Machine Learning for FIFA Data

Implementing Machine Learning Principles to Gain Insights into the World of Soccer

The University of Texas at Dallas: BUAN 6340

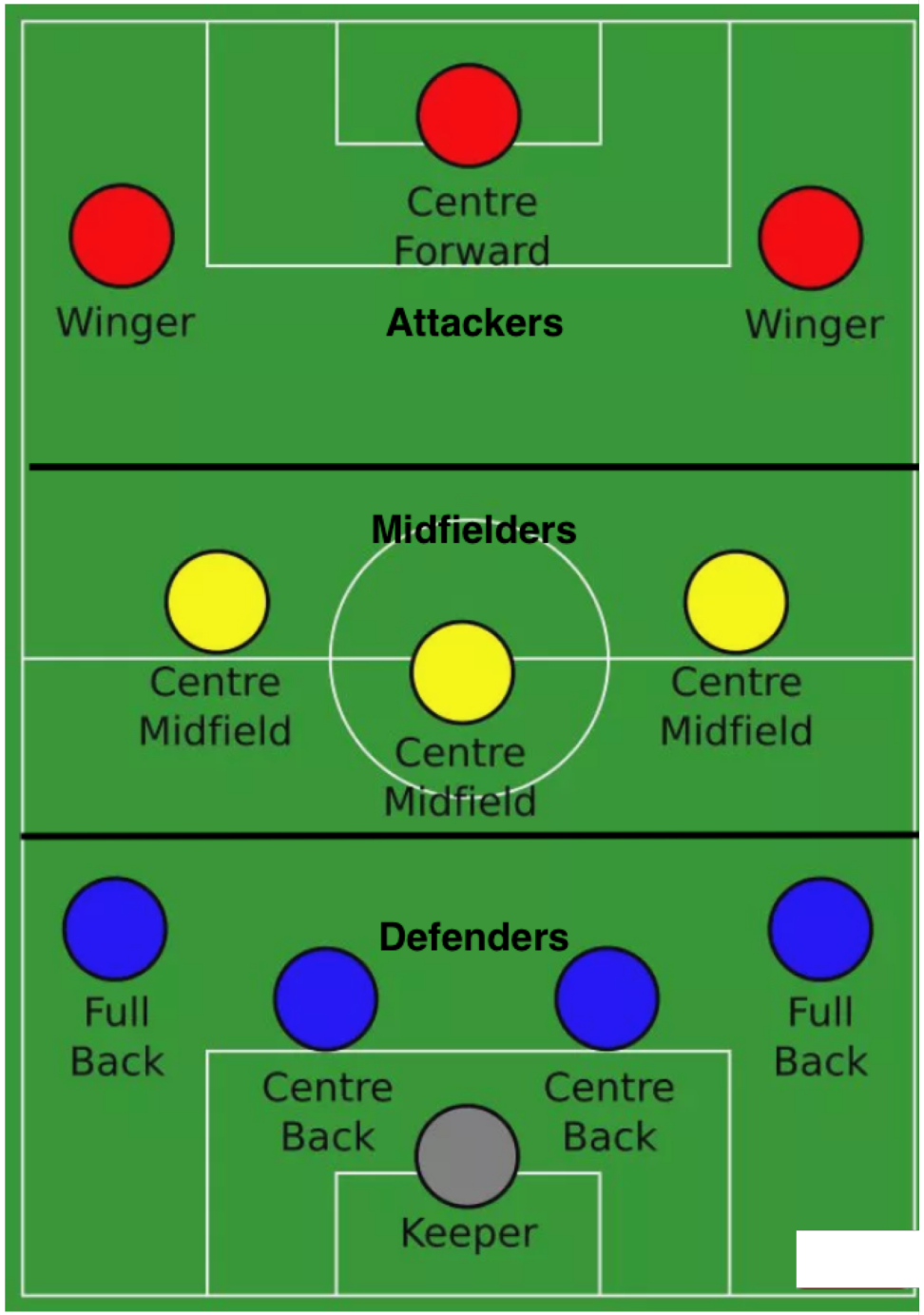
Team 3: Lucas Newman, Rohith Selvarajan, Megharjun Srinivasa

Purpose: Why use this Data?

- This dataset used in this project contains various metrics about every player in the FIFA 18 video game
- This dataset is viable for gaining insights on soccer players as it provides a simulated but, relevant and accurate view of each players characteristics and abilities
- There is a lot of data available from the FIFA video games, many standardized agreed upon metrics on how to judge and grade players
- Although the grading system isn't perfect, the simulated nature of the video game will be a more conducive environment for reinforcement learning, and deep learning models in future analysis
- All data was originally scraped from: *www. sofifa.com*

The Dataset and How it Was Created

- To see the origins of this FIFA data warehouse and how it was created please visit: https://github.com/Lucasnewman5732/FIFA_Analytics
- This data warehouse is roughly the merger of 3 different Kaggle datasets with certain variables modified and created
- Here is the link to the created data lake: https://www.kaggle.com/lucasnewman5732/buan6340project
- From this data warehouse the data tables that will be utilized are:
  - The main parent data set (with all players): "***Full_DF.csv***" and "full_fifa18_data.csv"
  - Data per position: "*goal_keepers.csv*", "*center_backs.csv*", "*full_backs.csv*", "center_mids.csv", "*wingers.csv*" , and lastly "*forwards.csv*"



Position Context:

- There are 4 position groups in soccer: midfielders, attackers, defenders and goal keepers
- In attack positions can be summed up as wingers(left/right wing) and forwards (center forwards and strikers)
- In midfield the positions can be summed up as a central midfielder(although there are also defensive and attacking midfielders)
- In defense positions can be summed up as center back or a fullback(left/right back)
- Lastly, the last line of defense is a goal keeper
- Note for the purposes of this project the positions have been divided into: wingers, forwards, center midfielders, center backs, full backs and goal keepers
- In soccer, the left/right footed nature of a person usually becomes more of a factor the wider their position is(i.e. fullbacks and wingers)

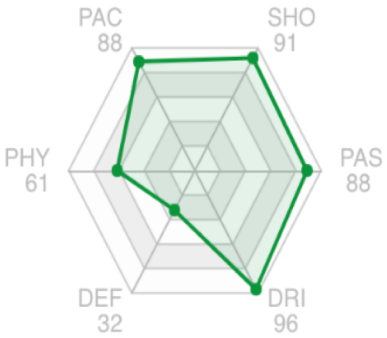
Data Reasoning: Main Data

L. Messi(ID: 158023)

Lionel Messi (Lionel Andrés Messi Cuccittini) CF RW ST Age 31 (Jun 24, 1987) 5'7" 159lbs

Overall Rating	Potential	Value	Wage
94	94	€110.5M	€565K

Preferred Foot Left
International Reputation 5★
Weak Foot 4★
Skill Moves 4★
Work Rate Medium/ Medium
Body Type Messi
Real Face Yes
Release Clause €226.5M



FC Barcelona	Argentina
86 ★★★★★	82 ★★★★½
Position RW	Position CF
Jersey Number 10	Jersey Number 10
Joined Jul 1, 2004	Joined Jul 1, 2004
Contract Valid Until 2021	Contract Valid Until 2021

#Dribbler #Distance Shooter #FK Specialist #Acrobat #Clinical Finisher #Complete Forward

Data Reasoning Continued: Features

Attacking

77	Crossing
95	Finishing
70	Heading Accuracy
90	Short Passing
86	Volleys

Skill

97	Dribbling
93	Curve
94	FK Accuracy
87	Long Passing
96	Ball Control

Movement

91	Acceleration
86	Sprint Speed
91	Agility
95	Reactions
95	Balance

Power

85	Shot Power
68	Jumping
72	Stamina
59	Strength
94	Long Shots

Mentality

48	Aggression
22	Interceptions
94	Positioning
94	Vision
75	Penalties
96	Composure

Defending

33	Marking
28	Standing Tackle
26	Sliding Tackle

Goalkeeping

6	GK Diving
11	GK Handling
15	GK Kicking
14	GK Positioning
8	GK Reflexes

Traits

Finesse Shot

Long Shot Taker (CPU AI Only)

Speed Dribbler

Playmaker (CPU AI Only)

One Club Player

Chip Shot (CPU AI Only)

Reasoning Continued

- These aggregated measures are based on a subset of measures such as the following
- We will use the more in depth measures as opposed to the aggregated card-measures

PACE		SHOOTING		PASSING		DRIBBLING		DEFENSE		PHYSICAL	
88		91		88		96		32		61	
Acceleration	91	Positioning	94	Vision	94	Agility	91	Interceptions	22	Jumping	68
Sprint Speed	86	Finishing	95	Crossing	77	Balance	95	Heading	70	Stamina	72
		Shot Power	85	Free Kick	94	Reactions	95	Marking	33	Strength	59
		Long Shots	94	Short Passing	90	Ball Control	96	Standing Tackle	28	Aggression	48
		Volleys	86	Long Passing	87	Dribbling	97	Sliding Tackle	26		
		Penalties	75	Curve	93	Composure	96				

Data Reasoning: Targets and Descriptive

- **Primary Target Variables**
  - `['Overall','Potential','Value (M)', 'ln_value', 'eur_value']`
  - Overall and Potential are ratings on a scale of (0-100)
  - In regards to value there are 3 measures used that reflect value: "**Value (M)**" is player value in millions of euros, "**eur_value**" is the value in euros, and "**ln_value**" is the *[ln(eur_value/100,000)+1]*
- **Descriptive Variables**
  - `['Name', 'ID', club','age', 'league', 'height_cm', 'weight_kg',club_id, league_id]`
  - *Note: to produce club and league ids a category coder was used "cat.codes"

Data Reasoning: Attribute Profile

- **Explanatory Variables which are the features focused on are ratings from (0-100)**:

```
attribute_profile=['age', 'height_cm', 'weight_kg', 'crossing',
'finishing', 'heading_accuracy',
      'short_passing', 'volleys', 'dribbling', 'curve',
'free_kick_accuracy',
      'long_passing', 'ball_control', 'acceleration', 'sprint_speed',
      'agility', 'reactions', 'balance', 'shot_power', 'jumping',
'stamina',
      'strength', 'long_shots', 'aggression', 'interceptions',
'positioning',
      'vision', 'penalties', 'composure', 'marking',
'standing_tackle',
      'sliding_tackle', 'gk_diving', 'gk_handling', 'gk_kicking',
      'gk_positioning', 'gk_reflexes']
```

Sample view of Full DataFrame

	Player_ID	ln_value	Name	Value (M)	Position	Overall	Potential	full_name	club	club_id	...	prefers_lcb	prefers_gk	foot_Left	foot_Right	att_rate_Hi
0	20801	6.862758	Cristiano Ronaldo	95.5	ST	94	94	C. Ronaldo dos Santos Aveiro	Real Madrid CF	467	...	False	0	0	1	
1	158023	6.957497	L. Messi	105.0	RW	93	93	Lionel Messi	FC Barcelona	220	...	False	0	1	0	
2	190871	7.115582	Neymar	123.0	LW	92	94	Neymar da Silva Santos Jr.	Paris Saint-Germain	433	...	False	0	0	1	
3	176580	6.878326	L. Suárez	97.0	ST	92	92	Luis Suárez	FC Barcelona	220	...	False	0	0	1	
4	167495	6.415097	M. Neuer	61.0	GK	92	92	Manuel Neuer	FC Bayern Munich	223	...	False	1	0	1	

5 rows × 190 columns

Basic Stats on Basic Attributes

- For at a glance ease the basic stats of the summarized card attributes have been calculated

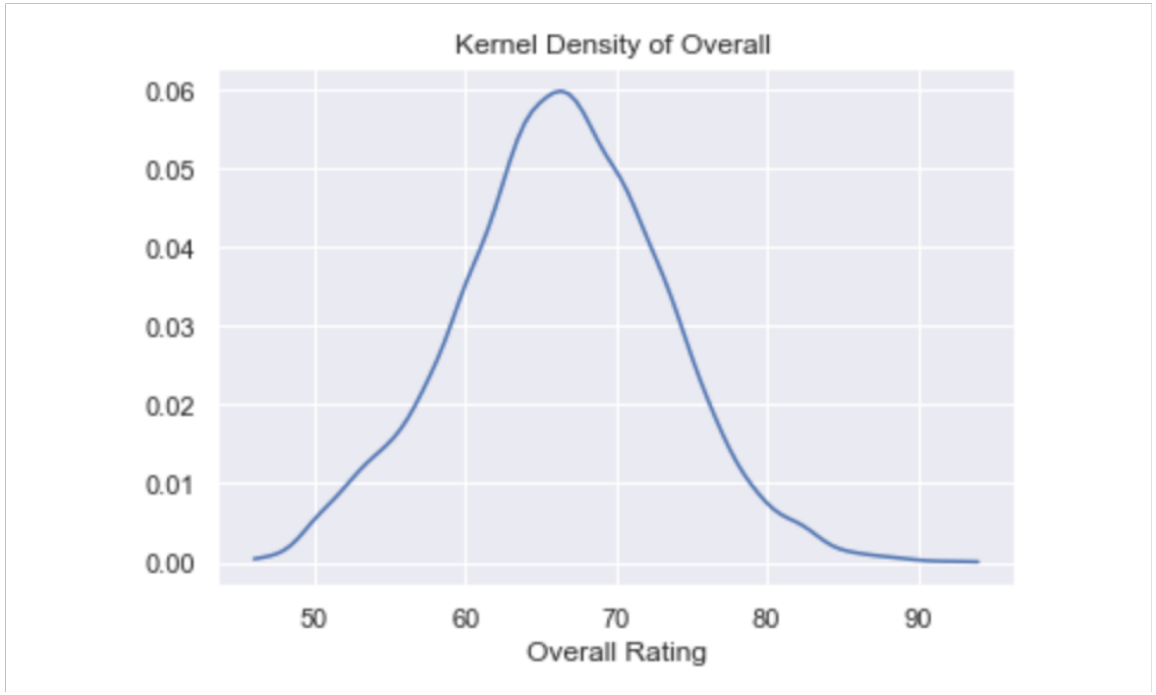
	count	mean	std	min	25%	50%	75%	max
pac	17739.0	67.800440	10.971605	21.0	61.0	68.0	75.0	96.0
sho	17739.0	53.595637	13.819056	14.0	44.0	56.0	64.0	93.0
pas	17739.0	57.650375	10.437276	24.0	51.0	59.0	65.0	95.0
dri	17739.0	62.696488	10.366483	24.0	57.0	64.0	70.0	96.0
defend	17739.0	49.495744	17.111116	12.0	34.0	52.0	64.0	90.0
phy	17739.0	64.874965	9.619584	27.0	59.0	66.0	72.0	92.0
reactions	17739.0	62.011387	9.115870	28.0	56.0	62.0	68.0	96.0
international_reputation	17739.0	1.123682	0.404942	1.0	1.0	1.0	1.0	5.0
skill_moves	17739.0	2.319578	0.748182	1.0	2.0	2.0	3.0	5.0
weak_foot	17739.0	2.949884	0.662293	1.0	3.0	3.0	3.0	5.0

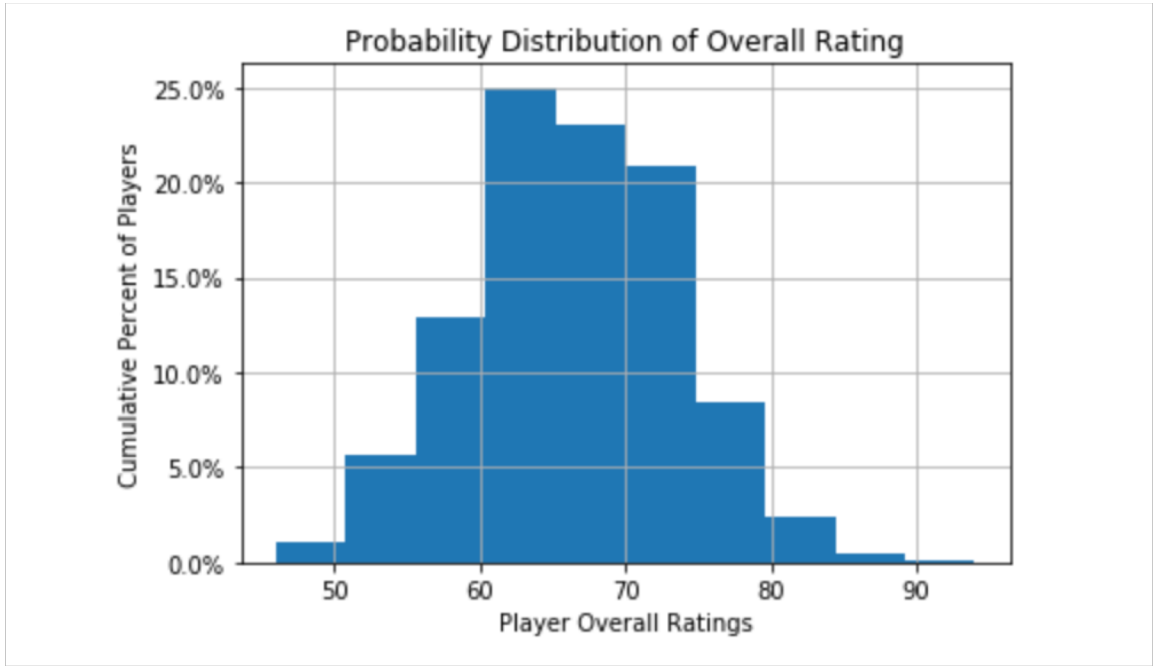
Basic Stats on Descriptive Features and Targets

	count	mean	std	min	25%	50%	75%	max
age	17739.0	25.160494	4.593598	16.0	22.0	25.0	28.0	47.0
height_cm	17739.0	181.277299	6.690376	155.0	177.0	181.0	186.0	205.0
weight_kg	17739.0	75.428660	6.996046	49.0	70.0	75.0	80.0	110.0

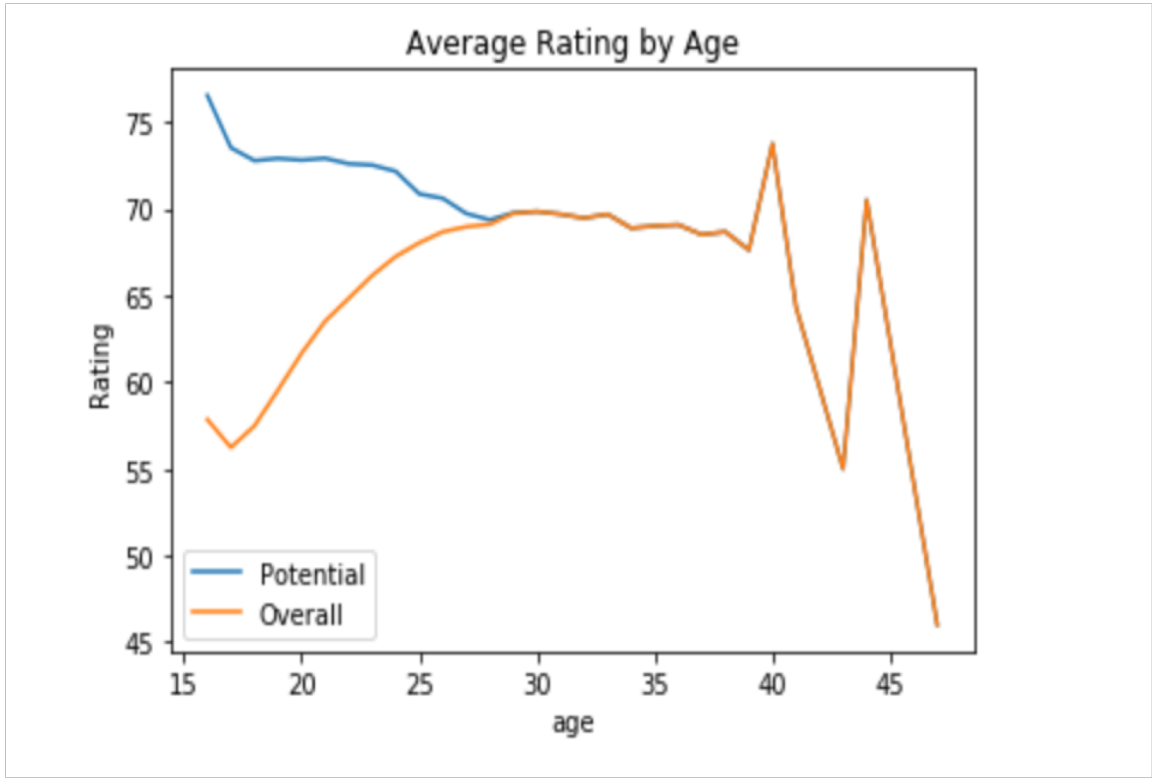
	count	mean	std	min	25%	50%	75%	max
Overall	17739.0	6.631687e+01	6.940958e+00	46.0	62.000000	66.000000	7.100000e+01	9.400000e+01
Potential	17739.0	7.121923e+01	6.103628e+00	46.0	67.000000	71.000000	7.500000e+01	9.400000e+01
Value (M)	17739.0	2.405757e+00	5.382428e+00	0.0	0.325000	0.700000	2.100000e+00	1.230000e+02
ln_value	17739.0	2.309136e+00	1.236539e+00	0.0	1.446919	2.079442	3.091042e+00	7.115582e+00
eur_value	17739.0	2.414659e+06	5.394812e+06	0.0	325000.000000	700000.000000	2.100000e+06	1.230000e+08

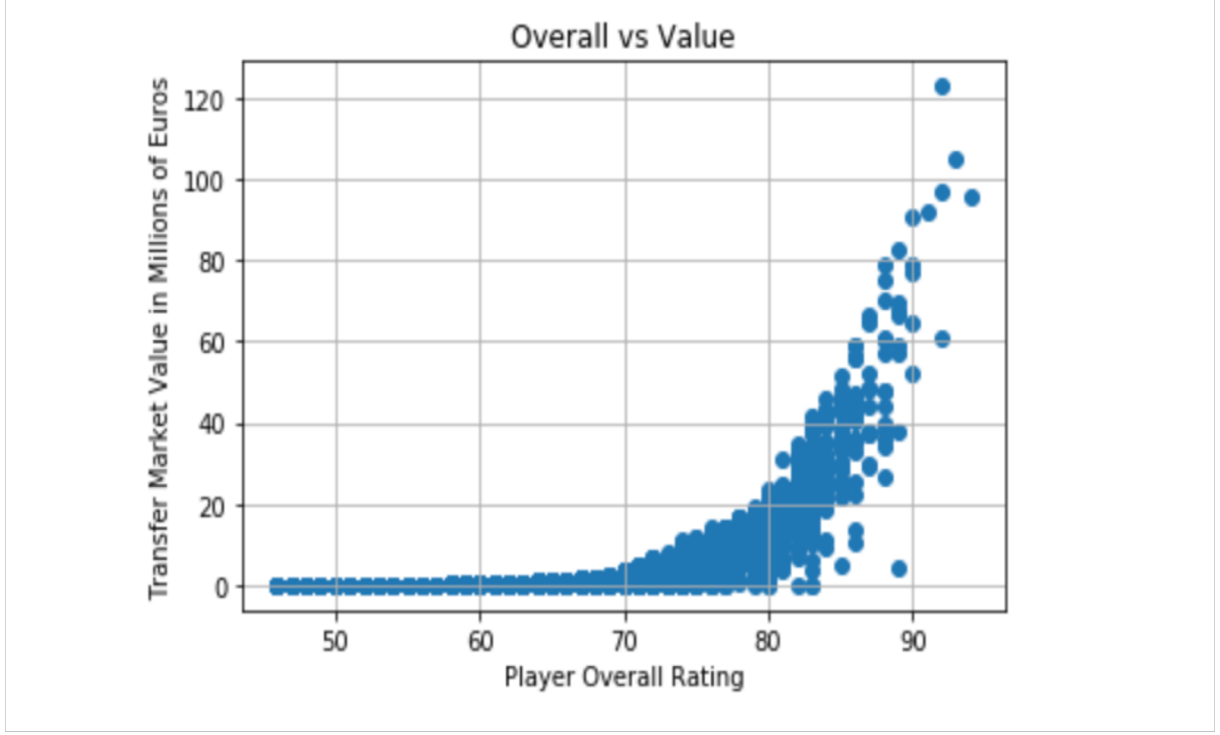
Distribution of Overall Rating



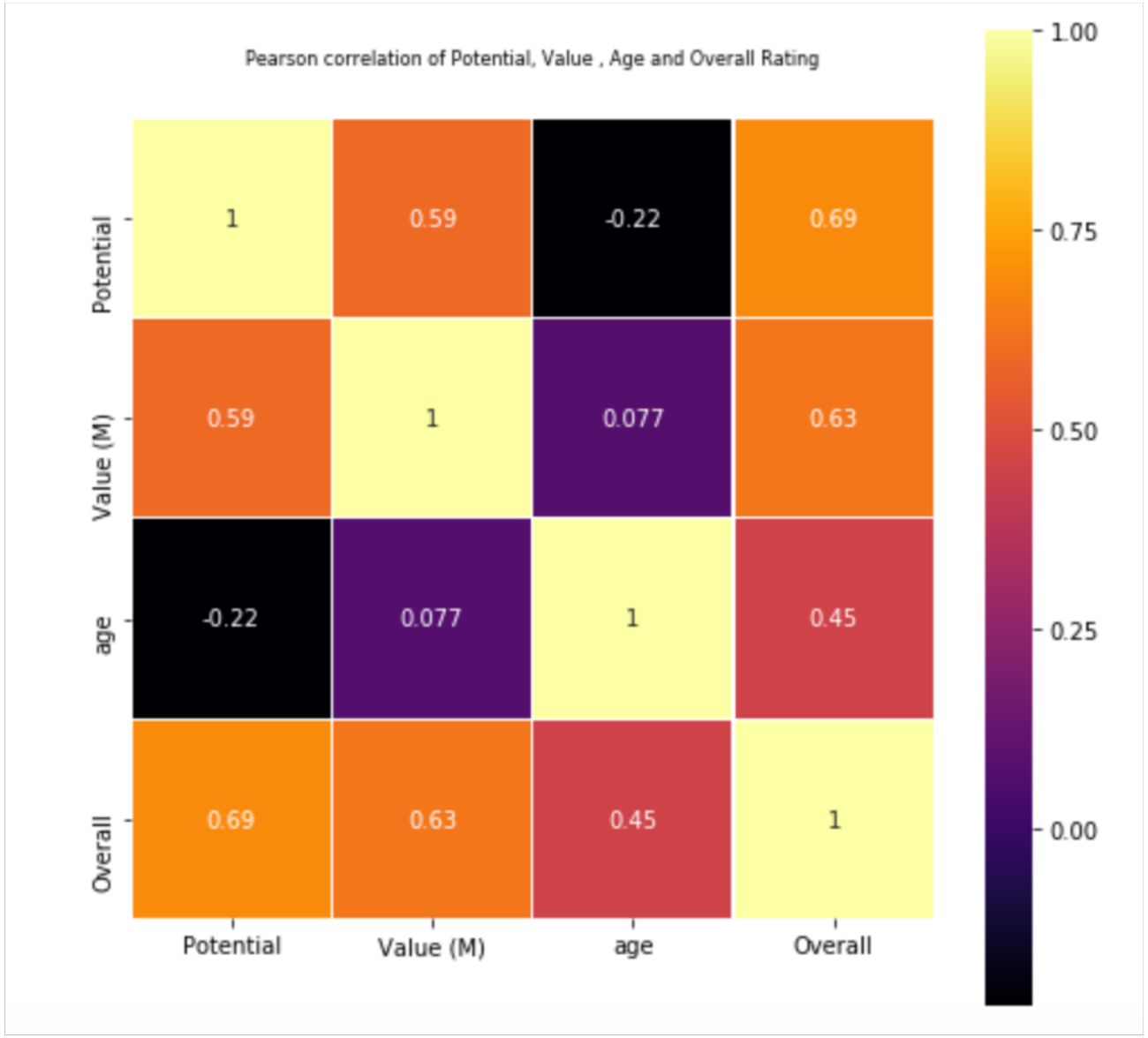


Overall vs Value & Change in Rating with Age





Pearson Correlation of Target Features with Age



Lucas Newman

Purpose, Motivation, & Critical Assumptions

- Purpose: To gain meaningful insights into the world of soccer by utilizing the features provided in the dataset

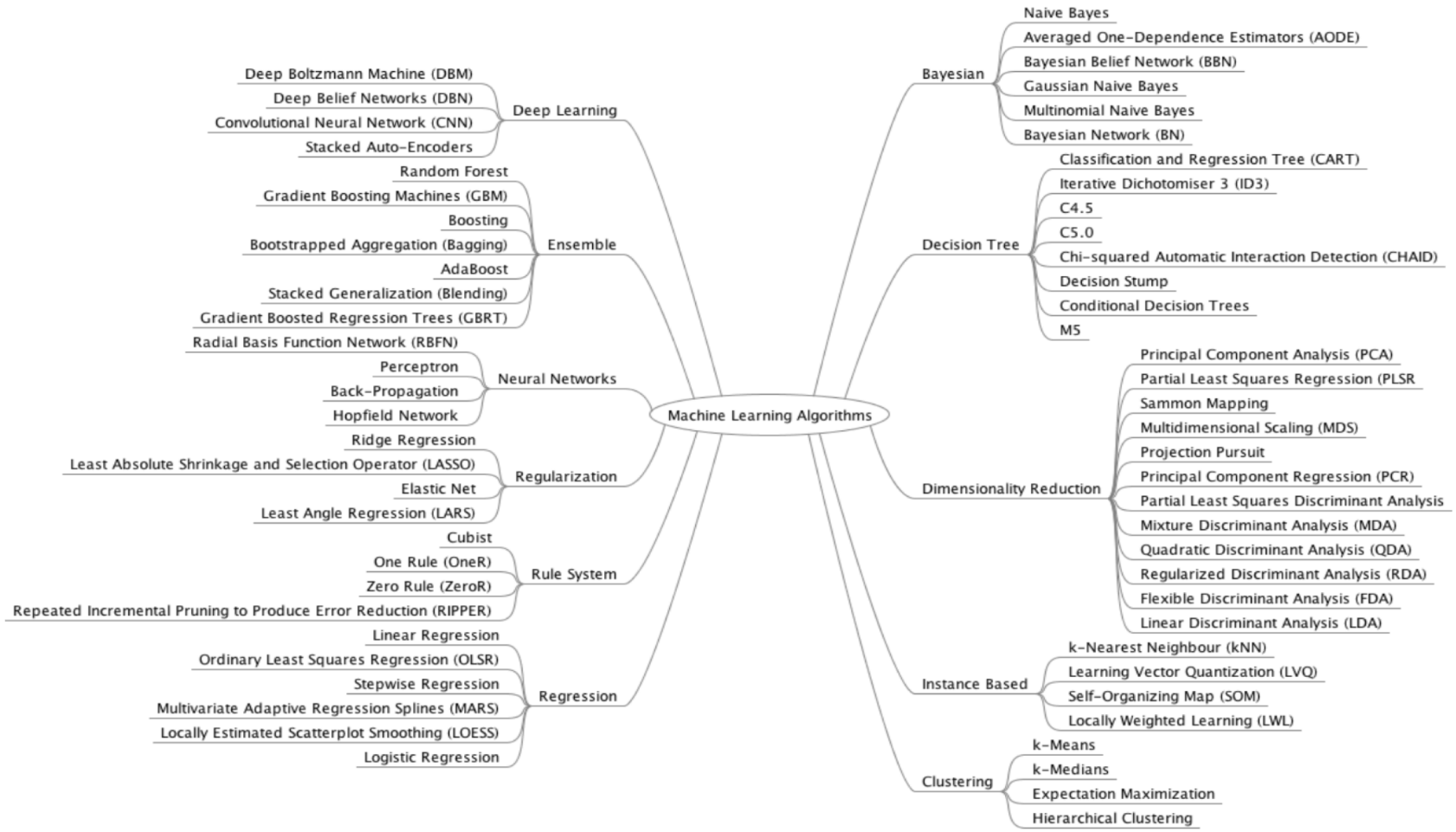
- Underlying assumption 1: Overall rating is an aggregated calculation based on all the player features previously mentioned
- Underlying assumption 2: Overall and Potential are highly correlated
- Underlying assumption 3: Overall and primary drivers of Value

- **When referred to X is attributes which are in "attribute_profile"**

Primary Research Questions

- Question 1: What sorts of basic relationships exist between our features?
- Question 2: What drives overall rating and potential?
- Question 3: What features are most important in determining these calculations?
- Question 4: Can an improved position-based value model be created?

Machine Learning Universe



Ref: Jason Brownlee 2013

Methods Used

- Regression
  - OLS with Robust SE and Gaussian GLM with Elastic Net
  - Lasso Regression
  - Ridge Regression
  - Robust Regression with Huber's M-Regression
- Classification
  - kNN instance based classifier
  - Bayesian classification
  - SVM
  - Kernel SVM
  - Logistic Regression
  - LDA
- Clustering
  - k-means clustering

Methods Used cont.

- Ensemble
  - Random Forrest Classification
  - Gradient Boosting Regression (Gradient Boosting Machines)
- Dimensionality Reduction
  - PCA
  - Lasso
  - Feature Ranking with GBM
- Model Tuning and Validation
  - Lasso CV to choose alpha
  - Ridge CV to choose alpha
  - k-folds cross validation on classification and regression models
  - Hyperparameter tuning with GridSearchCV for feature Importance
  - Confusion matrix for classifiers
- Feature Engineering

Regression Approach: Overall Rating

- OLS regression with Robust SE was used on all players to see which features were statistically significant at $\alpha = 5\%$
- Robust regression with Huber's Maximum likelihood estimation and comparison of statistically significant features with results from OLS
- Since we assume multicollinearity we conduct ridge regression with a weight=1 and observe the coefficients
- We conducted lasso regression with various α as a means of feature reduction
- Using L1 and L2 regularization helped us in preventing multicollinearity

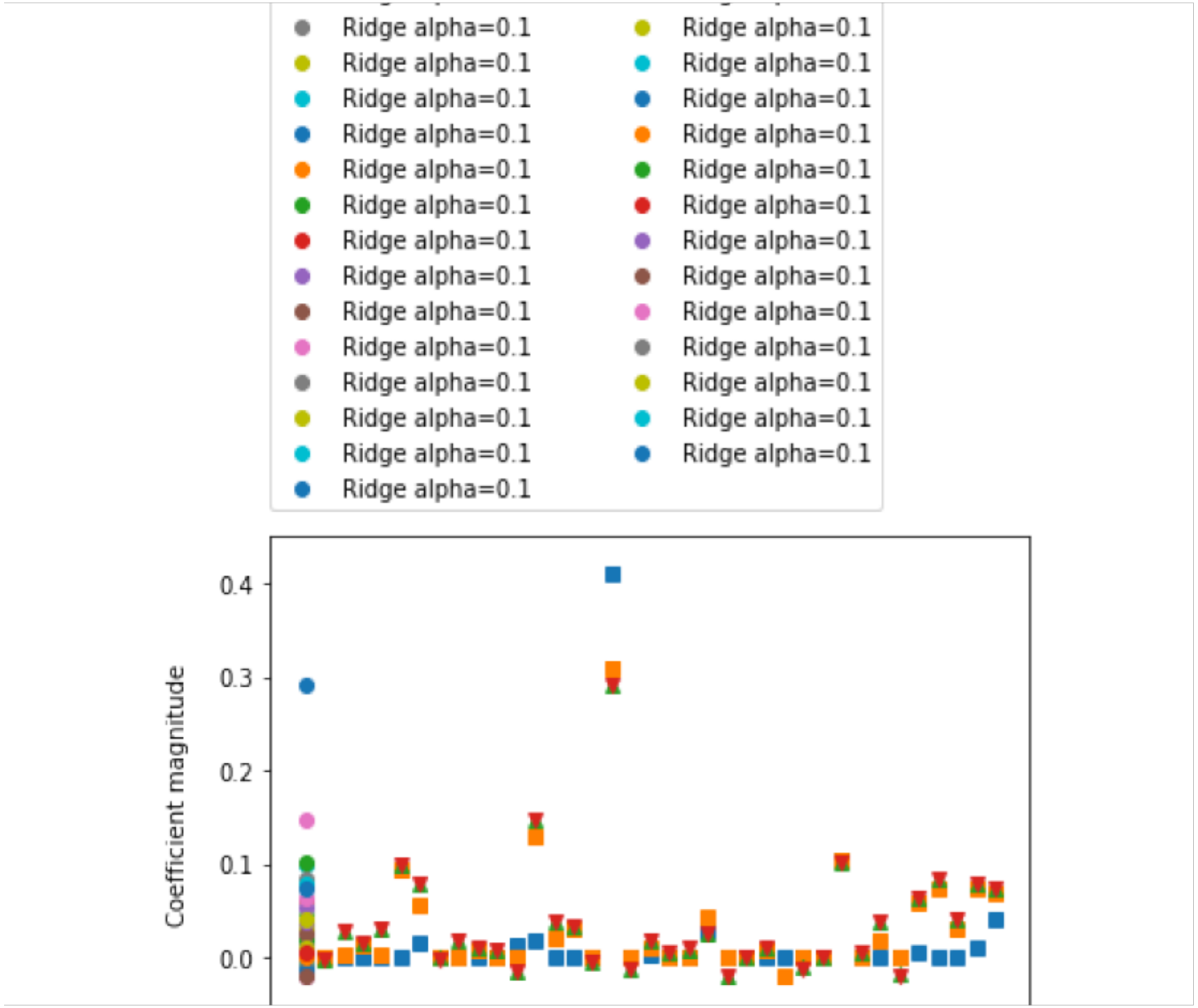
Challenges with Regression

- Heteroskedacity
- Multiple collinearity
- Overfitting
- **However, multiple collinearity was our primary challenge to overcome**

Regression Results Overall

- OLS model R-squared results: 0.845
- Ridge with α=1 R-square results=0.845
- Lasso with α=1 R-square results=0.845
- Lasso with α=8 and reduced features=0.8

Regression Results: Overall



Regression Approach: Value

- Purpose: what features are most important to determining value by position?
- When OLS used multicollinearity was an issue so we solved it by:
- Reduce amount of features by using lasso regression
- To find what value of alpha to use for lasso regression we used LassoCV, then we choose all non-zero features
- We calculated the Pearson correlations between the non-zero features
- We made a reduced model with the non-zero features and conducted ridge regression to deal with collinearity

Regression Approach Value cont

- To find the optimal alpha for ridge regression we used RidgeCV
- We calculated the R-square, RMSE, and conducted k-Folds CV on our reduced model for the ridge regression
- We then used gradient boosting regression and GridSearchCV on our reduced model and ranked the features
- Using our most important features we created interaction terms between the highly correlated features then re-conducted regression on value and ln_value (value elasticity) now that multicollinearity has been eliminated

Overcoming Multicollinearity: Procedure Summarized

- →To overcome collinearity a recursive process of lasso tuning->lasso regression->forming reduced model→
- →calculating correlation between features in reduced model→
- →ridge tuning-> ridge regression-> model scoring and k-folds cross validation→
- →gradient boosting tuning on reduced model->gradient boosting regression→
- → making a new model with interaction terms between correlated features→
- →reconducting regression on a reduced model not that collinearity is eliminated

Regression Value Model Results:

- The goal was to see if we could come up with a reduced and improved model to predict value and value elasticity for all positions and for all positions we succeeded

Regression Value Model Example: Winger

Out[519]:
```
Ridge(alpha=79.01, copy_X=True, fit_intercept=True, max_iter=None,
   normalize=False, random_state=None, solver='auto', tol=0.001)

[-37.96628836]
                0          1
0             age  -0.338955
1       height_cm  0.0354684
2       weight_kg -0.0150212
3        crossing  0.0405286
4       finishing  0.0567221
5   short_passing  0.0972522
6       dribbling  0.0548051
7    ball_control    0.12961
8    acceleration  0.0501735
9    sprint_speed -0.0163293
10      reactions   0.141411
11        stamina  0.0124369
12     long_shots -0.0305993
13    positioning  0.0484458
14         vision  0.0315826
15      composure   0.065729
```

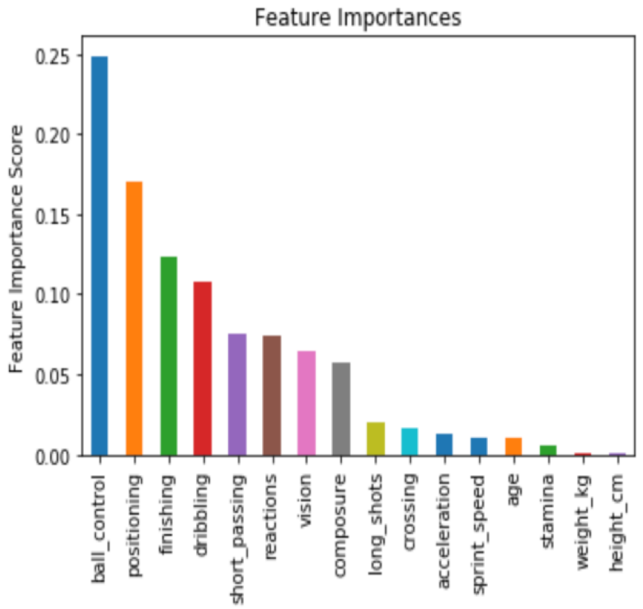
```
The training and test scores respectively are:  0.44352486274701625 and 0.5149970405114312

 Root Mean Square Error of test set is: 4.00026697162954

 K-folds cross validation scores: [  0.43742721  -8.6296417    0.50575815 -11.95888424   0.35889686]
```

GBR Tuned with GridSearch

```
Model Report
RMSE : 1.151
CV Score : Mean - 1.456 | Std - 1.165 | Min - 0.2721 | Max - 3.746
```




```python
def modelfit(alg, dtrain, features, performCV=True, printFeatureImportance=True, cv_folds=10):
    #Fit the algorithm on the data

    alg.fit(dtrain[features],dtrain["Value (M)"] )

    #Predict training set:
    dtrain_predictions = alg.predict(dtrain[features])

    #Perform cross-validation:
    cv_score =  cross_val_score(alg, dtrain[features], dtrain["Value (M)"], cv=cv_folds,
                                                scoring='neg_mean_squared_error')
    cv_score = np.sqrt(np.abs(cv_score))

    #Print model report:
    print ("\nModel Report")
    print ("RMSE : %.4g" % np.sqrt(metrics.mean_squared_error(dtrain["Value (M)"], dtrain_predictions)))
    print ("CV Score : Mean - %.4g | Std - %.4g | Min - %.4g | Max - %.4g" % (np.mean(cv_score),
                                                                              np.std(cv_score),np.min(cv_score),
                                                                               np.max(cv_score)))

    if printFeatureImportance:
        feat_imp = pd.Series(alg.feature_importances_, features).sort_values(ascending=False)
        feat_imp.plot(kind='bar', title='Feature Importances')
        plt.ylabel('Feature Importance Score')
```

New Model Definition for Winger

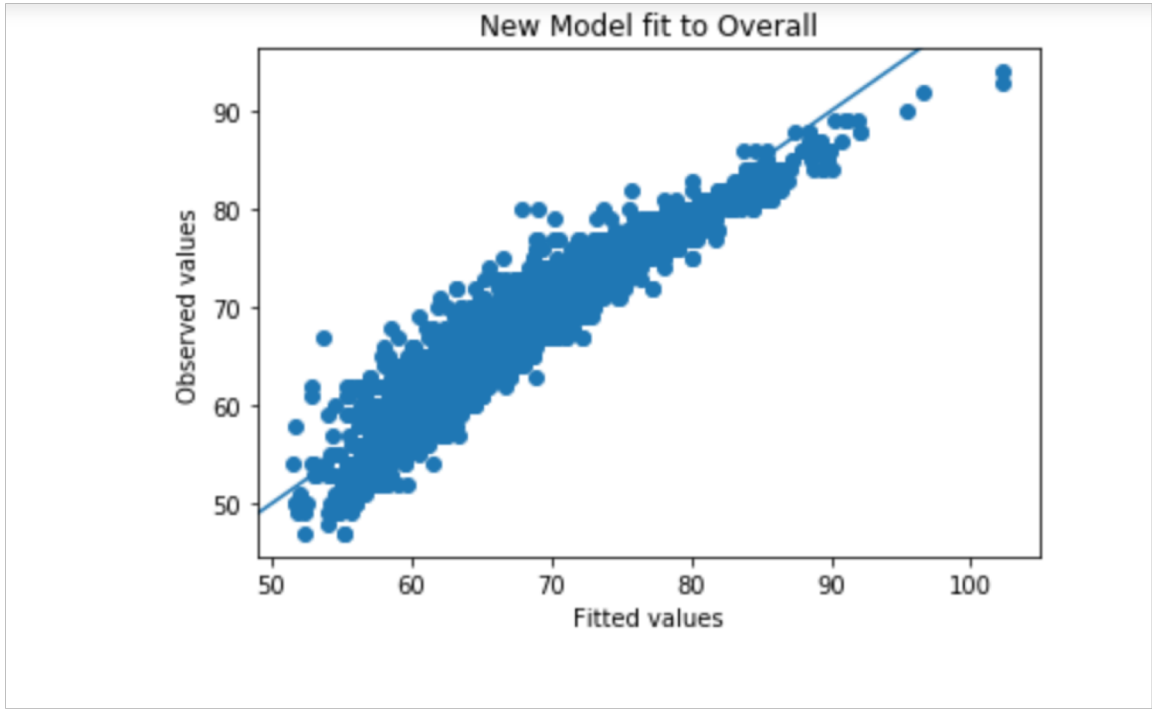
```
#Create our New Model with Interaction Terms

def w_skills (row):
    return row['short_passing'] * row['ball_control']  *row['dribbling']
def w_skills_off (row):
    return row['positioning']*row['finishing']
def w_ath_ment (row):
    return row['stamina'] * row['acceleration'] *row['sprint_speed']*row['composure'] *row['reactions'] * row['visi

#def w_mentality (row):
    #return row['composure'] *row['reactions'] * row['vision']

def w_phy (row):
    return row['height_cm'] * row['weight_kg']
```

Results of New Winger Model



OLS Regression Results

Dep. Variable:	Value (M)	R-squared:	0.601
Model:	OLS	Adj. R-squared:	0.601
Method:	Least Squares	F-statistic:	240.9
Date:	Mon, 26 Nov 2018	Prob (F-statistic):	3.06e-237
Time:	20:58:00	Log-Likelihood:	-18863.
No. Observations:	6685	AIC:	3.774e+04
Df Residuals:	6679	BIC:	3.778e+04
Df Model:	5		
Covariance Type:	HC0		

	coef	std err	z	P>|z|	[0.025	0.975]
const	3.1022	0.050	62.372	0.000	3.005	3.200
age	-1.1215	0.055	-20.209	0.000	-1.230	-1.013
w_skills	2.4689	0.126	19.569	0.000	2.222	2.716
w_skills_off	0.3022	0.081	3.729	0.000	0.143	0.461
w_ath_ment	2.8084	0.209	13.455	0.000	2.399	3.218
w_phy	0.2306	0.048	4.770	0.000	0.136	0.325

Omnibus:	8926.764	Durbin-Watson:	0.548
Prob(Omnibus):	0.000	Jarque-Bera (JB):	3249589.748
Skew:	7.372	Prob(JB):	0.00
Kurtosis:	110.000	Cond. No.	3.50

Warnings:
[1] Standard Errors are heteroscedasticity robust (HC0)

Classification Approach

- We try to predict player position with the attribute profile using k-NN classification
- We now think of overall as categorical data where the overall rating is a category (0-100)
- We use all the features to try and exactly predict overall
- For classification we tried Random Forrest, Bayesian classification, SVM, kernel SVM, logistic regression, and LDA
- We scored these classifiers to identify which is most performant for future analysis

Classification Results

- The k-NN classifier for prediction accurately predicted position roughly 45% of the time
- Of the other classification models tested on predicting exact overall rating the models that performed best were random Forrest and kernel SVM
- Random Forrest predicted 43% of the overall ratings where as kernel SVM predicted 28% of overall ratings

Random Forrest Example

```
# Fitting Random Forest Classification to the Training set
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators = 100, criterion = 'gini')
classifier.fit(X_train, y_train)
predictions = classifier.predict(X_test)
from sklearn.metrics import confusion_matrix
cm = pd.DataFrame(confusion_matrix(y_test, predictions))
cm.head()

#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics
# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, predictions))
from sklearn.metrics import mean_squared_error
RMSE_test=sqrt(mean_squared_error(y_true=y_test,y_pred=predictions))
print("Root Mean Square Error of test set is:" ,RMSE_test)
```

Random Forrest vs Kernel SVM Results

Random Forrest

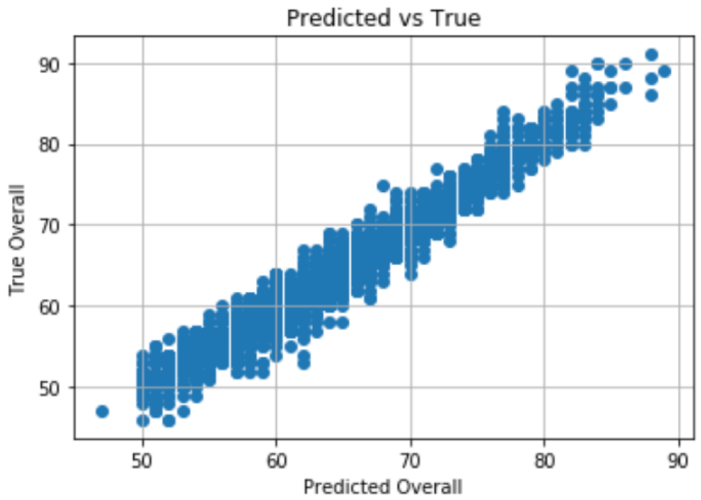
```
Accuracy: 0.4323562570462232
Root Mean Square Error of test set is: 1.4740649046443295

Out[471]: <matplotlib.collections.PathCollection at 0x258b43a978>

Out[471]: Text(0.5, 0, 'Predicted Overall')

Out[471]: Text(0, 0.5, 'True Overall')

Out[471]: Text(0.5, 1.0, 'Predicted vs True')
```



Kernel SVM

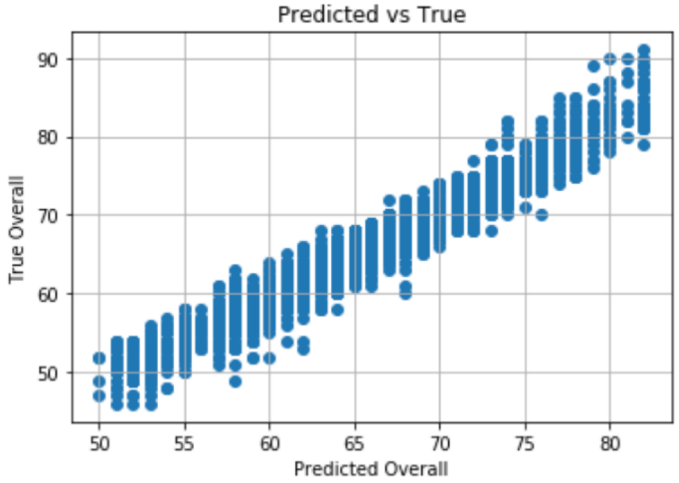
```
Accuracy: 0.2856069146937242
Root Mean Square Error of test set is: 1.715974857226964

Out[498]: <matplotlib.collections.PathCollection at 0x258a55a2b0>

Out[498]: Text(0.5, 0, 'Predicted Overall')

Out[498]: Text(0, 0.5, 'True Overall')

Out[498]: Text(0.5, 1.0, 'Predicted vs True')
```



Why Did Random Forrest and Kernel SVM Work Best?

- When we change overall to a category that needs to be predicted the presence of position makes the model non linear
- Random Forrest:
  - Powerful and accurate, good performance on many problems, including non linear problems
- Kernel SVM:
  - High performance on nonlinear problems, not biased by outliers, not sensitive to overfitting

Clustering Approach

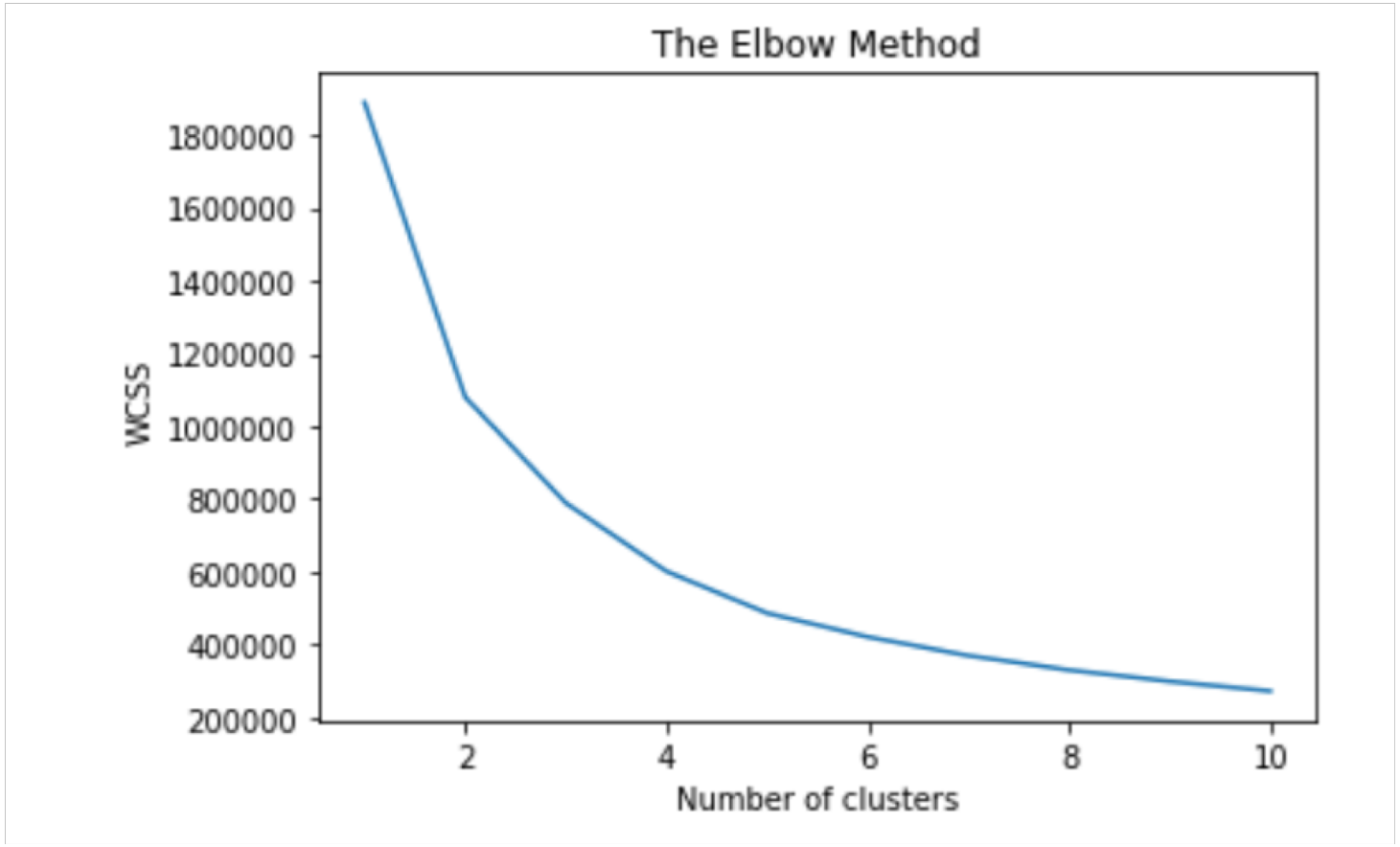
- Used k-means to establish clusters between the following features:
  - [‘Overall’, 'pac', 'sho', 'dri', 'pas', 'defend', 'phy', 'reactions’]
  - Why k-means: Simple to understand, easily adaptable, works well on small or large datasets, fast, efficient and performant
  - Con of k-means: we have to identify the right number of clusters

- Used elbow method to find ideal number of clusters in this case the ideal number of clusters was

```
from sklearn.cluster import KMeans
wcss = []
for i in range(1, 11):
        kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42)
        kmeans.fit(X)
       wcss.append(kmeans.inertia_)
```

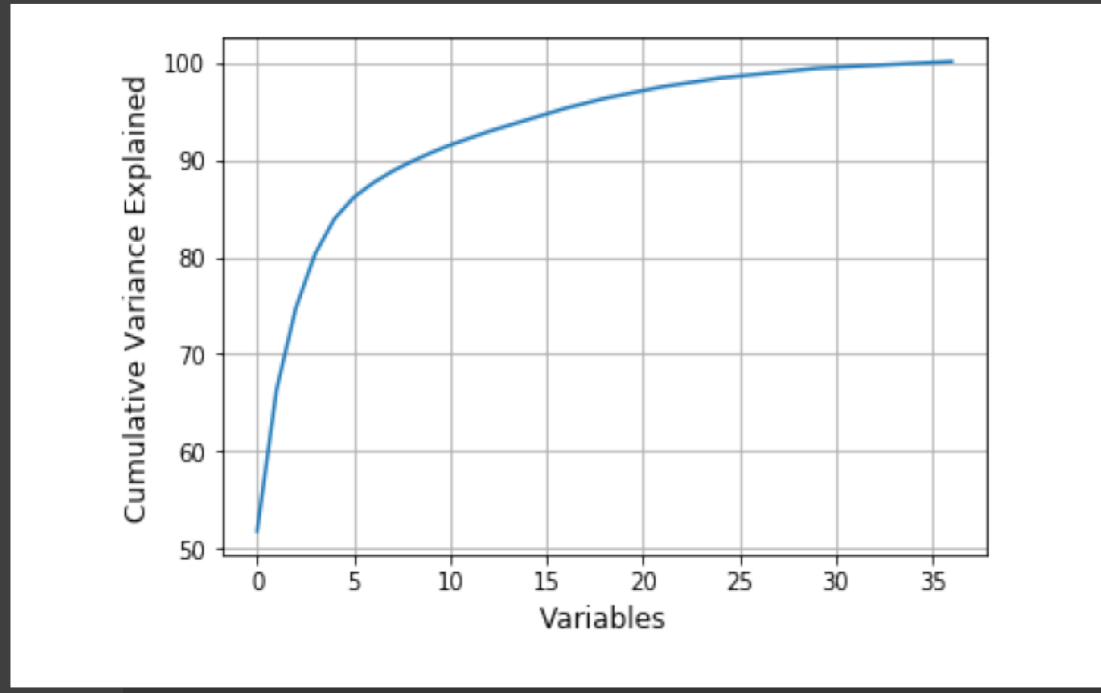
Clustering Results

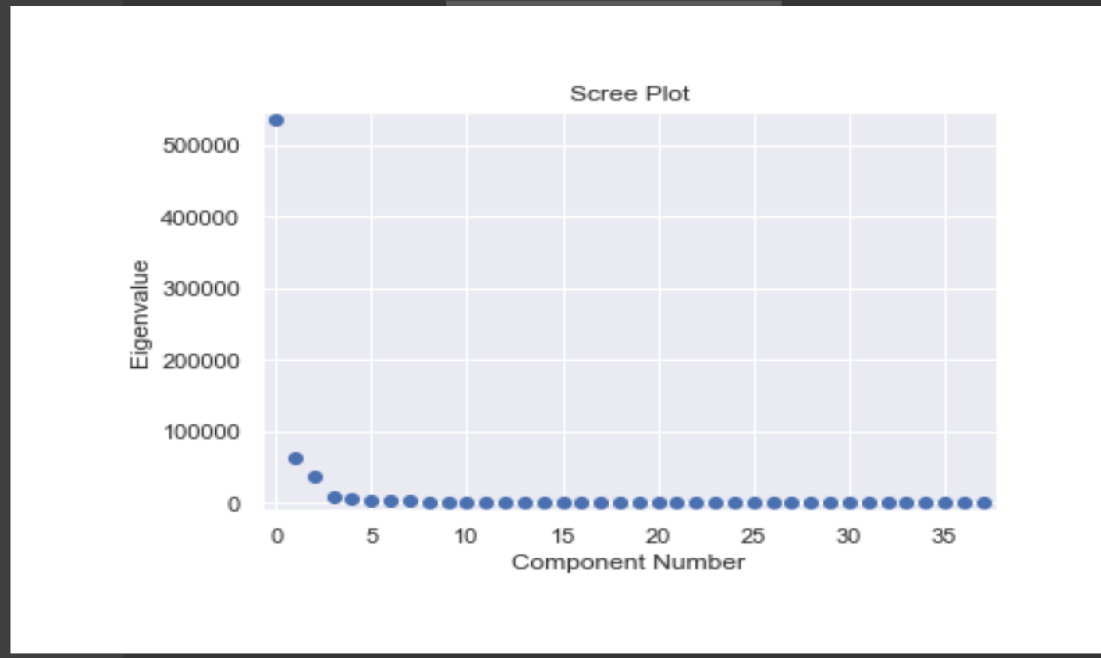
- Results of Elbow Method



Dimensionality Reduction Approach

- Utilized PCA on all features to compare to feature reduction by Lasso





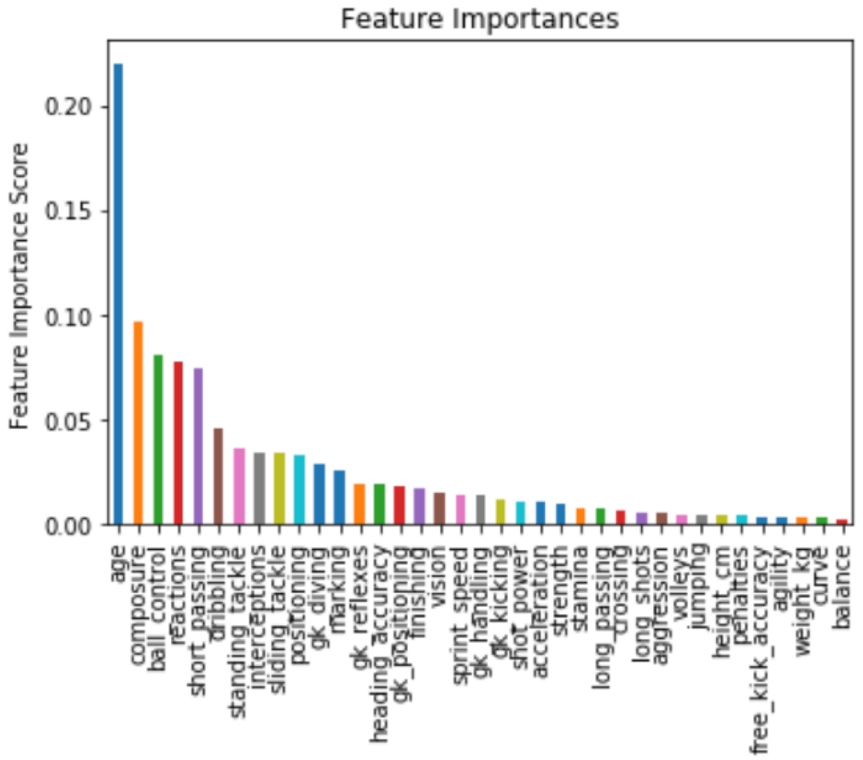
Ensemble Methods Used

- Implemented Gradient boosting regression with hyperparameter tuning done by GridSearch cross validation to rank features in regards to Overall rating and potential
- Conducted Random forest classification of overall

Example of Ensemble Methods Results

- Feature Ranking of Potential rating using gradient boosted regression

```
Model Report
RMSE : 1.244
CV Score : Mean - 2.289 | Std - 0.8957 | Min - 1.55 | Max - 4.226
```



Model Tuning and Validation Approach

- For model tuning in regression approaches we used k-folds cross validation
- For model tuning in our approaches to dimensionality reduction in lasso and ridge, we determined the value of alpha with RidgeCV and LassoCV
- For our hyperparameter tuning in the feature ranking models we used GridSearchCV
- For model tuning in our clustering we used to elbow method to determine the number of clusters

Conclusion

- In this analysis we were successfully able to get insights into what makes a good player and what determines their market value.
- We conducted dimensionality reduction with PCA and tuned lasso regression.
- We successfully created enhanced value models by position and conducted regression analysis on value and the elasticity of value.
- We created an enhanced and reduced value model that eliminates collinearity with feature engineering, and interaction terms as well as out performs the original model.
- We identified the optimal classification methods to predict a players overall rating.
- We came up with our own algorithm to rank features and used GridSearchCV for hyperparameter tuning.
- We clustered players by basic features.
- The implications of our results can help train AI systems on what to look for when scouting players, and help clubs financially structure contracts based on player attributes.

Future Methods

- Conduct further modeling with classification algorithms as opposed to seeing which model just performs best
- The primary focus of our future analysis will be regarding the development of a recommender system that recommends players to teams based on team style, budget, and current players on a team
  - Explore alternating least squares as a recommender system
- We will seek to use deep learning and reinforcement learning to identify future top prospects faster than scouts can so that a competitive advantage will be achieved

Further Commentary

References

[1] Kumar, Gunjan. (2013). Machine Learning for Soccer Analytics. 10.13140/RG.2.1.4628.3761.

[2] VanderPlas, J. T. (2017). Python Data Science Handbook: Essential Tools for Working with Data. Sebastopol, CA: OReilly Media.

[3] Müller, A. C., & Guido, S. (2017). Introduction to Machine Learning with Python: A Guide for Data Scientists. Beijing: OReilly.

[4] Hastie, T., Tibshirani, R., & Friedman, J. H. (2017). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. New York, NY: Springer.

[5] Bishop, C. M. (2016). Pattern Recognition and Machine Learning. New York, NY: Springer-Verlag.

Q/A

- Are there any Questions?