Feature Selection

Feature selection is the process of selecting a subset of relevant features for use in model construction. This can involve techniques such as Chi-squared test: Correlation Matrix and Logistics Regression Feature Importance.

The Chi-squared test is indeed used for carrying out univariate analysis on categorical variables, as it helps to determine the relationship between each categorical feature and the target variable individually. It is a useful technique for understanding the association between categorical features and a binary target variable.

The correlation matrix, on the other hand, is not a multivariate analysis technique specifically for feature selection. Instead, it is a way to measure the linear relationship between pairs of numerical features. In this context, it was used to assess potential multicollinearity among the numerical features, which can cause issues in linear models like logistic regression.

For a multivariate feature selection technique, we used logistic regression feature importance. This method assesses the importance of features in the context of other features by fitting a logistic regression model to the data and examining the coefficients associated with each feature. This approach takes into account the relationships among multiple features and their combined impact on the target variable, making it a multivariate analysis technique for feature selection.

Techniques for Feature Selection:

- **Chi-squared test:** we are going to use this test to determine the relationship between categorical features ('Education', 'City', 'PaymentTier', 'Gender', 'EverBenched') and the target variable ('LeaveOrNot'). The p-values resulting from the chi-squared test will show the significance of the association between each categorical feature and the target variable.
  Lower p-values indicate a stronger relationship. (NB: A p-value is a measure of the evidence against a null hypothesis.)

- **Correlation matrix:** For numerical features ('JoiningYear', 'Age', 'ExperienceInCurrentDomain'), we will calculate the correlation matrix to identify the linear relationships among these features. This method helps to assess the potential for multicollinearity, which can cause issues in linear models like logistic regression.
  - **Multicollinearity: It occurs when two or more independent variables in a regression model are highly correlated with each other. This can lead to unstable and unreliable estimates of the regression coefficients and make it difficult to determine the individual effect of each predictor on the dependent variable.** There are several methods for detecting and dealing with multicollinearity, such as calculating variance inflation factors (VIFs), centring and scaling the predictor variables, or using ridge regression or principal component analysis.

- **Logistic regression feature importance:**
  This Regression Analysis will used for estimating the relationship between a dependent variable (a.k.a, the outcome, or target variable) and one or more independent variables (a.k.a predictors, covariates, explanatory variables, or features.

Thus, we will use logistic regression as a model-based feature selection technique. By fitting a logistic regression model to the data, we can examine the coefficients associated with each feature. Larger absolute values of the coefficients indicate a more significant impact on the model's prediction. This analysis helps in understanding which features are the most important when predicting the target variable.

The goal of feature selection is to reduce the dimensionality of the data and improve model performance by removing irrelevant or redundant features.

```python
from sklearn.feature_selection import chi2
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder

# Divide the given list into categorical and numerical features
categorical_features = ['City_New Delhi', 'City_Pune', 'Education_Masters',
                        'Education_PHD', 'PaymentTier', 'Gender', 'EverBenched']

numerical_features = ['JoiningYear', 'Age', 'ExperienceInCurrentDomain']

# Perform chi-squared test for categorical features
X_cat = employee_df[categorical_features]
y = employee_df['LeaveOrNot']
chi_scores = chi2(X_cat, y)

# Display chi-squared test results in a dataframe
p_values = pd.Series(chi_scores[1], index=categorical_features)
p_values.sort_values(ascending=False, inplace=True)
print("Chi-squared test p-values for categorical features:")
print(p_values)

# Calculate the correlation matrix for numerical features
corr_matrix = employee_df[numerical_features].corr()
print("\nCorrelation matrix for numerical features:")
print(corr_matrix)

# Plot the correlation matrix heatmap
sns.heatmap(corr_matrix, annot=True, cmap="coolwarm")
plt.show()

# Feature importance using logistic regression
X_num = employee_df[numerical_features]
X = pd.concat([X_cat, X_num], axis=1)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

```python
# Standardize the numerical features
scaler = StandardScaler()
X_train[numerical_features] = scaler.fit_transform(X_train[numerical_features])
X_test[numerical_features] = scaler.transform(X_test[numerical_features])

# Train a logistic regression model
logreg = LogisticRegression(solver='liblinear')
logreg.fit(X_train, y_train)

# Display feature importance
feature_importance = pd.Series(logreg.coef_[0], index=X.columns)
feature_importance.sort_values(ascending=False, inplace=True)
print("\nFeature importance using logistic regression:")
print(feature_importance)

# Create a DataFrame for better visualization
importance_df = pd.DataFrame({'Feature': feature_importance.index, 'Importance': feature_importance.values})

# Visualize the feature importances using Plotly Express
fig = px.bar(importance_df, x='Feature', y='Importance', title='Feature Importance using Logistic Regression')
fig.show()
```

In the Python code above, the main objective is to perform feature selection on a dataset with both categorical and numerical features. The code follows several steps to achieve this goal:

Here's a description of what each code snippet does:

1. **from sklearn.feature_selection import chi2**
    - **Imports the chi-squared test function from the Scikit-learn library.**
2. **from sklearn.linear_model import LogisticRegression**
    - **Imports the Logistic Regression model from the Scikit-learn library.**
3. **from sklearn.model_selection import train_test_split**
    - **Imports the train_test_split function from the Scikit-learn library to split the dataset into training and testing subsets.**
4. **from sklearn.preprocessing import StandardScaler, LabelEncoder**
    - **Imports the StandardScaler for standardizing numerical features and the LabelEncoder for encoding categorical features from the Scikit-learn library.**
5. **Define categorical_features and numerical_features** using two separate lists, which represent the **categorical and numerical columns** of the dataset, respectively.
   categorical_features = [...]
    - **Creates a list of categorical feature column names.**

numerical_features = [...]

- **Creates a list of numerical feature column names.**

6. **X_cat = employee_df[categorical_features]**

    - **Creates a new DataFrame with only categorical features.**

7. **y = employee_df['LeaveOrNot']**

    - **Creates a target variable y representing the 'LeaveOrNot' column.**

8. **Conduct a Chi-squared test on the label-encoded categorical features (X_cat)** to calculate their p-values. P-values indicate the significance of each feature in relation to the target variable, 'LeaveOrNot'. Lower p-values suggest a stronger association between the feature and the target variable.

    chi_scores = chi2(X_cat, y)

    - **Performs chi-squared tests on the categorical features to determine their significance with respect to the target variable.**

    p_values = pd.Series(chi_scores[1], index=categorical_features)

    - **Creates a pandas Series to store p-values from the chi-squared test results.**

    p_values.sort_values(ascending=False, inplace=True)

    - **Sorts the p-values in descending order.**

9. **Display the p-values of the Chi-squared test results in a pandas Series, sorted in descending order. This helps identify the categorical features that have the strongest association with the target variable.**

    print("Chi-squared test p-values for categorical features:")

    - **Prints a message indicating that the following output will be the chi-squared test p-values for categorical features.**

    print(p_values)

    - **Prints the sorted p-values of the categorical features.**

10. **Calculate the correlation matrix for the numerical features to assess their pairwise linear relationships.** This matrix provides an overview of how the numerical features are related to one another.

    Display the correlation matrix for the numerical features. The matrix shows the correlation coefficients between each pair of numerical features.

    Plot a heatmap of the correlation matrix using Seaborn, a data visualization library. The heatmap visually represents the strength of correlations between numerical features through color intensity.

    corr_matrix = employee_df[numerical_features].corr()

    - **Calculates the correlation matrix for the numerical features.**

    print("\nCorrelation matrix for numerical features:")

    - **Prints a message indicating that the following output will be the correlation matrix for numerical features.**

    print(corr_matrix)

    - **Prints the correlation matrix.**

    sns.heatmap(corr_matrix, annot=True, cmap="coolwarm")

    - **Visualizes the correlation matrix using a heatmap.**

    plt.show()

    - **Displays the heatmap plot.**

11. **Combine the categorical features (X_cat)** and **numerical features (X_num) into a single DataFrame (X)** and **create the target variable series (y).** This step prepares the data for model training and testing.

    Split the dataset into training (70%) and testing (30%) sets using the train_test_split function. This allows for model evaluation on unseen data.

    X_num = employee_df[numerical_features]

    - **Creates a new DataFrame with only numerical features.**

12. **X = pd.concat([X_cat, X_num], axis=1)**

    - **Concatenates the categorical and numerical features into a single DataFrame X.**

13. **X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)**

    - **Splits the dataset into training and testing subsets.**

14. **Standardize the numerical features using the StandardScaler class.** Standardization scales the numerical features to have a mean of 0 and a standard deviation of 1. This is important because logistic regression is sensitive to feature scaling, and standardization helps improve model performance.

    Train a logistic regression model on the standardized training dataset using the LogisticRegression class. The solver **'liblinear'** is chosen for small datasets or datasets with a large number of features.

    Display the feature importance values based on the coefficients of the logistic regression model. These values represent the impact of each feature on the model's prediction. The higher the absolute value of the coefficient, the more important the feature is in predicting the target variable.

    scaler = StandardScaler()

    - **Creates a StandardScaler object to standardize the numerical features.**

    X_train[numerical_features] = scaler.fit_transform(X_train[numerical_features])

    - **Fits the scaler on the training dataset and standardizes the numerical features.**

    X_test[numerical_features] = scaler.transform(X_test[numerical_features])

    - **Standardizes the numerical features in the test dataset using the scaler fitted on the training dataset.**

    logreg = LogisticRegression(solver='liblinear')

    - **Creates a logistic regression model object.**

    logreg.fit(X_train, y_train)

    - **Fits the logistic regression model on the training data**

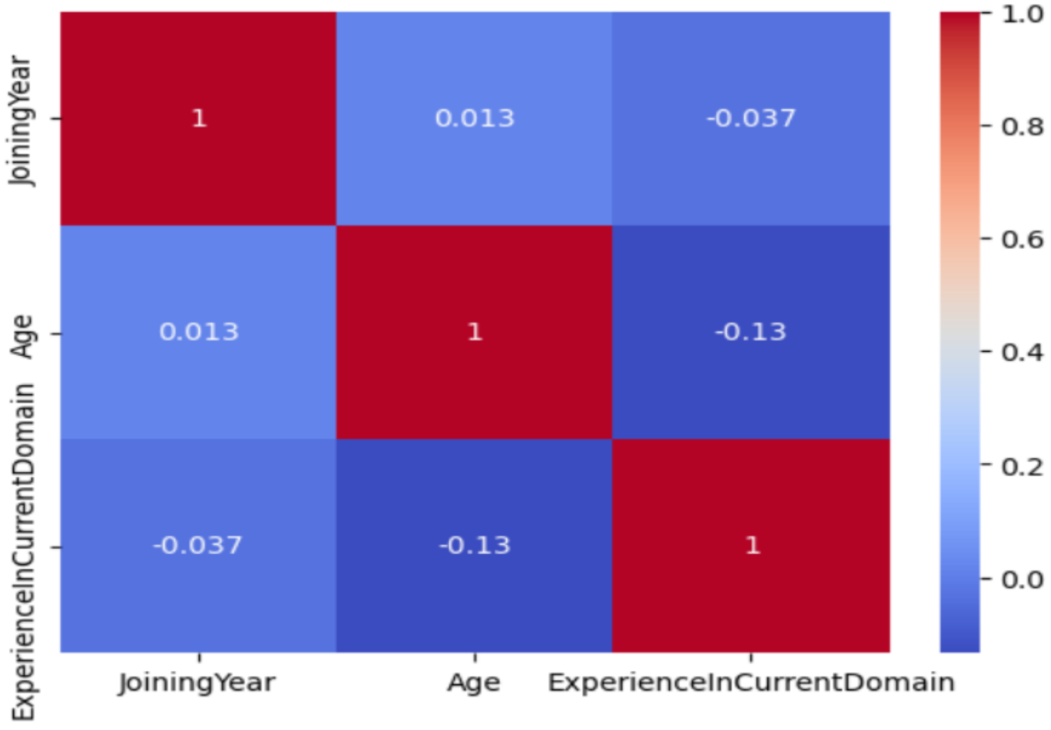
When executed, this code will output the following information:

- Chi-squared test p-values for categorical features.
- Correlation matrix for numerical features.
- Heatmap of the correlation matrix.
- Feature importance values using logistic regression.

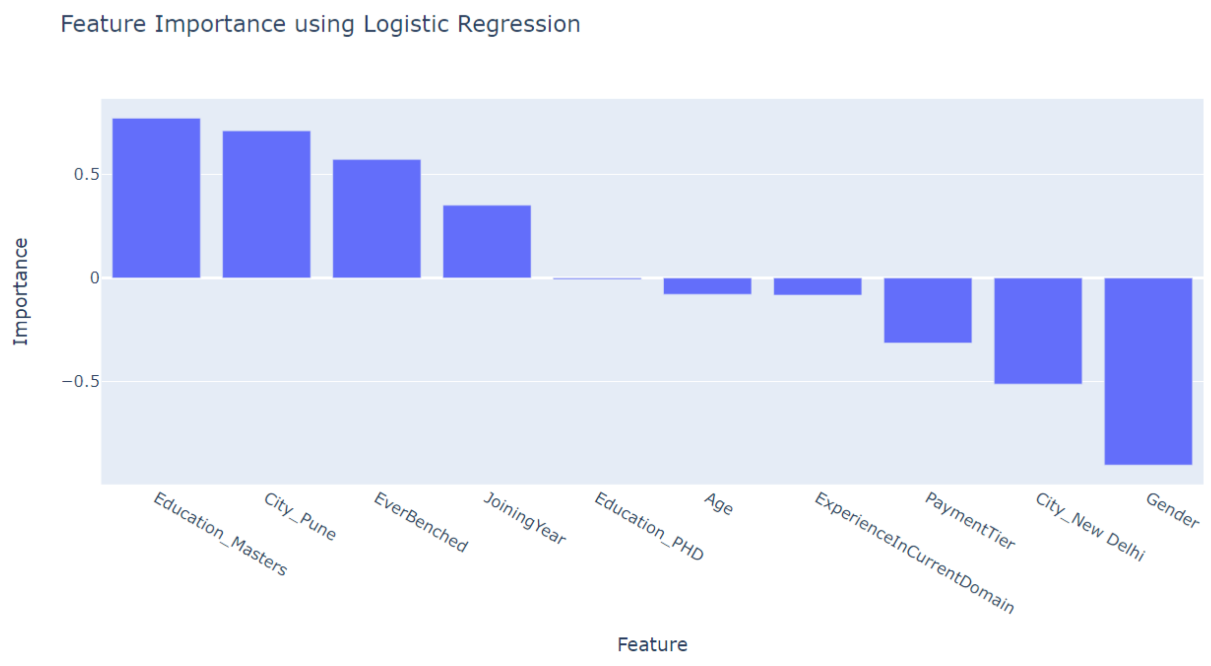
Results

```
Chi-squared test p-values for categorical features:
City_New Delhi       4.868372e-02
Education_PHD        9.200673e-03
PaymentTier          4.079040e-06
EverBenched          4.016083e-07
Education_Masters    3.128726e-19
Gender               1.216743e-21
City_Pune            3.526349e-33
dtype: float64

Correlation matrix for numerical features:
                           JoiningYear       Age  ExperienceInCurrentDomain
JoiningYear                   1.000000  0.013165                  -0.036525
Age                           0.013165  1.000000                  -0.134643
ExperienceInCurrentDomain    -0.036525 -0.134643                   1.000000
```



```
Feature importance using logistic regression:
Education_Masters            0.772361
City_Pune                    0.711685
EverBenched                  0.573131
JoiningYear                  0.351787
Education_PHD               -0.006417
Age                         -0.079001
ExperienceInCurrentDomain   -0.082824
PaymentTier                 -0.314216
City_New Delhi              -0.512243
Gender                      -0.903554
dtype: float64
```



Based on the results from the chi-squared test, correlation matrix, and logistic regression feature importance, we can provide the following data analytics report:

****Chi-squared Test for Categorical Features:****

The chi-squared test is used to determine whether there is a significant relationship between two categorical variables. In this case, we are testing the relationship between each categorical feature and the target variable 'LeaveOrNot'.

****The p-values from the test are as follows:****

- City_Pune: 3.53e-33

- Gender: 1.22e-21

- Education_Masters: 3.13e-19

- EverBenched: 4.02e-07

- PaymentTier: 4.08e-06

- Education_PHD: 9.20e-03

- City_New Delhi: 4.87e-02

Lower p-values indicate stronger relationships between the categorical feature and the target variable. In this case, **'City_Pune' and 'Gender' have the lowest p-values, suggesting that these features have the strongest relationships with the target variable.** On the other hand, 'City_New Delhi' has the highest p-value, indicating a weaker relationship with the target variable.

Correlation Matrix for Numerical Features:

The correlation matrix shows the correlation coefficients between numerical features. These coefficients range from -1 to 1, with values close to -1 or 1 indicating strong negative or positive correlations, respectively, and values close to 0 indicating weak or no correlation.

****

JoiningYear has a weak positive correlation with Age (0.013) and a weak negative correlation with ExperienceInCurrentDomain (-0.037).

Age has a weak negative correlation with ExperienceInCurrentDomain (-0.135). Therefore, the numerical features have weak correlations with each other.

****

**Logistic Regression Feature Importance: **

The logistic regression model assigns a coefficient to each feature, which indicates the strength and direction of the relationship between the feature and the target variable. Positive coefficients suggest that an increase in the feature value leads to an increased probability of the target variable being 1 (employee leaving), while negative coefficients suggest the opposite.

- City_Pune: 0.712 (positive relationship)
- Education_Masters: 0.772 (positive relationship)
- EverBenched: 0.573 (positive relationship)
- JoiningYear: 0.352 (positive relationship)

- Education_PHD: -0.006 (negative relationship)
- Age: -0.079 (negative relationship)
- ExperienceInCurrentDomain: -0.083 (negative relationship)
- PaymentTier: -0.314 (negative relationship)
- City_New Delhi: -0.512 (negative relationship)
- Gender: -0.904 (negative relationship)

Based on the logistic regression model, 'Gender' has the strongest negative relationship with the target variable, indicating that a change in gender has a significant impact on the likelihood of an employee leaving.

'Education_Masters' and 'City_Pune' have the strongest positive relationships, suggesting that having a master's degree and being located in Pune also have notable effects on the target variable.

<br>Other features such as 'EverBenched', 'JoiningYear', and 'PaymentTier' also show moderate relationships with the target variable.

Our Inference:

Taking into account the results from the chi-squared test, correlation matrix, and logistic regression feature importance, we can conclude that 'City_Pune', 'Gender', 'Education_Masters', and 'EverBenched' are the most important features for predicting whether an employee will leave or not.

Furthermore, 'PaymentTier' is a categorical variable representing different levels or categories of payment for employees. Its negative relationship implies that higher payment tiers are associated with a lower likelihood of employees leaving the company.

We can, therefore, include 'PaymentTier' among the selected features ('City_Pune', 'Gender','Education_Masters', and 'EverBenched') for your logistic regression model. The chi-squared test and the logistic regression feature importance analysis both suggest that 'PaymentTier' is significantly related to the target variable 'LeaveOrNot', and can potentially help improve the predictive performance of the model.

Adding 'PaymentTier' to the list of selected features would help capture the relationship between the different payment levels and employee attrition, providing additional information to the model. This comprehensive set of features, which also includes 'City_Pune', 'Gender', 'Education_Masters', and 'EverBenched', will allow the model to account for various factors that influence an employee's decision to leave or stay with the company.

Consequently, by considering the findings from the chi-squared test, correlation matrix, and logistic regression feature importance, we have identified 'City_Pune', 'Gender', 'Education_Masters', 'EverBenched', and 'PaymentTier' as the most important features for predicting employee attrition. Incorporating these features into your logistic regression model will improve its predictive performance and help you better understand the factors that drive employee turnover.