Duke Department of
STATISTICAL SCIENCE

DUKE UNIVERSITY

STA 841 CATEGORICAL DATA FINAL PROJECT

Lin Xiao, Dayi Fang

Contents

1 Introduction **1**

2 Data description **1**

3 Proposed questions and results **2**
3.1 Shot type effect: 2 pointers or 3 pointers? 2
3.1.1 Shot type with shot zone range 2
3.1.2 Shot type with shot distance 2
3.2 Home court effect: How does home advantage exist? 3
3.3 Opponent effect: When facing tough defender, shoot or drive? 3
3.4 Court location effect: Attack from center or both sides? 3
3.5 Years/age effect: The older, the further he shot? 4

4 Prediction and model selection **4**
4.1 Generalized linear models 4
4.2 Model comparisons with Classification models 5

5 Discussions and future work **6**

6 Appendix **6**
6.1 Geographical effect: Is plateau a harder place to play? 6
6.2 Feature selection 7
6.3 Python Code: for sklearn 7
6.4 R Code: for categorical analysis 11

1 Introduction

Kobe Bryant, a legendary basketball player, announced his retirement from NBA by scoring 60 points in his final game as a forever Los Angeles Laker several months ago. During his 20 years NBA career, Kobe has been always questioned for his "bad" shot selection. Recently before his retirement, his coach Bryan Scott saying he was fine with Kobe's shot selection was being controversial. Given this very detailed shooting information with locations and circumstances of every field goal Kobe attempted, we are trying to predict whether the basket went in.

The study purposes of this project consist of the following:

1. Explore the data, propose interesting questions and perform categorical analysis to justify/refute some of the opinions from the audience.

2. Improve categorical prediction model and make comparisons with some popular classification techniques.

In the end we will have a short discussion of the techniques in the report and propose some future work to make further improvements. This report stems from authors' passion for basketball as a sport, we hope the content is enjoyable.

2 Data description

The data used for this research comes from Kaggle online competition "Kobe Bryant Shot Selection" from stats.nba.com. It contains detailed data about every shot attempts Kobe Bryant made from 1996 to 2016. This dataset contains both training set (25697 pieces of information) and testing set (5000 pieces of information) for us to evaluate our model performances based on the predictions for the variable "shot_made_flag" (0 as missed and 1 as in/made). Table 1 includes the specific term names and explanations of all variables.

Name	Data type	Description
action_type	String	detailed shooting type; Jump Shot, Driving Dunk, etc.
combined_shot_type	String	general shooting type
minutes_remaining	Integer	minutes remaining in a quarter
period	Integer	categorical, game quarter, 1-4, and 5-7 represents over time
playoffs	Boolean	playoff(1) or regular season(0)
season	String	categorical, specific season
seconds_remaining	Integer	remaining seconds in that minute
shot_distance	Integer	**continuous**, shooting distance, in ft
shot_type	String	binary, 2 or 3 pointer
shot_zone_area	String	area location, left side, center, etc.
shot_zone_basic	String	basic zone, mid-range, restricted area, etc.
shot_zone_range	String	categorical, distance from basket in ft same as shot_distance
game_date	Datetype	date of the game
matchup	String	binary, home or away
shot_made_flag	**Boolean**	**target, in(1) or out(0)**

Table 1: Data description

shot type	miss	made
2PT Field Goal	0.5227	0.4773
3PT Field Goal	0.6707	0.3293

(a) shot type versus shot percentage

shot zone range	miss	made
16-24 ft.	0.5982	0.4018
24+ ft.	0.6675	0.3325
8-16 ft.	0.5645	0.4355
Back Court Shot	0.9861	0.0139
Less Than 8 ft.	0.4269	0.5731

(b) shot zone range versus shot percentage

Table 2: shot type, shot zone range and shot made

3 Proposed questions and results

Before jumping into the questions and results next, let's first explore the relations between predictors and responses, by simply running logistic regressions

$$\text{logit}\pi_i = \alpha + \beta_i^X, \quad \text{where X is each predictor}$$

the summaries of models indicates that, except for variable playoffs, all other predictors are significant related to responses. This gives us an initial idea that Kobe was a stable player across a single season.

Subsection 3.1.1 will be exploring conditional independence and marginal independence, 3.1.2 and 3.2 carry out ANOVA test, 3.3 and 3.4 use exact test, and 3.5 talks about the usage of proportional odds cumulative logit model[1].

3.1 Shot type effect: 2 pointers or 3 pointers?

3.1.1 Shot type with shot zone range

From above we have the conclusion that shot type is dependent of shot made. Table 2(a) brings us a puzzle: since $0.3293 \times 3/2 > 0.4773$, should Kobe attempt more 3pt? This conclusion seems reasonable, but let's take a step further by introducing shot zone range, we can see from table 2(b) that the area having the highest probability 0.5731 is the Less Than 8 ft area, which might be the first choice for any player. The model becomes

$$\text{logit}\pi_{ik} = \alpha + \beta_{ik}^X + \beta_k^Z$$

where i = 2pt or 3pt, k = 0-8 ft, 8-16 ft,..., back court shot, and $\beta^Z_{\text{(any k)}} = 0$ for identifiability. We can either perform a $2 \times 2 \times K$ logistic regression, or use Cochran-Mantel-Haenszel (CMH) test and obtain p-value = 0.9006 indicating conditional independence between Shot type and shot made conditioning on shot zone range.

3.1.2 Shot type with shot distance

Shot distance is the only continuous variable in the dataset(excluding remaining time in my analysis, could be further work), we can carry out ANOVA to compare different models so as to explore shot type and shot distance.

1. Common slope and common intercept: $\text{logit}\pi_i = \alpha + \beta_1 X_i$
2. Common slope and different intercept: $\text{logit}\pi_{ik} = \alpha + \beta_1 X_i + \beta_2 \mathbb{I}_{k=\text{2pt}}$
3. Different slope and different intercept: $\text{logit}\pi_{ik} = \alpha + \beta_1 X_i + \beta_2 \mathbb{I}_{k=\text{2pt}} + \beta_3 X_i \mathbb{I}_{k=\text{2pt}}$

ANOVA	p value
1 & 3	1.186e-07
2 & 3	1.909e-07

(a) ANOVA for shot type and distance

Coefs	estimates
α	0.36856932
β_1	-0.04516641
β_2	1.84624439
β_3	-0.07059524

(b) estimates for 3rd model with different slope and different intercept

ANOVA	p value
1 & 3	0.0491
2 & 3	0.1098

(c) ANOVA for shot distance and home

Table 3: ANOVA tests

Table 3(a) and 3(b) contains model comparisons results. The 3rd model improves the 1st and the 2nd model. The over-dispersion value is 0.9979, showing no over dispersion effect in this model.

3.2 Home court effect: How does home advantage exist?

By changing variable "shot type" to variable "home", we can carry out the same analysis as section 3.1.2(since both variables are binary). ANOVA is again being used here as table 3(c), but with a insignificant p value 3rd model is not improving the 2nd, altogether showing some evidence that the 2nd model(Common slope and different intercept) is the best among these three.

3.3 Opponent effect: When facing tough defender, shoot or drive?

Opponent/defender is a key factor to a player's performance. Unfortunately we don't have that type of data. So we change a direction and explore on the shot selection when facing San Antonio Spurs. The reason why choosing this team is because it's the only tough and stable team throughout Kobe's career. While shot type(2 or 3 pt) is independent of whether against SAS, this opponent is affecting Kobe's selection inside 3 pt line(24 ft) indicated by table 3(a) Fisher's exact test with p value 0.00754. As the best jump shooter in history, he chose to make more shots in the mid range(16-24 ft) instead of attacking the rim.

3.4 Court location effect: Attack from center or both sides?

Finally we want to know more about the routes Kobe is taking towards making shots. The initial goal was exploring if any preference from left(& left center) or right(& right center), the results are not shown here but with Fisher's exact test it ended up with insignificant p values 0.0964. This shows his ability of using both hands skillfully. Let's now focus on center effect.

Table 4 is exploring his tendency of making offense moves from center area. Table 4(b) shows that the shooting percentages of center and not center are 0.5256 and 0.3858, clearly showing a gap. The results could be justified again by Fisher's exact test with significant p value $<$ 2e-16.

Take a step further, we raise and try to answer a question: since he is a better scorer in the center area, would he attack more from center during playoff season? Table 4(c) is showing his attempts distribution, and a significant p value 0.0235 demonstrates his efforts of doing so, in order to make safer plays during tougher games.

Oppo.	0-16 ft	16-24 ft
SAS	12497	6492
! SAS	940	415

(a) Opponents' effect

Side	attempt	made
Center	11289	5933
! center	14336	5531

(b) Center versus shot percentage

Location	playoff	regular
Center	2031	12305
! center	1713	9576

(c) Attempts

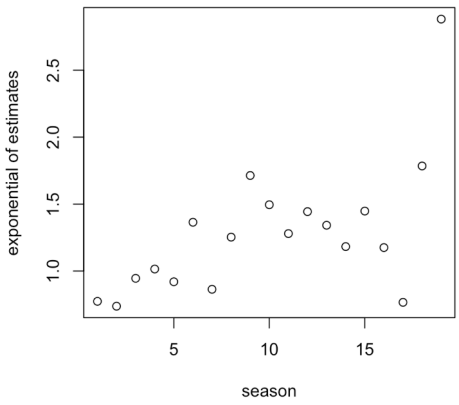
Table 4: Fisher's exact test

3.5 Years/age effect: The older, the further he shot?

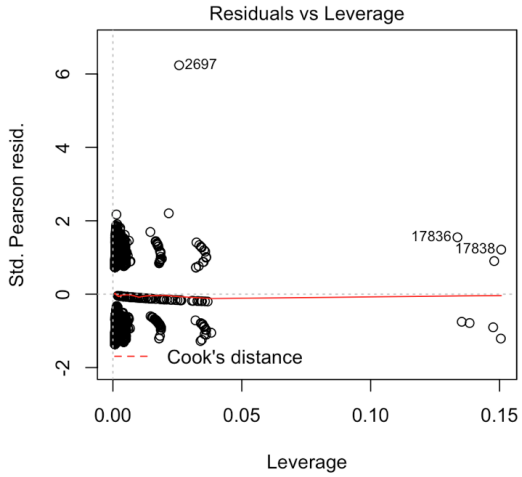
Players tend to attack the rim(or another way to say it, drive to the basket) less with their age increasing. We want to see from the data if this also happens to Kobe. Taking shot zone range as ordered response and season as categorical data, we can perform proportional odds cumulative logit model:

$$\text{logit P}(Y \leq j|x) = \alpha_j + x^{'}\beta, \quad \text{with x being categorical in this case}$$

Take exponential of estimates, we get the graph in figure 1(a). The interpretation of these exponential is that, or a one unit increase in season, i.e move forward a year/season, the odds of shooting anywhere more than 8 ft is exponential of coefficient time of less than 8 ft, same thing for other categories[2] . So the graph shows the trend as we thought. Note: he was injured most of time during 17th season. And for last season, he announced his retirement early, so he was somehow allowed to shoot more, and he was slow in motion at that time, leading to more outside shots.



(a) Proportional odds model estimates (exponential)



(b) Outlier identification

Figure 1

4 Prediction and model selection

4.1 Generalized linear models

Before we improve our GLM model, we run a stepwise function finding out that "playoffs" is the only variable got eliminated from the model, which accords with our finding earlier. And there is no clear difference between using logit and Probit link, so we will start with only

excluding "playoffs" and using logit link. Besides these variables, since some variables are nested in others, for example, action type nested in combined shot type, it would be more accurate the build a multilevel logistic regression such as

$$(1 + \text{shot\_distance}|\text{shot\_type/combined\_shot\_type/action\_type})$$

where shot distance has varying slope, action type nested in combined shot type and combined shot type nested in shot type. But due to unconvergence of this model, we simplify and use the model: $(1|\text{shot\_type/action\_type})$.

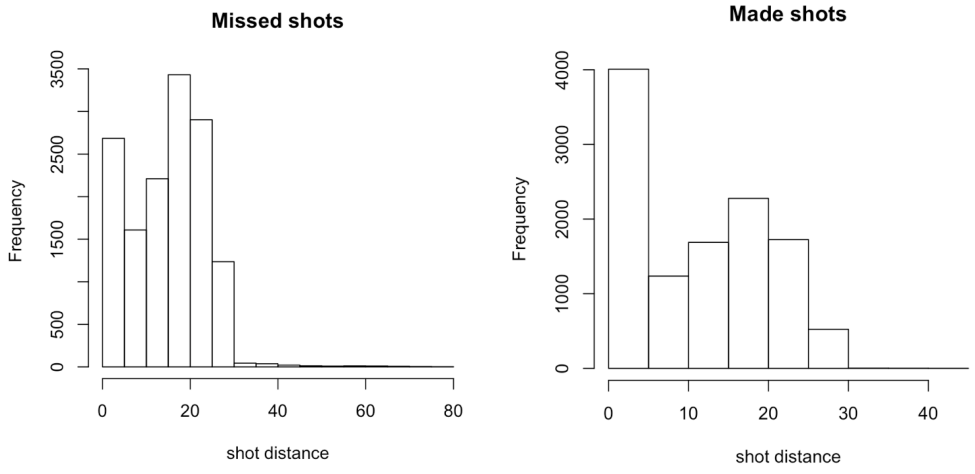
Figure 1(b) shows the outlier of the logistic model. We can see that the shot with id 2697 is clearly an outlier. Looking into the data we will find out that it was the only backward shot Kobe has made in his entire career, which means this is definitely an prediction very hard to get it right.

4.2 Model comparisons with Classification models

The problem is a binary classification problem, so we can introduce some basic classification algorithms and use misclassification rate to compare between models. We already have logistic model from above $(1|\text{shot\_type/action\_type})$, by calculating the misclassification rate over a grid of thresholds we obtain the highest rate. After creating dummy variables and performing feature selections, linear discriminant analysis, k nearest neighbors, CART and Naive Bayes together give us the following cross validation prediction power:

Algorithms	Misclassification rate
LDA	0.681
K-NN	0.602
CART	0.606
NB	0.647
LOGIT	0.682

Table 5: Model comparison



(a) distance distribution for missed shots (b) distance distribution for made shots

Figure 2: Non Gaussianity for shot distance

A comment here about LDA and logistic regression: LDA has approximately the same prediction power as logistic regression in this case, since the data has non-Gaussianity. Logistic regression is more robust to LDA under this situation. This also justify the opinion that Kobe is a rim attacker and mid range jump shooter with the spike between 0-5 ft and between 15-20 ft.

5 Discussions and future work

The project is a very good opportunity for us to explore utilizing models from the class and try to identify and solve some questions statistically. Kobe Bryant has been a controversial but excellent player throughout the history. Experts and fans have been trying to explore more on his behaviors on the court. There are definitely more to explore with more available data coming in.

Due to time limitation, we could only identify these aspects of the datasets and create some models for model selection. In the future to further explore based on the same data, we could probably carry out following models:

- Find a solution to the unconvergence of multilevel logistic regression models.
- Run Bayesian logit model on the data and do model comparisons.
- Carry out more machine learning classification algorithms such as adaboost and svm.
- Use unsupervised learning to identify more patterns in the data.

The project helps us develop a general understanding about the influential factors of making shots and guides us a potential future task about applying similar algorithms in basketball related fields such as player evaluation, draft, and video games.

6 Appendix

6.1 Geographical effect: Is plateau a harder place to play?

Let's further explore some geographical effect: is altitude affecting Kobe's performance? For example, Denver is a city with high altitude and thinner air, therefore it's harder for newcomers to breathe, even harder to play sports. With a 2×2 table like table 6 we can carry out Fisher's exact test and obtain a p value of 0.9455, indicating Kobe's performance is independent of this particular geographical effect. Good player.

location	attempt	made
Den	712	319
Not Den	24985	11146

Table 6: Altitude effect

6.2 Feature selection

- Variance Threshold[3]: Find all features with more than 90% variance in values.
- Variance importance: Use random forest to find out the 20 most important features
- Univariate feature selection: Chi-square test for feature selection
- Select 20 features from using recursive feature elimination (RFE) with logistic regression model.

And we pick the dummy features that are remaining for all these selection techniques.

6.3 Python Code: for sklearn

```
import pandas as pd
import warnings
warnings.filterwarnings('ignore')

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.decomposition import PCA, KernelPCA
from sklearn.cross_validation import KFold, cross_val_score
from sklearn.metrics import make_scorer
from sklearn.grid_search import GridSearchCV
from sklearn.feature_selection import VarianceThreshold, RFE, SelectKBest, chi2
from sklearn.preprocessing import MinMaxScaler
from sklearn.pipeline import Pipeline, FeatureUnion
from sklearn.linear_model import LogisticRegression
```

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.ensemble import BaggingClassifier, ExtraTreesClassifier,
    GradientBoostingClassifier, \
VotingClassifier, RandomForestClassifier, AdaBoostClassifier

data = pd.read_csv("/Users/LinXIAO/Desktop/841/final project/data_2.csv")
data.set_index('shot_id', inplace=True)
data["action_type"] = data["action_type"].astype('object')
data["combined_shot_type"] = data["combined_shot_type"].astype('category')
data["game_event_id"] = data["game_event_id"].astype('category')
data["game_id"] = data["game_id"].astype('category')
data["period"] = data["period"].astype('object')
data["playoffs"] = data["playoffs"].astype('category')
data["season"] = data["season"].astype('category')
data["shot_made_flag"] = data["shot_made_flag"].astype('category')
data["shot_type"] = data["shot_type"].astype('category')
data["team_id"] = data["team_id"].astype('category')

data_cl = data.copy() # create a copy of data frame
target = data_cl['shot_made_flag'].copy()

# Remove some columns
data_cl.drop('team_id', axis=1, inplace=True) # Always one number
data_cl.drop('lat', axis=1, inplace=True) # Correlated with loc_x
data_cl.drop('lon', axis=1, inplace=True) # Correlated with loc_y
data_cl.drop('game_id', axis=1, inplace=True) # Independent
data_cl.drop('game_event_id', axis=1, inplace=True) # Independent
data_cl.drop('team_name', axis=1, inplace=True) # Always LA Lakers
data_cl.drop('shot_made_flag', axis=1, inplace=True)

# Remaining time
data_cl['seconds_from_period_end'] = 60 * data_cl['minutes_remaining'] +
    data_cl['seconds_remaining']
data_cl['last_5_sec_in_period'] = data_cl['seconds_from_period_end'] < 5

data_cl.drop('minutes_remaining', axis=1, inplace=True)
data_cl.drop('seconds_remaining', axis=1, inplace=True)
data_cl.drop('seconds_from_period_end', axis=1, inplace=True)

## Matchup - (away/home)
data_cl['home_play'] = data_cl['matchup'].str.contains('vs').astype('int')
data_cl.drop('matchup', axis=1, inplace=True)

# Game date
```

```
data_cl['game_date'] = pd.to_datetime(data_cl['game_date'])
data_cl['game_year'] = data_cl['game_date'].dt.year
data_cl['game_month'] = data_cl['game_date'].dt.month
data_cl.drop('game_date', axis=1, inplace=True)

# Loc_x, and loc_y binning
data_cl['loc_x'] = pd.cut(data_cl['loc_x'], 25)
data_cl['loc_y'] = pd.cut(data_cl['loc_y'], 25)

# Replace 20 least common action types with value 'Other'
rare_action_types =
    data_cl['action_type'].value_counts().sort_values().index.values[:20]
data_cl.loc[data_cl['action_type'].isin(rare_action_types), 'action_type'] =
    'Other'

data_cl.drop('loc_x', axis=1, inplace=True)
data_cl.drop('loc_y', axis=1, inplace=True)
data_cll = data_cl.copy()

categorial_cols = [
    'action_type', 'combined_shot_type', 'period', 'season', 'shot_type',
    'shot_zone_area', 'shot_zone_basic', 'shot_zone_range', 'game_year',
    'game_month', 'opponent']

for cc in categorial_cols:
    dummies = pd.get_dummies(data_cl[cc])
    dummies = dummies.add_prefix("{}#".format(cc))
    data_cl.drop(cc, axis=1, inplace=True)
    data_cl = data_cl.join(dummies)

unknown_mask = data['shot_made_flag'].isnull()
data_submit = data_cl[unknown_mask]

# Separate dataset for training
X = data_cl[~unknown_mask]
Y = target[~unknown_mask]


threshold = 0.90
vt = VarianceThreshold().fit(X)

# Find feature names
feat_var_threshold = data_cl.columns[vt.variances_ > threshold * (1-threshold)]


model = RandomForestClassifier()
model.fit(X, Y)
```

```
feature_imp = pd.DataFrame(model.feature_importances_, index=X.columns,
    columns=["importance"])
feat_imp_20 = feature_imp.sort_values("importance", ascending=False).head(20).index


X_minmax = MinMaxScaler(feature_range=(0,1)).fit_transform(X)
X_scored = SelectKBest(score_func=chi2, k='all').fit(X_minmax, Y)
feature_scoring = pd.DataFrame({
        'feature': X.columns,
        'score': X_scored.scores_
    })

feat_scored_20 = feature_scoring.sort_values('score',
    ascending=False).head(20)['feature'].values


rfe = RFE(LogisticRegression(), 20)
rfe.fit(X, Y)

feature_rfe_scoring = pd.DataFrame({
        'feature': X.columns,
        'score': rfe.ranking_
    })

feat_rfe_20 = feature_rfe_scoring[feature_rfe_scoring['score'] ==
    1]['feature'].values


features = np.hstack([
        feat_var_threshold,
        feat_imp_20,
        feat_scored_20,
        feat_rfe_20
    ])

features = np.unique(features)
print('Final features set:\n')
for f in features:
    print("\t-{}".format(f))

data_cl = data_cl.ix[:, features]
data_submit = data_submit.ix[:, features]
X = X.ix[:, features]

seed = 123
processors=1
num_folds=10
num_instances=len(X)
```

```
kfold = KFold(n=num_instances, n_folds=num_folds, random_state=seed)

models = []
models.append(('LR', LogisticRegression()))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('K-NN', KNeighborsClassifier(n_neighbors=5)))
models.append(('CART', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
#models.append(('SVC', SVC(probability=True)))

# Evaluate each model in turn
results = []
names = []

for name, model in models:
    cv_results = cross_val_score(model, X, Y, cv=kfold, n_jobs=processors)
    results.append(cv_results)
    names.append(name)
    print("{0}: ({1:.3f}) +/- ({2:.3f})".format(name, cv_results.mean(),
        cv_results.std()))


clf = LinearDiscriminantAnalysis()
clf.fit(X,Y)
clf.score(X,Y)
```

6.4 R Code: for categorical analysis

```
library(MASS)
library(dplyr)
library(lme4)
library(arm)
library(DAAG)
data = read.csv("/Users/LinXIAO/Desktop/841/final project/data.csv", header=T)
test = data[is.na(data$shot_made_flag),]
train = data[!is.na(data$shot_made_flag),]

# add home or guest
train[grepl("@", train$matchup), "home"] = 0
train[!grepl("@", train$matchup), "home"] = 1
train[,"minutes_left"] <- train$minutes_remaining+12*(train$period-1)
train = train[,!(names(train) %in% c("game_date", "matchup","minutes_remaining"))]
train$home = as.factor(train$home)
train$playoffs = as.factor(train$playoffs)
train$period = as.factor(train$period)
```

```
trainmade <- train[ train $shot_made_flag==1,]
hist (trainmade$shot_distance , main="Made shots", xlab = "shot distance")
trainmiss <- train[ train $shot_made_flag==0,]
hist ( trainmiss $shot_distance , main="Missed shots", xlab = "shot distance")

attach ( train )
##############################################################################
# Q1: What is correlated to shot_made_flag?

for(i in 1:ncol( train )){
  summary(glm(train$shot_made_flag~ train [, i ], family = binomial("logit")))
}
 ttttt = summarise(group_by( train , home), count = n() , made = sum(shot_made_flag))
summary(glm(cbind( ttttt $made, ttttt $count- ttttt $made)~c(0,1) , family =
    binomial("logit")))

# playoff is independent, saying sth like this player is super professional ; make
    xtable here


##############################################################################
# shot_type and shot_zone range
# We still want to attack the rim since the shot_percentage is much higher for
    less than 8 feet
trainnew = train [,c("shot_zone_range", "shot_made_flag", "shot_type")]
table = summarise(group_by(trainnew, shot_zone_range, shot_type, shot_made_flag),
    count = n())
prop. table (xtabs(count~shot_type+shot_made_flag, table ),1)

prop. table (xtabs(count~shot_zone_range+shot_made_flag, table ),1)

table $shot_made_flag <- as. factor ( table $shot_made_flag)
table $count <- as.double(table $count)
table . partial = xtabs(count~shot_type+shot_made_flag+shot_zone_range, table )

# conditional independent
mantelhaen. test ( table . partial )

##############################################################################

# Ordered logistic regression , refer to Kobe's performance
# Proportional -odds cumulative logit model
by_tailnum <- group_by(train, shot_zone_range, season)
delay <- print(summarise(by_tailnum, count = sum(shot_made_flag)),n=100)
delay$shot_zone_range <- factor(delay$shot_zone_range, levels = c("Less Than 8
    ft .", "8-16 ft.", "16-24 ft.", "24+ ft .", "Back Court Shot"))
m.po = polr(shot_zone_range~season, weights = count, data = delay)
plot(exp(coef(m.po)), ylab = "exponential of estimates", xlab = "season")
```

```
#############################################################################################

#############################################################################################
# shot_distance and shot_type
## different intercept different slope is better model
m1 = glm(shot_made_flag~shot_distance,data = train, family = binomial("logit"))
m2 = glm(shot_made_flag~shot_distance+shot_type,data = train, family =
    binomial("logit"))
# indicates model does not fit well with lower than 0.05 pvalue
# hosmerlem(train$shot_made_flag, fitted(m2))
m3 = glm(shot_made_flag~shot_distance*shot_type, data = train,family =
    binomial("logit"))
anova(m2, m3, test="Chisq")
# no overdispersion effect
sum((residuals(m3, "pearson"))^2)/m3$df.residual
#############################################################################################


# what about home?

# shot distance and home
## different intercept different slope is better model
m1 = glm(shot_made_flag~shot_distance, data = train, family = binomial("logit"))
# common slope different intercept
m2 = glm(shot_made_flag~shot_distance+home, data = train, family =
    binomial("logit"))
# different intercept different slope
m3 = glm(shot_made_flag~shot_distance*home, data = train, family =
    binomial("logit"))
anova(m1, m3, test="Chisq")
anova(m2, m3, test="Chisq")
# common slope different intercept rules
sum((residuals(m3, "pearson"))^2)/m3$df.residual

#############################################################################################
# Generalized linear models
# dont see overdispersion : could be due to some reasons including underdispersion
    and overdispersion cancelling effect
m1 = glm(shot_made_flag~shot_distance, data = train, family = binomial("logit"))

step <- stepAIC(logit, direction="both")
```

```
logit =
    glm(shot_made_flag~shot_distance+factor(playoffs)+factor(period)+shot_zone_area+factor(shot_zo
        data = train, family = binomial("logit"))
probit =
    glm(shot_made_flag~shot_distance+factor(playoffs)+factor(period)+shot_zone_area+factor(shot_zo
        data = train, family = binomial("probit"))
cauchit =
    glm(shot_made_flag~shot_distance+factor(playoffs)+factor(period)+shot_zone_area+factor(shot_zo
        data = train, family = binomial("cauchit"))



# action type can replace combined shot type
p1 = glmer(shot_made_flag~shot_distance+(1|action_type), data = train, family =
    binomial("logit"))
p2 = glmer(shot_made_flag~shot_distance+(1|combined_shot_type), data = train,
    family = binomial("logit"))
p3 = glmer(shot_made_flag~shot_distance+(1|combined_shot_type)+(1|action_type),
    data = train, family = binomial("logit"))
p5 = glmer(shot_made_flag~shot_distance+(1|combined_shot_type/action_type), data =
     train, family = binomial("logit"))
p4 =
    glmer(shot_made_flag~shot_distance+(1+shot_distance|combined_shot_type/action_type),
    data = train, family = binomial("logit"))
p6 = glmer(shot_made_flag~shot_distance+(1|shot_type/action_type), data = train,
    family = binomial("logit"))
p9 =
    glmer(shot_made_flag~shot_distance+(1+shot_distance|combined_shot_type/shot_type),
    data = train, family = binomial("logit"))
p10 = glmer(shot_made_flag~shot_distance+(1+shot_distance|shot_type/action_type),
    data = train, family = binomial("logit"))

list = predict(p16, type = "response")

for(i in seq(0.35,0.6, length.out = 20)){
  pred = list>i
  print((dim(train)[1] - sum(abs(pred-train$shot_made_flag)))/dim(train)[1])
}

# Run on all
#train$playoffs = as.factor(train$playoffs)
#train$shot_distance = scale(train$shot_distance)
#train["minutes_left"] <- train$minutes_remaining+12*(train$period-1)
#train$minutes_left = scale(train$minutes_left)

sum((residuals(m3,"pearson"))^2)/m3$df.residual
```

```
p15 =
    glmer(shot_made_flag~shot_distance+(1|playoffs)+(1|period)+(1|shot_zone_area)+(1|shot_zone_ran
                (1|home)+(1|season)+(1|combined_shot_type/action_type), data =
                    train, family = binomial("logit")))
p16 =
    glmer(shot_made_flag~shot_distance+(1|playoffs)+(1|period)+(1|shot_zone_area)+(1|shot_zone_ran
                (1|home)+(1|season)+(1|shot_type/action_type), data = train, family
                    = binomial("logit")))

# Didnt converge for the following nested model
p17 =
    glmer(shot_made_flag~shot_distance+(1|period)+(1|shot_zone_area)+(1|shot_zone_range)+
                (1|home)+(1|season)+(1+shot_distance|shot_type/action_type), data =
                    train, family = binomial("logit")))
p18 =
    glmer(shot_made_flag~shot_distance+(1|period)+(1|shot_zone_area)+(1|shot_zone_range)+
                (1|home)+(1|season)+(1+shot_distance|combined_shot_type/action_type),
                    data = train, family = binomial("logit"))
# add another layer works better
p19 =
    glmer(shot_made_flag~shot_distance+(1|playoffs)+(1|period)+(1|shot_zone_area)+(1|shot_zone_ran
                (1|home)+(1|season)+(1+shot_distance|shot_type/combined_shot_type/action_type),
                    data = train, family = binomial("logit"))


########## consider add layer about shot zone
# run time too long

anova(p20, p21, test="Chisq")
# run anova
display(p41)

################################################################################

# bidirectional stepwise

################################################################################
# other topics:
# log odds ratio

# Professional player proven
# area effect? Denver
train1 = train
train1$opponent = as.character(train1$opponent)
train1[train1$opponent != "DEN","opponent"] <- "NOTDEN"
summarise(group_by(train1, opponent, home), count=n(), made = sum(shot_made_flag))

mat1 = matrix(c(712,319, 640+12500+11845, 300+5447+5399), byrow = T, ncol = 2)
```

```
summary(glm(cbind(mat1[,2], mat1[,1] - mat1[,2])~c(0,1), family =
    binomial("logit")))
fisher.test(mat1)

# Spurs?
# tend to shoot more middle range(16-24) compared to 0-8 and 8-16 when facing Spurs
train2 = train
train2$opponent = as.character(train2$opponent)
train2[train2$opponent != "SAS","opponent"] <- "NOTSAS"
ttt2 = summarise(group_by(train2, opponent, shot_zone_range), count=n(), made =
    sum(shot_made_flag))
mat2 = matrix(c(7315+5182, 6492, 542+398, 415), byrow = T, ncol = 2)
summary(glm(cbind(mat2[,2], mat2[,1])~c(0,1), family = binomial("logit")))



# right or left? Probably no different, indicating this guy is good at both side
summarise(group_by(train, shot_zone_area), count=n(), made = sum(shot_made_flag))
mat2 = matrix(c(3364+3132, 1215+1243, 3981+3859, 1523+1550), byrow = T, ncol = 2)
summary(glm(cbind(mat2[,2], mat2[,1] - mat2[,2])~c(0,1), family =
    binomial("logit")))

mat2 = matrix(c(3364, 1215, 3981, 1523), byrow = T, ncol = 2)
summary(glm(cbind(mat2[,2], mat2[,1] - mat2[,2])~c(0,1), family =
    binomial("logit")))


# center and non center are different
# He is shoting better in center
mat2 = matrix(c(11289, 5933, 3981+3859+3364+3132, 1215+1243+1523+1550), byrow =
    T, ncol = 2)
summary(glm(cbind(mat2[,2], mat2[,1] - mat2[,2])~c(0,1), family =
    binomial("logit")))

# So he takes more shots in center, especailly during playoffs, save play.
tmat2 = summarise(group_by(train, shot_zone_area, playoffs), count=n(), made =
    sum(shot_made_flag))
mat2 = matrix(c(sum(tmat2[c(6,8,10,12),"count"]), sum(tmat2[c(5,7,9,11),"count"]),
                as.numeric(tmat2[4,"count"]), as.numeric(tmat2[3,"count"])), byrow
                    = T, ncol = 2)
summary(glm(cbind(mat2[,2], mat2[,1])~c(0,1), family = binomial("logit")))

################################################################################

###  Bayesian??

################################################################################
```

```
### diagnotics

#############################################################################################

### predictions , compared to machine learning , on the test set (include LDA)

print ( sessionInfo () , locale = FALSE)
```

References

[1] Alan Agresti and Maria Kateri. *Categorical data analysis*. Springer, 2011.

[2] Tim Futing Liao. *Interpreting probability models: Logit, probit, and other generalized linear models*. Number 101. Sage, 1994.

[3] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.