Exploratory Analysis of Pre-Match Features vs SLS-F+

James Njoroge, Muhammad Raka Zuhdi, Fola Oladipo

October 27, 2025

1 Objective

We have made features that should be safe from data leakage, whereby we are not using the data that contributes to the target (that we are trying to predict) as a feature for our model to learn from. We achieve this by framing this problem as a time-series problem whereby we only access past information as a feature to make the current prediction. (Data Leakage in ML.)

We want to understand how our forward-looking, leakage-safe, pre-match features (e.g. attacking form, defensive concessions, rest, crowd context, composite AttackVsDefense / TempoSum / SoTSum) relate to the post-match liveliness score `SLS_Fplus`. We use `match_features_wide.csv` as the modeling table (one row per match), and produce global + per-round visuals.

2 Load Data

We load the feature table produced by the Python pipeline. This table has one row per match (Home vs Away), all pre-match features, and the target `SLS_Fplus`.

```
wide_df <- read_csv("feature_tables/match_features_wide.csv",
                    show_col_types = FALSE)

# Basic sanity check
head(wide_df)

## # A tibble: 6 x 31
##   Round HomeTeam          AwayTeam    Home_days_rest Away_days_rest DaysRestDiff
##   <dbl> <chr>             <chr>                <dbl>          <dbl>        <dbl>
## 1     0 Manchester United Fulham                   7              7            0
## 2     0 Ipswich Town      Liverpool                7              7            0
## 3     0 Arsenal           Wolverhamp~              7              7            0
## 4     0 Everton           Brighton &~              7              7            0
## 5     0 Newcastle United  Southampton              7              7            0
## 6     0 Nottingham Forest AFC Bourne~              7              7            0
## # i 25 more variables: Home_occ_prior <dbl>, LeagueAvg_xG_perMatch_sofar <dbl>,
## #   LeagueAvg_Corners_perMatch_sofar <dbl>, HomeFlag <dbl>,
## #   Home_xG_att_90 <dbl>, Home_SoT_att_90 <dbl>, Home_BigCh_att_90 <dbl>,
## #   Home_Corn_att_90 <dbl>, Home_ToB_att_90 <dbl>, Home_xGA_def_90 <dbl>,
## #   Home_SoT_agst_90 <dbl>, Home_BigCh_agst_90 <dbl>, Away_xG_att_90 <dbl>,
## #   Away_SoT_att_90 <dbl>, Away_BigCh_att_90 <dbl>, Away_Corn_att_90 <dbl>,
## #   Away_ToB_att_90 <dbl>, Away_xGA_def_90 <dbl>, Away_SoT_agst_90 <dbl>, ...
```

```r
summary(select(wide_df, SLS_Fplus,
               Home_AttackVsDefense, Away_AttackVsDefense,
               TempoSum, SoTSum,
               Home_occ_prior,
               DaysRestDiff,
               LeagueAvg_xG_perMatch_sofar,
               LeagueAvg_Corners_perMatch_sofar))
```

```
##    SLS_Fplus      Home_AttackVsDefense Away_AttackVsDefense    TempoSum
##  Min.   : 19.07   Min.   :1.353        Min.   :1.319        Min.   : 4.547
##  1st Qu.: 39.55   1st Qu.:2.324        1st Qu.:2.316        1st Qu.: 8.147
##  Median : 48.40   Median :2.661        Median :2.655        Median : 9.853
##  Mean   : 49.96   Mean   :2.720        Mean   :2.738        Mean   : 9.805
##  3rd Qu.: 59.00   3rd Qu.:3.047        3rd Qu.:3.138        3rd Qu.:11.179
##  Max.   :100.00   Max.   :4.904        Max.   :4.697        Max.   :18.568
##      SoTSum       Home_occ_prior    DaysRestDiff
##  Min.   : 3.979   Min.   :0.8652   Min.   :-23.0000
##  1st Qu.: 7.247   1st Qu.:0.9545   1st Qu.: -1.0000
##  Median : 8.337   Median :0.9753   Median :  0.0000
##  Mean   : 8.576   Mean   :0.9720   Mean   :  0.1474
##  3rd Qu.: 9.853   3rd Qu.:0.9869   3rd Qu.:  1.0000
##  Max.   :15.158   Max.   :1.0935   Max.   : 23.0000
##  LeagueAvg_xG_perMatch_sofar LeagueAvg_Corners_perMatch_sofar
##  Min.   :2.316               Min.   :10.00
##  1st Qu.:2.903               1st Qu.:10.41
##  Median :2.951               Median :10.67
##  Mean   :2.929               Mean   :10.70
##  3rd Qu.:2.979               3rd Qu.:10.98
##  Max.   :3.026               Max.   :15.00
```

3 Pairwise Relationships

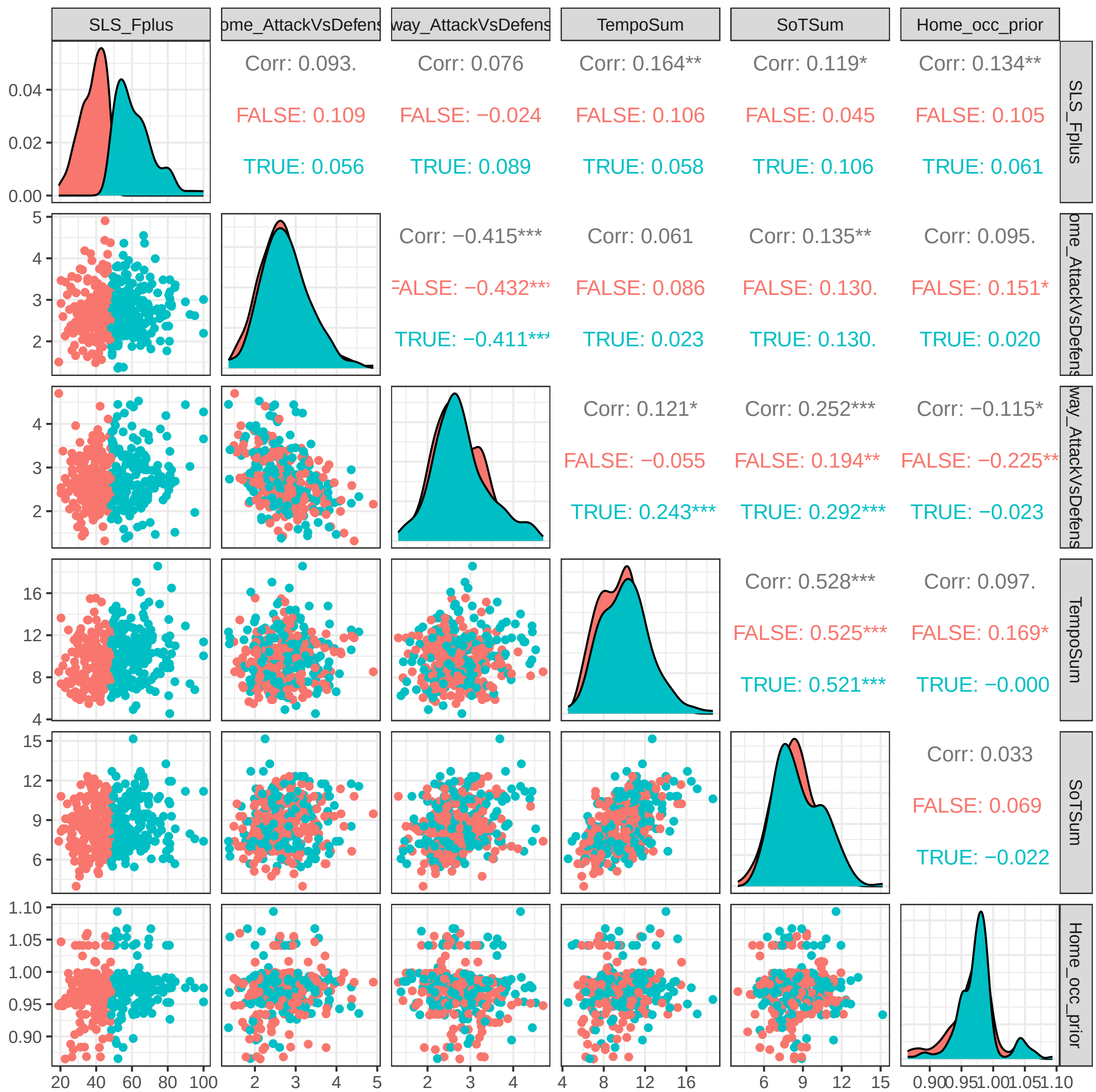
Below we inspect pairwise scatterplots between a subset of the most interpretably important numeric predictors and the target `SLS_Fplus`. This helps us see linear vs nonlinear trends, clustering, and outliers.

We pick:

- `Home_AttackVsDefense`, `Away_AttackVsDefense`
  (our core "can this get wild?" signal: attack strength of one side plus defensive weakness of the other);
- `TempoSum`, `SoTSum`
  (expected tempo and shot volume);
- `Home_occ_prior`
  (crowd intensity proxy);
- `SLS_Fplus`
  (the score we want to predict).

```r
pair_df <- wide_df |>
  select(SLS_Fplus,
         Home_AttackVsDefense, Away_AttackVsDefense,
         TempoSum, SoTSum,
         Home_occ_prior)

GGally::ggpairs(
  pair_df,
  columns = 1:ncol(pair_df),
  aes(color = SLS_Fplus > median(SLS_Fplus, na.rm=TRUE)),
  progress = FALSE
)
```



The diagonal panels show distributions for each variable. Off-diagonals are scatterplots. Color here just flags whether the match's liveliness (`SLS_Fplus`) is above the median, so we can visually see if "high SLS" games cluster anywhere.

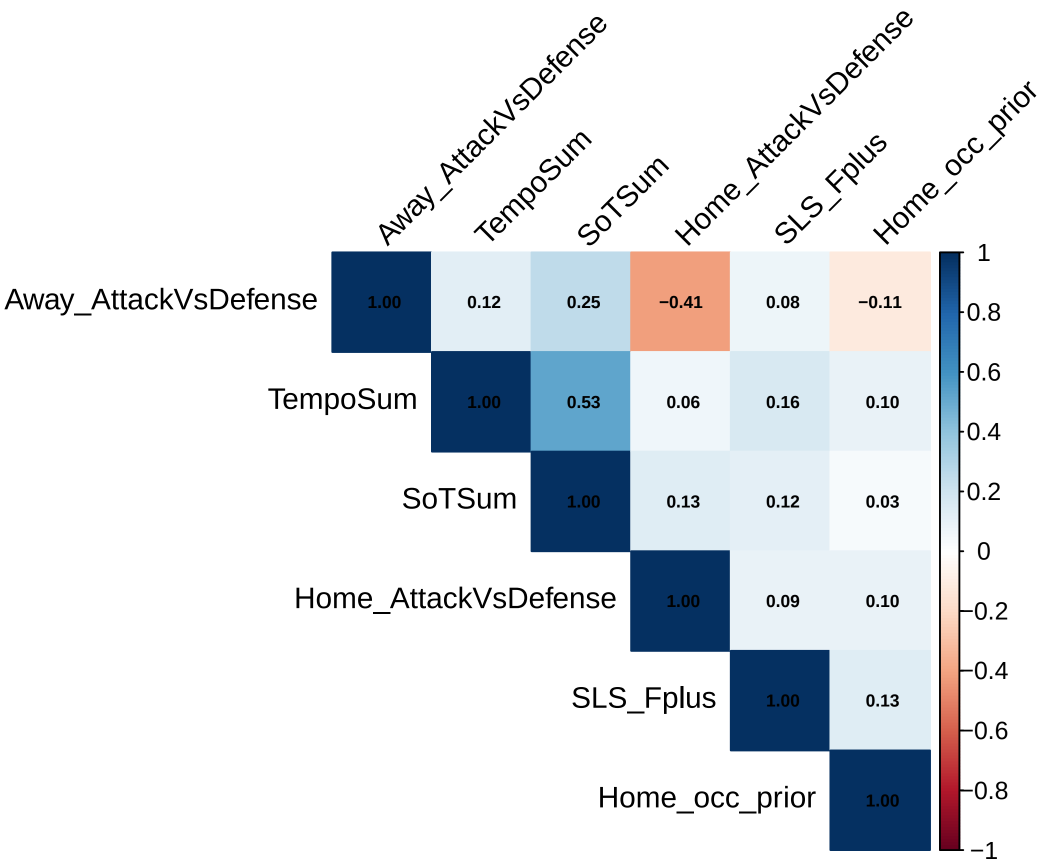
As you can tell, we currently cannot tell much apart from that there are clusters of data and they seem to correlate somehow. Surely, in higher dimensions, these clusters play off of each other and create better separations.

4 Correlation Heatmap

Next we quantify linear correlation among these predictors and the target. This helps flag multicollinearity (two features that are basically the same signal) or no-signal features.

```r
num_mat <- pair_df |>
  mutate(across(everything(), as.numeric)) |>
  cor(use = "complete.obs")

corrplot(num_mat,
         method="color", type="upper", order="hclust",
         addCoef.col="black", number.cex=0.6,
         tl.col="black", tl.srt=45)
```



We are looking for:

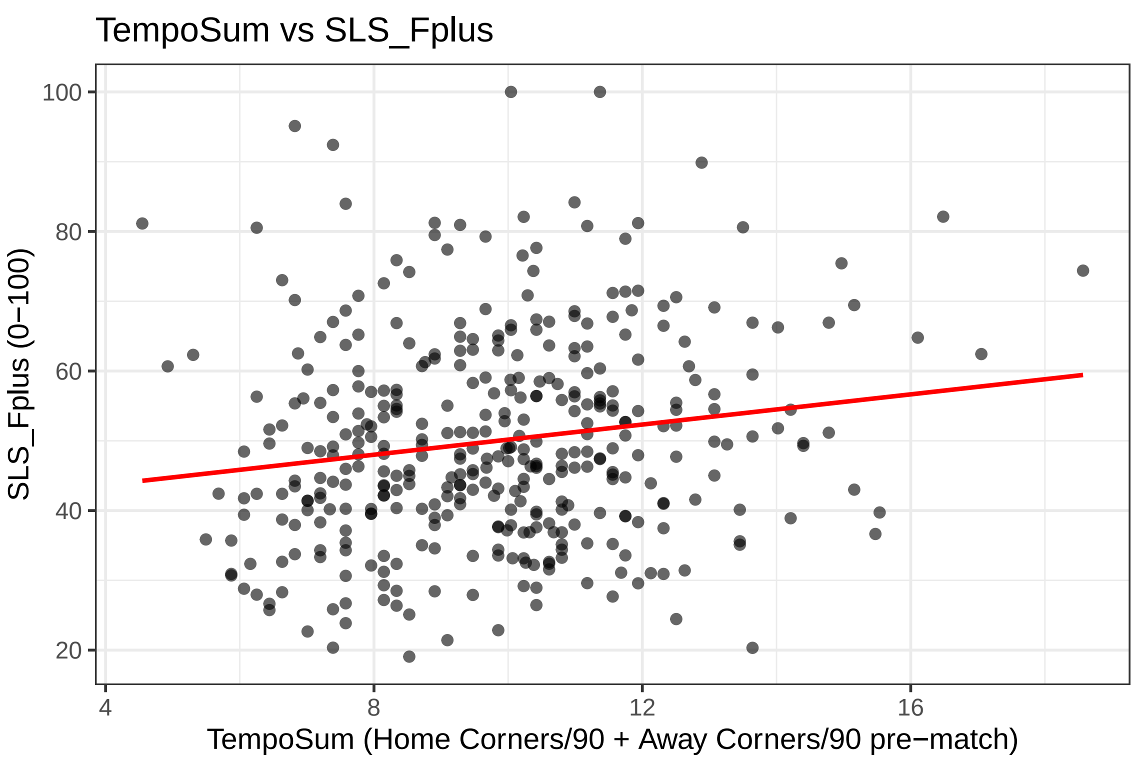
- Some correlation between `SLS_Fplus` and `TempoSum` / `SoTSum` / AttackVsDefense. That supports the idea that high-tempo, high-shot-volume matchups have effect on liveliness.
- Any predictors that are nearly identical to each other (very high correlation), which may cause instability in downstream models for us.

We find that some features are highly correlated to each other, but each of them have a weak correlation to the liveliness predictor.

5 Scatter vs Target (Annotated)

We now directly compare composite predictors to the target. The first plot shows whether matches with two aggressive teams (high combined shot tempo) tend to get higher `SLS_Fplus`. The red line is a simple linear fit.

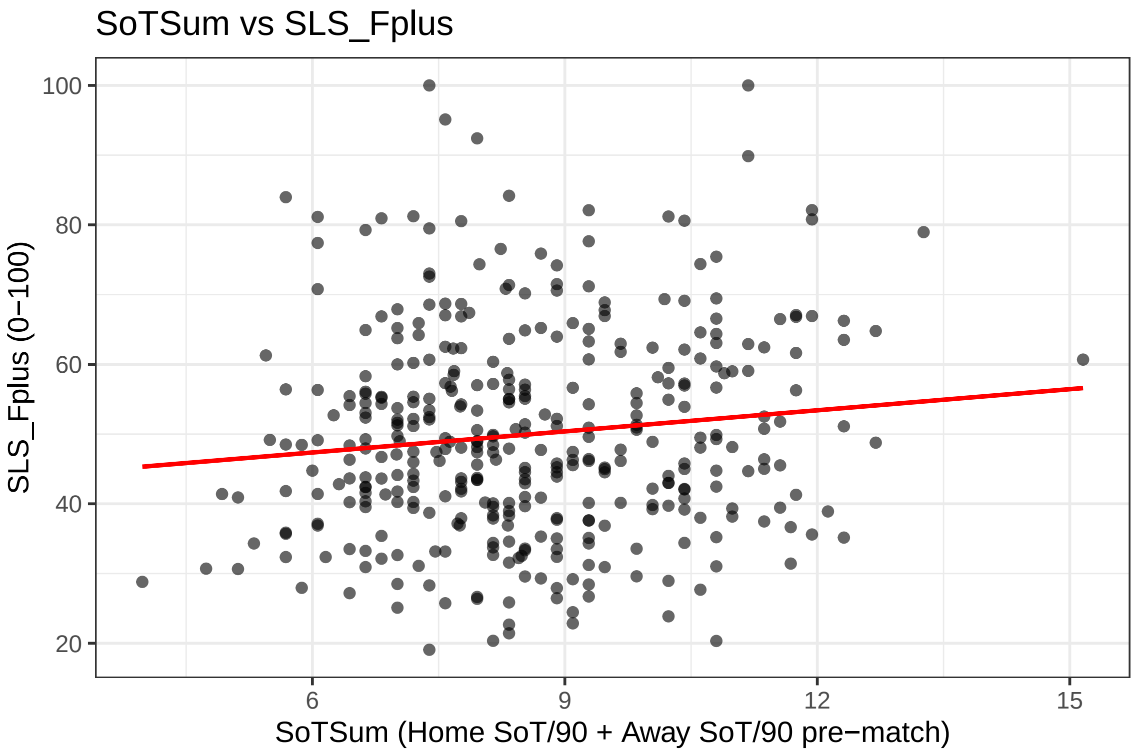
```
wide_df |>
  mutate(TempoSum_label = round(TempoSum,1)) |>
  ggplot(aes(x = TempoSum,
             y = SLS_Fplus)) +
  geom_point(alpha=0.6) +
  geom_smooth(method="lm", se=FALSE, linewidth=0.8, color="red") +
  labs(title="TempoSum vs SLS_Fplus",
       x="TempoSum (Home Corners/90 + Away Corners/90 pre-match)",
       y="SLS_Fplus (0{100)") +
  theme_bw()
```



Interpretation: This line trends upward, so "expected tempo" based on both teams' historical corners may mean that a match will be more lively.

We do the same for the combined SoT form.

```
wide_df |>
  ggplot(aes(x = SoTSum,
             y = SLS_Fplus)) +
  geom_point(alpha=0.6) +
  geom_smooth(method="lm", se=FALSE, linewidth=0.8, color="red") +
  labs(title="SoTSum vs SLS_Fplus",
       x="SoTSum (Home SoT/90 + Away SoT/90 pre-match)",
       y="SLS_Fplus (0{100)") +
  theme_bw()
```



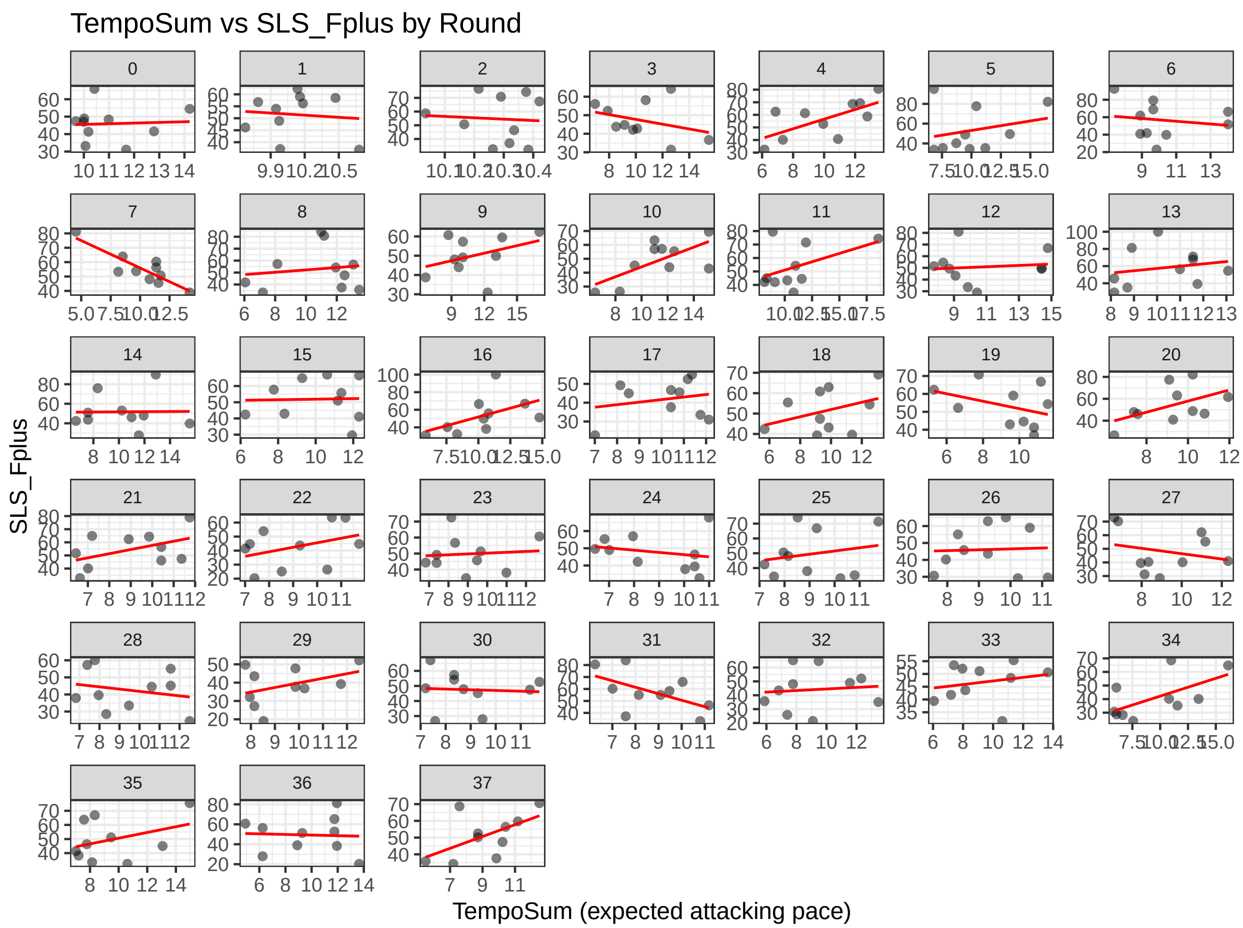
This feature also tracks **SLS_Fplus** well so this is a good sanity check for us that our targets and features positively relate.

6 Round-by-Round Facets

Finally, we check stability of these relationships over time. Early rounds use more league-average fallback because teams haven't built up 5-game histories yet. Later rounds use true rolling form. If the trend strengthens over rounds, that says the model's features become more predictive once the season has settled.

We facet TempoSum vs SLS_Fplus by round.

```
wide_df |>
  ggplot(aes(x = TempoSum,
             y = SLS_Fplus)) +
  geom_point(alpha=0.5, size=1.5) +
  geom_smooth(method="lm", se=FALSE, linewidth=0.6, color="red") +
  facet_wrap(~ Round, scales="free") +
  labs(title="TempoSum vs SLS_Fplus by Round",
       x="TempoSum (expected attacking pace)",
       y="SLS_Fplus") +
  theme_bw() +
  theme(strip.text = element_text(size=8))
```



The early-round facets look noisy or flat, that's expected: early rounds lean on fallback league priors. By mid-to-late rounds, each team's rolling metrics are "themselves," so we should start to see cleaner positive/negative slopes.

7 Takeaways

- The pre-match features (AttackVsDefense, TempoSum, SoTSum, rest, occupancy) are behaving in a way that's directionally consistent with our story about which matches "should" be lively.
- We are not leaking in-match data into pre-match features. Each row's features used only historical matches and league context *before* kickoff of that match.
- The target `SLS_Fplus` is from full-time stats, so it's valid to train a model that maps the pre-match view to that post-match label.
- Faceting by round helps confirm that predictive signal improves as the rolling windows become real (not priors).