Problem Statement:

Develop a predictive model using employee data to classify individuals as likely to stay or leave the company. This classification will assist in making informed decisions about employee retention strategies and workplace improvements.

Overview

The dataset contains 1350 rows and 15 columns, representing various employee metrics. The data aims to reflect realistic scenarios in a corporate setting, encompassing professional and personal employee metrics. Below is a brief overview of the dataset columns:

1. **JobSatisfaction**: Employee's job satisfaction level.
2. **Performance rating**: Performance rating given by the company.
3. **YearsAtCompany**: Total number of years the employee has been with the company.
4. **WorkLifeBalance**: Rating of how well the employee feels they balance work and personal life.
5. **DistanceFromHome**: Distance from the employee's home to the workplace.
6. **Monthly Income**: The monthly income of the employee.
7. **EducationLevel**: The highest level of education attained by the employee.
8. **Age**: The age of the employee.
9. **NumCompaniesWorked**: The number of companies the employee has worked at before joining the current company.
10. **Employee Role**: The role or position of the employee within the company.
11. **Annual Bonus**: Annual bonus received by the employee.
12. **Training hours**: Number of hours spent in training programs.
13. **Department**: The department in which the employee works.
14. **AnnualBonus_Squared**: Square of the annual bonus (a polynomial feature).
15. **AnnualBonus_TrainingHours_Interaction**: Interaction term between annual bonus and training hours.

Analysis

The analysis below contains the details of the Employee turnover dataset where we're using a Logistic regression for classification, i.e. to classify individuals as likely to stay or leave the company. We use different regularization techniques to see whether they perform any better compared to a simple logistic regression without any regularization.

After reading the dataset, we print the missing values in any column - there are no missing values -

```
In [33]: print(missing_values)
Job_Satisfaction                          0
Performance_Rating                        0
Years_At_Company                          0
Work_Life_Balance                         0
Distance_From_Home                        0
Monthly_Income                            0
Education_Level                           0
Age                                       0
Num_Companies_Worked                      0
Employee_Role                             0
Annual_Bonus                              0
Training_Hours                            0
Department                                0
Annual_Bonus_Squared                      0
Annual_Bonus_Training_Hours_Interaction   0
Employee_Turnover                         0
dtype: int64
```

After splitting the data into 70% train and 30% test data and scaling the X_train and X_test, a Logistic regression model with no regularization is first chosen to fit the data. The classification report for this model shows a F1 score of **90%**.

```
In [12]: print("Logistic Regression without regularization:\n", classification_report(y_test, y_pred))
Logistic Regression without regularization:
              precision    recall  f1-score   support

           0       0.90      0.89      0.90       203
           1       0.89      0.90      0.90       202

    accuracy                           0.90       405
   macro avg       0.90      0.90      0.90       405
weighted avg       0.90      0.90      0.90       405
```

To improve further, we try different types of regularization using different levels of regularization strengths and cross validation to pick the best C values.

The L1 regularization classification reports shows a 89% F1 score -

```
In [17]: print("L1 Regularization:\n", classification_report(y_test, y_pred_l1))
L1 Regularization:
              precision    recall  f1-score   support

           0       0.89      0.90      0.89       203
           1       0.90      0.89      0.89       202

    accuracy                           0.89       405
   macro avg       0.89      0.89      0.89       405
weighted avg       0.89      0.89      0.89       405
```

The L2 regularization classification reports similarly shows a 89% F1 score as well-

```
In [21]: print("L2 Regularization:\n", classification_report(y_test, y_pred_l2))
L2 Regularization:
              precision    recall  f1-score   support

           0       0.90      0.89      0.89       203
           1       0.89      0.90      0.89       202

    accuracy                           0.89       405
   macro avg       0.89      0.89      0.89       405
weighted avg       0.89      0.89      0.89       405
```

And finally, the Elastic net Regularization , which uses a 0.5 L1 ratio, shows a 90% F1 score, i.e. going back to the same level as the logistic regression model without any regularization -

```
In [25]: print("Elastic Net Regularization:\n", classification_report(y_test, y_pred_en))
Elastic Net Regularization:
              precision    recall  f1-score   support

           0       0.90      0.89      0.90       203
           1       0.89      0.90      0.90       202

    accuracy                           0.90       405
   macro avg       0.90      0.90      0.90       405
weighted avg       0.90      0.90      0.90       405
```

The best C values for each type of regularization are shown below -

```
In [29]: logistic_l2_cv.C_
Out[29]: array([12.68979592])

In [30]: logistic_l1_cv.C_
Out[30]: array([0.1])

In [31]: logistic_en_cv.C_
Out[31]: array([10.65918367])
```

Conclusions:

- **Regularization Strength (C)**:
  - C is the inverse of the regularization strength (α): α = 1/C
  - Larger C implies weaker regularization (less penalty on coefficients).
  - Smaller C implies stronger regularization (more penalty on coefficients).
- **Interpretation of Values**:
  - **L2 Regularization (C=12.69)**:
    - The relatively high C suggests weaker regularization.

    - L2 regularization generally encourages smaller coefficients but allows non-zero values for all features.
    - This indicates the model benefits from retaining more features with minimal penalty.
- **L1 Regularization (C=0.1)**:
    - The small C indicates strong regularization.
    - L1 regularization performs feature selection by driving some coefficients to exactly zero.
    - This suggests the model benefits from strong sparsity, removing less important features to focus on the most predictive ones.
- **Elastic Net (C=10.66)**:
    - The C value is closer to L2's C than L1's, suggesting the elastic net is leaning toward retaining more features while balancing sparsity.
    - The mix of L1 and L2 penalties allows the model to handle correlated features effectively, unlike pure L1.

For each of the different Logistic regression models with L1, L2 and Elastic Net regularization types, we look at the best C values against the mean cross val score and the C values and cross-validated accuracy for each regularization type are visualized in a bar chart shown below-.