Variable Selection and Regression Analysis of Smoking Cessation Outcomes Using Regularization: A Reanalysis of the BASC-Varenicline Trial Data

Abstract

TODO Background: Individuals with past or present major depressive disorder (MDD) experience unique barriers to smoking cessation. While varenicline has been shown to be more effective than traditional nicotine replacement therapy in achieving abstinence for individuals with and without MDD, its lower efficacy for those with MDD suggests that MDD-responsive treatment strategies are necessary. A recent determined that varenicline improved abstinence, but BASC did not, either with or without varenicline. In light of this result, we use data from this study to examine baseline variables as potential predictors of end-of-treatment abstinence and moderators of behavioral treatment's effect thereon.

Methods: We examine data from a randomized, placebo-controlled 2x2 factorial design study of 300 smokers with past or present MDD, who received either placebo or varenicline and standard treatment (ST) or behavioral activation for smoking cessation (BASC). Missing data was imputed through multiple imputation. A cross-validated lasso was applied at the optimally chosen value of lamda for five imputed data sets within n-many bootstrap re-samples. We set aside 20% of the sample for validation.

Results: **TODO**

Conclusion: * lasso identified moderators, predictors "with moderatly good discrimination power" * Limitations include the limited sample size, which produced multiple problems including convergence issues, and class imbalance. * TODO**

Introduction

TODO - make less detailed More than 30% of individuals with major depressive disorder (MDD) are daily smokers (Han et al. 2022; Smith et al. 2020; Weinberger et al. 2020). These individuals experience certain characteristic barriers to smoking cessation. Compared to smokers without MDD, they tend to be more likely to smoke heavily, to experience greater dependence on smoking, and to consider smoking to be more rewarding than other activities; it may thus be unsurprising that smokers with MDD experience more severe withdrawal symptoms (Breslau, Kilbey, and Andreski 1992; Spring, Pingitore, and McChargue 2003; Weinberger, Desai, and McKee 2010; Lyons et al. 2008). Smoking relapse is also associated with lower experience of reward and cognition, both of which tend to be impaired among those diagnosed with MDD (Leventhal et al. 2009; Cook et al. 2010; Patterson et al. 2009). Even past MDD adversely affects smoking cessation treatment outcomes (Hitsman, Papandonatos, et al. 2013).

Perhaps because of these challenges, individuals with MDD have typically been excluded from smoking treatment clinical trials (Hitsman et al. 2003; Hitsman, Papandonatos, et al. 2013; Talukder et al. 2023). Only six trials have explored smoking cessation in individuals with MDD (Evins et al. 2008; Anthenelli et al. 2013; Thorsteinsson et al. 2001; Hall et al. 2006; Minami et al. 2022; Hitsman et al. 2023), two of which had sample sizes of fewer than 50 participants (Thorsteinsson et al. 2001; Minami et al. 2022).

Additional research evaluating approaches to smoking cessation in this population is necessary. One such approach is treatment with varenicline, a pharmaceutical intervention shown to lessen cravings, withdrawal, withdrawal-related cognitive impairment, and reward from smoking (Patterson et al. 2009; Hitsman, Hogarth, et al. 2013; Perkins et al. 2010; Sofuoglu et al. 2009; West et al. 2008; McClure et al. 2012; Cinciripini et al. 2013). While cessation with varenicline was higher in smokers without any mental

health disorders (Anthenelli et al. 2016), one trial of varenicline for individuals with MDD found greater abstinence versus placebo at 52 weeks (28.5% vs. 17.5%) (Anthenelli et al. 2013). However, individuals with mental health disorders, including MDD, are more likely to be prescribed nicotine replacement therapy, a comparatively less effective therapy, than varenicline (Anthenelli et al. 2016; Taylor et al. 2020). This hesitance to prescribing varenicline may stem from a prior boxed warning that, as of 2016, has been removed based on studies evincing the safety of varenicline for individuals with and without mental health disorders (Anthenelli et al. 2016).

The data in our study is derived from a recent randomized, placebo-controlled trial which examined whether the effectiveness of varenicline for smoking cessation in adults with current or past MDD is enhanced by behavioral activation for smoking cessation (BASC) (Hitsman et al. 2023). Behavioral activation (BA) increases reward experience and decreases avoidance-based coping (35-37 Cuijpers, Van Straten, and Warmerdam 2007; Dimidjian et al. 2011; Hopko et al. 2003; MacPherson et al. 2010). A prior pilot study indicated that individuals with elevated depression symptoms, but not MDD, had better cessation results at 26 weeks with BA and nicotine replacement therapy compared to those with standard behavioral treatment (ST) and nicotine replacement therapy (14.3% vs. 0%) (MacPherson et al. 2010).

The Hitsman study from which we draw our data uses a 2x2 factorial design, with 300 participants with past or present MDD treated, over 12 weeks, with BASC or ST alongside placebo or varenciline. Although the Hitsman study found that BASC did not outperform ST with respect to abstinence at 27 weeks, with or without varenicline, it did find that varenicline improved short- and longer-term abstinence compared to placebo (Hitsman et al. 2023). The dual goals of our analysis of the Hitsman et al. (2023) data will be to assess whether and how baseline variables (1) moderate behavioral treatment's effect on end-of-treatment (EOT) abstinence and (2) predict EOT abstinence, controlling for pharmaceutical and behavioral treatments.

In order to accomplish these dual goals, we use regularized linear models. Regularization, also known as shrinkage, reduces variance by fitting a model with all p predictors, with the estimated coefficients shrunken towards zero relative to the least square estimates [Hastie, Tibshirani, and Friedman (2009); pp. 204]. We use this technique here because our goal can be understood as a variable selection problem. The lasso method of regularization, which comepls some of the coefficients to be exactly equal to zero, can also be used for variable selection [Hastie, Tibshirani, and Friedman (2009); pp. 204, 219]. Best subset regression seeks the optimal subset of predictors by evaluating all possible combinations, thereby ensuring selection of the model with the lowest prediction error, although at a computational cost that increases exponentially with the number of predictors (Hazimeh and Mazumder 2020; Hastie, Tibshirani, and Wainwright 2015). In contrast, L0+L2 regularization combines the sparsity-promoting L0 penalty, which counts the number of non-zero coefficients, with the L2 penalty that shrinks coefficients towards zero without zeroing them, effectively balancing variable selection with the need to manage multicollinearity and enhance model accuracy (Hazimeh and Mazumder 2020; Hastie, Tibshirani, and Wainwright 2015; Paul 2024). The relaxed lasso, a modification of the lasso method, initially applies the lasso to determine a subset of variables by shrinking some coefficients to zero, then refits these selected variables with a reduced or absent lasso penalty, potentially reducing bias while retaining the variable selection advantage of the original lasso approach (Meinshausen 2007). L0+L2 and relaxed lasso are considered the best trade-offs between variable selection and goodness of fit (Paul 2024). Best subset, L0+L2, and relaxed lasso are more computationally difficult than standard lasso.

Methods

Variables

First, we note that all covariates, other than abstinence (`abst`), are baseline variables. The following variables are binary: pharmacotherapy (`Var`); psychotherapy (`BA`); sex (`sex_ps`); indicators of whether the participant identifies as non-Hispanic white, Black, and/or Hispanic (`NHW`, `Black`, and `Hisp`); whether the individual smokes within five minutes of waking up (`ftcd.5.mins`); other lifetime DSM-5 diagnosis (`otherdiag`); whether the participant takes antidepressant medication (`antidepmed`); current or past MDD (`mde_curr`); and exclusive use of methylated cigarettes (`Only.Menthol`). The numeric variables are age (`age_ps`); cigarettes per day (`cpd_ps`); and Nicotine Metabolism Ratio (`NMR`). All remaining variables are ordinal. However, we treat income (`inc`) and education (`edu`) as ordinal, but (as we discuss below) treat the following variables as numeric: FTCD score (`ftcd_score`); `bdi_score_pq1`; cigarette reward value (`crv_total_pg1`); substitute reinforcer score (`hedonsum_n_pq1`); complementary reinforcer score (`hedonsum_y_pq1`); anhedonia (`shaps_score_pg1`); and readiness to quit smoking (`readiness`).

The `ftcd_score` is the participant's score on the Fagerstrom Test for Cigarette Dependence, a six-item survey intended to evaluate the quantity of cigarette consumption, dependence, and compulsion to smoke; answers are scored and summed to yield a total score of 0-10, with higher scores indicating a greater physical dependence on nicotine (Fagerström 2011).

The `bdi_score_pq1` is the participant's score on the Beck Depression Inventory, or BDI-II, a 21-item survey scored from 0 to 63 measuring the severity of depression, with higher scores indicating greater severity (Beck, Steer, and Brown 1996).

The `crv_total_pq1` score measures the participant's score on a questionnaire measuring preference for smoking over other traditionally rewarding activities, measured on a scale of 0 to 15, where 1 point is added for each time the participant chooses smoking (Spring, Pingitore, and McChargue 2003).

Substitute and complimentary reinforcer scores (`hedonsum_n_pq1` and `hedonsum_y_pg1`) come from a participant's score of the cross-product of frequency (measured from 0 to 2) and level of enjoyableness (0 to 2) for a 45-item version of the Pleasant Events Schedule (MacPhillamy and Lewinsohn 1982); based on the participant's report of whether an event is or is not associated with smoking, it is designated a complementary or substitute reinforcer, the cross products of which are summed.

Anhedonia (`shaps_score_pq1`) is assessed with the 14-item Snaith-Hamilton Pleasure Scale (SHAPS), where 14 statements are ranked on a 4-point Likert scale, with higher scores indicating greater enjoyment of typically rewarding experiences (Snaith et al. 1995).

Nicotine metabolism ratio, otherwise known as nicotine metabolite ratio (`NMR`), is calculated as the ratio of 3'hydroxycotinine [3HC] to cotinine; `NMR` is a biomarker where higher `NMR` is associated with faster metabolism of nicotine (Siegel et al. 2020).

Exploratory Data Analysis

The original paper uses a 2x2 factorial design. However, we choose to keep the treatment variables separate because the sample size is very small and further reducing the sample size to the four groups at issue would significantly limit the Lasso's ability to fit the data. That the 2x2 factorial design is not necessary to answer the research question validates our choice.

In addition to summary statistics available in Tables 1, 2, and 3 and the correlation matrix, we also inspect univariate histograms (omitted for space) for all variables. Bivariate plots with the outcome are

omitted because the outcome is binary. We note that there is still the assumption that there is a linear relationship between the log odds of predictors and the outcome.

We note that certain survey variables, specifically `ftcd_score`, `crv_total_pq1`, `readiness`, `shaps_score_pq1`, `hedonsum_y_pq1`, and `hedonsum_n_pq1` are ordinal but will be treated here as either numeric integers or continuous for the purpose of the modeling. This presents a limitation in that we are treating ordinal variables as evenly spaced. Doing so is more or less justifiable for different variables. For example, we are more justified in treating `hedonsum_y_pq1` and `hedonsuum_n_pq1` as continuous, given that each score is made up of the product of two ordinal scales; by using the cross-product, the study authors have made some assumptions about relative scale that obfuscates the ordinal nature of the original.

Nevertheless, treating these variables as ordinal is untenable here, as it would make individual cell sizes incredibly small and lead to parameter proliferation in modeling. Another potential solution would be grouping ordinal variables into discernible levels, as with `inc`, discussed below, but this would take an in-depth understanding of the meaning and spacing for each score that we do not have at present. Said another way, grouping the variables without fully understanding how relative distance between ordinal levels is expressed would lead to our making similar assumptions about the spacing between levels as we make in treating the variables as continuous. We note this is consistent with the handling of survey score variables in Table 1 of the original paper.

To ameliorate the potential loss of accurate fit from treating ordinal variables as numeric, we use polynomial fit for each variable, allowing non-linearities between the log odds ratio for these variables and the outcome. The default contrast handling for ordinal covariates in R is a polynomial contrast. Applying the polynomial score transformation approximates this relationship, without incurring the cost of a parameter for all but one level of the variable.

Other notable decisions include: to omit `Hisp` due to insufficient sample size (18 individuals), which is a limitation of our analysis. We considered taking the three race and ethnicity variables `NHW`, `Black`, and `Hisp` and producing a single race plus ethnicity variable with mutually exclusive groups for non-hispanic white, non-hispanic Black, Hispanic, and Other. However because there are only 18 individuals noted as `Hisp` (and only two of those individuals are `Black`), this hypothetical combined race plus ethnicity variable would've suffered from the same cell size issue as other factors with many levels.

We further note that based on our correlation matrix there is a very high correlation (well over 85%) between `NHW` and `Black`. This issue combined with the issue noted previously lead us to also drop the `NHW` variable. We also note that use of only menthol cigarettes is highly correlated with both `NHW`, `Black`, `inc`, and `edu`. This is a concern, but as menthol cigarette use may be a unique trait and we are conducting regularized regression, we elect to leave this variable in the model and allow the penalty to aid our choice.

We conclude by grouping levels of `edu` and `inc` to limit parameter proliferation and allow for estimation and model convergence, given that Table 1 shows that there is insufficient sample within each level when not grouped. And, to take the log of the one true continuous variable, `NMR`, to encourage normality. Regarding the last decision, we do not assume normality, but note that normalizing transformations can encourage convergence and simplify the loss landscape. We note that use of only menthol cigarettes is highly correlated with both `NHW`, `Black`, `inc`, and `edu`.

Finally, we note that there is a minor class imbalance issue (~2.1:10). The outcome is somewhat rare, and this could lead to model fitting issues. Correcting for this class imbalance is a potential area for improvement and a limitation.

Table 1: Summary of Variables

Variable	Type	Summary	Normal Distribution	Outlier(s) Present	Skewness	Kurtosis
abst [n (%)]	Categorical	No: 236 (78.67%), Yes: 64 (21.33%)	NA	NA	NA	NA
Var [n (%)]	Categorical	Placebo: 136 (45.33%), Varenicline: 164 (54.67%)	NA	NA	NA	NA
BA [n (%)]	Categorical	Standard: 149 (49.67%), BA: 151 (50.33%)	NA	NA	NA	NA
age_ps [Mean (SD) (Quantile)]	Numeric	49.99 (12.6) (41 - 59)	No	No	Left-skewed	Platykurtic
sex_ps [n (%)]	Categorical	Male: 135 (45%), Female: 165 (55%)	NA	NA	NA	NA
NHW [n (%)]	Categorical	No: 195 (65%), Yes: 105 (35%)	NA	NA	NA	NA
Black [n (%)]	Categorical	No: 143 (47.67%), Yes: 157 (52.33%)	NA	NA	NA	NA
Hisp [n (%)]	Categorical	No: 282 (94%), Yes: 18 (6%)	NA	NA	NA	NA
inc [n (%)]	Categorical	Less than $20,000: 110 (37.04%), $20,000–35,000: 68 (22.9%), $35,001–50,000: 46 (15.49%), $50,001–75,000: 38 (12.79%), More than $75,000: 35 (11.78%)	NA	NA	NA	NA
edu [n (%)]	Categorical	Grade School: 1 (0.33%), Some High School: 16 (5.33%), High School Graduate/GED: 76 (25.33%), Some College/Technical School: 116 (38.67%), College Graduate: 91 (30.33%)	NA	NA	NA	NA
ftcd_score [Mean (SD) (Quantile)]	Numeric	5.22 (2.14) (4 - 7)	No	No	Centered	Mesokurtic
ftcd.5.mins [n (%)]	Categorical	No: 162 (54%), Yes: 138 (46%)	NA	NA	NA	NA
bdi_score_w00 [Mean (SD) (Quantile)]	Numeric	18.72 (11.47) (10 - 26)	No	No	Centered	Mesokurtic
cpd_ps [Mean (SD) (Quantile)]	Numeric	15.15 (7.89) (10 - 20)	No	Yes	Right-skewed	Leptokurtic
crv_total_pq1 [Mean (SD) (Quantile)]	Numeric	7.19 (3.7) (5 - 9.75)	No	No	Centered	Platykurtic
hedonsum_n_pq1 [Mean (SD) (Quantile)]	Numeric	22.63 (19.6) (9 - 31)	No	Yes	Right-skewed	Leptokurtic
hedonsum_y_pq1 [Mean (SD) (Quantile)]	Numeric	25.43 (19.42) (12 - 35)	No	Yes	Right-skewed	Leptokurtic
shaps_score_pq1 [Mean (SD) (Quantile)]	Numeric	2.25 (3.16) (0 - 3)	No	Yes	Right-skewed	Leptokurtic
otherdiag [n (%)]	Categorical	No: 167 (55.67%), Yes: 133 (44.33%)	NA	NA	NA	NA
antidepmed [n (%)]	Categorical	No: 218 (72.67%), Yes: 82 (27.33%)	NA	NA	NA	NA
mde_curr [n (%)]	Categorical	Past: 153 (51%), Current: 147 (49%)	NA	NA	NA	NA
NMR [Mean (SD) (Quantile)]	Numeric	0.36 (0.23) (0.21 - 0.47)	No	Yes	Right-skewed	Leptokurtic
Only.Menthol [n (%)]	Categorical	No: 120 (40.27%), Yes: 178 (59.73%)	NA	NA	NA	NA

(continued)

Variable	Type	Summary	Normal Distribution	Outlier(s) Present	Skewness	Kurtosis
readiness [Mean (SD) (Quantile)]	Numeric	6.78 (1.24) (6 - 8)	No	No	Centered	Mesokurtic

Note:
Shapiro-Wilk test for normality; Grubb's test for outliers.

Table 2: Summary of Variables by Outcome

Variable	No	Yes	Sig.
Var [n (%)]	Placebo: 124 (52.54%), Varenicline: 112 (47.46%)	Placebo: 12 (18.75%), Varenicline: 52 (81.25%)	****
BA [n (%)]	Standard: 115 (48.73%), BA: 121 (51.27%)	Standard: 34 (53.12%), BA: 30 (46.88%)	ns
age_ps [Mean (SD) (Quantile)]	49.81 (12.63) (41 - 59)	50.62 (12.55) (43.75 - 59.25)	ns
sex_ps [n (%)]	Male: 107 (45.34%), Female: 129 (54.66%)	Male: 28 (43.75%), Female: 36 (56.25%)	ns
NHW [n (%)]	No: 162 (68.64%), Yes: 74 (31.36%)	No: 33 (51.56%), Yes: 31 (48.44%)	ns
Black [n (%)]	No: 107 (45.34%), Yes: 129 (54.66%)	No: 36 (56.25%), Yes: 28 (43.75%)	ns
Hisp [n (%)]	No: 221 (93.64%), Yes: 15 (6.36%)	No: 61 (95.31%), Yes: 3 (4.69%)	ns
inc [n (%)]	Less than $20,000: 88 (37.61%), $20,000–35,000: 56 (23.93%), $35,001–50,000: 36 (15.38%), $50,001–75,000: 30 (12.82%), More than $75,000: 24 (10.26%)	Less than $20,000: 22 (34.92%), $20,000–35,000: 12 (19.05%), $35,001–50,000: 10 (15.87%), $50,001–75,000: 8 (12.7%), More than $75,000: 11 (17.46%)	ns
edu [n (%)]	Grade School: 0 (0%), Some High School: 13 (5.51%), High School Graduate/GED: 60 (25.42%), Some College/Technical School: 97 (41.1%), College Graduate: 66 (27.97%)	Grade School: 1 (1.56%), Some High School: 3 (4.69%), High School Graduate/GED: 16 (25%), Some College/Technical School: 19 (29.69%), College Graduate: 25 (39.06%)	ns
ftcd_score [Mean (SD) (Quantile)]	5.46 (1.98) (4 - 7)	4.34 (2.46) (2 - 6)	*
ftcd.5.mins [n (%)]	No: 123 (52.12%), Yes: 113 (47.88%)	No: 39 (60.94%), Yes: 25 (39.06%)	ns
bdi_score_w00 [Mean (SD) (Quantile)]	19.14 (11.54) (11 - 26)	17.19 (11.19) (8 - 24)	ns
cpd_ps [Mean (SD) (Quantile)]	15.57 (7.81) (10 - 20)	13.58 (8.07) (7 - 20)	ns
crv_total_pq1 [Mean (SD) (Quantile)]	7.21 (3.71) (5 - 9)	7.09 (3.71) (5 - 10)	ns
hedonsum_n_pq1 [Mean (SD) (Quantile)]	21.9 (18.9) (8 - 30)	25.31 (21.96) (10 - 34.5)	ns
hedonsum_y_pq1 [Mean (SD) (Quantile)]	25.97 (19.27) (13 - 35)	23.47 (20) (9 - 32)	ns
shaps_score_pq1 [Mean (SD) (Quantile)]	2.42 (3.36) (0 - 4)	1.64 (2.22) (0 - 2)	ns
otherdiag [n (%)]	No: 127 (53.81%), Yes: 109 (46.19%)	No: 40 (62.5%), Yes: 24 (37.5%)	ns
antidepmed [n (%)]	No: 171 (72.46%), Yes: 65 (27.54%)	No: 47 (73.44%), Yes: 17 (26.56%)	ns

(continued)

Variable	No	Yes	Sig.
mde_curr [n (%)]	Past: 114 (48.31%), Current: 122 (51.69%)	Past: 39 (60.94%), Current: 25 (39.06%)	ns
NMR [Mean (SD) (Quantile)]	0.35 (0.22) (0.2 - 0.46)	0.42 (0.26) (0.25 - 0.53)	ns
Only.Menthol [n (%)]	No: 91 (38.89%), Yes: 143 (61.11%)	No: 29 (45.31%), Yes: 35 (54.69%)	ns
readiness [Mean (SD) (Quantile)]	6.8 (1.26) (6 - 8)	6.68 (1.17) (6 - 7.5)	ns

Note:
Kruskal–Wallis test for continuous variables, Chi-Square test for categorical variables. Bonferroni correction applied.
Note:
ns = P > 0.05, * = P ≤ 0.05, ** = P ≤ 0.01, *** = P ≤ 0.001, **** = P ≤ 0.0001

Table 3: Summary of Variables by BA

Variable	Standard	BA	Sig.
abst [n (%)]	No: 115 (77.18%), Yes: 34 (22.82%)	No: 121 (80.13%), Yes: 30 (19.87%)	ns
Var [n (%)]	Placebo: 68 (45.64%), Varenicline: 81 (54.36%)	Placebo: 68 (45.03%), Varenicline: 83 (54.97%)	ns
age_ps [Mean (SD) (Quantile)]	49.45 (11.85) (43 - 58)	50.52 (13.31) (41 - 60)	ns
sex_ps [n (%)]	Male: 66 (44.3%), Female: 83 (55.7%)	Male: 69 (45.7%), Female: 82 (54.3%)	ns
NHW [n (%)]	No: 102 (68.46%), Yes: 47 (31.54%)	No: 93 (61.59%), Yes: 58 (38.41%)	ns
Black [n (%)]	No: 66 (44.3%), Yes: 83 (55.7%)	No: 77 (50.99%), Yes: 74 (49.01%)	ns
Hisp [n (%)]	No: 140 (93.96%), Yes: 9 (6.04%)	No: 142 (94.04%), Yes: 9 (5.96%)	ns
inc [n (%)]	Less than $20,000: 55 (37.16%), $20,000–35,000: 35 (23.65%), $35,001–50,000: 25 (16.89%), $50,001–75,000: 14 (9.46%), More than $75,000: 19 (12.84%)	Less than $20,000: 55 (36.91%), $20,000–35,000: 33 (22.15%), $35,001–50,000: 21 (14.09%), $50,001–75,000: 24 (16.11%), More than $75,000: 16 (10.74%)	ns
edu [n (%)]	Grade School: 0 (0%), Some High School: 6 (4.03%), High School Graduate/GED: 38 (25.5%), Some College/Technical School: 62 (41.61%), College Graduate: 43 (28.86%)	Grade School: 1 (0.66%), Some High School: 10 (6.62%), High School Graduate/GED: 38 (25.17%), Some College/Technical School: 54 (35.76%), College Graduate: 48 (31.79%)	ns
ftcd_score [Mean (SD) (Quantile)]	5.27 (2.08) (4 - 7)	5.18 (2.2) (4 - 7)	ns
ftcd.5.mins [n (%)]	No: 76 (51.01%), Yes: 73 (48.99%)	No: 86 (56.95%), Yes: 65 (43.05%)	ns
bdi_score_w00 [Mean (SD) (Quantile)]	19.03 (11.57) (11 - 26)	18.42 (11.4) (9.5 - 26)	ns
cpd_ps [Mean (SD) (Quantile)]	14.7 (6.89) (10 - 20)	15.58 (8.77) (10 - 20)	ns
crv_total_pq1 [Mean (SD) (Quantile)]	7.04 (3.55) (5 - 9)	7.33 (3.84) (5 - 10)	ns
hedonsum_n_pq1 [Mean (SD) (Quantile)]	22.21 (19.74) (9 - 32)	23.05 (19.52) (9 - 31)	ns
hedonsum_y_pq1 [Mean (SD) (Quantile)]	26.09 (19.62) (12 - 37)	24.78 (19.27) (12 - 31.5)	ns

(continued)

Variable	Standard	BA	Sig.
shaps_score_pq1 [Mean (SD) (Quantile)]	2.29 (3.17) (0 - 3)	2.21 (3.16) (0 - 3)	ns
otherdiag [n (%)]	No: 81 (54.36%), Yes: 68 (45.64%)	No: 86 (56.95%), Yes: 65 (43.05%)	ns
antidepmed [n (%)]	No: 119 (79.87%), Yes: 30 (20.13%)	No: 99 (65.56%), Yes: 52 (34.44%)	ns
mde_curr [n (%)]	Past: 74 (49.66%), Current: 75 (50.34%)	Past: 79 (52.32%), Current: 72 (47.68%)	ns
NMR [Mean (SD) (Quantile)]	0.36 (0.24) (0.2 - 0.46)	0.37 (0.22) (0.22 - 0.48)	ns
Only.Menthol [n (%)]	No: 58 (39.19%), Yes: 90 (60.81%)	No: 62 (41.33%), Yes: 88 (58.67%)	ns
readiness [Mean (SD) (Quantile)]	6.82 (1.22) (6 - 8)	6.73 (1.27) (6 - 8)	ns

Note:
Kruskal–Wallis test for continuous variables, Chi-Square test for categorical variables. Bonferroni correction applied.
Note:
ns = P > 0.05, * = P ≤ 0.05, ** = P ≤ 0.01, *** = P ≤ 0.001, **** = P ≤ 0.0001

Table 4: Summary of Variables by Var

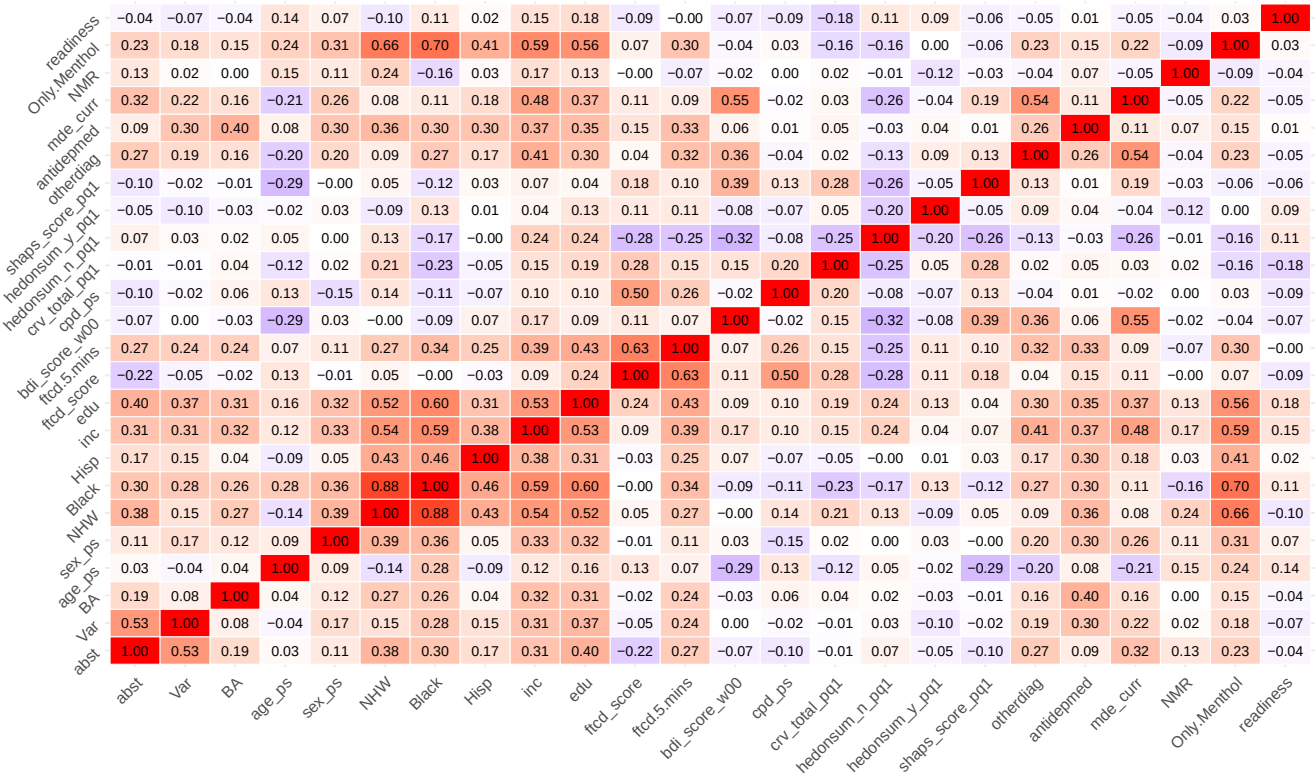
Variable	Placebo	Varenicline	Sig.
abst [n (%)]	No: 124 (91.18%), Yes: 12 (8.82%)	No: 112 (68.29%), Yes: 52 (31.71%)	****
BA [n (%)]	Standard: 68 (50%), BA: 68 (50%)	Standard: 81 (49.39%), BA: 83 (50.61%)	ns
age_ps [Mean (SD) (Quantile)]	50.53 (12.19) (42.75 - 59)	49.54 (12.94) (40.75 - 59)	ns
sex_ps [n (%)]	Male: 59 (43.38%), Female: 77 (56.62%)	Male: 76 (46.34%), Female: 88 (53.66%)	ns
NHW [n (%)]	No: 90 (66.18%), Yes: 46 (33.82%)	No: 105 (64.02%), Yes: 59 (35.98%)	ns
Black [n (%)]	No: 59 (43.38%), Yes: 77 (56.62%)	No: 84 (51.22%), Yes: 80 (48.78%)	ns
Hisp [n (%)]	No: 127 (93.38%), Yes: 9 (6.62%)	No: 155 (94.51%), Yes: 9 (5.49%)	ns
inc [n (%)]	Less than $20,000: 51 (37.78%), $20,000–35,000: 30 (22.22%), $35,001–50,000: 22 (16.3%), $50,001–75,000: 20 (14.81%), More than $75,000: 12 (8.89%)	Less than $20,000: 59 (36.42%), $20,000–35,000: 38 (23.46%), $35,001–50,000: 24 (14.81%), $50,001–75,000: 18 (11.11%), More than $75,000: 23 (14.2%)	ns
edu [n (%)]	Grade School: 1 (0.74%), Some High School: 5 (3.68%), High School Graduate/GED: 34 (25%), Some College/Technical School: 60 (44.12%), College Graduate: 36 (26.47%)	Grade School: 0 (0%), Some High School: 11 (6.71%), High School Graduate/GED: 42 (25.61%), Some College/Technical School: 56 (34.15%), College Graduate: 55 (33.54%)	ns
ftcd_score [Mean (SD) (Quantile)]	5.35 (2.05) (4 - 7)	5.12 (2.21) (4 - 7)	ns
ftcd.5.mins [n (%)]	No: 69 (50.74%), Yes: 67 (49.26%)	No: 93 (56.71%), Yes: 71 (43.29%)	ns
bdi_score_w00 [Mean (SD) (Quantile)]	18.71 (11.54) (9.75 - 27)	18.74 (11.45) (10 - 25)	ns
cpd_ps [Mean (SD) (Quantile)]	15.33 (8.2) (10 - 20)	14.99 (7.65) (10 - 20)	ns
crv_total_pq1 [Mean (SD) (Quantile)]	7.23 (3.74) (5 - 9)	7.15 (3.68) (5 - 10)	ns

(continued)

Variable	Placebo	Varenicline	Sig.
hedonsum_n_pq1 [Mean (SD) (Quantile)]	21.98 (20.16) (9 - 30)	23.17 (19.17) (9 - 34)	ns
hedonsum_y_pq1 [Mean (SD) (Quantile)]	27.54 (20.64) (13 - 36.25)	23.68 (18.23) (11.75 - 31.5)	ns
shaps_score_pq1 [Mean (SD) (Quantile)]	2.33 (3.3) (0 - 3)	2.18 (3.05) (0 - 3)	ns
otherdiag [n (%)]	No: 73 (53.68%), Yes: 63 (46.32%)	No: 94 (57.32%), Yes: 70 (42.68%)	ns
antidepmed [n (%)]	No: 93 (68.38%), Yes: 43 (31.62%)	No: 125 (76.22%), Yes: 39 (23.78%)	ns
mde_curr [n (%)]	Past: 73 (53.68%), Current: 63 (46.32%)	Past: 80 (48.78%), Current: 84 (51.22%)	ns
NMR [Mean (SD) (Quantile)]	0.36 (0.23) (0.21 - 0.44)	0.37 (0.23) (0.21 - 0.5)	ns
Only.Menthol [n (%)]	No: 52 (38.52%), Yes: 83 (61.48%)	No: 68 (41.72%), Yes: 95 (58.28%)	ns
readiness [Mean (SD) (Quantile)]	6.88 (1.35) (6 - 8)	6.7 (1.15) (6 - 8)	ns

Note:
Kruskal–Wallis test for continuous variables, Chi-Square test for categorical variables. Bonferroni correction applied.
Note:
ns = P > 0.05, * = P ≤ 0.05, ** = P ≤ 0.01, *** = P ≤ 0.001, **** = P ≤ 0.0001

Figure 1: Correlation Matrix



Missingness

We looked at the amount of missing data by variable and by sample. - ***TODO: insert missingness table*** After inspecting the data, we ran Little's test for MCAR. The null hypothesis for this test is that the data are not MCAR. The p-value was over 0.05, so we cannot reject the null hypothesis. We conclude that there is evidence that these data are missing completely at random. Missing data in covariates will be imputed during the bootstrap process to preserve the already limited sample available.

Modeling

We use a Lasso model to answer both questions at issue. To evaluate baseline variables that predict `abst`, we construct a Lass model that includes all main effects and polynomial terms for score variables. To evaluate baseline moderators of the effect of `BA` on `abst`, we also use a Lasso model, but the model that we use this time includes all main effects and polynomial terms of scores, as well as interactions of those terms with `BA`.

We choose a Lasso model due to its constraints on potential values of the coefficients, introducing desirable bias to the model when performed correctly, which prevent overfitting. We choose not to use L0, L0+L2, or relaxed Lasso: while all three are computationally more difficult and would have more convergence issues than Lasso, the former two also have a harsher penalty (Hazimeh and Mazumder 2020; Hastie, Tibshirani, and Wainwright 2015; Meinshausen 2007).

First, we perform an 80/20 validation training split, where each split has the same proportion of positive outcomes (where `abst`=1). We proceed with the bootstrap of the training data: first, through a bootstrap re-sample of the data, then imputing missing data, and by estimating our lasso on the imputed bootstrap data as many times as necessary. Coefficients are pooled and confidence intervals are estimated. Bootstrap is used to stabilize estimates due to volatility caused by low sample size and to estimate confidence intervals for coefficients.

After the estimates are derived from the bootstrap procedure, we evaluate the model's performance using the validation data. We tested the model's performance using the pooled estimates on validation data in order to assess calibration and discrimination. For the former, we use calibration plots comparing the model's abstinence predictions to the study's actual results. For the latter, we use AUC and ROC to test whether the model could correctly identify participants who were or were not abstinent.

Results

Baseline Variables Predict Abstinence

Results are presented for bootstrapped lasso regression on outcome abstinence, including main effects of baseline covariates and controlling for other treatments.

There are four significant baseline predictors of abstinence: `ftcd_score`, `Var`, `hedonsum_y_pq1`, and `logNMR`. That these are predictors of abstinence is not surprising. Although the relationship is not linear, `ftcd_score` is a measure of dependence, and thus would ***TODO*** varenicline (`Var`) is known to be

Table 5: Lasso Regression Coefficient Estimates

Predictor	Estimate (Median)	Odds Ratio	Lower 2.5%	Upper 97.5%	Proportion Non-Zero	Significant
(Intercept)	3.233	25.351	-20.760	29.260	1.00	No
BABA	-0.731	0.481	-1.476	0.583	1.00	No
VarVarenicline	2.383	10.841	1.223	4.105	1.00	Yes

Table 5: Lasso Regression Coefficient Estimates *(continued)*

Predictor	Estimate (Median)	Odds Ratio	Lower 2.5%	Upper 97.5%	Proportion Non-Zero	Significant
poly(ftcd_score, 3, raw = TRUE)1	-3.237	0.039	-6.173	-0.667	1.00	Yes
poly(ftcd_score, 3, raw = TRUE)2	0.775	2.171	0.134	1.456	1.00	Yes
poly(ftcd_score, 3, raw = TRUE)3	-0.056	0.946	-0.105	-0.013	1.00	Yes
poly(bdi_score_w(3, raw = TRUE)1	0.190	1.209	-0.107	0.484	0.98	No
poly(bdi_score_w00, 3, raw = TRUE)2	-0.009	0.991	-0.025	0.003	0.96	No
poly(bdi_score_w(3, raw = TRUE)3	0.000	1.000	0.000	0.000	0.96	No
poly(cpd_ps, 3, raw = TRUE)1	-0.481	0.618	-1.477	0.275	0.98	No
poly(cpd_ps, 3, raw = TRUE)2	0.021	1.021	-0.022	0.067	0.96	No
poly(cpd_ps, 3, raw = TRUE)3	0.000	1.000	-0.001	0.000	0.94	No
poly(crv_total_pq 3, raw = TRUE)1	-0.319	0.727	-1.528	0.938	1.00	No
poly(crv_total_pq1, 3, raw = TRUE)2	0.070	1.072	-0.111	0.304	0.96	No
poly(crv_total_pq 3, raw = TRUE)3	-0.003	0.997	-0.016	0.003	0.96	No
poly(hedonsum_n_pq1, 3, raw = TRUE)1	-0.051	0.950	-0.252	0.041	1.00	No
poly(hedonsum_n_ 3, raw = TRUE)2	0.001	1.001	-0.002	0.005	1.00	No
poly(hedonsum_n_pq1, 3, raw = TRUE)3	0.000	1.000	0.000	0.000	0.92	No
poly(hedonsum_y_ 3, raw = TRUE)1	-0.204	0.816	-0.460	-0.021	1.00	Yes
poly(hedonsum_y_pq1, 3, raw = TRUE)2	0.006	1.006	0.000	0.012	1.00	Yes
poly(hedonsum_y_ 3, raw = TRUE)3	0.000	1.000	0.000	0.000	1.00	No
poly(shaps_score_pq1, 3, raw = TRUE)1	-0.191	0.826	-1.723	0.864	1.00	No
poly(shaps_score_ 3, raw = TRUE)2	0.103	1.108	-0.182	0.631	0.98	No
poly(shaps_score_pq1, 3, raw = TRUE)3	-0.011	0.989	-0.051	0.007	1.00	No
poly(readiness, 3, raw = TRUE)1	0.466	1.594	-7.772	12.831	0.90	No
poly(readiness, 3, raw = TRUE)2	0.000	1.000	-1.749	1.245	0.70	No
poly(readiness, 3, raw = TRUE)3	-0.006	0.994	-0.064	0.076	0.92	No
age_ps	0.023	1.023	-0.018	0.083	0.98	No
logNMR	1.455	4.284	0.628	2.094	1.00	Yes
sex_psFemale	-0.334	0.716	-1.197	0.656	1.00	No
BlackYes	-1.100	0.333	-2.345	0.211	1.00	No
ftcd.5.minsYes	0.401	1.493	-1.018	1.760	0.98	No
otherdiagYes	-0.815	0.443	-2.207	0.593	1.00	No
antidepmedYes	0.164	1.178	-1.042	1.106	1.00	No
mde_currCurrent	-0.027	0.974	-0.946	1.305	0.98	No
Only.MentholYes	-0.126	0.882	-1.741	1.548	1.00	No

The polynomial variables are difficult to interpret in the same efficient manner as the continuous `logNMR`

or binary `Var`, therefore we produce plots to enhance our understanding.

Figure 2: Relationship between FTCD Score and Estimates

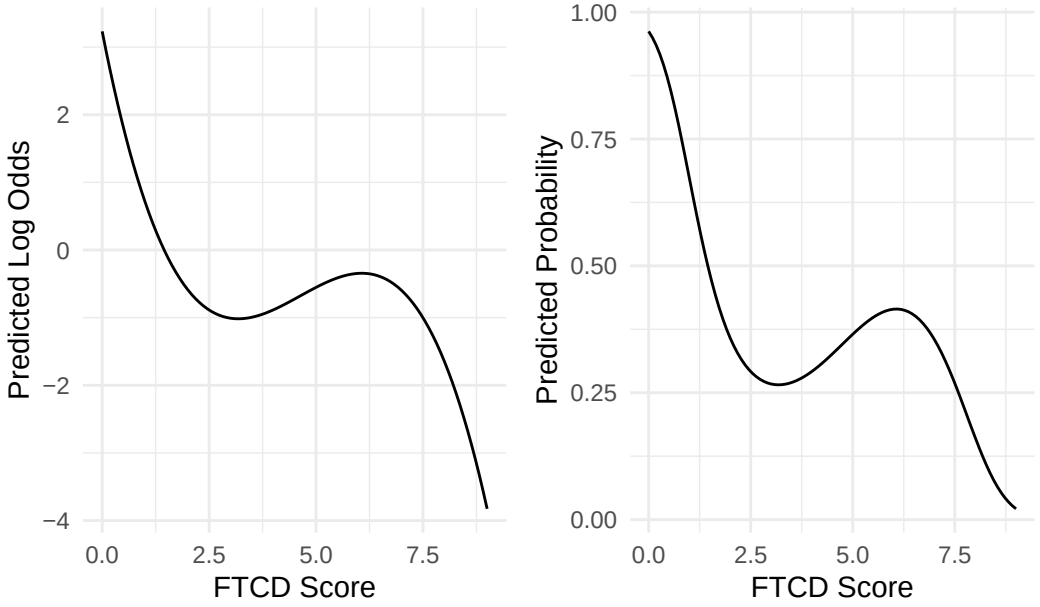
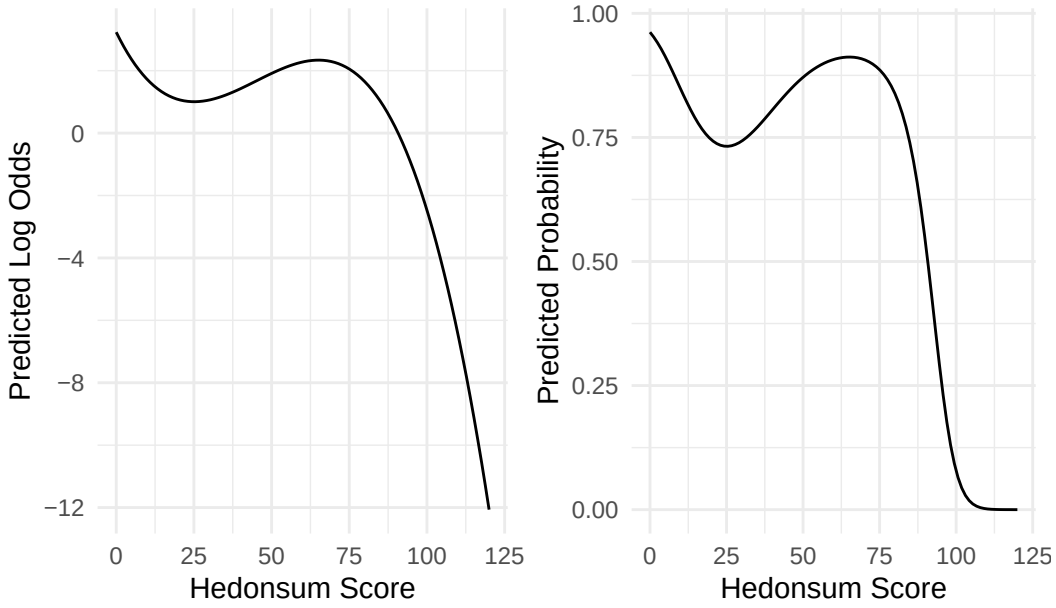
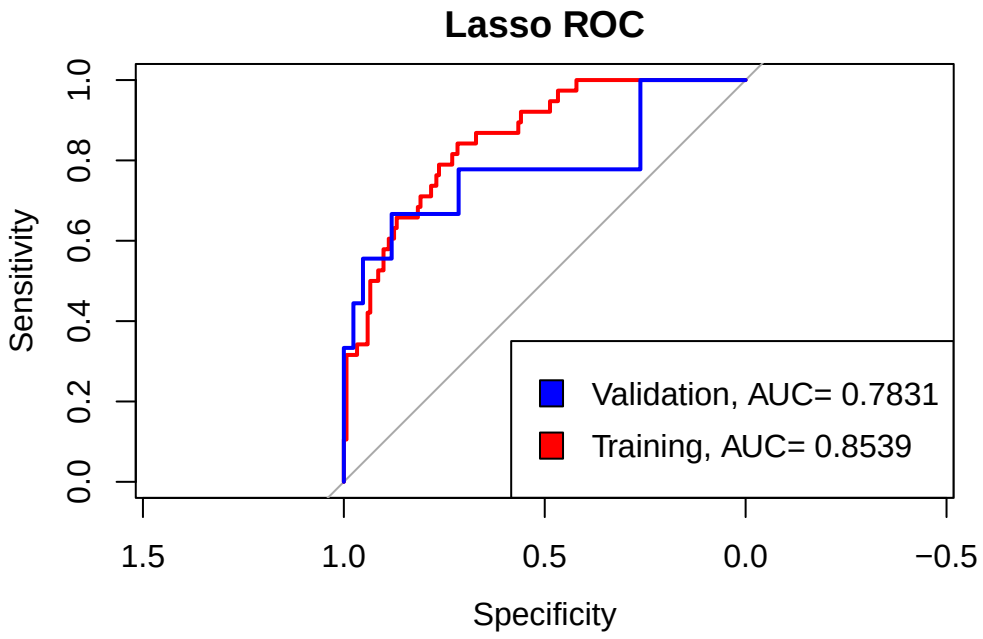


Figure 3: Relationship between Hedonsum Score and Estimates



Diagnostics



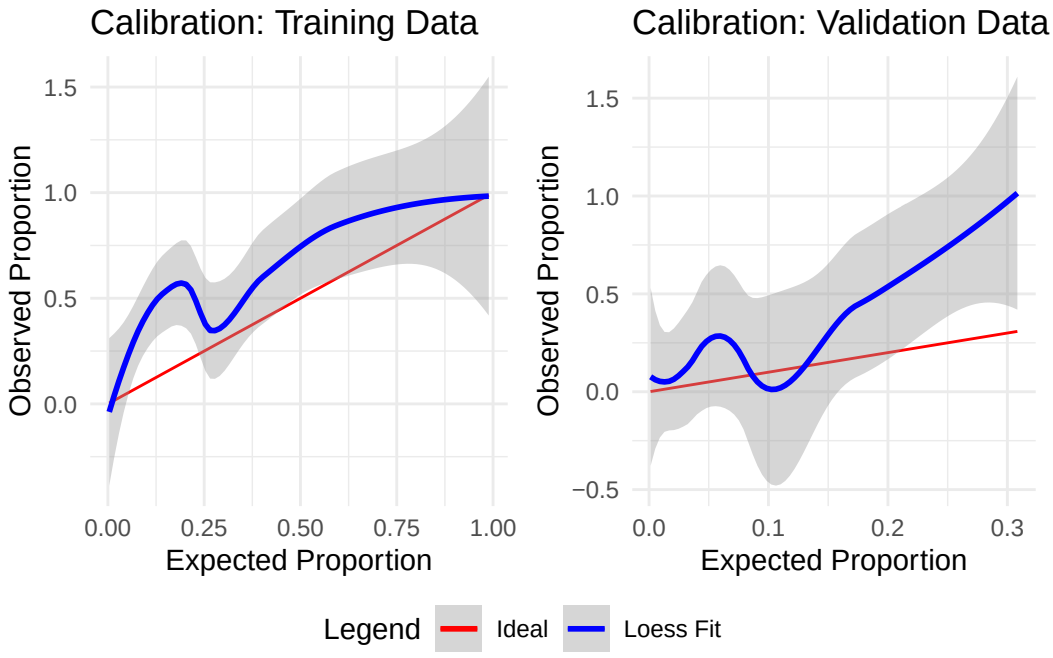
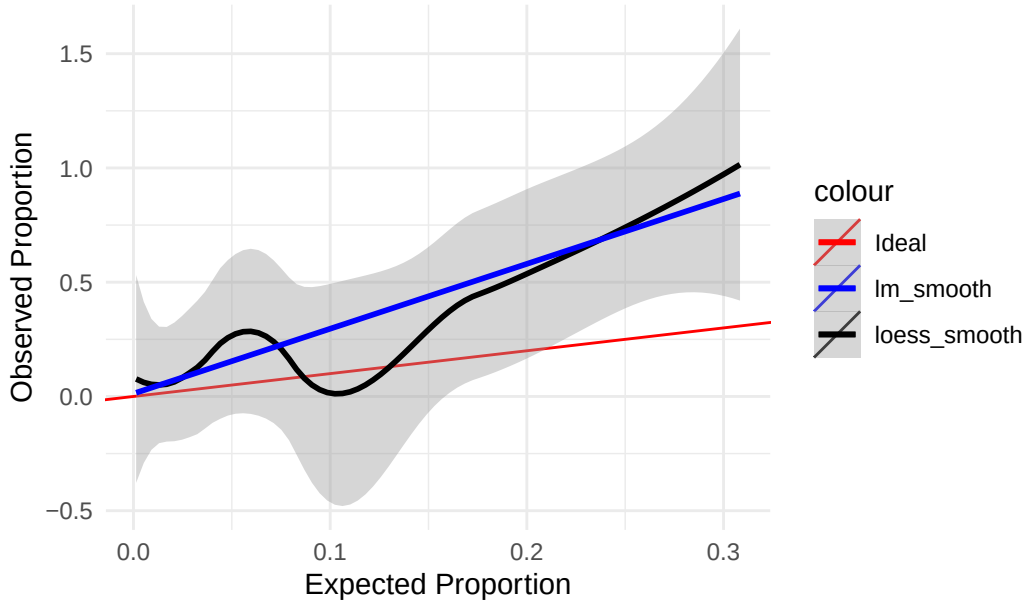
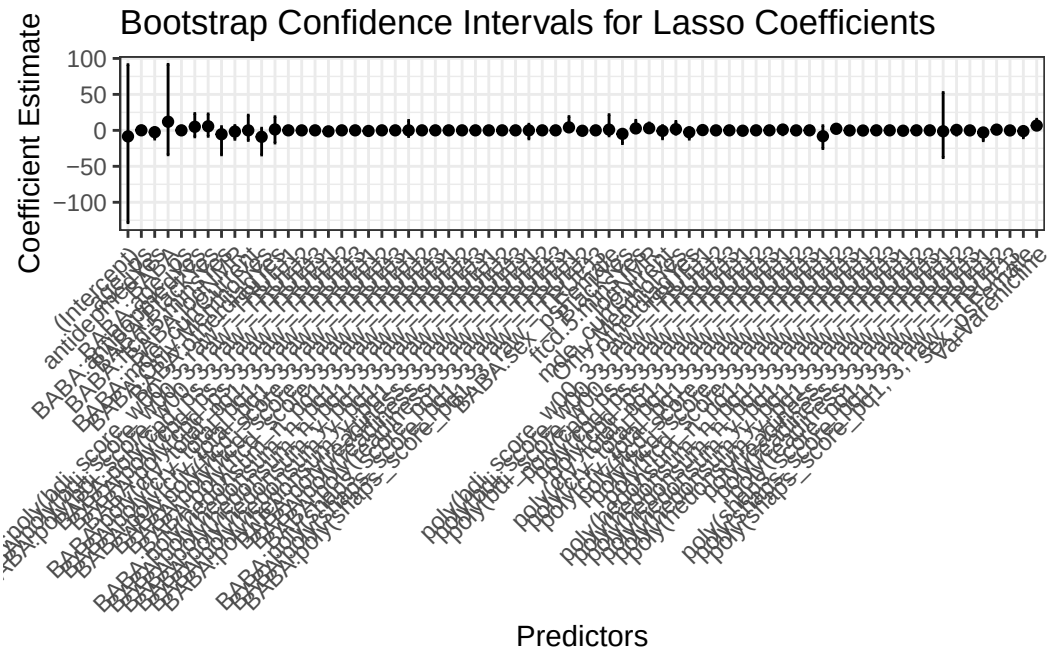


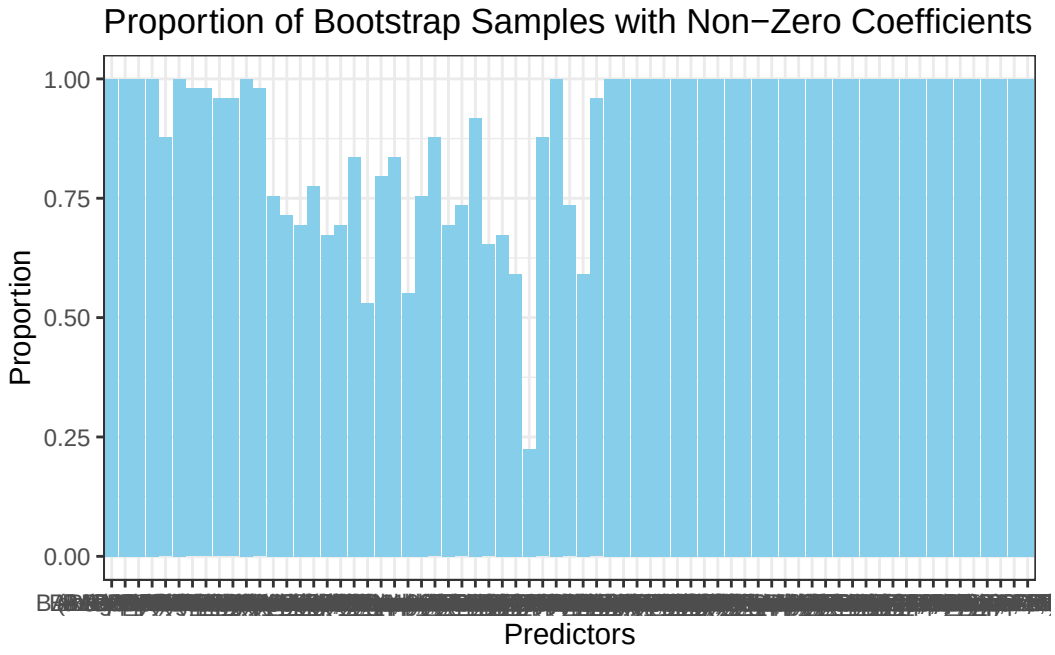
Figure 4: Lasso Calibration Plot on Validation Data

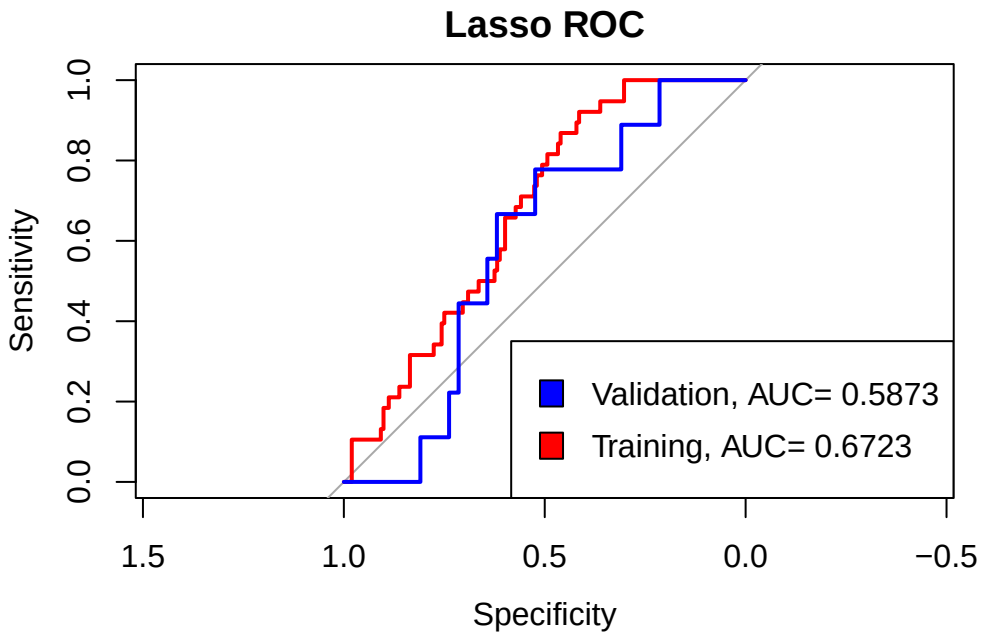


Baseline Variables as Moderators of BA

Results are presented for bootstrapped lasso regression on the outcome abstinence, with main effects included and unpenalized. Interactions of baseline covariates with `BA` are included and unpenalized, allowing the Lasso to select the interaction terms of consequence.




Proportion of Bootstrap Samples with Non−Zero Coefficients
1.00
0.75
0.50
0.25
0.00
Proportion
Predictors


Lasso ROC
Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
1.5
1.0
0.5
0.0
−0.5
Specificity
Validation, AUC= 0.5873
Training, AUC= 0.6723

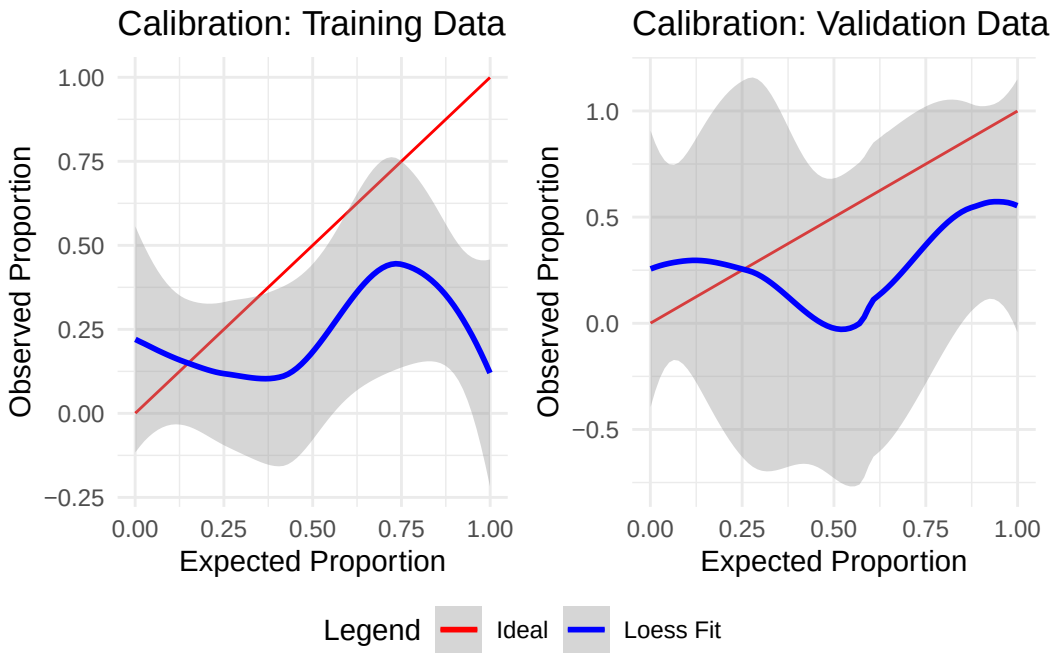
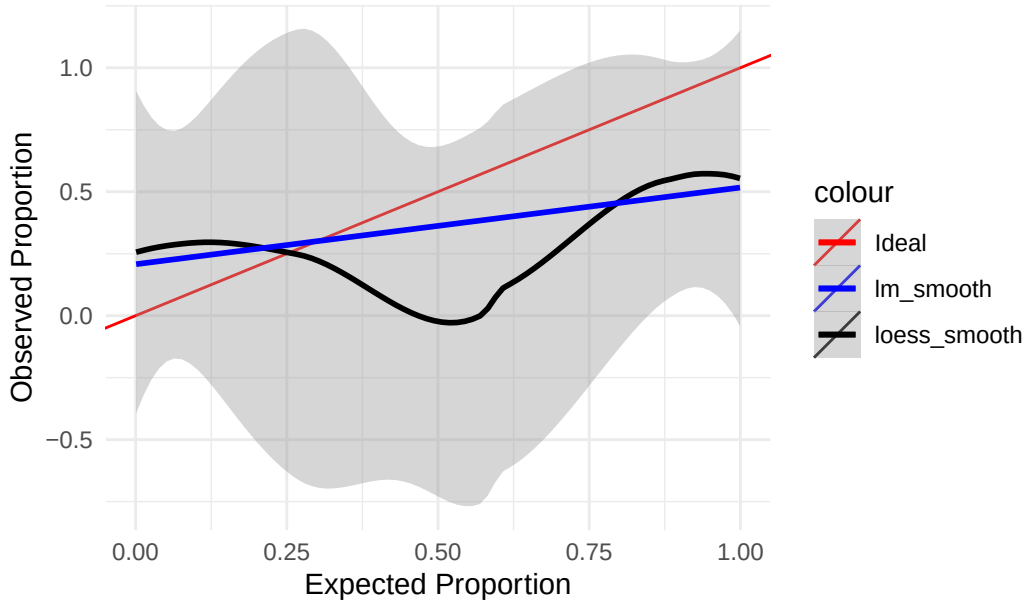


Figure 5: Lasso Calibration Plot on Validation Data



Conclusion

References

Anthenelli, Robert M, Neal L Benowitz, Robert West, Lisa St Aubin, Thomas McRae, David Lawrence, John Ascher, Cristina Russ, Alok Krishen, and A Eden Evins. 2016. "Neuropsychiatric Safety and Efficacy of Varenicline, Bupropion, and Nicotine Patch in Smokers with and Without Psychiatric Disorders (EAGLES): A Double-Blind, Randomised, Placebo-Controlled Clinical Trial." *The Lancet* 387 (10037): 2507–20.

Anthenelli, Robert M, Chad Morris, Tanya S Ramey, Sarah J Dubrava, Kostas Tsilkos, Cristina Russ, and Carla Yunis. 2013. "Effects of Varenicline on Smoking Cessation in Adults with Stably Treated Current or Past Major Depression: A Randomized Trial." *Annals of Internal Medicine* 159 (6): 390–400.

Beck, Aaron T, Robert A Steer, and Gregory Brown. 1996. "Beck Depression Inventory–II." *Psychological Assessment*.

Breslau, Naomi, M Marlyne Kilbey, and Patricia Andreski. 1992. "Nicotine Withdrawal Symptoms and Psychiatric Disorders: Findings from an Epidemiologic Study of Young Adults." *The American Journal of Psychiatry* 149 (4): 464–69.

Cinciripini, Paul M, Jason D Robinson, Maher Karam-Hage, Jennifer A Minnix, Cho Lam, Francesco Versace, Victoria L Brown, Jeffrey M Engelmann, and David W Wetter. 2013. "Effects of Varenicline and Bupropion Sustained-Release Use Plus Intensive Smoking Cessation Counseling on Prolonged Abstinence from Smoking and on Depression, Negative Affect, and Other Symptoms of Nicotine Withdrawal." *JAMA Psychiatry* 70 (5): 522–33.

Cook, Jessica, Bonnie Spring, Dennis McChargue, and Neal Doran. 2010. "Effects of Anhedonia on Days to Relapse Among Smokers with a History of Depression: A Brief Report." *Nicotine & Tobacco Research* 12 (9): 978–82.

Cuijpers, Pim, Annemieke Van Straten, and Lisanne Warmerdam. 2007. "Behavioral Activation Treatments of Depression: A Meta-Analysis." *Clinical Psychology Review* 27 (3): 318–26.

Dimidjian, Sona, Manuel Barrera Jr, Christopher Martell, Ricardo F Muñoz, and Peter M Lewinsohn. 2011. "The Origins and Current Status of Behavioral Activation Treatments for Depression." *Annual Review of Clinical Psychology* 7 (1): 1–38.

Evins, A Eden, Melissa A Culhane, Jonathan E Alpert, Joel Pava, Bruce S Liese, Amy Farabaugh, and Maurizio Fava. 2008. "A Controlled Trial of Bupropion Added to Nicotine Patch and Behavioral Therapy for Smoking Cessation in Adults with Unipolar Depressive Disorders." *Journal of Clinical Psychopharmacology* 28 (6): 660–66.

Fagerström, Karl. 2011. "Determinants of Tobacco Use and Renaming the FTND to the Fagerström Test for Cigarette Dependence." *Nicotine & Tobacco Research* 14 (1): 75–78.

Hall, Sharon M, Janice Y Tsoh, Judith J Prochaska, Stuart Eisendrath, Joseph S Rossi, Colleen A Redding, Amy B Rosen, Marc Meisner, Gary L Humfleet, and Julie A Gorecki. 2006. "Treatment for Cigarette Smoking Among Depressed Mental Health Outpatients: A Randomized Clinical Trial." *American Journal of Public Health* 96 (10): 1808–14.

Han, Beth, Nora D Volkow, Carlos Blanco, Douglas Tipperman, Emily B Einstein, and Wilson M Compton. 2022. "Trends in Prevalence of Cigarette Smoking Among US Adults with Major Depression or Substance Use Disorders, 2006-2019." *Jama* 327 (16): 1566–76.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2009. "An Introduction to Statistical Learning."

Hastie, Trevor, Robert Tibshirani, and Martin Wainwright. 2015. "Statistical Learning with Sparsity." *Monographs on Statistics and Applied Probability* 143 (143): 8.

Hazimeh, Hussein, and Rahul Mazumder. 2020. "Fast Best Subset Selection: Coordinate Descent and Local Combinatorial Optimization Algorithms." *Operations Research* 68 (5): 1517–37.

Hitsman, Brian, Belinda Borrelli, Dennis E McChargue, Bonnie Spring, and Raymond Niaura. 2003. "History of Depression and Smoking Cessation Outcome: A Meta-Analysis." *Journal of Consulting*

and Clinical Psychology 71 (4): 657.

Hitsman, Brian, Lee Hogarth, Li-Jung Tseng, Jordan C Teige, William G Shadel, Dana Britt DiBenedetti, Spencer Danto, Theodore C Lee, Lawrence H Price, and Raymond Niaura. 2013. "Dissociable Effect of Acute Varenicline on Tonic Versus Cue-Provoked Craving in Non-Treatment-Motivated Heavy Smokers." *Drug and Alcohol Dependence* 130 (1-3): 135–41.

Hitsman, Brian, George D Papandonatos, Jacqueline K Gollan, Mark D Huffman, Raymond Niaura, David C Mohr, Anna K Veluz-Wilkins, et al. 2023. "Efficacy and Safety of Combination Behavioral Activation for Smoking Cessation and Varenicline for Treating Tobacco Dependence Among Individuals with Current or Past Major Depressive Disorder: A 2× 2 Factorial, Randomized, Placebo-Controlled Trial." *Addiction* 118 (9): 1710–25.

Hitsman, Brian, George D Papandonatos, Dennis E McChargue, Andrew DeMott, María José Herrera, Bonnie Spring, Belinda Borrelli, and Raymond Niaura. 2013. "Past Major Depression and Smoking Cessation Outcome: A Systematic Review and Meta-Analysis Update." *Addiction* 108 (2): 294–306.

Hopko, Derek R, CW Lejuez, Kenneth J Ruggiero, and Georg H Eifert. 2003. "Contemporary Behavioral Activation Treatments for Depression: Procedures, Principles, and Progress." *Clinical Psychology Review* 23 (5): 699–717.

Leventhal, Adam M, Andrew J Waters, Christopher W Kahler, Lara A Ray, and Steve Sussman. 2009. "Relations Between Anhedonia and Smoking Motivation." *Nicotine & Tobacco Research* 11 (9): 1047–54.

Lyons, Michael, Brian Hitsman, Hong Xian, Matthew S Panizzon, Beth A Jerskey, Susan Santangelo, Michael D Grant, et al. 2008. "A Twin Study of Smoking, Nicotine Dependence, and Major Depression in Men." *Nicotine & Tobacco Research* 10 (1): 97–108.

MacPherson, Laura, Matthew T Tull, Alexis K Matusiewicz, Samantha Rodman, David R Strong, Christopher W Kahler, Derek R Hopko, Michael J Zvolensky, Richard A Brown, and CW3108050 Lejuez. 2010. "Randomized Controlled Trial of Behavioral Activation Smoking Cessation Treatment for Smokers with Elevated Depressive Symptoms." *Journal of Consulting and Clinical Psychology* 78 (1): 55.

MacPhillamy, Douglas J, and Peter M Lewinsohn. 1982. "The Pleasant Events Schedule: Studies on Reliability, Validity, and Scale Intercorrelation." *Journal of Consulting and Clinical Psychology* 50 (3): 363.

McClure, Erin A, Ryan G Vandrey, Matthew W Johnson, and Maxine L Stitzer. 2012. "Effects of Varenicline on Abstinence and Smoking Reward Following a Programmed Lapse." *Nicotine & Tobacco Research* 15 (1): 139–48.

Meinshausen, Nicolai. 2007. "Relaxed Lasso." *Computational Statistics & Data Analysis* 52 (1): 374–93.

Minami, Haruka, Shadi Nahvi, Julia H Arnsten, Hannah R Brinkman, Monica Rivera-Mindt, David W Wetter, Erika Litvin Bloom, et al. 2022. "A Pilot Randomized Controlled Trial of Smartphone-Assisted Mindfulness-Based Intervention with Contingency Management for Smokers with Mood Disorders." *Experimental and Clinical Psychopharmacology* 30 (5): 653.

Patterson, Freda, Christopher Jepson, Andrew A Strasser, James Loughead, Kenneth A Perkins, Ruben C Gur, Joseph M Frey, Steven Siegel, and Caryn Lerman. 2009. "Varenicline Improves Mood and Cognition During Smoking Abstinence." *Biological Psychiatry* 65 (2): 144–49.

Paul, Alice. 2024. "PHP 2550 - Variable Selection."

Perkins, Kenneth A, Melissa Mercincavage, Carolyn A Fonte, and Caryn Lerman. 2010. "Varenicline's Effects on Acute Smoking Behavior and Reward and Their Association with Subsequent Abstinence." *Psychopharmacology* 210: 45–51.

Siegel, Scott D, Caryn Lerman, Alex Flitter, and Robert A Schnoll. 2020. "The Use of the Nicotine Metabolite Ratio as a Biomarker to Personalize Smoking Cessation Treatment: Current Evidence

and Future Directions." *Cancer Prevention Research* 13 (3): 261–72.

Smith, Philip H, Mohammad Chhipa, Josef Bystrik, Jordan Roy, Renee D Goodwin, and Sherry A McKee. 2020. "Cigarette Smoking Among Those with Mental Disorders in the US Population: 2012–2013 Update." *Tobacco Control* 29 (1): 29–35.

Snaith, R Philip, Max Hamilton, S Morley, A Humayan, D Hargreaves, and P Trigwell. 1995. "A Scale for the Assessment of Hedonic Tone the Snaith–Hamilton Pleasure Scale." *The British Journal of Psychiatry* 167 (1): 99–103.

Sofuoglu, Mehmet, Aryeh I Herman, Marc Mooney, and Andrew J Waters. 2009. "Varenicline Attenuates Some of the Subjective and Physiological Effects of Intravenous Nicotine in Humans." *Psychopharmacology* 207: 153–62.

Spring, Bonnie, Regina Pingitore, and Dennis E McChargue. 2003. "Reward Value of Cigarette Smoking for Comparably Heavy Smoking Schizophrenic, Depressed, and Nonpatient Smokers." *American Journal of Psychiatry* 160 (2): 316–22.

Talukder, Saki Rubaiya, Julia M Lappin, Veronica Boland, Hayden McRobbie, and Ryan James Courtney. 2023. "Inequity in Smoking Cessation Clinical Trials Testing Pharmacotherapies: Exclusion of Smokers with Mental Health Disorders." *Tobacco Control* 32 (4): 489–96.

Taylor, Gemma MJ, Taha Itani, Kyla H Thomas, Dheeraj Rai, Tim Jones, Frank Windmeijer, Richard M Martin, Marcus R Munafò, Neil M Davies, and Amy E Taylor. 2020. "Prescribing Prevalence, Effectiveness, and Mental Health Safety of Smoking Cessation Medicines in Patients with Mental Disorders." *Nicotine and Tobacco Research* 22 (1): 48–57.

Thorsteinsson, Haraldur S, J Christian Gillin, Christi A Patten, Shahrokh Golshan, Laura D Sutton, Sean Drummond, Camellia P Clark, John Kelsoe, and Mark Rapaport. 2001. "The Effects of Transdermal Nicotine Therapy for Smoking Cessation on Depressive Symptoms in Patients with Major Depression." *Neuropsychopharmacology* 24 (4): 350–58.

Weinberger, Andrea H, Michael O Chaiton, Jiaqi Zhu, Melanie M Wall, Deborah S Hasin, and Renee D Goodwin. 2020. "Trends in the Prevalence of Current, Daily, and Nondaily Cigarette Smoking and Quit Ratios by Depression Status in the US: 2005–2017." *American Journal of Preventive Medicine* 58 (5): 691–98.

Weinberger, Andrea H, Rani A Desai, and Sherry A McKee. 2010. "Nicotine Withdrawal in US Smokers with Current Mood, Anxiety, Alcohol Use, and Substance Use Disorders." *Drug and Alcohol Dependence* 108 (1-2): 7–12.

West, Robert, Christine L Baker, Joseph C Cappelleri, and Andrew G Bushmakin. 2008. "Effect of Varenicline and Bupropion SR on Craving, Nicotine Withdrawal Symptoms, and Rewarding Effects of Smoking During a Quit Attempt." *Psychopharmacology* 197: 371–77.

Appendix I: Script Code

```
# --- Preamble ---
# Date of last update: Nov. 11, 2024
# R Version: 4.3.1
# Package Versions:
#   tidyverse: 2.0.0
#   knitr: 1.45
#   kableExtra: 1.3.4
#   ggplot2: 3.4.3
#   naniar 1.0.0
#   visdat 0.6.0
#   car 3.1-2
#   lme4 1.1-34
#   ggpubr 0.6.0
#   library(glmnet)
#   library(mice)
#   library(caret)


setwd("~/GitHub/smoking_cessation_proj")

# Knitr Engine Setup
knitr::opts_chunk$set(message=F,
                      warning=F,
                      error=F,
                      echo=F,
                      fig.pos = "H" ,
                      fig.align = 'center')

# Packages
options(kableExtra.latex.load_packages = FALSE) # Required to avoid floatrow error
library(knitr)
library(kableExtra)
library(ggplot2)
library(naniar) # For mcar_test()
library(visdat) # For vis_dat()
library(tidyverse)
library(car) # For qqPlot(), vif()
library(lme4)
library(lmerTest) # Satterthwaite approximation for computing p-values on lme4
library(ggpubr)
library(glmnet)
library(mice)
library(caret)
library(pROC)

#library(Hmisc)
#library(vcd)
```

```r
source("_helpers.R")

data <- read.csv("data/project2.csv", comment.char="#")




# Convert variables to appropriate types and levels
data <- data %>%
  mutate(
    abst = factor(abst, levels = c(0, 1), labels = c("No", "Yes")), # This should be a factor for
    Var = factor(Var, levels = c(0, 1),
                          labels = c("Placebo", "Varenicline")),
    BA = factor(BA, levels = c(0, 1), labels = c("Standard", "BA")),
    age_ps = as.numeric(age_ps),
    sex_ps = factor(sex_ps, levels = c(1, 2), labels = c("Male", "Female")),
    NHW = factor(NHW, levels = c(0, 1), labels = c("No", "Yes")),
    Black = factor(Black, levels = c(0, 1), labels = c("No", "Yes")),
    Hisp = factor(Hisp, levels = c(0, 1), labels = c("No", "Yes")),
    inc = factor(inc, levels = 1:5,
                 labels = c("Less than $20,000",
                            "$20,000-35,000",
                            "$35,001-50,000",
                            "$50,001-75,000",
                            "More than $75,000"), ordered = TRUE),
    edu = factor(edu, levels = 1:5,
                 labels = c("Grade School",
                            "Some High School",
                            "High School Graduate/GED",
                            "Some College/Technical School",
                            "College Graduate"), ordered = TRUE),
    ftcd_score = as.integer(ftcd_score), # This variable is actually ordinal not numeric, so this
    ftcd.5.mins = factor(ftcd.5.mins, levels = c(0, 1), labels = c("No", "Yes")),
    bdi_score_w00 = as.integer(bdi_score_w00), # This variable is actually ordinal not numeric, s
    cpd_ps = as.integer(cpd_ps),
    crv_total_pq1 = as.integer(crv_total_pq1), # This variable is actually ordinal not numeric, s
    hedonsum_n_pq1 = as.integer(hedonsum_n_pq1), # This variable is actually ordinal not numeric,
    hedonsum_y_pq1 = as.integer(hedonsum_y_pq1), # This variable is actually ordinal not numeric,
    shaps_score_pq1 = as.integer(shaps_score_pq1), # This variable is actually ordinal not numeri
    otherdiag = factor(otherdiag, levels = c(0, 1), labels = c("No", "Yes")),
    antidepmed = factor(antidepmed, levels = c(0, 1), labels = c("No", "Yes")),
    mde_curr = factor(mde_curr, levels = c(0, 1), labels = c("Past", "Current")),
    NMR = as.numeric(NMR),
    Only.Menthol = factor(Only.Menthol, levels = c(0, 1), labels = c("No", "Yes")),
    readiness = as.integer(readiness) # This variable is actually ordinal not numeric, so this is
  )
```

```r
# Note: using tbl-cap rather than caption= in kbl() breaks repeated title is span page


# Table 1
summary_table(data[-1]) %>%
  kbl(#caption = "Summary of Variables", # Conflicts with Quarto referencing
      booktabs = T,
      longtable = T, # LONGTABLE
      escape = T,
      align = "c") %>%
  column_spec(1, width="2.2cm", latex_valign = "m") %>%
  column_spec(2, width="1.3cm", latex_valign = "m") %>%
  column_spec(3, width="4.5cm", latex_valign = "m") %>%
  column_spec(4, width="1.0cm", latex_valign = "m") %>%
  column_spec(5, width="1.0cm", latex_valign = "m") %>%
  column_spec(6, width="1.25cm", latex_valign = "m") %>%
  column_spec(7, width="1.25cm", latex_valign = "m") %>%
  kable_styling(
    font_size = 8, # Added for LONGTABLE
    latex_options = c(#'HOLD_position', # Removed for LONGTABLE
    #'scale_down', # Removed for LONGTABLE
    "repeat_header",  # Added for LONGTABLE
    'striped'),
    full_width = F, # Note: TRUE does not work with LONGTABLE
    position = 'center'# Added for LONGTABLE
  ) %>%
  footnote(general = "Shapiro-Wilk test for normality;
            Grubb's test for outliers.", escape = F)


# Table 2
summary_table(data[-1], stratify_var = "abst") %>%
  kbl(
      booktabs = T,
      longtable = T, # LONGTABLE
      escape = T,
      align = "c") %>%
  column_spec(1, width="2.21cm", latex_valign = "m") %>%
  column_spec(2, width="4cm", latex_valign = "m") %>%
  column_spec(3, width="4cm", latex_valign = "m") %>%
  column_spec(4, width=".35cm", latex_valign = "m") %>%
  kable_styling(
    font_size = 7.6, # Added for LONGTABLE
    latex_options = c(#'HOLD_position', # Removed for LONGTABLE
    #'scale_down', # Removed for LONGTABLE
    "repeat_header",  # Added for LONGTABLE
    'striped'),
```

```
    full_width = F, # Note: TRUE does not work with LONGTABLE
    position = 'center'# Added for LONGTABLE
  ) %>%
  footnote(general = "ns = P > 0.05,
           * = P $\\\\leq$ 0.05,
           ** = P $\\\\leq$ 0.01,
           *** = P $\\\\leq$ 0.001,
           **** = P $\\\\leq$ 0.0001
           ", escape = F) %>%
  footnote(general = "Kruskal-Wallis test for continuous variables,
           Chi-Square test for categorical variables.
           Bonferroni correction applied.", escape = F)

# Table 2
summary_table(data[-1], stratify_var = "BA") %>%
  kbl(
      booktabs = T,
      longtable = T, # LONGTABLE
      escape = T,
      align = "c") %>%
  column_spec(1, width="2.21cm", latex_valign = "m") %>%
  column_spec(2, width="4cm", latex_valign = "m") %>%
  column_spec(3, width="4cm", latex_valign = "m") %>%
  column_spec(4, width=".35cm", latex_valign = "m") %>%
  kable_styling(
    font_size = 7.6, # Added for LONGTABLE
    latex_options = c(#'HOLD_position', # Removed for LONGTABLE
    #'scale_down', # Removed for LONGTABLE
    "repeat_header",  # Added for LONGTABLE
    'striped'),
    full_width = F, # Note: TRUE does not work with LONGTABLE
    position = 'center'# Added for LONGTABLE
  ) %>%
  footnote(general = "ns = P > 0.05,
           * = P $\\\\leq$ 0.05,
           ** = P $\\\\leq$ 0.01,
           *** = P $\\\\leq$ 0.001,
           **** = P $\\\\leq$ 0.0001
           ", escape = F) %>%
  footnote(general = "Kruskal-Wallis test for continuous variables,
           Chi-Square test for categorical variables.
           Bonferroni correction applied.", escape = F)

# Table 4
summary_table(data[-1], stratify_var = "Var") %>%
  kbl(
      booktabs = T,
      longtable = T, # LONGTABLE
      escape = T,
```

```
      align = "c") %>%
  column_spec(1, width="2.21cm", latex_valign = "m") %>%
  column_spec(2, width="4cm", latex_valign = "m") %>%
  column_spec(3, width="4cm", latex_valign = "m") %>%
  column_spec(4, width=".35cm", latex_valign = "m") %>%
  kable_styling(
    font_size = 7.6, # Added for LONGTABLE
    latex_options = c(#'HOLD_position', # Removed for LONGTABLE
    #'scale_down', # Removed for LONGTABLE
    "repeat_header",  # Added for LONGTABLE
    'striped'),
    full_width = F, # Note: TRUE does not work with LONGTABLE
    position = 'center'# Added for LONGTABLE
  ) %>%
  footnote(general = "ns = P > 0.05,
           * = P $\\\\leq$ 0.05,
           ** = P $\\\\leq$ 0.01,
           *** = P $\\\\leq$ 0.001,
           **** = P $\\\\leq$ 0.0001
           ", escape = F) %>%
  footnote(general = "Kruskal-Wallis test for continuous variables,
           Chi-Square test for categorical variables.
           Bonferroni correction applied.", escape = F)

# Histograms of variable distributions

# Identify numeric columns
numeric_cols <- sapply(data, is.numeric)

# Create histograms for each numeric column
for (col in names(data)[numeric_cols]) {
  hist(data[[col]], main = paste("Histogram of", col), xlab = col)
}

# Generate psuedo correlation matrix
psuedo_cor_matrix <- psuedo_cor_mat(data[-1])

# Plot the heatmap (with variable names and values)
ggplot(melt(psuedo_cor_matrix), aes(x = Var2, y = Var1, fill = value)) +
  geom_tile(color = "white") +
  geom_text(aes(label = sprintf("%.2f", value)), size = 2.8, color = "black") +
  scale_fill_gradient2(low = "blue",
                       high = "red",
                       mid = "white",
                       midpoint = 0,
                       limits = c(-1, 1),
                       space = "Lab",
                       name = "Correlation") +
  theme_minimal() +
```

```
  labs(x = "", y = "") +
  theme(axis.title.x = element_blank(),
        axis.text.x = element_text(angle = 45, hjust = 1, size = 9),
        axis.ticks.x = element_blank(),
        axis.title.y = element_blank(),
        axis.text.y = element_text(angle = 45, hjust = 1, size = 9),
        legend.position="none") +
  coord_fixed(ratio = .55)


# 1. Missing by Variable
missing_by_variable <- data %>%
  summarise(across(everything(), ~ sum(is.na(.)))) %>%
  pivot_longer(cols = everything(), names_to = "Variable", values_to = "Missing_Count") %>%
  mutate(Missing_Percentage = round((Missing_Count / nrow(data)) * 100, 2))

# Display Missing by Variable
cat("\n### Missing Data by Variable\n\n")
kbl(missing_by_variable, caption = "Missing Data by Variable")

# 2. Missing by Record
missing_by_record <- data %>%
  mutate(Row = row_number()) %>%
  mutate(Missing_Count = rowSums(is.na(across(everything())))) %>%
  select(Row, Missing_Count)

# Display Missing by Record
cat("\n\n### Missing Data by Record\n\n")
kbl(missing_by_record, caption = "Missing Data by Record")

# Display Missing by Record
cat("\n\n### Missing Data by Record\n\n")
kable(missing_by_record, caption = "Missing Data by Record")

# 3. Total Complete Cases
total_complete_cases <- data %>%
  summarise(Complete_Cases = sum(complete.cases(.))) %>%
  pull(Complete_Cases)

# Display Total Complete Cases
cat("\n\n### Total Complete Cases\n\n")
cat(total_complete_cases, "complete cases out of", nrow(data), "records.\n")
# Test for MCAR
naniar::mcar_test(data)


# prep data for analysis
data <- data %>%
  mutate(abst = as.numeric(abst) - 1,
```

```
    edu = factor(recode_factor(edu,
                               "Grade School" = "High School or Less",
                               "Some High School" = "High School or Less",
                               "High School Graduate/GED" = "High School or Less"),
                 levels = c("High School or Less",
                            "Some College/Technical School",
                            "College Graduate"),
                 ordered = TRUE),
    inc = factor(recode_factor(inc,
                               "$50,001-75,000" = "More than $50,000",
                               "More than $75,000" = "More than $50,000"),
                 levels = c("Less than $20,000",
                            "$20,000-35,000",
                            "$35,001-50,000",
                            "More than $50,000"),
                 ordered = TRUE)
        )

data <- data %>%
  mutate(logNMR = log(NMR)) %>%
  select(-c(NMR, Hisp, NHW))

# --- Validation / Train split ---

set.seed(123)
split_indices <- createDataPartition(y = data$abst, p = 0.8, list = FALSE)

# split data training / validation sets
train_data <- data[split_indices, ]
validation_data <- data[-split_indices, ]
data <- train_data
#GOOD; Q1; baseline

# Def All Params
B <- 50      # num of bootstraps
m <- 5      # num of multiple imputations per bootstrap
k <- 5      # num of cv folds for cv.glmnet
poly_degree <- 3  # degree for polynomial terms

# remove id
data_complete <- data %>% select(-id)

# Flag predictor vars
# exclude 'abst' from predictors
predictor_vars <- setdiff(names(data_complete), c("abst"))

# Flag trt vars
treatment_vars <- c("BA", "Var")
baseline_vars <- setdiff(predictor_vars, treatment_vars)
```

```r
# Flag numeric and int baseline vars
num_baseline_vars <- baseline_vars[sapply(data_complete[baseline_vars],
                                          function(x) is.numeric(x) & !is.integer(x))]

non_binary_factor_vars <- baseline_vars[sapply(data_complete[baseline_vars],
                                               function(x) is.integer(x))]
# Flag Binary baseline vars
binary_vars <- baseline_vars[sapply(data_complete[baseline_vars],
                                    function(x) length(unique(na.omit(x))) == 2)]

# Flag factor vars and store levels
factor_vars <- names(data_complete)[sapply(data_complete, is.factor)]
levels_list <- lapply(data_complete[, factor_vars, drop = FALSE], levels)

# --- Build Formula ---
# Main Effects
main_effects <- paste("BA + Var +",
                      paste(c(
                        paste0("poly(", non_binary_factor_vars,
                               ", ", poly_degree, ", raw=TRUE)"),
                        num_baseline_vars,
                        binary_vars
                      ), collapse = " + "))



# Full Model Formula
full_formula <- as.formula(paste("abst ~", main_effects))

# fix factor levels in the entire dataset (ensure consistency) BUGFIX
if (length(factor_vars) > 0) {
  for (var in factor_vars) {
    data_complete[[var]] <- factor(data_complete[[var]], levels = levels_list[[var]])
  }
}

# Gen master model matrix
master_model_matrix <- model.matrix(full_formula, data = data_complete)
master_coef_names <- colnames(master_model_matrix)
total_coeffs <- length(master_coef_names)

# init storage for coefs
coef_matrix <- matrix(NA, nrow = B, ncol = total_coeffs)
colnames(coef_matrix) <- master_coef_names

# init object for nonzero coef indicator
nonzero_matrix <- matrix(0, nrow = B, ncol = total_coeffs)
colnames(nonzero_matrix) <- master_coef_names
```

```r
# --- Bootstrap Loop ---
set.seed(1)

for (b in 1:B) {
  # Print progress
  cat("Bootstrap iteration:", b, "of", B, "\n")

  # Stratified resampling with replacement (maintain class distribution in 'abst')
  bootstrap_sample <- data_complete %>%
    group_by(abst) %>%
    sample_frac(size = 1, replace = TRUE) %>%
    ungroup()

  # impute data
  predictor_matrix <- make.predictorMatrix(bootstrap_sample)
  predictor_matrix[, "abst"] <- 0  # dont use 'abst' to predict others
  predictor_matrix["abst", ] <- 0  # dont impute 'abst' (not necessary bc no missing?)
  imp <- mice(bootstrap_sample, m = m, printFlag = FALSE, seed = b,
              predictorMatrix = predictor_matrix, method = 'pmm')

  # Stack imputations into one dataset
  stacked_data <- complete(imp, action = "long", include = FALSE)
  stacked_data <- stacked_data %>% dplyr::select(-.id, -.imp) # remove '.id' / '.imp'

  # fix factor levels in the entire dataset (ensure consistency) BUGFIX
  if (length(factor_vars) > 0) {
    for (var in factor_vars) {
      stacked_data[[var]] <- factor(stacked_data[[var]], levels = levels_list[[var]])
    }
  }

  # boot model matrix and X / y
  model_data_full <- model.matrix(full_formula, data = stacked_data)
  x <- model_data_full[, -1] # remove intercept for glmnet
  y_impute <- stacked_data$abst # response var for glmnet

  # Remove predictors with zero variance BUGFIX
  zero_var_cols <- apply(x, 2, function(x) var(x) == 0)
  if (any(zero_var_cols)) {
    x <- x[, !zero_var_cols]
    current_coef_names <- colnames(x)
  } else {
    current_coef_names <- colnames(x)}

  # align columns w/ master_coef_names excluding intercept BUGFIX
  master_coef_names_no_intercept <- master_coef_names[-1]  # now exclude intercept
  missing_cols <- setdiff(master_coef_names_no_intercept, current_coef_names)
```

```r
if (length(missing_cols) > 0) {
  zeros_matrix <- matrix(0, nrow = nrow(x), ncol = length(missing_cols))
  colnames(zeros_matrix) <- missing_cols
  x <- cbind(x, zeros_matrix)
}

# rm extra columns not in master_coef_names_no_intercept BUGFIX
extra_cols <- setdiff(current_coef_names, master_coef_names_no_intercept)
if (length(extra_cols) > 0) {
  x <- x[, !(colnames(x) %in% extra_cols)]
}

# reorder x colums to match master_coef_names_no_intercept BUGFIX
x <- x[, master_coef_names_no_intercept]



# Run cv.glmnet
cv_lasso <- tryCatch({
  cv.glmnet(
    x = x,
    y = y_impute,
    family = "binomial",
    alpha = 1,
    standardize = TRUE,
    relax = FALSE
  )
}, error = function(e) {
  cat("Error in cv.glmnet for bootstrap", b, ":", e$message, "\n")
  return(NULL)
})

if (is.null(cv_lasso)) {
  next  # skip/move to next bootstrap iteration
}

# Fit lasso
lasso_fit <- glmnet(
  x = x,
  y = y_impute,
  family = "binomial",
  alpha = 1,
  lambda = cv_lasso$lambda.min,
  standardize = TRUE,
  relax = FALSE
)


# grab coefs including intercept
```

```
  coef_values <- as.vector(coef(lasso_fit))
  coef_names <- rownames(coef(lasso_fit))
  names(coef_values) <- coef_names

  # init object to store coefs names
  aligned_coef_values <- setNames(rep(0, length(master_coef_names)), master_coef_names)

  # match coefs to the aligned vector
  matched <- names(coef_values) %in% master_coef_names
  aligned_coef_values[names(coef_values)[matched]] <- coef_values[matched]

  # store coefs
  coef_matrix[b, ] <- aligned_coef_values

  # store nonzero indicators
  nonzero_matrix[b, ] <- ifelse(aligned_coef_values != 0, 1, 0)
}



saveRDS(coef_matrix, file = "baseline_coef.rds")
saveRDS(nonzero_matrix, file = "baseline_nonzero.rds")

coef_matrix <- readRDS(file = "baseline_coef.rds")
nonzero_matrix <- readRDS(file = "baseline_nonzero.rds")


# --- Pool coefs ---

# identify bad runs
bad_runs <- apply(coef_matrix, 1, function(x) all(is.na(x)) | all(x == 0)) &
            apply(nonzero_matrix, 1, function(x) all(is.na(x)) | all(x == 0))

# rm bad runs from objects
coef_matrix <- coef_matrix[!bad_runs, ]
nonzero_matrix <- nonzero_matrix[!bad_runs, ]

# make coef data frame
coef_mean <- colMeans(coef_matrix, na.rm = TRUE)
coef_median <- apply(coef_matrix, 2, median, na.rm = TRUE)
coef_lower <- apply(coef_matrix, 2, quantile, probs = 0.025, na.rm = TRUE)
coef_upper <- apply(coef_matrix, 2, quantile, probs = 0.975, na.rm = TRUE)
proportion_nonzero <- colSums(nonzero_matrix, na.rm = TRUE) / sum(nonzero_matrix[,1])

coef_ci <- data.frame(
  Predictor = colnames(coef_matrix),
  Mean = coef_mean,
  Median = coef_median,
  Lower_2.5 = coef_lower,
```

```
  Upper_97.5 = coef_upper,
  Proportion_Nonzero = proportion_nonzero
)




# plot the CIs
ggplot(coef_ci, aes(x = Predictor, y = Median)) +
  geom_point() +
  geom_errorbar(aes(ymin = Lower_2.5, ymax = Upper_97.5), width = 0.2) +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(
    title = "Bootstrap Confidence Intervals for Lasso Coefficients",
    y = "Coefficient Estimate",
    x = "Predictors"
  )

# plot nonzero prop
ggplot(coef_ci, aes(x = Predictor, y = Proportion_Nonzero)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  theme_bw() +
  labs(
    title = "Proportion of Bootstrap Samples with Non-Zero Coefficients",
    y = "Proportion",
    x = "Predictors"
  )




# --- Enforce Sparsity ---

# Note: this is essentially skipped in this regression run because only having
# main effects in the model allows for few enough parameters that we are not overfitting.

coef_matrix_sparse <- coef_matrix


coef_mean <- colMeans(coef_matrix_sparse, na.rm = TRUE)
coef_median <- apply(coef_matrix_sparse, 2, median, na.rm = TRUE)
coef_lower <- apply(coef_matrix_sparse, 2, quantile, probs = 0.025, na.rm = TRUE)
coef_upper <- apply(coef_matrix_sparse, 2, quantile, probs = 0.975, na.rm = TRUE)
proportion_nonzero <- colSums(nonzero_matrix, na.rm = TRUE) / sum(nonzero_matrix[,1])

coef_ci_sparse <- data.frame(
  Predictor = colnames(coef_matrix_sparse),
  Mean = coef_mean,
```

```
  Median = coef_median,
  Lower_2.5 = coef_lower,
  Upper_97.5 = coef_upper,
  Proportion_Nonzero = proportion_nonzero
)

# Results Table

coef_ci_sparse <- coef_ci_sparse %>%
  mutate(
    Odds_Ratio = exp(Median)  # Odds Ratio
  )

coef_ci_sparse <- coef_ci_sparse %>%
  mutate(
    Rounded_Lower_2.5 = round(Lower_2.5, 4),
    Rounded_Upper_97.5 = round(Upper_97.5, 4),

    # Determine significance
    Significant = ifelse(
      Rounded_Lower_2.5 > 0 | Rounded_Upper_97.5 < 0,
      "Yes",
      "No"
    )
  ) %>%
  select(-Rounded_Lower_2.5, -Rounded_Upper_97.5)

table_data <- coef_ci_sparse %>%
  select(
    Predictor,
    Median,
    Odds_Ratio,
    Lower_2.5,
    Upper_97.5,
    Proportion_Nonzero,
    Significant
  ) %>%
  rename(
    "Estimate (Median)" = Median,
    "Odds Ratio" = Odds_Ratio,
    "Lower 2.5%" = Lower_2.5,
    "Upper 97.5%" = Upper_97.5,
    "Proportion Non-Zero" = Proportion_Nonzero
  )

table_data %>%
  kbl(row.names = F,
    booktabs = TRUE,
    longtable = TRUE,
```

```
    escape = TRUE,
    align = "c",
    digits = 3,
    caption = "Lasso Regression Coefficient Estimates"
  ) %>%
  column_spec(1, width = "2.21cm", latex_valign = "m") %>%
  column_spec(2, width = "2cm", latex_valign = "m") %>%
  column_spec(3, width = "2cm", latex_valign = "m") %>%
  column_spec(4, width = "2cm", latex_valign = "m") %>%
  column_spec(5, width = "2cm", latex_valign = "m") %>%
  column_spec(6, width = "2cm", latex_valign = "m") %>%
  column_spec(7, width = ".5cm", latex_valign = "m") %>%
  kable_styling(
    font_size = 7.6,
    latex_options = c("repeat_header", "striped"),
    full_width = FALSE,
    position = "center"
  )


# FTCD

ftcd_coefs <- coef_ci[grep("poly\\(ftcd_score, 3, raw = TRUE\\)", coef_ci$Predictor), ]


ftcd_range <- seq(min(na.omit(data$ftcd_score)),
                  max(na.omit(data$ftcd_score)), length.out = 100)

# calc predicted log odds
predicted_log_odds <- coef_ci$Median[coef_ci$Predictor == "(Intercept)"] +
                      ftcd_coefs$Median[1] * ftcd_range +
                      ftcd_coefs$Median[2] * ftcd_range^2 +
                      ftcd_coefs$Median[3] * ftcd_range^3

# convert to probabilities
predicted_probs <- 1/(1 + exp(-predicted_log_odds))

plot_data <- data.frame(
  ftcd_score = ftcd_range,
  log_odds = predicted_log_odds,
  probability = predicted_probs
)



# plot log odds
p1 <- ggplot(plot_data, aes(x = ftcd_score, y = log_odds)) +
  geom_line() +
  theme_minimal() +
```

```
  labs(x = "FTCD Score",
       y = "Predicted Log Odds",
       title = "")

# plot probabilities
p2 <- ggplot(plot_data, aes(x = ftcd_score, y = probability)) +
  geom_line() +
  theme_minimal() +
  labs(x = "FTCD Score",
       y = "Predicted Probability",
       title = "")


ftcd_plot <- ggarrange(
  p1,
  p2,
  ncol = 2,
  nrow = 1
)

print(ftcd_plot)

# Hedonsum
hedon_coefs <- coef_ci[grep("poly\\(hedonsum_y_pq1, 3, raw = TRUE\\)", coef_ci$Predictor), ]
hedon_range <- seq(min(na.omit(data$hedonsum_y_pq1)),
                   max(na.omit(data$hedonsum_y_pq1)), length.out = 100)

# calc predicted log odds
predicted_log_odds <- coef_ci$Median[coef_ci$Predictor == "(Intercept)"] +
                      hedon_coefs$Median[1] * hedon_range +
                      hedon_coefs$Median[2] * hedon_range^2 +
                      hedon_coefs$Median[3] * hedon_range^3

# convert to probabilities
predicted_probs <- 1/(1 + exp(-predicted_log_odds))

plot_data <- data.frame(
  hedonsum_score = hedon_range,
  log_odds = predicted_log_odds,
  probability = predicted_probs
)

# plot log odds
p1 <- ggplot(plot_data, aes(x = hedonsum_score, y = log_odds)) +
  geom_line() +
  theme_minimal() +
  labs(x = "Hedonsum Score",
       y = "Predicted Log Odds",
       title = "")
```

```r
# plot probabilities
p2 <- ggplot(plot_data, aes(x = hedonsum_score, y = probability)) +
  geom_line() +
  theme_minimal() +
  labs(x = "Hedonsum Score",
       y = "Predicted Probability",
       title = "")

hedon_plot <- ggarrange(
  p1,
  p2,
  ncol = 2,
  nrow = 1
)
print(hedon_plot)

# --- Generate Predictions ---

# fix factor levels in the entire dataset (ensure consistency) BUGFIX
if (length(factor_vars) > 0) {
  for (var in factor_vars) {
    validation_data[[var]] <- factor(validation_data[[var]], levels = levels_list[[var]])
  }
}

# compute ROC for valid data

# prepare valid data
model_data_valid_full <- model.matrix(full_formula, data = validation_data)
current_coef_names_valid <- colnames(model_data_valid_full) # align cols
missing_cols <- setdiff(master_coef_names, current_coef_names_valid)
if (length(missing_cols) > 0) {
  zeros_matrix <- matrix(0,
                         nrow = nrow(model_data_valid_full),
                         ncol = length(missing_cols))
  colnames(zeros_matrix) <- missing_cols
  model_data_valid_full <- cbind(model_data_valid_full, zeros_matrix)
}
extra_cols <- setdiff(current_coef_names_valid, master_coef_names)
if (length(extra_cols) > 0) {
  model_data_valid_full <-
    model_data_valid_full[, !(colnames(model_data_valid_full) %in% extra_cols)]
}

model_data_valid_full <- model_data_valid_full[, master_coef_names] # reorder
predictor_coefs <- coef_ci_sparse$Median
names(predictor_coefs) <- coef_ci_sparse$Predictor
predictor_coefs <- predictor_coefs[master_coef_names] # ensure match
```

```r
# predictions for valid data
linear_predictor <- as.vector(model_data_valid_full %*% predictor_coefs)
pred_prob <- 1 / (1 + exp(-linear_predictor))

# BUGFIX
if (length(validation_data) < length(pred_prob)) {
  validation_data <- validation_data[complete.cases(validation_data),]
}

validation_data$pred_prob <- pred_prob



# compute ROC for training data

# prepare training data
data_complete_cc <- data_complete[complete.cases(data_complete),]
model_data_train <- model.matrix(full_formula, data = data_complete_cc)
current_coef_names_train <- colnames(model_data_train)
missing_cols_train <- setdiff(master_coef_names, current_coef_names_train)
if (length(missing_cols_train) > 0) {
  zeros_matrix_train <- matrix(0, nrow = nrow(model_data_train),
                               ncol = length(missing_cols_train))
  colnames(zeros_matrix_train) <- missing_cols_train
  model_data_train <- cbind(model_data_train, zeros_matrix_train)
}
extra_cols_train <- setdiff(current_coef_names_train, master_coef_names)
if (length(extra_cols_train) > 0) {
  model_data_train <-
    model_data_train[, !(colnames(model_data_train) %in% extra_cols_train)]
}
model_data_train <- model_data_train[, master_coef_names]

# predictions for training data
linear_predictor_train <- as.vector(model_data_train %*% predictor_coefs)
pred_prob_train <- 1 / (1 + exp(-linear_predictor_train))
data_complete_cc$pred_prob <- pred_prob_train


# ROCs
roc_valid <- roc(validation_data$abst, validation_data$pred_prob)
roc_train <- roc(data_complete_cc$abst, data_complete_cc$pred_prob)

# plot ROC curves
plot(roc_train, col = "red", main = "Lasso ROC")
plot(roc_valid, add = TRUE, col = "blue")
legend("bottomright",
       legend = c(paste("Validation, AUC=", round(roc_valid$auc, 4)),
```

```
                  paste("Training, AUC=", round(roc_train$auc, 4))),
       fill = c("blue", "red"))

# calc residuals
data_complete_cc$residuals <- data_complete_cc$abst - data_complete_cc$pred_prob

# plot residuals v. fitted
ggplot(data_complete_cc, aes(x = pred_prob, y = residuals)) +
  geom_point() +
  geom_hline(yintercept = 0, color = "red", linetype = "dashed") +
  labs(x = "Predicted Probability", y = "Residuals") +
  ggtitle("Residuals vs. Predicted Probabilities")

# Histogram of Residuals
hist(data_complete_cc$residuals, main = "Histogram of Residuals")

# Q-Q Plot of Residuals
qqnorm(data_complete_cc$residuals)
qqline(data_complete_cc$residuals)
# calibration plot
num_cuts <- 100

# Training Set
calib_data_train <- data.frame(
  prob = data_complete_cc$pred_prob,
  bin = cut(data_complete_cc$pred_prob, breaks = num_cuts),
  class = as.numeric(as.character(data_complete_cc$abst))
)

calib_data_train <- calib_data_train %>%
  group_by(bin) %>%
  summarise(
    observed = sum(class) / n(),
    expected = mean(prob),
    se = sqrt(observed * (1 - observed) / n())
  )


calib_plot_train <- ggplot(calib_data_train) +
  geom_line(aes(x = expected, y = expected, color = "Ideal")) +
  geom_smooth(aes(x = expected, y = observed, color = "Loess Fit"), method = "loess") +
  labs(
    x = "Expected Proportion",
    y = "Observed Proportion",
    title = "Calibration: Training Data",
    color = "Legend"
  ) +
  theme_minimal() +
  scale_color_manual(values = c("Ideal" = "red", "Loess Fit" = "blue"))
```

```r
# Validation Set
calib_data_valid <- data.frame(
  prob = validation_data$pred_prob,
  bin = cut(validation_data$pred_prob, breaks = num_cuts),
  class = as.numeric(as.character(validation_data$abst))
)


calib_data_valid <- calib_data_valid %>%
  group_by(bin) %>%
  summarise(
    observed = sum(class) / n(),
    expected = mean(prob),
    se = sqrt(observed * (1 - observed) / n())
  )

calib_plot_valid <- ggplot(calib_data_valid) +
  geom_line(aes(x = expected, y = expected, color = "Ideal")) +
  geom_smooth(aes(x = expected, y = observed, color = "Loess Fit"), method = "loess") +
  labs(
    x = "Expected Proportion",
    y = "Observed Proportion",
    title = "Calibration: Validation Data",
    color = "Legend"
  ) +
  theme_minimal() +
  scale_color_manual(values = c("Ideal" = "red", "Loess Fit" = "blue"))

calib_plot_combined <- ggarrange(
  calib_plot_train,
  calib_plot_valid,
  ncol = 2,
  nrow = 1,
  common.legend = TRUE,
  legend = "bottom"
)

print(calib_plot_combined)

# calibration plot
num_cuts <- 100

# calibration for validation data
test_calib <- data.frame(
  prob = validation_data$pred_prob,
  bin = cut(validation_data$pred_prob, breaks = num_cuts),
  class = as.numeric(as.character(validation_data$abst))
)
```

```
test_calib <- test_calib %>%
  group_by(bin) %>%
  summarize(observed = sum(class)/n(),
            expected = sum(prob)/n(),
            se = sqrt(observed * (1-observed) / n()))

cols <- c("Ideal"="red","loess_smooth"="black","lm_smooth"="blue")
ggplot(test_calib) +
  geom_abline(aes(intercept = 0, slope = 1, color="Ideal")) +
  geom_smooth(aes(x = expected,
                  y = observed,
                  color="loess_smooth"), se=TRUE) +
  geom_smooth(aes(x = expected,
                  y = observed,
                  color="lm_smooth"), se=FALSE, method="lm") +
  scale_color_manual(values=cols) +
  labs(x = "Expected Proportion",
       y = "Observed Proportion",
       title="") +
  theme_minimal()
#GOOD; Q2; Moderator

# Def All Params
B <- 50      # num of bootstraps
m <- 5       # num of multiple imputations per bootstrap
k <- 5       # num of cv folds for cv.glmnet
poly_degree <- 3  # degree for polynomial terms

# remove id
data_complete <- data %>% select(-id)

# Flag predictor vars
# exclude 'abst' from predictors
predictor_vars <- setdiff(names(data_complete), c("abst"))

# Flag trt vars
treatment_vars <- c("BA", "Var")
baseline_vars <- setdiff(predictor_vars, treatment_vars)

# Flag numeric and int baseline vars
num_baseline_vars <- baseline_vars[sapply(data_complete[baseline_vars],
                                          function(x) is.numeric(x) & !is.integer(x))]

non_binary_factor_vars <- baseline_vars[sapply(data_complete[baseline_vars],
                                               function(x) is.integer(x))]
# Flag Binary baseline vars
binary_vars <- baseline_vars[sapply(data_complete[baseline_vars],
                                    function(x) length(unique(na.omit(x))) == 2)]
```

```r
# Flag factor vars and store levels
factor_vars <- names(data_complete)[sapply(data_complete, is.factor)]
levels_list <- lapply(data_complete[, factor_vars, drop = FALSE], levels)

# --- Build Formula ---
# Main Effects
main_effects <- paste("BA + Var +",
                      paste(c(
                        paste0("poly(", non_binary_factor_vars,
                               ", ", poly_degree, ", raw=TRUE)"),
                        num_baseline_vars,
                        binary_vars
                      ), collapse = " + "))

# Interaction Terms
interaction_terms <- paste(c(
  paste0("BA:", c(paste0("poly(", non_binary_factor_vars,
                         ", ", poly_degree, ", raw=TRUE)"), num_baseline_vars, binary_vars))
), collapse = " + ")

# Full Model Formula
full_formula <- as.formula(paste("abst ~", main_effects, "+", interaction_terms))

# fix factor levels in the entire dataset (ensure consistency) BUGFIX
if (length(factor_vars) > 0) {
  for (var in factor_vars) {
    data_complete[[var]] <- factor(data_complete[[var]], levels = levels_list[[var]])
  }
}

# Gen master model matrix
master_model_matrix <- model.matrix(full_formula, data = data_complete)
master_coef_names <- colnames(master_model_matrix)
total_coeffs <- length(master_coef_names)

# init storage for coefs
coef_matrix <- matrix(NA, nrow = B, ncol = total_coeffs)
colnames(coef_matrix) <- master_coef_names

# init object for nonzero coef indicator
nonzero_matrix <- matrix(0, nrow = B, ncol = total_coeffs)
colnames(nonzero_matrix) <- master_coef_names
# --- Bootstrap Loop ---
set.seed(1)

for (b in 1:B) {
  # Print progress
  cat("Bootstrap iteration:", b, "of", B, "\n")
```

```r
# Stratified resampling with replacement (maintain class distribution in 'abst')
bootstrap_sample <- data_complete %>%
  group_by(abst) %>%
  sample_frac(size = 1, replace = TRUE) %>%
  ungroup()

# impute data
predictor_matrix <- make.predictorMatrix(bootstrap_sample)
predictor_matrix[, "abst"] <- 0  # dont use 'abst' to predict others
predictor_matrix["abst", ] <- 0  # dont impute 'abst' (not necessary bc no missing?)
imp <- mice(bootstrap_sample, m = m, printFlag = FALSE, seed = b,
            predictorMatrix = predictor_matrix, method = 'pmm')

# Stack imputations into one dataset
stacked_data <- complete(imp, action = "long", include = FALSE)
stacked_data <- stacked_data %>% dplyr::select(-.id, -.imp) # remove '.id' / '.imp'

# fix factor levels in the entire dataset (ensure consistency) BUGFIX
if (length(factor_vars) > 0) {
  for (var in factor_vars) {
    stacked_data[[var]] <- factor(stacked_data[[var]], levels = levels_list[[var]])
  }
}

# boot model matrix and X / y
model_data_full <- model.matrix(full_formula, data = stacked_data)
x <- model_data_full[, -1] # remove intercept for glmnet
y_impute <- stacked_data$abst # response var for glmnet

# Remove predictors with zero variance BUGFIX
zero_var_cols <- apply(x, 2, function(x) var(x) == 0)
if (any(zero_var_cols)) {
  x <- x[, !zero_var_cols]
  current_coef_names <- colnames(x)
} else {
  current_coef_names <- colnames(x)}

# align columns w/ master_coef_names excluding intercept BUGFIX
master_coef_names_no_intercept <- master_coef_names[-1]  # now exclude intercept
missing_cols <- setdiff(master_coef_names_no_intercept, current_coef_names)
if (length(missing_cols) > 0) {
  zeros_matrix <- matrix(0, nrow = nrow(x), ncol = length(missing_cols))
  colnames(zeros_matrix) <- missing_cols
  x <- cbind(x, zeros_matrix)
}

# rm extra columns not in master_coef_names_no_intercept BUGFIX
extra_cols <- setdiff(current_coef_names, master_coef_names_no_intercept)
```

```r
if (length(extra_cols) > 0) {
  x <- x[, !(colnames(x) %in% extra_cols)]
}

# reorder x colums to match master_coef_names_no_intercept BUGFIX
x <- x[, master_coef_names_no_intercept]

# penalty factors
penalty_factors <- ifelse(grepl(":", master_coef_names_no_intercept), 1, 0)

# verify dims BUGFIX
if (ncol(x) != length(penalty_factors)) {
  stop("Mismatch between x columns and penalty_factors length")
}

# Run cv.glmnet
cv_lasso <- tryCatch({
  cv.glmnet(
    x = x,
    y = y_impute,
    family = "binomial",
    alpha = 1,
    penalty.factor = penalty_factors,
    standardize = TRUE,
    relax = FALSE
  )
}, error = function(e) {
  cat("Error in cv.glmnet for bootstrap", b, ":", e$message, "\n")
  return(NULL)
})

if (is.null(cv_lasso)) {
  next  # skip/move to next bootstrap iteration
}

# Fit lasso
lasso_fit <- glmnet(
  x = x,
  y = y_impute,
  family = "binomial",
  alpha = 1,
  lambda = cv_lasso$lambda.min,
  penalty.factor = penalty_factors,
  standardize = TRUE,
  relax = FALSE
)


# grab coefs including intercept
```

```
  coef_values <- as.vector(coef(lasso_fit))
  coef_names <- rownames(coef(lasso_fit))
  names(coef_values) <- coef_names

  # init object to store coefs names
  aligned_coef_values <- setNames(rep(0, length(master_coef_names)), master_coef_names)

  # match coefs to the aligned vector
  matched <- names(coef_values) %in% master_coef_names
  aligned_coef_values[names(coef_values)[matched]] <- coef_values[matched]

  # store coefs
  coef_matrix[b, ] <- aligned_coef_values

  # store nonzero indicators
  nonzero_matrix[b, ] <- ifelse(aligned_coef_values != 0, 1, 0)
}



saveRDS(coef_matrix, file = "moderators_coef.rds")
saveRDS(nonzero_matrix, file = "moderators_nonzero.rds")

coef_matrix <- readRDS(file = "moderators_coef.rds")
nonzero_matrix <- readRDS(file = "moderators_nonzero.rds")


# --- Pool coefs ---

# identify bad runs
bad_runs <- apply(coef_matrix, 1, function(x) all(is.na(x)) | all(x == 0)) &
            apply(nonzero_matrix, 1, function(x) all(is.na(x)) | all(x == 0))

# rm bad runs from objects
coef_matrix <- coef_matrix[!bad_runs, ]
nonzero_matrix <- nonzero_matrix[!bad_runs, ]

# make coef data frame
coef_mean <- colMeans(coef_matrix, na.rm = TRUE)
coef_median <- apply(coef_matrix, 2, median, na.rm = TRUE)
coef_lower <- apply(coef_matrix, 2, quantile, probs = 0.025, na.rm = TRUE)
coef_upper <- apply(coef_matrix, 2, quantile, probs = 0.975, na.rm = TRUE)
proportion_nonzero <- colSums(nonzero_matrix, na.rm = TRUE) / sum(nonzero_matrix[,1])

coef_ci <- data.frame(
  Predictor = colnames(coef_matrix),
  Mean = coef_mean,
  Median = coef_median,
  Lower_2.5 = coef_lower,
```

```
  Upper_97.5 = coef_upper,
  Proportion_Nonzero = proportion_nonzero
)




# plot the CIs
ggplot(coef_ci, aes(x = Predictor, y = Median)) +
  geom_point() +
  geom_errorbar(aes(ymin = Lower_2.5, ymax = Upper_97.5), width = 0.2) +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(
    title = "Bootstrap Confidence Intervals for Lasso Coefficients",
    y = "Coefficient Estimate",
    x = "Predictors"
  )

# plot nonzero prop
ggplot(coef_ci, aes(x = Predictor, y = Proportion_Nonzero)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  theme_bw() +
  labs(
    title = "Proportion of Bootstrap Samples with Non-Zero Coefficients",
    y = "Proportion",
    x = "Predictors"
  )




# --- Enforce Sparsity ---

# first criteria
threshold <- .90 # make this dynamic later
predictors_to_keep <- names(proportion_nonzero)[proportion_nonzero >= threshold]
coef_matrix_sparse <- coef_matrix

# columns (predictors) to zero out
predictors_to_zero <- setdiff(colnames(coef_matrix_sparse), predictors_to_keep)

# second criteria
zero_ci_predictors <- coef_ci$Predictor[
  round(coef_ci$Median, 4) == 0 &
  round(coef_ci$Lower_2.5, 4) == 0 &
  round(coef_ci$Upper_97.5, 4) == 0]
```

```r
predictors_to_zero <- union(predictors_to_zero, zero_ci_predictors)

# zero out the coefs
if(length(predictors_to_zero) > 0){
  coef_matrix_sparse[, predictors_to_zero] <- 0
}

coef_mean <- colMeans(coef_matrix_sparse, na.rm = TRUE)
coef_median <- apply(coef_matrix_sparse, 2, median, na.rm = TRUE)
coef_lower <- apply(coef_matrix_sparse, 2, quantile, probs = 0.025, na.rm = TRUE)
coef_upper <- apply(coef_matrix_sparse, 2, quantile, probs = 0.975, na.rm = TRUE)
proportion_nonzero <- colSums(nonzero_matrix, na.rm = TRUE) / sum(nonzero_matrix[,1])

coef_ci_sparse <- data.frame(
  Predictor = colnames(coef_matrix_sparse),
  Mean = coef_mean,
  Median = coef_median,
  Lower_2.5 = coef_lower,
  Upper_97.5 = coef_upper,
  Proportion_Nonzero = proportion_nonzero
)

# --- Generate Predictions ---

# fix factor levels in the entire dataset (ensure consistency) BUGFIX
if (length(factor_vars) > 0) {
  for (var in factor_vars) {
    validation_data[[var]] <- factor(validation_data[[var]], levels = levels_list[[var]])
  }
}

# compute ROC for valid data

# prepare valid data
model_data_valid_full <- model.matrix(full_formula, data = validation_data)
current_coef_names_valid <- colnames(model_data_valid_full) # align cols
missing_cols <- setdiff(master_coef_names, current_coef_names_valid)
if (length(missing_cols) > 0) {
  zeros_matrix <- matrix(0,
                         nrow = nrow(model_data_valid_full),
                         ncol = length(missing_cols))
  colnames(zeros_matrix) <- missing_cols
  model_data_valid_full <- cbind(model_data_valid_full, zeros_matrix)
}
extra_cols <- setdiff(current_coef_names_valid, master_coef_names)
if (length(extra_cols) > 0) {
  model_data_valid_full <-
    model_data_valid_full[, !(colnames(model_data_valid_full) %in% extra_cols)]
}
```

```r
model_data_valid_full <- model_data_valid_full[, master_coef_names] # reorder
predictor_coefs <- coef_ci_sparse$Median
names(predictor_coefs) <- coef_ci_sparse$Predictor
predictor_coefs <- predictor_coefs[master_coef_names] # ensure match

# predictions for valid data
linear_predictor <- as.vector(model_data_valid_full %*% predictor_coefs)
pred_prob <- 1 / (1 + exp(-linear_predictor))

# BUGFIX
if (length(validation_data) < length(pred_prob)) {
  validation_data <- validation_data[complete.cases(validation_data),]
}

validation_data$pred_prob <- pred_prob



# compute ROC for training data

# prepare training data
data_complete_cc <- data_complete[complete.cases(data_complete),]
model_data_train <- model.matrix(full_formula, data = data_complete_cc)
current_coef_names_train <- colnames(model_data_train)
missing_cols_train <- setdiff(master_coef_names, current_coef_names_train)
if (length(missing_cols_train) > 0) {
  zeros_matrix_train <- matrix(0, nrow = nrow(model_data_train),
                               ncol = length(missing_cols_train))
  colnames(zeros_matrix_train) <- missing_cols_train
  model_data_train <- cbind(model_data_train, zeros_matrix_train)
}
extra_cols_train <- setdiff(current_coef_names_train, master_coef_names)
if (length(extra_cols_train) > 0) {
  model_data_train <-
    model_data_train[, !(colnames(model_data_train) %in% extra_cols_train)]
}
model_data_train <- model_data_train[, master_coef_names]

# predictions for training data
linear_predictor_train <- as.vector(model_data_train %*% predictor_coefs)
pred_prob_train <- 1 / (1 + exp(-linear_predictor_train))
data_complete_cc$pred_prob <- pred_prob_train


# ROCs
roc_valid <- roc(validation_data$abst, validation_data$pred_prob)
roc_train <- roc(data_complete_cc$abst, data_complete_cc$pred_prob)
```

```r
# plot ROC curves
plot(roc_train, col = "red", main = "Lasso ROC")
plot(roc_valid, add = TRUE, col = "blue")
legend("bottomright",
       legend = c(paste("Validation, AUC=", round(roc_valid$auc, 4)),
                  paste("Training, AUC=", round(roc_train$auc, 4))),
       fill = c("blue", "red"))

# calc residuals
data_complete_cc$residuals <- data_complete_cc$abst - data_complete_cc$pred_prob

# plot residuals v. fitted
ggplot(data_complete_cc, aes(x = pred_prob, y = residuals)) +
  geom_point() +
  geom_hline(yintercept = 0, color = "red", linetype = "dashed") +
  labs(x = "Predicted Probability", y = "Residuals") +
  ggtitle("Residuals vs. Predicted Probabilities")

# Histogram of Residuals
hist(data_complete_cc$residuals, main = "Histogram of Residuals")

# Q-Q Plot of Residuals
qqnorm(data_complete_cc$residuals)
qqline(data_complete_cc$residuals)
# calibration plot
num_cuts <- 100

# Training Set
calib_data_train <- data.frame(
  prob = data_complete_cc$pred_prob,
  bin = cut(data_complete_cc$pred_prob, breaks = num_cuts),
  class = as.numeric(as.character(data_complete_cc$abst))
)

calib_data_train <- calib_data_train %>%
  group_by(bin) %>%
  summarise(
    observed = sum(class) / n(),
    expected = mean(prob),
    se = sqrt(observed * (1 - observed) / n())
  )


calib_plot_train <- ggplot(calib_data_train) +
  geom_line(aes(x = expected, y = expected, color = "Ideal")) +
  geom_smooth(aes(x = expected, y = observed, color = "Loess Fit"), method = "loess") +
  labs(
    x = "Expected Proportion",
    y = "Observed Proportion",
```

```
    title = "Calibration: Training Data",
    color = "Legend"
  ) +
  theme_minimal() +
  scale_color_manual(values = c("Ideal" = "red", "Loess Fit" = "blue"))

# Validation Set
calib_data_valid <- data.frame(
  prob = validation_data$pred_prob,
  bin = cut(validation_data$pred_prob, breaks = num_cuts),
  class = as.numeric(as.character(validation_data$abst))
)


calib_data_valid <- calib_data_valid %>%
  group_by(bin) %>%
  summarise(
    observed = sum(class) / n(),
    expected = mean(prob),
    se = sqrt(observed * (1 - observed) / n())
  )

calib_plot_valid <- ggplot(calib_data_valid) +
  geom_line(aes(x = expected, y = expected, color = "Ideal")) +
  geom_smooth(aes(x = expected, y = observed, color = "Loess Fit"), method = "loess") +
  labs(
    x = "Expected Proportion",
    y = "Observed Proportion",
    title = "Calibration: Validation Data",
    color = "Legend"
  ) +
  theme_minimal() +
  scale_color_manual(values = c("Ideal" = "red", "Loess Fit" = "blue"))

calib_plot_combined <- ggarrange(
  calib_plot_train, calib_plot_valid,
  ncol = 2, nrow = 1,
  common.legend = TRUE, legend = "bottom"
)

print(calib_plot_combined)

# calibration plot
num_cuts <- 100

# calibration for validation data
test_calib <- data.frame(
  prob = validation_data$pred_prob,
  bin = cut(validation_data$pred_prob, breaks = num_cuts),
```

```
  class = as.numeric(as.character(validation_data$abst))
)

test_calib <- test_calib %>%
  group_by(bin) %>%
  summarize(observed = sum(class)/n(),
            expected = sum(prob)/n(),
            se = sqrt(observed * (1-observed) / n()))

cols <- c("Ideal"="red","loess_smooth"="black","lm_smooth"="blue")
ggplot(test_calib) +
  geom_abline(aes(intercept = 0, slope = 1, color="Ideal")) +
  geom_smooth(aes(x = expected,
                  y = observed,
                  color="loess_smooth"), se=TRUE) +
  geom_smooth(aes(x = expected,
                  y = observed,
                  color="lm_smooth"), se=FALSE, method="lm") +
  scale_color_manual(values=cols) +
  labs(x = "Expected Proportion",
       y = "Observed Proportion",
       title="") +
  theme_minimal()
```

Appendix II: Functions Code

```
# Helper Functions for Smoking Cessation Project

# --- Preamble ---
# Date of last update: Oct. 17, 2024
# R Version: 4.3.1
# Package Versions:
#   outliers: 0.15
#   tidyverse: 2.0.0
#   reshape2: 1.4.4
#   moments: 0.14.1
#   vcd: 1.4-12


# Libraries for functions
suppressMessages(library(tidyverse))
library(outliers) # for grubbs.test()
library(reshape2) # for melt()
library(moments) # for skewness() and kurtosis()
library(vcd) # for assocstats()









# -- BEGIN Psuedo-correlation Matrix Section --

# Helper function to calculate Eta^2
eta_squared <- function(aov_model) {
  #' This function calculates the Eta^2 stat from an anova model. Eta^2 is a measure
  #' of effect size. Eta^2 describes the proportion of the total variance attributable
  #' to a given factor
  #'
  #' @param aov_model An aov object
  #' @return A numeric value of Eta^2 (proportion of the total variance explained)
  #'
  sum_of_squares_model <- summary(aov_model)[[1]]$"Sum Sq"[1]
  sum_of_squares_total <- sum(summary(aov_model)[[1]]$"Sum Sq")
  eta_sq <- sum_of_squares_model / sum_of_squares_total
  return(eta_sq)
}


# Main function to compute correlations or psuedo-correlations
# NOTE: Below still does not work with logical variables, but now fixed for NAs
```

```r
psuedo_cor_mat <- function(data) {
  #' Builds a pseudo correlation matrix for a given dataset, flexible for numerical
  #' and categorical variables. Uses Pearson correlation for numerical-numerical pairs,
  #' Cramer's V for categorical-categorical pairs, point biserial correlation for
  #' numerical-binary categorical pairs, and the square root of Eta^2 for
  #' numerical-multi-category pairs.
  #'
  #' @param data A df with any mix of numerical and categorical variables.
  #' @return A symmetric matrix with correlation coefficients
  #' (or equivalent measures) for all variable pairs.
  #'
  variables <- names(data)
  n <- length(variables)
  cor_matrix <- matrix(NA, n, n, dimnames = list(variables, variables))

  for (i in 1:n) {
    for (j in i:n) {
      if (i == j) {
        cor_matrix[i, j] <- 1
      } else {
        var_i <- data[[i]]
        var_j <- data[[j]]

        if (is.numeric(var_i) && is.numeric(var_j)) {
          # Pearson correlation for continuous-continuous
          cor_matrix[i, j] <- cor_matrix[j, i] <- cor(var_i, var_j,
                                                      use = "pairwise.complete.obs")

        } else if (is.factor(var_i) && is.factor(var_j)) {
          # Remove NAs for both variables
          valid_idx <- !is.na(var_i) & !is.na(var_j)
          var_i_clean <- var_i[valid_idx]
          var_j_clean <- var_j[valid_idx]

          # Drop unused levels
          var_i_clean <- droplevels(var_i_clean)
          var_j_clean <- droplevels(var_j_clean)

          # Cramer's V for categorical-categorical
          if (length(var_i_clean) > 0 && length(var_j_clean) > 0) {
            cramers_v <- sqrt(assocstats(table(var_i_clean, var_j_clean))$cramer)
            cor_matrix[i, j] <- cor_matrix[j, i] <- cramers_v
          } else {
            cor_matrix[i, j] <- cor_matrix[j, i] <- NA
          }

        } else {
          # Continuous-categorical pairs
          if (is.numeric(var_i) && (is.factor(var_j) || is.character(var_j))) {
```

```
            continuous <- var_i
            factor_var <- as.factor(var_j)
          } else if ((is.factor(var_i) || is.character(var_i)) && is.numeric(var_j)) {
            continuous <- var_j
            factor_var <- as.factor(var_i)
          } else {
            next  # Skip if variables are not appropriate types
          }

          # Remove NAs in both variables
          valid_idx <- !is.na(continuous) & !is.na(factor_var)
          continuous <- continuous[valid_idx]
          factor_var <- factor_var[valid_idx]

          # Drop unused levels
          factor_var <- droplevels(factor_var)

          if (length(unique(factor_var)) == 2) {
            # Point biserial correlation for binary factors
            factor_numeric <- as.numeric(factor_var) - 1
            cor_matrix[i, j] <- cor_matrix[j, i] <- cor(continuous, factor_numeric)
          } else if (length(unique(factor_var)) > 2) {
            # Eta-squared for multi-category factors
            if (length(factor_var) > 0) {
              aov_result <- aov(continuous ~ factor_var)
              eta_sq <- eta_squared(aov_result)
              cor_matrix[i, j] <- cor_matrix[j, i] <- sqrt(eta_sq)
            } else {
              cor_matrix[i, j] <- cor_matrix[j, i] <- NA
            }
          } else {
            # Cannot compute correlation if factor_var has insufficient levels
            cor_matrix[i, j] <- cor_matrix[j, i] <- NA
          }
        }
      }
    }
  }
  return(cor_matrix)
}

# -- END Psuedo-correlation Matrix Section --
```

```r
# Function to generate summary table aka "Table 1"
# NOTE: Does not like the following variable types: Date, `hms` num / difftime
# NOTE: Will likely need custom subroutines to handle dates / hms / difftime
summary_table <- function(df, stratify_var = NULL) {
  #' Create summary table for a data frame with option for stratification. It
  #' summarizes numeric variables by mean, standard deviation, and IQR.
  #' Categorical variables by count and percentage. Optionally stratifies by a
  #' specified variable.
  #'
  #' @param df A data frame
  #' @param stratify_var (Optional) A variable on which to stratify the summaries.
  #'
  #' @return A data frame (or tibble) containing the summarized statistics.

  # Internal function to handle the actual summarization
  summarize_internal <- function(df, stratify_var = NULL, is_recursive_call = FALSE) {

    # --- SUMMARIZER SECTION ---
    # Function to summarize continuous variable
    summarize_continuous <- function(data, var) {
      var_data <- data[[var]]
      mean_sd <- paste0(round(mean(var_data, na.rm = TRUE), 2),
                        " (",
                        round(sd(var_data, na.rm = TRUE), 2),
                        ")")
      quantiles <- quantile(var_data, probs = c(0.25, 0.75), na.rm = TRUE)
      quantile_range <- paste0("(",
                               round(quantiles[1], 2),
                               " - ",
                               round(quantiles[2], 2),
                               ")")
      paste(mean_sd, quantile_range)
    }

    # Func to summarize categorical variable
    summarize_categorical <- function(data, var) {
      var_data <- table(data[[var]])
      percentages <- prop.table(var_data) * 100
```

```
  paste(names(var_data), ": ",
        var_data, " (",
        round(percentages, 2),
        "%)",
        sep = "",
        collapse = ", ")
}

# Handler for strata
variable_names <- names(df)
if (!is.null(stratify_var) && stratify_var %in% variable_names) {
  variable_names <- variable_names[variable_names != stratify_var]
}

# Summarize all into a list for table
summary_list <- map(variable_names, ~ {
  var_type <- ifelse(is.numeric(df[[.x]]) | is.integer(df[[.x]]),
                     "Numeric", "Categorical")
  summarizer <- ifelse(var_type == "Numeric",
                       summarize_continuous,
                       summarize_categorical)
  summary <- summarizer(df, .x)
  variable_name <- ifelse(var_type == "Numeric",
                          paste(.x, "[Mean (SD) (Quantile)]"),
                          paste(.x, "[n (%)]"))
  tibble(Variable = variable_name, Type = var_type, Summary = summary)
})

# Bind to table
summary_table <- bind_rows(summary_list)

# --- NORMALITY AND OUTLIERS SECTION ---
# Perform Shapiro-Wilk and Grubbs tests only if it's not a recursive call
# and no stratify_var is given
if (!is_recursive_call && is.null(stratify_var)) {
  # Get numeric variables
  numeric_vars <- variable_names[sapply(df[variable_names],
                                        function(x) is.numeric(x) | is.integer(x))]

  # Init object for S-W / Grubbs results
  test_results <- map(numeric_vars, ~ {
    x <- df[[.x]]
    x <- na.omit(x)
    n <- length(x)
    normality <- NA
    outlier <- NA
    skewness_label <- NA
    kurtosis_label <- NA
```

```r
# Shapiro-Wilk test (Note: now accommodates n > 5000)
# NOTE: Consider in future -- KS test, AD test, Jarque-Bera test
if (n >= 3) {
  if (n > 5000) {
    shapiro_test <- shapiro.test(sample(x, 4999))
  } else {
    shapiro_test <- shapiro.test(x)
  }
  normality <- ifelse(shapiro_test$p.value < 0.05, "No", "Yes")
} else {
  normality <- "Insufficient data"
}

# Grubbs test
if (n >= 3) {
  grubbs_test <- grubbs.test(x)
  outlier <- ifelse(grubbs_test$p.value < 0.05, "Yes", "No")
} else {
  outlier <- "Insufficient data"
}

# Skewness and Kurtosis
if (n >= 2) {
  skewness <- skewness(x)
  kurtosis <- kurtosis(x)

  # Skewness label
  if (abs(skewness) < 0.5) {
    skewness_label <- "Centered"
  } else if (skewness > 0.5) {
    skewness_label <- "Right-skewed"
  } else {
    skewness_label <- "Left-skewed"
  }

  # Kurtosis label
  if (kurtosis < 2.5) {
    kurtosis_label <- "Platykurtic"
  } else if (kurtosis > 3.5) {
    kurtosis_label <- "Leptokurtic"
  } else {
    kurtosis_label <- "Mesokurtic"
  }
} else {
  skewness_label <- "Insufficient data"
  kurtosis_label <- "Insufficient data"
}

tibble(Variable = .x,
```

```
            `Normal Distribution` = normality,
            `Outlier(s) Present` = outlier,
            `Skewness` = skewness_label,
            `Kurtosis` = kurtosis_label)
  })

  # Bind test results into df
  test_results_df <- bind_rows(test_results)

  # Get actual variable names from summary_table for merge
  # Note: this could break in some datasets if the variable names include "["
  # therefore a more general solution later is preferred
  summary_table$ActualVariable <- gsub(" \\[.*\\]", "", summary_table$Variable)

  # Merge summary_table with test_results_df
  summary_table <- left_join(summary_table,
                             test_results_df,
                             by = c("ActualVariable" = "Variable"))

  # Remove temp matching ActualVariable column
  summary_table <- summary_table %>% dplyr::select(-ActualVariable)
} else if (!is.null(stratify_var)) {
  # --- STRATA SECTION ---

  # Stratification handling
  stratified_summaries <- df %>%
    group_by(!!sym(stratify_var)) %>%
    do(summarize_internal(., stratify_var = NULL, is_recursive_call = TRUE))

  summary_table <- stratified_summaries %>%
    ungroup() %>%
    dplyr::select(-group_cols()) %>% # This doesn't seem necessary, why'd I do it?
    pivot_wider(names_from = !!sym(stratify_var), values_from = Summary)

  # Function for significance testing (numeric variables)
  test_continuous <- function(data, var, group_var) {
    formula <- as.formula(paste0("`", var, "` ~ `", group_var, "`")) #BUGFIX: spaces
    num_groups <- length(unique(data[[group_var]]))
    if (num_groups == 2) {
      test_result <- t.test(formula, data = data)
      p_value <- test_result$p.value
    } else {
      test_result <- kruskal.test(formula, data = data)
      p_value <- test_result$p.value
    }
    p_value
  }

  # Function for significance testing (categorical variables)
```

```r
test_categorical <- function(data, var, group_var) {
  table_data <- table(data[[var]], data[[group_var]])
  table_data <- table_data[rowSums(table_data) > 0, # This prevents NaN error
                           colSums(table_data) > 0, # in ChiSq test.
                           drop = FALSE]
  test_result <- chisq.test(table_data)
  p_value <- test_result$p.value
  p_value
}

# Perform significance testing (if stratify_var is specified)
significance_tests <- map(variable_names, ~ {
  var_type <- ifelse(is.numeric(df[[.x]]) | is.integer(df[[.x]]),
                     "Numeric", "Categorical")
  tester <- ifelse(var_type == "Numeric",
                   test_continuous,
                   test_categorical)
  p_value <- tester(df, .x, stratify_var)
  tibble(Variable = .x, `P-value` = p_value)
})

significance_table <- bind_rows(significance_tests)

# Adjust p-values using Bonferroni correction
significance_table$`Adjusted P-value` <- p.adjust(significance_table$`P-value`,
                                                  method = "bonferroni")

# Get actual variable names from summary_table for merge
# Note: this could break in some datasets if the variable names include "["
# therefore a more general solution later is preferred
summary_table$ActualVariable <- gsub(" \\[.*\\]", "", summary_table$Variable)

# Merge summary_table with significance_table on actual variable names
summary_table <- left_join(summary_table,
                           significance_table,
                           by = c("ActualVariable" = "Variable"))

# Remove rows corresponding to stratify_var
summary_table <- summary_table %>% filter(ActualVariable != stratify_var)

# Create 'Sig.' column based on Adjusted P-value
summary_table$`Sig.` <- case_when(
  summary_table$`Adjusted P-value` <= 0.0001 ~ "****",
  summary_table$`Adjusted P-value` <= 0.001 ~ "***",
  summary_table$`Adjusted P-value` <= 0.01  ~ "**",
  summary_table$`Adjusted P-value` <= 0.05  ~ "*",
  TRUE                                      ~ "ns"
)
```

```
      # Remove temp matching ActualVariable, adjusted/unadjusted P-Values
      summary_table <- dplyr::select(summary_table,
                                     c(-ActualVariable,
                                       -`P-value`,
                                       -`Adjusted P-value`))
    }

    # Remove "Type" column for stratified tables
    if (!is.null(stratify_var) && !is_recursive_call) {
      summary_table <- summary_table %>% dplyr::select(-Type)
    }

    return(summary_table)
  }

  # Call internal function to initiate procedure
  summarize_internal(df, stratify_var)
}
```

```
# Function to count number of variables where missing data occours
count_missing_vars <- function(df) {
  #' Calc the number of variables in a data frame that have at least one missing
  #' value.
  #'
  #' @param df A data frame
  #'
  #' @return A data frame with a single row and column indicating the count
  #' of variables with any missing values.
  #'
  missing_count <- df %>%
    summarise_all(~ sum(is.na(.))) %>%
    pivot_longer(everything()) %>%
    filter(value > 0) %>%
    nrow()
```

```r
  return(data.frame(Num_Missing_Vars = missing_count))
}
```

```r
# Count by variable type
summarize_variables_types <- function(data) {
  #' Summarizes the types of variables in a dataset by type. Possible types:
  #' Numeric, or Categorical (factors or logical).
  #'
  #' @param data A data frame
  #' @return A table with the counts of variables categorized by type
  #'
  var_types <- sapply(data, function(x) {
    if(is.numeric(x) | is.integer(x)) {
      return("Numeric")
    }
    else {
      return("Categorical")
    }
  })

  var_counts <- table(var_types)
  return(var_counts)
}
```

```r
# Function to find groups of correlated variables
findCorrelatedGroups <- function(corr_matrix, threshold) {
  #' Find Groups of Correlated Variables
  #'
  #' This function identifies groups of variables in a correlation matrix that
  #' exceed a specified correlation threshold.
  #'
  #' @param corr_matrix A square symmetric matrix representing variable correlations.
  #' @param threshold A numeric threshold for defining significant correlation.
  #' @return A list of numeric vectors, each containing indices of a group of
  #' correlated variables.
  correlation <- abs(corr_matrix)
  diag(correlation) <- 0 # blank out diagonal to ignore self-correlation
  groups <- list()
  visited <- rep(FALSE, ncol(correlation))

  for (i in 1:ncol(correlation)) {
    if (!visited[i]) {
      # Find index of variables correlated w/ the current var
      high_cor_vars <- which(correlation[, i] > threshold)
      if (length(high_cor_vars) > 0) {
        group <- unique(c(i, high_cor_vars)) # include current var in the group
        groups[[length(groups) + 1]] <- group
        visited[group] <- TRUE
      }
      else {
        visited[i] <- TRUE
      }
    }
  }
  return(groups)
}




# Function to drop select highly correlated vars
reduceDataset <- function(df, corr_matrix, threshold) {
  #' Reduce Dataset by Dropping Highly Correlated Variables
  #'
  #' This function reduces a dataset by identifying and dropping highly correlated
  #' variables based on the provided correlation matrix and threshold.
  #'
  #' @param df A dataframe containing the dataset to be reduced.
  #' @param corr_matrix A square symmetric matrix of correlations between
  #' variables in `df`.
  #' @param threshold A numeric threshold to identify high correlations.
  #' @return A dataframe with reduced variables.
```

```r
  groups <- findCorrelatedGroups(corr_matrix, threshold)
  to_drop <- numeric()

  for (group in groups) {
    # Subroutine to select which to drop (inverse of variance explained)
    variances <- sapply(group, function(index) {
      # Below is to fix bug with fread/data.table (depending on how data is read in)
      if (is.data.table(df)) {
        var(as.vector(df[[index]]))
      } else {
        var(df[, index, drop = FALSE])
      }
    })

    # Drop variable with minimum variance
    min_variance_index <- which.min(variances)

    to_drop_var <- group[min_variance_index]

    to_drop <- c(to_drop, to_drop_var)
  }

  to_drop <- unique(to_drop)

  # only drop columns that exist in the dataset (bug fix)
  to_drop <- to_drop[to_drop <= ncol(df)]

  if (length(to_drop) > 0) {
    reduced_df <- dplyr::select(df, -to_drop)
  }
  else {
    reduced_df <- df # If nothing to drop, return the original dataset (bug fix)
  }

  return(reduced_df)
}
```

```r
check_mar <- function(data) {
  #' Check Missing At Random (MAR) in Data
  #'
```

```r
#' This function tests each variable in the dataset for missingness being at
#' random, conditional on all other variables, using logistic regression models.
#'
#' @param data A dataframe with variables to test for MAR.
#' @return A list where each entry corresponds to a variable in `data` and
#' contains results of MAR tests.


all_vars <- names(data)

# Init object for results
results <- list()

for (var in all_vars) {
  # Generate missing indicator
  missing_var_name <- paste0("missing_", make.names(var))
  data[[missing_var_name]] <- as.integer(is.na(data[[var]]))

  # formula for GLM
  formula_vars <- sapply(all_vars[all_vars != var], function(v) paste0("`", v, "`"))
  formula_str <- paste0(missing_var_name, " ~ ", paste(formula_vars, collapse = " + "))
  formula <- as.formula(formula_str)

  model <- tryCatch(
    {
      glm(formula, data = data, family = binomial())
    },
    error = function(e) {
      return(NULL)
    }
  )

  if (!is.null(model)) {
    # check for sig. variables
    summary_model <- summary(model)
    pvalues <- summary_model$coefficients[, "Pr(>|z|)"]
    significant_vars <- names(pvalues[pvalues < 0.05 & !is.na(pvalues)])

    # removve intercept from significant vars (if present)
    significant_vars <- significant_vars[significant_vars != "(Intercept)"]

    # rm backticks from significant vars for cleaner output
    significant_vars <- gsub("`", "", significant_vars)

    results[[var]] <- list(
      is_mar = length(significant_vars) == 0,
      significant_vars = significant_vars
    )
  } else {
```

```
      results[[var]] <- list(
        is_mar = NA,
        significant_vars = NA,
        error = "Error in fitting GLM"
      )
    }

    # remove temporary missing indicator col
    data[[missing_var_name]] <- NULL
  }

  return(results)
}




calculate_mse <- function(actual, predicted) {
  #' Calculate Mean Squared Error
  #'
  #' This function computes the mean squared error between actual and predicted
  #' numerical values.
  #'
  #' @param actual A numeric vector of actual values.
  #' @param predicted A numeric vector of predicted values.
  #' @return Numeric value representing the mean squared error.

  # inputs must be numeric
  actual <- as.numeric(actual)
  predicted <- as.numeric(predicted)

  # calc squared differences
  squared_diff <- (actual - predicted)^2

  # Remove NA / infinite values
  valid_diff <- squared_diff[is.finite(squared_diff) & !is.na(squared_diff)]

  # calc MSE
  mse <- sum(valid_diff) / length(valid_diff)

  return(mse)
}



variable_importance_reg <- function(model, data, target_column, n_iterations = 20) {
  #' Calculate Variable Importance for Linear Regression Models
  #'
  #' This function estimates the importance of each predictor variable in a
```

```r
  #' regression model by measuring the increase in prediction error after
  #' permuting each predictor variable.
  #'
  #' @param model A regression model object (e.g., lm, lmer).
  #' @param data A dataframe containing the data used in the model.
  #' @param target_column The name of the target (dependent) variable in `data`.
  #' @param n_iterations The number of iterations to perform for estimating
  #' importance (default is 20).
  #' @return A dataframe with variables and their relative importance scores.

  predictor_vars <- all.vars(formula(model))[-1]
  original_preds <- as.numeric(predict(model, data))
  original_mse <- calculate_mse(data[[target_column]], original_preds)

  importance_scores <- matrix(0, nrow = n_iterations, ncol = length(predictor_vars))
  colnames(importance_scores) <- predictor_vars

  for (i in 1:n_iterations) {
    for (var in predictor_vars) {
      data_shuffled <- data
      data_shuffled[[var]] <- sample(data_shuffled[[var]])

      shuffled_preds <- as.numeric(predict(model, data_shuffled))
      shuffled_mse <- calculate_mse(data_shuffled[[target_column]], shuffled_preds)

      importance_scores[i, var] <- shuffled_mse - original_mse
    }
  }

  avg_importance <- abs(colMeans(importance_scores))
  se_importance <- apply(importance_scores, 2, sd) / sqrt(n_iterations)

  result <- data.frame(
    Variable = predictor_vars,
    Importance = avg_importance,
    SE = se_importance
  )

  result <- result[order(-result$Importance), ]
  result$`Relative Importance` <- result$Importance / max(result$Importance) * 100

  return(result)
}
```

```r
select_best_model <- function(model_backward, model_forward) {
  #TODO: INSERT ROXYGEN
  # Compare the formulas of the two models
  if ((formula(model_backward) == formula(model_forward))) {
    # Formulas are the same, models are the same
    return(model_backward)
  } else {

    # --- Formulas are different ---

    # Extract terms from models
    terms_backward <- attr(terms(model_backward), "term.labels")
    terms_forward <- attr(terms(model_forward), "term.labels")

    # Check if models are nested -- BUG FIX 10/17/24
    if (all(terms_backward %in% terms_forward)) {
      # Backward model is nested within forward model
      smaller_model <- model_backward
      larger_model <- model_forward
    } else if (all(terms_forward %in% terms_backward)) {
      # Forward model is nested within backward model
      smaller_model <- model_forward
      larger_model <- model_backward
    } else {
      # Models are not nested; cannot perform LRT therefore we compare BIC to
      # select the best model
      aic_backward <- BIC(model_backward)
      aic_forward <- AIC(model_forward)

      if (aic_backward < aic_forward) {
        return(model_backward)
      } else {
        return(model_forward)
      }
    }

    # Perform LRT
    lrt_result <- tryCatch({
```

```
      anova(smaller_model, larger_model, test = "LRT")
    }, error = function(e) {
      stop("Error in performing LRT: ", e$message)
    })

    p_value <- lrt_result$`Pr(>Chi)`[2]

    if (is.na(p_value)) {
      stop("LRT failed: p-value is NA")
    }

    # Decide which model to choose based on LRT
    if (p_value < 0.05) {
      return(larger_model) # Larger model is significantly better
    } else {
      return(smaller_model) # Smaller model is sufficient

    }
  }}
```

```
create_cross_table <- function(data) {
  vars <- names(data)
  n_vars <- length(vars)

  results <- list()

  for (i in 1:(n_vars-1)) {
    for (j in (i+1):n_vars) {
      var1 <- vars[i]
      var2 <- vars[j]

      # variable types
      type1 <- class(data[[var1]])[1]
      type2 <- class(data[[var2]])[1]

      # result based on variable types
```

```
      if (type1 %in% c("factor", "character") && type2 %in% c("factor", "character")) {
        result <- table(data[[var1]], data[[var2]])

      } else if (type1 %in% c("numeric", "integer") && type2 %in% c("numeric", "integer")) {
        result <- cor(data[[var1]], data[[var2]], use = "complete.obs")

      } else {
        if (type1 %in% c("factor", "character")) {
          result <- tapply(data[[var2]], data[[var1]], mean, na.rm = TRUE)
        } else {
          result <- tapply(data[[var1]], data[[var2]], mean, na.rm = TRUE)
        }
      }

      #  result
      results[[paste(var1, "x", var2)]] <- result
    }
  }

  return(results)
}
```