Study of FIFA 22 Players' Wage and Affecting Factors

STAT 447 Group Project

Lorena Lu(yutingl7) Yi Yang(yiy14) Yufei Duan(yufeid3) Zean Li(zhaolin4)
Zhongwen Shen(zs30)

Contents

1 Introduction **2**

2 Setup **2**

3 Visualization **3**

4 Prediction **10**
4.1 Linear Regression 10
4.2 Lasso Regression 16

5 Summary **21**

1 Introduction

The FIFA World Cup will be held in Qatar in late 2022. And this competition is one of the grandest international contests, attracting people around the world. We have noticed that there is a trend that some star players' wage is much higher than the average wage, and certain potential players have their wage below market value compared to other players, and thus resulting in the uneven situation among the player trading market and the basic standard of the expected value. Therefore, we plan to analyze what factors influence FIFA soccer players' wages. We use dataset from "FIFA 2022", an EA game which is an up-to-date reflecction on real world FIFA, to perform our analysis.

In this project, we first plot graphs of several factors that we believe would affect one player's wage, such as ages, league level, and overall scores, etc. Then, we try to train the dataset to figure out the most influential factors of their annual salary based on their performance data in 2022 via various methodologies like linear regression and lasso regression. We will mainly focus on their objective physial skills and try to obtain a prediction model for their wage based on these factors.

2 Setup

```r
## load libraries

library(readr)
library(dplyr)
library(ISLR2)
library(glmnet)
library(ggplot2)
library(tidyverse)
library(data.table)
```

```r
## load dataset from local file location

player_general = read.csv("https://raw.githubusercontent.com/illinois-stat447/fa22-prj-yutingl7-zs30-zha
```

```r
## convert dataset into data.table or tibble for better operation

p_gen = as_tibble(player_general)
```

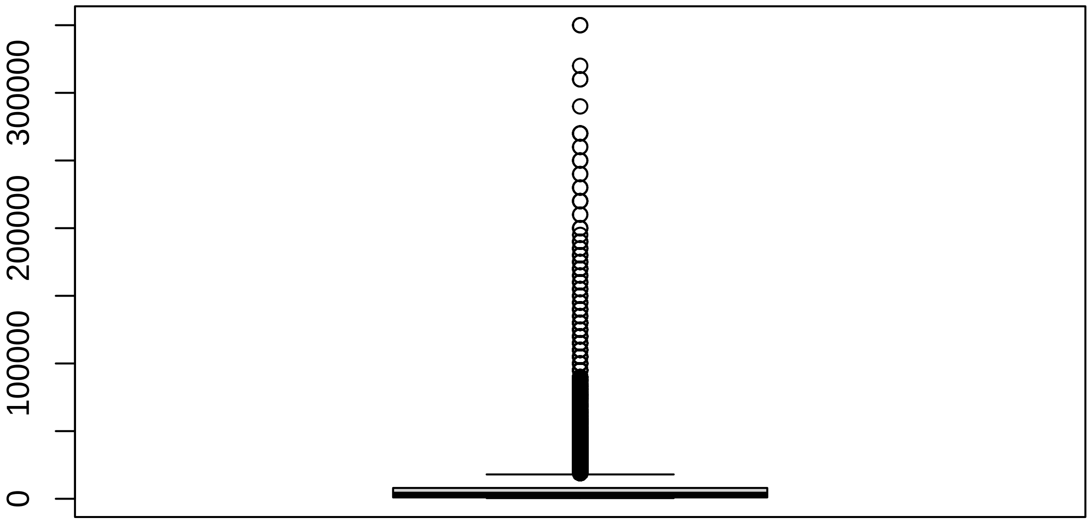
The "p_gen" dataset contains most of the personal information we need to analyze what factors influenced different player's wage, including their height, age, potentials, etc.

3 Visualization

```
summary(p_gen$wage_eur)
```

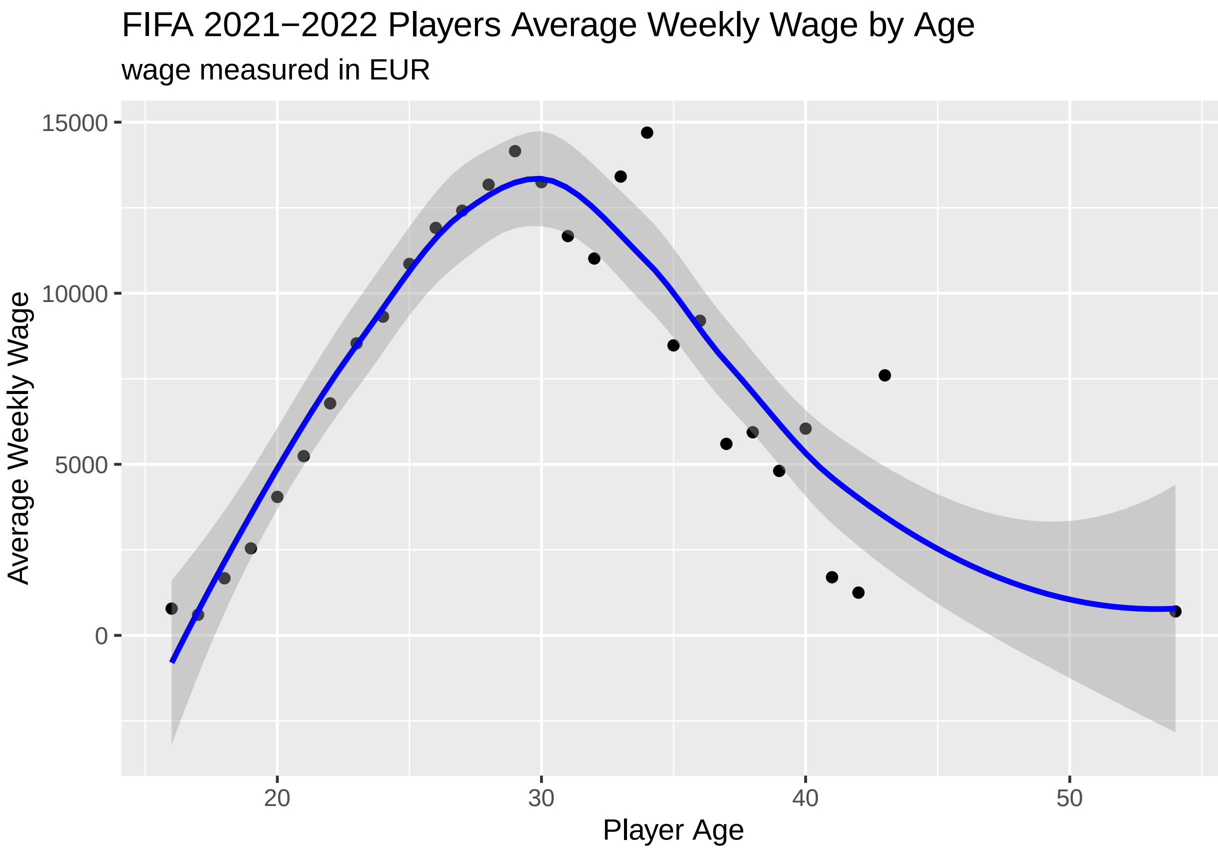
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##     500    1000    3000    9018    8000  350000      61
```

```
boxplot(p_gen$wage_eur)
```



We present a summary and a box plot of the average weekly wage. We can clearly see there are plenty of outliers in the boxplot. The third quartile is only €8000, but the max value is €350000. This is a huge difference and also represents an extreme income inequality. We will find out what factors caused this phenomenon.

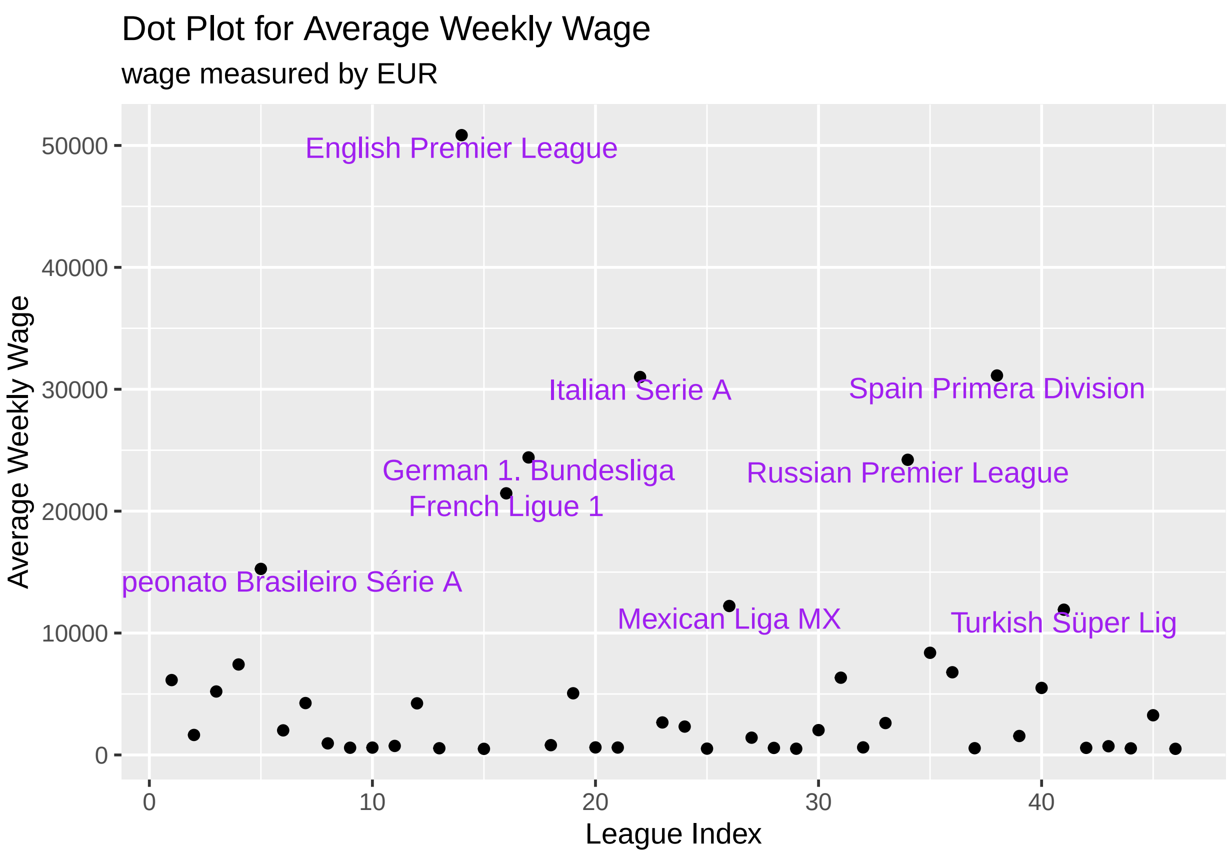
```
## plot weekly wage by players' age, and then analyze the factors that affect
## their wage, such as their potential

p_gen_age = p_gen |>
  group_by(age) |>
  summarise(wage = mean(na.omit(wage_eur)), .groups = "drop")

ggplot(data = p_gen_age) +
  geom_point(aes(x = age, y = wage), color = "black") +
  geom_smooth(aes(x = age, y = wage), method = "loess", formula = y ~ x,
              color = "blue") +
  ggtitle("FIFA 2021-2022 Players Average Weekly Wage by Age",
          subtitle = "wage measured in EUR") +
  xlab("Player Age") +
  ylab("Average Weekly Wage")
```



We first considered age factor, so we grouped the p_gen dataset by age and calculated the average wage of different age region. We can clearly see from the plot that the average wage increases at first and reach the peak at the age around 30, and the wage gradually goes down as the age decreased. We can also conclude an almost linear relationship between age and average wage, with a higher standard error from 40 to 50. This may due to their potential, which is their abilities, that varies their wage.

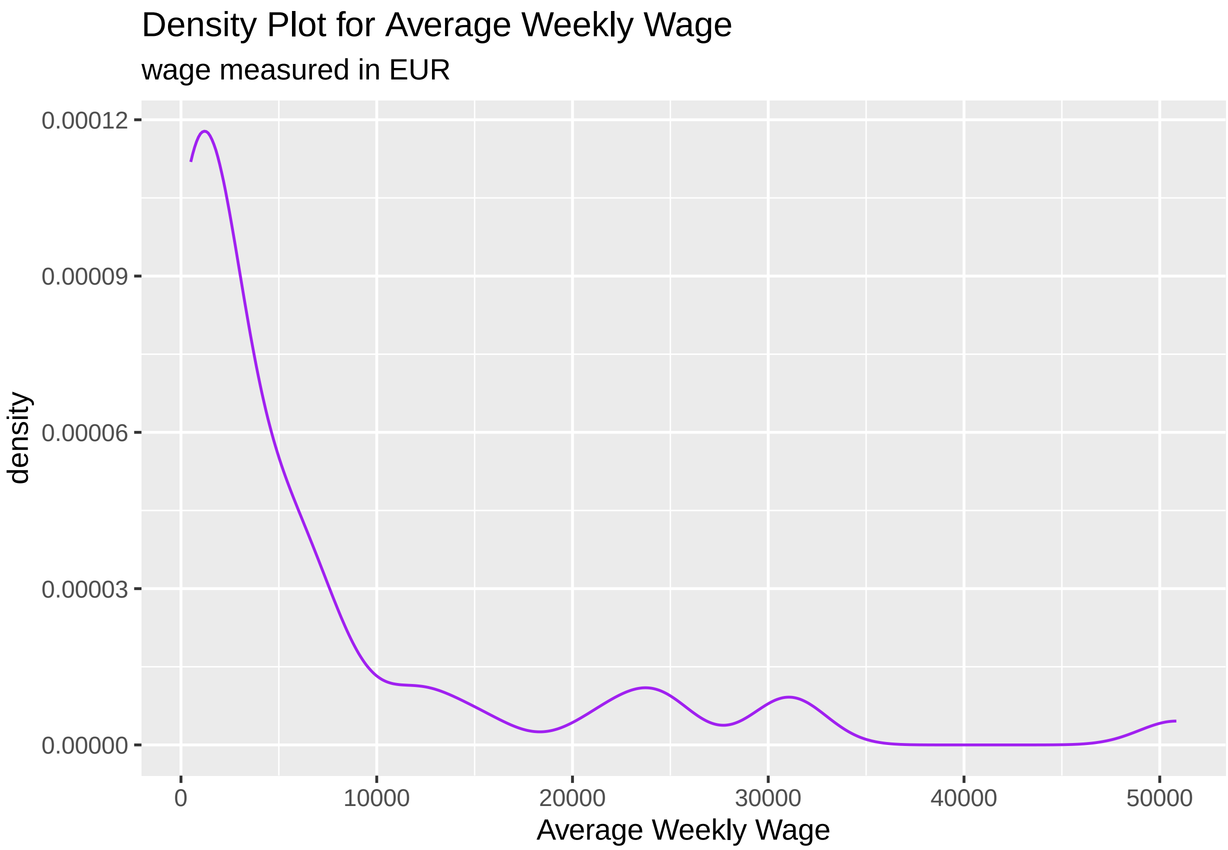
```
## plot weekly wage by players' league
##

p_gen_lea_avg = p_gen |>
  drop_na(league_name, wage_eur, league_level) |>
  filter(league_level == 1) |>
  select(league_name, wage_eur) |>
  group_by(league_name) |>
  summarise(avg_wage = mean(wage_eur),
            tot_wage = sum(wage_eur),
            med_wage = median(wage_eur),
            .groups = "drop")

p_gen_lea_avg_order = tibble::rowid_to_column(p_gen_lea_avg, "index")

p_gen_lea_avg_order_filter = p_gen_lea_avg_order |>
  filter(avg_wage >= 10000) |>
  mutate(avg_wage = avg_wage - 1000)

ggplot(data = p_gen_lea_avg_order) +
```

```r
  geom_point(aes(x = index, y = avg_wage)) +
  geom_text(data = p_gen_lea_avg_order_filter,
            aes(x = index, y = avg_wage, label = league_name), color = "purple") +
  ggtitle("Dot Plot for Average Weekly Wage", subtitle = "wage measured by EUR") +
  xlab("League Index") +
  ylab("Average Weekly Wage")
```



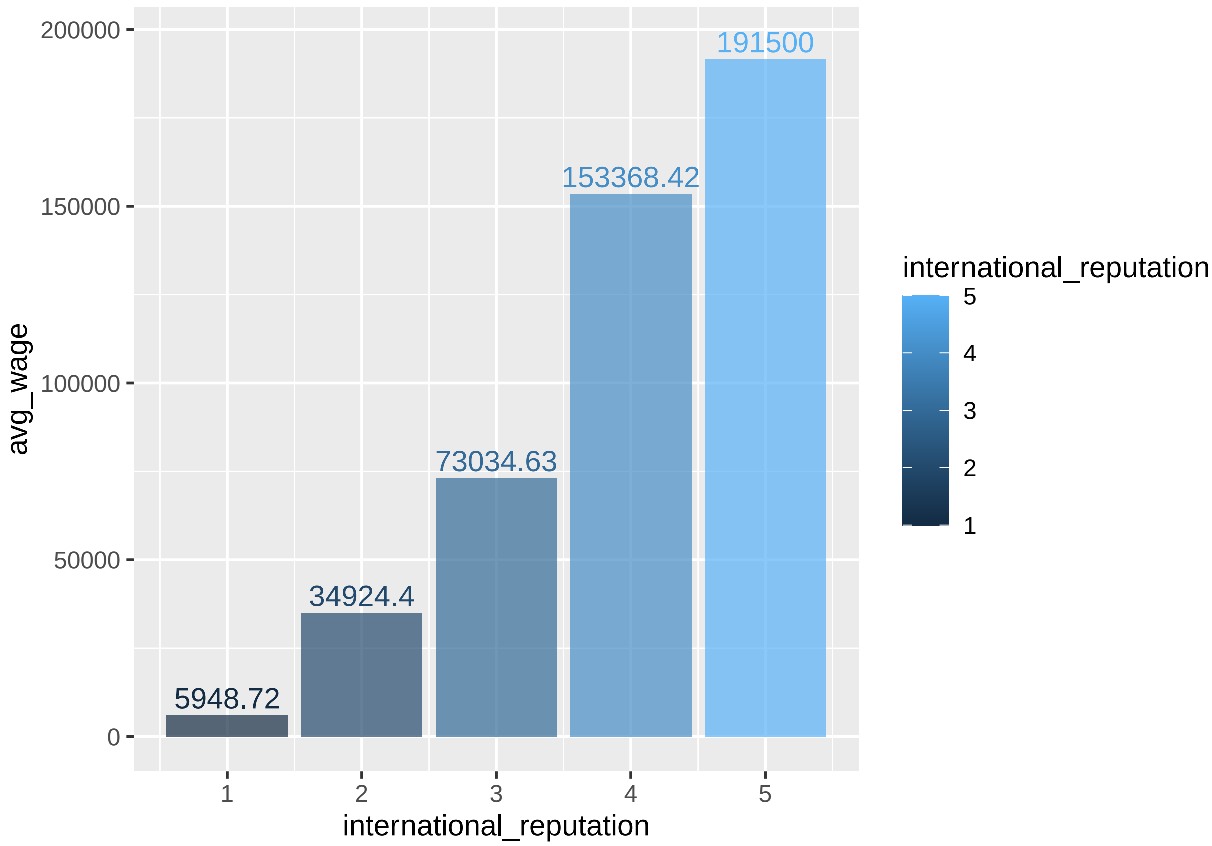
We then focused on the top ranked clubs in league level one which could be more representative. We calculated average, total, and median wage for each league and selected the league with average wage more than €10,000. From the dot plot we have highlighted 9 leagues that met the criteria, and these leagues will be the main target for our analysis.

```r
ggplot(data = p_gen_lea_avg_order) +
  geom_density(aes(x = avg_wage), color = "purple") +
  ggtitle("Density Plot for Average Weekly Wage", subtitle = "wage measured in EUR") +
  xlab("Average Weekly Wage")
```



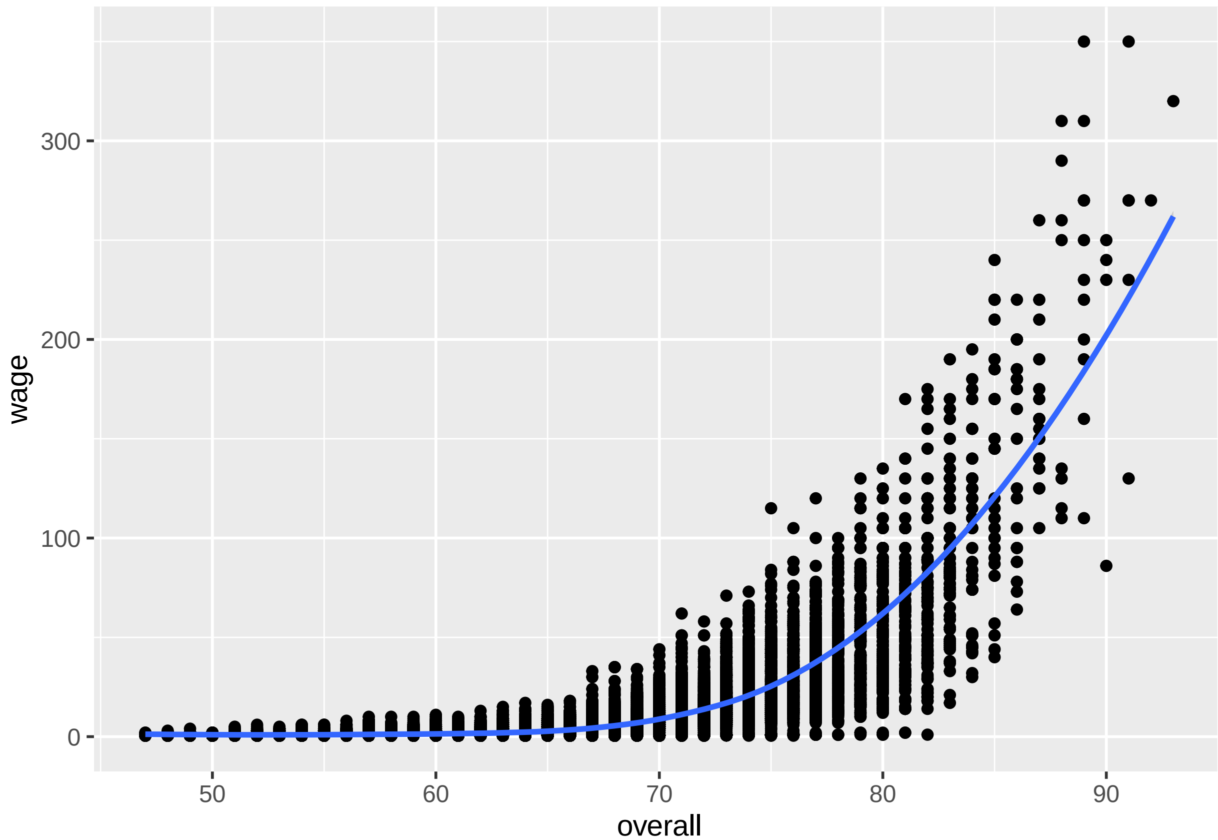
The density plot shows that there is a huge proportion of leagues with average weekly wage less than €10,000. There are three little peaks in €20,000, €30,000, and €50,000, corresponding to the highlighted leagues in the dot plot.

```
# international reputation

p_int = p_gen |>
  drop_na(international_reputation) |>
  group_by(international_reputation) |>
  summarise(avg_wage = round(mean(na.omit(wage_eur)), digits = 2), .groups = "drop") |>
  mutate(index = avg_wage + 5000)

ggplot(data = p_int) +
  geom_col(aes(x = international_reputation,
             y = avg_wage,
             fill = international_reputation), alpha = 0.7) +
  geom_text(aes(x = international_reputation,
              y = index,
              label = avg_wage,
              color = international_reputation))
```

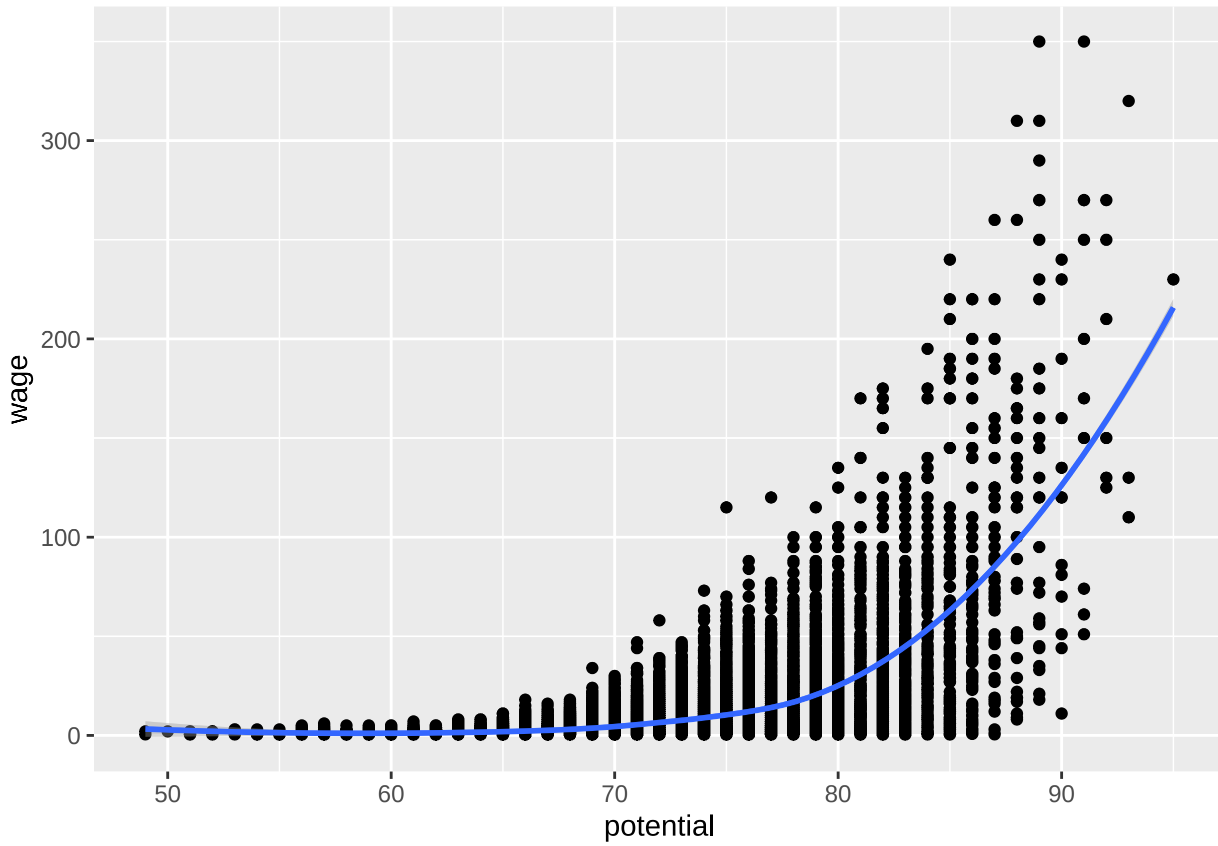


This plot shows player with higher international reputation will have higher average wage.

```
# wage by overall and potential

p_overall = p_gen |>
  drop_na(overall, wage_eur) |>
  mutate(wage = wage_eur / 1000) |>
  select(overall, wage)

ggplot(data = p_overall) +
  geom_point(aes(x = overall, y = wage)) +
  geom_smooth(aes(x = overall, y = wage), method = "loess", formula = y ~ x)
```




```
p_potential = p_gen |>
  drop_na(potential, wage_eur) |>
  mutate(wage = wage_eur / 1000) |>
  select(potential, wage)

ggplot(data = p_potential) +
  geom_point(aes(x = potential, y = wage)) +
  geom_smooth(aes(x = potential, y = wage), method = "loess", formula = y ~ x)
```



These two plots measure the overall performance and the potential of players, and it indicates the average weekly wage increases as the player performed better. We can conclude that average weekly wage has a positive relationship with player's overall performance and their potential.

4 Prediction

After plotting the data, we began focus on the relationship between the players' weekly wage and skills in different positions. We divided different positions into two groups: goalkeepers and non-goalkeepers. And for non-goalkeepers, we divided them into forward, midfield, and defender based on their position.

4.1 Linear Regression

We first conducted linear regression model for different positions.

For goalkeepers, we selected skills that are related with goalkeepers, such as goalkeeping handling and positioning, and conducted linear regression. We split the dataset into two parts: training dataset and testing dataset. We used our training dataset to obtain regression model and apply the model to our testing dataset to see the test error rate.

```r
#players that are GOAL KEEPER

p_gk = p_gen |>
  filter(club_position == "GK") |>
  select(9,12,73:78)|>
  na.omit()

set.seed(42)
train_gk = sample(c(TRUE,FALSE), nrow(p_gk), rep = TRUE)
test_gk = (!train_gk)
gk.train = p_gk[train_gk, ]
gk.test = p_gk[test_gk, ]

lm.fit_gk = lm(wage_eur~., data = gk.train)
lm.pred_gk = predict(lm.fit_gk, gk.test, type = "response")
summary(lm.fit_gk)
```

```
##
## Call:
## lm(formula = wage_eur ~ ., data = gk.train)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -35970  -8042  -2876   4460 194205
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            -222623.44   44792.45  -4.970 1.07e-06 ***
## height_cm                  346.62     236.54   1.465  0.14375
## goalkeeping_diving         331.41     461.07   0.719  0.47277
## goalkeeping_handling      1395.11     429.78   3.246  0.00129 **
## goalkeeping_kicking        257.10     225.60   1.140  0.25526
## goalkeeping_positioning   -339.93     378.36  -0.898  0.36960
## goalkeeping_reflexes       828.00     429.12   1.930  0.05451 .
## goalkeeping_speed          -32.05     117.27  -0.273  0.78480
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19690 on 337 degrees of freedom
## Multiple R-squared:  0.3904, Adjusted R-squared:  0.3777
## F-statistic: 30.83 on 7 and 337 DF,  p-value: < 2.2e-16
```

```
mean((lm.pred_gk - gk.test$wage_eur) ^ 2)
```

```
## [1] 274756588
```

We applied the same process for forward, midfield, and defender and obtained their linear regression model.

```
#players that are FORWARD

p_f = p_gen |>
  filter(club_position %in% c("ST", "CF", "RS", "LS", "RF", "LF", "RW", "LW")) |>
  select(9, 12, 44:72) |>
  na.omit()

set.seed(42)
train_f = sample(c(TRUE,FALSE), nrow(p_f), rep = TRUE)
test_f = (!train_f)
f.train = p_f[train_f, ]
f.test = p_f[test_f, ]

lm.fit_f = lm(wage_eur~., data = f.train)
lm.pred_f = predict(lm.fit_f, f.test, type = "response")
summary(lm.fit_f)
```

```
##
## Call:
## lm(formula = wage_eur ~ ., data = f.train)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -42953 -11708  -3002   6705 209176
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -3.334e+05  5.126e+04  -6.506 1.55e-10 ***
## height_cm                   6.264e+02  2.624e+02   2.388  0.01724 *
## attacking_crossing          5.427e+01  1.356e+02   0.400  0.68918
## attacking_finishing         6.237e+02  3.099e+02   2.012  0.04462 *
## attacking_heading_accuracy  2.658e+02  1.559e+02   1.705  0.08865 .
## attacking_short_passing     4.333e+01  3.145e+02   0.138  0.89046
## attacking_volleys          -1.308e+02  1.796e+02  -0.728  0.46683
## skill_dribbling            -1.115e+01  3.292e+02  -0.034  0.97298
## skill_curve                -1.405e+02  1.468e+02  -0.957  0.33876
## skill_fk_accuracy           2.213e+02  1.111e+02   1.992  0.04675 *
## skill_long_passing         -9.127e+01  1.779e+02  -0.513  0.60818
## skill_ball_control          8.954e+02  3.836e+02   2.334  0.01990 *
## movement_acceleration       4.573e+01  2.223e+02   0.206  0.83706
## movement_sprint_speed       4.052e+02  2.033e+02   1.993  0.04671 *
## movement_agility            3.133e+01  1.743e+02   0.180  0.85740
```

```
## movement_reactions           8.385e+02  2.910e+02   2.882  0.00409 **
## movement_balance             2.981e+02  1.449e+02   2.057  0.04011 *
## power_shot_power            -2.888e+02  2.528e+02  -1.142  0.25378
## power_jumping               -7.191e+01  9.773e+01  -0.736  0.46209
## power_stamina               -1.938e+02  1.279e+02  -1.516  0.13012
## power_strength              -5.635e+01  1.441e+02  -0.391  0.69597
## power_long_shots             3.148e+02  2.314e+02   1.360  0.17415
## mentality_aggression        -2.726e-01  9.399e+01  -0.003  0.99769
## mentality_interceptions     -1.296e+02  1.358e+02  -0.954  0.34023
## mentality_positioning        2.501e+02  3.519e+02   0.711  0.47756
## mentality_vision             3.921e+02  2.260e+02   1.735  0.08327 .
## mentality_penalties         -6.693e+01  1.544e+02  -0.434  0.66477
## mentality_composure         -6.228e+01  2.164e+02  -0.288  0.77358
## defending_marking_awareness -2.284e+02  1.192e+02  -1.916  0.05585 .
## defending_standing_tackle    3.328e+02  1.770e+02   1.880  0.06062 .
## defending_sliding_tackle    -5.990e+01  1.846e+02  -0.324  0.74570
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 25040 on 645 degrees of freedom
## Multiple R-squared:  0.4518, Adjusted R-squared:  0.4263
## F-statistic: 17.72 on 30 and 645 DF,  p-value: < 2.2e-16
```

```r
mean((lm.pred_f - f.test$wage_eur) ^ 2)
```

```
## [1] 712495719
```

```r
#players that are MIDFIELD
p_m = p_gen |>
  filter(club_position %in% c("RCM", "CDM", "RDM", "LCM", "CAM", "LDM", "LM",
                              "RM", "CM", "LAM", "RAM")) |>
  select(9, 12, 44:72) |>
  na.omit()

set.seed(42)
train_m = sample(c(TRUE,FALSE), nrow(p_m), rep = TRUE)
test_m = (!train_m)
m.train = p_m[train_m, ]
m.test = p_m[test_m, ]

lm.fit_m = lm(wage_eur~., data = m.train)
lm.pred_m = predict(lm.fit_m, m.test, type = "response")
summary(lm.fit_m)
```

```
##
## Call:
## lm(formula = wage_eur ~ ., data = m.train)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -37093  -8349  -2228   4022 281232
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   -2.171e+05  2.749e+04  -7.897 5.77e-15 ***
```

```
## height_cm                   3.195e+02  1.382e+02   2.313 0.020887 *
## attacking_crossing          6.744e+01  8.484e+01   0.795 0.426793
## attacking_finishing         3.481e+01  9.682e+01   0.360 0.719269
## attacking_heading_accuracy -3.104e+01  6.981e+01  -0.445 0.656677
## attacking_short_passing     4.620e+02  2.120e+02   2.179 0.029485 *
## attacking_volleys           8.495e+01  6.902e+01   1.231 0.218546
## skill_dribbling             8.106e+01  1.583e+02   0.512 0.608608
## skill_curve                -1.429e+02  7.820e+01  -1.828 0.067816 .
## skill_fk_accuracy           5.155e+01  6.516e+01   0.791 0.428955
## skill_long_passing          1.539e+02  1.387e+02   1.110 0.267288
## skill_ball_control          7.405e+02  2.115e+02   3.502 0.000476 ***
## movement_acceleration      -2.225e+01  1.124e+02  -0.198 0.843131
## movement_sprint_speed       2.199e+02  9.728e+01   2.261 0.023925 *
## movement_agility           -7.716e+01  9.638e+01  -0.801 0.423473
## movement_reactions          8.940e+02  1.258e+02   7.108 1.88e-12 ***
## movement_balance            1.218e+02  8.890e+01   1.370 0.170941
## power_shot_power           -1.330e+01  9.997e+01  -0.133 0.894194
## power_jumping               6.159e+01  5.293e+01   1.164 0.244788
## power_stamina              -3.400e+00  6.718e+01  -0.051 0.959646
## power_strength              1.058e+01  7.183e+01   0.147 0.882918
## power_long_shots           -1.142e+02  9.557e+01  -1.195 0.232413
## mentality_aggression        3.847e+01  5.959e+01   0.646 0.518599
## mentality_interceptions    -1.153e+02  8.321e+01  -1.385 0.166191
## mentality_positioning       3.608e+01  1.085e+02   0.333 0.739550
## mentality_vision           -6.229e+01  1.293e+02  -0.482 0.630107
## mentality_penalties        -2.696e+01  7.010e+01  -0.385 0.700647
## mentality_composure        -4.075e+01  1.074e+02  -0.379 0.704393
## defending_marking_awareness  1.138e+02  7.647e+01   1.488 0.137014
## defending_standing_tackle  -2.141e+00  1.215e+02  -0.018 0.985945
## defending_sliding_tackle    4.282e+01  1.153e+02   0.371 0.710440
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19280 on 1394 degrees of freedom
## Multiple R-squared:  0.3907, Adjusted R-squared:  0.3776
## F-statistic: 29.79 on 30 and 1394 DF,  p-value: < 2.2e-16
```

```
mean((lm.pred_m - m.test$wage_eur) ^ 2)
```

```
## [1] 325886608
```

```
#players that are DEFENDER

p_d = p_gen |>
  filter(club_position %in% c("LCB", "RCB", "LB", "RB", "CB", "RWB", "LWB")) |>
  select(9, 12, 44:72) |>
  na.omit()

set.seed(42)
train_d = sample(c(TRUE,FALSE), nrow(p_d), rep = TRUE)
test_d = (!train_d)
d.train = p_d[train_d, ]
d.test = p_d[test_d, ]
```

```r
lm.fit_d = lm(wage_eur~., data = d.train)
lm.pred_d = predict(lm.fit_d, d.test, type = "response")
summary(lm.fit_d)
```

```
##
## Call:
## lm(formula = wage_eur ~ ., data = d.train)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -34911  -6814  -1790   3891 164069
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -1.583e+05  2.201e+04  -7.196 1.00e-12 ***
## height_cm                   1.929e+02  1.122e+02   1.719  0.08589 .
## attacking_crossing         -6.710e+01  5.011e+01  -1.339  0.18071
## attacking_finishing         1.317e+02  5.915e+01   2.227  0.02611 *
## attacking_heading_accuracy  6.631e+01  7.781e+01   0.852  0.39423
## attacking_short_passing    -4.185e+01  1.102e+02  -0.380  0.70410
## attacking_volleys          -6.766e+01  5.458e+01  -1.240  0.21531
## skill_dribbling             1.115e+02  6.759e+01   1.650  0.09916 .
## skill_curve                 3.676e+01  5.147e+01   0.714  0.47530
## skill_fk_accuracy           3.578e+01  4.804e+01   0.745  0.45651
## skill_long_passing         -1.727e+01  6.924e+01  -0.249  0.80303
## skill_ball_control          4.348e+01  9.875e+01   0.440  0.65979
## movement_acceleration       7.478e+01  7.850e+01   0.953  0.34092
## movement_sprint_speed       2.084e+02  6.439e+01   3.237  0.00124 **
## movement_agility           -1.197e+02  6.116e+01  -1.956  0.05062 .
## movement_reactions          6.018e+02  1.090e+02   5.523 3.96e-08 ***
## movement_balance           -4.868e+01  5.852e+01  -0.832  0.40561
## power_shot_power           -3.328e+01  5.076e+01  -0.656  0.51217
## power_jumping               4.624e+01  4.233e+01   1.092  0.27487
## power_stamina               1.810e+01  5.088e+01   0.356  0.72209
## power_strength             -1.413e+02  6.959e+01  -2.030  0.04257 *
## power_long_shots           -7.798e+00  5.637e+01  -0.138  0.88999
## mentality_aggression       -8.956e+01  5.684e+01  -1.576  0.11534
## mentality_interceptions     1.004e+02  1.315e+02   0.763  0.44561
## mentality_positioning      -3.744e+01  5.605e+01  -0.668  0.50421
## mentality_vision            1.418e+02  5.894e+01   2.406  0.01626 *
## mentality_penalties        -2.377e+01  5.118e+01  -0.464  0.64245
## mentality_composure         1.038e+02  7.626e+01   1.361  0.17363
## defending_marking_awareness 3.174e+02  1.268e+02   2.504  0.01240 *
## defending_standing_tackle   4.994e+02  1.770e+02   2.821  0.00485 **
## defending_sliding_tackle    2.669e+02  1.615e+02   1.653  0.09856 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14770 on 1422 degrees of freedom
## Multiple R-squared:  0.4637, Adjusted R-squared:  0.4524
## F-statistic: 40.99 on 30 and 1422 DF,  p-value: < 2.2e-16
```

```r
mean((lm.pred_d - d.test$wage_eur) ^ 2)
```

```
## [1] 226951083
```

4.2 Lasso Regression

We applied the second strategy, lasso regression, to our dataset. We used the Lamda of cross-validation to find the best lasso model.

```r
### GK

train_gk.mat = model.matrix(wage_eur~., data = gk.train)
test_gk.mat = model.matrix(wage_eur~., data = gk.test)

set.seed(42)
cv.out_gk = cv.glmnet(train_gk.mat, gk.train$wage_eur, alpha = 1)
bestlam_gk = cv.out_gk$lambda.min
bestlam_gk
```

```
## [1] 768.5729
```

```r
lasso.mod_gk = glmnet(train_gk.mat, gk.train$wage_eur, alpha = 1)
lasso.pred_gk = predict(lasso.mod_gk, s = bestlam_gk, newx = test_gk.mat)
mean((lasso.pred_gk - gk.test$wage_eur) ^ 2)
```

```
## [1] 267817511
```

```r
lasso.coef_gk = predict(lasso.mod_gk, type = "coefficients", s = bestlam_gk)
length(lasso.coef_gk[lasso.coef_gk != 0])
```

```
## <sparse>[ <logic> ]: .M.sub.i.logical() maybe inefficient
```

```
## [1] 6
```

```r
lasso.coef_gk
```

```
## 9 x 1 sparse Matrix of class "dgCMatrix"
##                                    s1
## (Intercept)             -185588.2082
## (Intercept)                   .
## height_cm                   199.9548
## goalkeeping_diving          236.6291
## goalkeeping_handling       1227.0017
## goalkeeping_kicking         119.1639
## goalkeeping_positioning       .
## goalkeeping_reflexes        729.3857
## goalkeeping_speed             .
```

```r
### F

train_f.mat = model.matrix(wage_eur~., data = f.train)
test_f.mat = model.matrix(wage_eur~., data = f.test)

set.seed(42)
cv.out_f = cv.glmnet(train_f.mat, f.train$wage_eur, alpha = 1)
bestlam_f = cv.out_f$lambda.min
bestlam_f
```

```
## [1] 1215.353
```

```r
lasso.mod_f = glmnet(train_f.mat, f.train$wage_eur, alpha = 1)
lasso.pred_f = predict(lasso.mod_f, s = bestlam_f, newx = test_f.mat)
mean((lasso.pred_f - f.test$wage_eur) ^ 2)
```

```
## [1] 698632497
```

```r
lasso.coef_f = predict(lasso.mod_f, type = "coefficients", s = bestlam_f)
length(lasso.coef_f[lasso.coef_f != 0])
```

```
## <sparse>[ <logic> ]: .M.sub.i.logical() maybe inefficient
```

```
## [1] 11
```

```r
lasso.coef_f
```

```
## 32 x 1 sparse Matrix of class "dgCMatrix"
##                                         s1
## (Intercept)                  -1.892451e+05
## (Intercept)                   .
## height_cm                     .
## attacking_crossing            .
## attacking_finishing           6.227994e+02
## attacking_heading_accuracy    .
## attacking_short_passing       .
## attacking_volleys             .
## skill_dribbling               9.931592e+01
## skill_curve                   .
## skill_fk_accuracy             9.347063e+01
## skill_long_passing            .
## skill_ball_control            8.422616e+02
## movement_acceleration         .
## movement_sprint_speed         2.572154e+02
## movement_agility              .
## movement_reactions            8.106642e+02
## movement_balance              .
## power_shot_power              .
## power_jumping                 .
## power_stamina                 .
## power_strength                .
## power_long_shots              1.067886e+02
## mentality_aggression          .
## mentality_interceptions       .
## mentality_positioning         1.699747e+00
## mentality_vision              2.029903e+02
## mentality_penalties           .
## mentality_composure           .
## defending_marking_awareness   .
## defending_standing_tackle     5.332260e+00
## defending_sliding_tackle      .
```

```r
### M

train_m.mat = model.matrix(wage_eur~., data = m.train)
test_m.mat = model.matrix(wage_eur~., data = m.test)
```

```
set.seed(42)
cv.out_m = cv.glmnet(train_m.mat, m.train$wage_eur, alpha = 1)
bestlam_m = cv.out_m$lambda.min
bestlam_m
```

```
## [1] 447.7585
```

```
lasso.mod_m = glmnet(train_m.mat, m.train$wage_eur, alpha = 1)
lasso.pred_m = predict(lasso.mod_m, s = bestlam_m, newx = test_m.mat)
mean((lasso.pred_m - m.test$wage_eur) ^ 2)
```

```
## [1] 325410895
```

```
lasso.coef_m = predict(lasso.mod_m, type = "coefficients", s = bestlam_m)
length(lasso.coef_m[lasso.coef_m != 0])
```

```
## <sparse>[ <logic> ]: .M.sub.i.logical() maybe inefficient
```

```
## [1] 10
```

```
lasso.coef_m
```

```
## 32 x 1 sparse Matrix of class "dgCMatrix"
##                                        s1
## (Intercept)                 -1.712799e+05
## (Intercept)                  .
## height_cm                    1.433129e+02
## attacking_crossing           .
## attacking_finishing          .
## attacking_heading_accuracy   .
## attacking_short_passing      4.970238e+02
## attacking_volleys            .
## skill_dribbling              4.666464e-01
## skill_curve                  .
## skill_fk_accuracy            .
## skill_long_passing           8.613718e+00
## skill_ball_control           7.316160e+02
## movement_acceleration        .
## movement_sprint_speed        1.570555e+02
## movement_agility             .
## movement_reactions           8.601317e+02
## movement_balance             .
## power_shot_power             .
## power_jumping                3.968384e+01
## power_stamina                .
## power_strength               .
## power_long_shots             .
## mentality_aggression         .
## mentality_interceptions      .
## mentality_positioning        .
## mentality_vision             .
## mentality_penalties          .
## mentality_composure          .
## defending_marking_awareness  4.419659e+01
## defending_standing_tackle    .
## defending_sliding_tackle     .
```

```r
### D

train_d.mat = model.matrix(wage_eur~., data = d.train)
test_d.mat = model.matrix(wage_eur~., data = d.test)

set.seed(42)
cv.out_d = cv.glmnet(train_d.mat, d.train$wage_eur, alpha = 1)
bestlam_d = cv.out_d$lambda.min
bestlam_d
```

```
## [1] 141.9324
```

```r
lasso.mod_d = glmnet(train_d.mat, d.train$wage_eur, alpha = 1)
lasso.pred_d = predict(lasso.mod_d, s = bestlam_d, newx = test_d.mat)
mean((lasso.pred_d - d.test$wage_eur) ^ 2)
```

```
## [1] 226045284
```

```r
lasso.coef_d = predict(lasso.mod_d, type = "coefficients", s = bestlam_d)
length(lasso.coef_d[lasso.coef_d != 0])
```

```
## <sparse>[ <logic> ]: .M.sub.i.logical() maybe inefficient
```

```
## [1] 23
```

```r
lasso.coef_d
```

```
## 32 x 1 sparse Matrix of class "dgCMatrix"
##                                        s1
## (Intercept)                 -1.558632e+05
## (Intercept)                  .
## height_cm                    1.661054e+02
## attacking_crossing          -2.007549e+01
## attacking_finishing          7.294878e+01
## attacking_heading_accuracy   3.446315e+01
## attacking_short_passing      .
## attacking_volleys           -2.462207e+01
## skill_dribbling              7.124247e+01
## skill_curve                  .
## skill_fk_accuracy            3.793791e+00
## skill_long_passing           .
## skill_ball_control           2.104087e+01
## movement_acceleration        2.857307e+01
## movement_sprint_speed        2.066053e+02
## movement_agility            -6.228568e+01
## movement_reactions           6.050392e+02
## movement_balance            -2.164681e+01
## power_shot_power             .
## power_jumping                2.069286e+01
## power_stamina                .
## power_strength              -7.066988e+01
## power_long_shots             .
## mentality_aggression        -6.493845e+01
## mentality_interceptions      9.028638e+01
## mentality_positioning        .
## mentality_vision             1.022002e+02
## mentality_penalties          .
```

```
## mentality_composure          7.985868e+01
## defending_marking_awareness  3.115622e+02
## defending_standing_tackle    5.030108e+02
## defending_sliding_tackle     2.488225e+02
```

5 Summary

After comparing the test error rate (TRSS) of our linear regression and lasso regression, we decided to use lasso regression as it is a better model for our regression, and concluded four models for FIFA players' wage by their position on the court.

5.0.1 Goalkeeper

pred_gk = height_cm * 199.9548 + goalkeeping_diving * 236.6291 + goalkeeping_handling * 1227.0017 + goalkeeping_kicking * 119.1639 + goalkeeping_reflexes * 729.3857 - 185588.2082

5.0.2 Forward

pred_f = attacking_finishing * 6.227994e+02 + skill_dribbling * 9.931592e+01 + skill_fk_accuracy * 9.347063e+01 + skill_ball_control * 8.422616e+02 + movement_sprint_speed * 2.572154e+02 + movement_reactions * 8.106642e+02 + power_long_shots * 1.067886e+02 + mentality_positioning * 1.699747e+00 + mentality_vision * 2.029903e+02 + defending_standing_tackle * 5.332260e+00 - 1.892451e+05

5.0.3 Midfield

pred_m = height_cm * 1.433129e+02 + attacking_short_passing * 4.970238e+02 + skill_dribbling * 4.666464e-01 + skill_long_passing * 8.613718e+00 + skill_ball_control * 7.316160e+02 + movement_sprint_speed * 1.570555e+02 + movement_reactions * 8.601317e+02 + power_jumping * 3.968384e+01 + defending_marking_awareness * 4.419659e+01 - 1.712799e+05

5.0.4 Defender

pred_d = height_cm * 1.661054e+02 + attacking_crossing * -2.007549e+01 + attacking_finishing * 7.294878e+01 + attacking_heading_accuracy * 3.446315e+01 + attacking_volleys * -2.462207e+01 + skill_dribbling * 7.124247e+01 + skill_fk_accuracy * 3.793791e+00 + skill_ball_control * 2.104087e+01 + movement_acceleration * 2.857307e+01 + movement_sprint_speed * 2.066053e+02 + movement_agility * -6.228568e+01 + movement_reactions * 6.050392e+02 + movement_balance * -2.164681e+01 + power_jumping * 2.069286e+01 + power_strength * -7.066988e+01 + mentality_aggression * -6.493845e+01 + mentality_interceptions * 9.028638e+01 + mentality_vision * 1.022002e+02 + mentality_composure * 7.985868e+01 + defending_marking_awareness * 3.115622e+02 + defending_standing_tackle * 5.030108e+02 + defending_sliding_tackle * 2.488225e+02 - 1.558632e+05

Based on our graphs and regression models, we can conclude that soccer players start to grow as they enter this field and make the most money when they reach 30. Once a player get older than 30, there is a tendency for their wage to decline as their physical abilities start to weaken. Besides, as a player's international reputation, overall, and potential socre increase, their wage increase as well.

We chose our lasso regression model as our final prediction model. We can use the models to predict a player's wage based on their positions on the court and their physical ability scores. We hope our prediction can help FIFA better decide a player's wage and therefore create a relatively fair market.