Age Predicting Total Play Time Contribution to PLAICraft

Introduction:

We will be working with the Minecraft data from the Pacific Laboratory for Artificial Intelligence (PLAI). Minecraft is a popular open sandbox game, provides an open-ended environment where players can explore, build, and interact. PLAICraft, an initiative by PLAI, is interested in developing an advanced AI capable of mimicking human behaviors such as exploration, building, and decision-making. They want to analyze gameplay data and player demographics, to train AI to understand complex interactions and predict player engagement. This research not only advances AI capabilities in dynamic environments but also provides insights into user retention, adaptive systems, and AI safety.

The dataset provided by PLAI are the player.csv and sessions.csv datasets. The players.csv file contains 9 variables and 196 observations, providing descriptive characteristics for each player who has signed up for the project.

Variables in players dataset are listed below:

- experience - level of understanding with MineCraft - Pro, Veteran, Amateur, Beginner)(chr)
- hashedEmail - encrypted email address(chr)
- name - first name of player(chr)
- gender - player's gender(chr)
- individualId (lgl)
- subscribe (lgl)
- organizationName (lgl)
- played_hours - number of hours played on the platform in total(dbl)
- age - age in years of the player(dbL)

The sessions.csv file contains 5 variables and 1,535 observations, detailing each individual session a player has participated in on the site.

Variables

- hashedEmail - encrypted email address(chr)
- start_time - date and time of start of gameplay(chr)
- end_time - date and time of end of gameplay(chr)
- original_start_time (dbl)
- original_end_time (dbl)

The dataset we'll be using is only the players dataset. We will only be using the played_hours and age from the dataset to answer our question: "Is KNN regression or linear regression a better model to predict future contribution to PLAIcraft (total played hours) from age?"

Methods & Results:

In order to make our code reproducible, we loaded in the provided data set links as urls so another person can also load in the datasets without downloading it to their computer prior. Our desired variables, played_hours (response variable) and age, are only in the players.csv dataset which is already tidy. Thus, we do not need to wrangle this data into a tidy format. We then select only the desired variables to work with. In our exploratory data analysis (Figure 1) of these variables, we can see that the data appears to be skewed to the right with the majority of data points being within 0-25 years of age.

In [59]:
```r
library(tidyverse)
library(ggplot2)
#Link to the google docs for data

players_id <- "https://drive.google.com/uc?export=download&id=1Mw9vW0hjTJwRWx0bDXiSpYsO3gKogaPz"
sessions_id <- "https://drive.google.com/uc?export=download&id=14O91N5OlVkvdGxXNJUj5jIsV5RexhzbB"

#reading the data into R
players <- read_csv(players_id)
sessions <- read_csv(sessions_id)

#selecting the columns that are relevant
minecraft <- players |>
select(name, played_hours, age)
minecraft
```

```
Rows: 196 Columns: 9
── Column specification ────────────────────────────────────────────────────────
Delimiter: ","
chr (4): experience, hashedEmail, name, gender
dbl (2): played_hours, age
lgl (3): subscribe, individualId, organizationName

ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
Rows: 1535 Columns: 5
── Column specification ────────────────────────────────────────────────────────
Delimiter: ","
chr (3): hashedEmail, start_time, end_time
dbl (2): original_start_time, original_end_time

ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

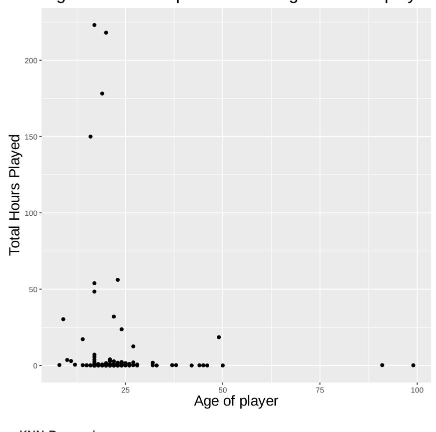
A tibble: 196 × 3

name	played_hours	age
<chr>	<dbl>	<dbl>
Morgan	30.3	9
Christian	3.8	17
Blake	0.0	17
Flora	0.7	21
Kylie	0.1	21
Adrian	0.0	17
Luna	0.0	19
Emerson	0.0	21
Natalie	0.1	17
Nyla	0.0	22
Lane	1.6	23
Daniela	0.0	17
Sarah	1.5	25
Thatcher	0.2	22
Niamh	0.0	17
Quinlan	0.0	22
Elodie	0.0	17
Xander	48.4	17
Marley	0.5	17
Ryker	0.6	19
Andy	0.3	8
Anastasia	0.1	17
Leah	1.0	17
Ren	0.0	17
Hugo	0.7	21
Kendall	0.6	28
Iman	0.0	17
Finn	0.0	23
Luca	1.8	23
Vivienne	0.1	18
⋮	⋮	⋮
Radwan	1.0	26
Ariana	0.3	17
Quinton	0.1	17
Rocco	0.1	17
Lyra	0.4	21
Amelia	1.8	32
Saif	0.0	20
Balthazar	0.0	50
Parker	0.0	17
Joaquim	0.3	17
Jesse	0.0	17
Jamie	2.7	21
Charlie	0.4	17
Finnian	0.1	17
Sebastián	2.1	24
Hunter	0.8	22
Liam	0.2	17
Sidney	32.0	22
Asher	1.7	17
Sam	0.1	18
Gabriela	0.1	44
Jasper	0.0	17
Lina	0.0	17
Orion	0.0	17
Rhys	0.0	20
Bailey	0.0	17
Pascal	0.3	22
Dylan	0.0	17
Harlow	2.3	17
Ahmed	0.2	91

The lack of older people data may result in our model unable to predict as well for higher ages (over 50 years old) and also often predict lower hours played overall due to little data of people with high played hours (above 50 hours).

In [60]:
```r
set.seed(5)

#Scatterplot of age vs played_hours
minecraft_plot <- minecraft |> ggplot(aes(x = age, y = played_hours)) +
    geom_point() +
    ggtitle("Figure 1: Scatter plot between age and total played hours") +
    labs(y = "Total Hours Played", x = "Age of player") +
    theme(title = element_text(size = 17))
minecraft_plot
```

Figure 1: Scatter plot between age and total played



KNN Regression

We implemented a KNN regression model alongside a linear regression model to predict future observations based on past numerical data. To evaluate the models' performance, we split the dataset into training (75%) and test (25%) sets. For the KNN model, we used cross-validation to determine the optimal value of KKK. Our process began by creating a preprocessing recipe that included standardization to ensure the data was properly prepared. Next, we defined a model specification for K-nearest neighbors regression. We then set up a 5-fold cross-validation object to enable an assessment of the model's performance. Finally, we combined the recipe and model specification into a single workflow.

In [61]:
```r
set.seed(5)

library(tidymodels)
library(gridExtra)
minecraft_split <- initial_split(minecraft, prop = 0.75, strata = played_hours)
minecraft_train <- training(minecraft_split)
minecraft_test <- testing(minecraft_split)
```

In [62]:
```r
set.seed(5)

#KNN recipe
minecraft_recipe <- recipe(played_hours ~ age, data = minecraft_train) |>
    step_scale(all_predictors()) |>
    step_center(all_predictors())
minecraft_recipe

minecraft_spec <- nearest_neighbor(weight_func = "rectangular",
    neighbors = tune()) |>
    set_engine("kknn") |>
    set_mode("regression")

minecraft_vfold <- vfold_cv(minecraft_train, v = 5, strata = played_hours)

minecraft_wkflw <- workflow() |>
    add_recipe(minecraft_recipe) |>
    add_model(minecraft_spec)
    minecraft_wkflw

gridvals <- tibble(neighbors = seq(from = 1, to = 100, by = 3))

minecraft_results <- minecraft_wkflw |>
    tune_grid(resamples = minecraft_vfold, grid = gridvals) |>
    collect_metrics() |>
    filter(.metric == "rmse")

# show the results
minecraft_results

minecraft_min <- minecraft_results |>
filter(mean == min(mean))

minecraft_min
```

```
── Recipe ──────────────────────────────────────────────────────────────────────

── Inputs

Number of variables by role

outcome:   1
predictor: 1

── Operations

• Scaling for: all_predictors()

• Centering for: all_predictors()
```

```
══ Workflow ════════════════════════════════════════════════════════════════════
Preprocessor: Recipe
Model: nearest_neighbor()

── Preprocessor ────────────────────────────────────────────────────────────────
2 Recipe Steps

• step_scale()
• step_center()

── Model ───────────────────────────────────────────────────────────────────────
K-Nearest Neighbor Model Specification (regression)

Main Arguments:
  neighbors = tune()
  weight_func = rectangular

Computational engine: kknn
```

A tibble: 34 × 7

neighbors	.metric	.estimator	mean	n	std_err	.config
<dbl>	<chr>	<chr>	<dbl>	<int>	<dbl>	<chr>
1	rmse	standard	23.95210	5	6.748632	Preprocessor1_Model01
4	rmse	standard	21.05845	5	7.496176	Preprocessor1_Model02
7	rmse	standard	20.12665	5	7.071529	Preprocessor1_Model03
10	rmse	standard	19.72705	5	7.014795	Preprocessor1_Model04
13	rmse	standard	19.63424	5	6.974554	Preprocessor1_Model05
16	rmse	standard	19.60907	5	6.951302	Preprocessor1_Model06
19	rmse	standard	19.60416	5	6.937198	Preprocessor1_Model07
22	rmse	standard	19.44468	5	6.989364	Preprocessor1_Model08
25	rmse	standard	19.52551	5	6.946234	Preprocessor1_Model09
28	rmse	standard	19.48854	5	6.899946	Preprocessor1_Model10
31	rmse	standard	19.94715	5	6.558536	Preprocessor1_Model11
34	rmse	standard	19.93844	5	6.625125	Preprocessor1_Model12
37	rmse	standard	19.81645	5	6.698000	Preprocessor1_Model13
40	rmse	standard	19.71402	5	6.701910	Preprocessor1_Model14
43	rmse	standard	19.86553	5	6.446024	Preprocessor1_Model15
46	rmse	standard	19.93881	5	6.429092	Preprocessor1_Model16
49	rmse	standard	19.73730	5	6.403030	Preprocessor1_Model17
52	rmse	standard	19.75421	5	6.428593	Preprocessor1_Model18
55	rmse	standard	19.68438	5	6.473138	Preprocessor1_Model19
58	rmse	standard	19.62350	5	6.509339	Preprocessor1_Model20
61	rmse	standard	19.60209	5	6.530529	Preprocessor1_Model21
64	rmse	standard	19.57527	5	6.546911	Preprocessor1_Model22
67	rmse	standard	19.55412	5	6.569539	Preprocessor1_Model23
70	rmse	standard	19.52984	5	6.589011	Preprocessor1_Model24
73	rmse	standard	19.50941	5	6.609739	Preprocessor1_Model25
76	rmse	standard	19.44930	5	6.636734	Preprocessor1_Model26
79	rmse	standard	19.44674	5	6.662440	Preprocessor1_Model27
82	rmse	standard	19.41546	5	6.662287	Preprocessor1_Model28
85	rmse	standard	19.50404	5	6.630114	Preprocessor1_Model29
88	rmse	standard	19.49416	5	6.616508	Preprocessor1_Model30
91	rmse	standard	19.56385	5	6.615039	Preprocessor1_Model31
94	rmse	standard	19.53877	5	6.599130	Preprocessor1_Model32
97	rmse	standard	19.53204	5	6.609026	Preprocessor1_Model33
100	rmse	standard	19.55488	5	6.604427	Preprocessor1_Model34

A tibble: 1 × 7

neighbors	.metric	.estimator	mean	n	std_err	.config
<dbl>	<chr>	<chr>	<dbl>	<int>	<dbl>	<chr>
82	rmse	standard	19.41546	5	6.662287	Preprocessor1_Model28

In the next step, we performed cross-validation over a grid of neighbor values ranging from 1 to 100. The optimal K, corresponding to the minimum RMSPE, was determined to be 82. We then evaluated our model on the unseen test data. To do this, we first retrained the model using K=82 neighbors. Next, we used the model to generate predictions on the test data and applied the metrics function to compute a summary of the regression quality. The RMPSE found was 39.58.

In [63]:
```r
set.seed(5)

#Evaluating on the test set
kmin <- minecraft_min |> pull(neighbors)

#Model for knn regression
minecraft_spec <- nearest_neighbor(weight_func = "rectangular", neighbors = kmin)|>
    set_engine("kknn") |>
    set_mode("regression")

#Fitting everything through the workflow onto the data
minecraft_fit <- workflow() |>
    add_recipe(minecraft_recipe) |>
    add_model(minecraft_spec) |>
    fit(data = minecraft_train)

#Finding the rmse
minecraft_summary <- minecraft_fit |>
    predict(minecraft_test) |>
    bind_cols(minecraft_test) |>
    metrics(truth = played_hours, estimate = .pred) |>
    filter(.metric == 'rmse')
minecraft_summary
```

A tibble: 1 × 3

.metric	.estimator	.estimate
<chr>	<chr>	<dbl>
rmse	standard	39.5795

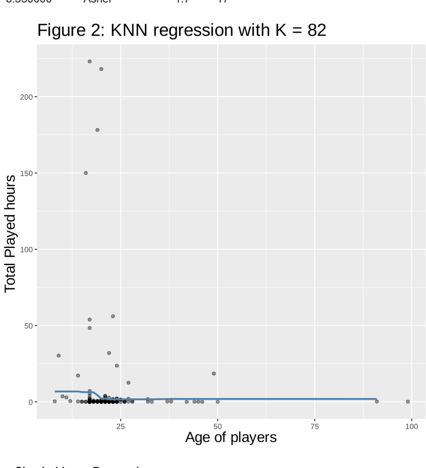
Final Plotting of the Regression Model

In [64]:
```r
set.seed(5)
#Final Plotting of the data
minecraft_preds <- minecraft_fit |>
    predict(minecraft_train) |>
    bind_cols(minecraft_train)
minecraft_preds

minecraft_plot_final <- minecraft |> ggplot(aes(x = age, y = played_hours)) +
    geom_point(alpha = 0.4) +
        geom_line(data = minecraft_preds,
        mapping = aes(x = age, y = .pred),
        color = "steelblue",
        linewidth = 1) +
    xlab("Age of players") +
    ylab("Total Played hours ") +
    ggtitle(paste0("Figure 2: KNN regression with K = ", kmin)) +
    theme(title = element_text(size = 17))
minecraft_plot_final
```

A tibble: 146 × 4

.pred	name	played_hours	age
<dbl>	<chr>	<dbl>	<dbl>
2.081707	Kylie	0.1	21
6.350000	Adrian	0.0	17
4.364634	Luna	0.0	19
6.350000	Natalie	0.1	17
1.609756	Nyla	0.0	22
6.350000	Daniela	0.0	17
6.350000	Niamh	0.0	17
1.609756	Quinlan	0.0	22
6.350000	Elodie	0.0	17
6.350000	Ren	0.0	17
6.350000	Iman	0.0	17
1.617073	Finn	0.0	23
6.185366	Vivienne	0.1	18
1.617073	Ayman	0.1	23
2.081707	London	0.1	21
6.350000	Farid	0.0	17
1.642683	Vasco	0.0	33
6.350000	Ishaan	0.0	17
6.350000	Yvette	0.1	17
2.081707	Mina	0.0	21
1.617073	Umar	0.0	24
6.350000	Winston	0.1	17
1.617073	Edmund	0.0	23
1.868293	Jude	0.0	42
2.020732	Kai	0.0	20
6.350000	Bijan	0.0	17
6.407317	Ophelia	0.1	15
2.081707	Magnus	0.0	21
1.609756	Hadi	0.0	22
6.350000	Faye	0.0	17
⋮	⋮	⋮	⋮
1.639024	Aiden	1.4	25
6.350000	Winslow	5.6	17
1.617073	Cyrus	2.2	24
1.639024	Isidore	12.5	27
6.185366	Pablo	0.9	18
6.780488	Rafael	2.9	11
6.780488	Zane	3.6	10
6.780488	Kyrie	17.2	14
6.350000	Alex	53.9	17
6.350000	Hiroshi	223.1	17
6.350000	Suki	1.0	17
6.350000	Delara	150.0	16
6.350000	Sakura	1.2	17
1.639024	Ibrahim	2.0	27
1.609756	Sophia	2.7	22
2.081707	Scarlett	4.0	21
6.350000	Ella	1.0	17
6.350000	Arash	7.1	17
1.868293	Dante	18.5	49
1.617073	Amelie	0.7	24
1.617073	Dana	56.1	23
2.020732	Caden	1.1	20
6.350000	Atlas	1.0	17
6.350000	Aaron	1.2	17
1.642683	Amelia	1.8	32
2.081707	Jamie	2.7	21
1.617073	Sebastián	2.1	24
1.609756	Hunter	0.8	22
1.609756	Sidney	32.0	22
6.350000	Asher	1.7	17

Figure 2: KNN regression with K = 82



Simple Linear Regression

For the simple linear regression, we will find the straight line of best through the training data and use it for predictions. First, we defined the model specification and created a preprocessing recipe. We then fit the simple linear regression model, which provided the coefficients for the best-fit line: the intercept 8.67 and the slope -0.181. Finally, we used the model to generate predictions on the test dataset to evaluate its performance. The RMPSE found was 39.198.

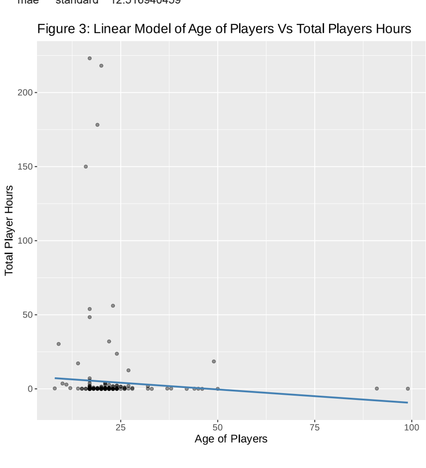
In [65]:
```r
set.seed(5)
#linear regression models
lm_spec <- linear_reg() |>
    set_engine("lm") |>
    set_mode("regression")

lm_recipe <- recipe(played_hours ~ age, data = minecraft_train)

lm_fit <- workflow() |>
    add_recipe(lm_recipe) |>
    add_model(lm_spec) |>
    fit(data = minecraft_train)
lm_fit

lm_test_results <- lm_fit |>
    predict(minecraft_test) |>
    bind_cols(minecraft_test) |>
    metrics(truth = played_hours, estimate = .pred)
lm_test_results

#Visualization of the linear model on the data
age_prediction_grid <- tibble(
    age = c(
        minecraft |> select(age) |> min(),
        minecraft |> select(age) |> max()))

minecraft_preds <- lm_fit |>
    predict(age_prediction_grid) |>
    bind_cols(age_prediction_grid)

#Plotting the linear line on the scatterplot data
lm_plot_final <- ggplot(minecraft, aes(x = age, y = played_hours)) +
    geom_point(alpha = 0.4) +
    geom_line(data = minecraft_preds,
              mapping = aes(x = age, y = .pred),
              color = "steelblue",
              linewidth = 1) +
    ggtitle("Figure 3: Linear Model of Age of Players Vs Total Players Hours") +
    xlab("Age of Players") +
    ylab("Total Player Hours") +
    theme(text = element_text(size = 13))
lm_plot_final
```

```
══ Workflow [trained] ══════════════════════════════════════════════════════════
Preprocessor: Recipe
Model: linear_reg()

── Preprocessor ────────────────────────────────────────────────────────────────
0 Recipe Steps

── Model ───────────────────────────────────────────────────────────────────────

Call:
stats::lm(formula = ..y ~ ., data = data)

Coefficients:
(Intercept)          age
     8.6668      -0.1814
```

A tibble: 3 × 3

.metric	.estimator	.estimate
<chr>	<chr>	<dbl>
rmse	standard	39.198033685
rsq	standard	0.001219333
mae	standard	12.516940459

Figure 3: Linear Model of Age of Players Vs Total Players Hours



Discussion:

Using KNN regression model, we found that the lowest RMSE for the number of neighbors is K = 82. However, we can inferentially assume that this is not a good K value as each data point will consider 82 neighbors. Having neighbours as 82 will cause underfitting due to the whole dataset only containing 196 observations, which our K value is almost half of. It will cause the model to likely ignore any smaller but obvious trends that are present in the dataset, giving an inaccurate representation of the data.

Using the simple linear regression model, we found a weak negative relationship between age and total player hours with a slope of -0.181 and intercept of 8.67. This negative correlation indicates that older generations are less likely to contribute to playing PLAIcraft. The RMPSE for a KNN regression was 39.58, whereas a simple linear regression was 39.198. A smaller RMPSE represents a more suitable model that fits and reflects the data more accurately. Thus, we can conclude that a linear model is better than the linear model for our selected data and experimental question because it has a smaller RMPSE.

This is what we expected to find as younger generations are known to engage with video games more frequently. Minecraft, launched in 2009, was designed to appeal to children and young adults, offering a fun, collaborative gaming platform. Additionally, Gen Z and Millennials grew up immersed in video games and computer usage, making them more familiar and comfortable with technology. In contrast, older generations did not have the same level of exposure to computers or the internet during their formative years, which likely contributes to lower engagement with video games among them.

The purpose of this study was to analyze gameplay data and player demographics, to train AI to understand complex interactions and predict player engagement. This teaches AI that the average player contributing the most hours in minecraft is often younger. Thus, game developers can begin to target a younger audience and appeal to them, as the younger generations are most likely to continue contributing to their game. Future questions this could lead to are investigating the older generation demographics like "how can we get older people to contribute to game play"? Additionally, we could explore younger generation demographics like "what aspects appeal towards younger generations"?