Homework1_HusikSargsyan

Husik Sargsyan

2024-10-18

#Humane Ai Pin

The choosen product for this homework is Humane Ai Pin (https://time.com/collection/best-inventions-2023/6327143/humane-ai-pin/). The Humane Ai Pin attaches to your clothing, it becomes your AI-powered personal assistant. Using a mix of proprietary software and OpenAI's GPT, the device lets you do everything from ask complex questions to make calls and send texts, all using just your voice.

#Similar Innovation

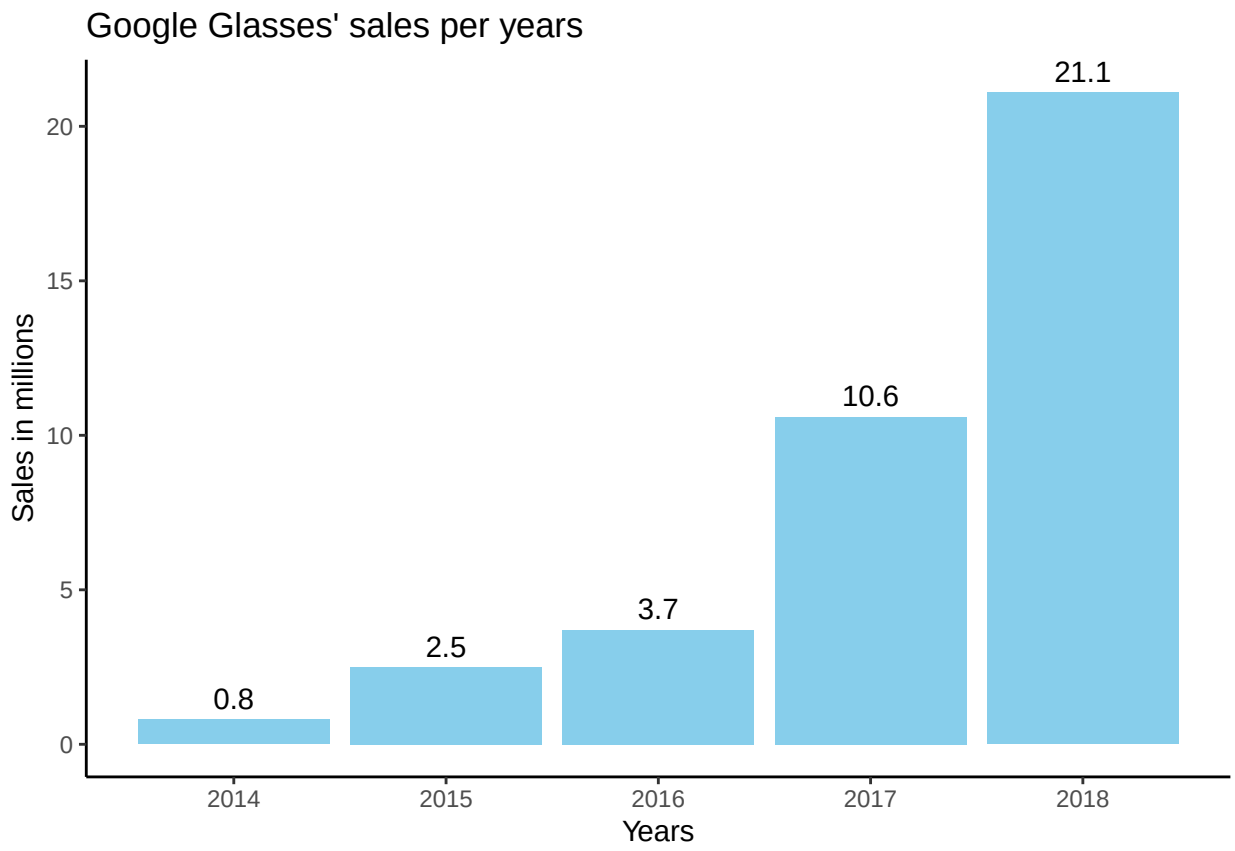
Similar to Humane AI Pin, Google Glass represented an early attempt to blend AI with clothes and accessories. Google Glass allows users to interact with specific apps and receive notifications or information directly through a small display on their glasses. The product suffered greatly, though, due to significant challenges, including privacy and the discomfort of its conspicuous style and ineptitude. Yet, this was exactly what Google Glass was designed to achieve: Aiming to demonstrate an artificial intelligence integration into the working pace of the user's life, even in the face of such evident limitations. On the other hand, Humane AI Pin took into consideration all the disadvantages that Google Glass experiments had. The innovation replaced the screen visibility and instead implemented direct voice, gesture, and sound interactions.

For the homework I will be using the following data, which represents the annual sales in millions for Google Glass. Reference Link: https://www.statista.com/chart/1143/forecast-of-annual-google-glass-sales/

```
library(ggplot2)

data <- data.frame(years = c(2014, 2015, 2016, 2017, 2018),
                   sales = c(0.8, 2.5, 3.7, 10.6, 21.1))

ggplot(data = data, aes(x = years, y = sales)) +
  geom_bar(stat = 'identity', fill = 'skyblue') +
  geom_text(aes(label = sales), vjust = -0.5) +
  ggtitle("Google Glasses' sales per years") +
  xlab("Years") +
  ylab("Sales in millions") +
  theme_classic()
```




```
sales <- c(0.8, 2.5, 3.7, 10.6, 21.1)
t = 1:length(sales)
start_values = list(m = sum(sales), p = 0.01, q = 0.3)

bass_model = nls(sales ~ m*(((p+q)^2/p)*exp(-(p+q)*t)) /
                   (1 + (q/p)*exp(-(p+q)*t))^2,
                 start = start_values,
                 algorithm = "port",
                 control = nls.control(maxiter = 200, warnOnly = TRUE))
summary(bass_model)
```

```
##
## Formula: sales ~ m * (((p + q)^2/p) * exp(-(p + q) * t))/(1 + (q/p) *
##     exp(-(p + q) * t))^2
##
## Parameters:
##    Estimate Std. Error t value Pr(>|t|)
## m 1.523e+02  1.403e+02   1.086   0.3911
## p 1.821e-03  7.542e-04   2.414   0.1372
## q 9.426e-01  2.023e-01   4.660   0.0431 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7459 on 2 degrees of freedom
##
## Algorithm "port", convergence message: relative convergence (4)
```

```r
p = coef(bass_model)["p"]
q = coef(bass_model)["q"]
m = coef(bass_model)["m"]

cat("Estimated p (innovation):", round(p, 4), "\n")
```

```
## Estimated p (innovation): 0.0018
```

```r
cat("Estimated q (imitation):", round(q, 4), "\n")
```

```
## Estimated q (imitation): 0.9426
```

```r
cat("Estimated M (market potential):", round(m, 4), "\n")
```

```
## Estimated M (market potential): 152.3434
```

```r
p <- 0.0018
q <- 0.9426
m <- 152.3434

bass_prediction <- function(t, m, p, q) {
  m * (((p + q)^2 / p) * exp(-(p + q) * t)) / (1 + (q / p) * exp(-(p + q) * t))^2
}

first_year <- 2024

num_years <- 5
predicted_adopters <- sapply(first_year:(first_year + num_years - 1),
                             function(year) bass_prediction(year - first_year, m, p, q))
print(predicted_adopters)
```

```
## [1]  0.2742181  0.7008846  1.7748070  4.3898269 10.2496596
```

Now let's take the market potential for Google glasses equal to 250, to see the difference it would make.

```r
p <- 0.0018
q <- 0.9426
m <- 250

bass_prediction <- function(t, m, p, q) {
  m * (((p + q)^2 / p) * exp(-(p + q) * t)) / (1 + (q / p) * exp(-(p + q) * t))^2
}

first_year <- 2024

num_years <- 5
predicted_adopters <- sapply(first_year:(first_year + num_years - 1),
                             function(year) bass_prediction(year - first_year, m, p, q))
print(predicted_adopters)
```

```
## [1]  0.450000  1.150172  2.912511  7.203835 16.819993
```

As we can see the results are almost twice higher in this case.