Electric Vehicle Energy Source

Group 5 AE: Hibah, Kenny, Neelum

2024-05-28

Abstract

an "executive summary" of no more than 100 words, that explains what did you do and what are your most important findings.

Introduction

Electric cars have become the new paper straw in regards to saving the planet. Just as plastic straws once symbolized environmental harm, the shift to paper straws highlighted the need for sustainability. However, while strides have been made to curb plastic pollution, industries like commercial fishing continue to significantly impact our oceans despite growing pressure for change. This begs the question: do electric vehicles (EVs) face a similar scrutiny? Are they truly more sustainable, or does their reliance on electricity sourced from non-renewable means still contribute to environmental degradation? In recent years, green washing across various media platforms has become increasingly common amidst a global push for sustainability. Traditional industries such as gas and coal mining have faced mounting criticism for their environmental toll, prompting a shift towards alternatives like electric vehicles. Yet, the narrative surrounding EVs often overlooks the source of their power. While they may not rely on traditional fuels, the electricity used to charge them may still be derived from non-renewable sources, undermining their eco-friendly image. Additionally, the mining of lithium for EV batteries raises ethical concerns surrounding labor practices. However, our focus lies in exploring the sustainability implications of EVs within the context of Washington State.Our project aims to unravel the intricacies of electric vehicle sustainability within Washington State. By delving into the energy sources powering EVs, we seek to gauge their true environmental impact. Leveraging predictive modeling, we analyze data to forecast the future prevalence of EVs in the state. Moreover, by examining each state's electricity data, we can identify regions where EV adoption would yield the greatest environmental benefits. This insight is invaluable for urban planning initiatives, including the conversion of gas stations into EV charging infrastructure. Understanding the nuanced sustainability implications of electric vehicles is crucial in navigating the transition towards greener transportation. Our report serves as a comprehensive exploration of EVs, shedding light on their true environmental footprint. By dissecting the sources of electricity fuelling these vehicles, we challenge the notion of EVs as a panacea for environmental woes. Readers will gain insight into the complexities of sustainable transportation and the importance of considering regional energy dynamics in assessing the viability of EVs. Ultimately, our findings offer valuable guidance for policymakers, urban planners, and environmentally-conscious consumers alike.

Data

The first dataset is from the U.S. Energy Information Administration (EIA), which displays each state's net generation for electrical utility broken down into categories for each state. The categories are coal, petroleum liquids, petroleum coke, natural gas, nuclear, conventional hydroelectric, other renewable, hydroelectric pumped storage, and all solar. This data source will provide insight on what the main form of energy generated is for each state, which will highlight what the electric vehicle is most likely powered by. The second dataset is from Kaggle and is titled Electric Vehicle Population, which was created by Ratik Kakkar. This dataset contains the number of electric vehicles on the roads in Washington State. This dataset has been taken from the data catalog of data.gov and it showcases the number and other technical information

about the electric vehicles that are currently on the roads of The United States of America. This data will be useful not only to figure out how many EVs Washington has but also to create a predictive model to see the increase in the EV population on the roads over a period of time. Both datasets are freely available to everyone. The EIA data is more accessible as you do not need an account to download or view the data, unlike the EV dataset for Kaggle, where you need to create a free account to access the data. According to the public domain and use of EIA content, U.S. government publications are in the public domain and are not subject to copyright protection. The EV data from Kaggle was collected from data.wa.gov, which is a U.S. government publication, meaning it is public domain and not subject to copyright. The EIA data originally had 26 columns and 573 rows. Each row represented a location and the type of electricity produced. In the filtered data for EIA, there are 6 rows and each row represents a fuel type and the information associated with each type of electricity. The EV data original data had 17 columns and 130,433 rows. Each row represents a unique EV car that is on the roads with the unique identifier being the DOL Vehicle ID. The cleaned data has 11 rows and each row represents a year from 2012 to 2022, and the total count of EVs by Model Year. So, that means each year represents how many new vehicles were added on the roads, not the total EVs for that year. The relevant variables are total_ev_count, which groups the EVs by the model year and counts how many are in each model year. This is the only data we have for time, so this is why we have to group by model year, but this can also cause an issue in the used car market because someone could have bought a 2016 Tesla Model S in 2022 but it would show up in the 2016 data because that is the model year. Another relevant variable is the type column in the EIA data because we had to separate the state and type, which were originally in one column in the dataset using this line of code: separate(description, into = c("state", "type"), sep = " : ", fill ="right"). In the unfiltered dataset from EIA, there are many NA values because the table has categories titled coal, petroleum liquids, petroleum coke, natural gas, nuclear, conventional hydroelectric, other renewables, hydroelectric pumped storage, and all solar for each state. However, not each state produces each type of electricity, so some states use 0 while some use the NA value to represent not producing that form of electricity. We used the code drop_na() %>% to filter out the NA values so it would not mess up the plotting and analysis. One important thing to add is that there are three regional power grids that allow for the transmission of electricity across state lines, which can affect how much energy one state is consuming and producing. The three grids are the Eastern Interconnection, Western Interconnection, and the Electric Reliability Council of Texas (ERCOT).

EIA Dataset: https://www.eia.gov/electricity/data/browser/#/topic/0?agg=2,0,1&fuel=vf04&geo=00fvvvvvvvvvo&sec=8&linechart=ELEC.GEN.ALL-CT-1.A&columnchart=ELEC.GEN.ALL-CT-1.A&map=ELEC.GEN.ALL-CT-1.A&freq=A&start=2001&end=2023&ctype=columnchart&ltype=pin&rtype=s&pin=&rse=0&maptype=0

Electric Vehicle Dataset: https://www.kaggle.com/datasets/ratikkakkar/electric-vehicle-population-data

Original Electric Vehicle Population Data: https://catalog.data.gov/dataset/electric-vehicle-population-data

Citations:

Source: U.S. Energy Information Administration (Oct 2008)

data.wa.gov. (2022). Electric Vehicle Population Data [Data set]. Kaggle. https://doi.org/10.34740/KAGGLE/DSV/4693417

Methods

EIA Methods:

For the EIA data, we first imported the CSV file. Then, when looking at the columns as discussed above, we separated the description into two variables, state and type, by using "sep =":" " as the divider. After that, we removed the NA values, then filtered to only get Washington as the state because that is what we will be comparing to for the beginning. To remove the NA values, we implemented the following code: "drop_na() %>%". This allowed us to perform calculations without getting errors because you cannot divide by NA. Later, we created another data frame that had all the states, and we filtered it by units == "thousand

megawatt hours" because all the titles of the states did not have any, so now the data frame only has the states with data. Then, we transformed the data from wide form to long form using pivot_longer and transformed each column name that was a year to a numeric value. Using ggplot, we got a baseline for Washington using the geom_bar to see what the main sources of energy produced are. This plot is helpful to then compare the national averages as well as each state. Using the long-form Washington State data, we were able to predict what the generation will be for the upcoming years. To do this, I researched how to predict trends using R, and I found that the easiest way was to install the forecast library and use the ARIMA test. I created two empty lists, one for models and one for forecast, and created a for loop for unique types. Then, I used my Python knowledge to fit the data using auto.arima, which is a function I researched online. Then, with the forecast library, I made it go for 10 years and created a data frame, which I then used ggplot to create a model. However, all the labels on the x-axis for the graph were scrambled together, so I had to use axis.text.x to angle the text so they would not be on top of each other.

EV Methods:

For the EV data, first, we began by loading in the CSV, then changing Model Year to Year. This is because when we were running the linear regression model later on, there was an error with the naming, and this was the easiest way to fix it. Then, we filtered the data to just Washington and between 2012 and 2022, and selected the columns: State, Postal Code, Year, Make, Electrical Vehicle Type, and we grouped by year, which is the model year of the car. Next, we plotted the graphs to show the change over the years using a line plot and a bar plot to see which one skewed the data less and was a more accurate representation. Following that, we created the linear regression model to predict what the change will be in the coming years and to see the growth. We implemented the lm function and created a data frame for the future years 2023 - 2032 and used the predict function to run the analysis. Then, we used rbind to mutate the total_ev_count and set that equal to the linear forecast. Finally, we plotted the graph and highlighted where the linear regression forecast began by using a red dot, not a black dot, so it stands out.

Merging both datasets:

Now that both predictive models have been created and we have the baseline, it is now time that we merge the models using a key that they both have to create one table. The key they had in common is year and model year, which is important to note that the year data for the EIA is for the specific year but the model year is for the model of the car year. Just to understand the relationship of the data, we used head() to get a glimpse, and we found that there are 6 rows and 6 columns that are displayed in the data frame. From here, we plotted this to better grasp the relationship between both the EVs and the Electricity. To plot this relationship, the x-axis was labeled years and the y-axis was the count for both the Generation and the Total EV count. There is also a legend on the right that labels this clearly using Data Type to differentiate both. Also, because it is hard to understand scale, we used geom_text to label the Total EV count line in blue that is towards the bottom of the graph. Next, we took the predictive models that we created above and merged them into this dataset as well to get the data all the way up to 2032. To merge the predictive models, we need to combine the predictive model for EVs and the one for generation. To do this, I used inner_join and used the year key to make sure they lined up. I used the same process to create the graph for the 2016-2022 data and the 2016-2032 data.

Results

Present your results. This section may touch the methodology and data, but it should focus on the results. It should be beefed up with figures and tables, and other kind of results that address your question and story line.

It is the section that answers the questions you asked above.

Discuss

Discuss your findings. While the previous section was about results, this should be about meaning of the results.

what do the results tell you regarding your question you asked in introduction, or about the story line you want to focus on?

What are the limitations–how far do you think you can extend your claims or story?

Give directions for future work–what analysis/data you might need to extend your story further?

Summary

At the end, it should include a summary with a takeaway message.

Code

Libraries

```
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyr)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v purrr     1.0.2
## v ggplot2   3.5.1     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(ggplot2)
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

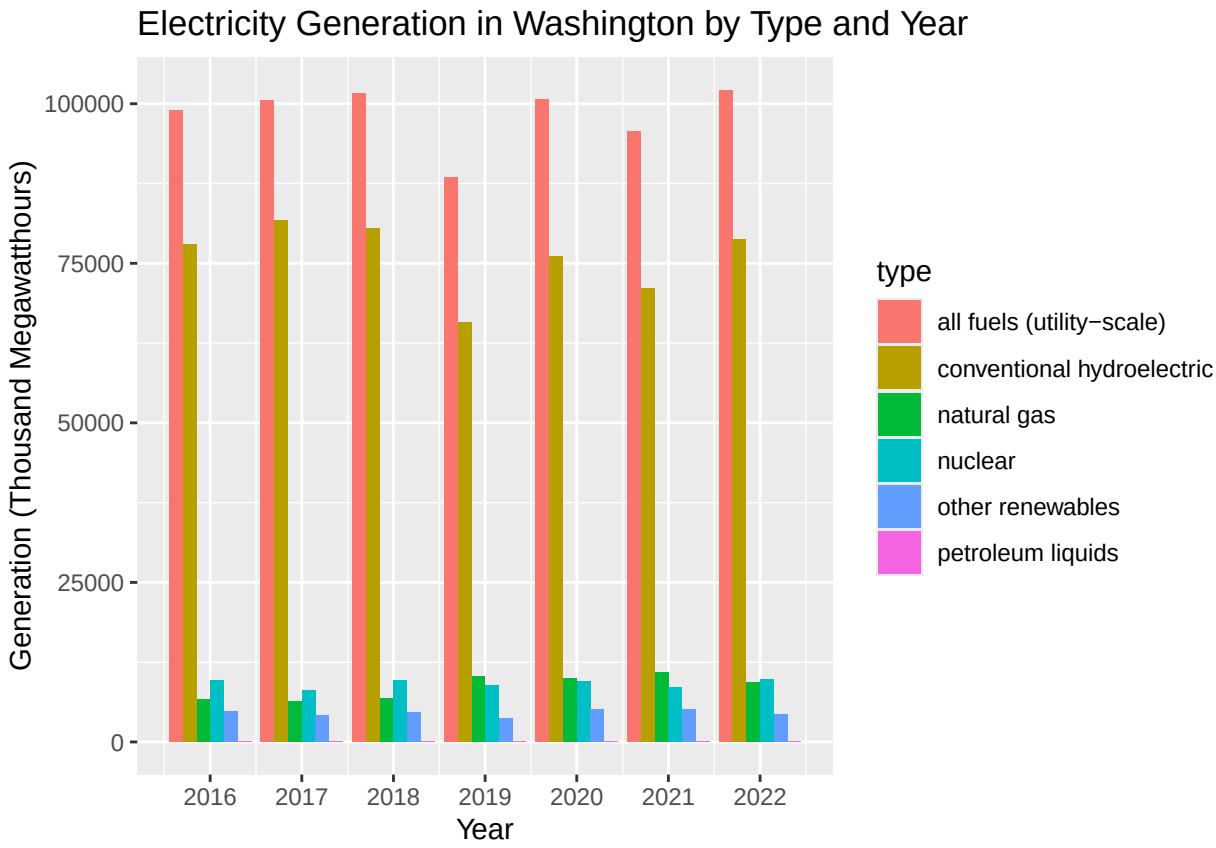
Washington EIA Data

```
generation_first <- read_delim("Net_generation_for_electric_utility.csv",
                               skip = 4, na = "--")
```

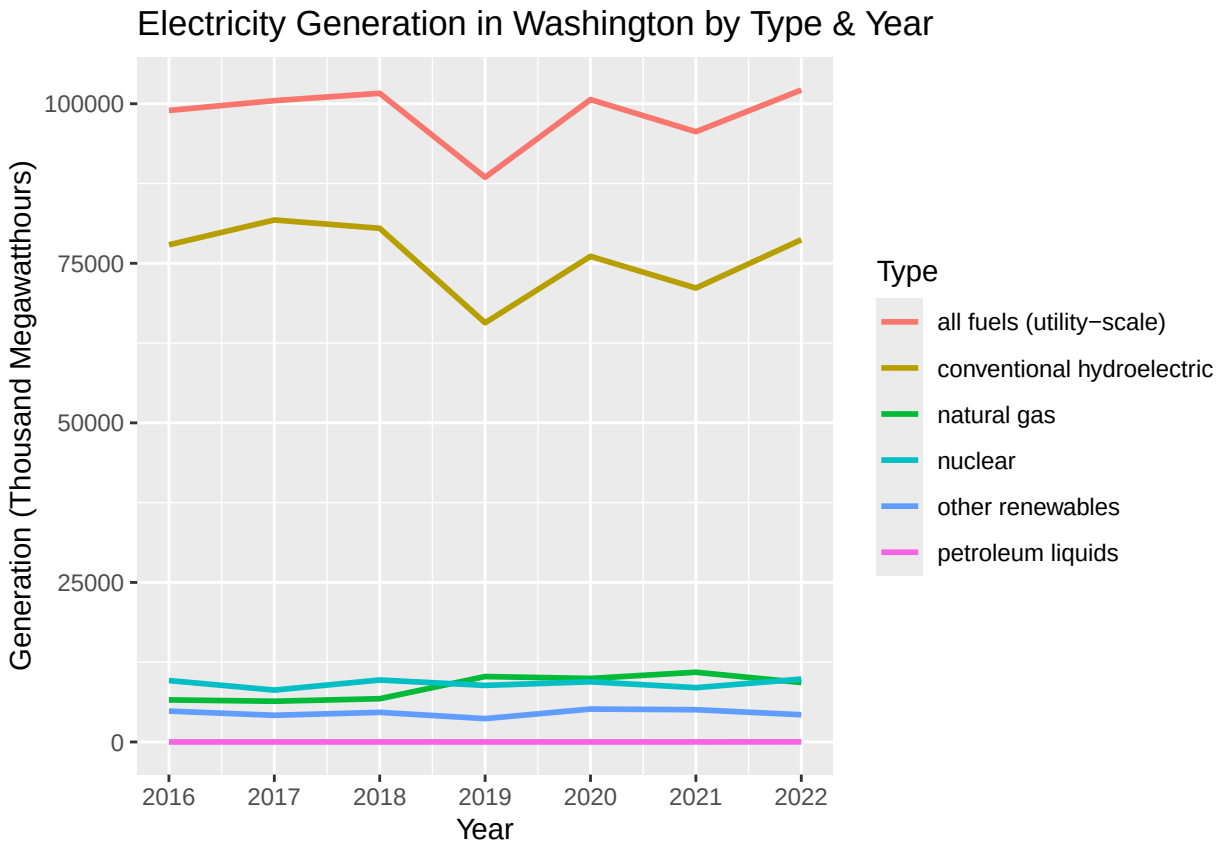
```
## Warning: One or more parsing issues, call `problems()` on your data frame for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)
```

```
## Rows: 573 Columns: 26
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (6): description, units, source key, 2011, 2013, 2023
## dbl (20): 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2012, ...
##
## 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.
```

```
generation <- generation_first[-(1:3), ]
generation <- generation %>%
  separate(description, into = c("state", "type"), sep = " : ",
           fill = "right") %>%
  drop_na() %>%
  filter(state == "Washington") %>%
  select(-c(`2001`:`2015`), -`2023`)

#transform data set from wide form to long form
generation_long <- generation %>%
# transform each column name to numeric values
  mutate(across(`2016`:`2022`, as.numeric)) %>%
  pivot_longer(cols = starts_with("20"), names_to = "year",
               values_to = "generation") %>%
  mutate(year = as.numeric(year)) %>%
  select(-`source key`)



# plot
generation_long %>%
  ggplot(aes(x = year, y = generation, fill = type)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Electricity Generation in Washington by Type and Year",
       x = "Year",
       y = "Generation (Thousand Megawatthours)") +
  scale_x_continuous(breaks = seq(2013, 2022, by = 1))
```



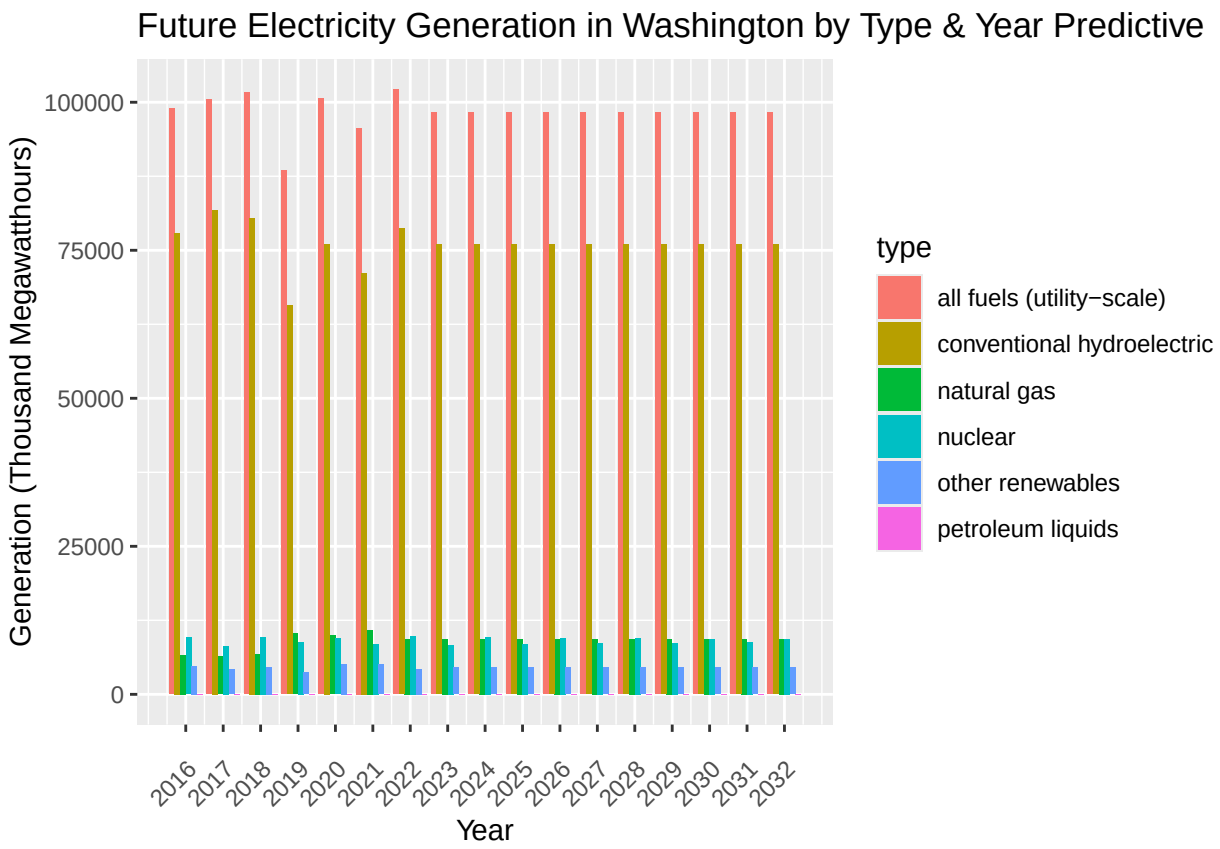

```
ggplot(generation_long, aes(x = year, y = generation, color = type)) +
  geom_line(linewidth = 1) +
  labs(title = "Electricity Generation in Washington by Type & Year",
       x = "Year",
       y = "Generation (Thousand Megawatthours)",
       color = "Type") +
  scale_x_continuous(breaks = seq(2013, 2022, by = 1))
```



##Predicting Model

```
generation_long <- generation_long %>%
  filter(!is.na(generation))
models <- list()
forecasts <- list()


unique_types <- unique(generation_long$type)
for (t in unique_types) {

  type_data <- generation_long %>% filter(type == t)
  ts_data <- ts(type_data$generation, start = min(type_data$year),
                frequency = 1)
  fit <- auto.arima(ts_data)
  models[[t]] <- fit
  fc <- forecast(fit, h = 10)
  forecasts[[t]] <- fc
}
forecast_data <- data.frame(
  year = rep((max(generation_long$year) + 1):(max(generation_long$year) + 10),
             times = length(unique_types)),
  generation = unlist(lapply(forecasts, function(x) x$mean)),
  type = rep(unique_types, each = 10)
)


combined_data <- bind_rows(generation_long, forecast_data)
```

```
ggplot(combined_data, aes(x = year, y = generation, fill = type)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Future Electricity Generation in Washington by Type & Year Predictive",
       x = "Year",
       y = "Generation (Thousand Megawatthours)") +
  scale_x_continuous(breaks = seq(min(combined_data$year),
                                  max(combined_data$year), by = 1)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1, vjust = 1,
                                   margin = margin(t = 10)))
```

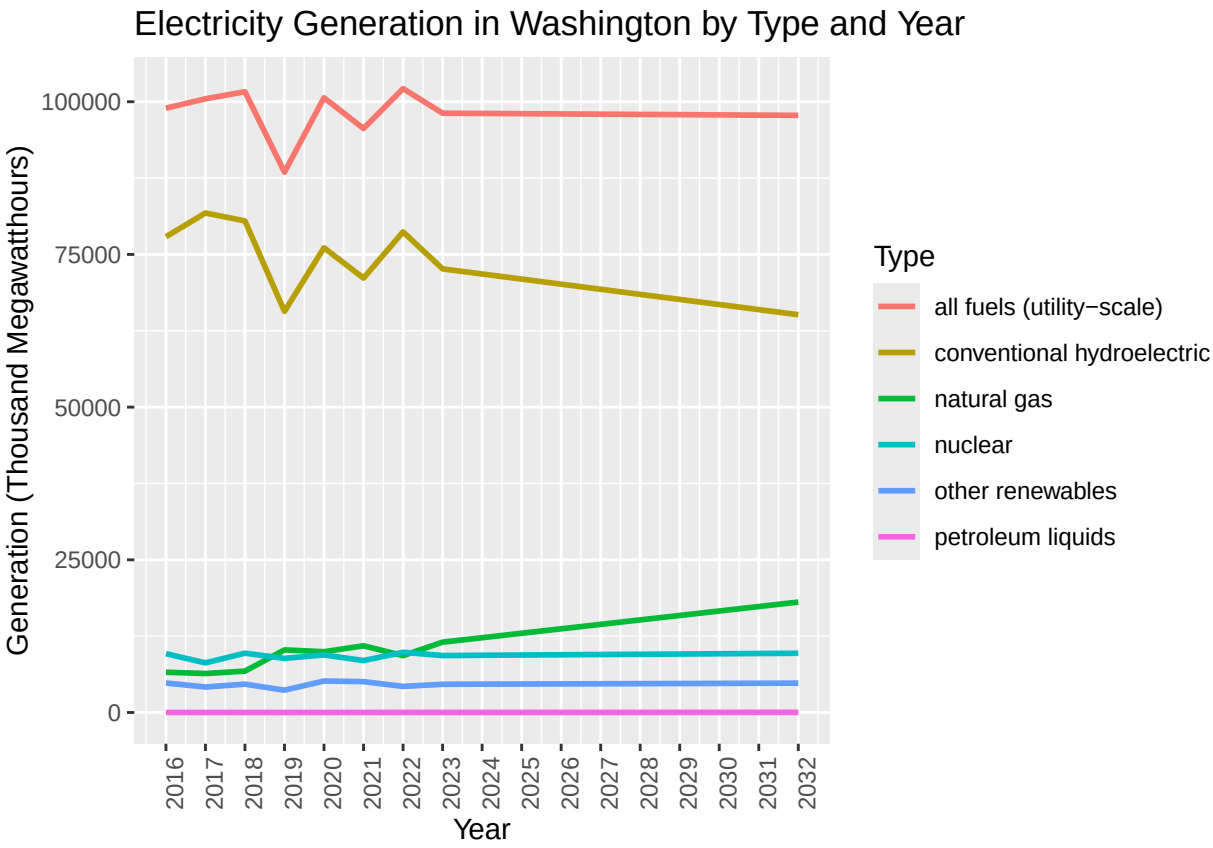


##Linear Model

```
models <- list()
forecasts <- list()

unique_types <- unique(generation_long$type)
for (t in unique_types) {
  type_data <- generation_long %>% filter(type == t)

  lm_model <- lm(generation ~ year, data = type_data)
  models[[t]] <- lm_model

  forecast_values <-
    predict(lm_model, newdata = data.frame(year = (max(type_data$year) + 1):
                                             (max(type_data$year) + 10)))
  forecasts[[t]] <- forecast_values
```

```
}

forecast_data <- data.frame(
  year = rep((max(generation_long$year) + 1):(max(generation_long$year) + 10),
             times = length(unique_types)),
  generation = unlist(forecasts),
  type = rep(unique_types, each = 10),
  state = rep("Washington", times = nrow(forecast_data)),
  units = rep("thousand megawatthours", times = nrow(forecast_data)))


combined_data <- bind_rows(generation_long, forecast_data)

ggplot(combined_data, aes(x = year, y = generation, color = type)) +
  geom_line(size = 1) +
  labs(title = "Electricity Generation in Washington by Type and Year",
       x = "Year",
       y = "Generation (Thousand Megawatthours)",
       color = "Type") +
  scale_x_continuous(breaks = seq(min(combined_data$year),
                                  max(combined_data$year), by = 1)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



All state EIA data

```
generation_states <- generation_first %>%
  separate(description, into = c("state", "type"), sep = " : ",
           fill = "right") %>%
  drop_na() %>%
  filter(units == "thousand megawatthours") %>%
  select(-c(`2001`:`2015`), -`2023`)

#transform data set from wide form to long form
generation_long_states <- generation_states %>%
# transform each column name to numeric values
  mutate(across(`2016`:`2022`, as.numeric)) %>%
  pivot_longer(cols = starts_with("20"),
               names_to = "year", values_to = "generation") %>%
  mutate(year = as.numeric(year)) %>%
  select(-`source key`)
```

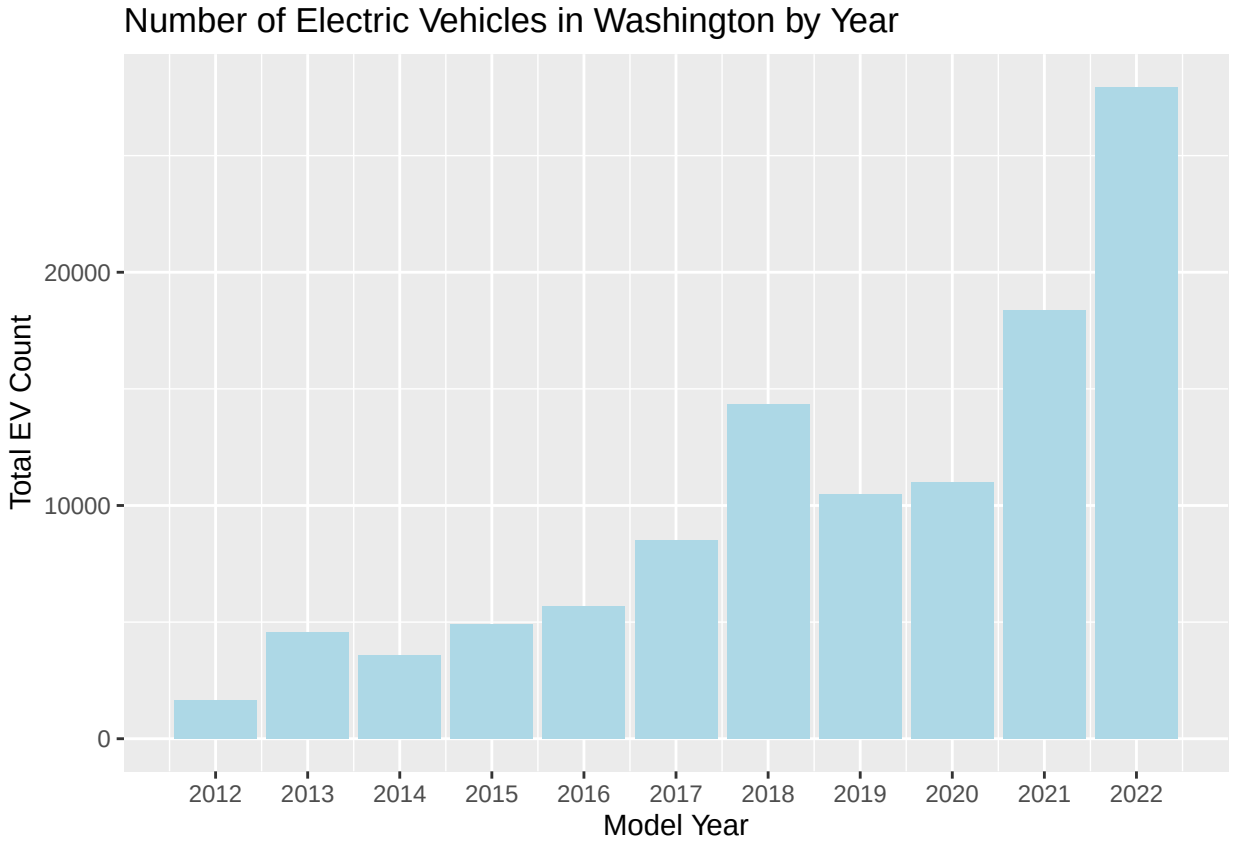
EV Data

```
ev_pop_first <- read_delim("Electric_Vehicle_Population_Data.csv")
```

```
## Rows: 130443 Columns: 17
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (10): VIN (1-10), County, City, State, Make, Model, Electric Vehicle Typ...
```

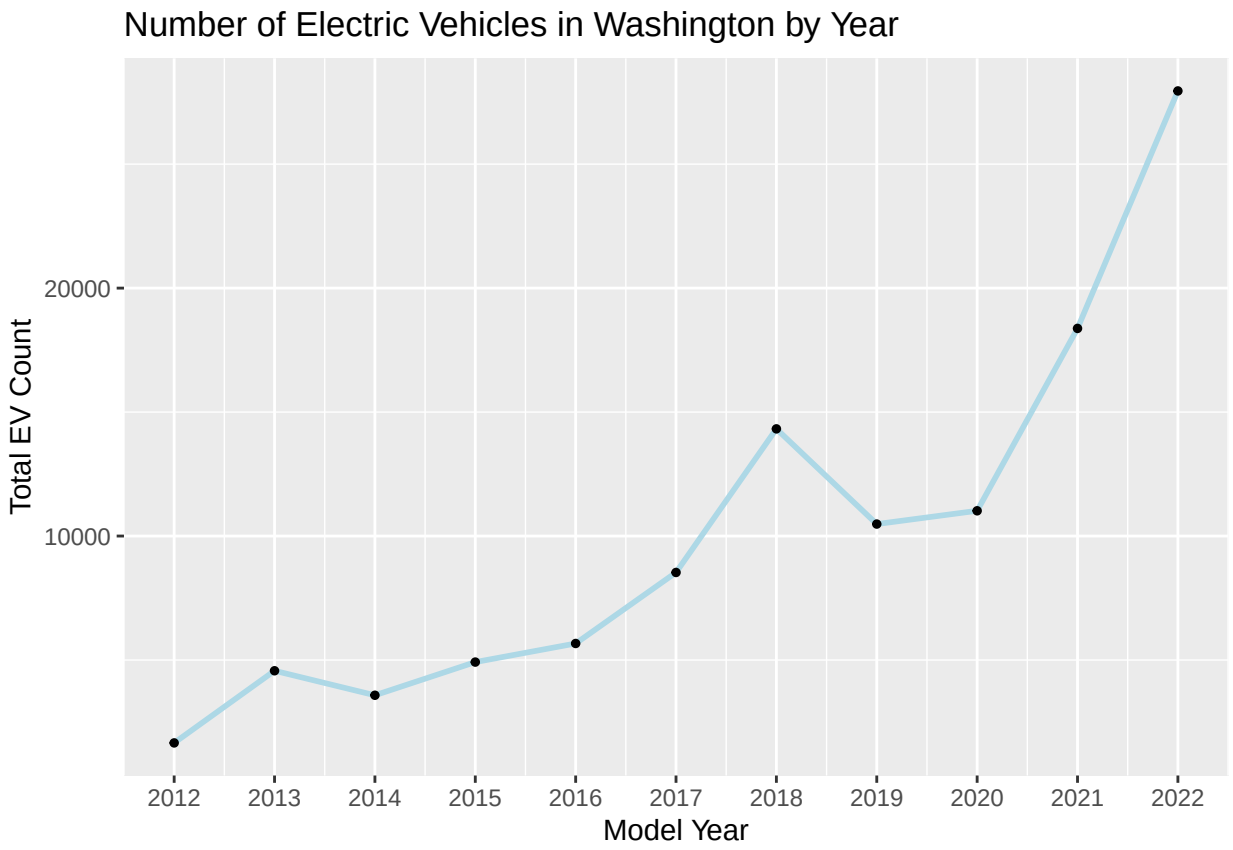
```
## dbl  (7): Postal Code, Model Year, Electric Range, Base MSRP, Legislative Di...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
ev_pop <- ev_pop_first %>%
  rename(Year = `Model Year`)

ev_pop <- ev_pop %>%
  filter(State == "WA", Year >= 2012 & Year <= 2022) %>%
  select(State, `Postal Code`, Year, Make, `Electric Vehicle Type`) %>%
  group_by(Year) %>%
  summarise(total_ev_count = n())

# plot
ev_pop_plot <- ev_pop %>%
  ggplot(aes(x = Year, y = total_ev_count)) +
  geom_bar(stat = "identity", fill = "lightblue") +
  labs(title = "Number of Electric Vehicles in Washington by Year",
       x = "Model Year",
       y = "Total EV Count") +
  scale_x_continuous(breaks = seq(2012, 2022, by = 1))
ev_pop_plot
```




```r
ev_pop_plot <- ev_pop %>%
  ggplot(aes(x = `Year`, y = total_ev_count)) +
  geom_line(color = "lightblue", size = 1) +
  geom_point(color = "black", size = 1) +
```

```
  labs(title = "Number of Electric Vehicles in Washington by Year",
       x = "Model Year",
       y = "Total EV Count") +
  scale_x_continuous(breaks = seq(2012, 2022, by = 1))

ev_pop_plot
```



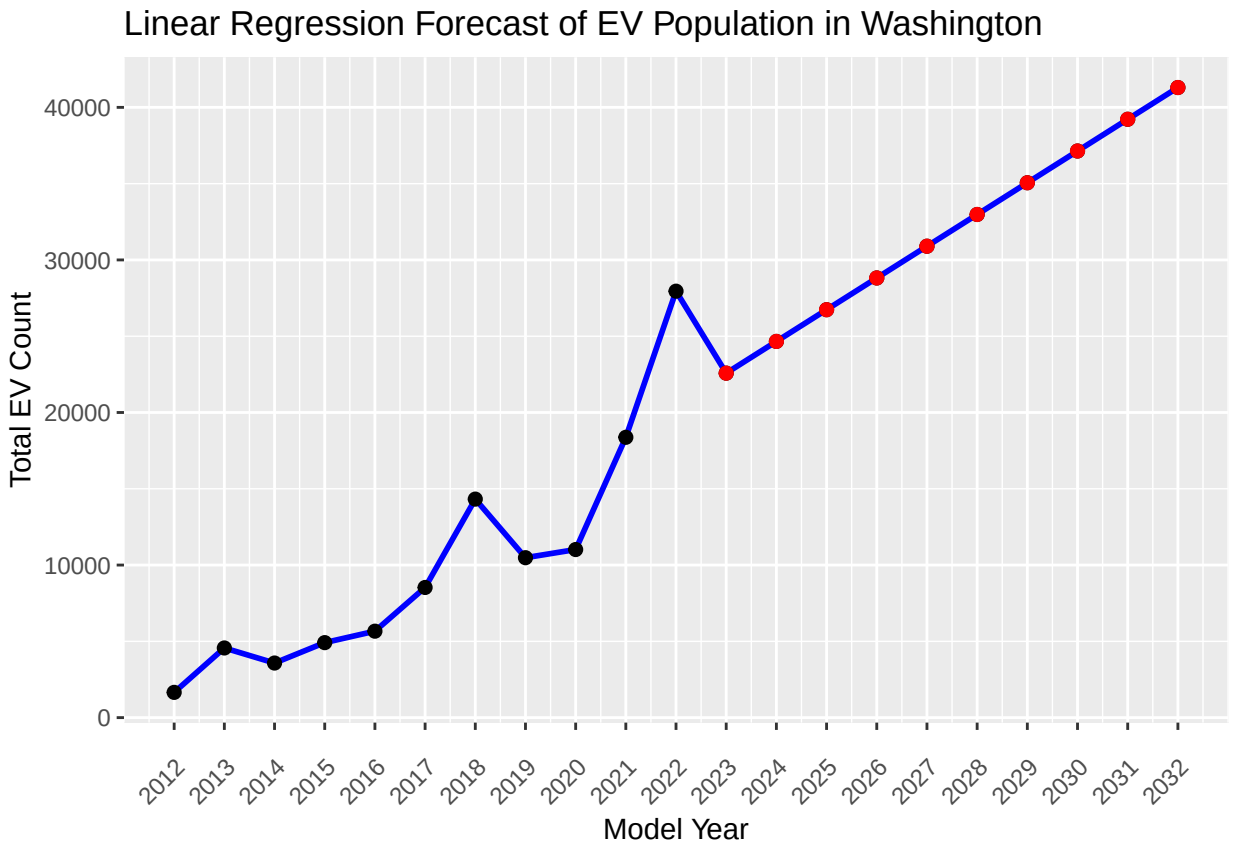
##Predictive Model

```
linear_model <- lm(total_ev_count ~ Year, data = ev_pop)

summary(linear_model)
```

```
##
## Call:
## lm(formula = total_ev_count ~ Year, data = ev_pop)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -5320  -1958   -278   2053   7455
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -4185776.7   712253.1  -5.877 0.000236 ***
## Year            2080.3      353.1   5.891 0.000232 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 3704 on 9 degrees of freedom
## Multiple R-squared:  0.7941, Adjusted R-squared:  0.7712
## F-statistic:  34.7 on 1 and 9 DF,  p-value: 0.0002316
```

```r
future_years <- data.frame(Year = 2023:2032)
linear_forecasts <- predict(linear_model, newdata = future_years)

predicted_data <- rbind(ev_pop, future_years %>%
                          mutate(total_ev_count = linear_forecasts))


linear_plot <- ggplot(predicted_data, aes(x = Year, y = total_ev_count)) +
  geom_line(color = "blue", size = 1) +
  geom_point(color = "black", size = 2) +
  geom_point(data = future_years %>% mutate(total_ev_count = linear_forecasts),
             aes(x = Year, y = total_ev_count), color = "red", size = 2) +
  labs(title = "Linear Regression Forecast of EV Population in Washington",
       x = "Model Year",
       y = "Total EV Count") +
  scale_x_continuous(breaks = seq(2012, 2032, by = 1)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1, vjust = 1,
                                   margin = margin(t = 10)))

linear_plot
```



##Combining both Data sets

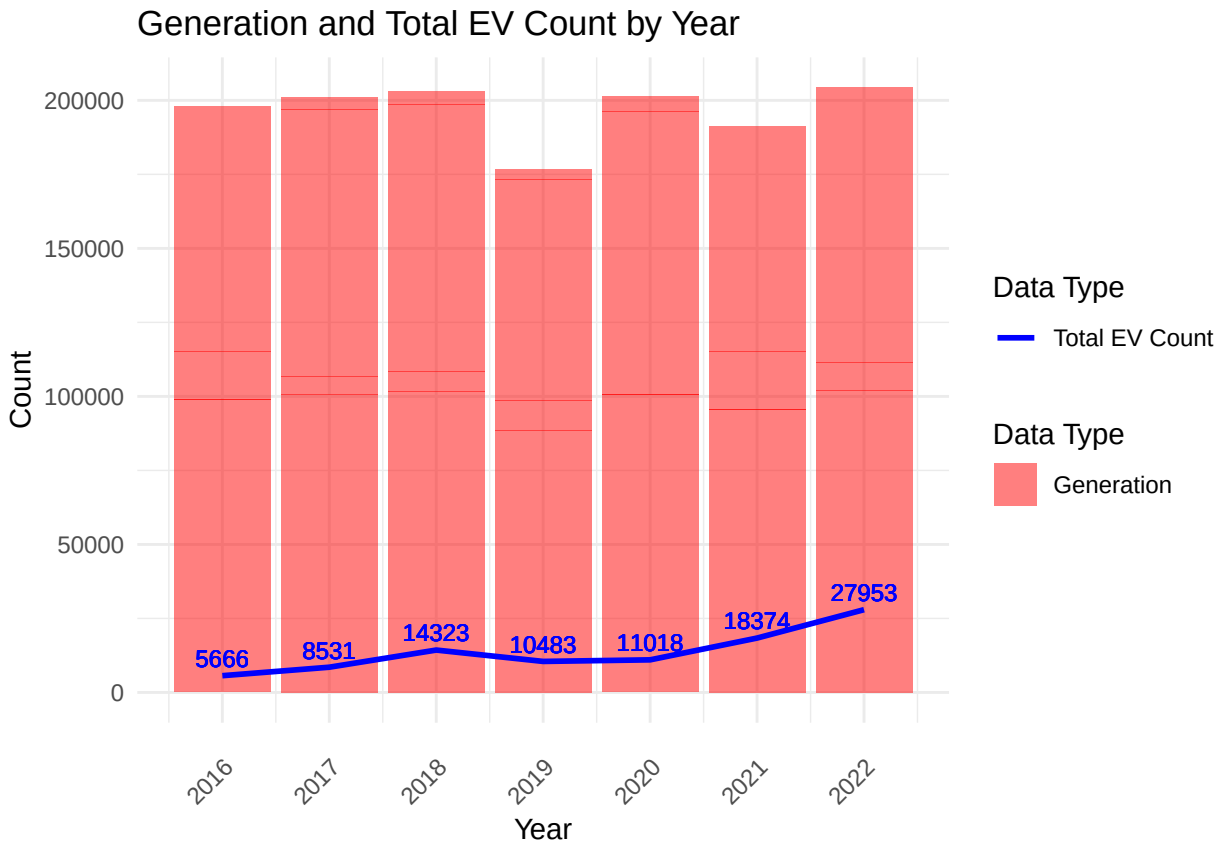
```r
# merge data sets
merged_data <- inner_join(generation_long, ev_pop, by = c("year" = "Year"))
head(merged_data)
```

```
## # A tibble: 6 x 6
##   state      type                       units      year generation total_ev_count
##   <chr>      <chr>                      <chr>     <dbl>      <dbl>          <int>
## 1 Washington all fuels (utility-scale) thousand~  2016      98948           5666
## 2 Washington all fuels (utility-scale) thousand~  2017     100482           8531
## 3 Washington all fuels (utility-scale) thousand~  2018     101638          14323
## 4 Washington all fuels (utility-scale) thousand~  2019      88467          10483
## 5 Washington all fuels (utility-scale) thousand~  2020     100653          11018
## 6 Washington all fuels (utility-scale) thousand~  2021      95617          18374
```

```r
merged_data_plot <- ggplot(merged_data, aes(x = year)) +
  geom_bar(aes(y = generation, fill = "Generation"), stat = "identity",
           alpha = 0.5) +
  geom_line(aes(y = total_ev_count, color = "Total EV Count"), size = 1) +
  geom_text(aes(y = total_ev_count, label = total_ev_count), vjust = -0.5,
            hjust = 0.5, color = "blue", size = 3) +
  scale_color_manual(values = "blue", guide = FALSE) +
  scale_fill_manual(values = "red", guide = FALSE) +
   scale_fill_manual(name = "Data Type", values = c("Generation" = "red"),
                     labels = c("Generation")) +
  scale_color_manual(name = "Data Type", values = c("Total EV Count" = "blue"),
                     labels = c("Total EV Count")) +
  labs(title = "Generation and Total EV Count by Year",
       x = "Year",
       y = "Count") +
  theme_minimal() +
  theme(legend.position = "right")+
  scale_x_continuous(breaks = seq(2015, 2022, by = 1)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1, vjust = 1,
                                   margin = margin(t = 10)))
```

```
## Scale for fill is already present.
## Adding another scale for fill, which will replace the existing scale.
## Scale for colour is already present.
## Adding another scale for colour, which will replace the existing scale.
```

```r
merged_data_plot
```



##Merging the Predictive Models

```
combined_data <- bind_rows(generation_long, forecast_data)
combined_data <- combined_data %>%
  filter(type == "all fuels (utility-scale)")

merged_data <- inner_join(combined_data, predicted_data,
                          by = c("year" = "Year"))

merged_data_plot <- ggplot(merged_data, aes(x = year)) +
  geom_bar(aes(y = generation, fill = "Generation"), stat = "identity",
           alpha = 0.5) +
  geom_line(aes(y = total_ev_count, color = "Total EV Count"), size = 1) +
  scale_color_manual(values = "blue", guide = FALSE) +
  scale_fill_manual(values = "red", guide = FALSE) +
  scale_fill_manual(name = "Data Type", values = c("Generation" = "red"),
                    labels = c("Generation")) +
  scale_color_manual(name = "Data Type", values = c("Total EV Count" = "blue"),
                     labels = c("Total EV Count")) +
  labs(title = "Predective Generation vs Total EV Count by Year",
       x = "Year",
       y = "Count") +
  theme_minimal() +
  theme(legend.position = "right") +
  scale_x_continuous(breaks = seq(2015, 2032, by = 1)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1, vjust = 1,
                                   margin = margin(t = 10)))
```

```
## Scale for fill is already present.
## Adding another scale for fill, which will replace the existing scale.
## Scale for colour is already present.
## Adding another scale for colour, which will replace the existing scale.
```

```
merged_data_plot
```