Data Completion and Interpolation

for

CS 536 : Final Project

Version 1.0

Prepared by Fatima AlSaadeh
Supervised by Charles Cowan

Rutgers University

December 17, 2020

Contents

1 Introduction

1.1 Purpose

This project is designed to predict and interpolate missing features in a dataset represented by a collection of categorical and numerical columns from the features that are present using classification and regression models. [1]

1.2 Project Scope

The purpose of this project is to find the best models to represent missing data by choosing a dataset, explore it, point out its categorical, numerical, and missing features, implement these models, fit, train, and evaluate it, then create a new dataset from the original one with the new predicted values. Figure 1.1

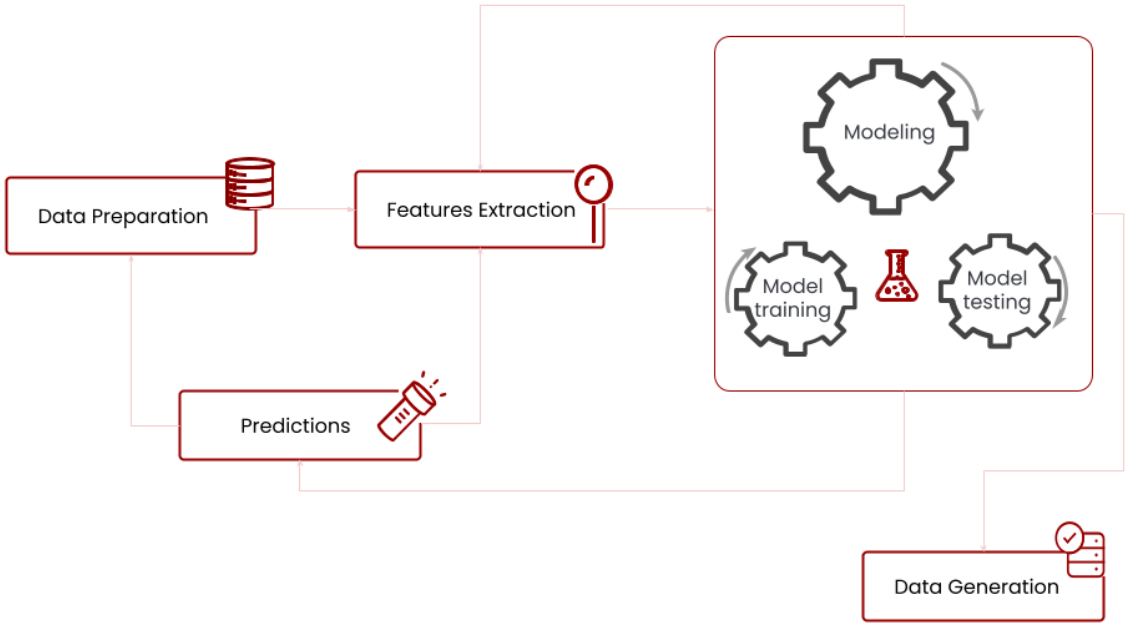

Figure 1.1: Project Architecture

2 Overall Description

2.1 Dataset Description and Exploration

2.1.1 Data Description

In this project, we are using the "Young People Survey" dataset from Kaggle. [2]
As mentioned in its description, the goal of this dataset is to explore the preferences, interests, habits, opinions, and fears of young people. It consists of 150 columns, 4 numerical and 146 categorical and 1010 records. The records represent adults' answers to survey questions.
The categorical columns are representing answers in a form of text categories or ranking which is a value from the classes [1,2,3,4,5] where 1 represent not interested/Strongly disagree/ Very afraid of and 5 represents very interested/ strongly agree/not afraid at all, please see in Appendix A, Table 7.1 questions and answers snippet from the dataset.

2.1.2 Data Exploration

- Figure 2.1 below show us the number of records in the dataset consist of null values, we will be splitting the data to a dataset with null values "responses_na" - 336 records and one for the rest of the records "responses" - 674 records. We will be using the responses dataset for the analysis, modeling, training and validation and at the end of the process we will use responses_na to make predictions to fill the dataset.

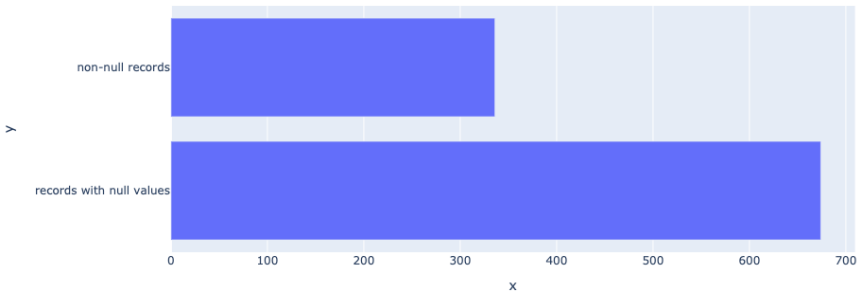

Figure 2.1: Number of records consist of null value

- Figure 2.2 below shows the count of missing values per feature, Height and Weight recorded the highest count of missing values and we found that 50% of the time where Height was missing Weight was missing too.

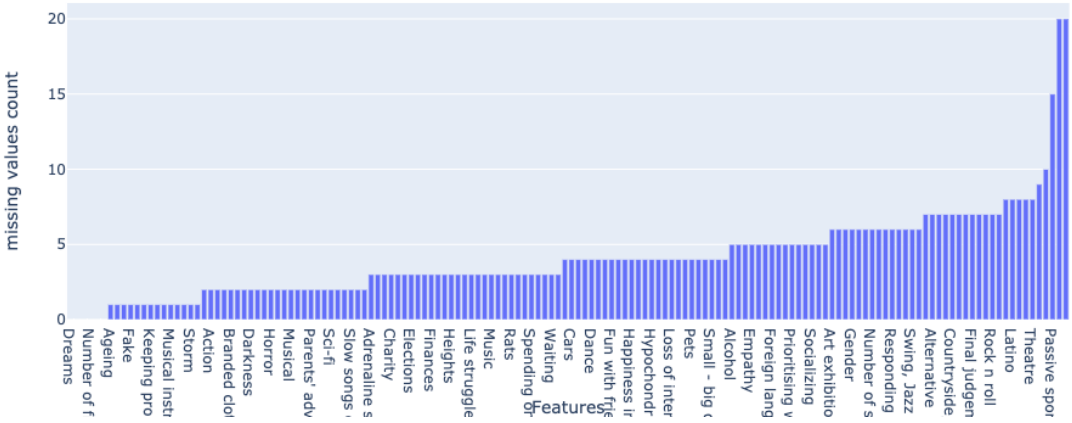

Figure 2.2: Count of missing values per feature

- Figure 2.3 shows further data exploration to learn more about the data we have and an overview about the participants, we found that the participants ages are normally distributed and around 66% of the participant are in the age range [19,23], the zero value in the figure represents the null values for the ages features occured 7 times for both genders, 59.6% of them are female and 62.9% in secondary school and 20.7% are in college pursuing a bachelor degree while 74.4% of the participants use the internet few hours a day.

- Figure 2.4 shows the correlation between the topics participants are interested in based on their answers, we found the participants who are interested in biology are also interested in chemistry and medicine, we anticipate that the correlated columns in the data will help us predict missing features.

$$\frac{\sum_{i=1}^{n}(x_i - \bar{x}) * (y_i - \bar{y}))}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 * \sum_{n=1}^{n}(y_i - \bar{y})^2}}$$

Figure 2.3: Overview about the participants

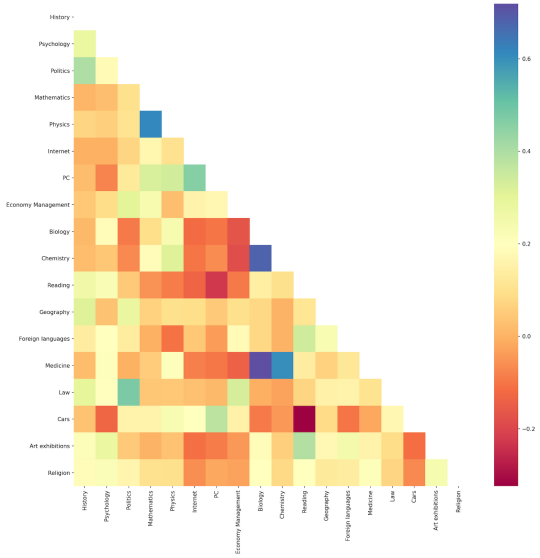

Figure 2.4: Correlation between topics participants are interested in

2.2 Data Preparation

2.2.1 Data Preparation and Splitting

- Data Preparation: The data provided was clean and ready for analysis, it consists of 33% of null values and by feature the maximum number of null values existed by 1.98% in height and weight features. The data records containing null values was saved in a different dataset to be used for data generation at the end of this process and no features were dropped as its all appears to be significant.
  Categorical column with textual classes were converted using one-hot encoding and this increased the number of columns from 150 to 173:
  - Example of categorical column was encoded in this step : Smoking habits column with values [Never smoked - Tried smoking - Former smoker - Current smoker] to [Smoking_current smoker, Smoking_former smoker, Smoking_never smoked, Smoking_tried smoking] features.
  - Categorical columns with values [1,2,3,4,5] were treated as integer values at this stage.

  It is important to keep in mind for now that the questions divided under the different topics and this information can help us know the dependent features, these topics are:
  - Music Preferences.
  - Movie Preferences.
  - Hobbies and Interests.
  - Phobias.
  - Health Habits.
  - Personality Traits.
  - Views On Life and Opinions.
  - spending Habits.
  - Demographics.
- Data Splitting: at this point the data was split 80% for training and 20% predictions for data generation. the training data is split to 80% training and 20% for validation.

3 Modeling - Describe your Model

In this section we are trying to answer the following question:
• **Describe your Model: What approach did you take? What design choices did you make, and why? How did you represent the data? How can you evaluate your model for goodness of fit? Did you make an effort to identify and exclude irrelevant variables? How did you handle missing data?**

3.1 Model Selection

As described in the data exploration section, the dataset contains categorical and numerical data, in order to select the best model, We implemented and train different classification and regression models, evaluate the results then selected the model that gave the best results based on the evaluation metrics will describe in the upcoming sections.

3.1.1 Data Representation

The categorical features are represented using one-hot encoding with values 0, 1 and the numerical features were normalized by dividing each value by the maximum value appeared for that feature to get values between 0 and 1 so the model give us stable results because for numerical features we have weight and height and that the model shouldn't treat as same value and this will give us values between 0 and 1 Normalization formula used for column X:

$$Normalized\_X = \frac{X - min(X)}{max(X) - min(X)}$$

3.1.2 Handling Missing Data and Design Approach

Our design approach to handle missing data in this project is as following:

- Start with the original data, encode the categorical features and normalize the numerical columns.
- Split the original data to : dataset A with records contains no missing values for any feature and dataset B with records contains all the records with missing data for any feature
- For dataset A split the data to 80%, 20% training and testing dataset splits. For training dataset:

  - Start with X: features 1,...n-1 and Y: feature n In case of numerical features we chose set of X where X and Y are correlated if y isn't correlated with any other column we pass all features - Y
  - Fit the model on X, Y from the training split
  - Evaluate the model on the testing split comparing true Y with the predicted Y
  - Repeat until all features are predicted as Y and evaluated
- Adjust the model hyperparameters until the evaluation of the model is successful (Low error rate).
- Predict the missing data on dataset B:
  - start with the feature with least missing value as target feature Y and select all the records where Y has a missing value.
  - start with X: features 1,...n-1 and Y: feature n In case of numerical features we chose set of X where X and Y are correlated if y isn't correlated with any other column we pass all features - Y
  - for each record, if it contains another ,missing feature other than the picked Y we move it from the predictors to be the target.
  - Fit the model on the dataset A with the same features selections we have until now to get the weights and parameters on dataset with no missing data.
  - Use the trained model to predict Y [one or multiple features]
  - Fill the predicted value back in dataset B to use it predicting features in the next step.
  - Repeat until there is no missing data
- Return full dataset with no missing values

In every step we made sure to take care of normalizing, de-normalizing, encoding and decoding.
• **Describe your Model: What approach did you take?**

3.2 Classification Model

3.2.1 Basic Mode Model

For categorical features predictions, We started by applying The basic classification completion agent mentioned in the project description:

- For a data point x missing feature i:
  - fill xi with the mode of feature i from the data set.
- Iterate over each feature missing i, completing x.

3.2.2 Neural Network Model

The neural network We built with single hidden layer, forward and backward propagation, figure 5.1 shows the neural network model architecture with description:

- Input X with n data points and m features and Y one or more target categorical feature to predict with n data points.
- Start with epochs = n and learning rate = L and number of nodes in the hidden layers = h.
- Initialize randomly the first layer weights w_input with size [h,m], and b_input biases with size [h,L].
- Initialize randomly the output layer weights w_output with size [1,m], and bias b_output with size [1,1].
- iterate n times to fit (X, Y):
  - Use forward propagation:

$$S1 = w\_input \cdot X^T + b\_input \tag{3.1}$$

$$Relu(Z) = maximum(0, X) \tag{3.2}$$

$$Hidden\_Layer\_out = Relu(S1) \tag{3.3}$$

$$Sigmoid(Z) = \frac{1}{1 + e^{-Z}} \tag{3.4}$$

$$S2 = output\_w \cdot Hidden\_Layer\_out + b\_output \tag{3.5}$$

$$output = Sigmoid(S2) \tag{3.6}$$

  - Use backward propagation:

$$Sigmoid\_Derivative(Z) = Sigmoid(Z) * (1 - Sigmoid(Z)) \tag{3.7}$$

$$Relu\_Derivative(Z) = Z > 0 \tag{3.8}$$

$$dZ2 = (output - Y^T) * Sigmoid\_Derivative(output) \tag{3.9}$$

$$d\_w\_output = \frac{1}{m} * dZ2 \cdot Hidden\_Layer\_out^T \tag{3.10}$$

$$d_b\_output = \frac{1}{m} * sum(dZ2) \tag{3.11}$$

$$dZ1 = (w\_output^T \cdot sZ2) * Relu\_Derivative(Hidden\_Layer\_out) \tag{3.12}$$

$$d\_w\_input = \frac{1}{m} * dZ1 \cdot X \tag{3.13}$$

$$d\_b\_output = \frac{1}{m} * sum(dZ1) \tag{3.14}$$

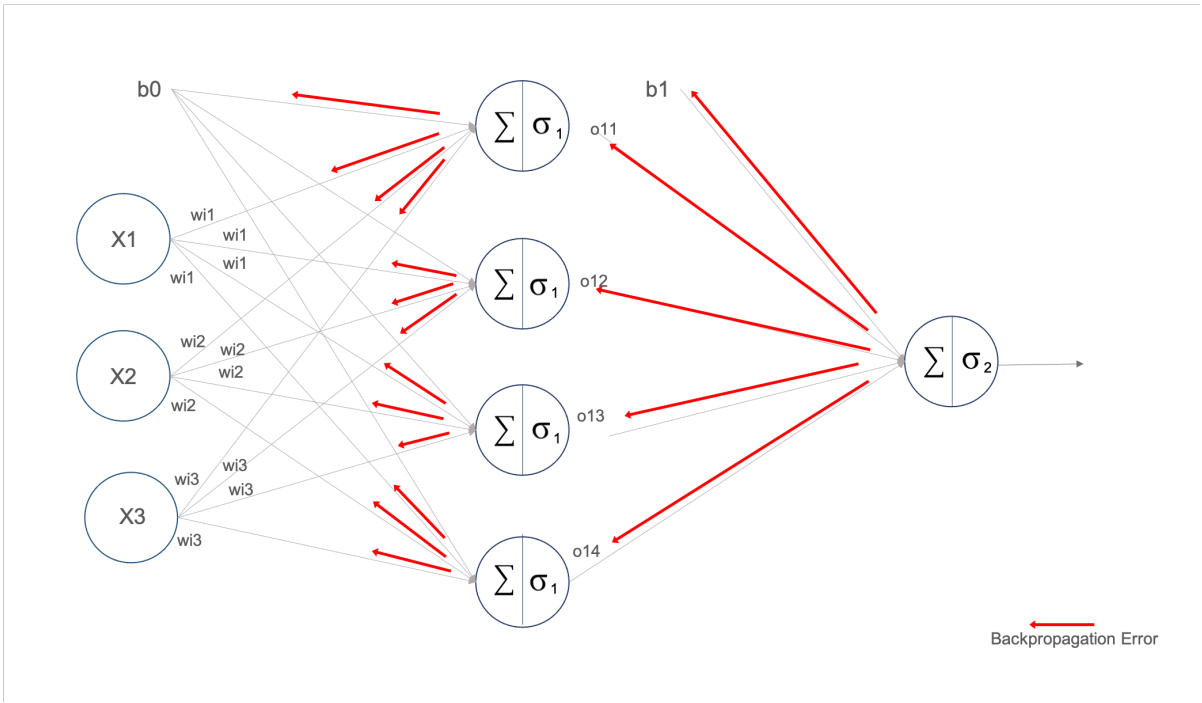

Figure 3.1: Neural Network Model Architecture

3.3 Regression Model

3.3.1 Basic Mean Model

For numerical features predictions, We started by applying The basic regression completion agent mentioned in the project description:

- For a data point x missing feature i:
  - fill xi with the mean of feature i from the data set.
- Iterate over each feature missing i, completing x.

3.3.2 Linear Regression Model

We built lasso and ridge regression models and compared them.

- Lasso Regression:
  - Input X with n data points and m features and Y one or more target features to predict with n data points.
  - Start with n iterations and lambda L.
  - Initialize weights w with size [1,m] with 0.01.
  - Iterate with n iterations

- Update the weights based on the lectures formula:

$$\hat{w}_i = \begin{cases} w_i - \lambda/2 & \text{if } w_i > \lambda/2 \\ w_i + \lambda/2 & \text{if } w_i < -\lambda/2 \\ 0 & \text{if } -\lambda/2 < w_i < \lambda/2. \end{cases}$$

- Repeated these steps to get the lambda that minimize the mse.

- Ridge Regression:
  - Input X with n data points and m features and Y with n data points.
  - Start with n iterations and lambda L.
  - Fit (X,Y) to find the weights values the solve the ridge algorithm:

$$\underline{\hat{w}} = \left[X^{\mathrm{T}}X + \lambda I\right]^{-1} X^{\mathrm{T}}X\underline{y}$$

3.4 Identifying Irrelevant Variables

For identifying irrelevant variables, We used the correlation approach by calculating the correlation between each feature and the target feature to see how they are related, select for each target the features with correlation value more than .5 and tried .3 threshold as well. This approach was applied on numerical features. Figure 2.4 shows selection of this heatmap and the JSON record below shows sample of these correlated columns. Lasso regression also helped us identifying the irrelevant features by pruning.

```
{
   "Rock":[
      "Metal or Hardrock",
      "Punk"
   ],
   "Metal or Hardrock":[
      "Rock",
      "Punk"
   ],
   "Punk":[
      "Rock",
      "Metal or Hardrock"
   ],
   "Biology":[
      "Chemistry",
      "Medicine"
   ],
   "Chemistry":[
      "Biology",
      "Medicine"
   ],
   "Medicine":[
      "Biology",
      "Chemistry"
   ],
   "Shopping":[
      "Shopping centres",
      "Spending on looks"
   ],
   "God":[
      "Religion",
      "Final judgement"
   ],
   "Height":[
      "Weight",
      "Gender_male"
   ],
   "Weight":[
      "Height",
      "Gender_male"
   ],
   "Gender_male":[
      "Height",
      "Weight"
   ]
}
```


Listing 1: Sample of Correlated Columns

4 Describe your Training Algorithm -Describe your Model Validation

4.1 Evaluation Question Answered

This section is to answer the following questions, but further explanation, analysis and plots are in the upcoming sections.
Given your model, how did you fit it to the data? What design choices did you make, and why? Were you able to train over all features? What kinds of computational tradeoffs did you face, and how did you settle them? How did you try to avoid overfitting the data? How did you handle the modest (in ML terms) size of the data set?

4.1.1 Classification Models

Basic Mode Model

In this basic model approach the model was able to replace every missing categorical variable with the mode of its column add these calculated values to the dataset, the approach was start with a feature, do the calculation, add back to the dataset, move to the next feature. This basic model was able to fit all the features with no problems and no tradeoff, We further used it as a benchmark to evaluate the performance of other models.

Neural Network Model

Given your model, how did you fit it to the data? What design choices did you make, and why? Were you able to train over all features? What kinds of computational tradeoffs did you face, and how did you settle them?
In neural network model, we started with fitting and testing the data on dataset split with no missing data, this split was further split to training and testing datasets, one for fitting the data and one for evaluating the performance and calculating the accuracy when we predict on it, The approach was starting with a categorical feature as target Y, and setting all the remaining features as predictors X, the predictors were all the numerical features only, fit, predict the output of the prediction was be a probability for each class in the categorical variable, we then selected the class index with highest probability to be the predicted value. This design choice was applied and tested over multiple hyperparameters to chose the ones that improve the performance of the models, we tested the model over number of iterations [350,1500,2500,5500], the learning rate

was tested over values .1,.5, .01,.001, .005. At the end we used the hyperparameters that makes the mode perform well on most of the features. This approach was to predict the categorical features in the dataset.
How did you try to avoid overfitting the data? How did you handle the modest (in ML terms) size of the data set? To fit and evaluate the model over all the categorical features avoiding overfitting, splitting the data into .8/.2 training and testing datasets to give the model a chance to predict on unseen data was one of the approaches, also keeping the portion with all missing data to the end when we generate the full dataset to predict the final results after training and testing. In addition to that We tuned and choose the hyperparameters carefully, monitoring the accuracy and loss plots to see when I'm using low or high learning rate and adjust accordingly.

4.1.2 Regression Models

Given your model, how did you fit it to the data? What design choices did you make, and why? Were you able to train over all features? What kinds of computational tradeoffs did you face, and how did you settle them? How did you try to avoid overfitting the data? How did you handle the modest (in ML terms) size of the data set?

4.1.3 Basic Mean Model

In this basic model approach the model was able to replace every missing numerical variable with the mean of its column add these calculated values to the dataset, the approach was start with a feature, do the calculation, add back to the dataset, move to the next feature. This basic model was able to fit all the features with no problems and no tradeoff, We further used it as a benchmark to evaluate the performance of other models.

Linear Regression Model

Given your model, how did you fit it to the data? What design choices did you make, and why? Were you able to train over all features? What kinds of computational tradeoffs did you face, and how did you settle them?
In linear regresstion model, we started with fitting and testing the data on dataset split with no missing data, this split was further split to training and testing datasets, one for fitting the data and one for evaluating the performance and calculating the MSE when we predict on it, The approach was starting with a numerical feature as target Y, and setting all the remaining features as predictors X, both categorical and numerical, fit, predict the output of the prediction was a numerical value normalized which we de-normalize to get the original value. This design choice was applied and tested over multiple hyperparameters to chose the ones that improve the performance of the lasso model, we tested the model over number of iterations [100,350,1000], the lambda was tested over values in the range [0, 20] with .05 step. The ridge model was tested over lambdas in the range [10, 100] with .05 step. At the end we used the hyperparameters

that makes the mode perform well on most of the features. This approach was to predict the numerical features in the dataset.
How did you try to avoid overfitting the data? How did you handle the modest (in ML terms) size of the data set? To fit and evaluate the model over all the categorical features avoiding overfitting, splitting the data into .8/.2 training and testing datasets to give the model a chance to predict on unseen data was one of the approaches, also keeping the portion with all missing data to the end when we generate the full dataset to predict the final results after training and testing. In addition to that We tuned and choose the hyperparameters carefully, monitoring the MSE plots to see when I'm using low or high lambda and adjust accordingly.

4.2 Evaluation Methods

- MSE and RMSE (Mean Squared Error):

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

  This metric is used to measure the performance of regression models

- Accuracy

$$Accuracy = \frac{|y_i! = \hat{y}_i|}{n}$$

  This metric is used to measure the performance of classification models

- Cross-entropy loss:

$$CE = -1 * \frac{\sum_{i=1}^{n}(y_i * \log \hat{y}_i) + (1 - y_i * \log 1 - \hat{y}_i)}{n}$$

  This metric is used to measure the performance of classification models (Loss).

4.3 Training and Validation Approach and Results

4.3.1 Classification Training and Validation

- Training and Evaluation Approach: For training and evaluation We used the 80%, 20% split approach, fit the model on the training dataset, predict on the evaluation dataset using the following algorithm for the neural network we built :

$$S1 = w\_input \cdot X^T + b\_input \tag{4.1}$$

$$Relu(Z) = maximum(0, X) \tag{4.2}$$

$$Hidden\_Layer\_out = Relu(S1) \tag{4.3}$$

$$Sigmoid(Z) = \frac{1}{1 + e^{-Z}} \tag{4.4}$$

$$S2 = output\_w \cdot Hidden\_Layer\_out + b\_output \tag{4.5}$$

$$output = Sigmoid(S2) \tag{4.6}$$

$$prediction = output^T \tag{4.7}$$

- After that We calculated the loss and accuracy using the metric mentioned in the evaluation methods section.
- These steps were repeated until we evaluate the model on all the category features.

4.3.2 Regression Training and Validation

- Training and Evaluation Approach: For training and evaluation We used the 80%, 20% split approach, fit the model on the training dataset, predict on the evaluation dataset using the following algorithm for the linear regression model we built :

$$predict = X \cdot W \tag{4.8}$$

- After that We calculated the mean squared error using the metric mentioned in the evaluation methods section.
- These steps were repeated until we evaluate the model on all the numerical features.

5 Evaluate your Model

Where is your model particularly successful, where does it lack? Does it need a certain amount of features in order to interpolate well? Are there some features it is really good at predicting and some it is really poor at predicting? Why do you think that is? How does your model stack up against the basic completion agent?

5.0.1 Classification Evaluation

We used the two models described in the model section to compare the results between them, the following steps were followed for the evaluation :

- For the training dataset set x of features to predict Y, with n values, Y here is the feature with the one hot encoding , for example for smoking feature Y = [0,1,0,0] for classes Never smoked - Tried smoking - Former smoker - Current smoker the Y here represent the Tried smoking category.
- fit the data on training X, Y and calculate the loss.
- predict on the testing dataset X.
- Evaluate using the accuracy metric.

Figure 5.1 shows the accuracy results for each categorical column predicted using all numerical as predictors and the neural network model with 550 neurons ,3000 iteration and learning rate of .001. These hyperparameters were pick after many steps of using different parameters and these gave the best results.
Comparing the neural network model results to the basic agent mode model, the results are shown in the figure 5.2. Using the neural network model 7 of the categorical features out of 11 were classified with more than 50% accuracy Left-Right handed record the highest accuracy here.
Comparing the neural network model results running on the learning rate .001 and .1 model, the results are shown in the figure **??**. We found that the model is preforming better in term of accuracy with learning rate 0.1 for punctuality, Laying and Internet usage features. The hyperparameters were chosen at the end after multiple stages of testing are in the table 5.1, and we picked it after monitoring the accuracy and loss plot for the neural network. Figure 5.4 compare the loss plots between all categorical features when the learning rate is .001. We picked the hyperparameters that made the model perform well on most of the features, while 5.5 shows this comparison on the learning rate .1. We found that the loss function is gradually decreasing. the learning rate .001 is performing better reducing the loss for most of the features.

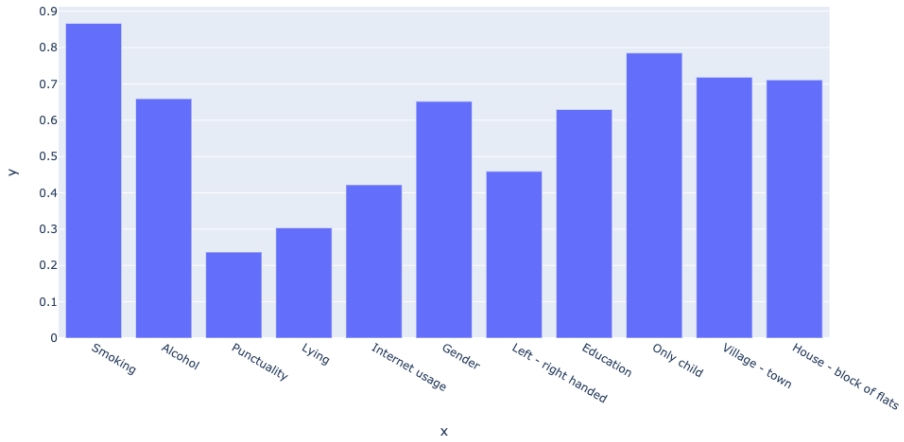

Figure 5.1: Neural Network Model Accuracy Per Categorical Column

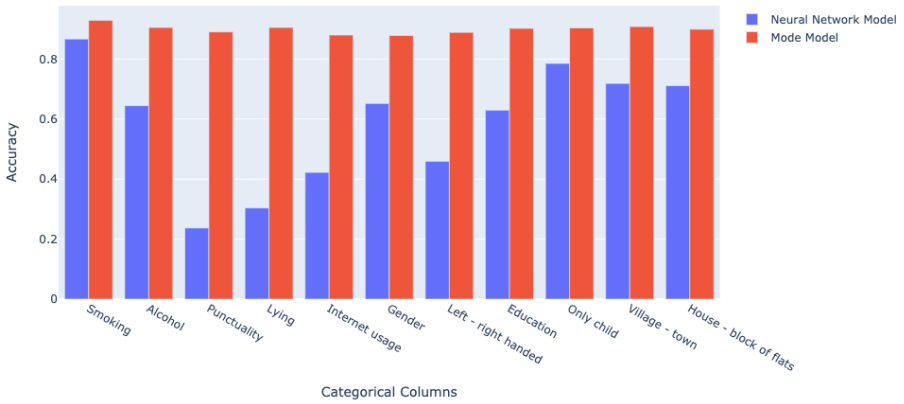

Figure 5.2: Accuracy Comparison between Classification Models

Hyperparameter	Value
number of iterations	1550
learning rate	.001 and .1
number of hidden layers	250

Table 5.1: Neural Network Models Hyperparameter

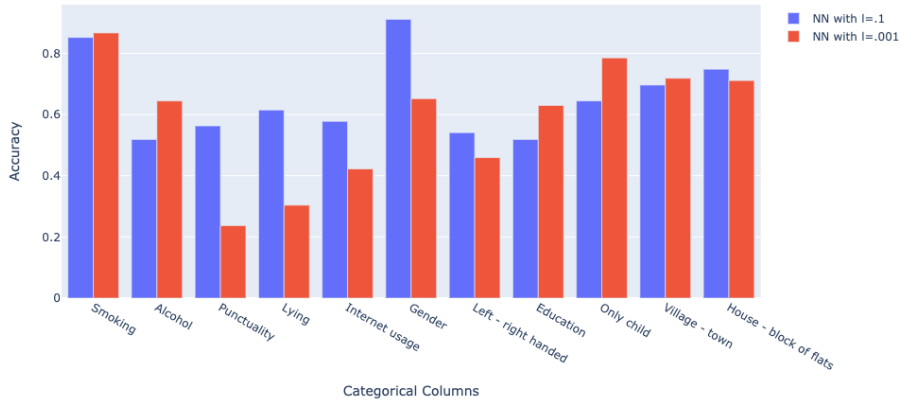

Figure 5.3: NN Accuracy Comparison between different learning rates

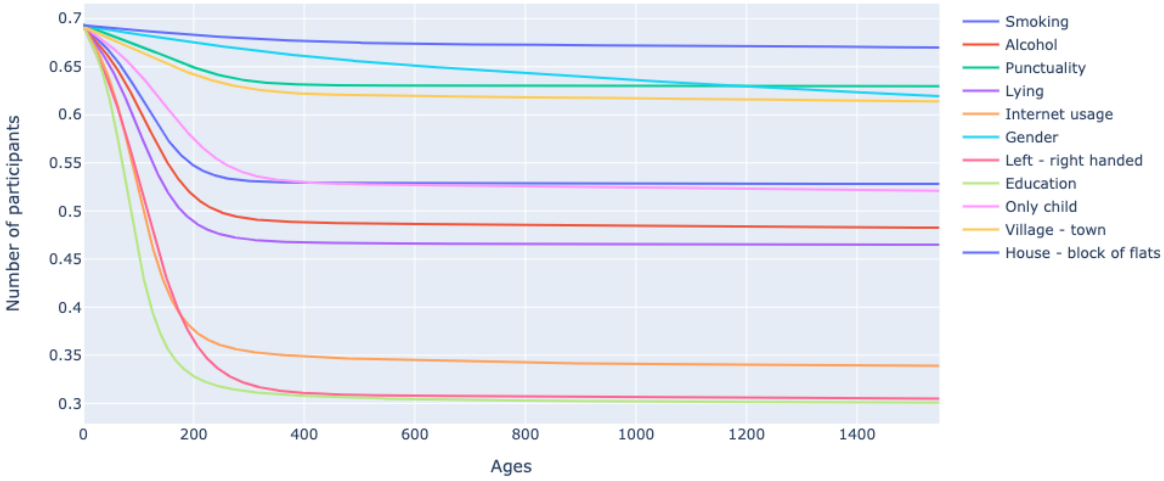

Figure 5.4: NN Loss plots features comparison on .001 learning rate

We tried to predict the categorical features using only numerical features predictors X, using different features, and different number of features to predict the categorical target the percentage of features used each time are [.1,.2,.3,.4,.5,.6,.7,.8,.9,1] of the features on learning rate .1 and 1550 iterations 5.6 most of the categorical features accuracy changed on changing the predictors.

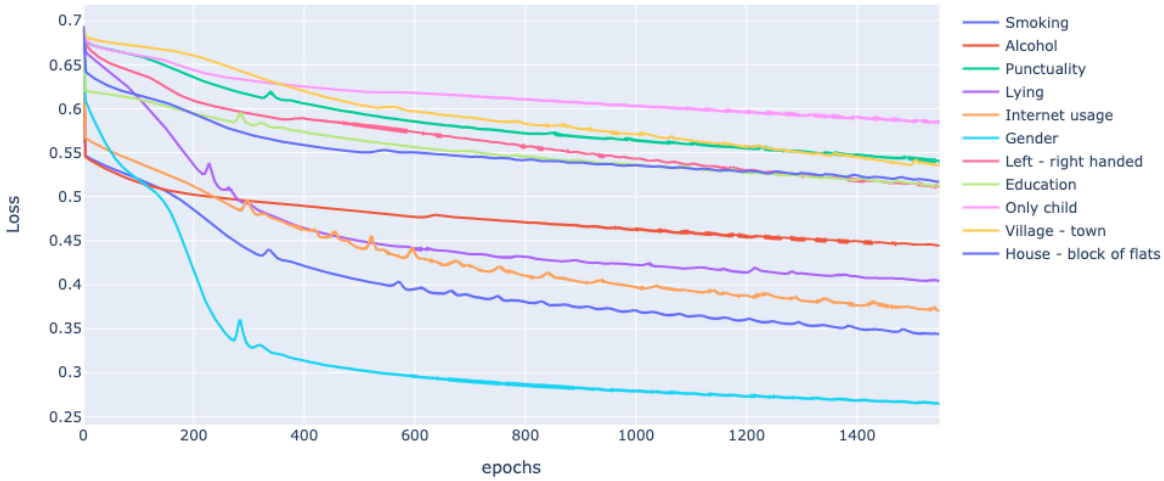

Figure 5.5: NN Loss plots features comparison on .1 learning rate

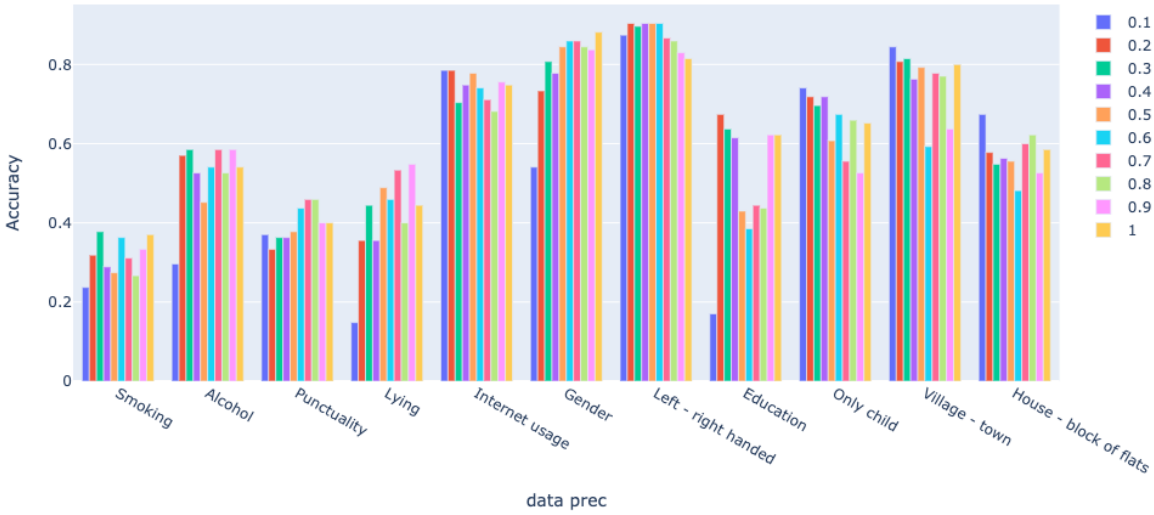

Figure 5.6: NN Accuracy Per Categorical columns using random portions of the numerical features

5.0.2 Regression Evaluation

We used the three models described in the model section to compare the results between them, the following steps were followed for the evaluation :

- For the training data and every numerical feature X set it as the target feature to predict Y, with n values.
- fit the data on training X, Y to find the lambda and number of iterations that reduce the error rate, Table 5.2 shows these parameters after tuning.
- on the calculated parameters predict on the testing dataset X.
- Evaluate using the MSE metric.

Figure 5.7 shows the best lambda was calculated on the test data to minimize the error for ridge regression and 5.8 showing that on lasso regression. Each numerical column mse calculation when predicted on ridge and lasso is showing in Figure 5.9 and 5.10 respectively. The results from lasso, ridge and basic mean model is showing in 5.11.

Model	Hyperparameters
Lasso Regression	Lambda = 1.9 , Iterations = 100
Ridge Regression	Lambda = 17.7

Table 5.2: Linear Regression Models Hyperparameter

The lasso and ridge regression showed around the same performance, Number of siblings and music showed lowest MSE, given the models parameters. Elections feature on the other side had the highest MSE = .15.

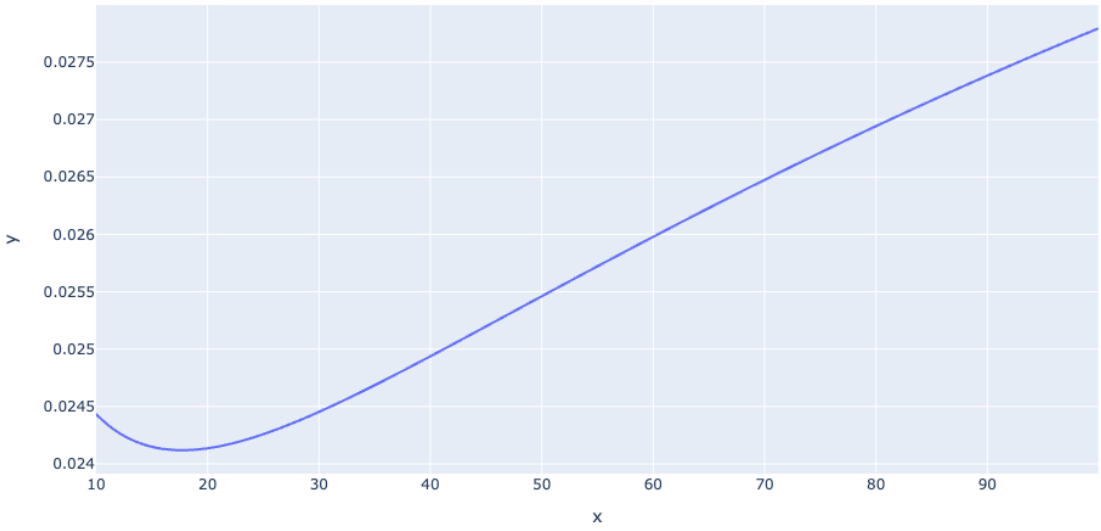

Figure 5.7: Find the lambda that minimize the error for Ridge regression

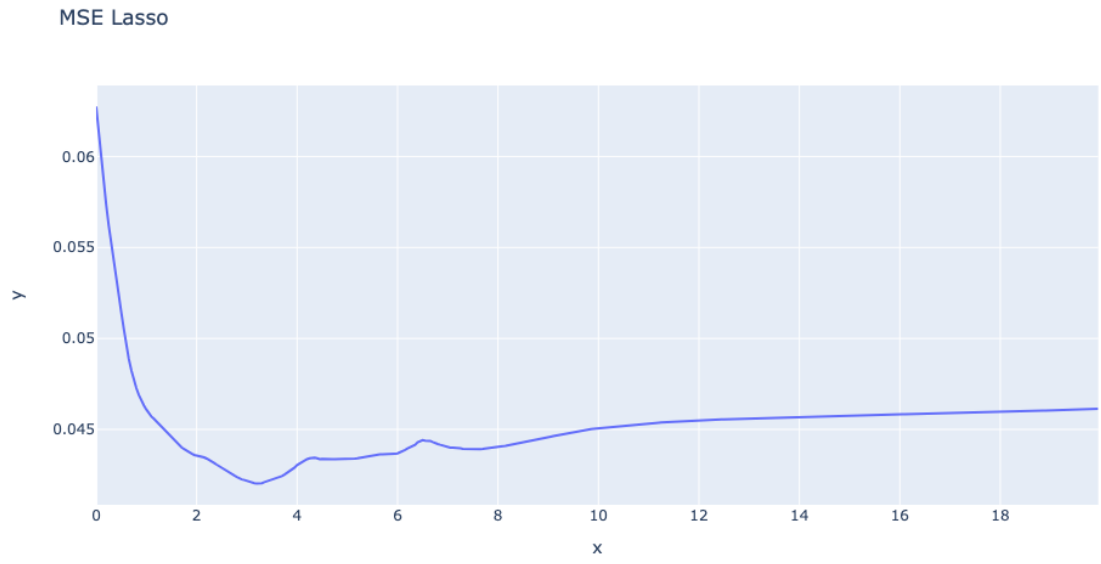

Figure 5.8: Find the lambda that minimize the error for Lasso regression

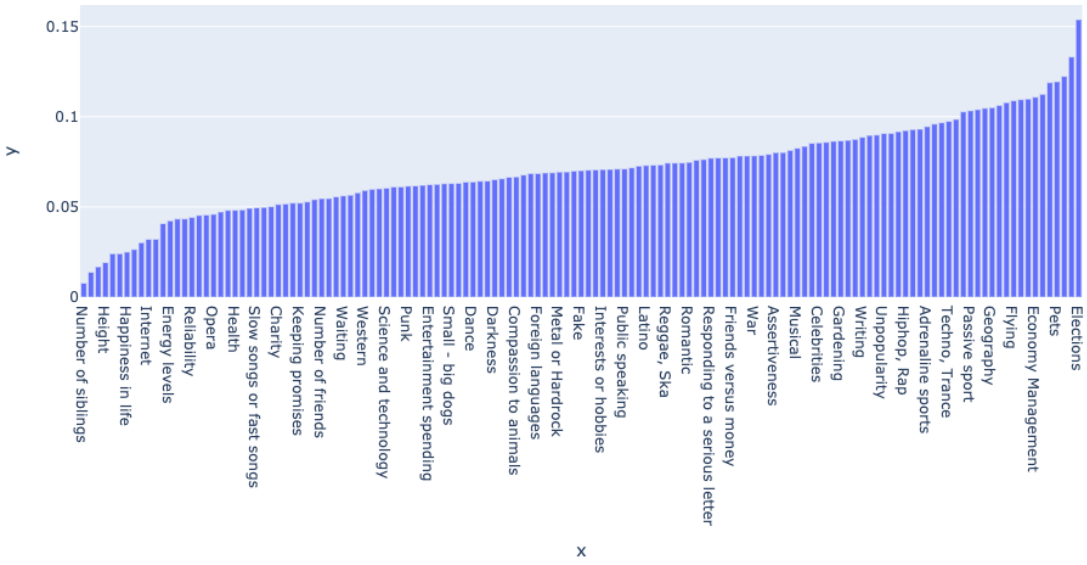

Figure 5.9: Ridge regression model - MSE

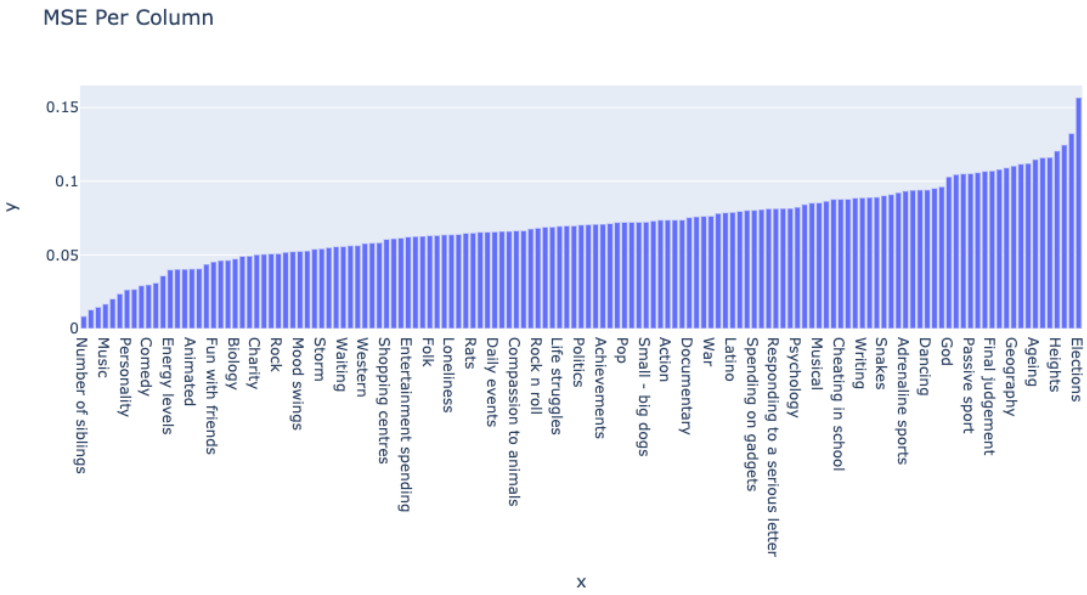

Figure 5.10: Ridge lasso model - MSE

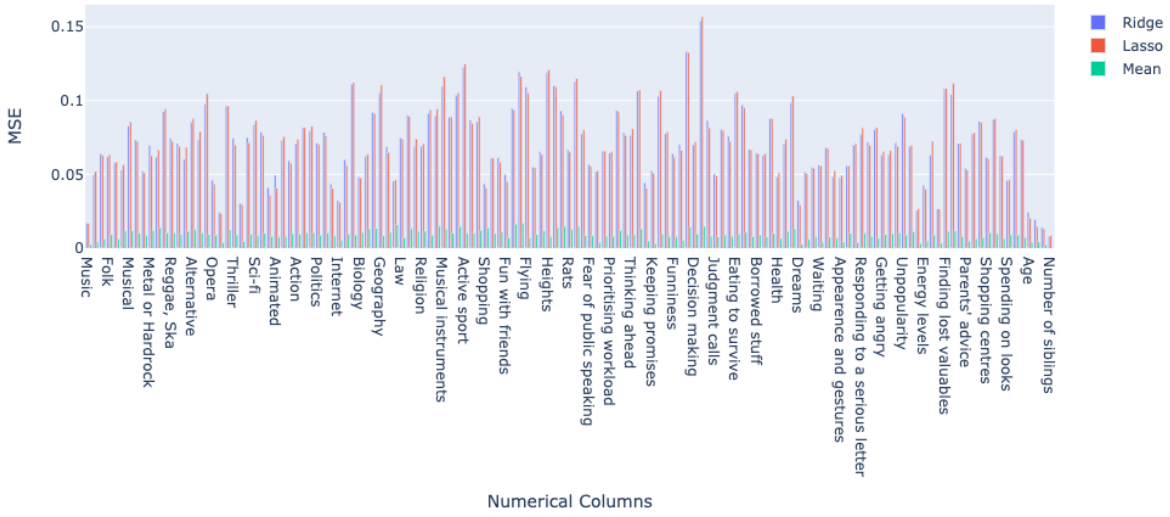

Figure 5.11: MSE Comparison between Regression Models

6 Data Analysis and Generation

What features were particularly valuable in predicting/interpolating? What features weren't particularly useful at all? Were there any features about the researcher/experimental environment that were particularly relevant to predicting answers - and if so, what conclusions can you draw about the replicability of those effects? Use your system to try to generate realistic data, and compare your generated data to the real data. How good does it look? What does it mean for it to 'look good'?
What features were particularly valuable in predicting/interpolating? What features weren't particularly useful at all?
For finding the useful features We used the lasso regression and checked the pruned features.
For X as predictors and Y as target, find pruned features save it, repeat until we have all features as target, then check the frequency of the pruned features in all steps, and the results showed that Weight, Height and Age were pruned in around 75% of the time, while comedy and movies pruned the least with 34% of the time. Please refer to the code results for full table. This results make sense for us as the features with less missing values are supposed to be more valuable, Fig 2.2 showed us that Weight, Height and Age features are with the most missing values.
Were there any features about the researcher/experimental environment that were particularly relevant to predicting answers - and if so, what conclusions can you draw about the replicability of those effects? The features were all relevant to predicting answers different combination of features where relevant to predict specific features, for example music preferences features where more relevant to predict features representing same topic like Rock, Rock n Roll, Metal or Hard rock. Or Chemistry, Medicine and Biology which appears to make sense the answers for questions under same topic will be related to each other eventually.

6.1 Data Generation

Fill the missing data - Numerical

- Start with the dataset with missing values which was around 40% of the original data X
- Start with numerical features
- Pick the features with the least na values a, get all records that a exists in with missing value (if the records contains more than one missing values we will add

them all as target Y to predict multiple numerical features at a time)

- We will end up with k number of features that are correlated to the target feature Y
- Get the k features from the testing data F
- Fit the the filtered training data T (with no missing data) on the regression model discussed before.
- Normalize the dataset we want to predict one or more target features on (X)
- Pass each record in dataset X for prediction
- Add the predicted value to X, so we will use it for other features predictions
- De-normalize and add it to the last dataset where we fill data with no missing values
- Repeat until we have no missing values

Fill the missing data - Categorical

- Start with the dataset with missing values which was around 40% of the original data X
- Next we predict the categorical features(get probability for each class and pick the one with highest as predicted value).
- Pick the features with the least na values a, get all records that a exists in with missing value, If any categorical data is missed at the same record both will be passed as target features.
- Get the k numerical features from the testing data F
- Fit the filtered training T (with no missing data) on the classification model
- Normalize the dataset we want to predict the target feature on (X)
- Pass each record in dataset X for prediction
- Find the class value with highest probability, decode and add it back to X, so we will use it for other features predictions
- Reverse the one-hot encoding and add the record to the last dataset where we fill data with no missing values
- Repeat until we have no missing values

This was followed up by processing, concatenating and converting to csv with no missing data, files are attached with the project files

6.2 Data Analysis

In addition to the evaluation metrics that we ran, In order to check the predicted values and the model performance. We ran the analysis on the new data after filling all the missing values. Figure 6.1 shows the count of participants per gender, the ages are still normally distributed with missing values predicted.

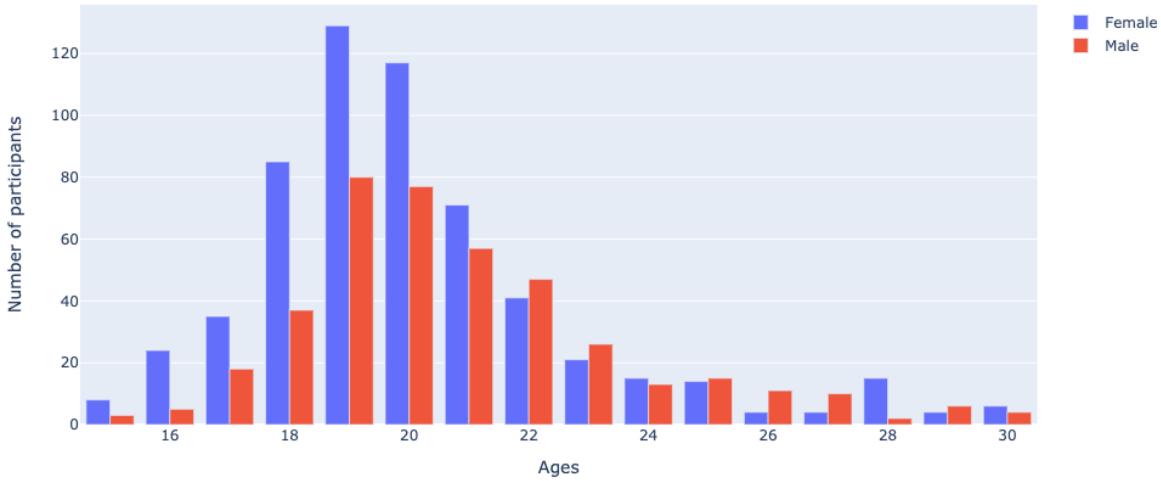

Figure 6.1: Ages per Gender After Data Generation

As we found some columns that where highly correlated like 'Chemistry','Biology', 'Medicine' we tried to compare our results with other existing records. One of the responses ranking their interested int Chemistry, Biology and Medicine the table shows comparison between existing responses and our predicted results in the table 6.1 and we found it very close, giving promising results.

Existed Responses			Predicted Responses		
Chemistry	Biology	Medicine	Chemistry	Biology	Medicine
1	1	1	1	1	1
3	2	3	3	2	1
2	2	1	2	2	1
4	4	5	4	4	5
3	3	2	3	3	3
2	1	3	2	1	1

Table 6.1: Compare Existed and Predicted Results

7 Appendix

7.1 Appendix A: Survey Questions and Answers Options

Quesion	Answers
I enjoy listening to music	Strongly disagree 1-2-3-4-5 Strongly agree (integer)
Musicals	Don't enjoy at all 1-2-3-4-5 Enjoy very much (integer)
History	Not interested 1-2-3-4-5 Very interested (integer)
Flying	Not afraid at all 1-2-3-4-5 Very afraid of (integer)
Smoking habits	Never smoked - Tried smoking - Former smoker - Current smoker
Age	(integer)
Height	(integer)
Weight	(integer)
How many siblings do you have?	(integer)

Table 7.1: Survey Questions and Answers Options snippet

7.2 Appendix B: Code and Notebook

Please Find it in the attached documents, html file is important to view plots in the notebook.

7.3 References

[1] *CS 536 : Final Project - Data Completion and Interpolation*. URL: https://drive.google.com/file/d/1erfg66Ue-6TTGS67zqJRXWNOV_IaWZpS/view?usp=sharing.

[2] *Kaggle Young People Survey*. URL: https://www.kaggle.com/miroslavsabo/young-people-survey.