

Real Estate Investment Prediction for San Jose

A Project Report
Presented to
The Faculty of the College of
Engineering
San Jose State University

In Partial Fulfillment
Of the Requirements for the Degree
Master of Science in Software Engineering

By
Mamatha Guntu
Srujana Koripalli
Prajakta Joshi
Umashankar Kumar

Abstract

House price forecasting is an important topic of real estate. Housing prices keep changing day in and day out and sometimes are hyped rather than being based on valuation. Predicting housing prices with real factors is the main crux of our research project. Here we aim to make our evaluations based on every basic parameter that is considered while determining the price. We use various regression techniques in this pathway, and our results are not sole determination of one technique rather it is the weighted mean of various techniques to give most accurate results. The results proved that this approach yields minimum error and maximum accuracy than individual algorithms applied. We also propose to use real-time neighborhood details by using different dataset to get exact real-world valuations.

Introduction

Millions of houses are sold every day. Today there is a large amount of data available on relevant statistics as well as on additional contextual factors, and it is natural to try to make use of these to improve our understanding of the industry. Every buyer asks himself some questions like what is the actual price of the house that this house deserves? Also, is it a good investment option? And is the price of the house is fair or not? In our project, we have implemented machine learning model which will predict a house price based on its neighborhood's data such as school, crime ratio etc. A model like this would be very much valuable for investor who can make use of the information to check the housing prices before investing.

Related work

For this project, we have used the CRISP-DM methodology.

1. Business understanding: House buyers or investors can gain an insight from this model whether to invest in a particular property or not. They can take this decision based on multiple factors such as price, rent estimate, bathroom, bedroom, nearby school, and neighborhood crime data.

2. Data understanding: We have used dataset from Zillow and performed data cleaning by removing the unwanted columns, null values and non-relevant columns. We also enriched our dataset by adding two more datasets to improve the dataset features.
3. Data preparation: We have scraped the data from Zillow and did the data cleaning. Also, we amalgamated multiple datasets with the original dataset by scraping crime index data and neighborhood school data.
4. Modeling: We implemented different machine learning algorithms to build training models and refine the data inputs. We applied algorithms like KNN and AdaBoostClassifier for final model.
5. Evaluation: We tried to interpret results of each used algorithms and tried to check if the algorithm output changes if we add some new features to the dataset. Also, used appropriate metrics like classification metrics, confusion matrix etc.
6. We build a flask rest API with user interface. We used React.js to develop front end.

Prediction

Fig 1. User interface to enter housing details

Data

Data for this project, we have scrapped from Zillow website.

```
[ ] # remove houses with price 0 or Nan
house_data = house_data[house_data['Price']>0]
house_data.head(10)
```

	Url	Zestimate	Price	Rent Zestimate	Days On Zillow	Bathrooms	Bedrooms	Living Area	Lot Size	Home Type	Street Address	City	Zip	State	Country	Broker Name	Has 3D Model	Has Image	Has Video	isZillowOwned	sgapt	statusText
0	https://www.zillow.com/homedetails/238-Kenbrook...	625700.0	655000	2849.0	Not specified	2.0	3.0	1164.0	NaN	CONDO	238 Kenbrook Cir	San Jose	95111	CA	USA	NaN	False	True	False	False	Unknown Listed By	Sold
1	https://www.zillow.com/homedetails/473-Toyon-A...	920100.0	920000	3499.0	Not specified	2.0	3.0	1287.0	NaN	SINGLE_FAMILY	473 Toyon Ave	San Jose	95127	CA	USA	NaN	False	True	False	False	Unknown Listed By	Sold
2	https://www.zillow.com/homedetails/6424-Nepo-C...	1451600.0	1420000	3959.0	Not specified	2.0	4.0	1958.0	NaN	SINGLE_FAMILY	6424 Nepo Ct	San Jose	95119	CA	USA	NaN	False	True	False	False	Unknown Listed By	Sold
3	https://www.zillow.com/homedetails/5887-Paddon...	1379100.0	1379000	3697.0	Not specified	2.0	3.0	1675.0	NaN	SINGLE_FAMILY	5887 Paddon Cir	San Jose	95123	CA	USA	NaN	False	True	False	False	Unknown Listed By	Sold
4	https://www.zillow.com/homedetails/443-Mignot-...	896800.0	900000	3820.0	Not specified	3.0	5.0	1617.0	NaN	SINGLE_FAMILY	443 Mignot Ln	San Jose	95111	CA	USA	NaN	False	True	False	False	Unknown Listed By	Sold
5	https://www.zillow.com/homedetails/307-Tradewi...	552000.0	552000	2385.0	Not specified	2.0	2.0	992.0	NaN	CONDO	307 Tradewinds Dr APT 9	San Jose	95123	CA	USA	NaN	False	True	False	False	Unknown Listed By	Sold
6	https://www.zillow.com/homedetails/461-N-15th-...	1108000.0	1150000	3200.0	Not specified	2.0	2.0	1428.0	NaN	SINGLE_FAMILY	461 N 15th St	San Jose	95112	CA	USA	NaN	False	True	False	False	Unknown Listed By	Sold
7	https://www.zillow.com/homedetails/470-Nolden-...	2144400.0	2030000	4857.0	Not specified	4.0	5.0	2204.0	NaN	SINGLE_FAMILY	470 Nolden Ave	San Jose	95117	CA	USA	NaN	False	True	False	False	Unknown Listed By	Sold
8	https://www.zillow.com/homedetails/5604-Makati...	675100.0	675000	2840.0	Not specified	2.0	2.0	1233.0	NaN	CONDO	5604 Makati Cir	San Jose	95123	CA	USA	NaN	False	True	False	False	Unknown Listed By	Sold
9	https://www.zillow.com/homedetails/2573-Island...	1115500.0	1114000	3399.0	Not specified	3.0	3.0	1406.0	NaN	SINGLE_FAMILY	2573 Island Palm Ct	San Jose	95133	CA	USA	NaN	False	True	False	False	Unknown Listed By	Sold

Fig 2. Scrapped Zillow data

We have preprocessed the data by removing unwanted columns, null values. Also, missing column and row values were detected, and appropriate column values were casted to relevant data type.

```
[ ] house_data["statusType"].unique()

array(['SOLD', 'FOR_RENT', 'FOR_SALE'], dtype=object)

[ ] house_data = house_data.drop(['Broker Name', 'Lot Size', 'Url', 'Has 3D Model', 'Has Image', 'Has Video', 'sgapt', 'Lot Size', 'City', 'State', 'Country', 'isZillowOwned', 'statusText'],axis=1)
```

The total count of null values.

```
[ ] house_data.isnull().sum()

Url          0
Zestimate    98
Price        0
Rent Zestimate 16
Days On Zillow 0
Bathrooms    26
Bedrooms     27
Living Area   13
Lot Size     399
Home Type     0
Street Address 0
City          0
Zip          0
State        0
Country      0
Broker Name   393
Has 3D Model  0
Has Image     26
Has Video     0
isZillowOwned 0
sgapt         0
statusText    0
statusType    0
dtype: int64
```

Methods

Using `corr()` to find the pairwise correlation of all columns in the dataset. `na` values will be excluded by using it.

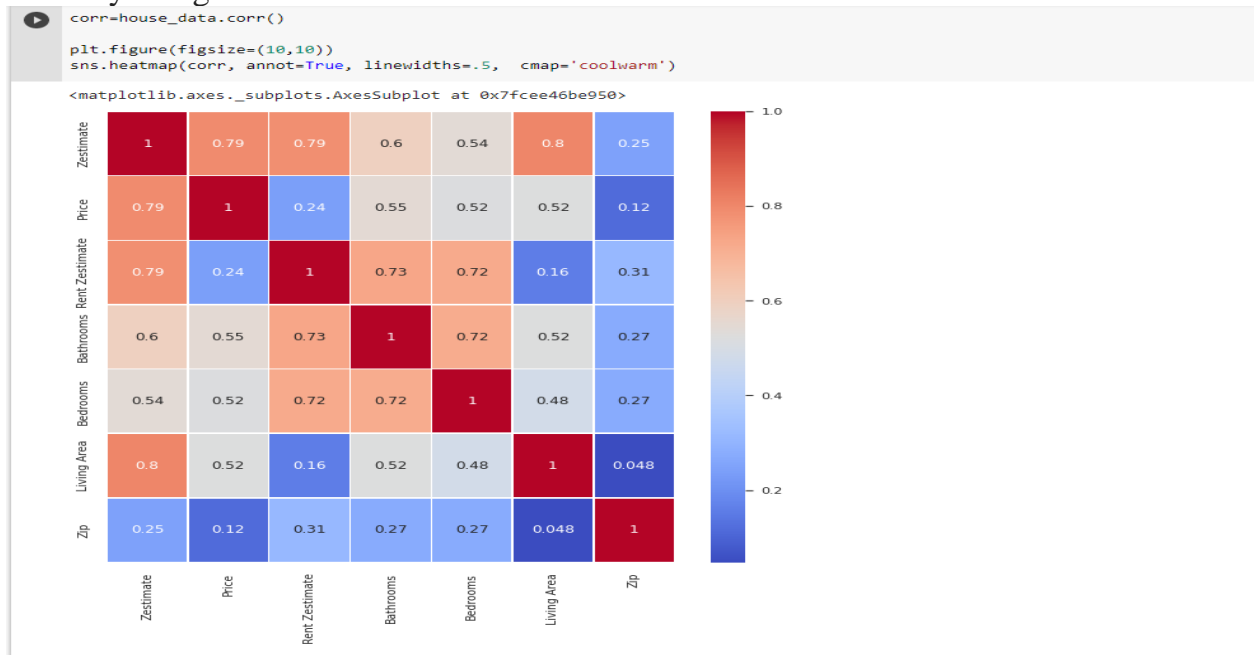


Fig 3. Heatmap and correlation of columns

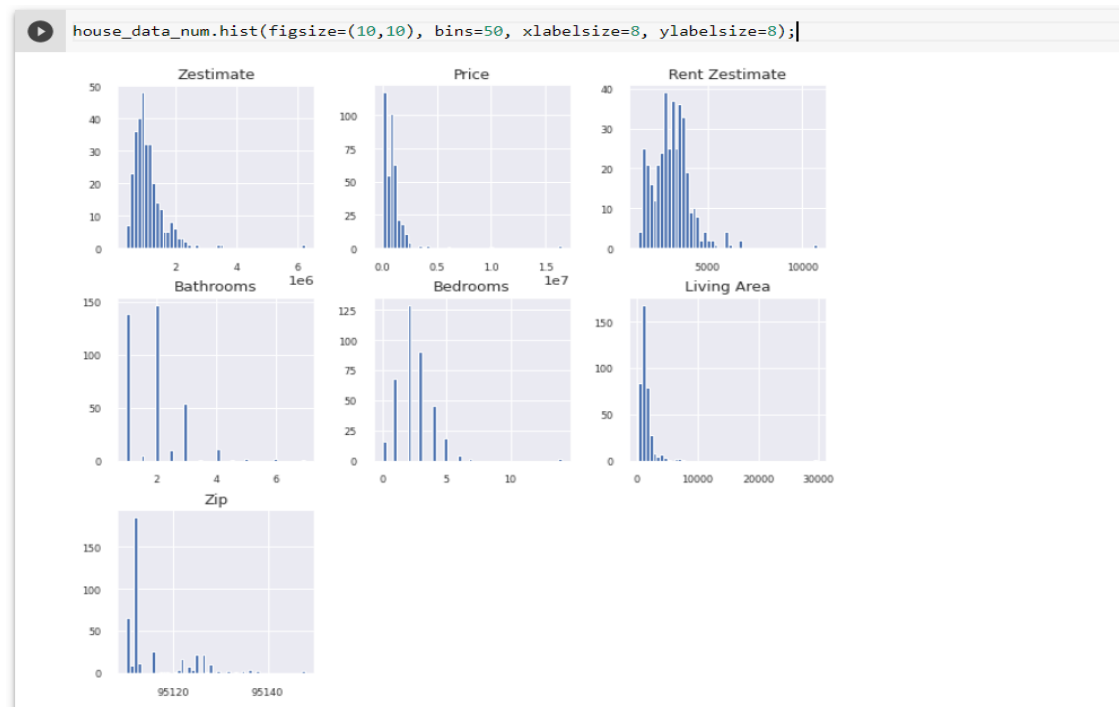


Fig 4. Understanding the distribution of numeric values.

Using `pairplot()` function, to plot multiple pairwise bivariate distributions of the dataset.

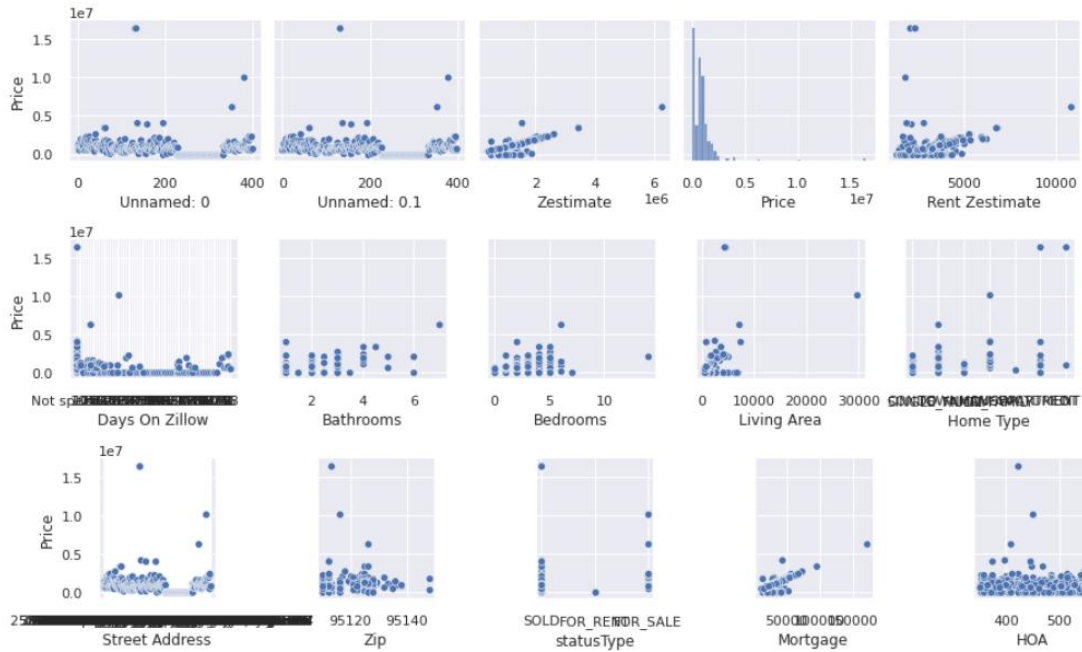


Fig 5. Pairwise bivariate distributions

Using PCA for Dimensionality reduction.



Applying K-means algorithm to visualize and plot the cluster.

```
[ ] plot_cluster(data_new)
```

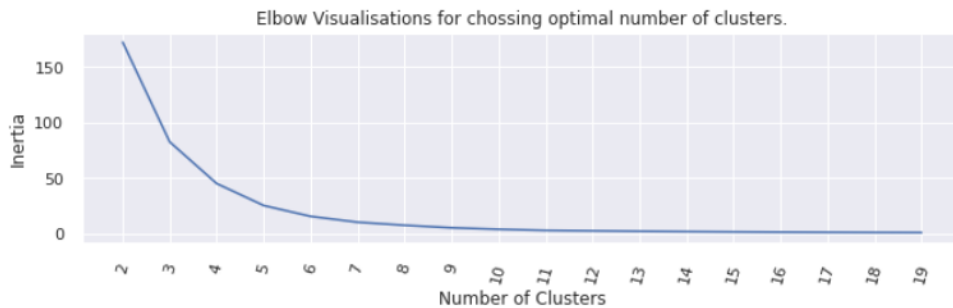


Fig 5. Elbow visualization

Experiments and Results

To predict the houses price, we split our data into two parts i.e., train data and test data and then performed linear regression to predict the house price.

Applied classifiers like GaussianProcessClassifier, DecisionTreeClassifier, AdaBoostClassifier, XGBClassifier. The final trained model is following.

AdaBoostClassifier - Final Model

```
[ ] from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
    from sklearn.model_selection import train_test_split

    feature_cols=['Price','RentalIncome','Change %','time_elapsed_after2years','increase_in_neighbourhood', 'Property_Crime','Violent_crime','commute_time']

    X = final_house_data[feature_cols]
    y = final_house_data['investment_label']

    M_train, M_test, n_train, n_test = train_test_split(X, y, test_size=0.3, random_state=1)
    # Create Decision Tree classifier object
    clf = AdaBoostClassifier()

    # Train Decision Tree Classifier
    clf = clf.fit(M_train,n_train)

    #Predict the response for test dataset
    y_pred = clf.predict(M_test)
    file = open('AdaBoostModel', 'wb')
    pickle.dump(clf, file)
    y_pred

array([[0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1,
        0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0,
        1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1,
        1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1,
        1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0,
        1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0,
        1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1])
```

Accuracy Score

```
[ ] from sklearn import metrics
    #accuracy
    print("accuracy:", metrics.accuracy_score(n_test,y_pred))
    #precision score
    print("precision:", metrics.precision_score(n_test,y_pred))
    #recall score
    print("recall" , metrics.recall_score(n_test,y_pred))
    print(metrics.classification_report(n_test, y_pred))
```

```
accuracy: 0.9833333333333333
precision: 0.975609756097561
recall 0.975609756097561

              precision    recall  f1-score   support

     0           0.99       0.99       0.99         79
     1           0.98       0.98       0.98         41

   accuracy                   0.98         120
  macro avg           0.98       0.98       0.98         120
 weighted avg           0.98       0.98       0.98         120
```

The accuracy score is .98

Confusion Matrix

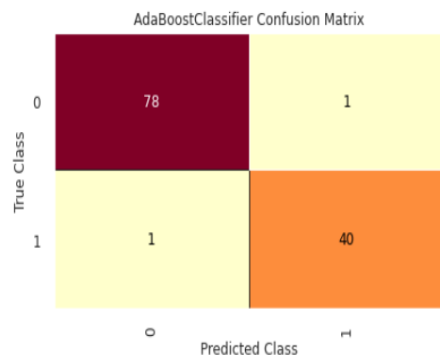
```
[ ] from sklearn.metrics import confusion_matrix
    fig, ax = plt.subplots()
    cm = ConfusionMatrix(clf, classes=[0, 1], ax=ax)

    #Fit fits the passed model. This is unnecessary if you pass the visualizer a pre-fitted model
    # cm.fit(X_train, y_train)
    cm.fit(M_train, n_train)

    #To create the ConfusionMatrix, we need some test data. Score runs predict() on the data
    #and then creates the confusion_matrix from scikit learn.
    # cm.score(X_test, y_test)
    cm.score(M_test, n_test)

    #How did we do?
    cm.poof()
```

/usr/local/lib/python3.7/dist-packages/sklearn/base.py:446: UserWarning: X does not have valid feature names, but AdaBoostClassifier was fitted with "X does not have valid feature names, but"

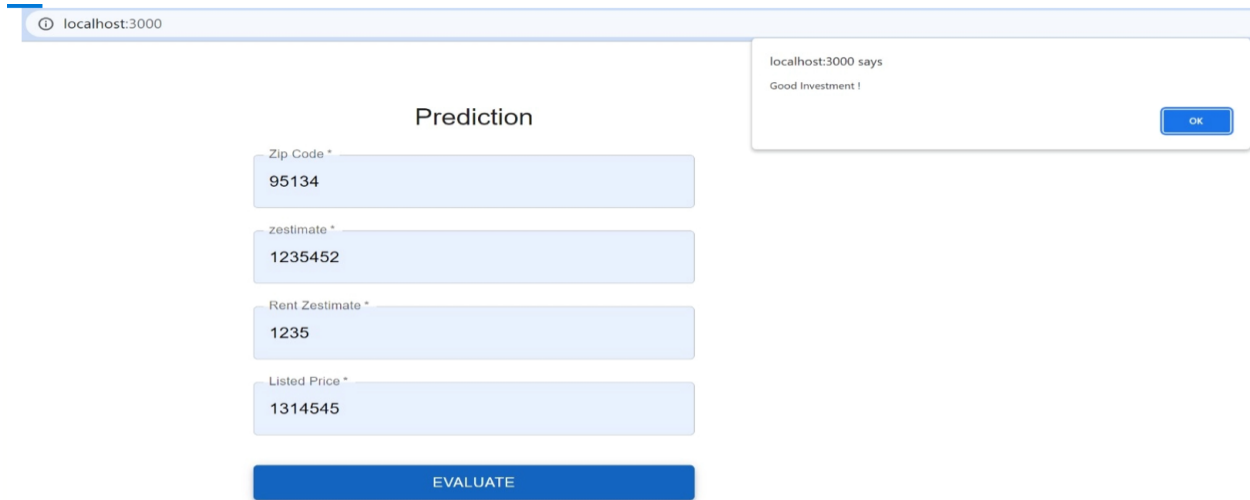


<matplotlib.axes._subplots.AxesSubplot at 0x7fcee52518d0>

Deployment

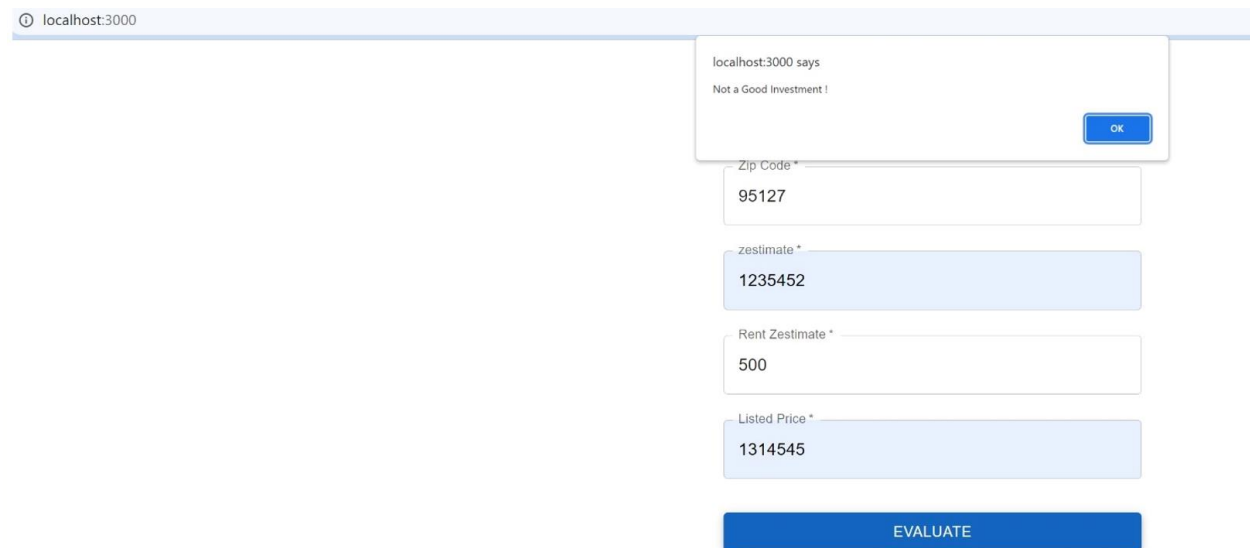
We have made rest API by using flask. And for UI development, we used React.js framework. User will provide inputs to our model and in turn we will get the result message either as good investment or not a good investment.

Input values entered by user. Model predicts it as a **good investment**.



The screenshot shows a web application running on localhost:3000. The main heading is "Prediction". Below it, there are four input fields: "Zip Code *" with value "95134", "zestimate *" with value "1235452", "Rent Zestimate *" with value "1235", and "Listed Price *" with value "1314545". At the bottom of these fields is a blue button labeled "EVALUATE". To the right of the input fields, a white box displays the message "localhost:3000 says" followed by "Good Investment !". An "OK" button is located at the bottom right of this message box.

Fig. 6 Model predicts good investment



The screenshot shows the same web application on localhost:3000. The input fields are: "Zip Code *" with value "95127", "zestimate *" with value "1235452", "Rent Zestimate *" with value "500", and "Listed Price *" with value "1314545". The "EVALUATE" button is at the bottom. To the right, a white box displays the message "localhost:3000 says" followed by "Not a Good Investment !". An "OK" button is located at the bottom right of this message box.

Fig. 7 Model predicts bad investment

http://127.0.0.1:5000/predictInvestment/154895/175452/1235/95134

GET http://127.0.0.1:5000/predictInvestment/154895/175452/1235/95134

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Headers 6 hidden

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (5) Test Results Status: 200 OK Time: 48 r

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "Not a Good investment !"
3 }
```

We called our API through **POSTMAN** also, to ensure all is working as expected.

References

1. <https://towardsdatascience.com/machine-learning-project-predicting-boston-house-prices-with-regression-b4e47493633d>
2. <https://towardsdatascience.com/predicting-house-prices-with-machine-learning-62d5bcd0d68f>
3. <https://www.kaggle.com/c/house-prices-advanced-regression-techniques>
4. <https://medium.com/codex/house-price-prediction-with-machine-learning-in-python-cf9df744f7ff>