Shawn John
Furqaan Ali
Bhavana laxmi Radharapu

CS 418 Project 2 Report

Task 1

To partition the merged dataset into a training set and validation set, we used the holdout method using the train_test_split function from the scikit-learn model_selection library. We used 80% of the merged dataset for the training set and the remaining 20% for the validation set. The x sets (x_train, x_val) included most of the features from the merged_train dataset. 'Democratic', 'Republican', and 'Party' were dropped from these sets since these are response variables and labels rather than predictor variables. 'State', 'County', and 'FIPS' were also dropped since they would not play a role in building or training the models. The y sets (y_train, y_val) only included 'Democratic', 'Republican', and 'Party' since these are the features we are trying to predict.

Task 3

When building a linear regression model for predicting the number of votes cast for the Democratic party in each county, we first considered the variables we deemed most important from project 1. These variables were 'Total Population', 'Percent White', 'Percent Black', and 'Percent Less Than Bachelors Degree'. The model built from these variables was quite good, with an R^2 value of 0.9485. Next, we tried removing different variables to see if any of these are not too significant for predicting democratic votes. After removing 'White Percent', the corresponding model had an R^2 value of 0.9491, indicating that 'White Percent' is not too relevant for this model. Finally, we added multiple variations of multiple variables and were able to achieve an R^2 value of 0.9495 using a total of 6 variables. However, because the adjusted R^2 value would likely be lower and the increase in R^2 was not significant given that we doubled the number of variables being used to build the model, we concluded that the best performing linear regression model for predicting the number of democratic votes is the model built using predictor variables 'Total Population', 'Percent Black' and 'Percent Less Than Bachelors Degree' with an R^2 value of 0.9491 and coefficient values of 69073, 1700, and -8948, respectively.

When building a linear regression model for predicting the number of votes cast for the Democratic party in each county, we also first considered the variables we deemed most important from project 1 (variables mentioned above). However, the R^2 value we achieved with this was only 0.6475, so we decided to add combinations of other variables that seem relevant specifically to Republican counties, such as 'Percent Age 65 and Older' and 'Percent Rural' and received an R^2 value of 0.6737. After removing some variables, we were able to build a model using only three variables ('Total Population', 'Percent White', 'Percent Rural') with an R^2 value of 0.6693. Although this wasn't the highest R^2 value we were able to achieve, we deemed this to be the best performing model for predicting the number of Republican votes because it's R^2 value was very close to the highest achieved (0.6948 using all 13 predictor variables) while using far fewer variables. Therefore, after considering the number of variables and the minute difference in R^2 values, we concluded that the best performing linear regression model for predicting the number Republican votes is the model built using predictor variables 'Total Population', 'Percent White', and 'Percent Rural' with an R^2 value of 0.6693.

Task 4

To predict which party the majority of the county will vote for we built several classification models. The three types of models we used are decision trees, K-nearest neighbors, and SVMs. We first decided to train each of these models on the variables that were weighted the most from project 1. The first model we created was a decision tree using Entropy as the criterion. Then we trained another decision tree but this time where the Gini Index was used to measure the impurity of nodes. Both models had an accuracy of 76%. However, the Entropy model had a F1 score of 58%, which is 5 % more to the Gini Index model's F1 score of 53%. So for decision trees the best parameter for the criterion was Entropy.

The next model we used was K-nearest neighbors. The first thing we did was create many different K-nearest neighbors models, all with different K values. The model with the highest accuracy would then be stored. In this case the most optimal K value was 3. The 3 nearest neighbor model did significantly better than the decision tree, with an 83% accuracy and 67% F1 score.

The final model we wanted to optimize was the SVM model. The first thing we did is to find the best kernel. So we trained all the models with different kernel functions. The best kernel was radial basis function which had the highest accuracy and highest F1 score compared to the other models with different kernels. After we found the most optimal kernel, we also wanted to optimize the c which is the regularization parameter for SVMs which allows misclassification with a penalty where the penalty is based on the value of c. We trained several SVMs with rbf as the kernel, with c values ranging from 0.001 up to 10. The best c value found based on accuracy was 5.0. So the most optimal SVM is with a radial basis function for the kernel and a c value of 5.0.

Now that we found our most optimal models it was now time to see if different combinations of variables would make our models better. The next combination of variables we trained our models in was Percent under age 25,median household income, percent unemployed, percent less than bachelor's degree, percent rural. With these combinations of variables we then trained on the most optimal models found before. The new decision tree was worse with only 73% accuracy and 52% F1 score compared to the original set of variables decision tree which had a 76% accuracy and a 58% F1 score. The 3 nearest neighbors models also did considerably worse using the new combination of variables. With an accuracy 8% less than the original 3 nearest neighbor model and a 22% decrease in the F1 score compared to the original model. Even the SVM performed worse than its original model. The fact that all three models when trained on these sets of variables compared to the original set of variables. Shows that these new set of variables are not as good as predicting party labels compared to the original set of variables.

The final combination of variables we wanted to try was all of the variables. We wanted to see if adding all of the variables to our models will provide us with some extra insight. Our initial expectations were that each variable was useful in predicting the party label. Sure some of them will be weighted more than others, which we found in project 1, but we initially thought that each variable will provide some useful information that will allow the models to predict party labels better. Now both k-nearest neighbors and decision tree models trained on all variables did worse compared to the models trained on the original set of variables. However, the SVM model did not change much compared to its original model. The accuracy increased by around .4% and the F1 score decreased by about 1%. So not that much change. The fact that decision trees and k-nearest neighbors did worse and SVM had similar results when trained on all

the variables. Shows that some of these variables are not helping the models at all and are in fact adding noise to the data which makes it more difficult to predict the party label.

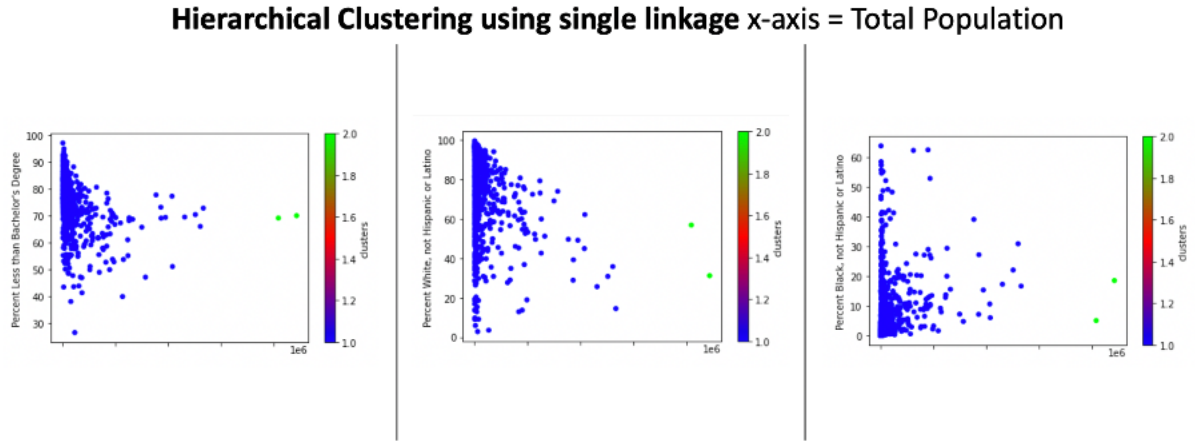
The best model was K-nearest neighbors trained on the set of variables found in project 1. The best k value was found by training the model on the original set of variables and choosing the model with the highest accuracy. This set of variables is the best to train on because the other combination of variables made our models to perform worse and training on all the variables also makes the models perform worse or in the case of the SVM model doesn't change the results. The 3-nearest model trained on variables 'Total Population', 'Percent White', 'Percent Black', and 'Percent Less Than Bachelors Degree' had one of the highest accuracy of 83% and the best F1 score of 67%. Even though the SVM trained on all variables had an accuracy of 84% its F1 score was considerably less with a value of 60%. So the best classification model for the data is K-nearest neighbors where K = 3.

Task 5

For our clustering analysis, we chose the variables that had a major impact in project 1 and we are trying to analyze if they have any cluster or any pattern that is advantageous for our analysis. The variables used are Total Population, Percent White, not Hispanic or Latino, Percent Black, not Hispanic or Latino, Percent Less than bachelor's degree. We are going to explore the data using the three clustering techniques: K-Means, DBSCAN and Hierarchical clustering. We will look at the output of different analyses below.

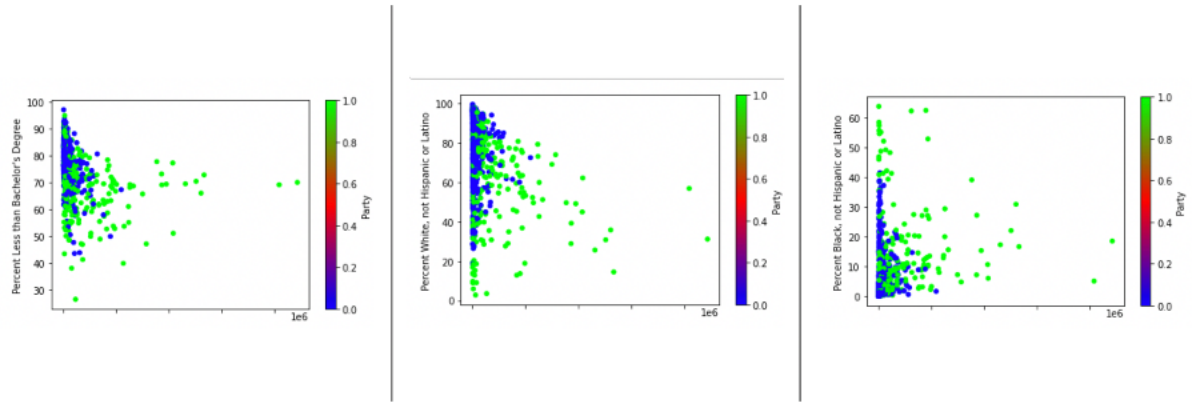
Hierarchical clustering

First, we looked at Hierarchical clustering using single linkage with two clusters, and distance calculated using Euclidean.



As we see above, we do not have any good clusters formed as our data is a real live data and it is very unlikely that is forms perfect cluster and if we look at our adjusted rand index which is 0.0056, very low and shows how our clusters are very different and our silhouette coefficient is 0.953 which tells us that this clustering method is really good for our data.

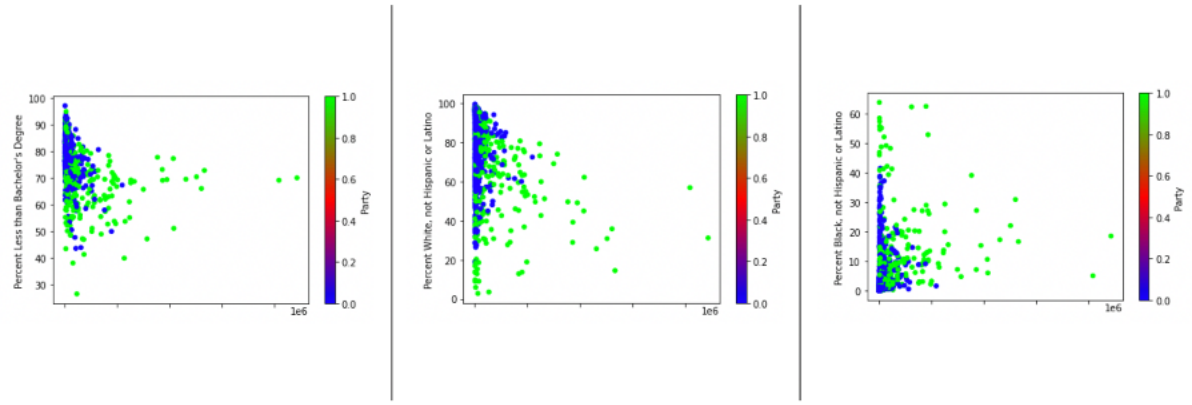
Hierarchical Clustering using complete linkage x-axis = Total Population



For our second clustering, we used complete linkage and our output didn't seem to change very much and also our rand index value remained the same. But the data in our graphs look so much different.

K-Means Clustering:

K Means Clustering x-axis = Total Population



For K-Means clustering we used 2 clusters and 1 iteration for our first case, and the rand index of is higher than hierarchical clustering which shows that our clusters are more similar, but silhouette coefficient is lower than Hierarchical clustering. For our second case we tried increasing the number of iterations to see if there would be any changes in our clustering. But we haven't really seen any noticeable difference, but our outputs remain the same. Rand index is 0.119 which is not very different from hierarchical clustering and the silhouette coefficient is 0.90 which is lower than Hierarchical clustering.

Task 6

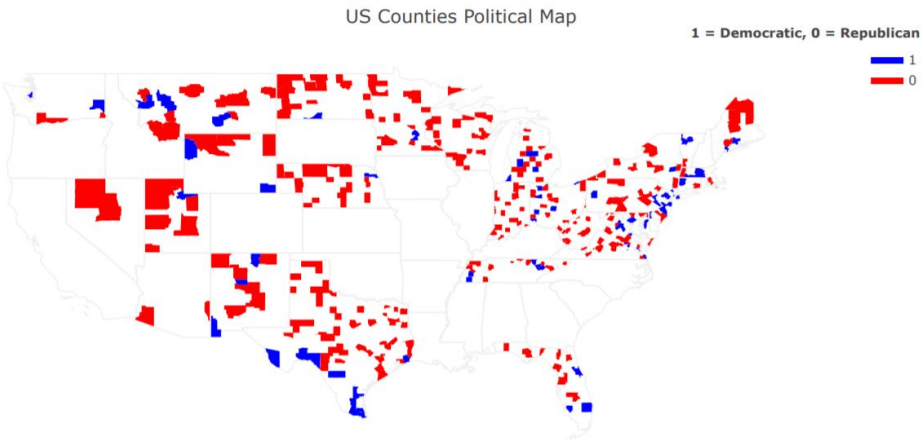


Figure 1

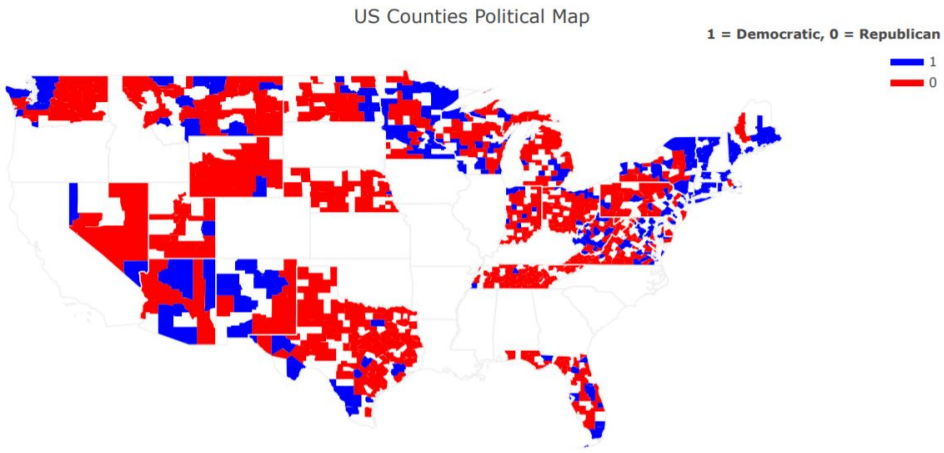


Figure 2

Figure 1 shows the predicted party label on the demographic_test set from the prediction of K-nearest neighbors model, with K =3, trained on the entire merged dataset. The Figure 2 shows how the counties voted in the merged dataset. Figure 1 fills in the states who have some counties labeled but other counties are unlabeled, meaning they are shaded in as white. For example in Nevada near the top region there is a county that is labeled white in Figure 2. However, Figure 1 has predicted that county to vote republican. If we combine the two figures some people might try to draw conclusions for how the state as a whole will vote. Again we use Nevada as an example, if we take figure 2 and combine it with figure 1 it will show that most of Nevada is shaded as republican. But Nevada doesn't necessarily have to vote as republican. Most of Nevada's population resides in the two blue counties. So we can't draw any conclusions on how the state as a whole will vote because counties don't contain the same amount of people. What we can conclude is what the majority of counties will vote in a given state. For example Texas, most of the counties in Texas are republican in Figure 2 and most of the counties in Texas are predicted to be republican in Figure 1. So we can conclude that the majority of counties in Texas will vote republican. So the plots let us conclude how the majority of counties will vote on the states whose counties are labeled.Figure 1 predicts more counties to be republican than democratic based on the information of those counties. So if we combine both Figures we can see that in the subset of states most counties will vote republican.

Task 7

The Output.csv holds the number of predicted votes, of a county, for both republicans and democrats using the best linear regression models for each party. Then uses K-nearest neighbors with K =3 to classify each county as either republican or democratic. Now some of the predicted votes are negative because linear regression models are not bound to non-negative numbers. So there will be some combination of variables that result in a negative number of predicted votes. An interesting thing to see will be how aligned our classification and linear regression models. Meaning will the classification model label a county as democratic and the linear regression model gives more votes to democrats and vice versa. If we look at the Output.csv and compare the linear regression models outputs with the classification models outputs we see that 297 of them match. Meaning that for 297 of the predictions the linear regression models gave more votes to democrats and the party was labeled democratic and predictions that gave more votes to republicans were labeled republican. So for 297/400 = 74% predictions the models came to the same conclusion. The disparity could be the result from how the classification models weighted the variables in the data could be different to the weights/coefficients in the linear regression models.