

```
In [30]: import pandas as pd
import numpy as np
from helpers import plot_histograms, plot

data_path = "tahkeer_data.csv"
all_data = pd.read_csv(data_path)

columns = all_data.columns.tolist()[1:]
columns.remove("smoking")
```

Feature engineering

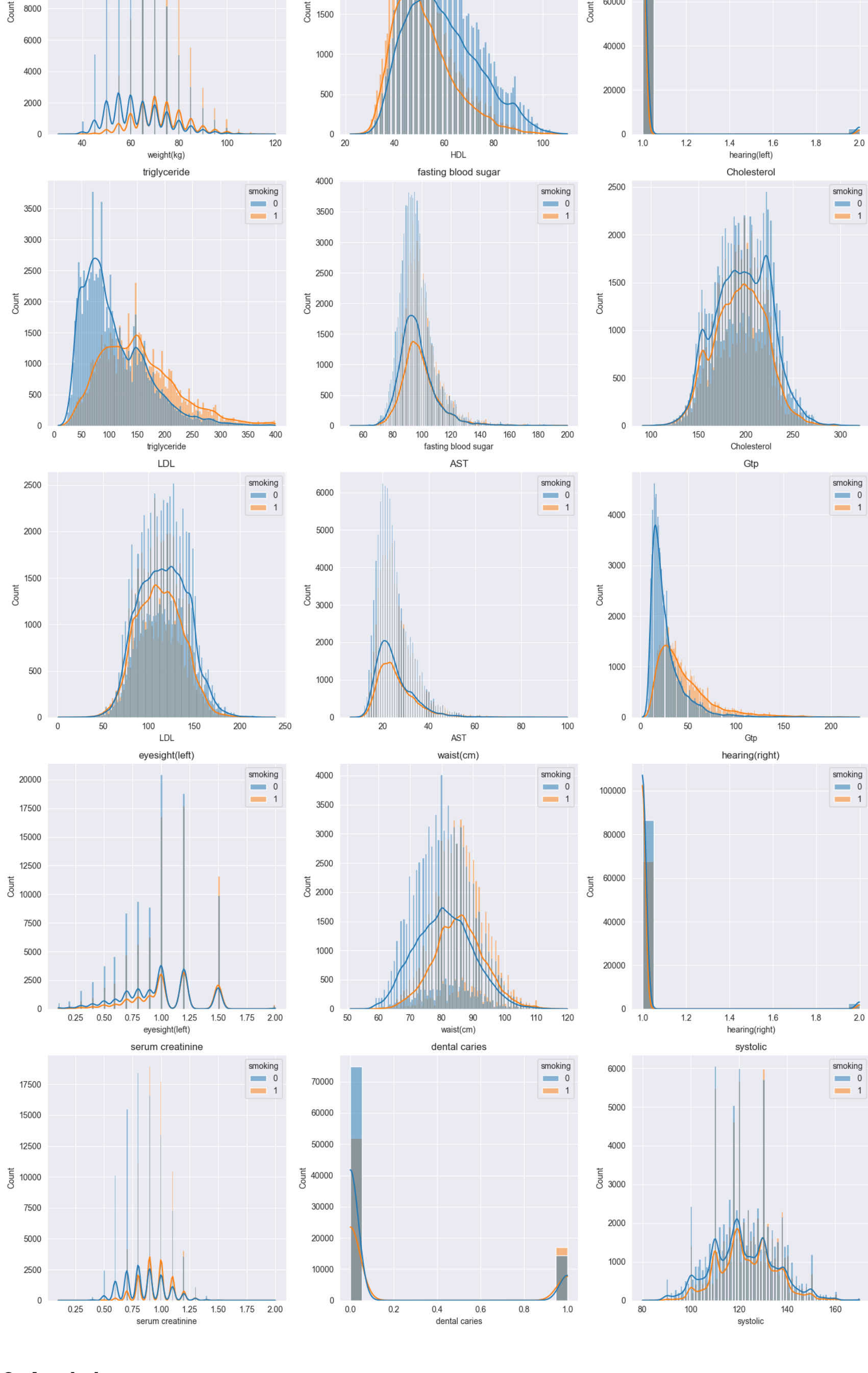
1. Remove outliers

We will remove outliers from the data to reduce the effect of large values on the model.

```
In [31]: columns_to_remove_outliers = {
    "weight(kg)": [30, 120],
    "HDL": [20, 110],
    "triglyceride": [0, 410],
    "fasting blood sugar": [50, 200],
    "Cholesterol": [80, 320],
    "LDL": [0, 260],
    "AST": [0, 100],
    "Gtp": [0, 230],
    "eyesight(left)": [0, 2],
    "waist(cm)": [50, 120],
    "serum creatinine": [0, 2],
    "systolic": [80, 170],
}

for column_name, values in columns_to_remove_outliers.items():
    min_value = values[0]
    max_value = values[1]
    all_data = all_data[(all_data[column_name] >= min_value) & (all_data[column_name] <= max_value)]
```

```
In [32]: plot_histograms(all_data, columns, 3)
```



2. Apply log

We will apply log to the data to reduce the effect of skewed data. For example, the data for Gtp and triglyceride is skewed to the right.

```
In [33]: logged_columns = ['Gtp', 'triglyceride', 'AST', 'LDL', 'Cholesterol', 'serum creatinine']

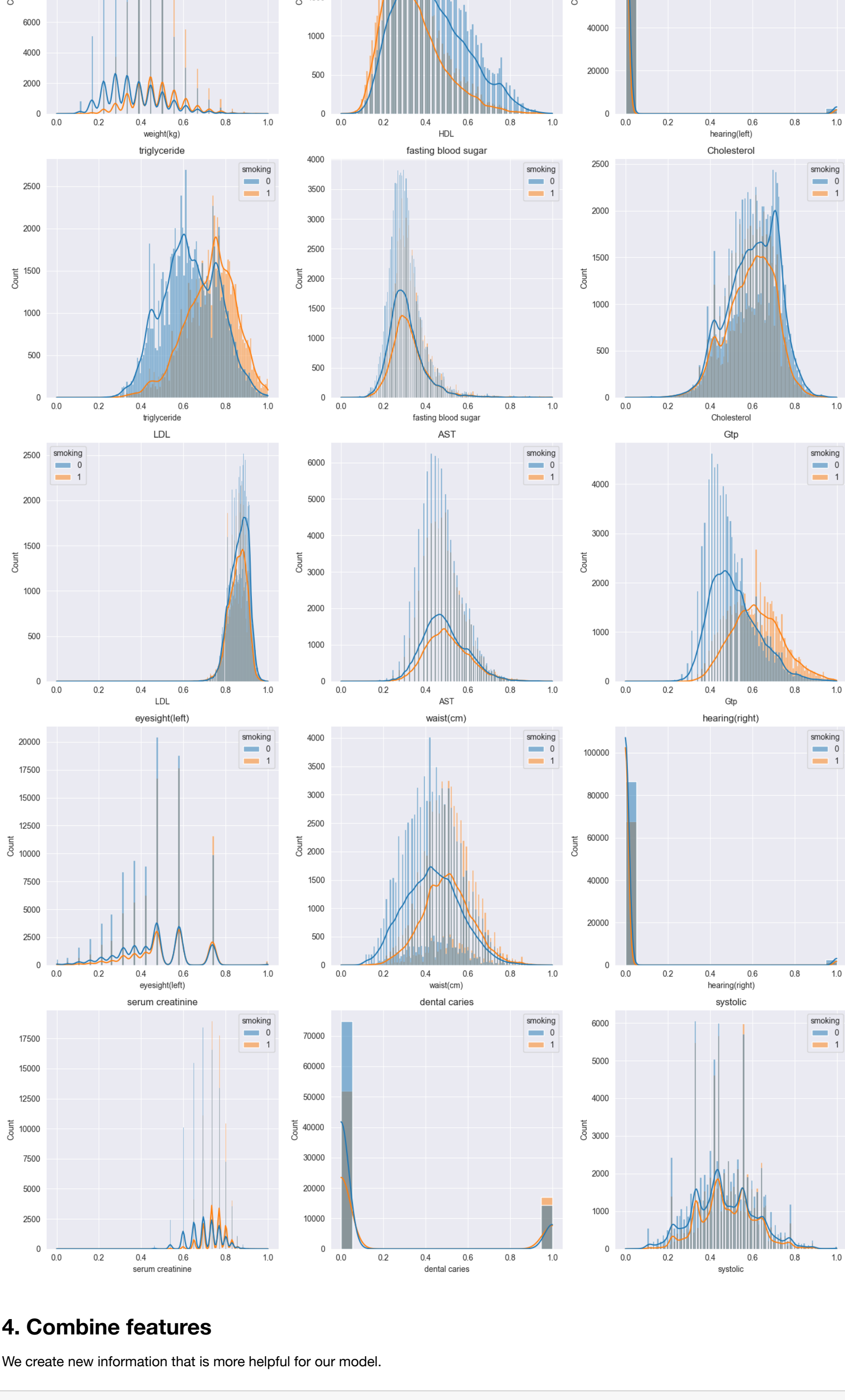
for column in logged_columns:
    all_data[column] = np.log(all_data[column])
```

3. Min-Max normalization

We will use min-max normalization to normalize the data to the range [0, 1] To reduce the effect of features with large values and make the features comparable.

```
In [34]: for column in columns:
    min_value = all_data[column].min()
    max_value = all_data[column].max()
    all_data[column] = (all_data[column] - min_value) / (max_value - min_value)
```

```
plot_histograms(all_data, columns, 3)
```



4. Combine features

We create new information that is more helpful for our model.

```
In [35]: weight = "weight(kg)"
waist = "waist(cm)"
all_data[weight + "_" + waist + "_interaction"] = all_data[weight] * all_data[waist]
columns.append(weight + "_" + waist + "_interaction")
columns.remove(weight)
columns.remove(waist)

cholesterol = "Cholesterol"
ldl = "LDL"
all_data[cholesterol + "_" + ldl + "_aggregation"] = all_data[[cholesterol, ldl]].mean(axis=1)
columns.append(cholesterol + "_" + ldl + "_aggregation")
columns.remove(cholesterol)
columns.remove(ldl)
```

5. Feature selection

We will use random forest to select the most important features and discard the rest.

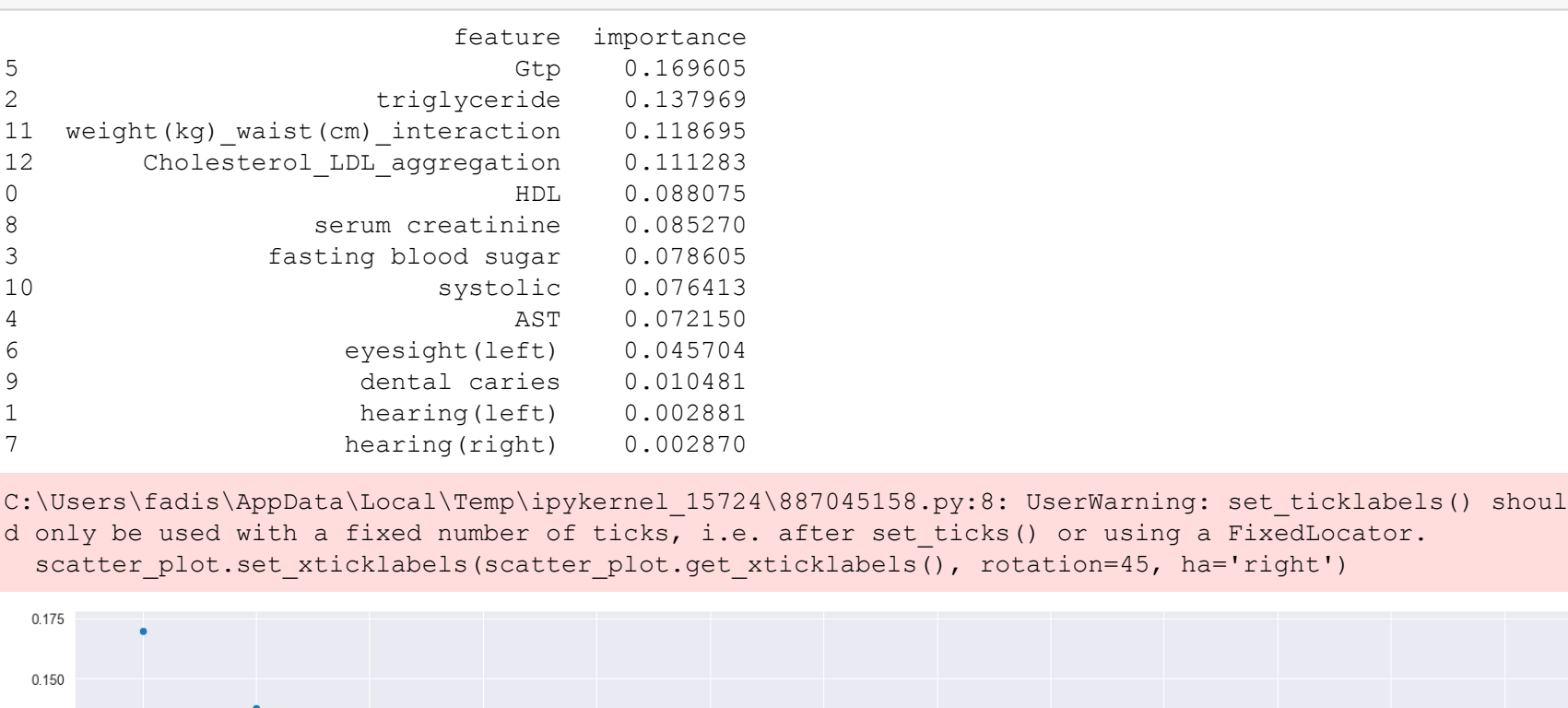
```
In [36]: from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier(random_state=42)
model.fit(all_data[columns].values, all_data['smoking'].values)

feature_importance = model.feature_importances_
feature_importance = pd.DataFrame({'feature': columns, 'importance': feature_importance})
feature_importance = feature_importance.sort_values(by='importance', ascending=False)
```

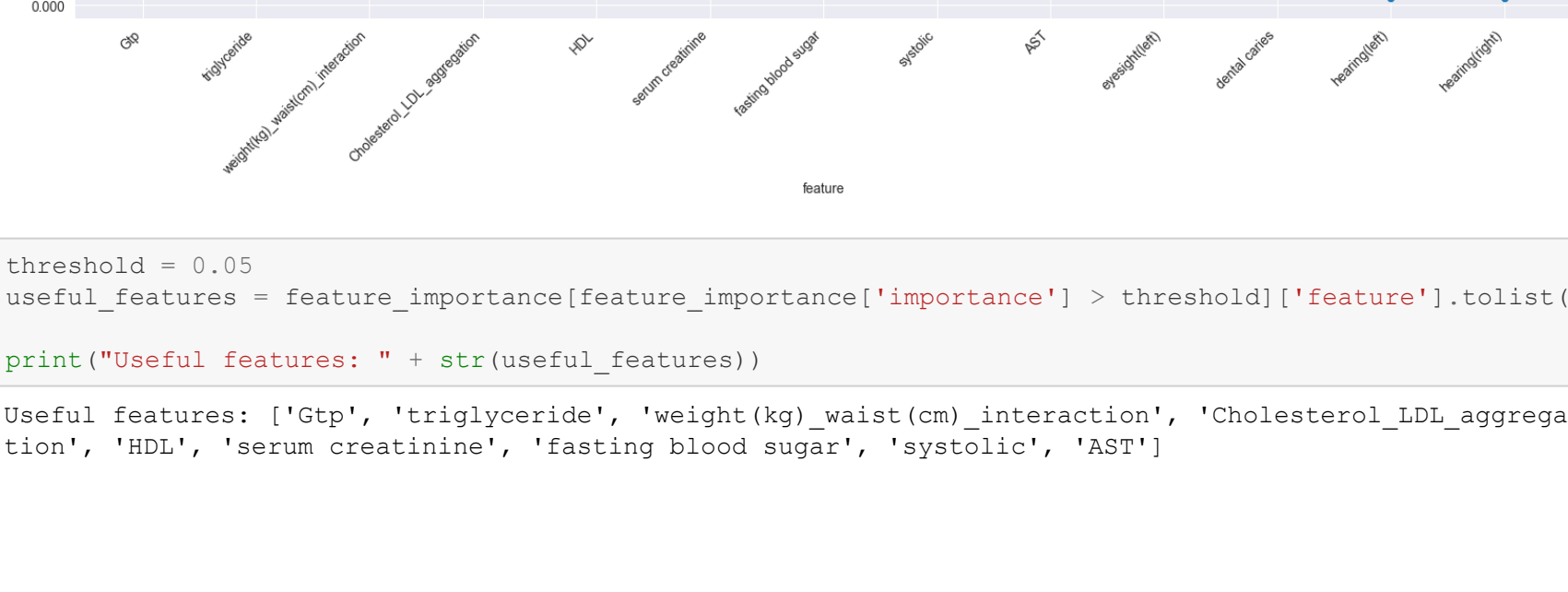
```
In [37]: import seaborn as sns
import matplotlib.pyplot as plt

print(feature_importance)
```



C:\Users\fadis\AppData\Local\Temp\ipykernel_15724\887045158.py:8: UserWarning: set_ticklabels() should only be used with a fixed number of ticks, i.e. after set_ticks() or using a FixedLocator.

```
scatter_plot.set_xticklabels(scatter_plot.get_xticklabels(), rotation=45, ha='right')
```

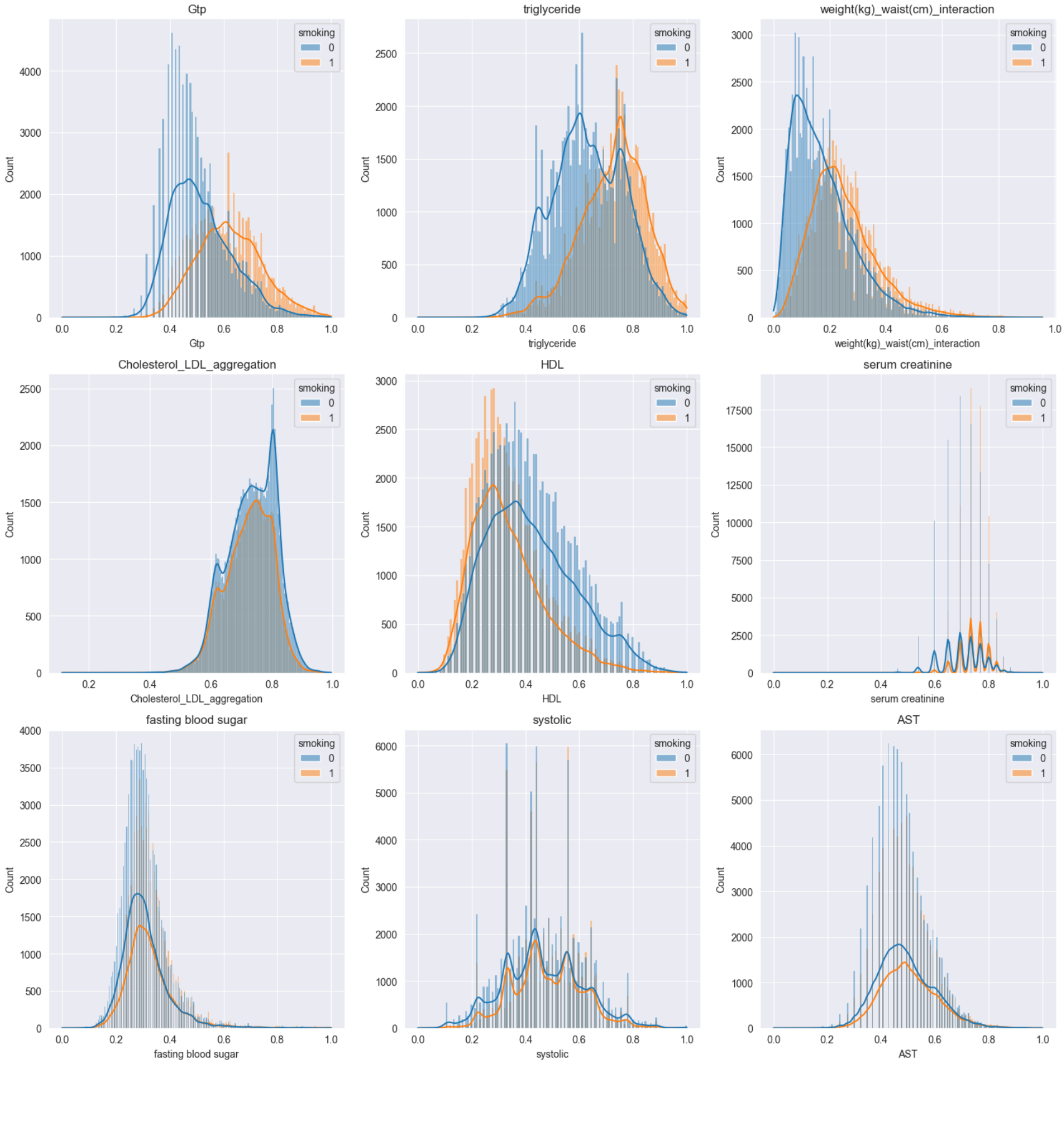


```
In [38]: threshold = 0.05
useful_features = feature_importance[feature_importance['importance'] > threshold]['feature'].tolist()

print("Useful features: " + str(useful_features))
```

Useful features: ['Gtp', 'triglyceride', 'weight(kg)_waist(cm) interaction', 'Cholesterol_LDL_aggregation', 'HDL', 'serum creatinine', 'fasting blood sugar', 'systolic', 'AST']

In [39]: plot_histograms(all_data, useful_features, 3)



```
In [11]: all_data = all_data[useful_features + ['smoking']]  
all_data.to_csv("tahkeer_data_cleaned.csv", index=False)
```