# 第 4 章: 予測
# (4.1. 選挙結果の予測)

今井耕介 著

『社会科学のためのデータ分析入門 (QSS)』

2026-03-09

# 4.1 選挙結果の予測

# 予測とは

- **予測**: 既存のデータ（訓練データ）からパターンを学び、未知のデータや未来の結果を推測すること。
- 社会科学の例：
  - 世論調査から選挙結果を予測する。
  - 経済指標から景気後退を予測する。
- 本節では、R のプログラミングの基礎（ループと条件分岐）を学びながら、2008 年米大統領選の予測に取り組みます。

# 4.1.1 Rでのループ (Loops)

# for ループの基本: 準備

- 同じ操作を繰り返し実行したい時に使用します。

```
values <- c(2, 4, 6) # 計算対象のベクトルを作成
n <- length(values)    # ベクトルの長さを取得
results <- rep(NA, n) # 結果保存用の空ベクトルを NA で初期化
```

# for ループの基本: 実行

```r
# i が 1 から n まで順に変わるループ
for (i in 1:n) {
    results[i] <- values[i] * 2 # values の i 番目を 2 倍して results の i 番目に代入
    cat(values[i], "times 2 is equal to", results[i], "\n") # 計算過程を表示
}
```

```
## 2 times 2 is equal to 4
## 4 times 2 is equal to 8
## 6 times 2 is equal to 12
```

```r
results # 最終的な結果を表示
```

```
## [1]  4  8 12
```

# 4.1.2 Rでの条件分岐 (Conditional Statements)

# if 文の基本: 準備

- 条件によって実行するコードを切り替えます。

```
values <- 1:5                          # 1 から 5 までの連続数を作成
results <- rep(NA, length(values)) # 結果保存用の空ベクトルを作成
```

# if 文の基本: 実行

```r
for (i in 1:length(values)) {
    x <- values[i] # 現在の要素を取り出す
    if (x %% 2 == 0) { # 2で割った余りが0（偶数）の場合
        cat(x, "is even. Adding itself.\n") # 偶数であることを表示
        results[i] <- x + x                 # 自分自身を足す
    } else {            # 奇数の場合
        cat(x, "is odd. Multiplying itself.\n") # 奇数であることを表示
        results[i] <- x * x                     # 自分自身を掛ける
    }
}
```

```
## 1 is odd. Multiplying itself.
## 2 is even. Adding itself.
## 3 is odd. Multiplying itself.
## 4 is even. Adding itself.
## 5 is odd. Multiplying itself.
```

```r
results # 結果を表示
```

```
## [1]  1  4  9  8 25
```

# 4.1.3 世論調査による予測

# 選挙データの読み込み (1)

- 2008 年大統領選の確定結果 (pres08.csv) と世論調査 (polls08.csv) を読み込みます。

```
pres08 <- read.csv("pres08.csv")    # 選挙結果データを読み込む
polls08 <- read.csv("polls08.csv") # 世論調査データを読み込む

#（参考）URL から直接読み込むことも可能
# pres08 <- read.csv("https://ayumu-tanaka.github.io/QSS/QSS_Data/pres08.csv")
# polls08 <- read.csv("https://ayumu-tanaka.github.io/QSS/QSS_Data/polls08.csv")
```

# データの準備 (2)

- 得票率のマージン計算と、日付データの処理を行います。

```
polls08$margin <- polls08$Obama - polls08$McCain # 世論調査の支持率差を計算
pres08$margin <- pres08$Obama - pres08$McCain    # 実際の選挙結果の得票率差を計算
polls08$middate <- as.Date(polls08$middate)        # 文字列を日付型に変換
# 選挙当日（2008-11-04）までの日数を計算
polls08$DaysToElection <- as.Date("2008-11-04") - polls08$middate
```

# 州ごとの最新調査に基づく予測: 準備

- 50 州＋ DC ごとに、選挙に最も近い日の調査結果を平均して予測値とします。

```
st.names <- unique(polls08$state)      # 州名のリスト（重複なし）を作成
poll.pred <- rep(NA, 51)               # 51 州分の予測値を保存するベクトル
names(poll.pred) <- as.character(st.names) # ベクトルの各要素に州名をつける
```

# 州ごとの最新調査に基づく予測: ループ処理

```
for (i in 1:51){
    # i 番目の州のデータのみを抽出
    state.data <- subset(polls08, subset = (state == st.names[i]))
    # その州の中で選挙日に最も近い（DaysToElection が最小）データを抽出
    latest <- subset(state.data, DaysToElection == min(DaysToElection))
    # 最新調査の支持率差の平均を予測値として保存
    poll.pred[i] <- mean(latest$margin)
}
head(poll.pred) # 予測値の先頭部分を表示
```

```
##    AL    AK    AZ    AR    CA    CO
## -25.0 -19.0  -2.5  -7.0  24.0   7.0
```

# 予測誤差の分析

- 実際の選挙結果と予測値の差を確認します。

```
errors <- pres08$margin - poll.pred # 実際の得票率差と予測値の差（誤差）を計算
names(errors) <- st.names           # 誤差ベクトルに州名を付与
mean(errors)                        # 平均予測誤差を表示
```

```
## [1] 1.062092
```

```
sqrt(mean(errors^2))                # RMSE（二乗平均平方根誤差）を表示
```

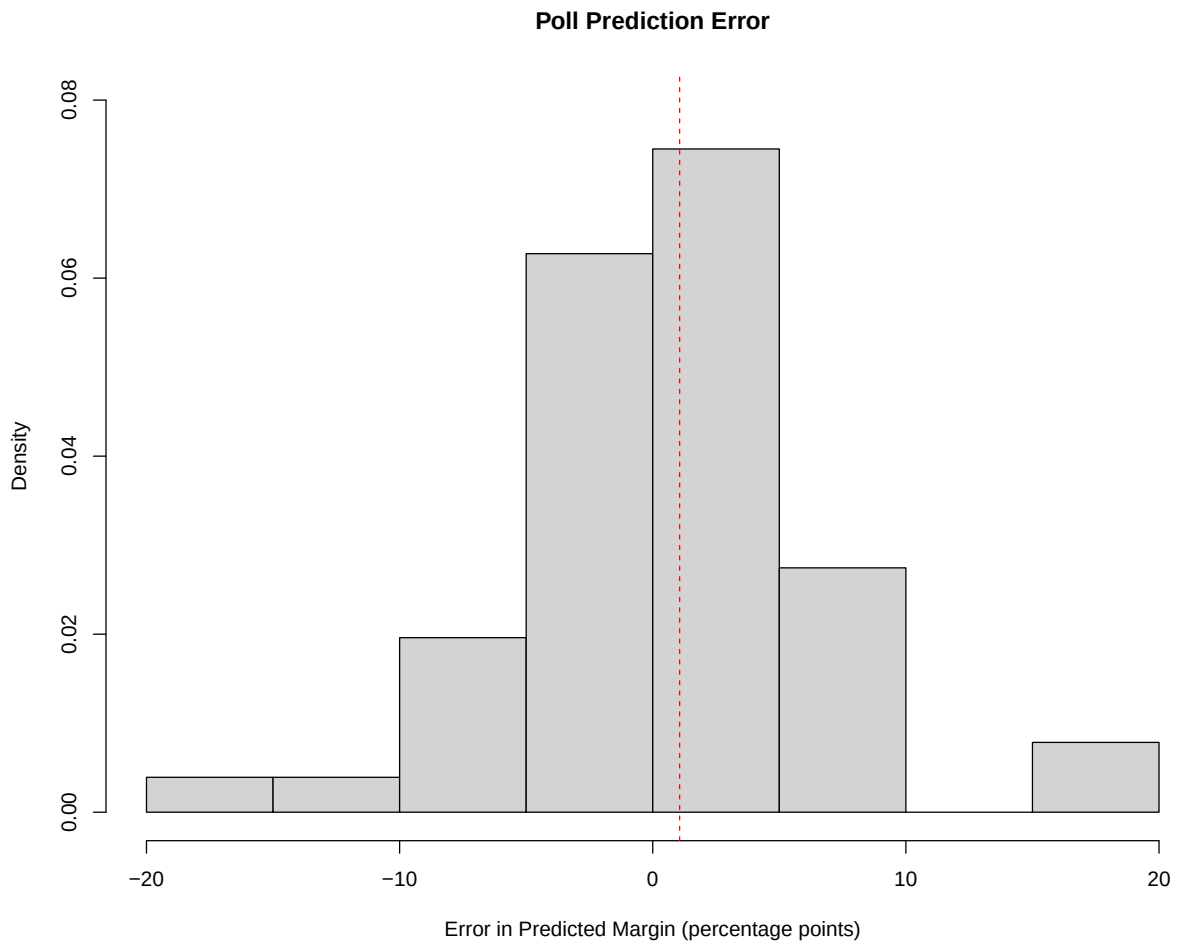
```
## [1] 5.90894
```

# 予測誤差の分布: コード

```r
# 予測誤差のヒストグラムを描画
hist(errors, freq = FALSE, ylim = c(0, 0.08),
     main = "Poll Prediction Error",
     xlab = "Error in Predicted Margin (percentage points)")

# 平均誤差の位置に垂直な赤い点線を追加
abline(v = mean(errors), lty = "dashed", col = "red")
```

# 予測誤差の分布: 描画結果



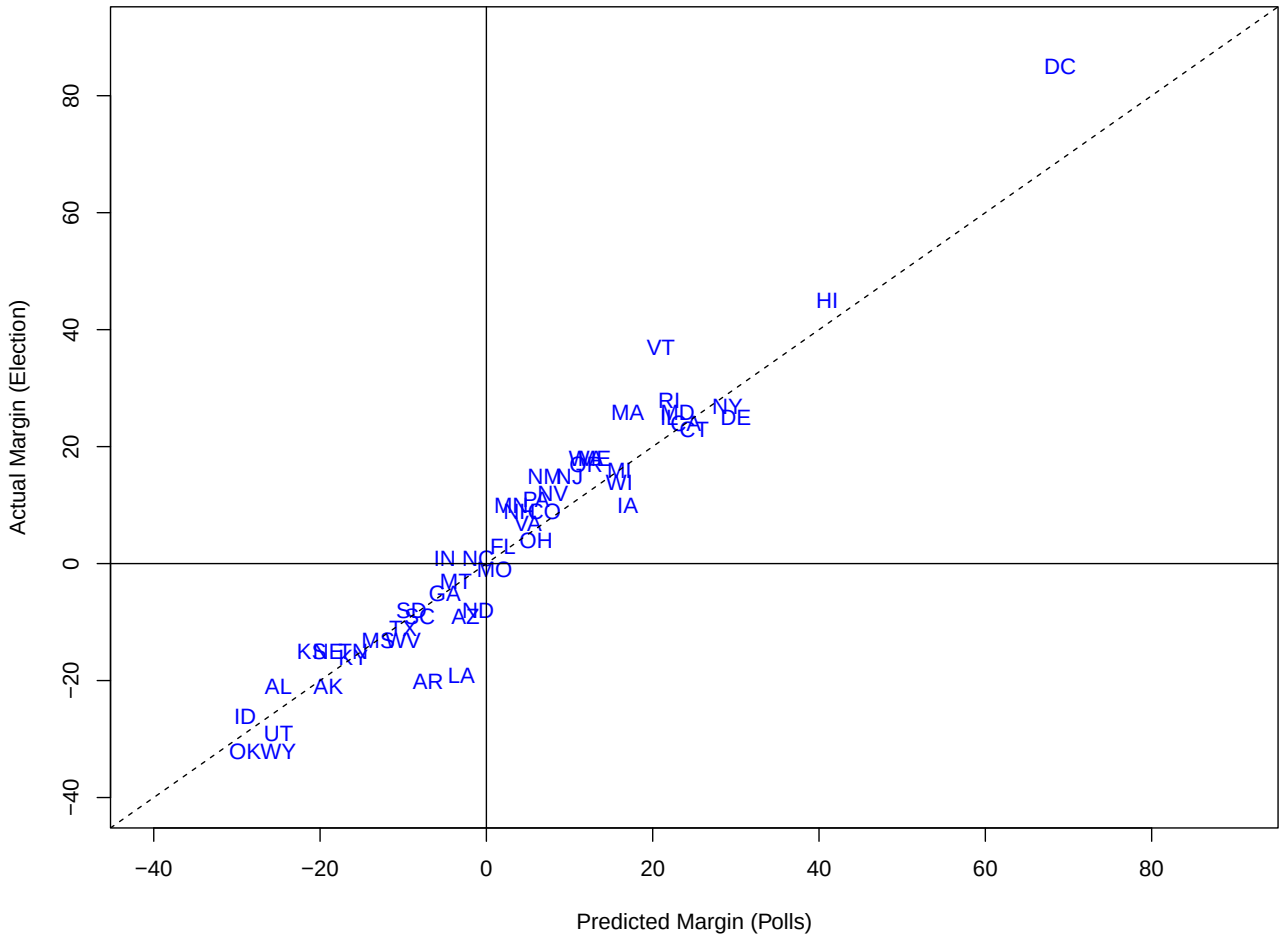
# 予測と実際の結果の比較: コード

```
# 空のプロット軸を作成
plot(poll.pred, pres08$margin, type = "n",
     xlim = c(-40, 90), ylim = c(-40, 90),
     xlab = "Predicted Margin (Polls)", ylab = "Actual Margin (Election)")

# 各州の値を略称（州名）でプロット
text(x = poll.pred, y = pres08$margin, labels = pres08$state, col = "blue")

# 予測と実績が一致する 45 度線を追加
abline(a = 0, b = 1, lty = "dashed")
abline(v = 0) # 垂直線（0）を追加
abline(h = 0) # 水平線（0）を追加
```

# 予測と実際の結果の比較: 描画結果



# 4.1.4 まとめ

# このセクションのまとめ (1)

- **予測のプロセス**: データのクリーニング（日付処理など）、サブセットの抽出、要約統計量の計算という一連の流れ。
- **プログラミングの道具**:
  - `for` ループ: 大量（今回は 51 州分）の繰り返し処理を自動化する。
  - `if` 文: 条件によって処理を分岐させる。

# このセクションのまとめ (2)

- **評価**: 実際の値と比較して、誤差（平均誤差や RMSE）を算出することで予測精度を客観的に評価する。
- **R の操作**: `as.Date()`, `for`, `if`, `unique()`, `min()`, `mean()`, `text()`。