```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

Загрузка и первичный анализ данных

```
from google.colab import files

uploaded = files.upload()

for filename in uploaded.keys():
  print('Uploaded file "{name}" with length
bytes'.format(name='Anaemia Prediction Dataset.csv',
length=len(uploaded['Anaemia Prediction Dataset.csv'])))
```

```
<IPython.core.display.HTML object>
```

```
Saving Anaemia Prediction Dataset.csv to Anaemia Prediction
Dataset.csv
Uploaded file "Anaemia Prediction Dataset.csv" with length bytes
```

```
data = pd.read_csv('Anaemia Prediction Dataset.csv', sep=",")

data
```


```
{"summary":"{\n  \"name\": \"data\",\n  \"rows\": 500,\n  \"fields\":
[\n    {\n      \"column\": \"Number\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 144,\n        \"min\": 1,\n
\"max\": 500,\n        \"num_unique_values\": 500,\n
\"samples\": [\n          362,\n          74,\n          375\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Sex\",\n      \"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 4,\n        \"samples\": [\n          \"F\",\n
\"F \",\n          \"M\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"%Red Pixel\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 2.9500168444205075,\n
\"min\": 36.8,\n        \"max\": 56.85,\n
\"num_unique_values\": 404,\n        \"samples\": [\n          45.92,\
n          41.95,\n          50.47\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"%Green pixel\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
1.8446234284497123,\n        \"min\": 24.15,\n        \"max\": 33.6,\n
\"num_unique_values\": 358,\n        \"samples\": [\n          27.94,\
n          28.22,\n          28.3\n        ],\n
```

\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"%Blue pixel\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
2.2159130731812993,\n   \"min\": 17.95,\n   \"max\": 31.3,\n
\"num_unique_values\": 371,\n   \"samples\": [\n   27.02,\
n   27.69,\n   24.08\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Hb\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 3.04094282361787,\n
\"min\": 4.0,\n   \"max\": 18.55,\n   \"num_unique_values\":
345,\n   \"samples\": [\n   4.83,\n   7.13,\n
15.56\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Anaemic\",\n   \"properties\": {\n   \"dtype\":
\"category\",\n   \"num_unique_values\": 2,\n   \"samples\":
[\n   \"No\",\n   \"Yes\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe","variable_name":"data"}

```
data = data.drop(columns='Number')
data
```

{"summary":"{\n \"name\": \"data\",\n \"rows\": 500,\n \"fields\":
[\n   {\n   \"column\": \"Sex\",\n   \"properties\": {\n
\"dtype\": \"category\",\n   \"num_unique_values\": 4,\n
\"samples\": [\n   \"F\",\n   \"F \",\n   \"M\"\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"%Red Pixel\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
2.9500168444205075,\n   \"min\": 36.8,\n   \"max\": 56.85,\n
\"num_unique_values\": 404,\n   \"samples\": [\n   45.92,\
n   41.95,\n   50.47\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"%Green pixel\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
1.8446234284497123,\n   \"min\": 24.15,\n   \"max\": 33.6,\n
\"num_unique_values\": 358,\n   \"samples\": [\n   27.94,\
n   28.22,\n   28.3\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"%Blue pixel\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
2.2159130731812993,\n   \"min\": 17.95,\n   \"max\": 31.3,\n
\"num_unique_values\": 371,\n   \"samples\": [\n   27.02,\
n   27.69,\n   24.08\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Hb\",\n   \"properties\": {\n
\"dtype\": \"number\",\n   \"std\": 3.04094282361787,\n
\"min\": 4.0,\n   \"max\": 18.55,\n   \"num_unique_values\":
345,\n   \"samples\": [\n   4.83,\n   7.13,\n
15.56\n   ],\n   \"semantic_type\": \"\",\n

```
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Anaemic\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          \"No\",\n          \"Yes\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"data"}
```

```
data.Sex
```

```
0       M
1       F
2       F
3       F
4      M
       ..
495     F
496     F
497    F
498     F
499    F
Name: Sex, Length: 500, dtype: object
```

```
data.Sex.unique()
```

```
array(['M', 'F', 'M ', 'F '], dtype=object)
```

```
data['Sex'] = data['Sex'].map({'F': 0, 'F ': 0, 'M': 1, 'M ': 1})
```

```
data.Sex.unique()
```

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

```
data.Anaemic.unique()
```

```
array(['Yes', 'No'], dtype=object)
```

```
data['Anaemic'] = data['Anaemic'].map({'Yes': 1, 'No': 0})
```

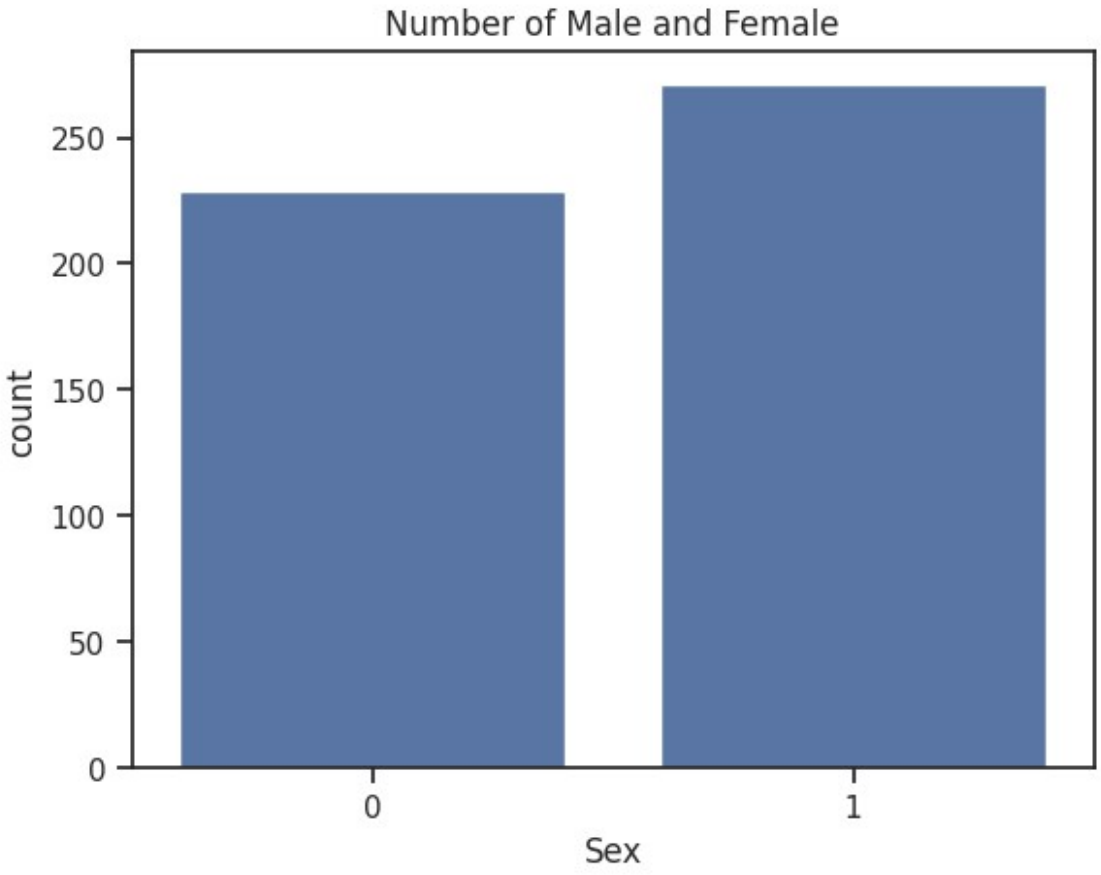
```
data.dtypes
```

```
Sex               int64
%Red Pixel      float64
%Green pixel    float64
%Blue pixel     float64
Hb              float64
Anaemic           int64
dtype: object
```

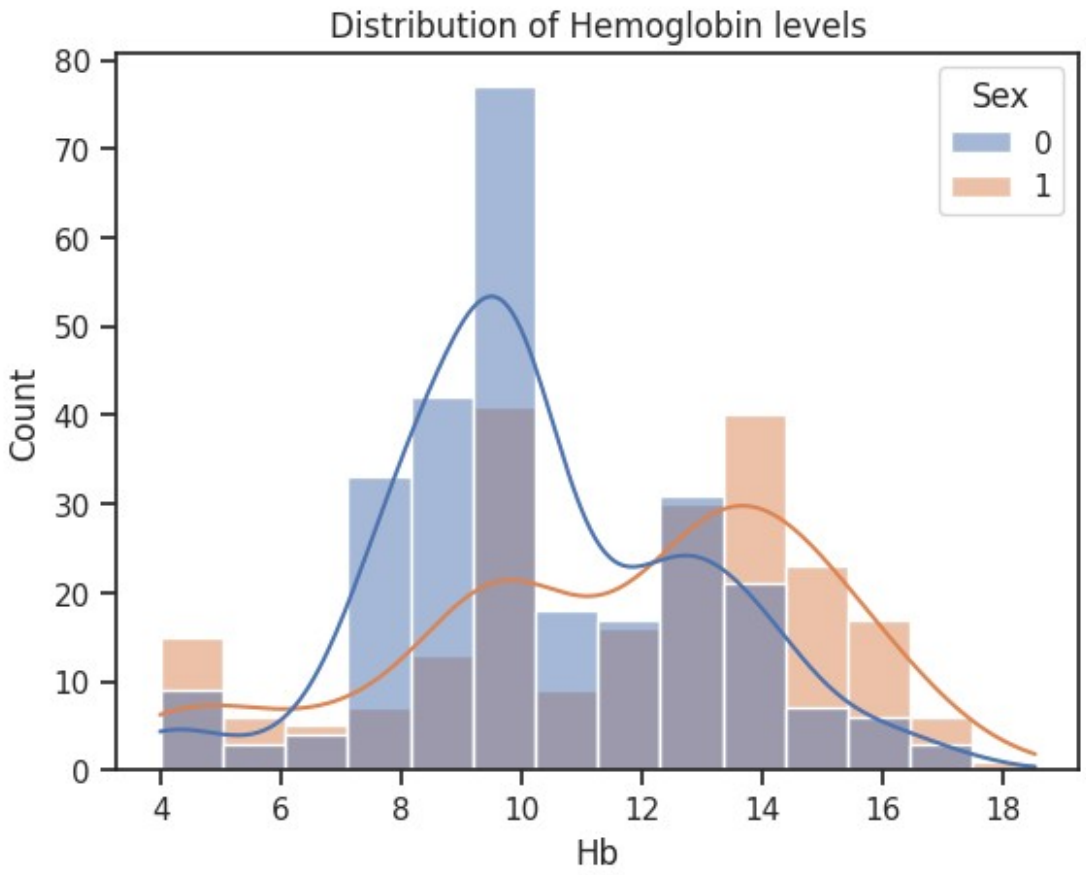
EDA

```
import matplotlib.pyplot as plt
import seaborn as sns
```

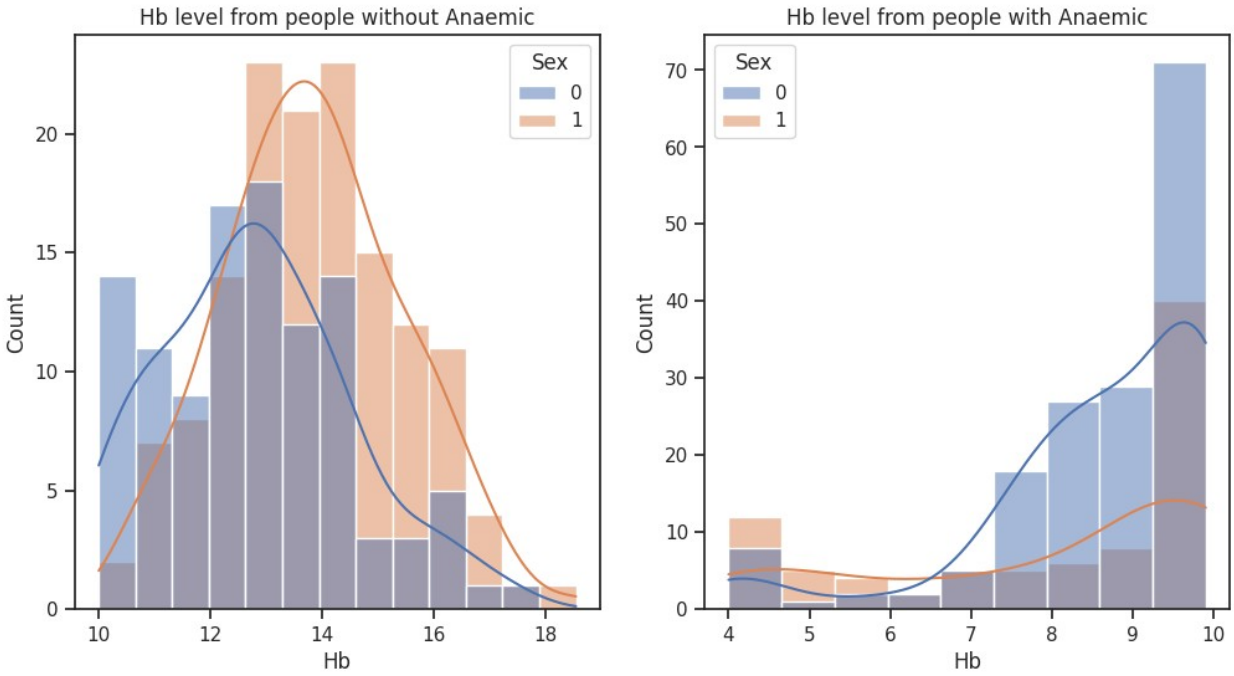
```
gender_count = data.Sex.value_counts() # data['Sex'].value_counts()
sns.barplot(data = data, x = 'Sex', y = gender_count)
plt.title('Number of Male and Female')

Text(0.5, 1.0, 'Number of Male and Female')
```




```
sns.histplot(data=data, x='Hb', hue='Sex', kde=True)
plt.title("Distribution of Hemoglobin levels")

Text(0.5, 1.0, 'Distribution of Hemoglobin levels')
```



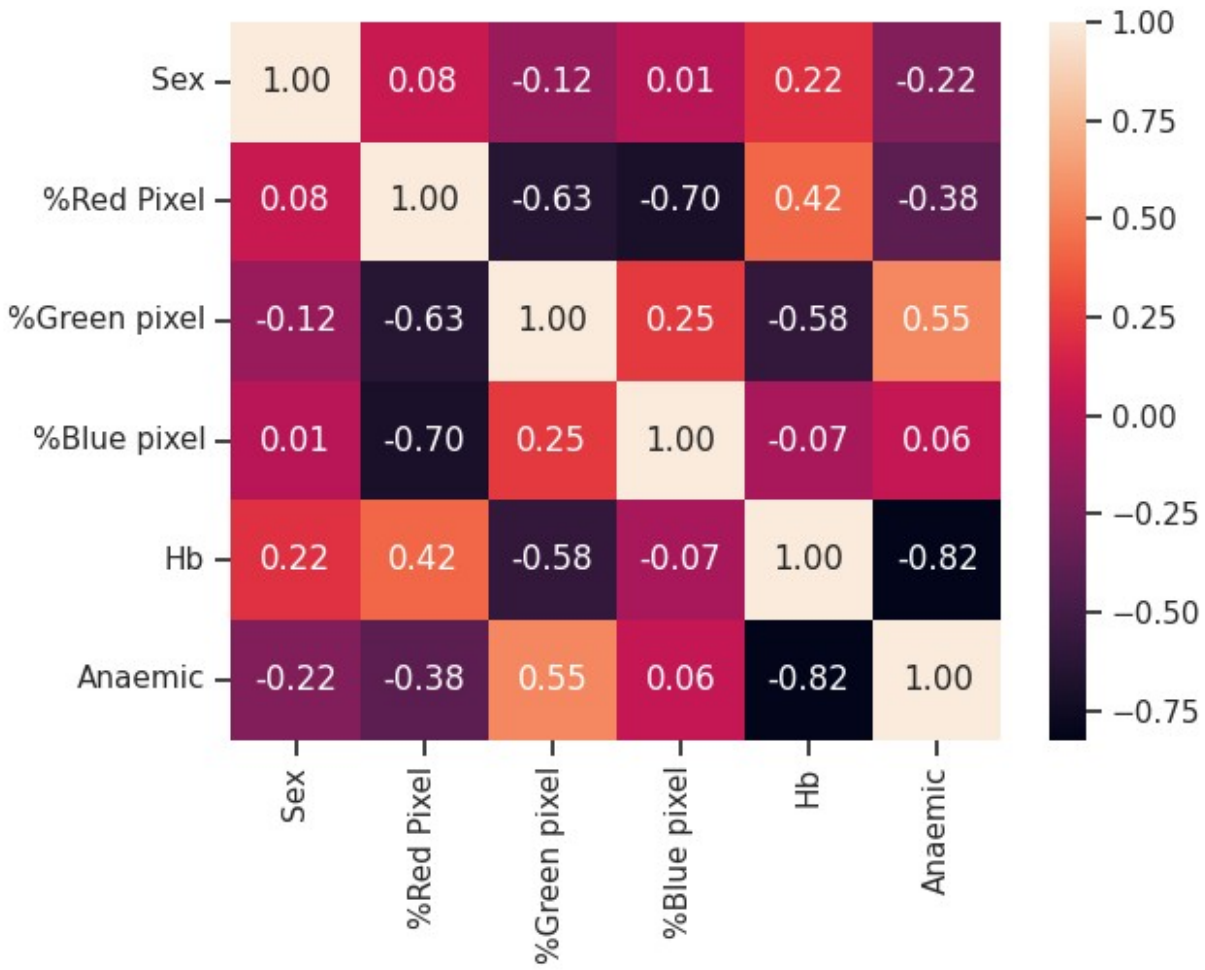

```
anaemic_data = data[data['Anaemic'] == 1]
not_anaemic_data = data[data['Anaemic'] == 0]

plt.figure(figsize=(12, 6))
plt.subplot(1,2,1)
sns.histplot(data=not_anaemic_data, x='Hb', hue='Sex', kde=True)
plt.title('Hb level from people without Anaemic')

plt.subplot(1,2,2)
sns.histplot(data=anaemic_data, x='Hb', hue='Sex', kde=True)
plt.title('Hb level from people with Anaemic')
plt.show()
```



people who didn't diagnose with anaemic have higher Hemoglobin level than people who diagnose with Anaemic.

```
corr = data.corr()

sns.heatmap(corr, fmt='.2f', annot=True)
```

```
<Axes: >
```



Split Dataset

```
X = data.drop(columns='Anaemic')
y = data['Anaemic']

X.shape, y.shape
```

```
((500, 5), (500,))
```

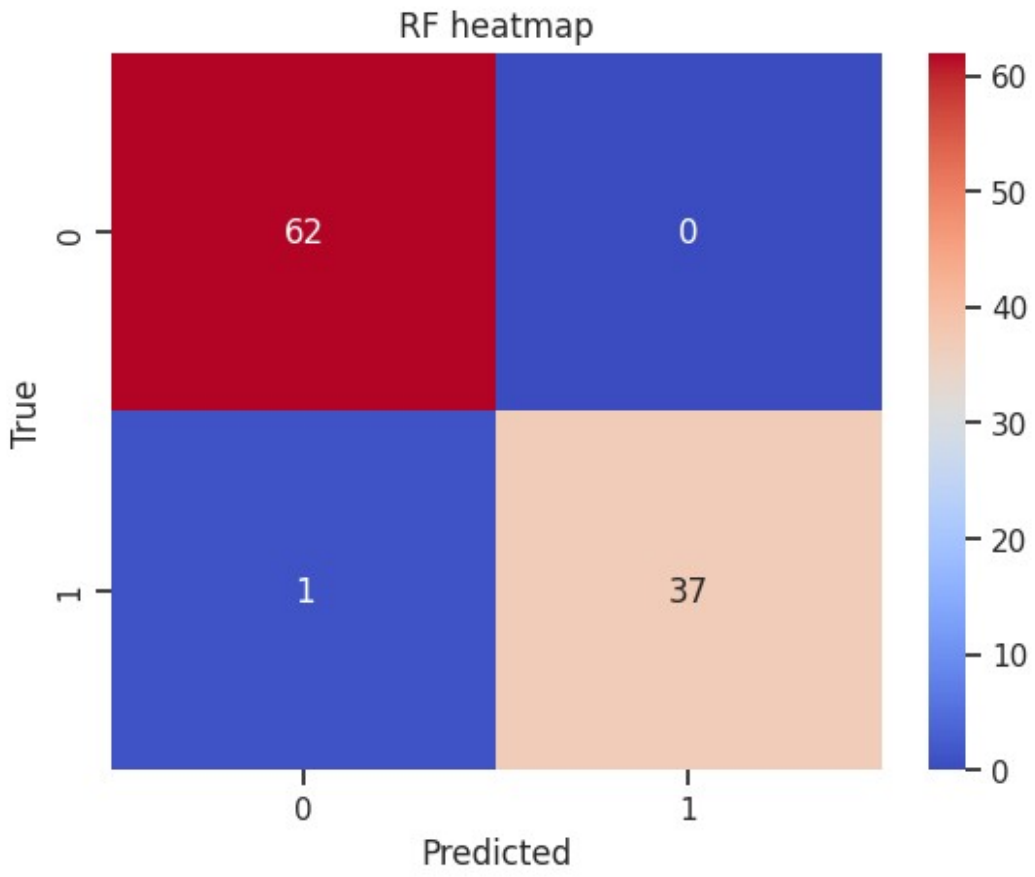
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
```

Define the Model

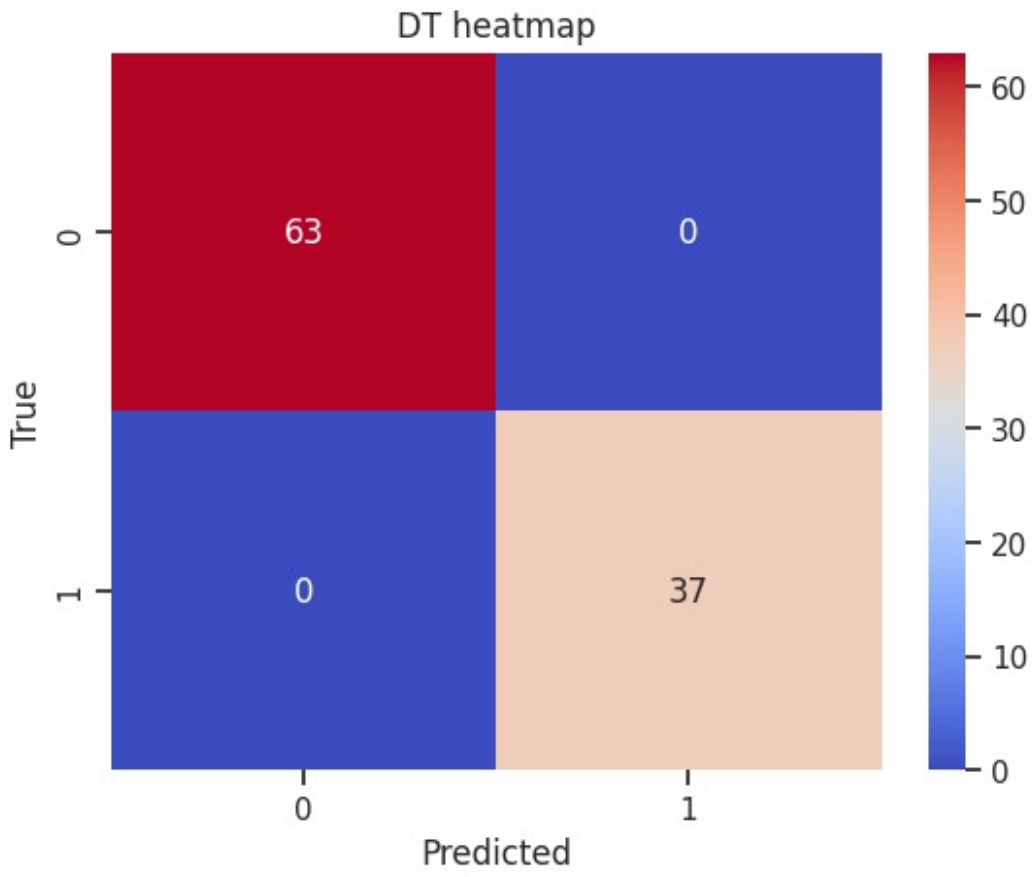
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
```

```python
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix

models = {
    'RF': RandomForestClassifier(),
    'DT': DecisionTreeClassifier(),
    'LogR': LogisticRegression()
}

for model_name, model in models.items():
  model.fit(X_train, y_train)
  y_pred = model.predict(X_test)
  accuracy = accuracy_score(y_pred, y_test)
  print(f"Model name: {model_name}\nAccuracy: {accuracy}\
n{classification_report(y_pred, y_test)}")

  cm = confusion_matrix(y_pred, y_test)
  sns.heatmap(cm, annot=True, fmt='d', cmap='coolwarm')
  plt.xlabel('Predicted')
  plt.ylabel('True')
  plt.title(f"{model_name} heatmap")
  plt.show()
```

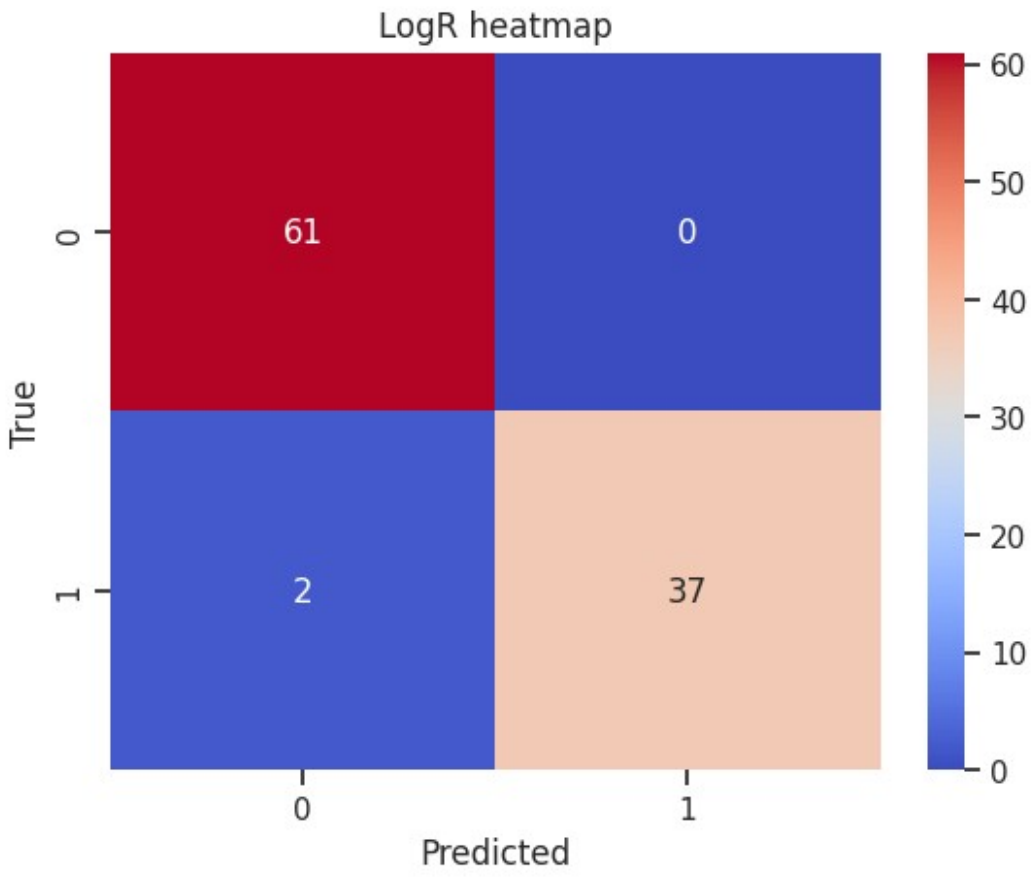
```
Model name: RF
Accuracy: 0.99
              precision    recall  f1-score   support

           0       0.98      1.00      0.99        62
           1       1.00      0.97      0.99        38

    accuracy                           0.99       100
   macro avg       0.99      0.99      0.99       100
weighted avg       0.99      0.99      0.99       100
```




```
Model name: DT
Accuracy: 1.0
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        63
           1       1.00      1.00      1.00        37

    accuracy                           1.00       100
   macro avg       1.00      1.00      1.00       100
weighted avg       1.00      1.00      1.00       100
```




```
Model name: LogR
Accuracy: 0.98
              precision    recall  f1-score   support

           0       0.97      1.00      0.98        61
           1       1.00      0.95      0.97        39

    accuracy                           0.98       100
   macro avg       0.98      0.97      0.98       100
weighted avg       0.98      0.98      0.98       100
```


LogR heatmap
61
0
2
37
True
0
1
Predicted
0
1
60
50
40
30
20
10
0