Homework 3

Due Wednesday, October 20th at 11:59pm on Moodle (https://moodle.macalester.edu/mod/assign/view.php?id=27981)

Deliverables: Please use this template (template_rmds/hw3.Rmd) to knit an HTML document. Convert this HTML document to a PDF by opening the HTML document in your web browser. *Print* the document (Ctrl/Cmd-P) and change the destination to "Save as PDF". Submit this one PDF to Moodle.

Alternatively, you may knit your Rmd directly to PDF if you have LaTeX installed.

```
# Library statements
library(dplyr);
library(readr);
library(broom);
library(ggplot2);
library(tidymodels);
tidymodels_prefer();
```

Project Work

(Note: This includes HW2 investigations plus a few tasks for dealing with non-linearity.)

Goal: Begin an analysis of your dataset to answer your **regression** research question.

Collaboration: Form a team (2-3 members) for the project and this part can be done as a team. Only one team member should submit a Project Work section. Make sure you include the full names of all of the members in your write up.

Data cleaning: If your dataset requires any cleaning (e.g., merging datasets, creation of new variables), first consult the R Resources page (r-resources.html) to see if your questions are answered there. If not, post on the #rcode-questions channel in our Slack workspace to ask for help. *Please ask for help early and regularly* to avoid stressful workloads.

Required Analyses:

1. **Initial investigation: ignoring nonlinearity (for now)**

Note: after this process, you might have a set of models (one of which has predictors chosen using LASSO, one model with all the predictors of interest, and perhaps some models with subsets of predictors that were chosen based on the data context rather than an algorithmic process)

a. Use ordinary least squares (OLS) by using the `lm` engine and LASSO (`glmnet` engine) to build a series of initial regression models for your quantitative outcome as a function of the predictors of interest. (As part of data cleaning, exclude any variables that you don't want to consider as predictors.)
    - You'll need two model specifications, `lm_spec` and `lm_lasso_spec` (you'll need to tune this one).
b. For each set of variables, you'll need a `recipe` with the `formula`, `data`, and pre-processing steps
    - You may want to have steps in your recipe that remove variables with near zero variance (`step_nzv()`), remove variables that are highly correlated with other variables (`step_corr()`), normalize all quantitative predictors (`step_normalize(all_numeric_predictors())`) and add indicator variables for any categorical variables (`step_dummy(all_nominal_predictors())`).
    - These models should not include any transformations to deal with nonlinearity. You'll explore this in the next investigation.

```
# Read in data
music <- read_csv("tcc_ceds_music.csv")

# Clean data
music_clean <- select(music, - "...1", -artist_name, -track_name)
head(music_clean)

set.seed(253)
```

```
# Creation of CV folds
music_cv <- vfold_cv(data = music_clean, v = 10)

# Model specs
lm_spec <-
    linear_reg() %>%
    set_engine(engine = 'lm') %>%
    set_mode('regression')

lm_lasso_spec_tune <-
  linear_reg() %>%
  set_args(mixture = 1, penalty = tune()) %>% ## mixture = 1 indicates Lasso
  set_engine(engine = 'glmnet') %>% #note we are using a different engine
  set_mode('regression')
```

```
# Recipes & workflows
data_rec <- recipe(release_date ~ danceability + loudness + acousticness + instrumentaln
ess + valence + energy, data = music_clean) %>%
    step_nzv(all_predictors()) %>% # removes variables with the same value
    step_corr()

ols_wf <- workflow() %>%
  add_recipe(data_rec) %>%
  add_model(lm_spec)

lasso_wf_tune <-  workflow() %>%
  add_recipe(data_rec) %>%
  add_model(lm_lasso_spec_tune)
```

```
# Fit & tune models
## Fit OLS model
ols_mod <- ols_wf %>%
  fit(data = music_clean)

## Fit & tune LASSO model
lasso_mod <- lasso_wf_tune %>%
  fit(data = music_clean)

penalty_grid <- grid_regular(
  penalty(range = c(-3, 1)), #log10 transformed
  levels = 30)

tune_output <- tune_grid( # new function for tuning parameters
  lasso_wf_tune, # workflow
  resamples = music_cv, # cv folds
  metrics = metric_set(rmse, rsq, mae),
  grid = penalty_grid # penalty grid defined above
)
```

c. Estimate the test performance of the models using CV. Report and interpret (with units) the CV metric estimates along with a measure of uncertainty in the estimate (`std_error` is readily available when you used `collect_metrics(summarize=TRUE)`).
    - Compare estimated test performance across the models. Which models(s) might you prefer?

```
# Calculate/collect CV metrics
## OLS model's CV metrics
ols_cv <- fit_resamples(ols_wf,
  resamples = music_cv,
  metrics = metric_set(rmse, rsq, mae)
)

ols_cv %>% collect_metrics(summarize=TRUE)

## LASSO models CV metrics
tune_output %>% collect_metrics(summarize=TRUE)

metrics_output <- collect_metrics(tune_output) %>%
  filter(.metric == 'mae')

metrics_output %>% ggplot(aes(x = penalty, y = mean)) +
  geom_point() +
  geom_line() +
  labs(x = 'Lambda', y = 'CV MAE') +
  scale_x_log10() +
  theme_classic()

## Best LASSO model's CV MAE
best_penalty <- select_best(tune_output, metric = 'mae') # choose penalty value based on
lowest cv mae
best_penalty

best_se_penalty <- select_by_one_std_err(tune_output, metric = 'mae', desc(penalty)) # c
hoose largest penalty value within 1 se of the lowest cv mae
best_se_penalty
```

MAE results from cross-validation of the OLS model and the best LASSO models (penalty = 0.001, and penalty = 0.1610262) are respectively 12.0784177 ± 0.04368178 years, 12.0838520 ± 0.043719815 years, and 12.11439 ± 0.04423626 years. In the best case scenario (subtracting the standard error from the mean MAE), predictions of these models would be off by 12.03473592 years, 12.040132185 years, and 12.07015374 years respectively. This means that on average, our predictions are about 12 years sooner or later than the real release year, with the OLS model having a slightly closer guess than the best LASSO models.

Comparison of the MAE of the models show that the OLS model (`ols_mod`) is the best, because it has the lowest average MAE after 10-fold cross-validation (so there is few to none overfitting).

```
d. Use residual plots to evaluate whether some quantitative predictors might be better m
odeled with nonlinear relationships.
```

```
ols_mod_output <- ols_mod %>%
    predict(new_data = music_clean) %>%
    bind_cols(music_clean) %>%
    mutate(resid = release_date - .pred) #observed - predicted

head(ols_mod_output)
```

```r
# Visualize residuals
## Residuals vs. release_date (outcome variable)
ggplot(ols_mod_output, aes(x = release_date, y = resid)) +
    geom_point() +
    geom_smooth() +
    geom_hline(yintercept = 0, color = "red")

## Residuals vs. danceability
ggplot(ols_mod_output, aes(x = danceability, y = resid)) +
    geom_point() +
    geom_smooth() +
    geom_hline(yintercept = 0, color = "red") +
    theme_classic()

## Residuals vs. loudness
ggplot(ols_mod_output, aes(x = loudness, y = resid)) +
    geom_point() +
    geom_smooth() +
    geom_hline(yintercept = 0, color = "red") +
    theme_classic()

## Residuals vs. acousticness
ggplot(ols_mod_output, aes(x = acousticness, y = resid)) +
    geom_point() +
    geom_smooth() +
    geom_hline(yintercept = 0, color = "red") +
    theme_classic()

## Residuals vs. instrumentalness
ggplot(ols_mod_output, aes(x = instrumentalness, y = resid)) +
    geom_point() +
    geom_smooth() +
    geom_hline(yintercept = 0, color = "red") +
    theme_classic()

## Residuals vs. valence
ggplot(ols_mod_output, aes(x = valence, y = resid)) +
    geom_point() +
    geom_smooth() +
    geom_hline(yintercept = 0, color = "red") +
    theme_classic()

## Residuals vs. energy
ggplot(ols_mod_output, aes(x = energy, y = resid)) +
    geom_point() +
    geom_smooth() +
    geom_hline(yintercept = 0, color = "red") +
    theme_classic()
```

The residual plots against all predictors do not show any clear trend, implying that there is no bias where predictions tend to be underestimated/overestimated at some ranges of the predictors' spectra. If we are to be strict, though, there is a slightly downward trend in the residual plot against `loudness`, which should prompt us to remodel this predictor using some nonlinear relationship.

Besides, our residual plot against predicted outcomes show that our model tends to underestimate the release year of songs from 1988 onward, and overestimate songs before that year. Since this residual plot has a very linear, upward trending pattern while the residual plots against predictors do not show any clear trend, we decided to see what happens by plotting the density plots of predictions and observed outcomes.

```
# visualize predicted vs. observed data
ggplot(ols_mod_output, aes(x = .pred)) +
    geom_density() +
    geom_vline(aes(xintercept=mean(.pred)),
            color="red", linetype="dashed", size=1) +
    theme_classic()

ggplot(ols_mod_output, aes(x = release_date)) +
    geom_density() +
    geom_vline(aes(xintercept=mean(.pred)),
            color="blue", linetype="dashed", size=1) +
    theme_classic()
```

It seems that we do not have a very strong prediction power in our model, because with all the predictors included, our model still tends to average out the predictions (shown by a normal distribution in the density plot). This is also quantified by the R-squared of `ols_mod`, which is only 37.01227%. Although it is rather disappointing to have an R-squared this low for the best model, in the context of this dataset and our research question, it makes sense, because musical trends are usually not clear-cut, and there can be a lot of "mix and match" where typical features of one "era" are brought into a later "era". Therefore, variations in songs' release years should not be accounted too much by musical features.

However, a look into the coefficients of these musical feature variables still tells us many interesting insights.

```
e. Which variables do you think are the most important predictors of your quantitative o
utcome? Justify your answer. Do the methods you've applied reach consensus on which vari
ables are most important? What insights are expected? Surprising?
    - Note that if some (but not all) of the indicator terms for a categorical predictor
are selected in the final models, the whole predictor should be treated as selected.
```

```
# Show coefficient estimates
## OLS model
ols_mod %>% tidy()
ols_mod_output %>% summarise(mean = mean(.pred), sd = sd(.pred), max = max(.pred), min =
min(.pred))

## LASSO models
final_wf <- finalize_workflow(lasso_wf_tune, best_penalty) # incorporates penalty value
 to workflow
final_fit <- fit(final_wf, data = music_clean)

final_fit %>% tidy()

final_wf_se <- finalize_workflow(lasso_wf_tune, best_se_penalty) # incorporates penalty
 value to workflow
final_fit_se <- fit(final_wf_se, data = music_clean)
tidy(final_fit_se)
```

Based on the absolute values of the coefficient estimates, the most important predictors in our model are in order: loudness, danceability, valence, and acousticness. This order of importance is also true for the best LASSO models.

From the coefficients, we can infer that in general, later songs are more danceable, a lot louder, less acoustic, slightly more instrumental, less positive (valence), and have just a little more energy. The trends of higher danceability and less acousticness are expected, but it is quite surprising that loudness plays such an important role (twice as much as danceability - the second most important predictor) in our model. This could be due to the advancement in audio recording technologies over time, given that 1975 marks the start of the Digital Era (https://en.wikipedia.org/wiki/History_of_sound_recording).

Another interesting insight is in the considerable, negative coefficient of valence, which is the musical positiveness (happy, cheerful, euphoric vs. sad, depressed, angry) conveyed by a track. Up to nearly 25 years is the estimated difference in release year between the happiest song and the saddest song possible. One plausible explanation for this trend is the rise of musical genres like Emo, Indie, Gothic Metal, Doom Metal, etc. However, this needs a closer look into the `genre` column of our dataset, since the dataset may not include songs of these musical genres or may not represent the total population with regards to these movements.

OLS has been the overall best model so far. Among our predictors, `energy` is the only one that seems irrelevant, whose coefficient is shrunk to 0 in LASSO model 17 (penalty = 0.1610262). However, because `energy` 's coefficient in the OLS model is only 0.2156978 years (which means it does not affect the outcome much given the outcome unit is years), and because OLS overall generates slightly more accurate predictions than LASSO, we chose OLS to be our final model. This is for the sake of both predictive accuracy and interpretability of the model output (while LASSO here only excludes one less relevant predictor, interpretation of OLS models are generally much simpler).

2. **Accounting for nonlinearity**
    - Update your models to use natural splines for some of the quantitative predictors to account for non-linearity (these are GAMs).
        - I recommend using OLS engine to fit these final models.

- You'll need to update the recipe to include `step_ns()` for each quantitative predictor that you want to allow to be non-linear.
- To determine number of knots (`deg_free`), I recommend fitting a smoothing spline and use `edf` to inform your choice.

In our residual plots above when assuming all linear terms, we see that `loudness` should surely be adjusted for nonlinearity. However, since other variables also have some degree of wiggliness in their residual plots against the predictor, we will first fit a model where all of them are smooth terms. Also based on the residual plots, our expectation is that `danceability` and `valence` will have smaller `edf` s than other variables.

Note that we choose to include `energy` , because OLS in the previous section has proven to be slightly more accurate compared to LASSO where `energy` coefficient is shrink down to 0.

```
# GAM model spec
gam_spec <-
  gen_additive_mod() %>%
  set_engine(engine = 'mgcv') %>%
  set_mode('regression')

# GAM model
gam_mod <- fit(gam_spec,
               release_date ~ s(danceability) + s(loudness) + s(acousticness) + s(instru
mentalness) + s(valence) + s(energy),
               data = music_clean)
```

```
gam_mod %>% pluck('fit') %>% summary()
gam_mod %>% pluck('fit') %>% plot( all.terms = TRUE, pages = 1)
```

As expected, the `danceability` and `valence` splines are more of a straight line than others. Their `edf` are also the smallest. Interestingly, all p-values are significantly small (<2e-16), and all splines are marked as "***", meaning they all are significant. This further supports our inclusion of `energy` . All in all, we decide to continue including all variables, with `loudness` , `acousticness` , `instrumentalness` , and `energy` being smooth terms, and `danceability` and `valence` being linear terms. This is to make the model simpler and less computationally intensive while still keeping the overall accuracy.

```
gam_mod_2 <- fit(gam_spec,
               release_date ~ danceability + s(loudness) + s(acousticness) + s(instrumen
talness) + valence + s(energy),
               data = music_clean)
```

```
gam_mod_2 %>% pluck('fit') %>% summary()

gam_mod_2 %>% pluck('fit') %>% plot()
```

Except for `loudness` , the `edf` s of all smooth terms slightly decrease. All p-values remain significantly small, and all variables continues being marked as "***" - highly significant. Our R-squared very slightly decreases (from 0.403 to 0.402), meaning the overall performance of this model (`gam_mod_2`) remains the same compared to `gam_mod` .

Finally, we check to see if the number of knots in any smooth terms should be increased. If nothing changes, the degree of freedom for `loudness`, `acousticness`, `instrumentalness` and `energy` will be respectively 8, 8, 7, 7 in our final GAM model with recipes.

```
# Diagnostics: Check to see if the number of knots is large enough (if p-value is low, i
ncrease number of knots)
par(mfrow=c(2,2))
gam_mod_2 %>% pluck('fit') %>% mgcv::gam.check()
```

The `edf` is not too close to k' for all smooth terms, and they all have high p-values. Therefore, we do not change the number of knots for any smooth terms.

Finally, we apply a GAM model with recipe (using `step_ns()`) using the edf from above.

```
# Model spec
lm_spec <-
  linear_reg() %>%
  set_engine(engine = 'lm') %>%
  set_mode('regression')

# General recipe
music_rec <- recipe(release_date ~ danceability + loudness + acousticness + instrumental
ness + valence + energy, data = music_clean)

# Adding splines
spline_rec <- music_rec %>%
    step_ns(loudness, deg_free = 8) %>%
    step_ns(acousticness, deg_free = 8) %>%
    step_ns(instrumentalness, deg_free = 7) %>%
    step_ns(energy, deg_free = 7)

# GAM workflow
spline_wf <- workflow() %>%
    add_model(lm_spec) %>%
    add_recipe(spline_rec)

# Evaluation metrics with cross-validation
fit_resamples(
    spline_wf,
    resamples = music_cv, # cv folds
    metrics = metric_set(mae,rmse,rsq)
) %>% collect_metrics()

# For convenience, here's the metrics from cross-validation on the OLS model
ols_cv %>% collect_metrics(summarize=TRUE)
```

```
- Compare insights from variable importance analyses here and the corresponding results
from the Investigation 1. Now after having accounted for nonlinearity, have the most rel
evant predictors changed?
```

For our OLS model, the significant values were ranked loudness, danceability, valence, acousticness, instrumentalness, and energy. We don't know how they are ranked for our spline model, because they're all listed as significant and have significantly low p-values.

```
- Do you gain any insights from the GAM output plots (easily obtained from fitting smoot
hing splines) for each predictor?
```

We can see that there are predictors that are more nonlinear than other terms and would benefit more from splines. We specifically noticed that `danceability` and `valence` looked very linear compared to other terms. When we fit a GAM model with those two as linear terms, our results of edfs, p-values and R-squared did not change much compared to the one where we have all predictors as splines.

```
- Compare model performance between your GAM models that the models that assuming linear
ity.
    - How does test performance of the GAMs compare to other models you explored?
```

After making our GAMs, we noticed that our MAE and RMSE lowered slightly - about 0.5 years, and our r-squared increase slightly - from 0.37 to 0.40. Overall, this means our predictive performance increases when using splines.

```
- Don't worry about KNN for now.
```

3. **Summarize investigations**
    - Decide on an overall best model based on your investigations so far. To do this, make clear your analysis goals. Predictive accuracy? Interpretability? A combination of both?

Based on our investigations so far, we decided the spline model is the best model we used so far since it has lower RMSE/MAE and higher r-squared. Our current analysis goal is to use the metrics gained from Spotify's machine learning algorithm in order to predict the release year of a song (metrics such as danceability, loudness, etc). As a result, we are aiming more for predictive accuracy, which the spline models gives us more of. The spline model remains interpretable but its interpretability is also harder than the OLS/LASSO models because it is a lot harder to see which variables are more significant than others when all the p-values are too small for R to give us exact values.

4. **Societal impact**
    - Are there any harms that may come from your analyses and/or how the data were collected?
    - What cautions do you want to keep in mind when communicating your work?

One harm we should consider is the possibility of the overgeneralization of music trends. One of our planned research questions is to see if we can predict the year a song was released based on its attributes like valence, danceability, etc. However, this essentially ties a specific year to a specific kind of genre and we do not want to perpetuate any stereotypes such as modern music being more about violence or being negative, etc. We just want to highlight some overall trends and not ignore the fact that at any year, music of many different genres is still made. Another caution we should keep in mind is that this dataset is limited to the Spotify library since these values are from the Spotify API. Our songs are also limited to English songs, so we only see trends for a specific

language. In addition, the values we are getting from the Spotify API are from AI, so the values have the possibility to be subjective. A caution from our analysis is that with the variables we used we see a trend in the data but the predictive power of our current model is low due to the R-squared value.