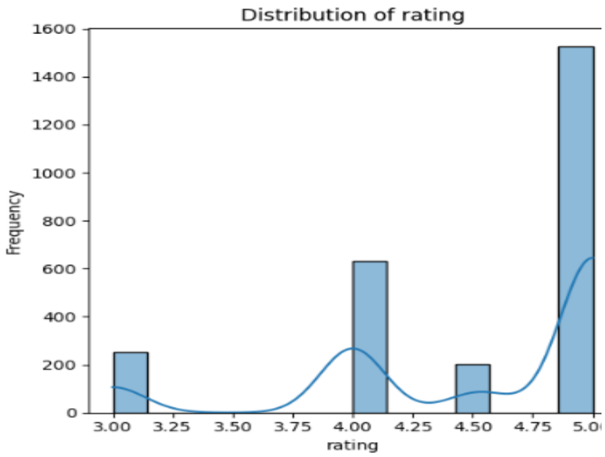
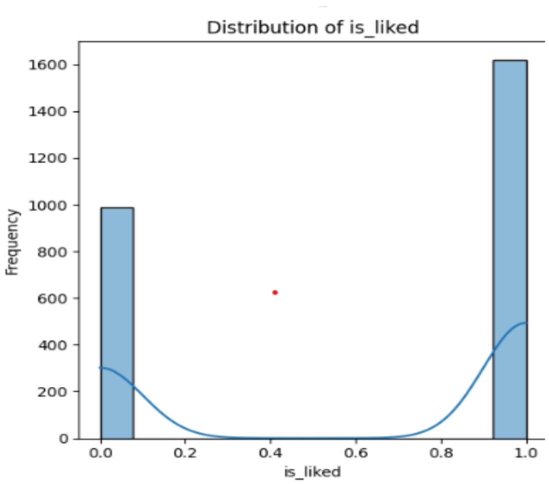
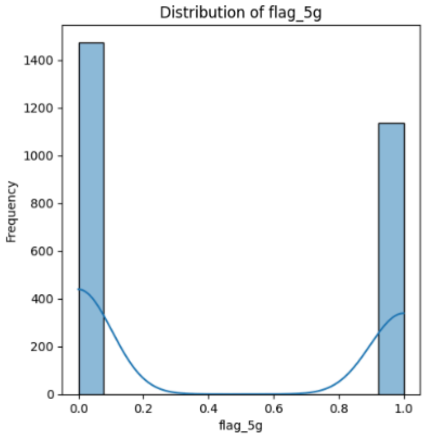
Following are the results of different types of analytics performed on Social Media Data

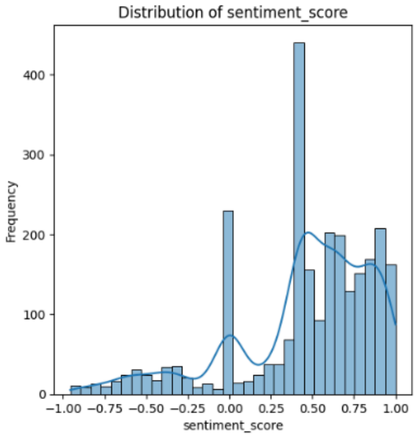
1. Descriptive Analytics

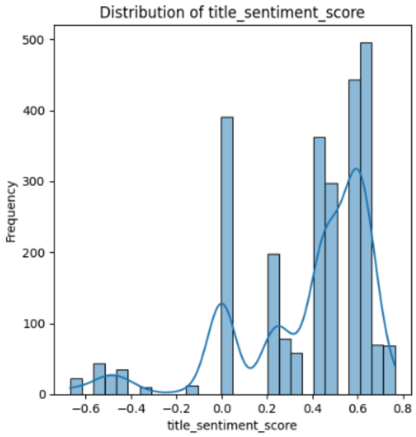




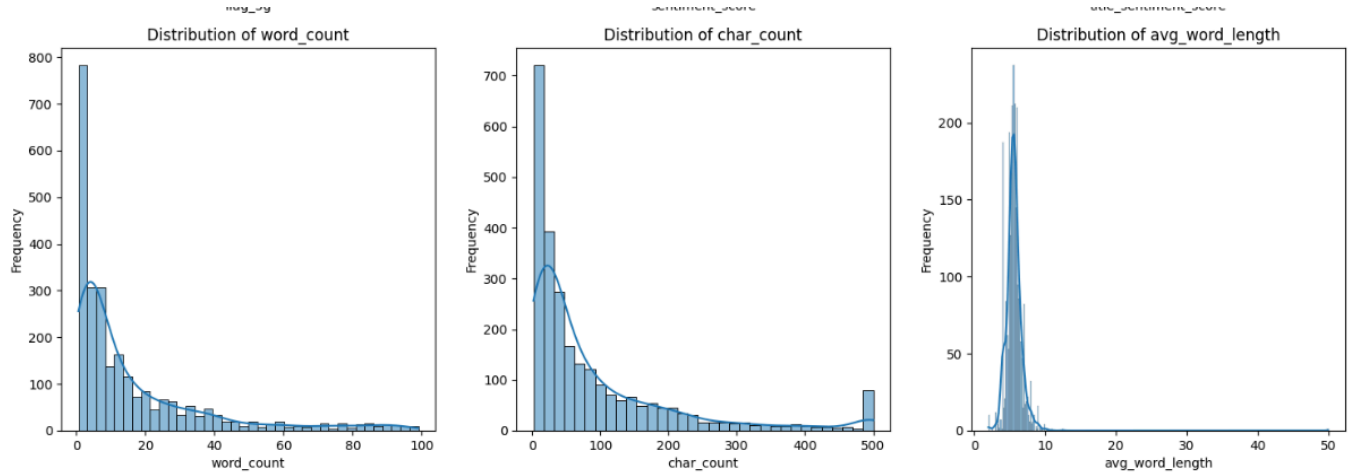
Rating distribution tells us that Motorola has the highest rating as 5 along with highest frequency, with lowest rating as 3, this tells us that customers are overall positive with company and does not have serious issues.



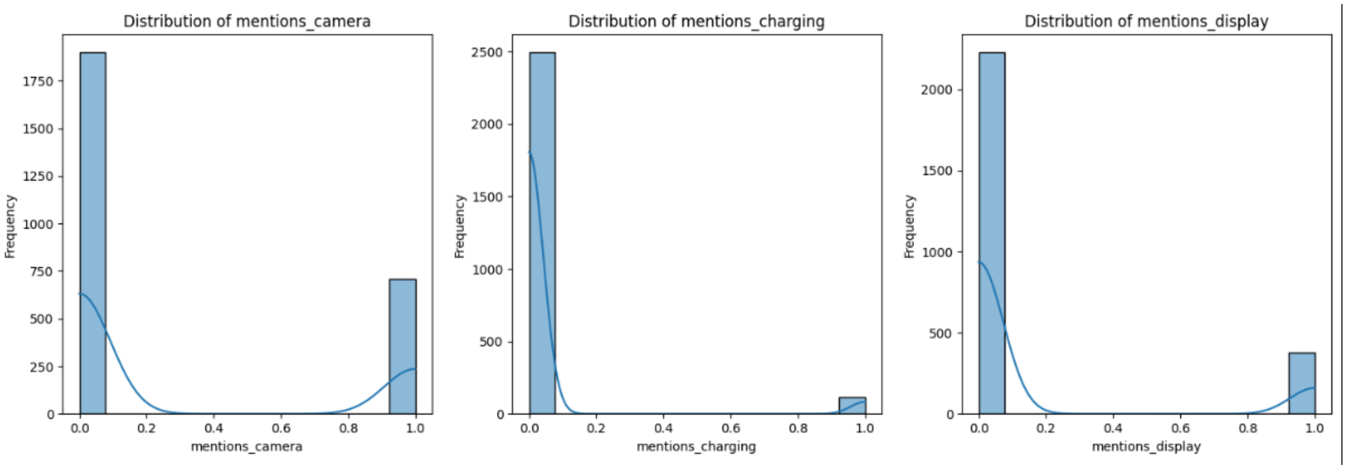




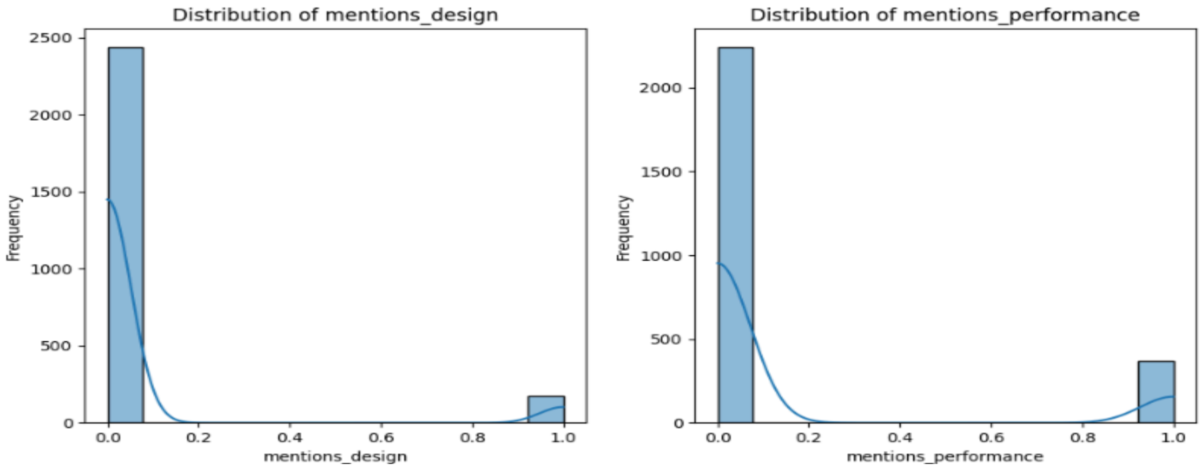
Motorola consistently launches 4g as well as 5g compatible devices.
The distribution of sentiments is skewed towards positive side.



These three distributions(word count, character count, and average word length) tells that normally reviews are short with limited detailed reviews.

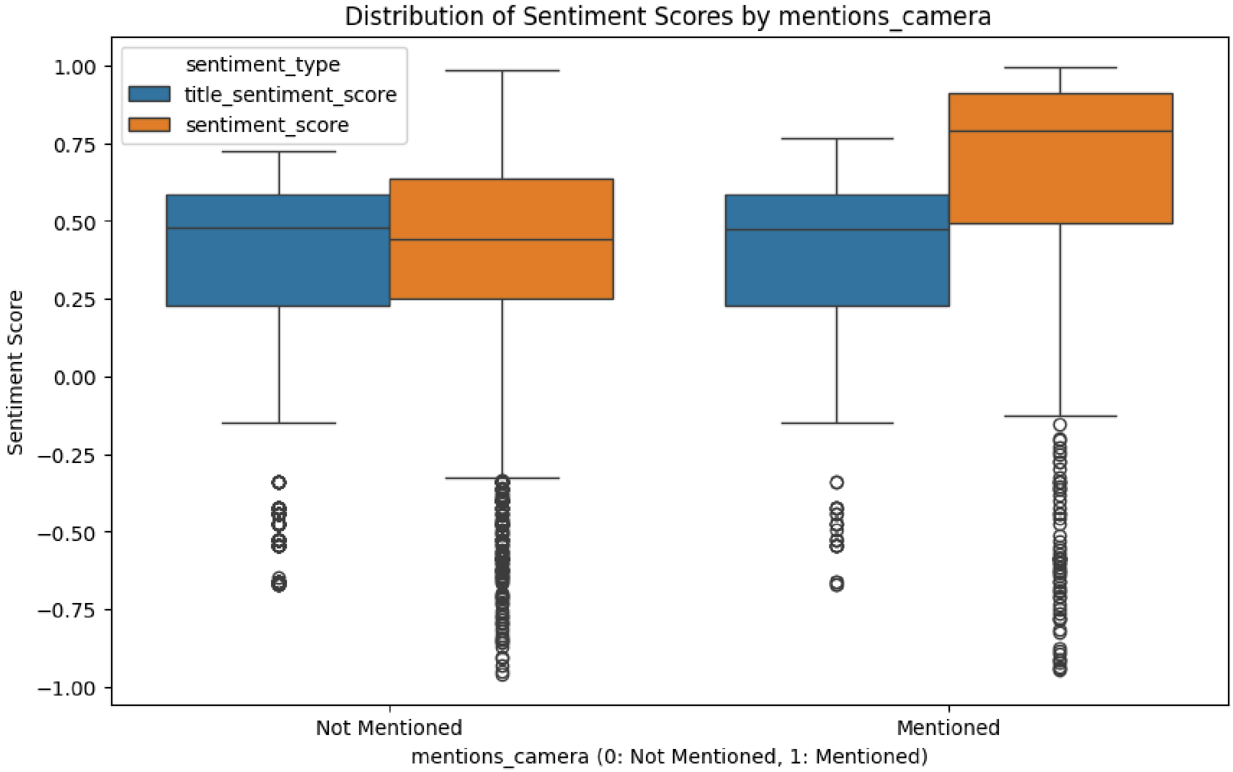


The distribution tells that customer mentioned camera in their reviews more that charging speed and display quality. Is it positive or negative will be understood in diagnostic statistics.

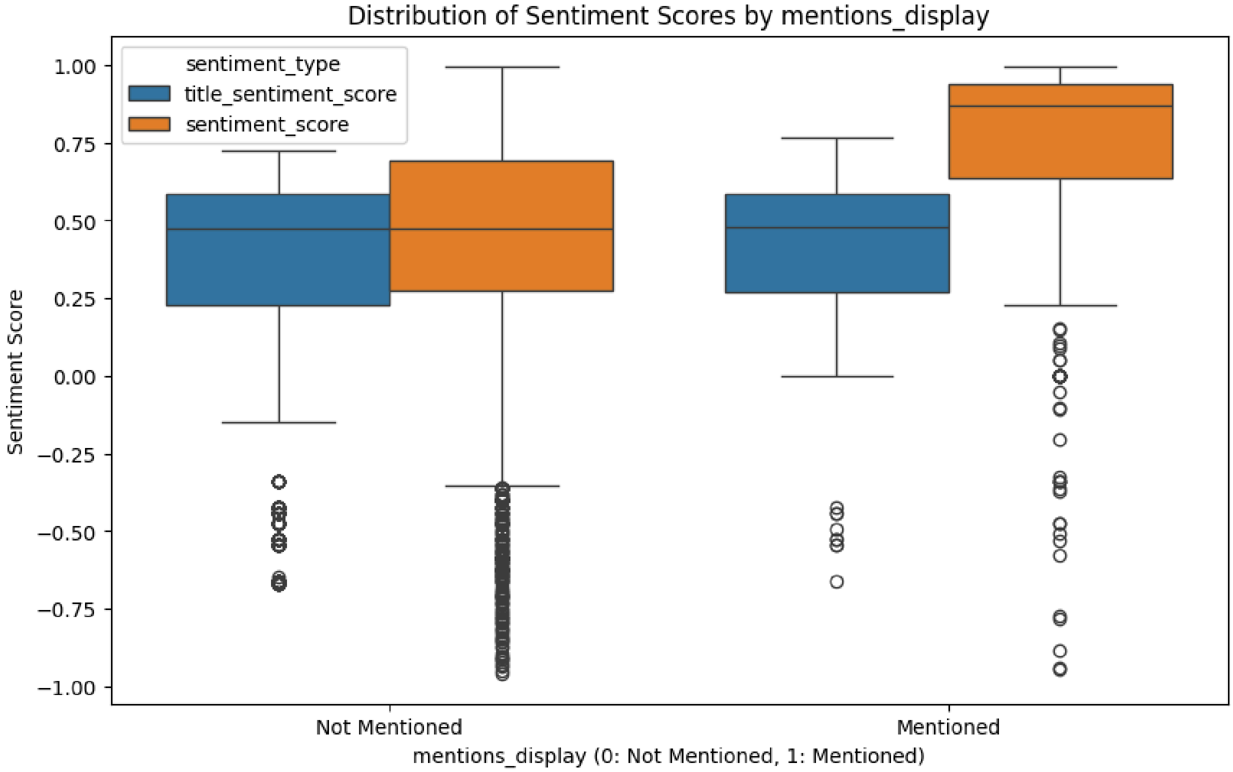


The distribution tells that customer mentioned performance in their reviews more than design of phone itself. Is it positive or negative will be understood in diagnostic statistics.

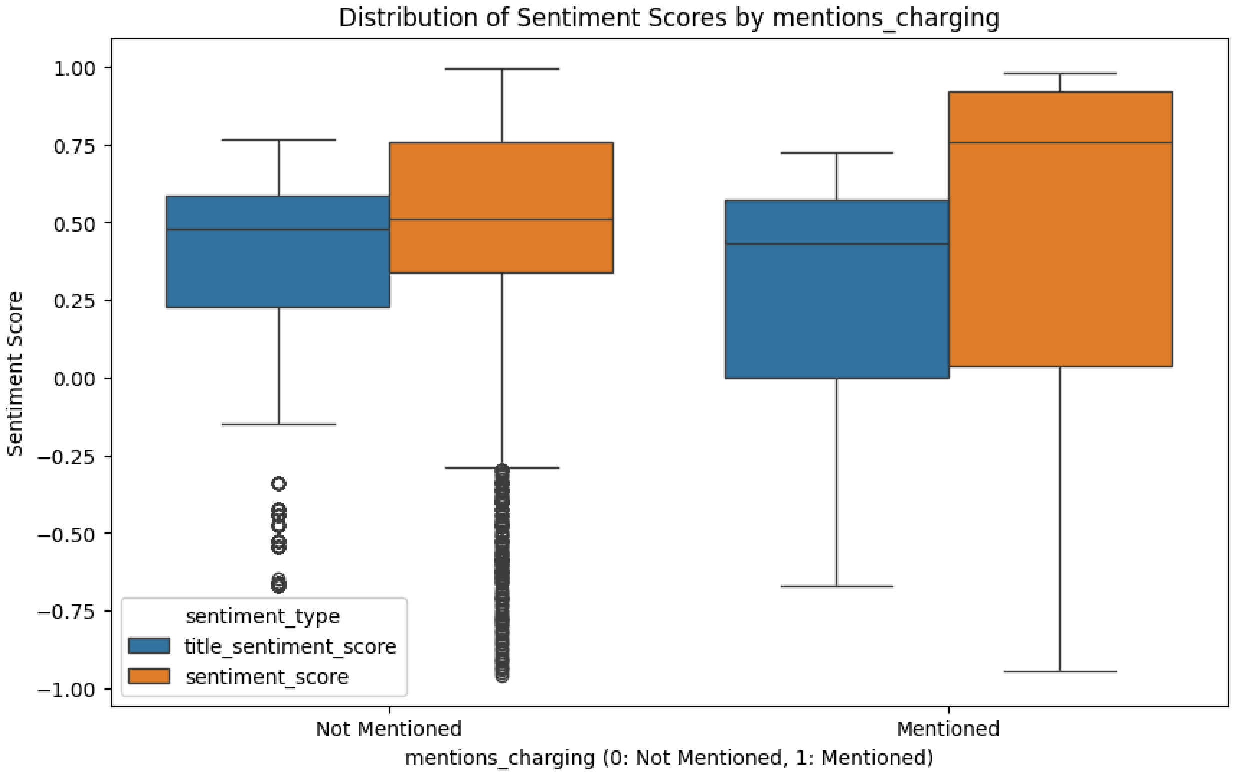
2. Diagnostic Analytics



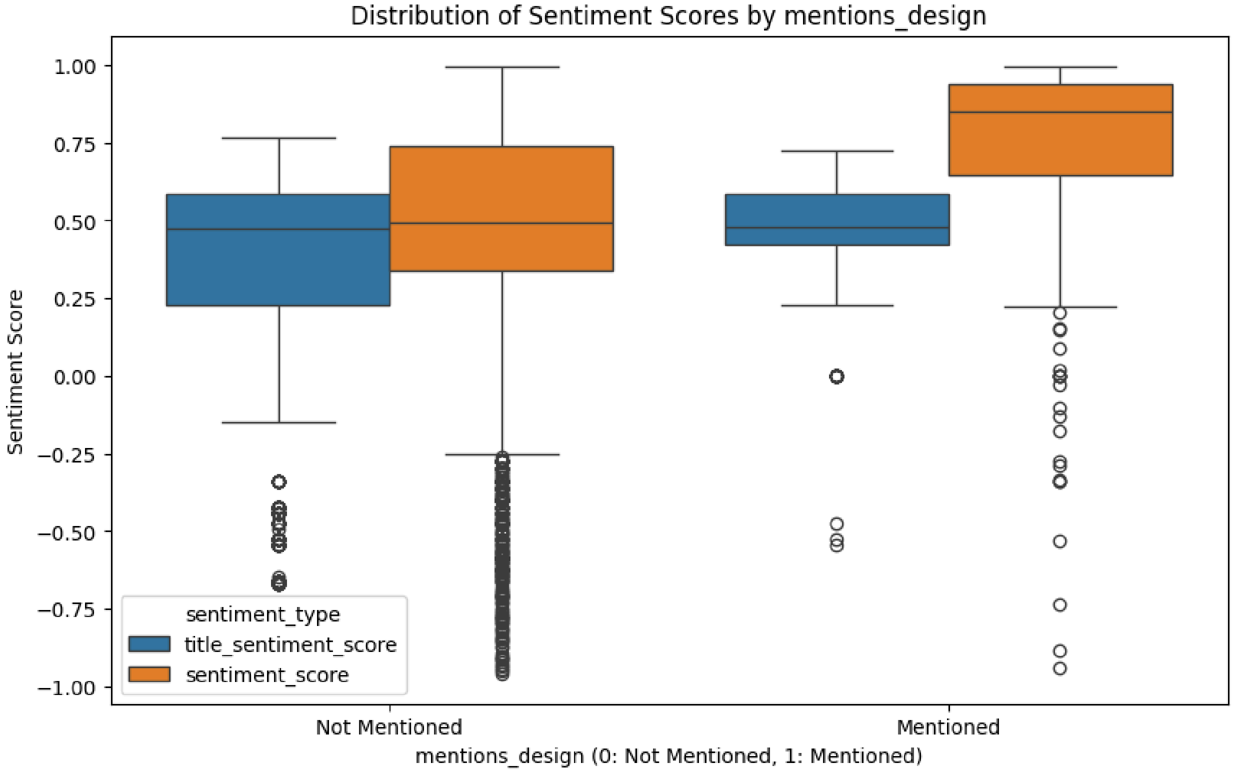
From this plot we could infer that motorola customers are overall positive with camera with some critics not happy with camera as seen in the outliers.



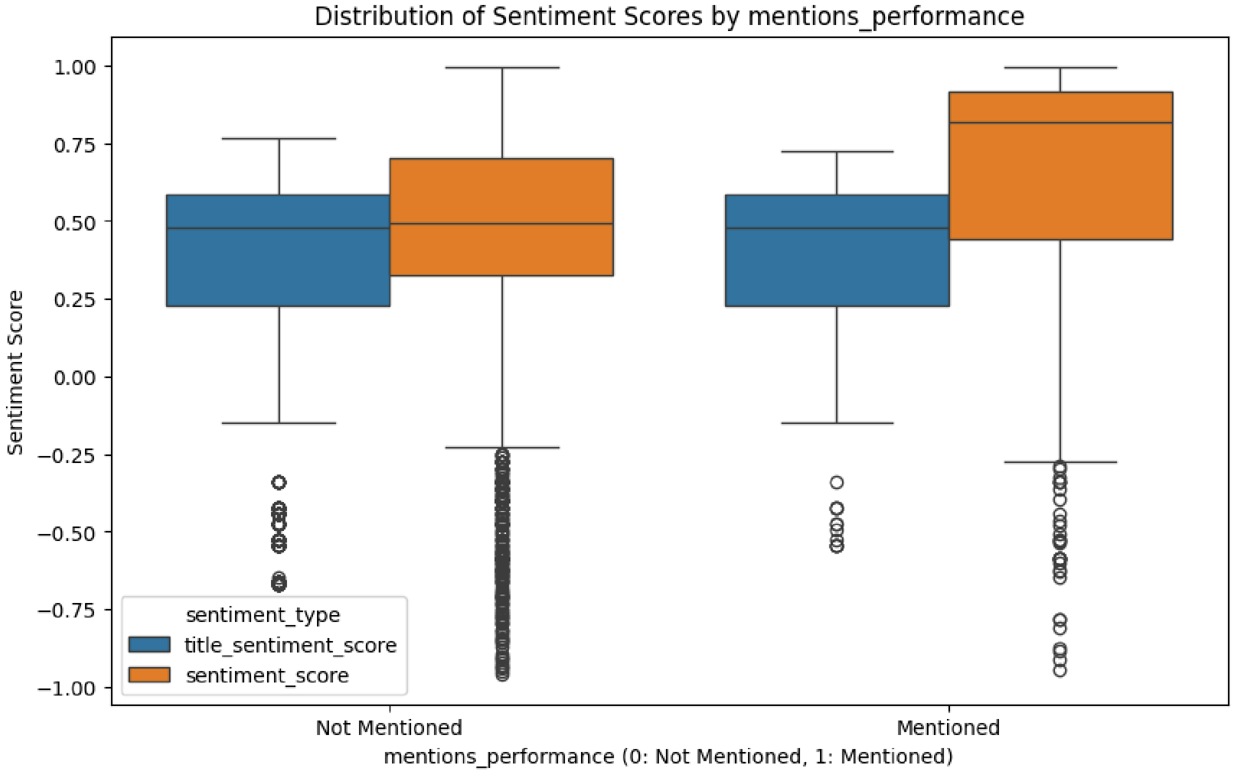
The motorola customer is overall positive with the display provided by motorola.



The motorola customers are overall neutral to positive in terms of charging speed provided by the brand since IQR varies from 0 to 0.75 score.

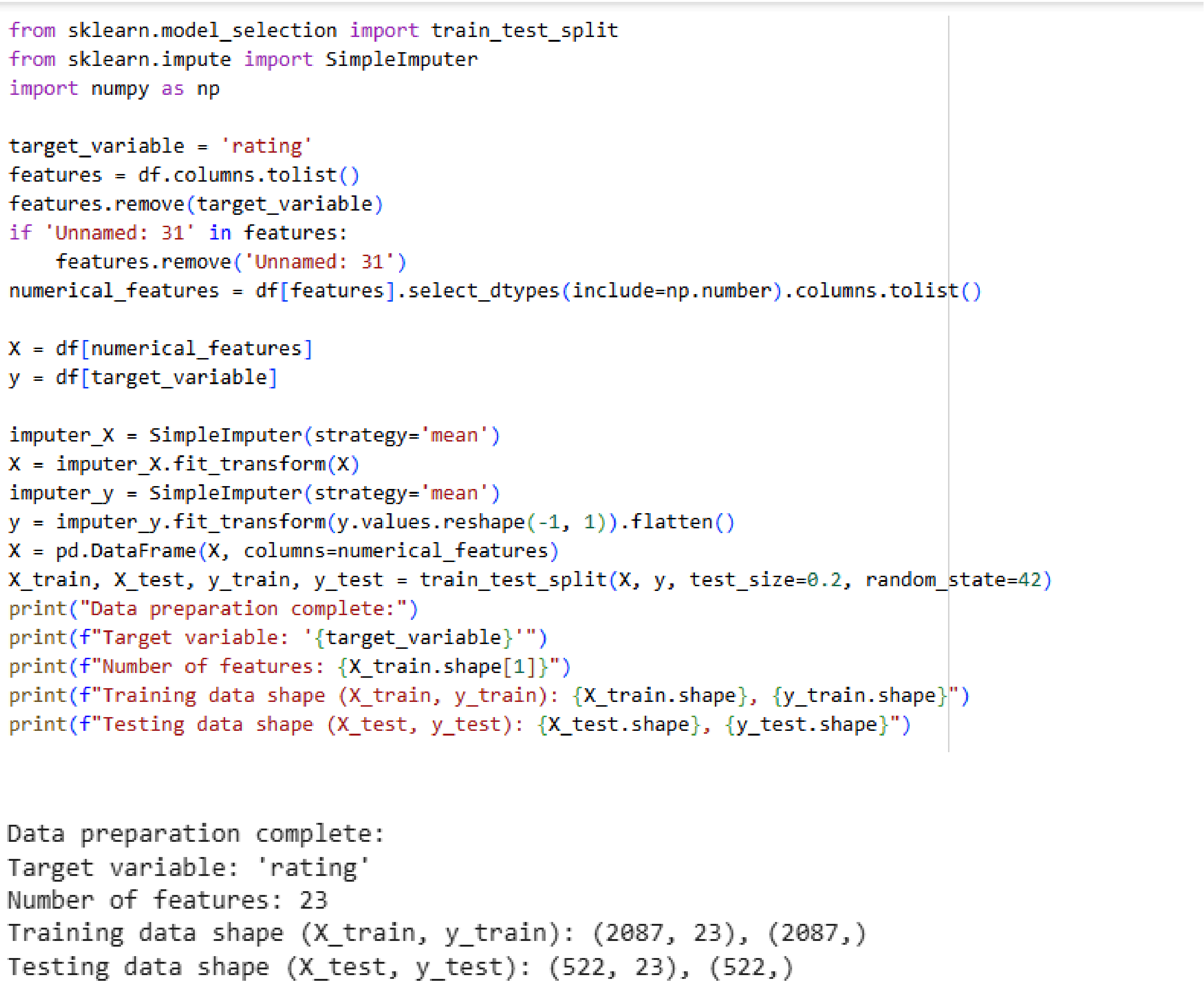


Customers are extremely satisfied with the design of the motorola smartphones. The IQR is above 0.5.



The above chart shows that customer is moderately satisfied with the performance offered by their smartphones.

3. Predictive Analytics



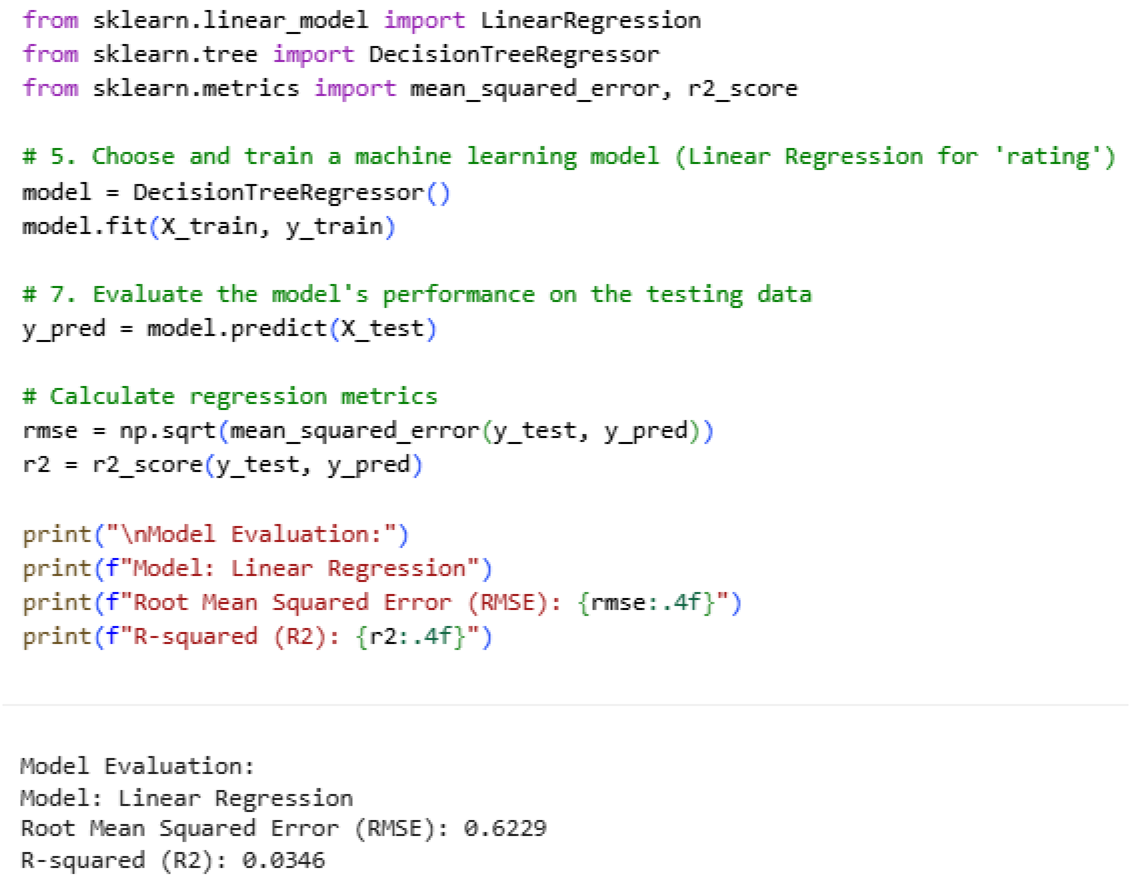
```
from sklearn.model_selection import train_test_split
from sklearn.impute import SimpleImputer
import numpy as np

target_variable = 'rating'
features = df.columns.tolist()
features.remove(target_variable)
if 'Unnamed: 31' in features:
    features.remove('Unnamed: 31')
numerical_features = df[features].select_dtypes(include=np.number).columns.tolist()

X = df[numerical_features]
y = df[target_variable]

imputer_X = SimpleImputer(strategy='mean')
X = imputer_X.fit_transform(X)
imputer_y = SimpleImputer(strategy='mean')
y = imputer_y.fit_transform(y.values.reshape(-1, 1)).flatten()
X = pd.DataFrame(X, columns=numerical_features)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
print("Data preparation complete:")
print(f"Target variable: '{target_variable}'")
print(f"Number of features: {X_train.shape[1]}")
print(f"Training data shape (X_train, y_train): {X_train.shape}, {y_train.shape}")
print(f"Testing data shape (X_test, y_test): {X_test.shape}, {y_test.shape}")
```

```
Data preparation complete:
Target variable: 'rating'
Number of features: 23
Training data shape (X_train, y_train): (2087, 23), (2087,)
Testing data shape (X_test, y_test): (522, 23), (522,)
```



```
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_squared_error, r2_score

# 5. Choose and train a machine learning model (Linear Regression for 'rating')
model = DecisionTreeRegressor()
model.fit(X_train, y_train)

# 7. Evaluate the model's performance on the testing data
y_pred = model.predict(X_test)

# Calculate regression metrics
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)

print("\nModel Evaluation:")
print(f"Model: Linear Regression")
print(f"Root Mean Squared Error (RMSE): {rmse:.4f}")
print(f"R-squared (R2): {r2:.4f}")
```

```
Model Evaluation:
Model: Linear Regression
Root Mean Squared Error (RMSE): 0.6229
R-squared (R2): 0.0346
```

The Decision Tree Regressor model is chosen with test data. Model achieved RSME of 0.6229.