Playoff Point Share Calculations Script

Adam Kiehl

7/8/20

This is a script that utilizes NHL Stanley Cup Playoff data scraped from Hockey Reference (https://www.hockey-reference.com) to calculate playoff-specific point shares for individual players and for teams. Point shares are a system of attributing a player's offensive, defensive, and goalie contributions to overall team points. The system was created by former Vice President of Sports Reference Justin Kubatko (https://www.hockey-reference.com/about/point_shares.html) and was based on Bill James's win shares system. To begin, team, skater, and goalie data is read into R. Note: Scraping script must be run to populate CSVs before this script will execute.

```
teams <- read_csv('./data/team_playoffs.csv')
skaters <- read_csv('./data/skaters_playoffs.csv')
goalies <- read_csv('./data/goalies_playoffs.csv')
```

Here, the positions LW, C, and RW are all replaced by F for later conditional grouping into forwards and defensemen.

```
for (i in 1:nrow(skaters)) {
  if (skaters$pos[i] %in% c('LW', 'C', 'RW')) {
    skaters$pos[i] = 'F'
  }
}
```

Defensive Point Shares (DPS)

The proportion of a team's total TOI played by a single skater.

```
toi.prop = c()
for (i in 1:nrow(skaters)) {
  toi.prop = c(
    toi.prop,
    skaters$toi[i] / sum(skaters$toi[which((skaters$year == skaters$year[i]) & (skaters$team == skaters$
  )
}
skaters <- skaters %>%
  mutate(toi.prop = toi.prop)
```

Marginal goals per point is calculated as the league average of goals per point for a season.

```
marginal.goals.per.point = c()
for (i in 1:nrow(skaters)) {
  league.goals = sum(teams$goals[which(teams$year == skaters$year[i])])
  league.points = sum((2 * (teams$wins[which(teams$year == skaters$year[i])] - teams$ot.wins[which(teams
                     + teams$ot.wins[which(teams$year == skaters$year[i])]
                     + teams$ot.losses[which(teams$year == skaters$year[i])])
  marginal.goals.per.point = c(
    marginal.goals.per.point,
    league.goals / league.points
  )
}
skaters <- skaters %>%
  mutate(marginal.goals.per.point = marginal.goals.per.point)
```

The proportion of a team's marginal goals against that is attributed to skaters. (fixed to use goalie minutes instead of skater time on ice on 7/6/20)

```
marginal.goals.against.prop = c()
for (i in 1:nrow(skaters)) {
  sa.min.team = sum(goalies$shots[which((goalies$year == skaters$year[i]) & (goalies$team == skaters$tea
    sum(goalies$mins[which((goalies$year == skaters$year[i]) & (goalies$team == skaters$team[i]))])
  sa.min.league = sum(goalies$shots[which(goalies$year == skaters$year[i])]) / sum(goalies$mins[which(go
  marginal.goals.against.prop = c(
    marginal.goals.against.prop,
    (7 - 2 * (sa.min.team / sa.min.league)) / 7
  )
}
skaters <- skaters %>%
  mutate(marginal.goals.against.prop = marginal.goals.against.prop)
```

A position adjustment defined by the proportion of defensive effort assigned to each position.

```
pos.adj = c()
for (i in 1:nrow(skaters)) {
  if (skaters$pos[i] == 'F') {
    pos.adj = c(
      pos.adj,
      5/7
    )
  }
  else if (skaters$pos[i] == 'D') {
    pos.adj = c(
      pos.adj,
      10/7
    )
  }
}
skaters <- skaters %>%
  mutate(pos.adj = pos.adj)
```

Team marginal goals against defines league goal performance against the team in question.

```
team.marginal.goals.against = c()
for (i in 1:nrow(skaters)) {
  team.games = teams$games[which((teams$team == skaters$team[i]) & (teams$year == skaters$year[i]))]
  league.goals.game = sum(teams$goals[which(teams$year == skaters$year[i])]) / sum(teams$games[which(tea
  team.goals.against = sum(goalies$goals[which((goalies$year == skaters$year[i]) & (goalies$team == ska
  team.marginal.goals.against = c(
    team.marginal.goals.against,
    (1 + (7 / 12)) * team.games * league.goals.game - team.goals.against
  )
}
skaters <- skaters %>%
  mutate(team.marginal.goals.against = team.marginal.goals.against)
```

Plus-minus adjustment defines skater's individual performance while on the ice.

```
plus.minus.adj = c()
for (i in 1:nrow(skaters)) {
  team.plus.minus = sum(skaters$plus.minus[which((skaters$year == skaters$year[i]) & (skaters$team == sk
  team.toi = sum(skaters$toi[which((skaters$year == skaters$year[i]) & (skaters$team == skaters$team[i]
  plus.minus.adj = c(
    plus.minus.adj,
    (1 / 7) * skaters$pos.adj[i] * (skaters$plus.minus[i] - skaters$toi[i] * (team.plus.minus / team.to
  )
}
skaters <- skaters %>%
  mutate(plus.minus.adj = plus.minus.adj)
```

Marginal goals against assigns a proportion of a team's marginal goals against to an individual skater based on his proportion of TOI, position adjustment, and plus-minus adjustment. (fixed to add plus-minus adjustment rather than multiply is on 7/6/20)

```
marginal.goals.against = c()
for (i in 1:nrow(skaters)) {
  marginal.goals.against = c(
    marginal.goals.against,
    skaters$toi.prop[i] * skaters$marginal.goals.against.prop[i] * skaters$pos.adj[i] * skaters$team.ma
  )
}
skaters <- skaters %>%
  mutate(marginal.goals.against = marginal.goals.against)
```

Attributes defensive point shares by dividing marginal goals against by marginal goals per point.

```
dps = c()
for (i in 1:nrow(skaters)) {
  dps = c(
    dps,
    skaters$marginal.goals.against[i] / skaters$marginal.goals.per.point[i]
  )
}
skaters <- skaters %>%
  mutate(dps = dps)
```

Offensive Point Shares (OPS)

Goals created factors assists into a skater's goal contribution.

```
goals.created = c()
for (i in 1:nrow(skaters)) {
  team.goals = teams$goals[which((teams$year == skaters$year[i]) & (teams$team == skaters$team[i]))]
  team.assists = sum(skaters$assists[which((teams$year == skaters$year[i]) & (teams$team == skaters$tea
  goals.created = c(
    goals.created,
    (skaters$goals[i] + .5 * skaters$assists[i]) * (team.goals / (team.goals + .5 * team.assists))
  )
}
skaters <- skaters %>%
  mutate(goals.created = goals.created)
```

Marginal goals for defines a skaters's individual share of team marginal goals for.

```
marginal.goals.for = c()
for (i in 1:nrow(skaters)) {
  goals.created.league = sum(skaters$goals.created[which((skaters$pos == skaters$pos[i]) & (skaters$yea
  toi.league = sum(skaters$toi[which((skaters$pos == skaters$pos[i]) & (skaters$year == skaters$year[i]
  marginal.goals.for = c(
    marginal.goals.for,
    skaters$goals.created[i] - (7 / 12) * skaters$toi[i] * (goals.created.league / toi.league)
  )
}
skaters <- skaters %>%
  mutate(marginal.goals.for = marginal.goals.for)
```

Attributes offensive point shares by dividing marginal goals for by marginal goals per point.

```
ops = c()
for (i in 1:nrow(skaters)) {
  ops = c(
    ops,
    skaters$marginal.goals.for[i] / skaters$marginal.goals.per.point[i]
  )
}
skaters <- skaters %>%
  mutate(ops = ops)
```

Goalie Point Shares (GPS)

Adjusts a goalie's shots against to playoff year.

```
shots.against.adjustment = c()
for (i in 1:nrow(goalies)) {
  shots.per.min = goalies$shots[i] / goalies$mins[i]
  league.shots.per.min = sum(goalies$shots[which(goalies$year == goalies$year[i])]) / sum(goalies$mins[
  shots.against.adjustment = c(
    shots.against.adjustment,
    shots.per.min / league.shots.per.min
  )
}
goalies <- goalies %>%
  mutate(shots.against.adjustment = shots.against.adjustment)
```

Assigns a proportion of a team’s goals against to an individual goalie.

```
marginal.goals.against = c()
for (i in 1:nrow(goalies)) {
  league.goals.per.min = sum(goalies$goals[which(goalies$year == goalies$year[i])]) / sum(goalies$mins[w
  marginal.goals.against = c(
    marginal.goals.against,
    (1 + (7 / 12)) * goalies$shots.against.adjustment[i] * goalies$mins[i] * league.goals.per.min - goal
  )
}
goalies <- goalies %>%
  mutate(marginal.goals.against = marginal.goals.against)
```

Marginal goals per point is calculated as the league average of goals per point for a season.

```
marginal.goals.per.point = c()
for (i in 1:nrow(goalies)) {
  league.goals = sum(teams$goals[which(teams$year == goalies$year[i])])
  league.points = sum((2 * (teams$wins[which(teams$year == skaters$year[i])] - teams$ot.wins[which(teams
                     + teams$ot.wins[which(teams$year == skaters$year[i])]
                     + teams$ot.losses[which(teams$year == skaters$year[i])])
  marginal.goals.per.point = c(
    marginal.goals.per.point,
    league.goals / league.points
  )
}
goalies <- goalies %>%
  mutate(marginal.goals.per.point = marginal.goals.per.point)
```

Attributes goalie point shares by dividing marginal goals against by marginal goals per point.

```
gps = c()
for (i in 1:nrow(goalies)) {
  gps = c(
    gps,
    (2 / 7) * (goalies$marginal.goals.against[i] / goalies$marginal.goals.per.point[i])
  )
}
goalies <- goalies %>%
  mutate(gps = gps)
```

Team Point Shares

A team’s points is defined by the sum of twice its regulation wins plus its overtime wins and losses.

```
points = c()
for (i in 1:nrow(teams)) {
  points = c(
    points,
    2 * (teams$wins[i] - teams$ot.wins[i]) + teams$ot.wins[i] + teams$ot.losses[i]
  )
}
teams <- teams %>%
  mutate(points = points)
```

Team defensive point shares are calculated as the sum of individual defensive point shares.

```
team.dps = c()
for (i in 1:nrow(teams)) {
  team.dps = c(
    team.dps,
    sum(skaters$dps[which((skaters$year == teams$year[i]) & (skaters$team == teams$team[i]))])
  )
}
teams <- teams %>%
  mutate(team.dps = team.dps)
```

Team offensive point shares are calculated as the sum of individual offensive point shares.

```
team.ops = c()
for (i in 1:nrow(teams)) {
  team.ops = c(
    team.ops,
    sum(skaters$ops[which((skaters$year == teams$year[i]) & (skaters$team == teams$team[i]))])
  )
}
teams <- teams %>%
  mutate(team.ops = team.ops)
```

Team goalie point shares are calculated as the sum of individual goalie point shares.

```
team.gps = c()
for (i in 1:nrow(teams)) {
  team.gps = c(
    team.gps,
    sum(goalies$gps[which((goalies$year == teams$year[i]) & (goalies$team == teams$team[i]))])
  )
}
teams <- teams %>%
  mutate(team.gps = team.gps)
```

Average DPS per game is calculated for each team.

```
dps.avg = c()
for (i in 1:nrow(teams)) {
  dps.avg = c(
    dps.avg,
    teams$team.dps[i] / teams$games[i]
  )
}
teams <- teams %>%
  mutate(dps.avg = dps.avg)
```

Average OPS per game is calculated for each team.

```
ops.avg = c()
for (i in 1:nrow(teams)) {
  ops.avg = c(
    ops.avg,
    teams$team.ops[i] / teams$games[i]
  )
}
teams <- teams %>%
  mutate(ops.avg = ops.avg)
```

Average GPS per game is calculated for each team.

```
gps.avg = c()
for (i in 1:nrow(teams)) {
  gps.avg = c(
    gps.avg,
    teams$team.gps[i] / teams$games[i]
  )
}
teams <- teams %>%
  mutate(gps.avg = gps.avg)
```

Team point shares are calculated as the sum of defensive, offensive, and goalie point shares.

```
team.ps = c()
for (i in 1:nrow(teams)) {
  team.ps = c(
    team.ps,
    teams$team.dps[i] + teams$team.ops[i] + teams$team.gps[i]
  )
}
teams <- teams %>%
  mutate(team.ps = team.ps)
```

Defensive point shares proportion describes the proportion of team points attributed to defensive contributions yielding a measure of a team's defensive mindedness.

```
dps.prop = c()
for (i in 1:nrow(teams)) {
  dps.prop = c(
    dps.prop,
    teams$team.dps[i] / teams$team.ps[i]
  )
}
teams <- teams %>%
  mutate(dps.prop = dps.prop)
```

A final adjustment subtracts the league average yielding a measure of a teams defensive mindedness relative to other playoff teams in that season.

```
dps.prop.adj = c()
for (i in 1:nrow(teams)) {
  league.mean = mean(teams$dps.prop[which(teams$year == teams$year[i])])
  dps.prop.adj = c(
    dps.prop.adj,
    teams$dps.prop[i] - league.mean
  )
}
teams <- teams %>%
  mutate(dps.prop.adj = dps.prop.adj)
```

Offensive point shares proportion describes the proportion of team points attributed to offensive contributions yielding a measure of a team's offensive mindedness.

```
ops.prop = c()
for (i in 1:nrow(teams)) {
  ops.prop = c(
    ops.prop,
    (teams$team.ops[i] / teams$team.ps[i])
  )
}
teams <- teams %>%
  mutate(ops.prop = ops.prop)
```

A final adjustment subtracts the league average yielding a measure of a teams offensive mindedness relative to other playoff teams in that season.

```
ops.prop.adj = c()
for (i in 1:nrow(teams)) {
  league.mean = mean(teams$ops.prop[which(teams$year == teams$year[i])])
  ops.prop.adj = c(
    ops.prop.adj,
    teams$ops.prop[i] - league.mean
  )
}
teams <- teams %>%
  mutate(ops.prop.adj = ops.prop.adj)
```

Full defensive point shares proportion describes the proportion of team points attributed to defensive and goalie contributions yielding a measure of a team's defensive mindedness.

```
dps.full.prop = c()
for (i in 1:nrow(teams)) {
  dps.full.prop = c(
    dps.full.prop,
    (teams$team.dps[i] + teams$team.gps[i]) / teams$team.ps[i]
  )
}
teams <- teams %>%
  mutate(dps.full.prop = dps.full.prop)
```

A final adjustment subtracts the league average yielding a measure of a teams defensive mindedness relative to other playoff teams in that season.

```
dps.full.prop.adj = c()
for (i in 1:nrow(teams)) {
  league.mean = mean(teams$dps.full.prop[which(teams$year == teams$year[i])])
  dps.full.prop.adj = c(
    dps.full.prop.adj,
    teams$dps.full.prop[i] - league.mean
  )
}
teams <- teams %>%
  mutate(dps.full.prop.adj = dps.full.prop.adj)
```

Here, an indicator variable `champ` is created to indicate whether or not a team won the Stanley Cup Finals in that given year. Conveniently, the championship team is always listed last in each year from the scraping process.

```
champ <- c()
for (i in 1:nrow(teams)) {
  if (i == nrow(teams)) {
    champ <- c(champ, 1)
  }
  else if(teams$year[i] != teams$year[i + 1]) {
    champ <- c(champ, 1)
  }
  else {
    champ <- c(champ, 0)
  }
}
teams <- teams %>%
  mutate(champ = champ)
```

Selection

```
teams <- teams %>% select(.)
skaters <- skaters %>% select(year, team, player, pos, goals, assists, plus.minus, toi, dps, ops)
goalies <- goalies %>% select(year, team, player, goals, shots, mins, gps)
```

Writing to CSV

Tables are rewritten to CSV for later analysis.

```r
teams %>% write_csv('./data/team_playoffs.csv')
skaters %>% write_csv('./data/skaters_playoffs.csv')
goalies %>% write_csv('./data/goalies_playoffs.csv')
```