Discussion

MoneyLion Machine Learning Scientist Take Home Assessment

Predict Bank Transaction Category Challenge by Johnston Yap

johnstonyap.jy@gmail.com | +6011-1113 3792

- Time Spent : 10 hours

1. Data Analysis and Model Development

The datasets used in this task include `bank_transactions` and `user_profiles`. The primary dataset, `bank_transactions`, contains various attributes such as client ID, bank ID, account ID, transaction ID, transaction date, description, amount, and category. The target variable for this task is the category of the transaction, which we aim to predict based on the description.

The initial data analysis involves checking for NaN values (only `categories` have NaNs with 257 rows), understanding the distribution of transaction categories, and identifying potential issues such as typos or irrelevant categories (e.g., 'Transfer', 'Uncategorized'). Entries like 'Transfer' cannot be defined as Deposit, Credit, or Debit, and 'Uncategorized' needs to be excluded due to its lack of guidance on training and validating datasets. The data was cleaned by removing such entries and merging similar categories (e.g., 'Bank Fee' and 'Bank Fees'), addressing typos in the labels.

Reasoning and Justification for Model Architecture

Given the text-based nature of the 'description' column, which is used to predict the transaction category, the following preprocessing steps and model choices were made:

a. Text Preprocessing

- Removing extra spaces, punctuation, and numeric values.
- Tokenizing the text.
- Lemmatization and removing stop words were considered but commented out to maintain features, which increased the F1-Score.

b. Feature Extraction on Techniques for Different Models:

TF-IDF (Term Frequency-Inverse Document Frequency): (Implemented)

- Importance Weighting: Adjusts term weights based on uniqueness to the document.
- Reduces Impact of Common Words: Lowers weights for terms that appear in many documents.
- Improved Classification: Enhances model performance by emphasizing discriminative terms.

BoW (Bag of Words):

- Equal Importance: Treats all words with equal weight, regardless of their commonality.

- High Dimensionality: Often results in large, less informative feature vectors.
- No Context Sensitivity: Ignores term importance across the corpus.

Model Impact:

- Naive Bayes (NB): TF-IDF generally yields better performance due to better term weighting.
- Logistic Regression (LR): Benefits from TF-IDF's informative feature set.
- Random Forest (RF): TF-IDF can reduce dimensionality and improve feature quality.
- Support Vector Machine (SVM): TF-IDF improves feature scaling and separation between classes.

Model	**Bag of Words**	**TF-IDF**
Logistic Regression	**Useful for term frequencies**	**Important for capturing term importance**
Naive Bayes	**Effective with term frequencies**	**Highly suitable for capturing term relevance**
Random Forests	**Less relevant for interactions**	**More suitable for term importance**
Support Vector Machines	**Useful for simple term vectors**	**Crucial for capturing term relevance**

c. Model Selection

- **Naive Bayes**: Suitable for text classification tasks due to its simplicity and efficiency.
- **Logistic Regression**: Effective for binary and multiclass classification tasks.
- **Random Forest**: A robust ensemble method that handles overfitting well and provides feature importance.
- **Support Vector Machine (SVM)**: A powerful classifier for high-dimensional spaces, though computationally intensive.

d. Hyperparameter Tuning with K-Fold Cross-Validation

To rigorously evaluate the performance of the models, 5-fold cross-validation was implemented. This method divides the dataset into five distinct folds, using each fold once as a validation set while the remaining four folds are used as training data. This process is repeated five times, ensuring every data point is used for both training and validation.

Key Benefits of 5-Fold Cross-Validation:

- **Reduced Overfitting**: Assesses the model's ability to generalize to unseen data by validating on multiple subsets.
- **Performance Metrics**: Provides a more reliable estimate of model performance by averaging metrics (accuracy, F1-score, etc.) across all folds.
- **Efficient Use of Data**: Ensures all data points are used for both training and validation, leading to a more comprehensive evaluation.

2. Functional Code and Reasoning

a. Loading Data:

- Read `bank_transaction.csv` data from CSV files.
- Display initial information for understanding the structure.

b. Data Visualization:

- Plot the distribution of transaction categories.
- Display description field from each category to examine differences.

c. Data Preprocessing:

- Initial data analysis involved checking for NaN values, understanding the distribution of transaction categories, and identifying potential issues such as typos or irrelevant categories (e.g., 'Transfer', 'Uncategorized'). The data was cleaned by removing such entries and merging similar categories (e.g., 'Bank Fee' and 'Bank Fees').
- Preprocessing descriptions to clean and tokenize text.

d. Feature Extraction:

Using `TF-IDF` to transform text data into numerical features.

- TF-IDF transforms text into weighted numerical features that enhance model performance by emphasizing term relevance, which benefits LR, NB, SVM, and RF in capturing important patterns in text data.

e. Data Splitting:

- Splitting the dataset into training and testing sets.

f. Model Training and Evaluation:

- Training and evaluating models (Naive Bayes, Logistic Regression, Random Forest, SVM).
- Implementing 5-Fold Cross-Validation ensures comprehensive evaluation.
- Classification Report and Confusion Matrix were used to determine Accuracy and F1-Score.

3. Model Performance and Effectiveness

Model Evaluation Results (With 5-Fold Cross Validation):

Model	Training Time (seconds)	Accuracy (%)	F1 Score
Naive Bayes	0.376576	79.89	0.78
Logistic Regression	35.14	88.62	0.88
Random Forest	201.46	89.93	0.90
SVM	365.07	89.51	0.89

Conclusion:

- The **Random Forest** model achieved the highest F1 score of 0.90, indicating it is the best-performing model among those evaluated.
- The **Logistic Regression** model achieved a high F1 score of 0.88 and a fast training time of 35.14 seconds, indicating it is the most efficient model among those evaluated.

4. Future Development Plans

Next 1 Month:

- **Hyperparameter Tuning**: Use grid search or random search to fine-tune model hyperparameters, alongside existing K-Fold Cross-Validation.
- **Feature Engineering**: Explore additional features like transaction amount, user profile information, and transaction date to enrich the model's input data.

Next 3 Months:

- **Feature Improvement**: Leverage data from `user_profile.csv` to enhance classification and potentially increase conversion rates.
- **Model Ensemble**: Combine predictions from different models to boost accuracy, possibly through a voting system.
- **Deep Learning Models**: Experiment with advanced models with word embeddings (e.g., Word2Vec or GloVe). to capture subtle information and improve F1 scores.
- **Data Feeding**: Implement a system for continuous updates with new data.
- **Deployment**: Develop a scalable API for real-time predictions and integrate it into a production environment.

Conclusion

This project focused on developing a machine learning model to predict bank transaction categories based on descriptions. Using `TF-IDF` proved to be an efficient choice for feature extraction across models. The `Logistic Regression` model demonstrated a good balance between accuracy and training time, while the `Random Forest` model performed best in terms of F1 score. Future work will involve improving model performance and preparing for practical deployment.