**Bias - Variance tradeoff**

What is all the fuss about bias and variance? Error is measured in two ways:

- Bias
- Variance

Bias: how far are you from target?
Variance: how consistent are you in your shoots?
(See target picture)

Question: Describe a setting for which bias is “worse” than variance:
⇒ casino earnings: because a lot of people are playing, casinos only care about bias: if they win by a slight margin all the time, they will win on average and they can sustain big losses (big gains for a player) from time to time.

Question: Describe a setting for which variance is “worse” than bias:
⇒ waiting in line: usually people don’t mind waiting more than expected as long as they know how long it takes. But people hate when they don’t know how long they have to wait. Do you agree?

Fun fact: Markov Chain theory shows that the expected value of the waiting time of a customer at a bus stop depends not only on the expected arrival time of buses but also on the variance of such system.

There are 3 types of prediction error: bias, variance, and irreducible error.

Irreducible error is also known as "noise," and it can't be reduced by your choice in algorithm. It typically comes from inherent randomness, a mis-framed problem, or an incomplete feature set.

The other two types of errors, however, can be reduced because they stem from your algorithm choice.

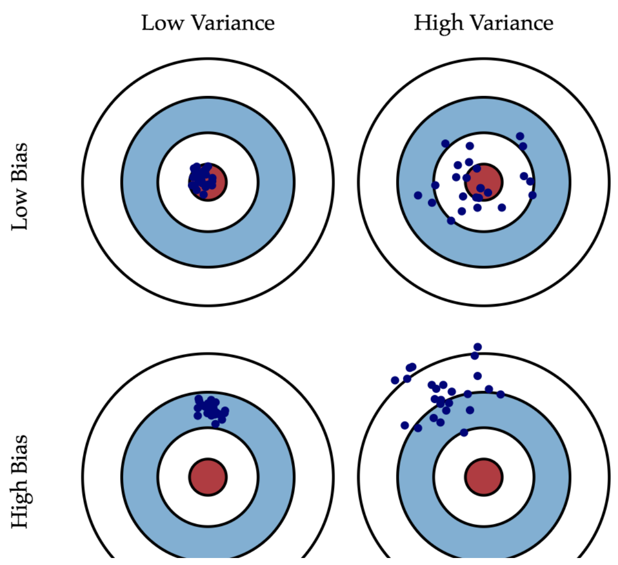
## Error from Variance

Variance refers to your algorithm's sensitivity to specific sets of training data.

High variance algorithms will produce drastically different models depending on the training set.

## Error from Bias

Bias is the difference between your model's expected predictions and the true values.

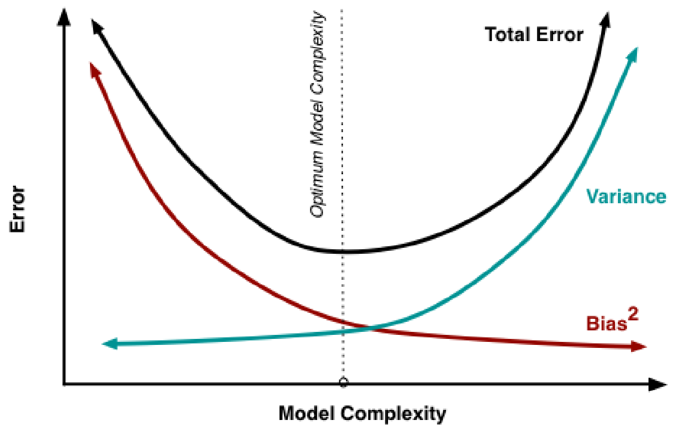


Let's return to the issue of picking $p$. We saw above that if we had 100,000 extra data, we could just fit the model with each $p$, and pick the $p$ that does best on the validation data. What would you do if you don't happen to have a spare 100,000 extra data sitting around? The first idea that comes to mind is to "hold out" some of our original training data.

1. Split TRAIN into $\text{TRAIN}_{\text{fit}}$ and $\text{TRAIN}_{\text{validation}}$. (e.g. half in each)
2. Fit each model to $\text{TRAIN}_{\text{train}}$, and evaluate how well it does on $\text{TRAIN}_{\text{validation}}$.
3. Output the model that has the best score on $\text{TRAIN}_{\text{validation}}$.

This can work reasonably well, but it "wastes" the data by only training on half, and only testing on half. We can make better use of the data by making several different splits of the data. Each datum is used once for testing, and the other times for training. This algorithm is called **K-fold cross validation**.

1. Split TRAIN into $K$ chunks
2. For $k = 1, 2, ..., K$:
   (a) Set $\text{TRAIN}_{\text{validation}}$ to be the $k$th chunk of data, and $\text{TRAIN}_{\text{fit}}$ to be the other $K - 1$ chunks.
   (b) Fit each model to $\text{TRAIN}_{\text{fit}}$ and evaluate how well it does on $\text{TRAIN}_{\text{validation}}$.



A proper machine learning workflow includes:

- Separate training and test sets
- Trying appropriate algorithms
- Fitting model parameters
- Tuning impactful hyperparameters
- Proper performance metrics
- Systematic cross-validation

Project advice example:

**Dealing with Text and Sentiment in Logistic Regression:**

1. **Feature Engineering:**
    - Selecting **V features** that are most relevant for classification.
    - Counting all possible **positive and negative words** in the dataset.
    - Using **word vectors** to capture both **semantic and syntactic structures**.
2. **Building a Dictionary for Encoding:**
    - Creating a **Word2Ind dictionary** (mapping words to indices).
        - **Why?** This allows efficient numerical representation of words for ML models.
    - Creating an **Ind2Word dictionary** (mapping indices back to words).
        - **Why?** Useful for interpretability and debugging.
3. **Processing the Data:**
    - **Padding** sequences: What size should you pad to? (Choose based on sequence length distribution).

- **Masking the loss** for padded tokens to prevent the model from learning from artificial padding.
- **Batching** the data efficiently for training (ensuring similar-length sequences are batched together).

## Additional Project Advice for ML Workflows:

### Handling Bias-Variance Tradeoff in Real-World Models:

4. **High Bias Issues** (underfitting):
   - Use **more complex models** (e.g., deeper trees, neural networks).
   - Engineer better **features** (e.g., interaction terms, embeddings).
   - Increase model **capacity** with additional layers or parameters.
5. **High Variance Issues** (overfitting):
   - Regularization techniques (**L1/L2 penalties, dropout**).
   - More **data augmentation** to improve generalization.
   - Use **cross-validation** to get more stable performance metrics.

   -