Data Science Bonus Models Documentation

1. Lasso Regression

How it works:

Lasso regression is a linear regression method that uses L1 regularization. It minimizes the sum of squared errors with an added penalty proportional to the absolute value of the coe icients. This penalty encourages sparsity, meaning some coe icients become exactly zero, e ectively performing feature selection.

Key steps:

* + Define the linear regression problem with L1 penalty.
  + Use optimization (e.g., coordinate descent) to estimate coe icients minimizing the penalized error.
  + Some coe icients shrink to zero, reducing model complexity.

Justification for choosing:

* + The dataset contains many correlated continuous features.
  + Lasso helps by selecting important features, improving interpretability and reducing overfitting.
  + Good baseline linear model with built-in feature selection.

Performance:

* + R² Score: 0.92896
  + Mean Squared Error (MSE): 0.63736

1. Elastic Net Regression

How it works:

Elastic Net combines L1 (Lasso) and L2 (Ridge) penalties in the loss function. This balances feature selection (L1) and coe icient shrinkage (L2), handling correlated variables better than Lasso alone.

Key steps:

Minimize squared error with both L1 and L2 regularization terms.

Tune mixing parameter to balance between L1 and L2.

* + Solve with optimization algorithms like coordinate descent.

Justification for choosing:

* + Features are highly correlated, and pure Lasso might arbitrarily select one.
  + Elastic Net stabilizes feature selection and coe icient estimates.
  + Provides better predictive performance when correlations exist.

Performance:

* + R² Score: 0.9253
  + Mean Squared Error (MSE): 0.6701

1. Random Forest Regressor

How it works:

Random Forest builds an ensemble of decision trees trained on random subsets of data and features. Each tree predicts the target, and the forest’s output is the average of all tree predictions. This reduces variance and overfitting compared to a single tree.

Key steps:

* + Bootstrap sampling to create training subsets for each tree.
  + Randomly select subsets of features for split decisions.
  + Train multiple trees independently.
  + Aggregate predictions by averaging.

Justification for choosing:

* + Can model nonlinear relationships and complex feature interactions.
  + Handles correlated features robustly without explicit feature engineering.
  + Less prone to overfitting than individual trees.
  + Suitable for datasets with moderate size (~1k entries).

Performance:

R² Score: 0.94123

Mean Squared Error (MSE): 0.5273

1. XGBoost Regressor

How it works:

XGBoost is a gradient boosting method that sequentially builds trees where each tree attempts to correct errors of the previous ones. It includes regularization to reduce overfitting and supports handling missing data internally.

Key steps:

* + Initialize with a simple prediction.
  + Iteratively add trees trained on residual errors.
  + Use gradient descent on the loss function with regularization terms.
  + Stop training when performance no longer improves or max trees reached.

Justification for choosing:

* + Powerful and scalable gradient boosting algorithm.
  + E ective at capturing complex nonlinear relationships.
  + Strong performance on many regression problems.
  + Handles missing data and regularization well.

Performance:

* + R² Score: 0.92117
  + Mean Squared Error (MSE): 0.7073

1. Logistic Regression

How it works:

Logistic regression models the probability of a binary outcome using the logistic function applied to a linear combination of input features. The model parameters are optimized to maximize the likelihood of observed outcomes.

Key steps:

Define logistic function σ(z)=11+e−z\sigma(z) = \frac{1}{1 + e^{-z}}σ(z)=1+e−z1.

Compute predicted probabilities for classes.

* + Use maximum likelihood estimation to fit parameters.
  + Predict class labels by thresholding probabilities.

Justification for choosing:

* + Baseline and interpretable model for binary classification.
  + E icient to train on large datasets.
  + Good if classes are linearly separable or nearly so.

Performance:

* + Accuracy: 0.9790
  + ROC AUC: 0.9979

1. XGBoost Classifier

How it works:

XGBoost classifier uses gradient boosting trees to optimize a classification objective (like logistic loss). Trees are added sequentially to correct previous mistakes, with regularization and e icient computation.

Key steps:

* + Initialize predictions.
  + Iteratively add trees focusing on misclassified samples.
  + Regularize to prevent overfitting.
  + Use softmax or logistic loss for classification.

Justification for choosing:

* + High predictive power for classification problems.
  + Handles complex feature interactions and missing values.
  + Often achieves state-of-the-art results.

Performance:

Training:

Accuracy: 0.9995

* + Precision, Recall, F1-score: 1.00 (perfect) Testing:
  + Accuracy: 0.9767
  + Precision, Recall, F1-score: ~0.98 for both classes