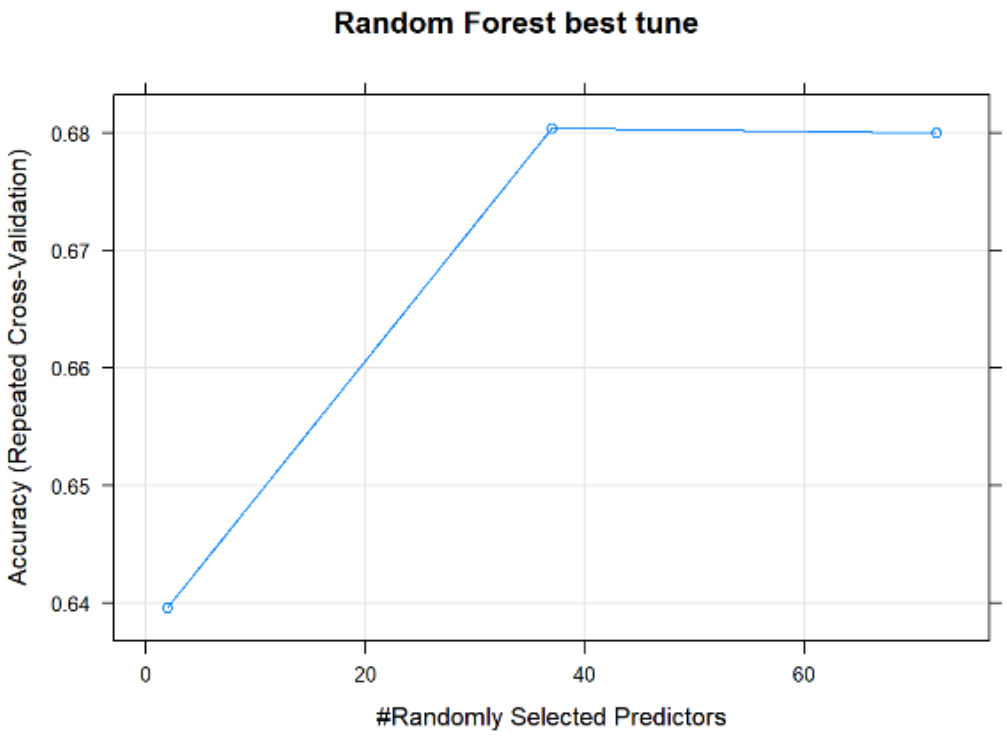

Figure 5 Random Forest best tuning parameters

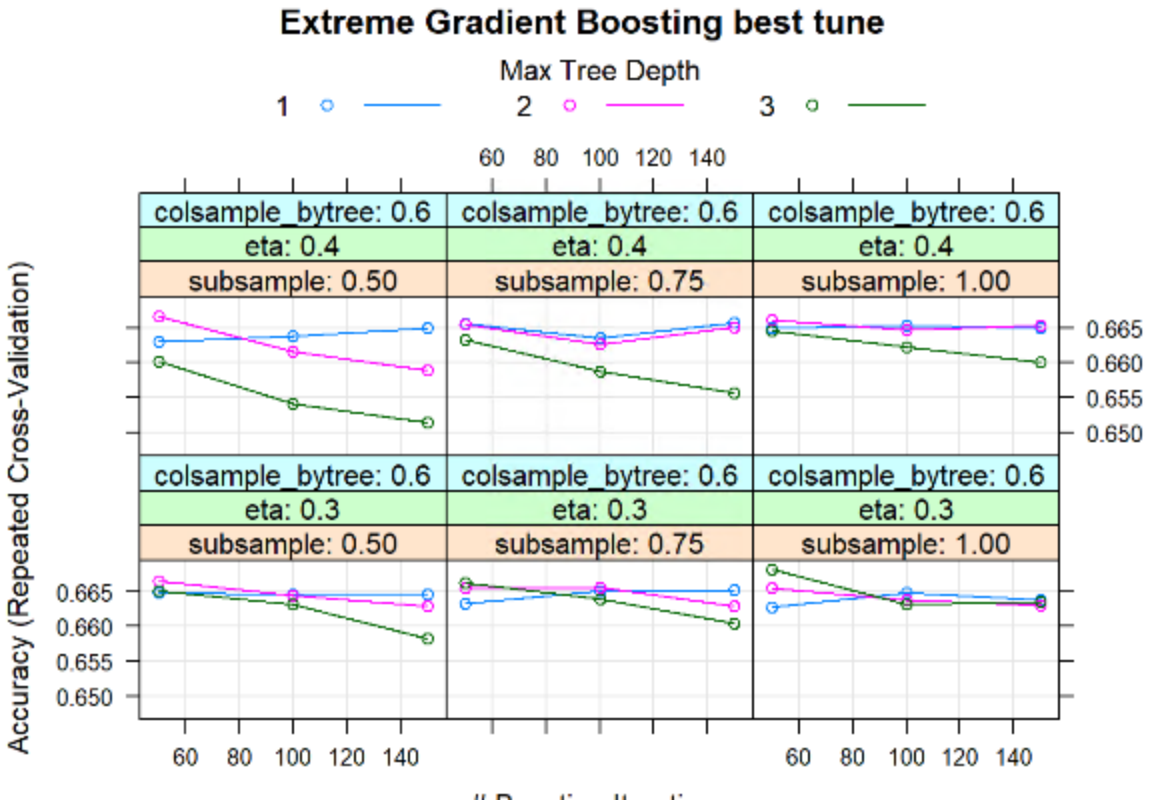

Figure 6 Extreme Gradient Boosting best tuning parameters

3.3 Performance on test set

The models were then used to predict preterm birth on the test set and their sensitivity and specificity can be found in Figures 7 and 8. The logistic regression model had the highest sensitivity and specificity with values of 0.66 and 0.69 respectively. The confusion matrix for each model can be found in tables 2-5 along with the accuracy on the test set.

The ROC plots along with the AUC for each of the models on the test set can be found in Figures 9 through Figure 12. Figure 13 shows each of the ROC curves overlaid on the same graph with random forest having the best ROC with an AUC of 0.731.

The variable importance plot for each of the models can be found in Figures 14-17. The logistic regression, neural network, and extreme gradient boosted tree models all had the number of prenatal visits as the most important variable. The random forest model had distance from the nearest major road or train tracks as the most influential variable to the model.

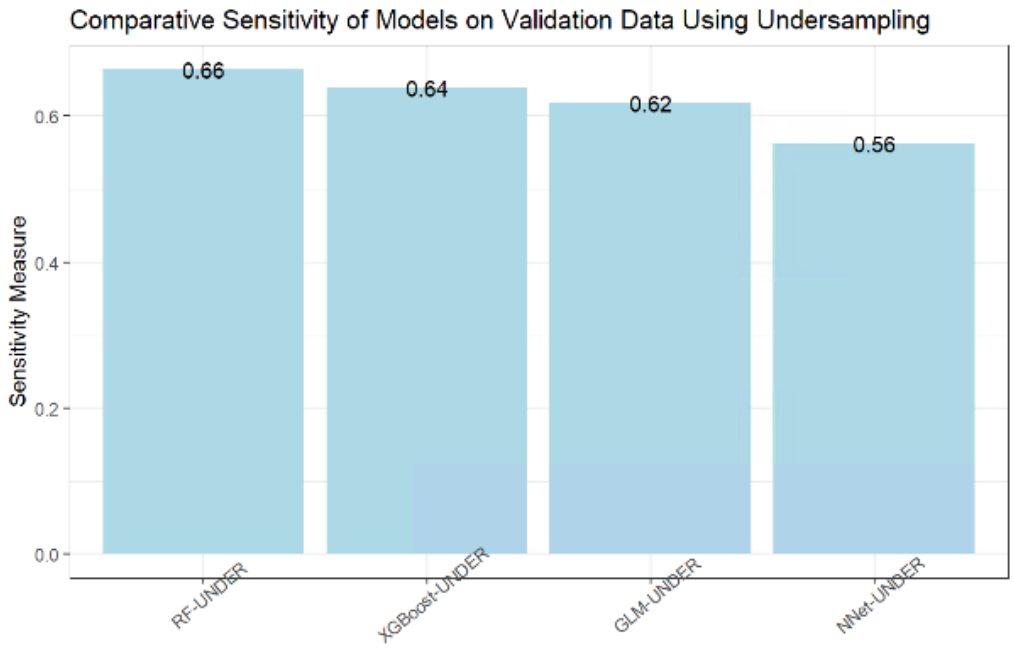

Figure 7 Sensitivity of models using the test set

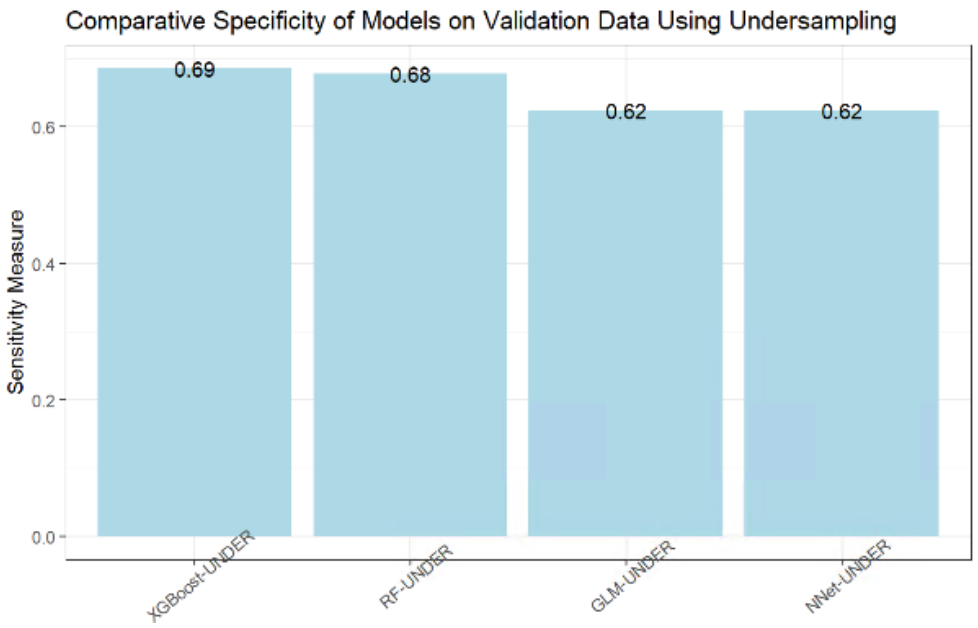

Figure 8 Sensitivity of models using the test set

Table 2 Generalized Linear Model Confusion Matrix

		Reference	
		Preterm	Term
Prediction	Preterm	1882	14657
	Term	1160	24373

Accuracy (average) = 0.6240

Table 3 Neural Network Confusion Matrix

		Reference	
		Preterm	Term
Prediction	Preterm	1711	10467
	Term	1331	28563

Accuracy (average) = 0.7196

Table 4 Random Forest Confusion Matrix

		Reference	
		Preterm	Term
Prediction	Preterm	2022	12558
	Term	1020	26472

Accuracy (average) = 0.6773

Table 5 Extreme Gradient Boosted Trees Confusion Matrix

		Reference	
		Preterm	Term
Prediction	Preterm	1945	12257
	Term	1097	26773

Accuracy (average) = 0.6826

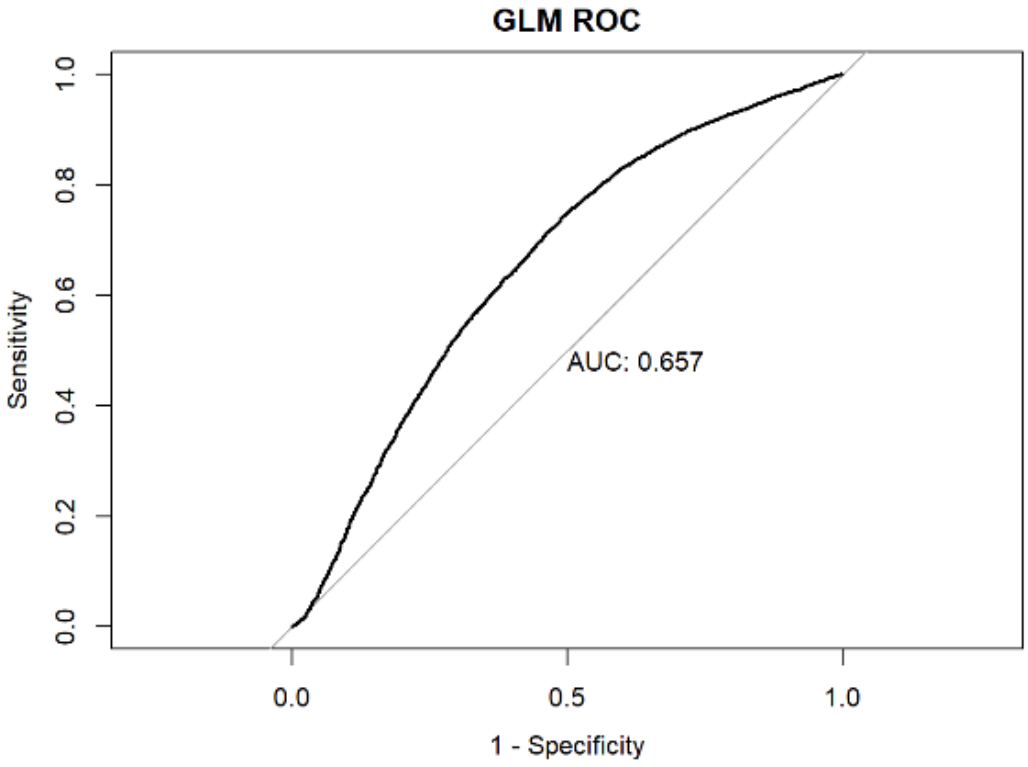

Figure 9 ROC plot for Generalized Linear Model

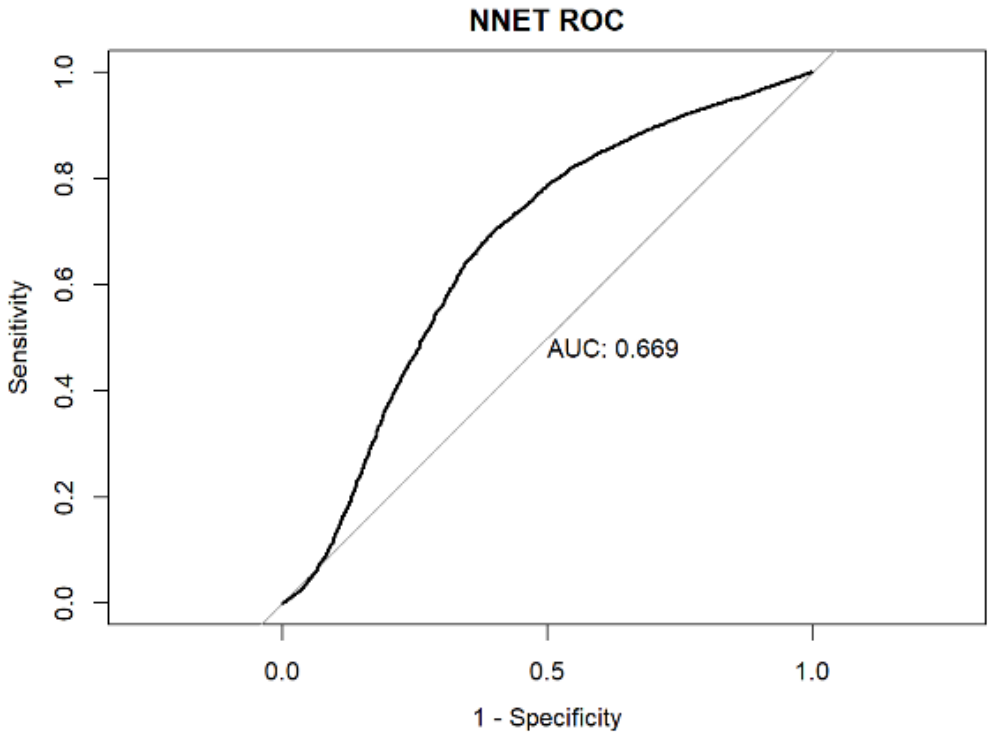

Figure 10 ROC plot for Neural Network

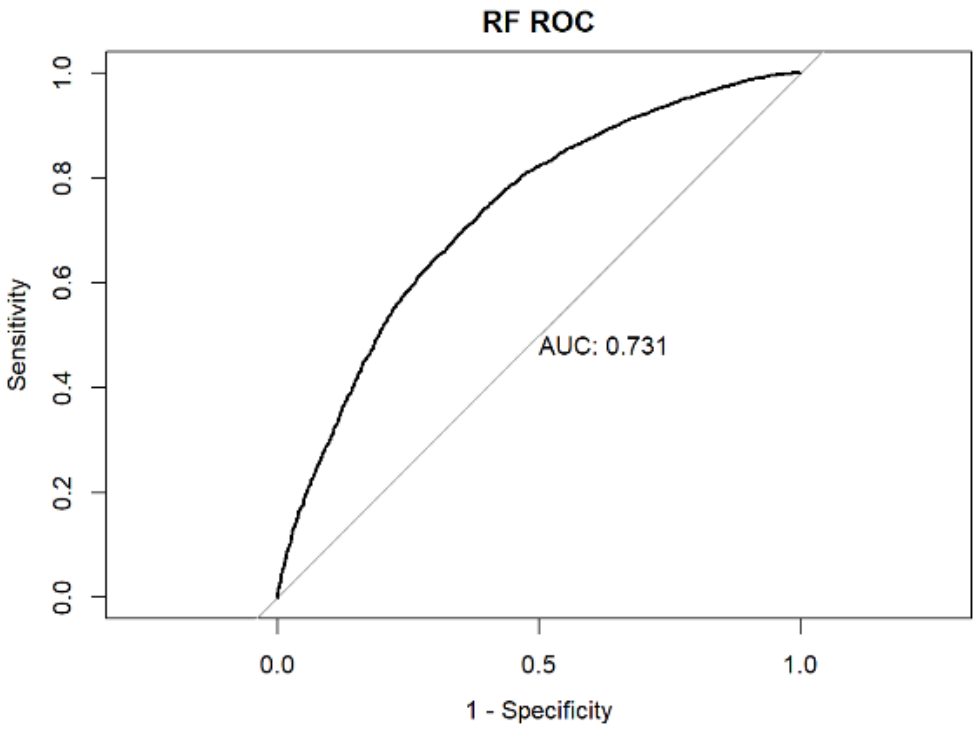

Figure 11 ROC plot for Random Forest

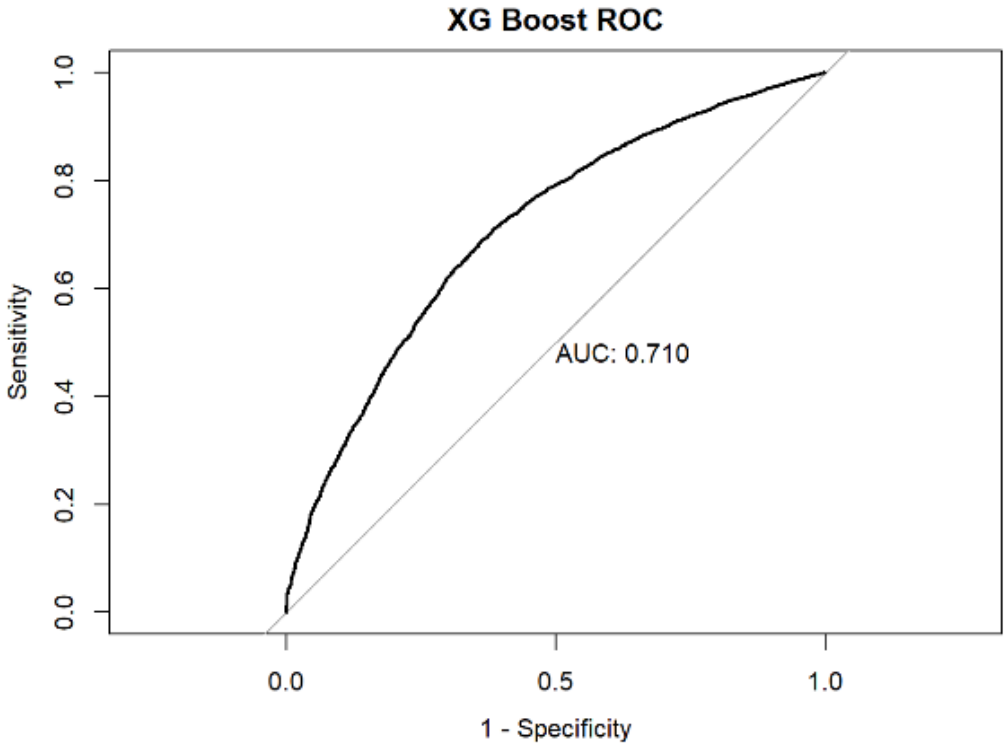

Figure 12 ROC plot for Extreme Gradient Boosting

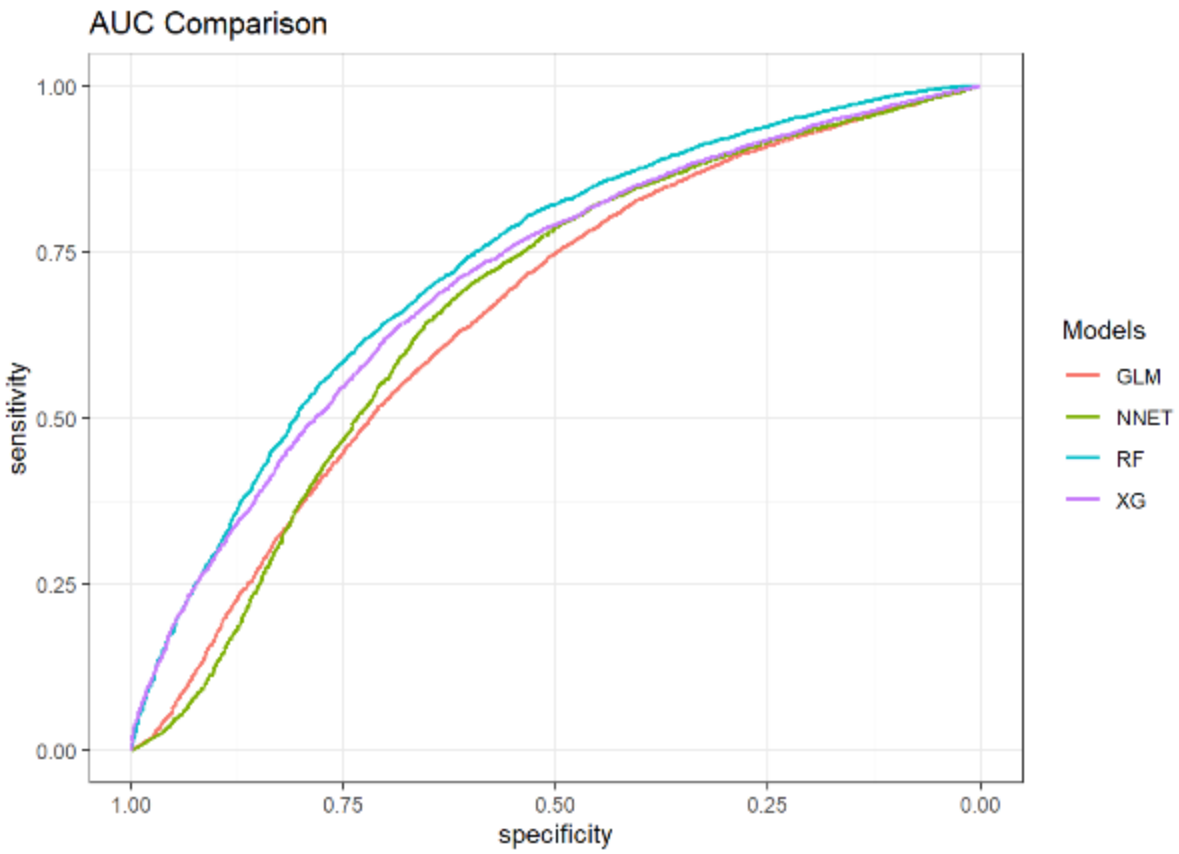

Figure 13 AUC comparison for all models

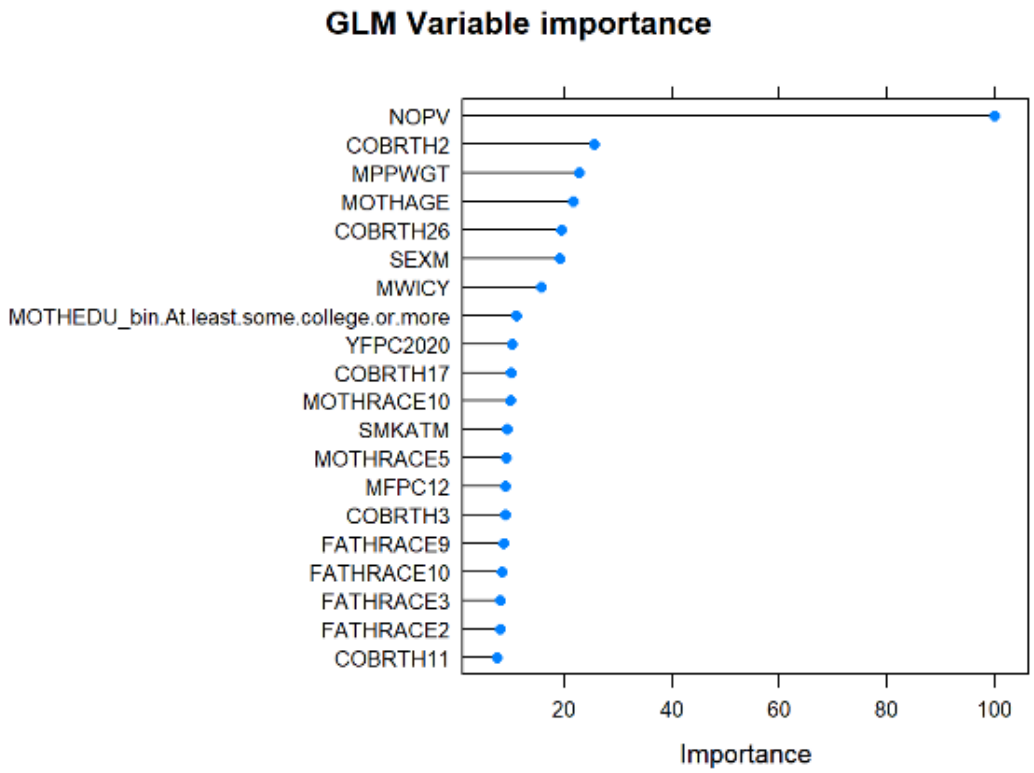

Figure 14 Generalized Linear Model variable importance

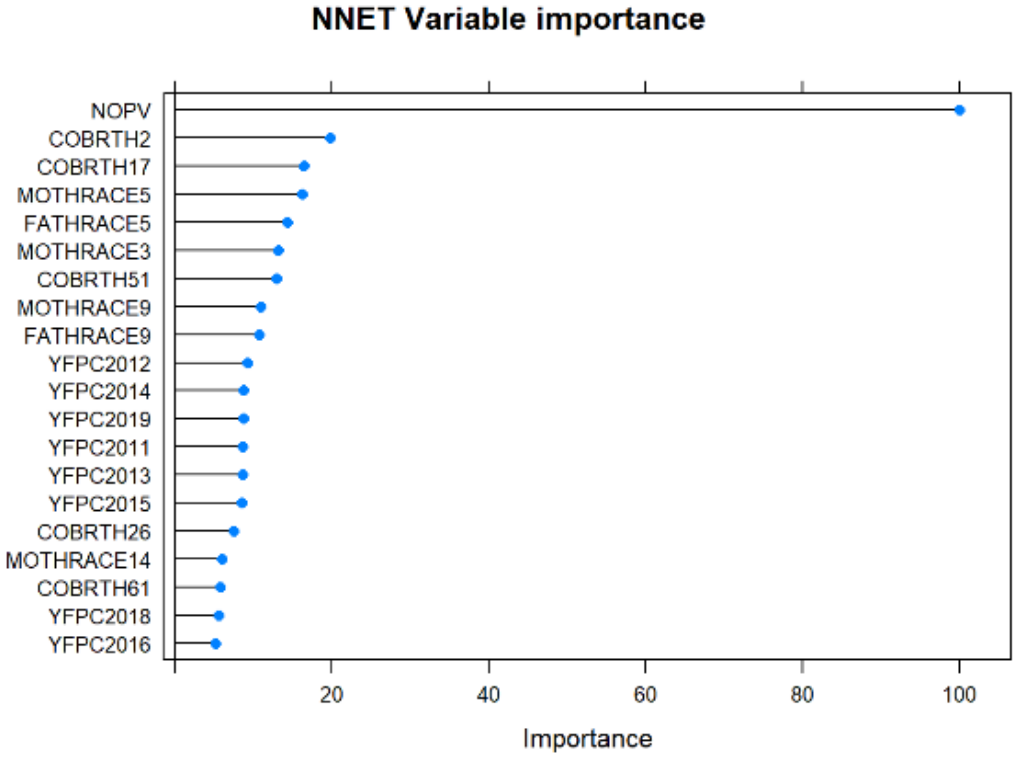

Figure 15 Neural Network variable importance

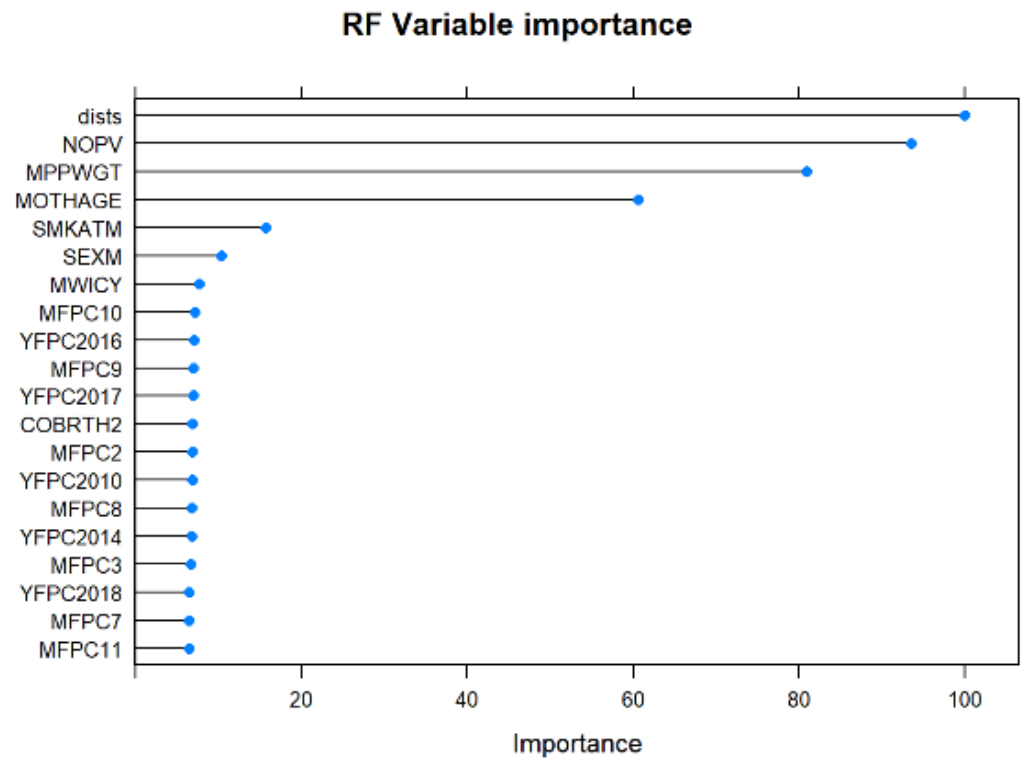

Figure 16 Random Forest variable importance

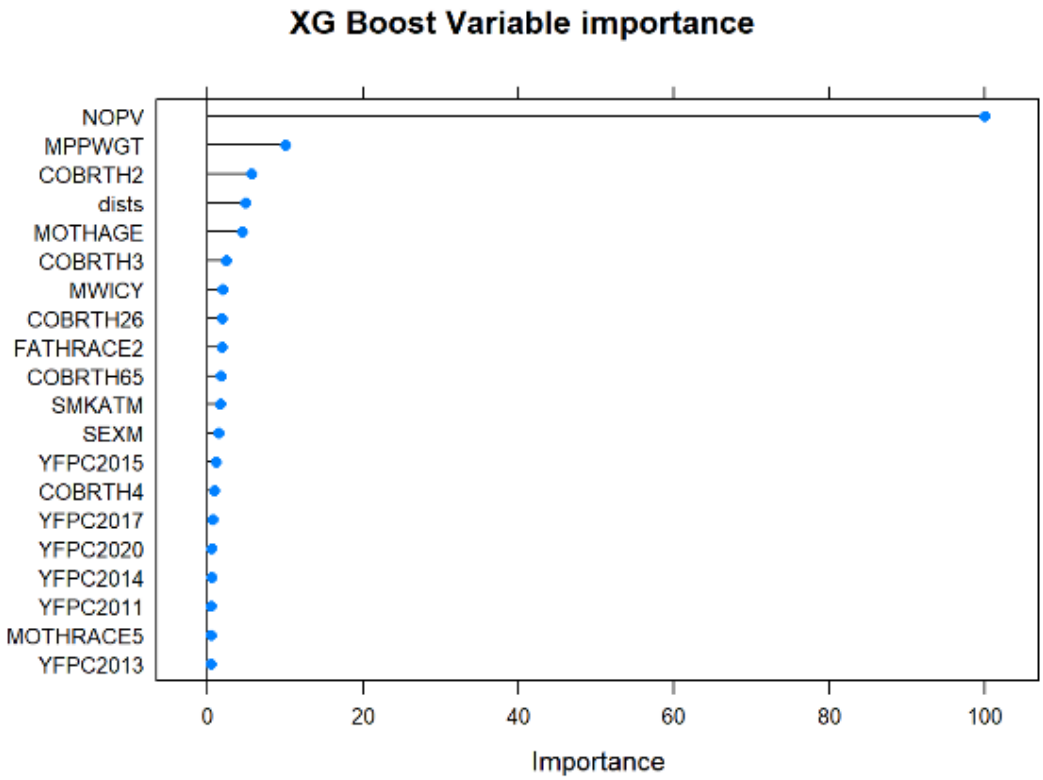

Figure 17 Extreme Gradient Boosting variable importance

4.0 Discussion

The goal of this thesis was to explore the relationships of preterm birth with demographic characteristics of the parents and the proximity of their residence to a major road or train tracks in Southwestern Pennsylvania and determine which of these variables most impacted preterm birth using machine learning techniques. After doing extensive exploratory data analysis, a subset of the data from the Health Effects of Hydraulic Fracturing Study was identified based on the variables of interest for this analysis. Using this subset of data, a 70/30 percent training/testing split was created for assessing performance of the dataset on observations not used to train the model. Undersampling of the term births in the 70% training set was then performed until an equal number of term and preterm were achieved so as not to violate the assumption of class equality of the outcome for logistic regression and the machine learning methods.

Implementing modeling techniques of neural network, random forest, and extreme gradient booted trees come with some limitations. The biggest limitations being a lack of interpretability of results. As part of the output of these models in R, there are no coefficients for each of the input variables. These coefficients would help give indication to the directional relationship that the input variable has with preterm birth as well as the magnitude of change in the outcome a single point of the input variable effects the model. Odds ratios, which aid in conveying the effects of variables on preterm birth, are not able to be computed from the output of the machine learning techniques.

This study furthers previous research done by Miranda et al. by expanding on their logistic regression analysis and applying machine learning to identify variables of importance by non-linear means. Miranda et al. found an association of preterm birth with proximity to roadways

using linear regression and this study concluded similar an association between these variables when using the random forest method. For the neural network and extreme gradient boosted trees models, the most important variable for the models was the total number of prenatal visits the mother had during pregnancy which aligned with the results of logistic regression. Meanwhile, the random forest model identified the distance of the mothers' residences as the driving variable in prediction of preterm birth followed closely by number of prenatal visits.

The random forest model performed the best in the metric of sensitivity and specificity with values of 0.66 and 0.68 respectively while the extreme gradient boosted trees was not far behind with 0.64 and 0.69 as seen in figures 7 and 8. Meanwhile, the accuracy of each model's ability to predict the observed outcome on the test set is seen in tables 2-5 with the neural network achieving the highest accuracy of 0.7196. These accuracy values are based off of which observations are selected for the training and testing set. The results could change if the seed for randomly deciding which observations are used for training and testing are altered. The final metric of comparison performed was the calculation of the area under the curve (AUC) of the receiver operator characteristic (ROC) curve. The individual ROC plots along with their corresponding AUC values are seen in figures 9-12 which indicate the random forest model as having the superior performance with and AUC of 0.731. In contrast, the neural network performs poorly in terms of AUC with a value of 0.669. For better clarification of the relation of the models ROC plots, figure 13 provides a side-by-side comparison of each model's performance. Considering all of these metrics together, the random forest model appears to have the best suited for modeling preterm birth in Southwestern Pennsylvania.

4.1 Conclusion

Preterm birth is a large and complex problem that has not clear-cut solution, but understanding the factors that contribute to it will help guide future public health decisions. The results of this study give indication that the number of prenatal visits has a large influence on preterm birth. There is also suggestive evidence that proximity to the nearest road also has bearing on preterm birth as seen in the random forest model selecting it as the most important measure. Using these results further study can be performed as to the effects of air pollutants on a pregnancy. The models have an accuracy around 66-71% suggest a fair ability to predict preterm birth but further exploration as an extension of this project should be done to iterate and improve upon the models and consider alternative methods.

Some possible extensions include exploring additional environmental measures to be included in models, testing different machine learning methods to implement, and including causal inference. For causal inference, determining the causal effect of the two most important variables of number of prenatal visits and distance of the mother's residence to the nearest major road should be considered rather than just the association on these variables with preterm birth.

Appendix A Analysis Executed in R

```
        library(tidyverse)
        library(RMariaDB)
        library(sf)

        con <- dbConnect(RMariaDB::MariaDB(),
                         default.file  = "D:/Brian/Road distance/.my.ini",
                         group = "fracking-group")

        birth_data <-dbReadTable(con, 'birth_data_Combined')

        #PA counties
        roads_allegheny             <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_42003_roads/tl_2020_42003_roads.shp')
        roads_armstrong             <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_42005_roads/tl_2020_42005_roads.shp')
        roads_beaver               <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_42007_roads/tl_2020_42007_roads.shp')
        roads_butler               <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_42019_roads/tl_2020_42019_roads.shp')
        roads_fayette              <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_42051_roads/tl_2020_42051_roads.shp')
        roads_greene               <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_42059_roads/tl_2020_42059_roads.shp')
        roads_washington            <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_42125_roads/tl_2020_42125_roads.shp')
        roads_westmoreland           <-            st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_42129_roads/tl_2020_42129_roads.shp')


        #PA border counties
        roads_lawrence              <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_42073_roads/tl_2020_42073_roads.shp')
        roads_mercer               <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_42085_roads/tl_2020_42085_roads.shp')
        roads_venango              <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_42121_roads/tl_2020_42121_roads.shp')
        roads_clarion              <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_42031_roads/tl_2020_42031_roads.shp')
        roads_jefferson             <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_42065_roads/tl_2020_42065_roads.shp')
        roads_indiana              <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_42063_roads/tl_2020_42063_roads.shp')
        roads_cambria              <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_42021_roads/tl_2020_42021_roads.shp')
        roads_somerset             <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_42111_roads/tl_2020_42111_roads.shp')




        #OH border counties
        roads_columbiana            <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_39029_roads/tl_2020_39029_roads.shp')


        #WV border counties
        roads_hancock              <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_54029_roads/tl_2020_54029_roads.shp')
        roads_brooke               <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_54009_roads/tl_2020_54009_roads.shp')
        roads_ohio                <-              st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_54069_roads/tl_2020_54069_roads.shp')
```

```
       roads_marshall             <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_54051_roads/tl_2020_54051_roads.shp')
       roads_wetzel             <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_54103_roads/tl_2020_54103_roads.shp')
       roads_monongalia            <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_54061_roads/tl_2020_54061_roads.shp')
       roads_preston             <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_54077_roads/tl_2020_54077_roads.shp')


       #MD border counties
       roads_garrett             <-             st_read('D:/Brian/Road             distance/Road
Shapefiles/tl_2020_24023_roads/tl_2020_24023_roads.shp')



       rails <- st_read('D:/Brian/Road distance/tl_2020_us_rails/tl_2020_us_rails.shp') # all us
rails
       pa_boundary <- st_read('D:/Brian/Road distance/tl_2020_us_state/tl_2020_us_state.shp') %>%
         filter(NAME == 'Pennsylvania')
       pa_rails <- st_intersection(rails, pa_boundary)



       # border county boundaries
       border_geoid <- c('39029', '54029', '54009', '54069', '54051', '54103', '54061', '54077',
'24023', '42073', '42085', '42121',
                         '42031', '42065', '42063', '42021', '42111')
       border_counties                          <-                          st_read('D:/Brian/Road
distance/tl_2020_us_county/tl_2020_us_county.shp') %>%
                           filter(GEOID %in% border_geoid)
       border_counties_boundary <- st_union(border_counties)

       border_county_rails <- st_intersection(rails, border_counties_boundary)




       #study area boundaries
       study_counties_geoid <- c('42003', '42005', '42007', '42019','42051', '42059', '42125',
'42129')
       study_counties                           <-                          st_read('D:/Brian/Road
distance/tl_2020_us_county/tl_2020_us_county.shp') %>%
         filter(GEOID %in% study_counties_geoid)
       study_counties_boundary <- st_union(study_counties)
       study_county_rails <- st_intersection(rails, study_counties_boundary)




       # S1100 = primary road
       # S1200 = secondary road
       # S1400 = local neighborhood road, rural road, city street
       # S1740 = private road for service vehicles
       road_types <- c('S1100', 'S1200')


       roads_pa   <-   rbind(roads_allegheny,   roads_armstrong,   roads_beaver,   roads_butler,
roads_fayette, roads_greene,
                         roads_washington, roads_westmoreland) %>% mutate(state = 'PA')
       roads_pa2 <- rbind(roads_lawrence, roads_mercer, roads_venango, roads_clarion,
                         roads_jefferson,  roads_indiana,  roads_cambria,  roads_somerset)  %>%
mutate(state = 'PA')
       roads_oh <- roads_columbiana %>% mutate(state = 'OH')
       roads_wv <- rbind(roads_hancock, roads_brooke, roads_ohio, roads_marshall, roads_wetzel,
roads_monongalia, roads_preston) %>%
                       mutate(state = 'WV')
       roads_md <-roads_garrett %>% mutate(state = 'MD')
```

```
roads_all <- rbind(roads_pa, roads_oh, roads_wv, roads_md, roads_pa2)

roads_filtered <- filter(roads_all, MTFCC %in% road_types)



roads_and_rails <- bind_rows(study_county_rails, border_county_rails, roads_filtered)



rr_simp <- st_simplify(roads_and_rails, dTolerance = 10000)




# sql_statement <- "select
#                      momrescounty,
#                      derived_lat,
#                      derived_long
#                   from
#                      birth_data_Combined
#                   where
#                                                                          momrescounty   regexp
'allegheny|armstrong|beaver|butler|greene|fayette|washington|westmoreland'
#                      and derived_lat is not null
#                      and derived_long is not null;"

sql_statement <- "select
                     momrescounty,
                     derived_lat,
                     derived_long,
                     Birth_ID
                  from
                     birth_data_Combined
                  where
                     derived_lat is not null
                     and derived_long is not null;"


birth_data <- dbGetQuery(conn = con, statement = sql_statement) %>%
                   st_as_sf(coords = c("derived_long", "derived_lat"),
                            crs = 'NAD83', agr = "constant")


birth_data_filtered <- st_intersection(birth_data, study_counties_boundary)



save(roads_and_rails,    birth_data_filtered,    birth_data,    file    =    'D:/Brian/Road
distance/res_and_roads_brian.RData')
save(study_counties,      study_counties_boundary,      border_counties,      pa_boundary,
border_counties_boundary, file = 'D:/Brian/Road distance/boundaries_brian.RData')
save.image('D:/Brian/Road distance/load_data_brian.RData')


load('D:/Brian/Road distance/res_and_roads_brian.RData')
load('D:/Brian/Road distance/roads_filtered.RData')

library(sf)

mom_res <- st_as_sf(birth_data_filtered,
                     coords = c("derived_long", "derived_lat"),
                     crs = 'NAD83', agr = "constant")


#all
roads_and_rails$id <- 1:nrow(roads_and_rails)
nearest_roads <- st_join(mom_res, roads_and_rails, join = st_nearest_feature)

dists <- st_distance(mom_res, roads_and_rails[nearest_roads$id, ], by_element = TRUE)
```

```
nearest_roads$dists <- dists

##st_distance(mom_res[1,], roads_and_rails[nearest_roads$id, ], by_element = TRUE)

nearest_roads_reduced <- nearest_roads %>%
  as.data.frame() %>%
  select(Birth_ID, dists)


save(mom_res, nearest_roads, dists, nearest_roads_reduced, file = 'D:/Brian/Road distance/dists_brian.RData')

---
title: "Brian Thesis EDA"
author: "Brian O'Connell"
date: "1/14/2022"
output: html_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

```{r, message=FALSE}
library(RMariaDB)
library(tidyverse)
library(naniar)
```
Linking the SQL server to R

```{r, include=TRUE}
con <- dbConnect(RMariaDB::MariaDB(),
default.file  = "C:/Users/brr99/.my.ini",
group = "fracking-group")
```

```{r}
dbListTables(con)
```

Getting birthdata table

```{r}
statement_1 <- 'select * from birth_data_Combined'
```

```{r}
## assign data to an object
birth_data_Combined <- dbGetQuery(conn = con, statement = statement_1)
head(birth_data_Combined)
```


```{r}
birth_data_Combined <- birth_data_Combined %>% filter(LOP > 21)
load('D:/Brian/Road distance/dists_brian.RData')
birth_data_Combined <- mutate(.data = birth_data_Combined, "preterm" = ifelse(LOP < 37, "preterm","term"))
```

```{r}
birth_data_Combined$SMKATM <- birth_data_Combined$SMKFTM + birth_data_Combined$SMKSTM + birth_data_Combined$SMKLTM

birth_data_Combined <- merge(birth_data_Combined,nearest_roads_reduced,by="Birth_ID")
```


```{r}
```
```

```
# create a reduced data set with only the variables of interest
reduced_df <- birth_data_Combined %>% select(Birth_ID, SEX, COBRTH, MOTHAGE, MOTHEDU, MOTHRACE, MOTHHISP, FATHRACE, FATHHISP,MFPC, YFPC, NOPV, MPPWGT, MWIC, SMKPR, SMKFTM, SMKSTM, SMKLTM, SMKATM, R2, LOP, LBIRTH, PLURAL, preterm, dists)
```

```{r}
# checking for missing values
reduced_df %>% purrr::map_dbl(~sum(is.na(.)))

```

```{r}
# remove the rows with na values
reduced_df <- reduced_df %>%  drop_na()

```

```{r}
reduced_df %>% purrr::map_dbl(~sum(is.na(.)))
```

```{r}
reduced_df$MOTHAGE[reduced_df$MOTHAGE==99] <- NA
reduced_df$MOTHEDU[reduced_df$MOTHEDU==9] <- NA
reduced_df$MOTHRACE[reduced_df$MOTHRACE==16] <- NA
reduced_df$MOTHRACE[reduced_df$MOTHRACE==17] <- NA
reduced_df$FATHRACE[reduced_df$FATHRACE==16] <- NA
reduced_df$FATHRACE[reduced_df$FATHRACE==17] <- NA
reduced_df$MFPC[reduced_df$MFPC==99] <- NA
reduced_df$MFPC[reduced_df$MFPC==88] <- NA
reduced_df$NOPV[reduced_df$NOPV==99] <- NA
reduced_df$SMKPR[reduced_df$SMKPR==99] <- NA
reduced_df$SMKPR[reduced_df$SMKPR==98] <- NA
reduced_df$SMKFTM[reduced_df$SMKFTM==99] <- NA
reduced_df$SMKFTM[reduced_df$SMKFTM==98] <- NA
reduced_df$SMKSTM[reduced_df$SMKSTM==99] <- NA
reduced_df$SMKSTM[reduced_df$SMKSTM==98] <- NA
reduced_df$SMKLTM[reduced_df$SMKLTM==99] <- NA
reduced_df$SMKLTM[reduced_df$SMKLTM==98] <- NA
reduced_df$LOP[reduced_df$LOP==99] <- NA
reduced_df$MPPWGT[reduced_df$MPPWGT==999] <- NA
reduced_df$YFPC[reduced_df$YFPC==9999] <- NA
reduced_df$YFPC[reduced_df$YFPC==8888] <- NA
reduced_df$MWIC[reduced_df$MWIC=='X'] <- NA
reduced_df$MWIC[reduced_df$MWIC=='U'] <- NA
reduced_df$LBIRTH[reduced_df$LBIRTH==99] <- NA
reduced_df$MOTHHISP[reduced_df$MOTHHISP==9] <- NA
reduced_df$FATHHISP[reduced_df$FATHHISP==9] <- NA
```

```{r}
reduced_df %>% purrr::map_dbl(~sum(is.na(.)))
```

```{r}
reduced_df2 <- reduced_df %>%  drop_na()

```

Sex of the baby

```{r}
options(scipen=10000)
```

```
reduced_df2 %>%
  ggplot(mapping = aes(x = SEX)) +
  geom_bar() +
  theme_bw() +
  ggtitle("Sex of the baby")
```

```{r}
table(reduced_df2$SEX)
```

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = SEX)) +
  geom_bar(mapping = aes(fill = preterm),
           position = 'dodge') +
  scale_fill_brewer(palette = "Dark2") +
  theme_bw()
```

County of birth

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = COBRTH)) +
  geom_bar() +
  theme_bw() +
  ggtitle("County of birth")
```

```{r}
mode <- function(codes){
  which.max(tabulate(codes))
}

summarise(.data = reduced_df2,
          mean = mean(COBRTH),
          median = median(COBRTH),
          mode = mode(COBRTH),
          max = max(COBRTH),
          min = min(COBRTH))
```

Age of the mother

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = MOTHAGE)) +
  geom_histogram(bins = 40) +
  theme_bw() +
  ggtitle("Age of the mother")
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(MOTHAGE),
          median = median(MOTHAGE),
          mode = mode(MOTHAGE),
          max = max(MOTHAGE),
          min = min(MOTHAGE))
```

Education of the mother

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = MOTHEDU)) +
  geom_bar(bins = 40) +


```

```
  theme_bw() +
  ggtitle("Education of the mother")
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(MOTHEDU),
          median = median(MOTHEDU),
          mode = mode(MOTHEDU),
          max = max(MOTHEDU),
          min = min(MOTHEDU))
```

```{r}
# set up cut-off values
breaks <- c(0,3,4,8)
# specify interval/bin labels
tags <- c("Less than Highschool","Highschool or GED", "At least some college or more")
# bucketing values into bins
reduced_df2$MOTHEDU_bin <- cut(reduced_df2$MOTHEDU,
                  breaks=breaks,
                  include.lowest=TRUE,
                  right=FALSE,
                  labels=tags)
# inspect bins
summary(reduced_df2$MOTHEDU_bin)
```

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = MOTHEDU_bin)) +
  geom_bar(mapping = aes(fill = preterm),
           position = 'dodge') +
  scale_fill_brewer(palette = "Dark2") +
  theme_bw()
```

Mothers race

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = MOTHRACE)) +
  geom_bar() +
  theme_bw() +
  ggtitle("Mothers race")
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(MOTHRACE),
          median = median(MOTHRACE),
          mode = mode(MOTHRACE),
          max = max(MOTHRACE),
          min = min(MOTHRACE))
```

Mother's Hispanic Origin

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = MOTHHISP)) +
  geom_bar() +
  theme_bw() +
  ggtitle("Mother's Hispanic Origin")
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(MOTHHISP),
          median = median(MOTHHISP),
          mode = mode(MOTHHISP),
          max = max(MOTHHISP),
          min = min(MOTHHISP))
```

```{r}
# set up cut-off values
breaks <- c(0,2,6)
# specify interval/bin labels
tags <- c("Not Hispanic","Hispanic")
# bucketing values into bins
reduced_df2$MOTHHISP_bin <- cut(reduced_df2$MOTHHISP,
                  breaks=breaks,
                  include.lowest=TRUE,
                  right=FALSE,
                  labels=tags)
# inspect bins
summary(reduced_df2$MOTHHISP_bin)
```

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = MOTHHISP_bin)) +
  geom_bar() +
  theme_bw() +
  ggtitle("Mother's Hispanic Origin Collapsed")
```


```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = MOTHHISP_bin)) +
  geom_bar(mapping = aes(fill = preterm),
           position = 'dodge') +
  scale_fill_brewer(palette = "Dark2") +
  theme_bw()
```


Fathers race

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = FATHRACE)) +
  geom_bar() +
  theme_bw() +
  ggtitle("Fathers race")
# need mapping for races
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(FATHRACE),
          median = median(FATHRACE),
          mode = mode(FATHRACE),
          max = max(FATHRACE),
          min = min(FATHRACE))
```

Father's Hispanic Origin

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = FATHHISP)) +
  geom_bar() +
  theme_bw() +
```

```
  ggtitle("Mother's Hispanic Origin")
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(FATHHISP),
          median = median(FATHHISP),
          mode = mode(FATHHISP),
          max = max(FATHHISP),
          min = min(FATHHISP))
```

```{r}
# set up cut-off values
breaks <- c(0,2,6)
# specify interval/bin labels
tags <- c("Not Hispanic","Hispanic")
# bucketing values into bins
reduced_df2$FATHHISP_bin <- cut(reduced_df2$FATHHISP,
                  breaks=breaks,
                  include.lowest=TRUE,
                  right=FALSE,
                  labels=tags)
# inspect bins
summary(reduced_df2$FATHHISP_bin)
```

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = FATHHISP_bin)) +
  geom_bar() +
  theme_bw() +
  ggtitle("Father's Hispanic Origin Collapsed")
```

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = FATHHISP_bin)) +
  geom_bar(mapping = aes(fill = preterm),
           position = 'dodge') +
  scale_fill_brewer(palette = "Dark2") +
  theme_bw()
```

Month of first paternal visit

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = MFPC)) +
  geom_bar() +
  theme_bw() +
  ggtitle("Month of first paternal visit")
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(MFPC),
          median = median(MFPC),
          mode = mode(MFPC),
          max = max(MFPC),
          min = min(MFPC))
```

Year of First Prenatal Care Visit; YYYY format

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = YFPC)) +
```

```
  geom_bar() +
  theme_bw() +
  ggtitle("Year of First Prenatal Care Visit; YYYY format")
# some missing values encoded?
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(YFPC),
          median = median(YFPC),
          mode = mode(YFPC),
          max = max(YFPC),
          min = min(YFPC))
```

Total Number of Prenatal Visits

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = NOPV)) +
  geom_histogram() +
  theme_bw() +
  ggtitle("Total Number of Prenatal Visits")
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(NOPV),
          median = median(NOPV),
          mode = mode(NOPV),
          max = max(NOPV),
          min = min(NOPV))
```

Mother's Pre-pregnancy Weight (in pounds)

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = MPPWGT)) +
  geom_histogram() +
  theme_bw() +
  ggtitle("Mother's Pre-pregnancy Weight (in pounds)")
```


```{r}
summarise(.data = reduced_df2,
          mean = mean(MPPWGT),
          median = median(MPPWGT),
          mode = mode(MPPWGT),
          max = max(MPPWGT),
          min = min(MPPWGT))
```

Did Mother get WIC food?

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = MWIC)) +
  geom_bar() +
  theme_bw() +
  ggtitle("Mother WIC food status")
```

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = MWIC)) +
  geom_bar(mapping = aes(fill = preterm),
           position = 'dodge') +
  scale_fill_brewer(palette = "Dark2") +
  theme_bw()
```

```

```{r}
table(reduced_df2$MWIC)
```

Number of cigarettes smoked three months prior

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = SMKPR)) +
  geom_histogram() +
  theme_bw() +
  ggtitle("Number of cigarettes smoked three months prior to pregnancy")
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(SMKPR),
          median = median(SMKPR),
          mode = mode(SMKPR),
          max = max(SMKPR),
          min = min(SMKPR))
```

Number of cigarettes smoked first three months of pregnancy

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = SMKFTM)) +
  geom_histogram() +
  theme_bw() +
  ggtitle("Number of cigarettes smoked first three months of pregnancy")
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(SMKFTM),
          median = median(SMKFTM),
          mode = mode(SMKFTM),
          max = max(SMKFTM),
          min = min(SMKFTM))
```

Number of cigarettes smoked second three months of pregnancy

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = SMKSTM)) +
  geom_histogram() +
  theme_bw() +
  ggtitle("Number of cigarettes smoked second three months of pregnancy")
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(SMKSTM),
          median = median(SMKSTM),
          mode = mode(SMKSTM),
          max = max(SMKSTM),
          min = min(SMKSTM))
```

Number of cigarettes smoked last three months of pregnancy

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = SMKLTM)) +
  geom_histogram() +
  theme_bw() +
```
```

```
  ggtitle("Number of cigarettes smoked last three months of pregnancy")
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(SMKLTM),
          median = median(SMKLTM),
          mode = mode(SMKLTM),
          max = max(SMKLTM),
          min = min(SMKLTM))
```

Number of cigarettes smoked during whole pregnancy

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = SMKATM)) +
  geom_histogram() +
  theme_bw() +
  ggtitle("Number of cigarettes smoked last three months of pregnancy")
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(SMKATM),
          median = median(SMKATM),
          mode = mode(SMKATM),
          max = max(SMKATM),
          min = min(SMKATM))
```

Risk Factors in this Pregnancy - Gestational Diabetes

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = R2)) +
  geom_bar() +
  theme_bw() +
  ggtitle("Risk Factors in this Pregnancy - Gestational Diabetes")
```

```{r}
table(reduced_df2$R2)
```

Obstetric Estimate of Gestation

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = LOP)) +
  geom_histogram() +
  stat_bin(bins = 20) +
  theme_bw() +
  ggtitle("Obstetric Estimate of Gestation")
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(LOP),
          median = median(LOP),
          mode = mode(LOP),
          max = max(LOP),
          min = min(LOP))
```

Plurality

```{r}


```

```
reduced_df2 %>%
  ggplot(mapping = aes(x = PLURAL)) +
  geom_histogram() +
  theme_bw() +
  ggtitle("PLurality - Number of fetuses at birth")
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(PLURAL),
          median = median(PLURAL),
          mode = mode(PLURAL),
          max = max(PLURAL),
          min = min(PLURAL))
```

Number of previous births

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = LBIRTH)) +
  geom_bar() +
  theme_bw() +
  ggtitle("Number of previous births")
```

```{r}
summarise(.data = reduced_df2,
          median = median(LBIRTH),
          max = max(LBIRTH),
          min = min(LBIRTH))
```

Preterm birth

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = preterm)) +
  geom_bar() +
  theme_bw() +
  ggtitle("Preterm birth")
```

```{r}
table(reduced_df2$preterm)
mean(reduced_df2$preterm == "preterm")
```

Distances

```{r}
reduced_df2 %>%
  ggplot(mapping = aes(x = dists)) +
  geom_histogram() +
  stat_bin(binwidth = 15 ,bins = 20) +
  theme_bw() +
  ggtitle("Distance of mother's residence to nearest major road or train tracks")
```

```{r}
summarise(.data = reduced_df2,
          mean = mean(dists),
          median = median(dists),
          max = max(dists),
          min = min(dists))
```

```{r}
reduced_df2 %>%
  select(MOTHAGE, MPPWGT, NOPV, SMKPR, SMKFTM, SMKSTM, SMKLTM, dists) %>%
  cor() %>%
  corrplot::corrplot(type = "upper", method = "number")
```

```{r}
reduced_df2 %>%
  select(MOTHAGE, MPPWGT, NOPV, SMKATM, dists) %>%
  cor() %>%
  corrplot::corrplot(type = "upper", method = "number")
```

```{r}
save(reduced_df2, file = 'C:/Users/brr99/Documents/Thesis Project/birth_data_Cleaned.RData')
```
---
title: "Model building random undersampling-1"
author: "Brian O'Connell"
date: "2/28/2022"
output: html_document

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

```{r, message=FALSE}
library(tidyverse)
library(caret)
library(smotefamily)
library(corrplot)
library(randomForest)
library(xgboost)
library(pROC)
library(ROSE)
library(tableone)
library(table1)
```

```{r}
load(file = 'birth_data_Cleaned.RData')
```

```{r}
reduced_df3 <- reduced_df2 %>% select(SEX, COBRTH, MOTHAGE, MOTHEDU_bin, MOTHRACE, MOTHHISP_bin, FATHRACE, FATHHISP_bin, MFPC, YFPC, NOPV, MPPWGT, MWIC, SMKATM, dists, preterm)
```

```{r}
sapply(reduced_df3, class)
```

```{r}
reduced_df3$COBRTH <- as.character(reduced_df3$COBRTH)
reduced_df3$MFPC <- as.character(reduced_df3$MFPC)
reduced_df3$YFPC <- as.character(reduced_df3$YFPC)
reduced_df3$MOTHRACE <- as.character(reduced_df3$MOTHRACE)
reduced_df3$FATHRACE <- as.character(reduced_df3$FATHRACE)
reduced_df3$MWIC <- as.character(reduced_df3$MWIC)
```

```{r}
dmy <- dummyVars("~ SEX + MOTHEDU_bin + COBRTH + MOTHRACE + MOTHHISP_bin + FATHRACE + FATHHISP_bin + MWIC", data = reduced_df3, fullRank = T)
trsf <- data.frame(predict(dmy, newdata = reduced_df3))
```

```{r}
reduced_df4 <- reduced_df3[,-c(1,2,4,5,6,7,8,13)]
reduced_df4 <- cbind(reduced_df4,trsf)
```

```{r}
table(reduced_df4$preterm)
```

```{r}
reduced_df4 <- subset(reduced_df4, select = -c(COBRTH14, COBRTH18, COBRTH19, COBRTH23,COBRTH25, COBRTH28, COBRTH31, COBRTH34, COBRTH40, COBRTH41, COBRTH5, COBRTH53, COBRTH54, COBRTH60, COBRTH62, COBRTH67, COBRTH8, COBRTH9, COBRTH15, COBRTH30, COBRTH36, COBRTH42, COBRTH6, COBRTH20, COBRTH22, COBRTH33, COBRTH35, COBRTH7, COBRTH47,MOTHRACE12, FATHRACE13, MOTHRACE11, MOTHRACE13, FATHRACE12, FATHRACE11))
```

```{r}
CreateTableOne(data = reduced_df4)
```

```{r}
table1(~ ., data=reduced_df4)
```

```{r}
# Train/test split
set.seed(2337)
train <- createDataPartition(reduced_df4$preterm,
                             p = 0.7,
                             times = 1,
                             list = F)
train.orig <- reduced_df4[ train,]
test       <- reduced_df4[-train,]
```

```{r}
table(train.orig$preterm)
```

```{r}
set.seed(1234)
data_balanced_under <- ovun.sample(preterm ~ ., data = train.orig, method = "under", N = 14198)$data
table(data_balanced_under$preterm)
```

```{r}
ctrl_acc <- trainControl(method = "repeatedcv", number = 5)
metric_acc <- "Accuracy"
```

```{r}
set.seed(1234)

fit_glm_acc_under <- train(preterm ~ ., data = data_balanced_under,
                     method = "glm",
                     metric = metric_acc,
                     preProcess = c("center", "scale"),
                     trControl = ctrl_acc)
fit_glm_acc_under

summary(fit_glm_acc_under)

confusionMatrix.train(fit_glm_acc_under)
```

Neural network

```{r}
set.seed(1234)
fit_nnet_acc_under <- train(preterm ~ ., data = data_balanced_under,
                     method = "nnet",
                     metric = metric_acc,
                     preProcess = c("center", "scale"),
                     trControl = ctrl_acc,
                     trace = FALSE)
fit_nnet_acc_under

confusionMatrix.train(fit_nnet_acc_under)

```


```{r}
fit_nnet_acc_under$bestTune
```



```{r}
plot(fit_nnet_acc_under, main = "Neural Network best tune")
```


Random Forest

```{r}
set.seed(1234)
fit_rf_acc_under <- train(preterm ~ ., data = data_balanced_under,
                     method = "rf",
                     metric = metric_acc,
                     trControl = ctrl_acc,
                     trace = FALSE)
fit_rf_acc_under
```


```{r}
confusionMatrix.train(fit_rf_acc_under)
```

```{r}
plot(fit_rf_acc_under, main = "Random Forest best tune")
```

```{r}
fit_rf_acc_under$bestTune
```

XGBoost

```{r}
set.seed(1234)
fit_XG_acc_under <- train(preterm ~ ., data = data_balanced_under,
                      method = "xgbTree",
                      metric = metric_acc,
                      trControl = ctrl_acc)
fit_XG_acc_under
```

```{r}
confusionMatrix.train(fit_XG_acc_under)
```

```{r}
fit_XG_acc_under$bestTune
```

```{r}
plot(fit_XG_acc_under,main = "Extreme Gradient Boosting best tune")
```

```{r}
my_results_acc_under <- resamples(list(GLM = fit_glm_acc_under,
                             NNET = fit_nnet_acc_under,
                             RF = fit_rf_acc_under,
                             XGB = fit_XG_acc_under))
```

```{r}
dotplot(my_results_acc_under)
```

GLM Undersampling model performance

```{r}
glm_under_pred <- predict(fit_glm_acc_under,test,type = "prob")
glm_under_test <- as.factor(ifelse(glm_under_pred$preterm > 0.5,"preterm","term"))
```

```{r}
precision_glm_under   <-   posPredValue(glm_under_test,as.factor(test$preterm),positive   =
"preterm")
sensitivity_glm_under     <- sensitivity(glm_under_test,as.factor(test$preterm),positive =
"preterm")
specificity_glm_under   <-   specificity(glm_under_test,as.factor(test$preterm),positive   =
"preterm")
F1_glm_under <- (2 * precision_glm_under * sensitivity_glm_under) / (precision_glm_under +
sensitivity_glm_under)
```

Neural network Undersampling model performance

```{r}
```

```
nnet_under_pred <- predict(fit_nnet_acc_under,test,type = "prob")
nnet_under_test <- as.factor(ifelse(nnet_under_pred$preterm > 0.5,"preterm","term"))
```

```{r}
precision_nnet_under  <-  posPredValue(nnet_under_test,as.factor(test$preterm),positive  = "preterm")
sensitivity_nnet_under     <- sensitivity(nnet_under_test,as.factor(test$preterm),positive = "preterm")
specificity_nnet_under  <-  specificity(glm_under_test,as.factor(test$preterm),positive  = "preterm")
F1_nnet_under    <-    (2    *    precision_nnet_under    *    sensitivity_nnet_under)    / (precision_nnet_under + sensitivity_nnet_under)
```

Random Forest Undersampling model performance

```{r}
rf_under_pred <- predict(fit_rf_acc_under,test,type = "prob")
rf_under_test <- as.factor(ifelse(rf_under_pred$preterm > 0.5,"preterm","term"))
```

```{r}
precision_rf_under   <-   posPredValue(rf_under_test,as.factor(test$preterm),positive   = "preterm")
sensitivity_rf_under      <- sensitivity(rf_under_test,as.factor(test$preterm),positive = "preterm")
specificity_rf_under   <-   specificity(rf_under_test,as.factor(test$preterm),positive   = "preterm")
F1_rf_under <- (2 * precision_rf_under * sensitivity_rf_under) / (precision_rf_under + sensitivity_rf_under)
```

Extreme gradient bossted tree Undersampling model performance

```{r}
XG_under_pred <- predict(fit_XG_acc_under,test,type = "prob")
XG_under_test <- as.factor(ifelse(XG_under_pred$preterm > 0.5,"preterm","term"))
```

```{r}
precision_XG_under   <-   posPredValue(XG_under_test,as.factor(test$preterm),positive   = "preterm")
sensitivity_XG_under      <- sensitivity(XG_under_test,as.factor(test$preterm),positive = "preterm")
specificity_XG_under   <-   specificity(XG_under_test,as.factor(test$preterm),positive   = "preterm")
F1_XG_under <- (2 * precision_XG_under * sensitivity_XG_under) / (precision_XG_under + sensitivity_XG_under)
```

```{r}
model_compare_sensitivity <- data.frame(Model = c('GLM-UNDER',
                                        'NNet-UNDER',
                                        'RF-UNDER',
                                        'XGBoost-UNDER'),
                                Sensitivity = c(sensitivity_glm_under,
                                       sensitivity_nnet_under,
```

```
                                        sensitivity_rf_under,
                                        sensitivity_XG_under))

        ggplot(aes(x      =      reorder(Model,-Sensitivity),y      =      Sensitivity),data      =
model_compare_sensitivity) +
          geom_bar(stat = 'identity', fill = 'light blue') +
          ggtitle('Comparative Sensitivity of Models on Validation Data Using Undersampling') +
          xlab('Models')  +
          ylab('Sensitivity Measure')+
          geom_text(aes(label = round(Sensitivity,2))) + theme_bw() +
          theme(axis.text.x = element_text(angle = 40))
        ```



        ```{r}
        model_compare_sensitivity <- data.frame(Model = c('GLM-UNDER',
                                                'NNet-UNDER',
                                                'RF-UNDER',
                                                'XGBoost-UNDER'),
                                    Sensitivity = c(specificity_glm_under,
                                                specificity_nnet_under,
                                                specificity_rf_under,
                                                specificity_XG_under))

        ggplot(aes(x      =      reorder(Model,-Sensitivity),y      =      Sensitivity),data      =
model_compare_sensitivity) +
          geom_bar(stat = 'identity', fill = 'light blue') +
          ggtitle('Comparative Specificity of Models on Validation Data Using Undersampling') +
          xlab('Models')  +
          ylab('Sensitivity Measure')+
          geom_text(aes(label = round(Sensitivity,2))) + theme_bw() +
          theme(axis.text.x = element_text(angle = 40))
        ```


        ## Confusion matrix on test set

        ### GLM

        ```{r}
        confusionMatrix(data = factor(glm_under_test), reference = factor(test$preterm))
        ```

        ### NNET

        ```{r}
        confusionMatrix(data = factor(nnet_under_test), reference = factor(test$preterm))
        ```


        ### RF

        ```{r}
        confusionMatrix(data = factor(rf_under_test), reference = factor(test$preterm))
        ```

        ### XG

        ```{r}
        confusionMatrix(data = factor(XG_under_test), reference = factor(test$preterm))
        ```

        ## ROC

        ### GLM


        ```{r}
```

```
roc_glm_point <- roc(response = test$preterm,factor(glm_under_test, ordered = TRUE))

plot(roc_glm_point, legacy.axe = TRUE, plot = TRUE, print.auc = TRUE, col = 1, print.auc.y=0.5, main = "GLM ROC")
```

```{r}
roc_glm <- roc(response = test$preterm,factor(glm_under_pred$preterm, ordered = TRUE))

plot(roc_glm, legacy.axe = TRUE, plot = TRUE, print.auc = TRUE, col = 1, print.auc.y=0.5, main = "GLM ROC")
```

NNET

```{r}
roc_nnet_point <- roc(response = test$preterm,factor(nnet_under_test, ordered = TRUE))

plot(roc_nnet_point, legacy.axe = TRUE, plot = TRUE, print.auc = TRUE, col = 1, print.auc.y=0.5, main = "NNET ROC")
```

```{r}
roc_nnet <- roc(response = test$preterm,factor(nnet_under_pred$preterm, ordered = TRUE))

plot(roc_nnet, legacy.axe = TRUE, plot = TRUE, print.auc = TRUE, col = 1, print.auc.y=0.5, main = "NNET ROC")
```

RF

```{r}
roc_rf_point <- roc(response = test$preterm,factor(rf_under_test, ordered = TRUE))

plot(roc_rf_point, legacy.axe = TRUE, plot = TRUE, print.auc = TRUE, col = 1, print.auc.y=0.5, main = "RF ROC")
```

```{r}
roc_rf <- roc(response = test$preterm,factor(rf_under_pred$preterm, ordered = TRUE))

plot(roc_rf, legacy.axe = TRUE, plot = TRUE, print.auc = TRUE, col = 1, print.auc.y=0.5, main = "RF ROC")
```

XG

```{r}
roc_XG_point <- roc(response = test$preterm,factor(XG_under_test, ordered = TRUE))

plot(roc_XG_point, legacy.axe = TRUE, plot = TRUE, print.auc = TRUE, col = 1, print.auc.y=0.5, main = "XG Boost ROC")
```

```{r}
roc_XG <- roc(response = test$preterm,factor(XG_under_pred$preterm, ordered = TRUE))

plot(roc_XG, legacy.axe = TRUE, plot = TRUE, print.auc = TRUE, col = 1, print.auc.y=0.5, main = "XG Boost ROC")
```

```{r}
auc_glm <- round(roc_glm$auc, 3)
auc_nnet <- round(roc_nnet$auc, 3)
auc_rf <- round(roc_rf$auc, 3)
auc_XG <- round(roc_XG$auc, 3)
```

```
ggroc(list(roc_glm, roc_nnet, roc_rf, roc_XG), size = .8, aes = c("color")) +
  scale_color_discrete(labels=c("GLM","NNET","RF", "XG")) +
  ggtitle('AUC Comparison') +
  labs(color = c("Models")) +
  theme_bw()
```

```{r}
auc_glm_point <- round(roc_glm_point$auc, 3)
auc_nnet_point <- round(roc_nnet_point$auc, 3)
auc_rf_point <- round(roc_rf_point$auc, 3)
auc_XG_point <- round(roc_XG_point$auc, 3)
```

```
ggroc(list(roc_glm_point, roc_nnet_point, roc_rf_point, roc_XG_point), size = .8, aes =
c("color")) +
  scale_color_discrete(labels=c("GLM","NNET","RF", "XG")) +
  ggtitle('AUC Comparison 0.5 cutpoint') +
  labs(color = c("Models")) +
  theme_bw()
```

Varaible importance

GLM

```{r}
plot(varImp(fit_glm_acc_under), main = "GLM Variable importance", top = 20)
```

NNET

```{r}
plot(varImp(fit_nnet_acc_under), main = "NNET Variable importance", top = 20)
```

RF

```{r}
plot(varImp(fit_rf_acc_under), main = "RF Variable importance", top = 20)
```

XG Boost

```{r}
plot(varImp(fit_XG_acc_under), main = "XG Boost Variable importance", top = 20)
```

Graphs for presentation

```{r}
reduced_df4 %>%
  ggplot(mapping = aes(x = NOPV)) +
  geom_bar(mapping = aes(fill = preterm),
           position = 'dodge') +
  scale_fill_manual(values = c("#736F6E", "#153E7E")) +
  ggtitle("Number of prenatal visits by preterm status") +
  theme_bw()
```

```{r}
reduced_df4 %>%
  ggplot(mapping = aes(x = NOPV)) +
  geom_bar(mapping = aes(fill = preterm), position = position_fill(reverse = TRUE)) +
  #scale_fill_brewer(palette = "Blues") +
  scale_fill_manual(values = c("#736F6E", "#153E7E")) +
  ggtitle("Proportion of preterm births by prenatal visits") +
  theme_bw()
```

Bibliography

[1] Al-Alaiyan, Saleh. “Call to establish a national lower limit of viability.” Annals of Saudi medicine vol. 28,1 (2008): 1-3. doi:10.5144/0256-4947.2008.1

[2] Evaluation and Comparison of Machine Learning Techniques for Rapid QSTS Simulations - Scientific Figure on ResearchGate. Available from:

[3] Evaluation and Comparison of Machine Learning Techniques for Rapid QSTS Simulations - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/Random-Forest-visualization_fig11_326560291 [accessed 28 Apr, 2022]

[4] Bensley, JG, De Matteo, R, Harding, R, Black, MJ. The effects of preterm birth and its antecedents on the cardiovascular system. Acta Obstet Gynecol Scand 2016; 95: 652– 663.

[5] Miranda, M., Edwards, S., Chang, H. et al. Proximity to roadways and pregnancy outcomes. J Expo Sci Environ Epidemiol 23, 32–38 (2013). https://doi.org/10.1038/jes.2012.78

[6] Ritz, Beate1 2; Yu, Fei3; Chapa, Guadalupe4; Fruin, Scott4 5 Effect of Air Pollution on Preterm Birth Among Children Born in Southern California Between 1989 and 1993, Epidemiology: September 2000 - Volume 11 - Issue 5 - p 502-511

[7] WHO Regional Office for Europe. Review of evidence on health aspects of air pollution – REVIHAAP Project: Technical Report [Internet]. Copenhagen: WHO Regional Office for Europe; 2013. C, Proximity to roads, NO2, other air pollutants and their mixtures. Available from: https://www.ncbi.nlm.nih.gov/books/NBK361807/

[8] Beam, A.L., Fried, I., Palmer, N. et al. Estimates of healthcare spending for preterm and low-birthweight infants in a commercially insured population: 2008–2016. J Perinatol 40, 1091–1099 (2020). https://doi.org/10.1038/s41372-020-0635-z

[9] Mishra, Satwik. "Handling imbalanced data: SMOTE vs. random undersampling." Int. Res. J. Eng. Technol 4.8 (2017): 317-320.

[10] Banerjee M, Reynolds E, Andersson HB, Nallamothu BK. Tree-Based Analysis [published correction appears in Circ Cardiovasc Qual Outcomes. 2019 Jun;12(6):e000056]. Circ Cardiovasc Qual Outcomes. 2019;12(5):e004879. doi:10.1161/CIRCOUTCOMES.118.004879