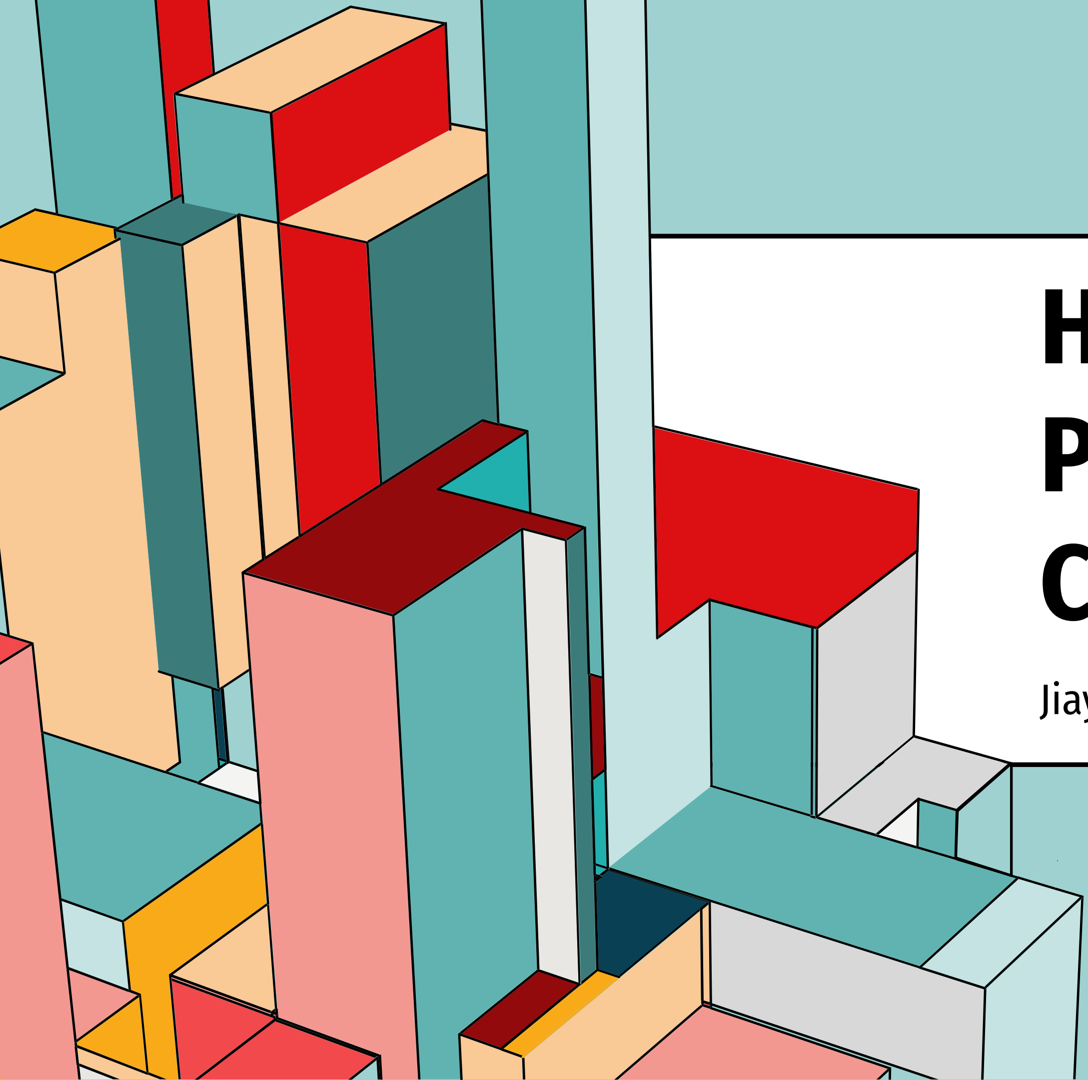


Housing Price Prediction by County

Jiayi Chen & Bell Luo for CSCI 0451

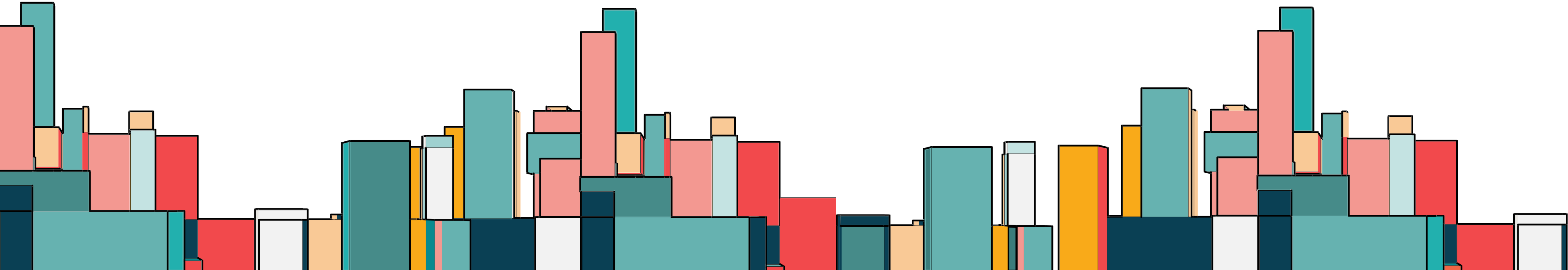
Motivation

Analyze the relationship between housing prices, demographic, and socioeconomic factors across counties

Inform and empower house buyers, sellers, renters, city planners, etc.

Goals

- **Combine** data from multiple sources to create a **comprehensive** dataset
- Create **interactive** maps to visualize housing price data by county
- Implement both **Classical Models** (including Linear Regression, SVM, Random Forest) and **Neural Networks** (through PyTorch) to analyze data



Data Sources

Zillow

- Typical home value
- By county & more
- Real-estate website
- Time series ready

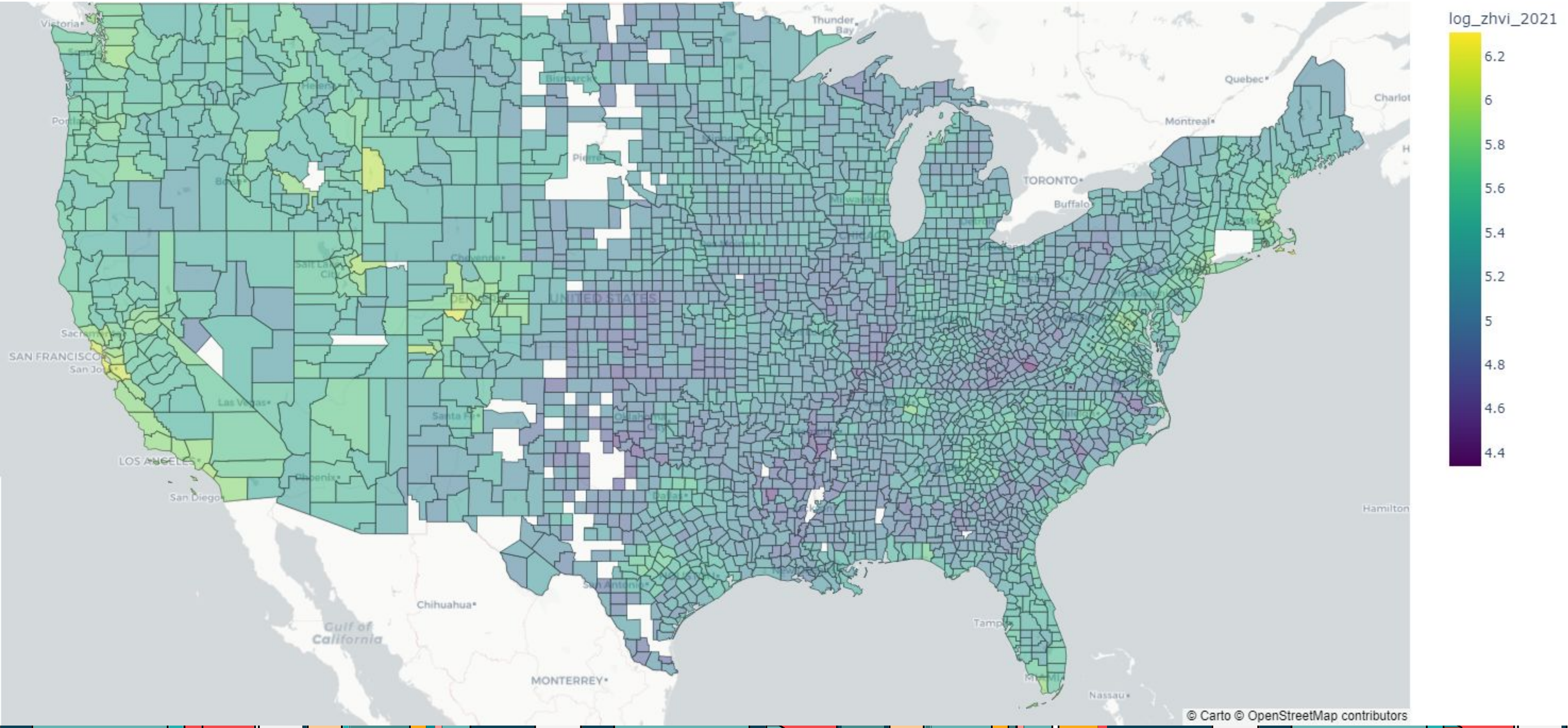
US Census

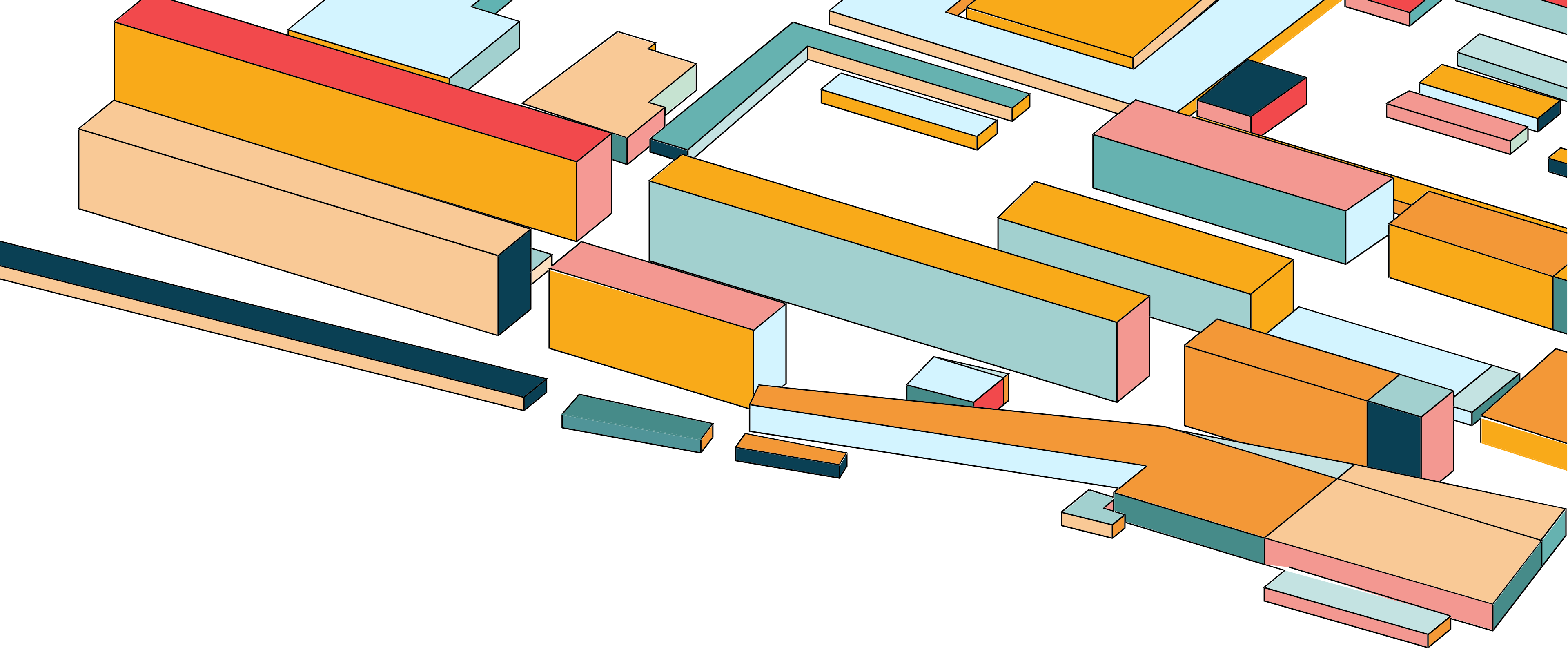
- ACS 5-Year Data
- Many predictors
- Population, income, education, etc.
- By county
- Time series ready

Miscellaneous

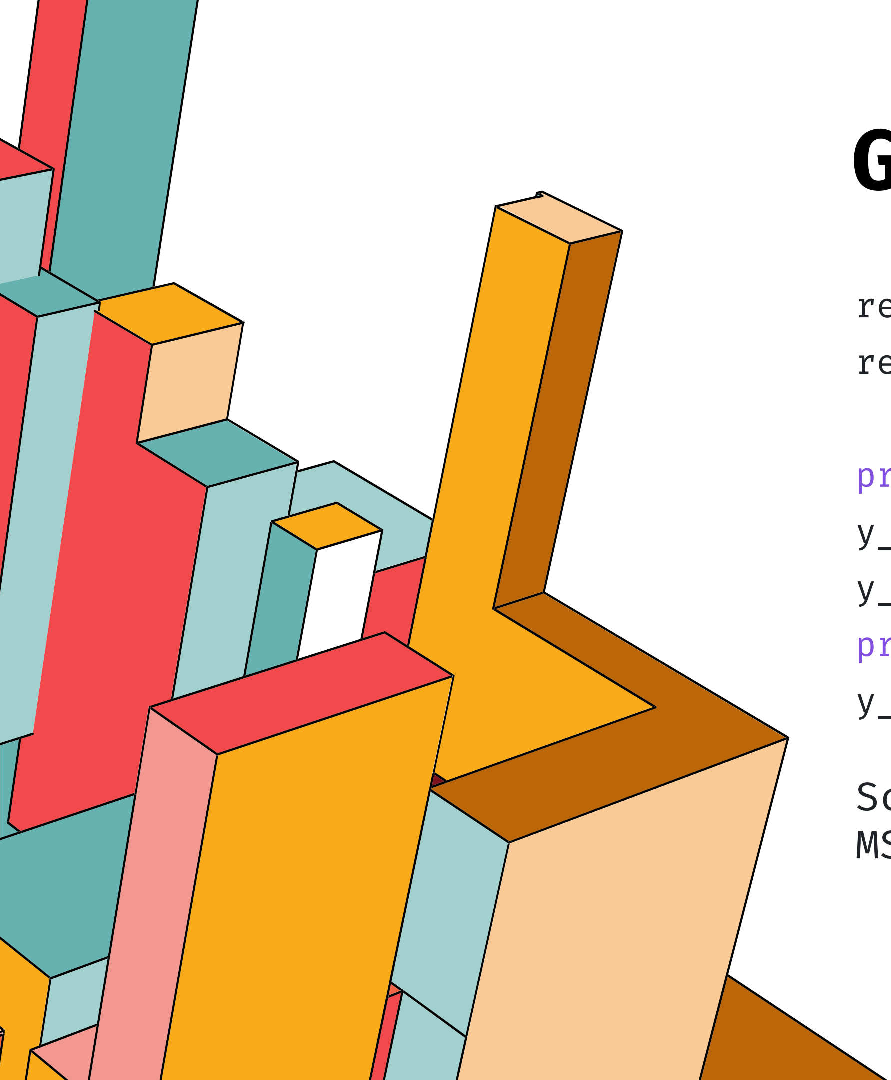
- More explanatory predictors
- Land area, Geospatial (longitude & latitude)
- By county

U.S. County-Level Housing Prices in 2021





Classical Models



Gradient Boosting

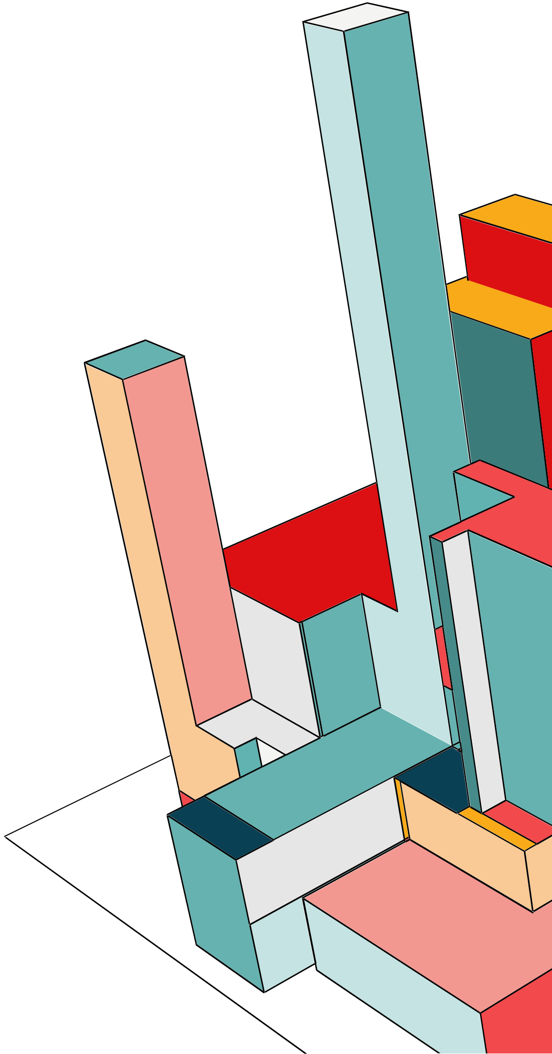
```
reg = GradientBoostingRegressor(**params)
reg.fit(X_train, np.ravel(y_train))

print(f"Score: {reg.score(X_test,
y_test):.4g}")
y_pred = reg.predict(X_test)
print(f"MSE: {mean_squared_error(y_test,
y_pred):.5e}")
```

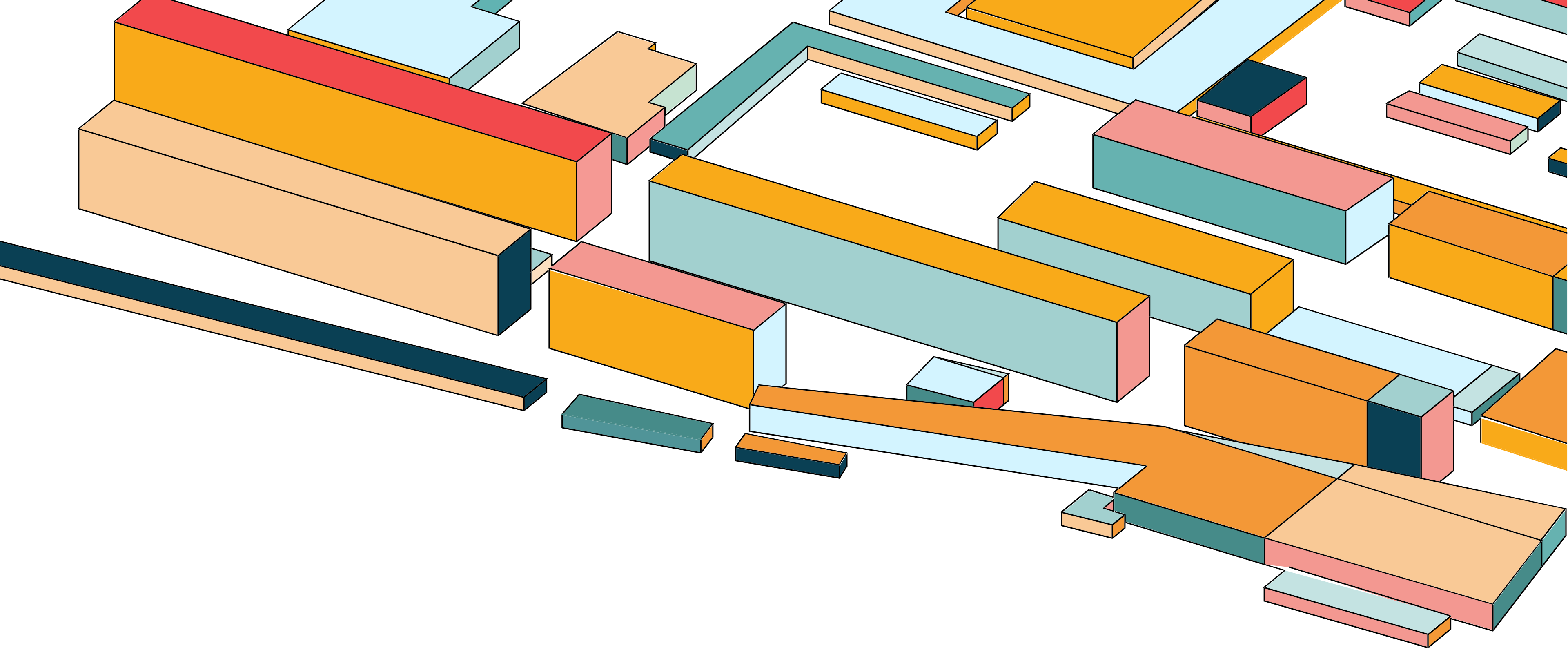
```
Score: 0.863
MSE: 2.47022e+09
```

Random Forest

```
regr = make_pipeline(StandardScaler(),
RandomForestRegressor(max_depth=20,
random_state=42))
regr.fit(X_train, np.ravel(y_train))
print(f"Score: {regr.score(X_test,
y_test):.4g}")
y_pred = regr.predict(X_test)
print(f"MSE: {mean_squared_error(y_test,
y_pred):.5e}")
```

```
Score: 0.9109
MSE: 1.60781e+09
```

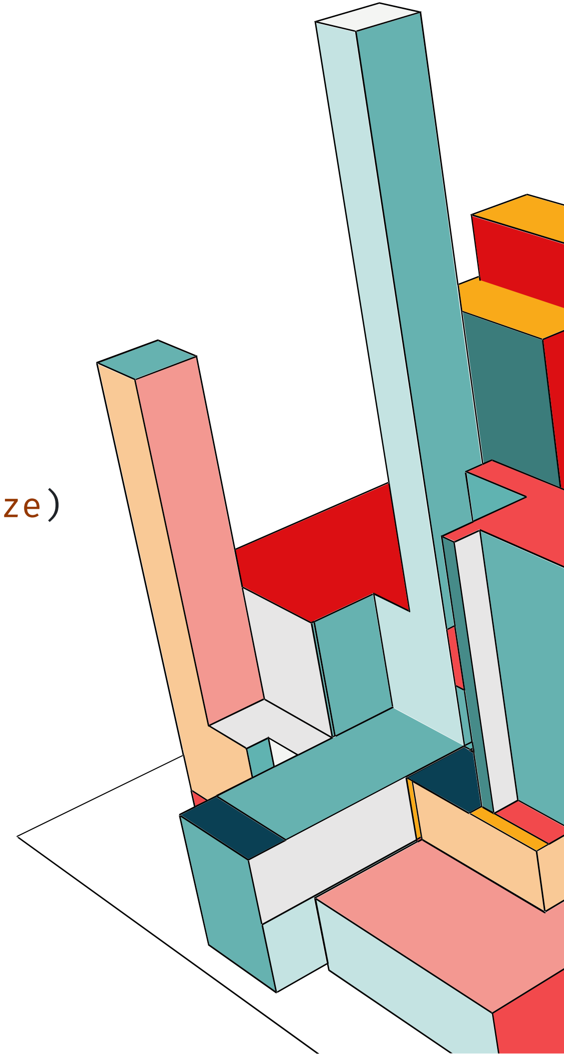




Neural Networks

Model Setup

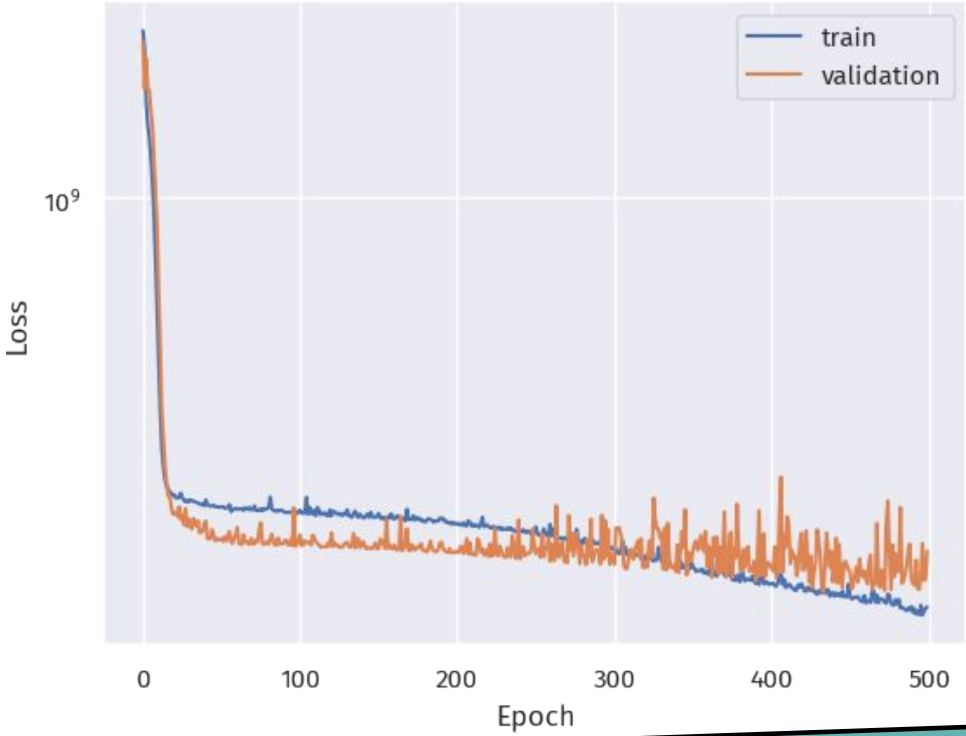
```
class LinearRegression(nn.Module):
    def __init__(self, input_size, hidden_size,
                 output_size, num_hidden_layers):
        super(LinearRegression, self).__init__()
        self.input_layer = nn.Linear(input_size, hidden_size)
        self.output_layer = nn.Linear(hidden_size,
                                      output_size)
        self.hidden_layers = nn.ModuleList()
        for _ in range(num_hidden_layers):
            self.hidden_layers.append(
                nn.Linear(hidden_size, hidden_size))
        self.activation = nn.ReLU()
```

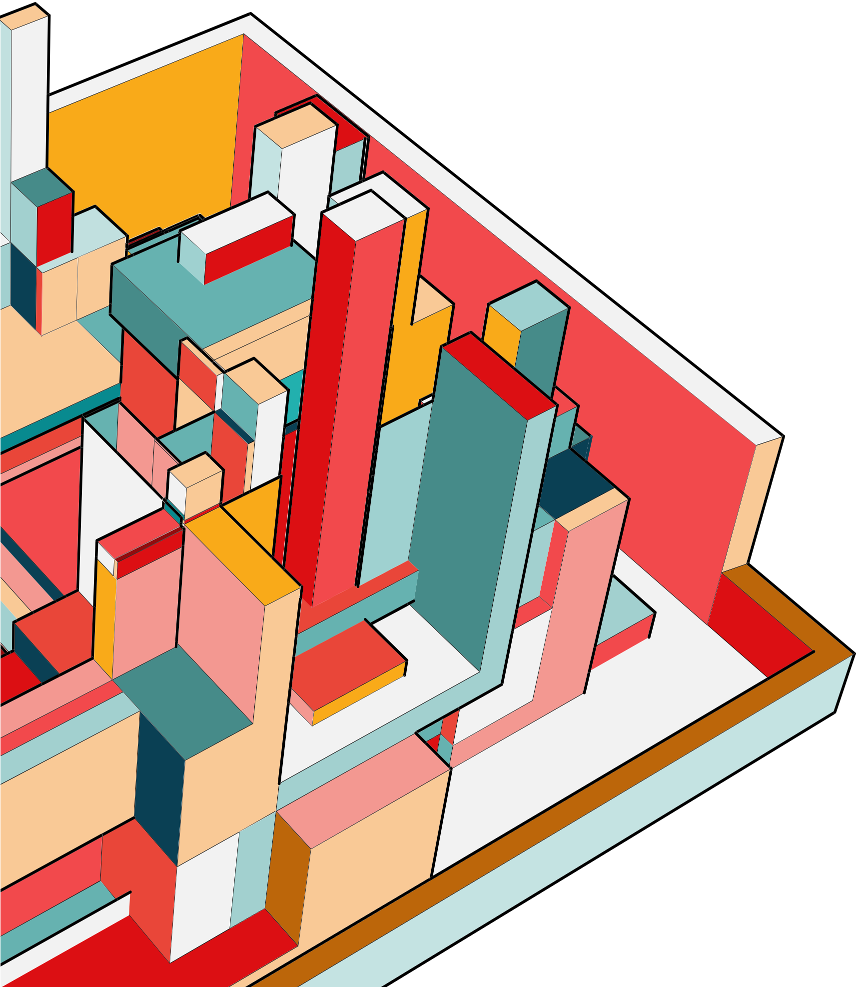


Training Loop

```
tlh, vlh = training_loop(model,
train_loader, test_loader, 1e-4, 500,
l1_lambda=1e-4)
```

```
epoch 000: val_loss = 1.76887e+09
epoch 050: val_loss = 2.88288e+08
epoch 100: val_loss = 2.92384e+08
epoch 150: val_loss = 2.78924e+08
epoch 200: val_loss = 2.75191e+08
epoch 250: val_loss = 2.73098e+08
epoch 300: val_loss = 2.62659e+08
epoch 350: val_loss = 2.76653e+08
epoch 400: val_loss = 2.59071e+08
```

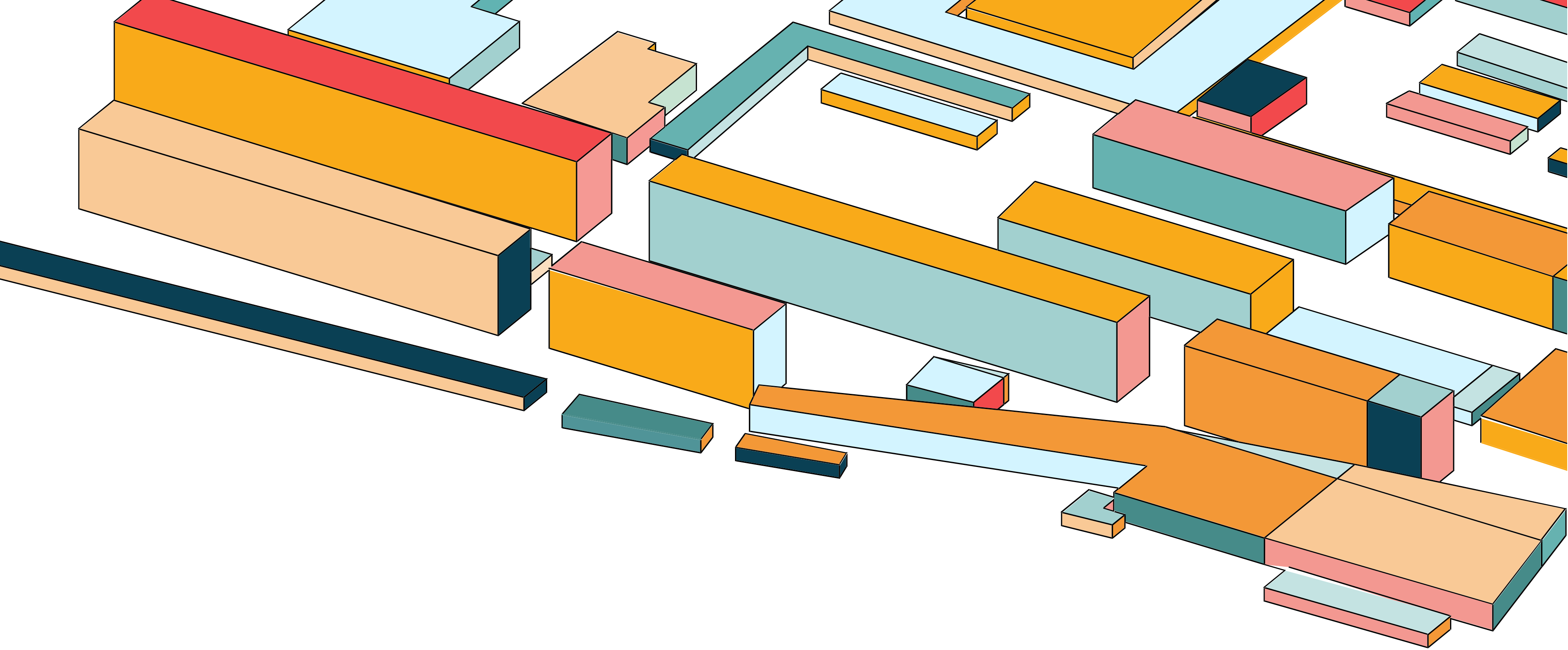




Further Tuning Needed

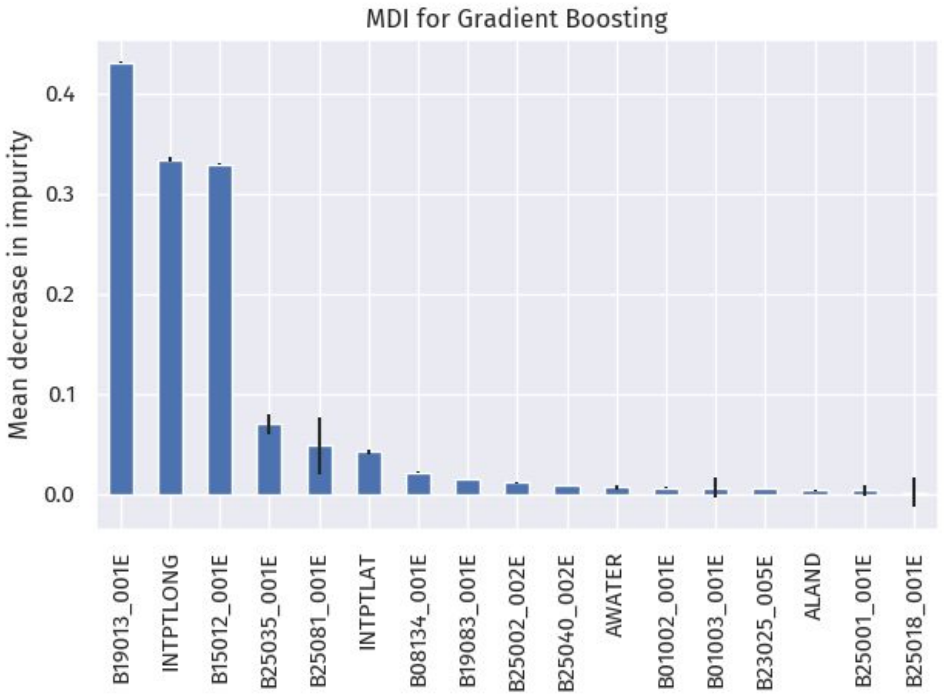
```
print(f"Score: {r2_score(y_test,
y_pred):.4g}")
print(f"MSE:
{mean_squared_error(y_test,
y_pred):.5e}")
```

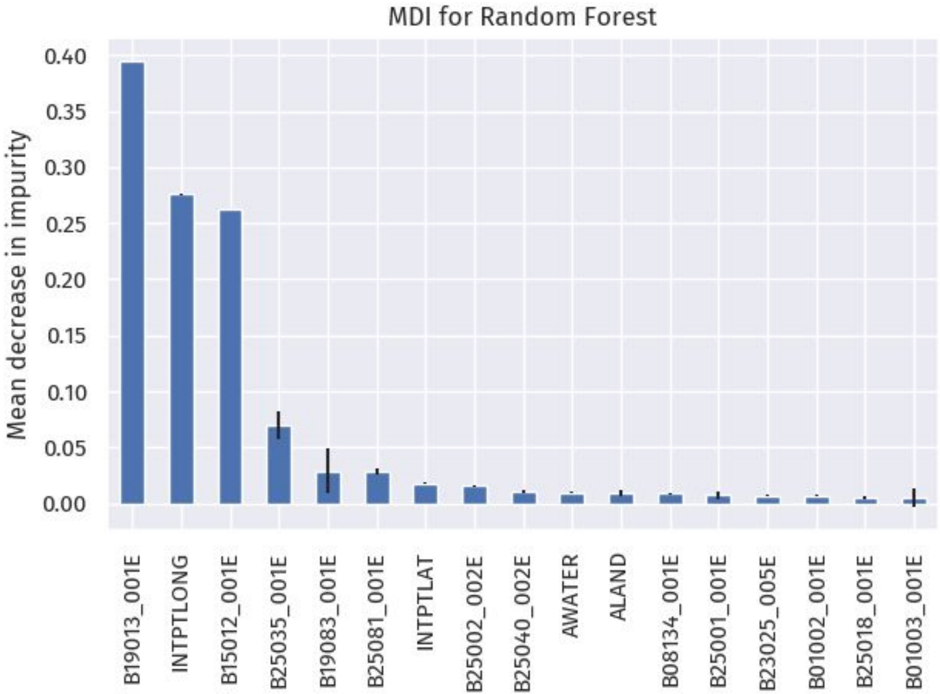
```
Score: 0.5231
MSE: 8.60080e+09
```



Findings

Mean Decrease in Impurity (Importance)





Important Features

Ordered by decreasing MDI:

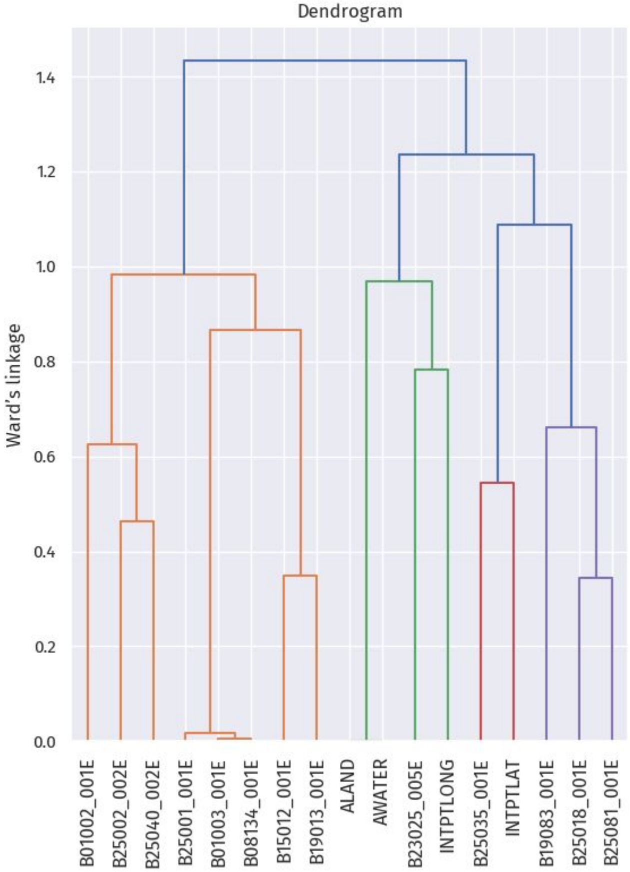
- `B19013_001E`: Median household income
- `INTPLONG`: Longitude
- `B15012_001E`: Count of Bachelor's degrees
- `B25035_001E`: Median year of construction

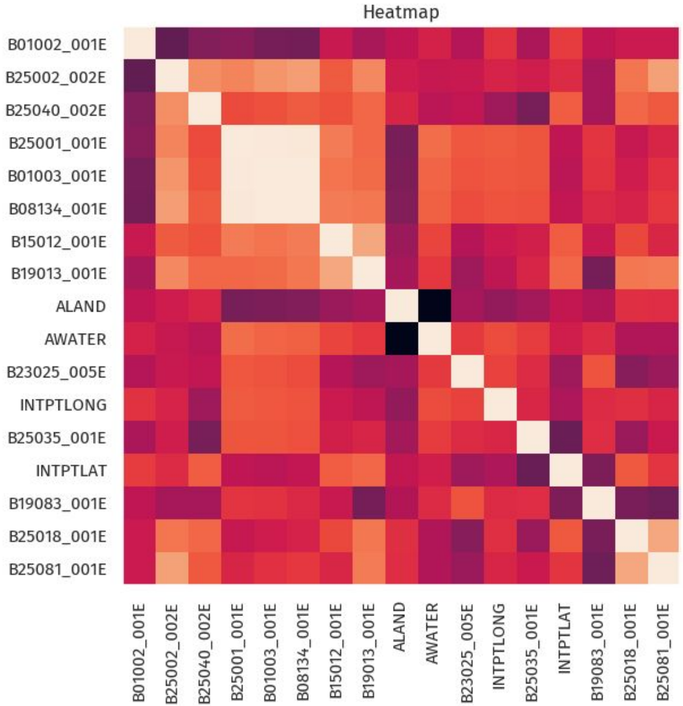
Model Performance

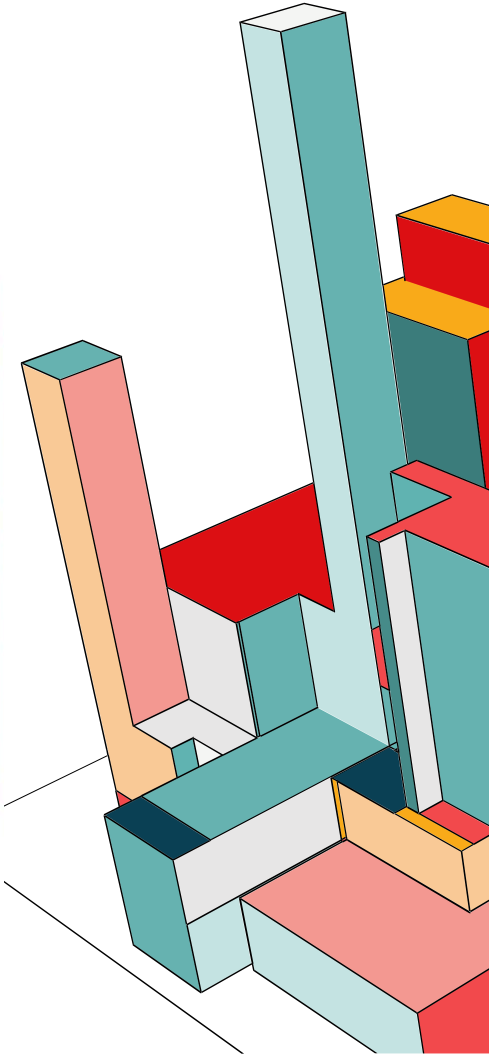
Ordered by decreasing test score:

- Random Forest Regression
- Gradient Boosting Regression
- Support Vector Regression
- Ridge Regression
- Linear Regression

Multicollinear or Correlated Features

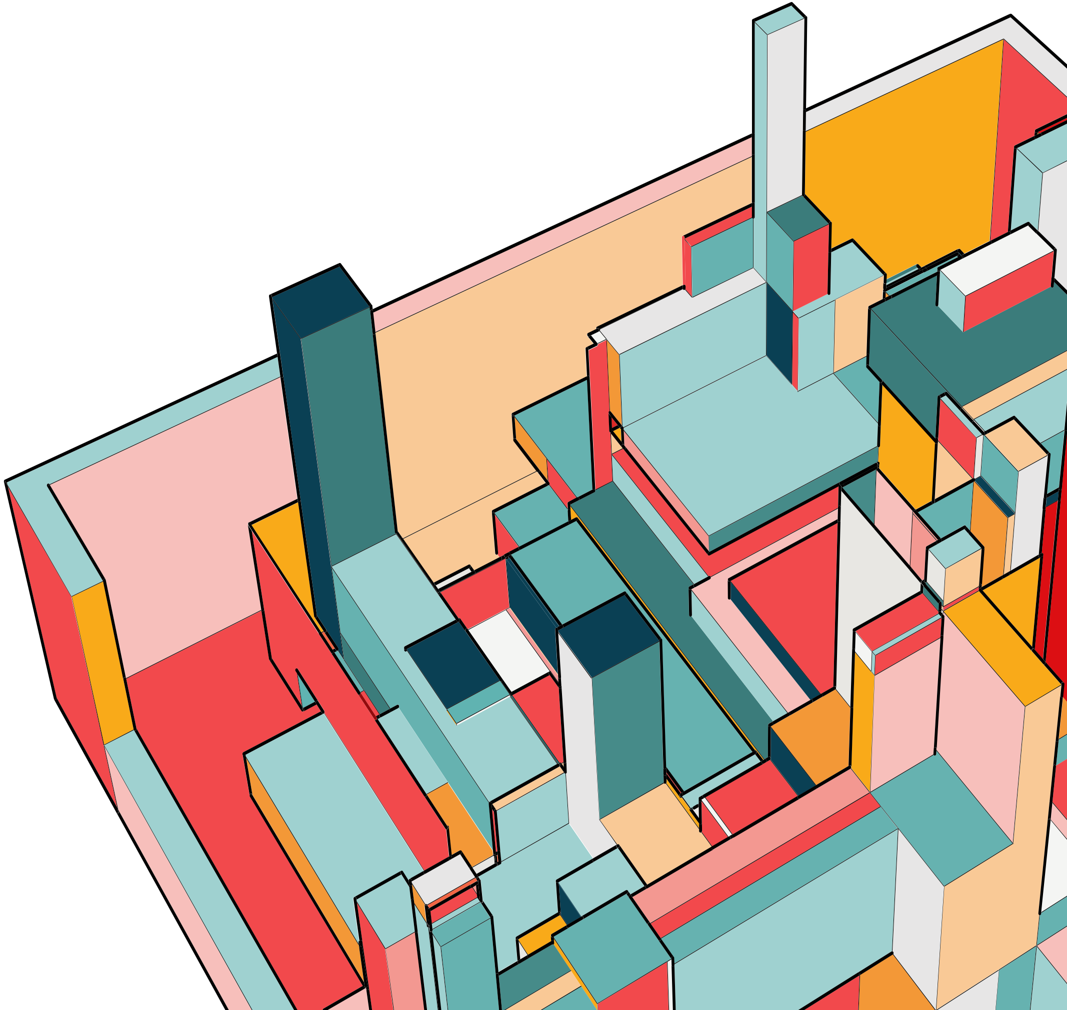






Future Work

- Time series with RNN
  - Past prices
  - Changes in ACS variables
  - Fed rates & inflation
- Auditing bias
  - Across race & other variables
  - Cross-sectional
  - Temporal
- Canada & México?



Thank you