Chess Rating Report

As a person who bet on sports, I want to predict if some chess player's rating will be higher or lower next month. I have a collection of datasets for every month from January 2019 to December 2021 with player's statistics – rating, titles, names etc. This is the statistics taken from FIDE official website, so the data is trustworthy.

The first dataset (for January 2019) contains 325205 rows and every next one is larger than the previous one. The statistical unit is a player. There are 15 columns (I lowercased their names). The table below shows column descriptions and data types.

Table 1. Initial set of columns

Title: meaning	Type
id number: unique player's id	int64
name: player's name	object
fed: player's homeland	object
sex: player's sex	object
tit: the highest title of the player	object
wtit: the highest title for playing woman	object
otit: non-player title (trainer, etc)	object
foa: FIDE online arena achievement	object
rat: player's rating	int64
gms: number of games played in a previous month	int64
k: development score	int64
b-day: player's year of birth	int64
flag: specific status of a player if any	object
year: current year	int64
mon: current month	int64

Initial data transformation

There are mixed types for 'b-day' and 'rat' columns in other datasets because of values like '0 1984' in b-day and 'M 2100' in rat, but it was easily cleaned. For convenience of the analysis, I turned the column 'b-day' into the column 'age'.

To predict the change of rating (lower or higher), there should be added the target column ('ratchg') for qualitative rating change with categorical values – 0 (means no change of rating), 1 (means that rating decreased) and 2 (means that rating increased). Additionally, a new column with quantitative rating change 'ratdiff' (computed as the difference between the 'rat' column of a current month and the 'rat' column of a previous month) could be helpful.

There is no change of the rating for the first month (January 2019), so for basic exploratory analysis I chose the second month (February 2019, 327897 rows). It is definitely not necessary to know the name of a player to predict its rating, so I dropped the 'name' column.

Missing values & categorical columns

There are many missing values – about 98% - in columns with titles and 'foa' column. These missing values can be interpreted as a lack of title or lack of achievement on online arena.

Columns 'tit', 'wtit', and 'foa' contain the following types of values:

- ‘tit’ : NaN, WFM, IM, GM, WIM, FM, CM, WCM, WGM, WH
- ‘wtit’ : NaN, WFM, WIM, WCM, WGM, WH
- ‘foa’ : NaN, ACM, AFM, AIM, AGM

Because all these columns contain data that can be interpreted as categorical, I replaced NaN values there with one more category ‘NT’ – no title. The column with common titles (‘tit’) contains all the values from the women titles column, so the column ‘wtit’ is redundant and can be dropped.

‘otit’ column contains more than 100 types of values of non-player titles. Given that there is about 98% of missing values and too many unique values for a categorical column, I decided to drop it as a non-informative one for the rating change prediction purpose.

There is about 46% of missing data in the ‘flag’ column. It has the next values: NaN, i, wi, w. Based on dataset description, I can interpret only the flag ‘i’, which means inactive in a last year, but there is no interpretation of the other values. I decided to drop this column as well.

Additionally, the columns ‘sex’ and ‘k’ contain values M, F and 10,15,20,40 respectively; therefore, I changed their type to categorical.

So far, I have the dataset with the following columns

Table 2. New set of columns (cat – categorical, obj – object)

column	id number	fed	sex	tit	foa	rat	gms	k	year	mon	ratdiff	ratchg	age
type	int64	obj	cat	cat	cat	int64	int64	cat	int64	int64	int64	cat	int64
comment											quantitative change of rating	qualitative change of rating	

Basic statistics

Table 3. Exploratory analysis

Column	**Mean**	**Min**	**Max**	**Std**	**2.5th percentile**	**50th percentile**	**97.5th percentile**
rat	1690	1001	2845	345	1062	1705	2305
gms	0.81	0	81	2.5	0	0	9
k	31	10	40	10.24	20	40	40
ratdiff	0.23	-362	340	0.57	-20	0	21
age	37.5	6	634	19.7	11	35	79

Based on the data in the table 3 it is obvious that there are some outliers in ‘gms’ column, because the 97.5th percentile is equal to 9 but the maximum value is 81 which is nonsense. I would replace these outliers with a median number of games played by a player in other months.

A number of games played can explain outliers in quantitative rating change (min 362, max 340).

Outliers in age column can be replaced by a median value of the column.

Importance of variables

Correlation matrix reflects a linear dependency between different columns.

Graph 1. Correlation matrix for numerical columns

The matrix above shows that the target column ‘ratchg’ has a positive correlation with a number of played games ‘gms’ and negative correlation with development score ‘k’ and quantitative change of rating ‘ratdiff’.

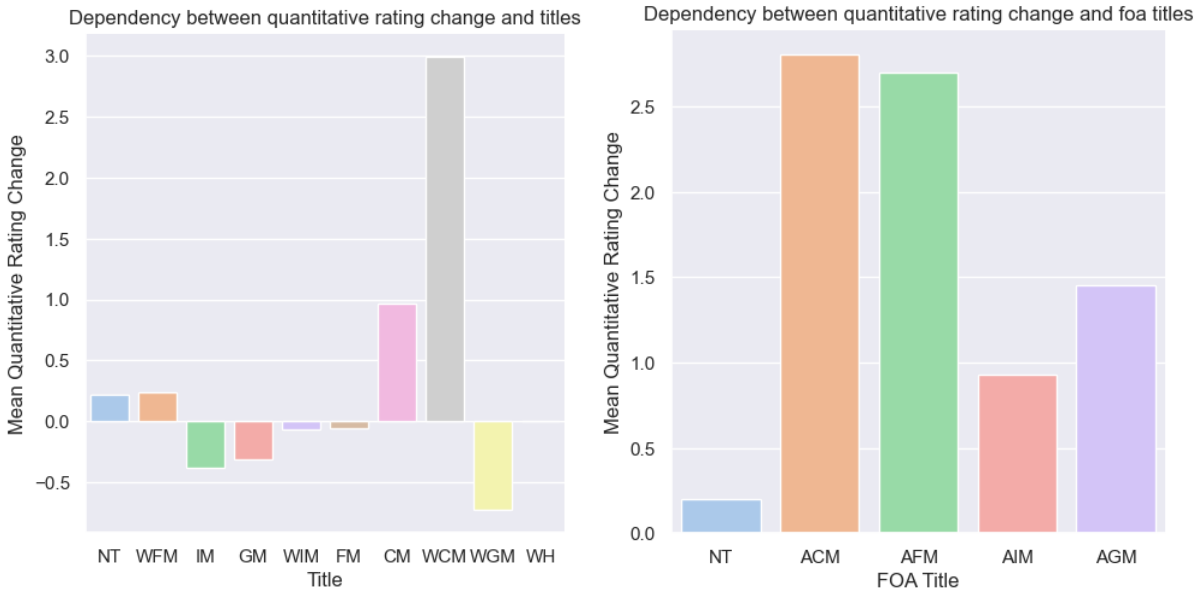
To analyze a rating change dependency of gender and age I have made barplots for different age groups.

Graph 2. Dependency between mean change of rating and gender across age groups

Both graphs shows that there is a non-linear dependency between mean change of rating and age, so I need to keep the ‘age’ column for a prediction purpose. Moreover, the difference between genders is also visible here; therefore, the ‘sex’ column is also significant for the model.

To analyze a dependency between rating change and titles I have made barplots for titles and FOA achievements.

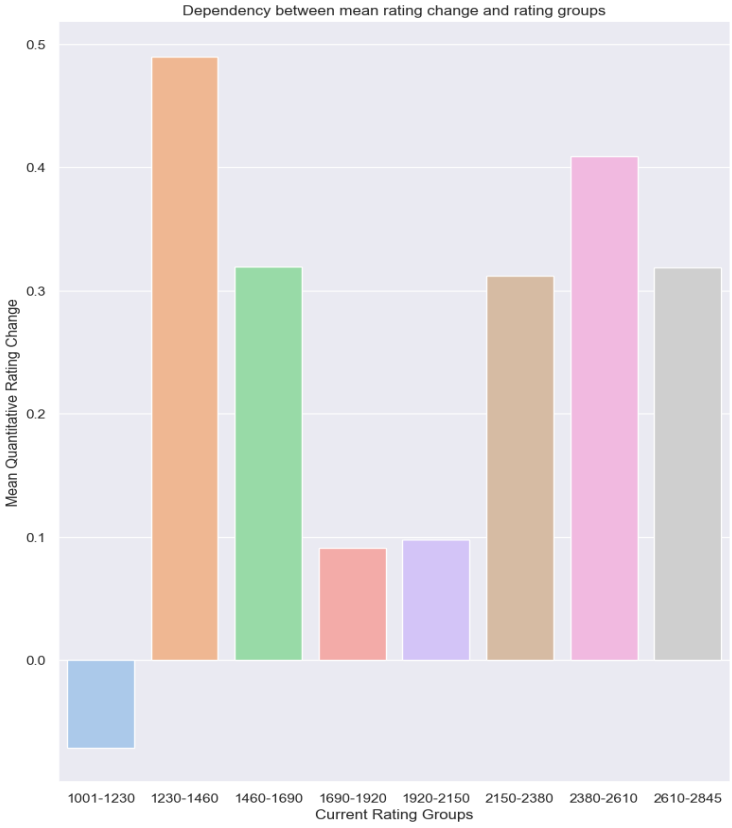
Graph 3. Dependency between mean change of rating and titles and FOA achievements



From the plot, it is possible to notice that the rating change is significantly different for different titles and achievements; therefore, I keep both columns – 'tit' and 'foa' – for further modelling.

The column 'rat' was used for rating change calculation and can be omitted due to its insignificance as shown on a graph below.

Graph 4. Dependency between mean rating change and current rating groups



I have divided the range of rating values to several groups and could not find any dependency between quantitative rating change and current rating.

Analyzing the significance of the homeland column 'fed' I grouped mean quantitative rating difference by homeland, sorted it and got its head and tail.

Table 4. Mean Rating Change by country (tail)

Fed	Mean Quantitative Rating Change
FIJ	-3.91
SSD	-3.46
SEY	-3.44
BDI	-2.36
AFG	-2.29

Table 5. Mean Rating Change by country (head)

Fed	Mean Quantitative Rating Change
NZL	1.14
MGL	1.15
NCA	1.19
MNC	1.66
SOM	2.00

Based on the significant difference between head and tail the 'fed' column can potentially be important for the prediction model, so I decided to keep it so far.

The final set of columns in dataset is shown in a table below.

Table 6. Final set of columns (cat – categorical, obj – object)

column	id number	fed	sex	tit	foa	gms	k	year	mon	ratdiff	ratchg	age
type	int64	obj	cat	cat	cat	int64	cat	int64	int64	int64	cat	int64
comment										quantitative change of rating	qualitative change of rating	

For the further modelling, I am going to transform data the same way for each month and concatenate them to one dataset.

Modelling 1. Data transformation

'All models are wrong, but some are useful.'
George E. P. Box

To be able to apply any model, I did some transformations of the dataset.

1. I have decided to remove 'id number' column, because its value does not give any information for the chosen classification task.
2. I replaced 'year' and 'month' columns with a new column called 'xp' meaning the experience (amount of months) of the player within scope of the data (players in the first month have 1 xp). Therefore, this column helps to distinguish newly added players in comparison with previous months.
3. I removed the 'ratdiff' column because it is the same column as the target 'ratchg', but quantitatively represented.
4. Column 'fed' was transformed to categorical.

After that, the dataset was converted to numpy array to be able to apply a model.

For further modelling, the dataset was splitted into two parts – training and test. I have decided to take 29 months for training and 7 months for test. Moreover, I decided to choose not just last 7 months, but to alternate test and training months, so in test set I have following months: 12.20, 02.21, 04.21, 06.21, 08.21, 10.21, and 12.21. The reason for this was my suggestion that the training exceptionally on old

data can make a weak model in case of some significant changes in last 7 months. Therefore, to make the experiment more consistent, this splitting strategy has been chosen.

Modelling 2. Models

1. **Logistic regression**

The first choice of the model for the classification task was the logistic regression (sklearn.linear_model.LogisticRegression) with default parameters. After training the model on training dataset, I also conducted a cross-validation on full data with metrics: accuracy, ROC AUC OVO (one-vs-one), ROC AUC OVR (one-vs-rest), F1 scores.

Accuracy on training/test data	Cross-validation accuracy 1	Cross-validation accuracy 2	Cross-validation accuracy 3	Cross-validation accuracy 4
0.96	0.94	0.95	0.95	0.94

ROC AUC OVO score 1	ROC AUC OVO score 2	ROC AUC OVO score 3	ROC AUC OVO score 4
0.83	0.84	0.81	0.82

ROC AUC OVR score 1	ROC AUC OVR score 2	ROC AUC OVR score 3	ROC AUC OVR score 4
0.97	0.97	0.94	0.96

F1 macro	F1 micro
0.46	0.96

Based on the cross-validation accuracy and ROC AUC scores, the model looks good, however, as it can be seen from F1 score values, the model has a good accuracy overall, but is not performing well on certain classes. F1 macro score evaluates the performance of the model on all classes individually. It can be well-illustrated using confusion matrix:

		Predicted	Predicted	Predicted
		0 (not changed)	1 (decreased)	2 (increased)
Actual	0 (not changed)	2458567	2948	4609
Actual	1 (decreased)	29454	623	20697
Actual	2 (increased)	36063	598	18652

From the model summary statistics (using statsmodels.MNLogit) I found out that **p-value** is extremely small (it displays 0.000) for all predictors, which means that all features are statistically significant for the prediction.

Attempts to improve the model using different solvers and feature engineering (polynomial features and one-hot encoding for categorical columns) did not improve its performance.

2. **Decision Tree**

To handle the significant class imbalance I have decided to use the Decision Tree Classifier (sklearn.tree.DecisionTreeClassifier) with default parameters, because decision trees are rather robust in that case.

F1 score values look much better for that model:

F1 macro	F1 micro
0.67	0.98

The confusion matrix is also showing a progress in comparison with Logistic Regression model.

		Predicted	Predicted	Predicted
		0 (not changed)	1 (decreased)	2 (increased)
Actual	0 (not changed)	2463636	1177	1311
Actual	1 (decreased)	1787	25377	23610
Actual	2 (increased)	1965	24357	28991

Unfortunately, Decision Trees can be easily overfitted, which was confirmed with cross-validation:

Accuracy on training/test data	Cross-validation accuracy 1	Cross-validation accuracy 2	Cross-validation accuracy 3	Cross-validation accuracy 4
0.98	0.40	0.14	0.28	0.93

3. Random Forest

Due to obvious overfitting, I have moved to Random Forest model (sklearn.ensemble.RandomForestClassifier) with default parameters, which is a combination of decision trees that improves its accuracy and reduce overfitting.

The **accuracy** of the model is **0.98**. F1 score values look a bit better than for Decision Tree model:

F1 macro	F1 micro
0.70	0.98

The confusion matrix is shown below and also reflects the progress:

		Predicted	Predicted	Predicted
		0 (not changed)	1 (decreased)	2 (increased)
Actual	0 (not changed)	2463645	1026	1453
Actual	1 (decreased)	709	26207	23858
Actual	2 (increased)	761	21574	32978

Instead of using cross-validation, which is time consuming in case of Random Forest model, I used OOB score to check the performance of the model (each tree in a forest is trained using a different bootstrap sample and the OOB score is computed by averaging the correct predictions on data that was not in that bootstrap sample). Additionally, I computed ROC AUC scores with one-vs-one and one-vs-rest strategies.

OOB score	ROC AUC OVO	ROC AUC OVR
0.96	0.86	0.99

Summary

1. The Logistic Regression model showed rather good overall performance and helped to figure out the significance of features, but appeared to be sensitive to class imbalance.
2. The Decision Tree model showed the great improvement at the first glance, but suffered from overfitting. Probably it is possible to improve the model with some parameter tuning, but I have decided to improve it using ensemble method.
3. Further investigation revealed the best model among all that was constructed – Random Forest model. It helped to overcome the Decision Tree overfitting and still had the Decision Tree advantage of handling class imbalance rather well.