Regularization

Regularization

- Regularization seeks to solve a few common model issues by:
    - Minimizing model complexity
    - Penalizing the loss function
    - Reducing model overfitting (add more bias to reduce model variance)

Regularization

- In general, we can think of regularization as a way to reduce model overfitting and variance.
    - Requires some additional bias
    - Requires a search for optimal penalty hyperparameter.

Regularization

- Three main types of Regularization:
  - L1 Regularization
    - LASSO Regression
  - L2 Regularization
    - Ridge Regression
  - Combining L1 and L2
    - Elastic Net

Regularization

- L1 regularization adds a penalty equal to the **absolute value** of the magnitude of coefficients.
  - Limits the size of the coefficients.
  - Can yield sparse models where some coefficients can become zero.

وزارة الاتصـــــالات
وتكنولوجيا المعلومات

رواد مصر الرقمية

Regularization

- L1 regularization adds a penalty equal to the **absolute value** of the magnitude of coefficients.

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}|\beta_j| = \mathrm{RSS}+\boxed{\lambda\sum_{j=1}^{p}|\beta_j|}$$

Regularization

- L2 regularization adds a penalty equal to the **square** of the magnitude of coefficients.
  - All coefficients are shrunk by the same factor.
  - Does not necessarily eliminate coefficients.

Regularization

- L2 regularization adds a penalty equal to the **square** of the magnitude of coefficients.

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}\beta_j^2=\mathrm{RSS}+\boxed{\lambda\sum_{j=1}^{p}\beta_j^2}$$

Regularization

- Elastic net combines L1 and L2 with the addition of an alpha parameter deciding the ratio between them:

$$\frac{\sum_{i=1}^{n}(y_i - x_i^J \hat{\beta})^2}{2n} + \lambda \left(\frac{1-\alpha}{2} \sum_{j=1}^{m} \hat{\beta}_j^2 + \alpha \sum_{j=1}^{m} |\hat{\beta}_j| \right)$$

Regularization

- Elastic net combines L1 and L2 with the addition of an alpha parameter deciding the ratio between them:

$\alpha = 0$

$$\frac{\sum_{i=1}^{n}(y_i - x_i^J \hat{\beta})^2}{2n} + \lambda \left(\frac{1-\alpha}{2} \sum_{j=1}^{m} \hat{\beta}_j^2 + \cancel{\alpha \sum_{j=1}^{m} |\hat{\beta}_j|} \right)$$

Regularization

- Elastic net combines L1 and L2 with the addition of an alpha parameter deciding the ratio between them:

$\alpha = 1$

$$\frac{\sum_{i=1}^{n}(y_i - x_i^J \hat{\beta})^2}{2n} + \lambda \left(\cancel{\frac{1-\alpha}{2} \sum_{j=1}^{m} \hat{\beta}_j^2} + \alpha \sum_{j=1}^{m} |\hat{\beta}_j| \right)$$

Regularization

- These regularization methods do have a cost:
    - Introduce an additional hyperparameter that needs to be tuned.
    - A multiplier to the penalty to decide the “strength” of the penalty.

Regularization

- Later on, we will actually cover L2 regularization (Ridge Regression) first, due to the intuition behind the squared term being easier to understand.

Regularization

- Before we dive straight into coding regularization with Scikit-Learn, we need to discuss a few more relevant topics:
  - Feature Scaling
  - Cross Validation

Feature Scaling

Feature Scaling

- Feature scaling provides many benefits to our machine learning process!
- Some machine learning models that rely on distance metrics (e.g. KNN) **require** scaling to perform well.
- Let’s discuss the main ideas behind feature scaling...

Feature Scaling

- Feature scaling improves the convergence of steepest descent algorithms, which do not possess the property of scale invariance.
- If features are on different scales, certain weights may update faster than others since the feature values x_j play a role in the weight updates.

Feature Scaling

- Critical benefit of feature scaling related to gradient descent.
- There are some ML Algos where scaling won't have an effect (e.g. CART based methods).

Feature Scaling

- Scaling the features so that their respective ranges are uniform is important in comparing measurements that have different units.
- Allows us directly compare model coefficients to each other.

Feature Scaling

- Feature scaling caveats:
  - Must always scale new unseen data before feeding to model.
  - Effects direct interpretability of feature coefficients
    - Easier to compare coefficients to one another, harder to relate back to original unscaled feature.

Feature Scaling

- Feature scaling benefits:
    - Can lead to great increases in performance.
    - Absolutely necessary for some models.
    - Virtually no “real” downside to scaling features.

رواد مصر الرقمية

Feature Scaling

وزارة الاتصالات
وتكنولوجيا المعلومات

- Two main ways to scale features:
  - Standardization
    - Rescales data to have a mean (μ) of 0 and standard deviation (σ) of 1.
  - Normalization
    - Rescales all data values to be between 0-1.

Feature Scaling

- Standardization:
  - Rescales data to have a mean (μ) of 0 and standard deviation (σ) of 1 (unit variance).

$$X_{changed} = \frac{X - \mu}{\sigma}$$

Feature Scaling

- Standardization:
  - Namesake can be confusing since this is also referred to as “Z-score normalization”.

$$X_{changed} = \frac{X - \mu}{\sigma}$$

Feature Scaling

- Normalization:
  - Scales all data values to be between 0 and 1.

$$X_{changed} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Feature Scaling

- Normalization:
  - Simple and easy to understand.

$$X_{changed} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Feature Scaling

- There are many more methods of scaling features and Scikit-Learn provides easy to use classes that “fit” and “transform” feature data for scaling.
- Let’s quickly discuss the fit and transform calls in more detail when it comes to scaling.

Feature Scaling

- A .fit() method call simply calculates the necessary statistics (Xmin,Xmax,mean, standard deviation).
- A .transform() call actually scales data and returns the new scaled version of data.
- Previously saw a similar process for polynomial feature conversion.

Feature Scaling

- Very important consideration for fit and transform:
    - We only **fit** to training data.
    - Calculating statistical information should only come from training data.
    - Don’t want to assume prior knowledge of the test set!

Feature Scaling

- Using the full data set would cause **data leakage**:
  - Calculating statistics from full data leads to some information of the test set leaking into the training process upon transform() conversion.

Feature Scaling

- Feature scaling process:
  - Perform train test split
  - Fit to training feature data
  - Transform training feature data
  - Transform test feature data

Feature Scaling

- Do we need to scale the label?
    - In general it is not necessary nor advised.
    - Normalising the output distribution is altering the definition of the target.
    - Predicting a distribution that doesn't mirror your real-world target.

Feature Scaling

- Do we need to scale the label?
  - Can negatively impact stochastic gradient descent.
- stats.stackexchange.com/questions/111467

Feature Scaling

- Now that we understand the benefits of feature scaling, let’s move on to understanding the benefits of cross-validation!