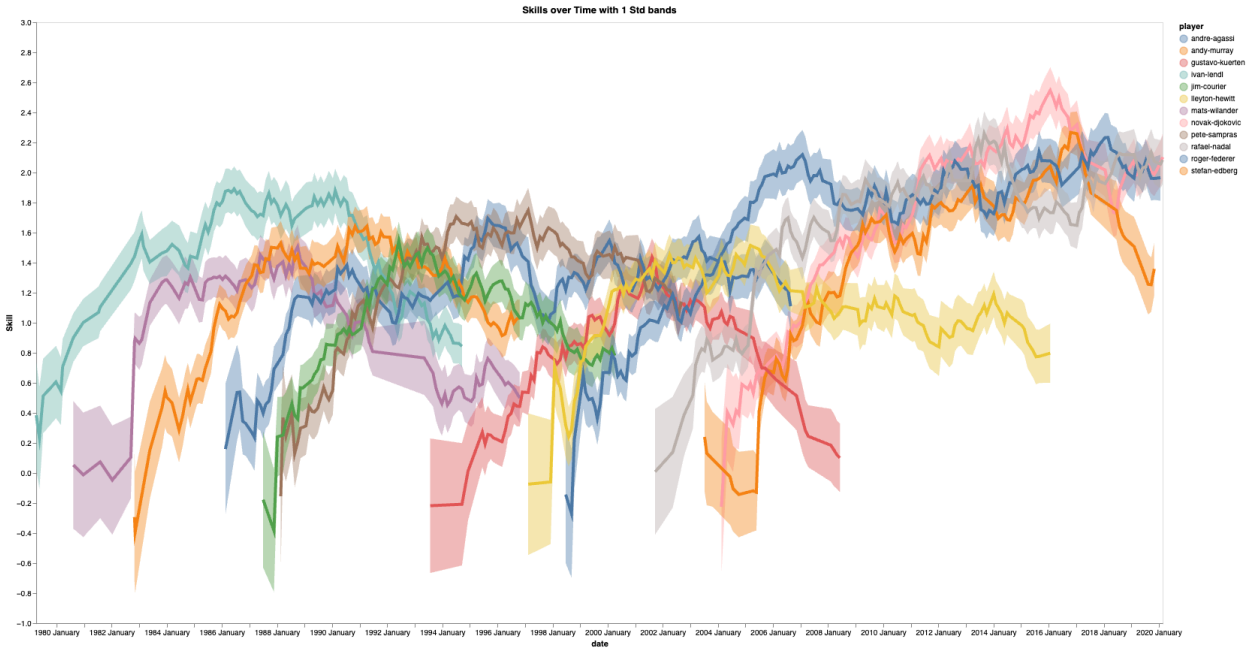
Data is downloaded from https://glandfried.github.io/TrueSkillThroughTime.jl/man/examples/

Apply TrueSkillThroughTime algo on the full dataset(326306 single matches & 18655 players):

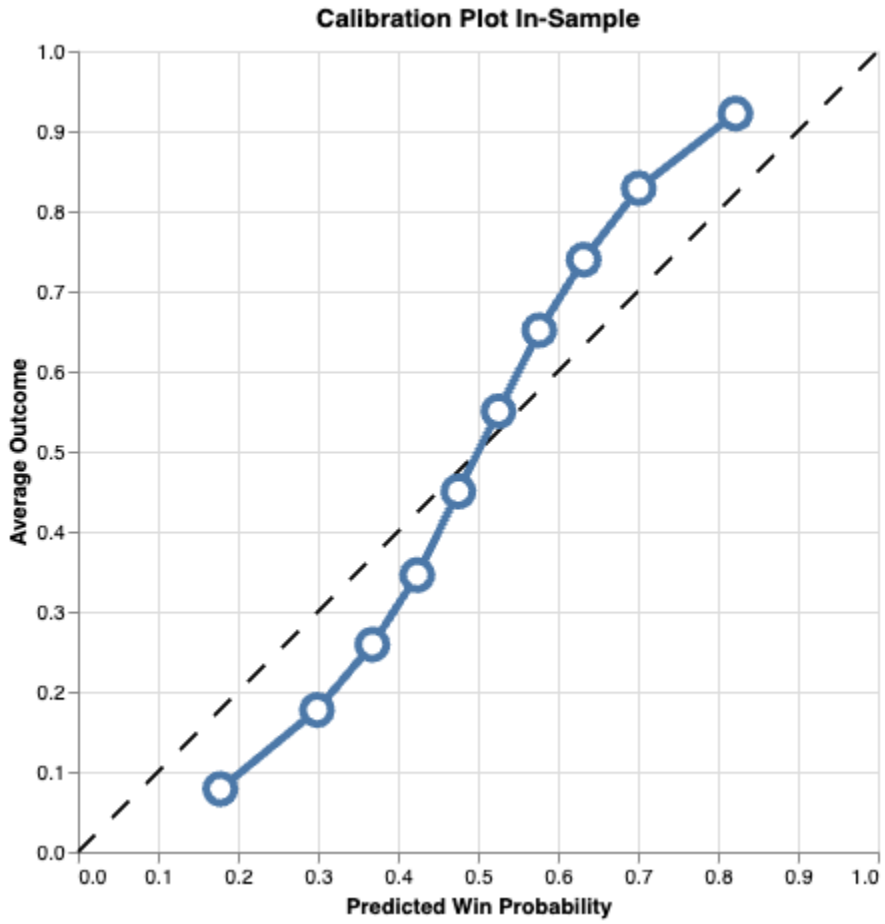
```
top_players_map = {
   'd643': "novak-djokovic",
   'f324': "roger-federer",
   's402': "pete-sampras",
   'l018': "ivan-lendl",
   'n409': "rafael-nadal",
   'a092': "andre-agassi",
   'h432': "lleyton-hewitt",
   'e004': "stefan-edberg",
   'c243': "jim-courier",
   'k293': "gustavo-kuerten",
   'mc10': "andy-murray",
   'w023': "mats-wilander"
}
```

```
self.plot_player_skills(players = top_players_id_lst, width=1500, burnin=0)
```



We can see from the skill curve plot that the GOAT should d643 novak-djokovic;
In addition, f324 roger-federer, n409 rafael-nadal share the runner-up in the recent days
We filter out those players who haven't played for a long time in their career(i.e. Those have large playing gaps), so that the learning curves won't have many straight lines.

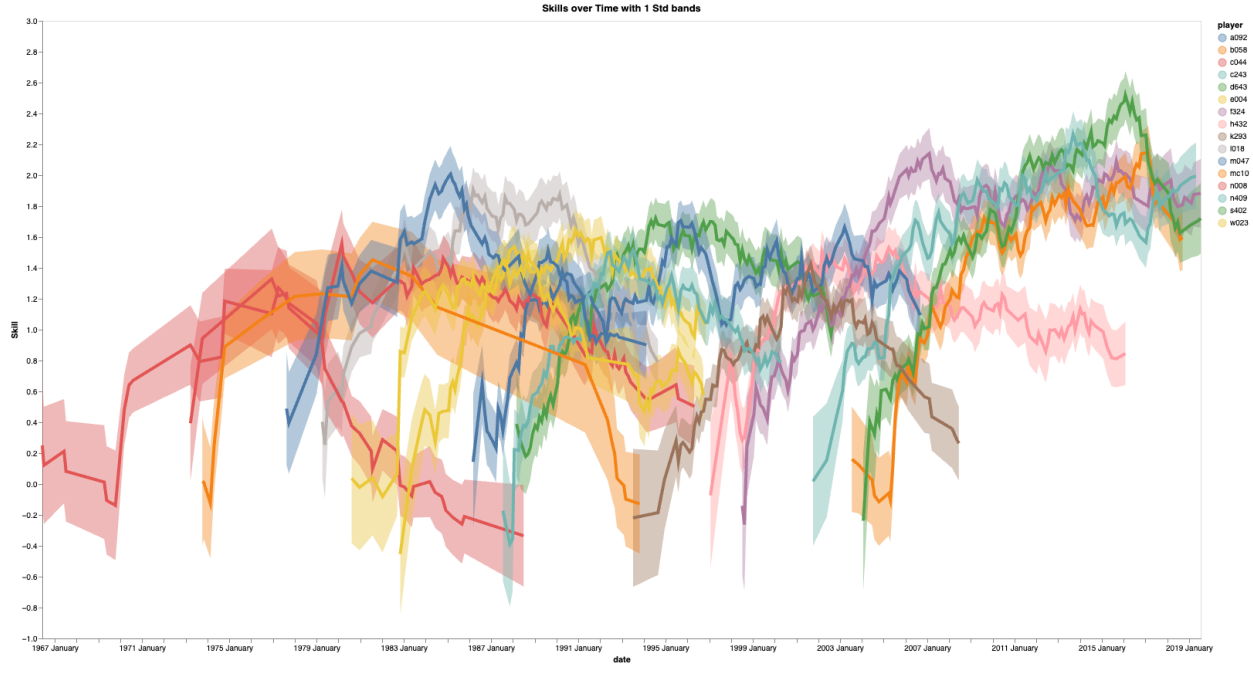
```
self.plot_calibration()
```



Out-of-sample evaluation:

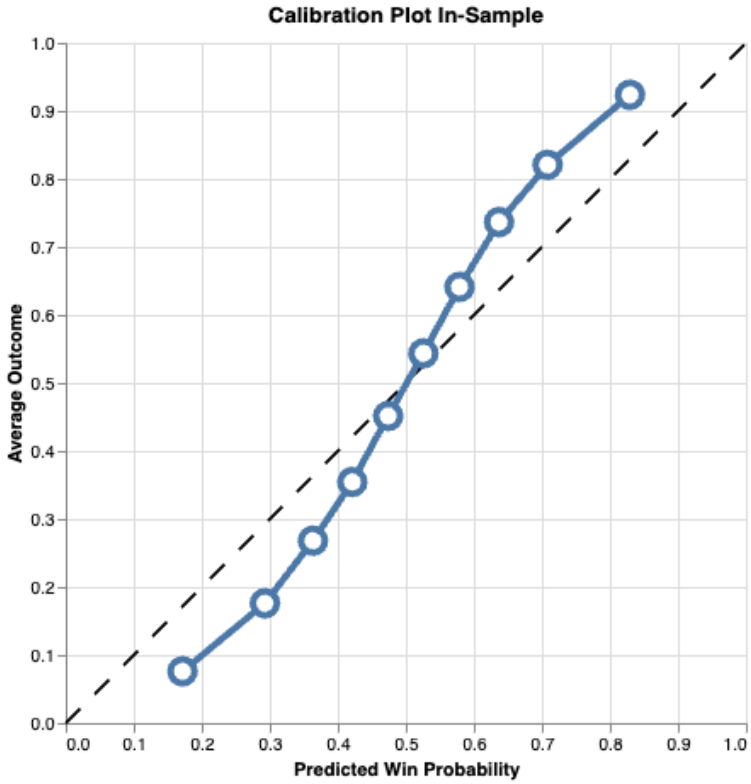
- We do a 80:20 train test split based on time for each one of 2502 players who have played at least 40 matches in their career.
    - data needs to be converted back to the winner-loser-timestamp format in order to apply TrueSkillThroughTime algo.
- We use the first 80% of a player's match to train the model and evaluate his last 20% of matches in the data.

```
self.plot_player_skills(players = top_players_id_lst, width=1500, burnin=0)
```

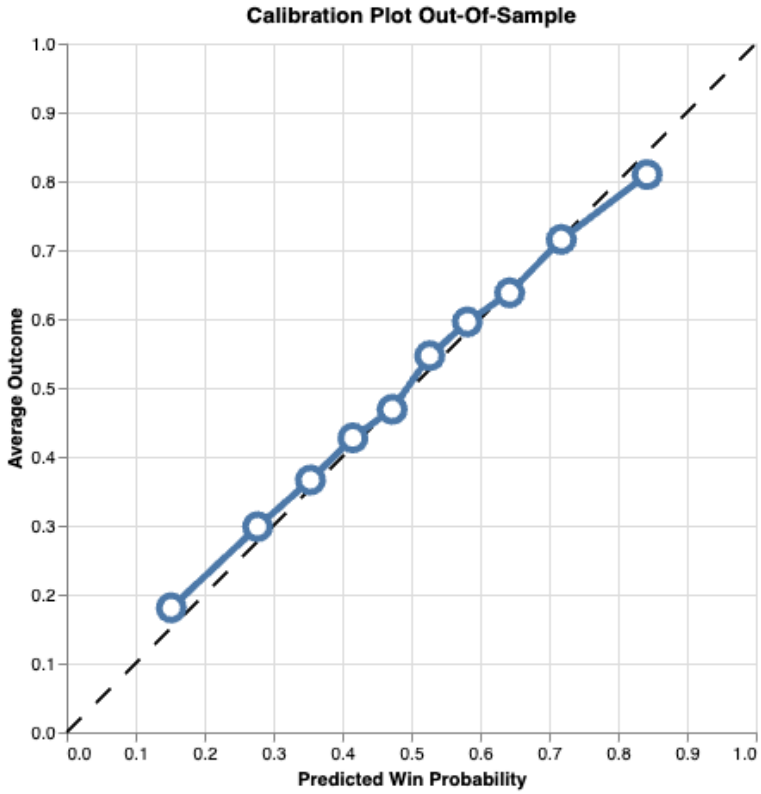


The overall trend and the absolute skill values for the train set are similar to the result of the full sample.

In-sample vs Out-of-sample calibration plot:



We filter out those matches that appear on both the train and test set, we do the OOS calibration plot again. The final test set has 26301 games.



As shown in both In-Sample and Out-Of-Sample calibration plot, we have a very close to diagonal line for pred_win_prob vs avg_outcome. It fits better than any other game we applied.

roc_auc_score:
Assign a match to positive if the winner is alphabetically lower than the loser and is negative otherwise. The dataset is relatively equally balanced over the classes.
If we evaluate on the entire dataset:
auc: 0.8135

If we evaluate on the 20% test dataset:
auc: 0.7130

Game Network Graph:

[Leave as blank]

There are too many regular players(over 2500), the game network should be very hard to visualize. We may have a stricter definition on regular players(>=100 games in career for example) if we want to compare with other games. There shouldn’t be many clusters in the network, meaning the regular players have played very frequently with each other.