Modul 215: Machine Learning

Linear Regression Methods II: Bias, Variance and Regularization

Wintersemester 2023/24

Tobias Kortus

Center for Technology and Transfer (ZTT)
University of Applied Sciences Worms

November 16, 2023

Recap: Linear Regression I

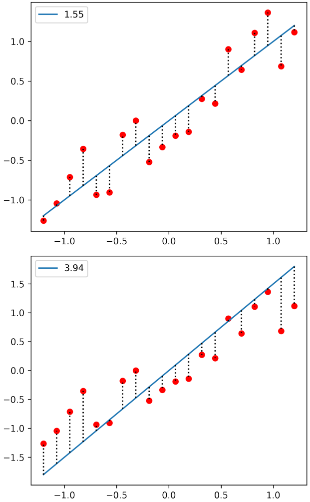
- **Goal:** find a function f that minimizes residual sum of squares:

$$RSS(\beta) = \sum_{i=1}^{N} \Big(\underbrace{y_i - f(x_i)}_{residual} \Big)^2 \qquad (1)$$

- with

$$f(x_1, \dots x_p) = \beta_0 + \sum_{j=1}^{p} \beta_j x_j \qquad (2)$$

- with y_i target value for input x_i and model parameters $\beta = \{\beta_0, \{\beta_j\}_{1:p}\}$



Recap: Linear Regression II

- We can reformulate equation (2) as matrix operations:

$$\hat{y} = \mathbf{X}'^T \beta, \tag{3}$$

- where $\mathbf{X}'$ is a $N\times$ (p+1) matrix containing N feature vectors with a leading 1 for the bias term. Now we can reformulate the RSS objective as:

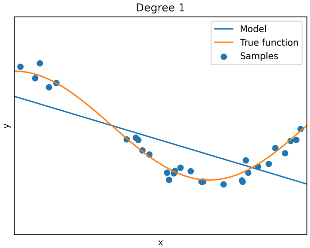
$$RSS(\beta) = \left(y - \mathbf{X}'^T \hat{\beta}\right)^T \left(y - \mathbf{X}'^T \hat{\beta}\right) \tag{4}$$

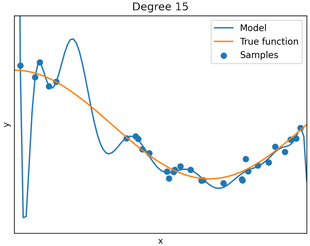
- By differentiating $RSS(\beta)$ w.r.t. β, setting the equation to zero, it gives us a unique solution for $\hat{\beta}$ ($\mathbf{X}'^T\mathbf{X}'$) must be invertible):

$$\hat{\beta} = \left(\mathbf{X}'^T \mathbf{X}'\right)^{-1} \mathbf{X}'^T y \tag{5}$$

Recap: Polynomial Feature Transform Interactions

- Transform our features into a higher-dimensional feature space $\rightarrow$ when selecting good non-linear features, the data might be linear separable.
- **Polynomial Features**:
  - $\phi(x) = [1, x_1, x_2, x_1^2, x_2^2, \dots]$
  - Can be expanded to any kind of nonlinear transformation (e.g. sin, cosine)
- **Interactions**:
  - $\phi(x) = [1, x_1, x_2, x_1 x_2, \dots]$
  - Modeled interactions (combinations) of x_1 with x_2 captures dependencies between those two variables





Motivation - Occam's Razor

"Numquam ponenda est pluralitas sine necessitate." ("Plurality must never be posited without necessity") —William of Ockham

- When you have two competing hypotheses that make the same predictions, the simpler one is most likely the better.
- **Question for this lecture**: How do we quantify the ability of our model? How can we find the best fitting solution with minimum manual effort?

Recap: Training and Generalization Error

- **Training error**: error (e.g. MSE) calculated on the training set.
- **Test error**: error calculated on the test set.
- **Generalization error**: expectation of our model's error were we to apply it to an infinite stream of additional data examples drawn from the same underlying data distribution as our original sample.
- For MSE we get the following equation for the generalization error:

$$\mathbb{E}_{(x,y)\sim P, D\sim P^n}\left[(h_D(x) - y)^2\right] \tag{6}$$

- Generalization error is a mathematical construct used in statistical learning theory. We can never calculate the generalization error exactly $\rightarrow$ we instead estimate the generalization error using the model error on a test set.

Decomposing the Generalization Error

- We can decompose our generalization error into three main parts:

$$\underbrace{\mathbb{E}_{(x,y,D)}\left[(h_D(x)-y)^2\right]}_{\text{Generalization error}} = \underbrace{\mathbb{E}_{x,D}\left[(h_D(x)-\overline{h}(x))^2\right]}_{\text{Variance}} + \underbrace{\mathbb{E}_{x}\left[(\overline{h}(x)-\overline{y}(x))^2\right]}_{Bias^2} \tag{7.1}$$

$$+ \underbrace{\mathbb{E}_{x}\left[(\overline{y}(x)-y(x))^2\right]}_{\text{Noise/ Irreducible Error}} \tag{7.2}$$

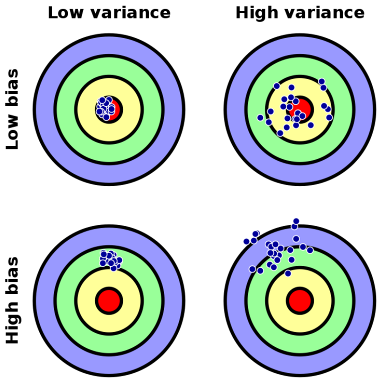
- This allows us to further analyze different impacts on the error.

Decomposing the Generalization Error: Bias Error

- **Recap**: The bias error is defined as:

$$\mathbb{E}_x \left[(\bar{h}(x) - \bar{y}(x))^2 \right] \quad (8)$$

- **Intuition** : What is the error of the model when trained on infinite data (averaged over all possibilities: $\bar{h}$) given the average over the label (e.g. noisy house prices)

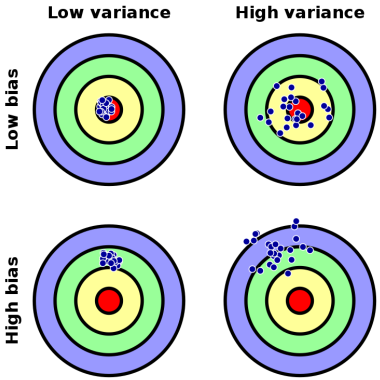


Decomposing the Generalization Error: Variance Error

- **Recap**: The variance error is defined as:

$$\mathbb{E}_{x,D}\left[(h_D(x) - \bar{h}(x))^2\right] \tag{9}$$

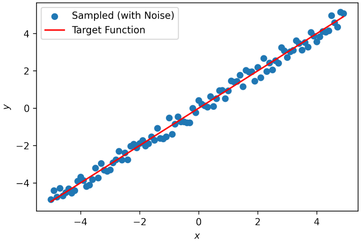
- **Intuition** : How much does our classifier change if we train on different data $\rightarrow$ captures over adaption to data.



Decomposing the Generalization Error: Noise

- **Recap**: Finally we have noise, defined as:

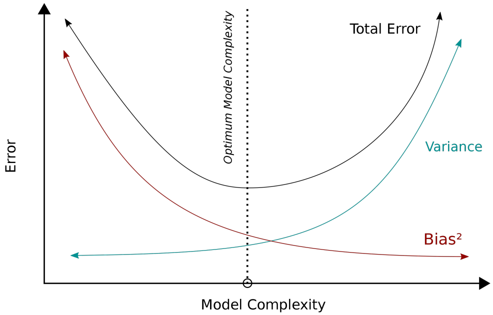
$$\mathbb{E}_x \left[(\bar{y}(x) - y(x))^2 \right] \tag{10}$$



- **Intuition** : How big is the data-intrinsic noise? This error measures ambiguity due to your data distribution and feature representation. You can never beat this (with your model), it is an aspect of the data.

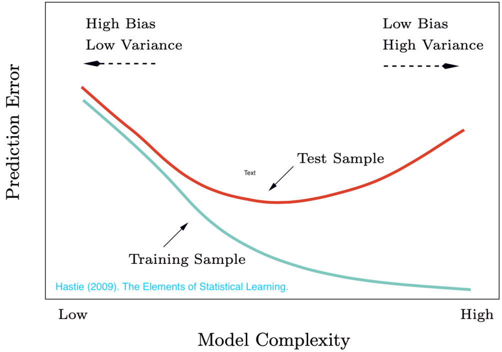
Bias Variance Tradeoff

- Increasing the bias will decrease the variance.
- Increasing the variance will decrease the bias.



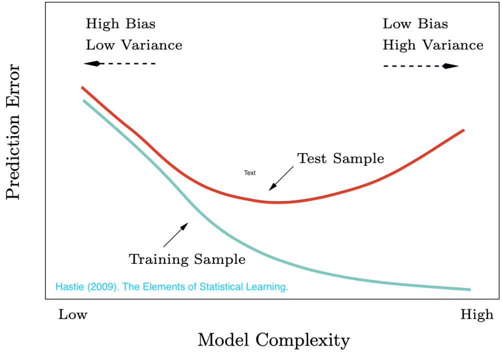
Detecting High Bias and High Variance

- **High Variance** (Regime #1)
- Symptoms
  - Training error is much lower than test error
  - Training error is lower than ϵ
  - Test error is higher than ϵ
- Remedies
  - Add more training data
  - Reduce model complexity
  - Model ensembles (bagging)

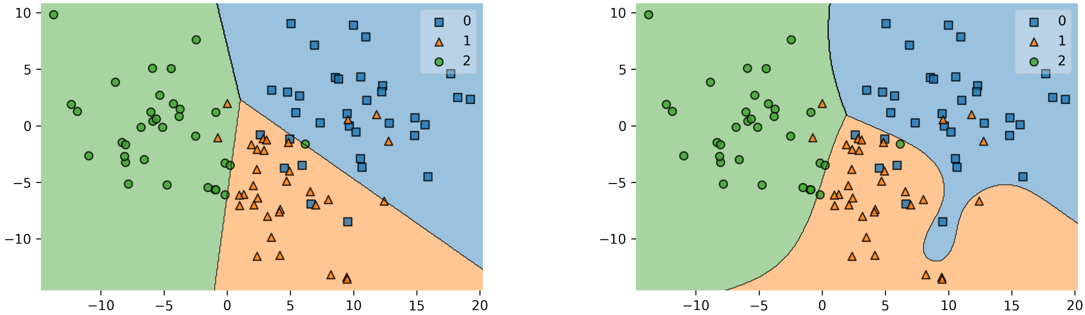


Detecting High Bias and High Variance

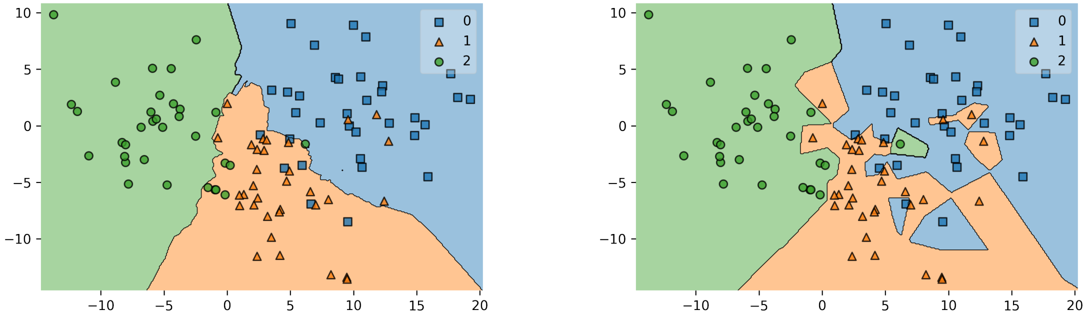
- **High Bias** (Regime #2)
- Symptoms
  - Training error is higher than ϵ
  - Test error is higher than ϵ
- Remedies
  - Use more complex model
  - Add features
  - Model ensembles (boosting)



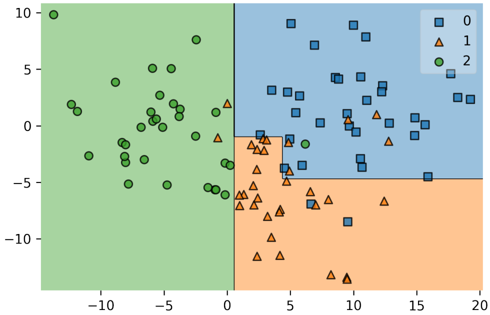
Examples of Overfitting: Logistic Regression

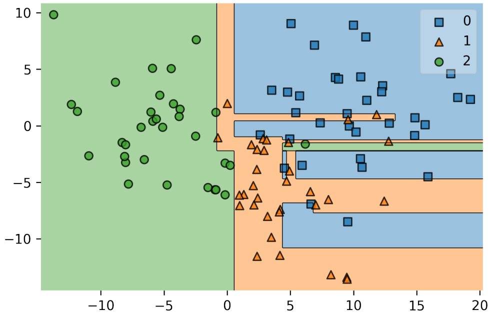


Examples of Overfitting: K-Nearest Neighbor



Examples of Overfitting: Decision Trees





Subset Selection

- One possibility to reduce the variance is to find a subset of the original features that produce the **best results**.
- This also allows for an **easier interpretation** since only a subset of the whole feature set is considered.
- There exist multiple various methods for selecting the best subset of features. We will explore them in the following slides:

Best-Subset Selection

- Find the best subset of $k \leq p$ features given an initial set of p features.
- Use independent validation set/ cross validation for model selection.
- Number of subset sizes k: $\binom{p}{k} \approx p^k$
- With efficient algorithms feasible for moderate feature sizes (30-40). [Hastie et al.]
- Not feasible for larger p

Forward- and Backward-Stepwise Selection

- For $p \gg 40$ [Hastie et al.] finding the best subset of features becomes feasible $\rightarrow$ we need good approximations.
- **Forward-Stepwise Selection**:
  - Start with intercept β_0
  - Sequentially add new variables to the model that most improve the fit $\rightarrow$ Quality of algorithm used to identify variables to test defines speed of the approach.
  - Use independent validation set/ cross validation for model selection.
- **Backward-Stepwise Selection**:
  - Start with full model.
  - Sequentially remove variables to the model that has the least impact on the fit.
  - Use independent validation set/ cross validation for model selection.

Shrinkage Methods

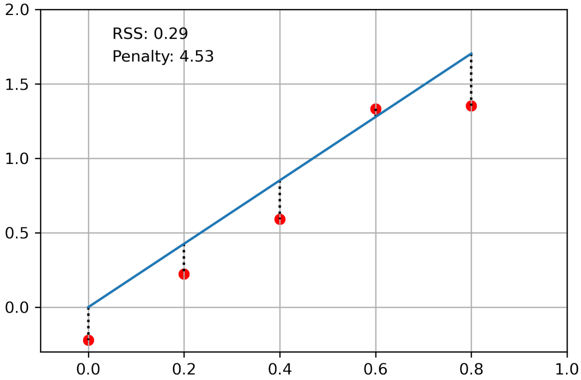
- Subset selection is a discrete process (variables are either retained or discarded) $\rightarrow$ often exhibits high variance $\rightarrow$ doesn't reduce the prediction error of the full model.
- We will explore more continuous alternatives to subset selection.

- **Intuition**: we want to penalize large parameters of the model in order to shrink them towards zero (and each other).
- We therefor penalize the parameters using the sum-of-squares.
- Same concept for neural networks (weight decay, l2-regularization).

$$\hat{\beta}^{ridge} = \arg\min_{\beta} \left\{ \sum_{i=1}^{N} (y_i - \beta_0 + \sum_{j=1}^{p} x_j^{(i)} \beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\} \tag{11}$$

- where λ is a tunable complexity/ hyperparameter that controls the amount of shrinkage

Ridge Regression II: A Simple Example



ridge_lasso_interactive.ipynb

Ridge Regression III: Closed Form Solution

- We can rewrite the preceding equation using matrix notation as:

$$RSS(\lambda) = \left(y - \mathbf{X}'^T \hat{\beta}\right)^T \left(y - \mathbf{X}'^T \hat{\beta}\right) + \lambda \beta^T \beta \tag{12}$$

- Similar to the basic OLS we can find a closed form solution:

$$\hat{\beta} = \left(\mathbf{X}'^T \mathbf{X}' + \lambda \mathbf{I}\right)^{-1} \mathbf{X}'^T y \tag{13}$$

- Where $\mathbf{I}$ is a $(p+1) \times (p+1)$ identity matrix.
- Solution of ridge regression are not equivariant under scaling of the inputs $\rightarrow$ standardization of features.

Recap: Standardization

- Standardization is a method used to normalize the range of independent variables or features of data.

$$x' = \frac{x - \overline{x}}{\sigma} \tag{14}$$

- where x is the original value, $\overline{x}$ is the mean value of x and σ is the standard deviation of the value.
- Transformed features have zero mean with unit (1) variance.

Ridge Regression IV: Limitations

- Ridge regression decreases the complexity of a model but does not reduce the number of variables since it never leads to a coefficient been zero, rather only minimizes it.
- Does not work well for feature reduction.

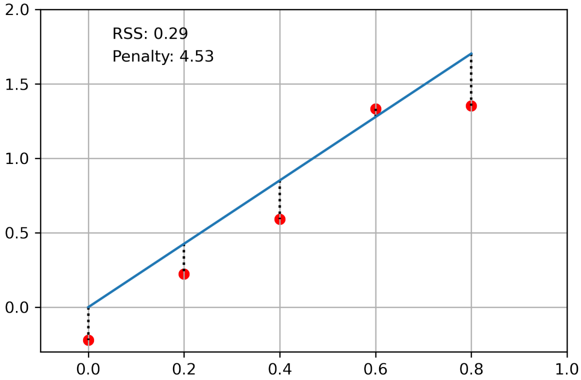
Lasso Regression

- Similar to ridge regression, we add a penalty term for our parameters β.
- Penalty uses l1 norm $\sum_j^p |\beta_j|$ ($|\beta|$) instead of l2 norm $\sum_j^p \beta_j^2$ ($\|\beta\|$)

$$\hat{\beta}^{lasso} = \underset{\beta}{\arg\min} \left\{ \sum_{i=1}^{N} (y_i - \beta_0 + \sum_{j=1}^{p} x_j^{(i)} \beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\} \quad (15)$$

- where λ is also a tunable complexity/ hyperparameter that controls the amount of shrinkage.
- Same concept is used for neural networks (L1 regularization)
- Lasso constraint makes the solution nonlinear $\rightarrow$ there is no closed form solution $\rightarrow$ quadratic programming problem.

Lasso Regression II: A Simple Example



ridge_lasso_interactive.ipynb

Lasso Regression III: Limitations

- If the number of predictors (p) is greater than the number of observations (n), Lasso will pick at most n predictors as non-zero, even if all predictors are relevant.
- If there are two or more highly collinear variables then LASSO regression select one of them randomly which is not good for the interpretation of data

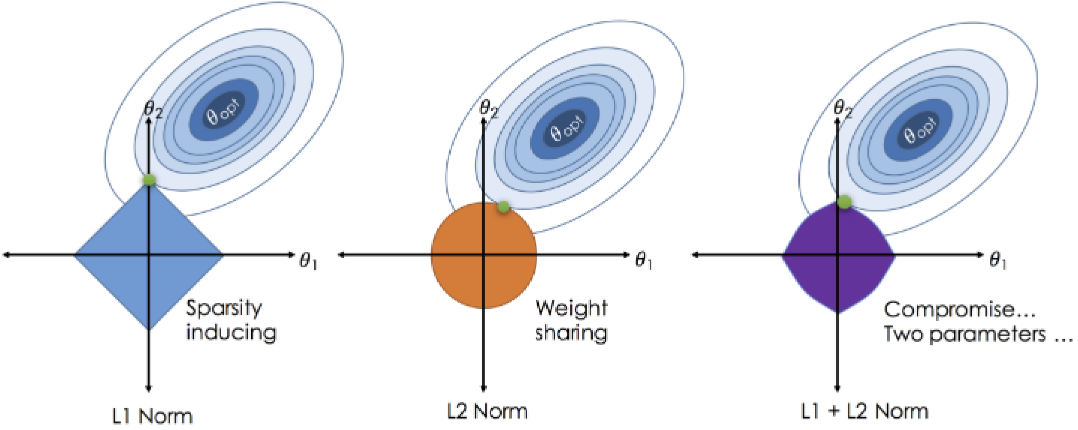
Elasticnet Regression

- Combination of Ridge & Lasso regression

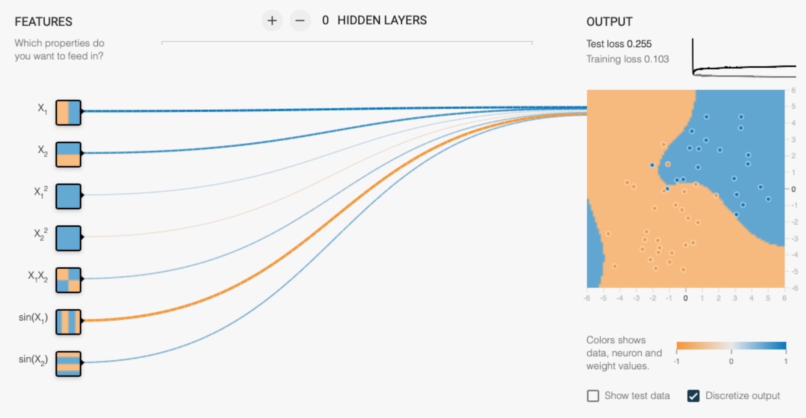
$$\hat{\beta} = \arg\min_{\beta} \left\{ \sum_{i=1}^{N} (y_i - \beta_0 + \sum_{j=1}^{p} x_j^{(i)} \beta_j)^2 + \lambda_2 \sum_{j=1}^{p} \beta_j^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| \right\} \quad (16)$$

- where λ_1 and λ_2 are **individual** regularization factors for ridge and lasso penalty terms.

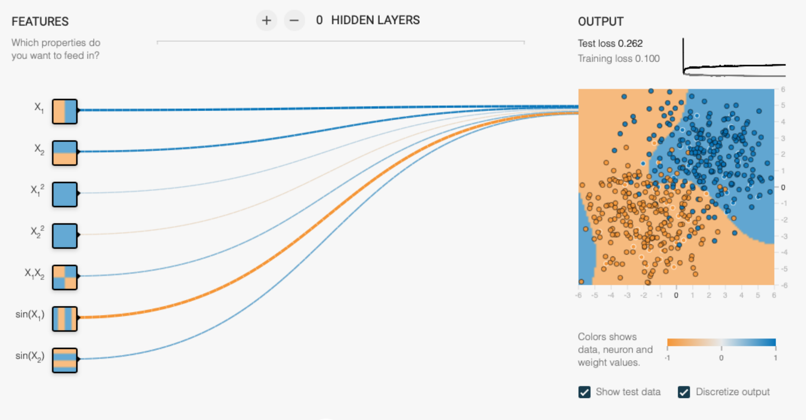
Ridge Regression vs. Lasso Regression



Shrinkage Methods: TensorFlow Playground

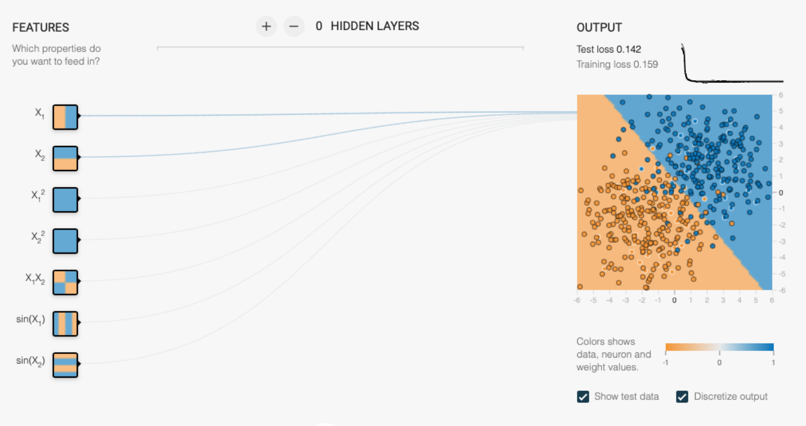


Shrinkage Methods: TensorFlow Playground



- When enabling test data (checkbox under output) we see that our model overfits to the training data.

Shrinkage Methods: TensorFlow Playground



- Activating regularization methods (L1/L2) allows us to control the model capacity.

Controlling Overfitting and Underfitting in Other Models

- Under- and overfitting is also a problem for other machine learning models.
- In most cases, reduce/increase model complexity using model hyperparameters:
- **Decision Tree**:
  1. max_depth
  2. min_samples_split
  3. min_samples_leaf
  4. ...
- **K-nearest neighbor**:
  1. n_neighbors
  2. ...

- Read documentation and try modifying parameters (either manual or hyperparameter optimization)

Final Notes to Linear Models

- We explored multiple linear models. However, there are still many more:
- `https://scikit-learn.org/stable/modules/classes.html`

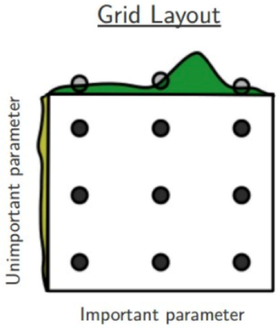
(Automatic) Hyperparameter Optimization

- Hyperparameter tuning is just an optimization loop on top of ML model learning to find the set of hyperparameters leading to the lowest error on the validation set.
- Manually optimizing hyperparameters is often not sufficient (requires lots of effort).
- We want to use search algorithms in order to find good/ optimal parameters.

- **Note**: All concepts from model evaluation train, test, eval split; cross-validation; etc. apply also for hyperparameter optimization

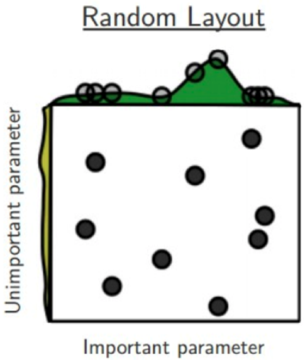
Grid Search

- Exhaustive search over all possible parameter configurations over a user defined grid.
- Requires preliminary knowledge on parameters $\rightarrow$ specify candidates in search space manually.
- Most straightforward search algorithm that leads to the most accurate predictions as long as sufficient resources are given.



Random Search

- Randomized search over hyper-parameters from certain distributions over possible parameter values.
- Searching process continues till the predetermined budget is exhausted, or until the desired accuracy is reached
- In most cases, random search is more effective than grid search, but it is still a computationally intensive method.



Further Optimization Methods

- Bayesian optimization
- Tree parzen estimators
- Evolutionary algorithms
- ...

- Further read:
  - Yu, T., Zhu, H. (2020). Hyper-Parameter Optimization: A Review of Algorithms and Applications. 1-56. http://arxiv.org/abs/2003.05689

A List of Personally Recommended Hyperparameter Optimization Frameworks

- Really simple: `https://scikit-learn.org/stable/modules/grid_search.html#`
- More advanced: `https://optuna.org`

References

- Hastie, T., Tibshirani, R., Friedman, J. H. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer.
- Deisenroth, M. P., Faisal, A. A.,, Ong, C. S. (2020). Mathematics for Machine Learning. Cambridge University Press.
- Yu, T., Zhu, H. (2020). Hyper-Parameter Optimization: A Review of Algorithms and Applications. 1-56. http://arxiv.org/abs/2003.05689
- https://proceedings.neurips.cc/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf
- Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011, NIPS 2011, 1-9.
- `https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote12.html`