Name: Sangeeth Kumar V
Register No: 16BIS0072

Artificial Intelligence with Python

Lab Task – 03 (L39 & L40)

Prof Hemprasad Yashwant Patil

Question:

Download the dataset from the following link:

https://www.kaggle.com/c/house-prices-advanced-regression-techniques/overview

The Ames Housing dataset was compiled by Dean De Cock for use in data science education. It's an incredible alternative for data scientists looking for a modernized and expanded version of the often cited Boston Housing dataset.

Ask a home buyer to describe their dream house, and they probably won't begin with the height of the basement ceiling or the proximity to an east-west railroad. But this playground competition's dataset proves that much more influences price negotiations than the number of bedrooms or a white-picket fence.

With 79 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa, you have to predict the final price of each home.

Goal

Predict the sales price for each house. For each Id in the test set, you have to predict the value of the SalePrice variable.

Metric

Submissions are evaluated on Root-Mean-Squared-Error (RMSE) between the logarithm of the predicted value and the logarithm of the observed sales price. (Taking logs means that errors in predicting expensive houses and cheap houses will affect the result equally.)

Submit the thoroughly commented code along with snapshots of RMSE taken in Spyder environment.

Participation to kaggle website competition is optional.

Code:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Sun Sep 1 04:26:39 2019

@author: astro
"""
# %%
import pandas as pd
```

```python
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_log_error

# %%
# Read csv file
train_file = '~/Documents/AIWP/Lab/LabTask_3\
/house-prices-advanced-regression-techniques\
/train.csv'
data = pd.read_csv(train_file)
summary = data.describe()

# %%
# Select features
y = data.SalePrice
data_features = [x for x in data.columns if str(data[x][0]).isdigit()][:-1]
X = data[data_features]
describe = X.describe()
head = X.head()

# %%
# Define and Fit model
data_model = DecisionTreeRegressor(random_state=1)
data_model.fit(X, y)

# %%
# Predict
test_file = '~/Documents/AIWP/Lab/LabTask_3\
/house-prices-advanced-regression-techniques\
/test.csv'
test_data = pd.read_csv(test_file)
X_test = test_data[data_features]
X_test = pd.DataFrame(X_test).fillna(X_test.mean())
print(X_test)
result = data_model.predict(X_test)
print(result)

# %%
# Evaluate model
submission_file = '~/Documents/AIWP/Lab/LabTask_3\
/house-prices-advanced-regression-techniques\
/sample_submission.csv'
submission_data = pd.read_csv(submission_file)
# RMSE && RMSLE
print('RMSE: ', (mean_absolute_error(
submission_data['SalePrice'], result)
) ** 0.5)
```

```python
print('RMSLE: ', (mean_squared_log_error(
submission_data['SalePrice'], result)
) ** 0.5)
```

Results:

Data:

data - DataFrame

Index	Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	LandSlope
0	1	60	RL	65	8450	Pave	nan	Reg	Lvl	AllPub	Inside	Gtl
1	2	20	RL	80	9600	Pave	nan	Reg	Lvl	AllPub	FR2	Gtl
2	3	60	RL	68	11250	Pave	nan	IR1	Lvl	AllPub	Inside	Gtl
3	4	70	RL	60	9550	Pave	nan	IR1	Lvl	AllPub	Corner	Gtl
4	5	60	RL	84	14260	Pave	nan	IR1	Lvl	AllPub	FR2	Gtl
5	6	50	RL	85	14115	Pave	nan	IR1	Lvl	AllPub	Inside	Gtl
6	7	20	RL	75	10084	Pave	nan	Reg	Lvl	AllPub	Inside	Gtl
7	8	60	RL	nan	10382	Pave	nan	IR1	Lvl	AllPub	Corner	Gtl
8	9	50	RM	51	6120	Pave	nan	Reg	Lvl	AllPub	Inside	Gtl
9	10	190	RL	50	7420	Pave	nan	Reg	Lvl	AllPub	Corner	Gtl
10	11	20	RL	70	11200	Pave	nan	Reg	Lvl	AllPub	Inside	Gtl
11	12	60	RL	85	11924	Pave	nan	IR1	Lvl	AllPub	Inside	Gtl
12	13	20	RL	nan	12968	Pave	nan	IR2	Lvl	AllPub	Inside	Gtl
13	14	20	RL	91	10652	Pave	nan	IR1	Lvl	AllPub	Inside	Gtl
14	15	20	RL	nan	10920	Pave	nan	IR1	Lvl	AllPub	Corner	Gtl
15	16	45	RM	51	6120	Pave	nan	Reg	Lvl	AllPub	Corner	Gtl
16	17	20	RL	nan	11241	Pave	nan	IR1	Lvl	AllPub	CulDSac	Gtl
17	18	90	RL	72	10791	Pave	nan	Reg	Lvl	AllPub	Inside	Gtl
18	19	20	RL	66	13695	Pave	nan	Reg	Lvl	AllPub	Inside	Gtl
19	20	20	RL	70	7560	Pave	nan	Reg	Lvl	AllPub	Inside	Gtl
20	21	60	RL	101	14215	Pave	nan	IR1	Lvl	AllPub	Corner	Gtl

Format Resize ✓ Background color ✓ Column min/max Save and Close Close

Feature Vector summary

Index	Id	MSSubClass	LotFrontage	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtFinSF2	BsmtUnfSF
count	1460	1460	1201	1460	1460	1460	1460	1460	1452	1460	1460	1460
mean	730.5	56.8973	70.05	10516.8	6.09932	5.57534	1971.27	1984.87	103.685	443.64	46.5493	567.24
std	421.61	42.3006	24.2848	9981.26	1.383	1.1128	30.2029	20.6454	181.066	456.098	161.319	441.867
min	1	20	21	1300	1	1	1872	1950	0	0	0	0
25%	365.75	20	59	7553.5	5	5	1954	1967	0	0	0	223
50%	730.5	50	69	9478.5	6	5	1973	1994	0	383.5	0	477.5
75%	1095.25	70	80	11601.5	7	6	2000	2004	166	712.25	0	808
max	1460	190	313	215245	10	9	2010	2010	1600	5644	1474	2336

Features present in train dataset

1. Id
2. MSSubClass
3. MSZoning
4. LotFrontage
5. LotArea
6. Street
7. Alley
8. LotShape
9. LandContour
10. Utilities
11. LotConfig
12. LandSlope
13. Neighborhood
14. Condition1
15. Condition2
16. BldgType
17. HouseStyle
18. OverallQual
19. OverallCond
20. YearBuilt
21. YearRemodAdd
22. RoofStyle
23. RoofMatl
24. Exterior1st
25. Exterior2nd
26. MasVnrType
27. MasVnrArea
28. ExterQual
29. ExterCond
30. Foundation
31. BsmtQual
32. BsmtCond
33. BsmtExposure
34. BsmtFinType1
35. BsmtFinSF1
36. BsmtFinType2
37. BsmtFinSF2
38. BsmtUnfSF
39. TotalBsmtSF
40. Heating
41. HeatingQC
42. CentralAir
43. Electrical
44. 1stFlrSF
45. 2ndFlrSF
46. LowQualFinSF
47. GrLivArea
48. BsmtFullBath

49. BsmtHalfBath
50. FullBath
51. HalfBath
52. BedroomAbvGr
53. KitchenAbvGr
54. KitchenQual
55. TotRmsAbvGrd
56. Functional
57. Fireplaces
58. FireplaceQu
59. GarageType
60. GarageYrBlt
61. GarageFinish
62. GarageCars
63. GarageArea
64. GarageQual
65. GarageCond
66. PavedDrive
67. WoodDeckSF
68. OpenPorchSF
69. EnclosedPorch
70. 3SsnPorch
71. ScreenPorch
72. PoolArea
73. PoolQC
74. Fence
75. MiscFeature
76. MiscVal
77. MoSold
78. YrSold
79. SaleType
80. SaleCondition
81. SalePrice

Out of them, numerical features are taken into consideration

Using DecisionTreeClassifier for model.fit

Console 1/A   00:08:16

```
   ...: data_model.fit(X, y)
Out[6]:
DecisionTreeRegressor(criterion='mse', max_depth=None,
max_features=None,
                      max_leaf_nodes=None,
min_impurity_decrease=0.0,
                      min_impurity_split=None, min_samples_leaf=1,
                      min_samples_split=2,
min_weight_fraction_leaf=0.0,
                      presort=False, random_state=1,
splitter='best')
```

IPython console   History log

Predicted Values

	0
0	127500
1	155000
2	223500
3	178000
4	213500
5	172400
6	176432
7	177500
8	185000

	A	B
1	Id	SalePrice
2	1461	169277.0524984
3	1462	187758.393988768
4	1463	183583.683569555
5	1464	179317.47751083
6	1465	150730.079976501
7	1466	177150.989247307
8	1467	172070.659229164
9	1468	175110.956519547
10	1469	162011.698831665
11	1470	160726.247831419
12	1471	157933.279456005
13	1472	145291.245020389
14	1473	159672.017631819
15	1474	164167.518301885
16	1475	150891.638244053

They indicate the predicted vs actual values from sample_submission.csv

Evaluating model:

```
Console 1/A                                                    00:09:41
    ...: submission_data = pd.read_csv(submission_file)
    ...: # RMSE && RMSLE
    ...: print('RMSE:  ', (mean_absolute_error(
    ...:         submission_data['SalePrice'], result)
    ...: ) ** 0.5)
    ...: print('RMSLE: ', (mean_squared_log_error(
    ...:         submission_data['SalePrice'], result)
    ...: ) ** 0.5)
RMSE:   242.24966850966211
RMSLE:  0.4131776419375089

In [9]:
IPython console    History log
```

Root Mean Square Error
RMSE: 242.24966850966211
Root Mean Square Log Error
RMSLE: 0.4131776419375089

Conclusion:

Thus we predicted the final price of each home in the given test.csv file by training train.csv, and eventually evaluated the model with the sample_submission.csv file.