Lecture 4 Exploratory Data Analysis -- EDA

More Concepts in DS

Tamer Çetin

Learning Outcomes

- Understand the importance and uses of EDA
- Understand distributions, visualize them, and describe the most important features
- Identify situations when extreme values matter and make decisions about extreme values
- Produce graphs and tables of presentation that are focused, informative, and easy to read
- Know the main features of the most important theoretical distributions and assess whether they are good approximations of distributions of variables in actual data

Motivation

- Learning common concepts in DS

Feature Selection and Extraction

- Big data –-> hundreds/thousands of features –-> overfitting
- To avoid overfitting, regularization and/or dimensionality reduction methods
- Feature selection aims to find best features that allow use to build optimized models.
- Feature extraction aims to reduce the number of features in a dataset by creating new features from the existing ones (and then discarding the original features)
- Using feature extraction/selection, we can reduce the curse of dimensionality in addition Accuracy improvements, Overfitting risk reduction, Speed up in training, Improved data visualization, and Increase in predictive power of our model

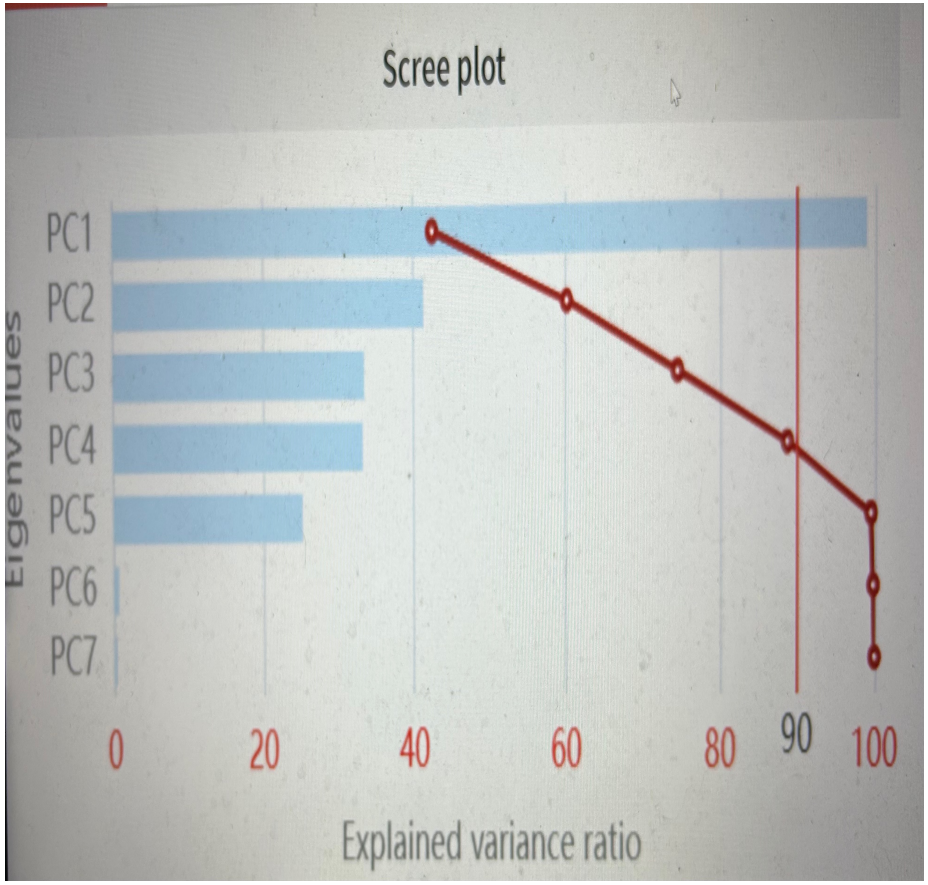
Principal Component Analysis (PCA)

- Powerful and practical DS technique used in ***high-dimensional datasets***
  - Data exploration
  - Visualization
  - ML
- The aim is to
  - Reduce the curse of dimensionality
  - While keeping variation maximum
- It transforms the original variables into a new set of linearly uncorrelated variables called principal components, orthogonality.

PCA

- PCA is a method that brings together:
- A measure of how each variable is associated with one another. (Covariance matrix.)
- The directions in which our data are dispersed. (Eigenvectors.)
- The relative importance of these different directions. (Eigenvalues.)
- PCA combines our predictors and allows us to drop the eigenvectors that are relatively unimportant.

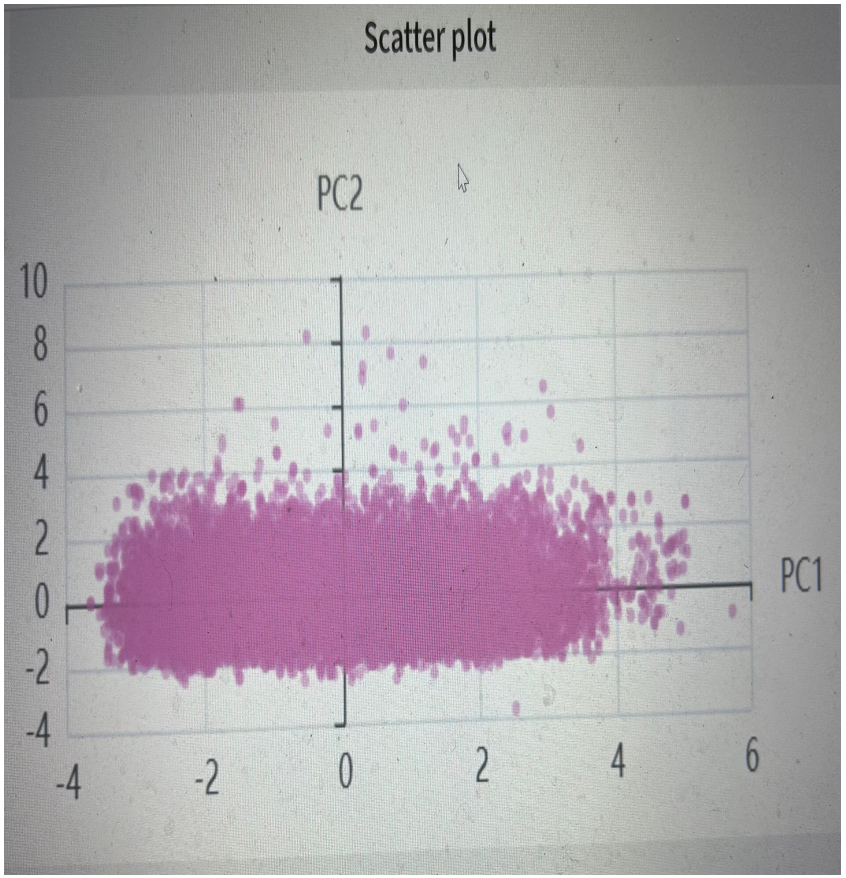
PCA



Scree Plot for PCA

- A scree plot is a popular graphical method for determining the optimal number of components to keep in PCA.
- It's a simple line segment plot that shows the fraction of total variance in the data as explained or represented by each PC.
- The x-axis of a scree plot indicates the eigenvalue (amount of variance explained), and the y-axis lists the number of each principal component, in order.
- The ratio of eigenvalues (also known as explained variance ratio) is a measure of the proportion of the total variance in data captured by each principal component.

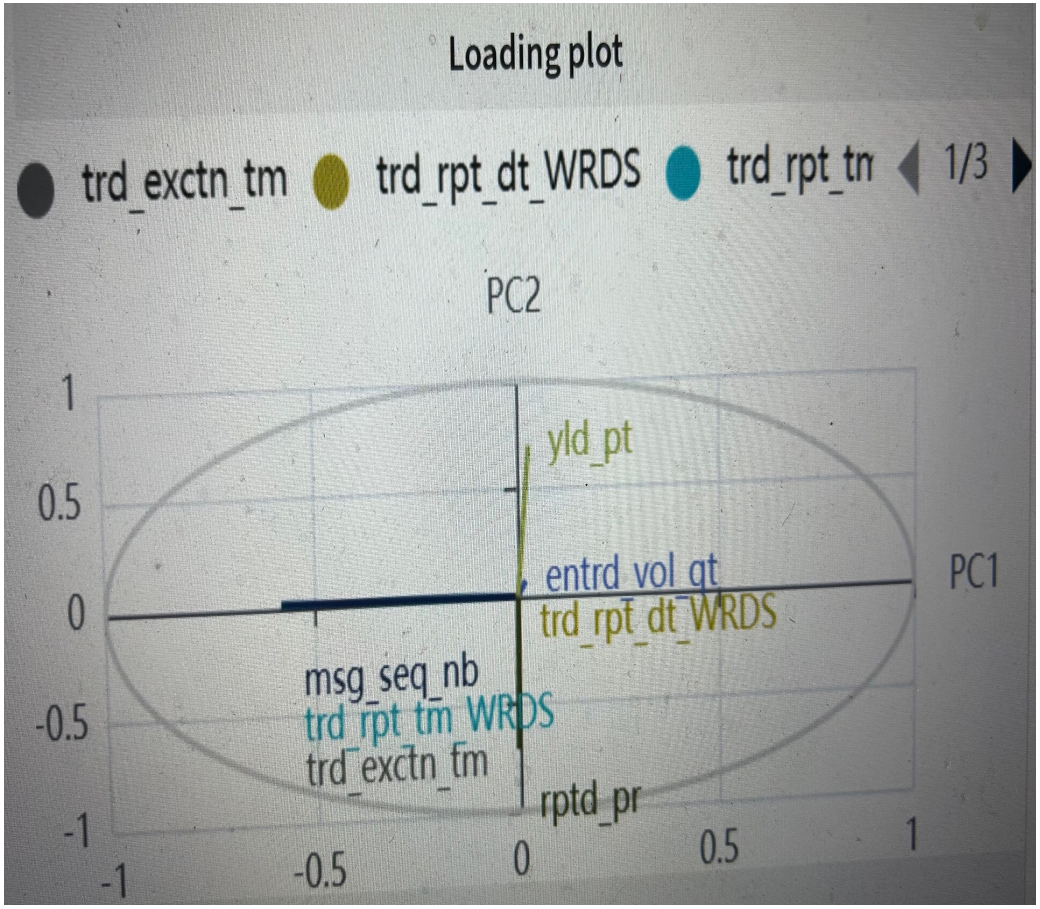
PCA



Scatter Plot for PC1 and PC2

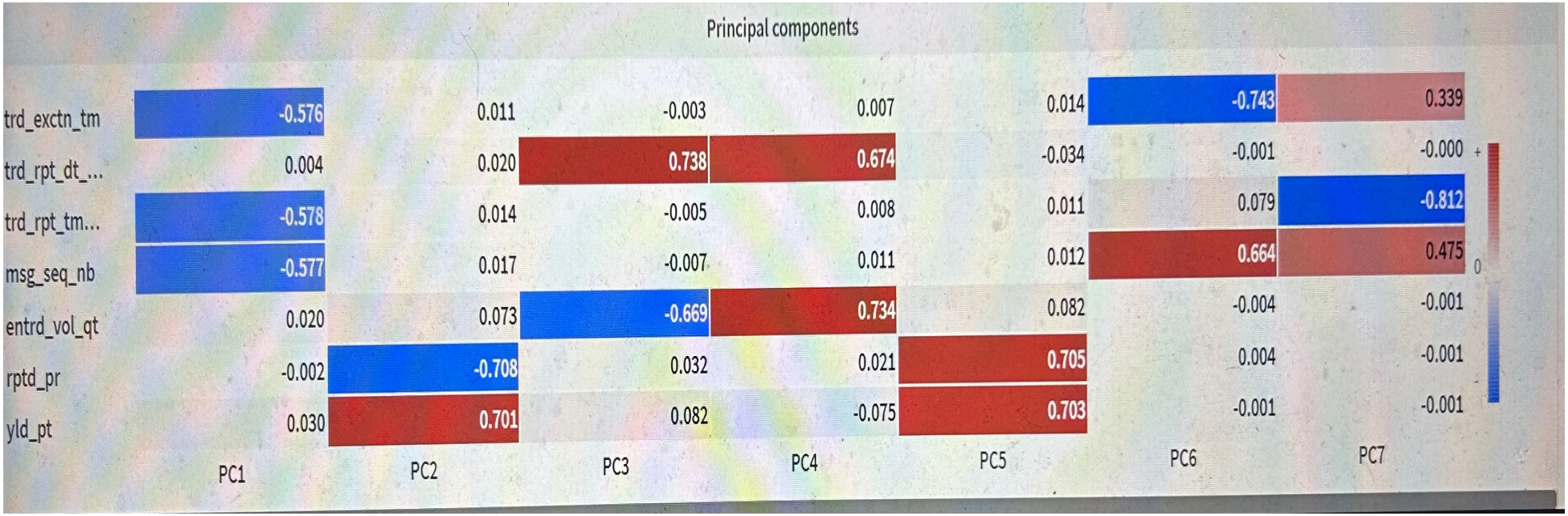
- Each point represents a single sample (row) from your dataset.
- The x-coordinate of each point is the value of PC1 for that sample, and the y-coordinate is the value of PC2 for that sample.
- Similar samples will cluster together.
- The distances between points reflect their dissimilarities.
- The axes of a PCA scatter plot don't represent individual features, but combinations of features.
- The first principal component (PC1) captures the direction of the highest variance in the data, and the second principal component (PC2) captures the second highest variance, orthogonal to the direction of the first.

PCA



Loading plot is a way to visualize how much each variable contributes to each principal component.
In the plot, each variable is represented as a vector.
1.Direction: The direction of the vectors (positive or negative) represents the correlation between the principal component and the original variables.
Magnitude: The magnitude (length) of the vectors represents the importance of a variable to the principal component. Larger absolute values (regardless of whether they are positive or negative) indicate that the variable has a strong contribution to the principal component.

PCA


Principal components
trd_exctn_tm
trd_rpt_dt_...
trd_rpt_tm...
msg_seq_nb
entrd_vol_qt
rptd_pr
yld_pt
-0.576
0.004
-0.578
-0.577
0.020
-0.002
0.030
0.011
0.020
0.014
0.017
0.073
-0.708
0.701
-0.003
0.738
-0.005
-0.007
-0.669
0.032
0.082
0.007
0.674
0.008
0.011
0.734
0.021
-0.075
0.014
-0.034
0.011
0.012
0.082
0.705
0.703
-0.743
-0.001
0.079
0.664
-0.004
0.004
-0.001
0.339
-0.000
-0.812
0.475
-0.001
-0.001
-0.001
+
0
-
PC1
PC2
PC3
PC4
PC5
PC6
PC7

PCA: Pros

- Reduces the Curse of Dimensionality
- Removes the unwanted noise present in the dataset and
- Preserves the signal required

PCA: Cons

1.Loss of Interpretability: After performing PCA, the original features (columns) are transformed into a set of new principal components, which are linear combinations of the original features.
2.Assumes Linear Relationships: PCA assumes that the principal components are a linear combination of the original features.
3.Variance vs. Relevance: PCA selects components based on the variance explained, not the relevance to the output variable.
A component that explains a lot of variance might not necessarily be important for prediction.
4. All Components are Orthogonal: PCA imposes orthogonality between the principal components, which may not always be the most meaningful representation of the data.

PCA: Cons

5.Data Scaling: PCA is sensitive to the scale of the features. If features are measured in different units or have different ranges, PCA might bias towards high variance features. To overcome this, standardization or normalization is typically performed before applying PCA, but this is an additional preprocessing step that needs to be remembered.

6. Data Structure: PCA assumes that the data points are continuous and lie in a Euclidean space. It may not work well with other types of data, like categorical or binary data.

7. Outliers: PCA is sensitive to outliers, which can significantly influence the direction of the principal components.

8. Loss of Information: Although the goal of PCA is to retain most of the variance in data, some information is lost because it reduces dimensions.

Bias-Variance Tradeoff

- To understand bias and variance in ML, we need to understand errors in ML first.
- Error in ML is used to account for the accuracy of prediction model.
- There are two main types of errors present in any ML model:
  Reducible Errors and Irreducible Errors.
- Irreducible errors are errors which will always be present in a machine learning model, because of unknown variables, and whose values cannot be reduced.
- Reducible errors are those errors whose values can be further reduced to improve a model.
- They are caused because our model's output function does not match the desired output function and can be optimized.
- The main reasons for reducible errors in ML models are bias and variance.

Bias-Variance Tradeoff

- Keep in mind, ML models analyze data and find patterns of association among data points to make prediction.
- The goal is to make generalizations about certain events/measurements in data using those patterns in the training set.
- After learning those patterns and/or generalizations based on those patterns, ML model applies that trained information to the test set to predict those events.

    ***Basically, algorithm learns a model from training data and uses that model to predict the outcome of unseen data.

    ***The goal is to make the model generalize well to unseen data.
- When there is bias/variance, which is inevitable, there will be less predictive power in ML model.

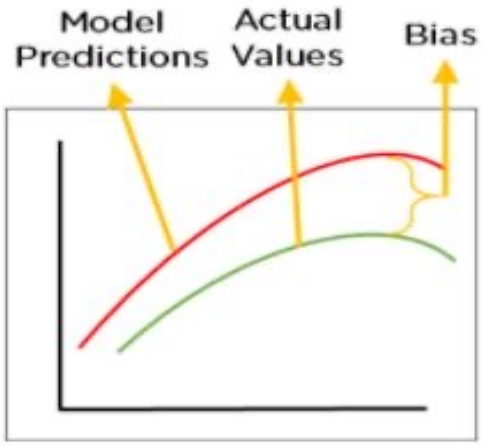
Bias-Variance Tradeoff

- So, the bias-variance tradeoff is a fundamental concept in ML models that helps us understand the model accuracy using the relationship between model complexity and model performance, especially in relation to overfitting and underfitting.
- Bias is the simple assumptions that our model makes about our data to be able to predict new data.
- In statistical terms, bias is indeed the difference between the expected (or average) prediction of our model and the correct value which we are trying to predict.
- More specifically, bias measures how far off in general ML predictions are from the actual value.
- In the machine learning context, bias also refers to the assumptions made by a model about the underlying data.
- A model with high bias pays very little attention to the training data and oversimplifies the model.
- It always leads to high error on training and test data.

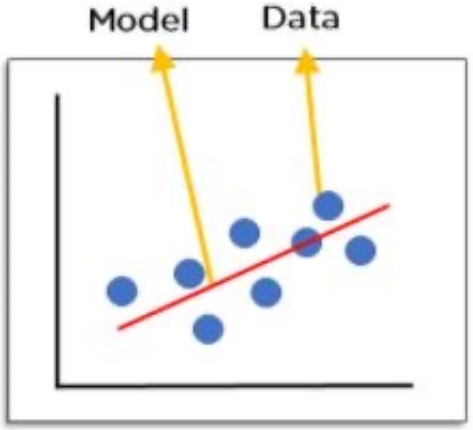
Bias-Variance Tradeoff

- When ML model is too simple (high bias), it may perform poorly because it cannot capture the complexity of data.

Bias



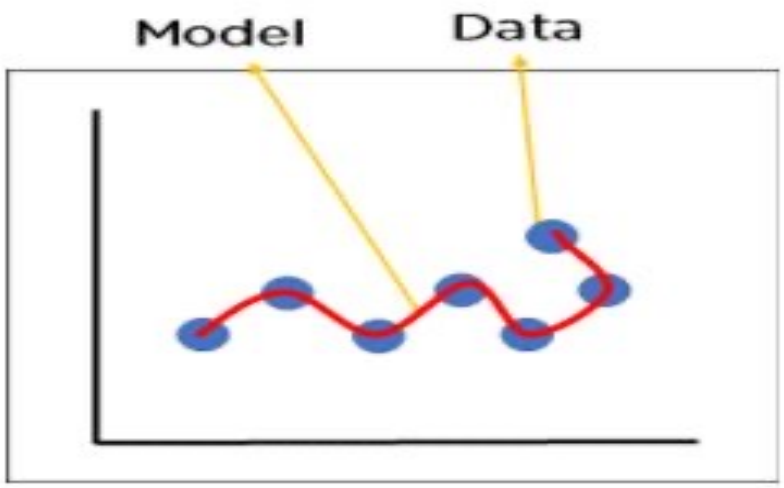
Underfitting



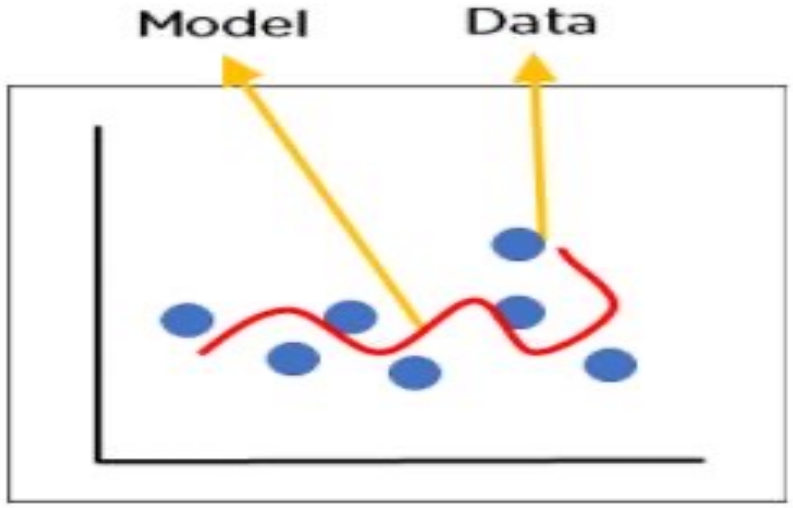
- When bias is high, assumptions made by our model are too basic.
- The model cannot capture the important features of our data or patterns in training set.
- It will not perform well on testing set and/or new data.
- The line of best fit is a straight line that does not pass through many of data points (underfitting).

Bias-Variance Tradeoff

- When ML model is too complex (high variance), it may perform poorly because it overfits the training data and fails to generalize to new, unseen data.



Overfitting on training data

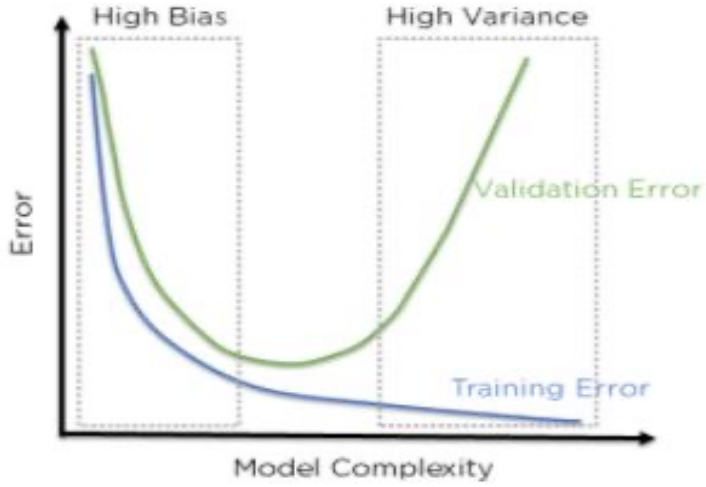


Overfitting on test data

- Variance is the model's sensitivity to fluctuations in data.
- When there is noise in data model can learn from noise and thus, it will perform well on testing data and get high accuracy but will fail to perform on unseen data (overfitting).

Bias-Variance Tradeoff

What is the reason for the trade-off between bias and variance?
Note that if the model is too simple it will perform poorly since it will not learn enough about data (high bias).
We need more complex models to reduce bias.
So using more complex models or minimizing bias will lead to the trade off because bias will decrease but variance will increase.

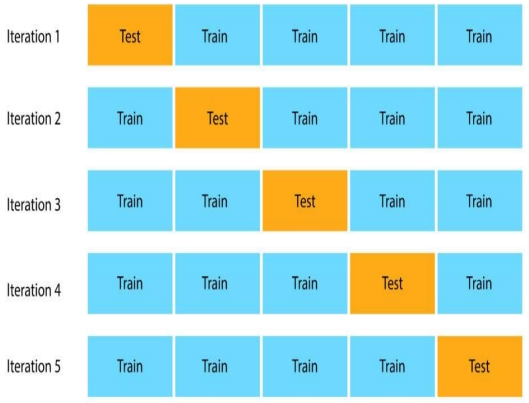


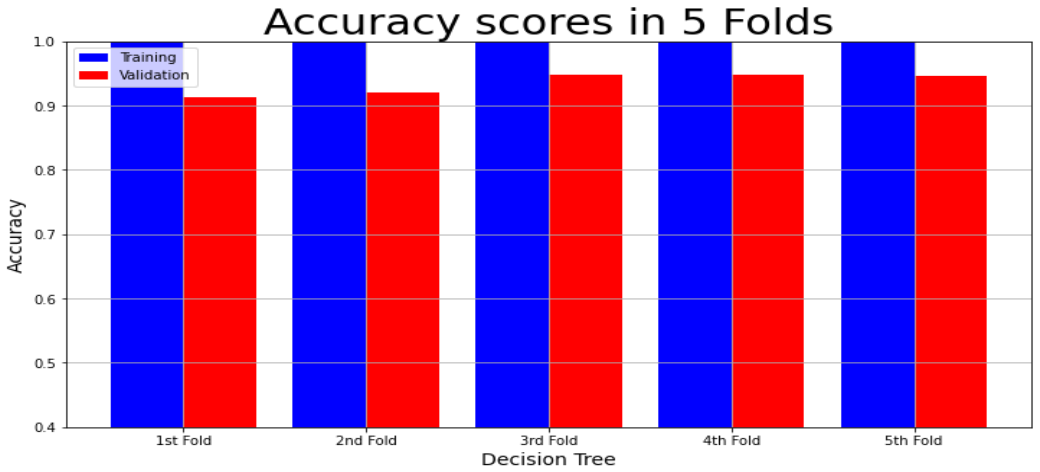
Bias-Variance Tradeoff

How to handle bias-variance tradeoff

1.Cross-validation:

- Use techniques like k-fold cross-validation to estimate the error rate of your model on unseen data.
- Divide your dataset into 'k' subsets and train and test your model 'k' times, each time using a different subset as your test set.
- Identify whether your model is overfitting or underfitting the data.





Bias-Variance Tradeoff

How to handle bias-variance tradeoff

2. Regularization: Regularization adds a penalty on the different parameters of the model to reduce the freedom of the model and in turn reduce overfitting.
L1 (Lasso) and L2 (Ridge) are two common types of regularization.

3. Ensemble Methods: Ensemble methods, like bagging or boosting, combine the predictions of several models in order to reduce the variance, and in some cases, also the bias.

4. Dimensionality Reduction: Reduce the number of input variables (dimensionality) in your model.
More input variables can make a model more complex, increasing the risk of overfitting (high variance).
Techniques like Principal Component Analysis (PCA) can be useful here.

Bias-Variance Tradeoff

How to handle bias-variance tradeoff

5. Model Selection: Use simpler models if you're suffering from high variance.

Use more complex models if you're suffering from high bias.

6. Adding more data: If possible, adding more data can help the algorithms detect the signal better.

However, it does not always work and is problem-dependent.

7. Early stopping: While training the machine learning model, you can measure how well each iteration of the model performs.

Up to a point, adding more iterations will improve the accuracy of the model.

After that point, the model's accuracy starts to decrease: this is when the model starts to overfit.

Stop here!