Garvit Audichya

23BAI10142

Case Study

Ans1. To handle Imbalance in dataset like churned customers are fewer than active ones we can apply following:

1. **Choose Proper Evaluation Metric**:

   The first technique to handle imbalanced data is choosing a proper evaluation metric. The accuracy of a classifier is the total number of correct predictions divided by the total number of predictions. For an imbalanced class dataset, the **F1 score** is a more appropriate metric. It is the harmonic mean of precision and recall and the expression is –

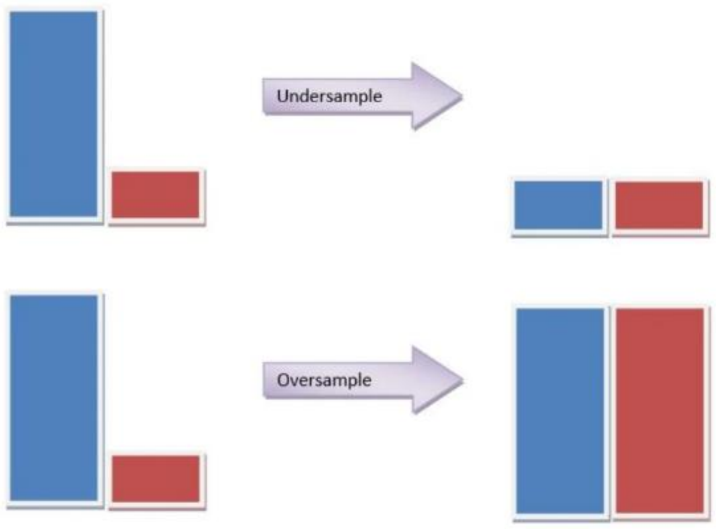
$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

   F1 score keeps the balance between precision and recall and improves the score only if the classifier identifies more of a certain class correctly.

2. **SMOTE:**
   The third technique to handle imbalanced data is the Synthetic Minority Oversampling Technique or SMOTE, which is another technique to oversample the minority class. Simply adding duplicate records of minority class often don't add any new information to the model. In SMOTE new instances are synthesized from the existing data. If we explain it in simple words, SMOTE looks into minority class instances and use k nearest neighbor to select a random nearest neighbor, and a synthetic instance is created randomly in feature space.

3. **Resampling (Under sampling or Oversampling) :**
   The second technique used to handle the imbalanced data is used to up sample or down sample the minority or majority class. When we are using an imbalanced dataset, we can oversample the minority class using replacement. This technique used to handle imbalanced data is called oversampling. Similarly, we can randomly delete rows from the majority class to match them with the minority class which is called under sampling. After sampling the data, we can get a balanced dataset for

both majority and minority classes. So, when both classes have a similar number of records present in the dataset, we can assume that the classifier will give equal importance to both classes.



Ans2. Most Important features of a Churn:

- **Tenure**: Lower tenure may indicate high rate of churn.
- **MonthlyUsageHours**: As usage decreases, interest dies and then one may churn.
- **SubscriptionPlan and MonthlyFee**: Customers might churn if they think the plan is not worth their money.
- **PaymentMethod**: Payment method can determine if a customer churn or not by analysing how many customers use credit cards or net banking and what is the ratio of churning among them.
- **Age**: Age can determine if one churn or not. We can analyse the ratio among Young Adults, Adults and Senior Citizens according to age and see how many churn or not.

Ans3. Explaining my Model:

- **Simplify the Explanation**: Using visual aids like bar charts or heatmaps to show feature importance and correlation among features.
- **Use Explainability Tools**:

  SHAP: SHAP (**SHapley Additive exPlanations**) is a game theoretic approach to explain the output of any machine learning model. It connects optimal credit allocation with local explanations using the classic Shapley values from game theory and their related extensions.

- **Illustrate with Examples:**
    - Provide real or hypothetical customer profiles to show how the model makes predictions:
    - "Customer A has a short tenure, uses the service for 2 hours a month, and has a basic plan. The model predicts they are likely to churn with a 75% probability."
    - "Customer B has been a loyal user for 24 months, uses the service for 15 hours a month, and has a premium plan. Their churn likelihood is only 10%.

Ans.4 Deploying the Model:

1) **Model Optimization**:

    - Ensure the model is well-trained, validated, and tested on unseen data to avoid overfitting.
    - Optimize the model's architecture and hyperparameters for production efficiency.

2) **Save the Trained Model**:

    - Export the trained model using a framework like TensorFlow or PyTorch (e.g., as a .h5 or .pth file).

3) **Develop an API for Predictions**:

    - Wrap the model in a REST API using tools like **Flask**, **FastAPI**, or **Django** to make it accessible.
    - Example: A customer ID is sent to the API, and the model returns the churn probability.

4) **Business Dashboard**:

    - Build a dashboard for stakeholders to view customer churn predictions and insights.
    - Provide summaries, like "Top 10% of customers at risk of churn," and actionable recommendations.