Statistical and Machine Learning

Individual Assessment

Bharanidharan Murugesan

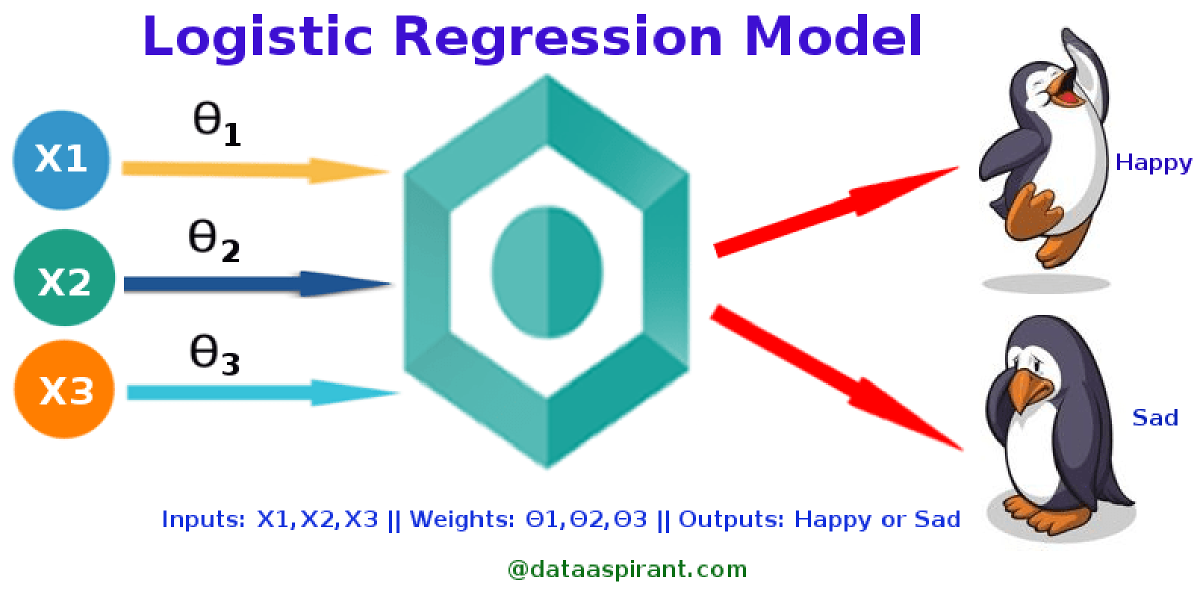
Logistic Regression

Logistic regression is a supervised machine learning algorithm used for classification tasks where the goal is to predict the probability that an instance belongs to a given class or not. Logistic regression is a statistical algorithm which analyze the relationship between two data factors. The article explores the fundamentals of logistic regression, it's types and implementations.

Logistic regression is a method that can help us predict categories. Assume we have a set of data points and you want to know whether something is true or untrue, yes or no, or belongs to one group or another.

It examines the relationship between our input data (age, salary, etc.) and the outcome we want to forecast (whether or not someone will buy a product). Instead of offering a clear answer, it estimates the likelihood (or chance) that something belongs to a specific category.

So it's a mechanism that allows us to make educated assumptions about categories based on our data.


Logistic Regression Model
X1
X2
X3
θ1
θ2
θ3
Happy
Sad
Inputs: X1,X2,X3 || Weights: Θ1,Θ2,Θ3 || Outputs: Happy or Sad
@dataaspirant.com

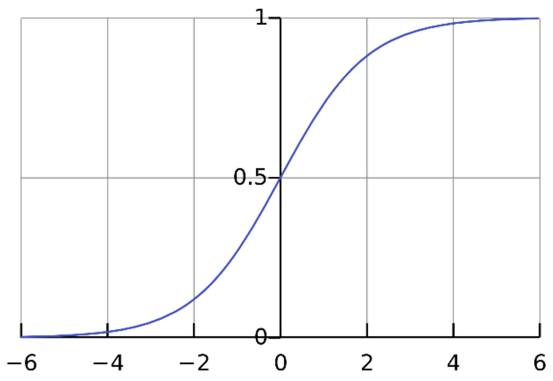
Types of Logistic Regression:

- Binary Logistic Regression
- Multinomial Logistic Regression
- Ordinal Logistic Regression

Key Properties of Logistic Regression:

1. Sigmoid Function
2. Straightforward Relationship
3. Coefficients
4. Best Guess

5. Best Assumptions
6. Probability



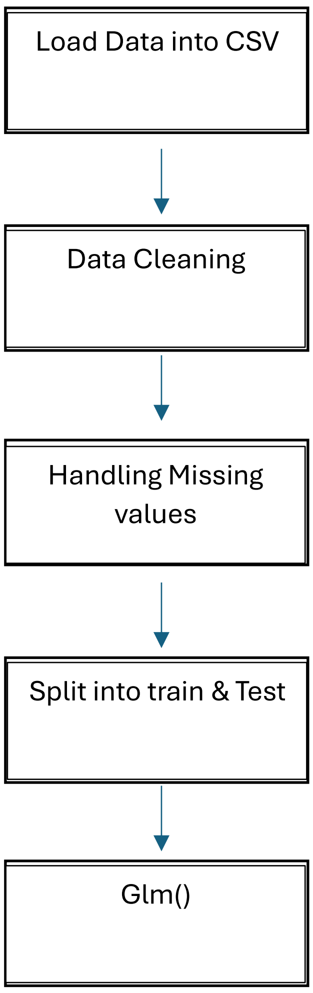
Without the Sigmoid function, Logistic Regression would just be Linear Regression. That means that the output of the model could range from -∞ to ∞.

The Sigmoid Function:

$$sigmoid(Z) = \frac{1}{1+e^{-Z}}$$

Algorithm Fitting Process

R makes it very easy to fit a logistic regression model. The function to be called is glm().



Advantages:

- Easy to implement and interpret yet efficient in training.
- The predicted parameters give inference about the importance of each feature.
- Performs well on low-dimensional data.
- Very efficient when the dataset has features that are linearly separable.
- Outputs well-calibrated probabilities along with classification results.

Disadvantages:

- Overfits on high dimensional data
- Assumes linearity between dependent and independent variables.
- Non linear problems can't be solved with logistic regression since it has a linear decision surface
- Fails to capture complex relationships.
- Only important and relevant features should be used otherwise model's predictive value will degrade.

Decision Tree

A decision tree is a non-parametric supervised learning algorithm, which is utilized for both classification and regression tasks. It has a hierarchical, tree structure, which consists of a root node, branches, internal nodes and leaf nodes.

A decision tree can be thought of as a decision-making flowchart. Imagine you're trying to decide whether to go outside or stay inside based on the weather and your mood.

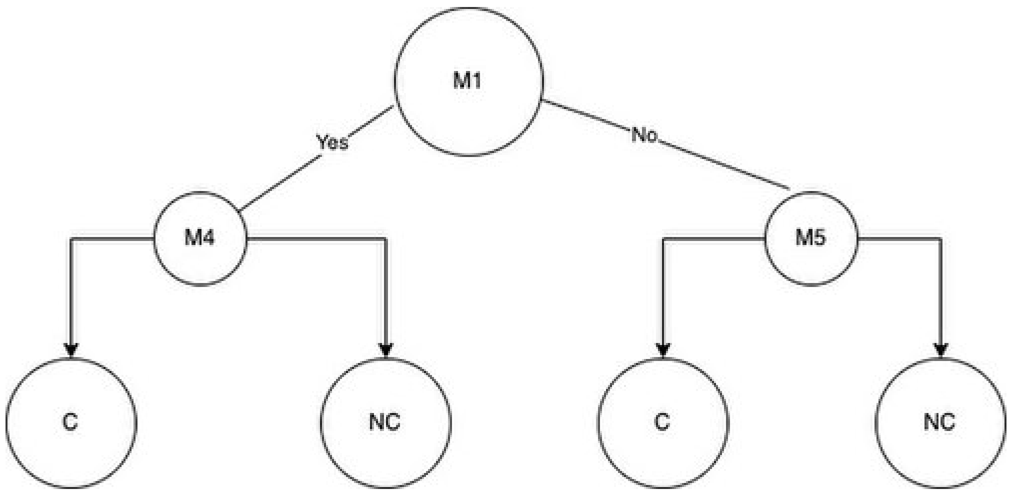
Root Node: This is the starting point for our decision tree. It poses a large inquiry, such as, "Is it raining?"

Branches: These are the several paths we can travel depending on the answers to our inquiries. For example, if the answer to "Is it raining?" is yes, we may inquire, "Do you have an umbrella?"

Internal Nodes: These are the queries we raise along the journey. They assist us narrow down our choices. For example, after inquiring about the rain, we could ask about the wind.

Leaf Nodes represent the final outcomes or decisions. In our example, it may either "stay inside" or "go outside with an umbrella."

So, a decision tree assists us in making decisions by posing a series of questions and sending us down various pathways based on our responses. It's a useful tool for both categorizing (determining whether an email is spam) and predicting values.



The decision trees are determined based on certain attributes such as Entropy, Information gain, Gini index, Gain ratio, Reduction in variance

The working of a Decision Tree in R involves several steps using packages like 'rpart' or 'partykit'.

1. Data Preparation
2. Model construction
3. Tree visualization
4. Prediction
5. Model evaluation
6. Tuning & Pruning
7. Interpretation
8. Ensemble Methods
9. Model Deployment

Advantages:

- Compared to other techniques, decision trees require less work to prepare data during preprocessing.
- Data normalization is not required when using a decision tree.
- Scaling data is not required with a decision tree.
- Missing values in the data have no significant effect on the decision tree building process.
- A decision tree model is simple and straightforward to communicate to both technical teams and stakeholders.

Disadvantages:

- A slight change in the data can result in a significant change in the structure of the decision tree, generating instability.
- Calculations for a decision tree can be significantly more complex than those for other algorithms.
- Training a decision tree model often takes longer.
- Decision tree training is relatively expensive due to its intricacy and time requirements.
- The Decision Tree technique is ineffective for regression and predicting continuous values.

Random Forest

Random Forest is a widely-used machine learning algorithm developed by Leo Breiman and Adele Cutler, which combines the output of multiple decision trees to reach a single result. Its ease of use and flexibility have fueled its adoption, as it handles both classification and regression problems.

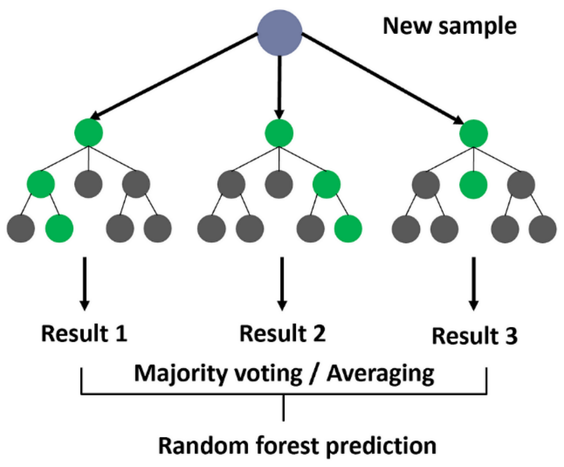
Assume you're trying to make a big decision, such as where to go for vacation. Instead of asking just one person for guidance, ask a group of friends with diverse backgrounds and tastes.

Random Forest: It's like asking a group of friends (trees) about your holiday spot. Each friend (tree) shares their thoughts based on their personal experiences and knowledge.

Combining Results: After hearing from all of your friends (trees), you combine their recommendations to get a final conclusion. Perhaps one companion like the seaside, another prefers the mountains, and still another advises a city trip. You consider all of these options before deciding on the best solution for everyone's preferences.

Random Forest can handle it all, whether you're picking between holiday kinds (such as beach, mountain, or city) or predicting something particular (such as the temperature on your preferred day).

So, Random Forest is analogous to a group decision-making method in which several "opinions" from separate decision trees are combined to assist generate a more accurate and well-rounded overall choice.



Implementing Random Forest in R:

randomForest()

1. Data Preparation

2. Split Data into Train and Test
3. Convert target variable into Factor form
4. Find optimized Value
5. Create Random Forest Model

Advantages:

- Able to learn non linear decisions
- Higher Accuracy
- Flexible and Robust
- Scalable
- Parallel processing

Disadvantages:

- Interpretability
- Complex Computation
- Noise sensitive

K-Nearest Neighbour

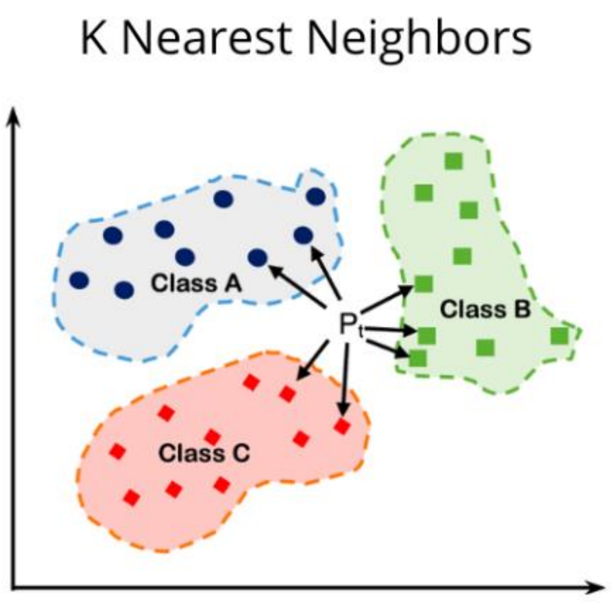
KNN is a simple, supervised machine learning (ML) algorithm that can be used for classification or regression tasks - and is also frequently used in missing value imputation. It is based on the idea that the observations closest to a given data point are the most "similar" observations in a data set, and we can therefore classify unforeseen points based on the values of the closest existing points. By choosing K, the user can select the number of nearby observations to use in the algorithm.

KNN (K-nearest neighbours): It's similar to asking your friends what genre they think the new music belongs to based on their favourite tunes.

Choosing "K": "K" represents how many friends you ask. If you ask three friends (K=3), they will give you three different judgments about the song's genre based on their favourite songs.

Finding Similar Songs: KNN searches for songs that are similar to the new song (the closest neighbours) and inquires about their genre. If most of them are rock, the new song may also be rock.

KNN can help you categorize music into categories (classification) or predict things like a song's popularity (regression).



Implementation of KNN

1. Determining the optimal value of K(no.of nearest neighbours)
2. Calculate the distance.(Euclidean)
3. Identify the nearest neighbours.

Advantages:

- Simple and scalable.
- Less computation Time
- Predictive Power is High
- Has relatively high accuracy

Disadvantages:

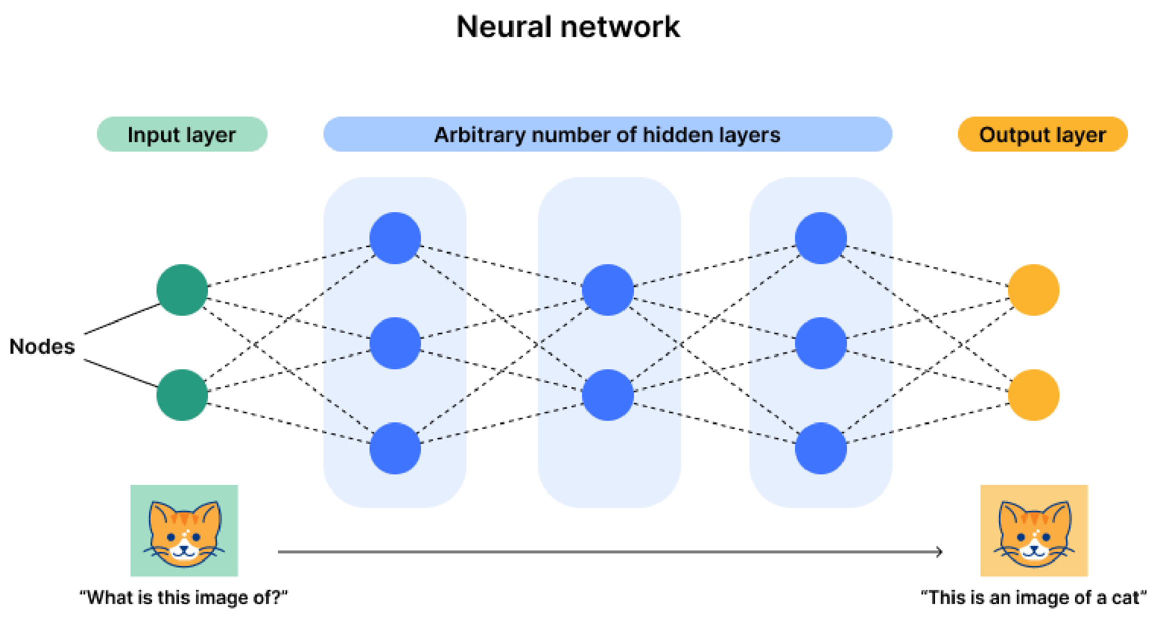
- Prediction slow for large datasets
- Unable to learn from the training set.
- Very sensitive to features.

Neural Networks

A neural network, or artificial neural network, is a type of computing architecture that is based on a model of how a human brain functions — hence the name "neural." Neural networks are made up of a collection of processing units called "nodes." These nodes pass data to each other, just like how in a brain, neurons pass electrical impulses to each other.

A neural network functions similarly to a group of people collaborating to solve an issue. Each member examines a specific aspect of the problem and reports back to the group. They collaborate to discover the greatest solution by merging everyone's perspectives.

So, a neural network is a collaborative effort in which each component contributes to the solution of complicated problems, inspired by the way human brains work.



Implementing NN :

1. Data preparation
2. Scaling The Data
3. Fit model in to the network
4. Plotting

Advantages:

- Provides effective visual analysis

- Processing of un organized data
- User friendly

Disadvantages:

- Require high performance machines
- Results might not be complete
- Data Dependant
- Black Box

Bench Marking the model

Project Definition:

The data set consist of customer information which provides about the telecommunication for certain campaign. The main motive of this project is to identify the potential customers that are likely to subscribe using various machine learning models that we came across above. We are implementing these models using Python.

Data Preprocessing:

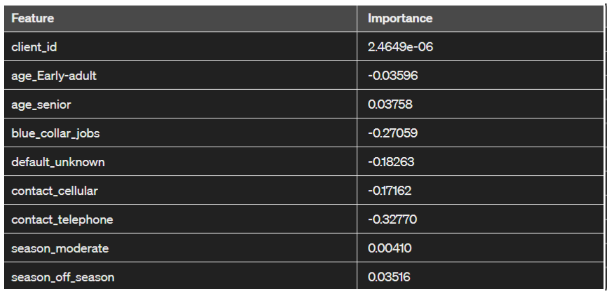
Loaded the data in to the notebook and inspected thoroughly. Based on the information it has been determined as the classification problem. So our course of action depends on implementing successful classification model. Handled possible null values within the dataset and dropped certain rows that aren't relevant to the modelling phase. Used one-hot encoding to convert category variables in binary variables. Once everything is done we have our basetable ready for the next step.

Split:

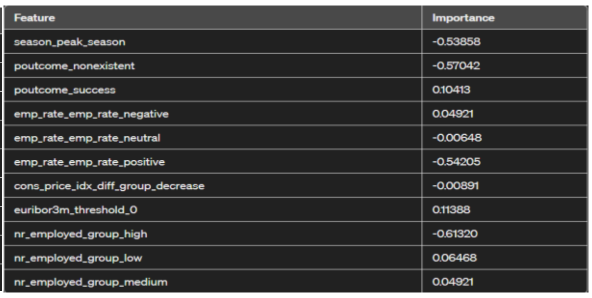
We are splitting the data into 70/30 ratio. As train and test sets respectively.

Feature Selection:

The next step the we proceeded is the feature selection in which we have selected the optimal features for our model. We have selected 20 random features using the chi-squared test, and transformed the dataset to only contain these features. Then we fit these features in to the Logistic Regression Model to calculate the coefficient of these features which gives us their importance.



Feature	Importance
client_id	2.4649e-06
age_Early-adult	-0.03596
age_senior	0.03758
blue_collar_jobs	-0.27059
default_unknown	-0.18263
contact_cellular	-0.17162
contact_telephone	-0.32770
season_moderate	0.00410
season_off_season	0.03516



Feature	Importance
season_peak_season	-0.53858
poutcome_nonexistent	-0.57042
poutcome_success	0.10413
emp_rate_emp_rate_negative	0.04921
emp_rate_emp_rate_neutral	-0.00648
emp_rate_emp_rate_positive	-0.54205
cons_price_idx_diff_group_decrease	-0.00891
euribor3m_threshold_0	0.11388
nr_employed_group_high	-0.61320
nr_employed_group_low	0.06468
nr_employed_group_medium	0.04921

Modelling:

Once the features has been selected we are applying them into the models to estimate their performance based on some metrics. The models that we used are:

'Gradient Boosting': GradientBoostingClassifier(),

'Random Forest':RandomForestClassifier(),

'Decision Trees': DecisionTreeClassifier(),

'Neural Networks': MLPClassifier(),

'K-Nearest Neighbors': KNeighborsClassifier(),

'Ridge': RidgeClassifier(),

'LogisticRegression':LogisticRegression()

And the performance metrics we are using to evaluate these models are:

'Accuracy': accuracy,

'Precision': precision,

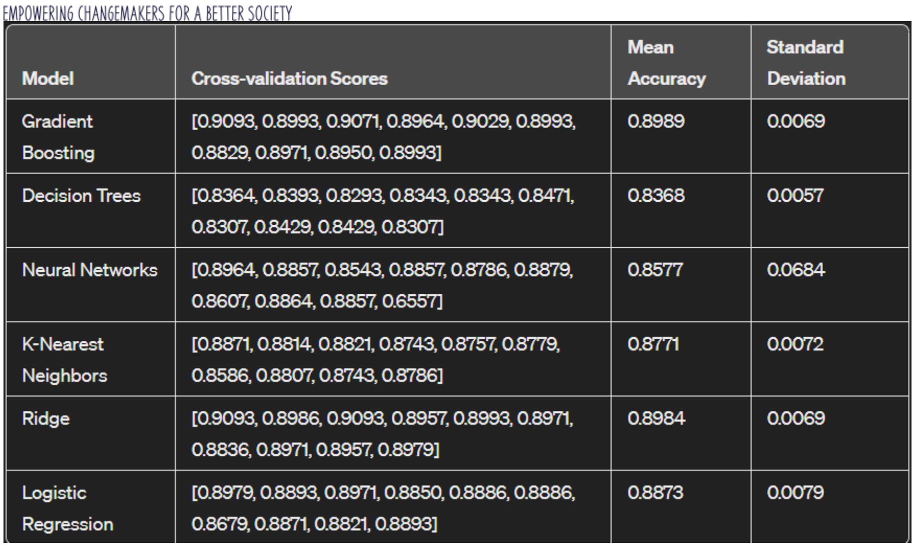
'Recall': recall,

'F1 Score': f1,

'Confusion Matrix': conf_matrix

Cross Validation :

We performed cross validation using K-fold. We have used 10 folds. using 10-fold cross-validation provides a more robust, generalized, and reliable estimate of a model's performance compared to a single train-test split. It helps in assessing the model's ability to generalize to unseen data, detecting overfitting, and tuning hyperparameters effectively.



Model	Cross-validation Scores	Mean Accuracy	Standard Deviation
Gradient Boosting	[0.9093, 0.8993, 0.9071, 0.8964, 0.9029, 0.8993, 0.8829, 0.8971, 0.8950, 0.8993]	0.8989	0.0069
Decision Trees	[0.8364, 0.8393, 0.8293, 0.8343, 0.8343, 0.8471, 0.8307, 0.8429, 0.8429, 0.8307]	0.8368	0.0057
Neural Networks	[0.8964, 0.8857, 0.8543, 0.8857, 0.8786, 0.8879, 0.8607, 0.8864, 0.8857, 0.6557]	0.8577	0.0684
K-Nearest Neighbors	[0.8871, 0.8814, 0.8821, 0.8743, 0.8757, 0.8779, 0.8586, 0.8807, 0.8743, 0.8786]	0.8771	0.0072
Ridge	[0.9093, 0.8986, 0.9093, 0.8957, 0.8993, 0.8971, 0.8836, 0.8971, 0.8957, 0.8979]	0.8984	0.0069
Logistic Regression	[0.8979, 0.8893, 0.8971, 0.8850, 0.8886, 0.8886, 0.8679, 0.8871, 0.8821, 0.8893]	0.8873	0.0079

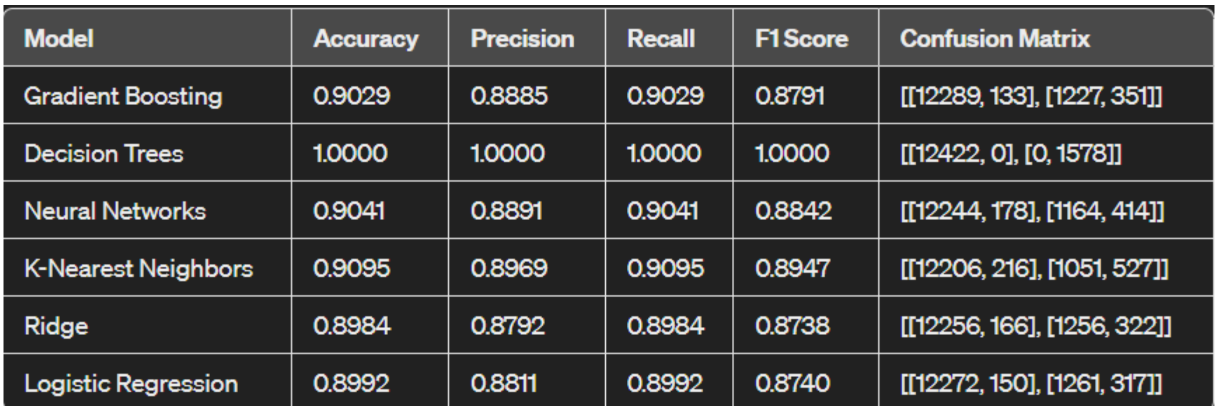
Based on the cross-validation results, K-Nearest Neighbors (KNN) stands out as the best-performing model among the ones evaluated. Here's why:

- Strong Mean Accuracy: KNN achieves a mean accuracy of 0.8771, which is among the highest across all models, indicating its effectiveness in classifying the data points.
- Consistent Performance: With a standard deviation of 0.0072, KNN demonstrates stable and consistent performance across different cross-validation folds, suggesting robustness.
- Simplicity and Interpretability: KNN is inherently simple and easy to understand, making it a practical choice for many classification tasks. Its straightforward nature allows for straightforward interpretation and implementation.

While Gradient Boosting and Ridge Regression also show competitive performance, KNN's strong mean accuracy, stability, and simplicity make it the best choice for this particular classification task based on the cross-validation results provided.

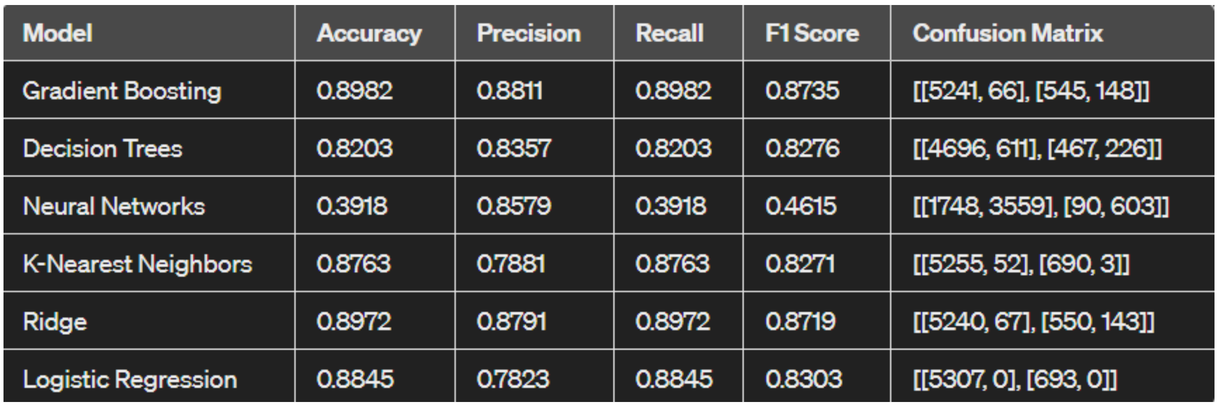
Benchmark Interpretation

Performance of models on Trainsets:



Model	Accuracy	Precision	Recall	F1 Score	Confusion Matrix
Gradient Boosting	0.9029	0.8885	0.9029	0.8791	[[12289, 133], [1227, 351]]
Decision Trees	1.0000	1.0000	1.0000	1.0000	[[12422, 0], [0, 1578]]
Neural Networks	0.9041	0.8891	0.9041	0.8842	[[12244, 178], [1164, 414]]
K-Nearest Neighbors	0.9095	0.8969	0.9095	0.8947	[[12206, 216], [1051, 527]]
Ridge	0.8984	0.8792	0.8984	0.8738	[[12256, 166], [1256, 322]]
Logistic Regression	0.8992	0.8811	0.8992	0.8740	[[12272, 150], [1261, 317]]

Performance of models on Test sets:



Model	Accuracy	Precision	Recall	F1 Score	Confusion Matrix
Gradient Boosting	0.8982	0.8811	0.8982	0.8735	[[5241, 66], [545, 148]]
Decision Trees	0.8203	0.8357	0.8203	0.8276	[[4696, 611], [467, 226]]
Neural Networks	0.3918	0.8579	0.3918	0.4615	[[1748, 3559], [90, 603]]
K-Nearest Neighbors	0.8763	0.7881	0.8763	0.8271	[[5255, 52], [690, 3]]
Ridge	0.8972	0.8791	0.8972	0.8719	[[5240, 67], [550, 143]]
Logistic Regression	0.8845	0.7823	0.8845	0.8303	[[5307, 0], [693, 0]]

- Gradient Boosting shows a slight drop in performance across all metrics but remains relatively stable.
- Decision Trees have a significant drop in accuracy and F1 score, indicating potential overfitting in the training phase.
- Neural Networks perform poorly on the test set compared to the training set, suggesting overfitting or insufficient generalization.
- K-Nearest Neighbors exhibits a minor drop in performance but still performs relatively well.
- Ridge and Logistic Regression show consistent performance between training and test sets, with Ridge slightly outperforming Logistic Regression.

Conclusion:

Given its consistent performance, high accuracy, balanced metrics, robustness to variations, and simplicity, K-Nearest Neighbors (KNN) stands out as the best model among the ones provided. While other models like Ridge Regression also show consistent performance, KNN's slightly higher accuracy and straightforward nature make it a preferable choice for this particular classification task.

Sources:

https://www.geeksforgeeks.org/understanding-logistic-regression/
https://cdn.analyticsvidhya.com/wp-content/uploads/2024/04/image-48.png
https://ml-explained.com/blog/logistic-regression-explained
https://www.r-bloggers.com/2015/09/how-to-perform-a-logistic-regression-in-r/
https://www.ibm.com/topics/decision-trees
https://upload.wikimedia.org/wikipedia/commons/thumb/4/43/Phi_Function_Tree.jpg/500px-Phi_Function_Tree.jpg
https://www.scaler.com/topics/decision-tree-in-r/
https://dhirajkumarblog.medium.com/
https://medium.com/@roiyeho/random-forests-98892261dc49
https://roboticsbiz.com/pros-and-cons-of-the-k-nearest-neighbors-knn-algorithm/
https://www.cloudflare.com/learning/ai/what-is-neural-network/
chat GPT & Professor session (SML).