Machine learning model training

Attrition Classification

Implement any classifier using library functions to predict whether an employee will leave the company or not.

Data: Test and Train data are given with 33 and 34 columns respectively and 1028 rows.
The column vector consists of:

```
Age, Attrition, BusinessTravel, DailyRate, Department,
DistanceFromHome, Education, EducationField, EmployeeCount,
EmployeeNumber, EnvironmentSatisfaction, Gender, HourlyRate,
JobInvolvement, JobLevel, JobRole, JobSatisfaction,
MaritalStatus, MonthlyIncome, MonthlyRate,
NumCompaniesWorked, OverTime, PercentSalaryHike,
PerformanceRating, RelationshipSatisfaction, StockOptionLevel,
TotalWorkingYears, TrainingTimesLastYear, WorkLifeBalance,
YearsAtCompany, YearsInCurrentRole, YearsSinceLastPromotion,
YearsWithCurrManager, ID
```

Data Processing:

- **Encoding**
  The first step taken was to encode the data such that we are left with numerical values only for the categorical data. Label Encoding converts categorical labels into numerical values. For the categorical data which have more than two unique values are encoded using One Hot Encoding. Features like, BusinessTravel, Department, OverTime, etc are converted based on their unique column values. One hot encoding is used to avoid any bias with large values assigned to data having more than two unique values.
- **Scaling**
  The classification algorithms works best for similarly scaled features. Thus feautre scaling is apllied using MinMaxScaler to range values between 0 to 1.
- **Removing Target and redundant features**
  The target feature : Attrition is removed from train data for model training.
  Also the features like- EmployeeNumber and ID which are unique for every row are removed as they will not help in any training. The feature called EmployeeCount which has same value all over is also removed.
  Lastly the feautures like, MonthlyRate, HourlyRate and DailyRate seems correlated and removed except one for reducing irrelevant features. Thus reducing efforts on model training.
  After above data processing, the columns for test and train data are,

  Age, DailyRate, DistanceFromHome, Education,
  EnvironmentSatisfaction, Gender, JobInvolvement, JobLevel,
  JobSatisfaction, MonthlyIncome, NumCompaniesWorked,
  OverTime, PercentSalaryHike, PerformanceRating,
  RelationshipSatisfaction, StockOptionLevel, TotalWorkingYears,
  TrainingTimesLastYear, WorkLifeBalance, YearsAtCompany,
  YearsInCurrentRole, YearsSinceLastPromotion,
  YearsWithCurrManager, BusinessTravel_Travel_Frequently,
  BusinessTravel_Travel_Rarely, Department_Research & Development,
  Department_Sales, EducationField_Life Sciences,
  EducationField_Marketing, EducationField_Medical,
  EducationField_Other, EducationField_Technical Degree,
  JobRole_Human Resources, JobRole_Laboratory Technician,
  JobRole_Manager, JobRole_Manufacturing Director,
  JobRole_Research Director, JobRole_Research Scientist,
  JobRole_Sales Executive, JobRole_Sales Representative,
  MaritalStatus_Married, MaritalStatus_Single

Train-Test data split & Cross validation

Using train_test_split helper function from scikit-learn, one can evaluate hyper paramaters for their classifier. However, there is a risk of overfitting on the test set because the hyper-parameters can be tuned for optimal values. By spliting the available data into two sets, the learning model can be deprived of the sufficient number of training samples. Also, a particluar choice of training and validation data may increase the accuracy of overall result but have poor predictability for the new data.
Cross validation – It helps to solve the above problem where training samples are split into k sets (k-folds) for validation. An average of the k folds can be taken as good measure of accuracy for that model.

Results

Without eliminating the irrelevant features (the last step of data processing) following results were obtained.

Classifier	Cross-Validate Accuracy
Logistic Regression	0.862568093385214
KNeighborsClassifier	0.836575875486381
Support Vector Machine	0.874513618677043
Random Forest	0.639105058365759
Decision Tree Classifier	0.529182879377432
Multi-layer Perceptron Classifier	0.837548638132296
Gaussian Naive Bayes	0.655642023346304

As observed above, the **Support Vector Machine** has highest accuracy of **87.45%**. This data when put into Kaggle for its test data **score** it came to be, **0.90909.** The linear regression has accuracy of 86.25%. Its Kaggle score was **0.89898.**

Now, performed both train-test split and cross validartion. Cross validation gave better results. The irrelevant data has been removed and a cross validation of 5 folds is performed for all classifiers.

Classifier	Cross-Validate Accuracy
Logistic Regression	0.86861381370251
KNeighborsClassifier	0.843380063855514
Support Vector Machine	0.87451381370251
Random Forest	0.84826280284753
Decision Tree Classifier	0.775313929585053
Multi-layer Perceptron Classifier	0.848173185488714
Gaussian Naive Bayes	0.678097215501492

As observed above, the accuracy of all the above classifiers have been improved. The **Support Vector Machine** still has highest accuracy of **87.45%**. This data when put into Kaggle for its test data **score** it came to be, **0.90909.**

Thus, for every classifier, cross validation is done to measure accuracy of a model. Folds are optimally selected. To find the optimal soltuion for value of k for k-folds validation, trail and errors were performed and following results shows the optimal accuracy obtained for the trained model belonging to that classifier.

Accuracy for Cross Validated Result:

Classifier	Cross-Validate Accuracy
Logistic Regression	0.868681032441392
KNeighborsClassifier	0.848303167420815
Support Vector Machine	0.87984126984127
Random Forest	0.846259280411193
Decision Tree Classifier	0.774290881401104
Multi-layer Perceptron Classifier	0.844
Gaussian Naive Bayes	0.664391776127927

As observed above, the **Support Vector Machine** has highest accuracy of **87.98%**. This data when put into Kaggle for its test data **score** it came to be, **0.91414.**