World Happiness Report and

Regression Comparison

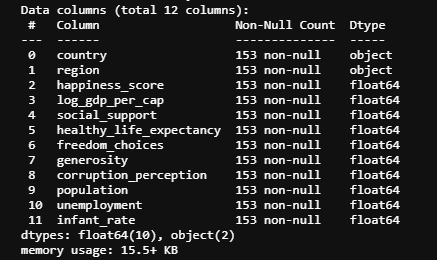
# Research Question

The purpose of this research is to determine how well one can predict the Happiness Index using various regression models and which regression tactic is the most effective. To resolve this question, we will compare the performance or Linear Regression, Ridge Regression, Lasso Regression, and Random Forest Regression models when predicting the Happiness Index for various countries in the 2020 World Happiness Report.

# Data Set Source and Description

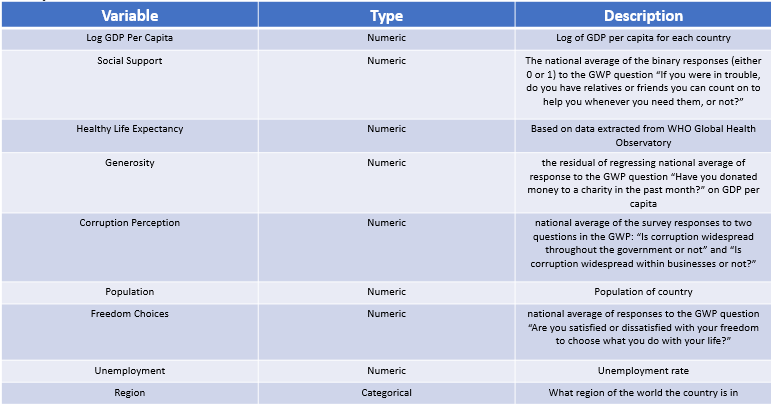
The data in question comes from the 2020 [World Happiness Report](https://worldhappiness.report/ed/2020/). Each year the WHR teams up with Gallup and a few other data sources to use various statistical strategies to attempt to quantify happiness throughout the world and assign a happiness score to each country. I downloaded the bulk of the data from [kaggle](a.%09https:/www.kaggle.com/londeen/world-happiness-report-2020) then proceeded to pull in a couple other interesting input variables from the [World Bank](https://data.worldbank.org/) data source.





Above we can see what these data look like and a snippet of the .info() output.

# Description of Variables and Data Preprocessing



The table above describes the input variables of interest.

Preprocessing steps included the following:

1. Read in world happiness data
2. Subset to relevant columns
3. Read in population data, infant mortality data, and unemployment data
4. Merge our data
5. StandardScaler() on our numeric columns
6. Since “region” is a non-ordinal categorical variables, I refactored this variable into multiple 0-1 columns (one for each region)

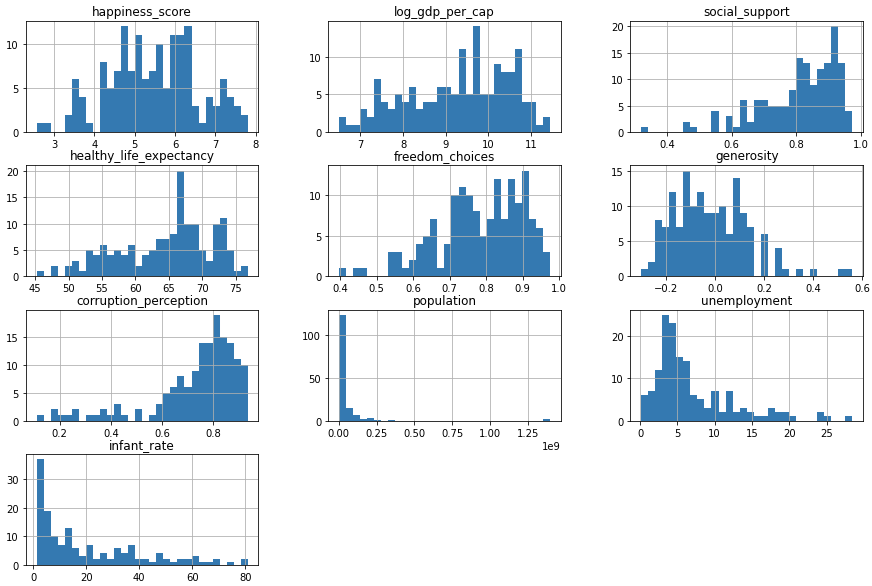
# Exploratory Data Analysis (EDA)

In our EDA steps we established a few key pieces of information including the distributions of our continuous variables, the frequencies of our categorical variables, and the correlations between variables.

First, I started off with some basic descriptive statistics using the .describe() function from the pandas package. The results are below



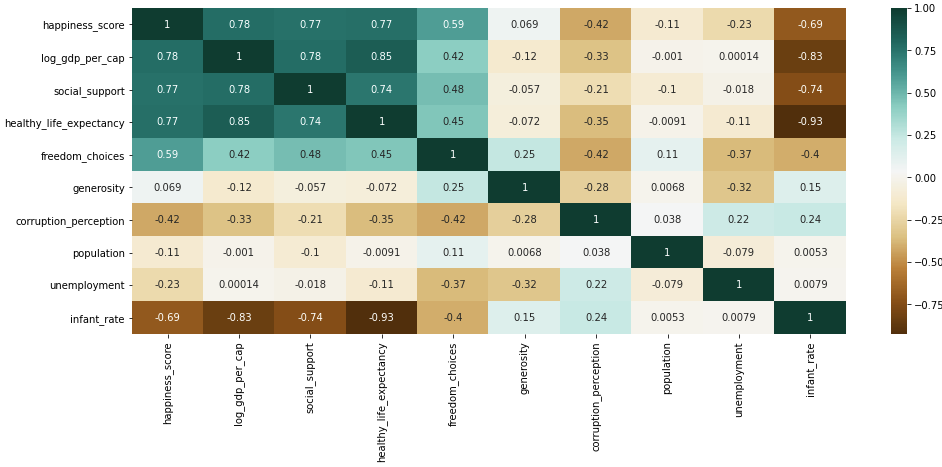
Next, I went on to observe these distributions with a collection of histograms.



The next step in our EDA was to observe the relevant counts in our “region” variable. Those counts are displayed below.

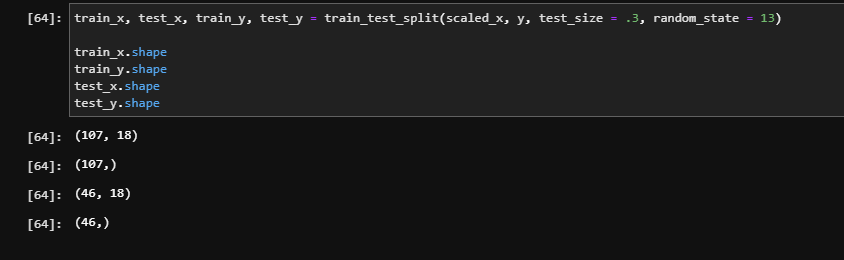


Finally, I created a correlation plot to observe the correlations between variables in our data. That plot is shown below.



# Data Partitioning

For this project, I chose to use the train test split method with a 30% testing size. I did this using the train\_test\_split() function in python. The size of the splits are shown below



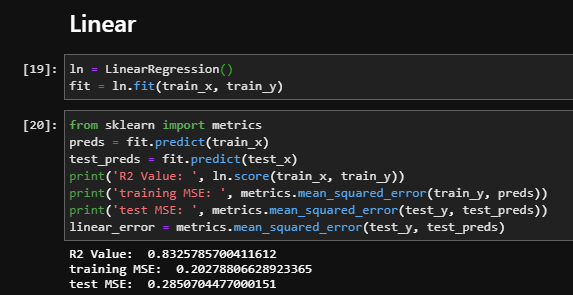
# Modeling

Finally, I moved on to building, tuning, and comparing the four different models.

## Linear Regression

The first model I built was a basic Linear Regression using the LinearRegression() method from the sklearn package.

For the linear regression model, I left the default hyperparameters. You will find the code fitting the model and displaying relevant statistics below.

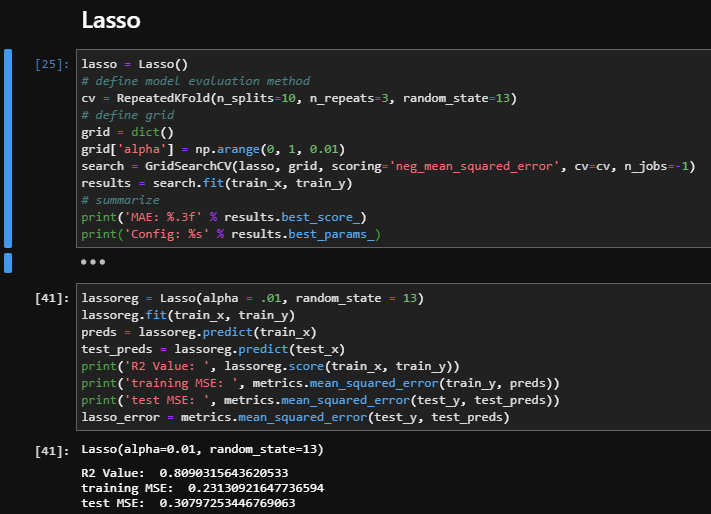


From the linear regression model we see an R2 value of .832 and a testing MSE of .285.

## Lasso Regression

The next model I used was the Lasso() method from the sklearn package. In this case, I added a couple tuning steps using GridSearchCV to ensure that the model was the best possible model for this case.

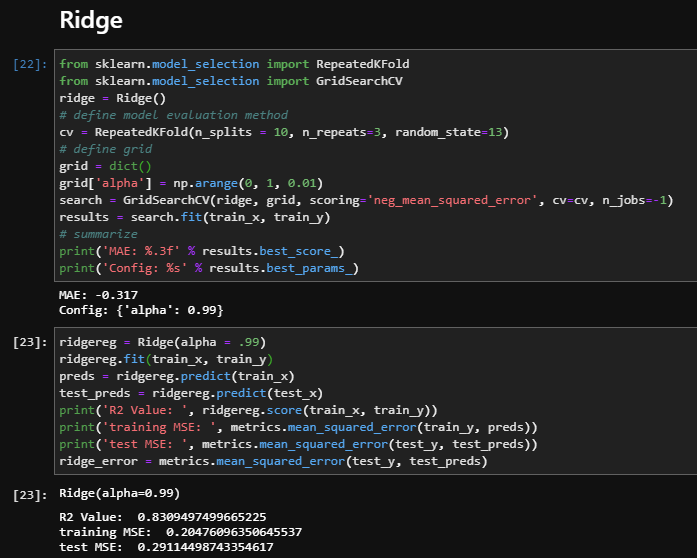
Below we see the code used to tune the alpha parameter, fit the model, and display relevant statistics.



From the Lasso model we see an R2 of .809 and a testing MSE of .308.

## Ridge Regression

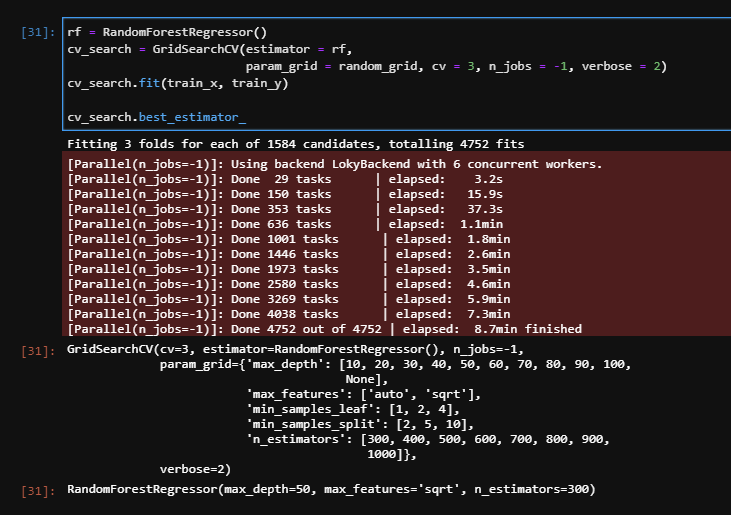
Very similar to the Lasso model, the next model is the Ridge() method also from the sklearn package. In this case I used the same strategies to tune the alpha value. The output is displayed below.

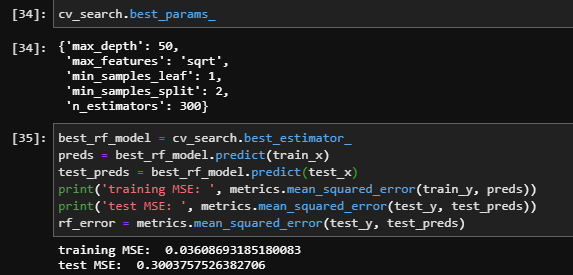


From the Ridge regression model we see an R2 value of .831 and a testing MSE of .291

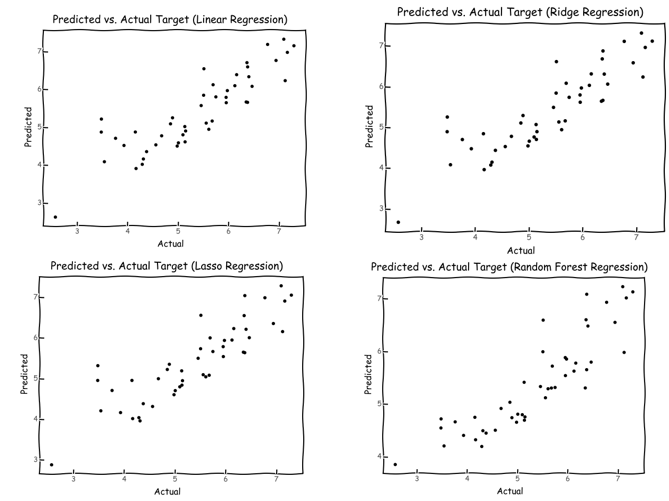
## Random Forest Regression

Finally, I decided to try a RandomForrestRegressor() method from the sklearn package. With this model I tuned a collection of different parameters including “max\_dept”, “max\_features”, “min\_samples\_leaf”, “min\_samples\_split”, and “n\_estimators”. The code used to tune and create this model is shown below.

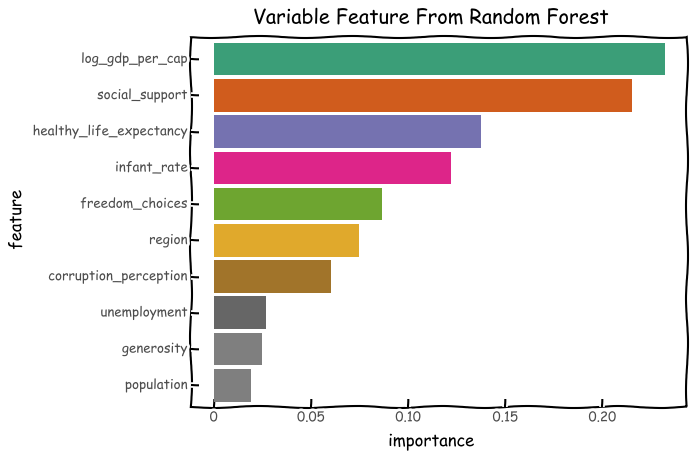




Next, I graphed the predicted output vs the actual output for each model and displayed the results.

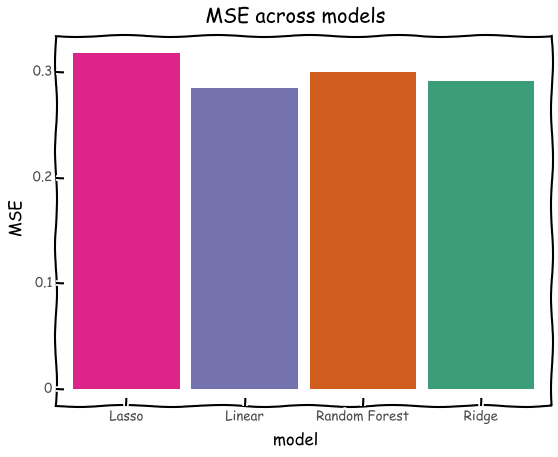


Next, I used the feature\_importances\_ call from the random forest model to create a graph that displays which variables were deemed most important in this model.



# Conclusion

Finally, we want to compare the four models to answer our research question. We can see on the graph below the comparison of the testing mean squared error across all models



Using the results from all our models, we see that the Linear Regression model has the lowest testing MSE of .285 and highest R2 value of .832.