

# Predicting County-Level Heart Disease Mortality in the United States

*Charlotte Abrams, Laura Cosgrove, Alyssa Vanderbeek*

*7 April 2019*

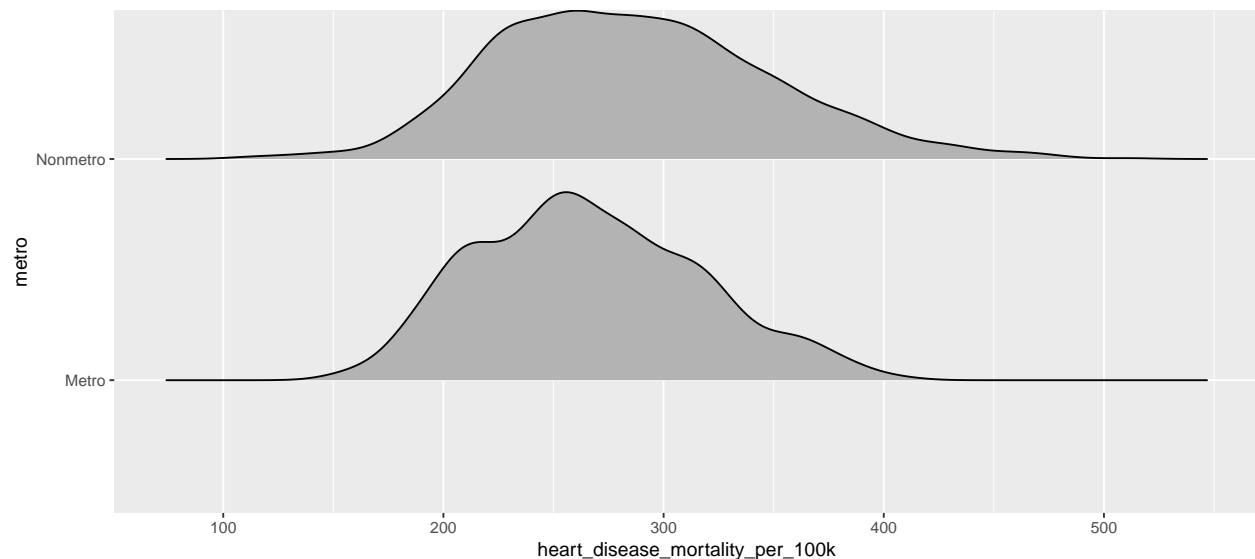
## Introduction

### Background

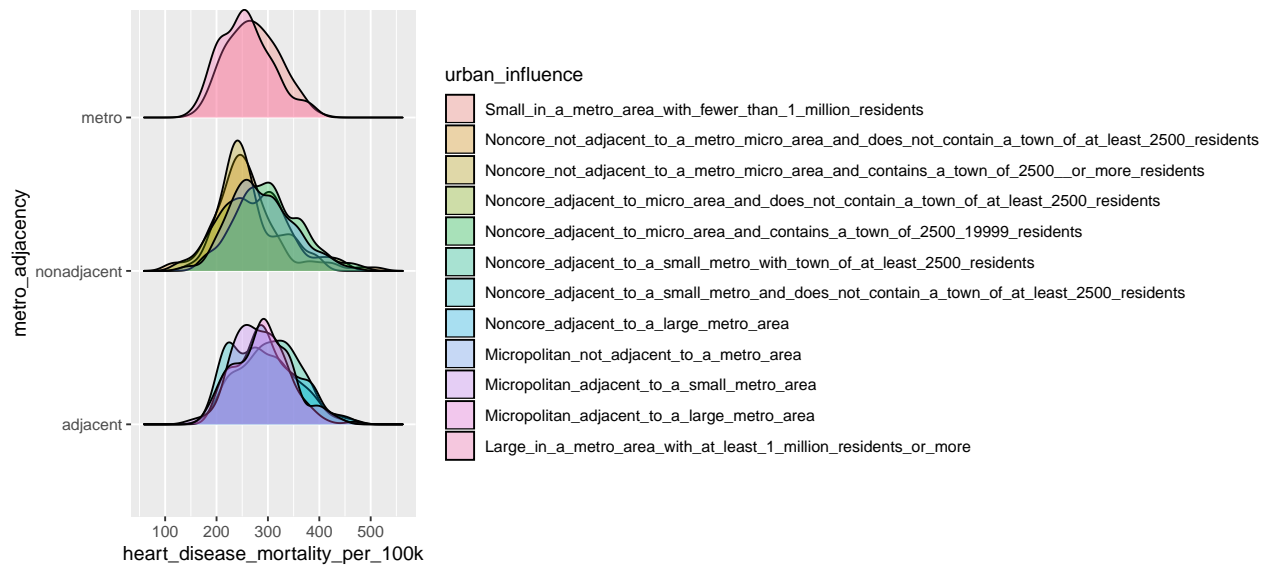
Heart disease remains one of the leading causes of death in adults in the US. Understanding risk factors for diseases of this kind is an important task in working towards reducing the number of lives lost. There is obvious benefit in doing this at an individual level, examining personal lifestyle, genetic profile, and family history. But there may be important environmental components that have predictive value in assessing risk for heart disease mortality. We examine economic, health, and demographic data for thousands of counties across the US. This data is synthesized from several sources, including the USDA Economic Research Service, Bureau of Labor Statistics, US Census, Behavioral Risk Factor Surveillance System, the CDC, and others. Our goal in this project is to most effectively predict the county-level heart disease mortality rate per 100,000 persons. We build 6 predictive models (stepwise linear regression, Lasso, Ridge, PCR, GAM, and MARS), and compare them on their predictive capacity quantified by the root mean squared error (RMSE).

### Exploratory Data Analysis (Laura)

## Picking joint bandwidth of 11.6



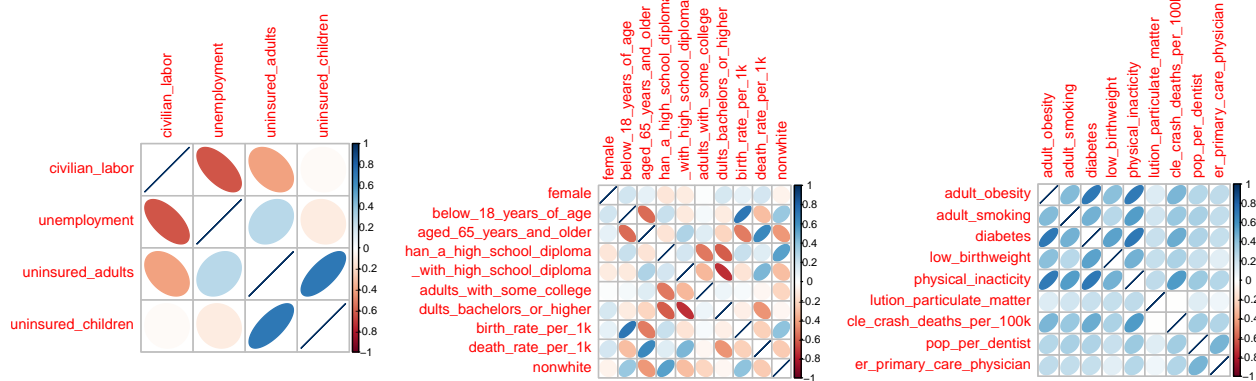
## Picking joint bandwidth of 16.7



The global mean of heart disease mortality per 100,000 residents is 279.37, with standard deviation 58.95. It is approximately normally distributed.

The distribution of heart disease mortality differs among levels of factor variables, but is broadly similar. Many of the factor variables included in this dataset contain similar information at different levels of interaction, or hierarchical division. For example, as shown in the second density plot, metro adjacency (a variable created from the original RUCCS hierarchical coding) is a less fine-grained look at urban influence, and urban influence may then function as an interaction between metro adjacency and population (in our dataset, `pure_population` is also created from the original RUCCS hierarchical coding in pre-processing). To get at these different specificities and as a “proxy” for including pseudo-interaction terms, we include all these factor variables for our initial model fit.

## Correlations and Collinearity



We investigated possible multicollinearity both between and within categories.

Health statistics related to healthy behaviors are highly intercorrelated: obesity, smoking, diabetes, low birthweight babies, excessive drinking, and physical inactivity. Less so are more environment- or “acts of God”-related health behaviors: particulate matter, homicides, motor vehicle crashes, and rates of dentists and doctors (though the last two are highly correlated).

For demography, there’s some categorical variables masquerading as separate predictors, leading to high intercorrelation within those categories: percentages of residents who are a given race accounts for 5 variables;

Table 1: Highly Correlated Predictors Between Categories

variable	with	correlation
demo_pct_adults_less_than_a_high_school_diploma	econ_pct_civilian_labor	-0.63
demo_pct_adults_less_than_a_high_school_diploma	econ_pct_uninsured_adults	0.67
demo_pct_adults_bachelors_or_higher	health_pct_adult_obesity	-0.63
demo_pct_adults_bachelors_or_higher	health_pct_physical_inactivity	-0.67
demo_pct_adults_with_high_school_diploma	health_pct_physical_inactivity	0.60

Table 2: Correlation with Outcome

variable	correlation
heart_disease_mortality_per_100k	1.00
health_pct_physical_inactivity	0.73
health_pct_diabetes	0.66
health_pct_adult_obesity	0.64
demo_pct_adults_bachelors_or_higher	-0.59
demo_pct_adults_less_than_a_high_school_diploma	0.58
health_pct_low_birthweight	0.53
demo_death_rate_per_1k	0.52
econ_pct_civilian_labor	-0.52
health_pct_adult_smoking	0.52
health_motor_vehicle_crash_deaths_per_100k	0.51
demo_pct_adults_with_high_school_diploma	0.48
econ_pct_uninsured_adults	0.40
econ_pct_unemployment	0.39
demo_pct_adults_with_some_college	-0.33
health_pop_per_dentist	0.33
health_pop_per_primary_care_physician	0.22
health_air_pollution_particulate_matter	0.17
demo_birth_rate_per_1k	0.15
demo_pct_nonwhite	0.12
demo_pct_below_18_years_of_age	0.10
demo_pct_female	0.08
econ_pct_uninsured_children	0.03
demo_pct_aged_65_years_and_older	0.02

age-related bins (less than 18, greater than 65) account for 2 variables; birth and death rate; and percentages of residents who complete a given level of education account for 4 variables.

For economics, unsurprisingly, the percent of adults and the percent of children without health insurance are highly correlated, as well as percent civilian labor and unemployment rate.

Between categories, education-related variables are highly correlated with % civilian labor, % uninsured adults, and % physical inactivity.

Predictors strongly associated with heart disease mortality are % physical inactivity, % diabetes, % adult obesity, education-related variables, % low birthweight, and overall death rate per 1,000. Many of the variables most strongly-associated with the outcome are associated with one another.

## Data Methods

**\*\*note:** transform test data by centering and scaling with TRAINING means and standard deviations. Does predict do this automatically when using `preProcess` in caret? – done

can consider imputation: say 10%, one possible solution. use the training set to build. Look at the `preProcess` function, can use `knn`. – done. Still dropped two columns because they had 90% missing data; imputation not reliable

Check/detect near-zero variance predictors: to decrease computational time and complexity. – done. A few of the categories for urban influence came back as having near-zero variance. These were examined as dummy variables; 2000 rows divided across 12 levels means that these are likely to be “rare”. We determined that we ought to leave them in the model since they are part of the larger variable “urban\_influence”.

## Linear Models

### Stepwise Selection (Alyssa)

We first fit a stepwise linear regression model.

### Lasso (Charlotte)

### Ridge (Laura)

### PCR (Charlotte)

## Nonlinear Models

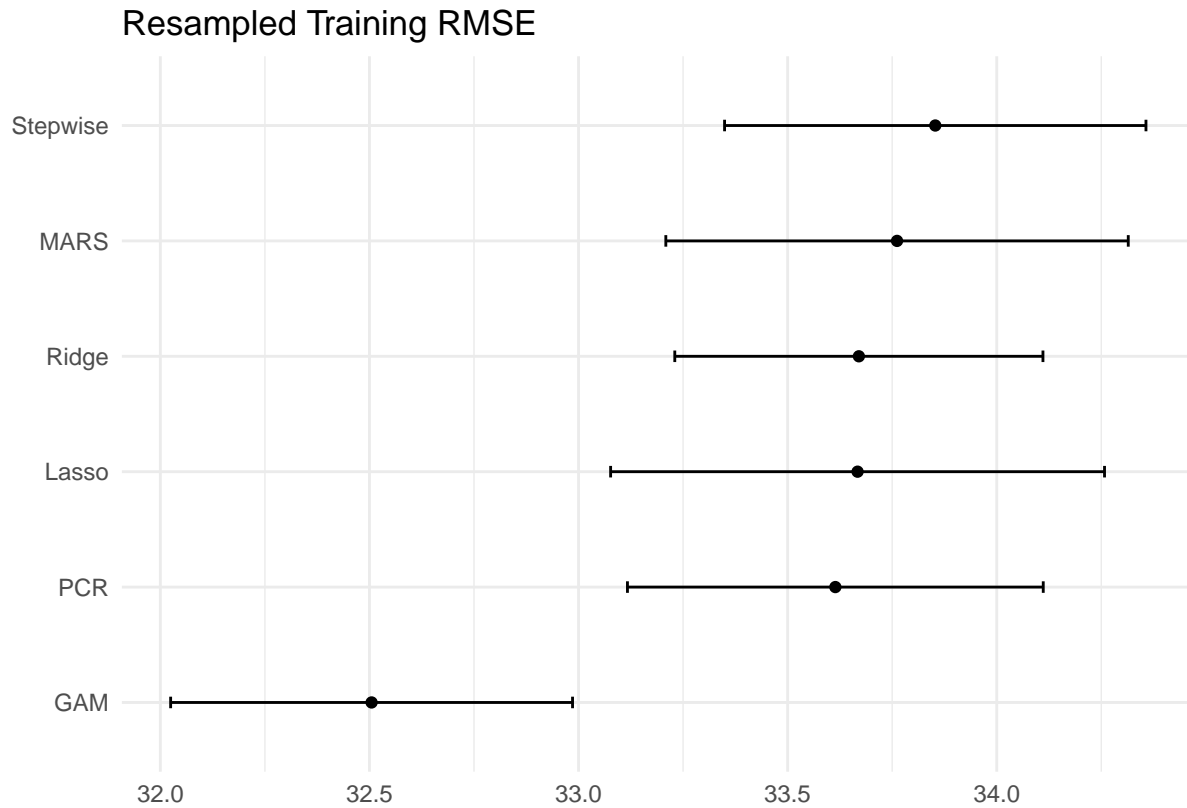
### GAM (Alyssa)

### MARS (Laura)

Note that we chose the most parsimonious model.

## Model Comparison

### Training RMSE



The more flexible model

### Test RMSE (Alyssa)

Table 3: Predictive RMSE on training and test data

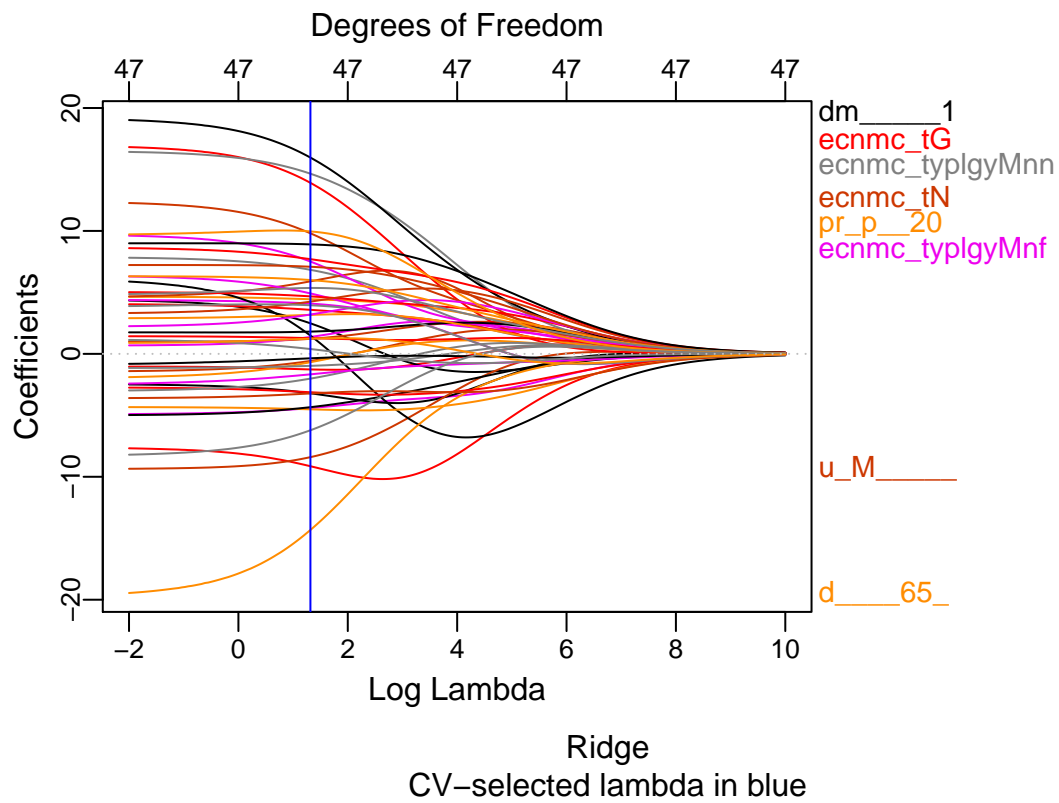
	Train RMSE	Test RMSE
Stepwise	33.85305	32.90754
Ridge	33.67035	33.04168
Lasso	33.66725	32.89532
PCR	33.61412	32.86776
GAM	32.50513	30.67289
MARS	33.76163	30.83814

Table 1 presents the RMSE of the predicted heart disease mortality rate for all models on both the training and testing datasets. Although GAM appeared to have the lowest RMSE when using the training data, MARS has similar performance to the GAM model in the test data.

### Interpretations (Laura)

*Coefficient Shrinkage: Lasso and Ridge*

```
## [1] 48 1
```



```
## integer(0)
```

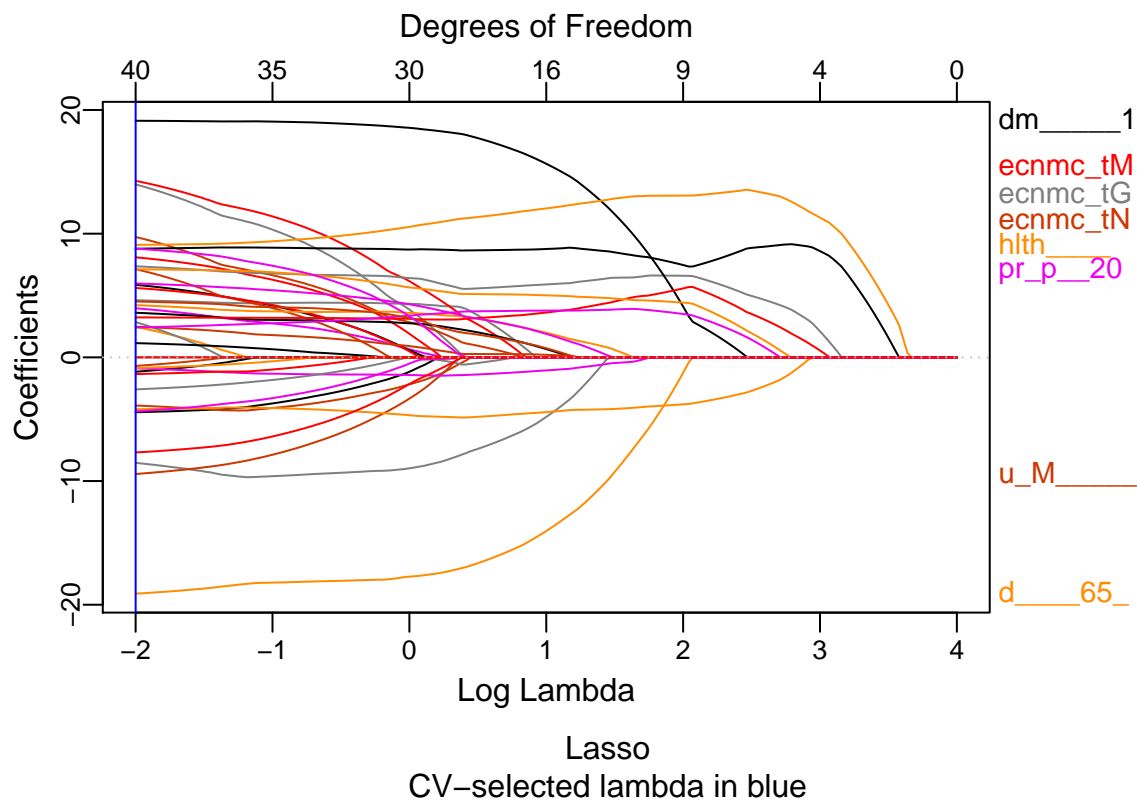


Table 4: Top Absolute-Values Coefficients, Ridge

row	column	value
(Intercept)	1	268.695291
demo_death_rate_per_1k	1	15.898754
economic_typologyMining	1	14.612043
economic_typologyGovernment	1	13.841795
pure_populationmore_than_20k	1	9.947095
economic_typologyNonspecialized	1	9.753701
health_pct_physical_inactivity	1	8.899230
health_pct_diabetes	1	7.686107
economic_typologyManufacturing	1	7.460263
health_pct_low_birthweight	1	7.079183

Table 5: Top Absolute-Valued Coefficients, Lasso

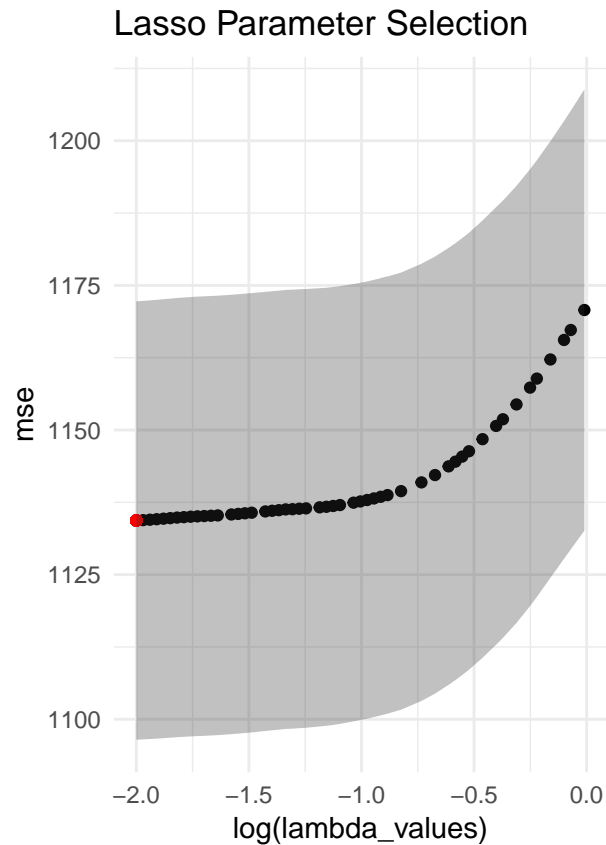
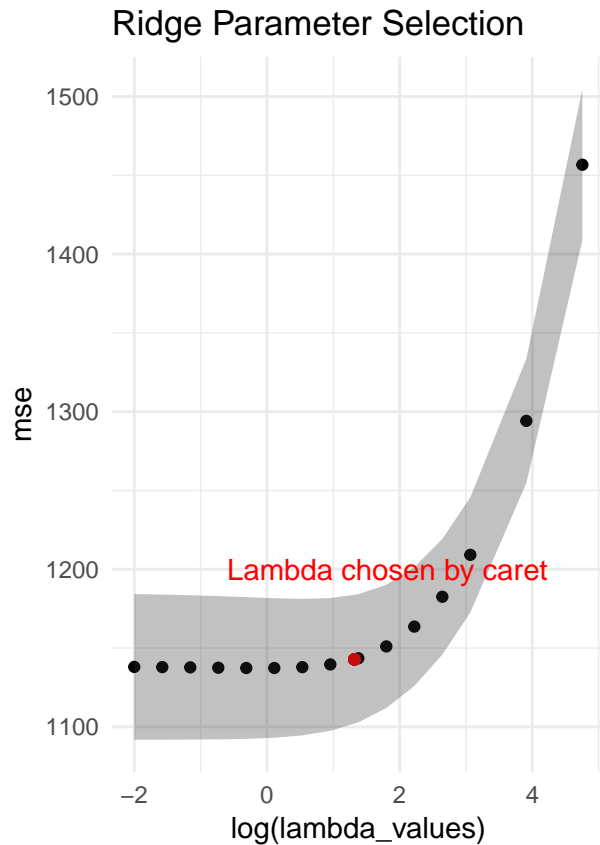
row	column	value
(Intercept)	1	269.100366
demo_death_rate_per_1k	1	19.125559
economic_typologyMining	1	14.248826
economic_typologyGovernment	1	13.960475
economic_typologyNonspecialized	1	9.716805
health_pct_physical_inactivity	1	9.097034
pure_populationmore_than_20k	1	8.797225
health_pct_diabetes	1	8.781068
urban_influenceLarge in a metro area with at least 1 million residents or more	1	8.028903
demo_pct_adults_less_than_a_high_school_diploma	1	7.367846

```
## integer(0)
## Selecting by value
## Selecting by value
```

For better visualization, a `glmnet` model was fit and the lambda value selected by `caret` was plotted.

Overall, this modeling problem was perhaps not the best fit for lasso and ridge. Neither of the methods shrunk many coefficients in such a way that completely removed them from the model and improved the RMSE. The best visualization of the effect of shrinkage can be seen in the ridge coefficient plot, where it can be seen that the vast majority of coefficients shrunk equally. Lasso's more stringent shrinking power did not improve MSE for this problem.

If lasso and ridge were more appropriate for the problem, parameter selection might have been more obvious. As it is, as shown below, the standard error of the mean cross-validated error is quite wide.

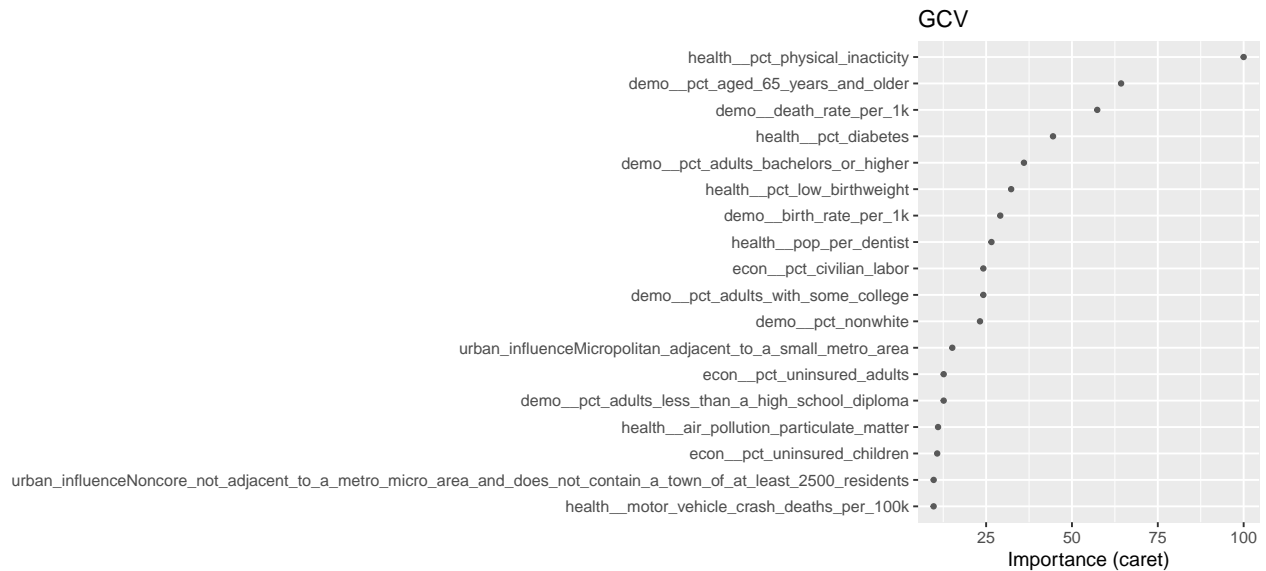


#### *Investigation of MARS*

The minimum generalized cross-validation error was achieved for a total of 25 features, including a two-way interaction term. Variable importance is assessed by tracking GCV for each predictor and accumulating the reduction in GCV when each predictor's feature is added to the model – the total reduction is used as the measure of variable importance. If a predictor was never used in any MARS basis function, it has an importance value of zero; 18 predictors were used in a MARS basis function.

MARS retained interactions between many sets of health-related variables, which one might expect given collinearity (and relatedness) between the within-category variable sets. Interaction terms were also present across categories.





```
## Warning: 'tidy.numeric' is deprecated.
```

```
## See help("Deprecated")
```

```
## # A tibble: 25 x 2
```

```
##   names
```

```
##   <chr>
```

```
## 1 (Intercept)
```

```
## 2 h(pct_physical_inactivity- -1.58282)
```

```
## 3 h(pct_aged_65_years_and_older- -0.609182) * h(pct_physical_ina~
```

```
## 4 h(pct_adults_bachelors_or_higher- -0.090806)
```

```
## 5 h(-0.090806-pct_adults_bachelors_or_higher)
```

```
## 6 h(pct_physical_inactivity- -1.58282) * h(0.103008-pct_nonwhite)
```

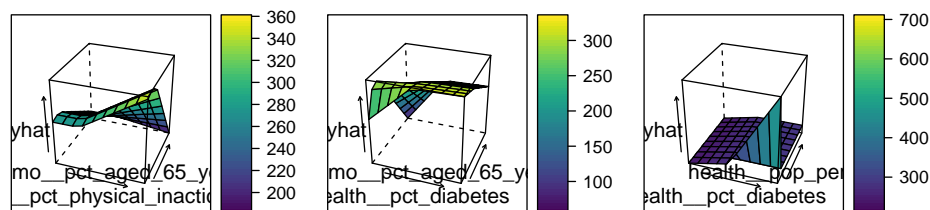
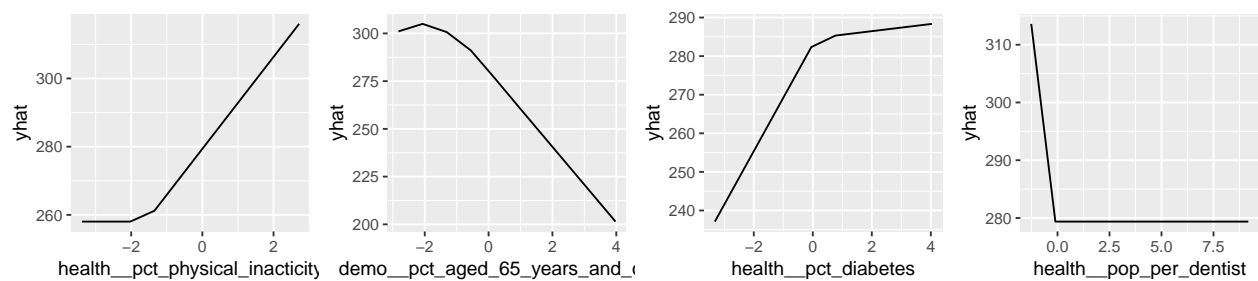
```
## 7 h(pct_aged_65_years_and_older- -1.78344) * h(0.122171-pct_diab~
```

```
## 8 h(-1.78344-pct_aged_65_years_and_older) * h(0.122171-pct_diabe~
```

```
## 9 h(pct_diabetes-0.122171) * h(-0.965113-pop_per_dentist)
```

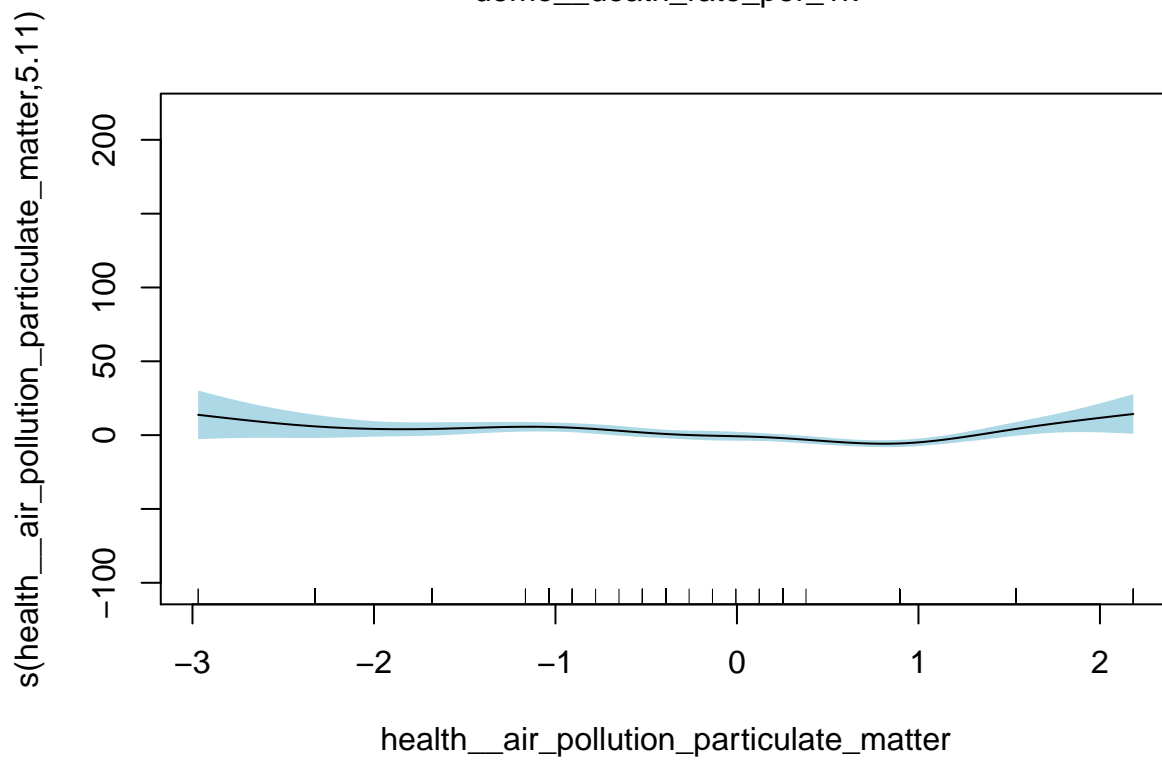
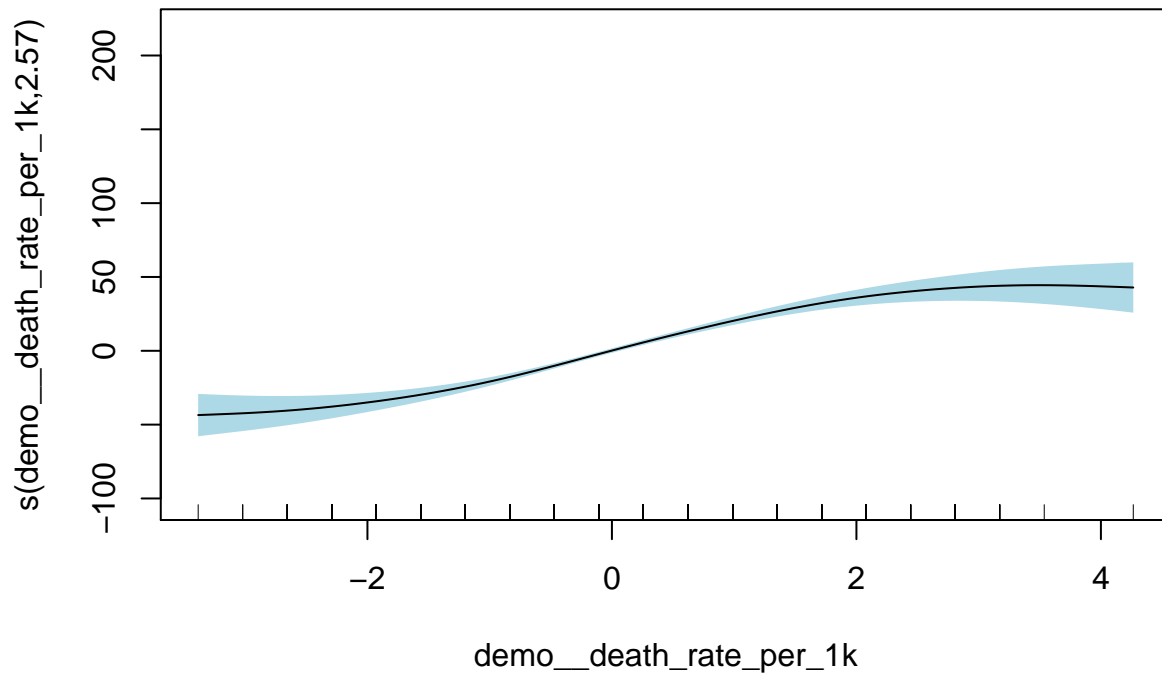
```
## 10 h(pct_low_birthweight- -1.11807)
```

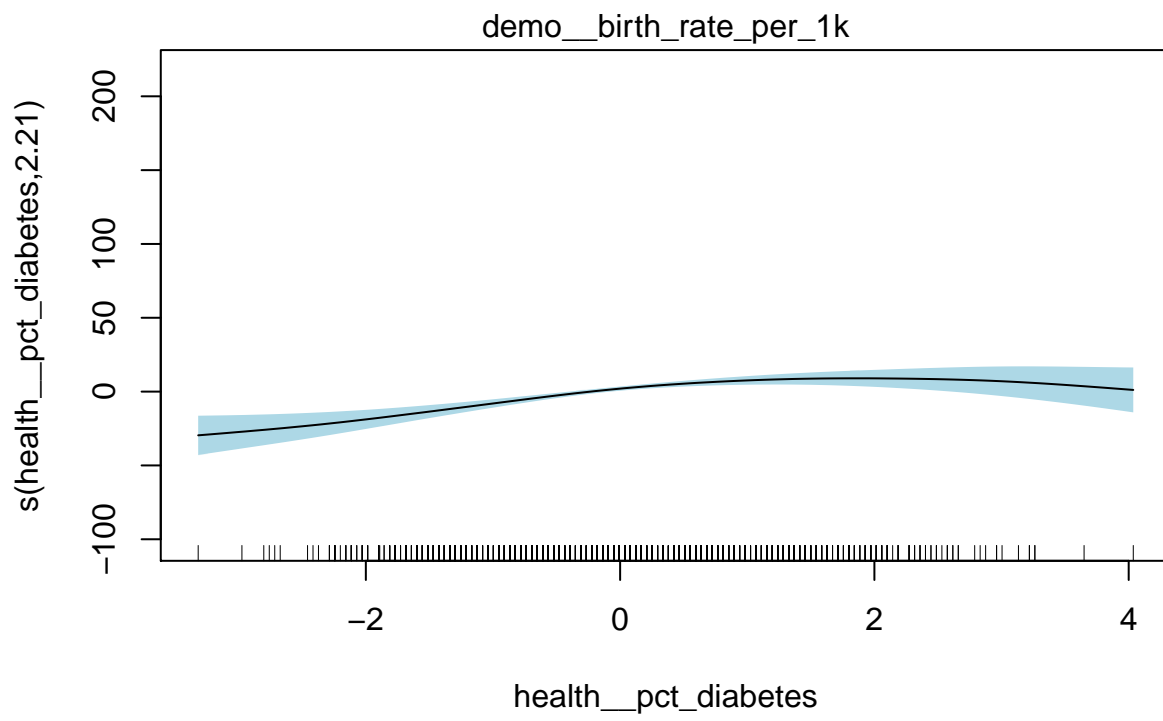
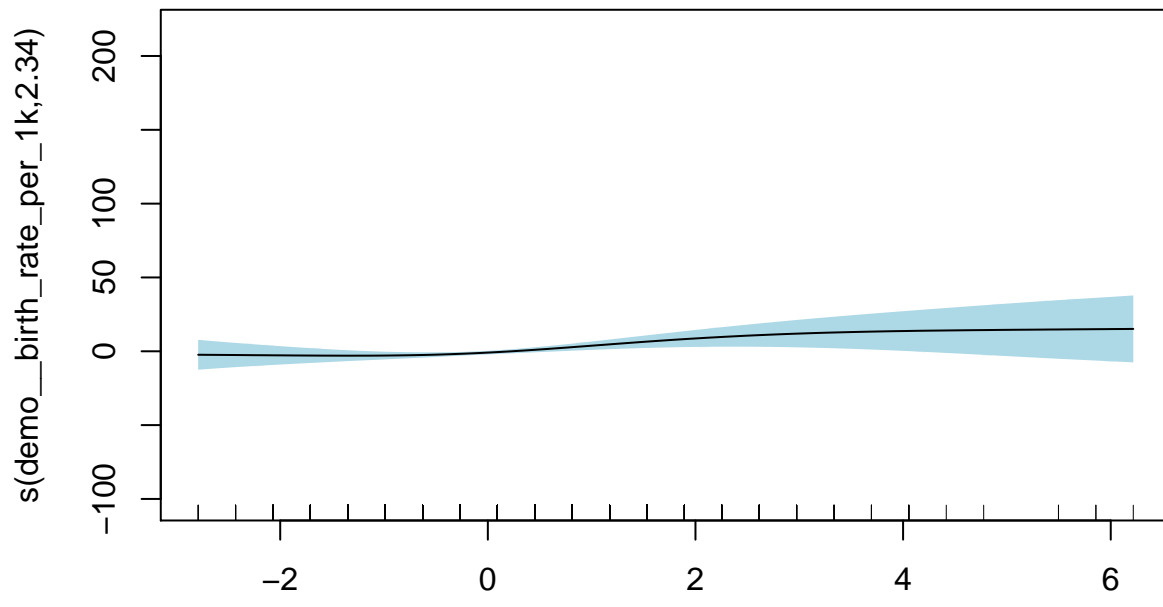
```
## # ... with 15 more rows
```

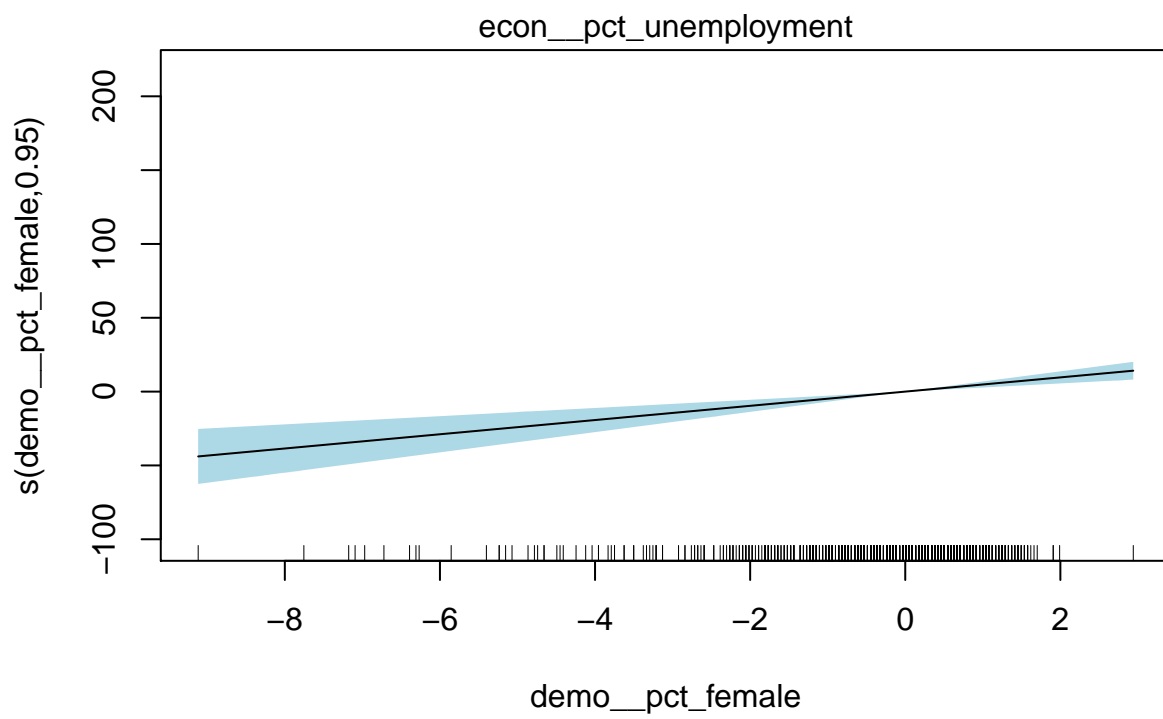
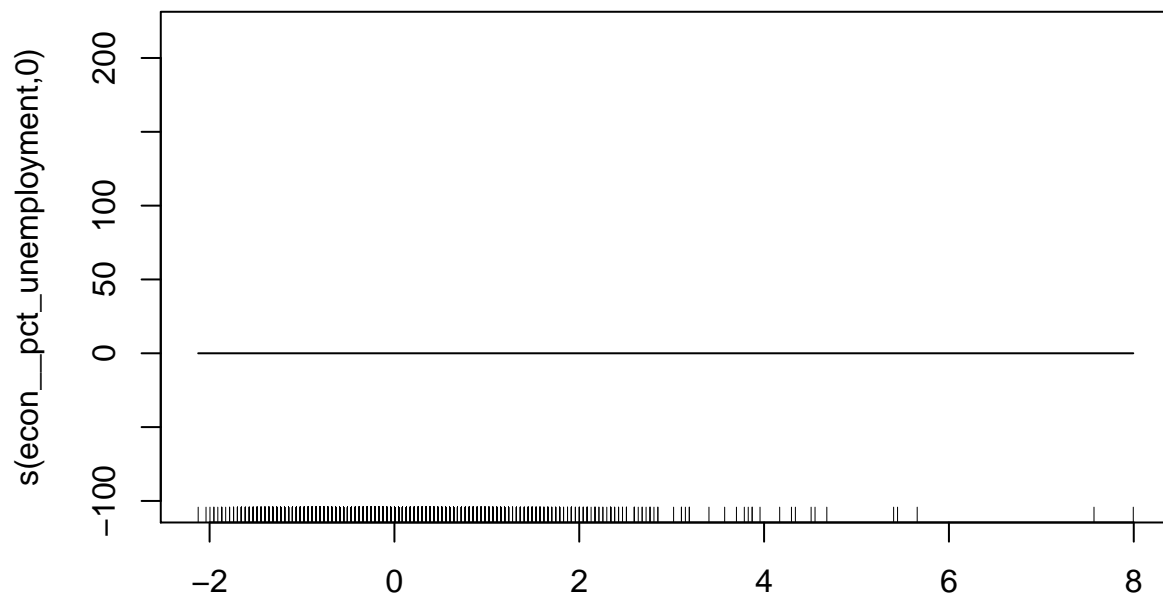


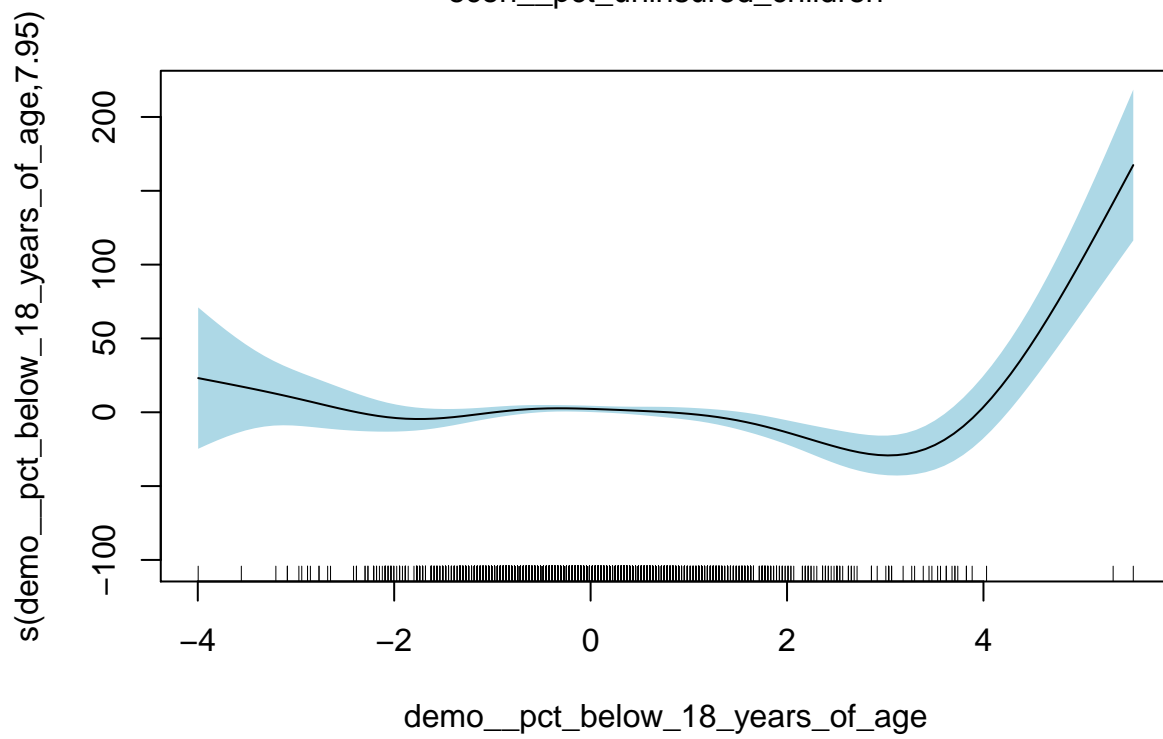
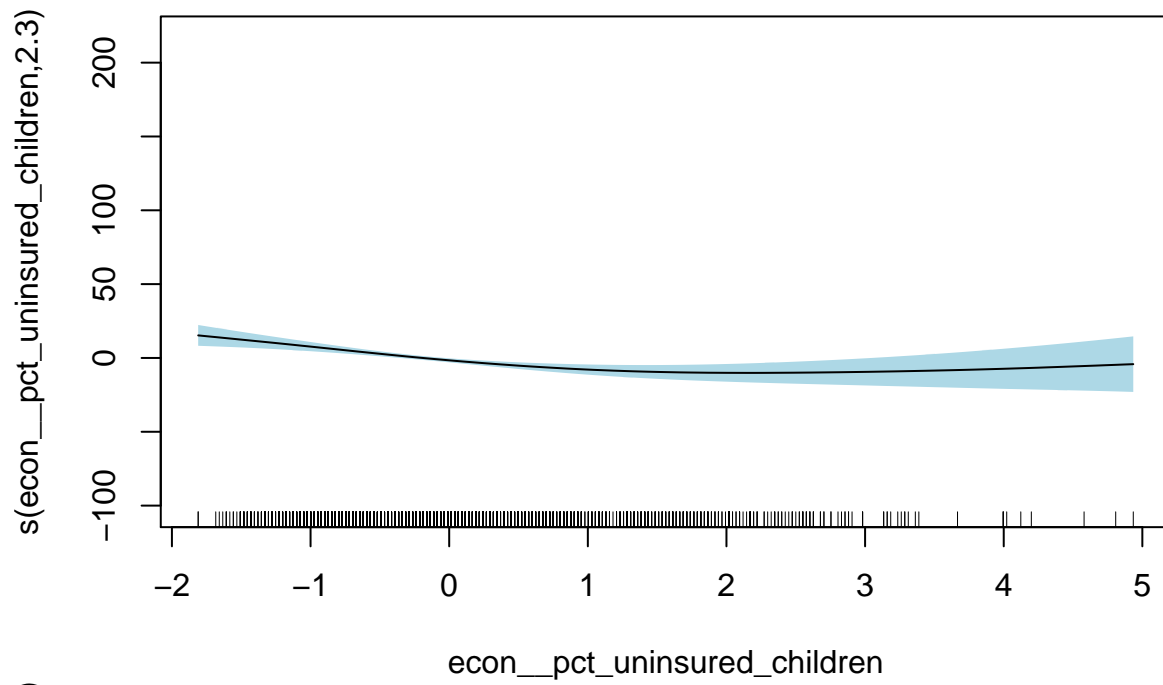
Investigation of GAM

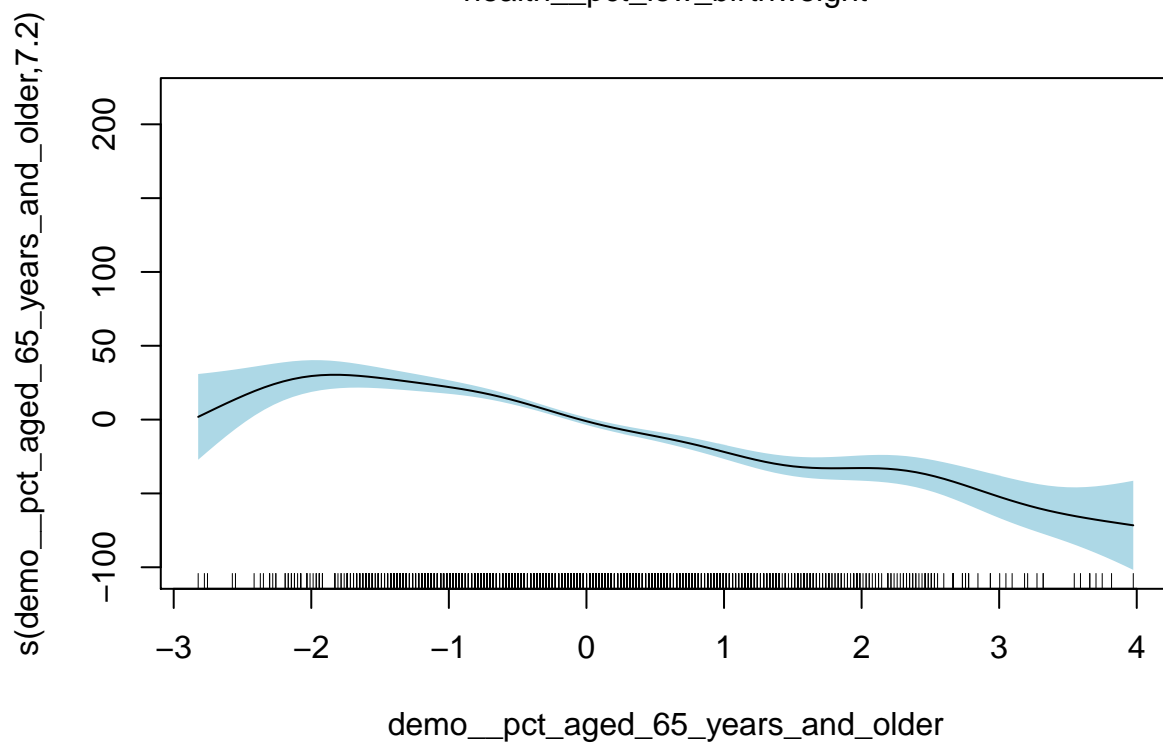
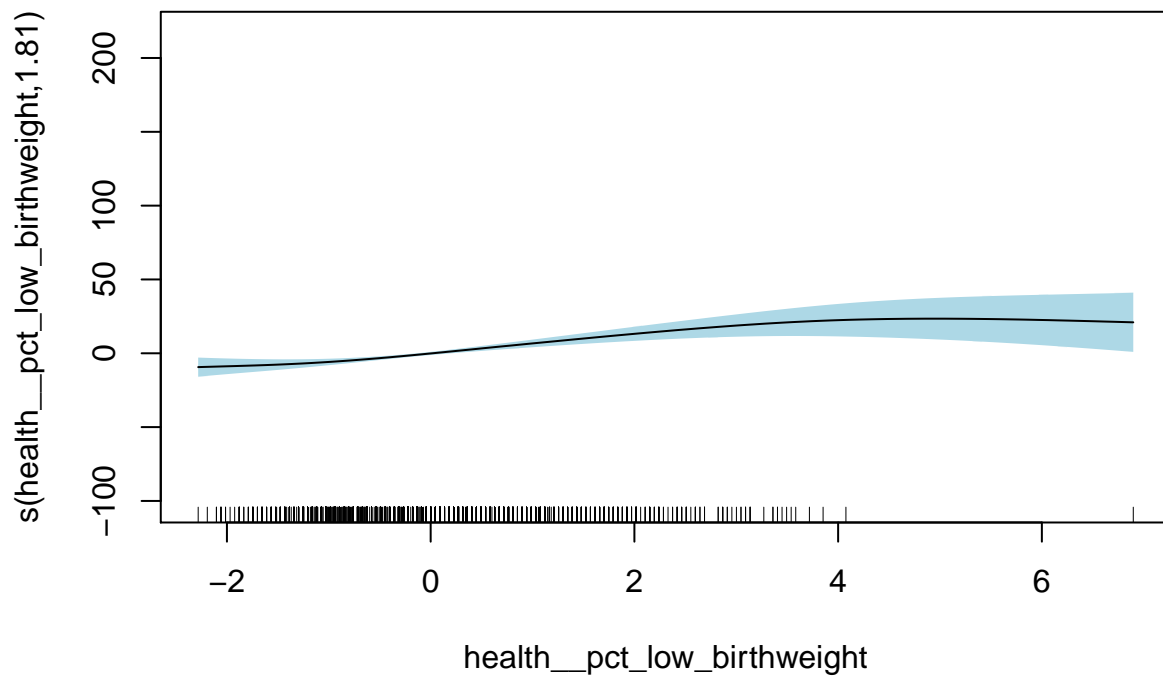
```
plot(gam_fit$finalModel, shade = TRUE, shade.col = "#add8e6")
```

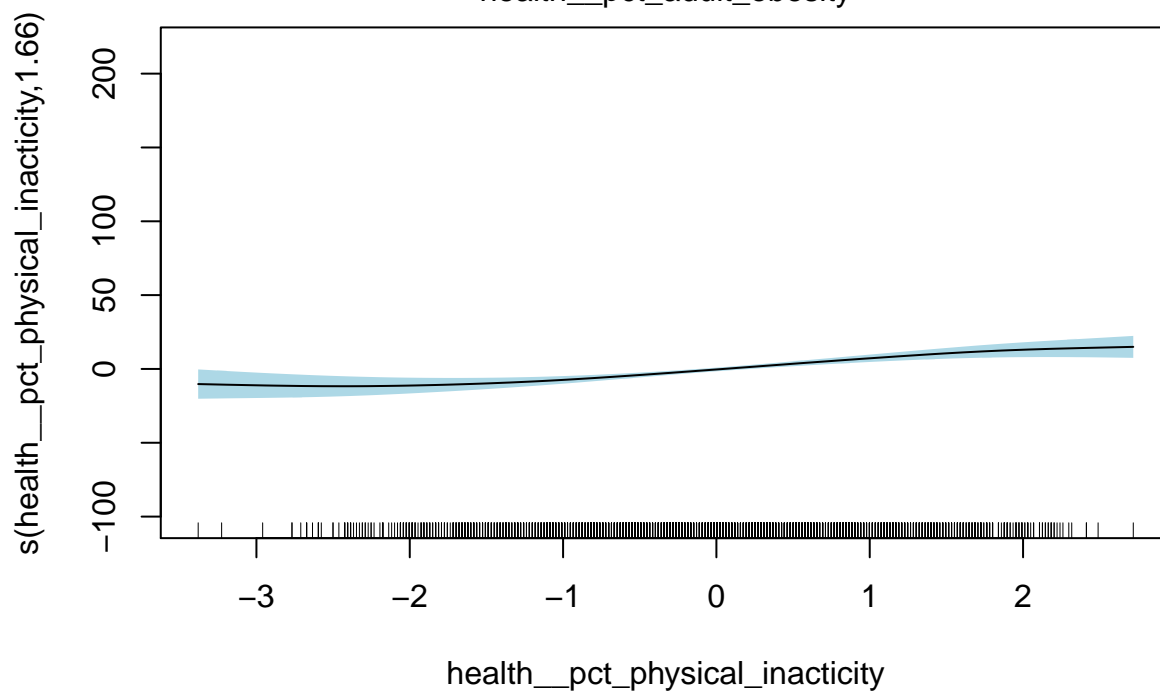
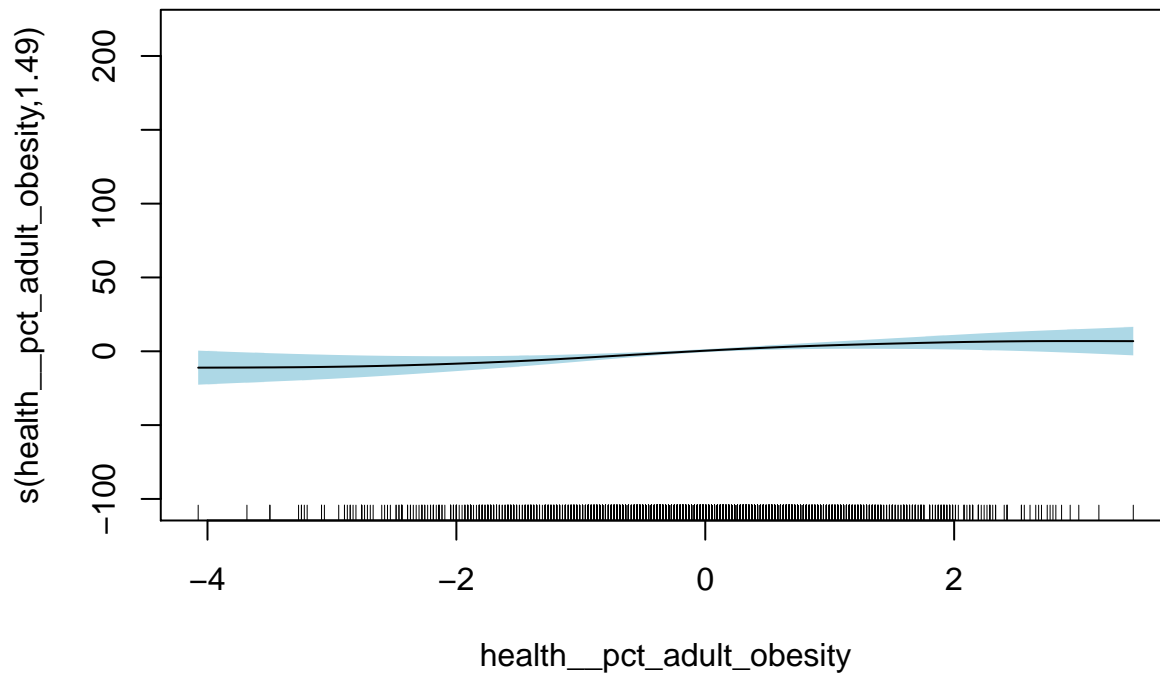


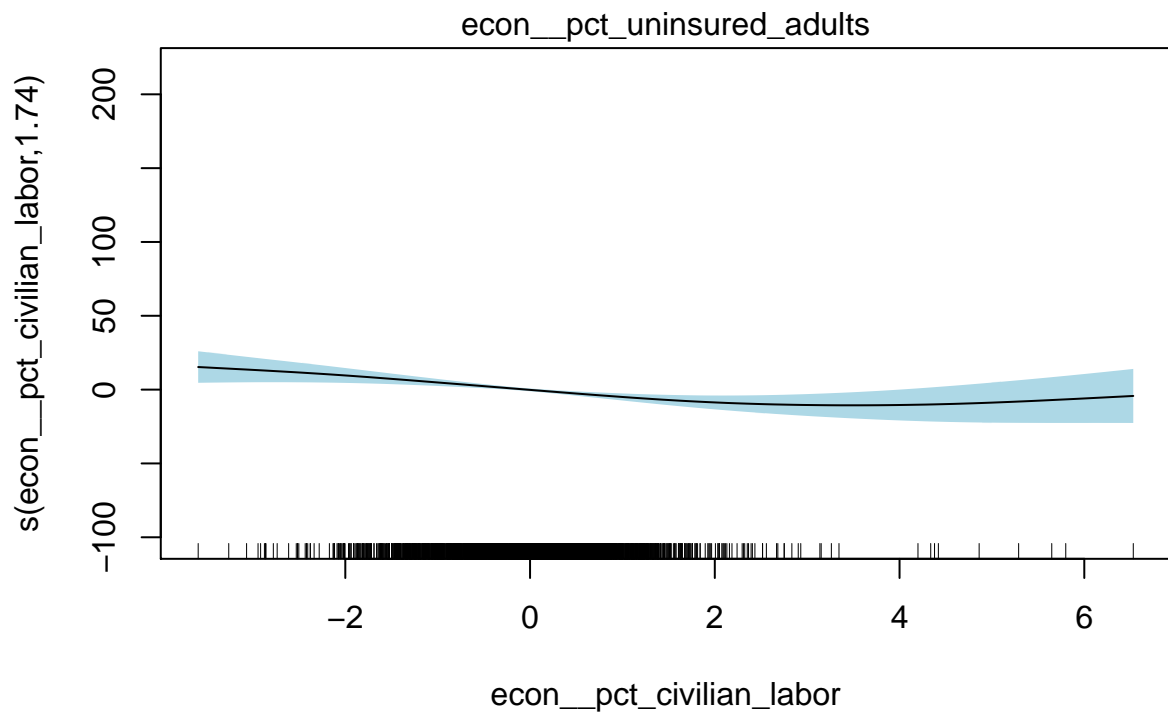
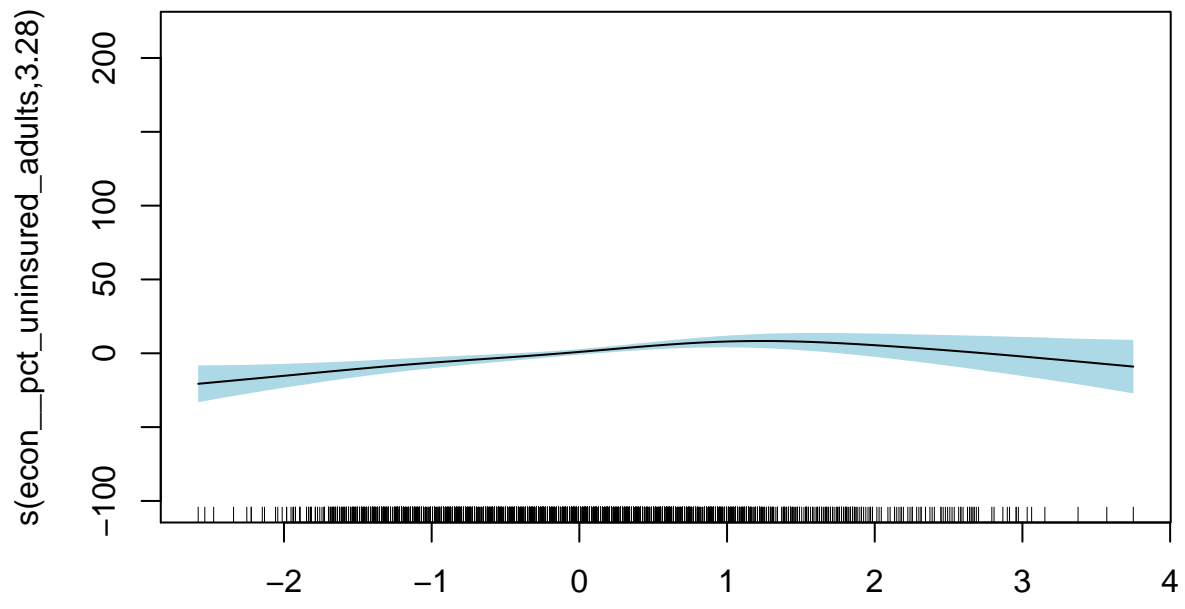




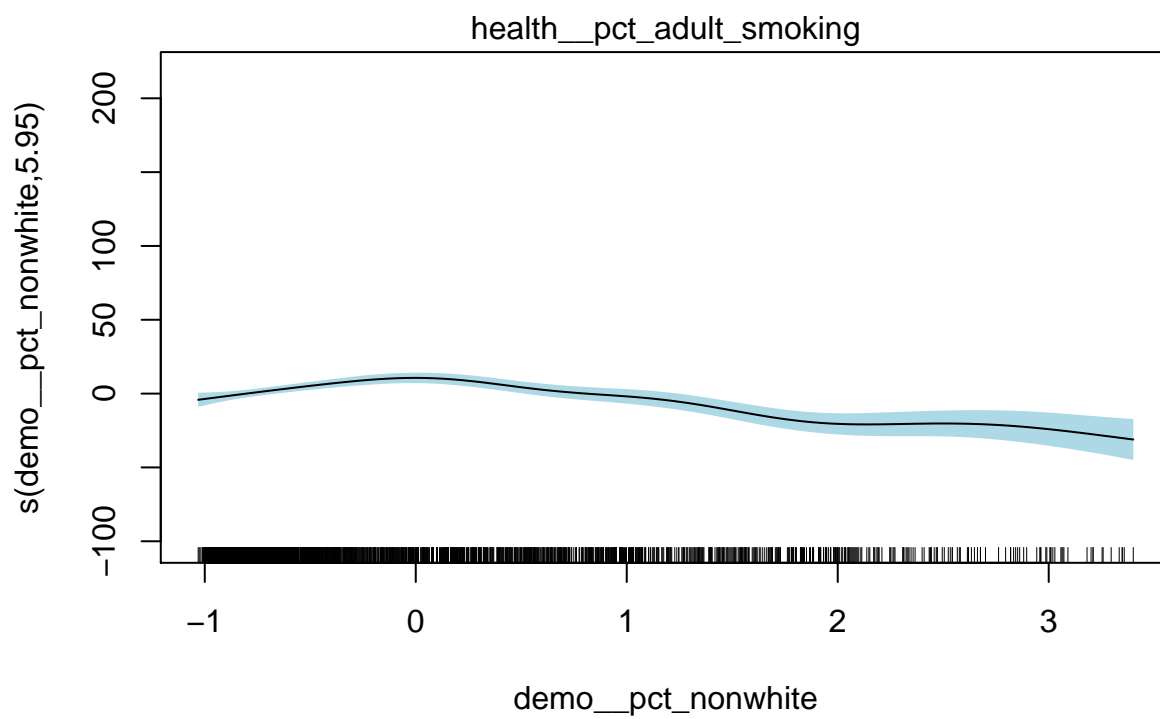
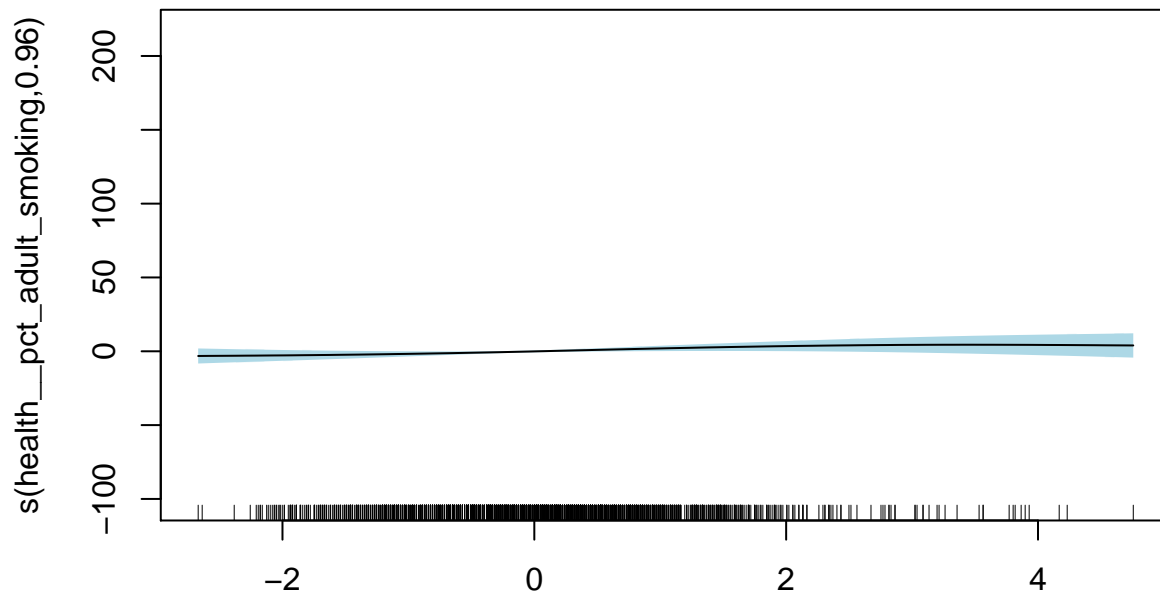


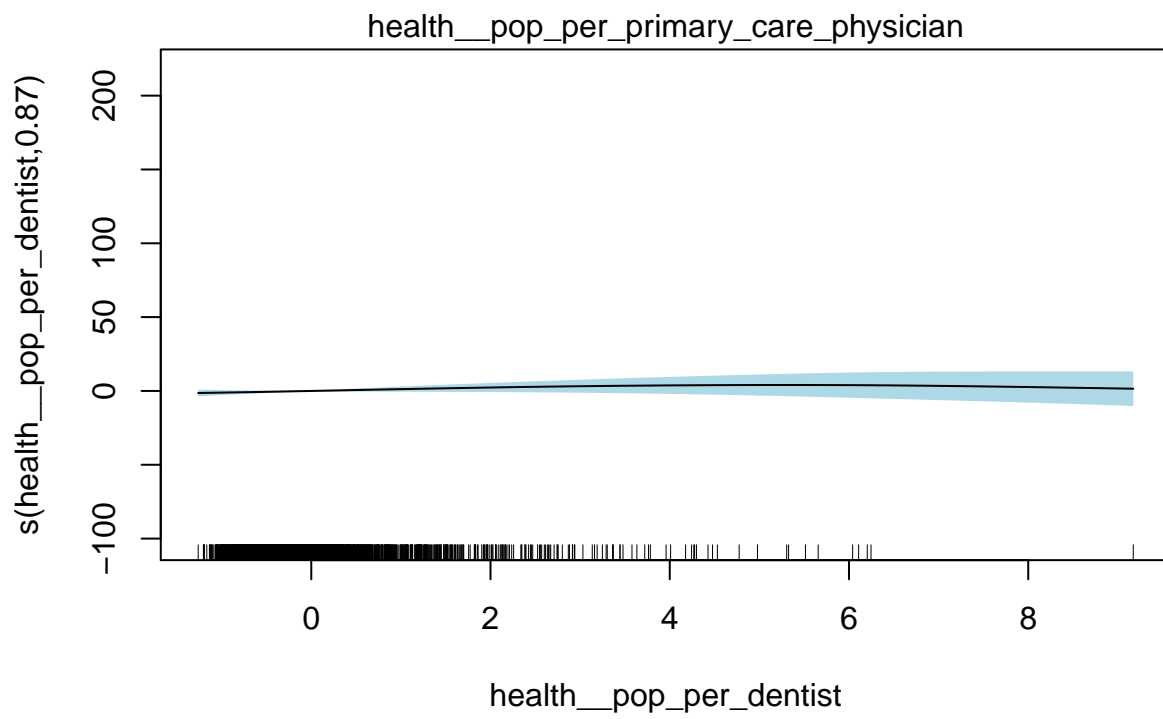
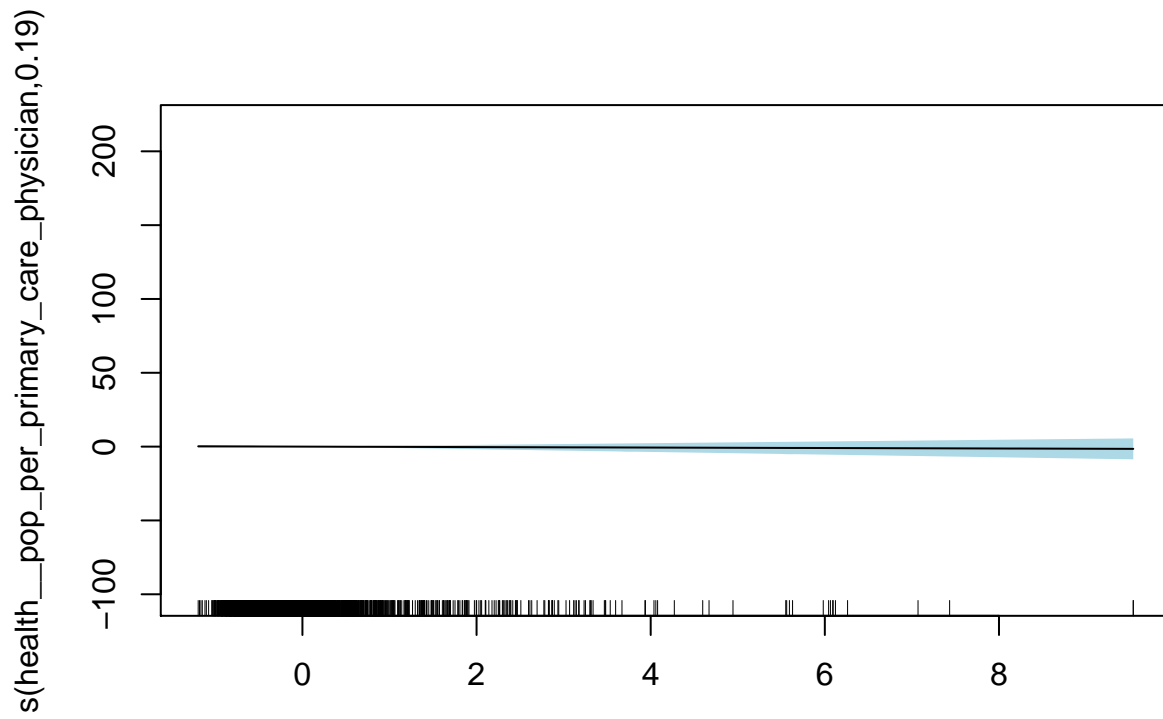


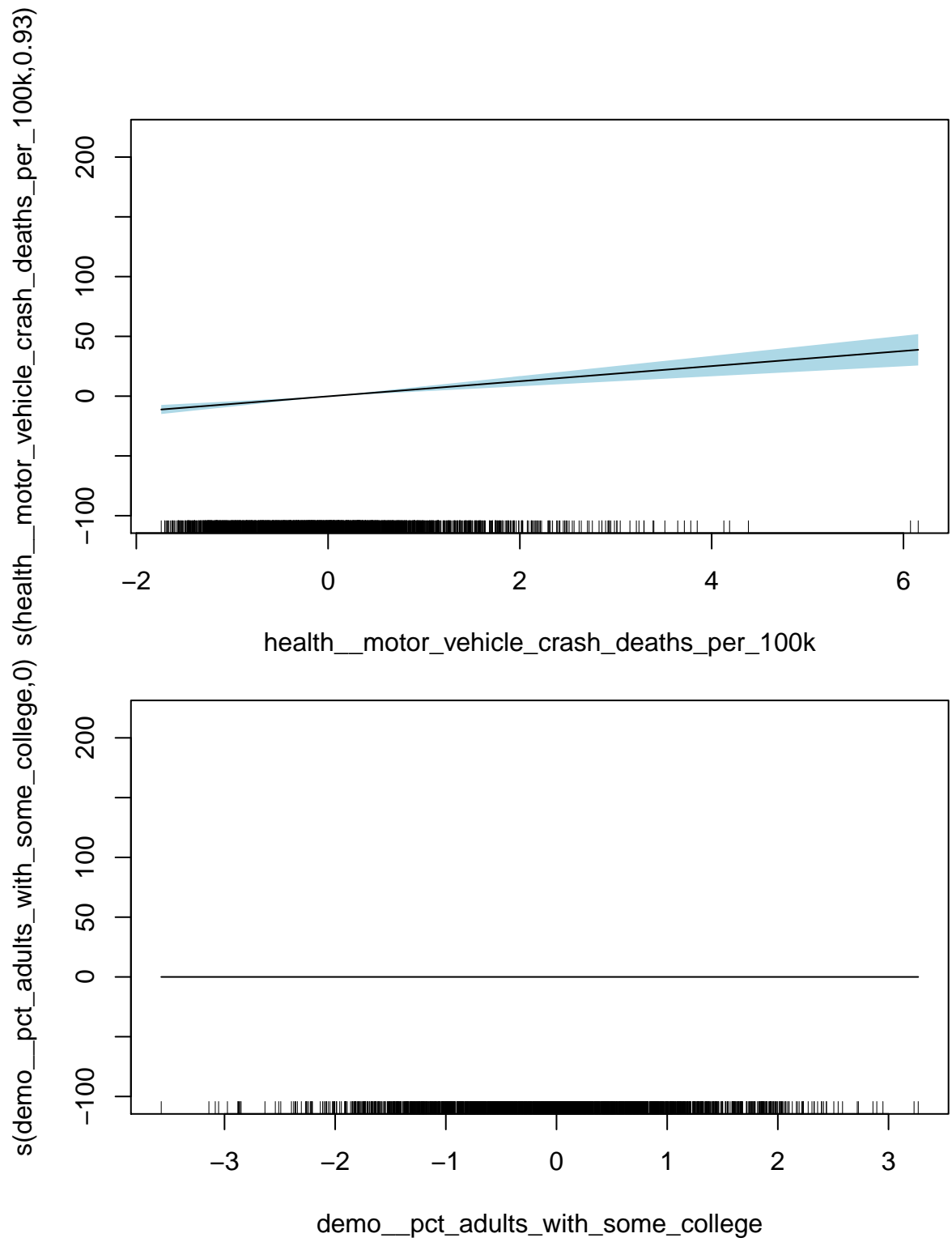




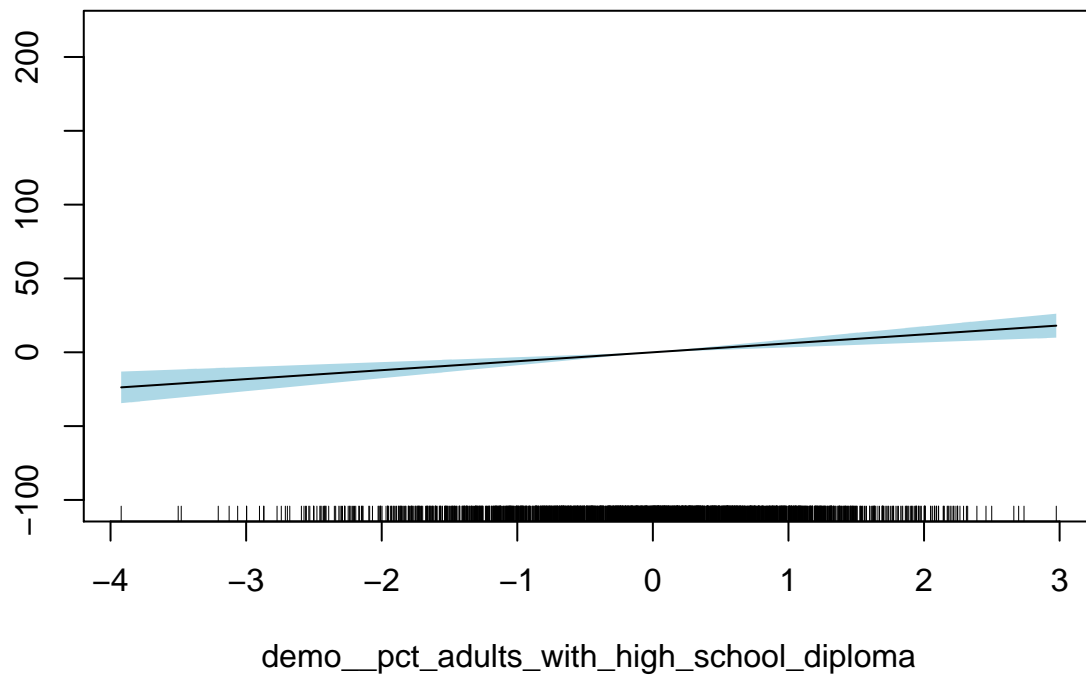
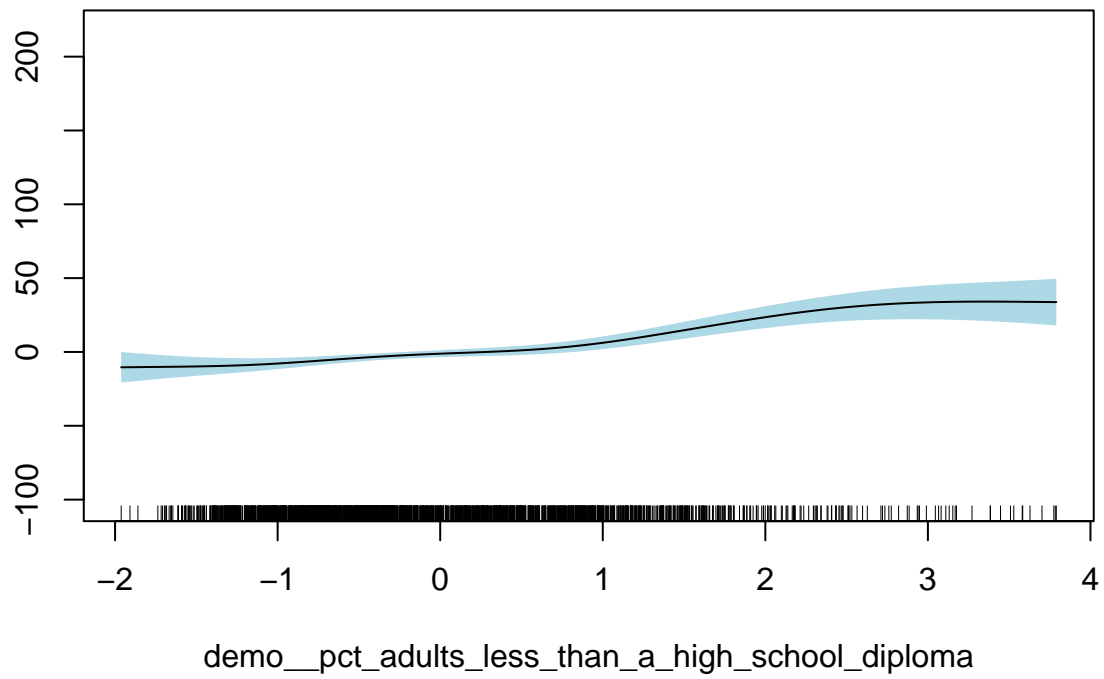


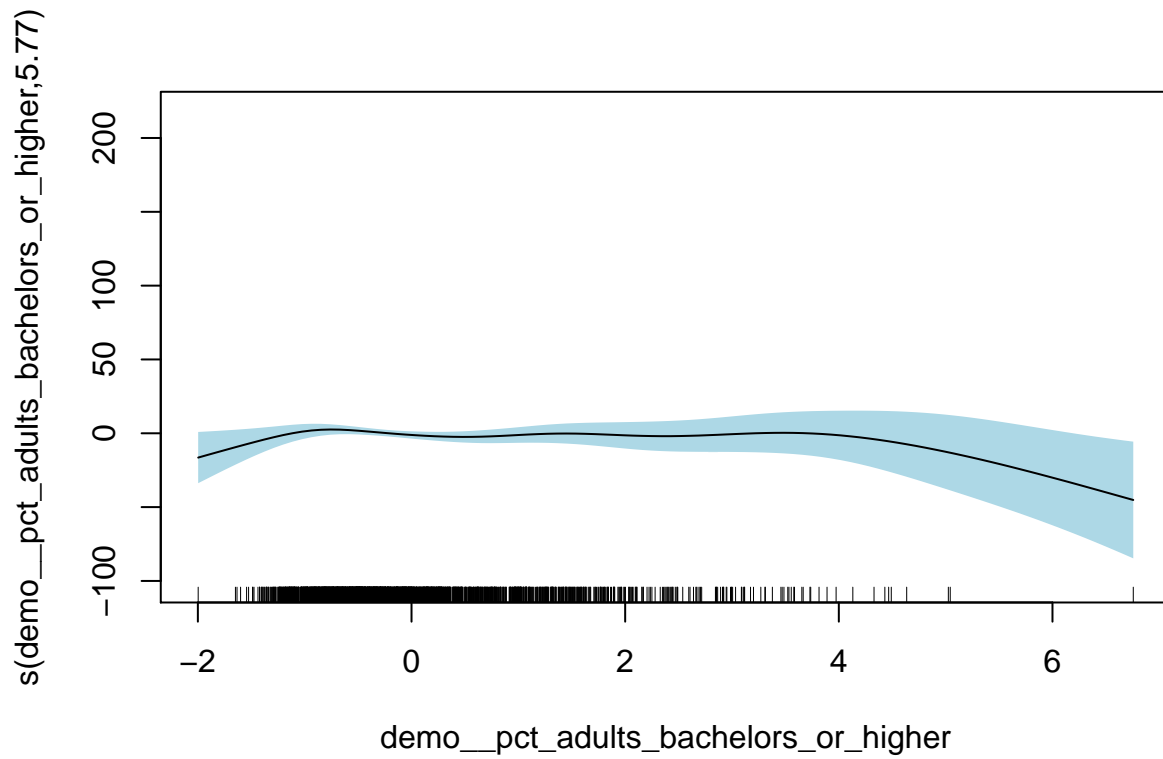






s(demo\_\_pct\_adults\_with\_high\_school\_diploma,0.9)s(demo\_\_pct\_adults\_less\_than\_a\_high\_school\_diploma,3.8)





## Discussion

GAM was our best-performing model. With GAM, we were able to achieve a cross-validated RMSE of median 32.45 and bootstrapped interquartile range of (31.18, 33.34). GAM, and MARS, also performed well on the test data.

However, simpler and more interpretable linear models were close in performance to the more flexible models.

## ## Appendix

### Data prep

```
## Data import
predictors <- read_csv("./data/Training_values.csv")
response <- read_csv("./data/Training_labels.csv")

## Manipulation
data <- response %>%
  full_join(predictors, by = "row_id") %>%
  separate(col = area_rucc, into = c('metro', 'population'), sep = ' - ') %>%
  rename(urban_influence = area_urban_influence,
         economic_typology = econ_economic_typology) %>%
  mutate(pure_population = fct_collapse(as.factor(population),
    "more_than_1mil" = "Counties in metro areas of 1 million population or more",
    "250k_to_1mil" = "Counties in metro areas of 250,000 to 1 million population",
    "less_than_250k" = "Counties in metro areas of fewer than 250,000 population",
    "more_than_20k" = c("Urban population of 20,000 or more, adjacent to a nonmetro area",
      "Urban population of 20,000 or more, not adjacent to a nonmetro area"),
    "2500_to_20k" = c("Urban population of 2,500 to 19,999, adjacent to a nonmetro area",
      "Urban population of 2,500 to 19,999, not adjacent to a nonmetro area"),
    "less_than_2500" = c("Completely rural or less than 2,500 urban population",
      "Completely rural or less than 2,500 urban population"),

    economic_typology = as.factor(recode(economic_typology,
      "Nonspecialized" = "Nonspecialized",
      "Manufacturing-dependent" = "Manufacturing",
      "Farm-dependent" = "Farming",
      "Federal/State government-dependent" = "Government",
      "Mining-dependent" = "Mining",
      "Recreation" = "Recreation")),

    metro = factor(metro,
      levels = c("Metro", "Nonmetro")),
    urban_influence = str_replace_all(urban_influence, " |/-", "_"), # replace problematic characters
    urban_influence = str_replace_all(urban_influence, ",", ""), # replace problematic characters
    demo_pct_nonwhite = demo_pct_hispanic + demo_pct_asian + demo_pct_american_indian_or_alaskan_native,
    urban_influence = fct_rev(urban_influence),
    metro_adjacency = fct_collapse(population,
      metro = c("Counties in metro areas of 1 million population or more",
        "Counties in metro areas of 250,000 to 1 million population",
        "Counties in metro areas of fewer than 250,000 population"),
      adjacent = c("Urban population of 20,000 or more, adjacent to a nonmetro area",
        "Urban population of 2,500 to 19,999, adjacent to a nonmetro area",
        "Completely rural or less than 2,500 urban population"),
      nonadjacent = c("Urban population of 20,000 or more, not adjacent to a nonmetro area",
        "Urban population of 2,500 to 19,999, not adjacent to a nonmetro area",
        "Completely rural or less than 2,500 urban population"),

    dplyr::select(-demo_pct_hispanic,
      -demo_pct_asian,
      -demo_pct_american_indian_or_alaskan_native,
      -demo_pct_non_hispanic_african_american,
      -demo_pct_non_hispanic_white,
```

```

-health_homicides_per_100k, # >90% missing
-health_pct_excessive_drinking, # >90% missing
-yr,
-population,
-row_id)

```

Training and testing data split. Imputation on missing data.

```

## training/test data
set.seed(1)
train_ind <- sample(seq_len(nrow(data)), size = 2/3*nrow(data)) # select rows in 2:1 ratio

train <- data[train_ind, ] # training dataset
test <- data[-train_ind, ] # testing dataset

# Imputation for missing values with caret, based on training data
training_preproc = caret::preProcess(train[, -1],
                                     method = "knnImpute", # automatically centers and scales data
                                     pcaComp = 10,
                                     na.remove = TRUE,
                                     k = 5,
                                     knnSummary = mean,
                                     outcome = NULL,
                                     fudge = .2,
                                     numUnique = 3,
                                     verbose = TRUE)

# Impute training imputation on both training and testing datasets
train_imputed = predict(training_preproc, train)
test_imputed = predict(training_preproc, test)

#save files to Rdata: was not saving the factor structure in read from csv
saveRDS(train_imputed, file = './data/train_imputed.Rdata')
saveRDS(test_imputed, file = './data/test_imputed.Rdata')

```

### Linear Models

Set up caret training control. We will use this for all models.

```

set.seed(100)
ctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 5)

```

Stepwise regression:

```

set.seed(2)
step.fit = caret::train(x, y,
                        method = 'glmStepAIC',
                        metric = "RMSE",
                        trControl = ctrl)
saveRDS(step.fit, "lm_step_imputed.rds")

```

Lasso:

```

set.seed(2)
lasso_fit <- caret::train(x, y,
                          method = "glmnet",
                          metric = "RMSE",

```

```

        tuneGrid = expand.grid(alpha = 1,
                               lambda = exp(seq(-2, 4, length = 200))),
        trControl = ctrl)
plot(lasso_fit, xTrans = function(x) log(x)) #in correct range

saveRDS(lasso_fit, "lasso_imputed.rds")

```

Ridge:

```

set.seed(100)

ridge_fit <- caret::train(x, y,
                          method = "glmnet",
                          tuneGrid = expand.grid(alpha = 0,
                                                  lambda = exp(seq(-2, 10, length = 200))),
                          trControl = ctrl1)

plot(ridge_fit, xTrans = function(x) log(x)) #in correct range

best_lambda_ridge = ridge_fit$bestTune$lambda

saveRDS(ridge_fit, "ridge.rds")

```

PCR:

```

set.seed(2)
pcr_fit <- caret::train(x, y,
                        method = "pcr",
                        trControl = ctrl,
                        metric = "RMSE",
                        tuneLength = 200)
saveRDS(pcr_fit, "pcr_imputed.rds")

```

*Non-linear models*

GAM:

```

set.seed(2)
gam_fit <- caret::train(x, y,
                        method = "gam",
                        metric = 'RMSE',
                        tuneGrid = data.frame(method = "GCV.Cp", select = c(TRUE, FALSE)),
                        trControl = ctrl)
summary(gam_fit)
saveRDS(gam_fit, "gam_fit_imputed.rds")

```

MARS:

```

mars_grid <- expand.grid(degree = 1:3, # degree: 1 vs 2 vs 3, no interaction vs. interaction;
                        nprune = 10:40) # nprune is number of coef

set.seed(2)

mars_fit <- caret::train(x, y,
                        method = "earth",
                        tuneGrid = mars_grid,

```



```

trControl = ctrl)

saveRDS(mars_fit, "mars.rds")

#based on initial results, choose parsimonious version
mars_grid_refined <- expand.grid(degree = 2, nprune = 25:40)

mars_fit_refined <- caret::train(x, y,
                                method = "earth",
                                tuneGrid = mars_grid_refined,
                                trControl = ctrl)
saveRDS(mars_fit_refined, "mars2.rds")

```