M158 Project Part 4

Riley Elliott and Anna Choi

2022-05-10

Sparse and Smooth Linear Models

Introduction

We scraped the data used in the following analysis from baseballsavant.com, the online depot for all publicly available advanced metrics on Major League Baseball players. Our sample is 131 of the 132 MLB hitters who "qualified" for end-of-year awards during the 2021 season by recording at least 502 plate appearances. One player was removed because preliminary data analysis revealed that he was an extreme outlier in average launch angle (LA), which feeds into some of our variables of interest. The population to which we infer results to, therefore, is all MLB players with positive LA who collect at least 502 plate appearances in any season.

Our response variable of interest is **Isolated Power (ISO)**, which attempts to quantify how much power a hitter demonstrates during games. It calculates the rate at which they hit for "extra" total bases. Where a single is 1 total base, a double is 2, a triple is 3, and a home run is 4, ISO is calculated by $ISO = \frac{\text{(Total Bases) - (Singles)}}{\text{(At Bats)}}$.

Our chief predictor variables of interest express some combination of how hard a batter hits the ball, the vertical angle at which they hit it, the horizontal direction in which they hit it, and the quality of their swing decisions. The four main predictors of interest are:

Barrel percentage (BRL%). BRL% combines information about the launch angles and exit velocities of a hitter's batted balls. If a batter hits the ball with a 98 MPH exit velocity, it must be hit between 26 and 30 vertical degrees to be defined as a barrel. If they hit it at 99 MPH, it must be hit between 25 and 31°. This pattern continues–every 1 MPH increase in exit velocity loosens the launch angle requirement by 2 total degrees (one in each direction).

Pull percentage (PULL%), the percentage of a hitter's batted balls that are hit to the same third of the field as the side of home plate from which they bat (e.g. right vs. left).

Walk rate (BB%), the percentage of plate appearances in which the batter draws a walk (by not swinging at 4 out-of-zone pitches).

Zone swing percentage (Z-Swing%), the percentage of pitches in the strike zone at which the batter swings.

Running Ridge Regression and LASSO

First, let's find the lambda value that best minimizes MSE and maximizes R^2 for Ridge Regression. We do this using cross validation.

Both shrinkage methods, RR and Lasso, start with all predictor variables in the model (though Lasso eliminates some of the less consequential ones during the process by making their coefficients equal to 0). Some feature engineering was required to make this possible. We turned the predictors FIRST_NAME and LAST_NAME into a single ID variable and turned all categorical variables except MODE_AMATEUR_ACQUISITION into dummy variables. MODE_AMATEUR_ACQUISITION was removed from the data set because one observation represented an entire category, which caused the number of variables in the training and test data sets to differ. We were not particularly interested in MODE_AMATEUR_ACQUISITION and its relationship to ISO, so we removed it to avoid the issue entirely. These decisions are discussed further during part 2 of this project (below).

The sample sizes of the training and test data sets were set at 2/3 and 1/3 of the total data set, respectively. As always, we will build each model using the training set and fit it to the test set to get an idea of how well it fits the population.

We constructed 50 ridge regression models using our CV training data. Below, we have plotted the $RMSE$ and R^2 values, respectively, against the penalties.

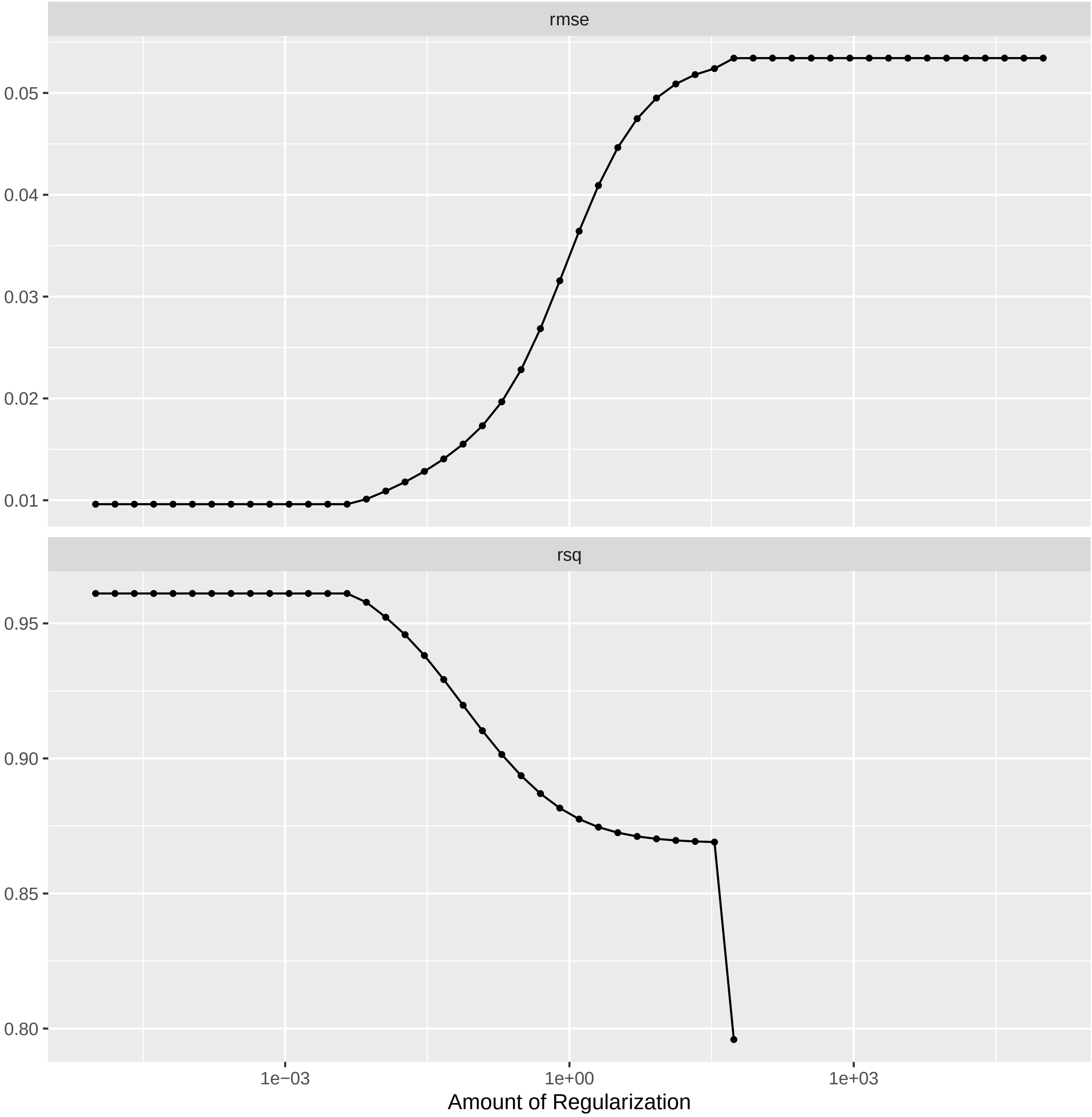


Figure 1: RMSE and R2 of RR models against lambda values

We can see that, according to both $RMSE$ and R^2, the superior models are those with lower penalties (those closest to OLS). The first 14 models return identical $RMSE$ and R^2 values, so the penalties associated with any of those models would be a good option. We chose model 1, which has lambda = 1.0e-05. A ridge regression model with this lambda value creates a model with the following coefficients for each of our predictor variables:

```
## # A tibble: 53 x 3
##    term          estimate penalty
##    <chr>            <dbl>   <dbl>
##  1 (Intercept)  0.189     0.00001
##  2 PLAYER_AGE  -0.000317  0.00001
##  3 PA           0.000398  0.00001
##  4 HR           0.00267   0.00001
##  5 HR_PERCENT   0.00270   0.00001
##  6 K_PERCENT    0.00101   0.00001
##  7 BB_PERCENT   0.000652  0.00001
##  8 BA          -0.0000652 0.00001
##  9 SLG          0.00253   0.00001
## 10 OBP          0.000435  0.00001
## # ... with 43 more rows
```

Now, we turn to LASSO.

We constructed 50 Lasso models using our CV training data. Below is a graph that plots the $RMSE$ and R^2 values of those models on the data against the lambda values.

Again, according to both $RMSE$ and R^2, the superior models are those with lower penalties (those closest to OLS). Any of the first 9 penalty values will do. We will choose the first model again. It also has lambda = 1.0e-5. A LASSO model with this lambda value creates a model with the following coefficients for each of our predictor variables:

```
## # A tibble: 53 x 3
##    term         estimate penalty
##    <chr>           <dbl>   <dbl>
##  1 (Intercept)  0.189    0.00001
##  2 PLAYER_AGE   0        0.00001
##  3 PA           0        0.00001
##  4 HR           0        0.00001
##  5 HR_PERCENT   0.0185   0.00001
##  6 K_PERCENT    0.000206 0.00001
##  7 BB_PERCENT   0.00109  0.00001
##  8 BA          -0.0145   0.00001
##  9 SLG          0.0413   0.00001
## 10 OBP          0        0.00001
## # ... with 43 more rows
```

We are aiming to compare RR, LASSO, and OLS models. We have what we need to build the first two. The OLS model was constructed previously in project 3.

Model Comparison: MLR, RR, and LASSO

The model with the .pred vs. ISO line that most closely resembles y=x (the black line in the plot above) will be most desirable as the predictions will most closely resemble the actual values. It is apparent that the LASSO-generated model, lasso_preds, is most desirable. It most closely resembles y=x in terms of both position and linearity.

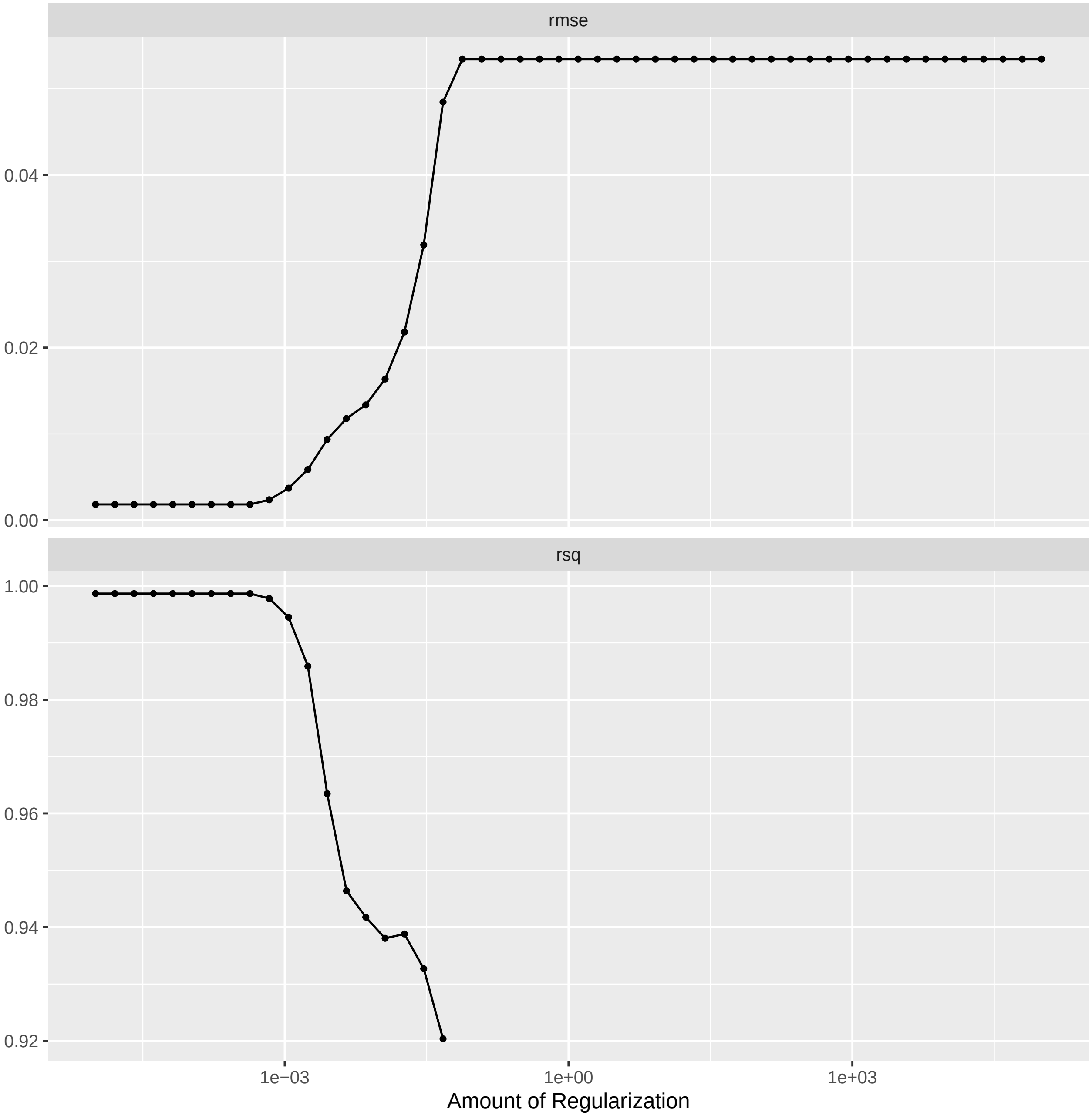


Figure 2: RMSE and R2 of LASSO models against lambda values

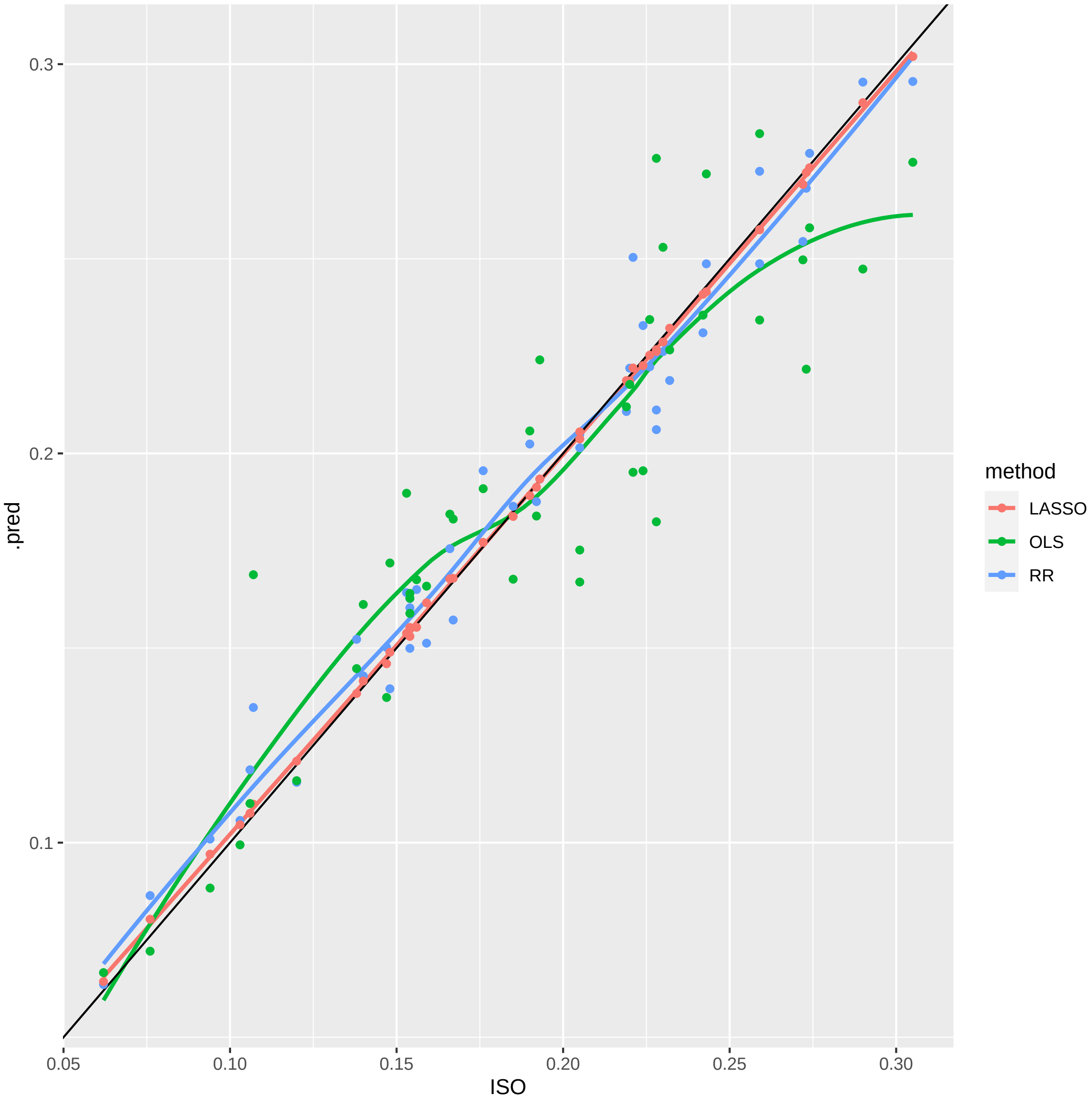


Figure 3: Predicted vs. Observed ISO Values for 3 Models

Regression Spline and Loess Smoother Methods

Smoothing methods all function on models with only one predictor variable. In the following analysis, we will use BRL% as our one predictor. Our previous analyses revealed that it had the highest bivariate correlation with our response (ISO). We therefore have more interest in using it as a single predictor of ISO. First, we will build a number of such models using regression splines. Depending on the degree of the polynomial used to model each region and the total number of regions (and therefore, degrees of freedom), these models will differ in regards to the bias-variance tradeoff.

This plot allows us to see how we can optimize $RMSE$ and R^2 by adjusting the aforementioned parameters. Generally, we can see that modeling the relationship with lower-degree polynomials typically yields the best fit. This makes sense, as previous examinations of our data revealed that the relationship between ISO and BRL_PERCENT is linear.

Using this plot, we selected 4 models that represented a good mix of the parameters but also did a good job of optimizing $RMSE$ and R^2. These are the models with df=10 and degree=1, df=15 and degree=2, df=5 and degree=3, and df=10 and degree=3. The tidy widgets and scatter plots for the 4 models are shown below.

```
## # A tibble: 11 x 5
##    term                estimate std.error statistic  p.value
##    <chr>                  <dbl>     <dbl>     <dbl>    <dbl>
##  1 (Intercept)           0.0640    0.0152      4.23 4.66e- 5
##  2 BRL_PERCENT_knot11    0.0631    0.0208      3.03 2.96e- 3
##  3 BRL_PERCENT_knot12    0.0880    0.0171      5.16 9.85e- 7
##  4 BRL_PERCENT_knot13    0.112     0.0184      6.06 1.62e- 8
##  5 BRL_PERCENT_knot14    0.115     0.0178      6.43 2.72e- 9
##  6 BRL_PERCENT_knot15    0.130     0.0180      7.26 4.10e-11
##  7 BRL_PERCENT_knot16    0.145     0.0176      8.25 2.33e-13
##  8 BRL_PERCENT_knot17    0.168     0.0174      9.66 1.13e-16
##  9 BRL_PERCENT_knot18    0.162     0.0176      9.25 1.05e-15
## 10 BRL_PERCENT_knot19    0.192     0.0177     10.8  1.98e-19
## 11 BRL_PERCENT_knot110   0.260     0.0239     10.9  1.30e-19
```

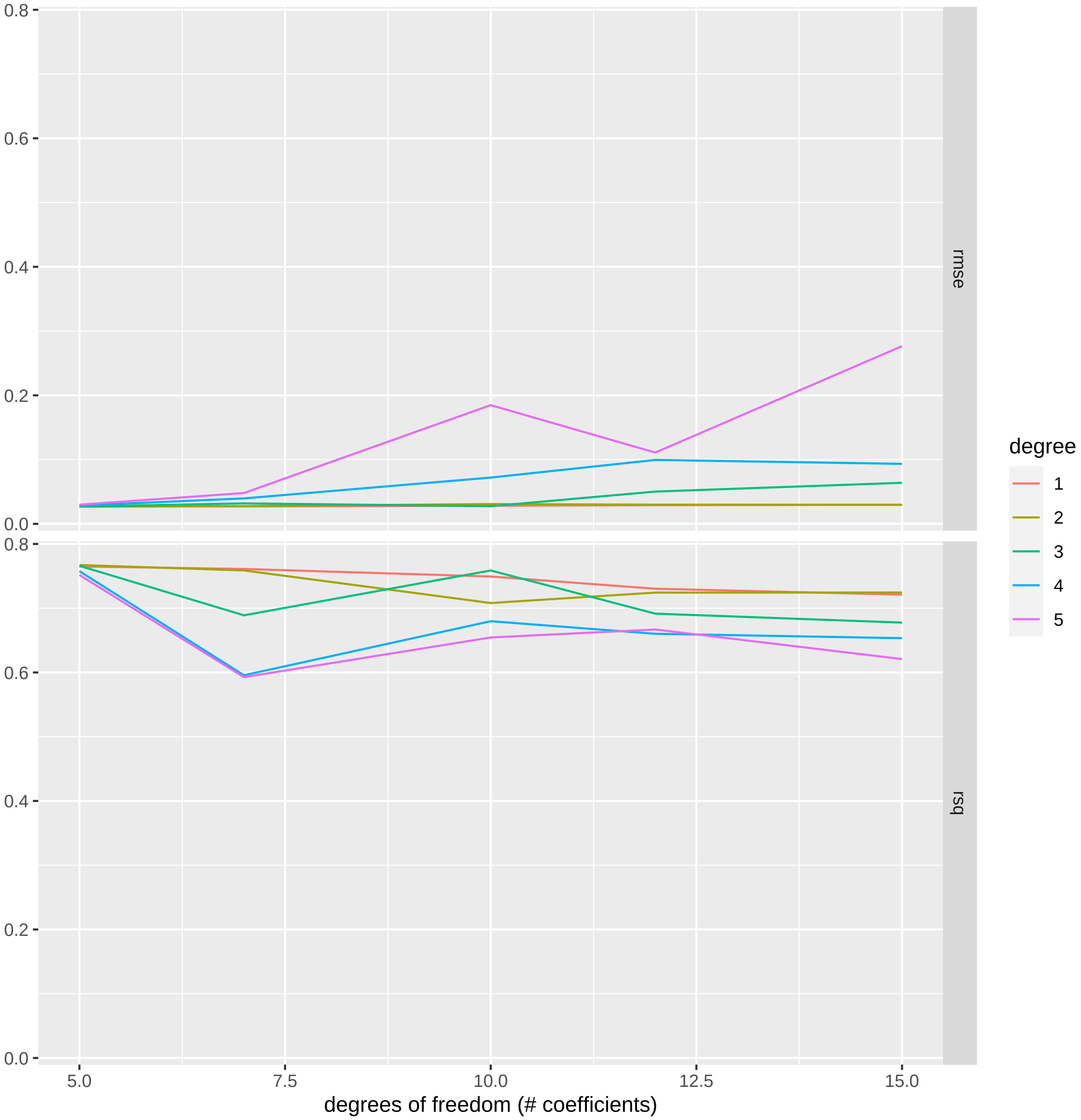
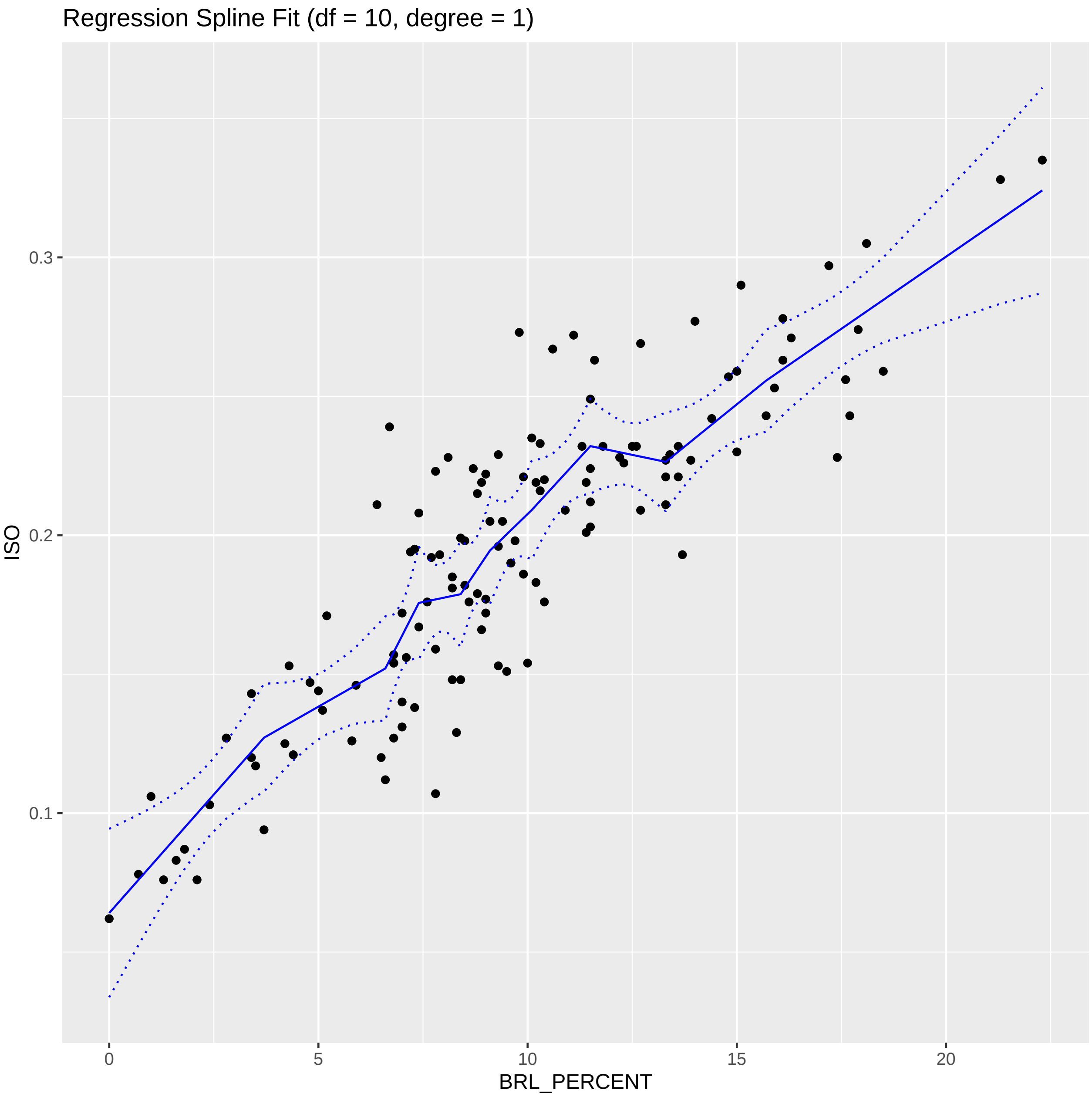
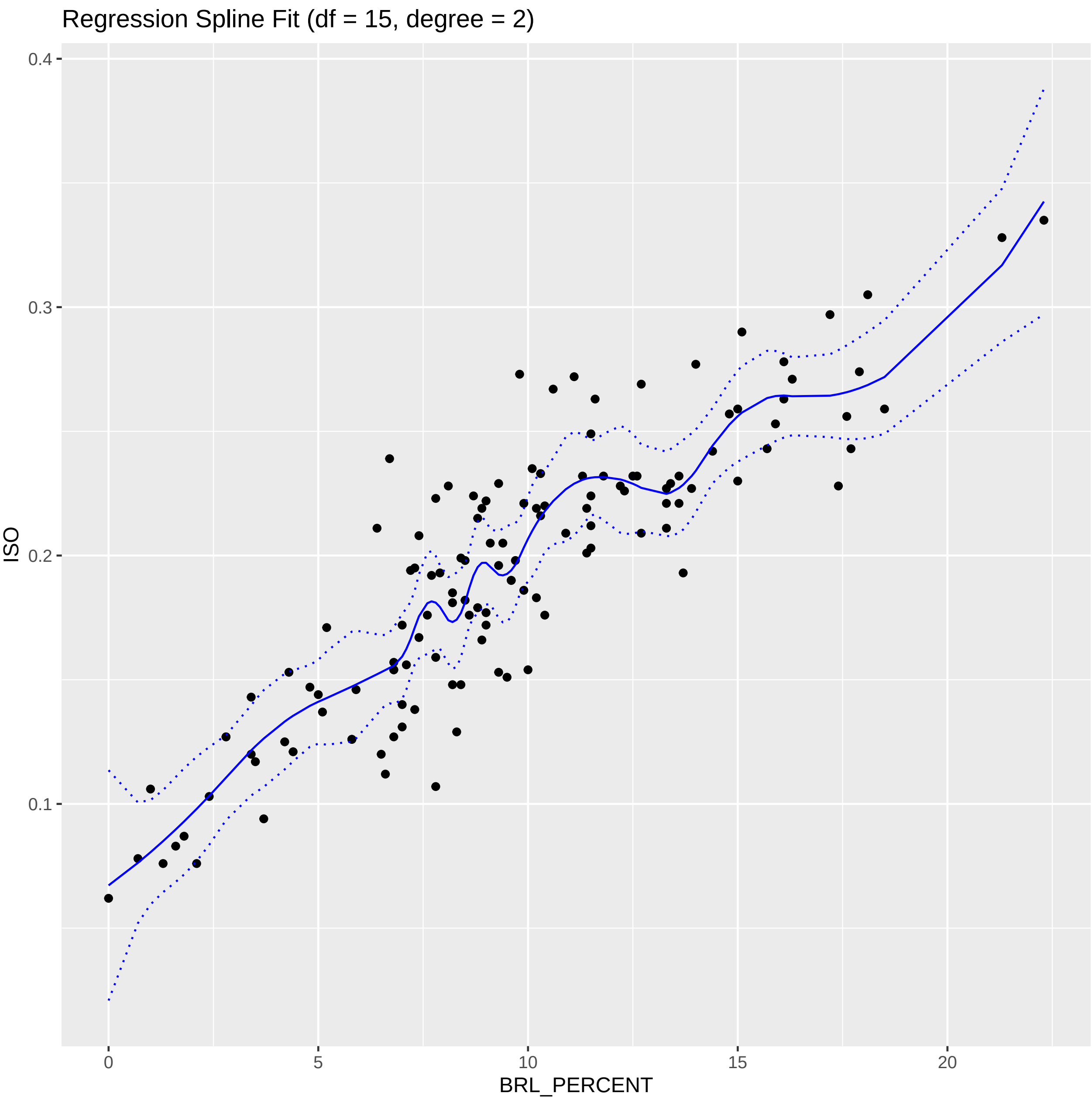


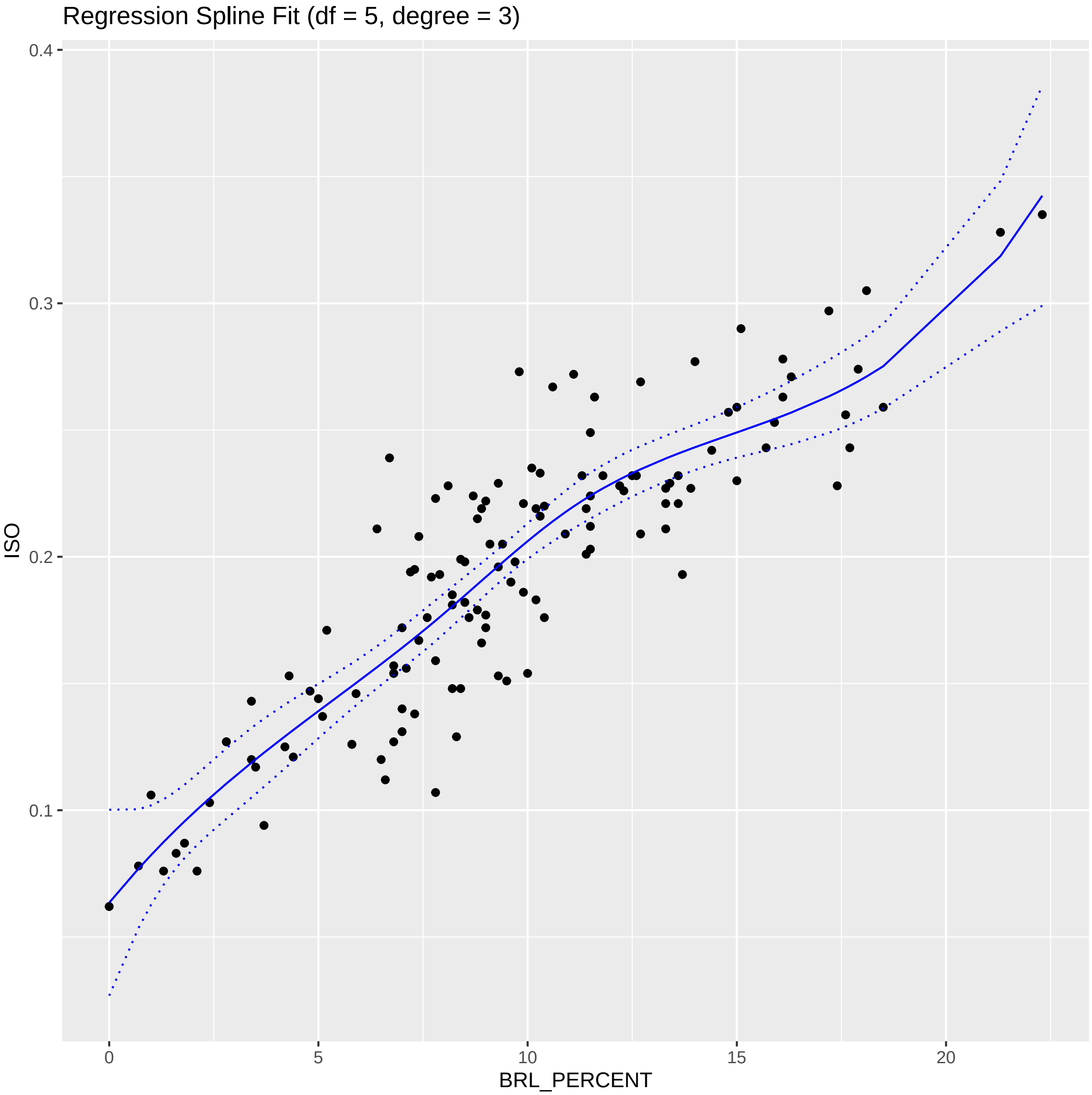
Figure 4: RMSE and R2 of models of varying degree, degrees of freedom



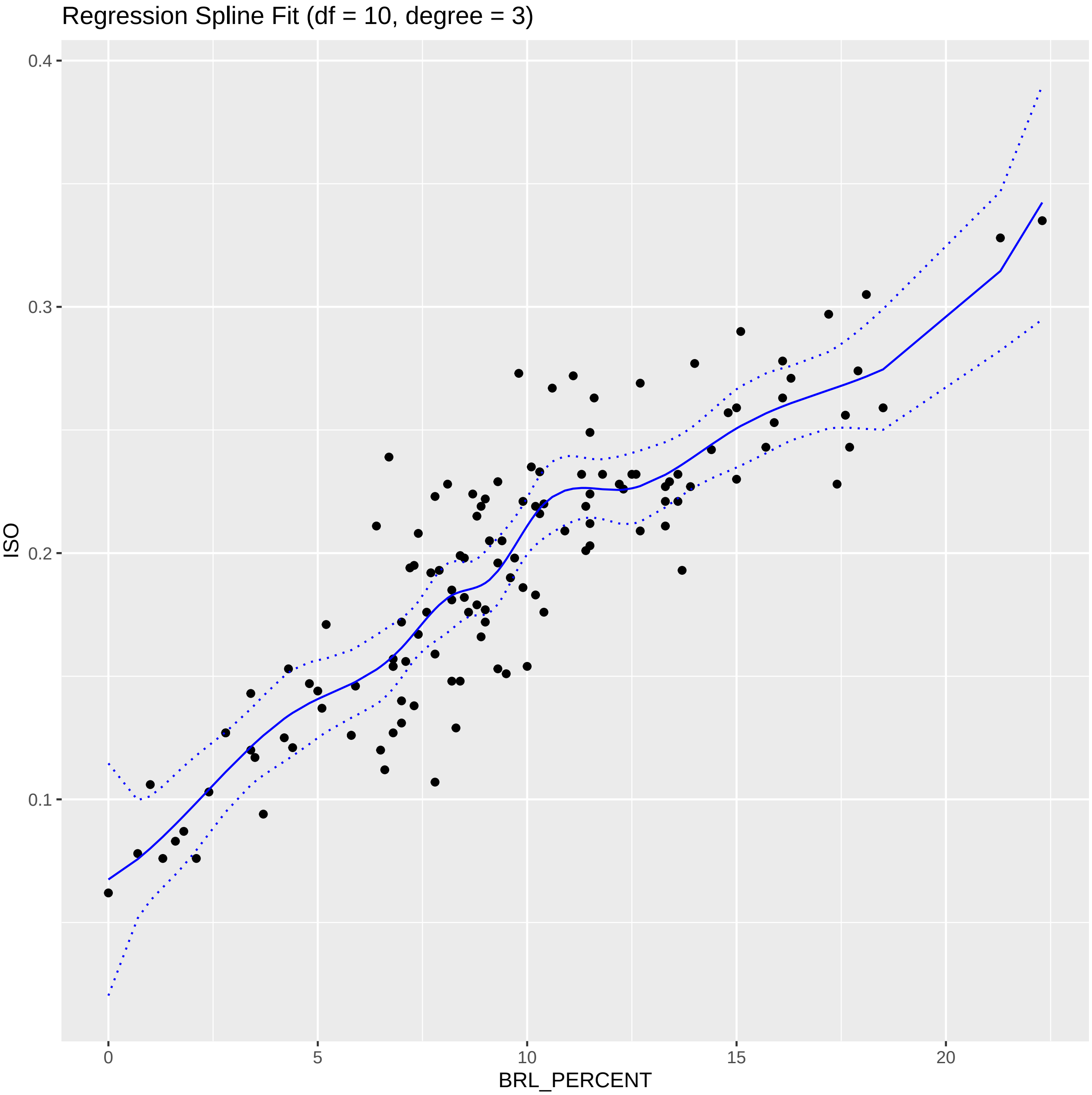
```
## # A tibble: 16 x 5
##    term                 estimate std.error statistic  p.value
##    <chr>                   <dbl>     <dbl>     <dbl>    <dbl>
##  1 (Intercept)            0.0672    0.0232     2.90  4.48e- 3
##  2 BRL_PERCENT_knot21     0.0181    0.0381     0.474 6.37e- 1
##  3 BRL_PERCENT_knot22     0.0667    0.0251     2.66  8.96e- 3
##  4 BRL_PERCENT_knot23     0.0807    0.0307     2.63  9.76e- 3
##  5 BRL_PERCENT_knot24     0.0906    0.0253     3.58  4.99e- 4
##  6 BRL_PERCENT_knot25     0.121     0.0282     4.30  3.60e- 5
##  7 BRL_PERCENT_knot26     0.0995    0.0272     3.66  3.82e- 4
##  8 BRL_PERCENT_knot27     0.135     0.0269     5.01  1.98e- 6
##  9 BRL_PERCENT_knot28     0.119     0.0278     4.29  3.74e- 5
## 10 BRL_PERCENT_knot29     0.144     0.0265     5.45  2.88e- 7
## 11 BRL_PERCENT_knot210    0.163     0.0288     5.65  1.20e- 7
## 12 BRL_PERCENT_knot211   0.166     0.0286     5.79  6.28e- 8
## 13 BRL_PERCENT_knot212   0.152     0.0267     5.71  8.95e- 8
## 14 BRL_PERCENT_knot213   0.199     0.0272     7.34  3.23e-11
## 15 BRL_PERCENT_knot214   0.190     0.0341     5.57  1.67e- 7
## 16 BRL_PERCENT_knot215   0.275     0.0324     8.49  8.37e-14
```



```
## # A tibble: 6 x 5
##   term                 estimate std.error statistic  p.value
##   <chr>                   <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)            0.0635    0.0183      3.46 7.31e- 4
## 2 BRL_PERCENT_knot31     0.0524    0.0338      1.55 1.24e- 1
## 3 BRL_PERCENT_knot32     0.0847    0.0182      4.65 8.25e- 6
## 4 BRL_PERCENT_knot33     0.203     0.0260      7.81 1.96e-12
## 5 BRL_PERCENT_knot34     0.179     0.0268      6.68 6.91e-10
## 6 BRL_PERCENT_knot35    0.279     0.0290      9.63 9.32e-17
```

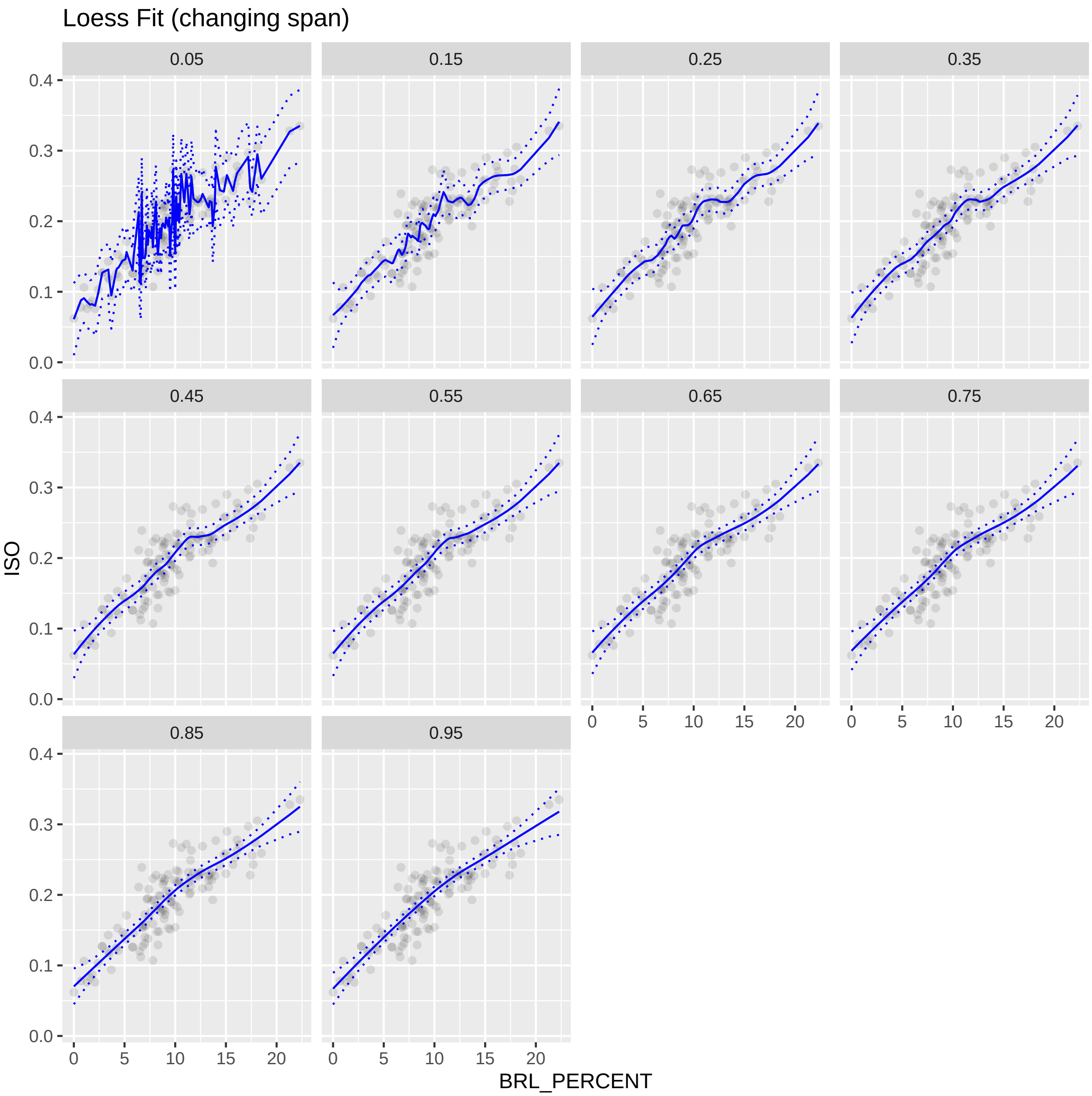


```
## # A tibble: 11 x 5
##    term                 estimate std.error statistic  p.value
##    <chr>                   <dbl>     <dbl>     <dbl>    <dbl>
##  1 (Intercept)            0.0675    0.0236     2.86  4.97e- 3
##  2 BRL_PERCENT_knot41     0.0142    0.0458     0.311 7.56e- 1
##  3 BRL_PERCENT_knot42     0.0747    0.0295     2.53  1.27e- 2
##  4 BRL_PERCENT_knot43     0.0797    0.0296     2.69  8.19e- 3
##  5 BRL_PERCENT_knot44     0.119     0.0257     4.65  8.72e- 6
##  6 BRL_PERCENT_knot45     0.116     0.0270     4.31  3.41e- 5
##  7 BRL_PERCENT_knot46     0.167     0.0272     6.14  1.09e- 8
##  8 BRL_PERCENT_knot47     0.149     0.0282     5.29  5.56e- 7
##  9 BRL_PERCENT_knot48     0.212     0.0379     5.60  1.41e- 7
## 10 BRL_PERCENT_knot49    0.195     0.0478     4.07  8.35e- 5
## 11 BRL_PERCENT_knot410   0.275     0.0334     8.22  2.74e-13
```

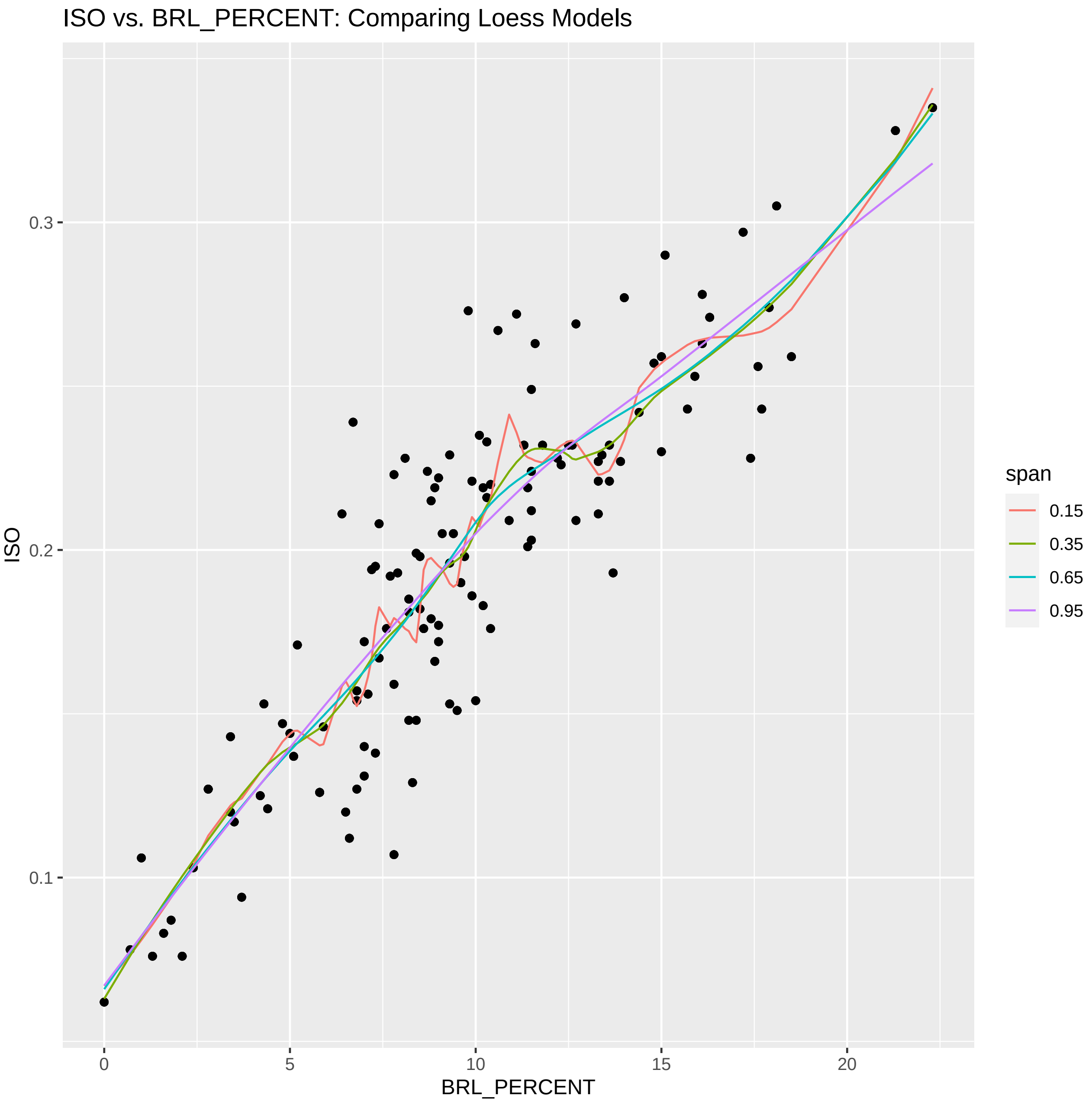


Among these models, we most prefer the one with df = 5 and degree = 3. It is the smoothest (indicating that it is the least overfit). It is also the most linear. We have a preference for more linear models here. We will explain this when we add the Loess models into the mix.

Now, we will create 4 more smoothing models using Loess. The models are differentiated by changing the span (the proportion of total observations with non-zero weights that is used to approximate an accordingly-sized section of the data). Below are 10 ISO vs. BRL_PERCENT plots created using every span size increment of 0.1 from .05 to .95.



We can see that the smallest span model is drastically overfit to the data, especially considering the probable linearity of the relationship between predictor and response. Below, we plot 4 of these models together to better compare them. The 4 models are those with span=0.15, 0.35, 0.65, 0.95. They were chosen because they are visually differentiable from one another when overlaid and represent the range of models above.



The lower span models are not very smooth, indicating overfit. When one considers the interpretation of the coefficients, there seems to be little reason for extreme local variations off of a y=-mx+b model. Why would an increase in BRL_PERCENT from one level differ from an identical increase from some other level in terms of how much it changes ISO? The higher-span models, which pay less attention to what appears to be random variance, predict a slight tapering off in slope as BRL_PERCENT increases. This seems far more viable. Even the .95 span model has significant curvature when compared to a line. We feel that this model sufficiently avoids becoming biased and limits excessive bias well. For the same reasons, we prefer this model to any of the regression spline models above.

Something New

Pre-processing

All save one variable are used, with FIRST_NAME and LAST_NAME merged to form the identity variable NAME. The categorical variable MODE_AMATEUR_ACQUISITION was removed since only one observation was labelled one of the categories, hence that observation was either in the training or testing data, causing the number of variables in the two datasets to differ by one and cause fitting issues. To convert categorical variables into numerical ones, through one hot encoding, such variables were transformed into vectors with binary values, the number of vectors being the number of categories. Moreover, in accordance with part 3 of this project, an outlier has been removed to leave 131 total observations.

To ensure our models are as representative of the population as possible according to the metric R^2, we split our dataset into training and testing sets by a 2:1 ratio. Within the training dataset, we used cross validation to find the optimal tuning parameter corresponding to the model in question: lambda for Lasso Regression, and k for k-nearest neighbors.

kNN Regression

Although we have produced rather high-performing models, the previous methods are all parametric models that have technical conditions assuming, for example, normally distributed errors, or that our response is related to the explanatory variables in the first place. However, there is no way to truly know if these assumptions were valid and representative of our population. Hence, here we use the nonparametric model of k-Nearest Neighbor regression which makes no such assumptions and is thus more robust. Now we can consider all explanatory variables in the dataset and no longer need to worry about multicollinearity, since its nonparametric nature means the algorithm considers all features altogether, instead of separately which would lead to disregard of correlation between those explanatory variables.

Instead of fitting a regression line through data points, for every response variable value, kNN makes a prediction by selecting the k closest points, measured here by Euclidean distance, and taking their mean. In the model below, each of the k points are weighted equally since the model was already performing excellently, instead of weighting closer points more than farther points.

Since our dataset is rather small, with the training set containing 87 observations, we used 4 folds to cross validate for k, with a grid search of all possible integers k between 1 and 60, the number of analysis observations in the fold. By R^2, the larger k, the better the model. However according to RMSE, a minimum is reached at k=18, with $k \in [11, 23]$ being the smallest values, where RMSE differed less than 0.001.

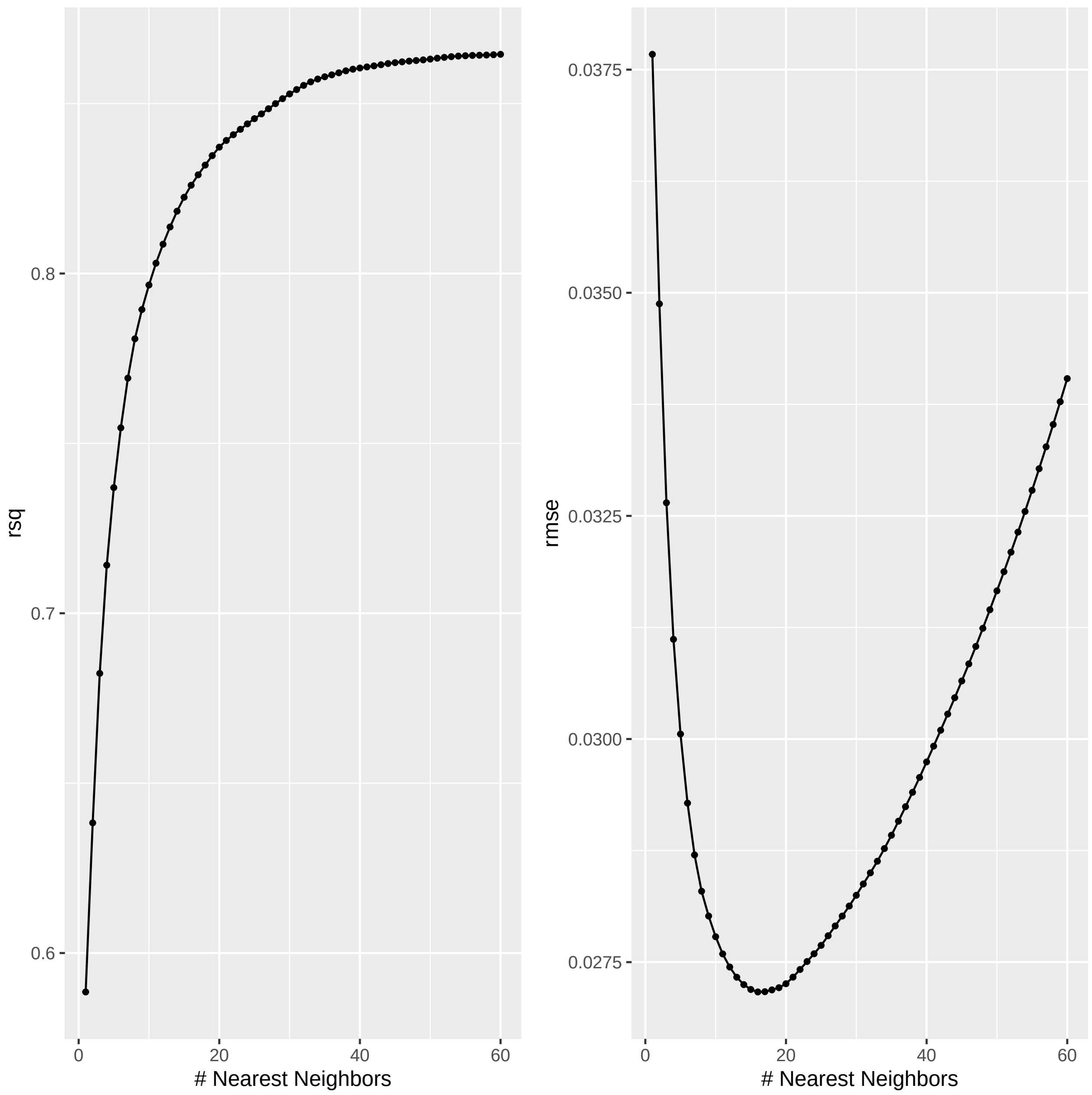
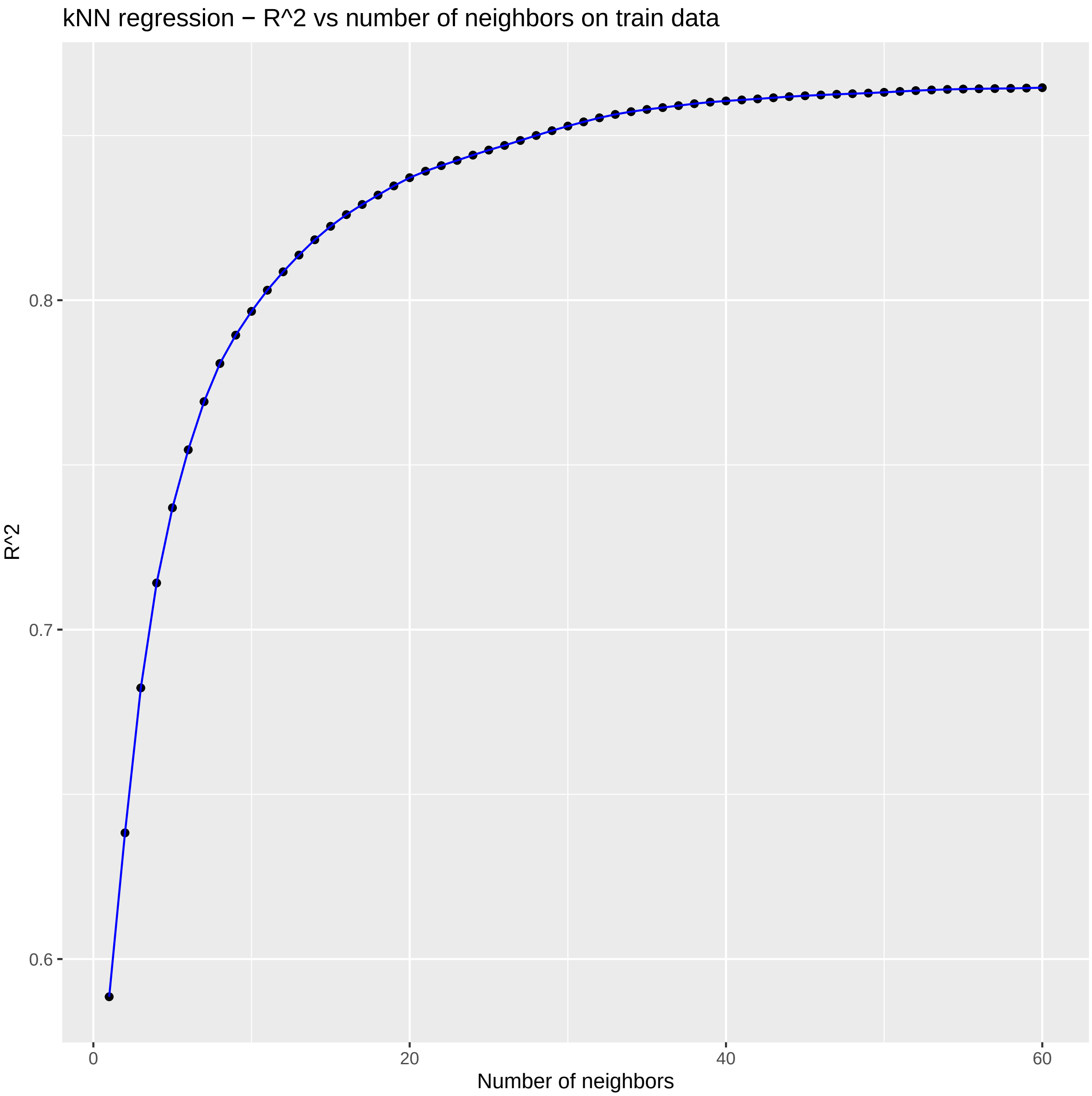


Figure 5: RMSE and R2 of kNN models fitted and predicted on training set



Since the test dataset has 44 observations, we also made predictions on the testing dataset based on each possible kNN model, $k \in [1, 44]$, fitted on the training dataset. In the figure, the dotted blue line represents kNN models predicted against training data, and the red line represents kNN models predicted against testing data. We can see R^2 only increases as more neighbors are included for each point's prediction, and surprisingly, each model reliably performed better when predicted against testing data rather than the training data the models were fit with. The best model according to both testing and training data is the one that includes all observations (the the results for $k \in [1, 44]$), but the literature states that a k around the square root of the number of observations (in our case $\sqrt{87} \approx 9$) yields a good model, although our dataset is on the smaller side. Thus, conclude that a kNN model is not necessary. However, this analysis proved that the relationship between our response variable ISO and other explanatory variables exists and is quite linear, thus for the sake of simplicity, we will not focus on a kNN model.

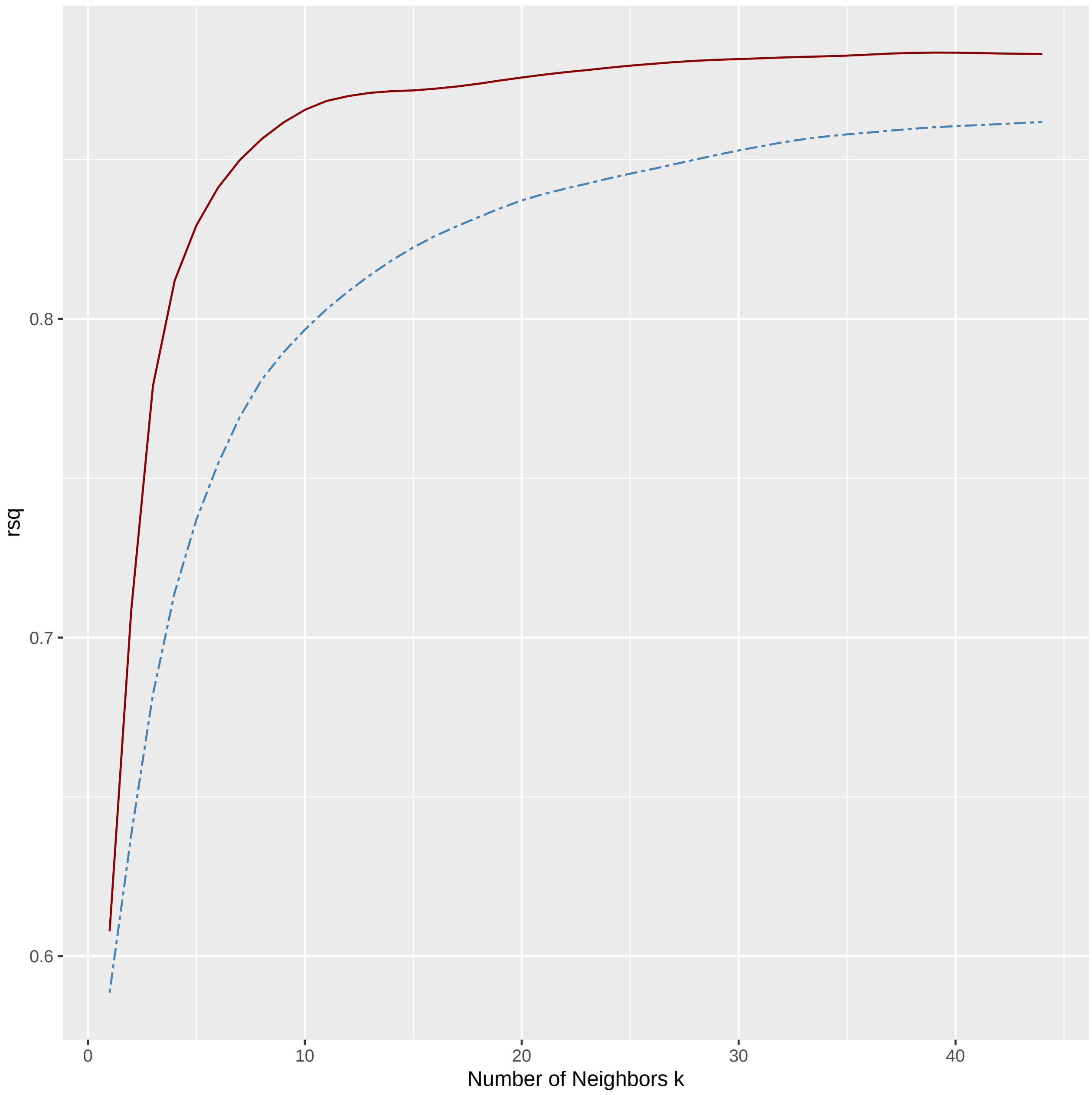


Figure 6: R2 of kNN models predicted against training and testing data, both fitted with training data

Summary

Through our analysis of this data set, we have found that the model that best predicts the ISO of a MLB player with a positive launch angle over a season of at least 502 plate appearances is the LASSO model we created, shown again here.

```
## # A tibble: 53 x 3
##    term        estimate penalty
##    <chr>          <dbl>   <dbl>
##  1 (Intercept)   0.194  0.00001
##  2 PLAYER_AGE    0      0.00001
##  3 PA            0      0.00001
##  4 HR            0      0.00001
##  5 HR_PERCENT    0.0139 0.00001
##  6 K_PERCENT     0      0.00001
##  7 BB_PERCENT    0      0.00001
##  8 BA           -0.0155 0.00001
##  9 SLG           0.0459 0.00001
## 10 OBP           0      0.00001
## # ... with 43 more rows
```

It includes the predictors HR_PERCENT, K_PERCENT, BB_PERCENT, BA, SLG, BRL_PERCENT, PULL_PERCENT, OPPO_PERCENT, GB_PERCENT, HANDEDNESS, POSITION, and DIVISION. With these variables, we can best estimate how much power a hitter will demonstrate in games (according to ISO) over the course of a season. The plot below shows how the ISO values we predicted from our test sample were remarkably close to the actual ISOs posted by those players during the 2021 season.

From an applicability standpoint, this model is interesting in that we can see the relative contributions/importance of various traits and statistics on game power. To see that handedness and position play a role is illuminating. However, if our goal is to predict a player’s ISO using only statistics that reflect player traits or skills, then our original MLR/OLS model (with predictors BRL_PERCENT, PULL_PERCENT, Z_SWING_PERCENT and BB_PERCENT) is most illuminating. While it does not predict ISO with as much certainty, it will allow us to predict future ISOs better because it relies on traits and skills more so than past statistics.

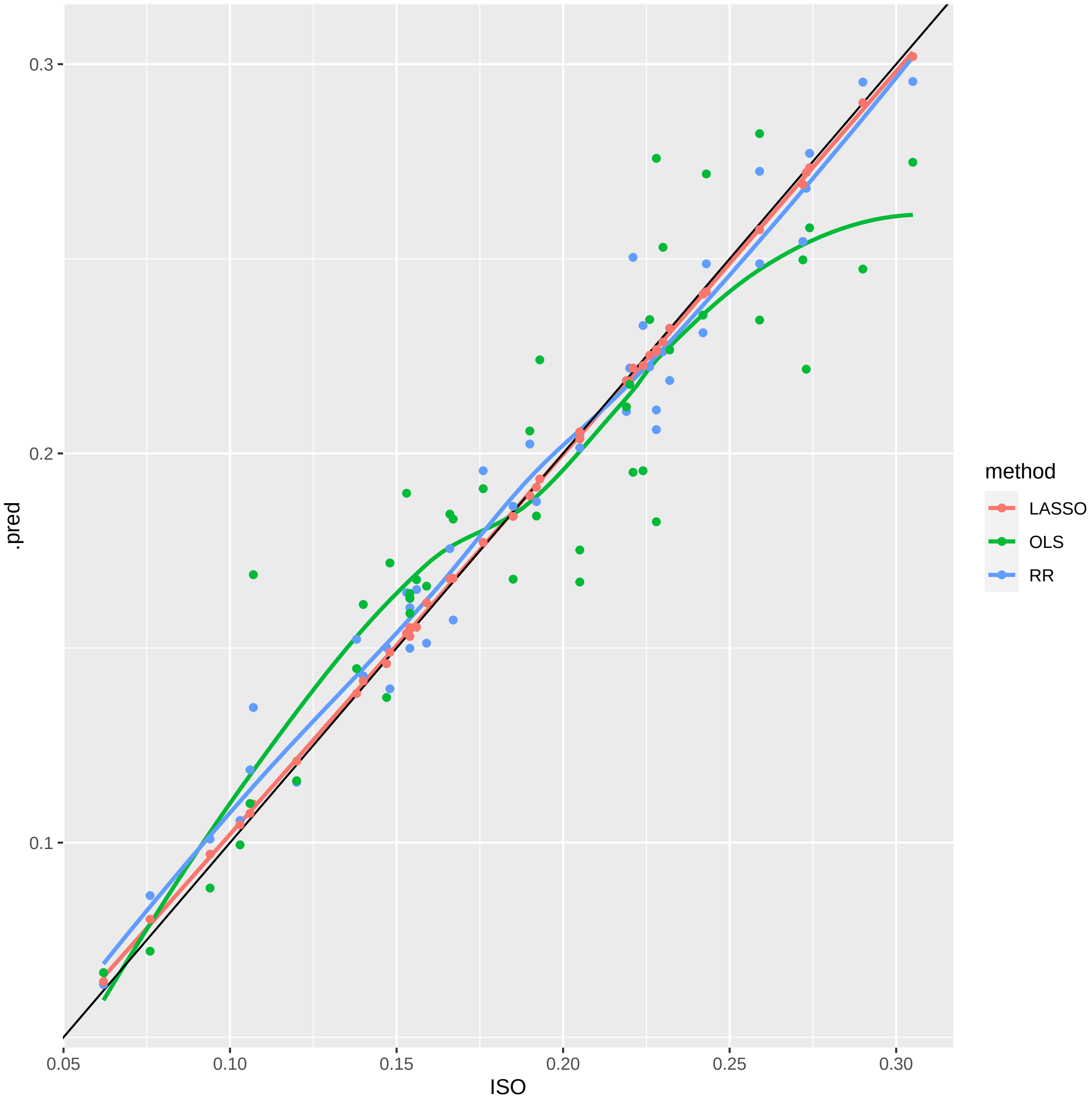


Figure 7: Predicted vs. Observed ISO Values for 3 Models