[1] UTKFace dataset.

Dataset link :

https://susanqq.github.io/UTKFace/

About dataset :

UTKFace dataset is a large-scale face dataset with long age span (range from 0 to 116 years old). The dataset consists of over 20,000 face images with annotations of age, gender, and ethnicity. The images cover large variation in pose, facial expression, illumination, occlusion, resolution, etc. This dataset could be used on a variety of tasks, e.g., face detection, age estimation, age progression/regression, landmark localization, etc.

Labels:

The labels of each face image is embedded in the file name, formated like [age]_[gender]_[race]_[date&time].jpg

[age] is an integer from 0 to 116, indicating the age.

[gender] is either 0 (male) or 1 (female).

[race] is an integer from 0 to 4, denoting White, Black, Asian, Indian, and Others (like Hispanic, Latino, Middle Eastern.

[date&time] is in the format of yyyymmddHHMMSSFFF, showing the date and time an image was collected to UTKFace.

Implementation details :

We used two algorithms K-Means and Logistic Regression

Pre-Processing:

Extracts features by HOG algorithm, And reading age, gender, and race in lists, Compile a list of age, gender, race and photo features in Dataset.

We got 2190 features per each image, but we sum all this features in one new feature called features.

Save this dataset into file .csv

Then normalize dataset by Min-Max Normalization algorithm. Split data into features and target

features: age, race and image

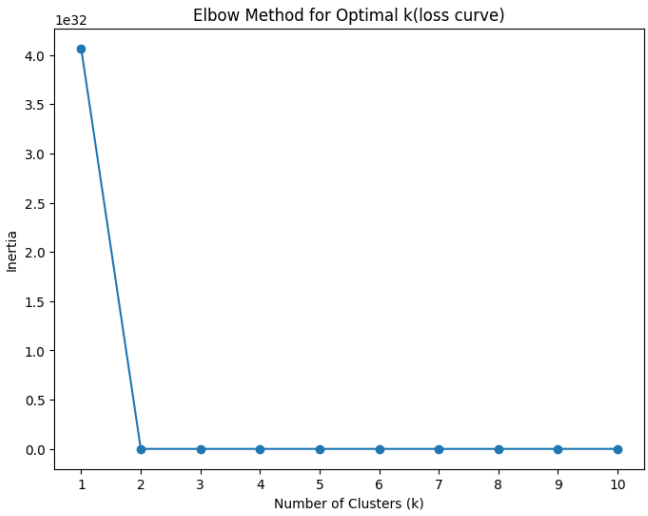
target: gender

And split dataset into 20% test and 80% train by train_test_slpit algorithm.

A) K-Means:

Goal: Classify humans into Male and Female (0 or 1)

Explain Code: select K randomly and loop from 2 to 11 , in each iteration save inertia in lest after end loop compare all inertias and plot Elbow to select right K.



Result:

-The Inertia: 15865.289326214694

-The silhouette score is: 0.9999067077152719

-Visualize the cluster:

B) Logistic Regression:

Goal: Detect Gender

Explain Code: We don't need to sum image features; we used dataset with all image features.

Result:

Train Score: 0.8769703960015379

Test Score: 0.7649769585253456

Classes is: [0 1]

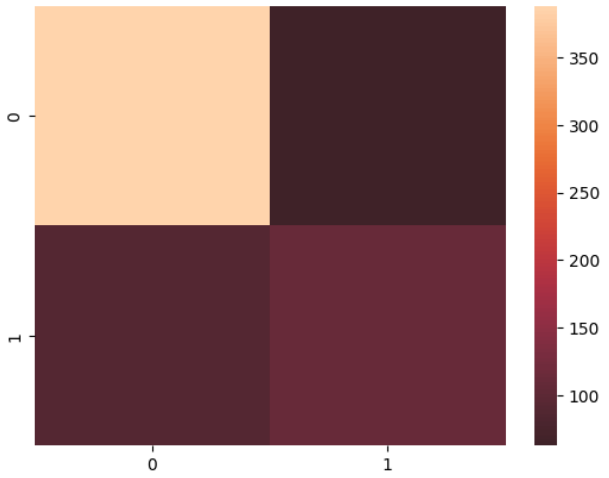
Iterations is: [212]

Intercept is: [-5.56254548]

Confusion Matrix:

[[388 63]

[84 116]]



Accuracy Score: 0.7649769585253456

Precision Score is: 0.7649769585253456

Precision Recall Score is: (0.7649769585253456, 0.7649769585253456, 0.7649769585253456, None)

Precision Value is: [0.30721966 0.64804469 1.]

Recall Value is: [1. 0.58 0.]

Thresholds Value is: [0 1]

AUC Value: 0.7201552106430156

fpr Value: [0. 0.13968958 1.]

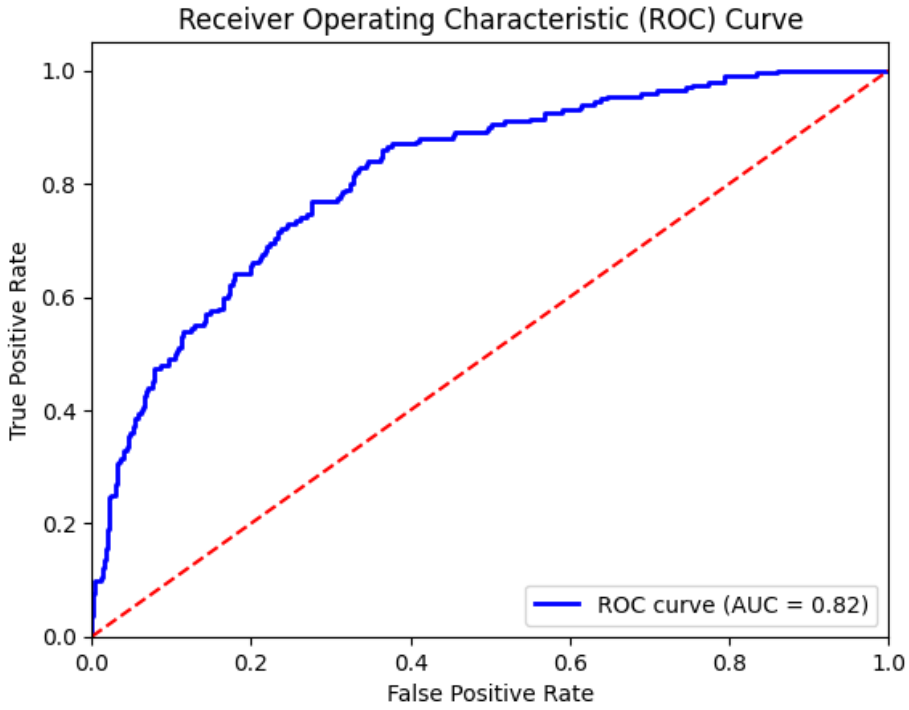
tpr Value: [0. 0.55 1.]

thresholds Value: [inf 1. 0.]

ROCAUC Score: 0.7051552106430156

Zero One Loss Value: 153

ROC curve:



[2] California housing prices

Dataset link:

https://www.kaggle.com/datasets/camnugent/california-housing-prices

About Dataset:

Context

This is the dataset used in the second chapter of Aurélien Géron's recent book 'Hands-On Machine learning with Scikit-Learn and TensorFlow'. It serves as an excellent introduction to implementing machine learning algorithms because it requires rudimentary data cleaning, has an easily understandable list of variables and sits at an optimal size between being to toyish and too cumbersome.

The data contains information from the 1990 California census. So although it may not help you with predicting current housing prices like the Zillow Zestimate dataset, it does provide an accessible introductory dataset for teaching people about the basics of machine learning.

Content

The data pertains to the houses found in a given California district and some summary stats about them based on the 1990 census data. Be warned the data aren't cleaned so there are some preprocessing steps required! The columns are as follows, their names are pretty self explanitory:

longitude

latitude

housing_median_age

total_rooms

total_bedrooms

population

households

median_income

median_house_value

ocean_proximity

Implementation details:

Pre-Processing:

a) Encoding all text into integer by oneHotEncoding algorithm

b) drop all null values

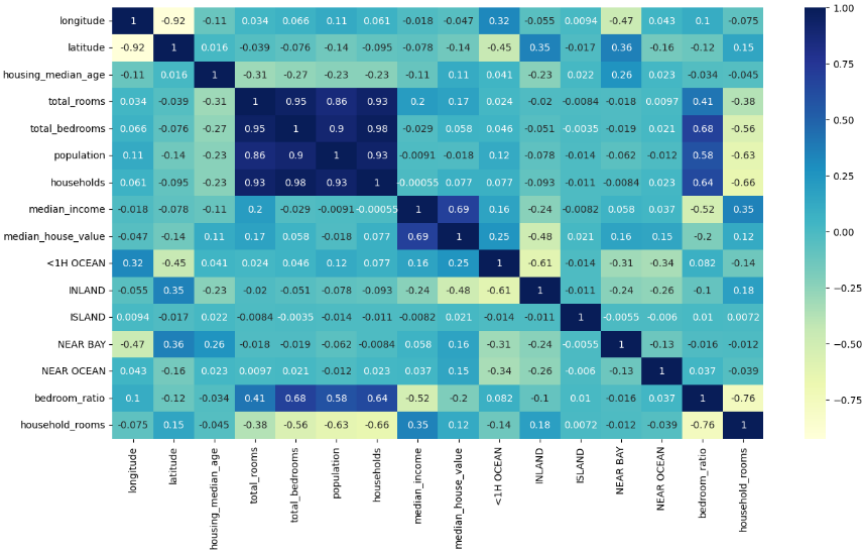
c) Check the correlation to determine which features is important

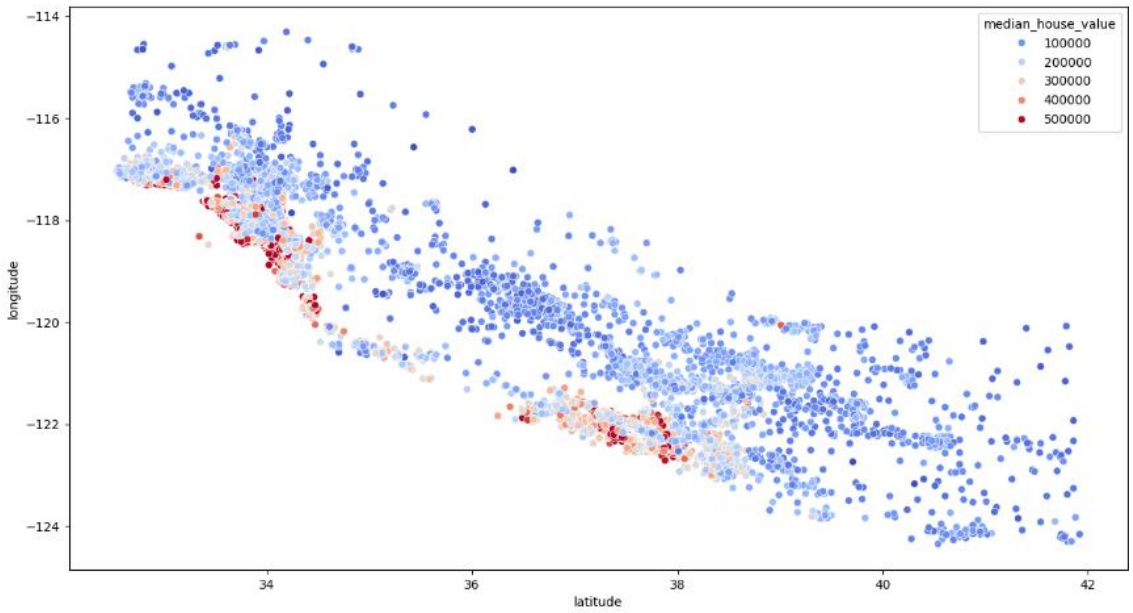
d) scale some features using standard scaler

e) create new features

f) Split data into features and target values

g) Split into train and test by train-test-split where test size is 20% and train size is 80%





a) Linear Regression

Goal: predict the price

Code: used linear regression from sklearn.lineae_model

Results:

Accuracy of linear regression: 0.6656584073714116

- Then we used random forest model to enhance the accuracy

The accuracy of random forest: 0.8128981238870866

- Then we tried to enhance the accuracy by using GridSearchCV algorithm, but we got the same accuracy of random forest.

b) K Nearest Neighbors (KNN)

Goal: predict the price

Code: set K to 9

Result:

The accuracy of knn is : 0.6801231119557607

Team members

ID	Name
20210537	عبدالسلام محمود عبدالسلام محمود
20210229	بلال حنفى محمد ابو العطا
20210077	أحمد عزت عبدالمرضي النساج
20210459	شهاب خلف عبدالمنعم عواد
20210094	أحمد محسن انور سعيد
20210370	زياد محمد أحمد على امبابى