HW 3

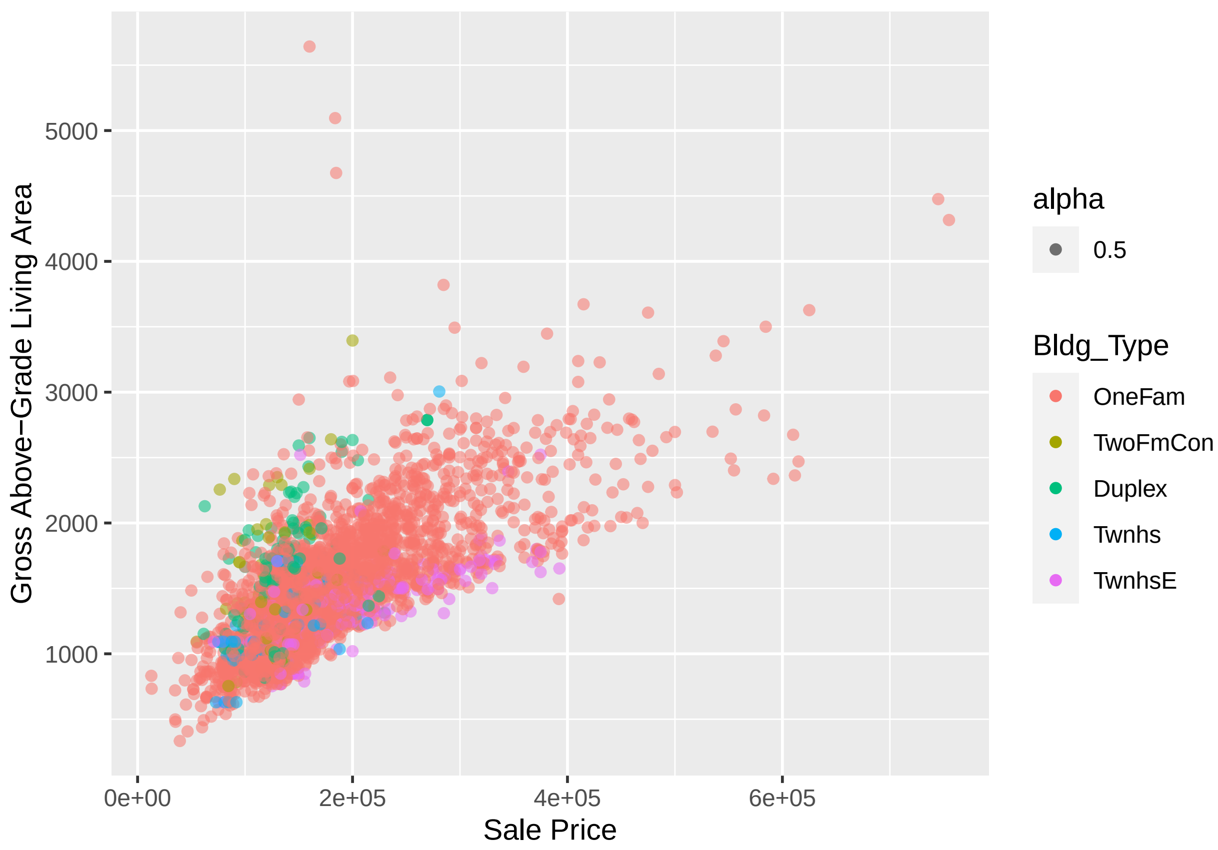
Marion Geary

Load Packages

```
library(tidyverse)
library(tidymodels)
library(modeldata)
data("ames")
```

Exploratory Data Analysis

```
ggplot(ames, aes(x=Sale_Price, y=Gr_Liv_Area)) +
  geom_jitter(aes(color=Bldg_Type, alpha=0.5)) +
  labs(x="Sale Price", y="Gross Above-Grade Living Area")
```



```r
ggplot(ames, aes(x=log(Sale_Price, base=10), y=log(Gr_Liv_Area, base=10))) +
  geom_jitter(aes(color=Bldg_Type, alpha=0.5)) +
  labs(x="Sale Price (log base 10)", y="Gross Above-Grade Living Area (log base 10)")
```

Through looking at the first graph, we see that there appears to be a relationship between the `Sale_Price` and the `Gr_Liv_Area`. In the second graph, we take the log of both variables and see that the relationship becomes even more linear, suggesting that `Gr_Liv_Area` could be a good predictor for `Sale_Price`. When we color by `Bldg_Type`, we see that different building types cluster around different `Sale_Price`, such as how `TwnhsE` clusters around higher sale prices while `TwoFmCon` clusters around lower sale prices.

Data Munging

```r
ames <- ames %>% mutate(log_sale_price = log(Sale_Price, base=10))
```

We alter the predictor, `Sale_Price` to prepare for modeling by taking the logarithm.

Data Spending

```r
set.seed(12345)
ames_split <- initial_split(ames, prop=0.8, strata=Sale_Price)
ames_train <- training(ames_split)
ames_test <- testing(ames_split)
```

Split the data into a test set and a training set. We stratify based on `Sale_Price` in order to keep the proportions similar in each set.

Feature Engineering

```
ames_recipe <-
  recipe(Sale_Price ~ Neighborhood + Gr_Liv_Area + Year_Built + Bldg_Type, data = ames_train) %>%
  step_log(Gr_Liv_Area, base = 10) %>%
  step_other(Neighborhood, threshold = 0.01) %>%
  step_dummy(all_nominal_predictors()) %>%
  step_interact( ~ Gr_Liv_Area:starts_with("Bldg_Type_") )
```

We create a recipe to prepare the data for modeling. We use `Neighborhood`, `Gr_Liv_Area`, `Bldg_Type`, and `Year_Built` to predict `Sale_Price`. We take the logarithm of `Gr_Liv_Area`, and turn our categorical variables into dummy variables. We also group the bottom 1% of neighborhoods into one dummy variable rather than have a multiple dummy variables with either one or no data points. This prevents these variables from creating problems for the model or skewing it. We also include an interaction step to reflect how the `Gr_Liv_Area` impacts the `Sale_Price` of each category of the `Bldg_Type` differently.

Create Model Workflow

```
lm_model <- linear_reg() %>% set_engine('lm')
lm_wflow <- workflow() %>%
  add_model(lm_model) %>%
  add_recipe(ames_recipe)
```

We combine our recipe with a linear regression engine to create a workflow.

Fit the Model

```
lm_fit <- fit(lm_wflow, ames_train)
```

We fit the workflow to our training data to get a model.

Test the Model

```
ames_test_res <-
  predict(lm_fit, new_data = ames_test %>%
  select(-Sale_Price))
ames_test_res <-
  bind_cols(ames_test_res, ames_test
    %>% select(Sale_Price))
ames_test_res
```

```
## # A tibble: 588 x 2
##      .pred Sale_Price
##      <dbl>      <int>
##  1 182334.     215000
##  2 192605.     189900
##  3 408878.     538000
##  4  78792.     141000
##  5 220612.     210000
##  6 146672.     149900
##  7 361537.     376162
##  8 322939.     306000
##  9 320809.     275000
## 10 179904.     180000
## # ... with 578 more rows
```

Now that we have a model, we test it on the test data. We use the model to predict `Sale_Price`. We create a table to compare the predicted outcome with the observed outcome.

Assess Goodness of Fit

```
ames_metrics <- metric_set(rmse, rsq)
ames_metrics(ames_test_res, truth = Sale_Price, estimate = .pred)
```

```
## # A tibble: 2 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard   40672.
## 2 rsq     standard       0.765
```

Using this result, we assess our model based on how well its predictions compared with the real outcomes. We create the function `metric_set()` to find the RMSE and R^2 for the model. We see that the R^2 value is 0.765, which shows that the model is a fairly accurate predictor of the `Sale_Price`. Our model is able to account for 76.5% of variation in `Sale_Price`. With a RMSE of 40672, we see that our model errs in prediction of `Sale_Price` by an average of $40,762. This is a fairly significant number based on the units for `Sale_Price` and indicates that our model could likely use some refining.