COMP3212 Computational Biology

Gongwei Shi : gs1n17
Eugene Teoh : wct1c16
Binti Kamarudin, Nur Afiqah : nabk1g17

1 Principal Components Analysis

Principal Components Analysis (PCA) is a method that reduces the dimensionality of data by extracting new features — principal components (PCs) — that are unrelated to each other but still retains the characteristics of the original features. The data is projected onto a lower-dimensional subspace using the directions given by the PCs. Each PC is a linear combination of the original features that maximises the variance in the projected data. Essentially, PCA clusters the data to show the most important features present in a dataset.

For analysis of cancer, we used SciKit-learn's PCA on RNA-seq gene expression data to analyse which genes (features) are the most important in explaining variation between cancer samples. The GDC TCGA cancer datasets (for cancers BLCA, BRCA, CESC, COAD, UCEC, ESCA, GBM) used had raw read counts for 60,483 genes. Before applying PCA, the data was standardised using mean normalisation to avoid PCA results being affected by differences in the range of each feature.

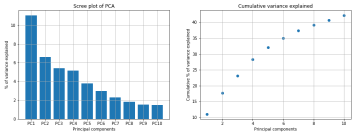

Figure 1: Left: Percentage of variance explained by each principal component. Right: Cumulative of percentage variance explained.

Using PCA, 10 principal components were found. Figure 1 shows the variance explained by each PC. The variance explained is the variance of the original data that is retained in the projection of the data. The total cumulative variance explained for 10 components is not very high, at approximately 45%. This indicates that only half of the variations in the data is described by the components found.

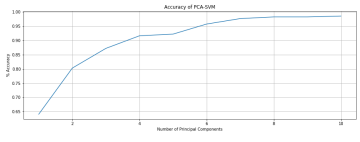

Figure 2: Accuracy of classification of cancer using SVM with respect to increasing number of principal components in PCA.

However, with respect to classification of cancer, 10 components were sufficient. To evaluate the usefulness of PCA for cancer classification, we trained a SciKit-learn SVM classifier on data that was transformed with PCA. Figure 2 shows the accuracy of the classifier with respect to different numbers of principal components. The accuracy curve plateaus at 8 PCs. Hence, for cancer classification, only the first 8 components are significant. We also plotted the confusion matrix for predicted vs true labels, which shows that cancers BRCA, COAD, and GBM are easiest to classify accurately, indicating that samples of these cancers are more separated from the rest of the samples in the PCA projections. Figure 3 show the distribution of the projected data on the first three components. The distribution of cancers BRCA, COAD and GBM are more separated from other cancers' clusters, unlike other samples that are overlapped with each other.

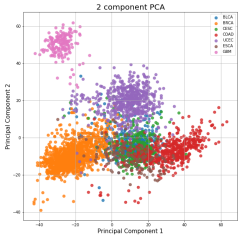

Figure 3: Projection of data on the first two PCs. Total variance explained = 17%.

$$Loadings = Eigenvectors \cdot \sqrt{Eigenvalues} \quad (1)$$

To meaningfully interpret the PCs found and evaluate which genes contributes to each PC, the loadings of each PC were evaluated. Loadings are the coefficients of the linear combinations of the original variables from which the PCs are constructed. Loadings measure the correlation between the original features and the unit-scaled principal components, given by equation 1. Positive loading values signify correlation while negative values signify anti-correlation. The higher the absolute value of a gene's loading, the more the gene contributes to the PC. The first 4 PCs had greater correlations with the original genes on average, indicating that these PCs were more descriptive of the original data. To qualitatively evaluate the genes found, we interpreted the sets of genes found for each PC based on their functional characteristics. Using the Gene Ontology knoweledgebase[1], we performed gene ontology (GO) enrichment analysis on

[1] http://geneontology.org/ accessed on 26th May 2020

Principal Component	Summary of GO terms
1	digestive system, protein dimerization
2	cornification, neural development
3	endopeptidase, cornification
4	N/A
5	phagocytosis, response to bacterium, humoral immune response
6	stem cell, respiratory burst in phagocytes, phagocytosis, response to bacterium, humoral immune response
7	metabolic process
8	N/A
9	cornification, adhesion between cells
10	N/A

Table 1: Summary of gene ontology analysis for top 40 significant genes in each component. Some components did not have results that were statistically significant (as defined by the tool used).

each set of genes. Table 1 summarises the results. Unfortunately, the tool were not able to give statistically significant results for all sets of genes. However, the rest of the results indicates the types of genes that may be useful in distinguishing between the cancers present in the dataset.

2 Support Vector Machine

Support Vector Machine (SVM) is a supervised learning model used in both classification and regression problems. In our application, we will be working with Support Vector Classification (SVC). Classification is a statistical problem where given several training samples from different classes, a model attempts to predict which class a new sample belongs to. SVCs are non-probabilistic classifiers, where the model outputs the most probable class, instead of outputing a probablility distribution of every class. We will be using a linear kernel. The optimisation problem is to minimize $\|\vec{w}\|$,with constraints for the decision boundary described as,

$$y_i(\vec{w} \cdot \vec{x_i} - b) \geq 1, \text{ for all } 1 \leq i \leq n. \qquad (2)$$

where x_i are the training examples and y_i is 1 for class 1, -1 for the other class for the ith example. $\vec{w}$ are the weights of the model, and b is the bias offset.

2.1 One-vs-One for Multi-Class Classification

SVMs are traditionally binary classifiers. To allow it to be used in multi-classification problems, such as predicting more than 2 types of cancers, the "One-vs-One" (OVO) approach is used. The OVO approach creates several number of classifiers and splits the dataset into binary classification problems. The number of classifiers created is equal to $(N \times (N-1))/2$, where N is the total number of classes. For example, suppose there are four classes: A, B, C and D, OVO splits the problem into:

- Classifier 1: A vs. B
- Classifier 2: A vs. C
- Classifier 3: A vs. D
- Classifier 4: B vs. C
- Classifier 5: B vs. D
- Classifier 6: C vs. C

2.2 Recursive Feature Elimination

Recursive feature elimination (RFE) is a feature selection method where a feature with the smallest weight is removed for every iteration until a specified number of features is reached [1]. The algorithm is summarized in Figure 4. RFE can also be used with K-fold cross validation to prevent over-fitting. In our analysis, we will be using 5-fold cross validation.

Algorithm 1: Recursive feature elimination

```
Tune/train the model on the training set using all predictors
Calculate model performance
Calculate variable importance or rankings
for Each subset size S_i, i = 1...S do
    Keep the S_i most important variables
    [Optional] Pre–process the data
    Tune/train the model on the training set using S_i predictors
    Calculate model performance
    [Optional] Recalculate the rankings for each predictor
end
Calculate the performance profile over the S_i
Determine the appropriate number of predictors
Use the model corresponding to the optimal S_i
```

Figure 4: RFE algorithm [2].

The datasets we will be using for the analysis will be similar to the PCA method. The gene expression data of GDC TCGA cancers: BLCA, BRCA, GBM will be used. Each dataset consist of 60,483 genes (or features). Each gene is z-score standardized to range [-1, 1].

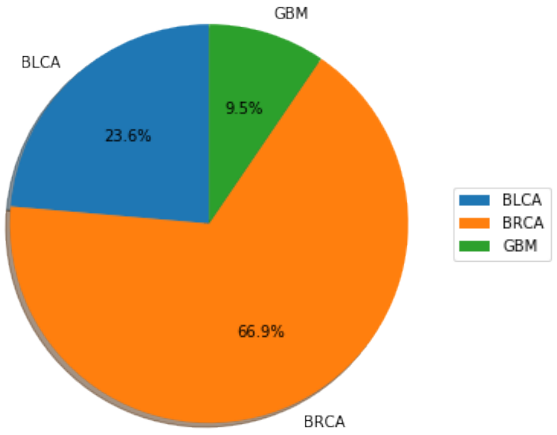

Figure 5: Dataset distribution for each cancer type.

2.3 Implementation

Figure 6 shows the first round of RFE with 3000 features per step. It shows that the optimal number of features to give the optimal accuracy is 3483 features.

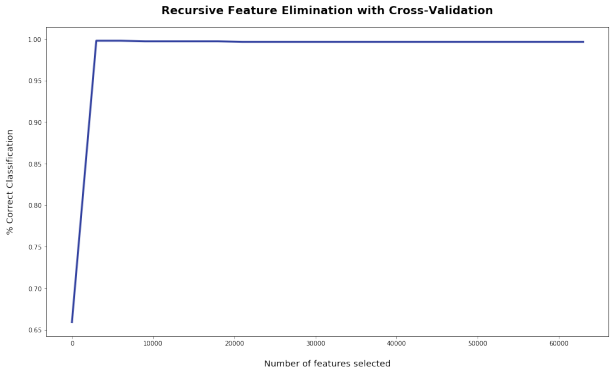

Figure 6: RFE results with 3000 features per step.

3 Random Decision Forests

Random Decisions Forests (RDF) is a tree-based classifier built on multiple decision trees in randomly selected subspaces of the feature space. Tress in each subspace contributes their classification to the final decision. This approach derives the importance of each feature.

3.1 Building RDF

The random forest instance is imported from the sklearn library, where a complete implementation of REF was provided. We need to define the number of decisions tree in the specify random forest instance; in our experiment, we have chosen 10k decision trees. With the increasing number of decision trees, random decision forests can make a more accurate decision on the importance of each feature.

For feature selection, we used the SelectFromModel provided in the sklearn library. It takes in the instance of RandomForests Classifier and returns a list of selected features. The feature selection based on the importance of weight, it automatically selects features with their importance weight over the threshold. The threshold can be manually defined; however, in our experiment, we have used the default threshold, which is the averaged importance weight. The fitting process can start after the SelectFromModel is initialized. We run the model on the training dataset. The model returns a list of Boolean variables indicating whether the index of the corresponding feature is selected, the selected feature associated with their importance weight, and we have sorted the selected feature in the decreasing order of importance.

The top 10 features with the highest feature important weight is in figure[7]

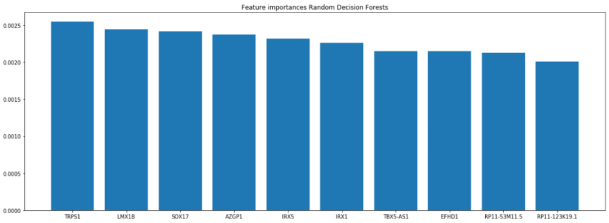

Figure 7: RFE algorithm [2].

References

[1] Isabelle Guyon et al. "Gene Selection for Cancer Classification Using Support Vector Machines". In: *Machine Learning* 46 (Jan. 2002), pp. 389–422. DOI: 10.1023/A:1012487302797.

[2] Max Kuhn. *The caret Package*. Mar. 2019. URL: https://topepo.github.io/caret/recursive-feature-elimination.html.