NBA Hackathon, Business Analytics Question - Tuned Random Forest Model for Predicting Instagram Engagments

Introduction

Presented with the task of predicting the number of engagements with Instagram posts of the *@nba* account from the past two years, our team elected to create a partitioned Random Forest Model from the *randomForest* package in *R*. Our models were partitioned based on post medium (The prompt seemed to allude to this as well, so we explored accordingly and confirmed the suggestion). Here we present our process for cleaning and modifying the data, visualizing trends, training the Random Forest, evaluating and tuning it, and finally applying it to predict *Engagements* in our holdout set.

First, we defined functions to calculate predicted values, absolute percent error of each prediction, and mean absolute percent error as follows (this will be useful for training later):

```
## Create useful functions for calculating mape
Calc_Mape <- function(labels, dataset, model)
  {
    actual = labels
    predicted = predict(model, dataset)
    APE = abs((actual - predicted)/actual) * 100
    mape = mean(APE)
    return(mape)
  }
Calc_Predicted <- function(model, dataset)
  {
    predicted = predict(model, dataset)
    return(predicted)
  }
Calc_PE <- function(labels, dataset, model)
  {
    actual = labels
    predicted = predict(model, dataset)
    PE = ((actual - predicted)/actual) * 100
    return (PE)
  }
```

Next we loaded in the necessary packages and read in the training set:

Predictors Preparation and Data Parsing

After sorting and examining the dataset across a number of predictors, we noticed that some of the posts did not contain descriptions and others only contained an emoji or symbol. To avoid any unnecessary influence from these observations, we deleted these entries entirely (while we recognize that imputing the mean for the predictors of these observations is likely a better approach, we felt that this wouldn't entirely be accurate given that they were text descriptions).

In addition, we created variables for the number of instagram accounts mentioned (num_ats) and the number of hashtags (num_hash) in each post. We also converted some of our categorical predictors into numerical predictors (useful for creating a Correlation Plot later).

Note: In an attached Python file, you will find some code with our initial modifications to the dataset, including using datetime functions, parsing out important information from the *Created* column and Instagram descriptions.

```
## Deleting missing data

# Order data by post description
df <- df[order(df$Description),]

# Delete descriptions with NAs, just symbols, or just emojis
# and remove season_yr and is_finals column

nba <- df[-c(1:14,20:35),-c(15,17)]
```

By the end, both our training and holdout sets contained the following predictors:

X - Index variable

Followers - # of Followers at time of posting

Created - Timestamp of posting

Type - Type of Post; three factors: *Videos, Albums, Photos*

Description - Post caption

datetime - Datetime variable parsed from *Created*

year - Year it was posted

month - Month posted (parsed *datetime*)

month_int - Numeric variable for months (*1* = January). Useful for correlation plot

weekday - Day of week posted

day_time - Time of day posted

is_playoffs - Boolean: whether the post was made in the playoffs (True) or not (False)

is_season - Boolean: whether the post was made during the season/playoffs (True) or off-season (False)

Followers Mentioned - Number of followers of nba team accounts mentioned in post *Description*

all_nba - Three factor variable: *0* if no all-nba or all-star players are mentioned in *Description*. *1* if an all-nba is mentioned in the *Description*, not including those to follow. *2* if Lebron James, Stephen Curry, Kyrie Irving, or Kobe Bryant (NBA players with highest number of instagram followers) are mentioned. Although Kobe is retired, posts with his mention appeared to garner more engagements.

league_type - *1* if *Description* mentioned the WNBA or G-League, *1* otherwise.

same_day_post - *1* if post was created within the same *day_time* on the same day. *0* otherwise. Parsed from *Created*.

num_ats - Number of hashtags in the *Description*.

num_hash - Number of @ symbols in the *Description*.

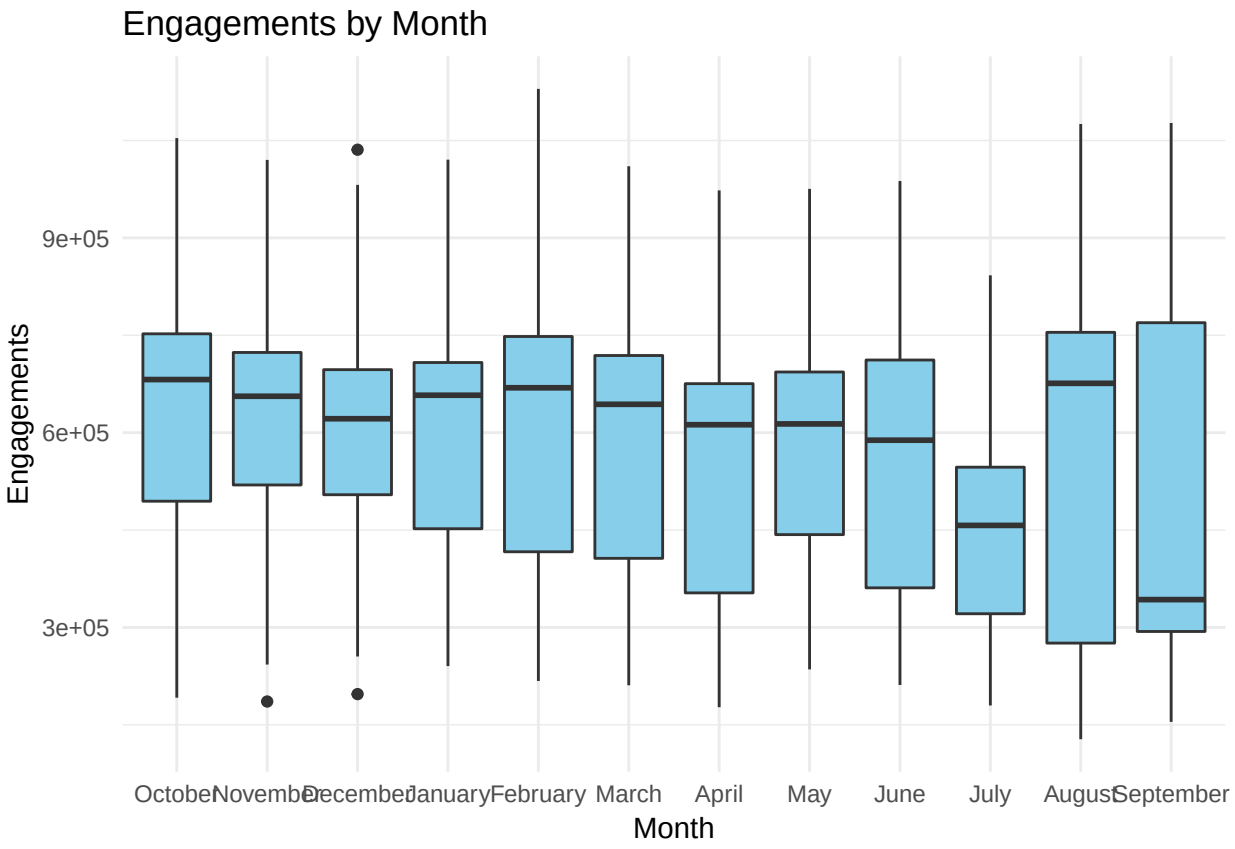
Type_num - Numerical variable for *Type* (*1* = Video, *2* = Album, *3* = Photo). Useful for correlation plot.

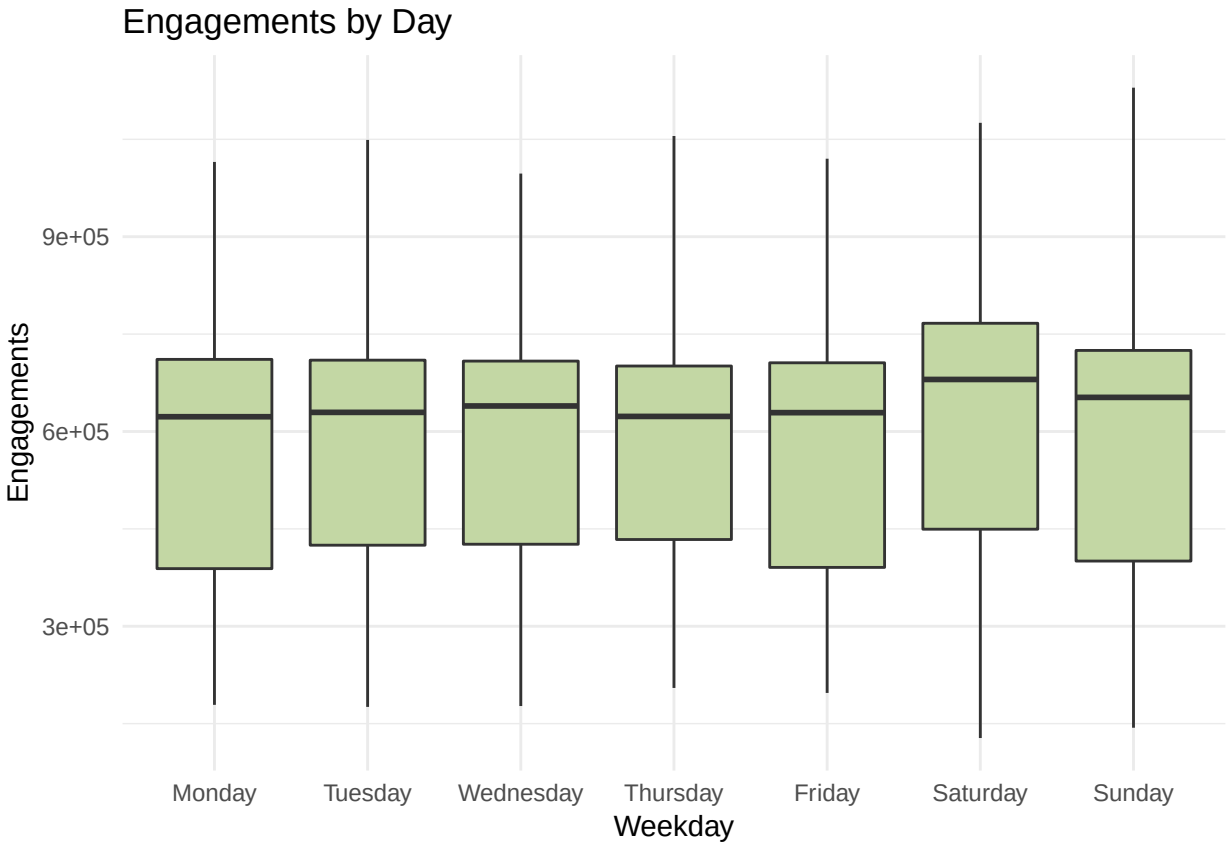
wkdy_num - Only in training. Numerical variable for *weekday* (*1* = Monday, *7* = Sunday)

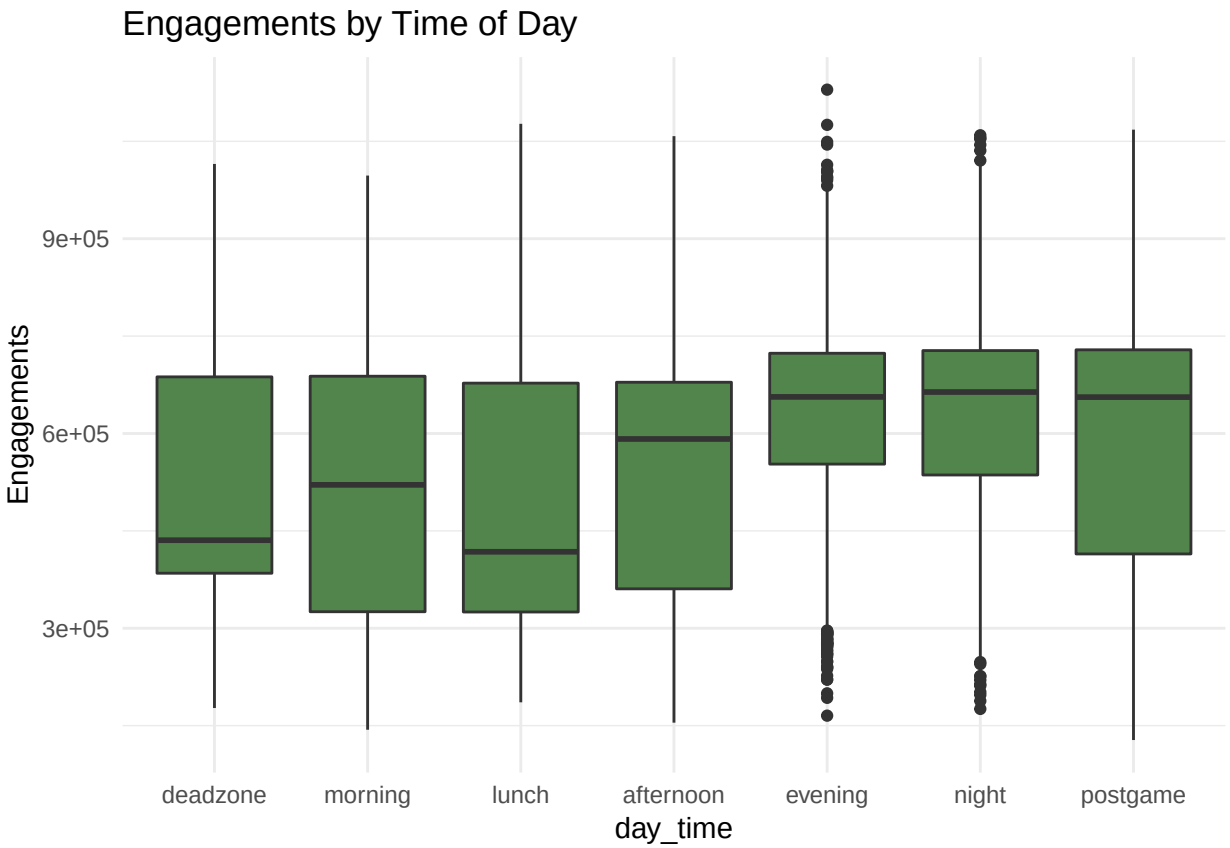
day_time_num - Only in training. Numerical variable for *day_time* (*1* = deazone, *7* = postgame)

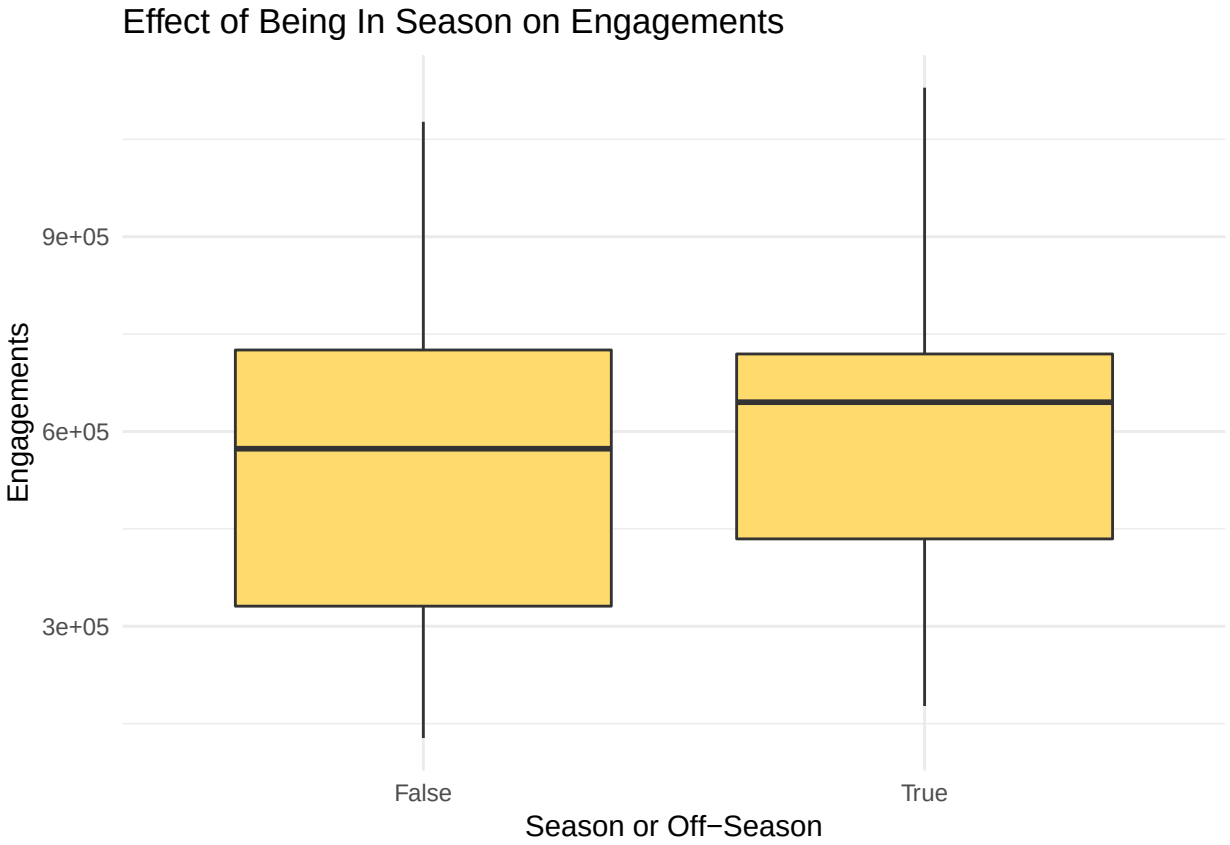
Data Visualizations

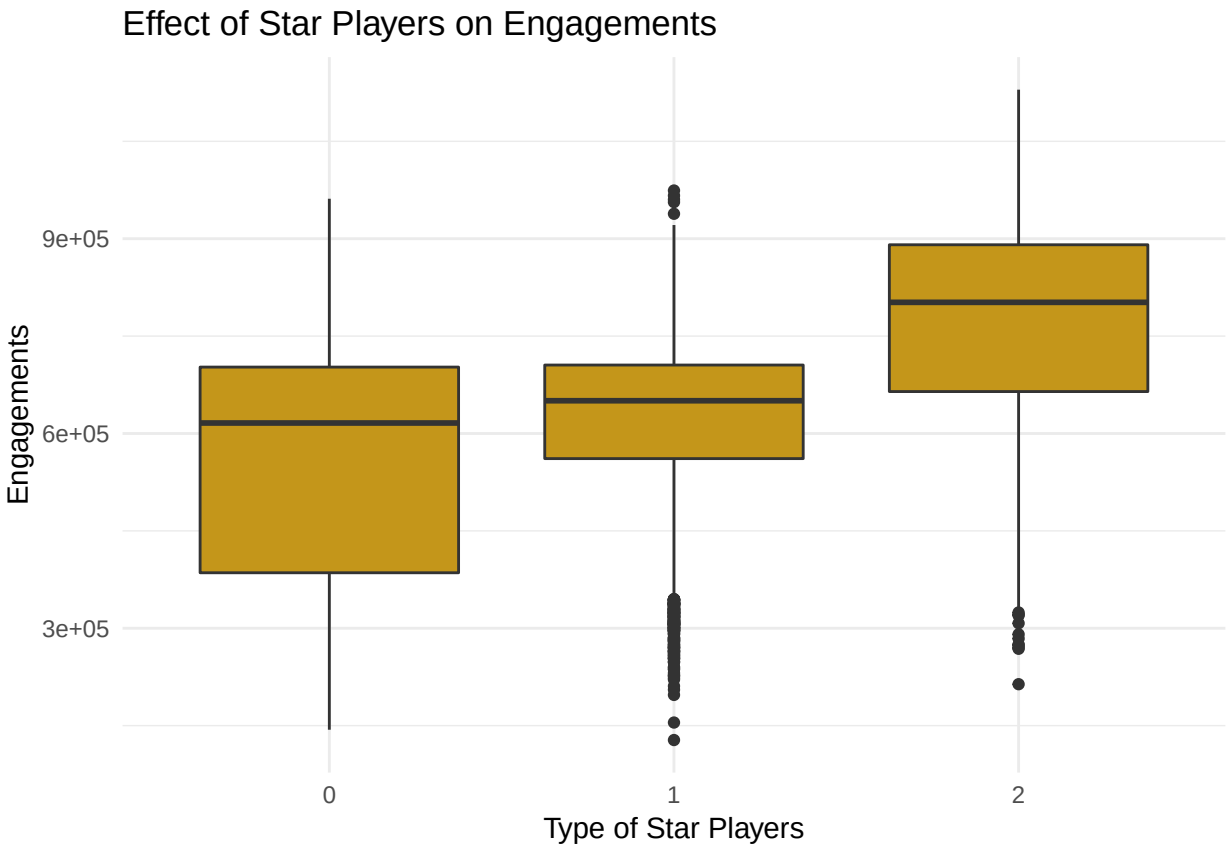
Here we provide some visualizations of our training data to understand the relationships between some of our predictors and Engagements. First some boxplots of predictors effects on *Engagements*.

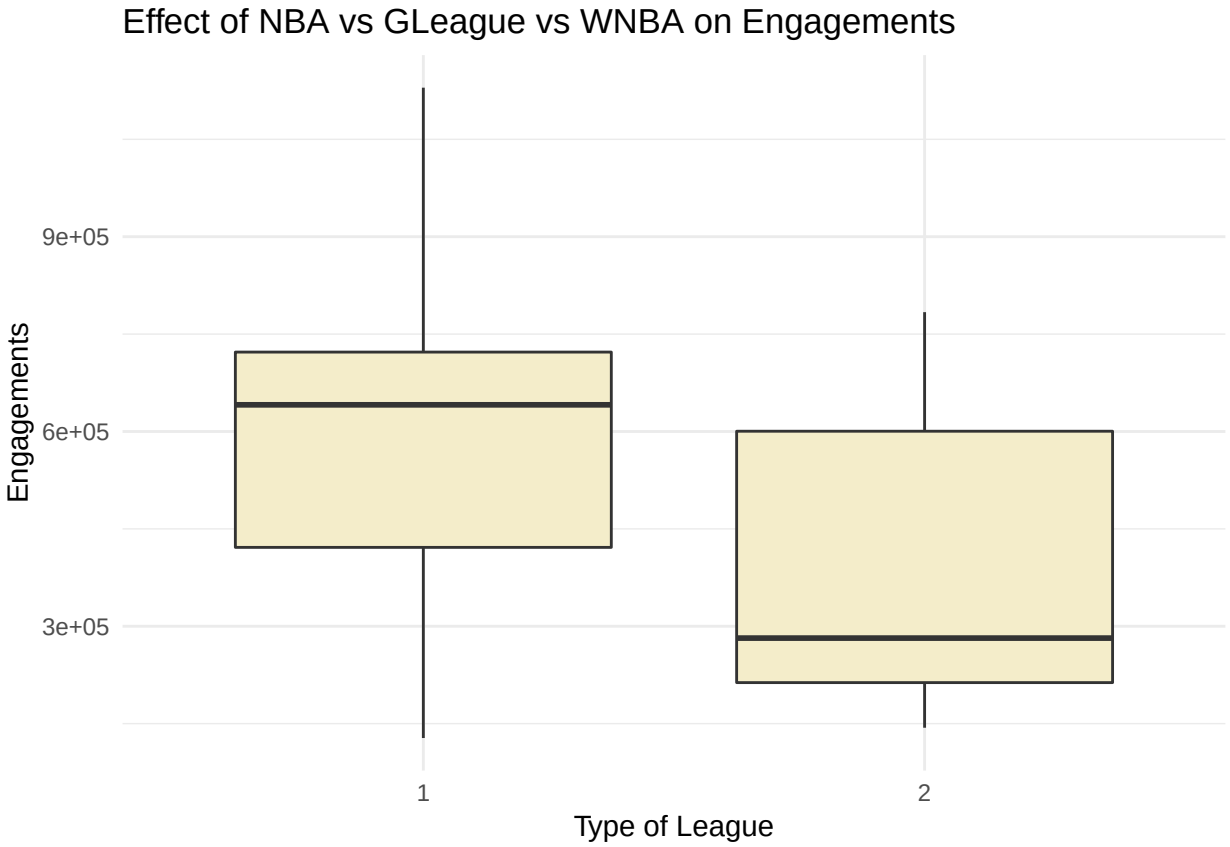


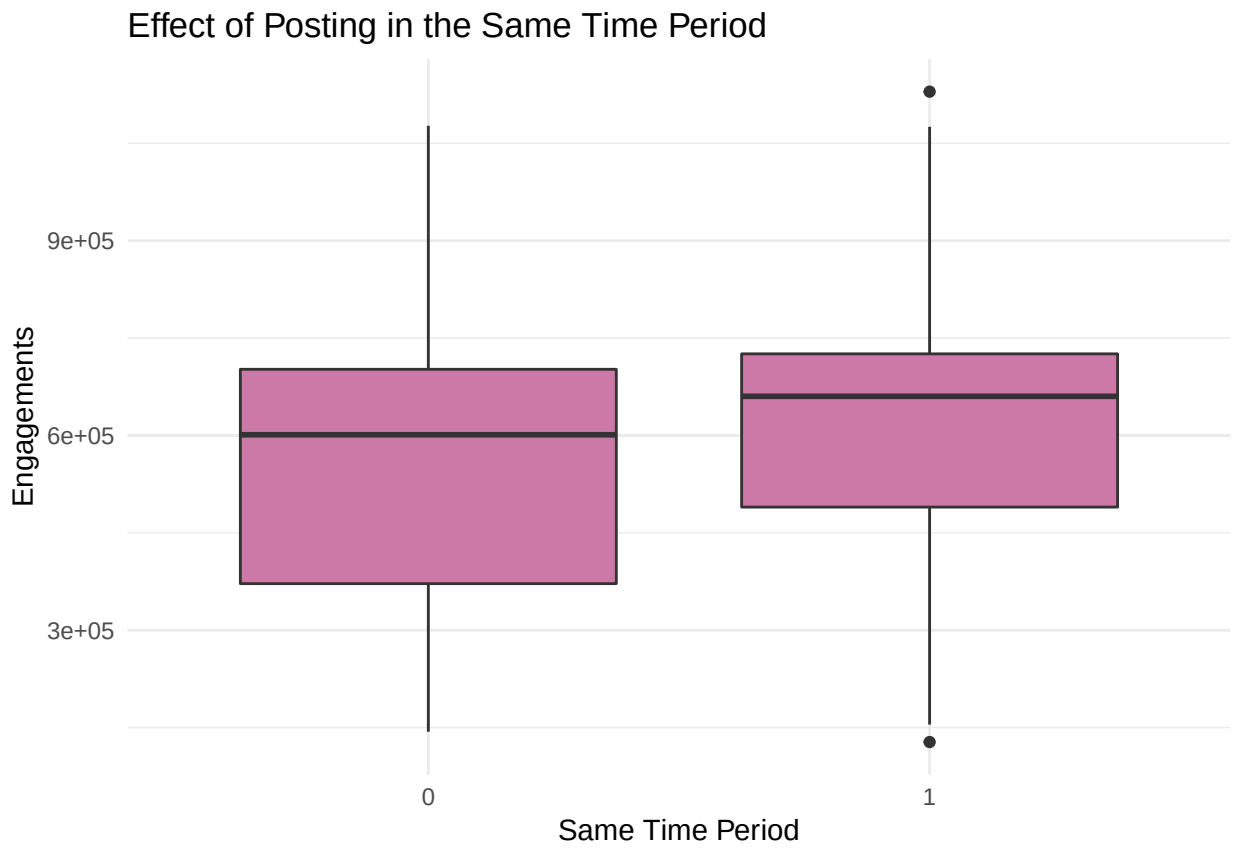

Engagements by Day
Engagements
9e+05
6e+05
3e+05
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
Weekday


Engagements by Time of Day
Engagements
9e+05
6e+05
3e+05
deadzone
morning
lunch
afternoon
evening
night
postgame
day_time

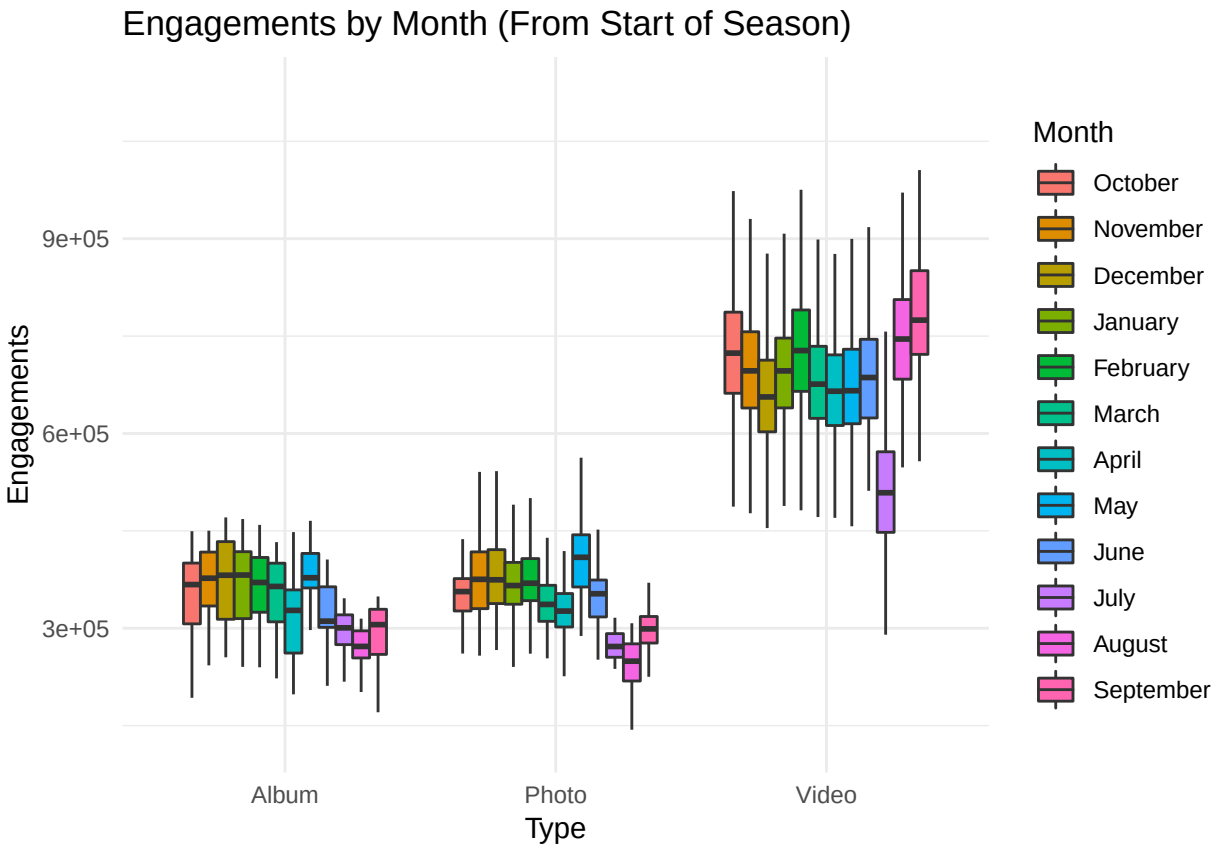

Effect of Being In Season on Engagements
Engagements
9e+05
6e+05
3e+05
False
True
Season or Off−Season

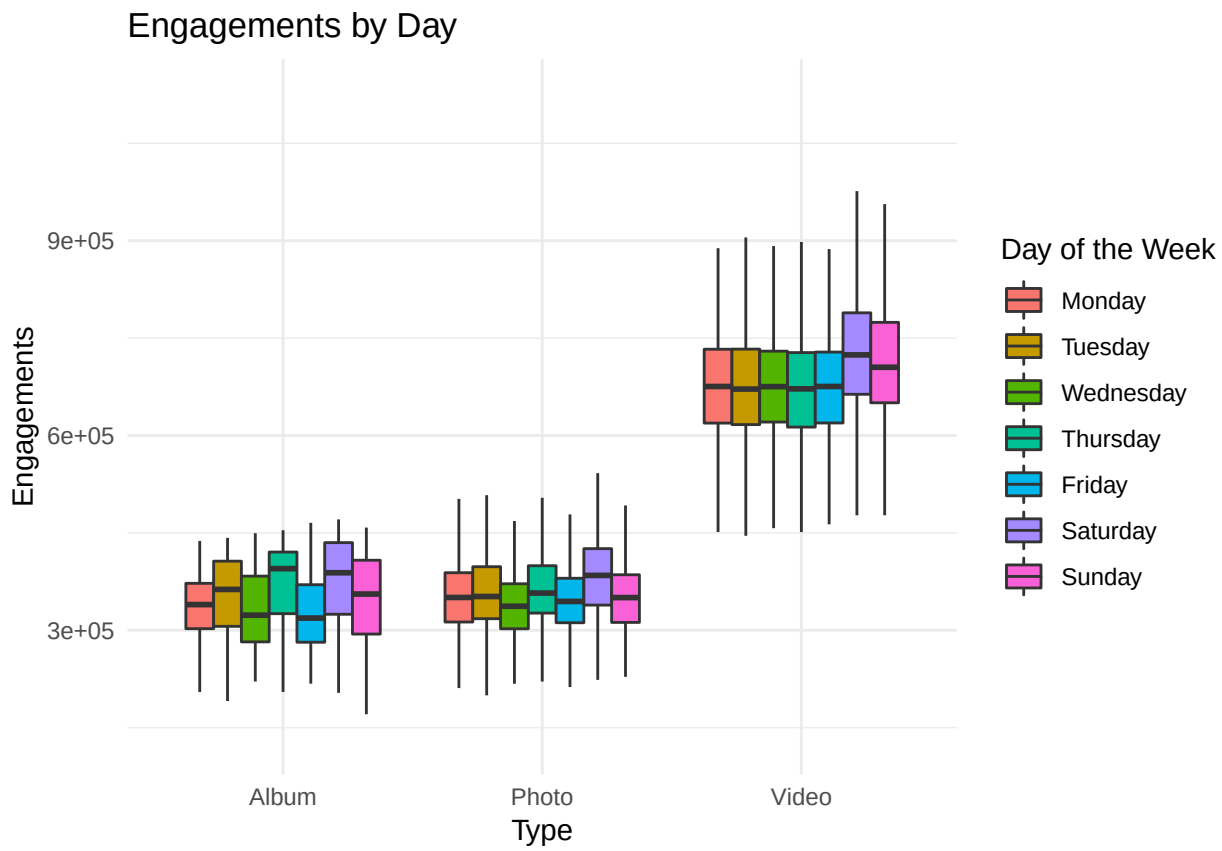

Effect of Star Players on Engagements
Engagements
9e+05
6e+05
3e+05
0
1
2
Type of Star Players

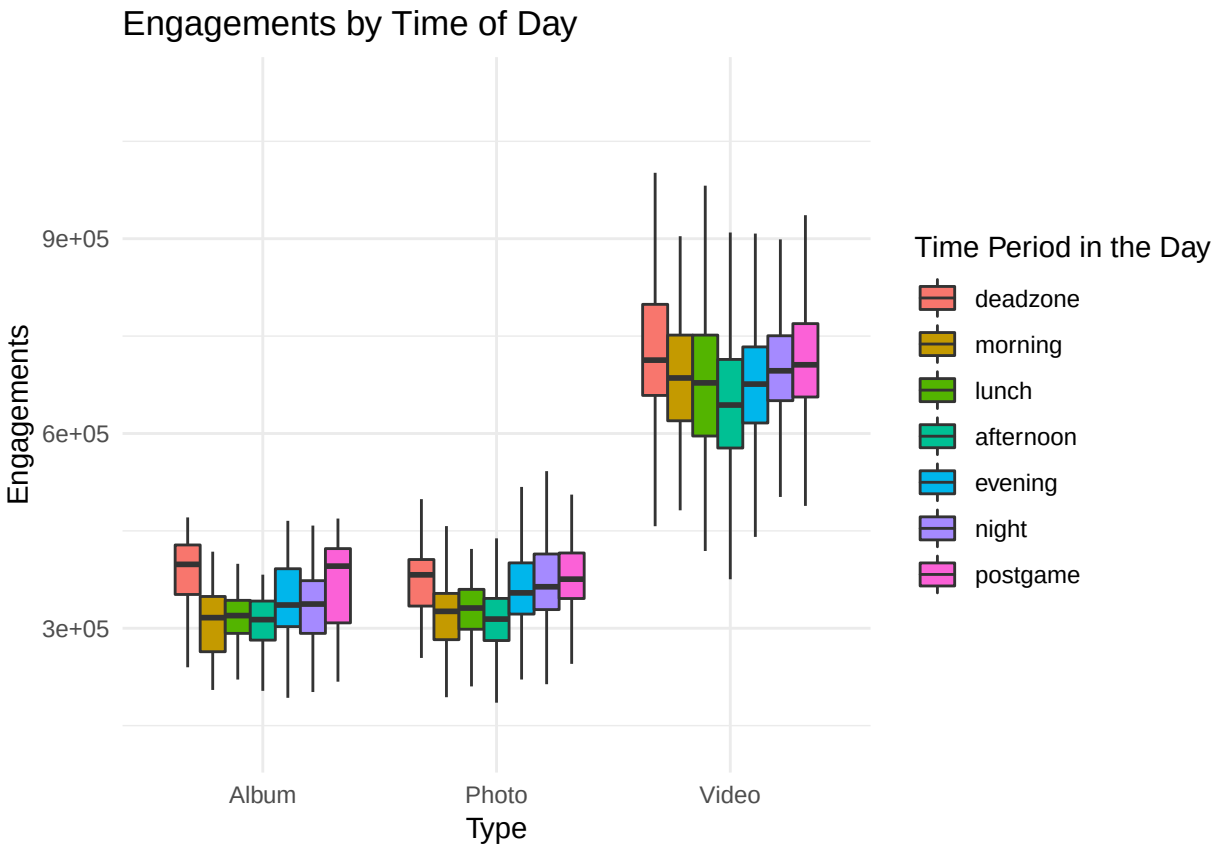

Effect of NBA vs GLeague vs WNBA on Engagements
Engagements
9e+05
6e+05
3e+05
1
2
Type of League

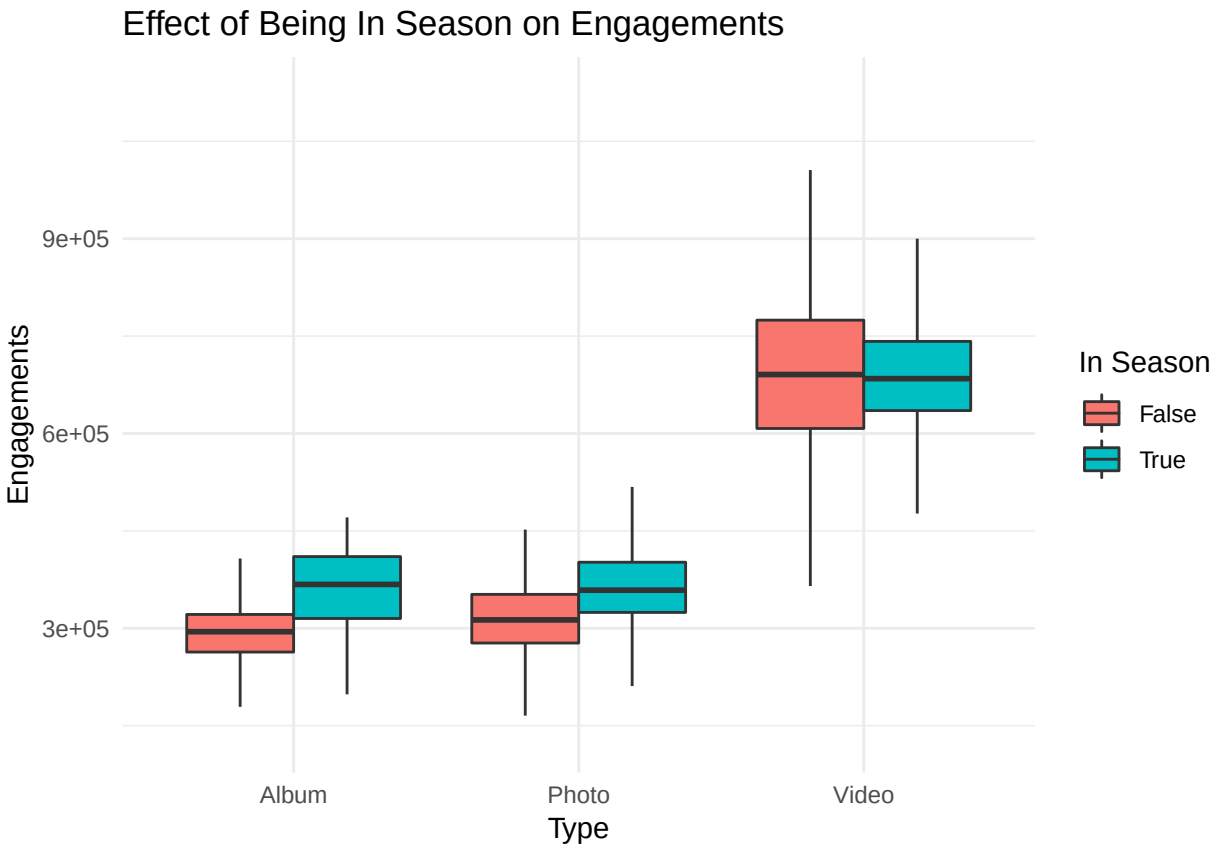


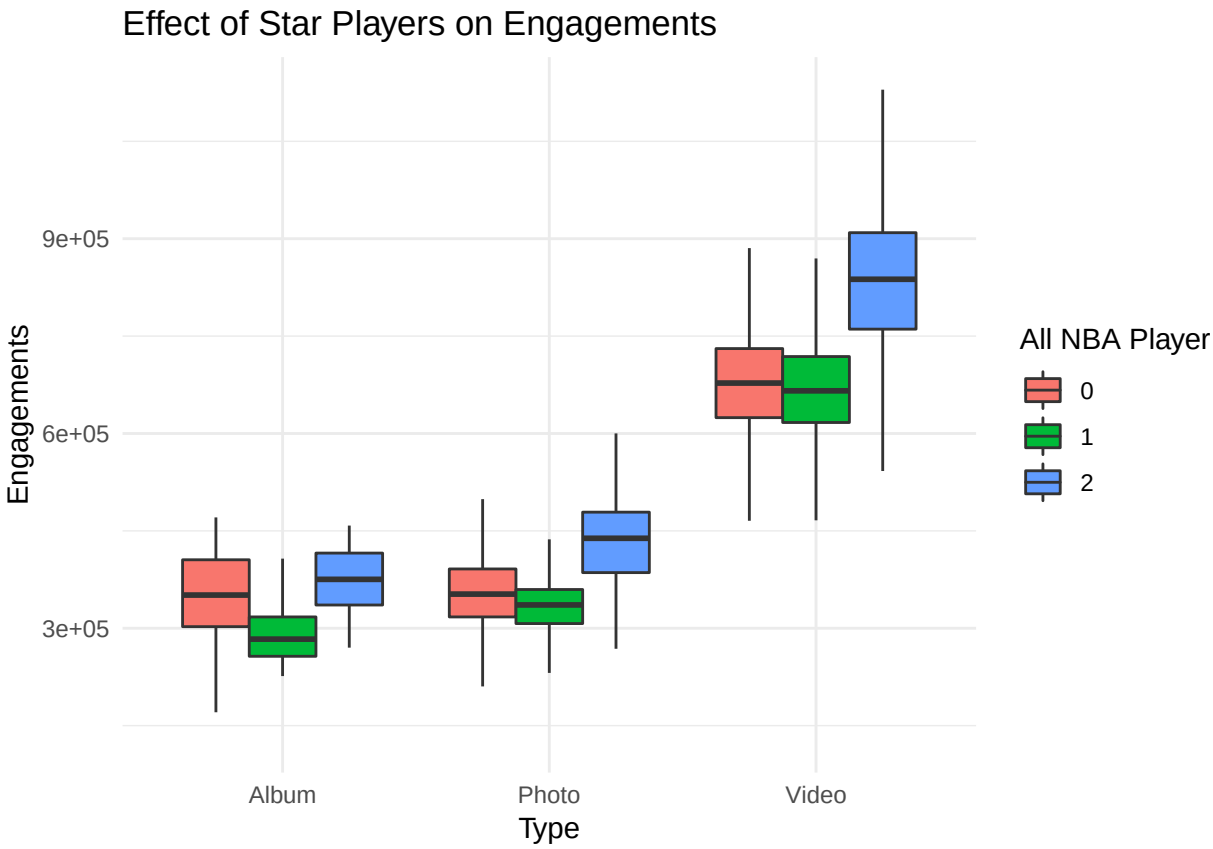
Now let's see how our data looks across post types.

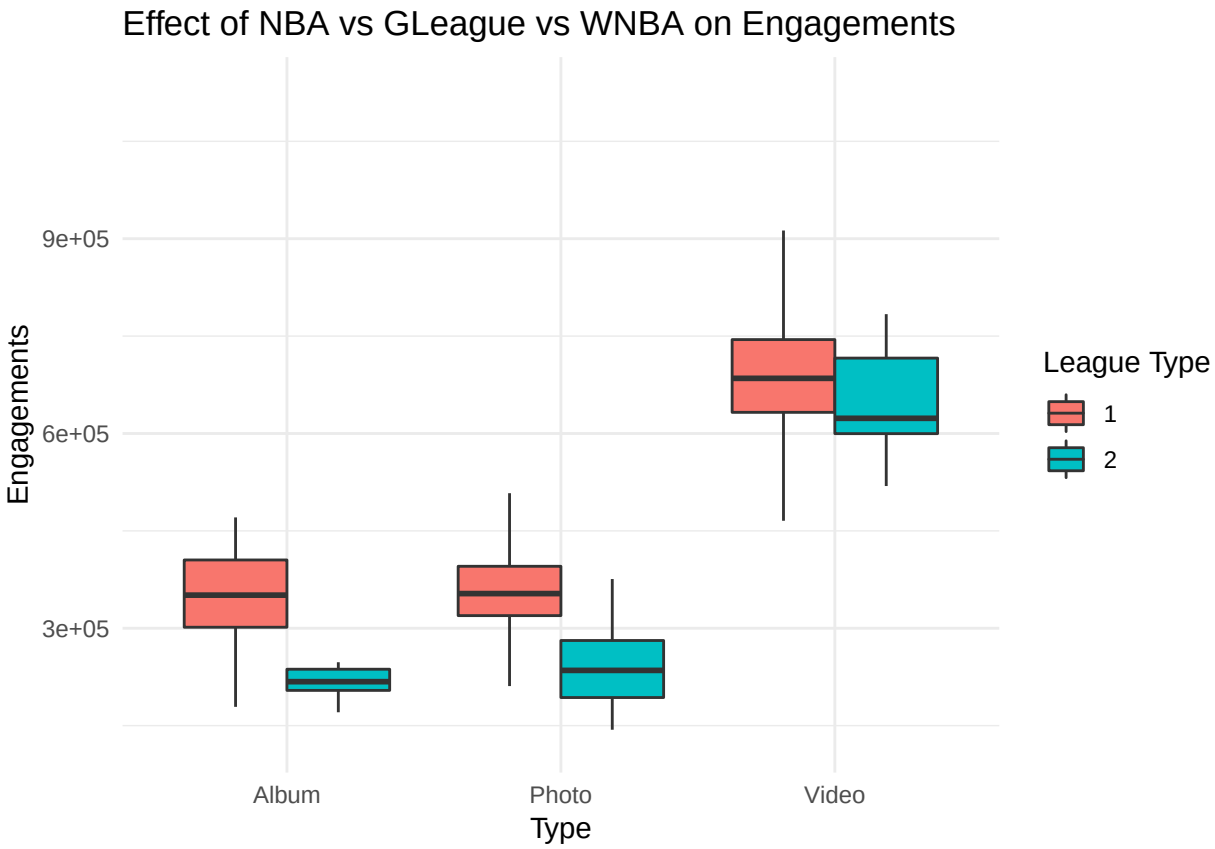

Engagements by Month (From Start of Season)
Engagements
9e+05
6e+05
3e+05
Album
Photo
Video
Type
Month
October
November
December
January
February
March
April
May
June
July
August
September

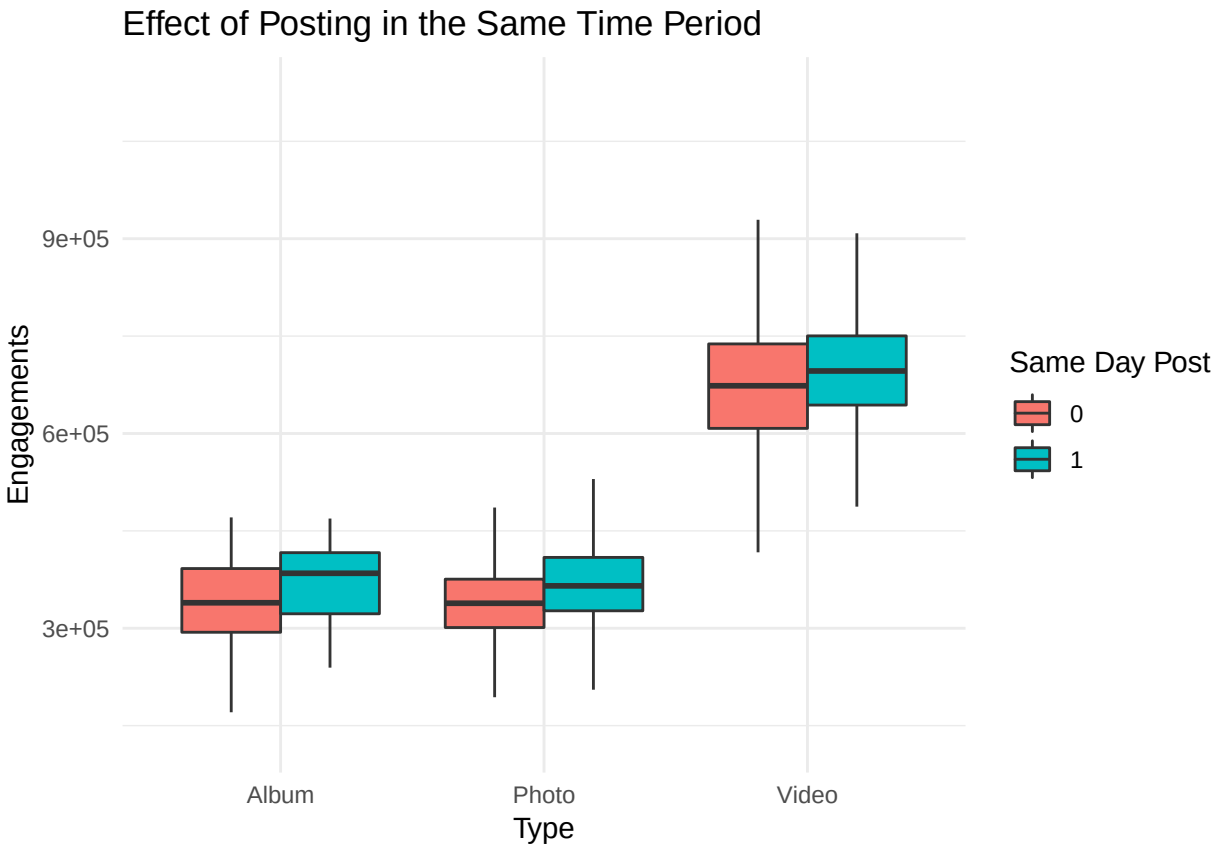

Engagements by Day
Engagements
9e+05
6e+05
3e+05
Album
Photo
Video
Type
Day of the Week
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday


Engagements by Time of Day
Engagements
9e+05
6e+05
3e+05
Album
Photo
Video
Type
Time Period in the Day
deadzone
morning
lunch
afternoon
evening
night
postgame


Effect of Being In Season on Engagements
Engagements
9e+05
6e+05
3e+05
Album
Photo
Video
Type
In Season
False
True


Effect of Star Players on Engagements
Engagements
9e+05
6e+05
3e+05
Album
Photo
Video
Type
All NBA Player
0
1
2


Effect of NBA vs GLeague vs WNBA on Engagements
Engagements
9e+05
6e+05
3e+05
Album
Photo
Video
Type
League Type
1
2



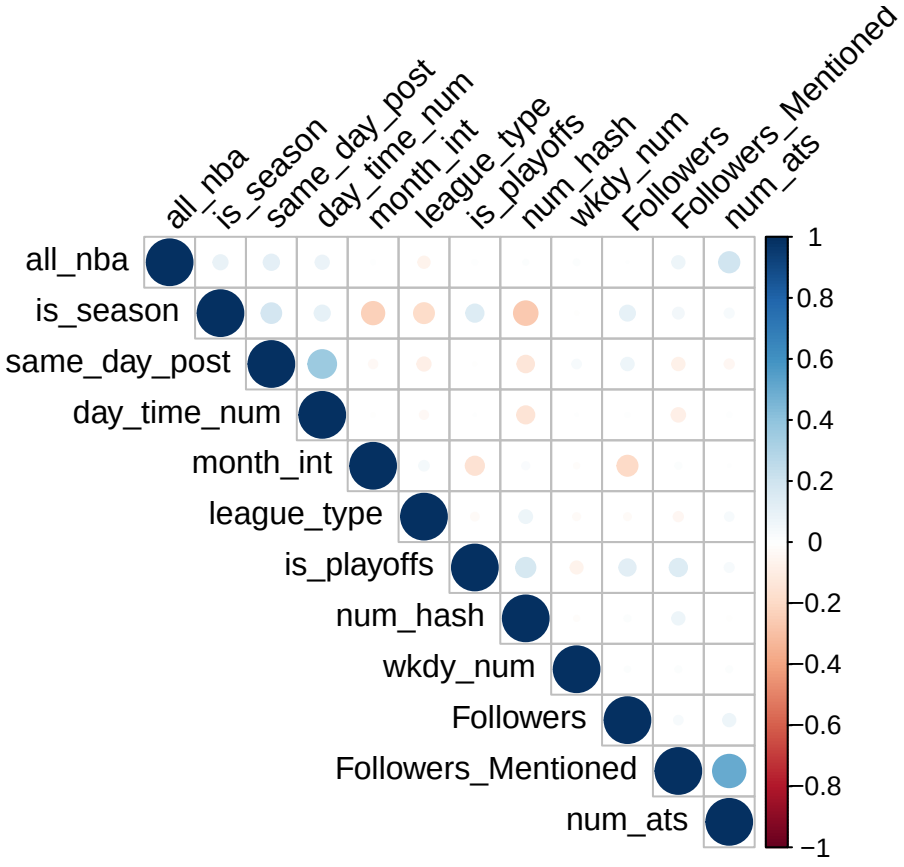
From our visualizations, one notable result sticks out: The effect of mentioning a superstar player (LBJ, Curry, Kobe, or Kyrie), is quite significant. Other trends seem to be relatively straightforward with some minor exceptions.

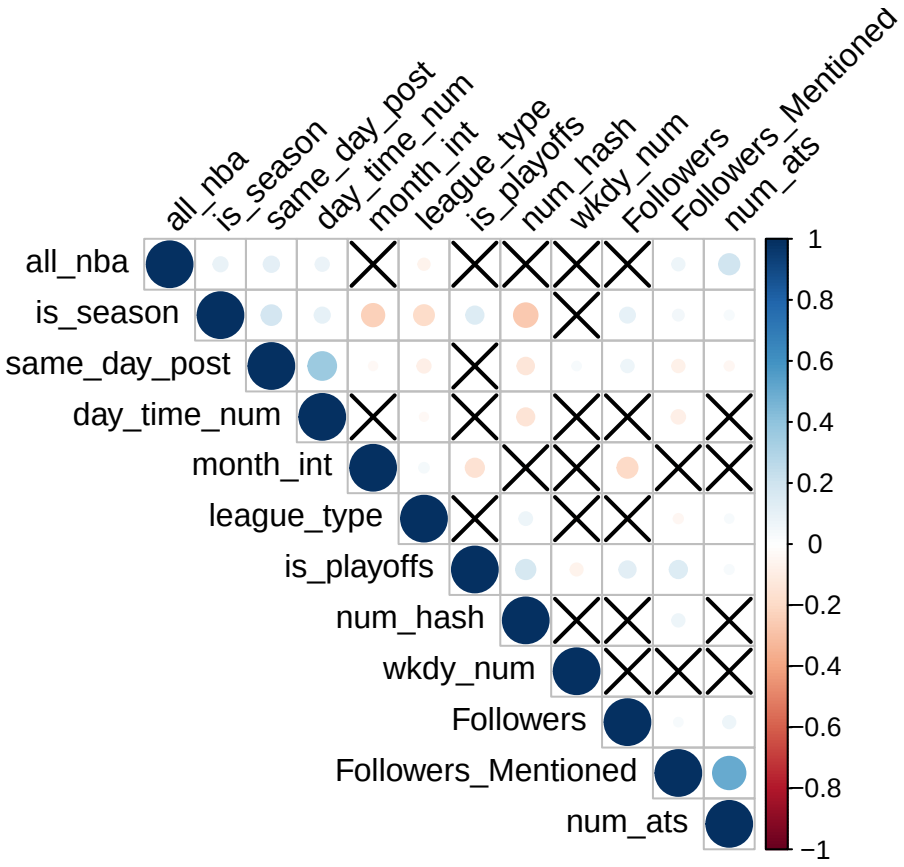
Multicollinearity

In order to assess multicollinearity in our predictors, we created a plot to visualize correlations between each variable The first plot displays the pearson correlation coefficient (R) between predictors, shown by the size and color of the circles at each point in the matrix. The second plot shows the same matrix, but this time with insignificant predictors (p-value < 0.01) crossed-out. With the remaining predictors, nothing in particular seems striking, except for two points:

1) The correlation between *Followers_Mentioned* and *num_@*. Intuitively this makes sense, a post must mention another team's IG account in order for the *Followers_Mentioned* to be a non-zero number. It may be worth looking into removing this variable after some cross validation.

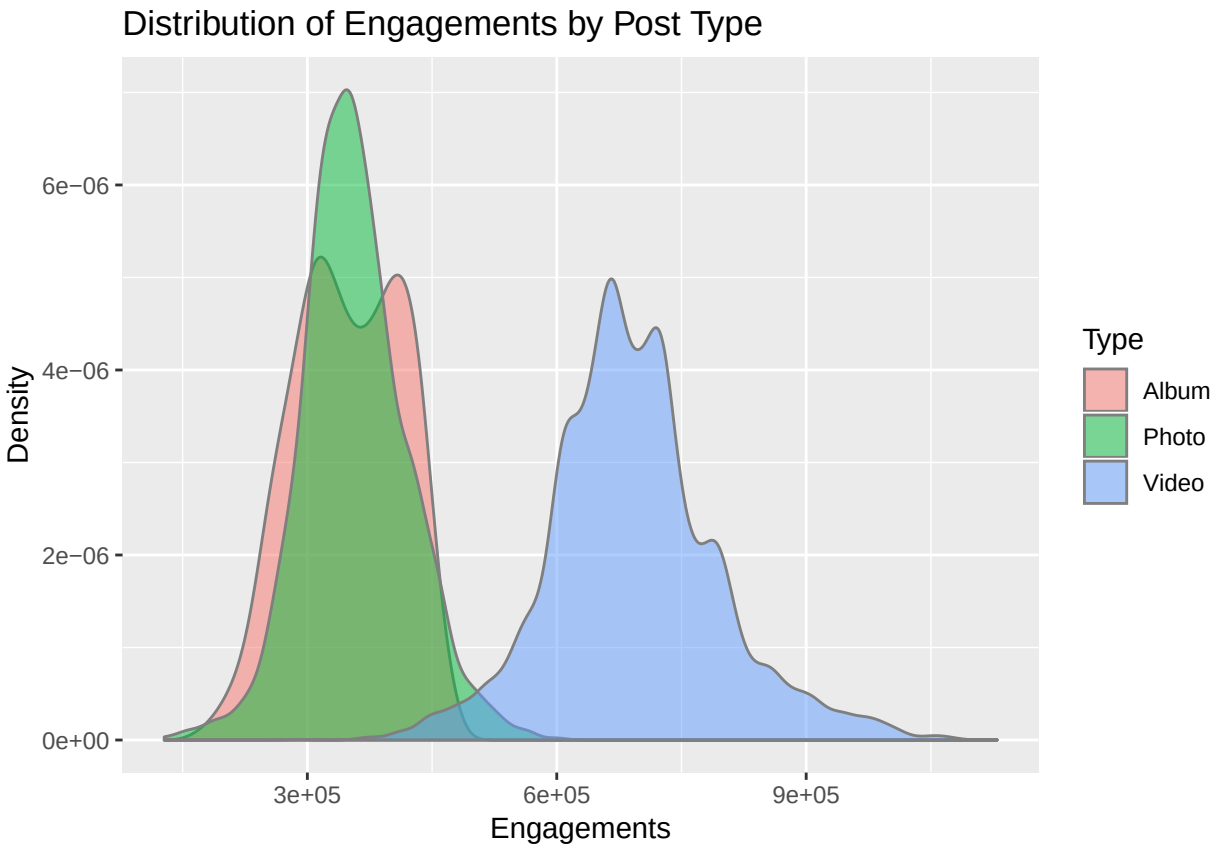
2) The correlation between *day_time_num* and *same_day_post*. Intuitively this makes sense, a post must be, by our definition, during the same time period as another for *same_day_post* to return a 1. It may be worth looking into removing one of these variables after some cross validation.


all_nba
is_season
same_day_post
day_time_num
month_int
league_type
is_playoffs
num_hash
wkdy_num
Followers
Followers_Mentioned
num_ats
1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1



Lastly, as hinted by the prompt, we viewed the distributions of our training data by post type (either Video, Album, or Photo) to determine whether they differed significantly.

While the distribution of Video and Photo distribution seems to be fairly normal, Album appears bimodal, and all three fall under different mean values for Engagements. Based on the plots, it does seem like each distribution is different enough to warrant each having their own model. In addition, the data set contains many more videos than it does photos or albums, so if we chose to simply include post type as a predictor in one model, we'd have to weight each level accordingly. Instead we chose to create three seperate models.



Model Training and Validation

First we scramble our training set in preparation for building our model.

```
## Subset data for just videos and scramble data
set.seed(92123)
nba <- nba[sample(1:nrow(nba)), ]
```

Although the code is not included here, we tried to create a number of different models, inlcuding a least squares regression model, a gradient boosted model, a model using NLP that vectorized the descriptions column, and even using predicted values from these models into other models. Ultimately, after testing the accuracy of each, we settled on a Random Forest, as it appeared to perform best and was the least overfitted.

In order to train and validate our Random Forest model, we first divide our training set by post type, and then within each division, we subset a small testing data set from our full training set and use it to calculate the MAPE.

More extensive cross-validation methods were used as well, but for brevity's sake, we don't include them here. Our cross-validation was useful in selecting the correct *mtry*, as well as eliminating two predictors that appeared to aid the model little: *same_day_post*, and *is_playoffs*. After eliminating these variables and tuning, our model's improved by over 1%.

```
# Deleting unneeded variables
nbanew <- nba[,c(2,3,4,5,6,9,11,12,15,16,17,18,20,21)]

# Creating Video/Photo/Album subsets from nba data
```

```r
nba_Video = nbanew[nbanew$Type == "Video",]
nba_Photo = nbanew[nbanew$Type == "Photo",]
nba_Album = nbanew[nbanew$Type == "Album",]

# Splitting Video/Album/Photo subsets into training and test sets
numberOfTrainingSamples_Video <- round(length(nba_Video$Engagements) * .80)
numberOfTrainingSamples_Photo <- round(length(nba_Photo$Engagements) * .80)
numberOfTrainingSamples_Album <- round(length(nba_Album$Engagements) * .80)

train_dataRF_Video <- nba_Video[1:numberOfTrainingSamples_Video,]
test_dataRF_Video <- nba_Video[-(1:numberOfTrainingSamples_Video),]

train_dataRF_Album <- nba_Album[1:numberOfTrainingSamples_Album,]
test_dataRF_Album <- nba_Album[-(1:numberOfTrainingSamples_Album),]

train_dataRF_Photo <- nba_Photo[1:numberOfTrainingSamples_Photo,]
test_dataRF_Photo <- nba_Photo[-(1:numberOfTrainingSamples_Photo),]
```

We then train the Random Forest model to predict *Engagements* on the test set and then calculate and append columns for predicted Engagements, percent error, and absolute percent error.

```
##
## Call:
##  randomForest(formula = Engagements ~ Followers + month + day_time +      weekday + Followers_Mentio
##                Type of random forest: regression
##                      Number of trees: 1000
## No. of variables tried at each split: 7
##
##           Mean of squared residuals: 2269818337
##                     % Var explained: 77.14
```

```
##
## Call:
##  randomForest(formula = Engagements ~ Followers + month + day_time +      weekday + Followers_Mentio
##                Type of random forest: regression
##                      Number of trees: 1000
## No. of variables tried at each split: 8
##
##           Mean of squared residuals: 850017793
##                     % Var explained: 78.54
```

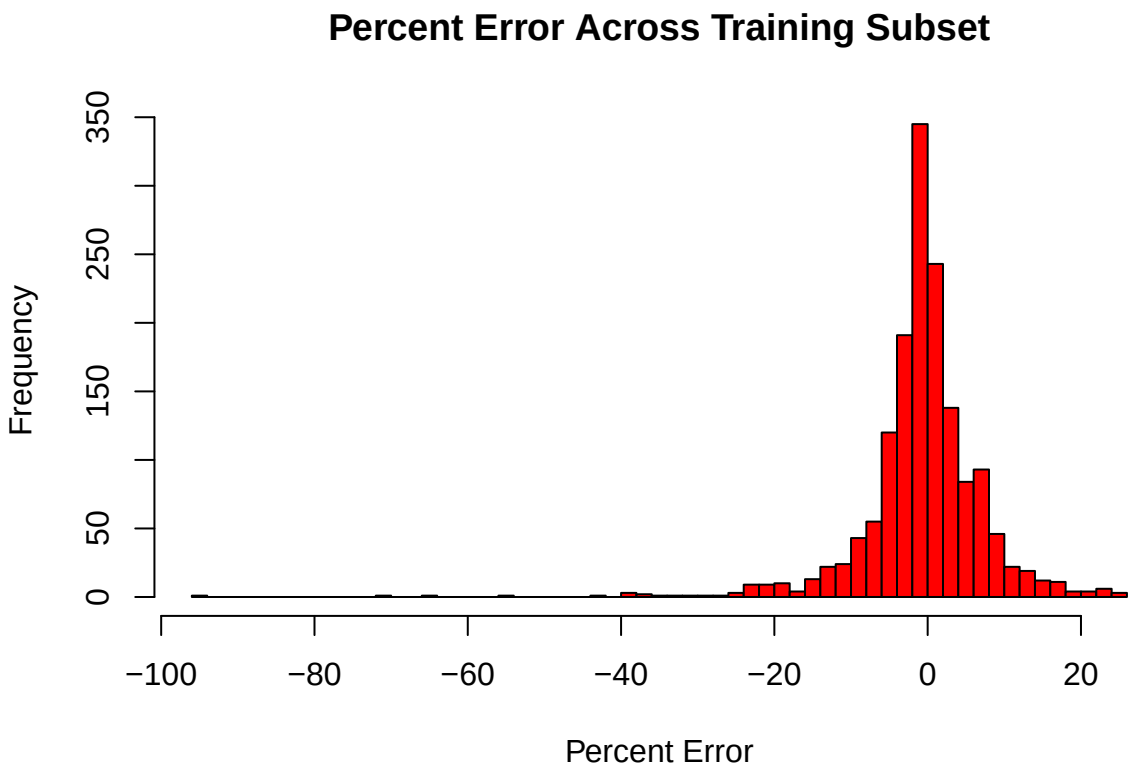
```
##
## Call:
##  randomForest(formula = Engagements ~ Followers + month + day_time +      weekday + Followers_Mentio
##                Type of random forest: regression
##                      Number of trees: 1000
## No. of variables tried at each split: 7
##
##           Mean of squared residuals: 727415628
##                     % Var explained: 81.61
```

```
mape_Rf
```

```
## [1] 4.93106
```

Our validation MAPE came out to about 5%. Not bad!

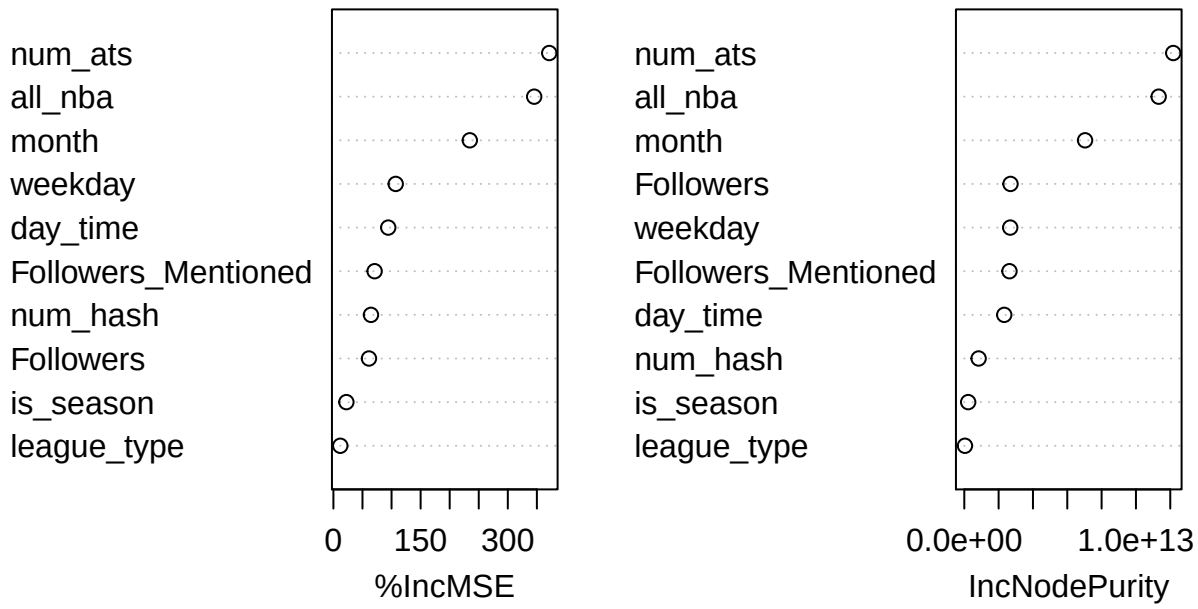
Let's take a look at how our percent error is distributed across the testing set.



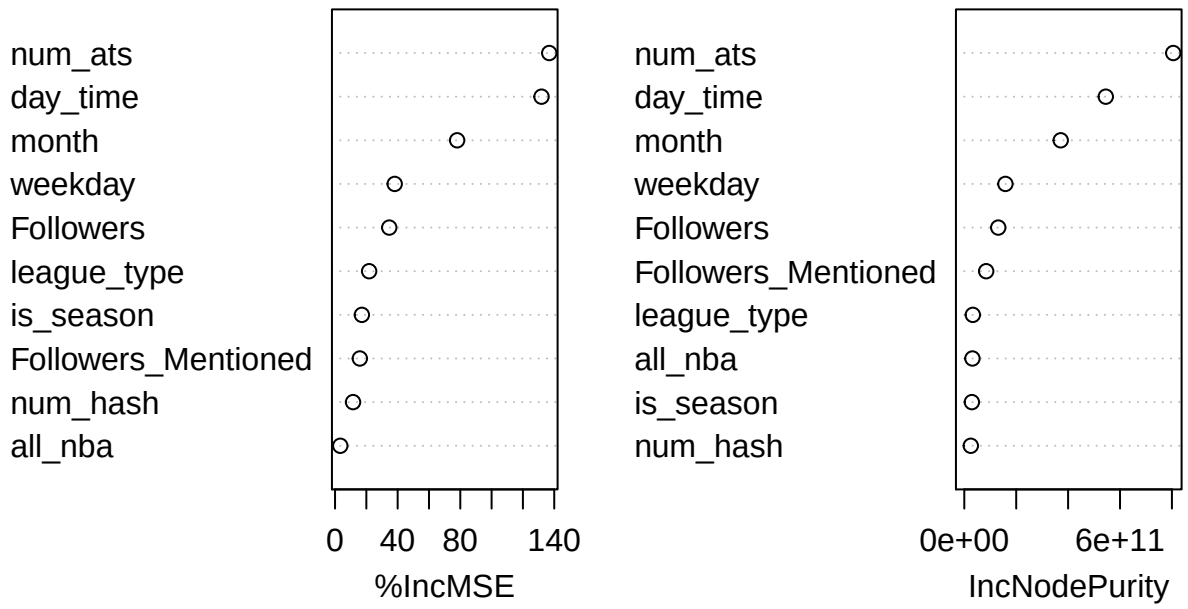
It appears that there are a few extreme outliers in our dataset that the model performs particularly poorly on. One in particular our model overshoots the most appears to be a post about Lebron James at a WNBA game. Intuitively, it makes sense that our model would perform poorly here. Mentioning Lebron would push the model to predict a higher engagement than would be normal for a WNBA game. Given how few outliers are, that the Random Forest is typically robust to outliers, and how rare a post like Lebron at a WNBA game is, we conclude that our model performs quite well.

Here we plot some of our variables to examine which are most critical to our model's accuracy. Notably, it appears that our variables vary in importance between the subsets for Videos, Albums, and Photos. This gives some credibility to our initial assumption that Video, Albums, and Photos should each have their own model.

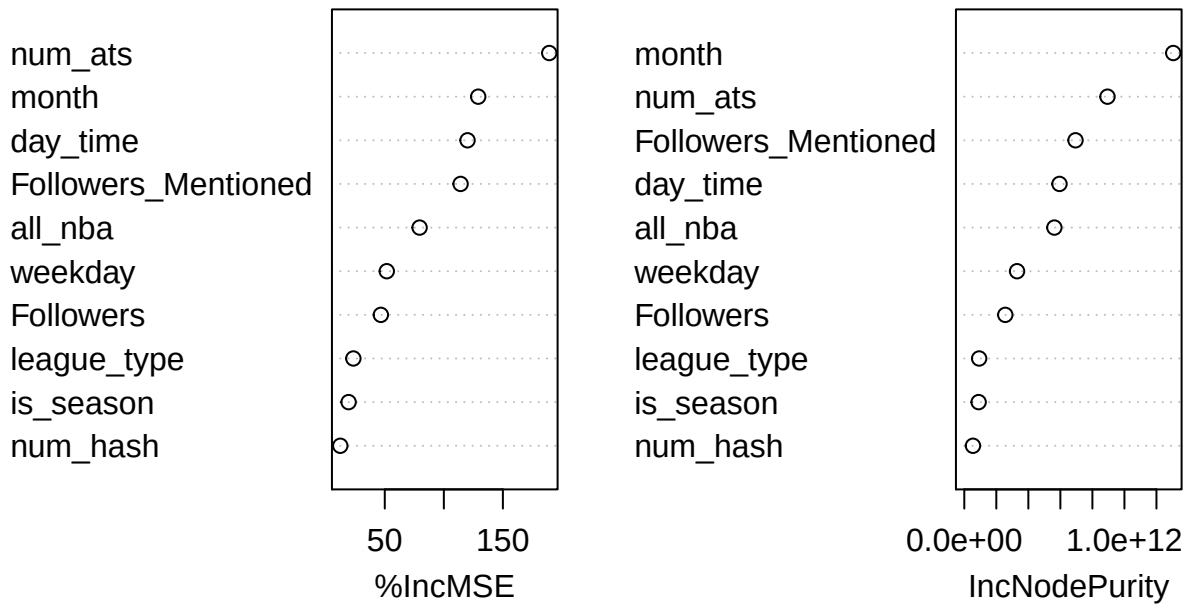
```
varImpPlot(RF_Video)
```




```
varImpPlot(RF_Album)
```




```
varImpPlot(RF_Photo)
```



Final Predictions

Finally, statisfied with our validation results, we train our Random Forest on the whole of the training set and apply it to the 1000-post holdout set.

```
# Read in holdout set and make same modifications as training set


holdout_pre <-
  read.csv("~/Desktop/NBA_Analytics/nba-business-hackathon/data/holdout_final.csv")

holdout <- holdout_pre[,c(3,4,5,6,7,11,13,14,16,19,20,21,23,24)]
holdout$Engagements <- as.integer(holdout$Engagements)
for (i in 1:length(holdout[,1])){
  if(holdout$is_season[i]==TRUE)(holdout$is_season[i]='True')
  if(holdout$is_season[i]==FALSE)(holdout$is_season[i]='False')
}
holdout$is_season <- as.factor(holdout$is_season)
```

```
## Final models for training and prediction

RF_Video_Final <- randomForest(Engagements ~ Followers+month+day_time
                               +weekday+Followers_Mentioned
                               +is_season+all_nba+league_type+num_ats+num_hash,
                               ntree=1000, mtry=7,importance=TRUE, data=nba_Video)
```

```
RF_Photo_Final <- randomForest(Engagements ~ Followers+month+day_time
                               +weekday+Followers_Mentioned
                               +is_season+all_nba+league_type+num_ats+num_hash,
                               ntree=1000, mtry=8,importance=TRUE, data=nba_Photo)

RF_Album_Final <- randomForest(Engagements ~ Followers+month+day_time
                               +weekday+Followers_Mentioned
                               +is_season+all_nba+league_type+num_ats+num_hash,
                               ntree=1000,mtry=7,importance=TRUE, data=nba_Album)
```

```
#Calculate predicted values and append to holdout dataframe


holdout_Video = holdout %>% filter(Type == "Video")
holdout_Photo = holdout %>% filter(Type == "Photo")
holdout_Album = holdout %>% filter(Type == "Album")
holdout_Video$Engagements = round(predict(RF_Video_Final, newdata=holdout_Video))
holdout_Photo$Engagements = round(predict(RF_Photo_Final, newdata=holdout_Photo))
holdout_Album$Engagements = round(predict(RF_Album_Final, newdata=holdout_Album))

holdout <- rbind(holdout_Album, holdout_Photo, holdout_Video)

write_csv(holdout, "holdout_final.csv")
```

The predictions are contained in the attached “holdout.csv” file.