STAT 486: Final Report

Madison Wozniak

Table of Contents

Introduction

Problem Statement

Food affordability in the Unites States has reached an all-time high in recent years, and is a struggle that many individuals and families are no stranger to. This is a critical issue for single mothers in particular, as it impacts their household's stability, health, and overall quality of life. In California especially, the average cost of living remains just barely achievable for many and women-headed households face unique economic challenges because of it. The ability to understand and predict the median income of women-headed households in California can provide valuable insights into financial realities, and help identify common trends.

Aim of this Report

This report aims to understand if food affordability and socioeconomic factors are accurate tools in predicting the median income of women-headed households in California. By looking at data that includes information on food costs, regional demographics and economic indicators, this analysis seeks to accomplish the following:

1. Understand the relationship between income and food affordability.
2. Develop predictive machine learning models on median income.
3. Address how these insights can influence policy changes and improve economic outcomes in the future.

Exploratory Data Analysis

To fully understand the important factors that may influence median income, we must first look at the raw data. This section gives an overview of the key variables by identifying relevant distributions and variations within the data. These summary statistics act as a foundation for deciding what potential predictors of median incme could be.

The dataset includes the following features:

- `race_eth_name` (string) Name of race/ethnic group
- `geotype` (string) Type of geographic unit place (PL), county (CO), region (RE), state (CA)
- `geoname` (string) Name of geographic unit
- `county_name` (string) Name of county the geotype is in
- `region_code` (string) Metropolitan Planning Organization (MPO)- based region code
- `cost_yr` (numeric) Annual food costs
- `median_income` (numeric) Median household income
- `affordability_ratio` (numeric) Ratio of food cost to household income
- `LL95_affordability_ratio` (numeric) Lower limit of affordability 95% confidence interval
- `UL95_affordability_ratio` (numeric) Upper limit of affordability confidence interval
- `rse_food_afford` (numeric) Relative standard error of affordability ratio
- `CA_RR_Affordability` (numeric) Ratio of affordability rate to California affordability rate
- `ave_fam_size` (numeric) Average family size - combined by race

With the variables defined above, the next step is to delve into the statistical properties of each, which includes their underlying distributions and variations. It is important to be aware of any potential outliers or missing data before selecting predictive features for future models.

	reg_code	cost	income	afford_rat	LL95	UL95	rse_afford	afford_decile	CA_RR_Afford	ave_fam_size
count	295362	264366	101836	101836	99445	99445	99445	26715	101836	280593
mean	11.388	7602.632	38038.998	0.325	0.113	0.769	53.362	5.804	1.219	3.294
standard deviation	3.341	1435.062	27050.591	0.42	0.116	3.599	174.806	2.685	1.578	0.636
minimum	1	3095.425	2500	0.021	0	0.041	0.684	1	0.08	1.36
Q2	9	6667.128	22417	0.153	0	0.229	12.574	4	0.576	2.88
median	14	7460.84	33103	0.227	0.092	0.351	25.596	6	0.852	3.25
Q4	14	8325.618	46418	0.349	0.174	0.561	53.824	8	1.312	3.61
maximum	14	16872.05	250000	4.852	0.851	108.784	5227.124	10	18.214	7.2

Table 2: Numeric Data Summary Satistics

Tables 2.1 and 2.2 act as a good baseline, but sometimes it can be easier to understand relationships with graphs. Below is a correlation matrix that depicts the strength of correlations between features.

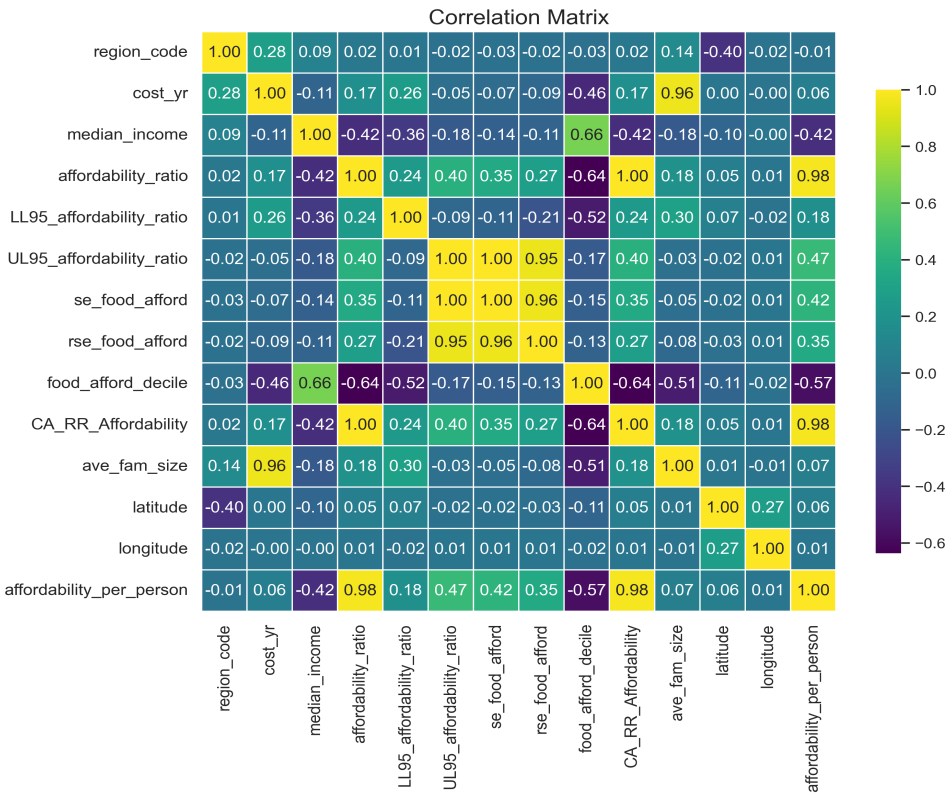

Figure 1: Correlation Matrix for Numeric Features

It can be noted from the matrix that there is a strong, almost perfect correlation between several variables. This is expected because many variables such as `UL96_affordability_ratio`, `LL95_affordability_ratio`, `se_food_afford`, `rse_food_afford`, `CA_RR_Affordability` and `affordability_per_person`are all calculated from information taken from `affordability_ratio`. Some more interesting strong correlations to take note of are between `cost_yr` and `ave_fam_size`, `med_income` and `food_afford_decile`. This is helpful because we can expect `food_afford_decile` to be an important feature in the model building process.

While not overwhelmingly apparent in the correlation matrix, there is also an important relationship between `median_income` and `ave_fam_size` that Figure 2 below illustrates.

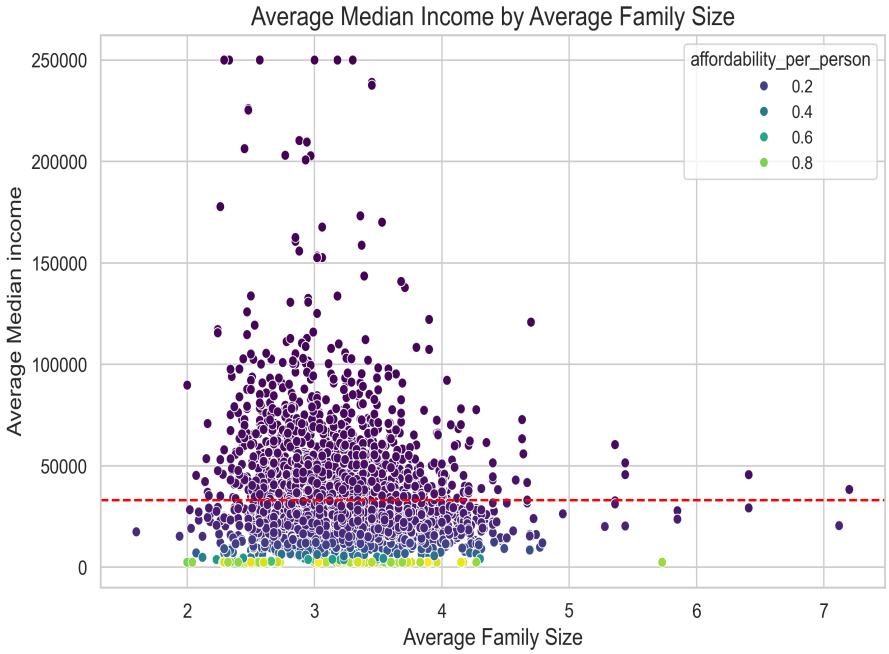

Figure 2: Scatterplot of Income Against Family Size

Figure 2 reveals a particularly interesting trend: while the spread of the points seems to suggest that most women-headed households earn enough to afford food for their families, the red line superimposed on the graph highlights the unfortunate reality. The median annual income in this dataset is $33,103.00. This is significantly below the median annual household income in the United States which stood at $80,610.00 as of 2022.

Although a majority of the data in this report is numeric, there are some categorical features that can still be examined. Table 3 below includes some summary statistics on these features.

	`race_eth_name`	`geotype`	`geoname`	`county_name`

	race_eth_name	geotype	geoname	county_name
count	295371	295371	295371	295236
unique	9	4	1581	58
top	AIAN	PL	Franklin CDP	Los Angeles
frequency	32819	298431	990	123966

Table 3: Categorical Data Summary Satistics

EDA Key Findings

The Exploratory Data Analysis presented in this section was useful to begin understanding basic structures, patterns, and relationships within this dataset. The following is a brief summary of the most important EDA insights:

- Many variables are calculated from the `affordability_ratio` feature - leading to high correlations
- There is a strong positive correlation between `med_income` and `food_afford_decile` which may be relevant in the feature selection process.
- This data contains a wide range of annual income ($2,500 - $250,000)
- The `med_income` feature is highly skewed, indicating a log transformation may be necessary.

These key findings from the EDA uphold that there is a complex relationship between income, family size, and food afforability among women-headed households in California. Moving forward, these insights will act as a foundation for finding a predictive model that best estimates median income based on economic and demographic factors.

Methods

Feature Engineering

Effective feature engineering for this dataset includes transforming the raw data obtained into meaningful, usable representations that can be fed into a model for accurate predictions. This section will identify the steps taken to engineer features effectively - specifically the process of enhancing relationships between income, household demographics, and food affordability.

1. Add Latitude and Longitude

The original dataset used for this report did not include coordinate information, but it can be helpful to include these features to visualize the regions this exploration focuses on. To gather this information, I imported a csv file from GitHub that contains all of the major counties, zipcodes, and coordinates in California. Using a list of counties in the original data, I mapped those names to the counties in the csv file and merged their corresponding latitude and longitude values.

2. Food Cost Index

The Food Cost Index is a measurement from the Consumer Price Index (CPI) used to understand the yearly costs of food in the United States. To assess where the women in this dataset stand in regards to the national average, the `food_cost_index` variable was calculated by dividing `cost_yr` by the national average cost of food. This value was calculated by averaging the median cost of food for high, medium, and low income households. (national average source)

3. Affordability Status

Understanding if food is affordable is a useful binary indicator that is implemented by calculating if the `affordability_ratio` is above 1 (easily affordable).

4. Affordability per Person

- `affordability_per_person` (numeric) Affordability ratio relative to family size

	latitude	longitude	affordability_per_person	food_cost_index
count	291717	291717	101733	264366
mean	35.218	-116.352	.097	72.955
standard deviation	2.655	8.745	.124	13.771
minimum	26.13	-123.73	.009	29.704
Q2	33.97	-119.67	.05	63.978
median	34.06	-118.26	.069	71.594
Q4	36.75	-117.32	.101	79.893
maximum	48.92	-71.35	.996	161.904

Table 4: Additional Numeric Data Summary Statistics

Preprocessing

4. Handling Categorical Features

All categorical features must be encoded into numeric formats that the computer can process when using modelling techniques. To do so, the `OneHotEncoder` package from the `sklearn` module is imported and fit to the predictors after they have been split into training and testing sets. This encoder works by creating binary columns for every unique category in all the categorical features contained in the data.

5. Handling Numeric Features

Similarly to the categorical features, numeric features also need some preprocessing before getting passed into relevant models. Since this analysis deals with features that are likely very skewed, the predictors need to be scaled using the `StandardScaler` package also from `sklearn`. The target `med_income` is also understood to be skewed, so taking the log of both the train and test target samples is a useful step to take for improved model interpretability.

6. Handling Missing Values

Referring back to Tables 2.1, 2.2, and 3, there are a handful of missing values that need to be handled before running any models. Using Scikit-Learn's `SimpleImputer` package, the missing numeric values can be filled with the average value of the given feature, and the most frequent value can be filled in categorical features.

include dimension reduction features

Supervised Learning Models

Model Name	**Description**	**Hyperparameters**	**Root Mean Squared Error (rMSE)**	**Time**	**Challenges**
K Nearest Neighbors Regressor	Predicting target by finding the average of the *k* number of neighbors surrounding a data point.	`n_neighbors`: 5 `weights`: 'uniform'	0.0227	80m11s	High computational cost
Ridge Regression	A linear regression model that uses Ridge regularization - a penalty term added to the cost function that pushes all coefficients towards zero.	`model__alpha`: 0.01	0.0647	3m7.3s	Poor rMSE score compared to other models
Decision Trees	Starting from the root node, branches move down with nodes split based on different patterns in the most important features until we reach the terminal node (leaves).	`model_max_depth`: 7 `model_min_samples_leaf`: 1 `model_min_samples_split`: 10	0.02495	2m32s	Prone to overfitting
XGBoost	Combines weak learning trees into strong learners by combining residuals and pruning.	`learning_rate`: 0.2 `max_depth`: 7 `n_estimators`: 100 `subsample`: 1.0	0.0063	8m25s	Training time slightly high
Deep Neural Network	Using Tensorflow, a DNN is an artificial neural network that includes multiple layers that can look for different patterns in the data.	`epochs`: 20 `batch_size`: 32 `validation_split`: 0.2	0.0385	8m34s	Low complexity in an effort to remain computationally cost effective

Table 5: Supervised Models

Discussion on Model Selection

Notable Patterns Across Models

1. Tree-Based Models Outperform Linear Models

As recorded in Table 5, the Decision Tree and XGBoost models outperformed Ridge Regression - indicating that there likely is not a strong linear relationship between the features.

2. Single Model Vs. Ensemble Method

While both the Decision Trees and XGBoost performed well, XGBoost's built-in regularization made it robust to overfitting, and more accurate. The DNN was flexible and handled the patterns in the data well, but it failed to outperform XGBoost, indicating that it may not be complex enough. While more layers and complexity could be added to this model, it is currently more computationally expensive than XGBoost, and adding more dimensions would only add to this time disparity.

Detailed Discussion on Best Model

After weighing all of the benefits and disadvantages of each of the models explored, the XGBoost model has proven to be the *best* based on its rMSE score, total time, and inherent ability to generalize well to new data.

XGBoost Hyperparameter Tuning

To build the most optimized version of the XGBoost Regressor, hyperparameter tuning was conducted using the grid search approach. The goal of this method was to provide many possible combinations of hyperparameters and identify the ones that minimize the model's prediction error on the training data, while also balancing the model's ability to generalize to unseen data. The follow hyperparameters were tested:

- `n_estimators`: Controls the number of trees in the model
- `learning_rate`: Controls the step size for updating model weights
- `max_depth`: Specify the maximum depth of a tree
- `subsample`: Fraction of observations randomly sampled for each tree

With these hyperparameter options, `GridSearchCV` tested all combinations, evaluated using 5-fold cross validation, which splits the data into 4 training subsets and one validation.

```
param_grid = {
    'n_estimators': [50, 100],
    'learning_rate': [0.01, 0.1, 0.2],
    'max_depth': [3, 5, 7],
    'subsample': [0.8, 1.0],
}
gb_grid_search = GridSearchCV(
    XGBRegressor(random_state=42),
    param_grid=param_grid,
    scoring='neg_mean_squared_error',
    cv=5,
    verbose = 1
)
gb_grid_search.fit(X_train_processed, y_train_log)

gb_grid_search.best_params_   #return the best parameters
```

This tuning process was effective in ensuring that the model was fit properly, and contained hyperparameters that control the model's over/under-fitting tendencies.

SHAP for Feature Importance

Shapley Additive Explanation (SHAP) values are a strong tool that can help interpret the information generated from the XGBoost Regressor's predictions. They act as a general framework that can be appplied to models, and explain their outputs by uniformly quantifying feature contributions.

One thing SHAP can be used for is to see how predicted values compare to actual values to evaluate model performance. From these comparisons, we can gather examples of true positives/negatives and false positives/negatives from the model's output.

1. True Positives/Negatives (TP/TN):

A predicted log income value of 10.5464 compared to the actual value of 10.54639 has an error of -0.00006 which is insignificant at the chosen 0.00001 level, thus suggesting that the model is accurate in this instance.

2. False Positives/Negatives (FP/FN):

A predicted log income value of 10.914 compared to the actual value of 10.9404 has an error of 0.026469 which is significant at the chosen 0.00001 level. Since the model incorrectly classified this observation higher than the actual income, this is a False Positive overprediction. An opposite instance with negative error would represent a false negative underprediction.

To continue interpreting the XGBoost model predictions, SHAP can be used to quanitfy and visualize feature contributions. This can be done on a global scale, identifying the most influential features across the dataset, and locally to visualize individual predictions.

Anomaly Detection

Dimension Reduction

Conclusion and Next Steps