Project 3
DSE 511

California Housing Prices Prediction

Avatar Team
(Albina, Amirehsan, Isidora and Pragya)

December 2021

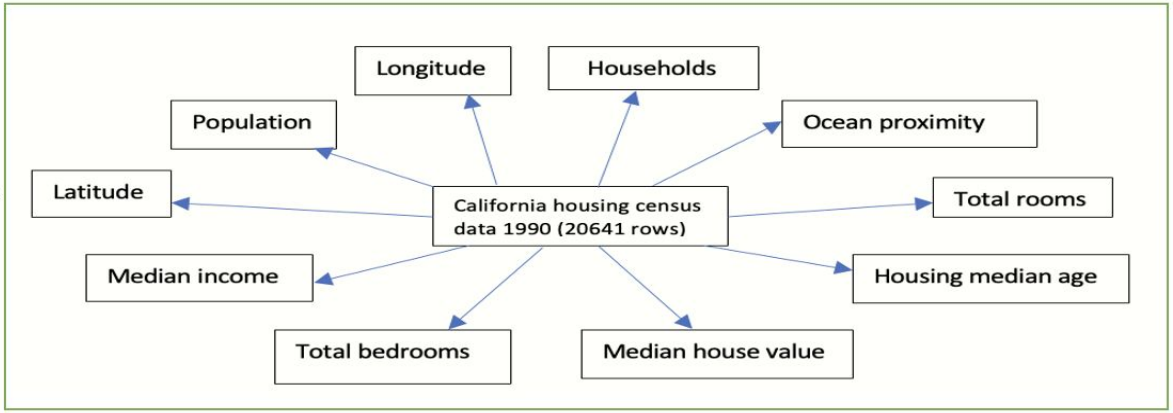
Introduction

New computing technologies widens scope of ML across various fields

Used in real estate for forecasting the house prices with maximum accuracy of the market and by building a model based on historical dataset

Prediction of house prices useful for homeowners, real estate agents, appraisers, mortgage lenders, property developers and investors

Looking into data

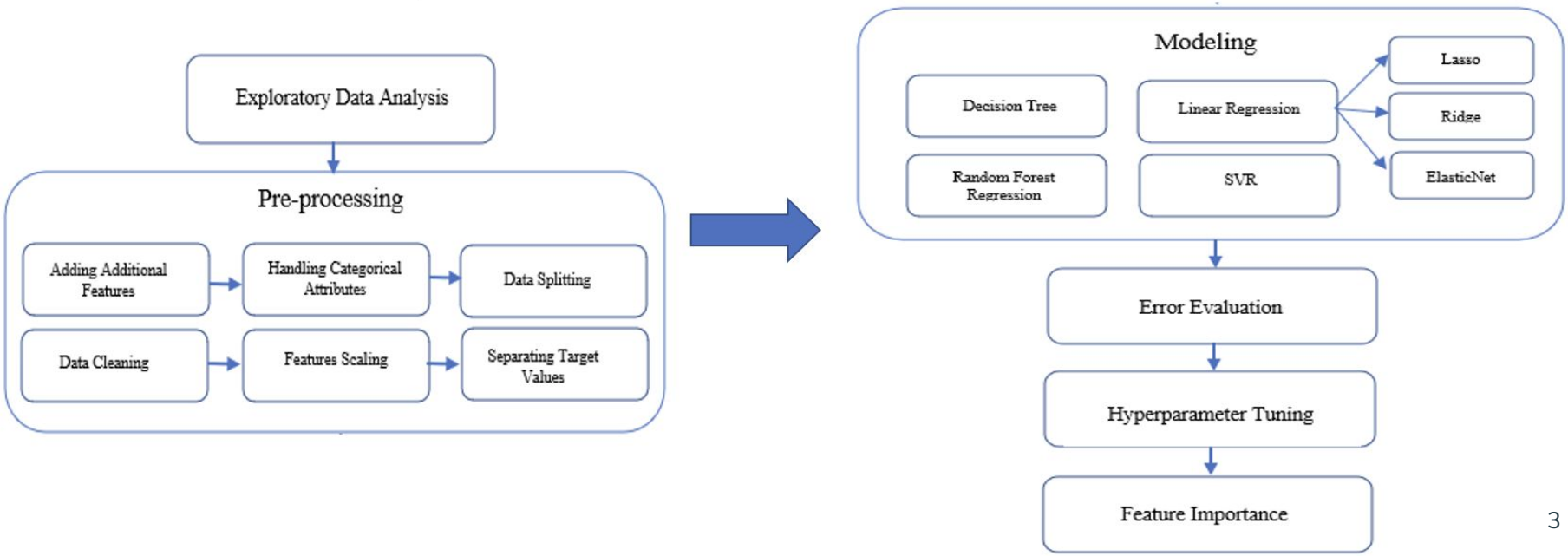


Hypothesis

Values of the houses ∝ ocean proximity, income

∝ 1/(housing median age, population in the area)

Methodology



Part1. Explanatory Data Analytics [Albina]

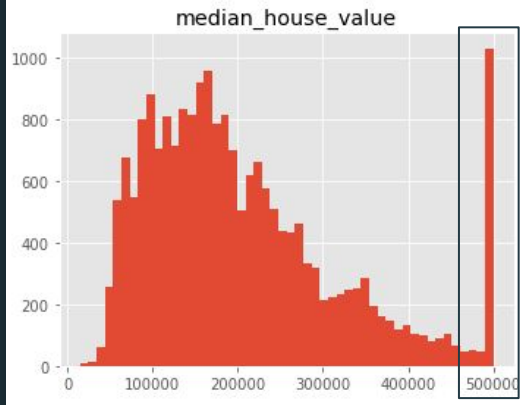


Figure A. Median house value (prices) distribution.

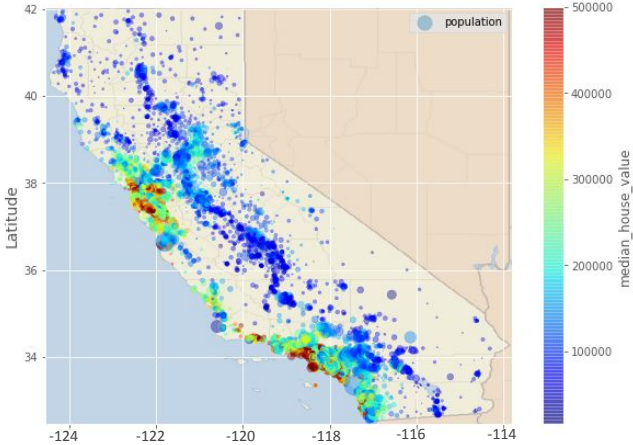


Figure B. Housing prices map

Figure C. Scatter plot of income vs prices with color coded ocean proximity.

- The potential issue with capped price values ($500,000) for models to predict prices in that range.
- Closer to the ocean & located in the regions of major cities Los Angeles, San Francisco and San Diego - more expensive houses.
- "Inland" houses - lower income-price region. Most of the "near bay" and "near ocean" houses - closer to the higher income-price region, while "<1 hour drive to ocean" - dispersed in the middle.
- The Pearson correlation coefficient between the income and price showed a strong positive correlation (0.68).

Part 2. Preprocessing [Ehsan]

Categorical feature handling

Gradient order (the lower the number the further away is the house from the ocean). This gradient is chosen for the better and easier interpretation of models' results (feature importances).

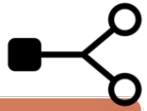
'NEAR OCEAN' = 4

'NEAR BAY' = 3

'<1H OCEAN' = 2

'INLAND' = 1

'ISLAND' = 0



Data split

To avoid data leakage and bias we split the data (80% train and 20% test sets) before cleaning and feature scaling.

Cleaning the missing values

"SimpleImputer" was used to insert the median values in the place of missing values.

Features scaling

Required this step as features had very different ranges, and ML algorithms generally do not perform well when the input numerical attributes have very different scales.

$$x_{scaled} = \frac{x - mean}{sd}$$

Part 3. Modeling [Albina]
Linear Regression/Lasso/Ridge/ElasticNet

- Linear Regression fits a linear model with coefficients that minimize the residual sum of squares between the observed and predicted by the linear approximation targets in the dataset.
- Ridge solves a regression model where the loss function is the linear least squares function and regularization is given by the L2-norm.
- Lasso penalizes the sum of their absolute values (L1 penalty). As a result, for high values of λ, many coefficients are exactly zeroed under Lasso, which is never the case in Ridge.
- ElasticNet combines the penalties of Ridge regression and lasso to get the best of both.

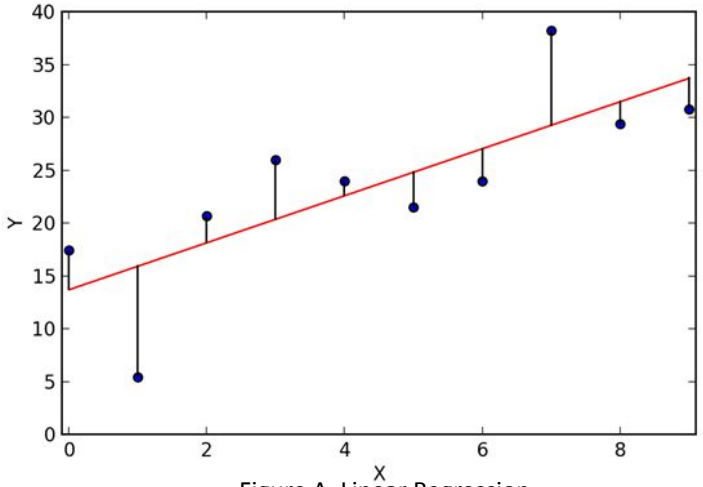


Figure A. Linear Regression

Ax_1 + Bx_2 + Cx_3 … = Y

- Y is target value (e.g., housing price)
- x_i are features (e.g., rooms, house age, etc.)
- A, B, C… are weights assigned to each feature
- LR finds the weights that minimize the distance between the predicted and true value

L1 Regularization

$$\text{Cost} = \sum_{i=0}^{N}(y_i - \sum_{j=0}^{M} x_{ij}W_j)^2 + \lambda \sum_{j=0}^{M} |W_j|$$

L2 Regularization

$$\text{Cost} = \underbrace{\sum_{i=0}^{N}(y_i - \sum_{j=0}^{M} x_{ij}W_j)^2}_{\text{Loss function}} + \underbrace{\lambda \sum_{j=0}^{M} W_j^2}_{\text{Regularization Term}}$$

Part 3. Modeling [Albina]
Linear Regression/Lasso/Ridge/ElasticNet

Table A. Best hyperparameters for models

Model	Best hyperparameter values
LR	NA
Lasso	alpha=100
Ridge	alpha=10
Elastic Net	l1_ratio=1 alpha=100

Table B. Model vs Error Evaluation Method. Capped Values

	RMSE	MAE	cross_val_score	score	wall time [train]	wall time [test]
LR	68949	49591	68471	0.637	17.9 ms	9.94 ms
Lasso	68949	49590	68471	0.637	48.8 ms	4.99 ms
Ridge	68949	49590	68471	0.637	18 ms	8.98 ms
Elastic Net	77862	58734	78592	0.537	25.9 ms	7.98 ms

Table C. Model vs Error Evaluation Method. Uncapped Values

	RMSE	MAE	cross_val_score	score	wall time [train]	wall time [test]
LR	60044	43727	59668	0.614	27.9 ms	7.98 ms
Lasso	59913	43705	59650	0.616	45.9 ms	4.94 ms
Ridge	60003	43721	59668	0.614	14.9 ms	5.98 ms
Elastic Net	67663	52149	68184	0.510	25.9 ms	7.98 ms

- Typical RMSE of approximately $69,000 for capped data is quite large (Considering that most of the prices range in $120,000 -$265,000).
- Lasso, Ridge and ElasticNet: even after hyperparameters tuning the results were not improved a lot. ElasticNet showed even worse performance.
- The largest improvement was observed on the uncapped dataset. The RMSE reduced by almost $10,000, which is quite significant. The model was trained better and thus, performed better, as these capped values significantly affected the patterns and their distribution. Moreover, they probably prevented the good generalization of modeling.
- Among all above discussed models, Lasso showed the best result on the uncapped dataset (it uses L1 regularization).

Part 3. Modeling [Albina]
Linear Regression/Lasso/Ridge/ElasticNet

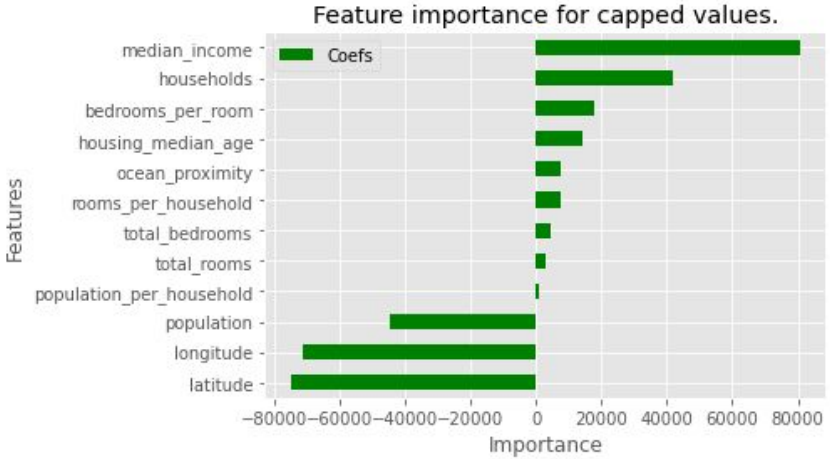


Figure A. Linear Regression feature importances (capped dataset)

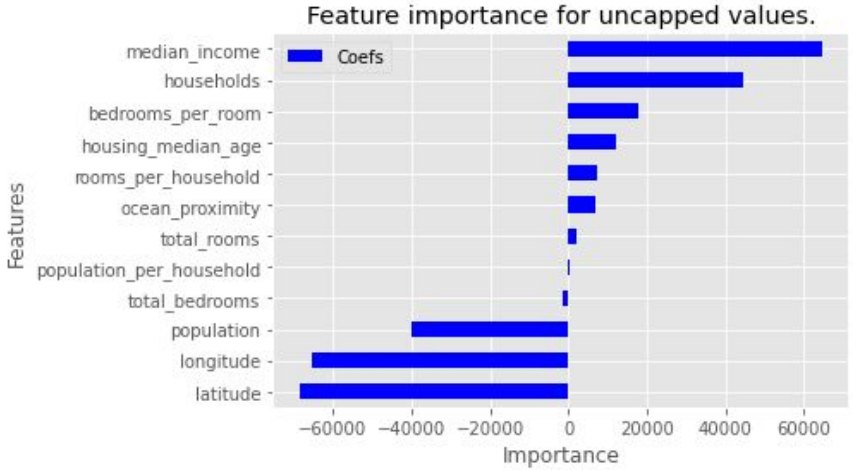


Figure B. Linear Regression feature importances (uncapped dataset)

To evaluate the features importance of the LR model, the coefficients of the model were extracted. .

1) **Income** has the highest positive significance, followed by the **households and bedrooms per room numbers**. Higher salaries correlate with buying more expensive houses, while larger households and number of bedrooms per room imply the bigger properties and thus, higher prices.
2) Strong negative coefficients are noticed for the **longitude, latitude** and the price. Moving more towards the continental part of the state and away from the coastal line decreases the prices.
3) Another strong negative relationship was observed for the **population**. The more private area - the higher the price.

Part 3. Modeling [Albina]
Linear Regression/Lasso/Ridge/ElasticNet

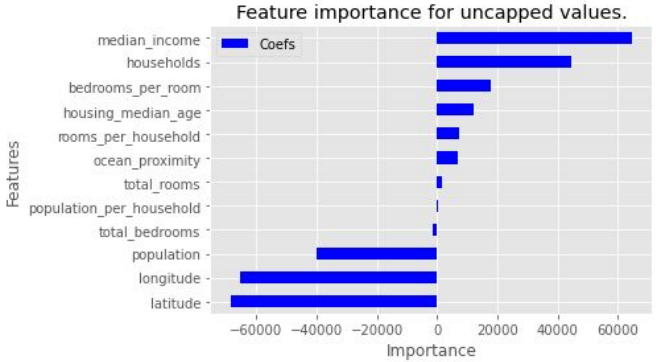


Figure A. Lasso feature importances (uncapped dataset)

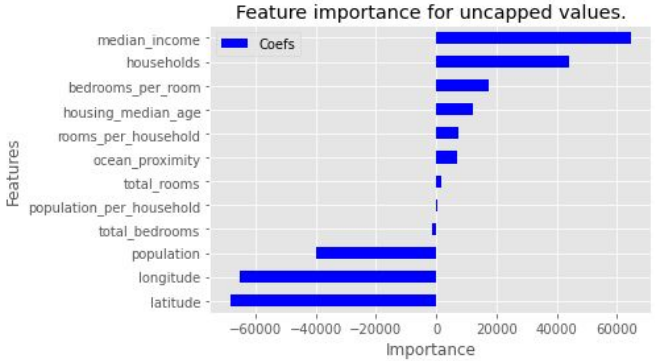


Figure B. Ridge feature importances (uncapped dataset)

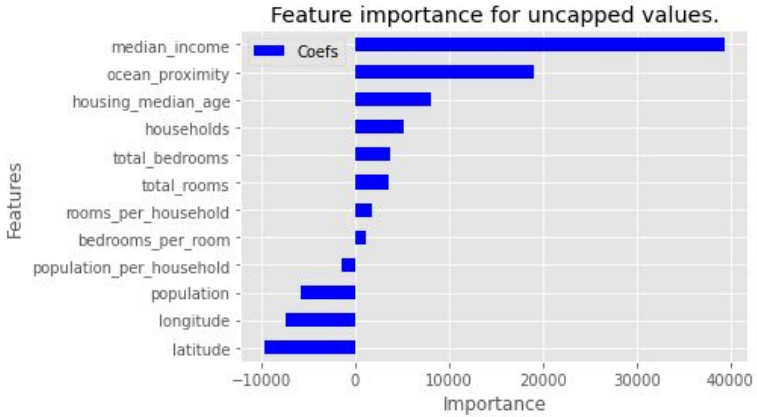


Figure C. ElasticNet feature importances (uncapped dataset)

- The feature importances for Lasso and Ridge were observed to be very similar to LR's.
- For ElasticNet, the second largest coefficient was that the ocean proximity, which was initially expected. As closer to the ocean locations might have higher prices.
- However, as ElasticNet performed worse than the baseline LR and Lasso/Ridge in terms of having largest errors, it would not be recommended to use this less reliable model.

Part 3. Modeling [Pragya]
Support Vector Regression (SVR)

- SVR is the ML approach that uses popular SVM algorithm to predict continuous variable
- Hyperparameter tuning increased accuracy and worked best on linear kernel

Table: Comparison of SVR before and after hyperparameter tuning

Errors	Before Tuning	After tuning($
MAE	76481	50235
RMSE	98318	68516

Not good ML approach as it's time consuming (> 10 min) and both MAE and RMSE was highest among all model used

Part 3. Modeling [Ehsan]
Decision Tree

- Decision tree is supervised ML approach where data is continuously split according to certain parameter

Model

```
dtr =  DecisionTreeRegressor ().fit(X_train, y_train)
```

Model scores

For training set: 1.0

For test set: 0.575

- 100% score on Training set. On testing set it was almost 63% score because no hyperparameters tuning was done, due to which depth of tree increased and our model did the overfitting. To solve this problem the hyperparameter tuning was utilized.

Part 3. Modeling [Ehsan]
Decision Tree

Best estimator

DecisionTreeRegressor(max_depth=8, max_leaf_nodes=100, min_samples_leaf=20,min_samples_split=20)

Table: Comparison of error before and after hyperparameter tuning

Errors	Before Tuning	After tuning
MAE	41939	38901
RMSE	63076	54244
R2 Score	0.57	0.68

Part 3. Modeling [Ehsan]

Decision Tree (Results: Visualization)

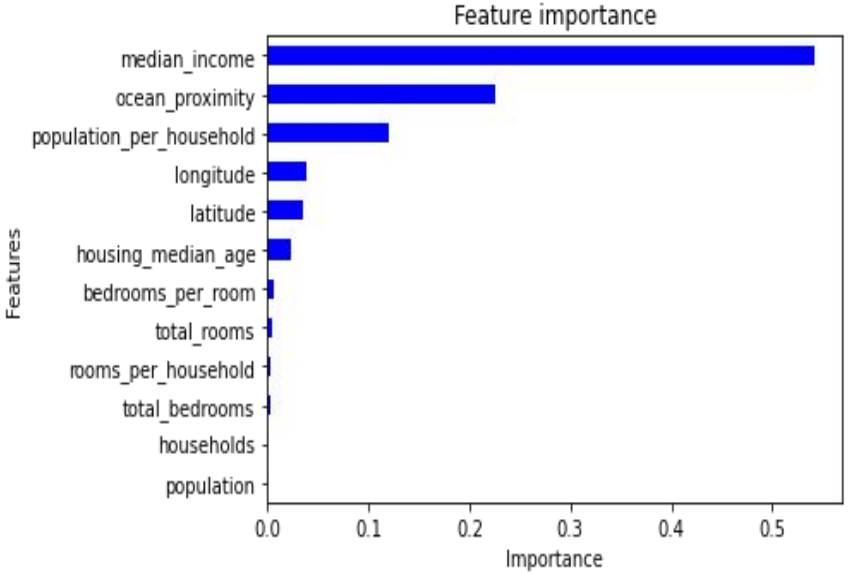


Figure A. Decision tree feature importances (capped dataset)

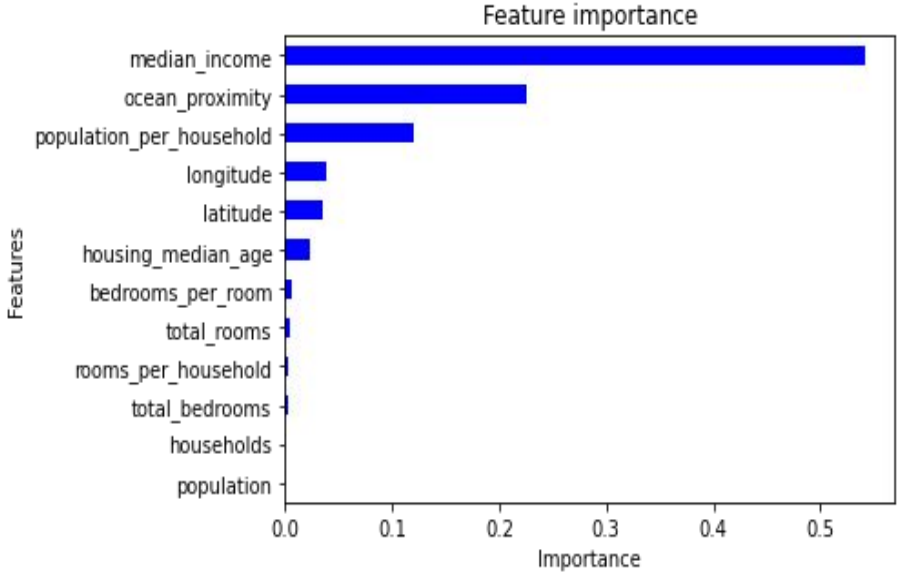


Figure B. Decision tree feature importances (uncapped dataset)

- Most important features: income and ocean proximity
- This was expected and corresponds to the results from the EDA section.

Part 3. Modeling [Ehsan]
Decision Tree (Results: Visualization)

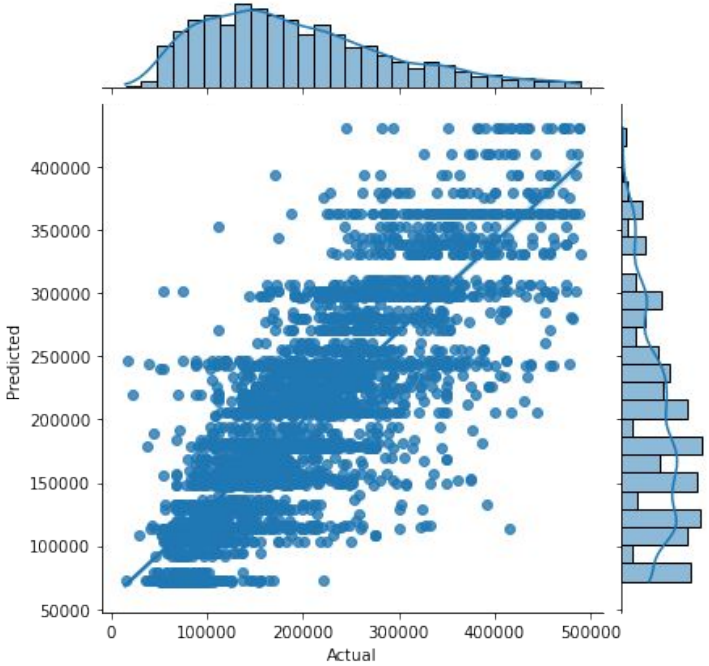


Figure A. Scatter plot of predicted and actual price values

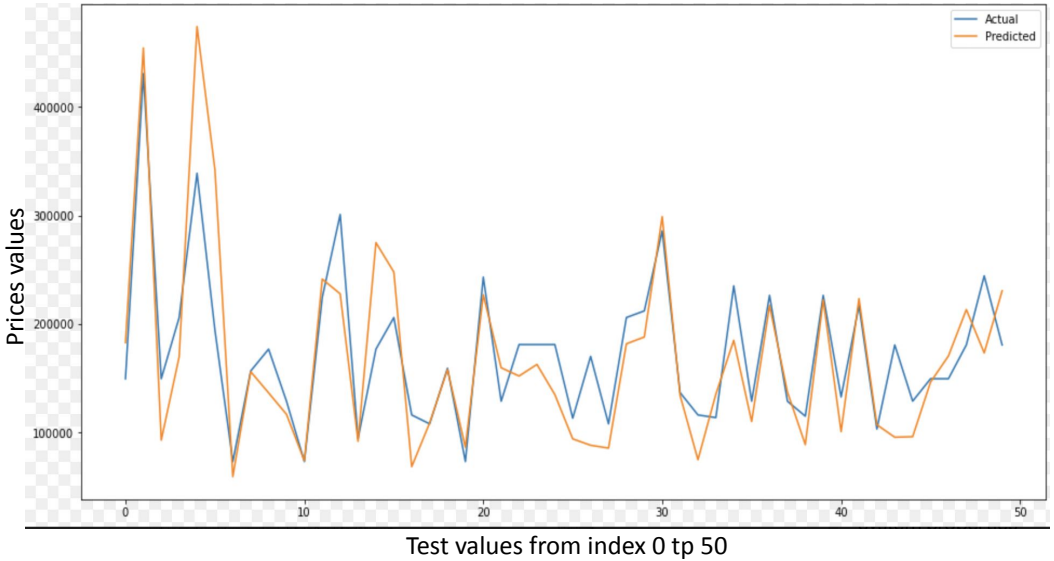


Figure B:. Actual and predicted prices (sample 0 to 50)

Part 3. Modeling [lsi]
Random Forest

- Random Forest is a supervised learning algorithm used to solve regression and classification problems.
- Random Forest Regressor is an estimator that generates a number of decision trees using various sub-samples of the data. Then, it averages the results obtained from the sub-samples in order to improve the predictive accuracy and control over-fitting.

Table. RMSE for Random Forest.

	RMSE with default parameters	RMSE with GridSearchCV
Capped	• $52500	$49300
Uncapped	• $46300	$43700

Part 3. Modeling [lsi]
Random Forest-Feature Importance

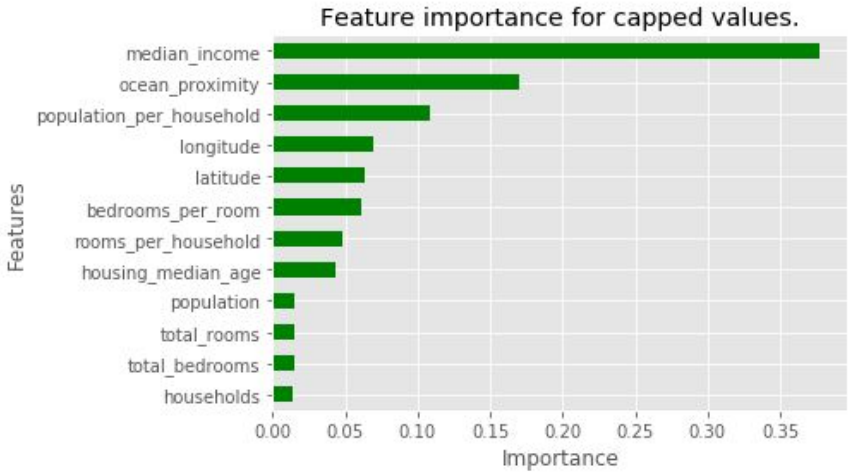


Figure A. Random Forest feature importances (capped dataset)

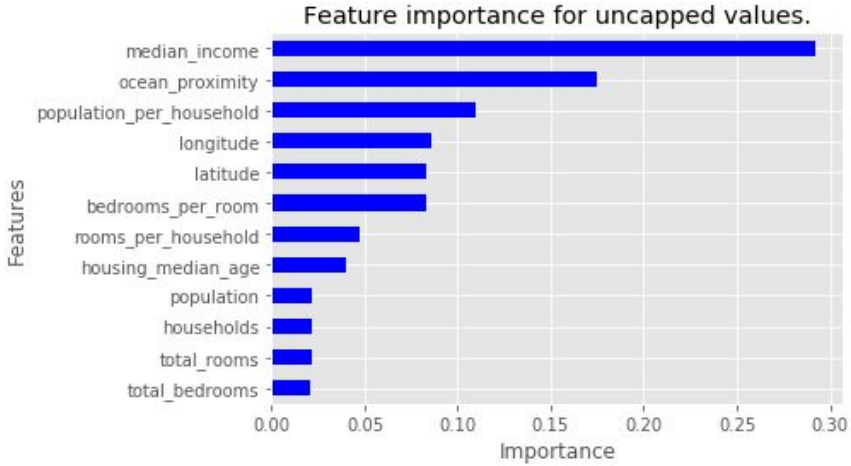


Figure B. Random Forest feature importances (uncapped dataset)

- Most important features: Ocean proximity/location and income.
- This was expected and can be explained by the results from the EDA section.

Conclusion [Isi]

Capped Dataset

	RMSE	MAE	cross_val_score	score	wall time [train]	wall time [test]
LR	68949	49591	68471	0.637	17.9 ms	9.94 ms
Lasso	68949	49590	68471	0.637	48.8 ms	4.99 ms
Ridge	68949	49590	68471	0.637	18 ms	8.98 ms
Elastic Net	77862	58734	78592	0.537	25.9 ms	7.98 ms
SVR	79738	56104	83467	0.460	14min19s	2.07s
Decision Tree	61041	42239	61301	0.715	3 min	3.89 ms
Random Forest Regression	49310	32349	49495	0.812	1.8 s	53.3 ms

Table A . Model vs Error Evaluation Method. Capped Values

Uncapped Dataset

	RMSE	MAE	cross_val_score	score	wall time [train]	wall time [test]
LR	60044	43727	59668	0.614	27.9 ms	7.98 ms
Lasso	59913	43705	59650	0.616	45.9 ms	4.94 ms
Ridge	60003	43721	59668	0.614	14.9 ms	5.98 ms
Elastic Net	67663	52149	68184	0.510	25.9 ms	7.98 ms
SVR	68516	50235	70561	0.461	12min3s	1.89s
Decision Tree	54244	38900	54812	0.685	2 min 29s	3.6 ms
Random Forest Regression	43777	29718	45104	0.795	1.24 s	36.9 ms

Table B . Model vs Error Evaluation Method. Uncapped Values

- The uncapped dataset was more accurate than the capped dataset, as predicted in the EDA section.

Conclusion [Isi]

- Models confirmed what was expected. Income and ocean proximity implied purchases of more expensive houses. Population was inversely proportional as predicted, but median age didn't follow what we expected, being directly proportional instead of inversely proportional.
- Best performing model: Random Forest. This model produced RMSE values between $43,000 and $44,000 for the uncapped dataset. The lowest values were obtained after hyperparameter tuning, and the best hyper parameters were 'max_features': 4 and 6 and 'n_estimators': 30.
- Random Forest results agreed with the other models, having income and location/ocean proximity as the most important features, which was expected
- This type of project offered us the opportunity to learn useful tools, like applied regression models.
- This project can be used to generate similar projects to evaluate pricing in other areas or to create other projects related to housing.
- Future work: PCA, NMF, etc.

References

- Kaggle. California Housing Priceshttps://www.kaggle.com/camnugent/california-housing-prices
- DSE511. Lecture slides

Github link to the repository: https://github.com/DSE511-Team-Avatar/Team-Avatar

Thank you!

Backup

Table A.Dataset: Features

Longitude	A measure of how far west a house is; a higher value is farther west
Latitude	A measure of how far north a house is; a higher value is farther north
housing_median_age	Median age of a house within a block; a lower number is a newer building
total_rooms	Total number of rooms within a block
total_bedrooms	Total number of bedrooms within a block
population	Total number of people residing within a block
households	Total number of households, a group of people residing within a home unit, for a block
median_income	Median income for households within a block of houses (measured in tens of thousands of US Dollars)
median_house_value	Median house value for households within a block (measured in US Dollars)
ocean_proximity	Location of the house w.r.t ocean/sea

Table B. Best hyperparameters for models

Model	**Best hyperparameter values**
Lasso	{alpha=100}
Ridge	{alpha=10}
Elastic Net	{l1_ratio=1 alpha=100}
SVR	{'C': 10, 'epsilon': 0.5, 'gamma': 1e-07, 'kernel': 'linear'}
Decision Tree	{'criterion': 'squared_error', 'max_depth': 8, 'max_leaf_nodes': 100, 'min_samples_leaf': 20, 'min_samples_split': 10}
Random Forest	{'max_features': 4, 'n_estimators': 30} {'max_features': 6, 'n_estimators': 30}

Backup

- Gaussian Processes (GP) are a generic supervised learning method designed to solve regression and classification, and optimization problems. GP is a useful technique that enables a non-parametric Bayesian approach to modeling
- The advantages of of GP: 1) the prediction interpolates the observations (at least for regular kernels); 2) the prediction is probabilistic so that it is possible to compute empirical confidence intervals and decide based on those if refit (online fitting, adaptive fitting) is needed for the prediction in some region of interest. 3) GP is versatile: different kernels can be specified.
- The disadvantages of GP that should be considered: 1) GP are not sparse, i.e., they use the whole samples/features information to perform the prediction. 2) GPs lose efficiency in high dimensional spaces – namely when the number of features exceeds a few dozens.

It was decided to try a radial-basis function kernel with noise and an offset, and the hyperparameters for the kernels (ConstantKernel, ConstantKernel, RBF, WhiteKernel) were suggested values and those were optimized during fitting.

In our case, as the dataset was too large, the GP model could not run it and the code crashed. This most probably happened due to the disadvantages of GP mentioned before, where GP are not sparse and they lose efficiency in high dimensional spaces.

```
MemoryError: Unable to allocate 3.98 GiB for an array with shape (16346, 16346, 2) and data type float64
```

- To check this model performance on the smaller dataset, our data was cut to 800 train and 200 test sets and modeled with GP again.
- This model performed worse than the baseline Linear Regression.
- The final results showed RMSE=94134 and MAE=76483.
- These large errors can be explained by the fact that not all data was used and only a small portion of it (5%), which does not allow the model to train and predict well. Thus, further investigation should be done on kernels and their parameters and more powerful hardware should be used for testing GP model.