HAPPY MOMENT PREDICTION BY THE ANALYSIS OF HAPPY DB

PROJECT REPORT

CMPE 256 - LARGE SCALE ANALYTICS

SUBMITTED TO:

MAGDALINI EIRINAKI

SAN JOSÉ STATE UNIVERSITY

SUBMITTED BY:

TEAM NAME - SPECTRUM

STUDENT NAME	SJSU ID
KEDHARA NETHRA THIRUVURU	011833640
MOHAMMED HAROON SHAREEF	011815479
LAVANYA KANDUKURI	011833588

GITHUB: https://github.com/mohammedharoon/CMPE_256_HappyDB_Analysis

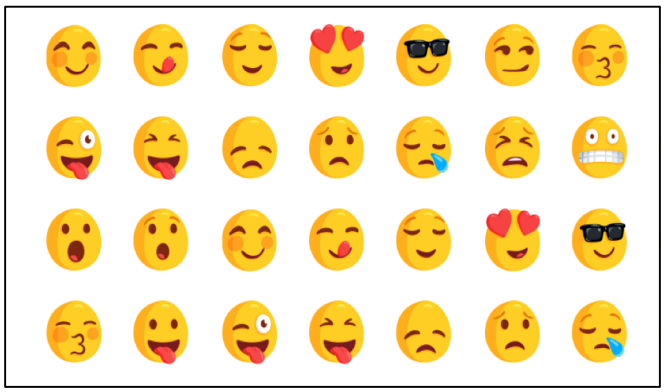
DATASET: https://www.kaggle.com/ritresearch/happydb

1. Introduction:

1.1. Happy DB:

Happy DB is a collection of 100,000+ happy experiences as documented by individuals. This information is gathered from Amazon's Mechanical Turk over a period of four months. The goal of this dataset is to advance the understanding of causes of happiness through text based reflection.

1.2. Motivation:



Who doesn't want to be happy? With our busy lives in modern days, we often find it difficult to manage stress, handle responsibilities, adapt to new situations along with maintaining a happy state of mind. Options like self-help books, blogs and stress-handling apps may not always be a solution. This might increase mental burden on us. Therefore, it is necessary to refocus on the everyday things that make us happy. This is what we are trying to do with Happy DB.

1.3. Objective:

Our objective is to find the correlation between demographic information of a person and the category of happiness by using classification algorithms and ultimately build a predictive model based on the correlation determined.

2. System Design and Implementation details:

2.1. Algorithm(s) considered/selected:

Since our use-case is related to predictive-modelling and our target variable is categorical, we have used classification algorithms for building our model.

Algorithms selected: Gaussian Naïve Bayes, Decision Trees Classifier, Random Forest Classifier, Support Vector Machine, K-Nearest Neighbor.

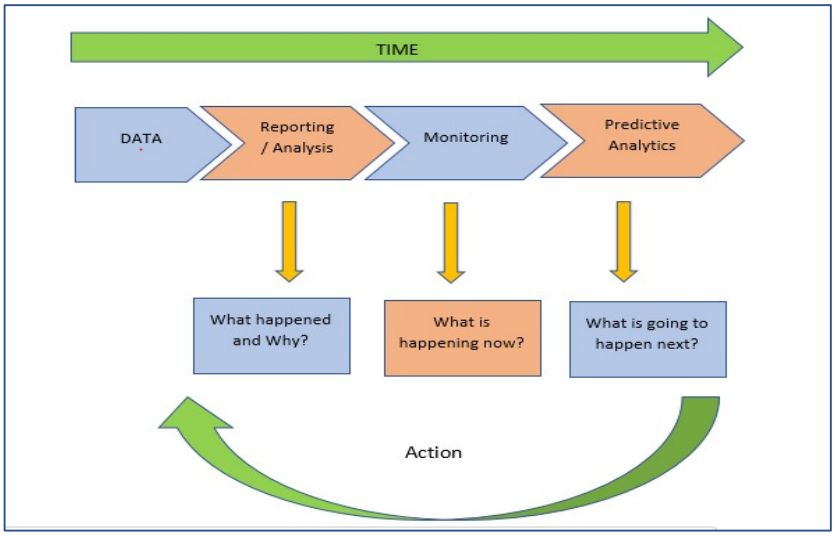


Figure-1: Predictive Analysis

Steps Performed:

- We first started with basic analysis of the dataset by using data exploration techniques to get some insight into the dataset.
- We observed that some features are categorical values while some are numerical values. Some features had a lot of missing values.
- We considered only the two files "**cleaned_hm.csv**" and "**demographic.csv**" from the dataset that are required for our analysis.
- We then cleaned these files by dropping the columns/features that are not required for predictive analysis. The columns like **wid**, **hmid**, **reflection_period**, **original_hm**, etc. are dropped.
- After dropping features, we merged both the files (datasets) and rearranged the columns (dependent and independent attributes) for better understanding. We have replaced features (**categorical** features) which had NaN values with corresponding mode value. We spent a lot of time in analyzing the dataset, preprocessing the data by removing unwanted features, imputing missing values and removing outliers.
- We then converted categorical variables to dummy variables and from every group of dummies generated, we dropped one column to prevent dummy variable trap. For example, categorical variable **gender** had three categories **male**, **female** and **others**. After dummy variables for each of these categories is created, we dropped column **others** to prevent the dummy variable trap.
- We tried to find relationship/correlation between different attributes by generating a heat map.
- We worked on different classifier algorithms to compare the results of each and to understand which would best serve our purpose of prediction. We evaluated these models by considering different metrics like precision, recall and accuracy.

2.2. Technologies & Tools used:

We have used the following tools and technologies for building our model:
Anaconda Jupyter Notebook, Spyder (Python 3.6), Python, sklearn, matplotlib

With Spyder, we could run the entire code at once which came in handy while experimenting during the initial stages and Jupyter Notebook provided better visualization of relationships between attributes. We used sklearn for preprocessing, building our model and cross-validation. We used matplotlib for data visualization.

2.3. System design/architecture/data flow/workflows as applicable:

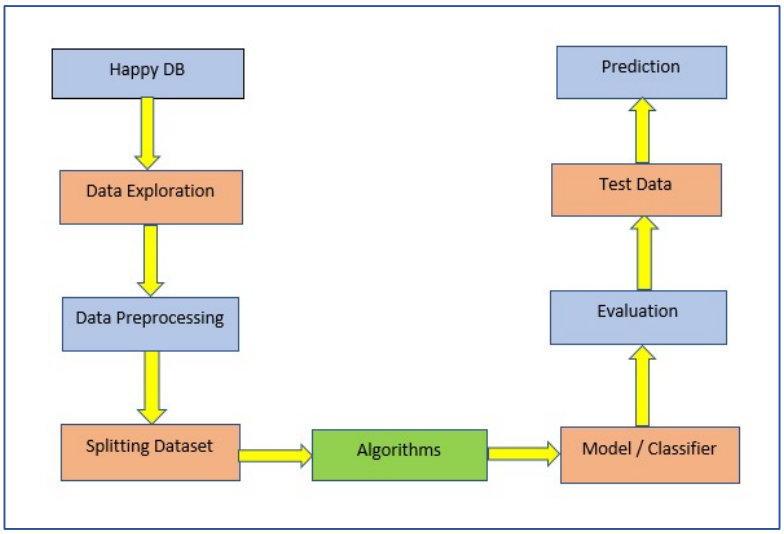


Figure-2: System Architecture Diagram

After selecting the dataset, we first started with data exploration to get a clear understanding of the dataset. In the preprocessing phase, we removed unwanted features and cleaned the data by imputing missing values and by removing outliers. We then split the dataset into train and test and trained our model using suitable algorithms. We evaluated the model and then predicted the happy moment category for the given test data.

2.4. Visualization/UI/ GUI/screenshots:

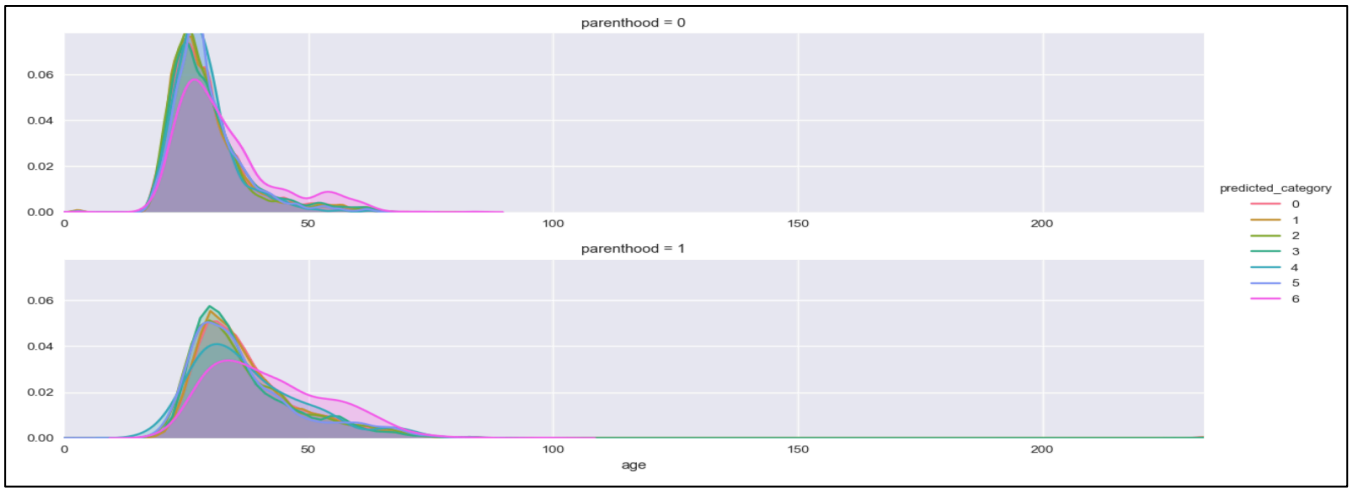


Figure-3: Distribution of age based on parenthood who contributed to Happy moment categories

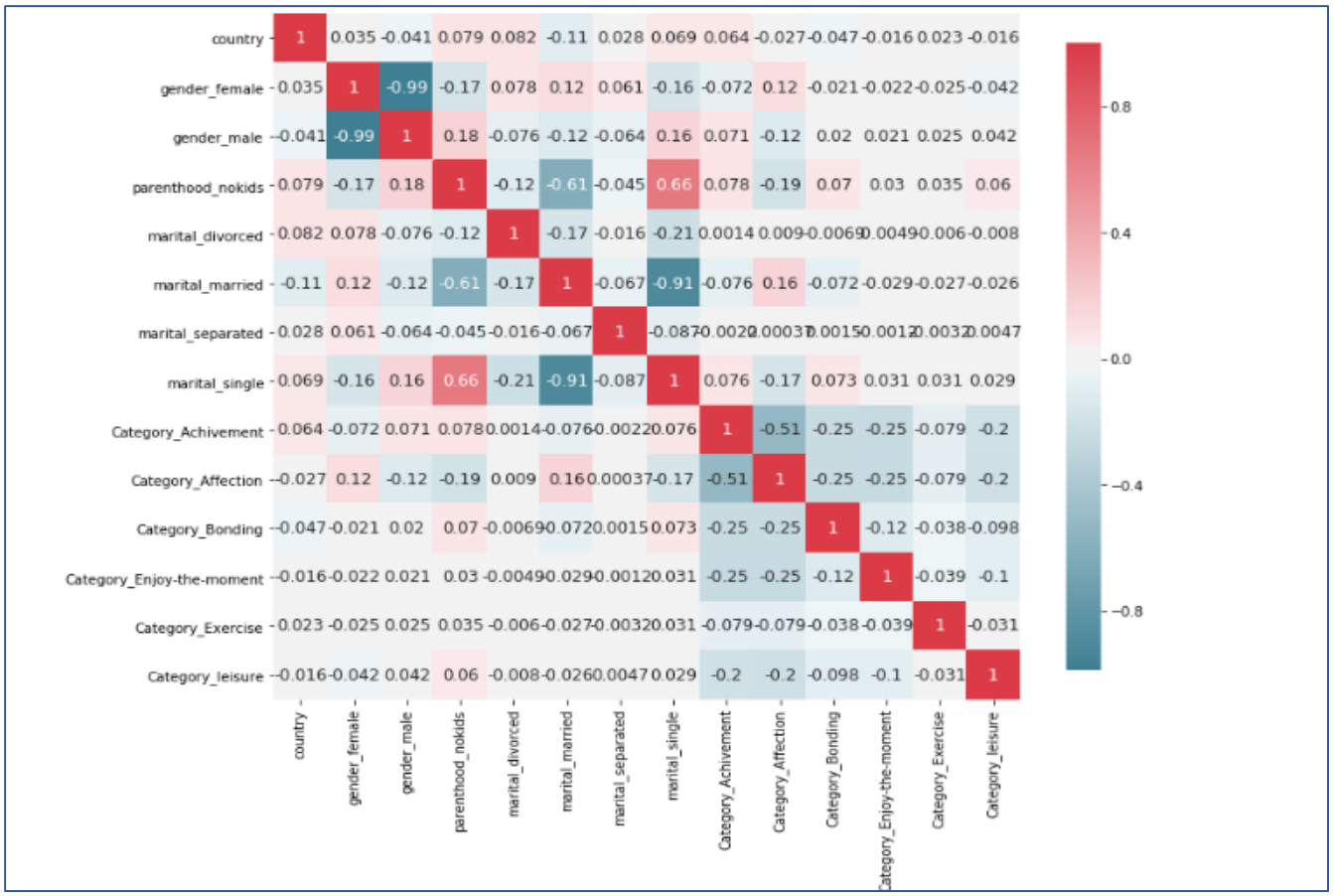


Figure-4: Correlation Map

Figure-4 shows relationship between various attributes with respect to target variable, predicted_category. We could see from the map that parenthood has highest correlation with target variable.

3. Experiments / Proof of concept evaluation:

3.1. Dataset(s) used:

- **Dataset Source:** https://www.kaggle.com/ritresearch/happydb
- **Dataset Name:** Happy DB
- **Dataset Size:** 22.5 MB
- **Dataset Files:** cleaned_hm.csv: 22.5 MB, demographic.csv: 265 KB
- **Number of records:** 100,000+
- **Preprocessing:** Data cleaning, removing outliers, imputing missing values.

Dataset Description:

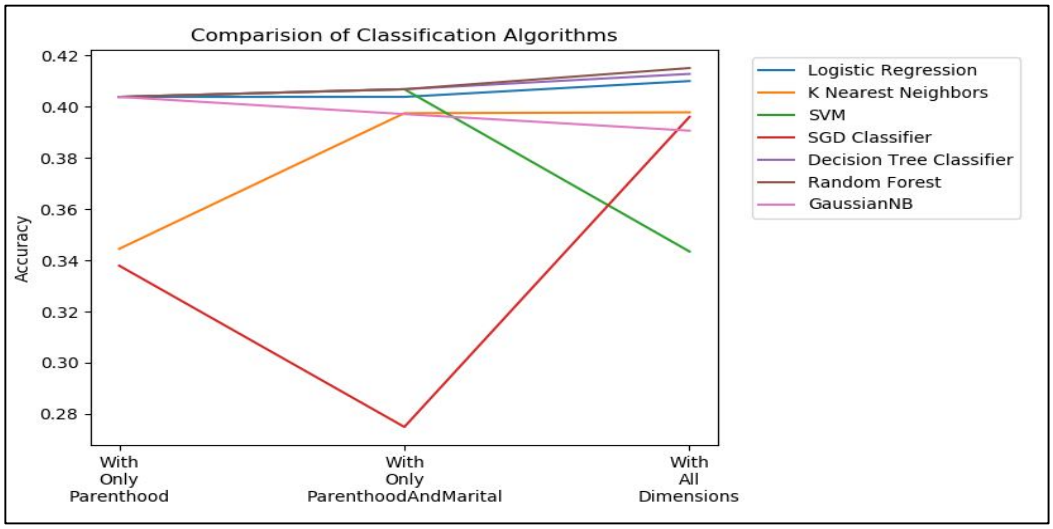
- Happy DB is a collection of various files of which we are using only cleaned_hm.csv and demographic.csv.
- cleaned_hm.csv has dimensions 100535*9 and demographic.csv has dimensions 10844*6. The feature that we are considering for our analysis in cleaned_hm.csv is “predicted_category”. We are considering features age, country, parenthood, marital, gender from demographic.csv.

- cleaned_hm.csv contains cleaned happy moments and demographic.csv contains demographic details of a person.

3.2. Data preprocessing decisions:

- We dropped unnecessary features like wid, hmid from cleaned_hm.csv and demographic.csv. Our target variable is predicted_category.
- We merged two datasets after dropping columns. We arranged features into depenedent and independent variables for clarity. We performed one hot encoding for all the categorical variables.
- We converted categorical variables to dummy variables by using get_dummies and also dropped one column to avoid dummy variable trap.
- We imputed missing values and removed outliers from age.
- We replaced NaN values with mean for age and with mode for categorical values.

3.3. Graphs showing different parameters/algorithms evaluated in a comparative manner:



3.4. Analysis of results:

- With the help of results and the evaluation graphs we can say that Logistic Regression has good accuracy results but the algorithm failed to predict most of the target categories.
- Tree based algorithms like Decision Tree and Random Forest performs the best in all three conditions, i.e with only Parenthood variable, Parenthood along with Marital and also with all the variables.
- It is also seen that Parenthood variable alone was able to give an accuracy of more than 40% for most of the algorithms except KNN and SGD.
- We can see that adding more dimensions does not improve our accuracy score. With 8 more variables as independent variables, we were able to improve the accuracy by only 1%.

4. Discussion & Conclusions:

4.1. Decisions made:

- We decided to work on a relatively new dataset and so we chose Happy DB which made our research more interesting.
- During data preprocessing phase, we decided to drop the columns that are not required for analysis and merged the files. We removed outliers in age like strings and values higher than 100(which is unusual) and imputed missing values with mean. Initially when we discovered that there were outliers in age, we decided to go with Robust Scaler. But with further analysis, we found that there were only 150 rows that were outliers and decided to drop those. We used Standard Scalar after dropping those outliers. For other features which were categorical variables, we imputed missing values with mode, which gave us better results.
- We decided not to create dummies for the feature "**country**" as we found that there is low correlation between country and the target variable(**predicted_category**).
- We converted categorical values to dummy variables using get_dummies function. This gave us numerical attributes which are suitable for working with various algorithms. We decided to find correlation between each independent variable and dependent variable (**predicted_category**) for better understanding of data. We observed that **parenthood** has maximum influence on the target variable.
- We decided to split the dataset in the ratio of 80% and 20% for training and testing for better results using cross validation measures.
- We considered different evaluation metrics like precision and recall to evaluate our model.
- We decided to compare various algorithms to understand which one best served our purpose.

4.2. Difficulties faced:

- We had difficulty in handling mixed data type values in features like **age**. It had mixed data types of Int, String, Float with missing values and outliers. We decided to format the data to make it consistent.
- We faced difficulty in transforming the feature "**country**" from categorical to numerical variable. As this feature had nearly 100 categories, converting these to dummy variables will introduce more than 100 features and so we decided not to create dummies for it. We also found out that there is low correlation between country and target variable(**predicted_category**) and thus we have decided not to create dummy columns for country.

4.3. Things that worked:

- Data preprocessing gave us much better results than the raw data. All the measures like data cleaning by removing unwanted features, outliers and imputing missing values improved the accuracy.
- Our decision to convert categorical to numerical values also helped in giving better results.

4.4. Things that didn't work well:

- Linear algorithms like SGD Classifier and Logistic Regression didn't work well in giving satisfactory results as anticipated. Logistic Regression only could predict two values out of a total of six values in predicted_category.

- Initially we thought the features like age and country will contribute towards target variable "predicted_category" which wasn't the case. Happy moment category is not dependent on demographic data is what we found out.

4.5. Conclusion:

- Through data exploration and preprocessing of Happy DB, we better understood the data we are dealing with and how features are interdependent. We could apply concepts learnt in the class which was a very good experience.
- Even after applying various algorithms, we couldn't predict the happy moment category with a good accuracy which shows that this category is independent of demographic information and this is counter-intuitive!
- Future scope is to perform sentiment analysis to determine happiness by specific types of experiences.

5. Project Plan / Task Distribution (1/2 page):

5.1. Who was assigned to what task?

Each one of us was assigned the task of data exploration, data preprocessing and data analysis individually. We decided to work on different classifier algorithms and divided the task (2 algorithms each).

5.2. Who ended up doing what task?

Analysis of Happy DB for Happy Moment Prediction	
Task	**Responsibility**
Dataset selection	All
Data Exploration	All
Data Preprocessing	All
Data Visualization	All
Research on Classification Algorithms	All
Logistic Regression	Kedhara Nethra
Support Vector Machine	Mohammed
Decision Tree Classifier	Lavanya
K-nearest neighbors	Mohammed
Random Forest Classifier	Kedhara Nethra
Naïve Bayes	Lavanya
Documentation	All

References:

- Akari Asai, Sara Evensen, Behzad Golshan, Alon Halevy, Vivian Li, Andrei Lopatenko, Daniela Stepanov, Yoshihiko Suhara, Wang-Chiew Tan, Yinzhan Xu, ``HappyDB: A Corpus of 100,000 Crowdsourced Happy Moments'', LREC '18, May 2018.

- https://github.com/rit-public/HappyDB
- https://www.kaggle.com/ritresearch/happydb
- http://scikit-learn.org/stable/supervised_learning.html
- http://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html#multiclass-settings
- https://pandas.pydata.org/pandas-docs/stable/visualization.html