Predicting Love – Machine Learning Models on The Bachelor Contestants

EE 660 Course Project

Project Type: (1) Design a system based on real-world data

Number of student authors: 1

Joy Gu – joyg@usc.edu

12/06/2022

1. Abstract

In this project we investigate the performance of machine learning algorithms in predicting when a contestant will be eliminated on The Bachelor, a reality TV dating show. The show's format involves one bachelor who dates approximately 30 female contestants over the course of several weeks, eliminating contestants until a winner remains. Our approach is to extract a variety of contestant features (age, hometown, first impression rose, week of first 1-1 date, and number of roses received in first three weeks) and predict the elimination week of the contestants. In our Supervised Learning (SL) section, we test Linear Regression, LASSO Regression, RIDGE Regression, and Random Forest models on full and reduced feature sets. As an extension task, we predict elimination week for a contestant on its sister show, The Bachelorette, from information learned on The Bachelor. We apply TrAdaBoost2, a Transfer Learning (TL) regression model. Our results show a Ridge Regression model on The Bachelor yielded a test error of about 1.4 weeks. When applying the same model to The Bachelorette data we also see a test error of about 1.4 weeks, and the TrAdaBoost2 method produced an error of 2.4 weeks. We conclude there exists trends and features that can be utilized in predicting who will last longer on the show (week of first date and number of roses received in first three weeks were important features). These trends may not differ that much between The Bachelor and The Bachelorette, as the application of the same SL model performed equally as well, and better than the TL method we tested. However, there is a lot of room for improvement given the limited data points (420 Bachelor contestants, 281 Bachelorette contestants) does not have more recent seasons, and there are many additional features to extract that could be better predictors.

2. Introduction

2.1. Problem Type, Statement and Goals

In this project we investigate how well machine learning algorithms perform in predicting when a Bachelor contestant will be eliminated. This is a regression problem, and the variable we are predicting is the elimination week. Our goal is to evaluate which model performs the best out of the following 4 techniques: Linear Regression, LASSO Regression, RIDGE Regression, and Random Forest. Additional goals include interpreting which features are the most predictive (feature importance) and to incorporate some dimensionality analysis during the model selection process by trying different subsets of the features. The datasets used for this problem pose several challenges – there is a significant amount of

preprocessing to stitch together the two sources of information. The data consists of a mix of continuous features and categorical features, and has high dimensionality, thus we need to define and extract out the important features.

As an extension, we also apply Transfer Learning (TL) on data for the sister show, The Bachelorette. The Bachelorette follows the same format and process as The Bachelor except the lead is a woman and the contestants are men. Thus, we evaluate how a model like TrAdaBoost performs against an SL approach in this related task.

This problem is a fun application of the material learned in class, and as an avid viewer of the show it is interesting to see whether a machine learning algorithm can predict who will fall in love.

2.2. Literature Review

There are 2 existing approaches we will discuss. The first is a paper titled "Predicting Winners of the Reality TV Dating Show The Bachelor Using Machine Learning Algorithms", in which the authors tested Random Forest, Neural Network, and Linear Regression on their own compiled dataset consisting of the following features: Age, Hometown, Job Category, Race, 1-1 Week, First Impression Rose (FIR). They used placement ranking rather than elimination week as the predicted variable, and adjusted which features were being used for each model. The results show the Random Forest Regression model achieved 30.5% accuracy rate, the Neural Network achieved 10% balanced accuracy score, and the Gaussian Process Regression model had an R value of 0.35 [1].

The second approach is a GitHub repository by Kwame V. Taylor, titled "The Bachelorette Predictor". This project uses the same datasets that we use in our project, and the author defines custom features like OneonOneScore to quantify importance of that date based on how many women are left, and FirstDate boolean if the contestant got the first date. The rest of the features are Age and Hometown regions. The author split the data into training, validation, and test data by season, rather than randomly. Linear Regression with OLS, Lasso & Lars Regression, Random Forest were used on validation data, and Random Forest is applied to test data to yield an RMSE=0.549 [2].

2.3. Our Prior and Related Work – None

2.4. Overview of Our Approach

The first step is to preprocess the data and extract the features. This will be covered in Section 3. Once we have our feature matrix, we apply the models to the data. We evaluate performance during the model selection process on the validation set, using mean squared error (MSE). After optimal parameters are selected, we retrain the models on the training and validation set combined, then compute several metrics to compare the algorithms: training MSE, test MSE, test error via mean absolute error, coefficient of determination R^2, and an estimate of

the generalization error bound. These metrics are repeated for the TL extension. The specific list of models and baseline systems are listed in the sections below.

2.4.1 Main topic (SL)

In the main supervised learning approach, the models used are:

- Linear Regression without Regularization
- Linear Regression with L1 Regularization (LASSO)
- Linear Regression with L2 Regularization (RIDGE)
- Random Forest Regressor

We test these algorithms using the full feature matrix, a subset of features pertaining to contestant biography data (Age, Hometown), and using Principal Component Analysis (PCA) with 3 principal components.

The baseline systems tested on the full feature matrix are:

- The trivial baseline: Randomly generated Elimination Week values across range, between [1,11]
- The non-trivial baseline: Linear Regression with single "Age" feature, and Polynomial Regression of degree 2 for single "Age" Feature

2.4.2 Extension (TL)

In the TL extension, the models used are:

- TrAdaBoost with RIDGE Regression estimator, on full feature matrix

The baseline system on the full feature matrix is:

- Directly applying the same SL RIDGE Regression model trained on source data directly on the new target domain

3. Implementation

3.1. Data Set

For the main SL task, we use two datasets.

The first dataset is a Kaggle Dataset from the analysis website FiveThirtyEight. It comprises of contestant data from both the Bachelor (Seasons 1-21) and the Bachelorette (Seasons 1-13) shows and weekly information about when/how they were eliminated, first-impression rose, as well as the order and type of dates (1-1 vs group date). There are 23 columns in total, so we have grouped related columns in the table below for easier interpretation. The following table will cover The Bachelor contestant data (569 contestants).

Feature Name/Group	Type	Cardinality or Range	Description
SHOW	Categorical	1	Show title: The Bachelor
SEASON	Numerical	[1, 21]	Season the contestant was on
CONTESTANT	Categorical	547	Unique ID for the contestant in the form "SEASON_FirstName_LastInitial"
ELIMINATION (Weeks 1-10)	Categorical	8	Weekly info regarding elimination or rose
DATES (Weeks 1-10)	Categorical	16	Weekly info on Dates and how many people on the date

Table 1: FiveThirtyEight Bachelor Data Summary

Additional information regarding the categories is summarized below from the Kaggle description [3]:

- Eliminates connote either an elimination (starts with "E") or a rose (starts with "R").
- "E" connotes a standard elimination, typically at a rose ceremony. "EQ" means the contestant quits. "EF" means the contestant was fired by production. "ED" connotes a date elimination. "EU" connotes an unscheduled elimination, one that takes place at a time outside of a date or rose ceremony.
- "R" means the contestant received a rose. "R1" means the contestant got a first impression rose.
- "D1" means a one-on-one date, "D2" means a 2-on-1, "D3" means a 3-on-1 group date, and so on.

The second dataset we pull from is also from Kaggle, which contains elimination week and biographical information such as age, occupation, and hometown [4]. The bachelor-contestants.csv file is summarized in the table below for the 423 contestants.

Feature Name/Group	Type	Cardinality or Range	Description
Name	Categorical	411	Contestant name
Age	Numerical	[21, 36]	Age (years)
Occupation	Categorical	306	Occupation
Hometown	Categorical	357	Hometown
Height	Categorical	14	Height (inches)
ElimWeek	Numerical	[1, 10]	Elimination Week
Season	Numerical	[1, 21]	Season the contestant was on

Table 2: Kaggle Dataset 2 – Bachelor Data Summary

3.2. Dataset Methodology

After combining the 2 datasets together (details in Section 3.3) and removing missing data, we are left with 420 contestants. We use the following split for the data: 70% training, 10% validation, 20% test. This yielded 294 training points, 42 validation points, and 84 test points. We use a random seed of 100, selected for making sure enough winners were in each set (9 in training, 2 in validation, 4 in test) and for reproducibility. We did this split upfront, so the validation and test sets are separate from the training set.

During the model selection process, we train the model using the training set and use the validation set to compute the MSE and determine the best parameters – there is no cross-validation loop in this process.

In the model performance phase, we use the parameters from model selection and retrain the models on the training and validation combined. Then the test set is used to measure our performance metrics for comparison across models.

We compute the generalization error bound via the following equations from lecture:

$$E_{out}(h_g) \leq E_{D_{test}}(h_g) + \epsilon_M, \epsilon_M = \sqrt{\frac{1}{2N}\ln\left(\frac{2M}{\delta}\right)}, \text{ with probability} \geq 1 - \delta.$$

Equation 1: Out of sample error bound equation

We are using the test set, so we have M=1. There are N=84 test samples, and let $\delta = 0.05$

$\epsilon_M = \sqrt{\frac{1}{2N}\ln(\frac{2M}{\delta})} = 0.148$, which gets added to the test error.

Our final model SL model will be used in the TL extension portion.

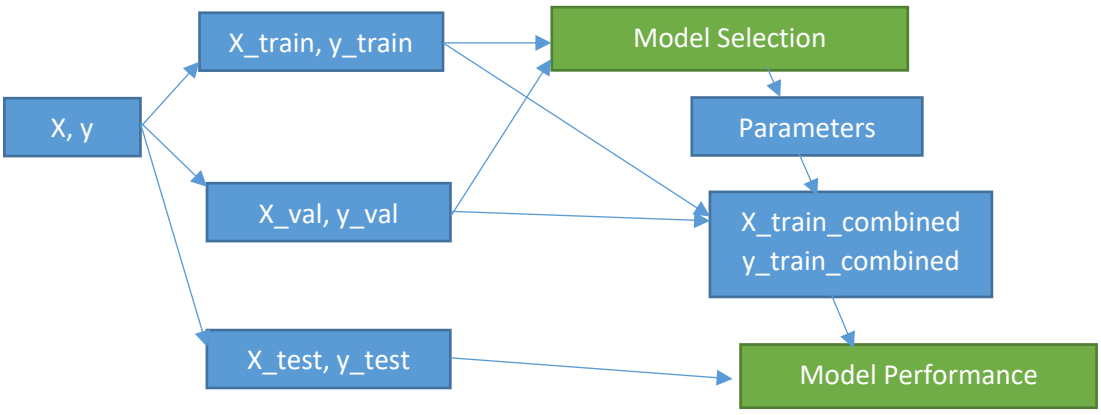

Figure 1: Diagram of Data Methodology

3.3. Preprocessing, Feature Extraction, Dimensionality Adjustment

We use Taylor's approach of combining the data by joining on the Contestant ID column, this requires us to add in a column on the second dataset and generate

the ID based on the season and the name [2]. After creating the ID, we remove duplicate IDs, and join the datasets together. There are still IDs that don't match due to misspellings and nicknames, and to get as much useable data as possible we cycle through the seasons and manually edit the IDs that correspond to the same person.

Once joined, we fill in the NaN's in the Elimination Week (ElimWeek) column. Most of the contestants with NaN's won their season, but a few were eliminated, and their week didn't get transcribed. Thus, we replace the NaN to either 11 for indicating they won and made it past the 10 weeks of the show (looking for "W" in the Elimination columns), or to the week that they were eliminated (looking for "E" in the Elimination columns). This ElimWeek column is the y value we will be predicting.

We extract a First Impression Rose (FIR) feature where 1 indicates the contestant received the FIR and 0 indicates they did not. We do this by searching through the Elimination columns for the "R1" category. We extract this feature because it may be a good indicator of frontrunners as the Bachelor usually selects one person to receive it the first night, although Season 17 had several women receive a FIR [5].

We also create a feature, called NumRoses3, to represent the number of roses received in the first 3 weeks, usually these are group date roses handed out. This value is computed by looking through the first 3 Elimination weeks and counting the number of Roses given to that contestant. Again, the idea is that this would be an indication of a good connection if they received roses early in the season.

We extract a FirstDate feature to capture the week that the contestant's first 1-1 Date happens. If they don't get a first date, then the value is 0. This is another potential indicator of a finalist, as an earlier date may mean a stronger relationship throughout the show's duration.

The final feature we extract is the Hometown region. The existing column in the datasets show the specific city and state, but for easier analysis we convert the city to a region: N – Northeast, W – West, M – Midwest, S – South, O – Other US region, and Other (Non-US country). Then this feature is one-hot-encoded into 6 columns.

We remove the rest of the columns from the original data that we don't use. There are some columns that are interesting to explore, such as Occupation, but due to time constraints we start with this smaller set of features.

We standardize the numerical features (Age, NumRoses3, FirstDate) according to the training statistics. Once we split the data into training, validation, and test sets, we standardize the validation and test set according to the mean and standard deviation of the training set. For model performance, we combine the training and validation set first and then use the combined data statistics to standardize the test set.

For dimensionality analysis, we modify the number of features we feed into our models. The Full Feature Matrix is composed of Age, FIR, NumRoses3, FirstDate, and the encoded Hometown Region. The Contestant Bio Data is only Age and Hometown Region. We also try PCA dimensionality reduction by using the training data to find the first 3 principal components, and then projecting the training, validation, and test data into this space. A summary of the input data for each model is below:

MODEL	INPUT FEATURES USED		
LINEAR REGRESSION, NO REGULARIZATION	Full Feature Matrix	Contestant Bio Data	PCA
LASSO REGRESSION	Full Feature Matrix	Contestant Bio Data	PCA
RIDGE REGRESSION	Full Feature Matrix	Contestant Bio Data	PCA
RANDOM FOREST	Full Feature Matrix	Contestant Bio Data	PCA
TRIVIAL BASELINE (RANDOM)	No features		
NONTRIVIAL BASELINES (LINEAR AND POLYNOMIAL REGRESSION)	Age feature		

Table 3: SL Model and corresponding input features used

3.4. Training Process

For all the linear regression models we set *intercept=False* since we standardized the numerical data, and for easier interpretation of the coefficients. We describe all the models and training process below, with the final parameters chosen listed in the table in the next section, Section 3.5.

Linear Regression without Regularization

This is a simple multiple linear regression such that the prediction y is a weighted sum of the input features. There are no hyperparameters to choose as it is solved via Least Squares, so after training we directly evaluate it on validation or test data. We chose Linear Regression as a simple model to investigate and interpret feature importance, as magnitude of the weight w_i can give insight into how much feature x_i affects the prediction.

$$\boldsymbol{y} = \sum_{i=1}^{n\_features} w_i \, \boldsymbol{x_i}$$

Equation 2: Linear Regression model

Linear Regression with L1 Regularization (LASSO)

We chose LASSO Regression to prevent overfitting that may occur without regularizing the coefficients. The training process involves selecting an optimal regularization coefficient λ. For each value within range $-10 \leq \log_2 \lambda \leq 10$ with step size of 0.5 for $\log_2 \lambda$, we train the LASSO model with that λ and compute the validation MSE. Then select the coefficient with the lowest MSE as the optimal value.

Linear Regression with L2 Regularization (RIDGE)

We repeat the same process as LASSO Regression but use the RIDGE Regression method since it also prevents overfitting without potentially reducing features to a zero weight.

Random Forest Regressor

We chose a Random Forest model as some of the features may not have linear relationship with the predicted value. We train the Random Forest and evaluate validation MSE across 2 parameters: we vary the number of trees in set [10, 50, 100, 150], and max depth in the set [None, 1, 2, 3, 4].

Trivial Baseline: Random

We chose this as a simple baseline to randomly assign the elimination week, we set the random seed to 500 for repeatability. No other parameters are needed, so baseline results will be shown test data only in Section 4.

Nontrivial Baseline: Simple Linear Regression

We use the Age feature for a simple linear regression analysis, as there may be a linear relationship between the contestant age and how far they make it. No other parameters are needed, so baseline results will be shown test data only in Section 4.

Nontrivial Baseline: Polynomial Regression

We also try a polynomial of degree 2 regression for the Age feature, as the thought is that contestants that are on the younger or older extremes don't usually make it to the end of the show. No other parameters are needed, so baseline results will be shown test data only in Section 4.

3.5. Model Selection and Comparison of Results

The optimal parameters the resulting validation MSE is displayed below. We see that the Contestant Bio data does not perform well, so we will not use that as input data in our final comparison. PCA transformation did not perform well in the three regression methods, but it performed the best when paired with the Random Forest method. Thus, for our final comparison of methods we will use the three Regression techniques with the Full Feature matrix, and Random Forest on both Full Feature and PCA Feature matrices.

Model	Full Feature	Contestant Bio	PCA
Linear Regression (no reg)	Val MSE = 2.917 No hyperparameters to choose	Val MSE = 7.938 No hyperparameters to choose	Val MSE = 14.856 No hyperparameters to choose
LASSO Regression	Val MSE = 2.850 $Log_2\ \lambda\ =$-4.5	Val MSE =7.743 $Log_2\ \lambda\ =$-7.0	Val MSE =14.667 $Log_2\ \lambda\ =$-2.5
RIDGE Regression	Val MSE = 2.877 $Log_2\ \lambda\ =$1.5	Val MSE =7.745 $Log_2\ \lambda\ =$ 0.0	Val MSE =14.726 $Log_2\ \lambda\ =$ 6.5
Random Forest	Val MSE =2.703 N_trees = 50 Depth = 2	Val MSE =7.705 N_trees = 10 Depth = 1	Val MSE = 2.520 N_trees = 10 Depth = 2

Table 4: SL Model Validation Results

The results of the down-selected models are below, where we retrain the models on the training and validation sets combined.

Model	Full Feature	PCA
Linear Regression (no reg)	Train MSE = 2.405 Test MSE = 3.028 Test Error = 1.385 Generalization Bound = 1.533 R^2 =0.621]	
LASSO Regression	Train MSE = 2.593 Test MSE = 3.303 Test Error = 1.434 Generalization Bound = 1.582 R^2 =0.587	
RIDGE Regression	Train MSE = 2.455 Test MSE = 3.101 Test Error = 1.389 Generalization Bound = 1.537 R^2 = 0.612	
Random Forest	Train MSE = 2.407 Test MSE = 2.771 Test Error = 1.392 Generalization Bound = 1.540 R^2 = 0.653	Train MSE = 2.409 Test MSE = 2.973 Test Error = 1.405 Generalization Bound = 1.553 R^2 = 0.628

Table 5: SL Model Test Results

In the figures below, we can see how the Random Forest models have a larger R squared value but seem to cluster the predictions towards the two ends, whereas the Regression techniques are better at spacing out the predictions.

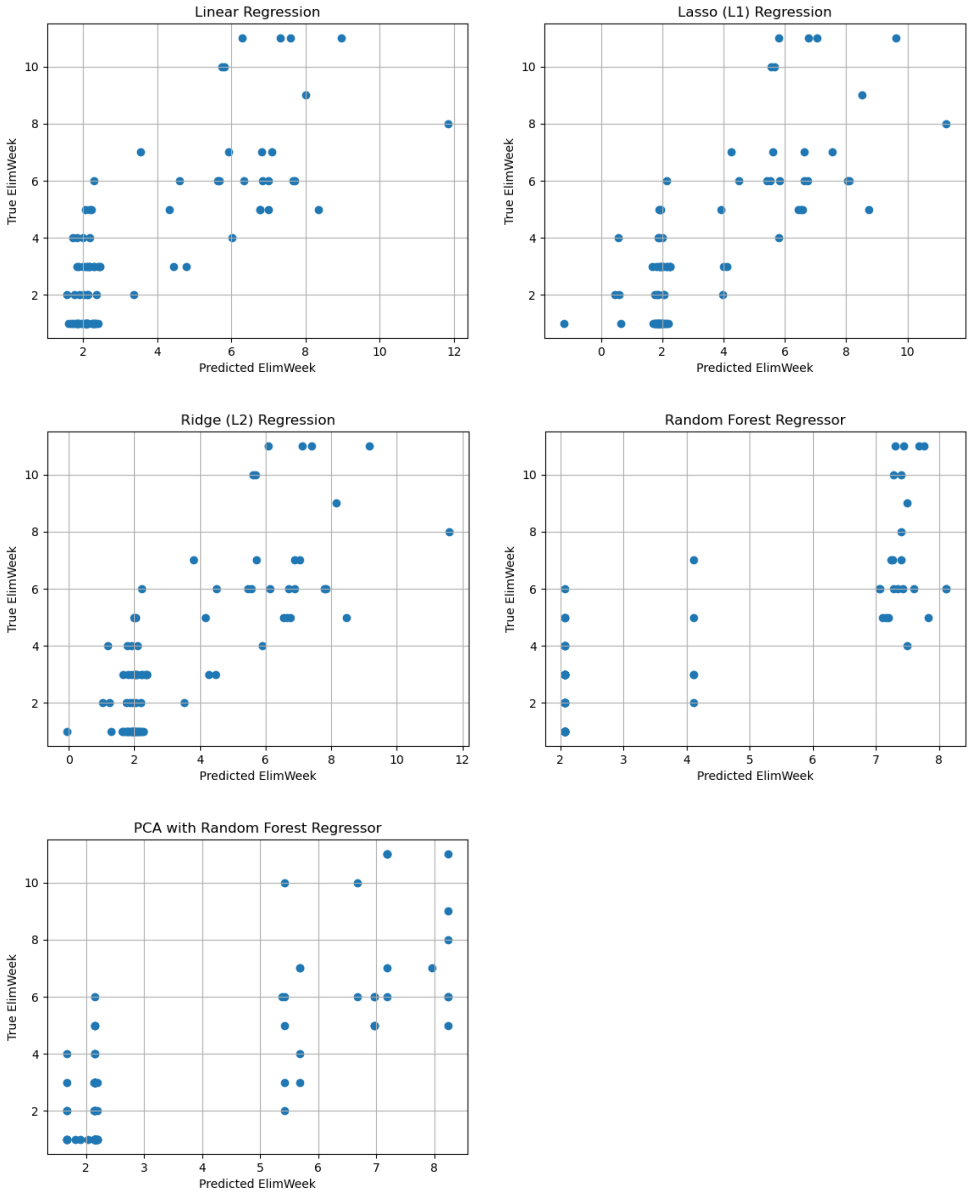

Figure 2: SL Results True Elimination Week vs Predicted Elimination Week

4. Final Results and Interpretation

The final system we choose is Ridge Regression on the Full Feature Matrix, using parameter $Log_2 \lambda = 1.5$. Though it does not have the lowest test MSE, we choose this system out of the candidates because we want to prevent overfitting that general Linear Regression doesn't provide. In this instance Ridge is preferred over LASSO so that important features don't end up with a zero coefficient, and the generalization bound is lower. It is also

preferred over the Random Forest techniques since the predictions do not favor specific elimination weeks and span the full range.

We compare this system with the baselines in the table below:

Model	Result
RIDGE Regression on Full Features	Train MSE = 2.455 Test MSE = 3.101 Test Error = 1.389 Generalization Bound = 1.537 $R^2 = 0.612$ Weights = [-0.14962212 -0.75869104 1.07706585 1.59609722 3.34387172 3.38916929 1.18577412 2.44203587 3.2303528 3.46509007]
Trivial Baseline: Random	Train MSE = 7.406 Test MSE = 28.095 Test Error = 4.405 Generalization Bound = 4.553 $R^2 = -2.514$
Nontrivial Baseline: Linear Regression on Age	Train MSE = 7.423 Test MSE = 7.874 Test Error = 2.230 Generalization Bound = 2.378 $R^2 = 0.015$ Weights = [-0.08286057]
Nontrivial Baseline: Polynomial Regression Deg= 2 on Age	Train MSE = 7.406 Test MSE =7.883 Test Error = 2.228 Generalization Bound = 2.376 $R^2 = 0.014$ Weights = [0.33547789 -0.00772099]

Table 6: SL Final Model and Baseline Result

We see that the training and test MSE values of our chosen system beats the baseline performance, and the prediction plots below confirm that the baseline predictions do not align well with the true Elimination Week.

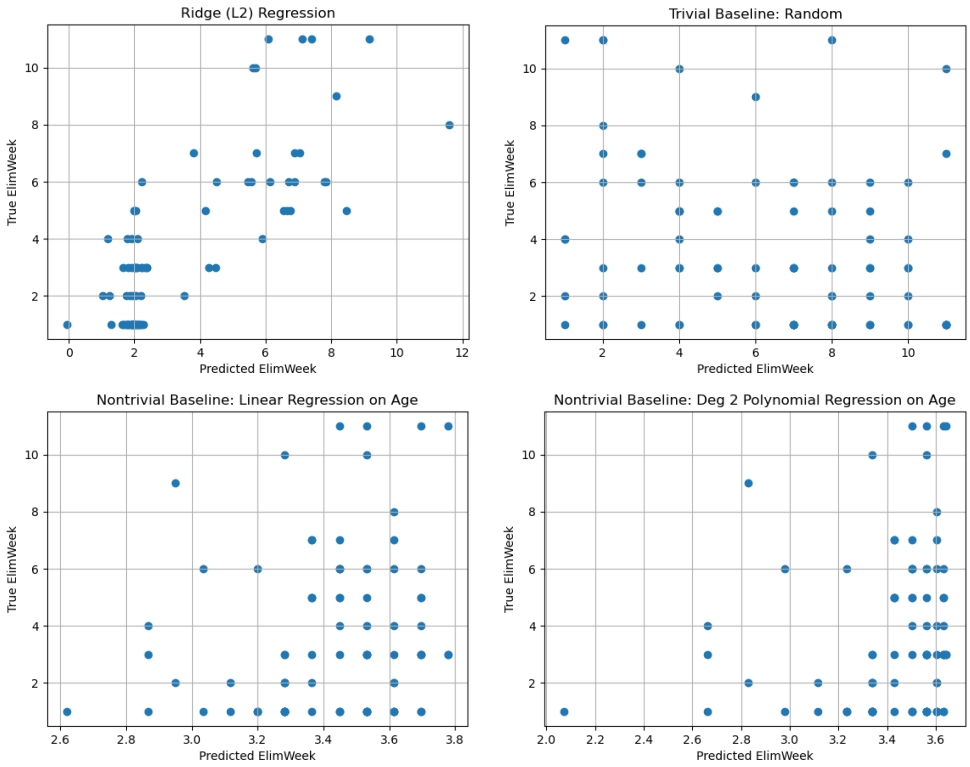

Figure 3: SL Final Model and Baseline True ElimWeek vs Predicted ElimWeek

For feature importance, the weights of the RIDGE regression correspond to the effect of the features in the following order: age, FIR, NumRoses3, FirstDate, Hometown Regions for Midwest, Northeast, Other US, Other International, South, West. NumRoses3 and FirstDate, with weights of 1.596 and 1.077 respectively, are relatively good predictors amongst the set of features that we standardized. For the one-hot-encoded regions, we see that being in a smaller region of the US and being international did not have high feature importance (1.186 and 2.44) compared to the other US regions. The West region had the highest weight value of 3.465. Initially we would have expected FIR to be a big predictor, but according to the Bachelor Data Analyst (an Instagram account dedicated to analyzing the show's data), getting a FIR on the Bachelor show is not as good in indicating winners as it is on the Bachelorette [6].

Comparing our results to the resources in our literature review, we had an R squared value of 0.612, or R = 0.782, which is higher than R=0.35 of the Gaussian Process Regression in [1]. The Random Forest Regressor in [2] achieved an RMSE of 0.5487. Our Ridge regression solution as a test MSE of 3.101, RMSE = 1.761, so there is still a lot of room for improvement in prediction. However, it is important to note a few differences in their methods that may affect this comparison – we predicted Elimination Week on a random test set, in [1] the authors predicted the Ranking of the contestants, and in [2] the author predicted Elimination Week on contestants from Seasons 11 and 12.

In conclusion, our test error shows we can predict elimination within 1.4 weeks of accuracy on average. To improve these results, we could investigate more nonlinear algorithms to try

since the Random Forest Regressor had the best MSE, but the predictions were too clustered.

5. Implementation for the extension

5.1. Data Set

For the TL task, we use the same two Kaggle datasets but look at the Bachelorette contestants instead of the Bachelor.

The FiveThirtyEight datasets for the Bachelorette (Seasons 1-13) is summarized in the table below for 350 contestants [3].

Feature Name/Group	Type	Cardinality or Range	Description
SHOW	Categorical	1	Show title: The Bachelorette
SEASON	Numerical	[1, 13]	Season the contestant was on
CONTESTANT	Categorical	350	Unique ID for the contestant in the form "SEASON_FirstName_LastInitial"
ELIMINATION (Weeks 1-10)	Categorical	8	Weekly info regarding elimination or rose
DATES (Weeks 1-10)	Categorical	14	Weekly info on Dates and how many people on the date

Table 7: FiveThirtyEight Bachelorette Data Summary

The second dataset from Kaggle, bachelorette-contestants.csv, is summarized in the table below for the 282 contestants [4].

Feature Name/Group	Type	Cardinality or Range	Description
Name	Categorical	279	Contestant name
Age	Numerical	[23, 42]	Age (years)
Occupation	Categorical	228	Occupation
Hometown	Categorical	242	Hometown
ElimWeek	Numerical	[1, 10]	Elimination Week
Season	Numerical	[1, 13]	Season the contestant was on

Table 8: Kaggle Dataset 2 - Bachelorette Data Summary

5.2. Dataset Methodology

The dataset methodology for the Bachelorette is identical to the Bachelor data in Section 3.2. We have a Bachelorette dataset of 281 contestants. We use the same split for the data: 70% training, 10% validation, 20% test. This yielded 196 training points, 28 validation points, and 57 points. We use a random seed of 15, which resulted in 7 winners in training, 2 winners n validation, 4 winners in test.

5.3. Preprocessing, Feature Extraction, Dimensionality Adjustment

This process of getting the input features is identical to Section 3.3 .

The models and what input features are used for the TL extension is summarized below:

MODEL	INPUT FEATURES USED
TRADABOOST WITH RIDGE	Full Feature Matrix
BASELINE: RIDGE REGRESSION FROM SL	Full Feature Matrix

Table 9: TL Models and Input Features

5.4. Training Process

We describe all the models and training process below, with the final parameters chosen listed in the table in the next section.

TrAdaBoost2, with Ridge Regression base model

We choose TrAdaBoost2, the regression version of TrAdaBoost, as our transfer learning approach based on the diagram from the Awesome Domain Adaptaion Python Toolbox (ADAPT) resource [7]. This is a supervised domain adaptation approach that uses instance weighting. We choose Ridge Regression as our base model as it was the final SL model we chose from the main task. We use the optimal parameter of $Log_2\ \lambda\ =$1.5 for the model. During training we want to select the optimal number of estimators and learning rate for the TrAdaBoostR2 model. We run a nested loop across the following number of estimators: [1, 5, 10, 15, 20]; and we use the following learning rates: [0.1, 0.5, 1,2]. We use the source and target training data for this process and compute the target validation MSE to find the optimal parameters.

Baseline: Ridge Regression from SL portion

As a simple baseline we want to test the Ridge Regression model from our main task and apply it to the new target domain data. The model will be trained on the source data, exactly as described in Section 3.4, so we load in the pre-trained model with the same parameters.

5.5. Model Selection and Comparison of Results

The optimal parameters and validation MSE for that system are listed in the chart below.

Model	Full Feature Matrix
TrAdaBoost2 with Ridge	Val MSE = 3.460 n_estimators = 1 learning_rate = 0.1 $log_2\ \lambda\ =$ 1.5, from SL section
Baseline: SL Ridge Regression	Val MSE = 2.850 $Log_2\ \lambda\ =$ 1.5, from SL section

Table 10: TL Models Validation Results

The results are surprising that the baseline model had a better validation result, as this was pre-trained on the source data.

6. Final Results and Interpretation for the extension

Our final Transfer Learning system is the TrAdaBoost2 model with 1 estimator, a learning rate of 0.1, and using Ridge regression as the base model with $Log_2\ \lambda\ = 1.5$. Test results on the target domain data set is in the chart and plots below.

Model	Result
TrAdaBoost2 with Ridge	Training MSE: 5.283 Test MSE: 7.362 Test Error: 2.045 R^2 = -0.013
Baseline: SL Ridge Regression	Training MSE: 2.455 Test MSE: 3.740 Test Error: 1.442 R^2 = 0.491

Table 11: TL Final Model and Baseline Test Results

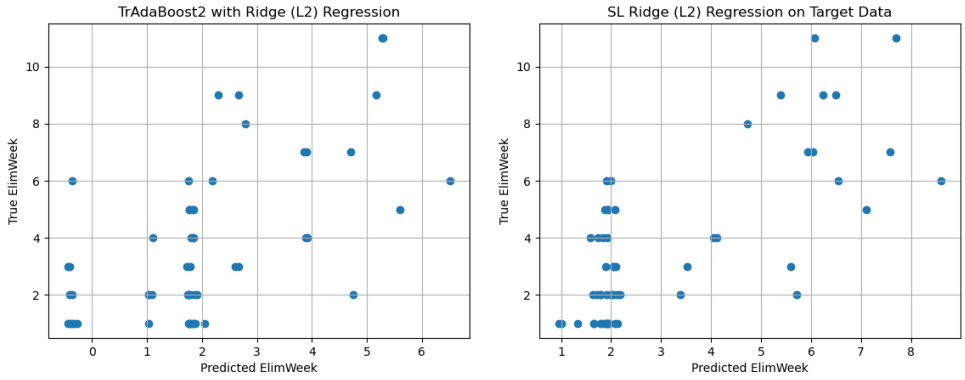

Figure 4: TL Model and Baseline True ElimWeek vs Predicted ElimWeek

The results show that the baseline has a better test MSE and R squared value compared to the TL technique we chose. Looking at the plots, it looks like the baseline has predicted lots of contestants to be eliminated in Week 2, but there's no prediction past 9 weeks. For TrAdaBoost2, this cluster around Week 2 is reduced, but we also don't have a prediction past 7 weeks. These characteristics are probably due to having insufficient data, as there's only 2 winners in the test set to measure how well our model is performing. Additionally, there may be minimal difference between the two shows on the features we've chosen – as we saw in the SL experiments, using only the contestant bio data (age being a main difference between the two populations) didn't yield good predictions, so perhaps there's less of a domain shift than we assumed. To improve this approach, we would want to do more statistical work in the feature sets of both domains to try and determine a more accurate importance weighting.

7. Contributions of each team member

N/A – Individual project

8. Summary and conclusions

Our key findings in this project include:

- For the Supervised Learning models tested for the Bachelor dataset, Ridge regression is the preferred model for having a low generalization error bound and reasonable R squared value.
- Using only contestant age and hometown is not a good predictor for how long someone will last on the show. Additionally reducing the feature space via PCA did not produce better results either.
- Applying the Supervised Learning model from the Bachelor on the Bachelorette dataset achieved better results than using the TrAdaBoost2 regression model, potentially indicating that the effects of the chosen features lead to similar results on both shows.

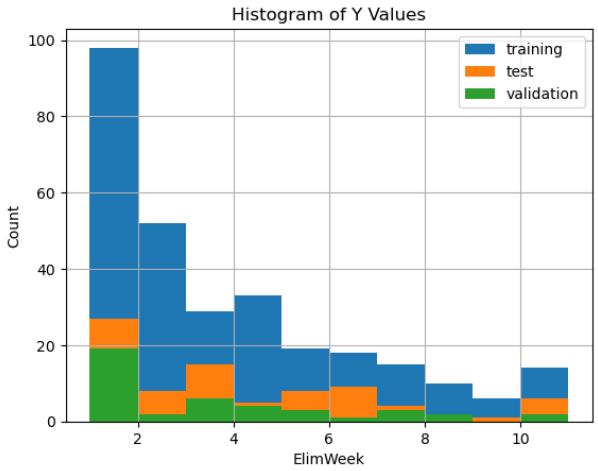

Figure 5: Histogram of Bachelor Contestants' Elimination Week

Given that the data I used is a few years old, it does not contain several of the most recent seasons. Thus, it would be interesting to see how the results differ if we add in more data or how well the current models perform when predicting results on these new seasons. In the histogram plot, we see the distribution of the elimination weeks has a known form (mostly lower numbers as many people leave the show early, and only a few winners), another extension would be to try to use this half-Guassian form in the model.

Additionally, there is an Instagram account (@bachelordata, The Bachelor Data Analyst) dedicated to analyzing data from the show, such as screentime and order of hometown dates, and it would be fun to incorporate some of their findings as new features.

References

[1] A. J. Lee, G. E. Chesmore, K. A. Rocha, A. Farah, M. Sayeed and J. Myles, "Predicting Winners of the Reality TV Dating Show The Bachelor Using Machine Learning Algorithms," 30 March 2022. [Online]. Available: https://arxiv.org/pdf/2203.16648.pdf. [Accessed Nov 2022].

[2] K. V. Taylor, "The Bachelorette Predictor," Jan 2021. [Online]. Available: https://github.com/KwameTaylor/bachelorette-predictor/blob/main/bachelorette-predictor.ipynb. [Accessed Nov 2022].

[3] FiveThirtyEight, "FiveThirtyEight Bachelorette Dataset," Feb 2018. [Online]. Available: https://www.kaggle.com/datasets/fivethirtyeight/fivethirtyeight-bachelorette-dataset . [Accessed Nov 2022].

[4] B. Gonzalez, "The Bachelor & Bachelorette Contestants," 2016. [Online]. Available: https://www.kaggle.com/datasets/brianbgonz/the-bachelorette-contestants. [Accessed Nov 2022].

[5] "Bachelor Nation Wiki - First Impression Rose," 2022. [Online]. Available: https://bachelor-nation.fandom.com/wiki/First_Impression_Rose. [Accessed Nov 2022].

[6] bachelordata, "First Impression Rose: How far do FIR recipients make it on The Bachelor?," Jan 2022. [Online]. Available: https://www.instagram.com/p/CYSLiSkPAnf/. [Accessed Nov 2022].

[7] ADAPT, "Selecting the right domain adaptation model," 2020. [Online]. Available: https://adapt-python.github.io/adapt/map.html. [Accessed Nov 2022].