Customer churn prediction using the telco customer churn

Problem Description :

The Telco Customer Churn dataset consists of data from a telecommunications company. The objective of this project is to predict customer churn, which is when a customer cancels their service or stops doing business with a company. The dataset contains information about customer demographics, account information, services used, and whether or not the customer churned. The goal of this project is to build a model that can accurately predict which customers are likely to churn, so that the company can take steps to prevent it.

Dataset Link: The dataset can be found on Kaggle – https://www.kaggle.com/blastchar/telco-customer-churn

Requirements:

- Python 3.x
- Jupyter Notebook or any other python environment
- Libraries: pandas, numpy, seaborn, matplotlib, sklearn

Deliverables:

- Exploratory Data Analysis (EDA) report
- Data Preprocessing report
- Feature Selection and Engineering report

- Model Building and Evaluation report

- Final report with conclusions and recommendations for the company.

Background: Customer churn is a common problem faced by businesses across many industries. In the telecommunications industry, customer churn can have a significant impact on revenue and profitability. Therefore, it is important for telecommunications companies to identify customers who are likely to churn and take steps to retain them. This project will use machine learning techniques to build a predictive model that can accurately identify customers who are likely to churn.

Framework

1.Problem definition

- Define the problem of customer churn and its importance in the telecommunications industry.
- Define the objective of the project, which is to predict which customers are most likely to churn, and identify the key factors that contribute to churn.

2. Data collection and analysis

- Collect the Telco Customer Churn dataset from Kaggle or other relevant sources.
- Perform exploratory data analysis to understand the data and its characteristics.
- Clean the data, handle missing values, and remove any irrelevant features.

3. Feature engineering and selection

- Engineer new features from the existing data that could potentially improve the performance of the model.
- Select the most relevant features that have the highest impact on customer churn.

4. Model selection and training

- Select appropriate machine learning algorithms for the classification task, such as logistic regression, decision trees, random forests, or gradient boosting.

- Split the data into training and testing sets, and evaluate the performance of each algorithm using appropriate metrics.

- Fine-tune the hyperparameters of the chosen algorithm using techniques such as grid search or randomized search.

5. Model evaluation and interpretation

- Evaluate the performance of the model on the testing set using appropriate metrics such as accuracy, precision, recall, and F1-score.

- Interpret the results and identify the most important factors that contribute to customer churn.

- Provide recommendations to the telecommunications company on how to reduce customer churn and improve customer retention.

6. Deployment and monitoring

- Deploy the model into production and integrate it with the existing customer database.

- Monitor the performance of the model over time and update it as needed to ensure its accuracy and effectiveness.

Code Explanation :

Here is the simple explanation for the code which is provided in the code.py file.

Introduction: In this project, we aim to predict customer churn using the Telco Customer Churn dataset. The dataset contains customer information such as demographics, usage behavior, and contract information. We will train a machine learning model to predict whether a customer is likely to churn or not.

Requirements: To run this code, we will need to have Python 3.x and the following libraries installed: pandas, numpy, matplotlib, seaborn, and scikit-learn.

Dataset: The Telco Customer Churn dataset can be downloaded from Kaggle or can be accessed through this link: https://www.kaggle.com/blastchar/telco-customer-churn

Sections:

1. Data preprocessing

2. Exploratory Data Analysis

3. Feature Engineering

4. Model Building and Evaluation

How to run the code:

1. First, download the dataset from the link mentioned above and save it in the same directory as the Python script.

2. Open the Python script in your preferred code editor.

3. Import the required libraries and load the dataset using pandas.

4. Preprocess the data by checking for missing values and converting categorical variables to numerical ones.

5. Perform exploratory data analysis by visualizing the data using matplotlib and seaborn.

6. Engineer new features that could improve the performance of the machine learning model.

7. Split the dataset into training and testing sets.

8. Build a machine learning model using scikit-learn and train it on the training set.

9. Evaluate the model on the testing set using accuracy, precision, recall, and F1 score.

10. Repeat steps 7-9 with different models and hyperparameters to find the best model for predicting customer churn.

Section 1: Data Preprocessing

- In this section, we are importing the necessary libraries for data preprocessing.

- The dataset is then loaded from a CSV file and missing values are handled by dropping the columns with missing values or filling them with appropriate values.

- Categorical variables are converted into numerical values using one-hot encoding.

- Finally, the dataset is split into training and testing sets using train_test_split from the sklearn library.

Section 2: Model Building

- In this section, we are importing the necessary libraries for building the model.
- The Sequential model is created using the Keras library and layers are added to the model.
- The model is then compiled using appropriate loss function, optimizer, and evaluation metric.

Section 3: Model Training

- In this section, we are training the model using the training set.
- The fit method is called on the model with the training data and the number of epochs.
- The model is trained on the training set with the specified number of epochs and the loss and accuracy metrics are calculated.

Section 4: Model Evaluation

- In this section, we are evaluating the performance of the model on the testing set.
- The predict method is called on the model with the testing data to obtain the predicted labels.
- The performance metrics, such as accuracy, precision, recall, and F1-score, are calculated using the predicted and actual labels.

To run this code, you would need to have the necessary libraries such as pandas, numpy, sklearn, keras, and tensorflow installed in your environment. The dataset can be obtained from Kaggle or any other relevant source. Once you have the dataset, you can run the code section by section to preprocess the data, build the model, train the model, and evaluate the performance.

Future Work :

Future Work for Customer Churn Prediction using Telco Customer Churn dataset

Now that we have built a basic model to predict customer churn, we can improve the performance and accuracy of the model by implementing various techniques. Below are some steps that can be taken to improve the model:

1. **Feature Engineering and Selection:** One of the most important steps in building a good machine learning model is to select relevant features. We can do some feature engineering by creating new features that are relevant to the problem. We can also use various techniques such as correlation analysis and feature importance ranking to select the most important features. This can help us to improve the accuracy of the model.

2. **Hyperparameter Tuning**: In the current model, we have used default hyperparameters for our models. However, we can further optimize the model by tuning the hyperparameters. We can use various techniques such as GridSearchCV or RandomizedSearchCV to find the best hyperparameters for our model.

3. **Ensemble Techniques:** Ensemble techniques can be used to improve the performance of the model. We can use techniques such as bagging, boosting, and stacking to improve the accuracy of the model.

4. **Other Machine Learning Models:** We can also try other machine learning models such as Random Forest, Gradient Boosting, and Neural Networks to see if they perform better than the current model.

5. **Imbalanced Data Handling:** The current dataset is imbalanced, which means that there are more examples of one class than the other. We can use techniques such as oversampling, undersampling, or SMOTE to handle imbalanced data.

Step-by-Step Guide to implement the Future Work:

1. **Feature Engineering and Selection:** To implement feature engineering, we can create new features such as the total amount of money spent by the customer, the total number of services

subscribed, or the length of time the customer has been with the company. We can use techniques such as correlation analysis to find the most important features. We can also use Lasso or Ridge regression to find the most relevant features.

2. **<u>Hyperparameter Tuning:</u>** We can use the GridSearchCV or RandomizedSearchCV methods to find the best hyperparameters for our models. These methods help us to search for the best hyperparameters by testing a range of values for each hyperparameter. This can help us to find the best combination of hyperparameters that gives us the highest accuracy.

3. **<u>Ensemble Techniques:</u>** We can use ensemble techniques such as bagging, boosting, and stacking to improve the accuracy of the model. Bagging involves building multiple models on different subsets of the data and then averaging the predictions. Boosting involves building multiple models sequentially, where each subsequent model tries to correct the errors of the previous model. Stacking involves combining multiple models, where the predictions of the models are used as input features for a final model.

4. **<u>Other Machine Learning Models:</u>** We can try other machine learning models such as Random Forest, Gradient Boosting, and Neural Networks to see if they perform better than the current model. These models can be implemented using the same steps as we did for the logistic regression model.

5. **<u>Imbalanced Data Handling:</u>** To handle imbalanced data, we can use techniques such as oversampling, undersampling, or SMOTE. Oversampling involves creating new examples of the minority class by replicating existing examples. Undersampling involves removing examples of the majority class to balance the dataset. SMOTE involves creating new examples of the minority class by synthesizing new examples based on the existing examples.

Overall, implementing these future work steps can help us to improve the accuracy and performance of our customer churn prediction model.

Exercise :

Try to answers the following questions by yourself to check your understanding for this project. If stuck, detailed answers for the questions are also provided.

1. **How can we improve the accuracy of the model?**
   Answer: We can try out different machine learning algorithms like Random Forest, SVM, or Neural Networks and compare the results. We can also perform feature selection to identify the most important features for prediction. Additionally, we can try out hyperparameter tuning to optimize the model.

2. **How can we visualize the performance of the model?**
   Answer: We can use confusion matrix to see the number of true positives, false positives, true negatives, and false negatives. We can also plot ROC curves to see the trade-off between sensitivity and specificity. Furthermore, we can use precision-recall curves to see the trade-off between precision and recall.

3. **How can we deal with missing values in the dataset?**
   Answer: We can either remove the rows with missing values or fill them with the mean, median, or mode of the feature. We can also use imputation techniques like KNN imputation or MICE imputation to estimate the missing values.

4. **How can we deal with imbalanced dataset?**
   Answer: We can use techniques like oversampling, undersampling, or SMOTE (Synthetic Minority Over-sampling Technique) to balance the dataset. We can also adjust the decision threshold to optimize the F1-score or ROC AUC score.

5. **How can we deploy the model for real-time prediction?**
   Answer: We can wrap the model in a Flask web application and deploy it to a cloud platform like AWS or Heroku. We can also use Docker to containerize the application for easier deployment and scalability. Additionally, we can use a load balancer to handle high traffic and ensure availability.