NBA_Final

May 19, 2022

[9]:
```
import warnings
warnings.filterwarnings('ignore')


import pandas as pd
import numpy as np
from plotnine import *

import matplotlib.pyplot as plt

from sklearn.tree import DecisionTreeClassifier # Decision Tree
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split

from sklearn import metrics
from sklearn.linear_model import LinearRegression # Linear Regression Model
from sklearn.linear_model import LogisticRegression # Logistic Regression Model
from sklearn.linear_model import Lasso
from sklearn.preprocessing import StandardScaler #Z-score variables

from sklearn.model_selection import train_test_split # simple TT split cv
from sklearn.model_selection import KFold # k-fold cv
from sklearn.cluster import KMeans
from sklearn.model_selection import LeaveOneOut #LOO cv
from sklearn.model_selection import cross_val_score # cross validation metrics
from sklearn.model_selection import cross_val_predict # cross validation metrics
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score␣
↪#model evaluation
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.metrics import plot_confusion_matrix
from sklearn.pipeline import make_pipeline
from sklearn.compose import make_column_transformer
from sklearn.decomposition import PCA
from sklearn.metrics import silhouette_score
from sklearn.model_selection import GridSearchCV

%precision %.7g
```

```
%matplotlib inline
```

[10]:
```
nba = pd.read_csv("https://query.data.world/s/fwbezkloolkpuahajrtzju5dxthw4p")
print(nba.columns)
```

```
Index(['Unnamed: 0', 'Team', 'Game', 'Date', 'Home', 'Opponent', 'WINorLOSS',
       'TeamPoints', 'OpponentPoints', 'FieldGoals', 'FieldGoalsAttempted',
       'FieldGoals.', 'X3PointShots', 'X3PointShotsAttempted', 'X3PointShots.',
       'FreeThrows', 'FreeThrowsAttempted', 'FreeThrows.', 'OffRebounds',
       'TotalRebounds', 'Assists', 'Steals', 'Blocks', 'Turnovers',
       'TotalFouls', 'Opp.FieldGoals', 'Opp.FieldGoalsAttempted',
       'Opp.FieldGoals.', 'Opp.3PointShots', 'Opp.3PointShotsAttempted',
       'Opp.3PointShots.', 'Opp.FreeThrows', 'Opp.FreeThrowsAttempted',
       'Opp.FreeThrows.', 'Opp.OffRebounds', 'Opp.TotalRebounds',
       'Opp.Assists', 'Opp.Steals', 'Opp.Blocks', 'Opp.Turnovers',
       'Opp.TotalFouls'],
      dtype='object')
```

1 Ronan Q's

1.1 Cleaning The Data

[11]:
```
nba = pd.read_csv("https://query.data.world/s/fwbezkloolkpuahajrtzju5dxthw4p")
nba.columns
nba.drop("Unnamed: 0",axis = 1)
```

[11]:
```
      Team  Game        Date  Home Opponent WINorLOSS  TeamPoints  \
0      ATL     1  2014-10-29  Away      TOR         L         102
1      ATL     2  2014-11-01  Home      IND         W         102
2      ATL     3  2014-11-05  Away      SAS         L          92
3      ATL     4  2014-11-07  Away      CHO         L         119
4      ATL     5  2014-11-08  Home      NYK         W         103
…    …    …          …    …        …         …           …
9835   WAS    78  2018-04-03  Away      HOU         L         104
9836   WAS    79  2018-04-05  Away      CLE         L         115
9837   WAS    80  2018-04-06  Home      ATL         L          97
9838   WAS    81  2018-04-10  Home      BOS         W         113
9839   WAS    82  2018-04-11  Away      ORL         L          92

      OpponentPoints  FieldGoals  FieldGoalsAttempted  …  Opp.FreeThrows  \
0                109          40                   80  …              27
1                 92          35                   69  …              18
2                 94          38                   92  …              27
3                122          43                   93  …              20
4                 96          33                   81  …               8
```

```
…              …         …                   …  …             …
9835          120        38                   72  …            18
9836          119        47                   94  …            22
9837          103        35                   87  …            16
9838          101        41                   83  …            22
9839          101        33                   95  …            22

      Opp.FreeThrowsAttempted  Opp.FreeThrows.  Opp.OffRebounds  \
0                          33            0.818               16
1                          21            0.857               11
2                          38            0.711               11
3                          27            0.741               11
4                          11            0.727               13
…                         …                …                …
9835                       27            0.667               10
9836                       28            0.786                5
9837                       23            0.696                7
9838                       27            0.815               13
9839                       27            0.815                6

      Opp.TotalRebounds  Opp.Assists  Opp.Steals  Opp.Blocks  Opp.Turnovers  \
0                    48           26          13           9              9
1                    44           25           5           5             18
2                    50           25           7           9             19
3                    51           31           6           7             19
4                    44           26           2           6             15
…                   …           …          …          …             …
9835                 46           26          13           3              9
9836                 35           26          10           3             16
9837                 50           24           5           5             18
9838                 44           22          14           1             16
9839                 42           20           6           7             16

      Opp.TotalFouls
0                 22
1                 26
2                 15
3                 30
4                 29
…              …
9835              14
9836              14
9837              22
9838              18
9839              27

[9840 rows x 40 columns]
```

1.2 Q1

[12]:
```
predictors = ['TeamPoints', 'OpponentPoints', 'FieldGoals',␣
 ↪'FieldGoalsAttempted',
       'FieldGoals.', 'X3PointShots', 'X3PointShotsAttempted', 'X3PointShots.',
       'FreeThrows', 'FreeThrowsAttempted', 'FreeThrows.', 'OffRebounds',
       'TotalRebounds', 'Assists', 'Steals', 'Blocks',
       'TotalFouls', 'Opp.FieldGoals', 'Opp.FieldGoalsAttempted',
       'Opp.FieldGoals.', 'Opp.3PointShots', 'Opp.3PointShotsAttempted',
       'Opp.3PointShots.', 'Opp.FreeThrows', 'Opp.FreeThrowsAttempted',
       'Opp.FreeThrows.', 'Opp.OffRebounds', 'Opp.TotalRebounds',
       'Opp.Assists', 'Opp.Steals', 'Opp.Blocks', 'Opp.Turnovers',
       'Opp.TotalFouls']
```

[13]:
```
X_train, X_test, y_train, y_test = train_test_split(nba[predictors],␣
 ↪nba["Turnovers"], test_size=0.2)
X_train.head()

z = StandardScaler()
X_train[predictors] = z.fit_transform(X_train[predictors])
X_test[predictors] = z.transform(X_test[predictors])

lr = LinearRegression()
lr.fit(X_train, y_train)

y_train_pred = lr.predict(X_train)
y_test_pred = lr.predict(X_test)

#Test Set
r2_test = r2_score(y_test, y_test_pred)
mse_test = mean_squared_error(y_test, y_test_pred)

#Train Set
r2_train = r2_score(y_train, y_train_pred)
mse_train = mean_squared_error(y_train, y_train_pred)
```

[14]:
```
y_pred = lr.predict(X_test)
true_vs_pred = pd.DataFrame({"predicted": y_pred,
                             "true": y_test})

true_vs_pred
```

[14]:

	predicted	true
2886	15.489034	19
9404	11.653713	13
6942	15.244070	14
6744	18.405501	17

```
3162   7.504999     9
…           …    …
1012  11.374762    10
157   16.652208    16
2470  16.759508    16
8818   9.730816    11
4733  13.193701    13

[1968 rows x 2 columns]
```

[15]:

```
print("R2 Train: ", r2_train)
print("MSE Train: ",mse_train)

print("R2 Test: ", r2_test)
print("MSE Test: ",mse_test)
```

```
R2 Train:  0.8082919267282684
MSE Train:  2.869532363662745
R2 Test:  0.809204957505012
MSE Test:  2.858744458928596
```

[16]:

```
z = StandardScaler()
nba[predictors] = z.fit_transform(nba[predictors])

pca = PCA()
pca.fit(nba[predictors])
pcaDF = pd.DataFrame({"expl_var" :
                      pca.explained_variance_ratio_,
                      "pc": range(1,34),
                      "cum_var":
                      pca.explained_variance_ratio_.cumsum()})

pcaDF
(ggplot(pcaDF, aes(x = "pc", y = "cum_var")) + geom_line(color = "pink") +
 geom_point(color = "pink") + theme_minimal() +geom_hline(yintercept = 0.90)+␣
 ↪labs(title = "Explained Cum Variance Plot"))
```

[16]: <ggplot: (8760369762337)>

Figure 1.

Explained cumulative variance plot showing the accumulated explainable variance per additional prinicipal component.

[17]:
```
pcomps10 = pca.transform(nba[predictors])
pcomps10 = pd.DataFrame(pcomps10[:, 0:14])

#modeMod1
lr1 = LogisticRegression()
lr1.fit(nba[predictors], nba["Turnovers"])
print("all data: ", lr1.score(nba[predictors], nba["Turnovers"]))

#modeMod1
lr2 = LogisticRegression()
lr2.fit(pcomps10, nba["Turnovers"])
print("14 PCs:   ", lr2.score(pcomps10, nba["Turnovers"]))
```

```
all data:  0.26148373983739837
14 PCs:    0.19654471544715446
```

[18]:
```
pca_y_pred = lr1.predict(X_test)
pca_true_vs_pred = pd.DataFrame({"predicted": pca_y_pred,
                             "true": y_test})
```

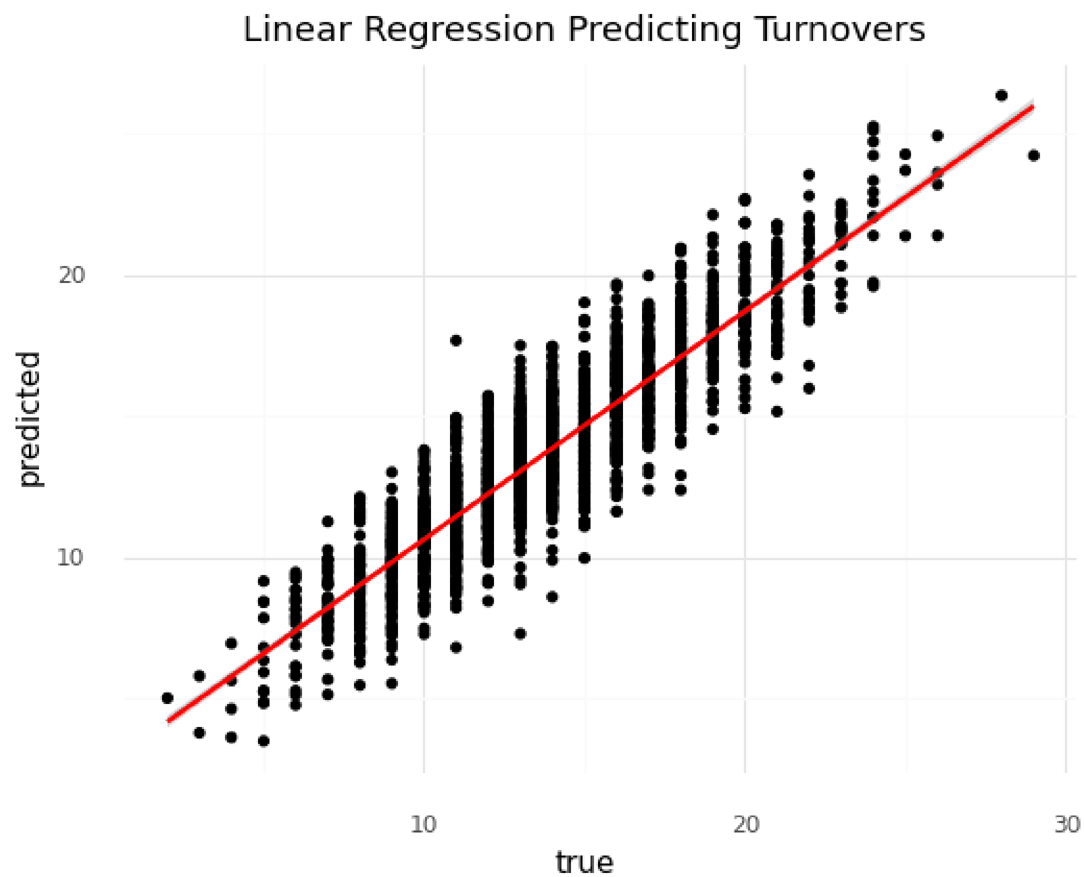
[19]:
```
pca_train = pca.transform(X_train)
pca_train = pd.DataFrame(pca_train[:,0:14])

pca_test = pca.transform(X_test)
pca_test = pd.DataFrame(pca_test[:,0:14])

lr.fit(pca_train, y_train)

pca_train_pred = lr.predict(pca_train)
pca_test_pred = lr.predict(pca_test)

#Train
pca_train_r2 = r2_score(y_train, pca_train_pred)
pca_train_mse = mean_squared_error(y_train, pca_train_pred)

#Test
pca_test_r2 = r2_score(y_test, pca_test_pred)
pca_test_mse = mean_squared_error(y_test, pca_test_pred)
```

[20]:
```
print("R2 Train: ", pca_train_r2)
print("MSE Train: ",pca_train_mse)

print("R2 Test: ", pca_test_r2)
print("MSE Test: ",pca_test_mse)
```

```
R2 Train:  0.6763502383483131
MSE Train:  4.8444671614578
R2 Test:  0.6844378078786812
MSE Test:  4.7281714261411105
```

Linear Regression Model

[21]:
```
(ggplot(true_vs_pred, aes(x = "true", y = "predicted")) + geom_point() +␣
 ↪theme_minimal() + geom_smooth(method = "lm", color = "red") + labs(title =␣
 ↪"Linear Regression Predicting Turnovers"))
```



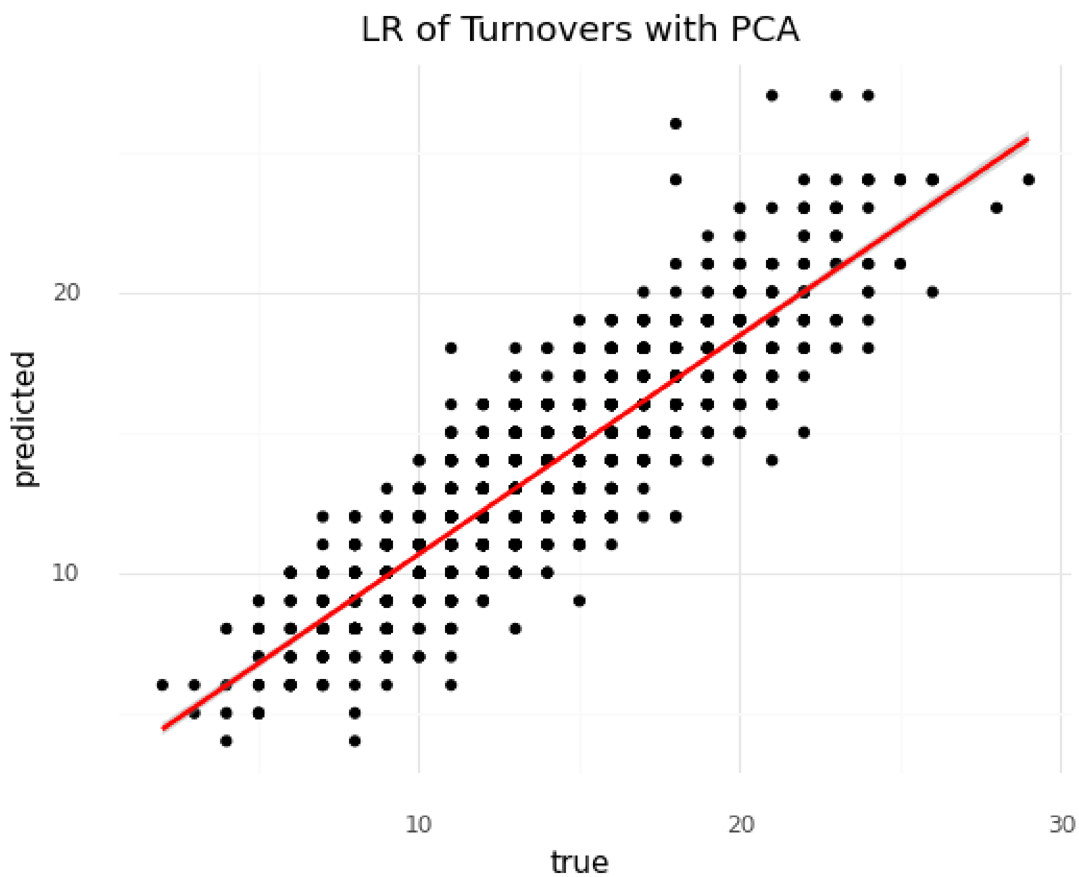
[21]: <ggplot: (8760369710777)>

Figure 2.

Plot of the true vs predicted turnovers for our linear regression model using all continuous and standardized variables.

Linear Regression with Dimensionality Reduction

[22]:
```
(ggplot(pca_true_vs_pred, aes(x = "true", y = "predicted")) + geom_point()+␣
↪theme_minimal() + geom_smooth(method = "lm", color = "red") + labs(title =␣
↪"LR of Turnovers with PCA"))
```



[22]: <ggplot: (8760369719221)>

Figure 3.

Plot of the true vs predicted amount of turnovers with dimensionality redcution using principal component analysis and 14 principal components

1.3 Q1 Dicussion

1.3.1 When comparing a model using PCA on all the continuous variables for the team (not turnovers), what is the difference in linear regression accuracy between number of components when predicting how often a team will turnover in a game to retain enough PCs that explain 90% of the variance?

Many statistics were recorded across the NBA season games from 2014 to 2018. We used these continuous predictors, such as points and fouls, across games from this period in a linear regression model. This model predicted the number of turnovers given the continuous predictors. The model performed accurately on the train and test set with r2 scores of .80 and .81, respectively. This shows that our model is not overfitting since the model performed more accurately on the test set. Therefore, the r2 of our model explains above 80% of the variance in both seen and unseen data. However, using the many continuous statistics recorded in NBA games means our model is slightly inefficient and data-heavy. We can compare this model to a linear regression model that utilizes fewer statistics using principal component analysis. It does this by using some of the relationships between certain NBA variables. We found that 14 principal components, which are linear combinations of some of the original variables, were needed to explain 90% of the cumulative variance in the data. Using these 14 principal components, we created the second linear regression model to compare accuracies. The r2 of the PCA linear regression model for the train and test was .68 and .69, respectively. This means that our model with 14 pcs explains around 10% less variance in seen and unseen data than the original model. The mean square error using 14 components was also about four, while the original models were around 2 in both train and test sets. These accuracy results are also reflected in the linear regression model score between the two models, which had a difference of only .07. So while our linear regression model with all variables is more accurate and has less error in predictions of the turnovers in a game, the model using dimensionality reduction through 14 components was only somewhat less precise.

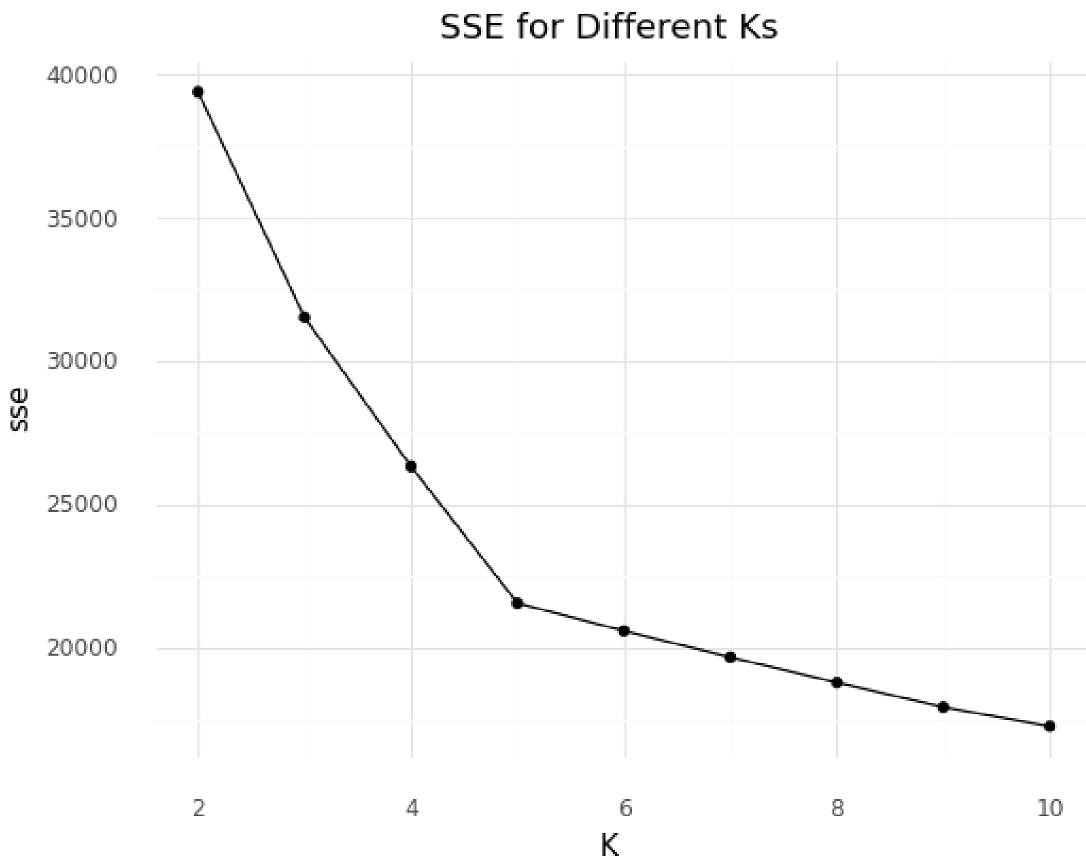
1.4 Q2

[23]:
```
features = ['FieldGoals', 'Steals', 'TotalFouls', 'Blocks']
X = nba[features]

z = StandardScaler()

X[features] = z.fit_transform(X[features])

#model
km = KMeans(n_clusters = 5)
km.fit(X)

membership = km.predict(X)
X["cluster"] = membership

print(silhouette_score(X[features], membership))
```

```
0.17807297416628884
```

[24]:
```
ks = [2,3,4,5,6,7,8,9,10]

sse = []
sils = []

for k in ks:
    km = KMeans(n_clusters = k)
    km.fit(X)

    sse.append(km.inertia_)
```

```
    sils.append(silhouette_score(X, km.predict(X)))

sse_df = pd.DataFrame({"K": ks,
                      "sse": sse,
                      "silhouette": sils})

(ggplot(sse_df, aes(x = "K", y = "sse")) + geom_point() +
geom_line() +
theme_minimal() +
labs(title = "SSE for Different Ks"))
```



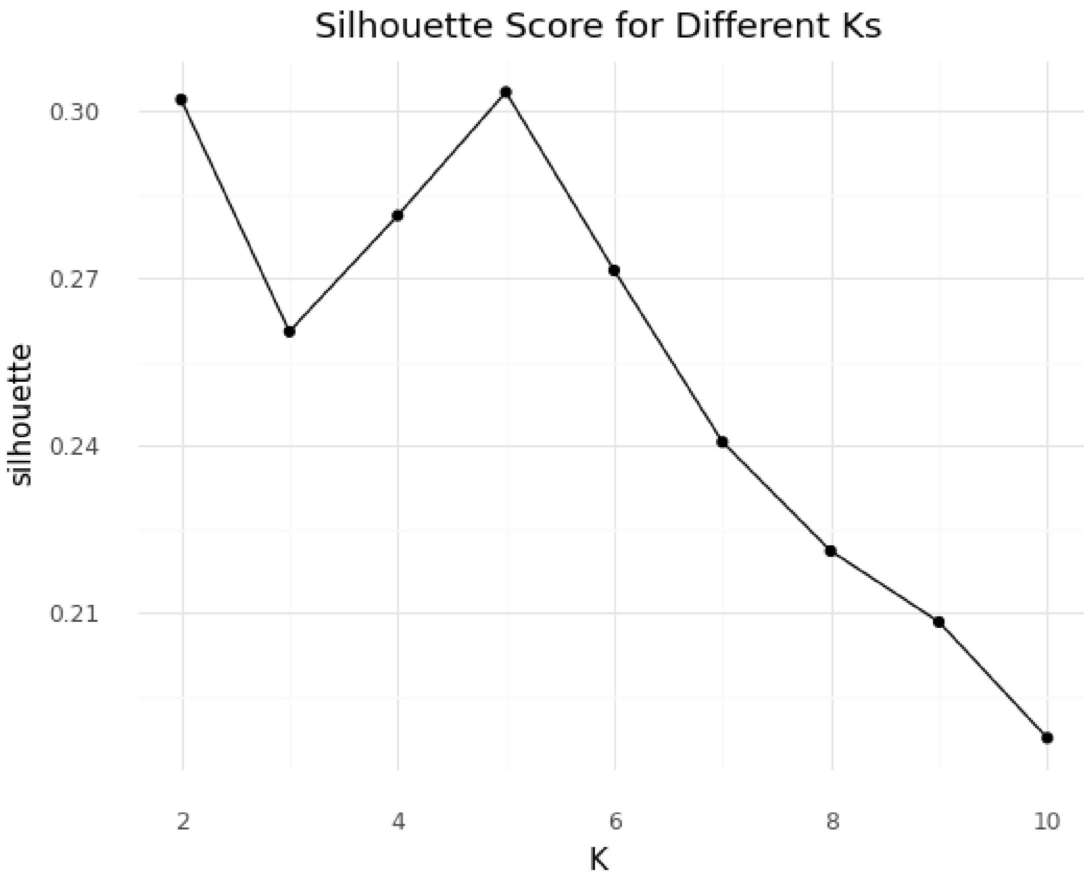
[24]: <ggplot: (8760369640437)>

Figure 4.

Plot of the sum of squared error for increasing k cluster centers using Field Goals, Steals, Total Fouls and Blocks as predictors.

[25]:
```
(ggplot(sse_df, aes(x = "K", y = "silhouette")) + geom_point() +
geom_line() +
```

```
theme_minimal() +
labs(title = "Silhouette Score for Different Ks"))
```

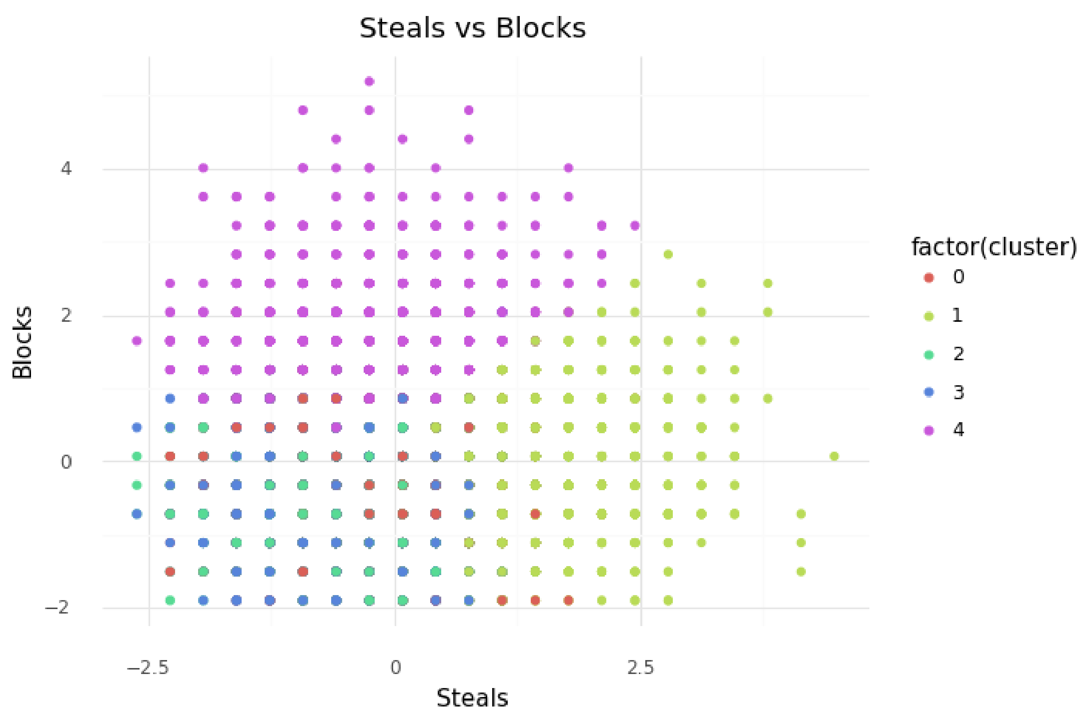


[25]: <ggplot: (8760369326489)>

Figure 5.

Plot of the silhouette scores for increasing k cluster centers. Selected k = 5

[26]:
```
(ggplot(X, aes(x = "Steals", y = "Blocks", color = "factor(cluster)")) +
 geom_point() + theme_minimal() + labs(title = "Steals vs Blocks"))
```

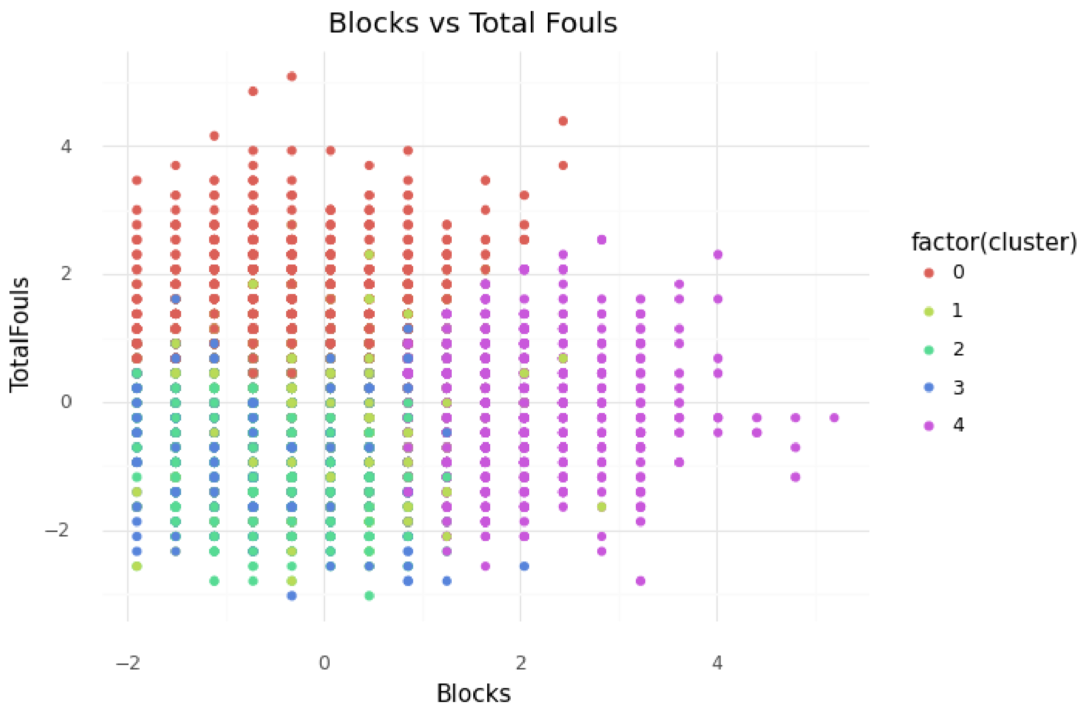



```
[26]: <ggplot: (8760369683665)>
```

Figure 6.

Plot of the cluster assignment for steals vs blocks

```
[27]: (ggplot(X, aes(x = "Blocks", y = "TotalFouls", color = "factor(cluster)")) +
       geom_point() + theme_minimal() + labs(title = "Blocks vs Total Fouls"))
```



[27]: <ggplot: (8760366553861)>

Figure 7.

Plot of the cluster membership for block vs total fouls

1.5 Q2 Discussion

1.5.1 (Clustering) When considering field goals, steals, total fouls, and blocks what clusters emerge and what characterizes clusters based on these characteristics?

Plotting the relationship between these statistics recorded across games, clusters of the data have specific characteristics related to these stats. We can use KMeans to cluster these unique relationships in a certain amount of clusters. Initially, we need to test the sum of squared error for the model using different amounts of clusters. We can plot this error alongside the increasing k clusters, which helped identify 5 clusters as optimal. I also planned the silhouette scores of each increasing k, revealing 5 clusters as a point with a relatively high score above 0.3. This silhouette score represents the quality of the clusters we created in terms of how well similar data points are clustered with each other. Since our score is close to zero, some of our clusters poorly perform with the overlapping and dense data. However, our model still produced meaningful clusters with functional characteristics. After identifying and choosing 5 clusters for our model, we plotted the cluster assignments between steals vs. blocks, blocks vs. total fouls, and field goals vs. steals. Steals vs. blocks show how a game is played defensively by what method they are protecting the ball from the opposing team, gaining possession and ultimately points. Of the 5 clusters in Steals vs. blocks, the red cluster has games with more steals and some blocks. The green cluster shows games with more blocks and only some steals. This indicates that we can cluster games with more defense led through steals or blocks. In the blocks vs. total fouls plot, we can see how defensive play with blocks affects total fouls. In teal cluster 2, there are low blocks and high fouls. In the green cluster, there are high blocks and somewhat low fouls. The teal cluster can be seen as games that receive more fouls from other types of play than defensive plan. The green cluster shows games played more defensively through blocks have fewer fouls through offensive play. While our model was able to identify these clusters, there is a low accuracy because of the density and overlapping of the data. Using another method of clustering, we could further analyze these cluster relationships. However, KMeans was somewhat accurately able to reveal these unique cluster relationships for these predictors.

2 Jared Q's

2.1 Jared Q1

When predicting the odds of a team winning a game, what predictors matter most?

[28]:
```
def WL_binary(x):
  if x == 'W': return 1
  else: return 0

nba['Win'] = nba['WINorLOSS'].apply(WL_binary)
nba.dropna()
nba.head()
```

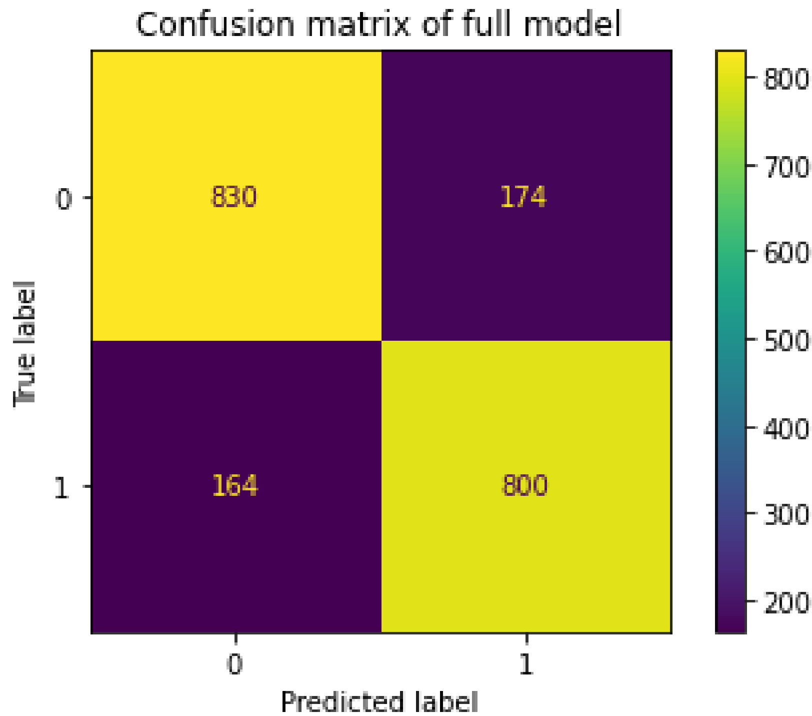
[28]:
```
   Unnamed: 0 Team  Game        Date  Home Opponent WINorLOSS  TeamPoints  \
0           1  ATL     1  2014-10-29  Away      TOR         L   -0.135577
1           2  ATL     2  2014-11-01  Home      IND         W   -0.135577
2           3  ATL     3  2014-11-05  Away      SAS         L   -0.956095
3           4  ATL     4  2014-11-07  Away      CHO         L    1.259303
4           5  ATL     5  2014-11-08  Home      NYK         W   -0.053525

   OpponentPoints  FieldGoals  …  Opp.FreeThrowsAttempted  Opp.FreeThrows.  \
0        0.438785    0.277860  …                 1.387078         0.533169
1       -0.956095   -0.716228  …                -0.236722         0.907120
2       -0.791992   -0.119775  …                 2.063661        -0.492800
3        1.505459    0.874312  …                 0.575178        -0.205145
4       -0.627888   -1.113863  …                -1.589888        -0.339384

   Opp.OffRebounds  Opp.TotalRebounds  Opp.Assists  Opp.Steals  Opp.Blocks  \
0         1.500527           0.698799     0.674144    1.774235    1.644787
1         0.187132           0.074784     0.478935   -0.929623    0.067945
2         0.187132           1.010806     0.478935   -0.253658    1.644787
3         0.187132           1.166810     1.650186   -0.591641    0.856366
4         0.712490           0.074784     0.674144   -1.943570    0.462156
```

```
   Opp.Turnovers  Opp.TotalFouls  Win
0      -1.198812        0.449684    0
1       1.127162        1.376170    1
2       1.385604       -1.171665    0
3       1.385604        2.302655    0
4       0.351838        2.071034    1

[5 rows x 42 columns]
```

[29]:
```
predictors = ['TeamPoints', 'FieldGoals.', 'X3PointShots.', 'FreeThrows.',␣
 ↪'OffRebounds',
              'TotalRebounds', 'Assists', 'Steals', 'Blocks', 'Turnovers',␣
 ↪'TotalFouls']

y = nba['Win']
X = nba[predictors]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

z = StandardScaler()
z.fit(X_train)
X_train = z.transform(X_train)
X_test = z.transform(X_test)


full_logit = LogisticRegression(penalty = "none")
full_logit.fit(X_train, y_train)


predvals_full_logit = full_logit.predict(X_test)

print("training mean squared error:", mean_squared_error(y_train, full_logit.
 ↪predict(X_train)))
print("testing mean squared error:", mean_squared_error(y_test,␣
 ↪predvals_full_logit))
print("accuracy score:", accuracy_score(y_test, predvals_full_logit))
plot_confusion_matrix(full_logit, X_test, y_test)
plt.title('Confusion matrix of full model')
```

```
training mean squared error: 0.18038617886178862
testing mean squared error: 0.1717479674796748
accuracy score: 0.8282520325203252
```

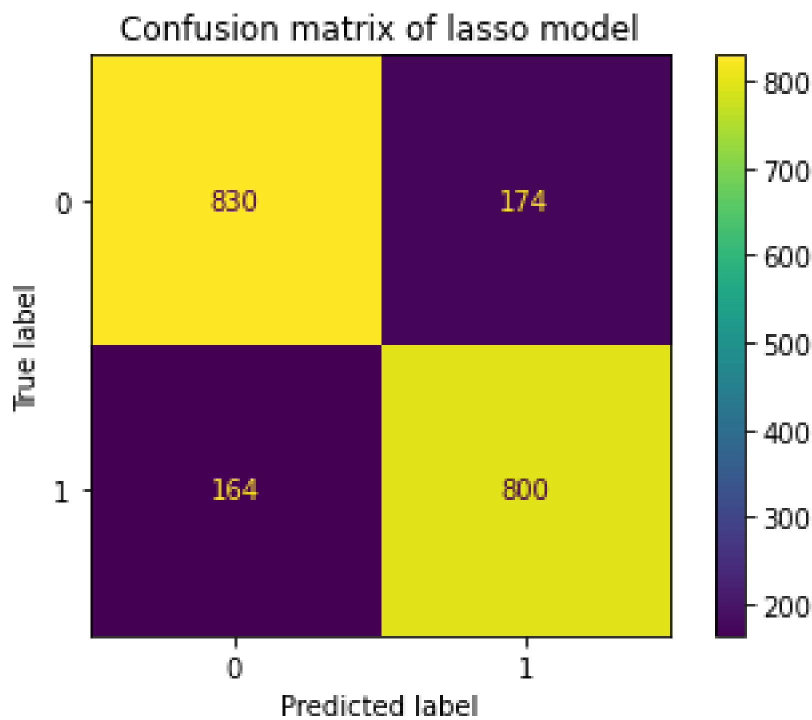
[29]:
```
Text(0.5, 1.0, 'Confusion matrix of full model')
```



[30]:
```
lasso_logit = LogisticRegression(penalty = "l1", solver="liblinear")
lasso_logit.fit(X_train, y_train)

predvals_lasso_logit = lasso_logit.predict(X_test)

print("training mean squared error:", mean_squared_error(y_train, lasso_logit.
 ↪predict(X_train)))
print("testing mean squared error:", mean_squared_error(y_test,␣
 ↪predvals_lasso_logit))
print("accuracy score:", accuracy_score(y_test, predvals_lasso_logit))
plot_confusion_matrix(lasso_logit, X_test, y_test)
plt.title('Confusion matrix of lasso model')
```

```
training mean squared error: 0.18076727642276422
testing mean squared error: 0.1717479674796748
accuracy score: 0.8282520325203252
```

[30]: Text(0.5, 1.0, 'Confusion matrix of lasso model')



[31]:
```
print(full_logit.coef_)
print(full_logit.intercept_)
print(lasso_logit.coef_)
print(lasso_logit.intercept_)
```

```
[[-0.23999873  1.94025377  0.6165299   0.44457963 -0.39518198  1.93424845
  -0.14223603  0.93086456  0.25871721 -0.83535996 -0.30253809]]
[0.05021917]
[[-0.23146583  1.92742035  0.61237887  0.44100354 -0.39324488  1.92374058
  -0.13982792  0.92512679  0.25744023 -0.82968805 -0.30224902]]
[0.04881413]
```

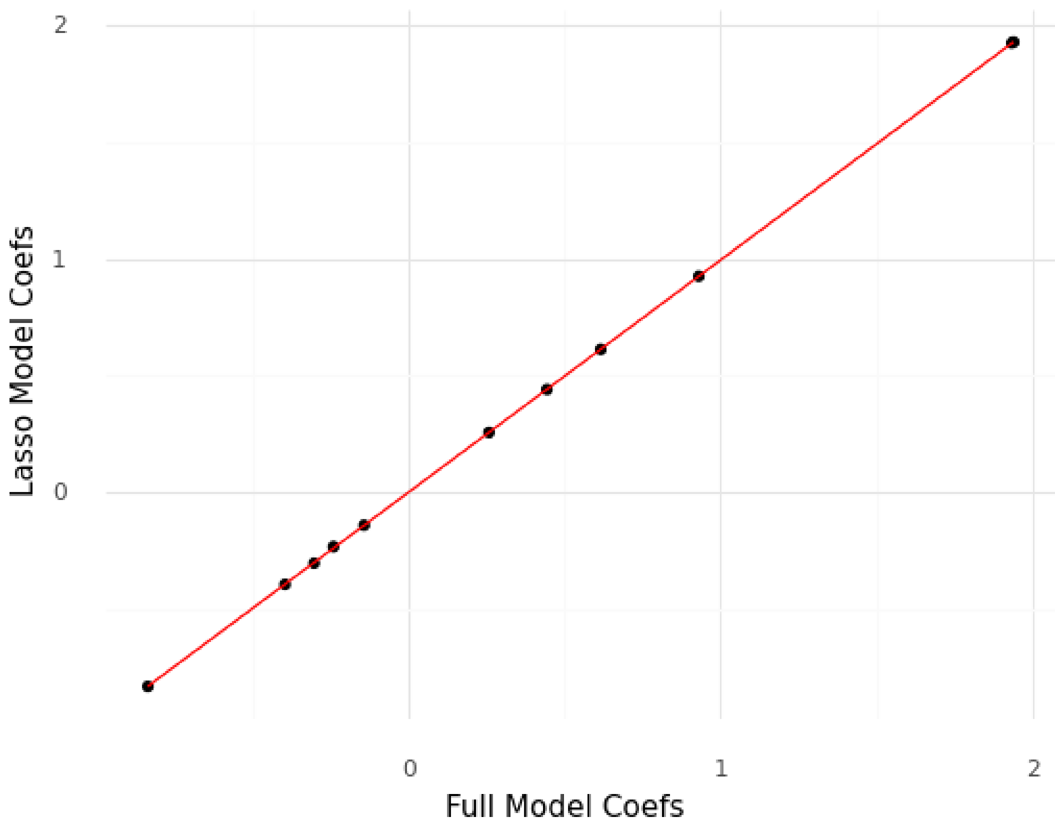
[32]:
```
coef_comp = pd.DataFrame({"Names": predictors,
                          "Full Model Coefs": full_logit.coef_[0],
                          "Lasso Model Coefs": lasso_logit.coef_[0]})
coef_comp["Full Model Odds Coefs"] = np.exp(coef_comp["Full Model Coefs"])
coef_comp["Lasso Model Odds Coefs"]= np.exp(coef_comp["Lasso Model Coefs"])
coef_comp
```

[32]:
```
             Names  Full Model Coefs  Lasso Model Coefs  Full Model Odds Coefs  \
0       TeamPoints         -0.239999          -0.231466               0.786629
1      FieldGoals.          1.940254           1.927420               6.960517
2   X3PointShots.          0.616530           0.612379               1.852489
3      FreeThrows.          0.444580           0.441004               1.559834
```

```
4     OffRebounds         -0.395182          -0.393245                 0.673557
5   TotalRebounds          1.934248           1.923741                 6.918842
6         Assists         -0.142236          -0.139828                 0.867416
7          Steals          0.930865           0.925127                 2.536701
8          Blocks          0.258717           0.257440                 1.295267
9       Turnovers         -0.835360          -0.829688                 0.433718
10     TotalFouls         -0.302538          -0.302249                 0.738940

    Lasso Model Odds Coefs
0                 0.793370
1                 6.871761
2                 1.844815
3                 1.554266
4                 0.674863
5                 6.846521
6                 0.869508
7                 2.522188
8                 1.293614
9                 0.436185
10                0.739154
```

[33]:

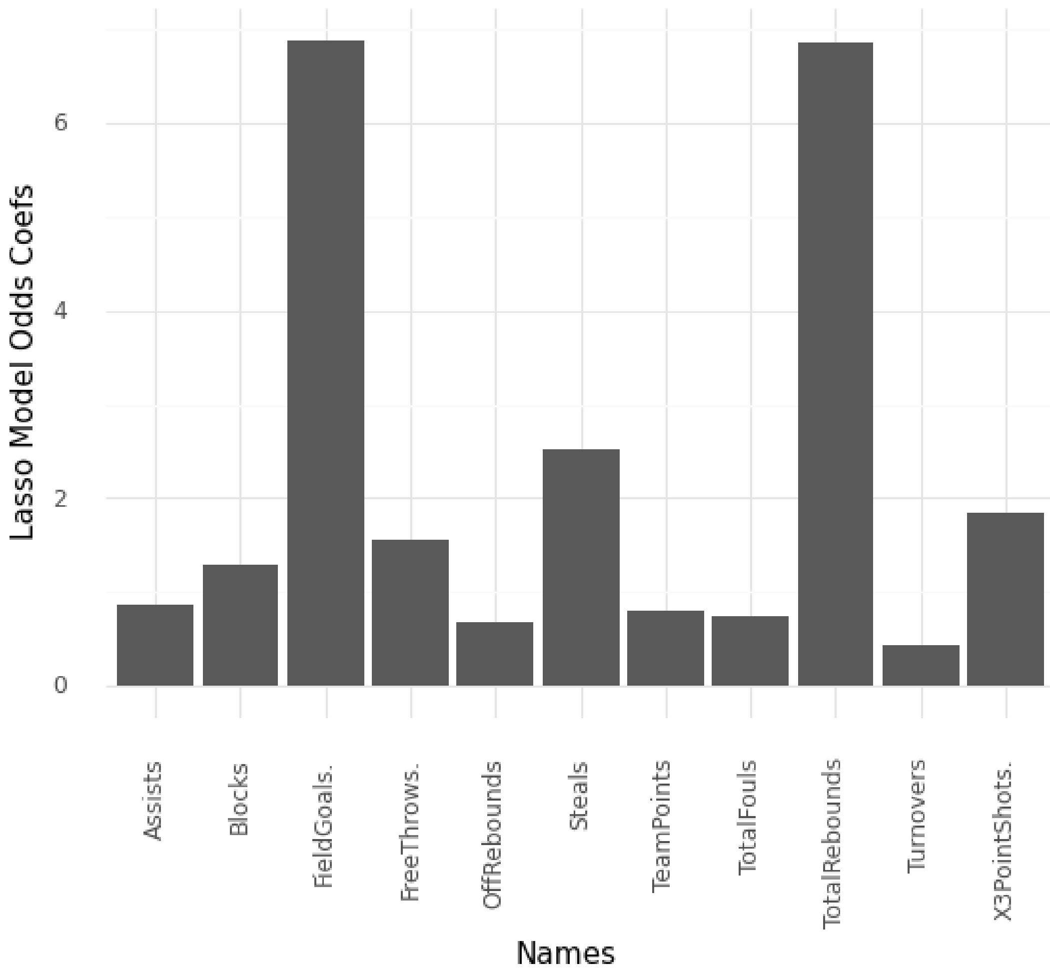
```
ggplot(coef_comp, aes("Full Model Coefs", "Lasso Model Coefs")) + geom_point()␣
↪+ theme_minimal() + geom_smooth(method = "lm", color = "red", size = .5,␣
↪alpha = 0.1)
```



[33]: <ggplot: (8760364707013)>

Adding a penalty to the model did not change the coefficient very much from the full model. The relationship is basically a 1:1 ratio.

[34]:
```
ggplot(coef_comp, aes(x = "Names", y = "Lasso Model Odds Coefs")) +␣
↪theme_minimal() + geom_bar(stat = "identity")+ theme(axis_text_x  =␣
↪element_text(angle = 90))
```



[34]: <ggplot: (8760366391053)>

Field goal percentage and total rebounds improve the odds of winning the most.

We see that adding a lasso penalty to the logisitic regression did not do much, so we do not have too many variables in the model. Lasso was used over ridge because we wanted to see if there was any automatic feature reduction, which there wasn't. If there was, we could instantly take those features out of consideration for which matter the most in choosing the most important predictors in winning. Since there wasn't much change from the full model to the lasso penalty model, we can assume that all of these features have some kind of contribution to winning, which makes sense since winning a basketball game does include many factors, many of which do not even have explicit stats for.

To answer the question, out of the features used, field goal percentage and total rebounds matter most to increase a team's odds of winning a basketball game. I came to this conclusion because increasing the field goal percentage and total rebounds by one standard deviation of their respective statistic, will increase the odds of winning much more than the other statistics (see last graph). However, correlation does not equal causation. I think it's pretty obvious that if a team shoots more efficiently, they will have a higher chance of winning. I was surprised to see the total rebounds affect the chance of winning so much though. I think we should look into possible interpretations of this though. Although rebounds don't directly affect the score, a rebound does mean there was a missed shot. If it was an offensive rebound, that means another chance to score for that team. If it was a defensive rebound, that means the opposing team missed and now you get a possession and a chance to score.

2.2 Jared Q2

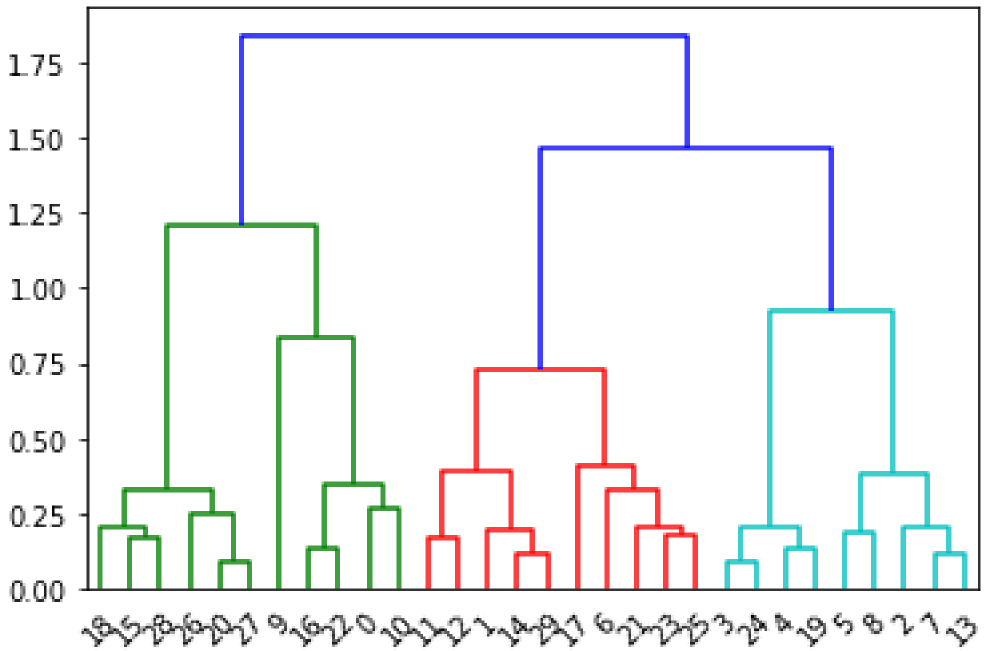
When considering steals, blocks, opp_turnovers, and opp_fieldgoals_2, are there clear defensively strong and weak teams?

[35]:
```
team_defense = nba.dropna().groupby(["Team"]).mean()
team_defense.columns
team_defense = team_defense.drop(columns = ['Unnamed: 0', 'Game', 'TeamPoints',␣
 ↪'OpponentPoints', 'FieldGoals',
        'FieldGoalsAttempted', 'FieldGoals.', 'X3PointShots',
        'X3PointShotsAttempted', 'X3PointShots.', 'FreeThrows',
        'FreeThrowsAttempted', 'FreeThrows.', 'OffRebounds', 'TotalRebounds',
        'Assists', 'Turnovers',
        'Opp.FieldGoals', 'Opp.FieldGoalsAttempted',
        'Opp.3PointShots', 'Opp.3PointShotsAttempted', 'Opp.3PointShots.',
        'Opp.FreeThrows', 'Opp.FreeThrowsAttempted', 'Opp.FreeThrows.',
        'Opp.OffRebounds', 'Opp.TotalRebounds', 'Opp.Assists', 'Opp.Steals',
        'Opp.Blocks', 'Opp.TotalFouls','TotalFouls', 'Win'])
team_defense.head()
```

[35]:

Team	Steals	Blocks	Opp.FieldGoals.	Opp.Turnovers
ATL	0.266711	0.033091	-0.155097	0.366021
BOS	0.108024	-0.273383	-0.174212	0.180069
BRK	-0.241293	-0.161610	0.188913	-0.202078
CHI	-0.272206	0.053523	-0.066880	-0.369908
CHO	-0.327850	0.078762	-0.041077	-0.256445

[36]:
```
features = ['Steals', 'Blocks', 'Opp.FieldGoals.', 'Opp.Turnovers']
X = team_defense[features]

hac = AgglomerativeClustering( affinity = "euclidean",
                              linkage = "ward")
hac.fit(X)

dendro = sch.dendrogram(sch.linkage(X, method='ward'))
```

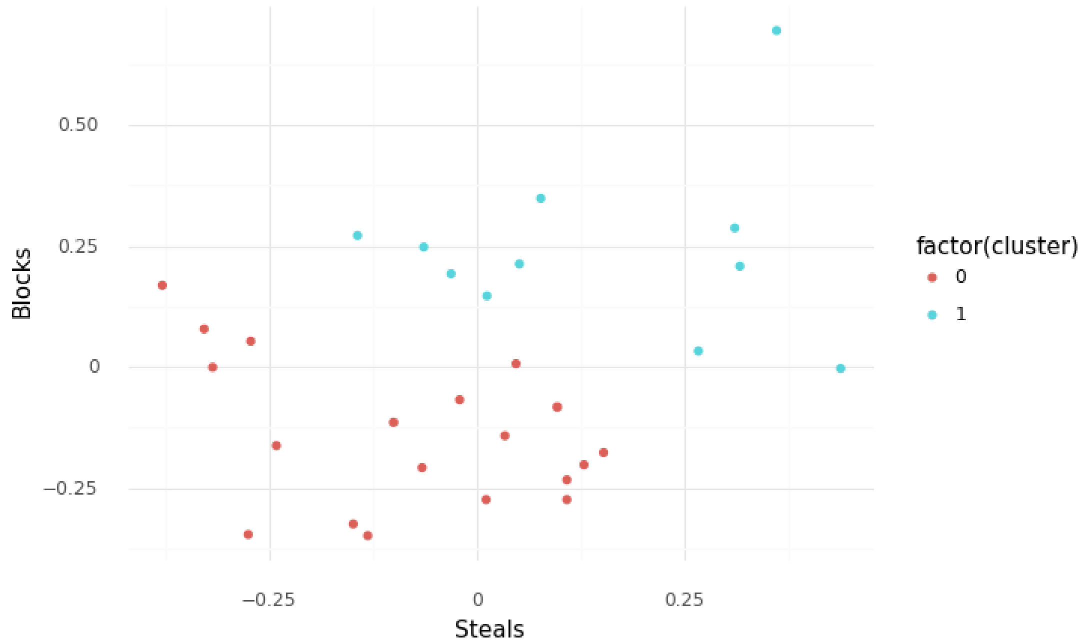


Dendrogram created two clusters of teams

[37]:
```
membership = hac.labels_
print(silhouette_score(X,membership))
team_defense['cluster'] = membership
```

```
0.2756858241796067
```

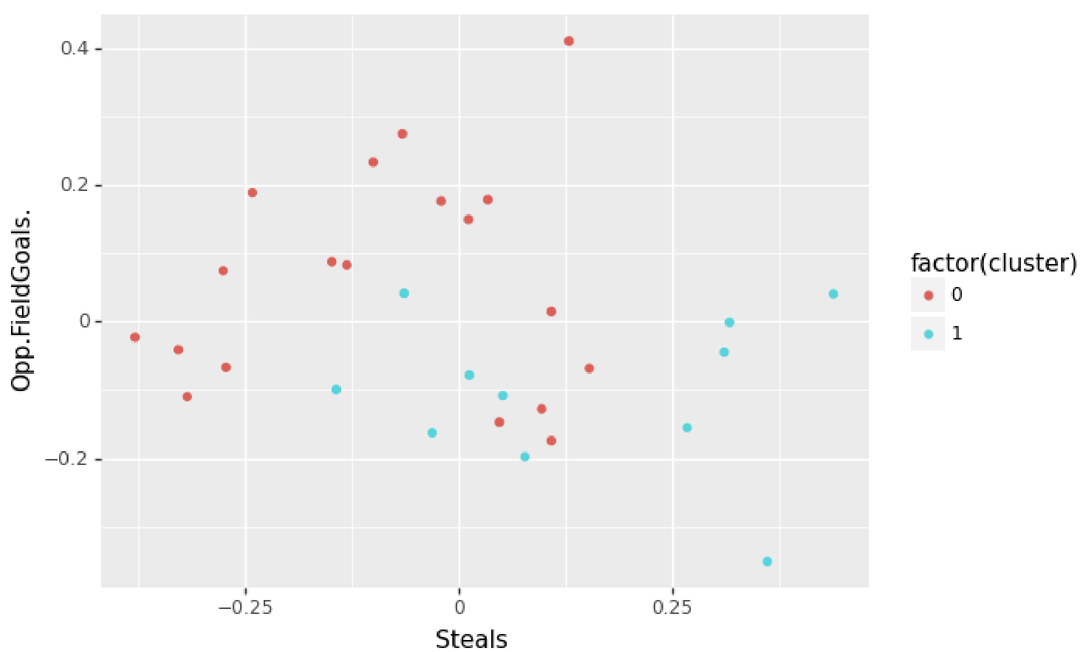
[38]:
```
(ggplot(team_defense, aes(x = "Steals", y = "Blocks")) + geom_point(aes(color = "factor(cluster)"))) + theme_minimal()
```



[38]: <ggplot: (8760364588625)>

Steals vs Blocks show two odd shaped clusters that are close but not overlapped.

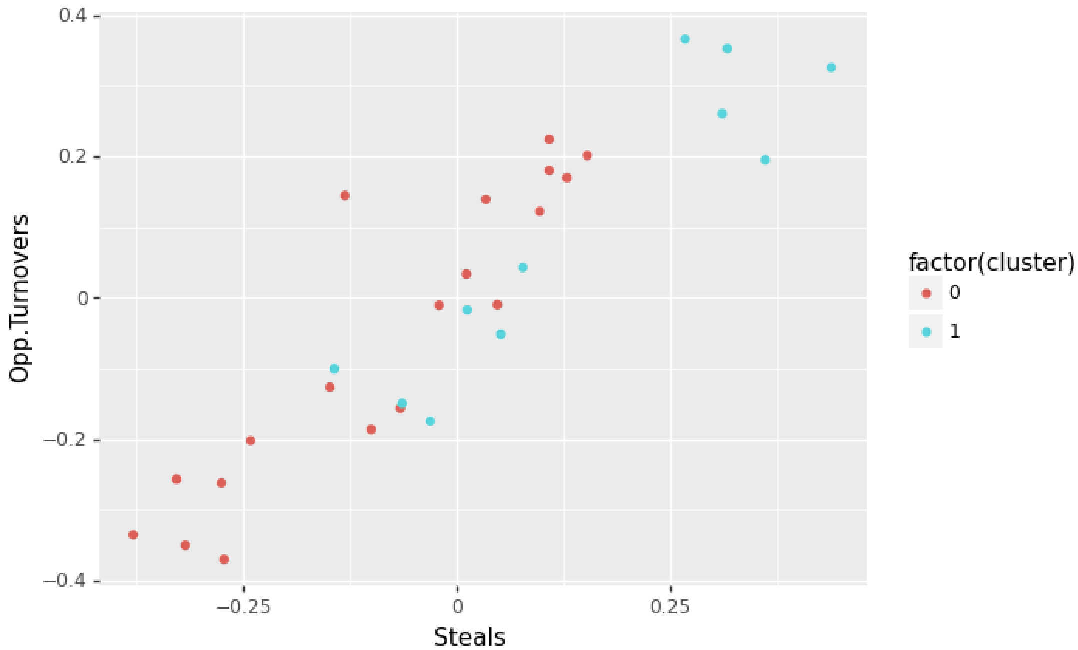
[39]:
```
(ggplot(team_defense, aes(x = "Steals", y = "Opp.FieldGoals.")) +
 geom_point(aes(color = "factor(cluster)")))
```



[39]: <ggplot: (8760364676305)>

Steals vs Opponent Field Goal Percentage shows two clusters that are close and overlapping a bit

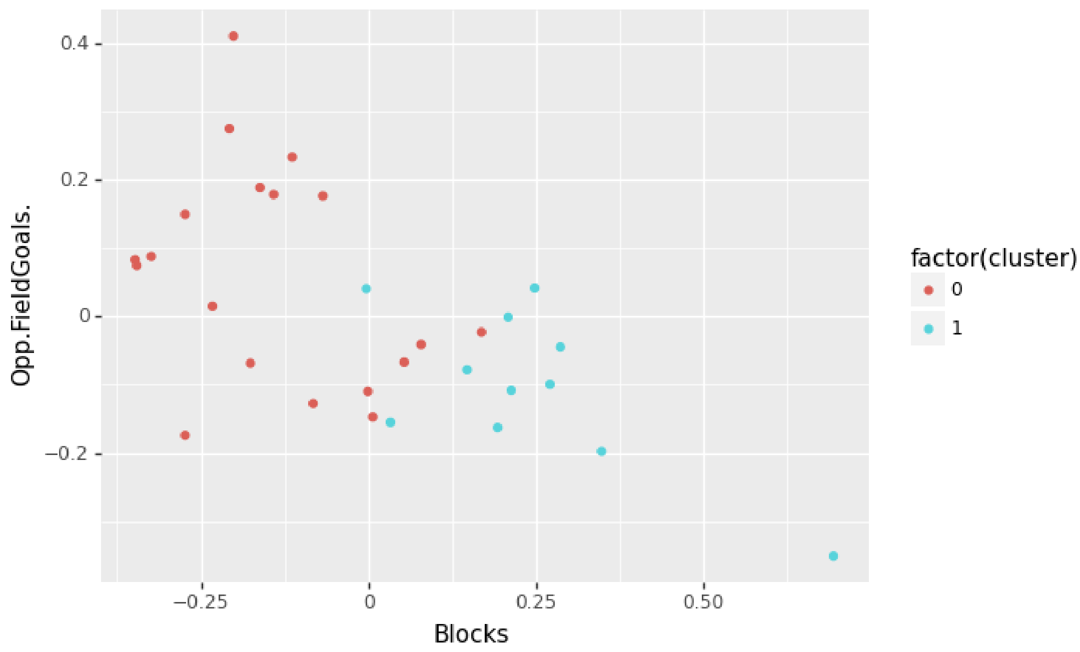
[40]:
```
(ggplot(team_defense, aes(x = "Steals", y = "Opp.Turnovers")) +
 ↪geom_point(aes(color = "factor(cluster)")))
```



[40]: <ggplot: (8760364560089)>

Steals vs Opponent Turnovers show two clusters that look like one ellipse shaped cluster, but cut in half. The they are close, but not overlapping.

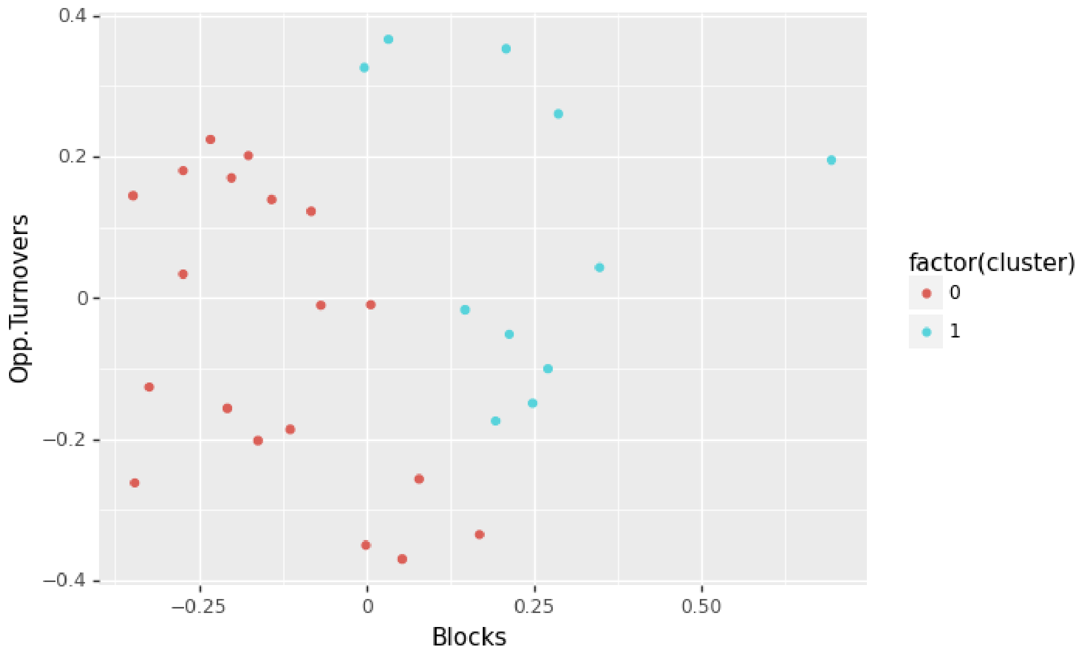
[41]:
```
(ggplot(team_defense, aes(x = "Blocks", y = "Opp.FieldGoals.")) +
 ↪geom_point(aes(color = "factor(cluster)")))
```



[41]: <ggplot: (8760364607765)>

Blocks vs Opponent Field Goal Percentage shows two very overlapping clusters and it's hard to make out anything from it. The clusters made do not show much relationship when it comes to blocks and field goal percent.

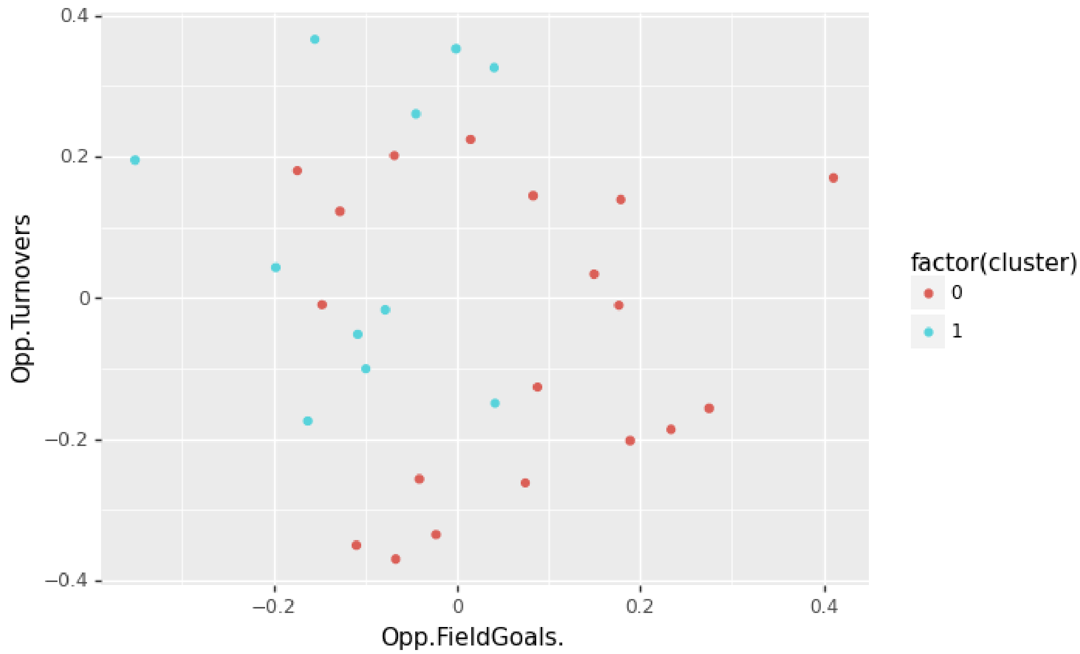
[42]:
```
(ggplot(team_defense, aes(x = "Blocks", y = "Opp.Turnovers")) +␣
↪geom_point(aes(color = "factor(cluster)")))
```



[42]: <ggplot: (8760364539809)>

Blocks vs Opponent Turnovers show two wide clusters. It looks like there is a split of higher turnovers and lower turnovers and blocks are varied and do not matter as much in splitting the clusters.

[43]:
```
(ggplot(team_defense, aes(x = "Opp.FieldGoals.", y = "Opp.Turnovers")) +␣
↪geom_point(aes(color = "factor(cluster)")))
```



[43]: <ggplot: (8760364545545)>

Opponent Field Goal Percentage vs Opponent Turnovers show two wide clusters. It looks like there is a split of higher turnovers and lower turnovers but the opponent field goal percentage does not carry much weight in the cluster splitting.

Just from looking at the dendrogram, we can see two clusters that are clearly split because the height of the first split. If we look deeper into the relationships of the clusters between the variables we see that there are certain variables that matter way more than others in clustering. Steals and Opponent Turnovers matter way more than the other variables. I think that this cluster model shows better and worse stealing teams as opposed to defense as a whole.

Defense is much more complicated than what the statistics show since blocks and steals are not the only things that create disruptions on the opponent's offense. This is why I added opponent's turnovers and field goal percentage. Not every opponent turnover is from a steal and not every missed shot is a block. A team could pressure the opponent into rushing a pass or obscuring their vision. They could also almost block a shot, which could make the opponent change their shot midair and miss. Also, it's hard to make all of your shots even if they are uncontested.

In conclusion, there are not completely clear defensively stronger teams when considering steals, blocks, opp_turnovers, and opp_fieldgoals_2 and defense is much harder to measure than just using the limited variables in this model.

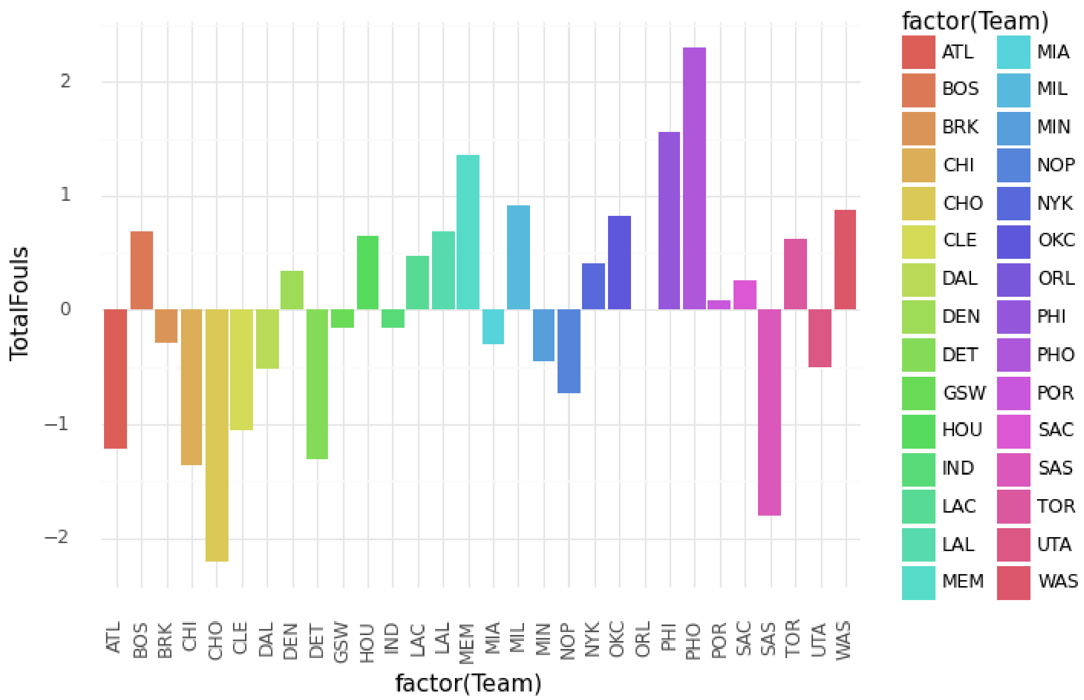
3 Darren Q's

3.1 Darren Q1

[44]:
```
wr = []
for i in range(0, len(nba)):
    if nba.iloc[i].WINorLOSS == "W" :
      wr.append(1)
    elif nba.iloc[i].WINorLOSS == "L" :
      wr.append(0)

nba["winrate"] = wr

foul_df = nba.groupby(["Team"], as_index = False)[["TotalFouls"]].mean()
```

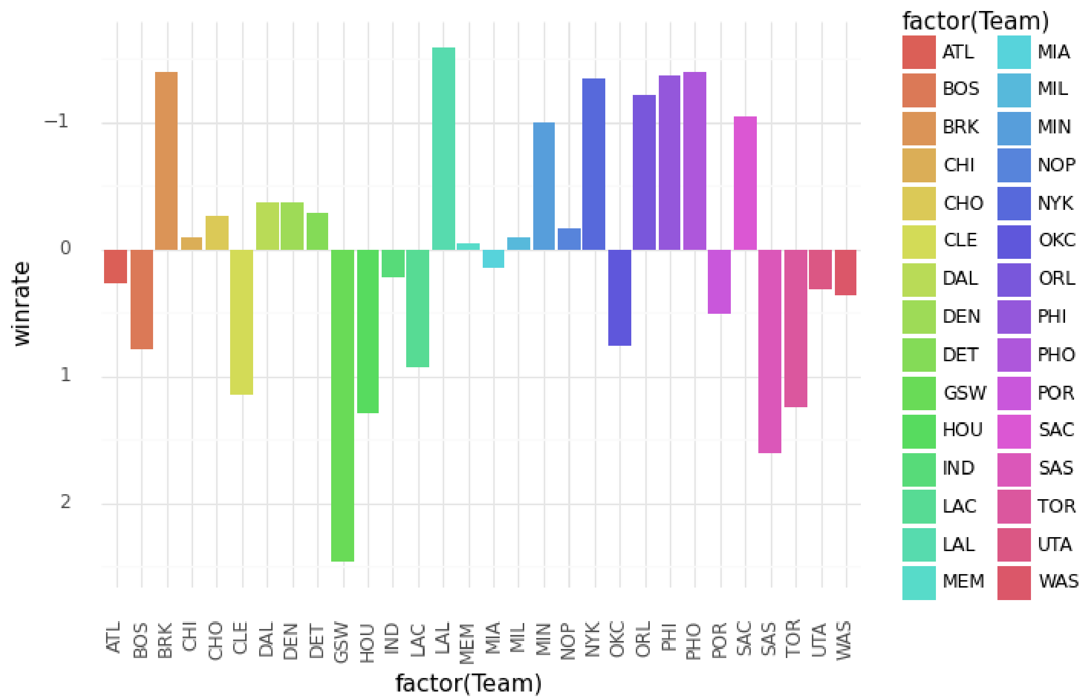
[45]:
```
wr_df = nba.groupby(["Team"], as_index = False)[["winrate"]].mean()
wr_df["AvgFouls"] = foul_df["TotalFouls"]

zscore3 = StandardScaler()
zscore3 = zscore3.fit(wr_df[["winrate"]])
wr_df["winrate"] = zscore3.transform(wr_df[["winrate"]])

zscore4 = StandardScaler()
zscore4 = zscore4.fit(foul_df[["TotalFouls"]])
foul_df["TotalFouls"] = zscore4.transform(foul_df[["TotalFouls"]])
```

[46]:
```
(ggplot(foul_df , aes(x = "factor(Team)", y= "TotalFouls", fill =␣
↪"factor(Team)")) + geom_bar(stat = "identity") +  theme_minimal() +␣
↪theme(axis_text_x  = element_text(angle = 90)))
```



[46]: <ggplot: (8760364517565)>

[47]:
```
(ggplot(wr_df , aes(x = "factor(Team)", y= "winrate", fill = "factor(Team)")) +
→geom_bar(stat = "identity") +  theme_minimal() + theme(axis_text_x  =
→element_text(angle = 90)) + scale_y_reverse())
```



[47]: <ggplot: (8760369688873)>

[48]:
```
pred_p1 = ["TotalFouls"]
X_p1  = nba[pred_p1]
y_p1 = nba["winrate"]

X_train_p1, X_test_p1, y_train_p1, y_test_p1 = train_test_split(X_p1,y_p1,␣
 ↪test_size = 0.15)
zscore_p1 = StandardScaler()
zscore_p1.fit(X_train_p1[pred_p1])
Xz_train_p1 = X_train_p1
Xz_test_p1 = X_test_p1
Xz_train_p1[pred_p1] = zscore_p1.transform(X_train_p1[pred_p1])
Xz_test_p1[pred_p1] = zscore_p1.transform(Xz_test_p1[pred_p1])

lr_p1 = LogisticRegression()
lr_p1.fit(Xz_train_p1,y_train_p1)
```

[48]: LogisticRegression()

[49]:
```
# prediction for test
rt_pred_test_p1 = lr_p1.predict(X_test_p1)
actual_vs_pred_test_p1 = pd.DataFrame({"predicted": rt_pred_test_p1,
```

```
                                            "actual": y_test_p1})

# prediction for train
rt_pred_train_p1 = lr_p1.predict(X_train_p1)
actual_vs_pred_train_p1 = pd.DataFrame({"predicted": rt_pred_train_p1,
                                             "actual": y_train_p1})
```

[50]:
```
# model evaluation
#r2 test
r2test_p1 = lr_p1.score(X_test_p1, y_test_p1)
print("test R2: " + str(r2test_p1))

#r2 train
r2train_p1 = lr_p1.score(X_train_p1, y_train_p1)
print("train R2:" + str(r2train_p1))
```

```
test R2: 0.5440379403794038
train R2:0.5399330463892874
```

[51]:
```
MSEtest_p1 = mean_squared_error(y_test_p1,rt_pred_test_p1 )
MSEtrain_p1 = mean_squared_error(y_train_p1,rt_pred_train_p1)
print("MSE test: " + str(MSEtest_p1))
print("MSE train: " + str(MSEtrain_p1))
```

```
MSE test: 0.4559620596205962
MSE train: 0.4600669536107126
```

[52]:
```
coef = pd.DataFrame({"Coefs": lr_p1.coef_[0],
                     "Names": pred_p1})

coef
```

[52]:

	Coefs	Names
0	-0.232406	TotalFouls

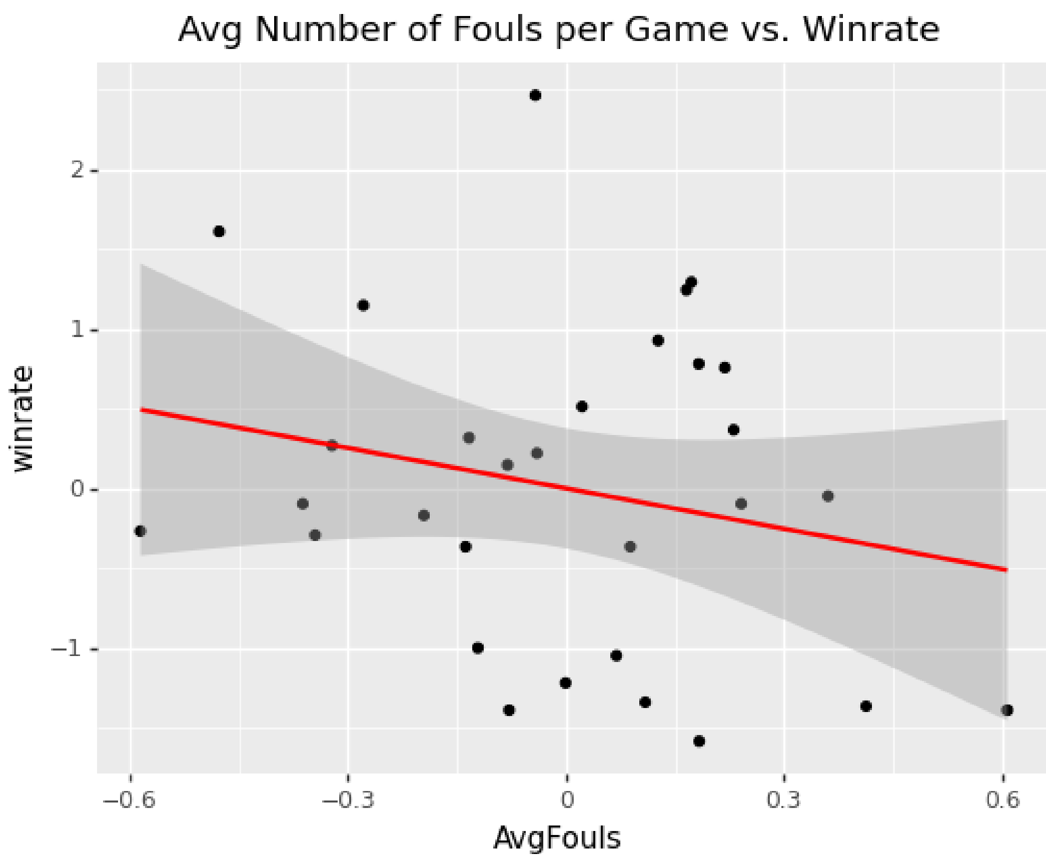
[53]:
```
print(wr_df)
```

```
   Team   winrate  AvgFouls
0   ATL  0.268394 -0.321445
1   BOS  0.780782  0.182755
2   BRK -1.390768 -0.077819
3   CHI -0.097598 -0.361697
4   CHO -0.268394 -0.584844
5   CLE  1.146774 -0.278369
6   DAL -0.365992 -0.137843
7   DEN -0.365992  0.088835
8   DET -0.292793 -0.344749
9   GSW  2.464343 -0.041805
```

```
10  HOU  1.293170  0.172869
11  IND  0.219595 -0.039686
12  LAC  0.927179  0.126968
13  LAL -1.585963  0.183461
14  MEM -0.048799  0.360708
15  MIA  0.146397 -0.079938
16  MIL -0.097598  0.241366
17  MIN -1.000377 -0.120895
18  NOP -0.170796 -0.195042
19  NYK -1.341969  0.109314
20  OKC  0.756383  0.218769
21  ORL -1.219972 -0.000141
22  PHI -1.366369  0.412964
23  PHO -1.390768  0.607159
24  POR  0.512388  0.022456
25  SAC -1.049176  0.069769
26  SAS  1.610363 -0.476801
27  TOR  1.244371  0.165807
28  UTA  0.317193 -0.132900
29  WAS  0.365992  0.230774
```

[54]:

```
(ggplot(wr_df, aes(x = "AvgFouls", y = "winrate")) + geom_point() +␣
↪geom_smooth(method = "lm", color = "red") + labs(title = "Avg Number of␣
↪Fouls per Game vs. Winrate"))
```



[54]: <ggplot: (8760364463581)>

[55]:
```
y_pred_p1 = lr_p1.predict(X_test_p1)

lrAcc = accuracy_score(y_pred_p1, y_test_p1)
print("Logistic Regression Accuracy:" + str(lrAcc))
```

```
Logistic Regression Accuracy:0.5440379403794038
```

3.1.1 Q1 Discussion

Fouls can be an indicator of "unsportsmanlike" behavior and this will be an analysis of how this kind of behavior can affect the winrate of the team. The graphs show some aggregated statistics for the teams throughout the season. For winrate, the y-axis has been inverted for effect to show similarities between the total number of fouls and the winrate. The team with the highest number of fouls is Phoenix. Overall, the graphs indicate that having a higher number of TotalFouls generally means a lower winrate for the team.

Graphing both statistics, there is a negative relationship between winrate and average number of fouls. This is another perspective from the data shown earlier, again confirming that higher number of fouls will lower winrate. A negative coefficient from the logistics regression model confirms the hypothesis about "unsportsmanlike" behavior lowering winrate. However, an interesting outlier is the teams with highest/lowest winrate are both close to the average. The accuracy score is not perfect, with only 52% of test data predicted correctly. This type of analysis can be important to coaches or other trainers who want to consider spending time with their teams on skills outside of performance. This example shows that sportsmanship can be proven to be beneficial to a team, both ethically and from a performance perspective.

3.2 Darren Q2

[56]:
```
pred_p2 = ["FreeThrows", "FreeThrowsAttempted", "FreeThrows.", "FieldGoals",␣
↪"FieldGoalsAttempted", "FieldGoals."]
X_p2  = nba[pred_p2]
y_p2 = nba["X3PointShots."]

X_train_p2, X_test_p2, y_train_p2, y_test_p2 = train_test_split(X_p2,y_p2,␣
↪test_size = 0.15)
zscore_p2 = StandardScaler()
zscore_p2.fit(X_train_p2[pred_p2])
Xz_train_p2 = X_train_p2
Xz_test_p2 = X_test_p2
Xz_train_p2[pred_p2] = zscore_p2.transform(X_train_p2[pred_p2])
Xz_test_p2[pred_p2] = zscore_p2.transform(Xz_test_p2[pred_p2])

lr_p2 = LinearRegression()
lr_p2.fit(Xz_train_p2,y_train_p2)
```

[56]: LinearRegression()

[57]:
```
# prediction for test
rt_pred_test_p2 = lr_p2.predict(X_test_p2)
actual_vs_pred_test_p2 = pd.DataFrame({"predicted": rt_pred_test_p2,
                                  "actual": y_test_p2})

# prediction for train
rt_pred_train_p2 = lr_p2.predict(X_train_p2)
actual_vs_pred_train_p2 = pd.DataFrame({"predicted": rt_pred_train_p2,
                                   "actual": y_train_p2})
```

[58]:
```
# model evaluation
#r2 test
r2test_p2 = lr_p2.score(X_test_p2, y_test_p2)
print("test R2: " + str(r2test_p2))

#r2 train
r2train_p2 = lr_p2.score(X_train_p2, y_train_p2)
```

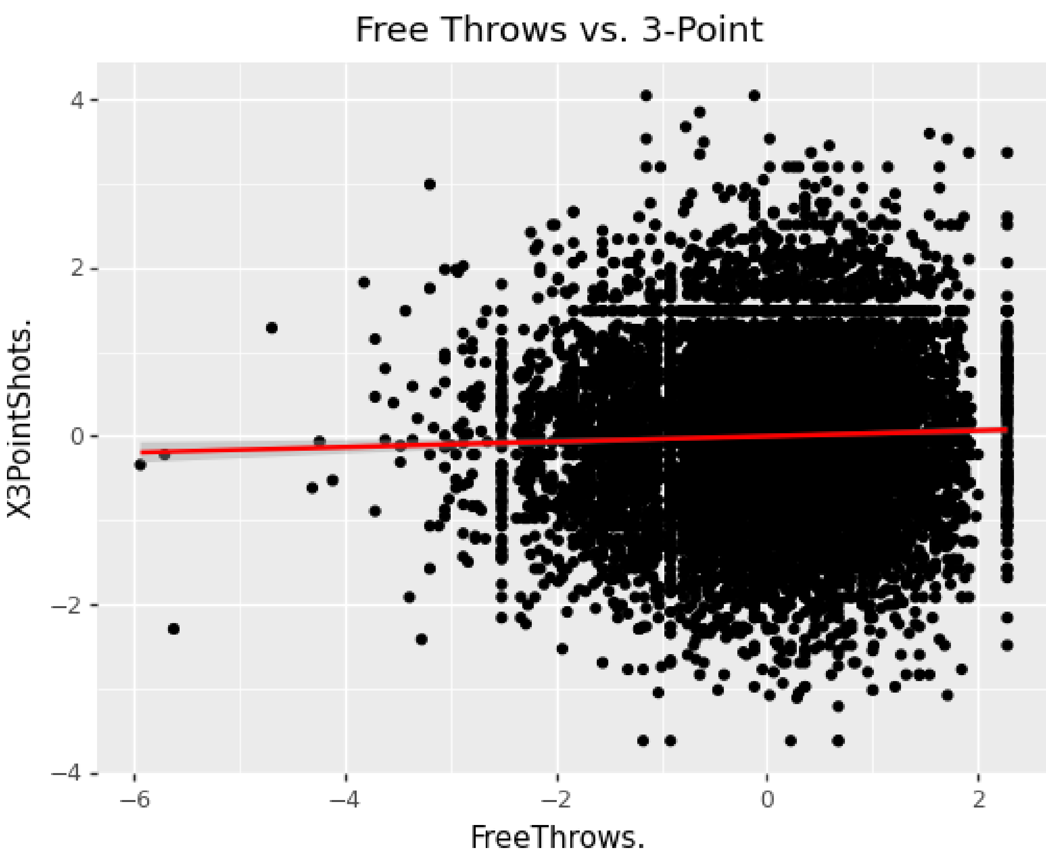
```
print("train R2:" + str(r2train_p2))
```

```
test R2: 0.24766142915156708
train R2:0.24775892850114745
```

[59]:
```
MSEtest_p2 = mean_squared_error(y_test_p2,rt_pred_test_p2 )
MSEtrain_p2 = mean_squared_error(y_train_p2,rt_pred_train_p2)
print("MSE test: " + str(MSEtest_p2))
print("MSE train: " + str(MSEtrain_p2))
```

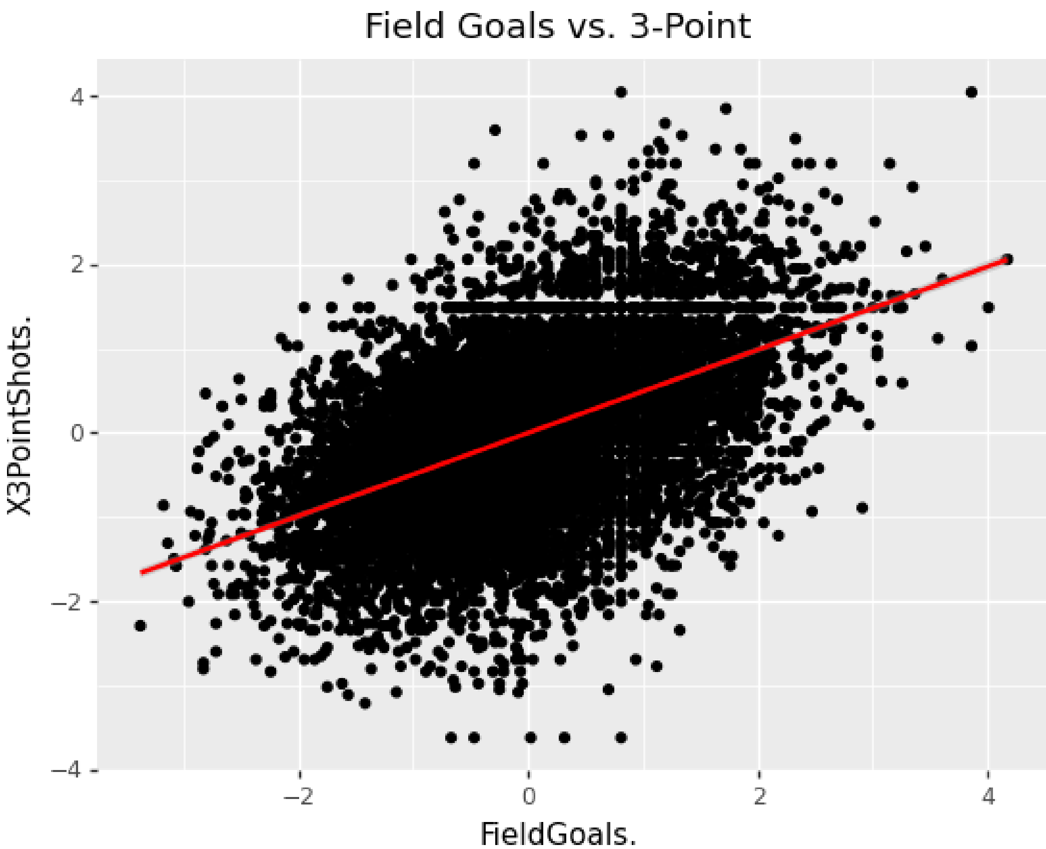
```
MSE test: 0.7645259080960332
MSE train: 0.7500800458452026
```

[60]:
```
(ggplot(nba, aes(x = "FreeThrows.", y = "X3PointShots.")) + geom_point() +␣
↪geom_smooth(method = "lm", color = "red") + labs(title = "Free Throws vs.␣
↪3-Point"))
```



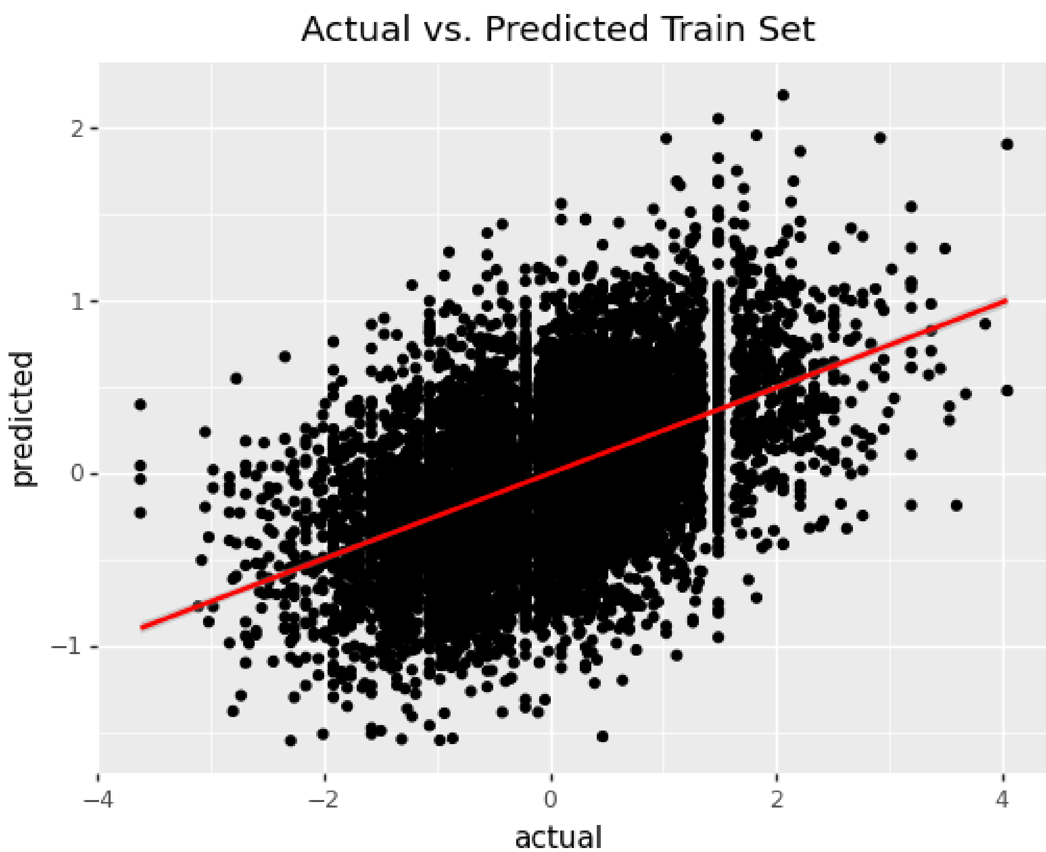
[60]: <ggplot: (8760364401797)>

[61]:
```
(ggplot(nba, aes(x = "FieldGoals.", y = "X3PointShots.")) + geom_point() +␣
↪geom_smooth(method = "lm", color = "red") + labs(title = "Field Goals vs.␣
↪3-Point"))
```



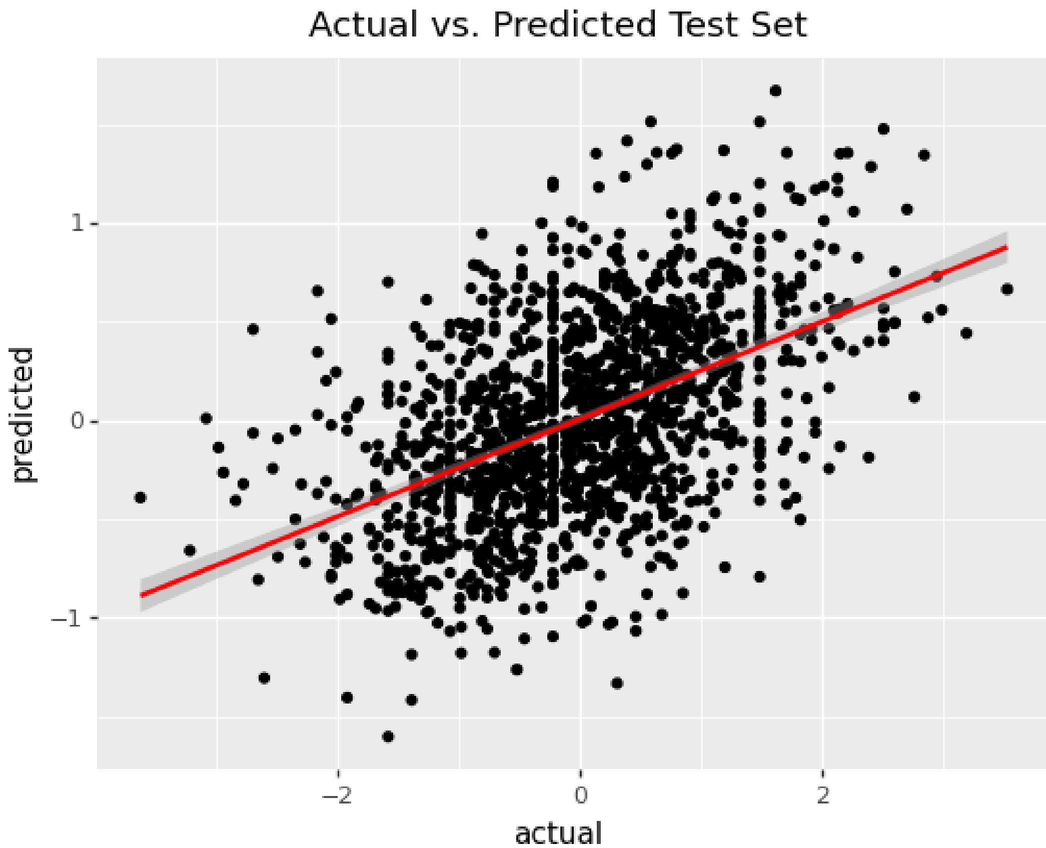
[61]: <ggplot: (8760364389289)>

[62]:
```
# actual vs. predicted train
(ggplot(actual_vs_pred_train_p2, aes(x = "actual", y = "predicted")) +␣
↪geom_point() + geom_smooth(method = "lm", color = "red") + labs(title =␣
↪"Actual vs. Predicted Train Set"))
```



[62]: <ggplot: (8760364282553)>

[63]:
```
# actual vs. predicted test
(ggplot(actual_vs_pred_test_p2, aes(x = "actual", y = "predicted")) +␣
↪geom_point() + geom_smooth(method = "lm", color = "red") + labs(title =␣
↪"Actual vs. Predicted Test Set"))
```



[63]: <ggplot: (8760364359261)>

[64]:
```
coef2 = pd.DataFrame({"Coefs": lr_p2.coef_[0],
                      "Names": pred_p2})

coef2
```

[64]:

	Coefs	Names
0	0.074354	FreeThrows
1	0.074354	FreeThrowsAttempted
2	0.074354	FreeThrows.
3	0.074354	FieldGoals
4	0.074354	FieldGoalsAttempted
5	0.074354	FieldGoals.

3.2.1 Q2 Discussion

In short, yes. Using a linear regression model, we can compare different goal averages to predict 3-point averages. This analysis serves the purpose of identifying if shot statistics at different ranges are related to each other and if performance at one indicates performance in other categories. From the raw data, the comparison graphs show that there is a strong relationship between Field Goals and 3-point shot averages. However, free throw averages has a weak relationship with 3-point average with most of the data biased towards higher free throw percentages.

The R2 and MSE for both data sets show very low performance for the Linear Regression Model. With only 24-25% of data being predicted accurately it is not a great model for predicting the 3-point average. This may be due to the high amount of data involved OR could be attributed to the weak relationship between free throw average and 3-point average. However, the graphs still show a positive relationship for the predictors and 3-point average, so in this case we conclude that yes there is a correlation between free throw average, field goal average, and 3-point average.

[ ]:
```
# doesn't show this cells output when downloading PDF
!pip install gwpy &> /dev/null

# installing necessary files
!apt-get install texlive texlive-xetex texlive-latex-extra pandoc
!sudo apt-get update
!sudo apt-get install texlive-xetex texlive-fonts-recommended␣
↪texlive-plain-generic

# installing pypandoc
!pip install pypandoc

# connecting your google drive
from google.colab import drive
drive.mount('/content/drive')

# copying your file over. Change "Class6-Completed.ipynb" to whatever your file␣
↪is called (see top of notebook)
!cp "drive/My Drive/Colab Notebooks/NBA_Final.ipynb" ./

# Again, replace "Class6-Completed.ipynb" to whatever your file is called (see␣
↪top of notebook)
!jupyter nbconvert --to PDF "NBA_Final.ipynb"
```

```
Reading package lists… Done
Building dependency tree
Reading state information… Done
pandoc is already the newest version (1.19.2.4~dfsg-1build4).
pandoc set to manually installed.
The following package was automatically installed and is no longer required:
  libnvidia-common-460
Use 'apt autoremove' to remove it.
The following additional packages will be installed:
  fonts-droid-fallback fonts-lato fonts-lmodern fonts-noto-mono fonts-texgyre
  javascript-common libcupsfilters1 libcupsimage2 libgs9 libgs9-common
```

```
  libijs-0.35 libjbig2dec0 libjs-jquery libkpathsea6 libpotrace0 libptexenc1
  libruby2.5 libsynctex1 libtexlua52 libtexluajit2 libzzip-0-13 lmodern
  poppler-data preview-latex-style rake ruby ruby-did-you-mean ruby-minitest
  ruby-net-telnet ruby-power-assert ruby-test-unit ruby2.5
  rubygems-integration t1utils tex-common tex-gyre texlive-base
  texlive-binaries texlive-fonts-recommended texlive-latex-base
  texlive-latex-recommended texlive-pictures texlive-plain-generic tipa
Suggested packages:
  fonts-noto apache2 | lighttpd | httpd poppler-utils ghostscript
  fonts-japanese-mincho | fonts-ipafont-mincho fonts-japanese-gothic
  | fonts-ipafont-gothic fonts-arphic-ukai fonts-arphic-uming fonts-nanum ri
  ruby-dev bundler debhelper gv | postscript-viewer perl-tk xpdf-reader
  | pdf-viewer texlive-fonts-recommended-doc texlive-latex-base-doc
  python-pygments icc-profiles libfile-which-perl
  libspreadsheet-parseexcel-perl texlive-latex-extra-doc
  texlive-latex-recommended-doc texlive-pstricks dot2tex prerex ruby-tcltk
  | libtcltk-ruby texlive-pictures-doc vprerex
The following NEW packages will be installed:
  fonts-droid-fallback fonts-lato fonts-lmodern fonts-noto-mono fonts-texgyre
  javascript-common libcupsfilters1 libcupsimage2 libgs9 libgs9-common
  libijs-0.35 libjbig2dec0 libjs-jquery libkpathsea6 libpotrace0 libptexenc1
  libruby2.5 libsynctex1 libtexlua52 libtexluajit2 libzzip-0-13 lmodern
  poppler-data preview-latex-style rake ruby ruby-did-you-mean ruby-minitest
  ruby-net-telnet ruby-power-assert ruby-test-unit ruby2.5
  rubygems-integration t1utils tex-common tex-gyre texlive texlive-base
  texlive-binaries texlive-fonts-recommended texlive-latex-base
  texlive-latex-extra texlive-latex-recommended texlive-pictures
  texlive-plain-generic texlive-xetex tipa
0 upgraded, 47 newly installed, 0 to remove and 42 not upgraded.
Need to get 146 MB of archives.
After this operation, 460 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic/main amd64 fonts-droid-fallback
all 1:6.0.1r16-1.1 [1,805 kB]
Get:2 http://archive.ubuntu.com/ubuntu bionic/main amd64 fonts-lato all 2.0-2
[2,698 kB]
Get:3 http://archive.ubuntu.com/ubuntu bionic/main amd64 poppler-data all
0.4.8-2 [1,479 kB]
Get:4 http://archive.ubuntu.com/ubuntu bionic/main amd64 tex-common all 6.09
[33.0 kB]
Get:5 http://archive.ubuntu.com/ubuntu bionic/main amd64 fonts-lmodern all
2.004.5-3 [4,551 kB]
Get:6 http://archive.ubuntu.com/ubuntu bionic/main amd64 fonts-noto-mono all
20171026-2 [75.5 kB]
Get:7 http://archive.ubuntu.com/ubuntu bionic/universe amd64 fonts-texgyre all
20160520-1 [8,761 kB]
Get:8 http://archive.ubuntu.com/ubuntu bionic/main amd64 javascript-common all
11 [6,066 B]
Get:9 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libcupsfilters1
```

```
amd64 1.20.2-0ubuntu3.1 [108 kB]
Get:10 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libcupsimage2
amd64 2.2.7-1ubuntu2.8 [18.6 kB]
Get:11 http://archive.ubuntu.com/ubuntu bionic/main amd64 libijs-0.35 amd64
0.35-13 [15.5 kB]
Get:12 http://archive.ubuntu.com/ubuntu bionic/main amd64 libjbig2dec0 amd64
0.13-6 [55.9 kB]
Get:13 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libgs9-common
all 9.26~dfsg+0-0ubuntu0.18.04.16 [5,093 kB]
Get:14 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libgs9 amd64
9.26~dfsg+0-0ubuntu0.18.04.16 [2,265 kB]
Get:15 http://archive.ubuntu.com/ubuntu bionic/main amd64 libjs-jquery all
3.2.1-1 [152 kB]
Get:16 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libkpathsea6
amd64 2017.20170613.44572-8ubuntu0.1 [54.9 kB]
Get:17 http://archive.ubuntu.com/ubuntu bionic/main amd64 libpotrace0 amd64
1.14-2 [17.4 kB]
Get:18 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libptexenc1
amd64 2017.20170613.44572-8ubuntu0.1 [34.5 kB]
Get:19 http://archive.ubuntu.com/ubuntu bionic/main amd64 rubygems-integration
all 1.11 [4,994 B]
Get:20 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 ruby2.5 amd64
2.5.1-1ubuntu1.11 [48.6 kB]
Get:21 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby amd64 1:2.5.1
[5,712 B]
Get:22 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 rake all
12.3.1-1ubuntu0.1 [44.9 kB]
Get:23 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby-did-you-mean all
1.2.0-2 [9,700 B]
Get:24 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby-minitest all
5.10.3-1 [38.6 kB]
Get:25 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby-net-telnet all
0.1.1-2 [12.6 kB]
Get:26 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby-power-assert all
0.3.0-1 [7,952 B]
Get:27 http://archive.ubuntu.com/ubuntu bionic/main amd64 ruby-test-unit all
3.2.5-1 [61.1 kB]
Get:28 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libruby2.5
amd64 2.5.1-1ubuntu1.11 [3,072 kB]
Get:29 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libsynctex1
amd64 2017.20170613.44572-8ubuntu0.1 [41.4 kB]
Get:30 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libtexlua52
amd64 2017.20170613.44572-8ubuntu0.1 [91.2 kB]
Get:31 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libtexluajit2
amd64 2017.20170613.44572-8ubuntu0.1 [230 kB]
Get:32 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 libzzip-0-13
amd64 0.13.62-3.1ubuntu0.18.04.1 [26.0 kB]
Get:33 http://archive.ubuntu.com/ubuntu bionic/main amd64 lmodern all 2.004.5-3
```

```
[9,631 kB]
Get:34 http://archive.ubuntu.com/ubuntu bionic/main amd64 preview-latex-style
all 11.91-1ubuntu1 [185 kB]
Get:35 http://archive.ubuntu.com/ubuntu bionic/main amd64 t1utils amd64 1.41-2
[56.0 kB]
Get:36 http://archive.ubuntu.com/ubuntu bionic/universe amd64 tex-gyre all
20160520-1 [4,998 kB]
Get:37 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 texlive-
binaries amd64 2017.20170613.44572-8ubuntu0.1 [8,179 kB]
Get:38 http://archive.ubuntu.com/ubuntu bionic/main amd64 texlive-base all
2017.20180305-1 [18.7 MB]
Get:39 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive-fonts-
recommended all 2017.20180305-1 [5,262 kB]
Get:40 http://archive.ubuntu.com/ubuntu bionic/main amd64 texlive-latex-base all
2017.20180305-1 [951 kB]
Get:41 http://archive.ubuntu.com/ubuntu bionic/main amd64 texlive-latex-
recommended all 2017.20180305-1 [14.9 MB]
Get:42 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive all
2017.20180305-1 [14.4 kB]
Get:43 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive-pictures
all 2017.20180305-1 [4,026 kB]
Get:44 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive-latex-
extra all 2017.20180305-2 [10.6 MB]
Get:45 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive-plain-
generic all 2017.20180305-2 [23.6 MB]
Get:46 http://archive.ubuntu.com/ubuntu bionic/universe amd64 tipa all 2:1.3-20
[2,978 kB]
Get:47 http://archive.ubuntu.com/ubuntu bionic/universe amd64 texlive-xetex all
2017.20180305-1 [10.7 MB]
Fetched 146 MB in 5s (31.0 MB/s)
Extracting templates from packages: 100%
Preconfiguring packages …
Selecting previously unselected package fonts-droid-fallback.
(Reading database … 155629 files and directories currently installed.)
Preparing to unpack …/00-fonts-droid-fallback_1%3a6.0.1r16-1.1_all.deb …
Unpacking fonts-droid-fallback (1:6.0.1r16-1.1) …
Selecting previously unselected package fonts-lato.
Preparing to unpack …/01-fonts-lato_2.0-2_all.deb …
Unpacking fonts-lato (2.0-2) …
Selecting previously unselected package poppler-data.
Preparing to unpack …/02-poppler-data_0.4.8-2_all.deb …
Unpacking poppler-data (0.4.8-2) …
Selecting previously unselected package tex-common.
Preparing to unpack …/03-tex-common_6.09_all.deb …
Unpacking tex-common (6.09) …
Selecting previously unselected package fonts-lmodern.
Preparing to unpack …/04-fonts-lmodern_2.004.5-3_all.deb …
Unpacking fonts-lmodern (2.004.5-3) …
```

```
Selecting previously unselected package fonts-noto-mono.
Preparing to unpack …/05-fonts-noto-mono_20171026-2_all.deb …
Unpacking fonts-noto-mono (20171026-2) …
Selecting previously unselected package fonts-texgyre.
Preparing to unpack …/06-fonts-texgyre_20160520-1_all.deb …
Unpacking fonts-texgyre (20160520-1) …
Selecting previously unselected package javascript-common.
Preparing to unpack …/07-javascript-common_11_all.deb …
Unpacking javascript-common (11) …
Selecting previously unselected package libcupsfilters1:amd64.
Preparing to unpack …/08-libcupsfilters1_1.20.2-0ubuntu3.1_amd64.deb …
Unpacking libcupsfilters1:amd64 (1.20.2-0ubuntu3.1) …
Selecting previously unselected package libcupsimage2:amd64.
Preparing to unpack …/09-libcupsimage2_2.2.7-1ubuntu2.8_amd64.deb …
Unpacking libcupsimage2:amd64 (2.2.7-1ubuntu2.8) …
Selecting previously unselected package libijs-0.35:amd64.
Preparing to unpack …/10-libijs-0.35_0.35-13_amd64.deb …
Unpacking libijs-0.35:amd64 (0.35-13) …
Selecting previously unselected package libjbig2dec0:amd64.
Preparing to unpack …/11-libjbig2dec0_0.13-6_amd64.deb …
Unpacking libjbig2dec0:amd64 (0.13-6) …
Selecting previously unselected package libgs9-common.
Preparing to unpack …/12-libgs9-common_9.26~dfsg+0-0ubuntu0.18.04.16_all.deb
…
Unpacking libgs9-common (9.26~dfsg+0-0ubuntu0.18.04.16) …
Selecting previously unselected package libgs9:amd64.
Preparing to unpack …/13-libgs9_9.26~dfsg+0-0ubuntu0.18.04.16_amd64.deb …
Unpacking libgs9:amd64 (9.26~dfsg+0-0ubuntu0.18.04.16) …
Selecting previously unselected package libjs-jquery.
Preparing to unpack …/14-libjs-jquery_3.2.1-1_all.deb …
Unpacking libjs-jquery (3.2.1-1) …
Selecting previously unselected package libkpathsea6:amd64.
Preparing to unpack …/15-libkpathsea6_2017.20170613.44572-8ubuntu0.1_amd64.deb
…
Unpacking libkpathsea6:amd64 (2017.20170613.44572-8ubuntu0.1) …
Selecting previously unselected package libpotrace0.
Preparing to unpack …/16-libpotrace0_1.14-2_amd64.deb …
Unpacking libpotrace0 (1.14-2) …
Selecting previously unselected package libptexenc1:amd64.
Preparing to unpack …/17-libptexenc1_2017.20170613.44572-8ubuntu0.1_amd64.deb
…
Unpacking libptexenc1:amd64 (2017.20170613.44572-8ubuntu0.1) …
Selecting previously unselected package rubygems-integration.
Preparing to unpack …/18-rubygems-integration_1.11_all.deb …
Unpacking rubygems-integration (1.11) …
Selecting previously unselected package ruby2.5.
Preparing to unpack …/19-ruby2.5_2.5.1-1ubuntu1.11_amd64.deb …
Unpacking ruby2.5 (2.5.1-1ubuntu1.11) …
```

```
Selecting previously unselected package ruby.
Preparing to unpack …/20-ruby_1%3a2.5.1_amd64.deb …
Unpacking ruby (1:2.5.1) …
Selecting previously unselected package rake.
Preparing to unpack …/21-rake_12.3.1-1ubuntu0.1_all.deb …
Unpacking rake (12.3.1-1ubuntu0.1) …
Selecting previously unselected package ruby-did-you-mean.
Preparing to unpack …/22-ruby-did-you-mean_1.2.0-2_all.deb …
Unpacking ruby-did-you-mean (1.2.0-2) …
Selecting previously unselected package ruby-minitest.
Preparing to unpack …/23-ruby-minitest_5.10.3-1_all.deb …
Unpacking ruby-minitest (5.10.3-1) …
Selecting previously unselected package ruby-net-telnet.
Preparing to unpack …/24-ruby-net-telnet_0.1.1-2_all.deb …
Unpacking ruby-net-telnet (0.1.1-2) …
Selecting previously unselected package ruby-power-assert.
Preparing to unpack …/25-ruby-power-assert_0.3.0-1_all.deb …
Unpacking ruby-power-assert (0.3.0-1) …
Selecting previously unselected package ruby-test-unit.
Preparing to unpack …/26-ruby-test-unit_3.2.5-1_all.deb …
Unpacking ruby-test-unit (3.2.5-1) …
Selecting previously unselected package libruby2.5:amd64.
Preparing to unpack …/27-libruby2.5_2.5.1-1ubuntu1.11_amd64.deb …
Unpacking libruby2.5:amd64 (2.5.1-1ubuntu1.11) …
Selecting previously unselected package libsynctex1:amd64.
Preparing to unpack …/28-libsynctex1_2017.20170613.44572-8ubuntu0.1_amd64.deb
…
Unpacking libsynctex1:amd64 (2017.20170613.44572-8ubuntu0.1) …
Selecting previously unselected package libtexlua52:amd64.
Preparing to unpack …/29-libtexlua52_2017.20170613.44572-8ubuntu0.1_amd64.deb
…
Unpacking libtexlua52:amd64 (2017.20170613.44572-8ubuntu0.1) …
Selecting previously unselected package libtexluajit2:amd64.
Preparing to unpack
…/30-libtexluajit2_2017.20170613.44572-8ubuntu0.1_amd64.deb …
Unpacking libtexluajit2:amd64 (2017.20170613.44572-8ubuntu0.1) …
Selecting previously unselected package libzzip-0-13:amd64.
Preparing to unpack …/31-libzzip-0-13_0.13.62-3.1ubuntu0.18.04.1_amd64.deb …
Unpacking libzzip-0-13:amd64 (0.13.62-3.1ubuntu0.18.04.1) …
Selecting previously unselected package lmodern.
Preparing to unpack …/32-lmodern_2.004.5-3_all.deb …
Unpacking lmodern (2.004.5-3) …
Selecting previously unselected package preview-latex-style.
Preparing to unpack …/33-preview-latex-style_11.91-1ubuntu1_all.deb …
Unpacking preview-latex-style (11.91-1ubuntu1) …
Selecting previously unselected package t1utils.
Preparing to unpack …/34-t1utils_1.41-2_amd64.deb …
Unpacking t1utils (1.41-2) …
```

```
Selecting previously unselected package tex-gyre.
Preparing to unpack …/35-tex-gyre_20160520-1_all.deb …
Unpacking tex-gyre (20160520-1) …
Selecting previously unselected package texlive-binaries.
Preparing to unpack …/36-texlive-
binaries_2017.20170613.44572-8ubuntu0.1_amd64.deb …
Unpacking texlive-binaries (2017.20170613.44572-8ubuntu0.1) …
Selecting previously unselected package texlive-base.
Preparing to unpack …/37-texlive-base_2017.20180305-1_all.deb …
Unpacking texlive-base (2017.20180305-1) …
Selecting previously unselected package texlive-fonts-recommended.
Preparing to unpack …/38-texlive-fonts-recommended_2017.20180305-1_all.deb …
Unpacking texlive-fonts-recommended (2017.20180305-1) …
Selecting previously unselected package texlive-latex-base.
Preparing to unpack …/39-texlive-latex-base_2017.20180305-1_all.deb …
Unpacking texlive-latex-base (2017.20180305-1) …
Selecting previously unselected package texlive-latex-recommended.
Preparing to unpack …/40-texlive-latex-recommended_2017.20180305-1_all.deb …
Unpacking texlive-latex-recommended (2017.20180305-1) …
Selecting previously unselected package texlive.
Preparing to unpack …/41-texlive_2017.20180305-1_all.deb …
Unpacking texlive (2017.20180305-1) …
Selecting previously unselected package texlive-pictures.
Preparing to unpack …/42-texlive-pictures_2017.20180305-1_all.deb …
Unpacking texlive-pictures (2017.20180305-1) …
Selecting previously unselected package texlive-latex-extra.
Preparing to unpack …/43-texlive-latex-extra_2017.20180305-2_all.deb …
Unpacking texlive-latex-extra (2017.20180305-2) …
Selecting previously unselected package texlive-plain-generic.
Preparing to unpack …/44-texlive-plain-generic_2017.20180305-2_all.deb …
Unpacking texlive-plain-generic (2017.20180305-2) …
Selecting previously unselected package tipa.
Preparing to unpack …/45-tipa_2%3a1.3-20_all.deb …
Unpacking tipa (2:1.3-20) …
Selecting previously unselected package texlive-xetex.
Preparing to unpack …/46-texlive-xetex_2017.20180305-1_all.deb …
Unpacking texlive-xetex (2017.20180305-1) …
Setting up libgs9-common (9.26~dfsg+0-0ubuntu0.18.04.16) …
Setting up libkpathsea6:amd64 (2017.20170613.44572-8ubuntu0.1) …
Setting up libjs-jquery (3.2.1-1) …
Setting up libtexlua52:amd64 (2017.20170613.44572-8ubuntu0.1) …
Setting up fonts-droid-fallback (1:6.0.1r16-1.1) …
Setting up libsynctex1:amd64 (2017.20170613.44572-8ubuntu0.1) …
Setting up libptexenc1:amd64 (2017.20170613.44572-8ubuntu0.1) …
Setting up tex-common (6.09) …
update-language: texlive-base not installed and configured, doing nothing!
Setting up poppler-data (0.4.8-2) …
Setting up tex-gyre (20160520-1) …
```

```
Setting up preview-latex-style (11.91-1ubuntu1) …
Setting up fonts-texgyre (20160520-1) …
Setting up fonts-noto-mono (20171026-2) …
Setting up fonts-lato (2.0-2) …
Setting up libcupsfilters1:amd64 (1.20.2-0ubuntu3.1) …
Setting up libcupsimage2:amd64 (2.2.7-1ubuntu2.8) …
Setting up libjbig2dec0:amd64 (0.13-6) …
Setting up ruby-did-you-mean (1.2.0-2) …
Setting up t1utils (1.41-2) …
Setting up ruby-net-telnet (0.1.1-2) …
Setting up libijs-0.35:amd64 (0.35-13) …
Setting up rubygems-integration (1.11) …
Setting up libpotrace0 (1.14-2) …
Setting up javascript-common (11) …
Setting up ruby-minitest (5.10.3-1) …
Setting up libzzip-0-13:amd64 (0.13.62-3.1ubuntu0.18.04.1) …
Setting up libgs9:amd64 (9.26~dfsg+0-0ubuntu0.18.04.16) …
Setting up libtexluajit2:amd64 (2017.20170613.44572-8ubuntu0.1) …
Setting up fonts-lmodern (2.004.5-3) …
Setting up ruby-power-assert (0.3.0-1) …
Setting up texlive-binaries (2017.20170613.44572-8ubuntu0.1) …
update-alternatives: using /usr/bin/xdvi-xaw to provide /usr/bin/xdvi.bin
(xdvi.bin) in auto mode
update-alternatives: using /usr/bin/bibtex.original to provide /usr/bin/bibtex
(bibtex) in auto mode
Setting up texlive-base (2017.20180305-1) …
mktexlsr: Updating /var/lib/texmf/ls-R-TEXLIVEDIST…
mktexlsr: Updating /var/lib/texmf/ls-R-TEXMFMAIN…
mktexlsr: Updating /var/lib/texmf/ls-R…
mktexlsr: Done.
tl-paper: setting paper size for dvips to a4:
/var/lib/texmf/dvips/config/config-paper.ps
tl-paper: setting paper size for dvipdfmx to a4:
/var/lib/texmf/dvipdfmx/dvipdfmx-paper.cfg
tl-paper: setting paper size for xdvi to a4: /var/lib/texmf/xdvi/XDvi-paper
tl-paper: setting paper size for pdftex to a4:
/var/lib/texmf/tex/generic/config/pdftexconfig.tex
Setting up texlive-fonts-recommended (2017.20180305-1) …
Setting up texlive-plain-generic (2017.20180305-2) …
Setting up texlive-latex-base (2017.20180305-1) …
Setting up lmodern (2.004.5-3) …
Setting up texlive-latex-recommended (2017.20180305-1) …
Setting up texlive-pictures (2017.20180305-1) …
Setting up tipa (2:1.3-20) …
Regenerating '/var/lib/texmf/fmtutil.cnf-DEBIAN'… done.
Regenerating '/var/lib/texmf/fmtutil.cnf-TEXLIVEDIST'… done.
update-fmtutil has updated the following file(s):
        /var/lib/texmf/fmtutil.cnf-DEBIAN
```

```
        /var/lib/texmf/fmtutil.cnf-TEXLIVEDIST
If you want to activate the changes in the above file(s),
you should run fmtutil-sys or fmtutil.
Setting up texlive (2017.20180305-1) …
Setting up texlive-latex-extra (2017.20180305-2) …
Setting up texlive-xetex (2017.20180305-1) …
Setting up ruby2.5 (2.5.1-1ubuntu1.11) …
Setting up ruby (1:2.5.1) …
Setting up ruby-test-unit (3.2.5-1) …
Setting up rake (12.3.1-1ubuntu0.1) …
Setting up libruby2.5:amd64 (2.5.1-1ubuntu1.11) …
Processing triggers for mime-support (3.60ubuntu1) …
Processing triggers for libc-bin (2.27-3ubuntu1.3) …
/sbin/ldconfig.real: /usr/local/lib/python3.7/dist-
packages/ideep4py/lib/libmkldnn.so.0 is not a symbolic link

Processing triggers for man-db (2.8.3-2ubuntu0.1) …
Processing triggers for fontconfig (2.12.6-0ubuntu2) …
Processing triggers for tex-common (6.09) …
Running updmap-sys. This may take some time… done.
Running mktexlsr /var/lib/texmf … done.
Building format(s) --all.
        This may take some time… done.
Get:1 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Hit:2 http://archive.ubuntu.com/ubuntu bionic InRelease
Get:3 http://archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:4 https://cloud.r-project.org/bin/linux/ubuntu bionic-cran40/ InRelease
[3,626 B]
Hit:5 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64
InRelease
Get:6 http://archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Hit:7 http://ppa.launchpad.net/c2d4u.team/c2d4u4.0+/ubuntu bionic InRelease
Ign:8 https://developer.download.nvidia.com/compute/machine-
learning/repos/ubuntu1804/x86_64  InRelease
Hit:9 https://developer.download.nvidia.com/compute/machine-
learning/repos/ubuntu1804/x86_64  Release
Hit:10 http://ppa.launchpad.net/cran/libgit2/ubuntu bionic InRelease
Get:11 http://security.ubuntu.com/ubuntu bionic-security/multiverse amd64
Packages [22.8 kB]
Get:12 http://security.ubuntu.com/ubuntu bionic-security/universe amd64 Packages
[1,503 kB]
Get:13 http://security.ubuntu.com/ubuntu bionic-security/restricted amd64
Packages [932 kB]
Get:14 http://security.ubuntu.com/ubuntu bionic-security/main amd64 Packages
[2,765 kB]
Hit:15 http://ppa.launchpad.net/deadsnakes/ppa/ubuntu bionic InRelease
Get:16 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu bionic InRelease
[21.3 kB]
```

```
Get:17 http://archive.ubuntu.com/ubuntu bionic-updates/multiverse amd64 Packages
[29.8 kB]
Get:18 http://archive.ubuntu.com/ubuntu bionic-updates/restricted amd64 Packages
[966 kB]
Get:19 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 Packages
[2,277 kB]
Get:20 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 Packages
[3,199 kB]
Get:22 http://ppa.launchpad.net/graphics-drivers/ppa/ubuntu bionic/main amd64
Packages [44.3 kB]
Fetched 12.0 MB in 2s (5,826 kB/s)
Reading package lists… Done
Reading package lists… Done
Building dependency tree
Reading state information… Done
texlive-fonts-recommended is already the newest version (2017.20180305-1).
texlive-fonts-recommended set to manually installed.
texlive-plain-generic is already the newest version (2017.20180305-2).
texlive-plain-generic set to manually installed.
texlive-xetex is already the newest version (2017.20180305-1).
The following package was automatically installed and is no longer required:
  libnvidia-common-460
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 47 not upgraded.
Collecting pypandoc
  Downloading pypandoc-1.8.tar.gz (31 kB)
```