Analyzing the Relationship Between Bike Shares and Seasonal Conditions

Data Analysis Project Report - STAT 420, Summer 2021, D. Unger

Introduction

In this report, we intend to model bike share frequency in London based on seasonal conditions (e.g. weather conditions, whether it's a holiday, etc.) for purposes of prediction. We are creating this model because these insights would be useful to bike/scooter sharing companies and transport authorities. We also find this data ideal for exploring & applying the techniques we've learned in STAT 420 whilst remaining easily interpretable.

Report Layout

In this report, the Methods section will offer a narrative of the step-by-step decision-making process we followed to arrive at our final model. In the Results section, we will show numerical and graphical summaries of the final model. In the Discussion section, we will discuss our results and frame them in the context of the data.

Dataset Background Information

The data set we chose is the London bike sharing dataset from Kaggle. It consists of the number of new rentals (shares) of bikes owned by the London transport authority (Transport for London). Counts are provided for (nearly) every hour between January 4th, 2015 and January 3rd, 2017. This count is accompanied by weather & date information pertinent to the time the count was recorded (e.g. humidity, wind speed, whether it's a holiday, current season, etc.)

The data set is a data frame with 10 variables and 17,414 observations.

Variable	Description
`timestamp`	Timestamp (hourly)
`cnt`	Number of new bike shares/rentals
`t1`	Real temperature (°C)
`t2`	Relative temperature (°C)
`hum`	Relative humidity (%)
`wind_speed`	Wind speed (km/h)
`weather_code`	Weather type (1 = Clear, 2 = Partly cloudy, 3 = Mostly cloudy, 4 = Overcast, 7 = Rain, 10 = Thunderstorm, 26 = Snow)
`is_holiday`	Is a holiday (0 = non-holiday, 1 = holiday)
`is_weekend`	Is the weekend (0 = weekday, 1 = weekend)
`season`	Meteorological season (0 = spring, 1 = summer, 2 = fall, 3 = winter)

Methods

In this section, we will provide a narrative step-by-step decision-making process throughout our analysis as we adjust the model and attempt to validate model assumptions. Below, we will make use of strategies such as multiple linear regression, dummy variables, interaction terms, residual diagnostics, outlier diagnostics, transformations of the response, and model selection.

Read and Clean the Dataset

A description of the dataset can be found in the Introduction. We converted categorical variables to factors, dropped the timestamp (unusable), and dropped an observation with `cnt = 0` to prevent issues with Box-Cox later.

```
ds = read.csv("london_merged.csv")

ds$weather_code = as.factor(ds$weather_code)
ds$is_holiday = as.factor(ds$is_holiday)
ds$is_weekend = as.factor(ds$is_weekend)
ds$season = as.factor(ds$season)

ds = ds[, setdiff(names(ds), c("timestamp"))] # drop `timestamp` column

ds = ds[ds$cnt > 0, ] # drop 1 observation with `cnt` = 0
```

Perform Train-test Split

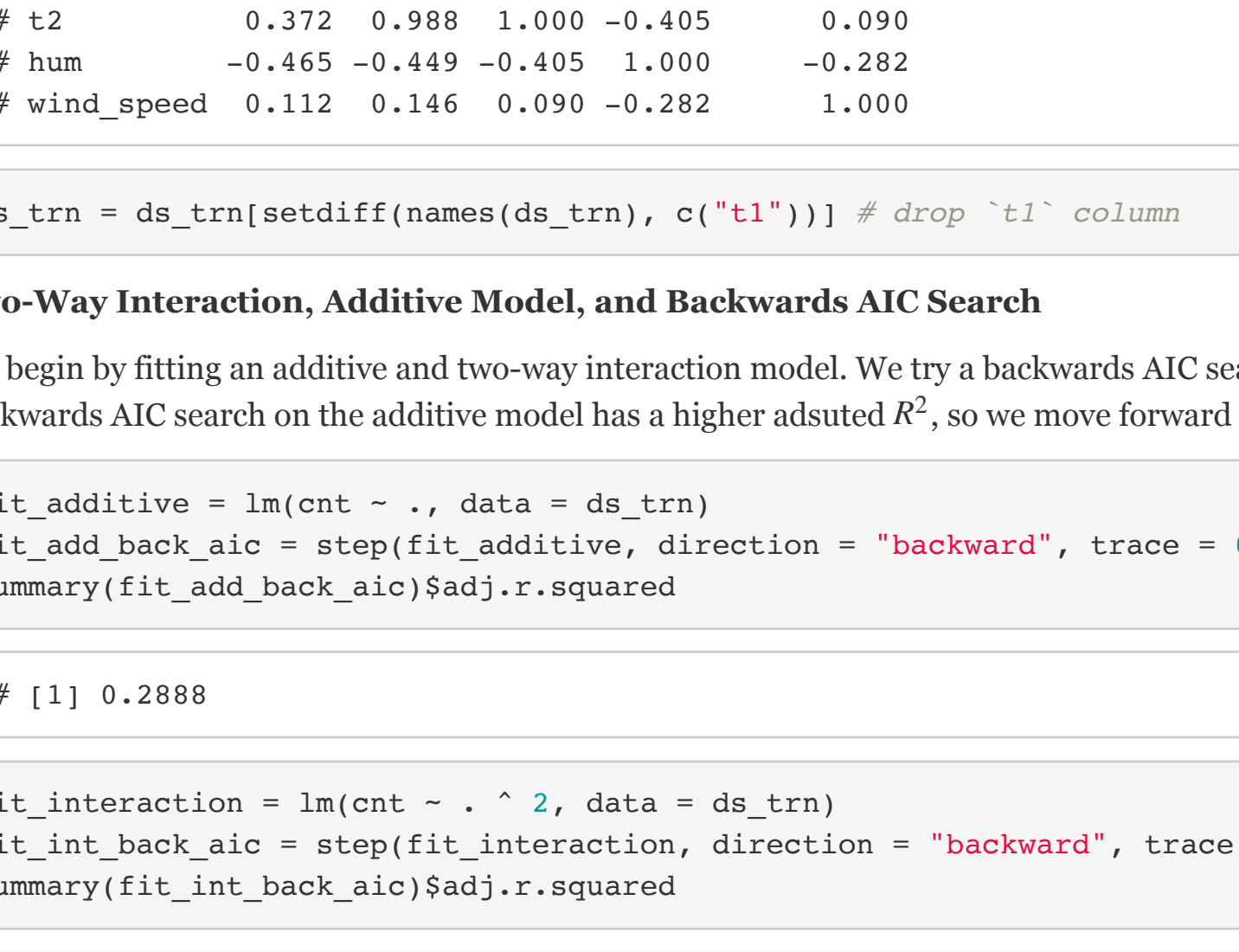
We split the data into a train set & test set so we can get an estimate of model performance on unseen data.

```
set.seed(19990420)
ds_idx = sample(1:nrow(ds), (0.80)*nrow(ds))
ds_trn = ds[ds_idx,]
ds_tst = ds[-ds_idx,]
```

Check Pairwise Correlations

Here, we check the pairs plot and correlation matrix for highly correlated variables. We remove t1 because it's highly correlated with t2.

```
pairs(ds_trn, col="dodgerblue")
```




```
round(cor(ds_trn[, sapply(ds_trn, is.numeric)]), 3)
```

```
##               cnt     t1     t2    hum wind_speed
## cnt         1.000  0.391  0.372 -0.465      0.112
## t1          0.391  1.000  0.988 -0.449      0.146
## t2          0.372  0.988  1.000 -0.405      0.090
## hum        -0.465 -0.449 -0.405  1.000     -0.282
## wind_speed  0.112  0.146  0.090 -0.282      1.000
```

```
ds_trn = ds_trn[setdiff(names(ds_trn), c("t1"))] # drop `t1` column
```

Two-Way Interaction, Additive Model, and Backwards AIC Search

We begin by fitting an additive and two-way interaction model. We try a backwards AIC search on both. The model found via backwards AIC search on the additive model has a higher adsuted R^2, so we move forward with it.

```
fit_additive = lm(cnt ~ ., data = ds_trn)
fit_add_back_aic = step(fit_additive, direction = "backward", trace = 0)
summary(fit_add_back_aic)$adj.r.squared
```

```
## [1] 0.2888
```

```
fit_interaction = lm(cnt ~ . ^ 2, data = ds_trn)
fit_int_back_aic = step(fit_interaction, direction = "backward", trace = 0)
summary(fit_int_back_aic)$adj.r.squared
```

```
## [1] 0.3353
```

Drop Insignificant Variables From Interaction Model

Here, we analyze the significance of the predictors. We decide to drop `windspeed` and `is_holiday` because they have the lowest `Pr(>|t|)`. We fit a new model with this reduced set of predictors.

```
summary(fit_int_back_aic)$call
```

```
## lm(formula = cnt ~ t2 + hum + wind_speed + weather_code + is_holiday +
##     is_weekend + season + t2:hum + t2:is_holiday + t2:is_weekend +
##     t2:season + hum:wind_speed + hum:weather_code + hum:is_holiday +
##     hum:is_weekend + hum:season + wind_speed:weather_code + wind_speed:is_holiday +
##     wind_speed:is_weekend + wind_speed:season + weather_code:is_holiday +
##     weather_code:is_weekend + weather_code:season + is_holiday:season +
##     is_weekend:season, data = ds_trn)
```

```
summary(fit_int_back_aic)$coef[1:15, ] # windspeed and is_holiday are insignificant
```

```
##                  Estimate Std. Error t value  Pr(>|t|)
## (Intercept)     1727.5835    188.441  9.1678 5.500e-20
## t2                88.1914      9.431  9.3512 9.973e-21
## hum              -17.9135      2.340 -7.6554 2.054e-14
## wind_speed         0.6677      5.934  0.1125 9.104e-01
## weather_code2     54.5473    145.761  0.3742 7.082e-01
## weather_code3    -86.4679    172.991 -0.4998 6.172e-01
## weather_code4  -2210.6179    263.564 -8.3874 5.446e-17
## weather_code7    351.9950    245.921  1.4313 1.524e-01
## weather_code10  1126.0745   4056.305  0.2776 7.813e-01
## weather_code26   415.6930   3373.443  0.1232 9.019e-01
## is_holiday1      949.2060    580.054  1.6364 1.018e-01
## is_weekend1      946.1718    139.060  6.8040 1.059e-11
## season1         -377.8921    220.697 -1.7123 8.687e-02
## season2          781.8681    198.463  3.9396 8.201e-05
## season3          264.9122    199.248  1.3296 1.837e-01
```

```
fit_rm_insig = lm(cnt ~ t2 + hum + weather_code + is_weekend + season + t2:hum + t2:is_weekend +
t2:season + hum:weather_code + hum:is_weekend + hum:season + weather_code:is_weekend + weather_c
ode:season + is_weekend:season, data = ds_trn)
```

Checking Model Assumptions & Performance

Here, we check for model assumptions & overfitting. We find that the LOOCV-RMSE is infinite (likely due to high-leverage points). We also find the adjusted R^2 for comparison. The Fitted vs Residuals shows that the constant variance and linearity assumptions are suspect. The Q-Q Plot shows that the normality assumption is suspect.

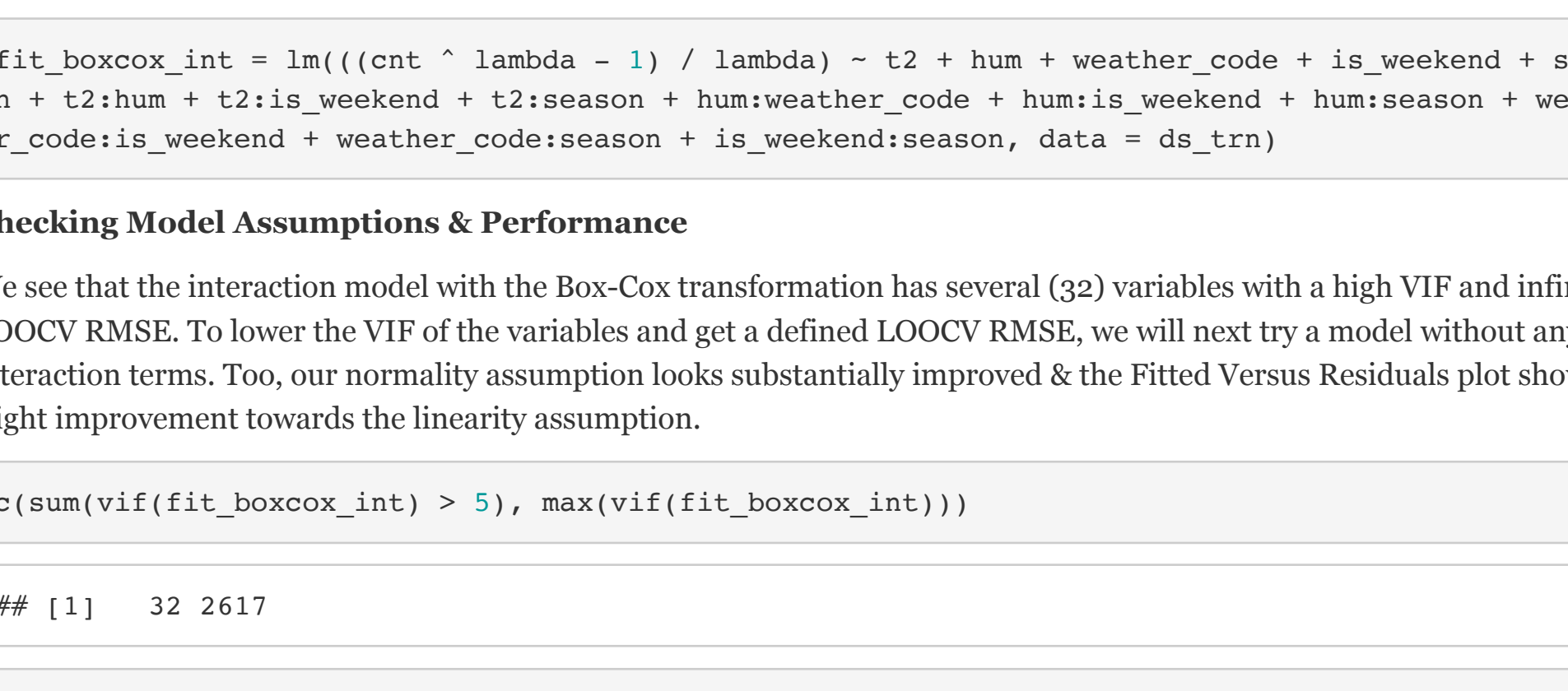
```
calc_loocv_rmse = function(model) {
  sqrt(mean((resid(model) / (1 - hatvalues(model))) ^ 2))
}

calc_loocv_rmse(fit_rm_insig)
```

```
## [1] Inf
```

```
summary(fit_rm_insig)$adj.r.squared
```

```
## [1] 0.3296
```

```
plot_diagnostics(fit_rm_insig)
```



Box-Cox Transformation of the Response

Here, we try a box-cox transformation of the response variable.

```
bc = boxcox(fit_rm_insig, plotit = TRUE, lambda = seq(0.2, 0.35, by = 0.01))
```

```
(lambda = bc$x[which.max(bc$y)])
```

```
## [1] 0.2712
```

```
fit_boxcox_int = lm(((cnt ^ lambda - 1) / lambda) ~ t2 + hum + weather_code + is_weekend + seaso
n + t2:hum + t2:is_weekend + t2:season + hum:weather_code + hum:is_weekend + hum:season + weathe
r_code:is_weekend + weather_code:season + is_weekend:season, data = ds_trn)
```

Checking Model Assumptions & Performance

We see that the interaction model with the Box-Cox transformation has several (32) variables with a high VIF and infinite LOOCV RMSE. To lower the VIF of the variables and get a defined LOOCV RMSE, we will next try a model without any interaction terms. Too, our normality assumption looks substantially improved & the Fitted Versus Residuals plot shows slight improvement towards the linearity assumption.

```
c(sum(vif(fit_boxcox_int) > 5), max(vif(fit_boxcox_int)))
```

```
## [1]   32 2617
```

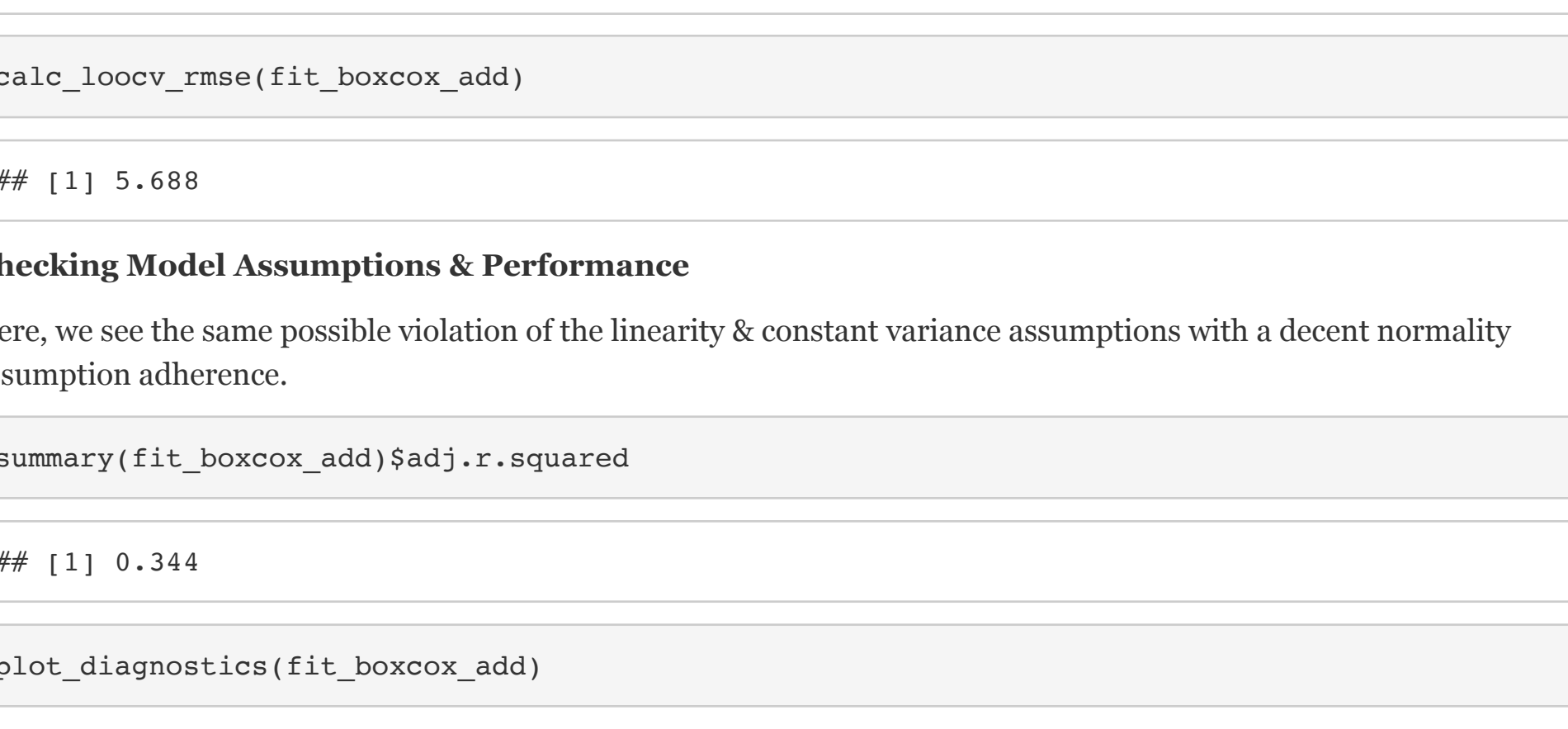
```
calc_loocv_rmse(fit_boxcox_int)
```

```
## [1] Inf
```

```
summary(fit_boxcox_int)$adj.r.squared
```

```
## [1] 0.3742
```

```
plot_diagnostics(fit_boxcox_int)
```



Remove Interaction Terms and Compare

Here, we modify the Box-Cox model to remove interaction terms. We prefer a simpler model where possible. This new model has a good LOOCV RMSE and none of its predictors have a high VIF (> 5).

```
fit_boxcox_add = lm(formula = ((cnt ^ lambda - 1) / lambda) ~ hum + season + weather_code + t2 +
is_weekend, data = ds_trn)

sum(vif(fit_boxcox_add) > 5)
```

```
## [1] 0
```

```
calc_loocv_rmse(fit_boxcox_add)
```

```
## [1] 5.688
```

Checking Model Assumptions & Performance

Here, we see the same possible violation of the linearity & constant variance assumptions with a decent normality assumption adherence.

```
summary(fit_boxcox_add)$adj.r.squared
```

```
## [1] 0.344
```

```
plot_diagnostics(fit_boxcox_add)
```

High leverage, Outlier, and Influential Observations

Here, we check for high leverage, outlier, and influential observations. We see that about 4% of the observations are influential (high leverage AND large residual).

```
unusual_observations(fit_boxcox_add)
```

```
## [1] "High Leverage: 62  points, 0.45% of points"
## [1] "Outliers:      654 points, 4.69% of points"
## [1] "Influential:   435 points, 3.12% of points"
```

Remove influential points

Here, we refit the Box-Cox additive model after removing influential observations. The coefficients change but not drastically.

```
cooks_dists = cooks.distance(fit_boxcox_add)
fit_boxcox_rm_infl = lm(formula = ((cnt ^ lambda - 1) / lambda) ~ hum + season + weather_code +
t2 + is_weekend, data = ds_trn, subset = cooks_dists <= 4 / length(cooks_dists))

coef(fit_boxcox_rm_infl)
```

```
##    (Intercept)            hum        season1        season2        season3
##        29.9074        -0.2292        -1.0325         1.2067         1.6361
##  weather_code2  weather_code3  weather_code4  weather_code7 weather_code10
##         2.4961         2.8226        -0.7725         0.7407        -0.4841
## weather_code26             t2    is_weekend1
##        -1.5231         0.3160        -0.4829
```

Check Model Assumptions and Performance

Here, we see a greatly improved Adjusted R^2, still good VIFs, and a good LOOCV RMSE. We see that this model has the best adjusted R^2 yet. The model still doesn't quite follow the constant variance or linearity assumption, but the normality assumption does seem to be satisfied.

```
summary(fit_boxcox_rm_infl)$adj.r.squared
```

```
## [1] 0.4186
```

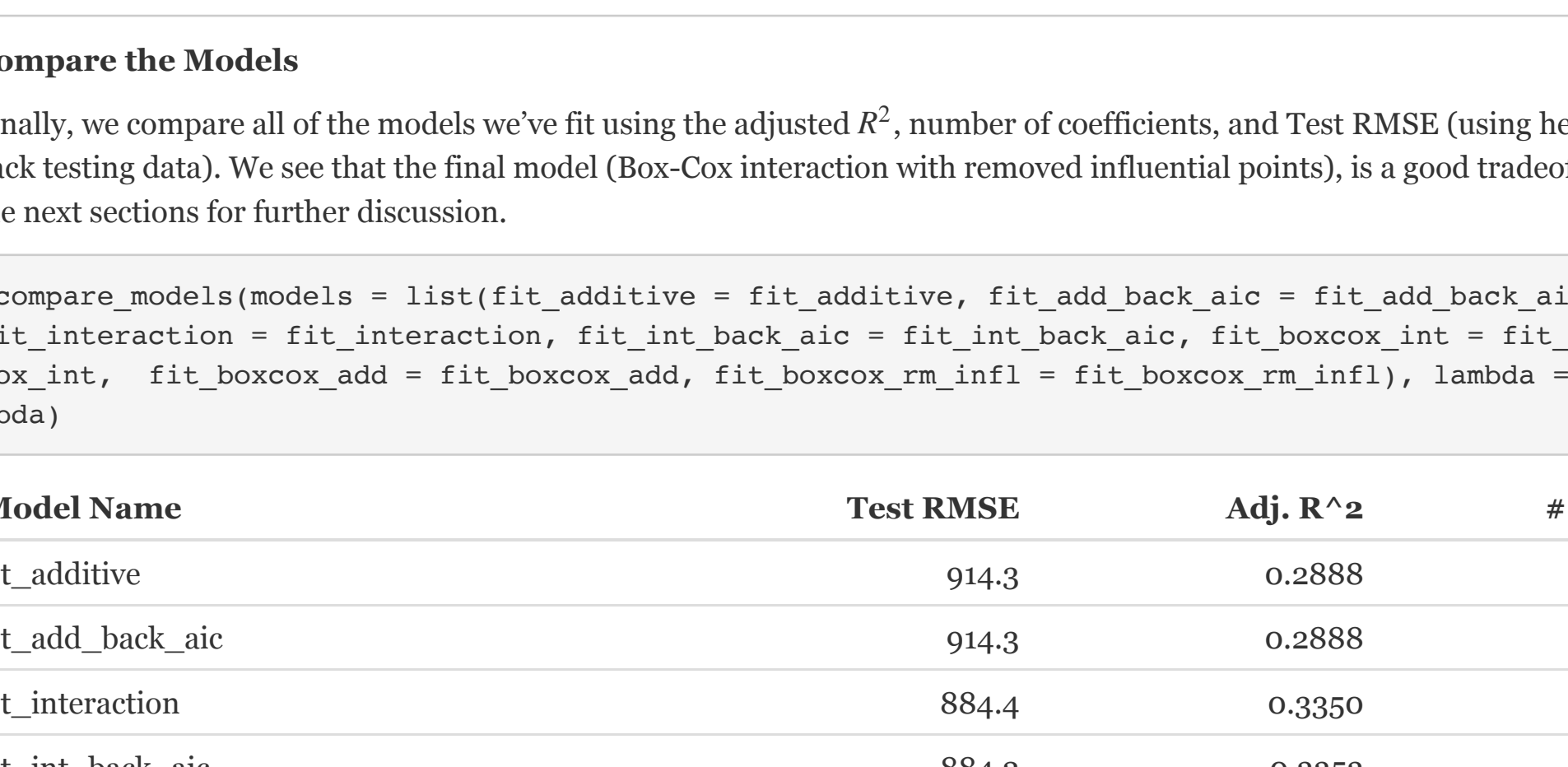
```
vif(fit_boxcox_rm_infl)
```

```
##            hum        season1        season2        season3  weather_code2
##          1.598          2.330          1.732          1.633          1.273
##  weather_code3  weather_code4  weather_code7 weather_code10 weather_code26
##          1.341          1.204          1.338          1.001          1.009
##             t2    is_weekend1
##          2.415          1.011
```

```
calc_loocv_rmse(fit_boxcox_rm_infl)
```

```
## [1] 5.224
```

```
plot_diagnostics(fit_boxcox_rm_infl)
```



High Leverage, Outlier, and Influential Observations

Here, we see the proportion of influential points has improved greatly and the percent of outliers and high leverage points has also been reduced.

```
unusual_observations(fit_boxcox_rm_infl)
```

```
## [1] "High Leverage: 27  points, 0.2% of points"
## [1] "Outliers:      603 points, 4.47% of points"
## [1] "Influential:   282 points, 2.09% of points"
```

Compare the Models

Finally, we compare all of the models we've fit using the adjusted R^2, number of coefficients, and Test RMSE (using held-back testing data). We see that the final model (Box-Cox interaction with removed influential points), is a good tradeoff. See the next sections for further discussion.

```
compare_models(models = list(fit_additive = fit_additive, fit_add_back_aic = fit_add_back_aic, f
it_interaction = fit_interaction, fit_int_back_aic = fit_int_back_aic, fit_boxcox_int = fit_boxc
ox_int,  fit_boxcox_add = fit_boxcox_add, fit_boxcox_rm_infl = fit_boxcox_rm_infl), lambda = lam
bda)
```

Model Name	Test RMSE	Adj. R^2	# Coefs
fit_additive	914.3	0.2888	15
fit_add_back_aic	914.3	0.2888	15
fit_interaction	884.4	0.3350	88
fit_int_back_aic	884.2	0.3353	80
fit_boxcox_int	928.3	0.3742	55
fit_boxcox_add	950.2	0.3440	13
fit_boxcox_rm_infl	958.8	0.4186	13

Results

The previous section (Methods) discussed our approach to finding a good model for our dataset. Readers should refer to it for the step-by-step narrative and intermediate models. The model we settled on was of the form `(cnt ^ 0.2712 - 1) / 0.2712 ~ hum + season + weather_code + t2 + is_weekend`.

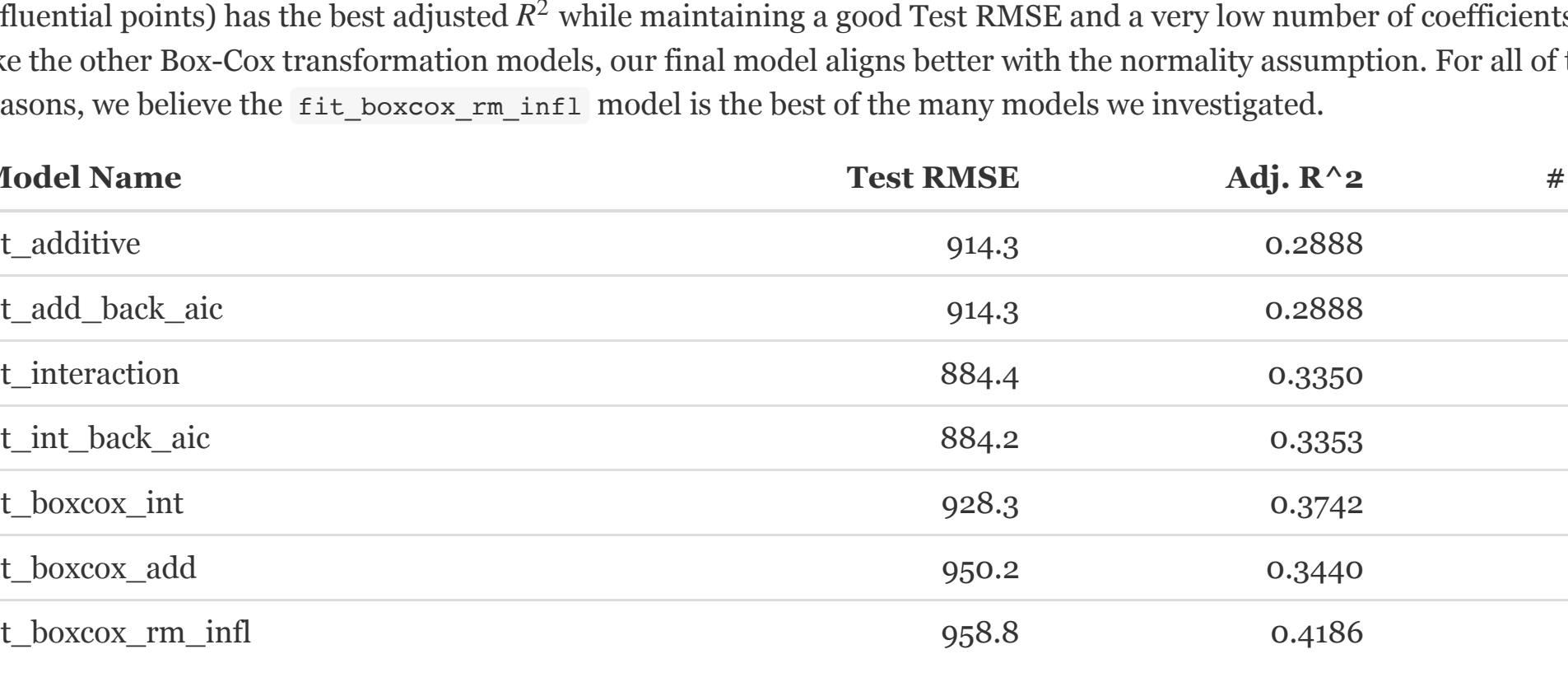
Final Model Model Assumptions and Performance

Our final model has the highest adjusted R^2 we could find. It also has good VIFs and a good LOOCV-RMSE. The final model, like the others doesn't seem to adhere to constant variance or linearity assumptions, but the normality assumption does seem to be satisfied.

```
## [1] 0.4186
```

```
##            hum        season1        season2        season3  weather_code2
##          1.598          2.330          1.732          1.633          1.273
##  weather_code3  weather_code4  weather_code7 weather_code10 weather_code26
##          1.341          1.204          1.338          1.001          1.009
##             t2    is_weekend1
##          2.415          1.011
```

```
## [1] 5.224
```



Final Model High Leverage, Outlier, and Influential Observations

Our final model has low proportions of high leverage, outlier, and influential observations.

```
## [1] "High Leverage: 27  points, 0.2% of points"
## [1] "Outliers:      603 points, 4.47% of points"
## [1] "Influential:   282 points, 2.09% of points"
```

Final Model Comparison

To further justify our choice of final model, we compare all of the models we've fit using the adjusted R^2, number of coefficients, and Test RMSE (using held-back testing data). We see that the final model (Box-Cox interaction with removed influential points) has the best adjusted R^2 while maintaining a good Test RMSE and a very low number of coefficients. Too, like the other Box-Cox transformation models, our final model aligns better with the normality assumption. For all of these reasons, we believe the `fit_boxcox_rm_infl` model is the best of the many models we investigated.

Model Name	Test RMSE	Adj. R^2	# Coefs
fit_additive	914.3	0.2888	15
fit_add_back_aic	914.3	0.2888	15
fit_interaction	884.4	0.3350	88
fit_int_back_aic	884.2	0.3353	80
fit_boxcox_int	928.3	0.3742	55
fit_boxcox_add	950.2	0.3440	13
fit_boxcox_rm_infl	958.8	0.4186	13

Discussion

In the Methods section, we offered a narrative of the step-by-step decision-making process we followed to arrive at our final model. In the results section, we showed numerical and graphical summaries of the final model. In this section, we will discuss our results and frame them in the context of the data.

The final model we settled on was of the form `(cnt ^ 0.2712 - 1) / 0.2712 ~ hum + season + weather_code + t2 + is_weekend`. It has the highest adjusted R^2 we could find. It also has good VIFs and a good LOOCV-RMSE. The final model, like the others doesn't seem to adhere to constant variance or linearity assumptions, but the normality assumption does seem to be satisfied. An addition, we compared all of the models we've fit using the adjusted R^2, number of coefficients, and Test RMSE (using held-back testing data). We see that the final model (Box-Cox interaction with removed influential points) has the best adjusted R^2 while maintaining a good Test RMSE and a very low number of coefficients. Too, like the other Box-Cox transformation models, our final model aligns better with the normality assumption. For all of these reasons, we believe the `fit_boxcox_rm_infl` model is the best of the many models we investigated.

The original goal of this model is to predict bike share frequency in London based on seasonal conditions (e.g. weather conditions, whether it's a holiday, etc.) for purposes of prediction. Our final model could offer useful insights to bike/scooter sharing companies and transport authorities. This model also helped us explore & apply the techniques learned in STAT 420 whilst remaining easily interpretable.

We believe our model achieves this goal as it was able to perform with a 959 RMSE on a held-out test dataset. It also comes closest to meeting LINE assumptions. Because it has few coefficients and no interaction terms, it remains easy to interpret and understand. Too, it has the highest adjusted R^2 of the models we tried.

So, we are satisfied with the final model we've found and believe it would prove useful in the real-world problem of predicting bike share frequency based on seasonal conditions.

Appendix

Helper Functions

Diagnostics Plots:

This function plots the Fitted versus Residual and Q-Q Plot for a model.

```
plot_diagnostics = function(model, pcol = "grey", lcol = "dodgerblue") {
  layout(matrix(c(1,2), nrow = 1, ncol = 2, byrow = TRUE))
  par(mfrow=c(1,2))
  plot(fitted(model), resid(model), col = pcol, xlab = "Fitted", ylab = "Residuals", main = "Fit
ted versus Residuals")
  abline(h = 0, col = lcol)
  qqnorm(resid(model), col = pcol, main = "Normal Q-Q Plot")
  qqline(resid(model), col = lcol)
}
```

Unusual Observations

This function prints out the number and proportion of high leverage, outlier, and influcential observations.

```
unusual_observations = function(model) {
  leverages = hatvalues(model)
  print(paste("High Leverage: ", sum(leverages > 2*mean(leverages)), " points, ", round(mean(lev
erages > 2*mean(leverages)) * 100, 2), "% of points", sep = ""))

  rstandard_abs = abs(rstandard(model))
  print(paste("Outliers:      ", sum(rstandard_abs > 2), " points, ", round(mean(rstandard_abs >
2) * 100, 2), "% of points", sep = ""))

  cooks_dists = cooks.distance(model)
  cd_cutoff = 4 / length(cooks_dists)
  print(paste("Influential:   ", sum(cooks_dists > cd_cutoff), " points, ", round(mean(cooks_dis
ts > cd_cutoff) * 100, 2), "% of points", sep = ""))
}
```

Compare Models

This function outputs a table comparing models using the test RMSE, adjusted R^2, and number of betas.

```
compare_models = function(models, lambda) {
  results = data.frame(name = character(), test_rmse = numeric(), adj_r_squared = numeric(), num
_betas = numeric())

  for (name in names(models)) {
    model = models[[name]]
    pred = predict(model, newdata = ds_tst)
    test_rmse = ifelse(grepl("boxcox", name, fixed = TRUE), rmse(ds_tst$cnt, (pred  * lambda + 1
) ^ (1 / lambda)), rmse(ds_tst$cnt, pred))
    adj_r_squared = summary(model)$adj.r.squared
    num_betas = length(coef(model))
    results[nrow(results) + 1, ] = list(name, test_rmse, adj_r_squared, num_betas)
  }

  kable(results, col.names = c("Model Name", "Test RMSE", "Adj. R^2", "# Coefs"))
}
```


Group Members

- Darci Peoples (darciap2)
- Alex Koczwara (alexk3)
- Yash Patel (ypatel42)