HOSTEL PRICE PREDICTION USING MACHINE LEARNING ALGORITHMS

Table of Contents

Introduction

The data for these hostels was obtained through personal interractions (interviews) with student residents. The dataset contains 151 rows, each reprent a room (i.e number of beds in the room) and 16 hostel features and hostel names. Below is a short description of features:

hostel - name of hostel

location - general location of hostel

grade - average value of how students evaluate their hostel

rank - overrall quality of the hostel

beds - number of beds in a particular room

study_room - a binary variable for wheather the hostel has a study room or not

tv_room - a binary variable for wheather the hostel has a tv room or not

security - a binary variable for wheather the hostel has a security post or not

food_joint - a binary variable for wheather the hostel has a food joint $\leq$ 5min walk from hostel

ext_power - a binary variable for wheather the hostel has either a plant, generator, solar or not

ac - a binary variable for wheather the room has ac installed

proximity - distance from hostel to Aboagye Menyeh Complex, College of Science (KNUST) measured in meters

post_code - post code of area hostel is located

latitude - north-south position of the hostel on Earth's surface

longitude - east-west postion of the hostel on Earth's surface

price2018 - price in Ghana cedis of room for 2018/19 academic year

price2019 - price in Ghana cedis of room for 2019/20 academic year

```
clear, clc
```

Importing Data

```
% generated function to open a partially cleaned data
dataset = importKHorig('knust_orig.csv');
hostels = importKH('knust_hostels.csv');
head(hostels)
```

Data Exploration

Geographical location of hostels

```
% based on general location
geobubble(dataset,"latitude","longitude","ColorVariable","location", ...
    "SizeVariable","proximity")
title('Geographical location of hostels')
% based on area post code
geobubble(dataset,"latitude","longitude","ColorVariable","post_code")
title('Postcode location of hostels')
```

The target variable

```
% distribution of target variable
histogram(hostels.price2020)
title('Distribution of target feature')
xlabel("Price2020 (in cedis)")
txt = {'Skewness:', skewness(hostels.price2020)};
text(9000,72,txt,'HorizontalAlignment','right')
```

```
% numerical features
num_data = hostels(:,["grade","beds","proximity","latitude", ...
    "longitude","price2018","price2019","price2020"]);
% categorical features
cat_data = hostels(:,["location","rank","study_room","tv_room", ...
    "security","food_joint","ext_power","ac","post_code"]);
```

Correlation Matrix

```
corrplot(num_data, ...
    "varNames",{'grade','beds','prox','lat','long','p18','p19','p20'})
% most correlating numerical feature
boxplot(num_data.price2020,num_data.beds)
title('box plot of beds againts target feature')
xlabel("Number of Beds")
ylabel("price2020 (in cedis)")
```

Analysis of variance (ANOVA)

```
rank = cat_data.rank; study = cat_data.study_room;
tv = cat_data.tv_room; security = cat_data.security;
fj = cat_data.food_joint; power = cat_data.ext_power;
ac = cat_data.ac; pc = cat_data.post_code;
loc = cat_data.location;
anovan(hostels.price2020,{loc rank study tv security fj power ac pc}, ...
    'varnames',["location","rank","study_room","tv_room","security", ...
    "food_joint","ext_power","ac","post_code"],'model','linear');
```

Data Preparation

Feature Interpolation

```
% reducing skewness in response feature
hostels.price2020 = log(hostels.price2020);
% distribution of response feature
histogram(hostels.price2020)
title('natural log distribution of target feature')
xlabel("ln(price2020)")
txt = {'Skewness:', skewness(hostels.price2020)};
text(9,37,txt,'HorizontalAlignment','right')
```

Missing data

```
% checking number of missing data
numMissing = nnz(ismissing(hostels));
% summary statistics of features
summary(hostels(:,2:end));
% removing columns with more than 30 missing entries
hostels = rmmissing(hostels, 2, "MinNumMissing", 30);
% filling missing data
hostels.grade = fillmissing(hostels.grade, "constant", ...
    mean(hostels.grade,'omitnan'));
hostels.rank = fillmissing(hostels.rank, "nearest");
```

Outliers

```
boxplot(hostels.price2020,"Orientation","horizontal")
title('Boxplot of price2020')
ylabel("ln(Price2020)")
% we'll use 99% of the dataset to reduce the
% effect of huge values behaving like outliers
hostels = rmoutliers(hostels,"percentiles",[0,99], ...
    "DataVariables","price2020");
histogram(hostels.price2020)
title('Distribution of 99% of target feature')
xlabel("ln(Price2020)")
txt = {'Skewness:', skewness(hostels.price2020)};
text(8.8,37,txt,'HorizontalAlignment','right')
```

Feature selection

```
% % cosidering AYEDUASE and KOTEI as a submarket
% % % uncomment the 2 lines of codes below and rerun the entire code
% % % to analyse how the models behave on the large submarkets
% submarket = "AYEDUASE";
% hostels = hostels(hostels.location == submarket,:);
hostels = hostels(:,["study_room","tv_room","security", ...
    "ext_power","ac","beds","post_code","rank","price2020"]);
```

Feature Engineering

Categorical features

```
% ordinal categorical data
hostels.rank = reordercats(hostels.rank,{'fair','average','good'});
hostels.rank = double(hostels.rank) + 1;
% norminal categorical data
hostels = enCode(hostels);
% post_code and beds features
hostels = toDummy(hostels);
```

Splitting dataset

```
% partition dataset into test and train sets
rng(1) % for reproducibility
cv = cvpartition(size(hostels,1), 'HoldOut', 0.35);
idx = cv.test;
dataTrain = hostels(~idx,:);
dataTest = hostels(idx,:);
% further splitting of train sets and test sets
X_train = dataTrain(:,1:end-1); y_train = dataTrain(:,end);
X_test = dataTest(:,1:end-1); y_test = dataTest(:,end);
```

Predictive Modelling

Machine Learning Algorithms

1. **Multiple Linear Regression**
2. **Ridge Regression**
3. **Neural Network**

Multiple Linear Regression

```
load lrBEST.mat; % loads saved mlr model
% linMdl = fitlm(dataTrain,"linear","ResponseVar","price2020");
beta = linMdl.Coefficients.Estimate;
```

Implementation

```
ypredLM = predict(linMdl,X_test);
plot(table2array(y_test),ypredLM,'o', ...
    ypredLM,ypredLM,'-',"LineWidth",2)
title('Prediction: Multiple Linear Regression Model')
xlabel('True Response')
ylabel('Predicted Response')
legend('Observations','Prediction','Location','northwest')
```

Model Evaluation

Test data

```
fprintf('Evaluation of test data')
[T_lm, resLM] = Metrics(table2array(y_test),ypredLM);
T_lm
```

Train data

```
fprintf('Evaluation of train data')
[T_lm2, ~] = Metrics(table2array(y_train),predict(linMdl,X_train));
T_lm2
```

Analysis of residuals

```
% (1) equal error variance
```

```
sz = 50;
scatter(table2array(y_test),resLM,sz,"filled","red")
xlabel('actual values (price2020)')
ylabel('MLR residuals')
title('Residual plot')
grid on
line([7 9],[0 0],"Color","black","LineWidth",2.5)
% (2) normality of error
[H, p] = ttest(resLM);
normplot(resLM)
grid off;
txt = {'pValue:',p};
text(-0.2,0.97,txt,'HorizontalAlignment','left')
```

Feature Importance

```
x = linMdl.CoefficientNames; x = categorical(x(1,2:end));
y = linMdl.Coefficients.Estimate; y = y(2:end,1);
bar(x,y)
title('MLR model feature importance')
ylabel("Coefficinets, \beta_i")
```

Ridge Regression

```
load regBEST.mat; % loads saved ridge model
% [rMdl, FitInfo] = fitrlinear(table2array(X_train),table2array(y_train), ...
%     "Learner","leastsquares","Regularization",'ridge');
lambda = rMdl.Lambda;
Beta = [rMdl.Bias;rMdl.Beta];
```

Implementation

```
ypredR = predict(rMdl,table2array(X_test));
plot(table2array(y_test),ypredR,'o', ...
    ypredR,ypredR,'-',"LineWidth",2)
title('Prediction: Regularized (Ridge) Regression Model')
xlabel('True Response')
ylabel('Predicted Response')
legend('Observations','Prediction','Location','northwest')
```

Model Evaluation

Test data

```
fprintf('Evaluation of test data')
[T_r, resR] = Metrics(table2array(y_test),ypredR);
T_r
```

Train data

```
fprintf('Evaluation of train data')
[T_r2, ~] = Metrics(table2array(y_train),predict(rMdl,table2array(X_train)));
T_r2
```

Feature Importance

```
y = rMdl.Beta;
bar(x,y)
ylim([-0.4 0.8])
title('RR model feature importance')
ylabel("Coefficinets, \beta_i")
```

Neural Network

```
load nnBEST.mat; % loads saved neural net model
% nnMdl = fitrnet(dataTrain,"price2020","Activations","relu");
weights1 = nnMdl.LayerWeights{1};
weights2 = nnMdl.LayerWeights{2};
biases1 = nnMdl.LayerBiases{1};
biases2 = nnMdl.LayerBiases{2};
```

Implementation

```
ypredNN = predict(nnMdl,X_test);
plot(table2array(y_test),ypredNN,'o', ...
    ypredNN,ypredNN,'-',"LineWidth",2)
title('Prediction: Neural Network Model')
xlabel('True Response')
ylabel('Predicted Response')
legend('Observations','Prediction','Location','northwest')
```

Model Evaluation

Test data

```
fprintf('Evaluation of test data')
[T_nn, resNN] = Metrics(table2array(y_test),ypredNN);
T_nn
```

Train data

```
fprintf('Evaluation of train data')
[T_nn2, ~] = Metrics(table2array(y_train),predict(nnMdl,X_train));
T_nn2
```