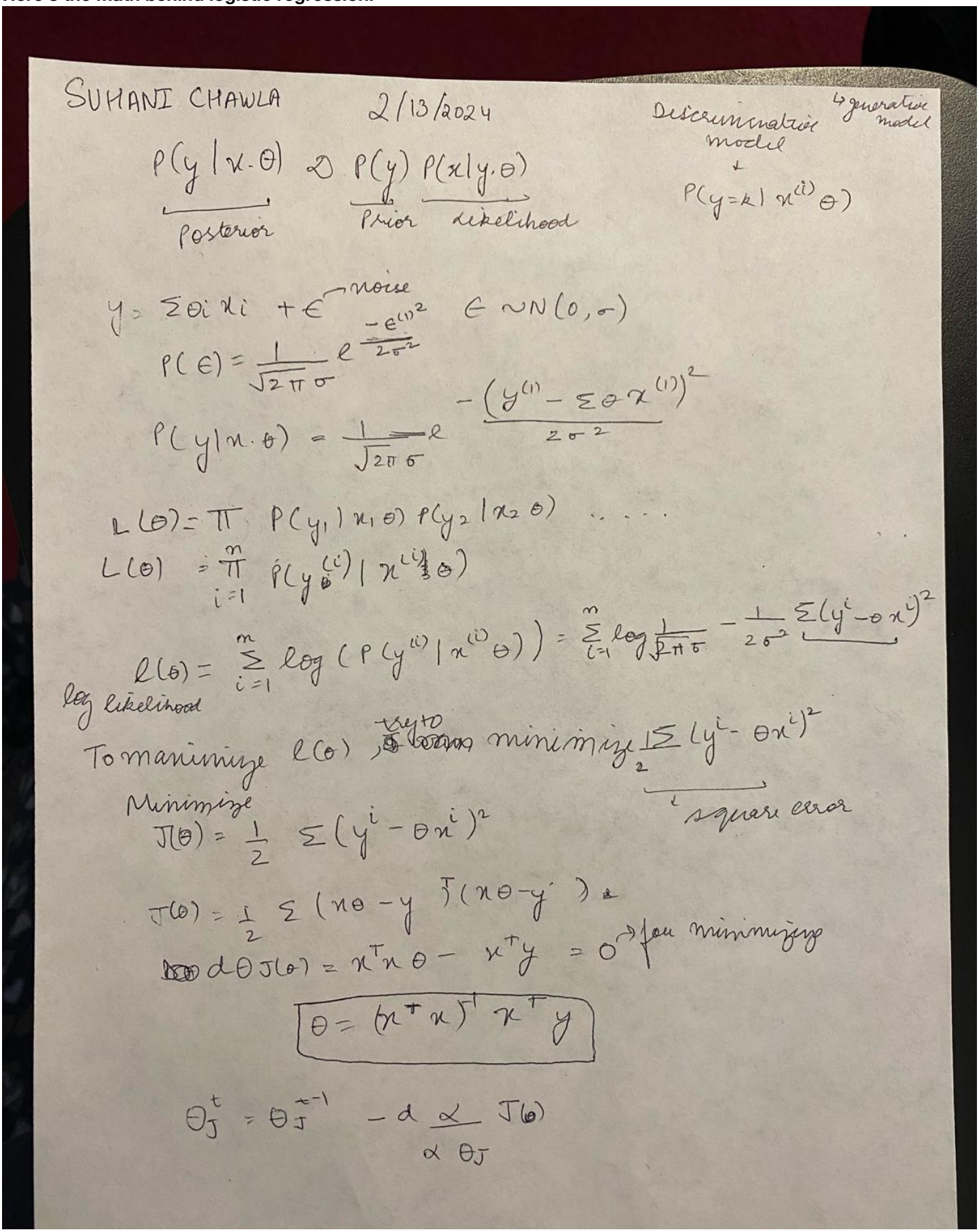
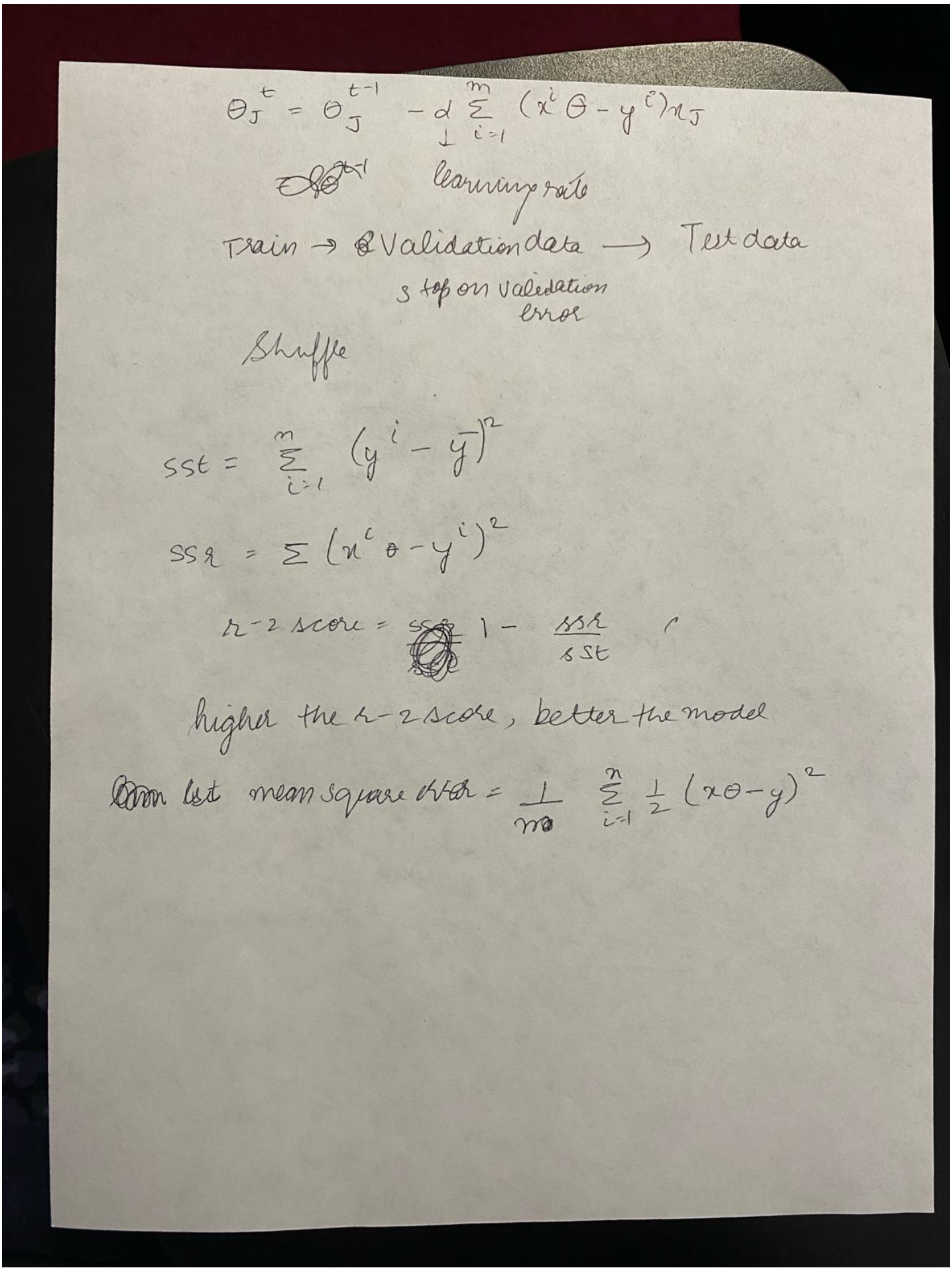
In [106]:

```
import numpy as np
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
import matplotlib.axes as ax
from IPython.display import clear_output
```

Here's the math behind logistic regression:





In [128]:

```
from sklearn import datasets
iris = datasets.load_iris()
```

```
iris_data = iris.data
iris_target = iris.target
len(iris.data)
print(iris_data[:10])
#print(iris_target[49])
```

```
[[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]
 [5.  3.6 1.4 0.2]
 [5.4 3.9 1.7 0.4]
 [4.6 3.4 1.4 0.3]
 [5.  3.4 1.5 0.2]
 [4.4 2.9 1.4 0.2]
 [4.9 3.1 1.5 0.1]]
```

In [130]:

```
feature_names = iris['feature_names']
print(feature_names)
print(iris_target)
```

```
['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2]
```

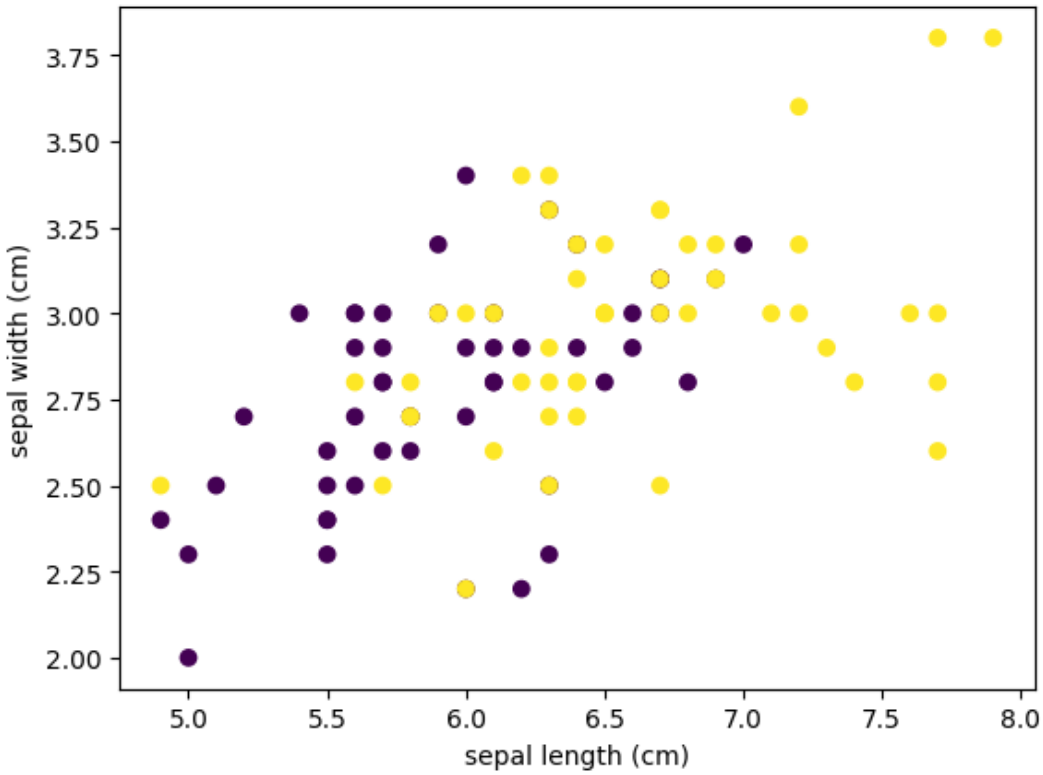
In [115]:

```
iris_data   = iris_data[50:]
iris_target = iris_target[50:]-1 # making 0-1 classification

plt.scatter(iris_data[:,0],iris_data[:,1], c=iris_target)
plt.xlabel(feature_names[0])
plt.ylabel(feature_names[1])
#plt.plot()
```

Out[115]:

```
Text(0, 0.5, 'sepal width (cm)')
```



In [116]:

```
iris_target
```

Out[116]:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1])
```

In [117]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(iris_data, iris_target, test_size=0.
60, random_state=42)
```

In [118]:

```
from sklearn import linear_model
model = linear_model.LogisticRegression(C=1e5, solver='lbfgs')
model.fit(X_train, y_train)
```

Out[118]:

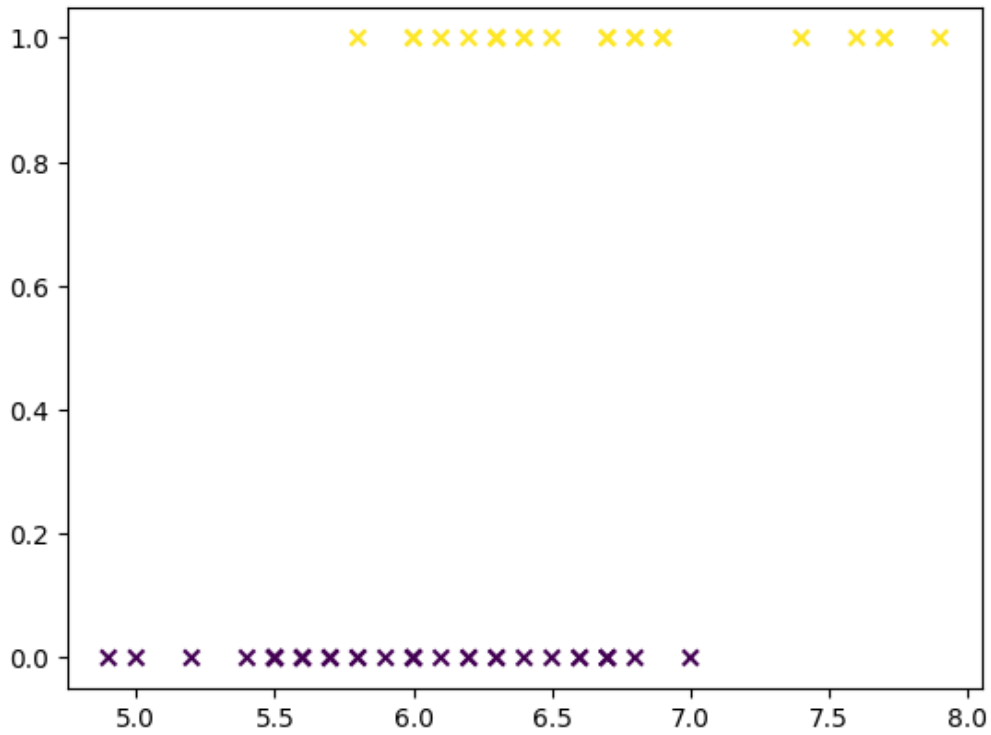
```
▾ LogisticRegression
LogisticRegression(C=100000.0)
```

In [119]:

```
plt.scatter(X_test[:,0],y_test, marker= 'x',c=y_test,label = feature_names[0])
```

Out[119]:

```
<matplotlib.collections.PathCollection at 0x7f502f3a4d00>
```



In [120]:

```
y_pred = model.predict(X_test)
colors = []
for y in y_pred:
    if y==0:
        colors.append([0,0,1])
    else:
        colors.append([1,0,0])
```

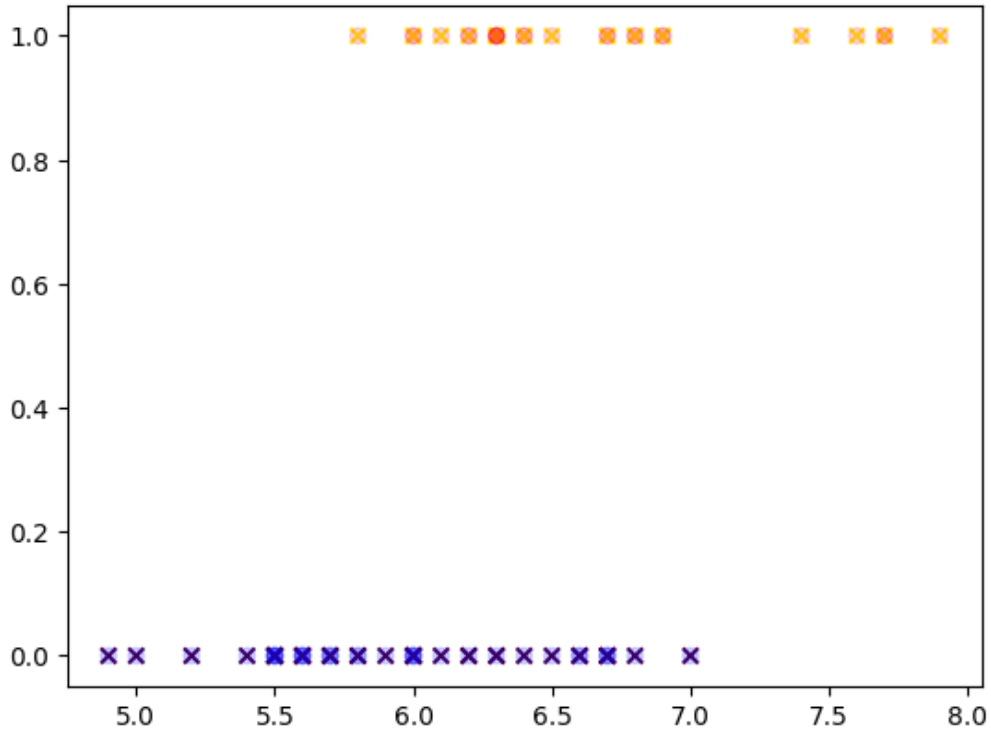
In [121]:

In [121]:

```
plt.scatter(X_test[:,0],y_test, marker='x', c=y_test,label = feature_names[0])
plt.scatter(X_test[:,0],y_pred, marker='o',c= colors,label = feature_names[0],alpha=0.15
)
```

Out[121]:

```
<matplotlib.collections.PathCollection at 0x7f502e301ab0>
```



In [122]:

```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
y_pred
```

```
              precision    recall  f1-score   support

           0       0.97      0.92      0.94        37
           1       0.88      0.96      0.92        23

    accuracy                           0.93        60
   macro avg       0.93      0.94      0.93        60
weighted avg       0.94      0.93      0.93        60
```

Out[122]:

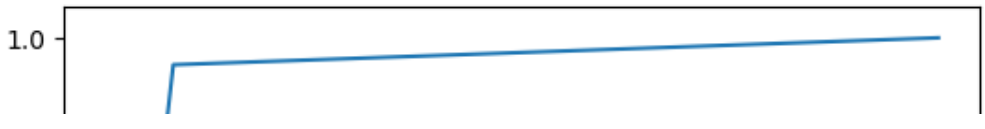
```
array([1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0,
       0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1,
       0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1])
```

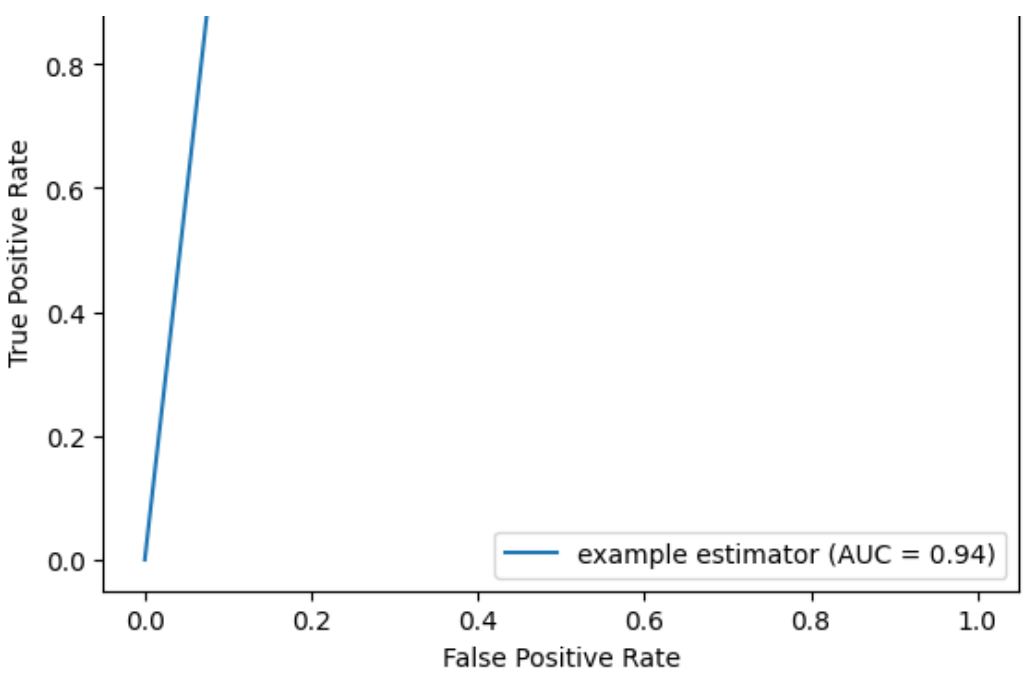
In [123]:

```
from sklearn import metrics
fpr, tpr, thresholds = metrics.roc_curve(y_test, y_pred)
roc_auc = metrics.auc(fpr, tpr)
display = metrics.RocCurveDisplay(fpr=fpr, tpr=tpr, roc_auc=roc_auc, estimator_name='exa
mple estimator')
display.plot()
```

Out[123]:

```
<sklearn.metrics._plot.roc_curve.RocCurveDisplay at 0x7f502fb89d50>
```





In [125]:

```
metrics.confusion_matrix(y_test, y_pred)
```

Out[125]:

```
array([[34,  3],
       [ 1, 22]])
```

In [126]:

```
y_test
```

Out[126]:

```
array([1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1,
       0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1,
       0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1])
```

In [127]:

```
y_pred
```

Out[127]:

```
array([1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0,
       0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1,
       0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1])
```

In [ ]:

```

```