Feature selection and extraction

Feature selection means: given a set of potential features, some are selected and rest are discarded. Feature selection is applied either **to prevent redundancy and/or irrelevancy existing in the features** or just to get a limited number of features to **prevent from overfitting**.

Feature extraction means **the translation of raw data into the inputs** that a particular Machine Learning algorithm requires. Features must represent the information of the data in a format that will best fit the needs of the algorithm that is going to be used to solve the problem.

- The unique case when we wouldn't need any feature extraction is when our algorithm can perform feature extraction by itself as in the deep learning neural networks, that can get a low dimensional representation of high dimensional data.
- We should apply feature selection, when there is a suspicion of **redundancy or irrelevancy**, since these affect the model accuracy or simply add noise at best.
- Feature selection may be performed only to reduce the number of features, in order **to favor interpretability and computing feasibility or to avoid the curse of dimensionality phenomena**, i.e., too many features to describe not enough samples.

Feature selection

Feature selection has four different approaches such as filter approach, wrapper approach, embedded approach, and hybrid approach.

1. ***Filter based:*** We specify some metric and based on that filter features. An example of such a metric could be correlation/chi-square.
2. ***Wrapper-based:*** Wrapper methods consider the selection of a set of features as a search problem. Example: Recursive Feature Elimination
3. ***Embedded:*** Embedded methods use algorithms that have built-in feature selection methods. For instance, Lasso and RF have their own feature selection methods.
4. **Hybrid approach :** Both filter and wrapper-based methods are used in hybrid approach. This approach first selects the possible optimal feature set which is further tested by the wrapper approach. It hence uses the advantages of both filter and wrapper-based approach.

3 Feature selection techniques that are most commonly used and also gives good results-

1. Univariate Selection
2. Feature Importance
3. Correlation Matrix with Heatmap

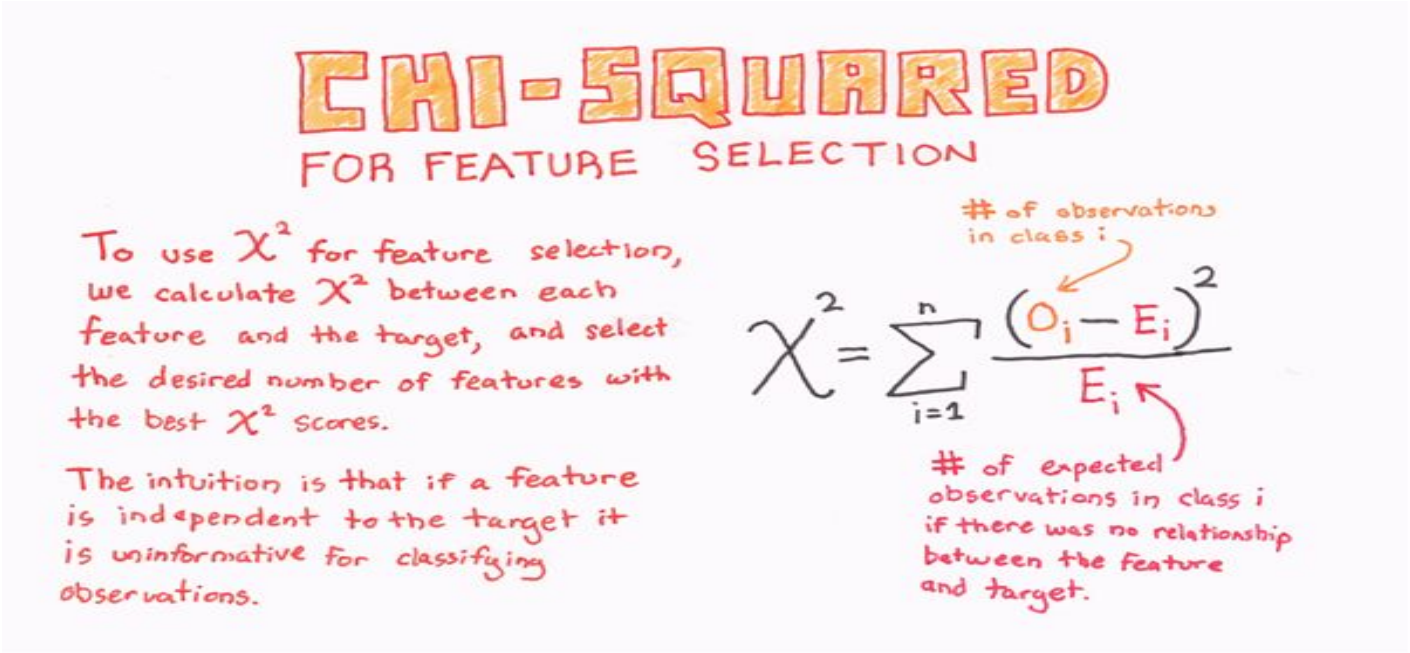
Univariate Selection

- Statistical tests(eg. chi-squared (chi^2)) can be used to select those features that have the strongest relationship with the output variable.

- The scikit-learn library provides the SelectKBest class that can be used with a suite of different statistical tests to select a specific number of features.

Chi-squared (chi²) test

Mathematically, a Chi-Square test is done on two distributions two determine the level of similarity of their respective variances. In its **null hypothesis**, it assumes that the given distributions are independent. The higher the value of , the more dependent the output label is on the feature and higher the importance the feature has on determining the output.



Let the feature in question have m attribute values and the output have k class labels. A contingency table(m*k) shows the distribution of one variable in rows and another in columns. It is used to study the relation between two variables

Exited\ Gender	Yes	No	Total
Male	38	178	216
Female	44	140	184
Total	82	318	400

Exited\Gender	Yes	No
Male	44	172
Female	38	146

Gender,Exited	O	E	O-E	Square of O-E	(Square of O-E) / E
Male,Yes	38	44	-6	36	0.818181818
Male,No	178	172	6	36	0.209302326
Female,Yes	44	38	6	36	0.947368421
Femal,No	140	146	-6	36	0.246575342
Chi Square Value					2.221427907

Now, the parameters for similarity of features are classified based on two factors –

1. ***The Similarity of information contributed by the features*** **:**

CORRELATION- The features are classified as associated or similar mostly based on their correlation factor. In the data set, we have many features which are correlated. Now the problem with having correlated features is that, if f1 and f2 are two correlated features of a data set, then the classifying or regression model including both f1 and f2 will give the same as the predictive model compared to the scenario where either f1 or f2 was included in the dataset. This is because both f1 and f2 are correlated and hence they contribute the same information regarding the model in the data set. There are various methods to calculate the correlation factor, however, Pearson's correlation coefficient is most widely used.

CORR(X,Y): cov(X, Y)/ sigma(X)* sigma(Y)

Where,

cov(X, Y) - covariance = $\mathrm{Cov}\,(X, Y) = \frac{\sum(X_i - \overline{X})(Y_j - \overline{Y})}{n}$

sigma(X) - standard deviation of X

sigma(Y) - standard deviation of Y

2. *Quantum of information contributed by the features*

ENTROPY- Entropy is the measure of the average information content. The higher the entropy, the higher is the information contribution by that feature. Entropy (H) can be formulated as:

$$H(x) = E(I(X)) = E(-\ln(P(X)))$$

Where,

X - discrete random variable X

P(X) - probability mass function

E - expected value operator,

I - information content of X.

I(X) - a random variable.

In Data Science, entropy of a feature f1 is calculated by excluding feature f1 and then calculating the entropy of the rest of the features. Now, the lower the entropy value (excluding f1) the higher will be the information content of f1. In this manner the entropy of all the features is calculated. At the end, either a threshold value or further relevancy check determines the optimality of the features on the basis of which features are selected. Entropy is mostly used for Unsupervised Learning as we do have a class field in the dataset and hence entropy of the features can give substantial information.

The Mathematical formula for Entropy is as follows - $$E(S) = \sum_{i=1}^{c} - p_i \log_2 p_i$$

'Pi'=frequentist probability of an element/class 'i' in our data. For a decision tree that uses Information Gain, the algorithm chooses the attribute that provides the *greatest* Information Gain (this is also the attribute that causes the greatest *reduction* in entropy)

Credit Rating	Liability		
	Normal	High	Total
Excellent	3	1	4
Good	4	2	6
Poor	0	4	4
Total	7	7	14

$$E(Liability) = -\frac{7}{14}\log_2\left(\frac{7}{14}\right) - \frac{7}{14}\log_2\left(\frac{7}{14}\right)$$

$$= -\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right)$$

$$= 1$$

$$E(Liability \mid CR = Excellent) = -\frac{3}{4}\log_2\left(\frac{3}{4}\right) - \frac{1}{4}\log_2\left(\frac{1}{4}\right) \approx 0.811$$

$$E(Liability \mid CR = Good) = -\frac{4}{6}\log_2\left(\frac{4}{6}\right) - \frac{2}{6}\log_2\left(\frac{2}{6}\right) \approx 0.918$$

$$E(Liability \mid CR = Poor) = -0\log_2(0) - \frac{4}{4}\log_2\left(\frac{4}{4}\right) = 0$$

Weighted Average:

$$E(Liability \mid CR) = \frac{4}{14} \times 0.811 + \frac{6}{14} \times 0.918 + \frac{4}{14} \times 0$$

$$= 0.625$$

Information Gain:

$$IG(Liability, CR) = E(Liability) - E(Liability \mid CR)$$

$$= 1 - 0.625$$

$$= 0.375$$

Feature Importance

- Feature importance gives you a score for each feature of your data, the higher the score more important or relevant is the feature towards your output variable.
- Feature importance is an inbuilt class that comes with Tree Based Classifiers.

❖ Extra Tree Classifier for Feature Selection

Extremely Randomized Trees Classifier(Extra Trees Classifier) is a type of ensemble learning technique which aggregates the results of multiple de-correlated decision trees collected in a “forest” to output it’s classification result

Each Decision Tree in the Extra Trees Forest is constructed from the original training sample. Then, at each test node, Each tree is provided with a random sample of k features from the feature-set from which each decision tree must select the best feature to split the data based on some mathematical criteria (typically the Gini Index). This random sample of features leads to the creation of multiple de-correlated decision trees.

To perform feature selection using the above forest structure, during the construction of the forest, for each feature, the normalized total reduction in the mathematical criteria used in the decision of feature of split (Gini Index if the Gini Index is used in the construction of the forest) is computed. This value is called the Gini Importance of the feature. To perform feature selection, each feature is ordered in descending order according to the Gini Importance of each feature and the user selects the top k features according to his/her choice.

Consider the following data:-

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Let us build a hypothetical Extra Trees Forest for the above data with **five decision trees** and the value of k which decides the number of features in a random sample of features be **two**. Here the decision criteria used will be Information Gain.
For the given data, the **entropy** is:- $$Entropy(S) = -\frac{9}{14}log(\frac{9}{14}) - \frac{5}{14}log(\frac{5}{14})$$
$$\Rightarrow Entropy(S) = 0.940$$

Note that the formula for Information Gain is:- $$Gain(S, A) = Entropy(S) - \sum_{v \epsilon Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

1st Decision Tree gets data with the features Outlook and Temperature:

$$Gain(S, Outlook) = 0.940 - (\frac{5}{14}(\frac{-2}{5}log_2(\frac{2}{5}) + \frac{-3}{5}log_2(\frac{3}{5})) + \frac{4}{14}(\frac{-4}{4}log_2(\frac{4}{4}) + \frac{-0}{4}log_2(\frac{0}{4})) + \frac{5}{14}(\frac{-3}{5}log_2(\frac{3}{5}) + \frac{-2}{5}log_2(\frac{2}{5})))$$
$$\Rightarrow Gain(S, Outlook) = 0.246$$
$$Gain(S, Temperature) = 0.029$$

2nd Decision Tree gets data with the features Temperature and Wind:

$$Gain(S, Wind) = 0.048 \quad Gain(S, Temperature) = 0.029$$

3rd Decision Tree gets data with the features Outlook and Humidity:

$$Gain(S, Outlook) = 0.246 \qquad Gain(S, Humidity) = 0.151$$

4th Decision Tree gets data with the features Temperature and Humidity:

$$Gain(S, Temperature) = 0.029 \quad Gain(S, Humidity) = 0.151$$

5th Decision Tree gets data with the features Wind and Humidity:

$$Gain(S, Wind) = 0.048 \qquad Gain(S, Humidity) = 0.151$$

Computing total Info Gain for each feature:-

Total Info Gain for Outlook = 0.246+0.246 = 0.492

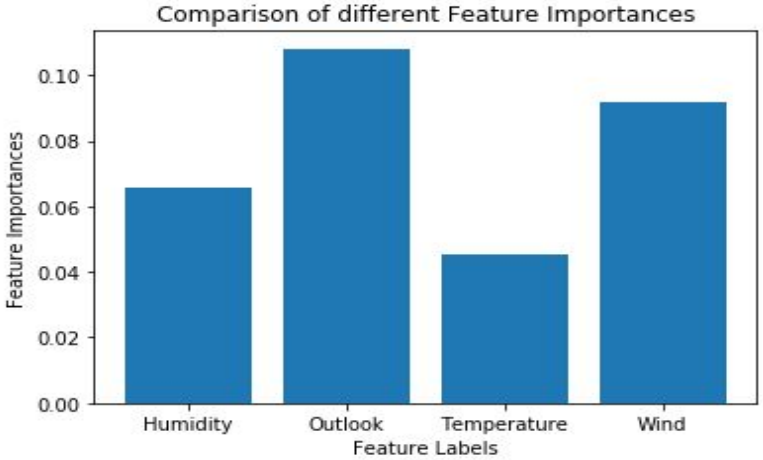
Total Info Gain for Temperature = 0.029+0.029+0.029 = 0.087

Total Info Gain for Humidity = 0.151+0.151+0.151 = 0.453

Total Info Gain for Wind = 0.048+0.048 = 0.096

Thus the most important variable to determine the output label according to the above constructed Extra Trees Forest is the feature "**Outlook**".

Using ExtraTreesClassifier(n_estimators = 5,criterion ='entropy', max_features = 2):



Gini index

$$GiniIndex = 1 - \sum_j p_j^2$$

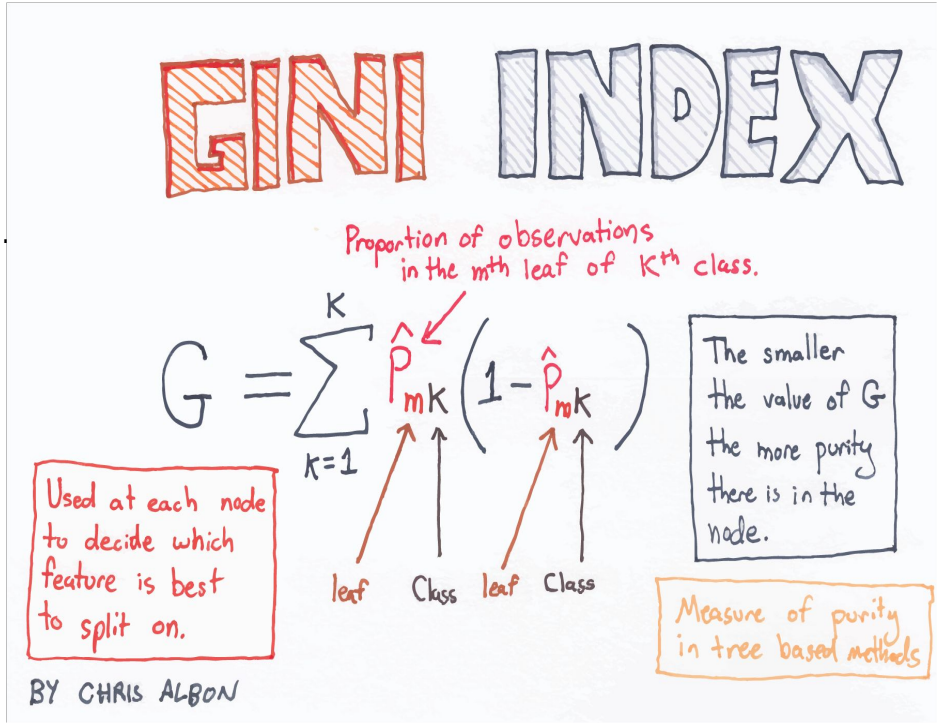
1.compute the gini index for data-set
2.for every attribute/feature:

1..calculate gini index for all categorical value.

2.take average information entropy for the current attribute.

3. pick the best gini gain attribute.

4. Repeat until we get the tree we desired.



https://www.geeksforgeeks.org/decision-tree-introduction-example/

Correlation Matrix with Heatmap

Correlation states how the features are related to each other or the target variable.

CORR(X,Y): cov(X, Y)/ sigma(X)* sigma(Y)

	A	B	C
0	45	38	10
1	37	31	15
2	42	26	17
3	35	28	21
4	39	33	12

```
corrMatrix = df.corr()

sn.heatmap(corrMatrix, annot=True)

plt.show()
```

Curse of Dimensionality

Potential issues that arise when your data has a huge number of dimensions:

1. If we have more features than observations than **we run the risk of massively overfitting our model — this would generally result in terrible out of sample performance.**

2. When we have too many features, observations become harder to cluster — believe it or not, **too many dimensions causes every observation in your dataset to appear equidistant from all the others.** And because clustering uses a distance measure such as Euclidean distance to quantify the similarity between observations, this is a big problem. **If the distances are all approximately equal, then all the observations appear equally alike (as well as equally different), and no meaningful clusters can be formed.**

In machine learning classification problems, there are often too many factors on the basis of which the final classification is done. The higher the number of features, the harder it gets to visualize the training set and then work on it. Sometimes, most of these features are correlated, and hence redundant. This is where dimensionality reduction algorithms come into play. Dimensionality reduction is the process of reducing the number of random variables under consideration, by obtaining a set of principal variables. It can be divided into feature selection and feature extraction.

- **Feature selection:** In this, we try to find a subset of the original set of variables, or features, to get a smaller subset which can be used to model the problem. It usually involves three ways:
  1. Filter
  2. Wrapper
  3. Embedded
- **Feature extraction:** This reduces the data in a high dimensional space to a lower dimension space, i.e. a space with lesser no. of dimensions

Other various methods used for dimensionality reduction include:

- Principal Component Analysis (PCA)
- Linear Discriminant Analysis (LDA)
- Generalized Discriminant Analysis (GDA)

Principal Component Analysis (PCA)

It works on a condition that while the data in a higher dimensional space is mapped to data in a lower dimension space, **the variance of the data in the lower dimensional space should be maximum**.

PCA is an **unsupervised learning algorithm** used for data preprocessing. In real world,the training instances are not generally spread uniformly across all the dimensions. Most of the instances actually lie close to each other. Hence ideally we should try to decrease the dimension of our dataset.This can be done by projecting our training set onto a lower-dimensional hyperplane. And we should try to project it such a way that most of the information of our data is contained by our algorithm.

It involves the following steps:

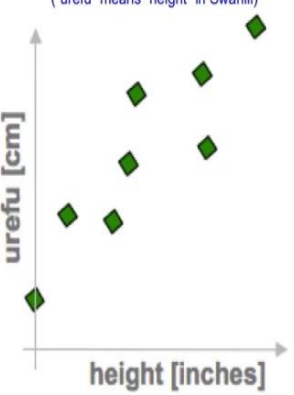
- Construct the **covariance matrix** of the data.
- Compute the **eigenvectors of this matrix**.
- **Eigenvectors corresponding to the largest eigenvalues** are used to reconstruct a large fraction of variance of the original data.

Hence, we are left with a lesser number of eigenvectors, and there might have been some data loss in the process. But, the most important variances should be retained by the remaining eigenvectors.

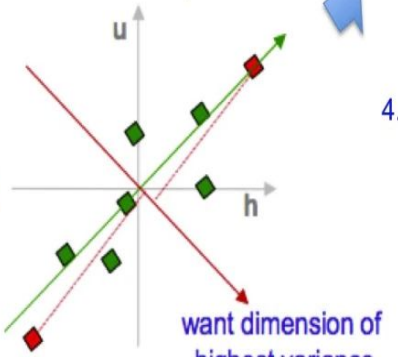
PCA in a nutshell

1. correlated hi-d data

("urefu" means "height" in Swahili)



2. center the points



3. compute covariance matrix

$$\begin{array}{c} \\ h \\ u \end{array}\begin{array}{c} \begin{array}{cc} h & u \end{array} \\ \begin{pmatrix} 2.0 & 0.8 \\ 0.8 & 0.6 \end{pmatrix} \end{array} \rightarrow \mathrm{cov}(h,u) = \frac{1}{n}\sum_{i=1}^{n} h_i u_i$$

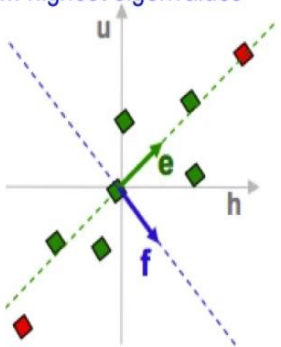
4. eigenvectors + eigenvalues

$$\begin{pmatrix} 2.0 & 0.8 \\ 0.8 & 0.6 \end{pmatrix}\begin{pmatrix} e_h \\ e_u \end{pmatrix} = \lambda_e \begin{pmatrix} e_h \\ e_u \end{pmatrix}$$

$$\begin{pmatrix} 2.0 & 0.8 \\ 0.8 & 0.6 \end{pmatrix}\begin{pmatrix} f_h \\ f_u \end{pmatrix} = \lambda_f \begin{pmatrix} f_h \\ f_u \end{pmatrix}$$

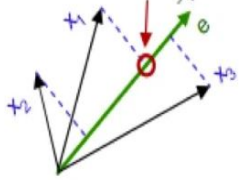
```
eig(cov(data))
```

5. pick m<d eigenvectors w. highest eigenvalues

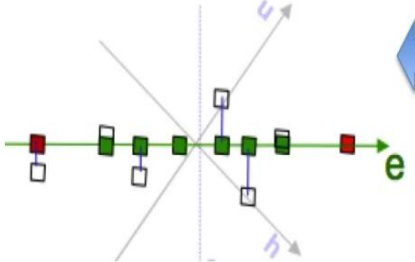


6. project data points to those eigenvectors

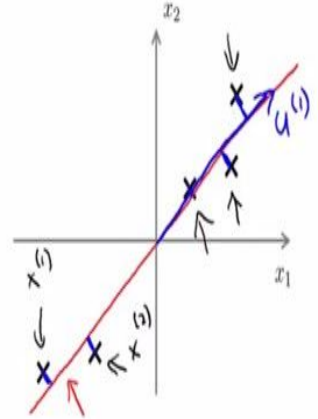
$$x_e^i = x^i e = \sum_{j=1}^{a} x_{ij} e_j$$



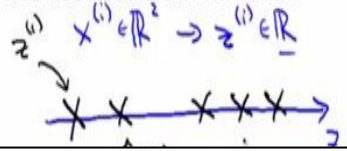
7. uncorrelated low-d data

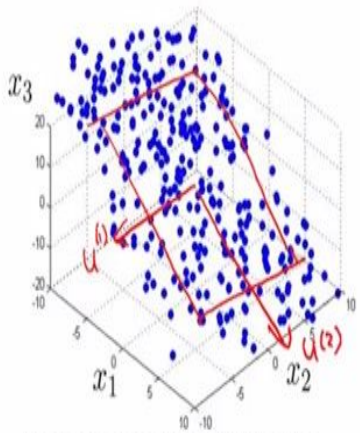


Principal Component Analysis (PCA) algorithm



Reduce data from 2D to 1D





Reduce data from 3D to 2D

Advantages of Dimensionality Reduction

- It helps in data compression, and hence reduced storage space.
- It reduces computation time.
- It also helps remove redundant features, if any.

Disadvantages of Dimensionality Reduction

- It may lead to some amount of data loss.
- PCA tends to find linear correlations between variables, which is sometimes undesirable.
- PCA fails in cases where mean and covariance are not enough to define datasets.
- We may not know how many principal components to keep- in practice, some thumb rules are applied

Classification

- Find how much important each feature is determining whether a CV is getting shortlisted or not.
- Accordingly, assign score to those features.
- Use these scores to evaluate the new CVs.

Feature selection- https://www.geeksforgeeks.org/parameters-feature-selection/?ref=rp

https://towardsdatascience.com/the-5-feature-selection-algorithms-every-data-scientist-need-to-know-3a6b566efd2

https://towardsdatascience.com/feature-selection-using-random-forest-26d7b747597f (using Random Forrest)

Chi-squared test-
https://www.geeksforgeeks.org/chi-square-test-for-feature-selection-mathematical-explanation/?ref=rp

https://towardsdatascience.com/chi-square-test-for-feature-selection-in-machine-learning-206b1f0b8223

Gini index-
https://medium.com/@riyapatadiya/gini-index-cart-decision-algorithm-in-machine-learning-1a4ed5d6140d

Entropy- https://towardsdatascience.com/entropy-how-decision-trees-make-decisions-2946b9c18c8

Decision tress-
https://medium.com/deep-math-machine-learning-ai/chapter-4-decision-trees-algorithms-b93975f7a1f1

Extra-tree classifier-

https://www.geeksforgeeks.org/ml-extra-tree-classifier-for-feature-selection/#:~:text=Extremely%20Randomized%20Trees%20Classifier(Extra,to%20output%20it's%20classification%20result.

Dimensionality reduction- https://www.geeksforgeeks.org/dimensionality-reduction/