Aid Allocation

Predicting poverty in Costa Rica

Business Problem

- Lack of econometric data in developing countries makes it difficult for donors to assess and prioritize needs
- Apart from econometric data, donors are curious about the underlying factors that contribute to poverty
- As a step towards a solution, the Inter-American Development Bank has created a non-econometric dataset for Costa Rica
- The features in this dataset relate to household size and education level rather than financial information

Solution

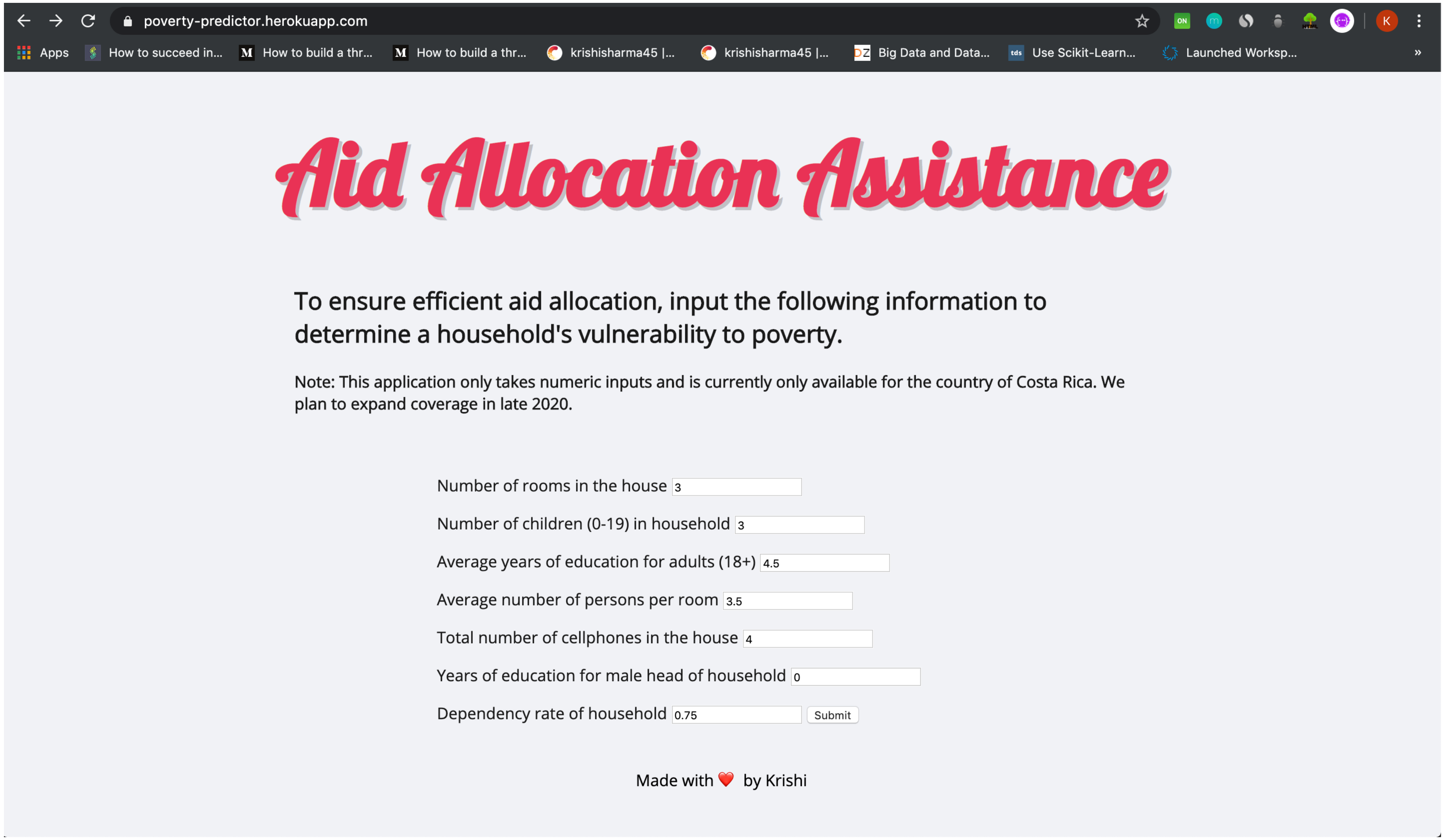
- To help donors, I’ve created a machine learning model using the Inter-American Development Bank’s dataset

- The dataset classifies households on their vulnerability to poverty on a scale of 1 to 4, with 4 being ‘Not Vulnerable’

- A full machine learning pipeline was created to test classification algorithms and a web app designed for donors was developed

- Deep learning techniques were explored, however due to accuracy, was not implemented in the final product

Final Product: App



Aid Allocation Assistance

To ensure efficient aid allocation, input the following information to determine a household's vulnerability to poverty.

Note: This application only takes numeric inputs and is currently only available for the country of Costa Rica. We plan to expand coverage in late 2020.

Number of rooms in the house 3

Number of children (0-19) in household 3

Average years of education for adults (18+) 4.5

Average number of persons per room 3.5

Total number of cellphones in the house 4

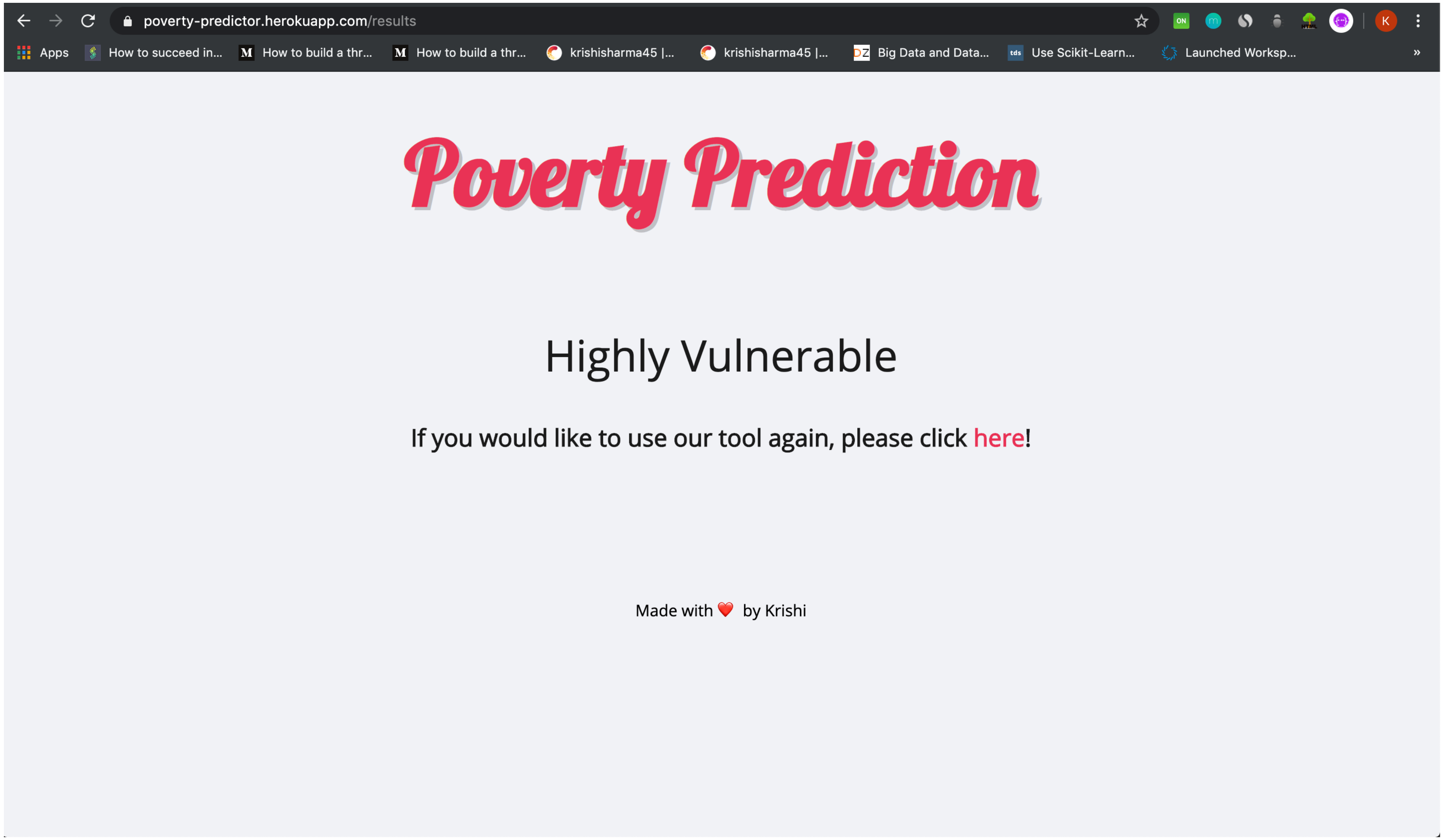
Years of education for male head of household 0

Dependency rate of household 0.75

Made with ❤️ by Krishi

This application is hosted on Heroku. Anyone can enter information related to a specific house (currently only available for Costa Rica), submit it.

App Predication



Poverty Prediction

Highly Vulnerable

If you would like to use our tool again, please click here!

Made with ❤️ by Krishi

After submission, the application returns the model's prediction in a user friendly format. This information can be used to help the donor prioritize aid.

Approach

- To create this machine learning application, I took the following steps:

  1. Cleaned and explored the dataset

  2. Created a machine learning pipeline

  3. Optimized parameters for best algorithm

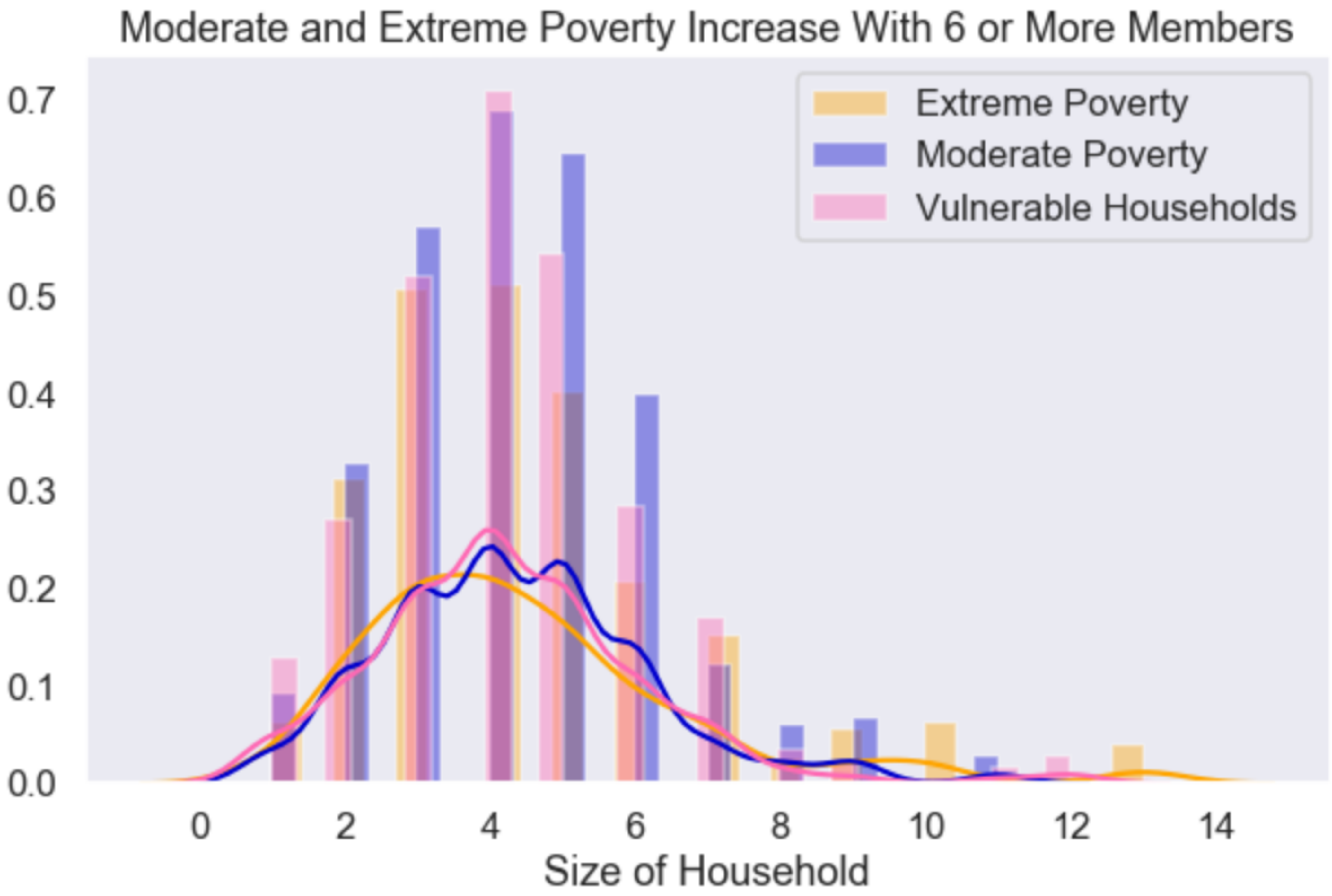
  4. Created app that used model to classify poverty level

Exploratory Data Analysis

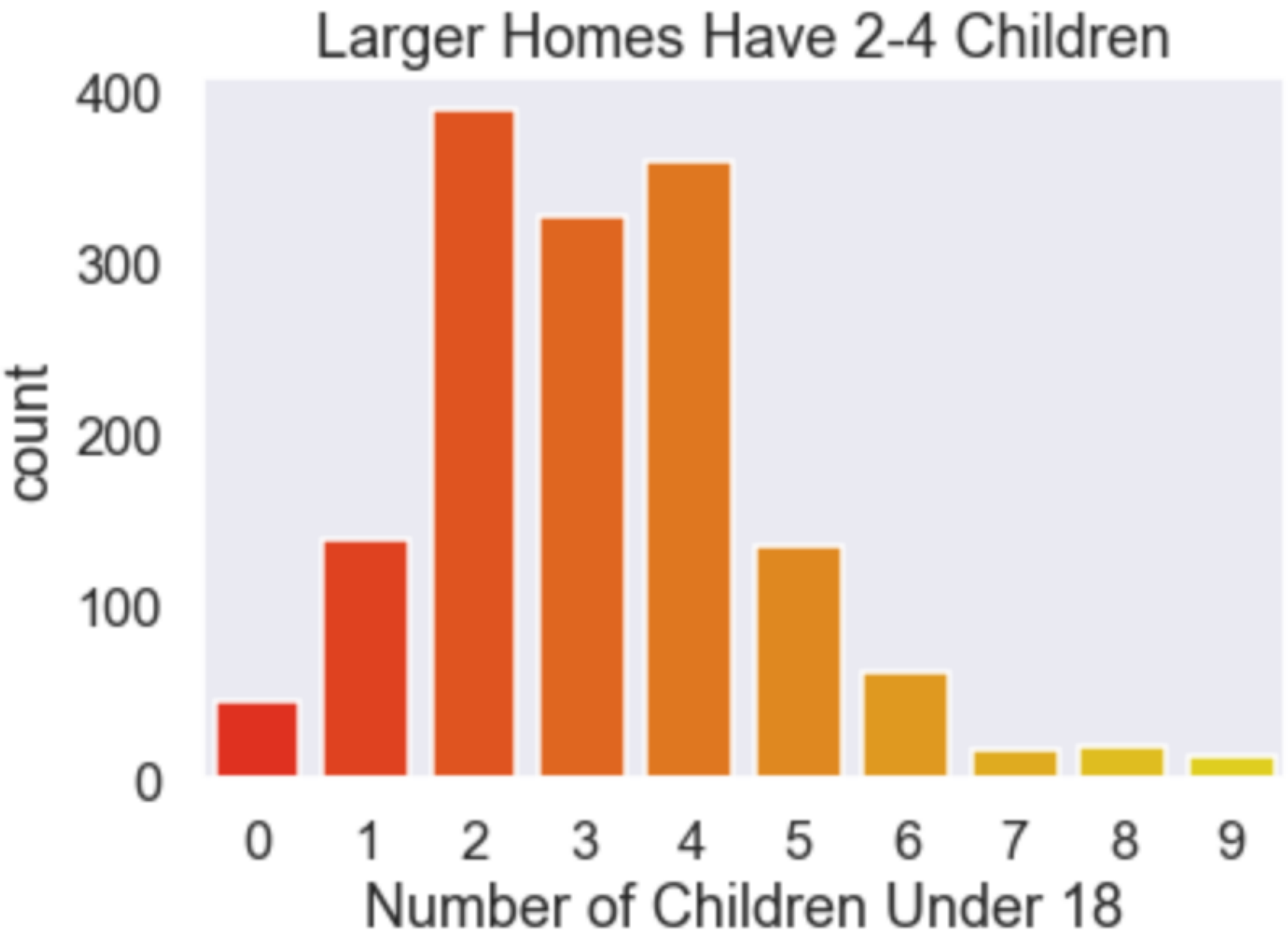
- To explore the dataset, I removed null values, insignificant non-numeric values and converted significant string values to numeric values

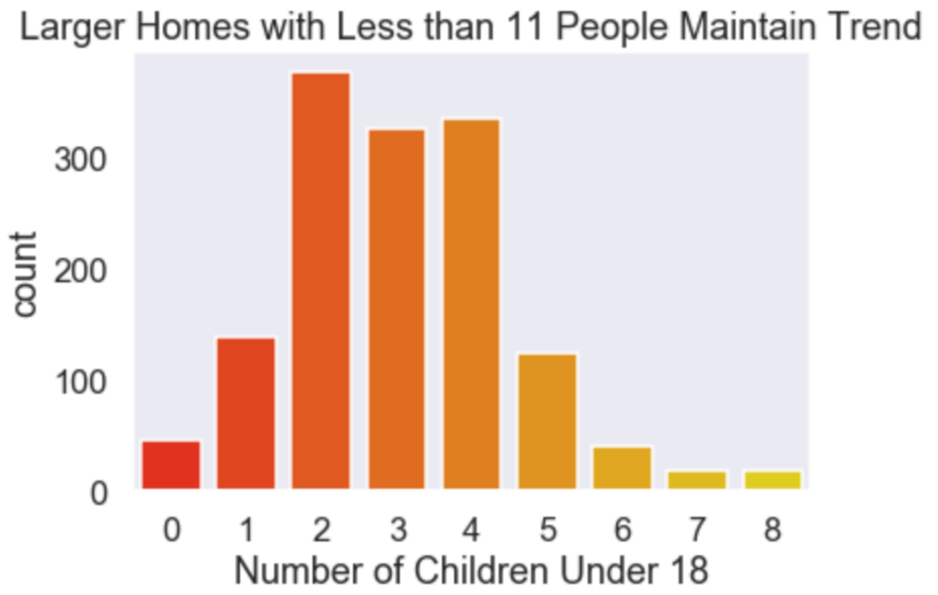
- I produced useful visualizations in Python using the matplotlib and seaborn libraries.

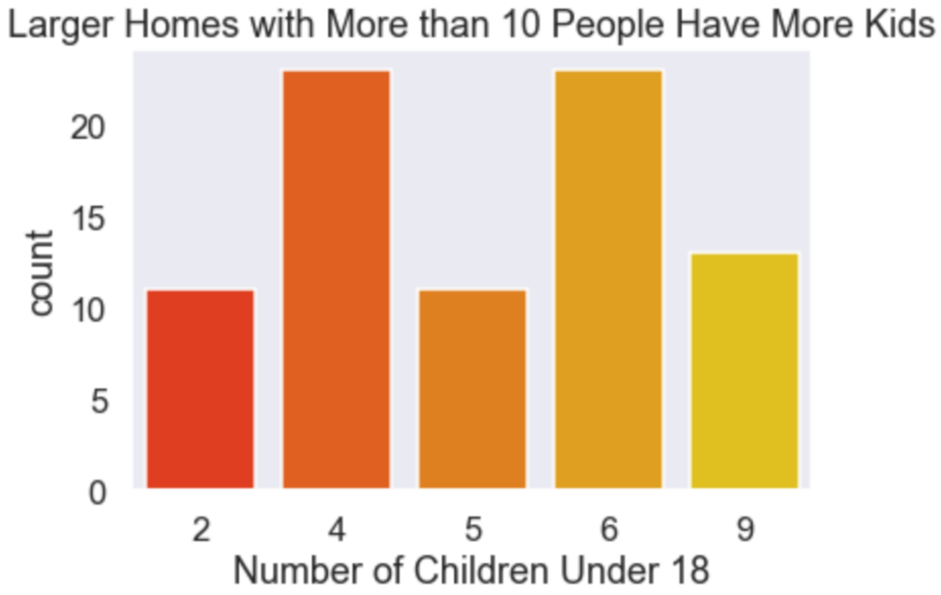
- These visualisations showed that the number of children in a household, average education level of the household and household size correlated with certain levels of poverty



The size of the household directly correlates with vulnerability to poverty.
Extreme poverty outpaces the less severe forms of poverty at 10 people.







Larger households tend to have more kids.
This means that there are less people contributing income towards rent.

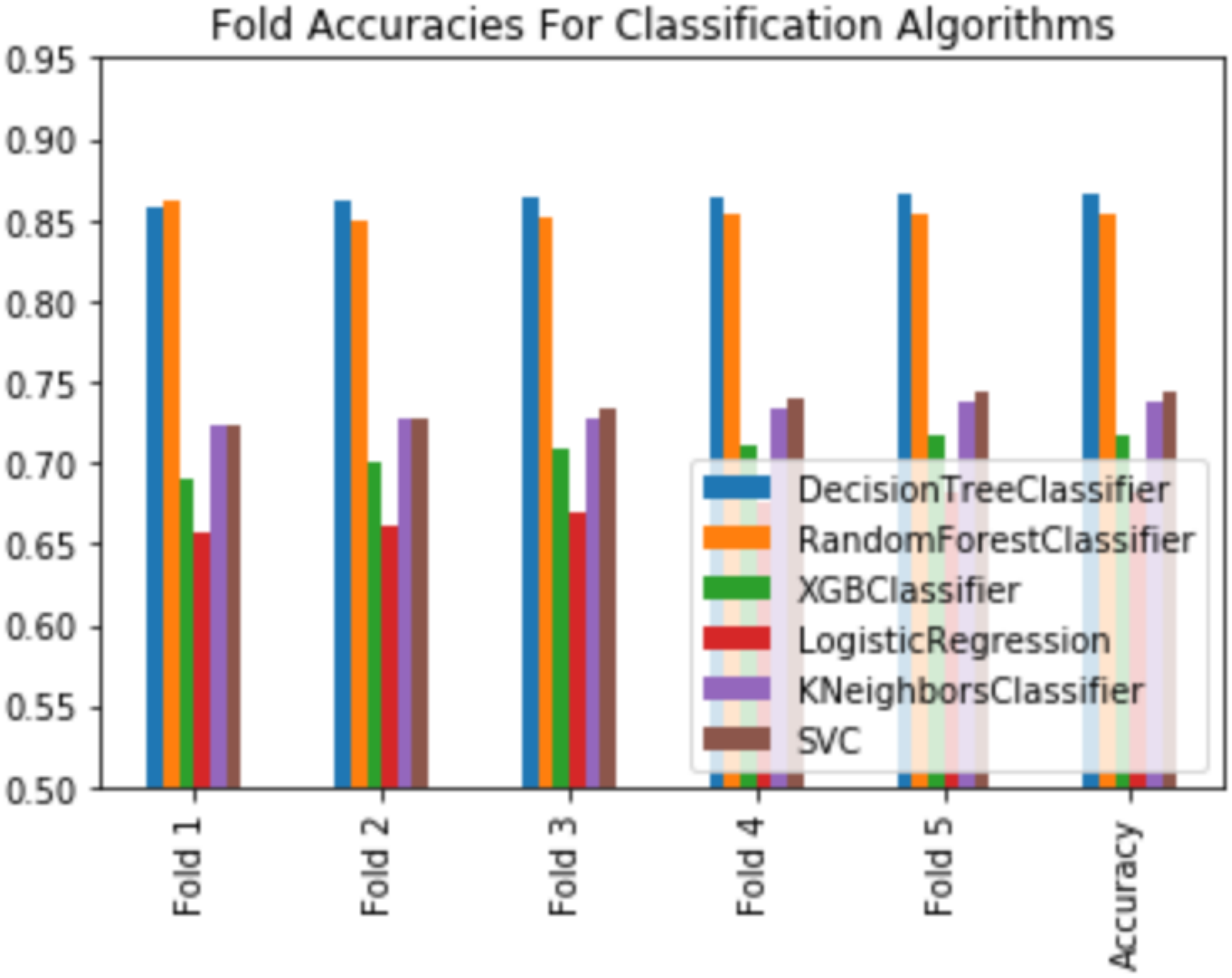
Machine Learning Pipeline

- After gaining some insight behind the numbers, I created a machine learning pipeline.

- I tried a few popular classifiers and cross validated each classifier across 5 folds.

- The accuracy was then averaged across the 5 folds for each classifier.

- The classifiers with the highest accuracy, as seen in the next slide, were Decision Tree and Random Forest.


Fold Accuracies For Classification Algorithms
0.95
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0.55
0.50
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Accuracy
DecisionTreeClassifier
RandomForestClassifier
XGBClassifier
LogisticRegression
KNeighborsClassifier
SVC

Hyperparameter Optimization

- Having nailed down the two algorithms I wanted to explore, I created a hyperparameter optimisation pipeline to determine which select hyperparameters made the algorithm perform best given this dataset.

- The GridSearchCV Python library was used, as well as the CrossValidation library, to perform this pipeline search.

- Random Forest had a higher accuracy after this hyper parameter optimisation search was conducted.

Random Forest Classifier

- Random Forests are collections of decision trees whose results are aggregated into one final result

- Limit overfitting without substantial increasing error makes them powerful

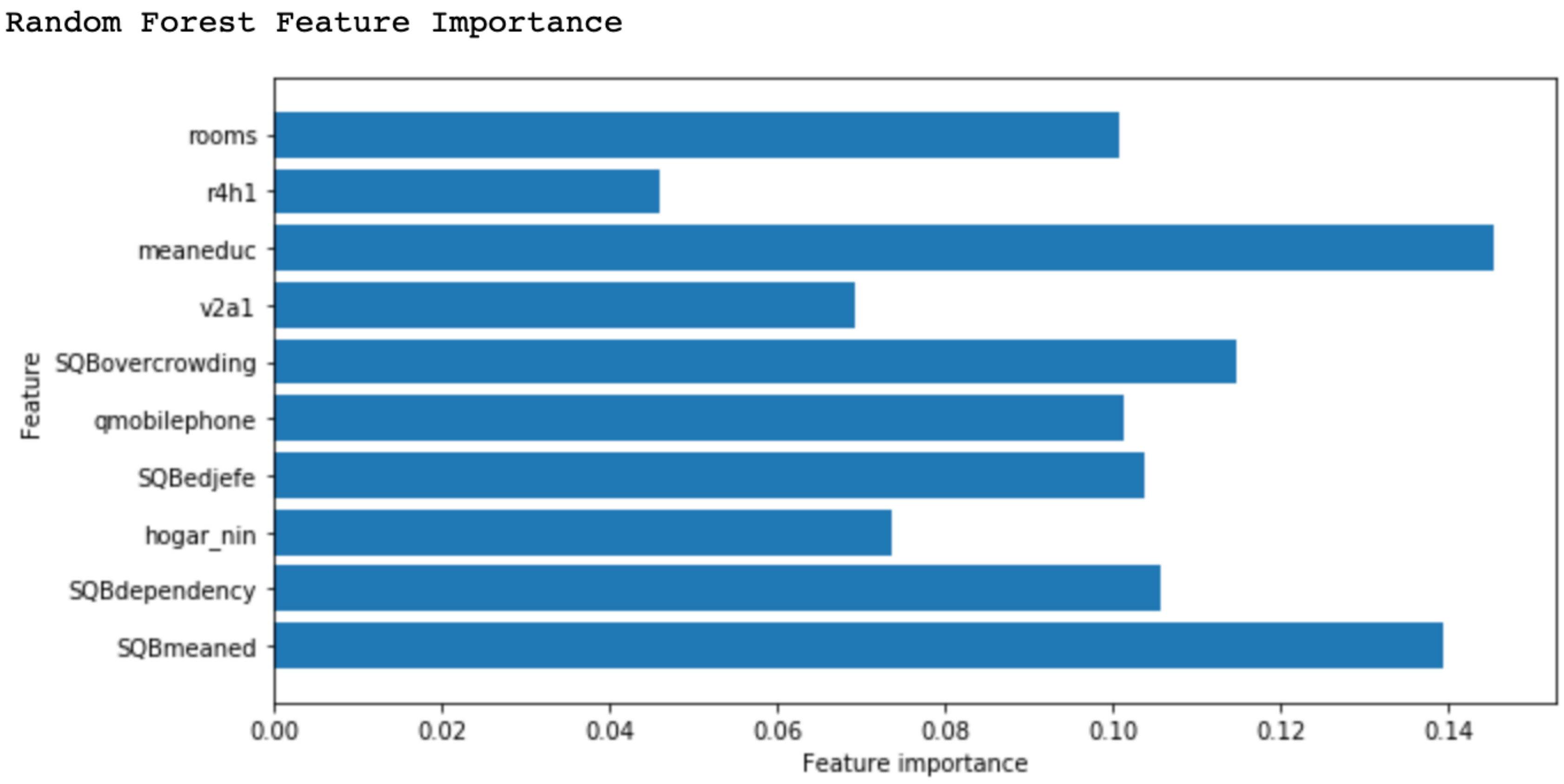
- They train on random samples of and random subset of features; using many trees in forest provides strong result while reducing overfitting

- More robust than a single decision tree

- My final random forest classifier had 350 trees with a maximum depth of 15 for each tree

Feature Importance

To further improve the accuracy of the Random Forest classifier, I saw which features were the most important in determining the class



Final Classifier

- The final classifier used these ten features from the dataset to train the optimised Random Forest classifier

- The training score was 96.47%

- The testing score was 93.36%

- The difference between the two scores was 3.11%

- This final classifier was scaled using Dask for future performance enhancement and saved in a pickle file

Deep Learning Approach

- Artificial Neural Networks were created to see how deep learning compared to traditional machine learning
- 4 layers (1 input layer, 2 hidden layers and 1 output layer) were created with an input dimension of 10
- Keras and Tensorflow were the main Python packages used to create the model
- I tried different values for the loss functions, activation functions and optimiser to obtain the best accuracy
- Final training accuracy was 65.05%, testing was 64.54%

Application Deployment

- To create the application, I used Flask and integrated the final machine learning classifier (Random Forest) into the backend

- The frontend consists of a basic HTML form that captures the user's information related to the features that the model uses for prediction (10 in total, 7 user entered)

- The app.py file runs calculations (squares certain numbers) from the user inputted data to get to the 10 features that flow into the model

Future Enhancements

- Phase II (a) of this project consists of tailoring feedback based on which of the four classes the predicted result turns out to be.

- Phase II (b) would be to create a login for users so that they could run the model on several different households. They could get a dashboard which priorities which households should receive aid first, and even how much aid (given a certain dollar amount that the donor has to offer in total donations across all households) based on the predictions given by the model

- Phase III includes expanding coverage of the model to different countries and including features specific to certain areas