Multi-Armed Bandit Implementations

Brendan Callender

```r
# This function gets the result from pulling the arm for bandit == bandit_number
# Returns 1 if success and 0 otherwise
# bandits = probabilities of success for each bandit in vector
# bandit_number = bandit in which to pull arm for
pull_arm <- function(bandits, bandit_number) {
  p <- bandits[[bandit_number]]
  rbinom(1, 1, p)
}

# This function determines which arm to pull for epsilon-greedy method or takes random actio
# eps = probability in which function will pick random bandit
# avg_rewards = Average reward winnings for each bandit in a vector
take_epsilon_greedy_action <- function(eps, avg_rewards) {
  res <- runif(1)
  if_else(res < eps, sample(1:length(avg_rewards), 1), which.max(avg_rewards))
}

# This function returns the sample thetas from each bandit distribution based on current esti
# Returns sampled thetas in a vector
sample_thetas <- function(alphas, betas) {
  res <- numeric(length(alphas))
  for (k in 1:5) {
    alpha_k <- alphas[[k]]
    beta_k <- betas[[k]]

    res[[k]] <- rbeta(1, alpha_k, beta_k)
  }
  res
}

# This function determines which arm to pull for the Thompson Sampling alorithm
# Returns the larges theta from the sample_thetas function
```

```r
take_TS_action <- function(thetas) {
  which.max(thetas)
}
```

ϵ-Greedy Method

```r
run_eps_greedy <- function(bandits, eps, iter) {

  total_rewards <- rep(0,5)
  total_attempts <- rep(0,5)
  avg_rewards <- rep(0.0,5)

  for (i in 1:iter) {

    action <- take_epsilon_greedy_action(eps, avg_rewards)
    reward <- pull_arm(bandits, action)

    # Store restult
    total_rewards[[action]] <- total_rewards[[action]] + reward
    total_attempts[[action]] <- total_attempts[[action]] + 1
    avg_rewards[[action]] <- total_rewards[[action]] / total_attempts[[action]]

  }
  list(
    best_bandit = which.max(avg_rewards),
    total_rewards = total_rewards,
    total_attempts = total_attempts,
    avg_rewards = avg_rewards,
    total_reward = sum(total_rewards)

  )

}
```

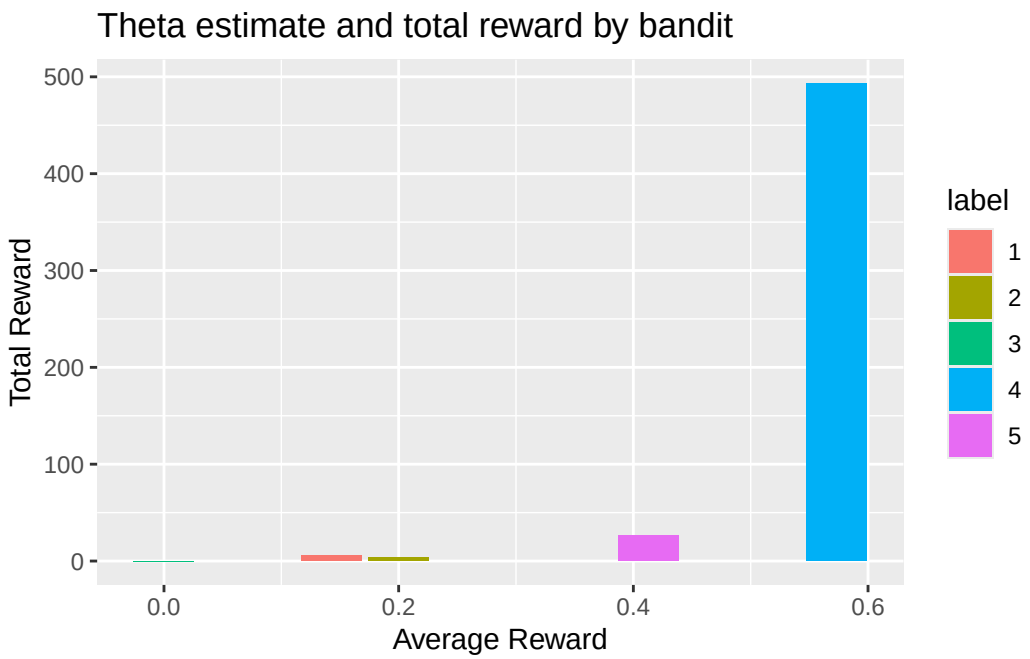
```r
p1 <- 0.1
p2 <- 0.3
p3 <- 0.05
p4 <- 0.55
p5 <- 0.4
```

```r
bandits <- c(p1,p2,p3,p4,p5)
iter <- 1000
eps <- 0.1

run_eps_greedy(bandits, eps, iter)
```

```
$best_bandit
[1] 4

$total_rewards
[1]   7   3   0 456   6

$total_attempts
[1]  97  20  22 844  17

$avg_rewards
[1] 0.07216495 0.15000000 0.00000000 0.54028436 0.35294118

$total_reward
[1] 472
```

```r
plot_eps_greed <- function(eg_obj) {
  tibble(x = eg_obj$avg_rewards, y = eg_obj$total_rewards, bandit = 1:5) %>%
    mutate(label = factor(bandit)) %>%
    ggplot(aes(x = x, y = y, fill = label)) +
    geom_bar(stat = "identity") +
    labs(title = "Theta estimate and total reward by bandit",
         y = "Total Reward", x = "Average Reward")

}
```

```r
eg_obj <- run_eps_greedy(bandits, eps, iter)
plot_eps_greed(eg_obj)
```



Thompson Sampling Method

```
run_ts <- function(bandits, iter) {

  alphas <- rep(1, 5)
  betas <- rep(1, 5)
  total_rewards <- 0

  for (i in 1:iter) {

    # Sample Model
    thetas <- sample_thetas(alphas, betas)

    # Select and apply action
    action <- take_TS_action(thetas)
    reward <- pull_arm(bandits, action)

    # update distribution
    alphas[[action]] <- alphas[[action]] + reward
    betas[[action]] <- betas[[action]] + 1 - reward
    total_rewards <- total_rewards + reward
```

```
  }

  list(
    alphas = alphas,
    betas = betas,
    theta_estimates = alphas / (alphas + betas),
    total_reward = sum(alphas)
  )

}
```

```
p1 <- 0.1
p2 <- 0.3
p3 <- 0.05
p4 <- 0.55
p5 <- 0.4

bandits <- c(p1,p2,p3,p4,p5)
iter <- 1000

run_ts(bandits, iter)
```

```
$alphas
[1]   2  10   1 393 116

$betas
[1]  12  22   9 317 128

$theta_estimates
[1] 0.1428571 0.3125000 0.1000000 0.5535211 0.4754098

$total_reward
[1] 522
```

```
plot_ts <- function(ts_object) {

  x <- seq(0,1,0.001)
  df_res <- as_tibble(label = c(), x = c(), dens = c())

  for(k in 1:length(ts_object$alphas)) {
      a <- ts_object$alphas[[k]]
```

```
      b <- ts_object$betas[[k]]

      bandit <- rep(k, length(x))
      dens <- dbeta(x, a, b)
      res <- cbind(bandit, x, dens) %>% as_tibble()

      df_res <- rbind(df_res, res)
  }
  df_res %>%
    mutate(bandit = factor(bandit)) %>%
    ggplot(aes(x = x, y = dens, color = bandit)) +
    geom_line() +
    theme_bw() +
    labs(title = "Posterior Distributions for Bandit Thetas")

}
```

```
ts_obj <- run_ts(bandits, iter)
plot_ts(ts_obj)
```