Documentation Article

Corbin Brinkerhoff, Matthew Wilson

Statistical Model

Mathematical Model

$$\text{Salary}_i = \beta_0 + \beta_1 \,\text{Sex}_i + \beta_j \,\text{Major}_{ij} + \beta_3 \,\text{GPA}_i + \varepsilon_i, \qquad \varepsilon_i \sim \mathcal{N}(0, \sigma^2).$$

Here the $Major_{ij}$ indicates that the ith quantity for major is the jth major. And β_j is the beta coefficient for the jth major

Prediction Justification

We performed k-fold CV and used the metric of RMSE. We can see that our model does decently well in predicting the variance in salary compared to the variance of the salaries in the data set. We can also see that the results are rather consistent across folds. Model Description

Sect 2

Sect 3

We created a linear model using major to predict salary. We then looked at the F-statistic for the model and saw that major was significant. We then looked at the beta coefficients of the different majors to determine which majors had the largest positive coefficients as that indicated which majors had the highest salaries on average.

Sect 4

Sect 5

We compared models with and without interactions. The F-test showed the model with interactions was significant. But their adjusted R-squared values are similar, and the K-fold CV results show that the predictive RMSE is similar. So we consider the simpler model to be a better choice. We also compared our predictive rmse to the sd of the dataset and saw that our rmse was significantly smaller than the sd, which means it better explained the variance in the data.

Code Appendix

```
#library for data cleaning and graphs
library(tidyverse)

# reading in data and creating factors
sal_df <- vroom::vroom("Salary.csv")

sal_df <- sal_df |>
  mutate(
    MajorCategory = factor(MajorCategory),
    Sex = factor(Sex)
  )

# creating a linear model. Salary predicted my Major, GPA and sex. No interaction
sal_lm<-lm(Salary~ .,sal_df)
#linear model that includes an interaction between GPA and MajorCategory
sal_lm_int_GPA_MAJ  <- lm(Salary ~ GPA * MajorCategory + Sex, data = sal_df)

head(sal_df)

# 1) Pull GPA beta (estimate) and p-value from the summary object
smry <- summary(sal_lm)
coef_mat <- smry$coefficients

gpa_estimate <- coef_mat["GPA", "Estimate"]
gpa_pvalue   <- coef_mat["GPA", "Pr(>|t|)"]

# 2) Confidence interval (default 95%)
gpa_ci <- confint(sal_lm, "GPA", level = 0.95)

# Print nicely
```

```r
cat("GPA beta (Estimate):", round(gpa_estimate, 4), "\n")
cat("GPA p-value        :", signif(gpa_pvalue, 4), "\n")
cat("GPA 95% CI         : [",
    round(gpa_ci[1], 4), ", ", round(gpa_ci[2], 4), "]\n", sep = "")

# Recommended contrasts for Type III tests
options(contrasts = c("contr.sum", "contr.poly"))
library(car)
car::Anova(sal_lm_int_GPA_MAJ, type = 3)

# code to perform f-test between base lm, and an lm that doesn't include Major
print(anova(sal_lm, lm(Salary~ -MajorCategory,sal_df)))

# we see that major is significant,
# so now we view the summary of the model with major.
summary(sal_lm)

maj_df <- expand.grid(Sex = "F", GPA = 3.5,
                      MajorCategory = unique(sal_df$MajorCategory))

maj_preds <- predict(sal_lm, new_data = as.data.frame(maj_df))

# make df for diff majors holding constant of female w/ 3.5 GPA
maj_df <- expand.grid(Sex = "F", GPA = 3.5,
                      MajorCategory = unique(sal_df$MajorCategory))

# predict salaries for diff majors
maj_preds <- predict(sal_lm, newdata = maj_df)

# Ensure you used the correct arg for base R lm
maj_preds <- predict(sal_lm, newdata = maj_df, se.fit = TRUE)

# Bind predictions back to maj_df
plot_df <- maj_df %>%
  mutate(
    .pred = maj_preds$fit,
  ) %>%
  # Order majors by predicted value for a nicer chart
  mutate(MajorCategory = reorder(MajorCategory, .pred))

library(scales)
```

```r
ggplot(plot_df, aes(x = MajorCategory, y = .pred)) +
  geom_col(fill = "#4472C4") +
  coord_flip() +  # if you prefer horizontal bars (often easier to read)
  scale_y_continuous(
    breaks = scales::breaks_pretty(n = 5),
    minor_breaks = waiver(),
    labels = label_number(big.mark = ",")
  ) +
  theme_minimal(base_size = 12) +
  theme(
    panel.grid.minor.y = element_line(color = "grey85"),
    panel.grid.major.y = element_line(color = "grey80")
  ) +
  labs(x = "Major category",
       y = "Predicted Salary",
       title = "Predicted Salary by Major")

sal_lm_sex_major <- lm(
  Salary ~ GPA + Sex * MajorCategory,
  data = sal_df
)

summary(sal_lm_sex_major)

# linear model using all predictors and no interactions.
# As seen above, just defined again for clarity
sal_lm<-lm(Salary~ .,sal_df)

# linear model using all predictors and interactions found to be significant:
sal_interaction_lm <- lm(Salary ~ MajorCategory * Sex + MajorCategory * GPA,
                         data = sal_df)

# function to perform k-fold cv for a single fold.
# take input of k, the kth fold
# calculates rmse for the models with and without interactions
# returns difference of rmse of the 2
cross_validate <- function(k){
  val_set <- sal_df |> filter(folds == k)
  train_set <- sal_df |> filter(folds != k)

  preds_1 <- predict.lm(lm(Salary ~ MajorCategory * Sex + MajorCategory * GPA,
                           data = train_set),
```

```
            newdata = val_set)
  rmse_1 <- sqrt(mean((preds_1 - val_set[["Salary"]])^2))

  preds_2 <- predict.lm(lm(Salary ~ MajorCategory + Sex + GPA, data = train_set),
            newdata = val_set)
  rmse_2 <- sqrt(mean((preds_2 - val_set[["Salary"]])^2))

  rmse_1 - rmse_2
}

K <- 20

# randomly creates partitions for folds
folds <- rep(1:K, length = nrow(sal_df))|>
  sample()

rmse_results <- sapply(1:K, FUN = cross_validate)

# results of k-fold validation comparison
print(summary(rmse_results))
hist(rmse_results)

# function similar to above
#instead of finding difference, just calclates rmse for 1 and returns it
cross_validate_one <- function(k){
  val_set <- sal_df |> filter(folds == k)
  train_set <- sal_df |> filter(folds != k)

  preds <- predict.lm(lm(Salary ~ Sex + MajorCategory + GPA, data = train_set),
            newdata = val_set)
  sqrt(mean((preds - val_set[["Salary"]])^2))
}

K <- 20

# see above cell
folds <- rep(1:K, length = nrow(sal_df))|>
  sample()

rmse_results <- sapply(1:K, FUN = cross_validate_one)

# statistics from k_folds
```

```r
print(sd(sal_df[["Salary"]]))
print(mean(rmse_results))
print(summary(rmse_results))

set.seed(545)

# randomly selecting a person from the df
pick <- sample(1:length(sal_df$Salary),1)

# gets explanatory variables for that person
info <- as.data.frame(sal_df)[pick,2:4]

#makes prediction for salary
prediction <- predict(sal_lm, info)

#prints explanatory variables, and also difference between prediciton and actual salary.
print(info)
print(prediction - sal_df$Salary[pick])

# creates a plot of various lm assumptions
par(mfrow = c(2,2))
plot(sal_lm)
```