Final Project Presentation

Ricardo Oliveira
Mario Rincon
Richard Vo
KC Kim

The problem

Stakeholders

Information accessed could potentially replace/ help scouters of various types of sports (i.e take large amounts of data and give output on prominent future stars) not just including basketball. Of course given that data is provided.

Context

Similarly to predicting a good wine using machine learning, we will create a model using methods learned in class to scout for future prominent potential stars going to the NBA. Using the data and the ncaa statistics for each player.

- Columns such as three pointers, games played, points etc.

Problem statement

Does a good performance in the NCAA correlate to a good performance in the NBA, using a player's stats from their NCAA career?

Datasets - Players(data.world), All_seasons, Games, Games_details, nba_players, rankings, teams. (7 total)

Players stats from their time playing for NCAA and NBA from 1947 to 2018

Personal info

- Active years
- Birthdate
- College
- Position
- Weight
- Height
- Name

NCAA and NBA Performance

- 3 pts %
- 3 pts per game
- Effective field goals %
- Field goals %
- Field goals per game
- Free throw %
- Free throw per game
- Games
- Points per game

Method

Data Cleaning

This task includes reading in the data as well as cleaning the data from dropping, renaming, and dealing with NaN values so we can apply prediction models

Prediction Models

We used Linear Regression with L2 Regularization, Random Forest Regression, Support Vector Machines, and Neural Networks algorithms to predict **NBA performance stats**

Findings/Conclusions

Compare each method with evaluation metrics, such as MSE and R^2, from a test set to define the best algorithm to forecast basketball player performance based on NCAA and personal info

Data Cleaning

Load the dataset:

In [2]:
```
df = pd.read_csv('players.csv', index_col=0)
df
```

Out[2]:

	active_from	active_to	birth_date	college	height	name	position	url	weight	NBA__3ptapg	...	NCAA__3ptpg	NCAA_efgpct
0	1991	1995	June 24, 1968	Duke University	6-10	Alaa Abdelnaby	F-C	/players/a/abdelal01.html	240.0	0.0	...	0.0	NaN
1	1969	1978	April 7, 1946	Iowa State University	6-9	Zaid Abdul-Aziz	C-F	/players/a/abdulza01.html	235.0	NaN	...	NaN	NaN
2	1970	1989	April 16, 1947	University of California, Los Angeles	7-2	Kareem Abdul-Jabbar	C	/players/a/abdulka01.html	225.0	0.0	...	NaN	NaN
3	1991	2001	March 9, 1969	Louisiana State University	6-1	Mahmoud Abdul-Rauf	G	/players/a/abdulma02.html	162.0	2.3	...	2.7	NaN
4	1998	2003	November 3, 1974	University of Michigan, San Jose State University	6-6	Tariq Abdul-Wahad	F	/players/a/abdulta01.html	223.0	0.3	...	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...
4571	2018	2018	January 4, 1997	NaN	6-11	Ante Zizic	F-C	/players/z/zizican01.html	250.0	0.0	...	NaN	NaN
4572	1983	1983	December 20, 1953	Kent State University	7-1	Jim Zoet	C	/players/z/zoetji01.html	240.0	0.0	...	NaN	NaN
4573	1971	1971	June 7, 1948	Duquesne University	6-1	Bill Zopf	G	/players/z/zopfbi01.html	170.0	NaN	...	NaN	NaN
4574	2017	2018	March 18, 1997	NaN	7-1	Ivica Zubac	C	/players/z/zubaciv01.html	265.0	0.0	...	NaN	NaN

Data Cleaning

Rename some of the NBA columns to be consistent in naming with names of NCAA columns:

In [4]:
```
df = df.rename({'NBA_fg%': 'NBA_fgpct',
                'NBA_fg_per_game': 'NBA_fgpg',
                'NBA_fga_per_game': 'NBA_fgapg',
                'NBA_ft%': 'NBA_ftpct',
                'NCAA_ft': 'NCAA_ftpct',
                'NBA_ft_per_g': 'NBA_ftpg',
                'NBA_fta_p_g': 'NBA_ftapg',
                'NBA_g_played': 'NBA_games'
                }, axis=1)
# Lists of NBA and NCAA columns
NBA_columns = [c for c in df.columns if 'NBA' in c]
NCAA_columns = [c for c in df.columns if 'NCAA' in c]
# Sort lists of columns to have more consistency later
NBA_columns.sort()
NCAA_columns.sort()
# Move NBA_efgpct to the end, because there will be no corresponding NCAA column
NBA_columns.remove('NBA_efgpct')
NBA_columns.append('NBA_efgpct')
```

Data Cleaning

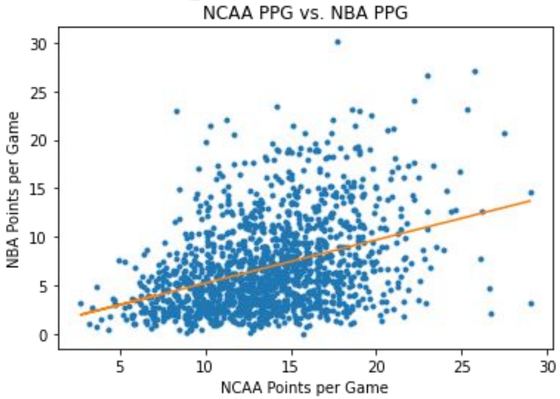
Data cleaning

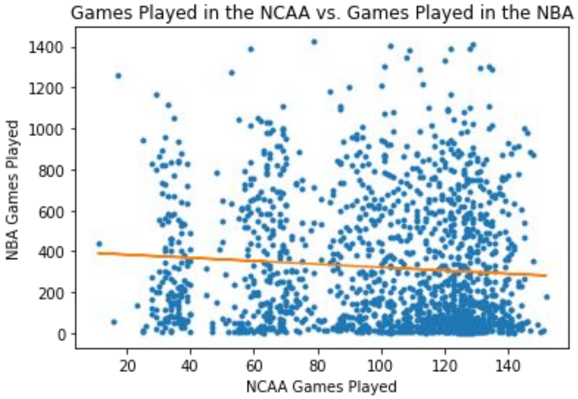
We can see that most of the columns contain null values. All values in the NCAA_efgpct column are null, therefore we can discard the whole column. There are 2598 non-null values in columns NCAA_games, NCAA_ppg and NCAA_ftpg. We will assume that players with null values in the NCAA_games column did not go to college and remove them completely from the dataset. Since we are interested in predicting performance in the NBA, we will also discard all rows for which at least one NBA column contains a missing value. The remaining missing values in the NCAA columns we will replace using the k-Nearest Neighbors approach - for each missing value, we will find the average value in that column of k other players that have most similar NCAA statistics.

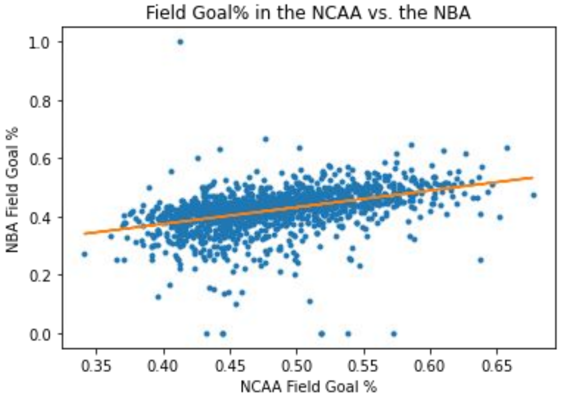
In [5]:

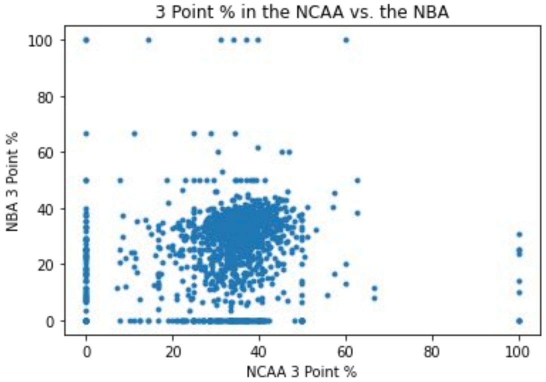
```
# Remove the all-null column
df.drop('NCAA_efgpct', axis=1, inplace=True)
NCAA_columns.remove('NCAA_efgpct')
# Remove rows with null values in NCAA_games
df = df[df['NCAA_games'].notnull()]
# Remove rows with any of NBA values missing
df = df[df[NBA_columns].notnull().all(axis=1)]
# Use KNNImputer with 10 neighbors to impute missing values
imputer = KNNImputer(n_neighbors=10)
df[NCAA_columns] = imputer.fit_transform(df[NCAA_columns])
```

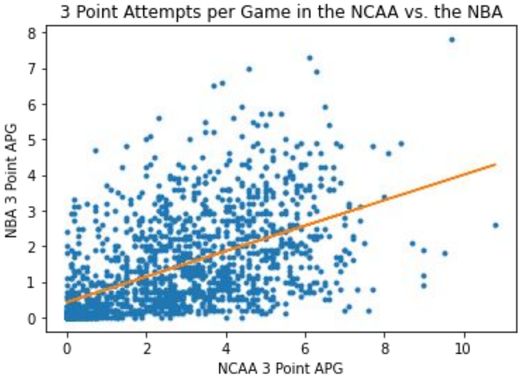
EDA and Visualization

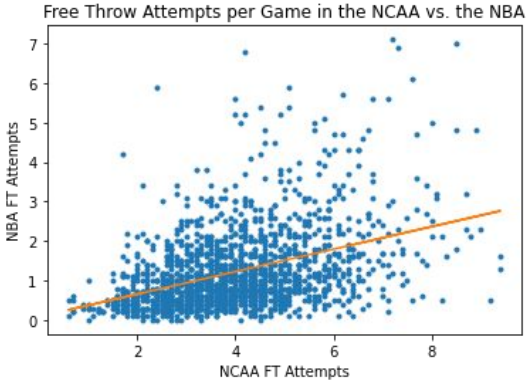




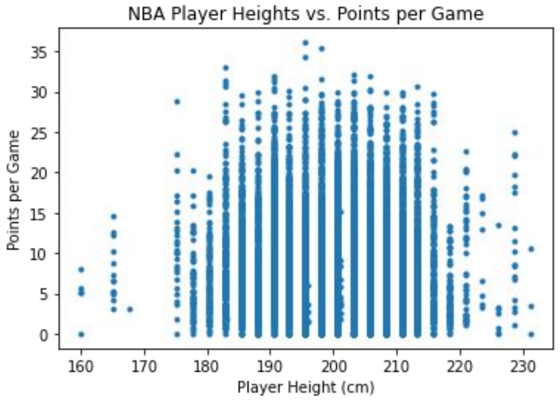


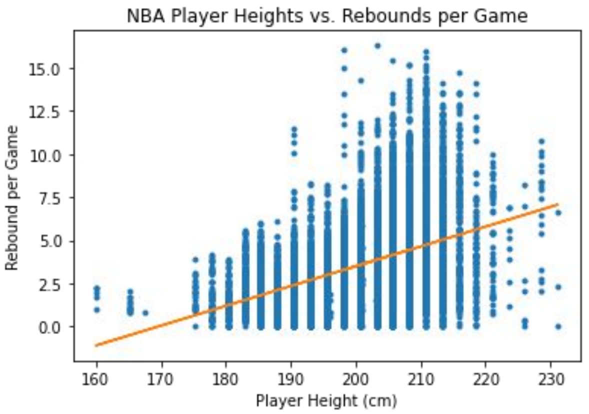


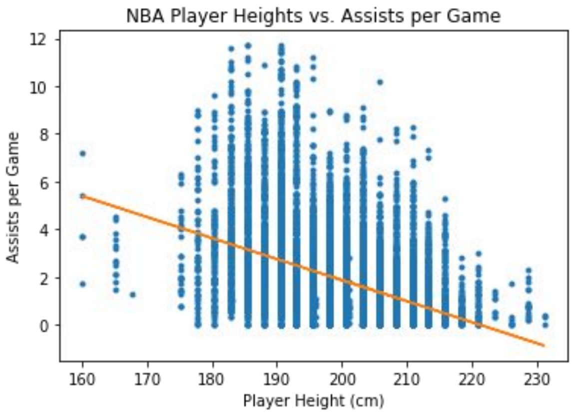




EDA and Visualization Continued

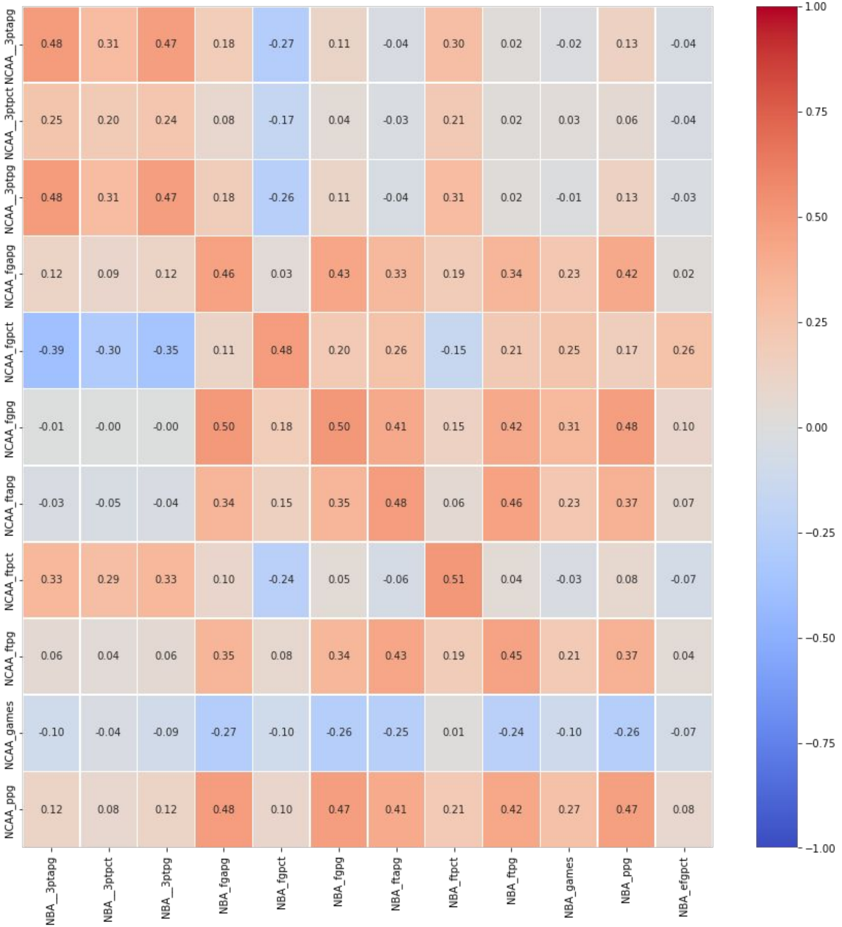






Correlation Analysis

- We used a heat map in a correlation matrix to identify highly correlated features that could potentially identicate collinearity in the models



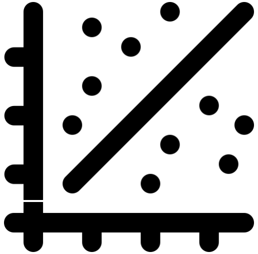
Predictive models

Evaluation metric	Linear regression	Random Forest	SVM	Neural Network
R^2 - mean	0.265	0.196	0.210	0.189
R^2 - std. dev.	0.084	0.067	0.083	0.099
MSE - mean	0.668	0.730	0.718	0.735
MSE - std. dev.	0.158	0.155	0.163	0.162

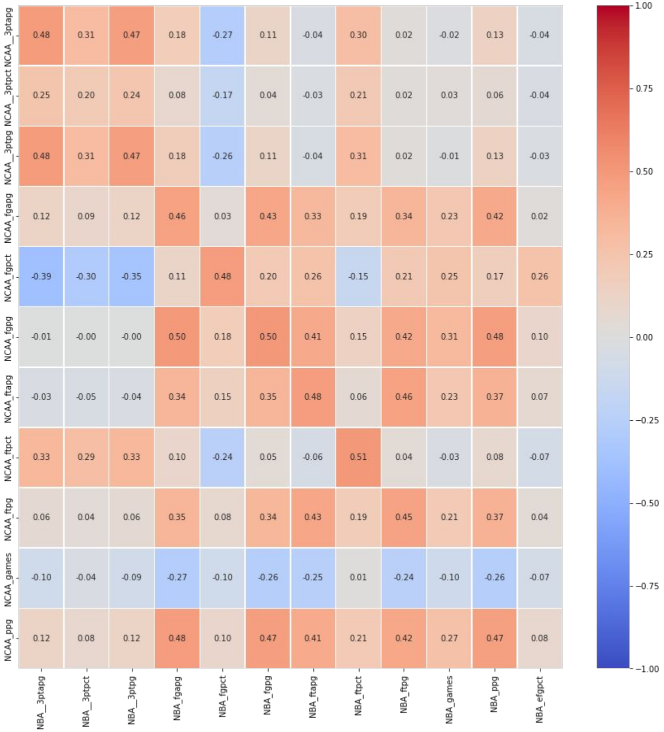
Predictive models

Linear Regression

Result & Conclusion



- High correlation indicates success to predict NBA stats from NCAA
- Collinearity within NCAA features points for redundancy and limitations in the models



Result & Conclusion

Linear Regression With L2 Regularization

```
[ ] # Use grid search cross-validation to find the best value of the paramater alpha
    model = GridSearchCV(Ridge(), param_grid = {'alpha': np.linspace(0.0, 2.0, 21)})
    model.fit(X_train, y_train)
    print('Best alpha parameter:', round(model.best_params_['alpha'], 2))
    y_pred = model.predict(X_test)
    stats_ridge = eval(y_test, y_pred, 'Linear Regression With L2 Regularization')
```

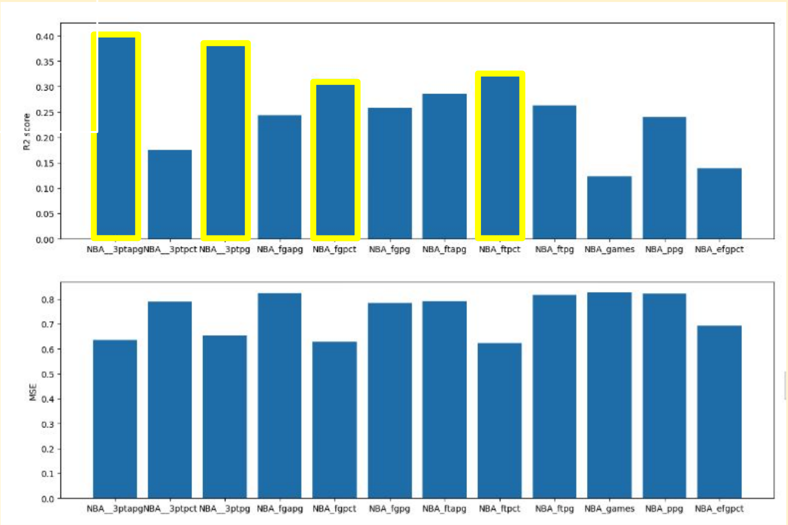
```
Best alpha parameter: 0.7
Average R2 score: 0.263
R2 score standard deviation: 0.085
Average MSE: 0.740
MSE standard deviation: 0.082
```

	NBA__3ptapg	NBA__3ptpct	NBA__3ptpg	NBA_fgapg
R2 score	0.405343	0.175598	0.384402	0.243308
Mean Squared Error	0.636253	0.789145	0.653166	0.824364

	NBA_fgpct	NBA_fgpg	NBA_ftapg	NBA_ftpct	NBA_ftpg
R2 score	0.313851	0.257589	0.285713	0.326494	0.263424
Mean Squared Error	0.627809	0.784100	0.790453	0.622194	0.816455

	NBA_games	NBA_ppg	NBA_efgpct
R2 score	0.12267	0.239419	0.139324
Mean Squared Error	0.82684	0.820584	0.691901

Linear Regression With L2 Regularization



Result & Conclusion

- Linear regression was the best predictor, but still with high MSE
- Every model had a high variance of accuracy of target columns
- We could try to improve the model by finding more independent features of NCAA performance

Thank you!