Universidad Politécnica De Yucatán

Machine Learning

Solution to most common problems in ML - Portfolio evidence

- Esteban Rodríguez Kumul

Date: 15/09/2023

Overfitting and Underfitting in Machine Learning

Overfitting and Underfitting are the two main problems that occur in machine learning and degrade the performance of the machine learning models.

The main goal of each machine learning model is to generalize well. Here generalization defines the ability of an ML model to provide a suitable output by adapting the given set of unknown input. It means after providing training on the dataset, it can produce reliable and accurate output. Hence, the underfitting and overfitting are the two terms that need to be checked for the performance of the model and whether the model is generalizing well or not.

Before understanding the overfitting and underfitting, let's understand some basic term that will help to understand this topic well:

- *Signal:* It refers to the true underlying pattern of the data that helps the machine learning model to learn from the data.
- *Noise:* Noise is unnecessary and irrelevant data that reduces the performance of the model.
- *Bias:* Bias is a prediction error that is introduced in the model due to oversimplifying the machine learning algorithms. Or it is the difference between the predicted values and the actual values.
- *Variance:* If the machine learning model performs well with the training dataset, but does not perform well with the test dataset, then variance occurs.

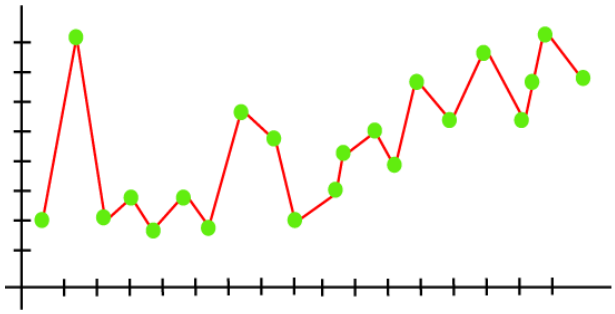
Overfitting

Overfitting occurs when our machine learning model tries to cover all the data points or more than the required data points present in the given dataset. Because of this, the model starts caching noise and inaccurate values present in the dataset, and all these factors reduce the efficiency and accuracy of the model. The overfitted model has low bias and high variance.

The chances of occurrence of overfitting increase as much we provide training to our model. It means the more we train our model, the more chances of occurring the overfitted model.

Overfitting is the main problem that occurs in supervised learning.

Example: The concept of the overfitting can be understood by the below graph of the linear regression output:



As we can see from the above graph, the model tries to cover all the data points present in the scatter plot. It may look efficient, but in reality, it is not so. Because the goal of the regression model to find the best fit line, but here we have not got any best fit, so, it will generate the prediction errors.

How to avoid the Overfitting in Model

Both overfitting and underfitting cause the degraded performance of the machine learning model. But the main cause is overfitting, so there are some ways by which we can reduce the occurrence of overfitting in our model.

- *Cross-Validation*
- *Training with more data*
- *Removing features*
- *Early stopping the training*
- *Regularization*
- *Ensembling*

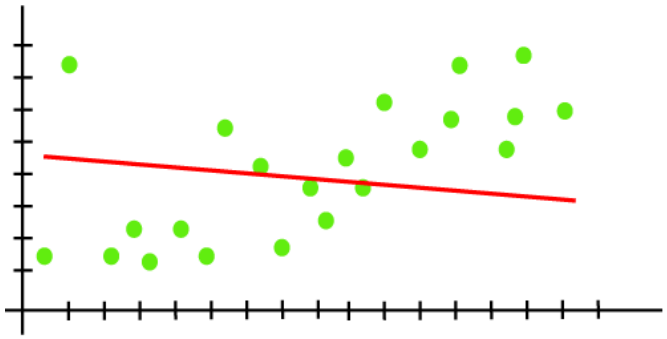
Underfitting

Underfitting occurs when our machine learning model is not able to capture the underlying trend of the data. To avoid the overfitting in the model, the fed of training data can be stopped at an early stage, due to which the model may not learn enough from the training data. As a result, it may fail to find the best fit of the dominant trend in the data.

In the case of underfitting, the model is not able to learn enough from the training data, and hence it reduces the accuracy and produces unreliable predictions.

An underfitted model has high bias and low variance.

Example: We can understand the underfitting using below output of the linear regression model:



As we can see from the above diagram, the model is unable to capture the data points present in the plot.

How to avoid underfitting:

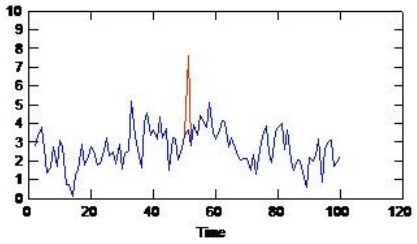
- By increasing the training time of the model.
- By increasing the number of features.

Outliers

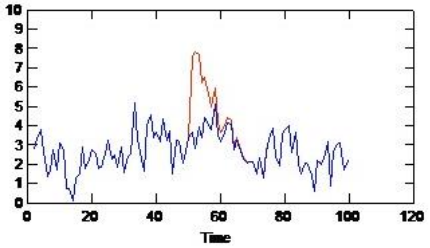
Shifts in the level of a time series that cannot be explained are referred to as outliers. These observations are inconsistent with the remainder of the series and can dramatically influence the analysis and, consequently, affect the forecasting ability of the time series model.

The following figure displays several types of outliers commonly occurring in time series. The blue lines represent a series without outliers. The red lines suggest a pattern that might be present if the series contained outliers. These outliers are all classified as deterministic because they affect only the mean level of the series.

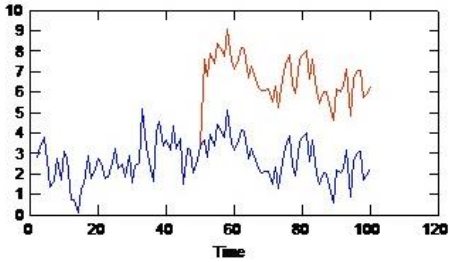
- *Additive Outlier.* An additive outlier appears as a surprisingly large or small value occurring for a single observation. Subsequent observations are unaffected by an additive outlier. Consecutive additive outliers are typically referred to as additive outlier patches.



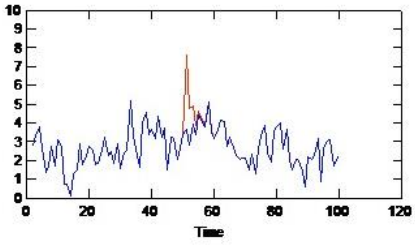
- *Innovational Outlier.* An innovational outlier is characterized by an initial impact with effects lingering over subsequent observations. The influence of the outliers may increase as time proceeds.



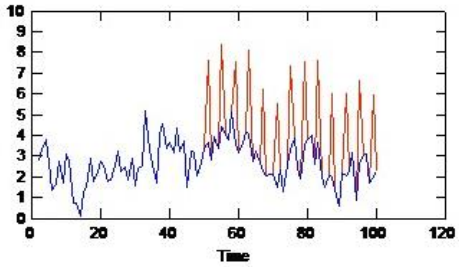
- *Level Shift Outlier.* For a level shift, all observations appearing after the outlier move to a new level. In contrast to additive outliers, a level shift outlier affects many observations and has a permanent effect.



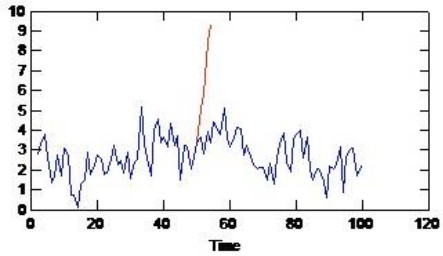
- *Transient Change Outlier.* Transient change outliers are similar to level shift outliers, but the effect of the outlier diminishes exponentially over the subsequent observations. Eventually, the series returns to its normal level.



- *Seasonal Additive Outlier.* A seasonal additive outlier appears as a surprisingly large or small value occurring repeatedly at regular intervals.



- *Local Trend Outlier.* A local trend outlier yields a general drift in the series caused by a pattern in the outliers after the onset of the initial outlier.



Outlier detection in time series involves determining the location, type, and magnitude of any outliers present. Tsay (1988) proposed an iterative procedure for detecting mean level change to identify deterministic outliers. This process involves comparing a time series model that assumes no outliers are present to another model that incorporates outliers. Differences between the models yield estimates of the effect of treating any given point as an outlier.

Solutions for overfitting, underfitting and presence of outliers in datasets.

Overfitting, underfitting, and the presence of outliers are common challenges in machine learning and data analysis. Each of these issues requires specific strategies to address them effectively. Let's discuss the most common solutions for each problem:

1. Overfitting:

Overfitting occurs when a model learns to perform very well on the training data but fails to generalize to new, unseen data. It typically happens when a model is too complex relative to the amount of training data available.

Common solutions for overfitting:

a. *Reduce model complexity:* Use simpler models with fewer parameters or features, such as linear regression instead of complex neural networks.
b. *Cross-validation:* Employ techniques like k-fold cross-validation to assess the model's performance on different subsets of the data. This helps detect overfitting and allows for tuning model hyperparameters accordingly.
c. *Regularization:* Add regularization terms like L1 (Lasso) or L2 (Ridge) to the model's loss function. This penalizes large coefficients and encourages simpler models.
d. *Increase training data:* Collect more data to improve the model's ability to generalize. More data helps the model learn underlying patterns instead of memorizing the training set.
e. *Feature selection:* Choose relevant features and eliminate irrelevant or redundant ones. Feature selection can help reduce model complexity and improve generalization.

2. Underfitting:

Underfitting occurs when a model is too simple to capture the underlying patterns in the data. It results in poor performance on both the training and test datasets.

Common solutions for underfitting:

a. *Increase model complexity:* If the model is too simple, consider using a more complex model or adding more layers to a neural network.
b. *Feature engineering:* Create new features or transform existing ones to provide more relevant information to the model.
c. *Collect more data:* A larger dataset can help a model better understand complex relationships in the data.
d. *Tune hyperparameters:* Adjust hyperparameters like learning rate, the number of layers, or the number of neurons in a neural network to find a better fit for the data.

3. Presence of Outliers:

Outliers are data points that significantly deviate from the general trend in the dataset. They can distort model training and lead to inaccurate predictions.

Common solutions for handling outliers:

a. *Identify and remove outliers:* Use statistical methods such as the Z-score or the interquartile range (IQR) to detect and remove outliers from the dataset. Be cautious when removing outliers, as they may contain valuable information or indicate anomalies.
b. *Transformations:* Apply data transformations like log-transform or Box-Cox to make the data more normally distributed, reducing the impact of extreme values.
c. *Robust models:* Use robust machine learning algorithms that are less sensitive to outliers, such as decision trees or random forests.
d. *Feature engineering:* Create features that are less affected by outliers or encode categorical data in a way that is robust to extreme values.
e. *Winsorization:* Replace extreme values with values within a certain range (e.g., replacing values below the 1st percentile with the 1st percentile value and values above the 99th percentile with the 99th percentile value).

In practice, it's essential to evaluate the effectiveness of these solutions using techniques like cross-validation or hold-out validation to ensure that the chosen approach improves model performance while avoiding overfitting or underfitting. Additionally, domain knowledge and context play a crucial role in deciding which strategies to employ.

Describe the dimensionality problem.

The dimensionality problem, often referred to as the curse of dimensionality, is a common challenge in various fields, including machine learning, data analysis, and optimization. It arises when working with datasets or problems that have a large number of features or dimensions. This problem can have significant implications for the efficiency and effectiveness of algorithms and data analysis techniques. Here's a more detailed description of the dimensionality problem:

1. *Increased Computational Complexity:* As the number of dimensions in a dataset increases, the computational requirements for processing and analyzing that data grow exponentially. Many algorithms that perform well in low-dimensional spaces become impractical or inefficient in high-dimensional spaces. This is because the number of possible combinations and calculations increases exponentially with dimensionality.
2. *Data Sparsity:* In high-dimensional spaces, data points tend to be sparse, meaning that there is limited data available for any given combination of feature values. This sparsity can make it challenging to find meaningful patterns or relationships in the data, as there may not be enough instances or data points in each region of the feature space.
3. *Increased Risk of Overfitting:* High-dimensional datasets are more prone to overfitting, where a model captures noise and random variations in the data rather than the underlying patterns. This can lead to models that perform well on the training data but generalize poorly to new, unseen data.
4. *Difficulty in Visualization:* Visualizing data becomes increasingly challenging as the number of dimensions grows. While we can easily visualize data in two or three dimensions, it becomes nearly impossible to visualize or interpret data in spaces with dozens or hundreds of dimensions.
5. *Curse of Sparsity:* In high-dimensional spaces, most data points tend to be far from each other, making distance-based similarity measures less meaningful. This can affect clustering and nearest-neighbor search algorithms.
6. *Increased Data Requirements:* To maintain statistical significance and reliable model performance, high-dimensional datasets often require a substantial increase in the amount of training data. Collecting and annotating such large datasets can be costly and time-consuming.

Solutions to the Dimensionality Problem:

1. *Feature Selection:* Choose a subset of the most relevant features while discarding irrelevant or redundant ones. This reduces dimensionality and can improve model performance.
2. *Feature Extraction:* Use techniques like principal component analysis (PCA) or autoencoders to transform high-dimensional data into a lower-dimensional representation while retaining as much useful information as possible.
3. *Regularization:* Apply techniques like L1 (Lasso) or L2 (Ridge) regularization to penalize certain feature weights, effectively encouraging the model to use a subset of features.
4. *Dimensionality Reduction:* Explore dimensionality reduction methods like t-SNE or UMAP for visualization and exploratory data analysis.
5. *Curse of Dimensionality-Aware Algorithms:* Some machine learning algorithms are specifically designed to handle high-dimensional data efficiently. Examples include random forests and certain deep learning architectures.
6. *Collect More Data:* Increasing the amount of training data can help mitigate the curse of dimensionality to some extent by providing more information for the model to learn from.

The dimensionality reduction process.

Dimensionality reduction is a data preprocessing technique used to reduce the number of features or dimensions in a dataset while preserving as much of the relevant information as possible. This

process is valuable for various reasons, including simplifying complex datasets, improving the efficiency of machine learning algorithms, and aiding in data visualization. Here's an overview of the dimensionality reduction process:

1. *Data Preprocessing:*

Start with a dataset that contains a high number of features or dimensions. This dataset could be used for tasks such as classification, clustering, or visualization.

2. *Feature Scaling (Optional):*

Before applying dimensionality reduction techniques, it's often a good practice to scale or normalize the features to ensure that they have similar scales. Common scaling methods include Min-Max scaling and standardization (mean normalization).

3. *Choosing a Dimensionality Reduction Technique:*

Select an appropriate dimensionality reduction technique based on the nature of your data and your specific objectives. There are two main categories of dimensionality reduction methods:

- Feature Selection: These methods involve selecting a subset of the original features while discarding the rest. Common techniques include:
  - Univariate feature selection: Select features based on statistical tests like chi-squared, ANOVA, or mutual information.
  - Recursive feature elimination (RFE): Iteratively remove less important features based on a model's performance.
- Feature Extraction: These methods transform the original features into a lower-dimensional space while preserving as much information as possible. Common techniques include:
  - Principal Component Analysis (PCA): A linear technique that identifies orthogonal axes (principal components) along which data variance is maximized.
  - Linear Discriminant Analysis (LDA): A supervised method that finds a projection that maximizes class separability.
  - Non-linear techniques like t-Distributed Stochastic Neighbor Embedding (t-SNE) and Uniform Manifold Approximation and Projection (UMAP) are used for visualization and capturing non-linear relationships in the data.

4. *Parameter Tuning (if applicable):*

Depending on the chosen dimensionality reduction technique, there may be hyperparameters to tune, such as the number of components to retain in PCA or the perplexity in t-SNE. Cross-validation can help determine the optimal parameter values.

5. *Applying Dimensionality Reduction:*

Use the selected dimensionality reduction technique to transform the original dataset. For example, in PCA, this involves projecting the data onto the principal components.

6. *Interpreting Results (if necessary):*

If you're using dimensionality reduction for visualization or exploratory data analysis, interpret the transformed data to gain insights into the relationships between data points.

7. *Using Reduced-Dimension Data:*

The reduced-dimension dataset can be used for various purposes, such as training machine learning models, clustering, or visualization. Ensure that you apply the same dimensionality reduction transformation to both the training and test datasets when using machine learning algorithms.

8. *Validation and Evaluation:*

Assess the impact of dimensionality reduction on your specific task or analysis. Measure the performance of your models or the effectiveness of your analysis using appropriate evaluation metrics.

9. *Iterate (if necessary):*

Depending on the results and your objectives, you may need to iterate on the dimensionality reduction process, trying different techniques or parameter settings to achieve the desired outcome.

Dimensionality reduction is a powerful tool in data analysis and machine learning, but it should be used judiciously, taking into account the trade-offs between simplicity and information loss. It can significantly improve the efficiency and interpretability of models, especially when dealing with high-dimensional data.

Explain the bias-variance trade-off.

The bias-variance trade-off is a fundamental concept in machine learning and statistical modeling that deals with finding the right balance between two sources of error in predictive models: bias and variance. These two sources of error can impact a model's ability to generalize well to unseen data.

1. Bias: Bias refers to the error introduced by approximating a real-world problem with a simplified model. A model with high bias is one that makes strong assumptions about the underlying data distribution. Such models tend to be overly simplistic and may not capture the true underlying patterns in the data. This can lead to underfitting, where the model performs poorly both on the training data and on new, unseen data.

    - Characteristics of high bias models:
    - They are too simple and have few parameters.
    - They make strong assumptions about the data.
    - They are unable to capture complex relationships in the data.

2. Variance: Variance refers to the error introduced due to the model's sensitivity to small fluctuations in the training data. A model with high variance is one that is very flexible and captures noise in the training data, rather than just the underlying patterns. Such models tend to perform well on the training data but poorly on new, unseen data, a phenomenon known as overfitting.

- Characteristics of high variance models:
- They are complex with many parameters.
- They fit the training data very closely, including the noise.
- They are highly sensitive to variations in the training data.

The trade-off arises because, as you increase a model's complexity (e.g., by adding more features or increasing the degree of a polynomial regression model), you typically reduce bias but increase variance, and vice versa. The goal in machine learning is to strike the right balance between bias and variance to build models that generalize well to new data. Here's how it works:

- High bias, low variance models: These models are too simplistic and do not capture the underlying patterns well. They underfit the data. To reduce bias and improve performance, you might need to use more complex models or feature engineering techniques.
- Low bias, high variance models: These models are very flexible and fit the training data closely, including noise. They overfit the data. To reduce variance and improve generalization, you might consider simplifying the model, using regularization techniques, or increasing the amount of training data.
- Balanced models: The goal is to find a model that achieves a good balance between bias and variance, typically at the sweet spot where it minimizes the total prediction error on unseen data. This is often achieved through techniques like cross-validation and hyperparameter tuning.

References

- IBM Documentation. (n.d.). IBM - Deutschland | IBM. https://www.ibm.com/docs/en/spss-modeler/18.1.0?topic=series-outliers
- Introduction to Dimensionality Reduction for Machine Learning - MachineLearningMastery.com. (n.d.). MachineLearningMastery.com. https://machinelearningmastery.com/dimensionality-reduction-for-machine-learning/#:~:text=Dimensionality%20reduction%20refers%20to%20techniques,as%20the%20curse%20of%20dimensionality.
- Overfitting and Underfitting in Machine Learning - Javatpoint. (n.d.). www.javatpoint.com. https://www.javatpoint.com/overfitting-and-underfitting-in-machine-learning
- What is the Curse of Dimensionality? | Data Basecamp. (n.d.). Data Basecamp. https://databasecamp.de/en/ml/curse-of-dimensionality-en
- Singh, S. (2018, May 20). *Understanding the Bias-Variance Tradeoff*. Towards Data Science. https://towardsdatascience.com/understanding-the-bias-variance-tradeoff-165e6942b229