**Data Evaluation**

Retrieved from Kaggle, I have a 250 records \* 300 features training dataset and a 19750 records \* 300 features testing dataset. During evaluation, I noticed that:

1. **Data is well-constructed:**

The dataset appears to be well-constructed, indicating that it is organized, consistent, and without missing values or errors.

1. **Limited training set:**

The training set consists of only 250 records. A small training set may pose challenges for machine learning models, particularly complex ones, as they may struggle to generalize well with limited data. Techniques such as cross-validation would be necessary to mitigate overfitting.

1. **High dimensionality:**

The dataset exhibits with 300 features. We consider this as high-dimensional data as it is larger than the number of records (i.e. 250). Such phenomenon can lead to several issues, including the curse of dimensionality, increased computational complexity, and overfitting. Dimensionality reduction would be required for the training.

一張含有 文字, 字型, 印刷術, 設計 的圖片

自動產生的描述

Fig.1 Data is imported as Pandas Dataframe

**Data Preprocessing**

For boosting performance, we tend to preprocess data before using them to train a model. The following operations are performed:

1. **Data Standardization**

In general, we perform standardization for each column in the data. This can reduce the bias of each record, retrieving a more generalized result.

1. **Principle Component Analysis (PCA)**

Since the training dataset has a high dimensionality, we reduce it with PCA. To conclude, PCA reduces dimensionality, captures underlying patterns, and enhances interpretability by transforming features into linear combinations of original variables, retaining most variance, and aiding in data exploration and modeling.

However, while reducing dimensionality can improve computational efficiency and mitigate the curse of dimensionality, it also occurs a loss of information. PCA may not preserve all important characteristics of the data, leading to a potential loss of discriminative power. Therefore, PCA will be used when training kNN models only.

**Model Selection**

To facilitate my study in the competition, I proposed a variety of base models, including k-nearest neighbours, support vector machine, and logistic regression. In addition, a few ensemble models with different base models are proposed. Under the sense of collective wisdom, these models genuinely performs better than the base models. Therefore, experiments around the hypothesis are conducted.

Foe each model, I will examine the accuracy by cross validation with 5 folds. In addition, to ensure that each model has its hyperparameters fine-tuned, grid search approach will be applied. The proposed hyperparameters in this procedure are provided by ChatGPT. The accuracy of shown plot are the maximum score of models which used the corresponding hyperparameter.

**Base Model**

1. **K-Nearest Neighbors**

kNN is a simple and intuitive algorithm used for classification and regression tasks. It classifies data points based on the majority class of their k nearest neighbors in the feature space, making it non-parametric and robust to noisy data. It is known that the curse of dimensionality often occurs when using kNN model due to its sensitiveness to dimensionality. This results a reduction of model accuracy. Therefore, it is suggested to perform PCA on the dataset before training [1].

Through cross-validation, the KNN model attained its highest accuracy of 73.2% with a value of k set to 17. This process involved systematically evaluating different values of k and selecting the one that yielded the optimal performance on unseen data.

一張含有 字型, 文字, 螢幕擷取畫面 的圖片

自動產生的描述 一張含有 文字, 字型, 螢幕擷取畫面 的圖片

自動產生的描述

Fig.2 Cross Validation result of kNN

1. **Support Vector Machine**

SVM optimize a hyperplane to effectively classify data points in feature space, leveraging support vectors to define this boundary's position and orientation. By maximizing the margin between support vectors of distinct classes and utilizing kernel functions for nonlinear relationships, SVM achieves robustness against overfitting and enhances its generalization capabilities.

After fine-tuning the SVM model using cross-validation, we achieved the highest accuracy of 73.2%. This optimal performance was obtained with the following hyperparameters: C set to 0.1, gamma set to 'scale', and kernel set to 'rbf'.

一張含有 文字, 字型, 螢幕擷取畫面, 數字 的圖片

自動產生的描述

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

Fig.3 Cross Validation result on SVM

1. **Logistic Regression**

In essence, Logistic Regression offers a straightforward yet effective approach to binary classification, estimating the probability of a sample belonging to a specific class using the sigmoid function. By iteratively refining model parameters through optimization techniques like maximum likelihood estimation or gradient descent, Logistic Regression achieves accurate predictions by adjusting to the underlying data distribution and applying a threshold to delineate class boundaries based on calculated probabilities.

Following cross-validation, the Logistic Regression model achieved its highest accuracy of 74.8%. This optimal performance was attained using the following hyperparameters: C value set to 0.1, penalty set to 'l1', and solver set to 'liblinear'.

一張含有 字型, 文字, 螢幕擷取畫面 的圖片

自動產生的描述

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

Fig.4 Cross Validation result on logistic regression

The specific issue found in SVM predictions is that the values are nearly all 0, which should consider bias with previous experiences. Thus, considering the high computational expense and adnominal performance, it is better to consider other models.

All predictions provided by these models remain a private score of 0.5. Thus, we are expecting ensemble models have a higher performance.

**Ensemble Models**

After retrieving the optimized base models, we are able to work with ensemble models. There are two types of models, including bagging and boosting. For each ensemble model, we will try different base model and retrieve the best model among them.

**Bagging**

For each bagging classifier, it contains 2000 base models for prediction. To facilitate the understanding of performance difference, each classifier with varied base model will be implemented 5 times. The average score determines the overall performance of a classifier.

**Excluding SVM-based Ensemble Model**

Both kNN and LR models are used as base models. SVM is excluded because of its extreme computational cost on **each iteration**. It is tested that SVM-based Bagging has only 0.5 accuracy score. Considering the low return, the model is excluded from further testing.

**Random Forest Classifier**

Inspired by the lectures, RandomForestClassfier() is included to compare with other bagging classifiers. Same as previously, the n\_estimator is set to 500 for control experimenting purpose. Since decision tree is likely to suffer from the curse of dimensionality, PCA will be performed beforehand.

By grid search, we retrieved 0.732 validation score with the best hyperparameters max\_depth = 10, min\_simples\_leaf = 1, and min\_samples\_split = 2.

**Performance result of bagging models**

By submitting the prediction on Kaggle, the following private accuracy score is given:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Iteration | kNN Bagging | SVM Bagging | LR Bagging | RandomForestClassifier() |
| 1 | 0.500 | 0.500 | 0.499 |  |
| 2 | 0.499 | 0.500 | 0.500 |  |
| 3 | 0.500 | 0.500 | 0.499 |  |
| 4 | 0.500 | 0.500 | 0.500 |  |
| 5 | 0.500 | 0.500 | 0.499 |  |