Predicting Marriage Outcomes Using Couples Therapy Questions

Name: Zachary Stiles Email: zmstiles@crimson.ua.edu

*Abstract*—This paper is an analysis of the process, execution, and results of machine learning models developed to predict if a marriage will end in divorce. This analysis compares the research conducted here with research previously done on the dataset used. Through a feature selection process designed to maximize correlation and predictive power, and minimize redundancy, the dataset was reduced from 54 indicative features to seven. These features were evaluated on three machine learning models implementing the strategies of logistic regression, decision tree classification, and randomized forest classification. These models showed promising accuracy in the evaluating the test data used. While in practice it is hard to be sure if these models can predict a divorce in advance, they have the potential to evaluate or identify if there are present problems in a relationship that could lead to a divorce in the future.

Keywords—divorce prediction, data science, machine learning models

# Introduction

Divorce has increasingly become an issue worldwide, gradually increasing in rate over the past five decades. Many couples seek help through means of therapy when the health of their relationships start to decline. Unfortunately for these couples, often it is too late before the problems in a marriage are cognitively recognized an action is taken. If these couples could have an indication of what the struggles in their relationship projects for the future, could they make changes before it is too late for their marriage?

# Research Question

My research question is “Can the outcome of a marriage be predicted by a spouse’s opinions of the state of their marriage?” In other words, “Can we use a spouse’s current feelings and views about his or her marriage to predict whether or not that marriage will end in a divorce?” This question is a simple binary classification problem, with the two classes being “will divorce” and “will not divorce.”

This question is obviously not to be taken seriously. The outcomes of any models created for this project are not intended to diagnose the state of a marriage or suggest a married couple file for divorce.

# Related Work

This same topic was discussed by Dr. Yöntem et al. (henceforth, “Researchers”) on the same dataset used to answer my research question. Researchers used three different models to evaluate varying numbers of features in their research. Researchers received their features from the Gottman Insitute, a Turkish-owned couples therapy business. Gottman couples therapy contains a 54 question survey known as the Divorce Predictors Scale (DPS). Each question served as an initial feature in their research. They obtained the data via in-person interviews and Google Drive survey submissions from married and divorced couples. This is the same dataset that I used for my own analysis (discussed more in IV.).

After feature acquisition, Researchers used a method of correlation-based feature selection to reduce the total features down from 54 to six, selecting only the features with the highest correlation to a couple’s divorce: Q2: “I know we can ignore our differences even if things get though sometimes.”; Q6: “We don't have a common time we spent together at home.”; Q11: “We don't have a common time we spent together at home.”; Q18: “We have similar ideas with my spouse about how a marriage should be.”; Q26: “I know the basic concerns of my spouse.”; Q40: “We're starting to fight before I know what's going on.”

Researchers ran both sets of 54 and six features on three different machine learning models: artificial neural networks (ANN), random forest, and radial basiss function (RBF) classification. For the 54 feature set, their models predicted with similar degrees of accuracy: ANN: 97.64%; RBF: 98.23%; random forest: 97.64%. Interestingly, the six feature set scored better, and one of them scored worse: ANN: 98.82%; RBF: 97.64%; random forest: 97.64% [1].

While this research was on the same dataset, I hoped to provide a more refined approach to selecting features and use additional models to further explore the data and take of note of any differences in performance of the models used in this project.

# Methods

## Data Acquisition

As discussed in III., the data used for this project was previously collected through in-person interviews and Google Drive surveys for the research discussed above. The dataset was obtained on Kaggle from [2], sourced from a different party than Researchers above. The file for the data was obtained in a

The data came in the form of a .csv file with semicolons as delimiters, containing 55 columns and 170 rows. The first 54 of these columns represented answers to the 54 DPS questions, respectively, ranging from integers zero (0) to four (4). These numbers represent a linear scale of how much a couple or spouse agrees or disagrees with the question or statement: zero (0) meaning “strongly disagree” or “never”, one (1) meaning “disagree” or “seldom”, two (2) meaning “neutral” or “averagely”, three (3) meaning “agree” or “frequently”, and four (4) meaning “strongly agree” or “always”. The last column contained a binary value of zero (0) or one (1) to represent whether a couple was divorced or not divorced, respectively. Each row of the data contained the entire set of answers and marital status of a single couple [2].

## Data Cleaning and Labeling

First it was important to ensure the data contained no null values. I decided to remove any rows that contained a null or missing value from the data. Fortunately, every row was fully populated.

One of the best qualities of the dataset is the zero (0) to four (4) scale that every value has. Normally on a numerical dataset, it would need to be standardized or normalized so that the varying ranges of numbers do not interfere with the feature engineering process. However, since all values are on the same scale, there is no need to standardize or normalize the data.

Another benefit of the dataset is the “Divorce” column. Because I am trying to predict whether a not a couple will get a divorce, this column serves as the target variable for the machine learning models. However, I found it confusing that zero (0) was used to indicate “divorced” and one (1) was used to indicate “not divorced.” Following the tradition of zero (0) representing false and one (1) representing true in binary, I decided to switch the values in the dataset to make it less confusing, which is the extent of the data labeling done in this project. Lastly, I decided to create a count plot to display how balanced the target variable is for the dataset [3]. Fortunately, there was a very even balance of “not divorced” to “divorced” couples in the dataset, which will greatly improve the reliability of the models to be developed.

Chart, bar chart

Description automatically generated

Fig.1. A count plot of the target variable.

## Feature Engineering

I found the research in III. to be overly simplistic for the feature engineering process, so I wanted to expand on the subject. However, I believe looking at correlation values was a great place to start. I decided to create a heatmap of the absolute magnitude of the correlation values for every feature in the dataset [4]. Furthermore, I wanted to use the Predictive Power Score (PPS) as a metric for feature selection. While correlation is fine for summarizing linear relationships between variables, it completely neglects nonlinear relationships. Furthermore, a correlation plot or heatmap is symmetric across the diagonal, but the PPS of one variable to another is completely different from the PPS of the second variable to the first [5]. Thusly, I created a heatmap of the PPS as well. The main rows to look at for the first step of the feature engineering process are the “Divorce” rows in each heatmap. Lighter colored intersections with these rows represent higher correlation or PPS with the variable.

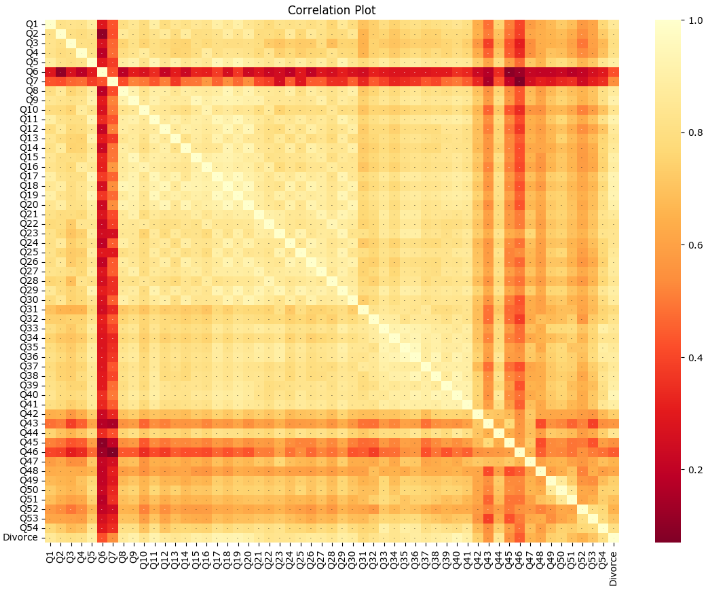


Fig. 2. The correlation heatmap between all 54 features and target variable “Divorce”.

**A picture containing text

Description automatically generated**

Fig. 3. The Predictive Power Score heatmap between all 54 features and target variable “Divorce”.

Looking at the heatmaps, it is interesting to note that there is a very high correlation between most features. Also, the row containing the highest values on the PPS heatmap is the “Divorce” row, which is promising for model performance.

I started by selecting features using PPS in reference to the target variable. Because the dataset is so small, I needed to drastically reduce the number of features to prevent overfitting of models [6]. I set a minimum requirement of 0.7 PPS, which immediately eliminated half of the features. With this subset, I took after the original researchers of the project and took only the features with the highest correlation values, setting a minimum of 0.8 for the correlation with the “Divorce” variable.

After this first stage of feature selection, there was a subset of 9 values remaining. While the original translation from Turkish to English was poor, I referenced the Gottman Institute’s website to find the appropriate translation: Q5: “The time I spend with my spouse is special for the both of us.”; Q9: “I look forward to vacations with my spouse and enjoy the travel we do together.”; Q17: “My spouse and I have similar ideas about how we find happiness in life.”; Q18: “My spouse and I have similar ideas about how marriage should be.”; Q19: “My wife and I have similar ideas about how roles should be in marriage.”; Q20: “My wife and I have similar values in trust.”; Q29: “I know my spouse very well.”; Q36: “I am humble in discussions with my spouse.”; Q40: “I know why my partner is upset before we have an argument.” [7]. It is important to note that this changed the meaning of questions 36 and 40. Question 40 was used in the original research of this dataset, but notably had a translation that no longer exists on the Gottman couples therapy’s questionnaire [1].

Even after greatly reducing the number of features, I wanted to reevaluate the correlation plot once again. The heatmap displayed very high correlation values between most of the features, so for good measure I created new heatmaps for both correlation and PPS and take a closer look.

Chart, treemap chart

Description automatically generated

Fig. 4. The correlation heatmap between the reduced set of nine features and target variable “Divorce”.

**Chart, treemap chart

Description automatically generated**

Fig. 5. The correlation heatmap between the reduced set of nine features and target variable “Divorce”.

With the reduced feature set, the correlation and PPS values could be displayed on the heatmaps. However, the correlation between many of the features is very high. The first goal of this feature engineering process was to select features that maximize correlation and predictive power, but the second is to reduce redundancy. Having variables with too high of correlation can cause models to overcompensate for these features and skew prediction results [8], [9].

Thusly, I decided to remove “duplicates” of features having correlation values over 0.95 with each other. While this seems like a very high value, I think a degree of leniency is required when evaluating these particular features, as they are based on psychological answers to questions about personal relationships; I do not want to discount the relevancy of a spouse’s opinion of their marriage. However, this small filter eliminated questions 19 and 20 from the feature set, as they overlapped with questions 17 and 18, respectively.

This reduced the total number of features for predicting the target variable to seven. Interestingly, the only features in my reduced set that overlap with the original researchers’ feature set were questions 18 and 40.

## Model Development

I decided to develop three models for this research using the methods of logistic regression, decision tree classification, and randomized forest classification.

The first method I used was logistic regression, a classic classification method despite being called a “regression” technique. While linear regression is also fine for classification problems, predicting whether a couple will get divorced is not fully black and white; there may be a chance closer to 50 percent that the couple will indeed divorce. Logistic regression will benefit the deployment of this model as the probabilities of each category, “not divorced” and “divorced” can be displayed alongside whatever classification the model predicts [10]. I decided to maintain the default values used in scikit-learn’s logistic regression model. Most notable of these parameters is the default 100 maximum number of iterations permitted to converge [11].

I decided to develop a decision tree classification model as the second model, largely because I wanted to compare its performance to a randomized forest classification’s performance. For the decision tree classifier, I used Gini impurity for the classification criteria, which calculates the probability of for incorrectly classifying features [12]. I did not set a maximum depth of the decision trees, as I found setting values lower than five worsened the prediction performance of the model [13].

Qr code

Description automatically generated

Fig. 6. A decision tree used on the training dataset assigned at random state 13.

The last model I created was a randomized forest classification. Because random forest models take several decision tree classifiers on subsets of the data and then averages results to increase accuracy, I created the third model to compare its performance to that of the second model. I also was interested in comparing this model’s performance to that of the randomized forest model used for this dataset’s original research project. I decided to maintain use of the Gini impurity criteria and not set a maximum tree depth; the same settings used for the decision tree classification model [14].

## Model Training

While model training, I opted to split the dataset to allow for 75 percent training set size. This resulted in a training size of 127, leaving the remaining 25 percent for a test size of 43. Initially, I divide the dataset for a 70 percent training size, but upon increasing it to 75 percent saw an increase in prediction accuracy. Further increasing it to 80 percent resulted in negligible change. All models used the same training and test datasets, which were split at a random state of 27, although changing this value did not affect performance. As aforementioned, most hyperparameters were left in default values and the decision tree classification and randomized forest classification models used Gini impurity.

## Model Evaluation and Validation

To evaluate the three models, I used two main techniques: accuracy scoring and a confusion matrix. While essentially these display the same values in a binary classification problem, the confusion matrix helps achieve the data visualization aspect of data science [15].

## Model Deployment

To deploy a model to the web, I first used Streamlit to create an application [16]. After viewing the performance results (see V.), I opted to use my logistic regression model for the application. For the application, I used the seven question features to create a sidebar that allows a user to select answers from integers zero (0) to four (4) on sliders, with a default setting of two (2) [17]. The values of these sliders served as datapoints for the model to predict the target variable. I also utilized the logistic regression model’s predicted probabilities to create a confidence level graph to show how strong or weak the model’s prediction is. This technique was previously used by Xishi Zhu in the third homework assignment [18]. Lastly, I displayed all the test data used to train the model at the bottom of the application.

After creating a repository on GitHub for the application, I used Streamlit Cloud to publish the app freely. All that was required was to create a “requirements.txt” file in the GitHub repository using the pipreqs [19], [20]. The app has maintained deployment since its publishing.

# Results

## Model Performance and Evaluation

All three models performed very well. For each model, there was an accuracy test run on the predictions made for the test data subset. Then, a confusion matrix was created to visualize the results.

The first model was the logistic regression model, which had the best results of any model, boasting a 100 percent accuracy score.

Chart, treemap chart

Description automatically generated

Fig. 7. The confidence matrix for the first model, logistic regression.

The second model was the decision tree classification model, which had the worst performance at 95.35 percent. The model incorrectly predicted two “not divorced” outcomes as “divorced” outcomes.

Chart, treemap chart

Description automatically generated

Fig. 8. The confidence matrix for the second model, decision tree classification.

The third and final model was the randomized forest classification, which had slightly better performance than the decision tree classification model, which was to be expected. This valued at an accuracy score of 97.67 percent, which is only a slight improvement over the original researchers’ random forest model used on this dataset. This model only incorrectly predicted one “not divorced” outcome as a “divorced” outcome.

Chart, treemap chart

Description automatically generated

Fig. 9. The confidence matrix for the third model, randomized forest classification.

The overall model performance was impressive. While one of the models performed worse than the original research done on this project, even with the refined feature engineering process, the logistic regression model exceeded expectations with its perfect accuracy.

## Feature Importance

Despite the similar performance levels of these models and the original researchers’ models, the features used were largely different. As aforementioned, only two of the same features were used for model training and testing, those being questions 18 and 40.

When a user first opens the model’s application, the default values for all answers are two (2), which results in a very high confidence in the prediction that the couple will not get divorced, being upwards of 95 percent. As the answers to the question increase in value, the likelihood of not getting divorced continues to increase. Likewise, as the answers to each question decrease in value, the likelihood of getting a divorce increase. Also of note is that setting all answers to a value of one (1) generates a prediction of getting a divorce with a confidence of roughly 65 percent.

Chart, bar chart, box and whisker chart

Description automatically generated

Fig. 10. The confidence matrix for the second model, decision tree classification.

Individually increasing or decreasing the values of the various answers helps demonstrate the influence each feature has over the overall prediction. For example, increasing the value of the answer to question 40 to two (2) not only changes the predicted class, but the confidence in the prediction increases to roughly 70 percent.

Chart, bar chart, box and whisker chart

Description automatically generated

Fig. 11. The confidence matrix for the second model, decision tree classification.

No other feature has such drastic impacts on the model’s prediction. The answers to question 29 hold the next-greatest weight over the model’s prediction, and then afterwards questions 5, 17, 18, and 36. Interestingly, answers to question 9 have very little impact on the model’s prediction.

## Runtime Monitoring

I invited John Holmquist and Sam Perry to test my model hosted on Streamlit Cloud. Each tested the model with ten to fifteen different data points.

John immediately made note of the weight question 40 had over the model’s predictions. Humorously, he decided to use his previous short-term relationships as references for his testing data. John said that if the women he had previously dated were his wives, the model accurately predicted how roughly half of the marriages with them would have ended. He was also surprised at the confidence level in the prediction of “will not divorce” for the default answers to the questions, having values of two (2).

Sam attempted to use the model on a mobile device after also checking John’s model on a mobile device, which was likewise hosted on Streamlit Cloud. Sam remarked that Streamlit was not very mobile optimized, specifically the sidebar feature. He also noticed the weakness of the question 9 feature and its influence on the model’s prediction.

# Discussion

## Answers to Research Questions

While model accuracy is very high and thusly the models predict the marital status of a couple, there is no accurate way to test this model without decades of data accumulation. Several factors exist within a marriage that cannot be pinned down explicitly in a machine learning model. While some of these factors could be interpreted by a machine learning model, such as demographics, not all of them can. In theory, the research suggests a machine learning model can be developed to accurately predict if a couple will get divorced or not, yet in application it cannot be concluded.

I would once again like to stress that these models are not to be taken seriously and are more experimentative rather than practical. By no means should these models be used to influence a decision to file for divorce.

However, the results of these models can have some practical application. While they should not be used to encourage a couple’s divorce, they have the potential to be used to evaluate if problems in a marriage are present and encourage a couple to seek therapy for their marriage if they believe that would be helpful.

## Limitations of your research.

In order to practically and effectively test the model, data points would need to be collected from married couples as answers to the Gottman couples therapy questionnaire. Then data would once again need to be collected on if that couple got divorced, or their marriage lasted until one of the spouses died. As one can imagine, the time it would take to evaluate the model’s predictions is not worth the wait.

Aside from this difficulty, there are other problems with the dataset itself. For instance, almost every feature in the original dataset is highly correlated to other features, which adds varying degrees of redundancy to model fitting. Furthermore, properly translating question 40 proved difficult and may still be inaccurate, as Gottman’s website did not always have obvious one-to-one translations. An additional issue with the dataset is the small size, which likely reduces prediction accuracy and when combined with the vast feature set, overfitting is extremely difficult to avoid.

Lastly, finding users to test the model proved difficult for two reasons. First, no students in CS 451 or CS 551 are married, so no accurate data could be accumulated through classmates. The few married couples I asked to try out my model were hesitant to use it out of superstition.

## Lessons Learned

Over a week of working with this dataset and on this research project, I learned a variety of lessons to consider when pursuing data science and machine learning in the future.

As for the dataset itself, I learned to fully investigate the data before coming up with a research question. This dataset had many flaws to it and required an in-depth analysis of sources outside of where it was obtained and the dataset itself. In hindsight, I would have chosen a different project if I knew the complications listed above. However, being able to comb the data and perform extensive feature engineering and selection would not have been possible without a dataset with as many issues.

While I have had experience using Kubernetes (K8s) before, I also learned how to deploy models using Streamlit Cloud, which was much easier than K8s since it is optimized for deploying Streamlit applications.

This project was what I was looking to get out of CS 451. While the lectures focused heavily on the theory, I was hoping to get more practice working with data that the homework assignments could not give alone. I have learned a great deal about the fundamentals of developing machine learning models.

##### Acknowledgment

I would like to thank my fellow students, John “Jack” Holmquist and Sam Perry, for helping me test my model. For testing the model as a couple, I would like to thank my roommates, Jacob Pacheco and Abigail Halloran. I would also like to thank my girlfriend, Madeline Haase, for suggesting that I perform research on a divorce-based dataset.

The dataset used for this research can be found here:

<https://www.kaggle.com/datasets/andrewmvd/divorce-prediction>

The code for this research can be found here:

<https://github.com/zmstiles/divorce-prediction-cs451>

The web-based model can be found here:

<https://zmstiles-divorce-prediction-cs451-app-zskec7.streamlit.app/>

##### References

1. M. A. K. İ. T. &. K. S. Yöntem, "Divorce Prediction Using Correlation Based Feature Selection And Artificial Neural Networks," Nevşehir Hacı Bektaş Veli Üniversitesi SBE Dergisi, vol. 9, no. 1, pp. 259-273, 2019.
2. "Divorce Prediction," 30 July 2020. [Online]. Available: https://www.kaggle.com/datasets/andrewmvd/divorce-prediction.
3. M. Waskom, "seaborn.countplot," seaborn, [Online]. Available: https://seaborn.pydata.org/generated/seaborn.countplot.html. [Accessed 27 November 2022].
4. M. Waskom, "seaborn.heatmap," seaborn, [Online]. Available: https://seaborn.pydata.org/generated/seaborn.heatmap.html. [Accessed 27 November 2022].
5. "ppscore 1.3.0," PyPI, [Online]. Available: https://pypi.org/project/ppscore/. [Accessed 27 November 2022].
6. "The Questionnaire," The Gottman Institute, [Online]. Available: https://checkup.gottman.com/couple/questionnaire/intro. [Accessed 1 December 2022].
7. U. Malik, "Applying Filter Methods in Python for Feature Selection," Stack Abuse, 30 October 2018. [Online]. Available: https://stackabuse.com/applying-filter-methods-in-python-for-feature-selection/. [Accessed 27 November 2022].
8. S. S. Skiena, The Data Science Design Manual, Stony Brook: Springer International Publishing AG, 2017.
9. J. Thorn, "Logistic Regression Explained," 8 February 2020. [Online]. Available: <https://towardsdatascience.com/logistic-regression-explained-9ee73cede081>.
10. "sklearn.linear\_model.LogisticRegression," scikit learn, [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.linear\_model.LogisticRegression.html. [Accessed 27 November 2022].
11. A. Ortner, "Top 10 Binary Classification Algorithms [a Beginner’s Guide]," Towards Data Science, 28 May 2022. [Online]. Available: https://towardsdatascience.com/top-10-binary-classification-algorithms-a-beginners-guide-feeacbd7a3e2. [Accessed 26 November 2022].
12. "sklearn.tree.DecisionTreeClassifier," scikit learn, [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html. [Accessed 27 November 2022].
13. "sklearn.ensemble.RandomForestClassifier," scikit learn, [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. [Accessed 27 November 2022].
14. "sklearn.metrics.confusion\_matrix," scikit learn, [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion\_matrix.html. [Accessed 27 November 2022].
15. "Create an app," Streamlit, [Online]. Available: https://docs.streamlit.io/library/get-started/create-an-app. [Accessed 1 December 2022].
16. "st.slider," Streamlit, [Online]. Available: https://docs.streamlit.io/library/api-reference/widgets/st.slider. [Accessed 1 December 2022].
17. X. Zhu, "wine-example-app," GitHub, 16 October 2022. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion\_matrix.html. [Accessed 27 November 2022].
18. "Deploy an app," Steamlit, [Online]. Available: https://docs.streamlit.io/streamlit-cloud/get-started/deploy-an-app. [Accessed 5 December 2022].
19. "pipreqs 0.4.11," PyPI, [Online]. Available: https://pypi.org/project/pipreqs/. [Accessed 5 December 2022].