regression-pr-04

January 6, 2024

[1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

[3]:
```
df =pd.read_csv('../08-Linear-Regression-Models/Advertising.csv')
```

[6]:
```
df.head()
```

[6]:

	TV	radio	newspaper	sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	9.3
3	151.5	41.3	58.5	18.5
4	180.8	10.8	58.4	12.9

[7]:
```
X=df.drop('sales',axis=1)
```

[8]:
```
y=df['sales']
```

[9]:
```
from sklearn.preprocessing import PolynomialFeatures
```

[10]:
```
convertor=PolynomialFeatures(degree=3,include_bias=False)
```

[11]:
```
poly_features=convertor.fit_transform(X)
```

[12]:
```
from sklearn.model_selection import train_test_split
```

[14]:
```
X_train,X_test,y_train,y_test=train_test_split(poly_features,y,test_size=0.
↪3,random_state=101)
```

[15]:
```
from sklearn.preprocessing import StandardScaler
```

[16]:
```
scaler=StandardScaler()
```

[17]:
```
scaler.fit(X_train)
```

[17]:
```
StandardScaler()
```

[18]:
```
X_train=scaler.transform(X_train)
```

[19]:
```
X_test=scaler.transform(X_test)
```

[20]:
```
from sklearn.linear_model import Ridge
```

[21]:
```
ridge_model=Ridge(alpha=10)
```

[22]:
```
ridge_model.fit(X_train,y_train)
```

[22]:
```
Ridge(alpha=10)
```

[23]:
```
test_predictions=ridge_model.predict(X_test)
```

[24]:
```
from sklearn.metrics import mean_absolute_error,mean_squared_error
```

[25]:
```
MAE=mean_absolute_error(y_test,test_predictions)
```

[26]:
```
MSE=mean_squared_error(y_test,test_predictions)
RMSE=np.sqrt(MSE)
```

[27]:
```
from sklearn.linear_model import RidgeCV
```

[43]:
```
ridge_cv_model=RidgeCV(alphas=(0.1,1.0,10),scoring='neg_mean_absolute_error')
```

[44]:
```
ridge_cv_model.fit(X_train,y_train)
```

[44]:
```
RidgeCV(alphas=(0.1, 1.0, 10), scoring='neg_mean_absolute_error')
```

[45]:
```
ridge_cv_model.alpha_
```

[45]:
```
0.1
```

[40]:
```
from sklearn.metrics import get_scorer_names
```

[42]:
```
get_scorer_names()
```

[42]:
```
['accuracy',
 'adjusted_mutual_info_score',
 'adjusted_rand_score',
 'average_precision',
 'balanced_accuracy',
 'completeness_score',
 'explained_variance',
 'f1',
```

```
'f1_macro',
'f1_micro',
'f1_samples',
'f1_weighted',
'fowlkes_mallows_score',
'homogeneity_score',
'jaccard',
'jaccard_macro',
'jaccard_micro',
'jaccard_samples',
'jaccard_weighted',
'matthews_corrcoef',
'max_error',
'mutual_info_score',
'neg_brier_score',
'neg_log_loss',
'neg_mean_absolute_error',
'neg_mean_absolute_percentage_error',
'neg_mean_gamma_deviance',
'neg_mean_poisson_deviance',
'neg_mean_squared_error',
'neg_mean_squared_log_error',
'neg_median_absolute_error',
'neg_negative_likelihood_ratio',
'neg_root_mean_squared_error',
'normalized_mutual_info_score',
'positive_likelihood_ratio',
'precision',
'precision_macro',
'precision_micro',
'precision_samples',
'precision_weighted',
'r2',
'rand_score',
'recall',
'recall_macro',
'recall_micro',
'recall_samples',
'recall_weighted',
'roc_auc',
'roc_auc_ovo',
'roc_auc_ovo_weighted',
'roc_auc_ovr',
'roc_auc_ovr_weighted',
'top_k_accuracy',
'v_measure_score']
```

[46]:
```
test_predictions=ridge_cv_model.predict(X_test)
```

[47]:
```
MAE=mean_absolute_error(y_test,test_predictions)
RMSE=np.sqrt(mean_squared_error(y_test,test_predictions))
```

[48]:
```
ridge_cv_model.coef_
```

[48]:
```
array([ 5.40769392,  0.5885865 ,  0.40390395, -6.18263924,  4.59607939,
       -1.18789654, -1.15200458,  0.57837796, -0.1261586 ,  2.5569777 ,
       -1.38900471,  0.86059434,  0.72219553, -0.26129256,  0.17870787,
        0.44353612, -0.21362436, -0.04622473, -0.06441449])
```

[49]:
```
ridge_cv_model.best_score_
```

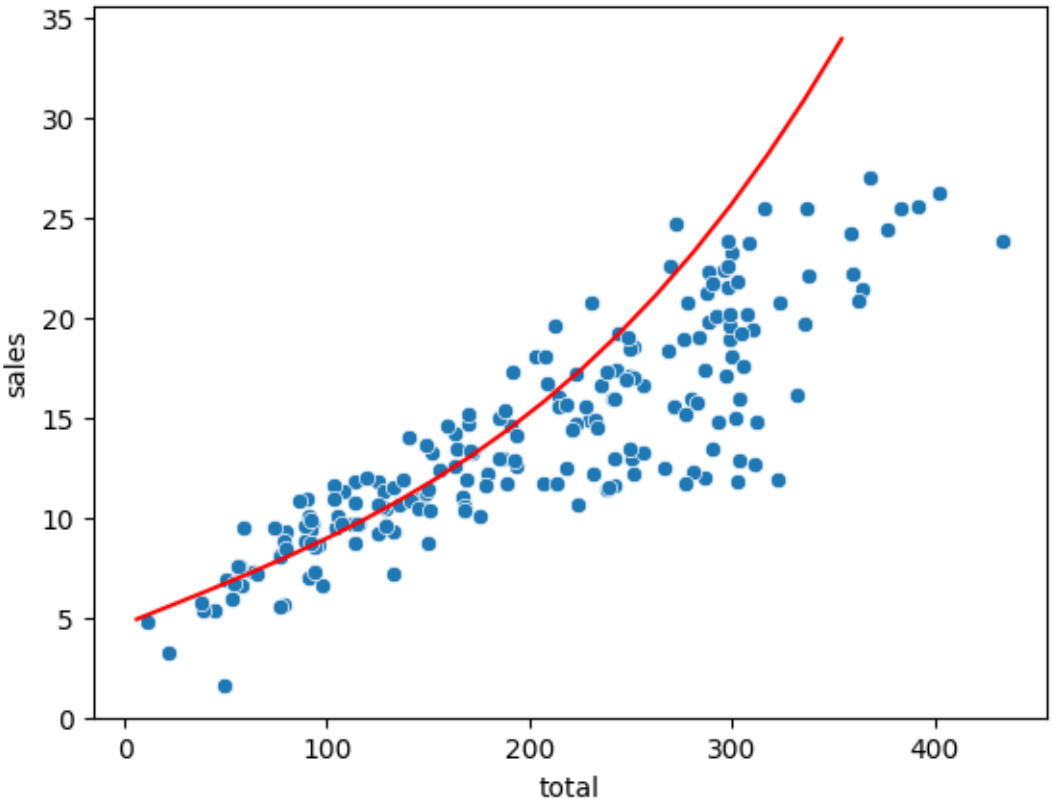
[49]:
```
-0.3749223340292964
```

[68]:
```
plt_data=df.copy()
plt_data['total']=plt_data['TV']+plt_data['radio']+plt_data['newspaper']
test_data=np.linspace(0,120,60).reshape(20,3)
test_data_sum=test_data.sum(axis=1)
test_data=poly_features=convertor.fit_transform(test_data)

test_data=scaler.transform(test_data)
test_data
predictions=ridge_cv_model.predict(test_data)
```

[69]:
```
sns.scatterplot(data=plt_data,x='total',y='sales')
plt.plot(test_data_sum,predictions,color='red')
```

[69]:
```
[<matplotlib.lines.Line2D at 0x7f00f28a7d90>]
```


35
30
25
20
15
10
5
0
sales
0
100
200
300
400
total

[ ]: