4.1 Linear Regression

Regression

Some Examples:

- Predicting future stock price
- Predicting if someones going to drop out of college or not
- Estimating the income price for banks

What are Linear Models?

An approximation method that we can use to describe behavior within data and predict its attributes in a timeframe.

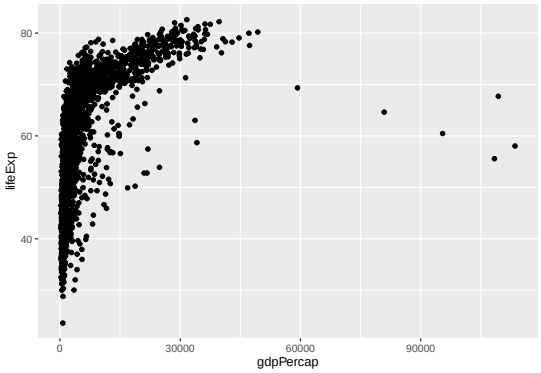
```
library(ggplot2)
```

```
## Warning: replacing previous import 'lifecycle::last_warnings' by
## 'rlang::last_warnings' when loading 'tibble'
```

```
## Warning: replacing previous import 'lifecycle::last_warnings' by
## 'rlang::last_warnings' when loading 'pillar'
```

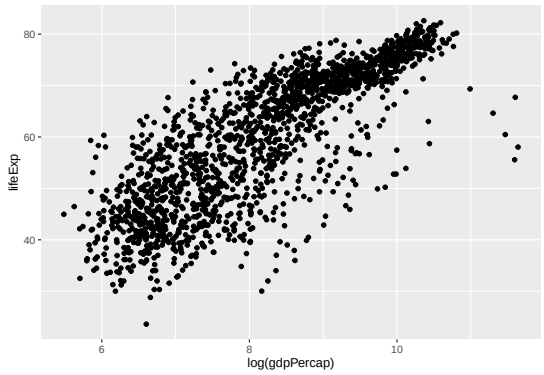
```
library(gapminder)
# Model fitting in R, the Basics:

ggplot(gapminder, aes(x = gdpPercap, y = lifeExp )) +
  geom_point()
```



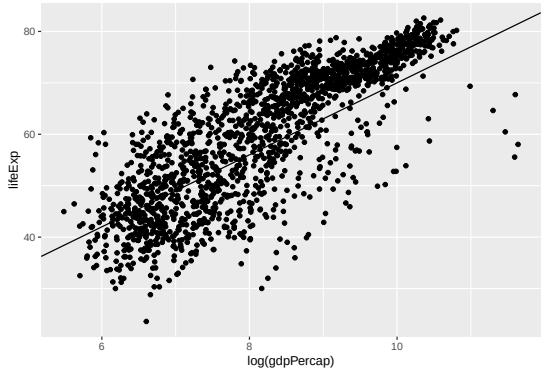
Note: Given that a column has large numbers, we can use logarithm to reduce the values. And make the data easier to understand.

```
# Make the same chart, but look at the log of gdp per capita instead
ggplot(gapminder, aes(log(gdpPercap), lifeExp ) ) +
  geom_point()
```




```
# Use geom_abline to guess the relationship
ggplot(gapminder, aes(log(gdpPercap), lifeExp ) ) +
  geom_point() +
  geom_abline(incercept = 0, slope = 7)
```

```
## Warning: Ignoring unknown parameters: incercept
```



Manually creating a model with 1 x variable

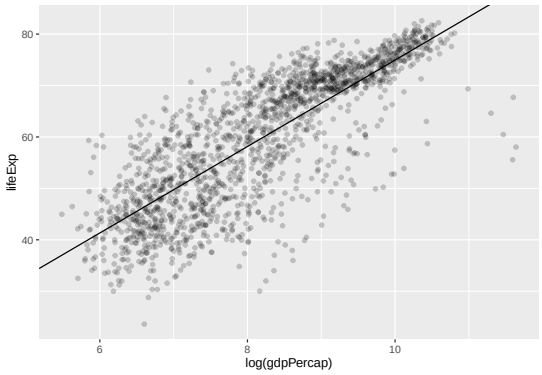
```
y <- gapminder$lifeExp
x <- log(gapminder$gdpPercap)


x_bar <- mean(x)
y_bar <- mean(y)

x_diff <- x - x_bar
y_diff <- y - y_bar

beta_1 <- sum(y_diff * x_diff/sum(x_diff^2))
beta_0 <- y_bar - (beta_1 * x_bar)

ggplot(gapminder, aes(x=log(gdpPercap), y=lifeExp )) +
  geom_point(alpha = .2) +
  geom_abline(intercept = beta_0, slope = beta_1 )
```



Model fitting in R syntax

"Formula syntax" takes the form: lm(y_var ~ x_var, data = dataset)

The formula syntax is more commonly used and we will focus on that today

Predicting gdpPerCap

In order to illustrate linear models, models with different features are ran.

a. lifeExp is used solely to get the gdpPercap.
b. pop is used solely to get the gdpPercap.
c. [year, pop, gdpPercap] are used to get the gdpPercap.

```
# a. x = lifeExp
model1 <- lm( gdpPercap ~ lifeExp, data = gapminder)

# b. x = pop
model2 <- lm( gdpPercap ~ pop, data = gapminder)

# b. x = [year, pop, gdpPercap]
model3 <- lm(gdpPercap ~ pop + year + lifeExp, data = gapminder)
```

Making predictions

In order to make predictions, we use the predict function.

```
predictions <- predict(model1, gapminder)
```

Validating models

In order to decide which features should be used, we run the loss functions to see which model has a high accuracy and lower error.

```
# Getting the predictions
model1_pred <- predict(model1, gapminder)
model2_pred <- predict(model2, gapminder)
model3_pred <- predict(model3, gapminder)

# Loss functions

# MAE
mae <- function(y_true, y_pred) { sum(abs(y_true - y_pred)) / NROW(y_true) }
```

```r
# MSE
mse <- function(y_true, y_pred) { sum((y_true - y_pred) ** 2 / NROW(y_true)) }

# RMSE
rmse <- function(y_true, y_pred) { sqrt(sum((y_true - y_pred) ** 2 / NROW(y_true)) ) }

error <- c("MAE", "RMSE", "MSE")
error_model1 <- c(mae(model1_pred, gapminder$gdpPercap), rmse(model1_pred, gapminder$gdpPercap), mse(mo
error_model2 <- c(mae(model2_pred, gapminder$gdpPercap), rmse(model2_pred, gapminder$gdpPercap), mse(mo
error_model3 <- c(mae(model3_pred, gapminder$gdpPercap), rmse(model3_pred, gapminder$gdpPercap), mse(mo

errors <- data.frame(error, error_model1, error_model2, error_model3)
head(errors)
```

```
##   error error_model1 error_model2 error_model3
## 1   MAE     4801.936     6535.112     4763.591
## 2  RMSE     8001.558     9851.332     7972.770
## 3   MSE 64024938.216 97048744.296 63565055.580
```