ABORTION RATES IN AMERICA

NATIONAL AND STATE

ABORTION AND PREGNANCY

RATES FROM 1973 - 2017

Lauren Chlebove Shweta Ale Tsering Lhamo Philip Park

INTRODUCTION

Since the landmark Supreme Court decision of Roe v. Wade in 1973, the right to abortion has been protected in the United States at the federal level. However, in a country as large and diverse as America, opinions on abortion have always varied greatly between states. Until recently, states were unable to ban abortion outright. What does the data show about abortion rates across the country?

We begin our exploration by distinguishing **RED** and **BLUE** states.

RED States

Typically a "Red" state is a state that leans towards conservative, Republican values. **Abortion** is generally opposed.

BLUE States

Typically a "Blue" state is a state that leans towards liberal, Democratic values. **Abortion** is generally supported.

HYPOTHESIS: Abortion rates will be lower in red states due to state laws that limit abortion.

KEY ATTRIBUTES

PREGNANCY RATES

Columns for teen pregnancy rate and pregnancy as a whole

ABORTION RATES

RESPONSE VARIABLE

YEAR

Categorical

STATE

Categorical

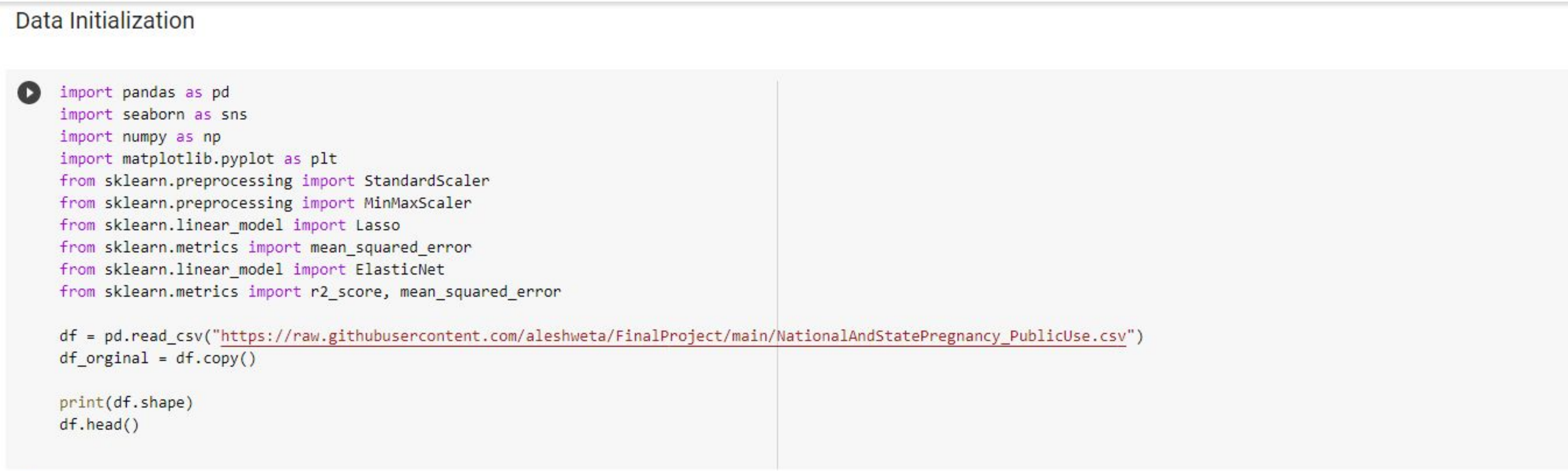
BIRTH RATES

Columns for teen birth rate and birth as a whole

STATE COLOR

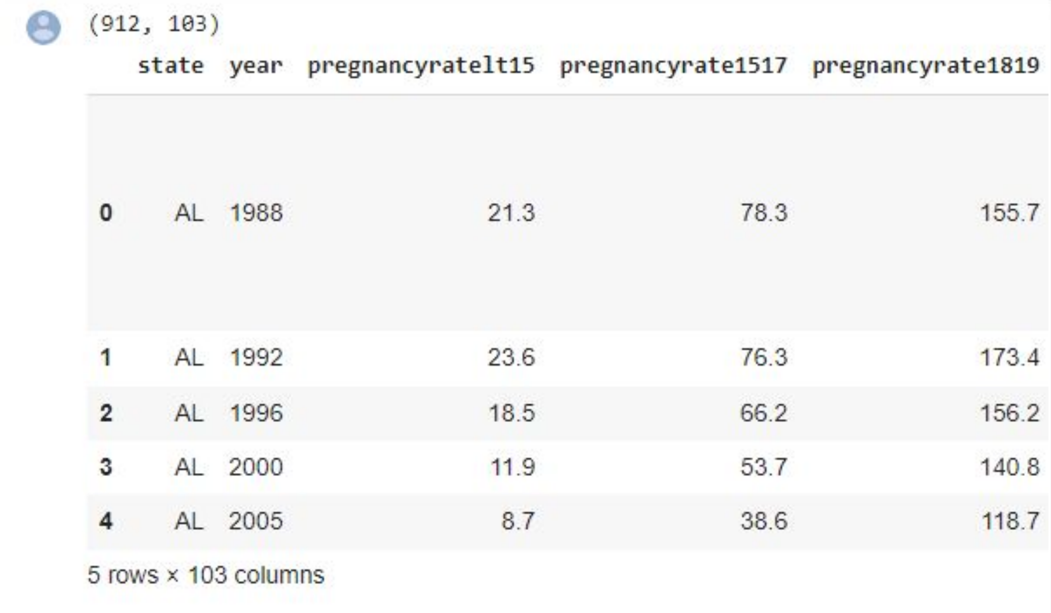
Denotes political affiliation of state

DATA INITIALIZATION & CLEANING



Data Initialization

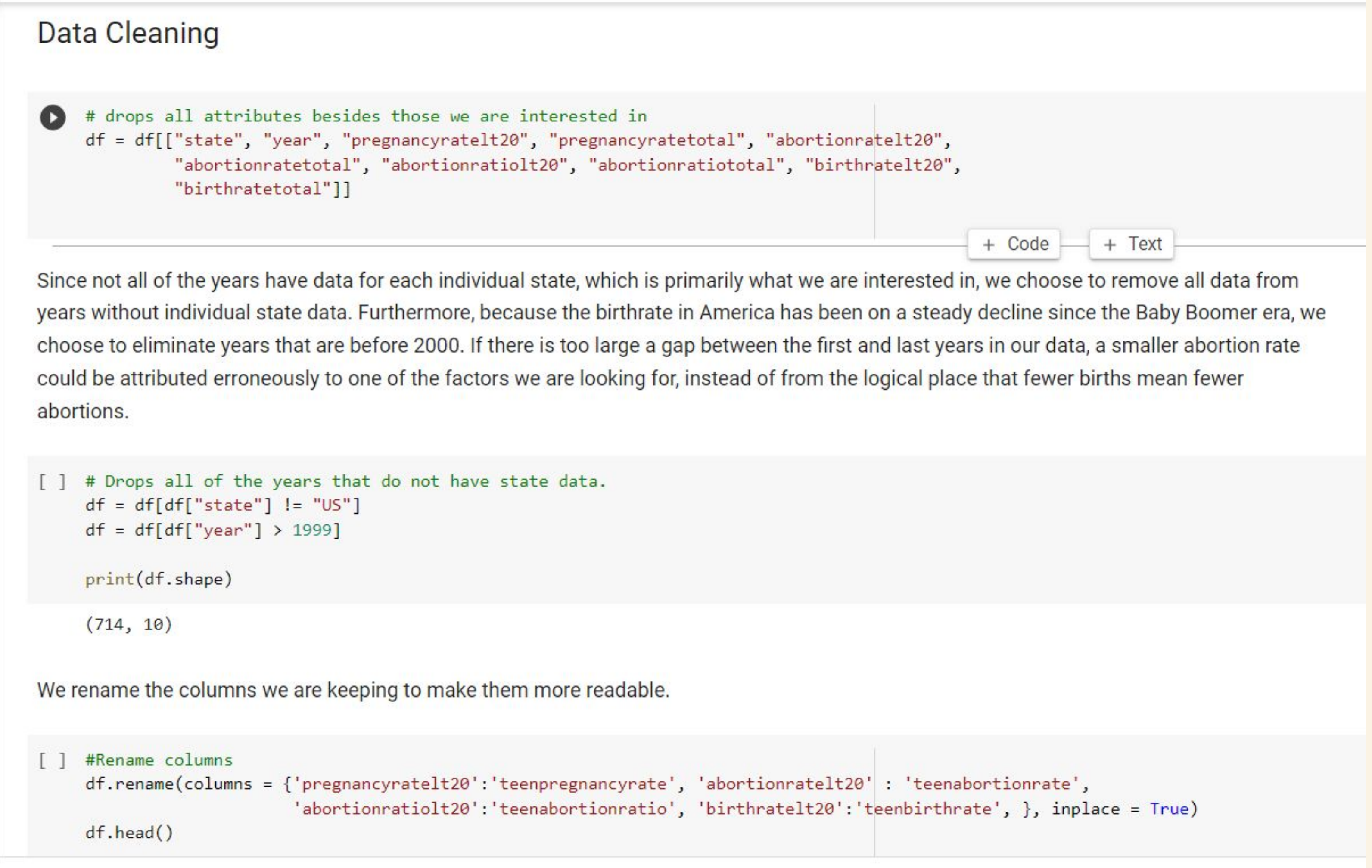
```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from sklearn.linear_model import Lasso
from sklearn.metrics import mean_squared_error
from sklearn.linear_model import ElasticNet
from sklearn.metrics import r2_score, mean_squared_error

df = pd.read_csv("https://raw.githubusercontent.com/aleshweta/FinalProject/main/NationalAndStatePregnancy_PublicUse.csv")
df_orginal = df.copy()

print(df.shape)
df.head()
```



```
(912, 103)
```

	state	year	pregnancyratelt15	pregnancyrate1517	pregnancyrate1819
0	AL	1988	21.3	78.3	155.7
1	AL	1992	23.6	76.3	173.4
2	AL	1996	18.5	66.2	156.2
3	AL	2000	11.9	53.7	140.8
4	AL	2005	8.7	38.6	118.7

5 rows × 103 columns



Data Cleaning

```
# drops all attributes besides those we are interested in
df = df[["state", "year", "pregnancyratelt20", "pregnancyratetotal", "abortionratelt20",
        "abortionratetotal", "abortionratiolt20", "abortionratiototal", "birthratelt20",
        "birthratetotal"]]
```

Since not all of the years have data for each individual state, which is primarily what we are interested in, we choose to remove all data from years without individual state data. Furthermore, because the birthrate in America has been on a steady decline since the Baby Boomer era, we choose to eliminate years that are before 2000. If there is too large a gap between the first and last years in our data, a smaller abortion rate could be attributed erroneously to one of the factors we are looking for, instead of from the logical place that fewer births mean fewer abortions.

```
# Drops all of the years that do not have state data.
df = df[df["state"] != "US"]
df = df[df["year"] > 1999]

print(df.shape)
```

```
(714, 10)
```

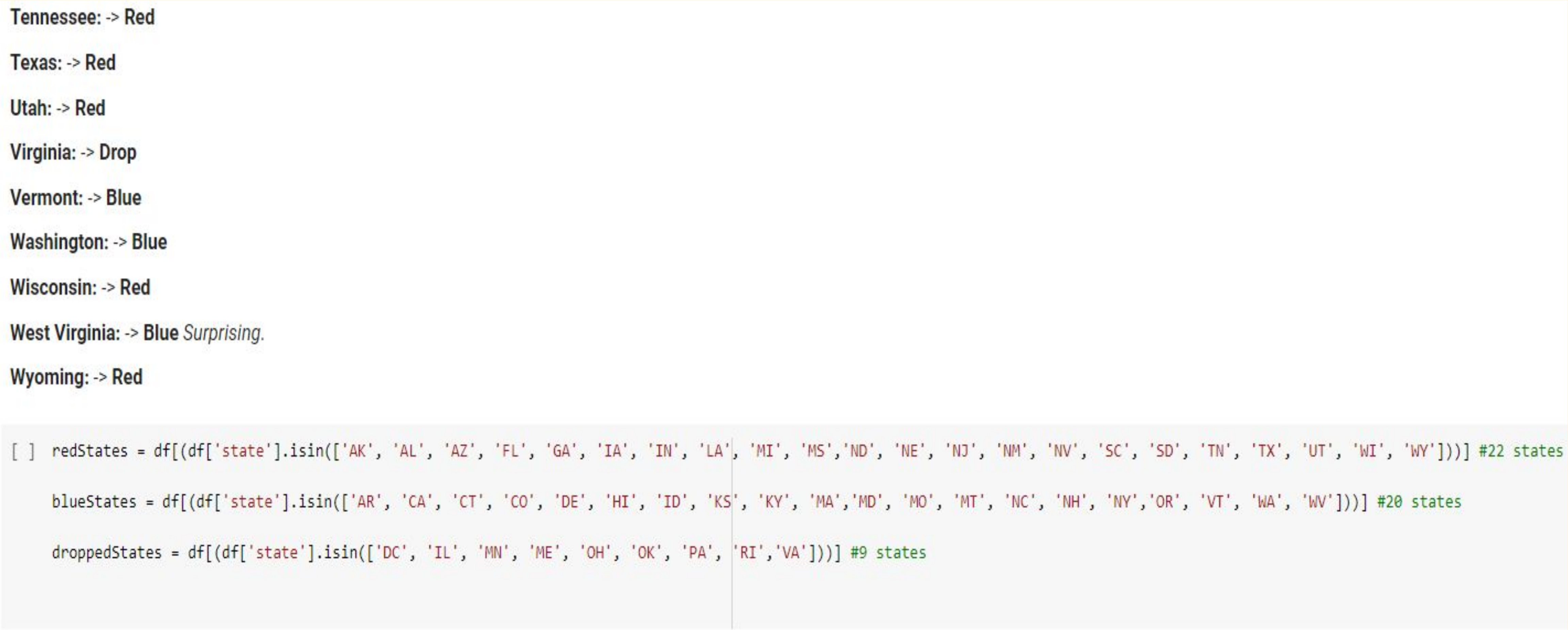
We rename the columns we are keeping to make them more readable.

```
#Rename columns
df.rename(columns = {'pregnancyratelt20':'teenpregnancyrate', 'abortionratelt20' : 'teenabortionrate',
                     'abortionratiolt20':'teenabortionratio', 'birthratelt20':'teenbirthrate', }, inplace = True)
df.head()
```

OVERVIEW

- Initialize the data and import relevant libraries.
- Survey the dataset.
- Keep the attributes that are relevant for our analysis.
- Drop earlier years from the set since there has been a steady decline of pregnancies over the last 60+ years (since the Baby Boom).

RED AND BLUE DATAFRAMES (PREPPING OUR MODELS)



Tennessee: -> **Red**

Texas: -> **Red**

Utah: -> **Red**

Virginia: -> **Drop**

Vermont: -> **Blue**

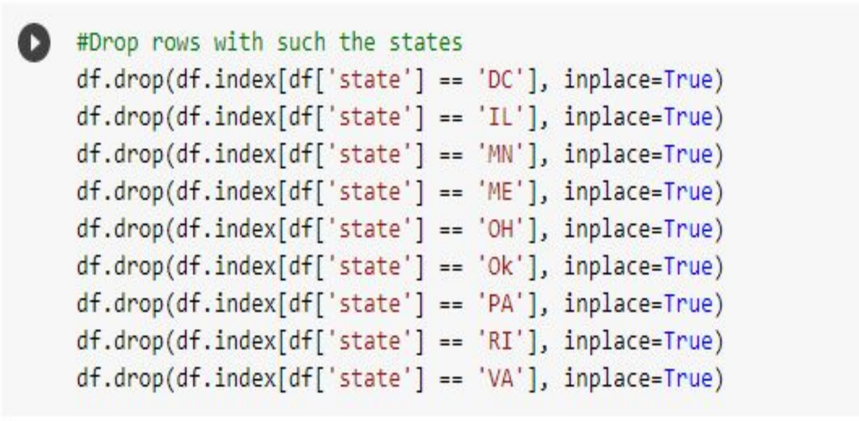
Washington: -> **Blue**

Wisconsin: -> **Red**

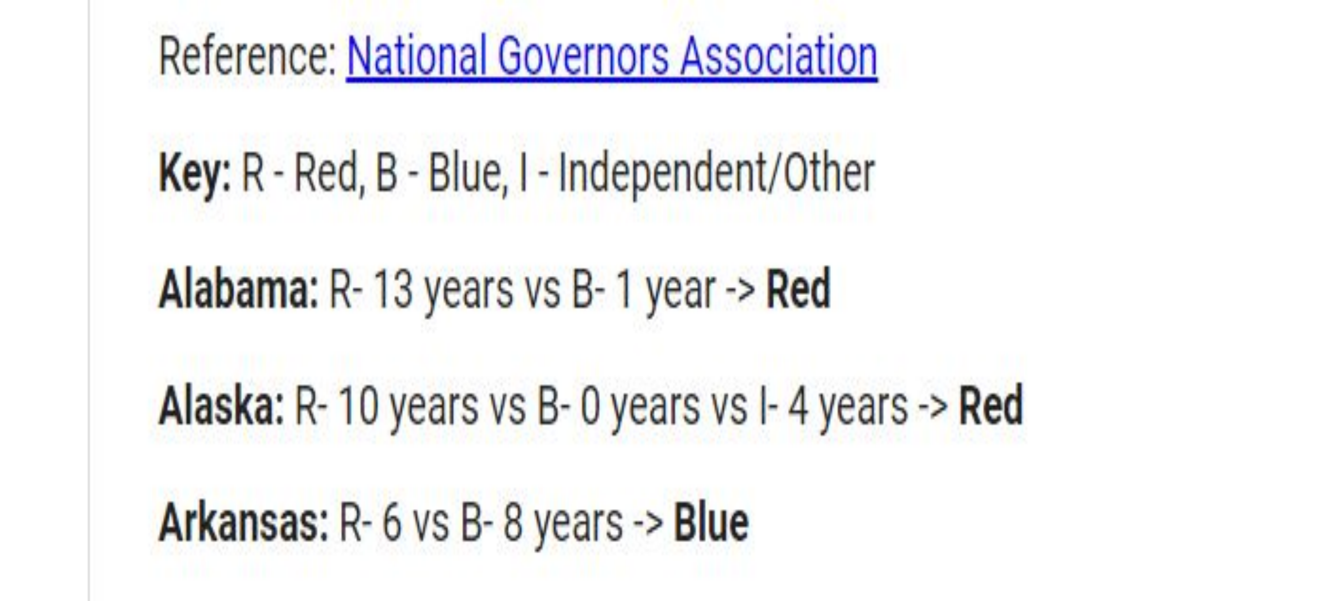
West Virginia: -> **Blue** *Surprising.*

Wyoming: -> **Red**

```
[ ] redStates = df[(df['state'].isin(['AK', 'AL', 'AZ', 'FL', 'GA', 'IA', 'IN', 'LA', 'MI', 'MS','ND', 'NE', 'NJ', 'NM', 'NV', 'SC', 'SD', 'TN', 'TX', 'UT', 'WI', 'WY']))] #22 states

    blueStates = df[(df['state'].isin(['AR', 'CA', 'CT', 'CO', 'DE', 'HI', 'ID', 'KS', 'KY', 'MA','MD', 'MO', 'MT', 'NC', 'NH', 'NY','OR', 'VT', 'WA', 'WV']))] #20 states

    droppedStates = df[(df['state'].isin(['DC', 'IL', 'MN', 'ME', 'OH', 'OK', 'PA', 'RI','VA']))] #9 states
```



```
#Drop rows with such the states
df.drop(df.index[df['state'] == 'DC'], inplace=True)
df.drop(df.index[df['state'] == 'IL'], inplace=True)
df.drop(df.index[df['state'] == 'MN'], inplace=True)
df.drop(df.index[df['state'] == 'ME'], inplace=True)
df.drop(df.index[df['state'] == 'OH'], inplace=True)
df.drop(df.index[df['state'] == 'Ok'], inplace=True)
df.drop(df.index[df['state'] == 'PA'], inplace=True)
df.drop(df.index[df['state'] == 'RI'], inplace=True)
df.drop(df.index[df['state'] == 'VA'], inplace=True)
```



Reference: National Governors Association

Key: R - Red, B - Blue, I - Independent/Other

Alabama: R- 13 years vs B- 1 year -> **Red**

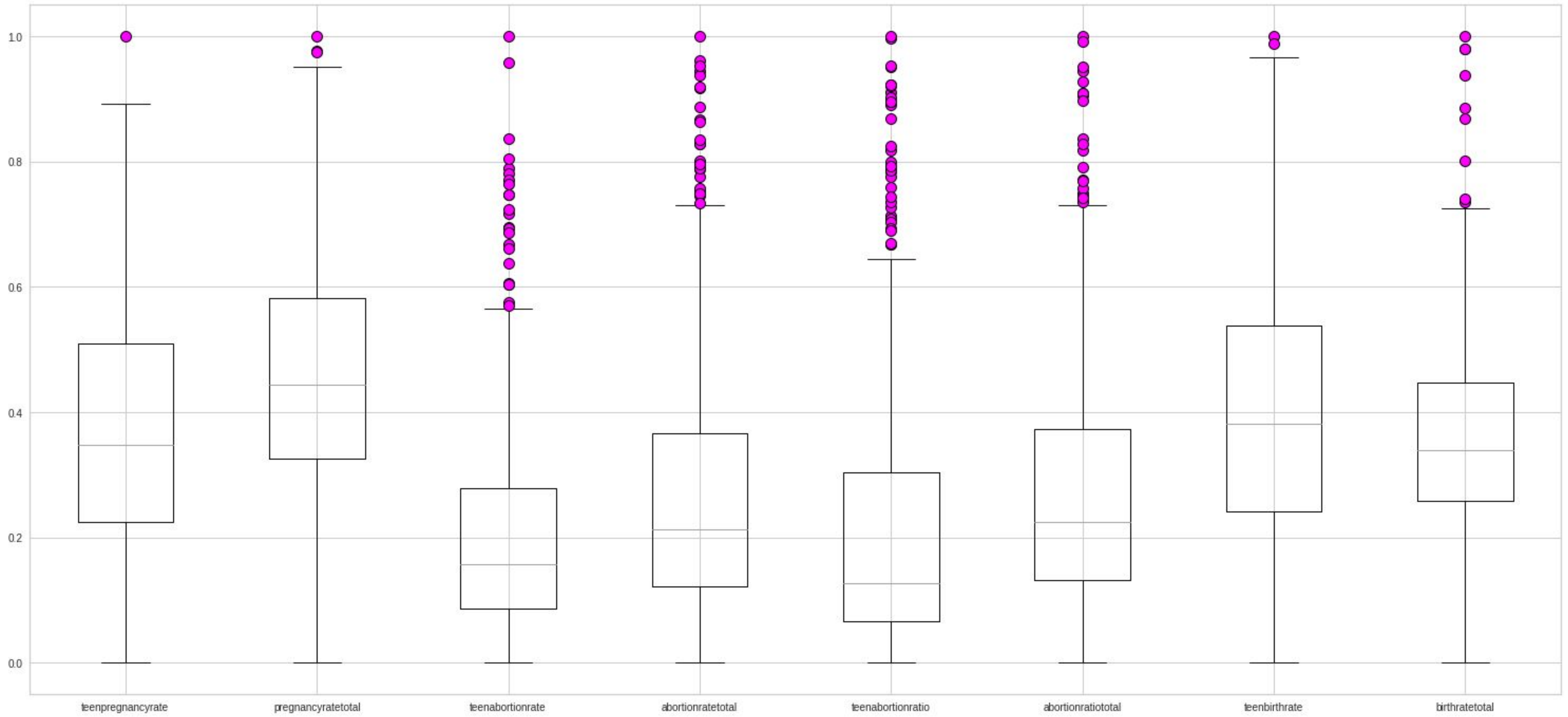
Alaska: R- 10 years vs B- 0 years vs I- 4 years -> **Red**

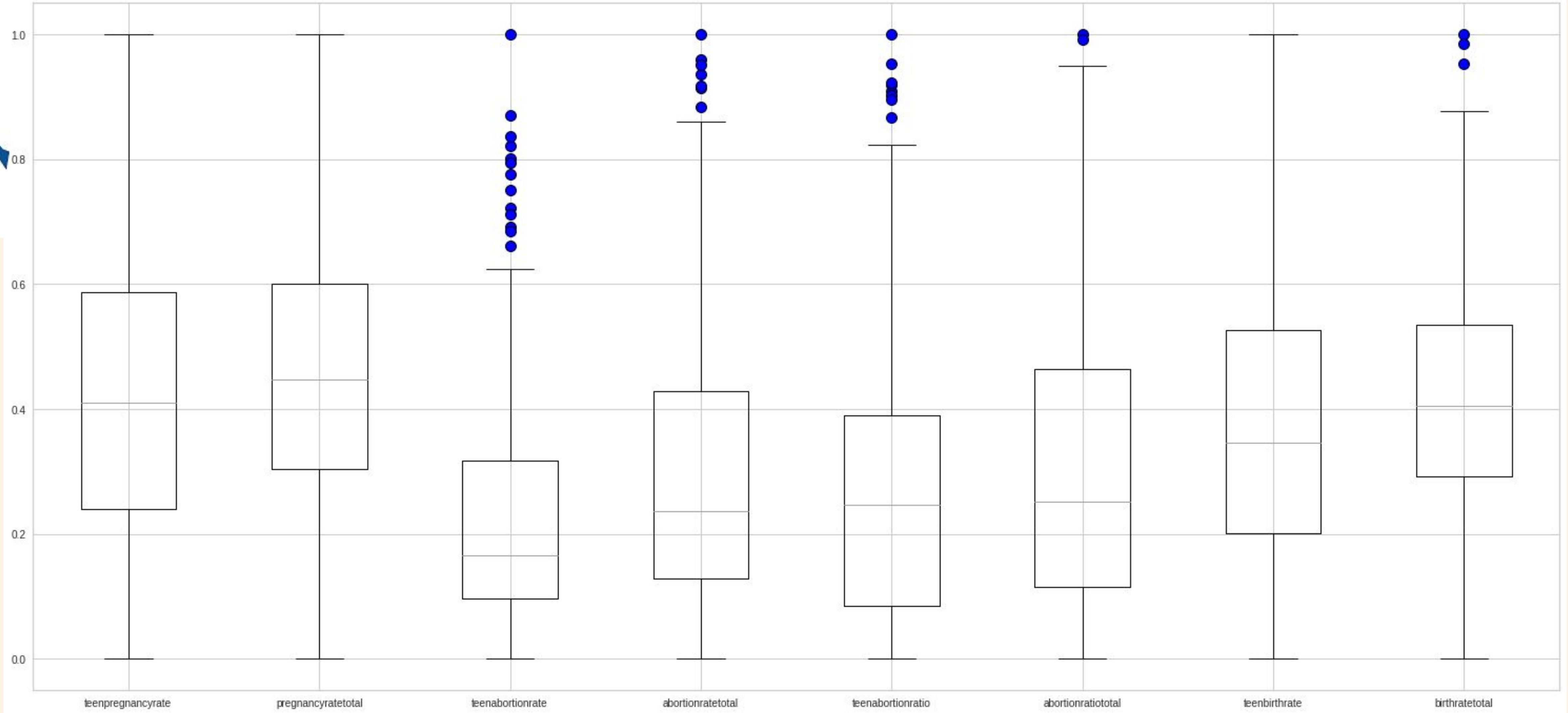
Arkansas: R- 6 vs B- 8 years -> **Blue**

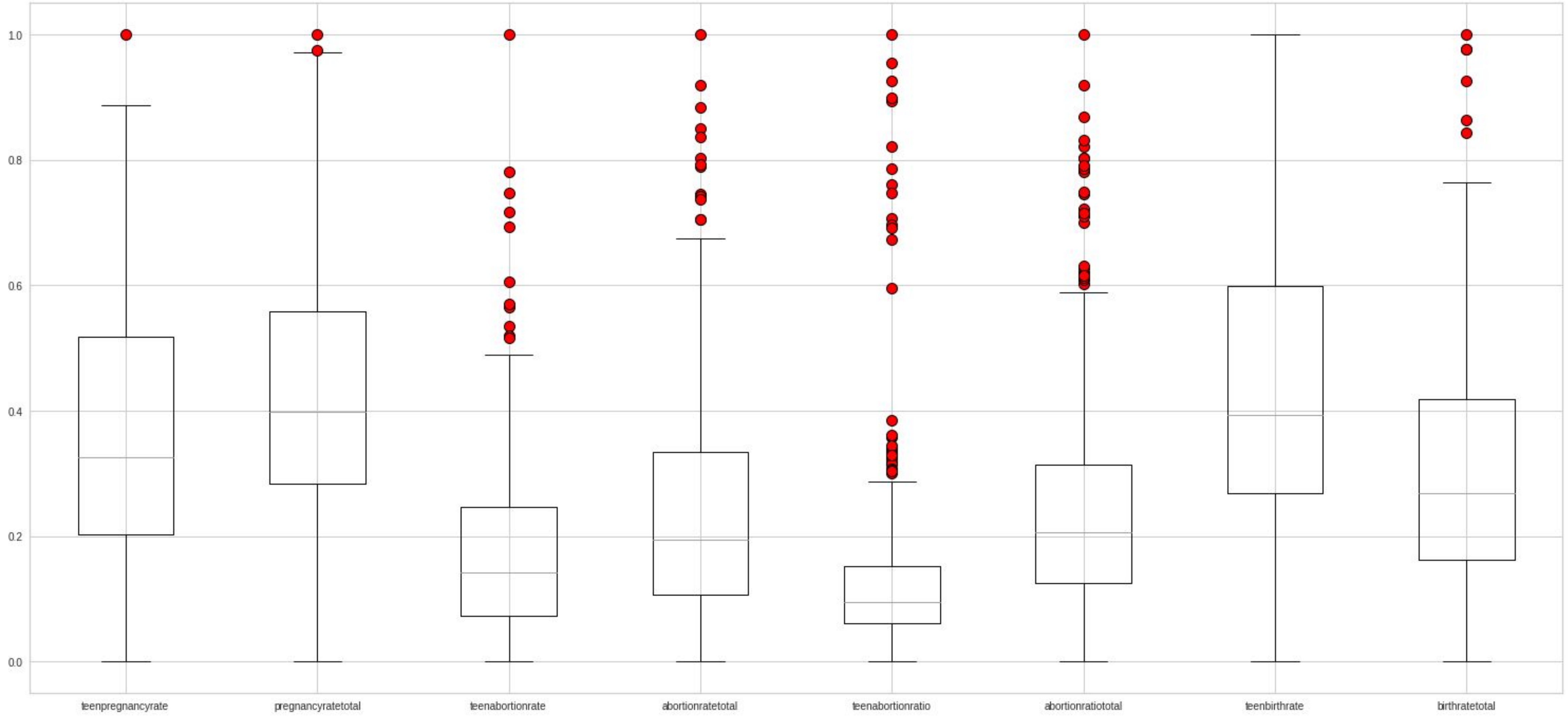
Logical Categorization

1. Admit the complexity and imperfections of categorizing Red and Blue states.
2. Create 3 distinctions: R (Red), B (Blue), and I (Independent/Other)
3. Assign distinctions based on the sitting governor for said year. Our reasoning: as elected heads of states, they have the final say on restrictive or relaxed abortion laws.
4. Tally up points. The “winning” color has to be at least 2 points higher than the “losing” color.
5. Populate the 3 dataframes (Red, Blue, Dropped).

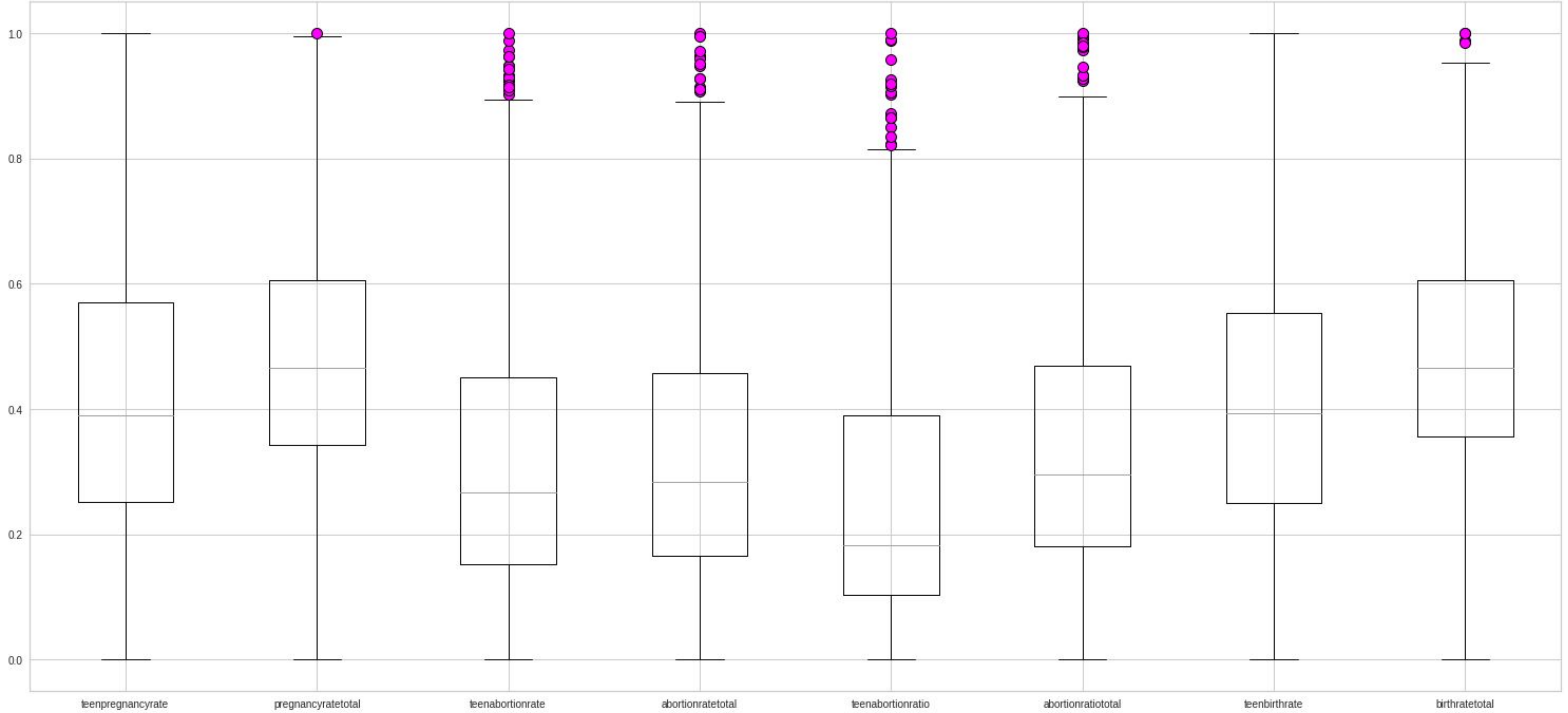
Tackling Outliers

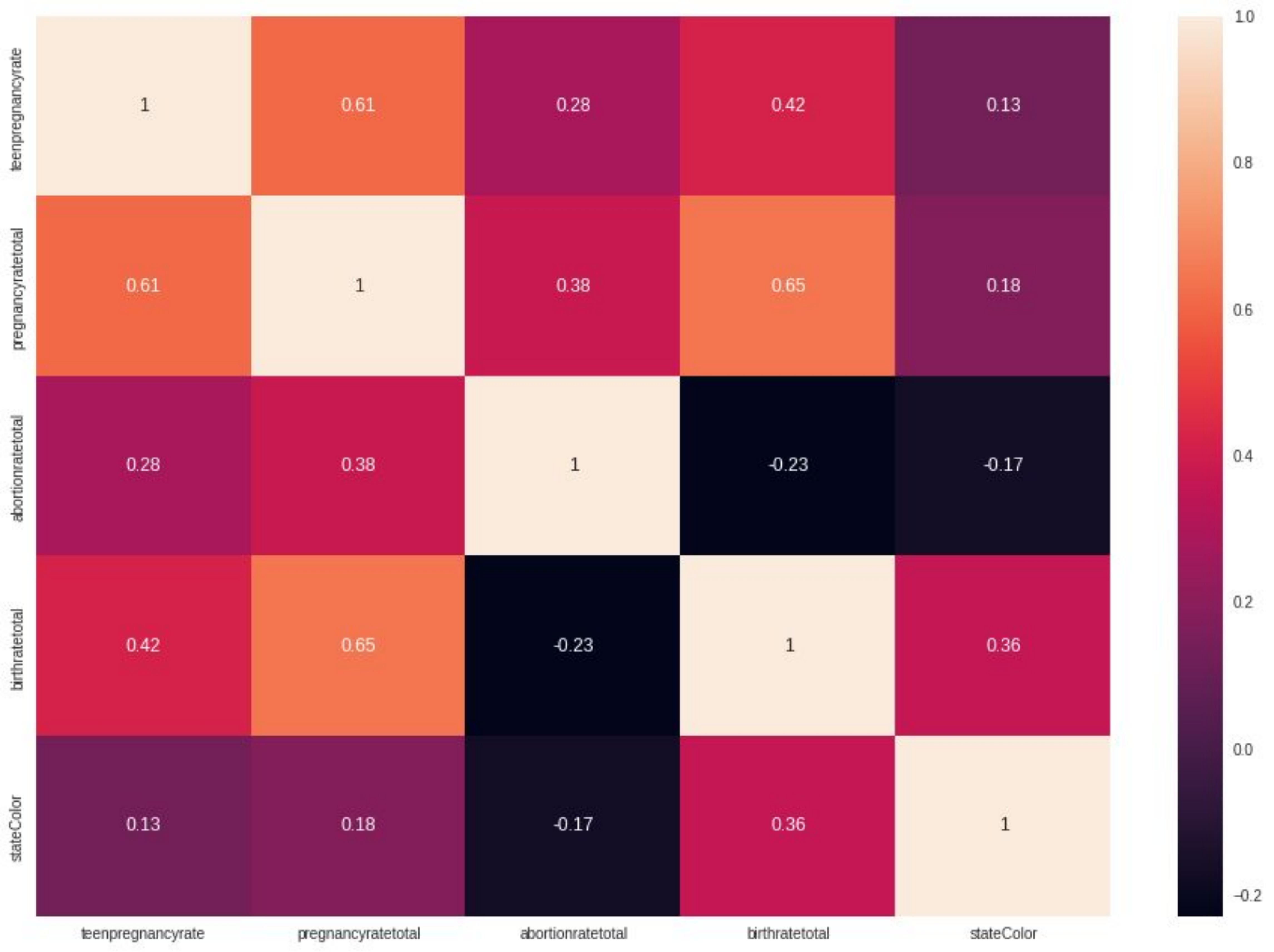



1.0
0.8
0.6
0.4
0.2
0.0
teenpregnancyrate
pregnancyratetotal
teenabortionrate
abortionratetotal
teenabortionratio
abortionratiototal
teenbirthrate
birthratetotal


1.0
0.8
0.6
0.4
0.2
0.0
teenpregnancyrate
pregnancyratetotal
teenabortionrate
abortionratetotal
teenabortionratio
abortionratiototal
teenbirthrate
birthratetotal

Tackling Outliers Part 2





HEATMAP

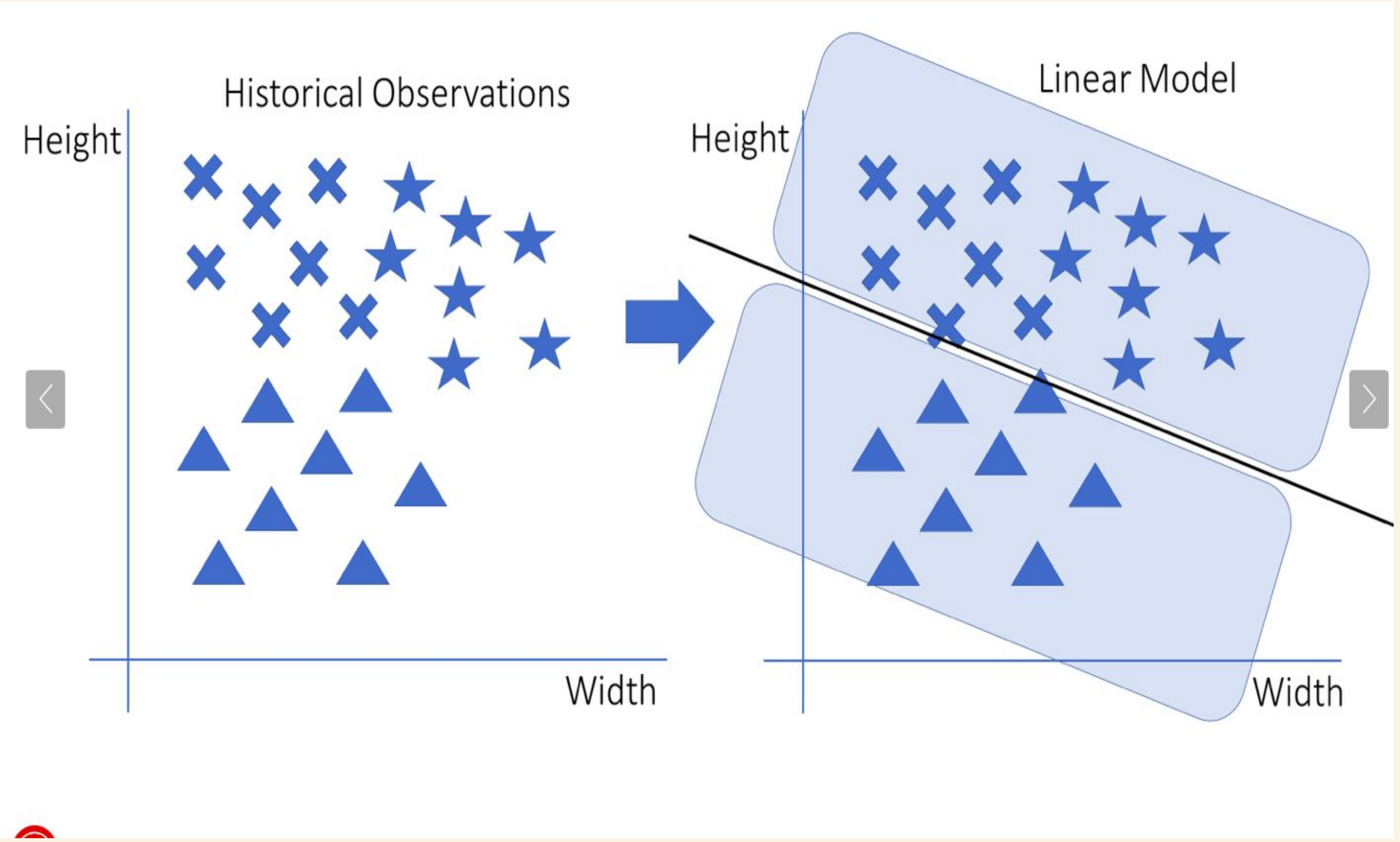
Trivial:

- Teen Pregnancy vs Pregnancy Total
- Birth Total vs Pregnancy Total
- Birth Total vs Teen Pregnancy

Potentially Interesting:

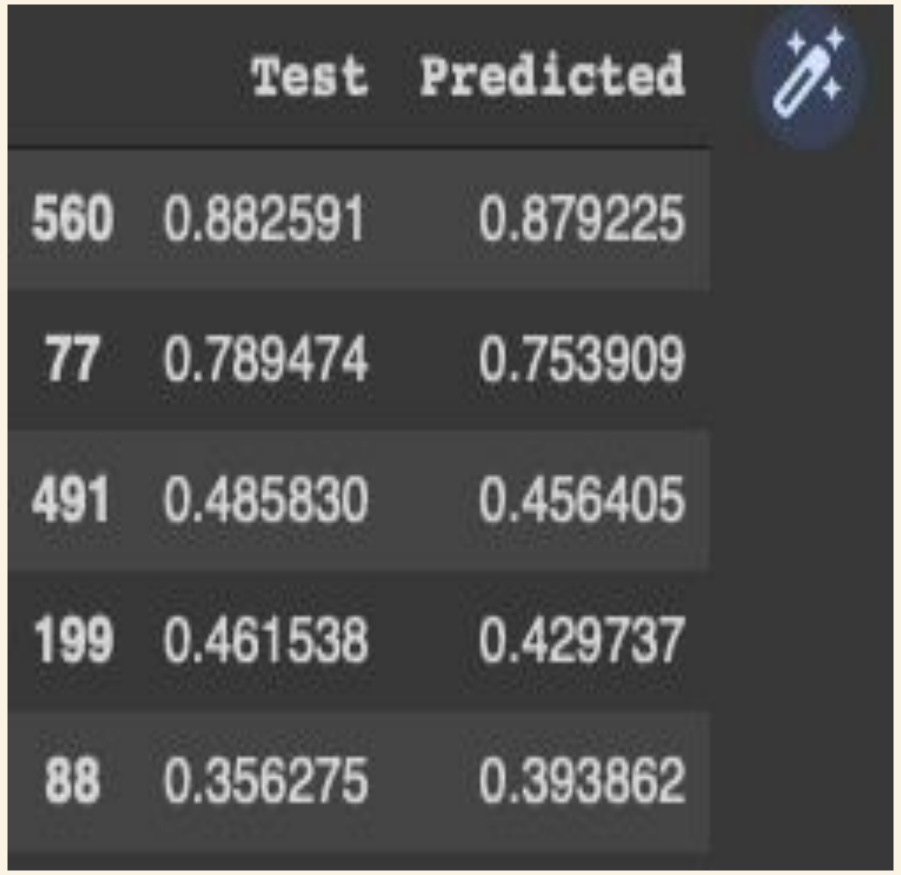
- State Color vs Birth Total

LINEAR REGRESSION



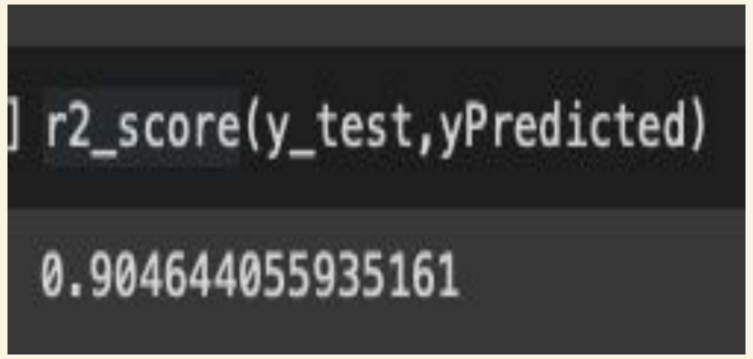
- Linear regression is a **linear** model that predicts using lines or hyperplanes like the picture in RHS were a single line separates triangles from non-triangles.
- For our analysis as well, we tested bunch of models to make accurate prediction for our target/ dependent attribute i.e “abortionratetotal” by training 80% of their data with 80% of other independent attributes from our dataset excluding the “abortionratetotal”.

LINEAR REGRESSION



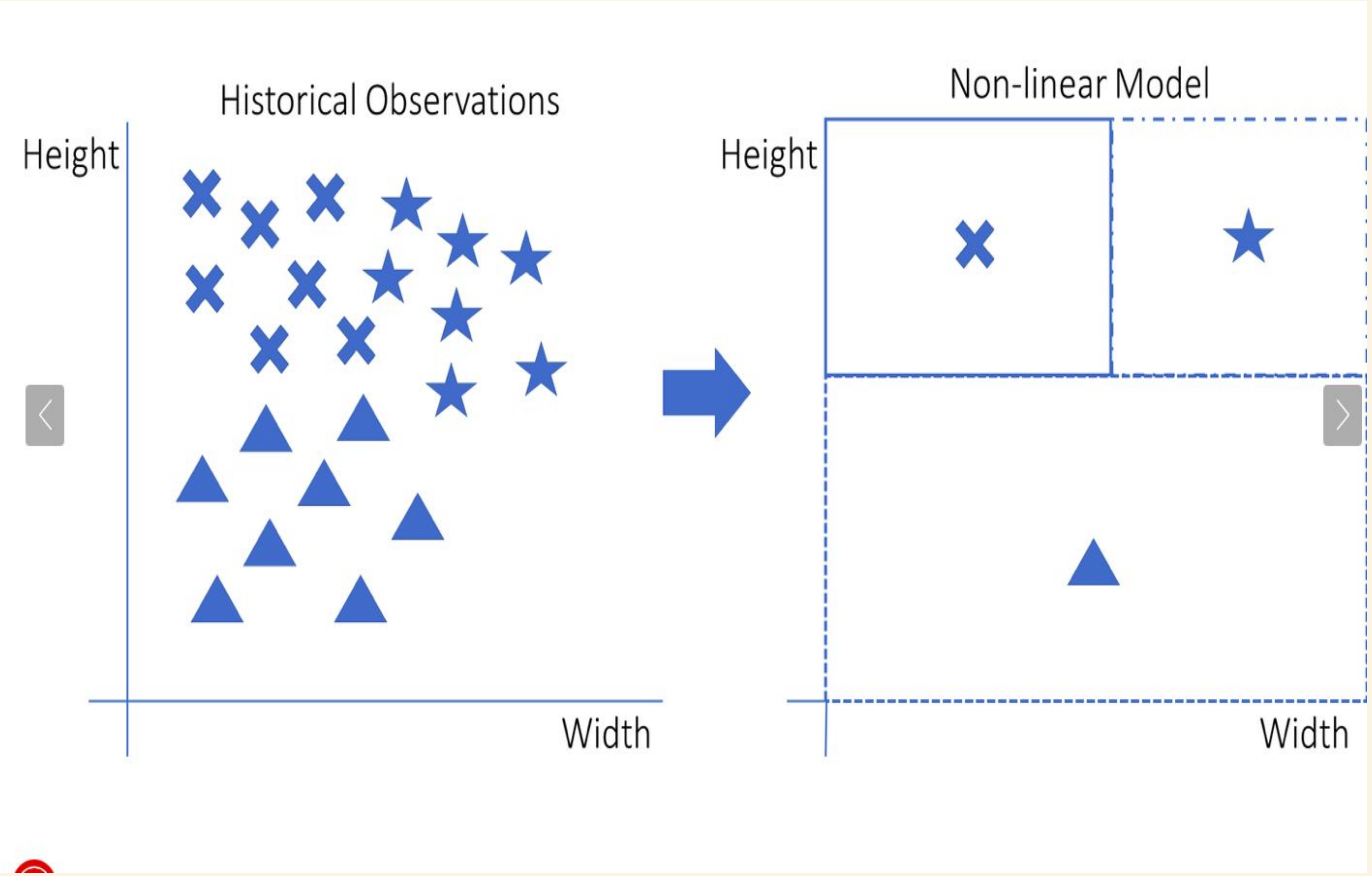
	Test	Predicted
560	0.882591	0.879225
77	0.789474	0.753909
491	0.485830	0.456405
199	0.461538	0.429737
88	0.356275	0.393862

- Results:
- RHS has our results for actual value of our test("abortionratetotal") dataset and value of what our model predicted.
- LHS shows our r2_score which shows our results are 90%accurate.



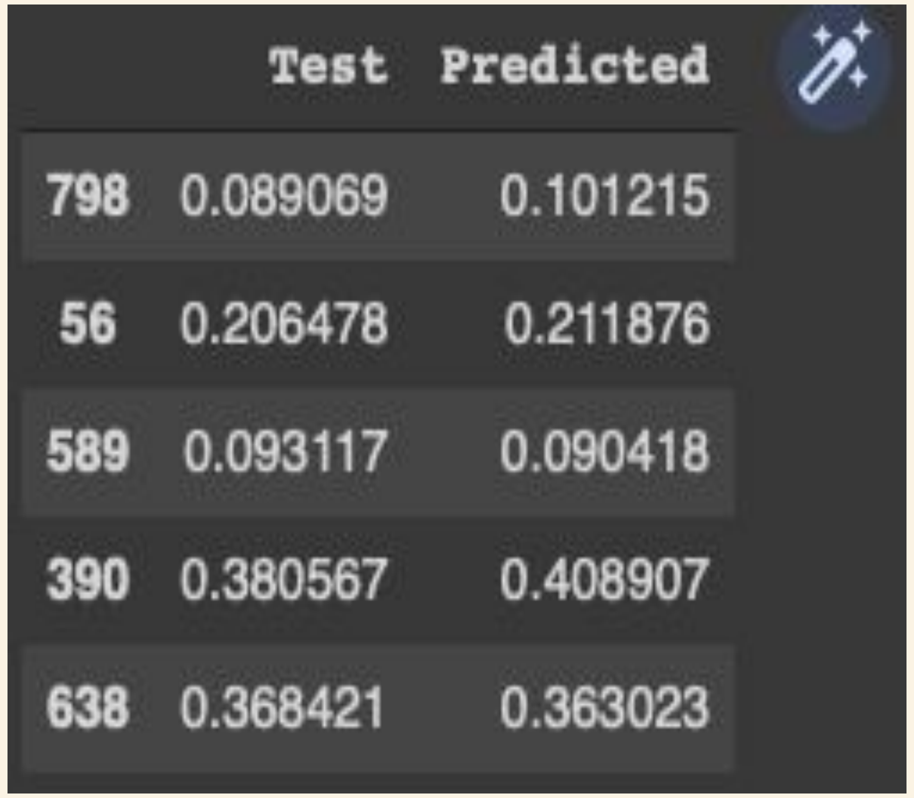
```
] r2_score(y_test,yPredicted)

0.904644055935161
```

KNN REGRESSION



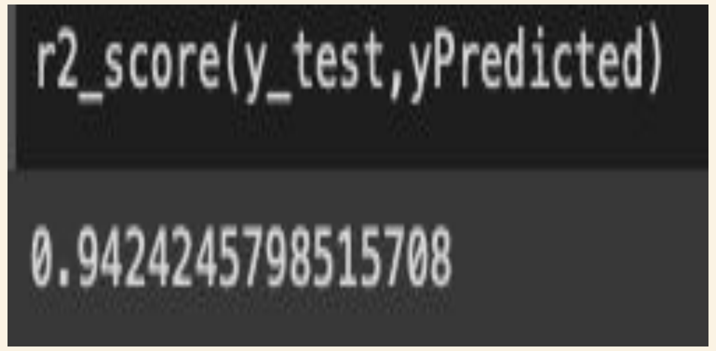
- K-Nearest Neighbor as Regression is a non linear model which uses any forms other than lines to separate their cases.
- RHS it uses squares to show how stars, triangle and cross are different from each other.
- We will be again using same x and y attributes as before but we also feed them n_neigbors = 3 which means when a new data point arrives, the algorithm will start by finding the 3 nearest neighbors of this new data point.
- Then it takes the values of those 3 neighbors and uses them as a prediction for the new data point.

KNN REGRESSION



	Test	Predicted
798	0.089069	0.101215
56	0.206478	0.211876
589	0.093117	0.090418
390	0.380567	0.408907
638	0.368421	0.363023

- Results:
- RHS has our results for actual value of our test("abortionratetotal") dataset and value of what our model predicted.
- LHS shows our r2_score which shows our results are 94%accurate.



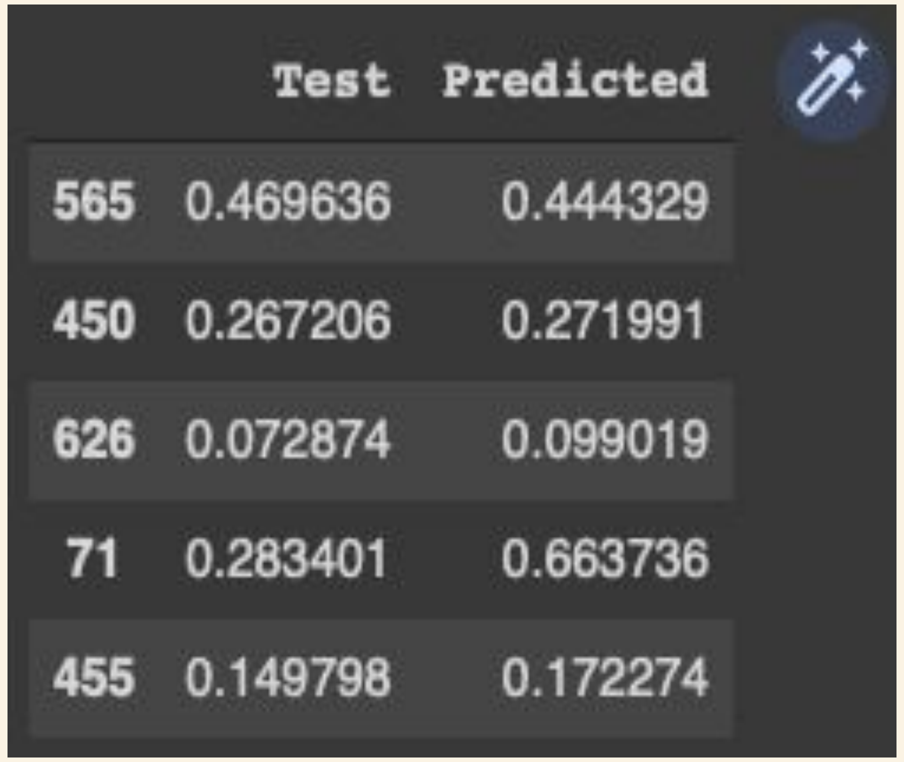
```
r2_score(y_test,yPredicted)
```

```
0.9424245798515708
```

RIDGE REGRESSION

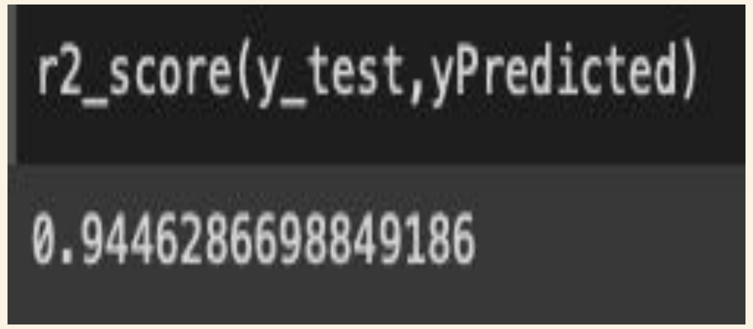
- It is a regularized form of linear regression i.e a regularization technique, used to overcome multicollinearity in linear regression.
- The main purpose of Ridge Regression is, to find the coefficients that minimize the sum of error squares by applying a penalty to these coefficients.
- Alpha is penalty term, which we determined by importing RidgeCV from sklearn library.
- It basically reduces the model complexity by coefficient shrinkage.

RIDGE REGRESSION



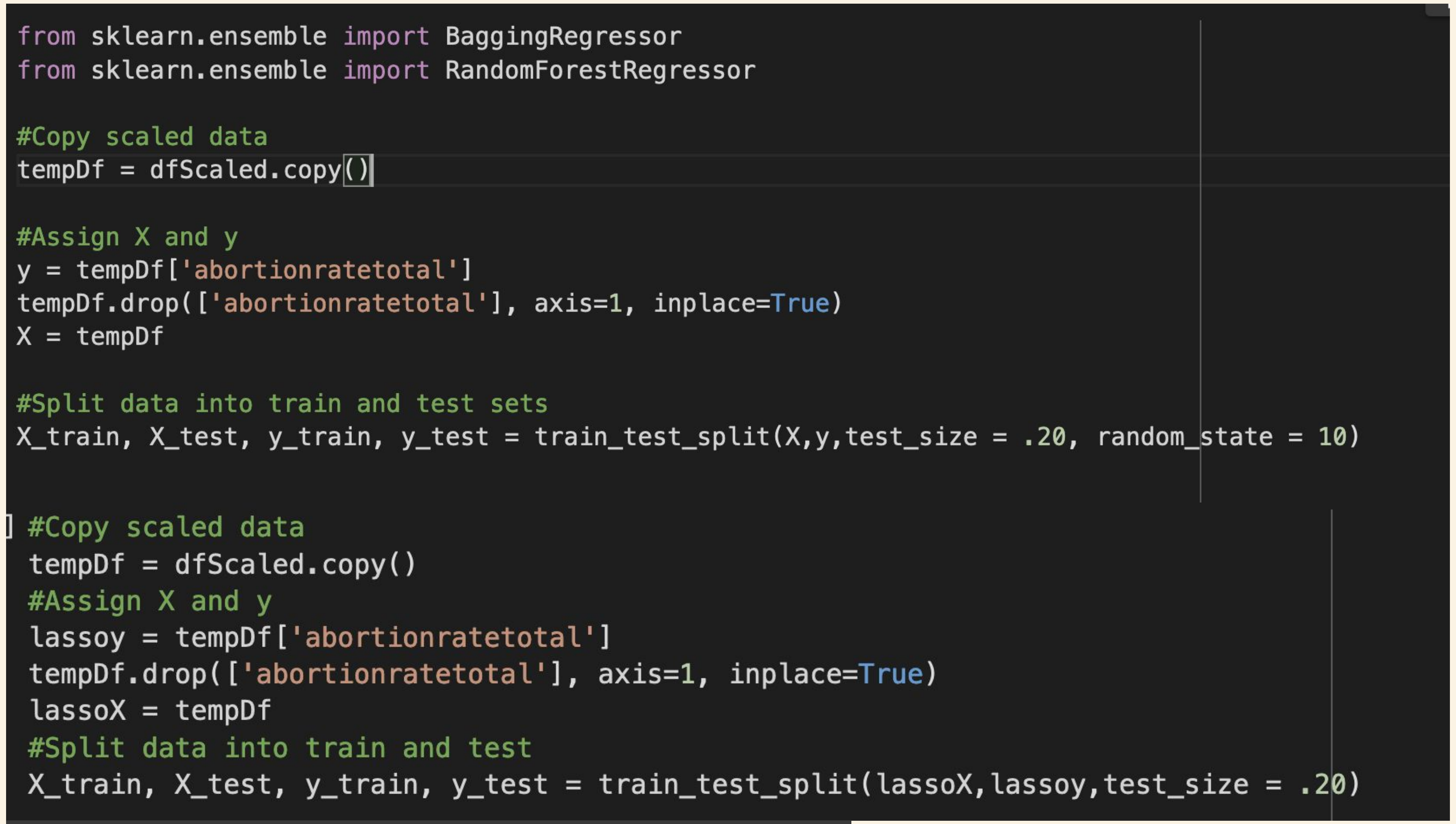
	Test	Predicted
565	0.469636	0.444329
450	0.267206	0.271991
626	0.072874	0.099019
71	0.283401	0.663736
455	0.149798	0.172274

- Results:
- RHS has our results for actual value of our test(“abortionratetotal”) dataset and value of what our model predicted.
- LHS shows our r2_score which shows our results are 94%accurate.



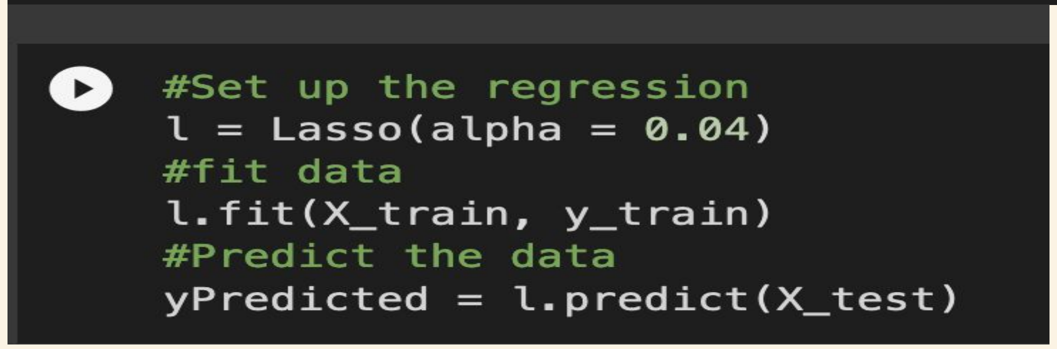
```
r2_score(y_test,yPredicted)
0.9446286698849186
```

LASSO REGRESSION

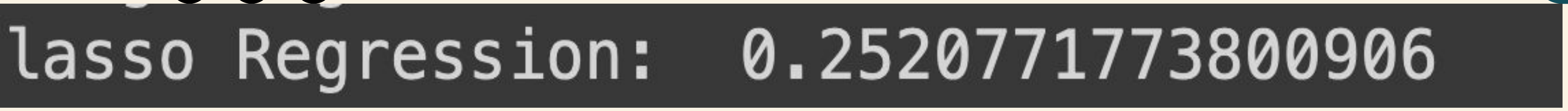


```
from sklearn.ensemble import BaggingRegressor
from sklearn.ensemble import RandomForestRegressor

#Copy scaled data
tempDf = dfScaled.copy()

#Assign X and y
y = tempDf['abortionratetotal']
tempDf.drop(['abortionratetotal'], axis=1, inplace=True)
X = tempDf

#Split data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = .20, random_state = 10)
```

```
#Copy scaled data
tempDf = dfScaled.copy()
#Assign X and y
lassoy = tempDf['abortionratetotal']
tempDf.drop(['abortionratetotal'], axis=1, inplace=True)
lassoX = tempDf
#Split data into train and test
X_train, X_test, y_train, y_test = train_test_split(lassoX,lassoy,test_size = .20)
```



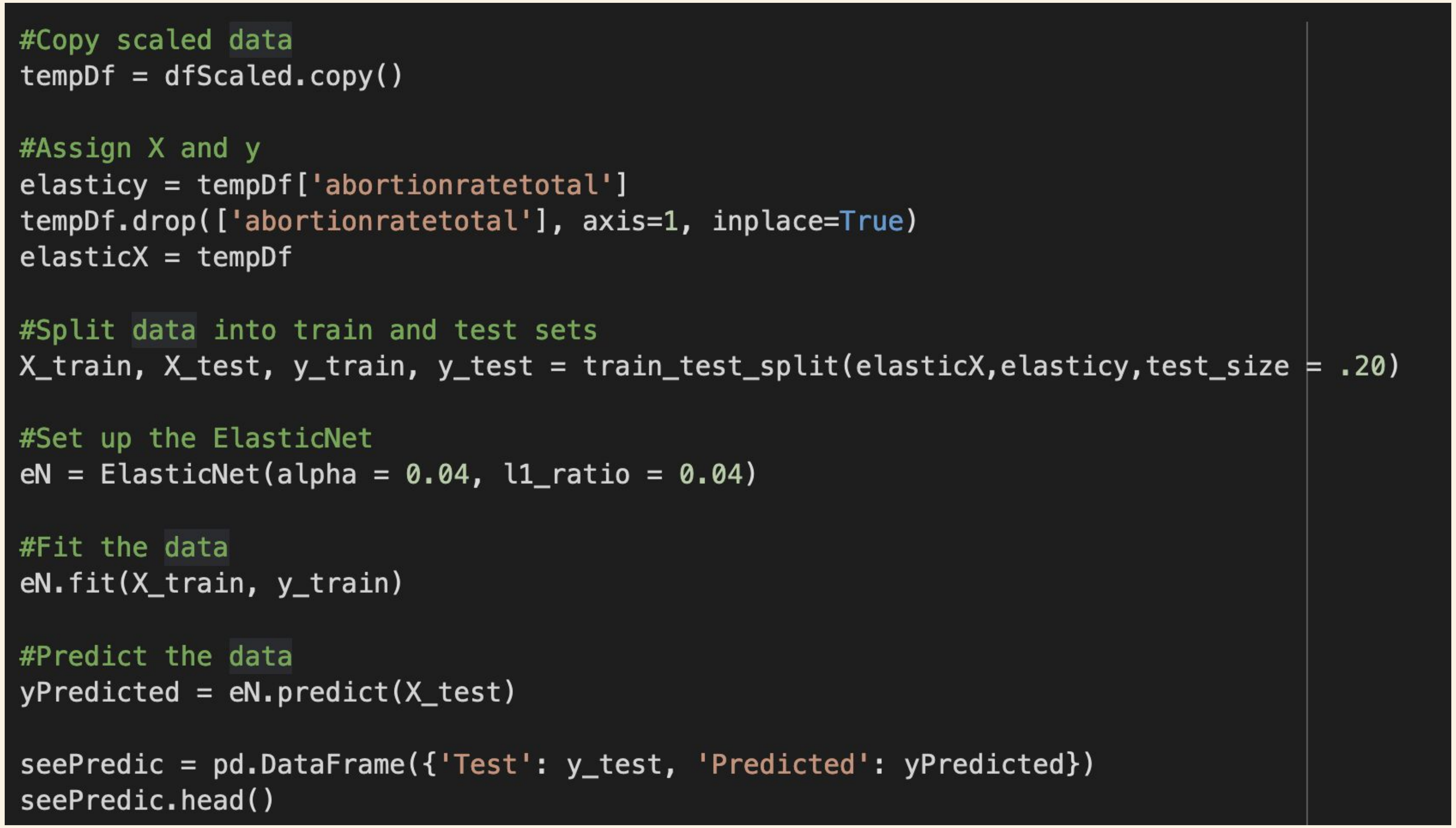
```
#Set up the regression
l = Lasso(alpha = 0.04)
#fit data
l.fit(X_train, y_train)
#Predict the data
yPredicted = l.predict(X_test)
```



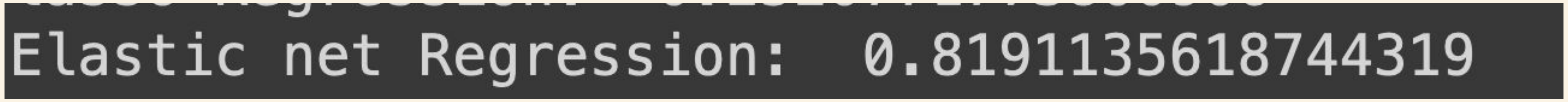
```
lasso Regression:  0.2520771773800906
```

- Lasso Regression works best when your model includes many useless variables. Lasso Regression has a L1 penalty which has the effect of shrinking the coefficient for those input variables that do not contributes much to the prediction.

ELASTIC NET REGRESSION

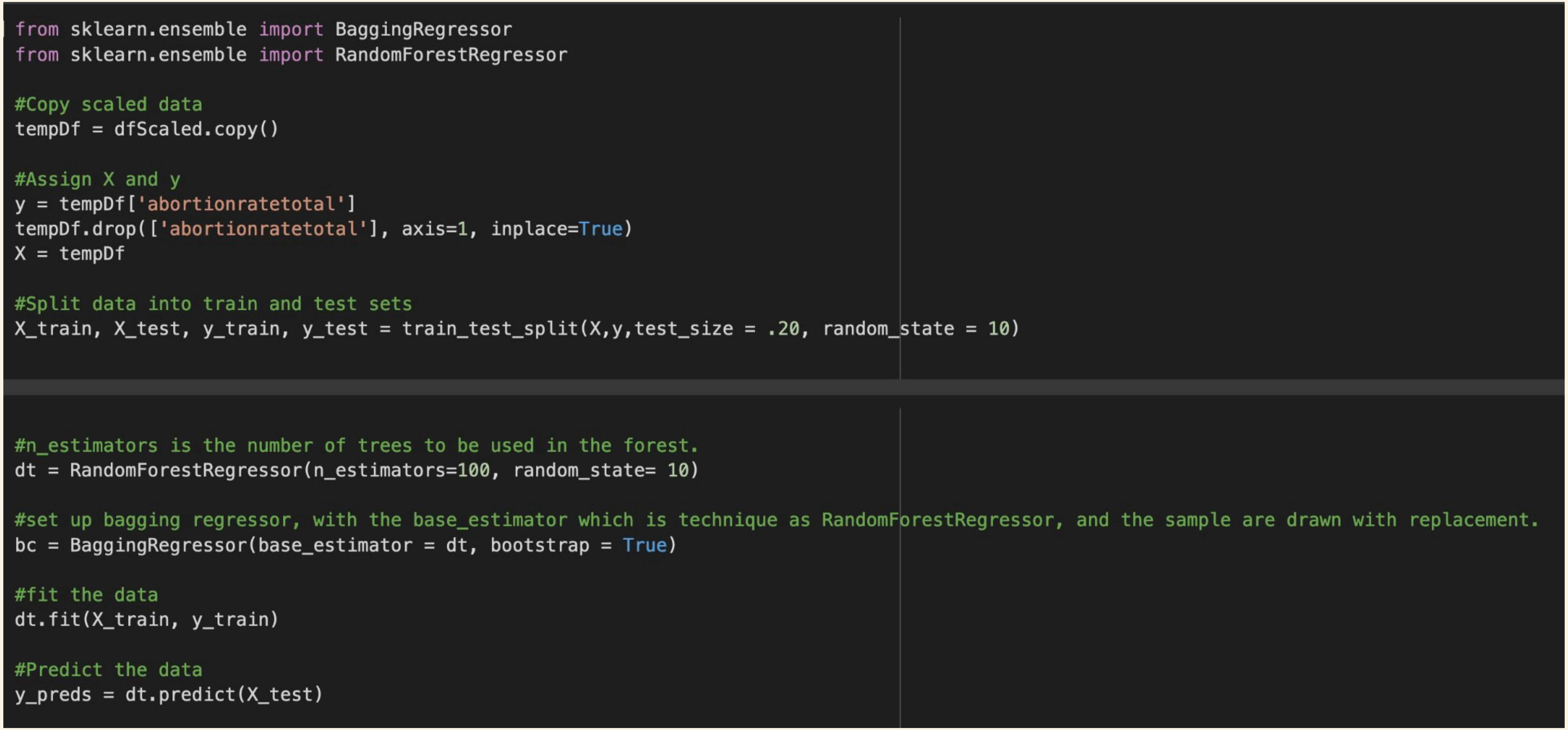


```
#Copy scaled data
tempDf = dfScaled.copy()

#Assign X and y
elasticy = tempDf['abortionratetotal']
tempDf.drop(['abortionratetotal'], axis=1, inplace=True)
elasticX = tempDf

#Split data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(elasticX,elasticy,test_size = .20)

#Set up the ElasticNet
eN = ElasticNet(alpha = 0.04, l1_ratio = 0.04)

#Fit the data
eN.fit(X_train, y_train)

#Predict the data
yPredicted = eN.predict(X_test)

seePredic = pd.DataFrame({'Test': y_test, 'Predicted': yPredicted})
seePredic.head()
```



```
Elastic net Regression:  0.8191135618744319
```

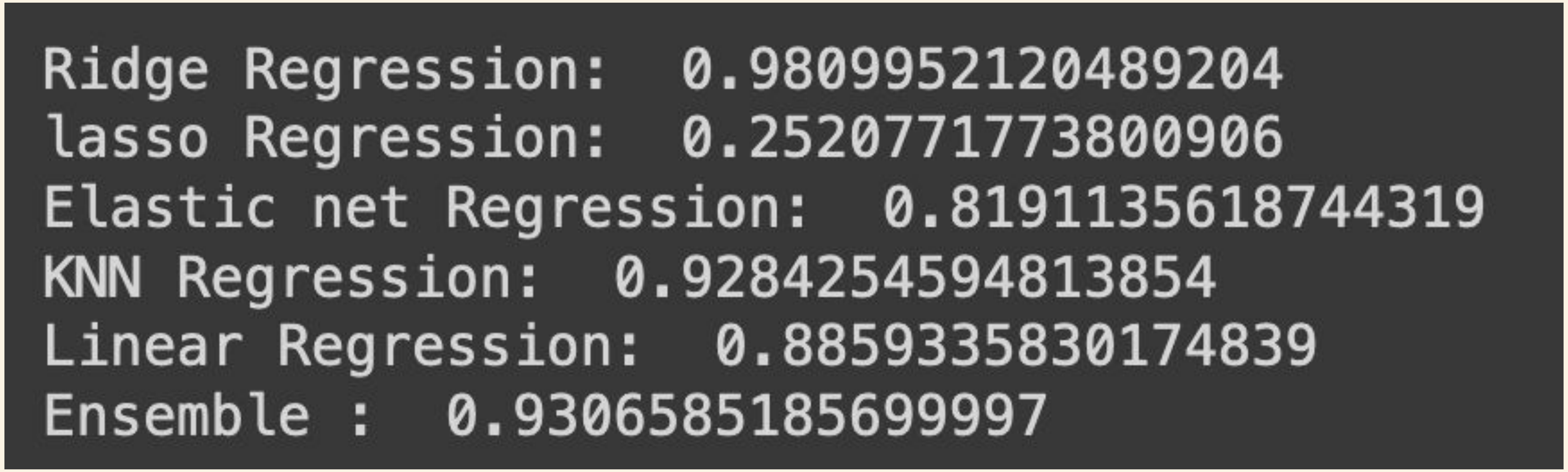
ENSEMBLE MODEL



```
from sklearn.ensemble import BaggingRegressor
from sklearn.ensemble import RandomForestRegressor

#Copy scaled data
tempDf = dfScaled.copy()

#Assign X and y
y = tempDf['abortionratetotal']
tempDf.drop(['abortionratetotal'], axis=1, inplace=True)
X = tempDf

#Split data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = .20, random_state = 10)
```

```
#n_estimators is the number of trees to be used in the forest.
dt = RandomForestRegressor(n_estimators=100, random_state= 10)

#set up bagging regressor, with the base_estimator which is technique as RandomForestRegressor, and the sample are drawn with replacement.
bc = BaggingRegressor(base_estimator = dt, bootstrap = True)

#fit the data
dt.fit(X_train, y_train)

#Predict the data
y_preds = dt.predict(X_test)
```

Conclusion For Best Model



```
Ridge Regression:  0.9809952120489204
lasso Regression:  0.252077177380906
Elastic net Regression:  0.8191135618744319
KNN Regression:  0.9284254594813854
Linear Regression:  0.8859335830174839
Ensemble :  0.9306585185699997
```

By comparing our regression models we concluded that the ridge regression model ,which had the r-squared value of 98%, did the best in predicting the data. Therefore, we think the ridge regression model is the best model among the 6 models for our dataset

CONCLUSION

Our findings cannot prove that red states have a lower abortion rate than blue states. While some of the data does point in this direction, the correlations are too small to be regarded with any level of statistical significance.

References

https://en.wikipedia.org/wiki/Abortion_in_Alaska
https://www.guttmacher.org/united-states/abortion
https://www.nga.org/

Further Exploration

A couple of interesting points to explore would be the differences in sex education and availability of contraception, as well as as well as potential religious factors. Furthermore, since the reversal of Roe v Wade just recently, there will likely be a noticeable shift in the data in the coming months and years that should be of great interest and importance to study. Abortion is a complicated issue and it cannot be boiled down to party affiliation alone.