Machine Learning

WNBA Playoff Predictions

G51

Francisco Cardoso - 202108793 (0.33)

João Fernandes - 202108867 (0.33)

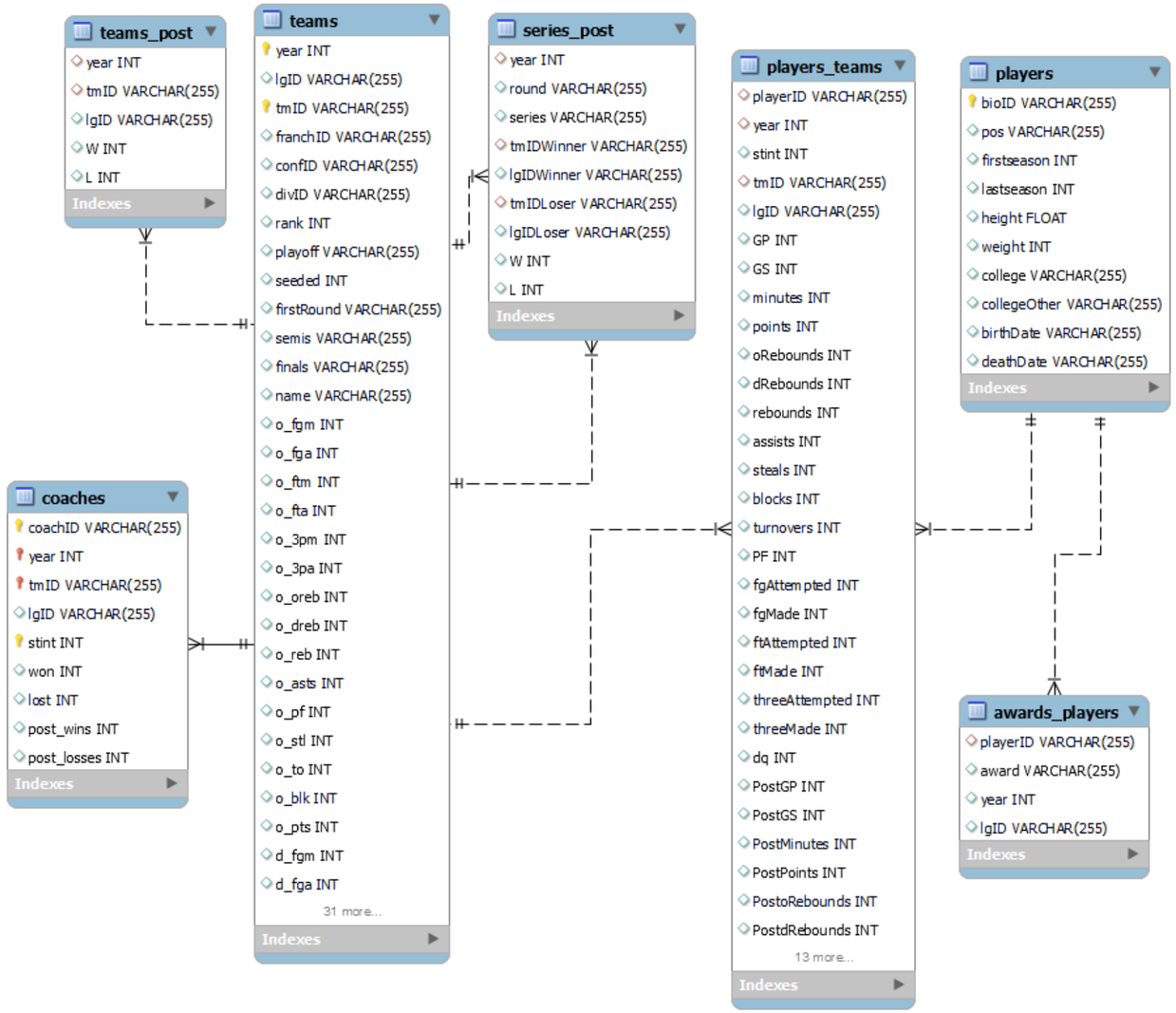
Tomás Palma - 202108880 (0.33)

Domain Description

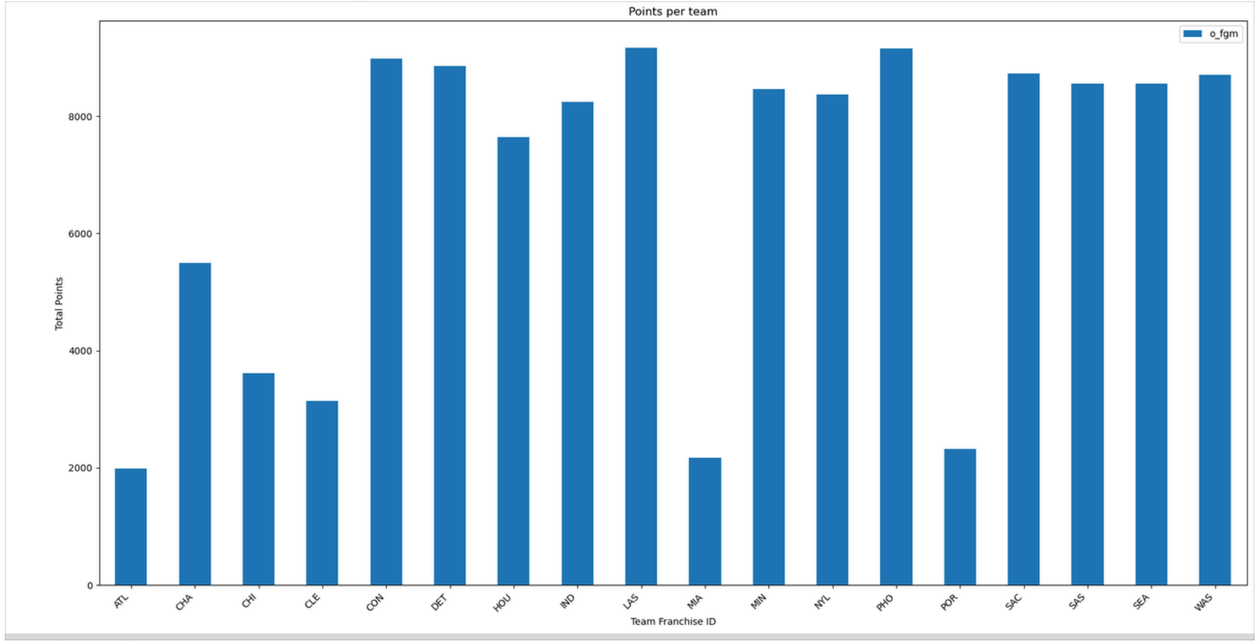
Objective: which teams qualify for playoff?

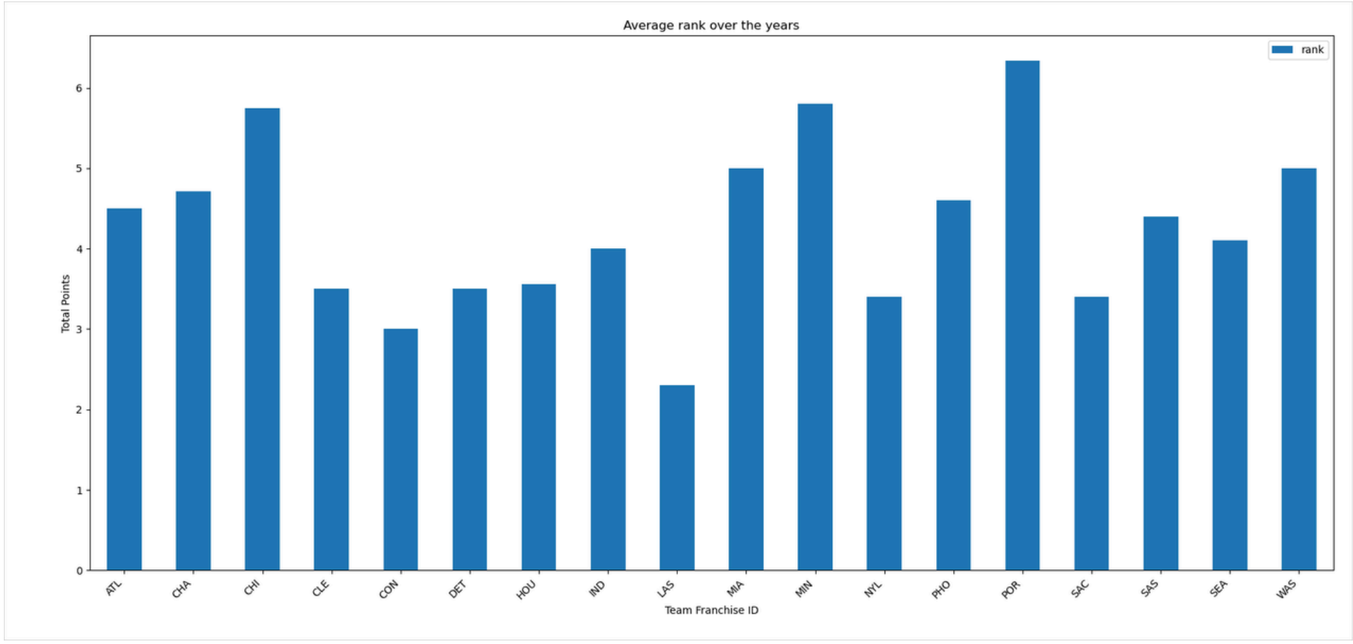
57 coaches, 893 players, 10 seasons and 18 teams

players, teams, games, awards and coaches statistics for every season

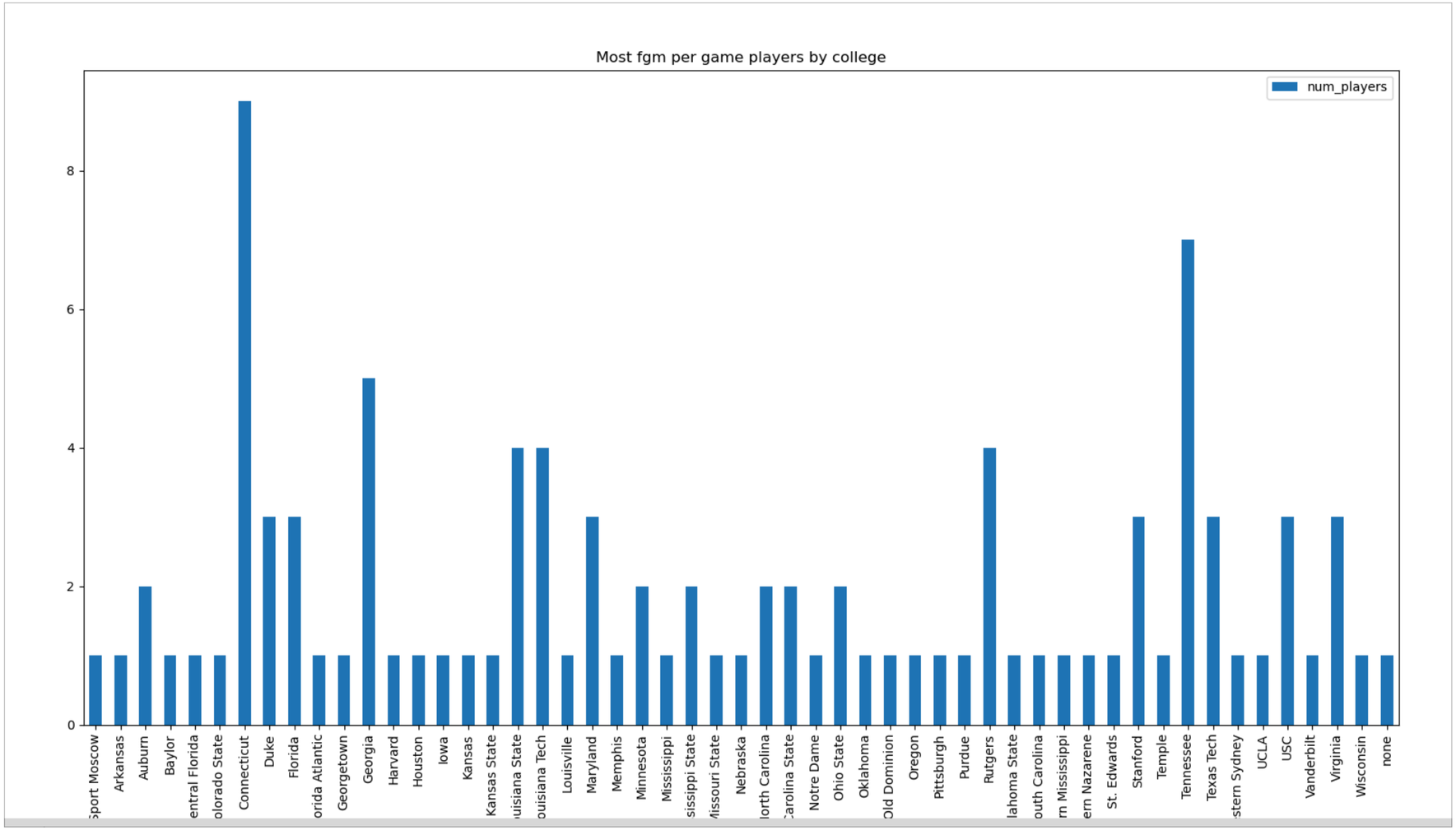


Data Analysis - Main findings (1/3)

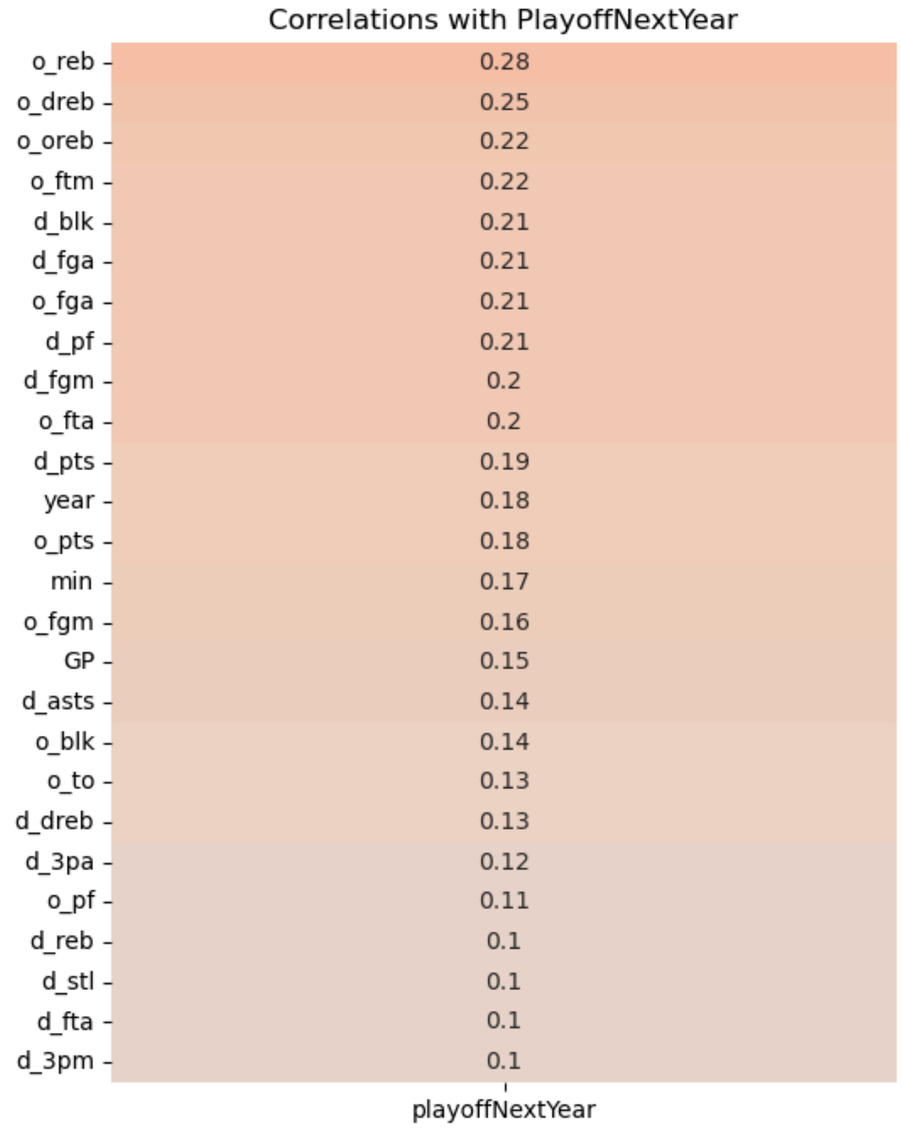


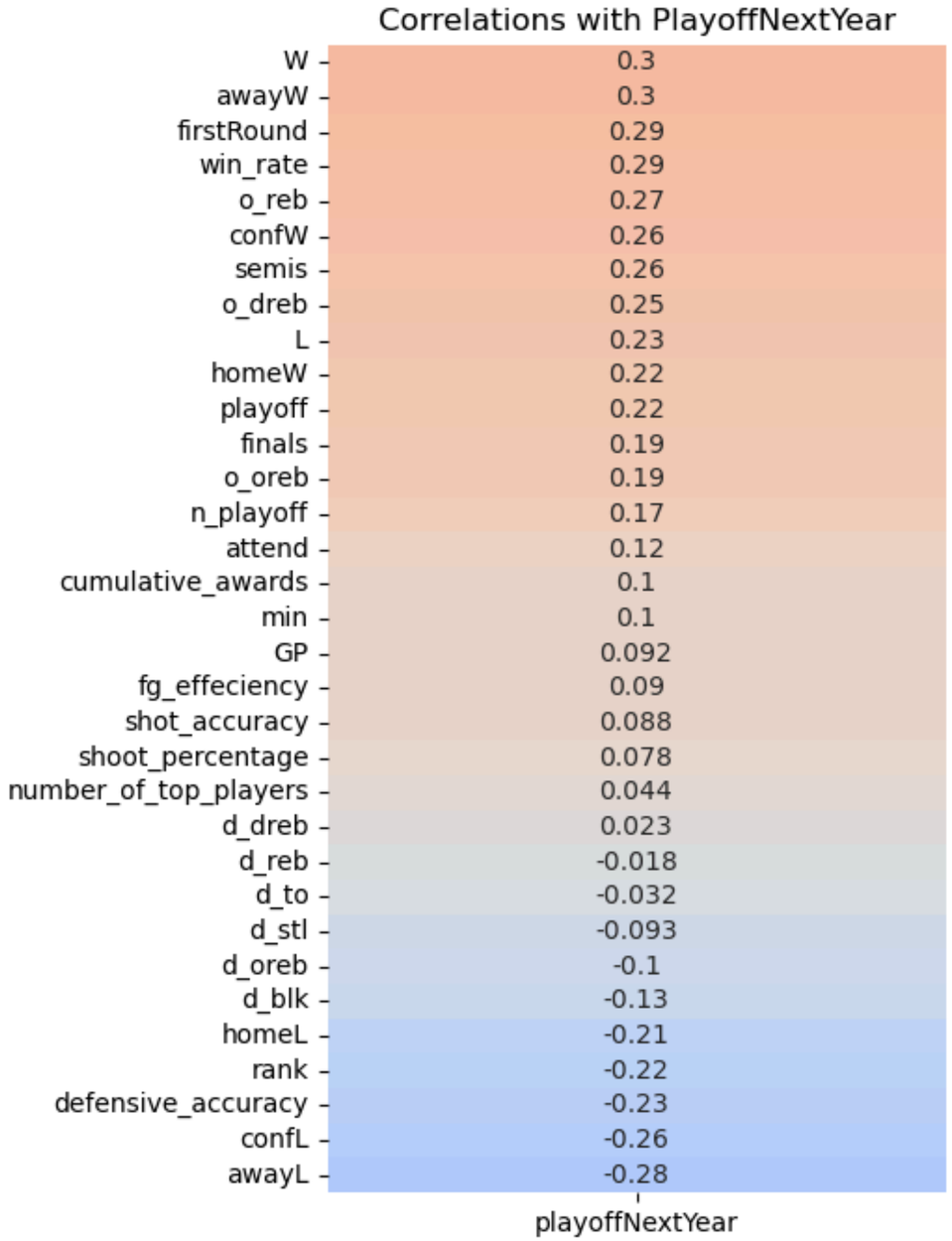


Data Analysis - Main findings (2/3)



Data Analysis - Main findings (3/3)





Data preparation

Before starting to train models the data was cleaned.

General

Removed ids from all the supplied CSVs.

Players

Removed personal information irrelevant to performance (birth and death date).

Removed awards not based on player skill (e.g. sportmanship awards).

Teams

Removed conference details (e.g. the conference id).

Removed name and arena.

Data preparation

Feature creation and selection

Teams

Use every relevant column for each team and year.

Added shot accuracy, defensive accuracy, shoot percentage, field goals accuracy, win rate tendencies and past playoff appearences.

Players

Added how many top 5 players of all time are playing on each team and year.

Awards

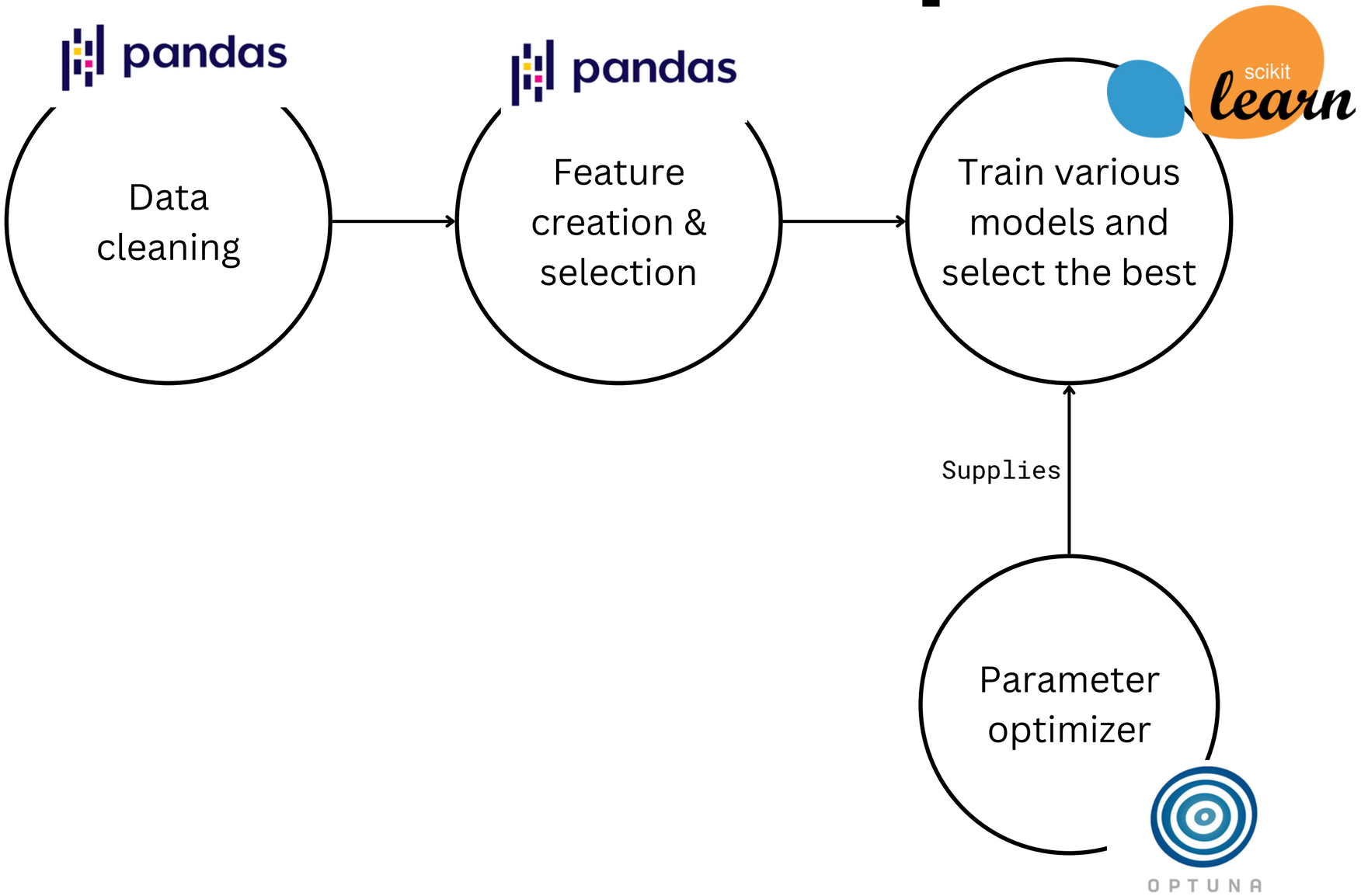
Added how many relevant awards each team won before that year.

Coaches

Added first and second coaches overall win rate.

Note: The awards and coaches features are cumulative, which means that for a year N, we consider the sum (or average) of the values until that year.

Experimental setup



A note about the setup

We used both normal python files and jupyter notebooks

Jupyter notebook

- Used to showcase the used features so the way they were calculated are better seen

Python files

- Contains the logic of the whole pipeline
  - Can be run using *python3 project.py*

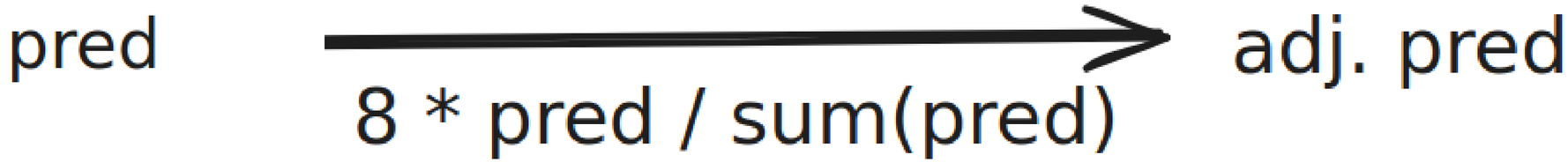
Models used

- Decision tree classifier
- Neural networks
- Logistic regression
- Gaussian Naive Bayes
- Ada Boost
- Gradient boost
- K-Nearest Neighbor
- Random forest

How were parameters chosen?


```python
def optimize_dtc(X_train, y_train, X_test, y_test):
    def objective(trial):
        param_grid = {
            'criterion': trial.suggest_categorical('criterion', ['gini', 'entropy', 'log_loss']),
            'max_depth': trial.suggest_int('max_depth', 2, 1000),
            'min_samples_split': trial.suggest_int('min_samples_split', 2, 1000),
            'min_samples_leaf': trial.suggest_int('min_samples_leaf', 1, 1000),
            'random_state': trial.suggest_int('random_state', 1, 100000),
            'splitter': trial.suggest_categorical('splitter', ['best', 'random']),
        }

        model = DecisionTreeClassifier(
            criterion=param_grid['criterion'],
            max_depth=param_grid['max_depth'],
            min_samples_split=param_grid['min_samples_split'],
            min_samples_leaf=param_grid['min_samples_leaf'],
            random_state=param_grid['random_state'],
            splitter=param_grid['splitter']
        )

        model.fit(X_train, y_train)
        y_pred = model.predict_proba(X_test)
        return eval.error_eval(y_test, y_pred)

    study = optuna.create_study(direction='minimize')
    study.optimize(objective, n_trials=1000)

    return study.best_params
```

What is the error evaluation function?



$$error = sum(|adj.\ pred - label|)$$

label is the real result (0-1)

pred is the probability obtained by the *model.predict_proba*

adj.pred is the probabilities normalized

Training and test sizes

There were noticeable differences on the behaviour of the models depending on the training and testing samples

When the training data was smaller (year < 6) neural networks were easier to optimize

When the training data was larger (year < 8) decision tree classifiers performed the best were easier to optimize

Best results for each model

Model	Accuracy	Error (0 - 12)
Decision Tree	84%	2,5
Neural Networks	70%	4,6
Logistic Regression	54%	6,4
Ada Boost	46%	6,3
Gradient Boost	54%	6,6
K(28)-Nearest Neighbors	61%	5,3
Gaussian Naive Bayes	61%	5,6
Random Forest	69%	6,0

Feature improvements (1/2)

- Cumulative Team Awards
- Coaches Win Rate

For **both** tables, we used information on the following year to try to get a more realistic and intuitive correlation with playoffNextYear (see next slide).

For the awards table, we expanded the scope to include coaches, as previously, we had only included players.

Feature improvements (2/2)

Initial Approach:
Each team/year was assigned the cumulative awards of its players for that year.

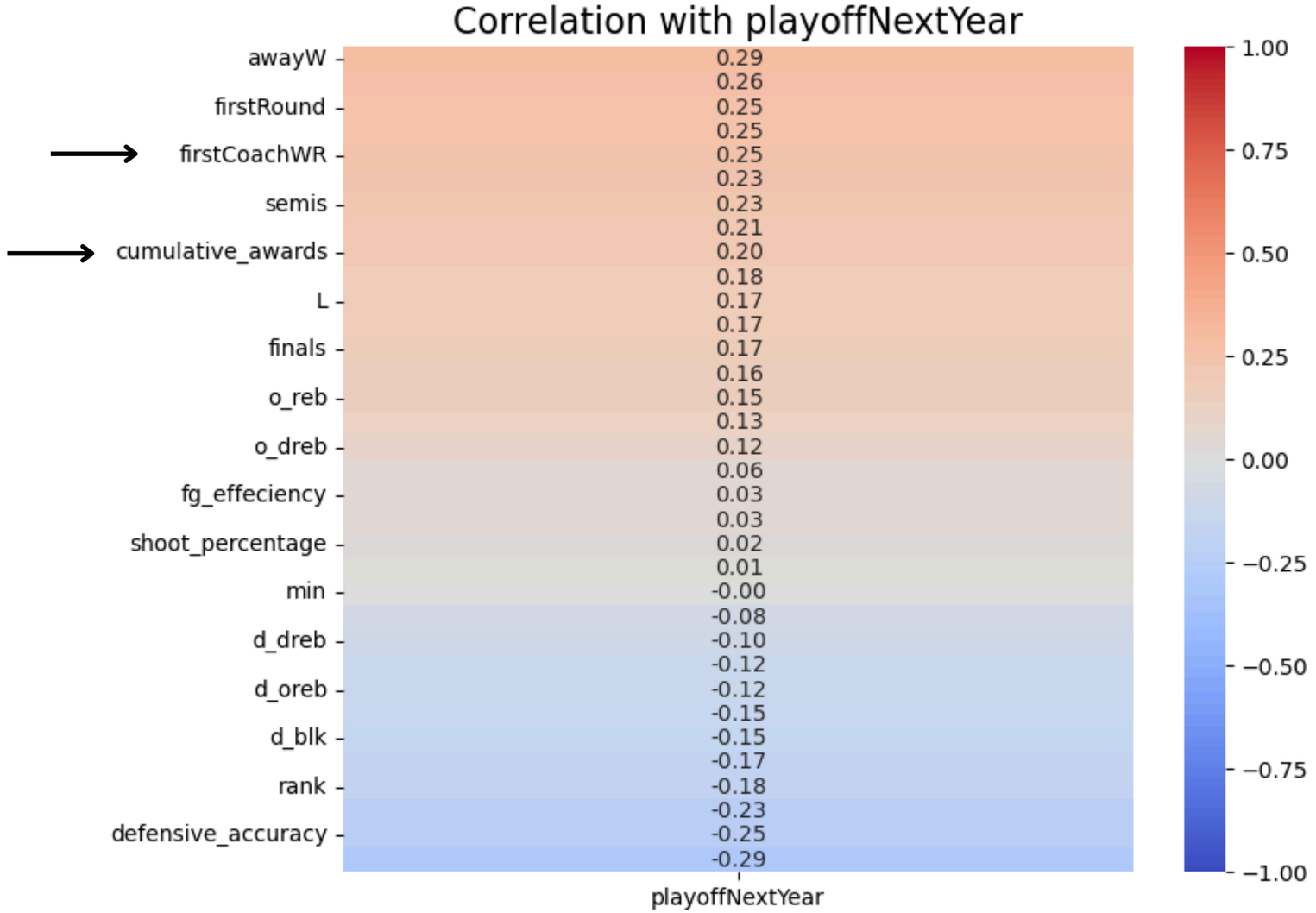
- *E.g. Team 1 in Year 1 gets 8 awards.*

Improved Approach:
Teams are assigned the current cumulative awards of players they will have next year.

- *E.g. Team 1 in Year 1 gets 3 awards, as Player B joins in Year 2.*

Players			
playerID	year	tmID	awards
playerA	1	team1	8
playerA	2	team2	10
playerB	1	team2	3
playerB	2	team1	5

Updated Data Analysis



Pipeline for submission with year 11 (1/2)

1. Added a new df with the data from season 11 according to the provided CSVs.

2. Concat these new DF with the original one from the training data besides season 11

Pipeline for submission with year 11 (2/2)

- In the different submissions we did in the competition we chose the best model from our pipeline, which gives us the models with error. However, the features used varied in each submission.

- In the first submission we did not yet have into account the season 11 data. In the second one we had already performed what is described in the previous slides. Finally, in the third one we added improved cumulative features about the coaches and player awards.

Final Submission

tmID	ATL	CHI	CON	IND	LAS	MIN	NYL	PHO	TUL	SAS	SEA	WAS
% of playoff on season 11	51	69	86	55	95	60	35	58	94	44	66	33

Conclusions & Limitations

A more complex feature selection method using scikit learn was not implemented.

In order to achieve better results, we could have added more complex evaluation mechanisms such as sliding windows