MACHINE LEARNING FOR COMMUNICATIONS LAB REPORT

MEIC501P

TASK 4

Adaptive modulation Coding prediction using Logistic regression

Name : Durga Sitalakshmi S

Registration Number: 24MEC0024

Phone Number: +91 9731850288

Email ID: durgasitalakshmi.s2024@vitstudent.ac.in

TASK 4 : ADAPTIVE MODULATION CODING – LOGISTIC REGRESSION

1) Importing the required libraries

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.linear_model import LogisticRegression  # Logistic Regression
from sklearn.model_selection import train_test_split
```

2) Training the model with logistic regresion

```
def train_model(X_train, y_train):
    model = LogisticRegression(multi_class='multinomial', solver='lbfgs', max_iter=1000, random_state=42)
    model.fit(X_train, y_train)
    return model

def predict(model, X_test):
    return model.predict(X_test)
```

3) Defining the main function , the encoding data and loading the training and testing files

```
def main():
    # Load the training data
    X_train = pd.read_csv('vehicular_communication_data_train.csv')

    # Encode the categorical 'modulation_and_coding_scheme' column
    label_encoder = LabelEncoder()
    y_train_encoded = label_encoder.fit_transform(X_train['modulation_and_coding_scheme'])

    # Encode the categorical 'traffic_requirements' column using one-hot encoding
    onehot_encoder = OneHotEncoder(sparse_output=False, drop='first')
    traffic_requirements_encoded = onehot_encoder.fit_transform(X_train[['traffic_requirements']])

    # Combine the encoded 'traffic_requirements' with other features
    X_train = np.column_stack((X_train['channel_conditions'], traffic_requirements_encoded))

    # Train the model
    model = train_model(X_train, y_train_encoded)
```

4) checking the accuracy of the model with training dataset 1

```
    # Load the testing data
    X_test = pd.read_csv('vehicular_communication_data_test1.csv')

    # Encode the categorical 'modulation_and_coding_scheme' column of the testing data
    y_test_encoded = label_encoder.transform(X_test['modulation_and_coding_scheme'])

    # Encode the categorical 'traffic_requirements' column of the testing data using one-hot encoding
    traffic_requirements_encoded = onehot_encoder.transform(X_test[['traffic_requirements']])

    # Combine the encoded 'traffic_requirements' with other features of the testing data
    X_test = np.column_stack((X_test['channel_conditions'], traffic_requirements_encoded))

    # Evaluate the model on the test set
    y_pred = predict(model, X_test)
    accuracy = np.mean(y_pred == y_test_encoded)

    print(f"Test accuracy: {accuracy:.4f}")

if __name__ == "__main__":
    main()
```

```
Test accuracy: 0.6667
```

5) Checking the accuracy of the model with training dataset 2

```
    # Load the testing data
    X_test = pd.read_csv('vehicular_communication_data_test2.csv')

    # Encode the categorical 'modulation_and_coding_scheme' column of the testing data
    y_test_encoded = label_encoder.transform(X_test['modulation_and_coding_scheme'])

    # Encode the categorical 'traffic_requirements' column of the testing data using one-hot encoding
    traffic_requirements_encoded = onehot_encoder.transform(X_test[['traffic_requirements']])

    # Combine the encoded 'traffic_requirements' with other features of the testing data
    X_test = np.column_stack((X_test['channel_conditions'], traffic_requirements_encoded))

    # Evaluate the model on the test set
    y_pred = predict(model, X_test)
    accuracy = np.mean(y_pred == y_test_encoded)

    print(f"Test accuracy: {accuracy:.4f}")

if __name__ == "__main__":
    main()
```

```
Test accuracy: 0.3333
```

INFERENCE

- The dataset generated by ChatGPT may not fully reflect the complexity and variability of real-world data.
- Inconsistent correlations between parameters are observed across tests (e.g., weak correlation of 0.33 in test 1, stronger correlation of 0.66 in later tests).
- This variation suggests that the data generation process or underlying assumptions might be influencing the relationships between parameters.
- If this pattern continues, further analysis will be needed to assess whether the dataset accurately mirrors realistic correlations.
- Low covariance and weak correlations between features could lead to reduced accuracy in predictive models.
- Adjustments may be necessary to ensure the dataset is more representative of real-world scenarios.

CONCLUSION

- **Model Performance**: The logistic regression model was successfully trained and evaluated on vehicular communication data. The test accuracy metric provides a quantitative measure of how well the model performs in predicting the modulation_and_coding_scheme using features like channel_conditions and traffic_requirements.
- **Encoding Impact:** The use of LabelEncoder for encoding themodulation_and_coding_scheme and OneHotEncoder for the traffic_requirements ensures that categorical data is converted into a format suitable for the logistic regression model. The inclusion of encoded traffic requirements adds valuable information, which likely contributed to the model's predictive performance.
- **Model Fit:** Since the model uses the multinomial option in logistic regression, it's capable of handling multiple classes in the target variable (modulation_and_coding_scheme). The use of the lbfgs solver with increased iterations allows the model to converge effectively.
- **Insights on Accuracy:** The accuracy reported at the end of the experiment shows how closely the model's predictions match the true modulation and coding schemes in the test dataset. If accuracy is high, this suggests that the model is able to generalize well to

unseen data, capturing the relationships between channel conditions, traffic requirements, and modulation schemes. If accuracy is low, it might indicate that the model needs more feature engineering, tuning, or that the dataset lacks sufficient variance in certain areas.