Cooper Wolf
CMPS 5232 AI
Assignment 2
2/10/26

Assignment 2

Part 1

Sentiment Analysis on Movie Reviews

The first problem is a binary classification task modeled using an end-to-end machine learning workflow to classify movie reviews as positive or negative. The model takes as input sequences of integers, where each integer corresponds to a specific word in a dictionary, representing the movie reviews. Since the model cannot interpret raw text, the original string-based reviews were converted into integer values, padded to a uniform length, and transformed into tensors using multi-hot encoding, while the labels were also vectorized. The neural network consists of two dense layers with 16 nodes each using ReLU activation, followed by a sigmoid output layer, and was trained using binary cross-entropy as the loss function and the Adam optimizer. The model began to overfit after four epochs, so training was limited to four epochs, resulting in a final accuracy of 88%.

Newswire classification

This model performs multiclass classification by categorizing newswire articles into one of 46 different topics. The input consists of news text, and the output is a probability distribution across the 46 possible newswire categories. During data preparation, the text was vectorized using multi-hot encoding. The neural network architecture includes two dense hidden layers with 64 nodes each and ReLU activation functions, followed by an output layer with 46 nodes using softmax activation. The model was trained using categorical cross-entropy as the loss function and the Adam optimizer. Training was conducted for nine epochs, resulting in an accuracy of 80%.

House price prediction

This regression model predicts the median home price in different California cities in 1990. The model takes as input several features, including the home’s location, the median age of houses in a block, block population, number of households, household income, and the total number of rooms and bedrooms. The output is the model’s predicted house price. Data preparation began with feature-wise normalization to address the wide range of input values, and the target variable (house prices) was also scaled by dividing by 100,000. The neural network consists of two dense hidden layers with 64 nodes each using ReLU activation, followed by a single linear output node. After training, the model achieves an average prediction error of approximately $31,000.

Part 2

Compare Binary vs multiclass

Binary classification and multiclass classification differ in that a binary model determines whether an input belongs to one of two possible classes, while a multiclass model classifies an input into one of more than two categories. Because of this, their output layers and loss functions are different. In binary classification, the output layer contains a single node, since only one value is needed to represent a binary decision. In contrast, multiclass classification requires an output layer with a number of nodes equal to the number of possible classes. Binary classification models typically use binary cross-entropy or mean squared error as the loss function, whereas multiclass classification models use categorical cross-entropy.

Class vs Regression

The main difference between classification models and regression models lies in their output possibilities. Classification models predict an outcome from a fixed set of categories, whereas regression models produce continuous values with no predefined constraints. As a result, their output layers differ: classification models use activation functions in the output layer to introduce nonlinearity and produce probabilities or class scores, while regression models typically use no activation function in the output layer, so the output remains linear.

Part 3

Five Models

For the first model, I adapted an existing workflow from Kaggle that closely aligned with the goals of this project and modified it to simulate a user cold-start scenario. The cold-start user was selected as user 1 from the dataset. This scenario was created by withholding all but five of the user's ratings. This model was trained on the remaining data and then used to predict the ratings for the movies that were withheld.

The second model used the same base approach as the first but evaluated a different cold-start user, replacing user 1 with user 2. For the third model, user 2 was again used as the test user, but the hyperparameters of the SVD model were adjusted to analyze the impact of tuning on prediction performance. In the fourth model, the tuned hyperparameters were retrained while the test user was changed to user 6 in order to assess model consistency across different users.

For the final model, a GridSearch approach was employed to train multiple SVD models with varying hyperparameter combinations. This process identified the optimal set of hyperparameters that minimized RMSE and MAE, proving the best overall model performance.

Results

Model	1	2	3	4	5
Score(RMSE):	1.2768	.9553	.9417	1.0063	.9417
Score (MAE):	1.0378	.7595	.7352	.8684	.7333

My discussion/observations

One reason Model 2 outperformed Model 1 is the difference in the amount and quality of available user data. User 1, used in Model 1, had rated only 20 movies, whereas User 2, used in Model 2, had rated 76. Additionally, the five retained ratings for User 1 were all between 2.5 and 3.5, providing limited insight into the user's preferences. In contrast, User 2's retained ratings ranged from 3 to 4, offering slightly clearer information about the types of movies the user tended to enjoy.

Model 4 was initially expected to outperform the previous models because it used a user with more polarizing rating behavior. However, the results showed that this was not consistently the case, suggesting that rating extremity alone does not guarantee improved cold-start performance.

For the final model, the use of GridSearch for hyperparameter tuning was expected to produce the strongest performance. While this approach did result in lower RMSE and MAE compared to the other models, the improvement was smaller than anticipated, indicating that hyperparameter optimization provides incremental gains rather than dramatic performance improvements in this cold-start setting.

Source

https://www.kaggle.com/code/ibtesama/getting-started-with-a-movie-recommendation-system/notebook