Model building

Logistic Regression

In [279…
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix,classification_report
```

In [281…
```
model=LogisticRegression(solver='liblinear')
model
```

Out[281…
```
LogisticRegression(solver='liblinear')
```

In [283…
```
model.fit(x_train,y_train)
```

Out[283…
```
LogisticRegression(solver='liblinear')
```

In [285…
```
y_pred=model.predict(x_test)
y_pred
```

Out[285…
```
array([0, 0, 0, ..., 0, 0, 0])
```

In [289…
```
accuracy_log=accuracy_score(y_test,y_pred)
accuracy_log
```

Out[289…
```
0.8497287337496161
```

In [291…
```
print("Accuracy :",accuracy_log)
```

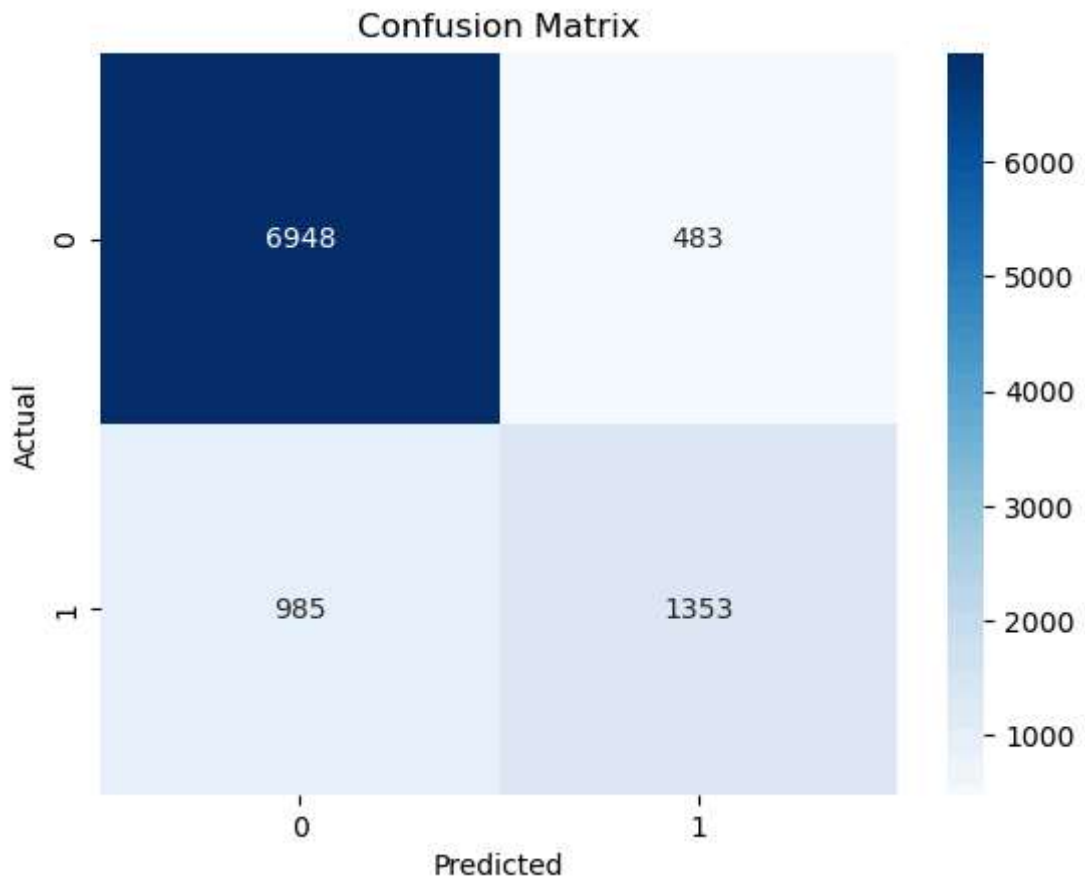
```
Accuracy : 0.8497287337496161
```

In [294…
```
confusion_matrix(y_test,y_pred)
```

Out[294…
```
array([[6948,  483],
       [ 985, 1353]], dtype=int64)
```

In [296…
```
sns.heatmap(confusion_matrix(y_test,y_pred),annot=True,fmt='d',cmap='Blues')
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
```

Out[296…
```
Text(50.722222222222214, 0.5, 'Actual')
```



In [297…
```
print("classification Report :\n")
print(classification_report(y_test,y_pred))
```

```
classification Report :

              precision    recall  f1-score   support

           0       0.88      0.94      0.90      7431
           1       0.74      0.58      0.65      2338

    accuracy                           0.85      9769
   macro avg       0.81      0.76      0.78      9769
weighted avg       0.84      0.85      0.84      9769
```

Random Forest

In [301…
```
from sklearn.ensemble import RandomForestClassifier
```

In [303…
```
model=RandomForestClassifier(n_estimators=50,random_state=100)
model
```

Out[303…
```
RandomForestClassifier(n_estimators=50, random_state=100)
```

In [305…
```
model.fit(x_train,y_train)
```

Out[305…
```
RandomForestClassifier(n_estimators=50, random_state=100)
```

In [306…
```
y_pred=model.predict(x_test)
y_pred
```

Out[306…
```
array([0, 0, 0, ..., 1, 0, 0])
```

In [310…
```
accuracy_rand=accuracy_score(y_test,y_pred)
accuracy_rand
print("Accuracy :",accuracy_rand)
```

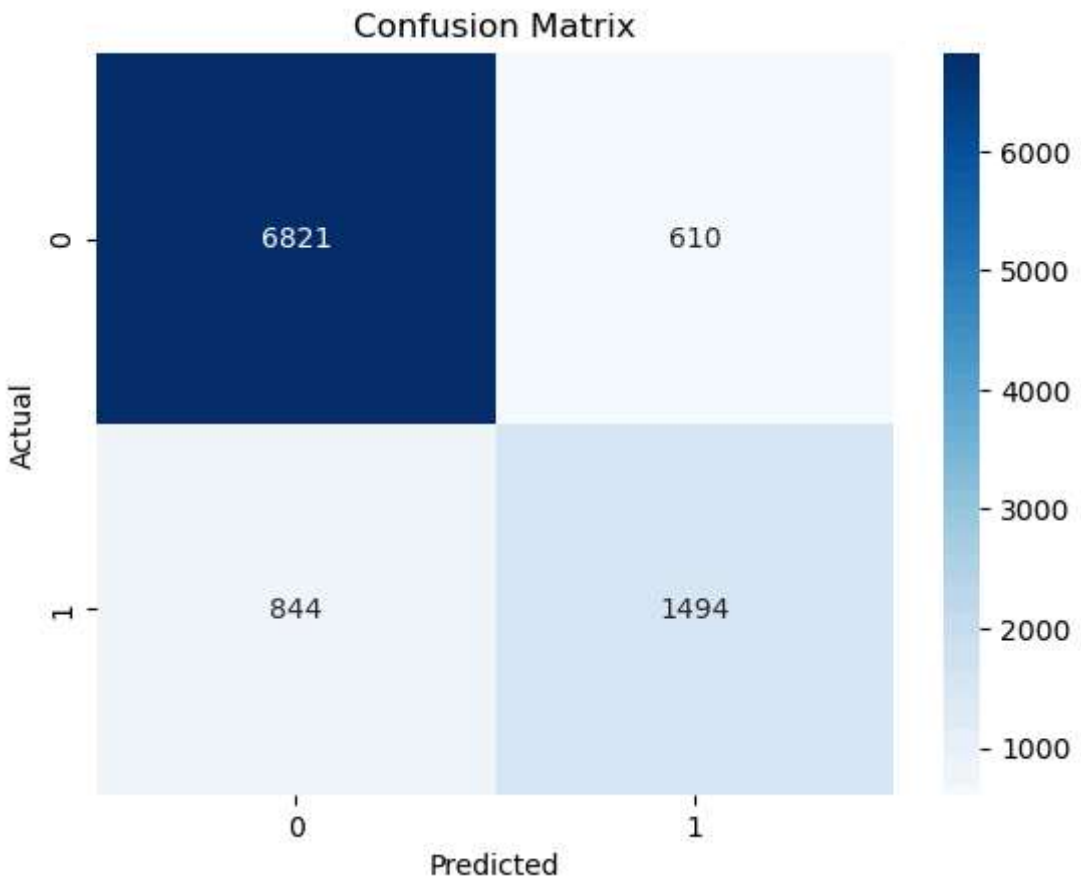
```
Accuracy : 0.8511618384686253
```

In [313…
```
confusion_matrix(y_test,y_pred)
```

Out[313…
```
array([[6821,  610],
       [ 844, 1494]], dtype=int64)
```

In [315…
```
sns.heatmap(confusion_matrix(y_test,y_pred),annot=True,fmt='d',cmap='Blues')
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
```

Out[315…
```
Text(50.722222222222214, 0.5, 'Actual')
```



In [317…
```
print("classification Report :\n")
print(classification_report(y_test,y_pred))
```

```
classification Report :

              precision    recall  f1-score   support

           0       0.89      0.92      0.90      7431
           1       0.71      0.64      0.67      2338

    accuracy                           0.85      9769
   macro avg       0.80      0.78      0.79      9769
weighted avg       0.85      0.85      0.85      9769
```

Decision tree classification

In [320…
```
from sklearn.tree import DecisionTreeClassifier
```

In [322…
```
model=DecisionTreeClassifier(criterion='gini',max_depth=3)
model
```

Out[322…
```
DecisionTreeClassifier(max_depth=3)
```

In [324…
```
model.fit(x_train,y_train)
```

Out[324…
```
DecisionTreeClassifier(max_depth=3)
```

In [326…
```
y_pred=model.predict(x_test)
y_pred
```

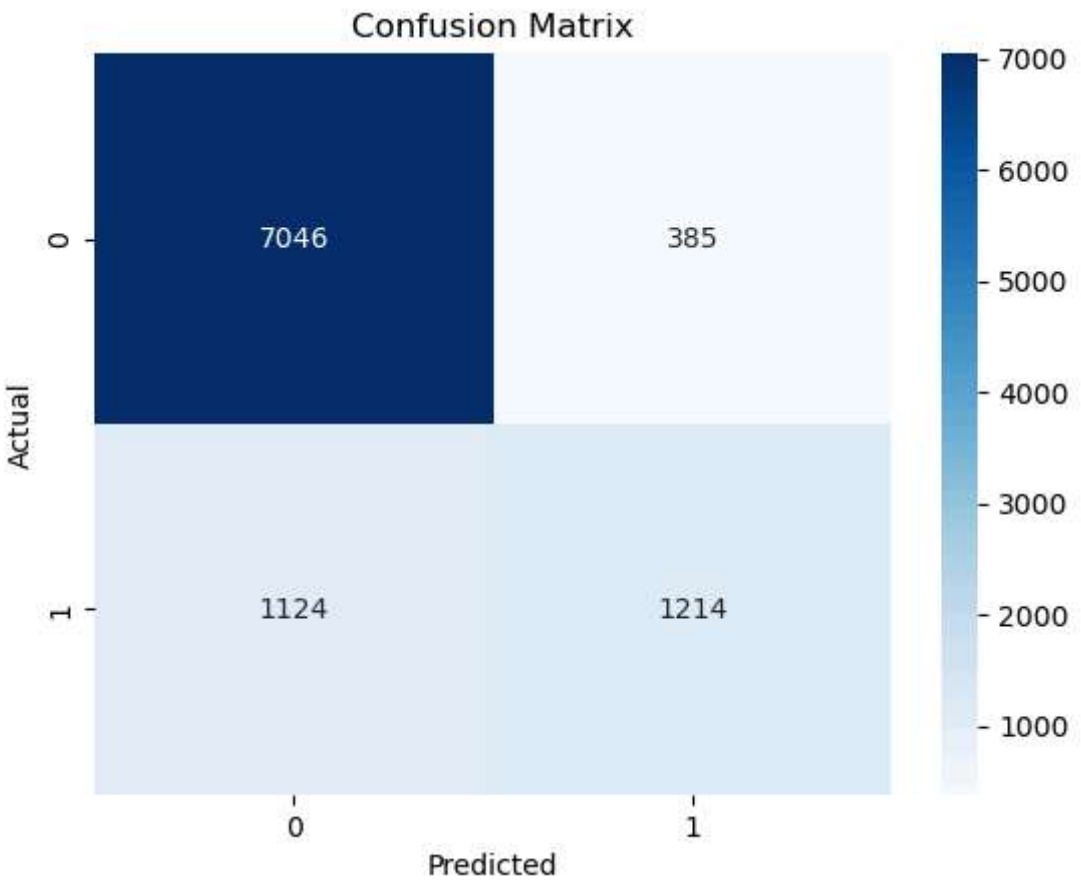
Out[326…
```
array([0, 0, 0, ..., 0, 0, 0])
```

In [329…
```
confusion_matrix(y_test,y_pred)
```

Out[329…
```
array([[7046,  385],
       [1124, 1214]], dtype=int64)
```

In [331…
```
sns.heatmap(confusion_matrix(y_test,y_pred),annot=True,fmt='d',cmap='Blues')
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
```

Out[331…
```
Text(50.722222222222214, 0.5, 'Actual')
```



In [334…
```
accuracy_tree=accuracy_score(y_test,y_pred)
accuracy_tree
```

Out[334…
```
0.8455317842153751
```

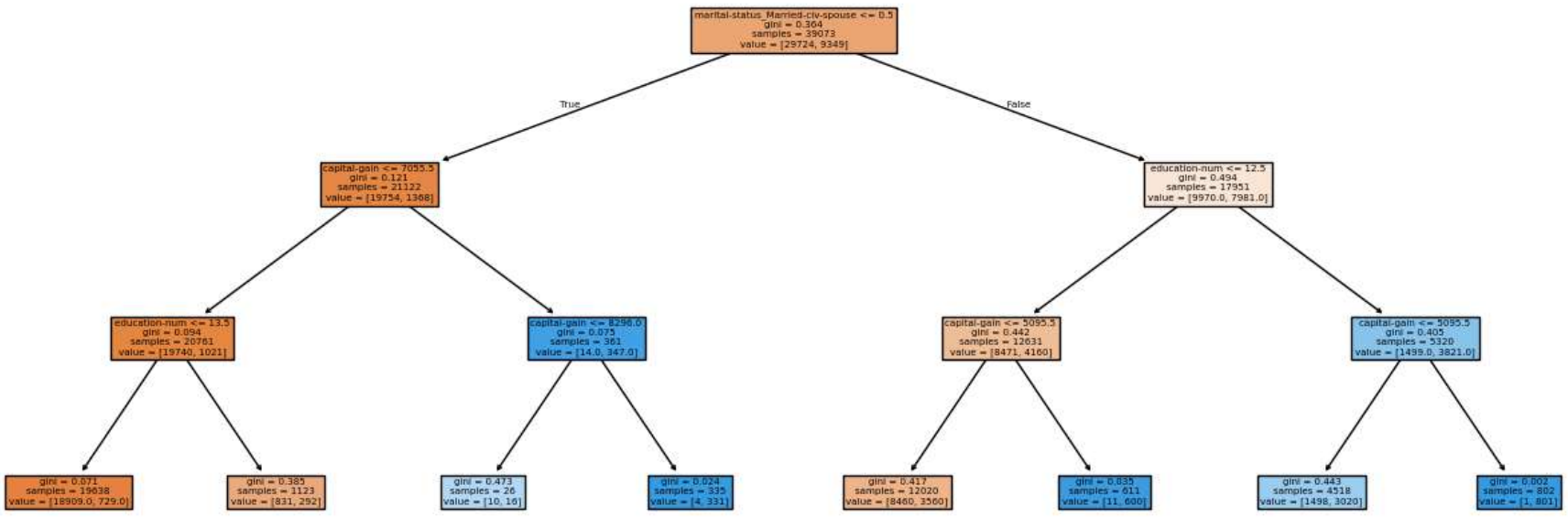
In [336…
```
print("classification Report :\n")
print(classification_report(y_test,y_pred))
```

```
classification Report :

              precision    recall  f1-score   support

           0       0.86      0.95      0.90      7431
           1       0.76      0.52      0.62      2338

    accuracy                           0.85      9769
   macro avg       0.81      0.73      0.76      9769
weighted avg       0.84      0.85      0.83      9769
```

Tree ploting

In [339…
```
from sklearn.tree import plot_tree
```

In [341…
```
plt.figure(figsize=(16,6))
plot_tree(model,feature_names=x.columns,filled=True)
plt.show()
```



K-Nearest Neighbors (KNN)

In [344…
```
from sklearn.neighbors import KNeighborsClassifier
```

In [346…
```
model=KNeighborsClassifier(n_neighbors=3)
model
```

Out[346…
```
KNeighborsClassifier(n_neighbors=3)
```

In [348…
```
model.fit (scaled_x_train,y_train)
```

Out[348…
```
KNeighborsClassifier(n_neighbors=3)
```

In [350…
```
y_pred=model.predict(scaled_x_test)
y_pred
```

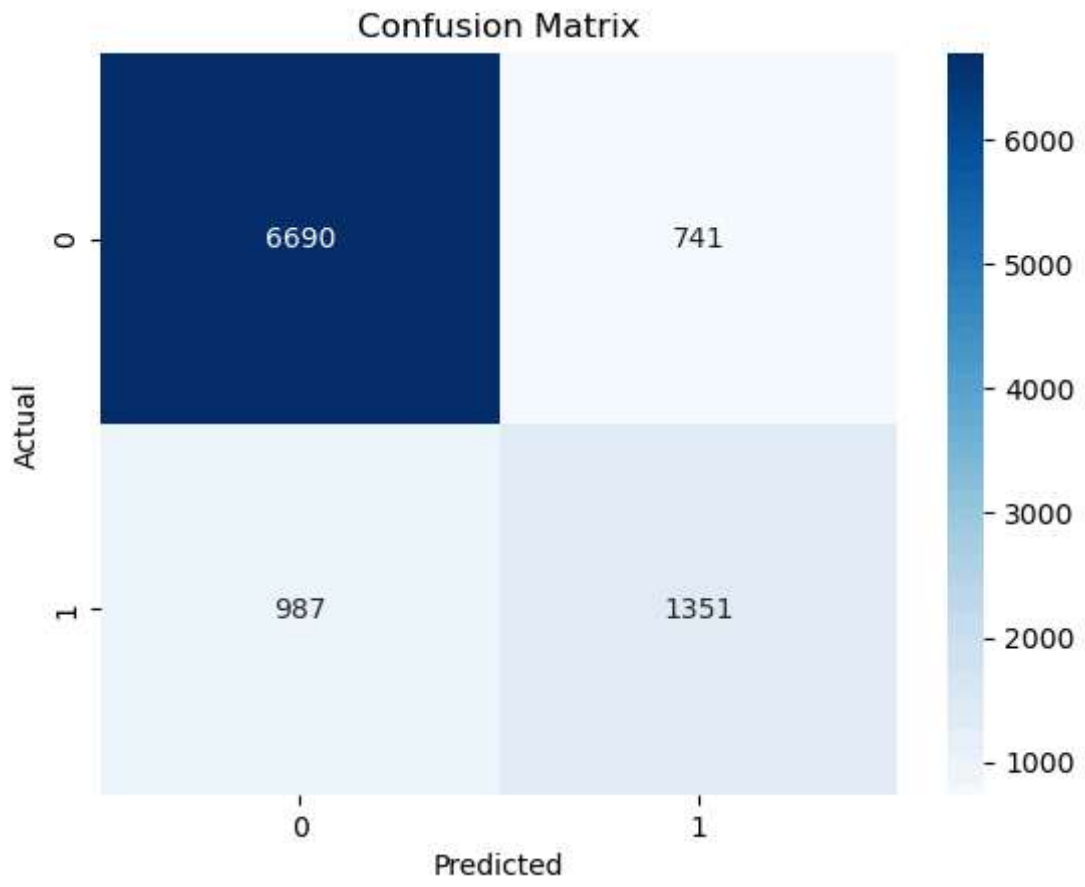
Out[350…
```
array([0, 0, 0, ..., 0, 0, 0])
```

In [352…
```
confusion_matrix(y_test,y_pred)
```

Out[352…
```
array([[6690,  741],
       [ 987, 1351]], dtype=int64)
```

In [353…
```
sns.heatmap(confusion_matrix(y_test,y_pred),annot=True,fmt='d',cmap='Blues')
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
```

Out[353…
```
Text(50.722222222222214, 0.5, 'Actual')
```



In [355…
```
accuracy_knn=accuracy_score(y_test,y_pred)
accuracy_knn
```

Out[355…
```
0.8231139318251612
```

In [356…
```
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.87      0.90      0.89      7431
           1       0.65      0.58      0.61      2338

    accuracy                           0.82      9769
   macro avg       0.76      0.74      0.75      9769
weighted avg       0.82      0.82      0.82      9769
```