ChatGPT Instructions on LASSO Logistic Regression with Interaction Terms

To perform Lasso logistic regression with scikit-learn, we need to use the LogisticRegression class with the penalty parameter set to 'l1'. The PolynomialFeatures class is used to generate the polynomial and interaction features based on the degree specified. In this case, we set the degree to 3 to include all possible 2-way and 3-way interactions. Here's the code:

```
import pandas as pd
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import r2_score

# Load the training data
train_df = pd.read_csv('training.csv')

# Split the training data into input features and output variable
X_train = train_df.iloc[:, :-1].values
y_train = train_df.iloc[:, -1].values

# Load the test data
test_df = pd.read_csv('test.csv')

# Split the test data into input features and output variable
X_test = test_df.iloc[:, :-1].values
y_test = test_df.iloc[:, -1].values

# Create polynomial features with 2-way and 3-way interactions
poly = PolynomialFeatures(degree=3, interaction_only=True, include_bias=False)
X_poly_train = poly.fit_transform(X_train)
X_poly_test = poly.transform(X_test)

# Scale the input features
scaler = StandardScaler()
X_scaled_train = scaler.fit_transform(X_poly_train)
X_scaled_test = scaler.transform(X_poly_test)

# Create Lasso Logistic Regression model
clf = LogisticRegression(penalty='l1', solver='liblinear', random_state=42)

# Fit the model to the training data
clf.fit(X_scaled_train, y_train)

# Predict the output variable for the test data
y_pred = clf.predict(X_scaled_test)
```

```
# Calculate the pseudo R-squared statistic
r2 = r2_score(y_test, y_pred)

print("Pseudo R-squared: {:.2f}".format(r2))
```

In this code, we first load the training and test data from the CSV files using the read_csv function from Pandas. We split the input features and output variable for both the training and test data. We then create polynomial features with 2-way and 3-way interactions, scale the input features using StandardScaler, and fit a Lasso logistic regression model to the training data.

We predict the output variable for the test data using the fitted model and calculate the pseudo R-squared statistic using the r2_score function from scikit-learn's metrics module. The pseudo R-squared statistic measures the proportion of variance explained by the model, similar to R-squared in linear regression. A value of 1 indicates a perfect fit, while a value of 0 indicates that the model does not explain any of the variance in the data.

The resulting pseudo R-squared statistic is printed to the console.