linear_regression_multi_variable_machine_learning

multivariable

price= m1*area+m2*bedrooms+m3*age+b

price is dependent variable

m1,m2,m3 is coefficients

area,bedrooms,age arer independent variables(features of dependent variable)

y=m1x1+m2x2+m3x3+b

Topics

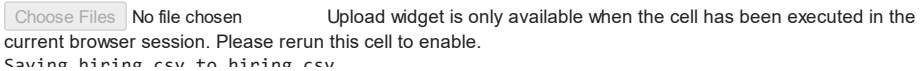
Data Preprocessing : Handling NA value

Linear Regression using Multiple Variables

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")
from sklearn import linear_model
```

before doing or applyiing any algorithm first we need to clean the data or preprocess the data and check if there is any missing or NAN values and deal with it.

```
from google.colab import files
uploaded = files.upload()
```



Saving hiring.csv to hiring.csv

fit is used for train your data and uses training method

```
df=pd.read_csv("hiring.csv")
df
```

	experience	test_score(out of 10)	interview_score(out of 10)	salary($)
0	NaN	8.0	9	50000
1	NaN	8.0	6	45000
2	five	6.0	7	60000
3	two	10.0	10	65000
4	seven	9.0	6	70000
5	three	7.0	10	62000
6	ten	NaN	7	72000
7	eleven	7.0	8	80000

```
df.head(3)
```

	experience	test_score(out of 10)	interview_score(out of 10)	salary($)
0	NaN	8.0	9	50000
1	NaN	8.0	6	45000

```
df.tail(2)
```

	experience	test_score(out of 10)	interview_score(out of 10)	salary($)
6	ten	NaN	7	72000
7	eleven	7.0	8	80000

```
!pip install word2number
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting word2number
  Downloading word2number-1.1.zip (9.7 kB)
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: word2number
  Building wheel for word2number (setup.py) ... done
  Created wheel for word2number: filename=word2number-1.1-py3-none-any.whl size=5582 sha256=4dfec96111d40952d3bc3252adb403666329208b34c64140
  Stored in directory: /root/.cache/pip/wheels/cb/f3/5a/d88198fdeb46781ddd7e7f2653061af83e7adb2a076d8886d6
Successfully built word2number
Installing collected packages: word2number
Successfully installed word2number-1.1
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8 entries, 0 to 7
Data columns (total 4 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   experience                  6 non-null      object 
 1   test_score(out of 10)       7 non-null      float64
 2   interview_score(out of 10)  8 non-null      int64  
 3   salary($)                   8 non-null      int64  
dtypes: float64(1), int64(2), object(1)
memory usage: 384.0+ bytes
```

```
from  word2number import w2n
```

```
df.experience =df.experience.fillna("zero")
df
```

	experience	test_score(out of 10)	interview_score(out of 10)	salary($)
0	zero	8.0	9	50000
1	zero	8.0	6	45000
2	five	6.0	7	60000
3	two	10.0	10	65000
4	seven	9.0	6	70000
5	three	7.0	10	62000
6	ten	NaN	7	72000
7	eleven	7.0	8	80000

```
df.experience= df.experience.apply(w2n.word_to_num)
```

```
df
```

	experience	test_score(out of 10)	interview_score(out of 10)	salary($)
0	0	8.0	9	50000
1	0	8.0	6	45000
2	5	6.0	7	60000
3	2	10.0	10	65000
4	7	9.0	6	70000
5	3	7.0	10	62000
6	10	NaN	7	72000
7	11	7.0	8	80000

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8 entries, 0 to 7
Data columns (total 4 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   experience                  8 non-null      int64  
 1   test_score(out of 10)       7 non-null      float64
 2   interview_score(out of 10)  8 non-null      int64  
 3   salary($)                   8 non-null      int64  
dtypes: float64(1), int64(3)
memory usage: 384.0 bytes
```

```
df.isnull().sum()
```

```
experience                    0
test_score(out of 10)         1
interview_score(out of 10)    0
salary($)                     0
dtype: int64
```

```
import math
mean_test= math.floor(df['test_score(out of 10)'].mean())
mean_test
```

```
7
```

```
df['test_score(out of 10)'] =df['test_score(out of 10)'].fillna(math.floor(df['test_score(out of 10)'].mean()))
df
```

	experience	test_score(out of 10)	interview_score(out of 10)	salary($)
0	0	8.0	9	50000
1	0	8.0	6	45000
2	5	6.0	7	60000
3	2	10.0	10	65000
4	7	9.0	6	70000
5	3	7.0	10	62000
6	10	7.0	7	72000
7	11	7.0	8	80000

```
xtrain=df.iloc[:,0:3]
xtrain
```

	experience	test_score(out of 10)	interview_score(out of 10)
0	0	8.0	9
1	0	8.0	6
2	5	6.0	7
3	2	10.0	10
4	7	9.0	6
5	3	7.0	10
6	10	7.0	7
7	11	7.0	8

```
ytrain = df.loc[:,'salary($)']
ytrain
```

```
0    50000
1    45000
2    60000
3    65000
4    70000
5    62000
6    72000
7    80000
Name: salary($), dtype: int64
```

```
lr=linear_model.LinearRegression()
lr.fit(xtrain,ytrain)
```

```
LinearRegression()
```

```
# 2yr exp,9 test score,6 interview
lr.predict([[2,9,6]])
```

```
array([53713.86677124])
```

```
lr.coef_
```

```
array([2922.26901502, 2221.30909959, 2147.48256637])
```

```
lr.intercept_
```

```
14992.65144669314
```

```
#salary = m1*exp+m2*test+m3*interview+b
2922.26901502*2+2221.30909959*9+2147.48256637*6+14992.65144669314
```

```
53713.86677126314
```

```
# 12 yr exp,10 test score,10 interview
lr.predict([[12,10,10]])
```

```
array([93747.79628651])
```

```
2922.26901502*12+2221.30909959*10+2147.48256637*10+14992.65144669314
```

```
93747.79628653315
```

```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(xtrain,ytrain)
ypred=lr.predict(xtest)
```