logistic-regression-insurance

November 10, 2024

[40]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[41]:
```
plt.rcParams['figure.figsize']=[19,8]
```

[42]:
```
import warnings
warnings.filterwarnings('ignore')
```

[43]:
```
insurance_df=pd.read_csv("C:\\Users\\DSU-CSE513-25\\Downloads\\insurance_data␣
↪(1).csv")
```

[44]:
```
insurance_df.shape
```

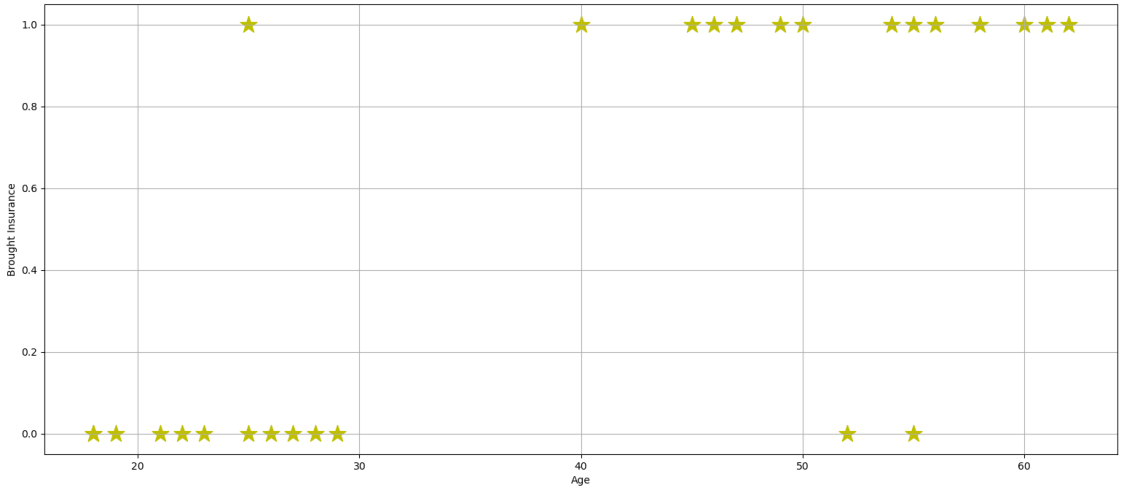
[44]:
```
(27, 2)
```

[45]:
```
insurance_df.head()
```

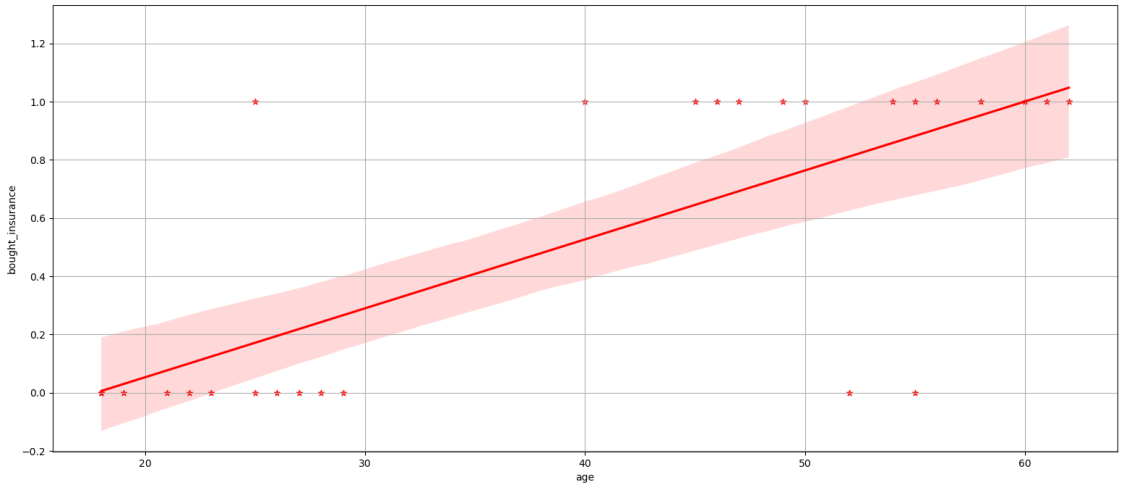
[45]:

	age	bought_insurance
0	22	0
1	25	0
2	47	1
3	52	0
4	46	1

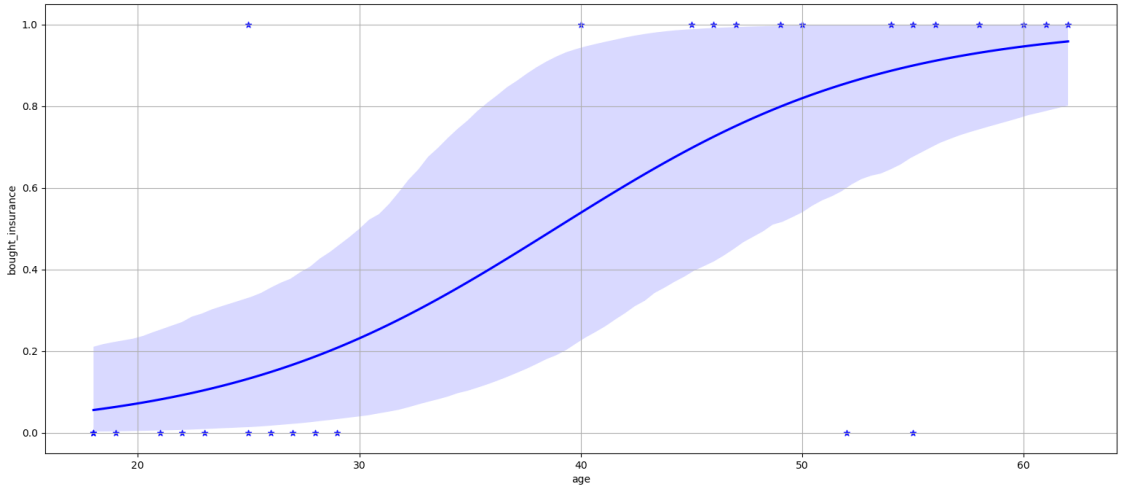
[72]:
```
plt.
↪scatter(data=insurance_df,x='age',y='bought_insurance',marker='*',s=300,color='y')
plt.grid()
plt.xlabel('Age')
plt.ylabel('Brought Insurance')
plt.show()
```



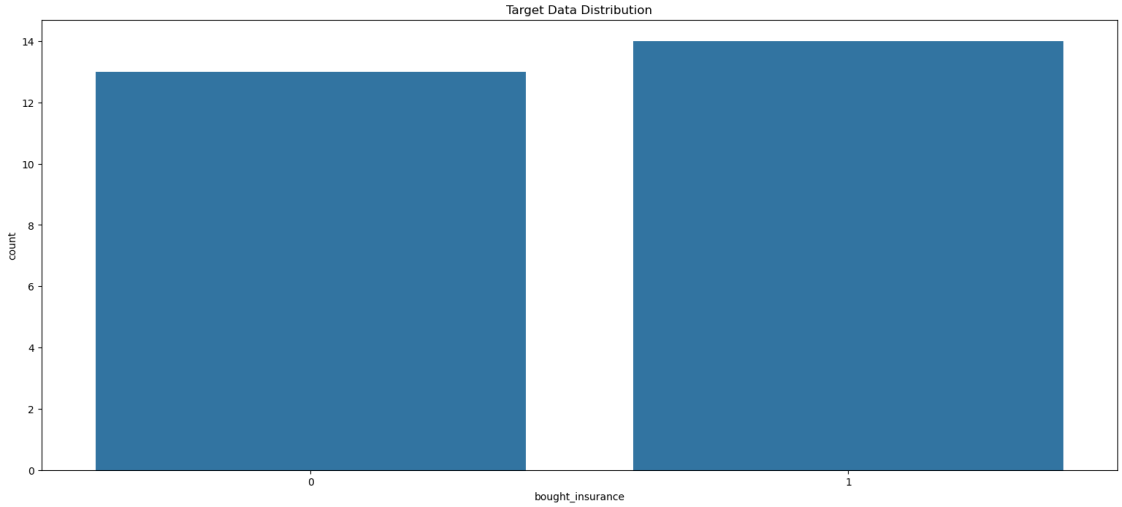
[47]:
```
sns.regplot(data=insurance_df,x='age',y='bought_insurance',marker='*',color='r')
plt.grid()
plt.show()
```



[48]:
```
sns.
↪regplot(data=insurance_df,x='age',y='bought_insurance',logistic=True,marker='*',color='b')
plt.grid()
plt.show()
```



[49]:
```
sns.countplot(data=insurance_df,x='bought_insurance')
plt.title('Target Data Distribution')
plt.show()
```



[50]:
```
#Retrieve the independent and dependent variables
x=insurance_df['age'].values.reshape(-1,1)
y=insurance_df['bought_insurance'].values.reshape(-1,1)
```

[51]:
```
#split the data into train and test
from sklearn.model_selection import train_test_split
```

[52]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=1)
```

[53]:
```
#Model Training
from sklearn.linear_model import LogisticRegression
```

[54]:
```
logit_model=LogisticRegression()
```

[55]:
```
logit_model.fit(x_train,y_train)
```

[55]:
```
LogisticRegression()
```

[56]:
```
#Model Evaluation
logit_model.score(x_train,y_train)
```

[56]:
```
0.9047619047619048
```

[57]:
```
logit_model.score(x_test,y_test)
```

[57]:
```
0.8333333333333334
```

[58]:
```
x_test
```

[58]:
```
array([[58],
       [49],
       [19],
       [52],
       [45],
       [18]], dtype=int64)
```

[59]:
```
y_test
```

[59]:
```
array([[1],
       [1],
       [0],
       [0],
       [1],
       [0]], dtype=int64)
```

[65]:
```
y_predict=logit_model.predict(x_test)
y_predict
```

[65]:
```
array([1, 1, 0, 1, 1, 0], dtype=int64)
```

[66]:
```
logit_model.predict_proba(x_test)
```

[66]:
```
array([[0.04795745, 0.95204255],
       [0.15806064, 0.84193936],
       [0.93775974, 0.06224026],
       [0.10800805, 0.89199195],
```

```
       [0.25198884, 0.74801116],
       [0.9457649 , 0.0542351 ]])
```

[67]:
```
logit_model.predict([[36]])
```

[67]:
```
array([0], dtype=int64)
```

[68]:
```
logit_model.predict([[63]])
```

[68]:
```
array([1], dtype=int64)
```

[69]:
```
from sklearn.metrics import confusion_matrix
```

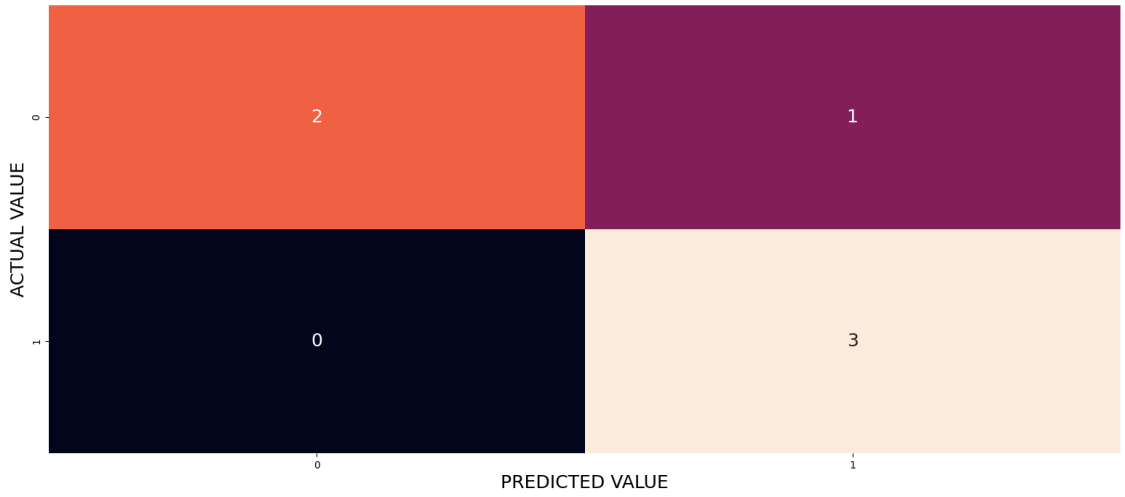
[70]:
```
confusion_matrix=confusion_matrix(y_test,y_predict)
confusion_matrix
```

[70]:
```
array([[2, 1],
       [0, 3]], dtype=int64)
```

[71]:
```
print('Test Data:',y_test.reshape(-1))
print('Predicted:',y_predict)
```

```
Test Data: [1 1 0 0 1 0]
Predicted: [1 1 0 1 1 0]
```

[74]:
```
sns.heatmap(confusion_matrix,annot=True,cbar=False,annot_kws={"fontsize":18})
#sns.set(font_scale=2)
plt.xlabel('PREDICTED VALUE',fontsize=18)
plt.ylabel('ACTUAL VALUE',fontsize=18)
plt.show()
```



[ ]: