

100% MATCH IN DATING APPS

STAT 6309



Yichien chou

zherui lin

JINGCHEN LIANG

# Introduction

Does your heart beat faster when you are dating someone? In this paper, we would like to investigate the types of factors that might affect personal performance in dating. Our major task of this project is to decide whether a person is willing to accept second date information based on the first date; thus, "match" is the response variable. In data preprocessing, we impute categorical missing values in the original dataset. Then, several machine techniques such as model selection, resampling method, and lasso regression analysis are applied to catch all predictors for further predictive modeling. Predictively, logistic regression and random forest models perform well that accurately predict the date decision, each with an accuracy of approximately 88%.

# Preliminary Investigation

This dataset contains 552 total people (277 males, 274 females) who were participating in the speed dating experiment from 2002 to 2004. Data can be found on the website “<https://www.kaggle.com/annavictoria/speed-dating-experiment>”. Each participant had a four-minute “first date” with other participants of the opposite sex. Finally, participants were asked whether they want a second date based on six attributes: attractiveness, ambition, intelligence, fun, sincerity, and shared interests. They were also required to complete questionnaires including 5 fields: dating habits, demographic information, belief, lifestyle, and self-perception.

* There are 195 variables in total. The following Table 1 shows a basic review of some important variable names:



*Table 1. Variables for Consideration*

More variables will be clearly explained in the following analysis.

# Data Cleaning

* Firstly, we checked to see if there were any null values in the dataset.
* Secondly, we removed all the variables which had more than 1000 missing values and we left with 101 variables. Then, we imputed all the rest of the variables by a method contained in the scikit-learn library.
* We also scaled the relevant variables for the further linear regression model.

Moreover, we find there were three variable which can’t be quantified and impossible to transfer to dummy variables. (However, we discuss some features later in exploratory data analysis part)

* field= field of study
* from = Where are you from originally?
* career= What is your intended career?

Lastly, we removed the ‘dec\_o’ variable which represents the ‘decision of partner the night of event’. Since the participants cannot acquire the partners' real thoughts, we decided to remove it as well.

Exploratory Data Analysis

To summarize the main characteristics of this dating experiment, the most important issue of this dataset is repetition. Since each participant was asked to date with others multiple times, information was recorded once for each math. For example, an Asian male got a date with 10 female, he/she may be counted as 10 Asian people. Thus, this dataset has 8378 total observations, with lots of repetitions. Since repetitions can cause bias, we would like to change to unique entries, i.e., 551 unique participants. Then, we investigate several essential features such as age, race, field of study, intentions, hobbies, and assessment to understand what features are most essential in this experiment.

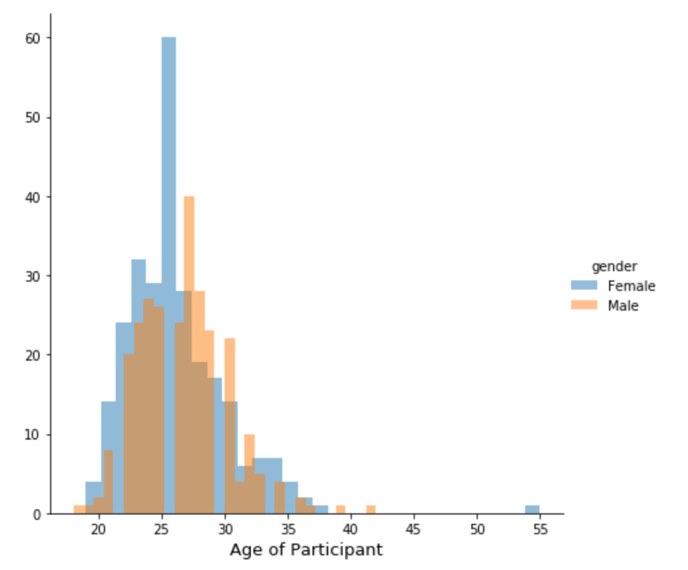
A picture containing tree

Description automatically generated

*Figure 1. Heatmap*

In the beginning, we construct a heatmap of the correlation between all the variables. we found that there has the collinearity of the multiple variables in the dataset. Then we analyze five specific features (variables) respectively, with important summaries as follows:

* **Age and Race**

A close up of text on a white background

Description automatically generated

*Figure 2. Age and Race*

A screenshot of a cell phone

Description automatically generated

*Figure 3. Importance of race*

1. There doesn’t exist a huge difference in age, but we could find that the males are slightly older than the female participants (Figure 2).

2. Most participants are Caucasian and Asian participants. Thus, data were probably collected from a specific area or a cultural event (Figure 2).

3. For “other” and “White” groups, more males than females (Figure 2).

4. Males are older than females except Asian group (Figure 2).

5. Given the definition of “imprace” from the data that “how important is it to you (on a scale of 1-10) that a person you date be of the same racial/ethnic background”, from Figure 3, we find that white people seem to consider more important to race than other races.

* **Field of Study**

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

*Figure 4. Field of Study*

1. Most of them work in the business field. More males work in the business field than females.
2. More males work in engineering, law fields, while more females work in education, medical science, and social work.
3. For biological science political science, the number of men and women is roughly equal.

* **Intention**

A screenshot of a cell phone

Description automatically generated

*Figure 5. Intentions*

1. Most of the people come here to have fun and want to meet new people.
2. Females are approximately twice as many as males who want to find a date in this event. Women are probably more serious about dating.

**Hobbies**

**A picture containing fence

Description automatically generated**

**A pencil and paper

Description automatically generated**

*Figure 6. Hobbies*

Females seem to have a higher average interest than males in every kind of activity except sports, tvsports, and gaming.

* Asians prefer tv
* Hispanic reading or watching movies
* White prefer doing exercise
* Black like clubbing and music
* **Assessment**

A screenshot of a cell phone

Description automatically generatedA close up of a map

Description automatically generated

*Figure 7. Assessment*

1. Match rate is not high in each wave, especially low for wave 2, 12, 18
2. Fun has the largest difference in the decision while being Sincere or Ambitious look not influential.

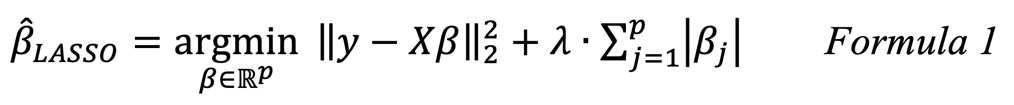
# Model Selection

After the data cleaning process, the above heatmap provides the correlations between matches and other numeric predictors. Due to the potential collinearity issue among the predictor variables with many predictor variables, we planned to use three penalized regression methods: Ridge, Lasso, and Elastic Net, to select the most appropriate regression model to predict if subjects had successfully matched. Table 2 summarizes RMSE results for these three methods.

|  |  |
| --- | --- |
| **Competing Techniques** | **RMSE** |
| Ridge | 0.2878 |
| Lasso | 0.2909 |
| ElasticNEt | 0.3709 |

*Table 2. Comparing three methods*

We choose Lasso method. From the definition formula of Lasso (Formula 1), the regularization term in lasso method is in absolute value, which will influence the trade-off between underfitting and overfitting a model. Thus, Lasso not only punished high coefficients β values but also setting to 0 if they are not relevant, thereby maintaining more valuable features in our model.



After the data cleaning process, there were 84 predictors left. From Figure 8 lasso selection plot, we can see the top 20 significant predictors. Furthermore, the top three positive predictors are “dec”, “attr\_o”, and “like\_o”. On the other hand, the top three negative predictors are “met\_o”, “gender”, and “samerace”.

一張含有 螢幕擷取畫面 的圖片

自動產生的描述

*Figure 8. Lasso Selection Plot*

# Resampling Method

To test the accuracy rate of our model, we split data into training and testing set using cross-validation. This method involved randomly dividing the available set of observations into two parts, a training set, and a testing set. Our regression model is fit on the training set, and the fitted model predicted the response for the predictors in the testing set. The resulting testing set accuracy rate, which was typically assessed using MSE in the case of a quantitative response. We assigned 70% of the observations to the training set and assigned the remaining 30% to the testing set.

# Logistic Regression

Our goal was to predict if the two partners were successfully matched. Our response variable, “match”, was a categorical variable. Therefore, it was appropriate to build a logistic regression model.

We put the top four predictors selected from lasso selection into a logistic regression model, then test the accuracy rate using training set and testing set. The model accuracy of the logistic regression is approximately 89.9% (Figure 9).



*Figure 9. Logistic Regression Result*

# Random Forest

一張含有 地圖 的圖片

自動產生的描述

*Figure 10. Random Forest*

For each subset, it will build a decision tree and generate the characteristic (not all the characteristic would benefit the final decision). Then, the algorithm will combine all the characteristics of the subset and distribute the final characteristic.

We set an accuracy rate applying random forest. The model accuracy of the random forest is approximately 87.7% (Figure 11). Compared to logistic regression, the accuracy rate of random forest is lower than that of logistic regression.



*Figure 11. Random Forest Result*

# Further Improvement

Although we can conclude some important information or interesting findings from our predictive modeling (See Conclusion), it is unwise to rely solely on statistical models to judge or select dating partners. These data were probably collected from a specific event, so conclusions cannot be generalized to the population. Also, data were recorded from 2002 to 2004, which are not appropriate to be an indicator of finding love among current young people. Furthermore, given abundant repetitions and missing values, more data are necessary to build a more accurate model; however, it is difficult to collect demographic data because a lot of people don't want to disclose their true information. Finally, love is a mysterious thing, and statistical and mathematical theories certainly cannot explain it all. Psychologists and sociologists may give us novel thoughts by looking at love from different angles.

# Conclusion

* We present a relationship between gender and race, and how important are racial background, with gender has a much higher weight.
* Race is not much important. However, people tend to give positive feedback about their date if the partner was of their same racial backgrounds.
* Many participants were fairly confident before the date. However, in rating their expectations after every date, they looked not so confident if they liked the partner, especially if they were attractive.
* How much the participant found their partners attractive (or other key attributes) is way more important of shared interest, race, and field of study.
* Several features may increase/decrease the chance of match (next slide).
* Our final models had 89.9% and 88.4% accuracy in predicting outcomes, respectively. That is, the logistic regression and random forest showed a similar accuracy rate.

Besides, several features may increase/decrease the chance of match:

***Positive***

* Participants were willing to date with their partners for the next time.
* Partners thought he/she was attractive.
* Partners liked him/her.
* Partners thought he/she was funny.
* How probable do their partners think you will say ‘yes’ (1 to 10)
* Participants and partners had similar interests.

**Negative**

* Participants had met their partner before.
* Gender. (Males are more difficult to match)
* Participants and partners were the same race.
* Partners thought participants were sincere.
* Partners thought participants were ambitious.
* How often do participants go out? (more score means less activity)

# Appendix

#!/usr/bin/env python

# coding: utf-8

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

import matplotlib

%matplotlib inline

from pandas.plotting import scatter\_matrix

from statsmodels.graphics.gofplots import ProbPlot

import statsmodels.api as sm

import statsmodels.formula.api as smf

import sklearn.linear\_model as skl\_lm

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score

from sklearn.discriminant\_analysis import LinearDiscriminantAnalysis

from sklearn.discriminant\_analysis import QuadraticDiscriminantAnalysis

pd.options.display.max\_rows = 1000

from sklearn.impute import SimpleImputer

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.ensemble import GradientBoostingClassifier

from sklearn import metrics

from sklearn.feature\_selection import VarianceThreshold

from sklearn.base import TransformerMixin

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import RidgeCV

from sklearn.linear\_model import LassoCV

from sklearn.linear\_model import ElasticNetCV

from sklearn.metrics import mean\_squared\_error

import warnings;

warnings.filterwarnings('ignore');

# import the dataset

url='https://raw.githubusercontent.com/Gin7/Speed-Dating/master/Speed\_Dating\_Data.csv'

SpeedDating = pd.read\_csv(url,parse\_dates=[0],encoding='latin1')

SpeedDating.head(5)

personality = ['gender', 'field', 'field\_cd', 'age','undergra',

       'mn\_sat', 'tuition', 'race', 'imprace', 'from', 'imprelig',

       'zipcode', 'income', 'goal', 'date', 'go\_out', 'career','museums',

       'career\_c', 'sports', 'exercise', 'tvsports','dining',

        'hiking', 'gaming', 'reading', 'tv', 'art','theater', 'clubbing',

       'movies',  'music', 'concerts','shopping',  'exphappy','yoga',

       'expnum','match\_es']

attributes = ['dec','attr', 'sinc', 'intel', 'fun', 'amb', 'shar', 'like', 'prob',

       'met','match',]

assessment = ['length', 'satis\_2','numdat\_2']

question = ['them\_cal','you\_call', 'date\_3', 'numdat\_3',

       'num\_in\_3']

SpeedDating1 = SpeedDating[['iid', 'wave'] + personality + assessment + question].drop\_duplicates().copy()

SpeedDating1.head()

SpeedDating1.shape

# count null values for all variables

SpeedDating.isnull().sum()

NaN\_col = np.where(SpeedDating.isnull().sum()>1000)

#NaN\_col

SpeedDating1 = SpeedDating.drop(SpeedDating.columns[NaN\_col], axis=1)

SpeedDating1.shape

# Then, we decide to impute the rest values having missing values

class DataFrameImputer(TransformerMixin):

    def \_\_init\_\_(self):

        """Impute missing values.

        Columns of dtype object are imputed with the most frequent value

        in column.

        Columns of other types are imputed with mean of column.

        """

    def fit(self, X, y=None):

        self.fill = pd.Series([X[c].value\_counts().index[0]

            if X[c].dtype == np.dtype('O') else X[c].mean() for c in X],

            index=X.columns)

        return self

    def transform(self, X, y=None):

        return X.fillna(self.fill)

SpeedDating3 = DataFrameImputer().fit\_transform(SpeedDating1)

#SpeedDating3.isnull().sum()

## EDA part

# Matches variable

pd.crosstab(index=SpeedDating2['gender'],columns="count")

# age variable

plt.hist(SpeedDating2.age, bins = 30)

plt.xlabel('Age of Participant')

plt.ylabel('Frequency')

plt.show()

age\_graph = sns.FacetGrid(SpeedDating1, hue='gender', height = 6)

age\_graph.map(plt.hist, 'age', alpha= 0.5, bins=30)

age\_graph.set\_xlabels('Age of Participant', fontsize=13)

age\_graph.add\_legend()

# Race variable

SpeedDating1['race'] = SpeedDating1.race.map({1: 'Black', 2: 'White', 3: 'Hispanic',

                          4: 'Asian', 6: 'Other'}).fillna(SpeedDating1.race)

SpeedDating1['gender'] = SpeedDating1.gender.map({1 : 'Male', 0 : 'Female'}).fillna(SpeedDating1.gender)

print(SpeedDating1[['race','age']].groupby(['race'])\

.agg(['mean', 'median', 'min', 'max', 'count']))

SpeedDating1['race'] = SpeedDating1.race.map({1: 'Black', 2: 'White', 3: 'Hispanic',

                          4: 'Asian', 6: 'Other'}).fillna(SpeedDating1.race)

SpeedDating1['gender'] = SpeedDating1.gender.map({1 : 'Male', 0 : 'Female'}).fillna(SpeedDating1.gender)

print(SpeedDating1[['race','age','gender']].groupby(['race','gender',])\

.agg(['mean', 'median', 'min', 'max', 'count']))

SpeedDating1['field\_cd'] = SpeedDating1.field\_cd.map({1: 'Law', 2: 'Math', 3: 'Social Science', 4: 'Medical Science',

                                 5: 'Engineering', 6: 'Journalism', 7: 'History', 8: 'Business', 9: 'Education',

                                 10: 'Biological Science', 11: 'Social Work', 12: 'Undergrad', 13: 'Political Science',

                                 14: 'Film', 15: 'Arts', 16:'Languages', 17: 'Architecture', 18: 'Other'}).fillna(SpeedDating1.field\_cd)

plt.figure(figsize = (11,6))

ax = sns.countplot(x="field\_cd", data=SpeedDating1)

plt.title('Field of study', fontsize=14)

ax.set\_xticklabels(ax.get\_xticklabels(),rotation=45)

plt.ylim(0, 160)

plt.xlabel('')

#for i in ax.patches:

##    ax.text(i.get\_x()+.2, i.get\_height()+3, \

            #str(round((i.get\_height()), 1)), fontsize=12)

field = SpeedDating1[['gender', 'field\_cd']].groupby(['field\_cd', 'gender']).size().unstack().fillna(0)

ax = field.plot(kind='bar', figsize=(11,5),stacked=True)

ax.set\_xticklabels(ax.get\_xticklabels(), fontsize=11, rotation=45)

ax.set\_title('Field of study by gender', fontsize=14)

ax.set\_xlabel('')

# imprace variable - ethical background

print(SpeedDating1[['race', 'imprace']].groupby(['race']).agg(['mean', 'median', 'min', 'max', 'count']))

# goal variable

SpeedDating1['goal'] = SpeedDating1.goal.map({1: 'Have fun', 2: 'Meet new people', 3: 'Get a date',

                          4: 'For a serious relationship', 5: 'To say I did it', 6: 'Other'}).fillna(SpeedDating1.goal)

goal = SpeedDating1[['gender', 'goal']].groupby(['goal', 'gender']).size().unstack()

ax = goal.plot(kind='bar', figsize=(11,5), ylim=(0,140))

ax.set\_xticklabels(ax.get\_xticklabels(), fontsize=12, rotation=45)

ax.set\_title('Purpose of Dating Event', fontsize=14)

# hobbies: different type of interests

hobbies = ['sports', 'tvsports', 'exercise', 'dining', 'museums',

       'art', 'hiking', 'gaming', 'clubbing', 'reading', 'tv', 'theater',

       'movies', 'concerts', 'music', 'shopping', 'yoga']

hob = SpeedDating1[['gender']+ hobbies].groupby(['gender']).mean().stack().unstack(0)

ax = hob.plot(kind='bar', figsize=(11,5))

ax.set\_xticklabels(ax.get\_xticklabels(), fontsize=10, rotation=45)

ax.set\_title('Hobbies', fontsize=14)

race\_hob = SpeedDating1[['race']+ hobbies].groupby(['race']).mean().stack().unstack(0)

ax = race\_hob.plot(kind='bar', figsize=(14,5),color='rybmg')

ax.set\_xticklabels(ax.get\_xticklabels(), fontsize=11, rotation=45)

ax.set\_title('Hobbies ~ race', fontsize=14)

## Assessment(match, wave)

ax = SpeedDating[['wave', 'match']].groupby('wave').mean().plot(kind='bar', legend=False, figsize=(14,4),

                                                        ylim=(0,0.5))

ax.set\_xticklabels(ax.get\_xticklabels(), fontsize=10, rotation=45)

ax.set\_xlabel('Wave', fontsize=10)

vals = ax.get\_yticks()

ax.set\_yticklabels(['{:,.0%}'.format(x) for x in vals], fontsize=10)

ax.set\_title('Match Rate Per Wave', fontsize=13)

for i in ax.patches:

    ax.text(i.get\_x(), i.get\_height()+.01, \

            str(round((i.get\_height())\*100, 1))+'%', fontsize=10)

ax = SpeedDating[['wave', 'dec']].groupby('wave').mean().plot(kind='bar', legend=False, figsize=(15,5), ylim=(0,0.8))

ax.set\_xticklabels(ax.get\_xticklabels(), fontsize=10, rotation=45)

ax.set\_xlabel('Wave', fontsize=10)

vals = ax.get\_yticks()

ax.set\_yticklabels(['{:,.0%}'.format(x) for x in vals], fontsize=10)

ax.set\_title('Positive feedback rate per wave', fontsize=13)

for i in ax.patches:

    ax.text(i.get\_x(), i.get\_height()+.01, \

            str(round((i.get\_height())\*100, 1))+'%', fontsize=10)

rating = SpeedDating[['iid', 'race', 'gender', 'field\_cd', 'dec', 'match', 'int\_corr', 'samerace', 'met',

             'attr', 'sinc', 'intel', 'fun', 'amb', 'shar', 'like', 'prob']].copy()

rating.head(5)

features = ['attr', 'sinc', 'intel', 'fun', 'amb', 'shar', 'like', 'prob']

num = len(features)

rows = int(num/2) + (num % 2 > 0)

fig, ax = plt.subplots(rows, 2, figsize=(13, 5 \* (rows)))

i = 0

j = 0

for feat in features:

    SpeedDating[SpeedDating.dec==0][feat].hist(label='No', ax=ax[i][j], bins=11, alpha=0.6)

    SpeedDating[SpeedDating.dec==1][feat].hist(label='Yes', ax=ax[i][j], bins=11, alpha=0.6)

    ax[i][j].set\_title(feat, fontsize=12)

    ax[i][j].grid(False)

    ax[i][j].legend()

    j = (j+1)%2

    i = i + 1 - j

plt.subplots\_adjust()

fig.suptitle('Rate Your Partner', fontsize=18)

corrMatrix = SpeedDating.corr()

corrMatrix

#### model selection

SpeedDating4 = SpeedDating3.drop(['field', 'from', 'career','dec\_o'], axis=1)

X = SpeedDating4.drop(['iid','pid','match','condtn'],axis=1).astype(float)

y  = SpeedDating4.match

### Ridge Regrression

ridge\_reg = RidgeCV(alphas=[1, 0.1, 0.001, 0.0005]).fit(X, y)

ridge\_reg.fit(X,y)

### Lasso

lasso\_reg = LassoCV(alphas=[1, 0.1, 0.001, 0.0005]).fit(X, y)

lasso\_reg.fit(X,y)

### Elastic Net

Elastic\_reg = ElasticNetCV(cv=5, random\_state=0)

Elastic\_reg.fit(X, y)

print("Ridge RMSE:", np.sqrt(mean\_squared\_error(y, ridge\_reg.predict(X))))

print("Lasso RMSE:", np.sqrt(mean\_squared\_error(y, lasso\_reg.predict(X))))

print("ElasticNet RMSE:", np.sqrt(mean\_squared\_error(y, Elastic\_reg.predict(X))))

# choose Lasso Regression

coef = pd.Series(lasso\_reg.coef\_, index = X.columns)

print(coef.sort\_values())

imp\_coef = pd.concat([coef.sort\_values().head(10),

                     coef.sort\_values().tail(10)])

matplotlib.rcParams['figure.figsize'] = (8.0, 10.0)

imp\_coef.plot(kind = "barh")

plt.title("Coefficients in the Lasso Model")

print("Lasso picked " + str(sum(coef != 0)) + " variables and eliminated the other " +  str(sum(coef == 0)) + " variables")

#### Resampling methods

# preparing the data

X = X[['dec','attr\_o','like\_o','fun\_o','sinc\_o','samerace','met\_o','gender']].values

y = SpeedDating3.match.values

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=1, stratify=y)

#### Logistic reg

#Keep 13 variables

lr = LogisticRegression(C=1, random\_state=0)

lrc = lr.fit(X\_train, y\_train)

predict\_train\_lrc = lrc.predict(X\_train)

predict\_test\_lrc = lrc.predict(X\_test)

#print('Training Accuracy:', accuracy\_score(y\_train, predict\_train\_lrc))

print('Logistic Regression Model Validation Accuracy:', accuracy\_score(y\_test, predict\_test\_lrc))

#### Random forest

model = RandomForestClassifier()

rf\_model = model.fit(X\_train, y\_train)

predict\_train\_rf = rf\_model.predict(X\_train)

predict\_test\_rf = rf\_model.predict(X\_test)

#print('Training Accuracy:', metrics.accuracy\_score(y\_train, predict\_train\_rf))

print('Random Forest Validation Accuracy:', metrics.accuracy\_score(y\_test, predict\_test\_rf))