Candace Edwards
ICS 635: Machine Learning
Final Project Report: **Data is a Drag**
Full Project Python Notebook: Link
Project Repo: https://github.com/CS-Edwards/ml_final

Introduction:

In this project we explore the performance of classical machine learning algorithms on small and imbalanced datasets. Here, we use machine learning to predict the winner of the international hit television series, RuPaul's Drag Race. At the time of writing, the series has completed its 15th season.

RuPaul's Drag Race is a competitive reality show where each season a number of queens compete to determine who will be crowned America's Next Drag Superstar. Throughout the course of the competition, contestants compete in challenges culminating in a runway look and lip sync elimination, until the winner is decided.

This project applies four classical machine learning algorithms to the show data to solve the binary classification problem of determining whether or not the contestant is the winner of their season.

The project compares the performance of each model to determine the best classifier for this dataset.

The full code for this project can be found in this repository[1] and accessed via the Google Colab notebook: here.

Machine Learning Related Work:

Binary classification is a supervised learning task where elements of the dataset are placed into one of two discrete categories. The goal of the classifier is to learn the appropriate relationship among the data input features in order to correctly classify new data points. In this project we use three classifiers: Logistic Regression, Naive-Bayes and Decision Trees, in addition to one ensemble method, Random Forest classification.

Logistic regression is a foundational tool in binary classification, widely used in numerous fields including medicine [1] [2], finance [3], and machine learning. Functionally, logistic regression takes the learned feature input weights and applies the sigmoid function to output a classification value of 1 or 0. The model learns the feature weights using gradient descent; which can be optimized using L1: Lasso Regularization, L2: Ridge Regularization or ElasticNet (hybrid). The purpose of gradient descent is to find optimal weights for the model, which minimize the loss function.

Ho and Wookey [4], explore variations of the cross-entropy loss function for logistic regression and their impact on imbalanced datasets. Given the class imbalance in our dataset, we explore the impact of cross-entropy loss (log loss) on overall model performance.

Naive Bayes and Decision Trees are also classifiers that can be used in binary classification. Functionally, Naive Bayes learns the probability distribution of the features in each class and then applies Bayes' theorem to calculate the probability of the class. Kim and Lee **[5],** highlight that 'naive Bayes (NB) sometimes fails at predicting minority instances owing to its sensitivity to class distribution'. Given the class imbalance of our dataset we explore the impact of the class_prior parameter on the accuracy of the model performance.

Decision Trees recursively split the data into subsets based on an impurity metric such as the Gini index or entropy. The goal is to minimize the impurity metric based on the feature choice, to arrive at the

[1] https://github.com/CS-Edwards/ml_final

purest subsample (leaf node). Once the tree is built, the new data point traverses the tree structure and is classified accordingly. Chaabane, Guermazi and Hammami **[6]**, note that " in imbalanced data distributions, standard pruning procedures may remove branches describing the minority concept which hinders performance towards the focus class". In other words, pruning as an approach to manage overfitting in decision trees may have a detrimental impact on the tree's ability to classify minority classes in an imbalanced dataset.

In our approach to the data we explore the impact of the class_weight parameter to deal with imbalance. Additionally, we use the ensemble method, Random Forest to improve the generalization of the Decision Tree model.

Bal et al **[7]**, compares the performance of several classifiers on progressively larger datasets including: Naive Bayes, Logistic Regression, Decision Tree and Random Forest. In their experiment with a dataset of size 100 samples, the relevant algorithms ranked in descending order of average performance are: Simple Logistic Regression, Naive Bayes, Decision Tree and Random Forest (see Appendix A).

Ala'raj,Majdalawieh & Abbod**[8]**, propose using k-nearest neighbors to filter parameter selection for classification models, prior to utilizing Logistic Regression, Naive Bayes, Decision Trees, Random Forest and other classifiers to improve model performance. According to their report, using k-nn successfully resulted in increased accuracy from previously 'low-accuracy' datasets; however given the small size of our dataset with relatively few parameters, this method was not applied in this project.

Working with small datasets in machine learning poses several challenges. Given the size of the dataset, the model has limited training examples to learn from, which increases the risk of overfitting. In addition to optimization via hyperparameter tuning, cross validation can also be used to manage overfitting.

Cross validation can be used for both feature selection, and model selection [9].A common cross validation technique, k-fold cross validation. In k-fold, the data is split into *k-number* subgroups. In each iteration one subgroup is withheld while the model is trained on the other groups. And the withheld subgroup is used for testing. Stratified K-fold, a modification to k-fold, ensures a representative sample of each class in the fold. Stone[10], proposed leave-one-out cross-validation (LOOCV), for estimating model parameters. In LOOCV, one sample is omitted from the training set and the omitted sample is used for testing. The LOOCV approach ensures that each datapoint is used exactly once for model and training validation. This approach could be more effective on a small dataset given the granularity of the approach. In the k-fold approach, a data point will likely be used in both training and testing – unless the k=n, where n was the number of samples;which is equivalent to LOOCV. A drawback of cross validation approaches in general, and LOOCV specifically is the computational expense[11], where the runtime of the k-fold approach is O(k * n), and LOOCV is O(n), where n = number of samples. However, for this particular dataset of 184 samples, these approaches are well within computational feasibility, a benefit of a small dataset.

Data:

The data for this project is originally sourced from the {dragraceR} CRAN package, created by GitHub User SVMILLER[2]. The package contains three R Data Files, which provide episodic and contestant statistics from seasons 1-14 of RuPaul's Drag Race. Prior to modeling, the data was preprocessed in a Python notebook using the NumPy and Pandas libraries. During preprocessing numerical feature values were normalized (ie. rank per episode was

[2] https://github.com/svmiller/dragracer

averaged, challenging statistics are normalized for the season).

An original feature metric called the 'top-to-safe-proportion', captures top challenge performance placements relative to safe placements, where x=win, y=high, z=safe:

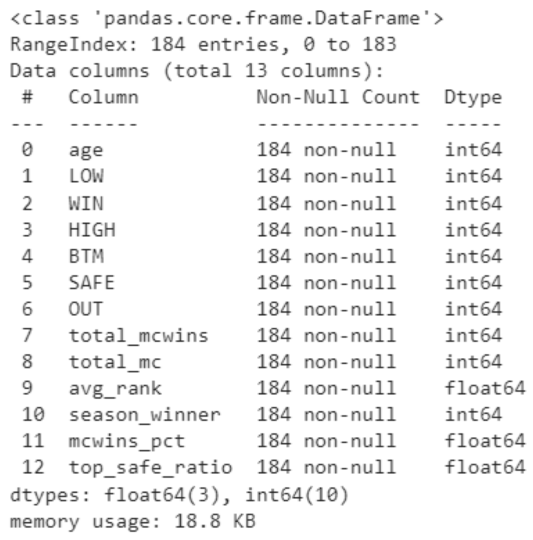
Equation 1. Top-to-Safe Proportion

$$f(x,y,z) = \begin{cases} \left(\frac{x+y}{x+y+z}\right) & \text{if } x+y+z>0 \\ -1 & \text{if } x+y+z=0 \end{cases}$$

The feature data set consists of normalized episodic contest data, the label array is a binary value {0,1} where 1 indicates the winner of the season. The winner class represents <8% of the label data, given that there is only one winner each season, our data classes are naturally imbalanced.

The final dataframe for the models contains 184 samples, 13 features and is ~18.8kb.

Figure 1.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 184 entries, 0 to 183
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   age             184 non-null    int64
 1   LOW             184 non-null    int64
 2   WIN             184 non-null    int64
 3   HIGH            184 non-null    int64
 4   BTM             184 non-null    int64
 5   SAFE            184 non-null    int64
 6   OUT             184 non-null    int64
 7   total_mcwins    184 non-null    int64
 8   total_mc        184 non-null    int64
 9   avg_rank        184 non-null    float64
 10  season_winner   184 non-null    int64
 11  mcwins_pct      184 non-null    float64
 12  top_safe_ratio  184 non-null    float64
dtypes: float64(3), int64(10)
memory usage: 18.8 KB
```

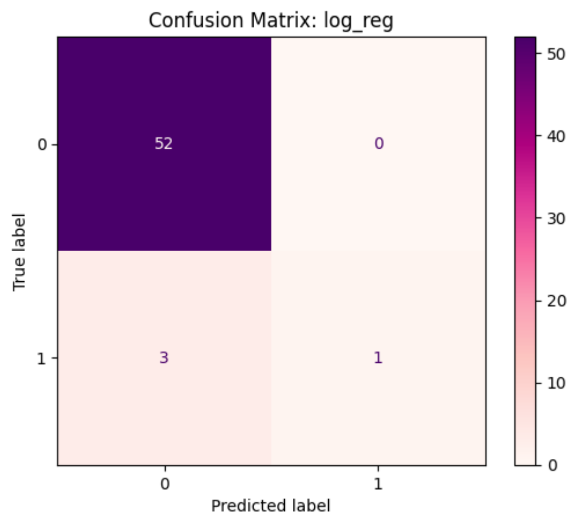
Methods:

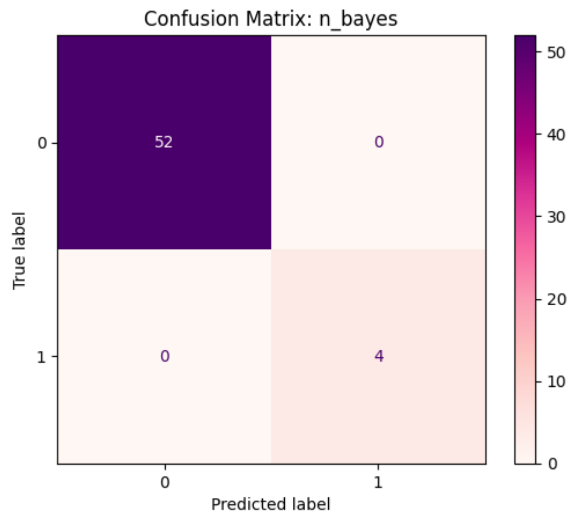
The full experiment code can be found in this repository[3] and accessed via the Google Colab notebook: here.

In the first experiment, we trained the model using vanilla (out-of-the-box) classifiers. For each classifier we created a confusion matrix and recorded the test performance of each model using the following metrics: accuracy, recall, precision, f1 and log-loss.

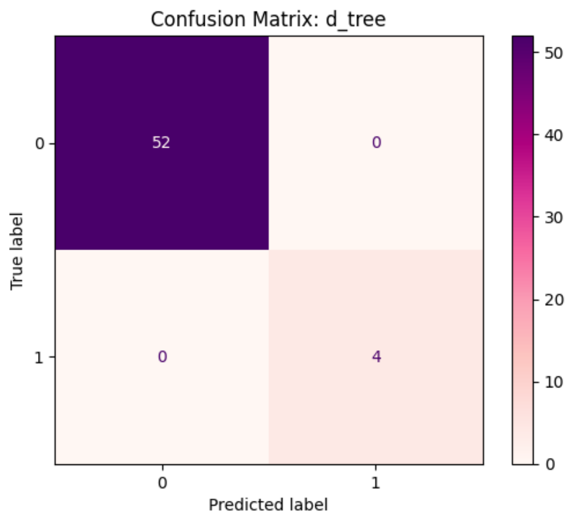
In part one of the second experiment, each classifier was tuned separately using GridSearchCV, and Stratified K-Fold Cross Validation where k=5. The F1 score was used as the scoring metric in GridSearchCV. Test data was applied to the classifiers best performing hyper parameters.

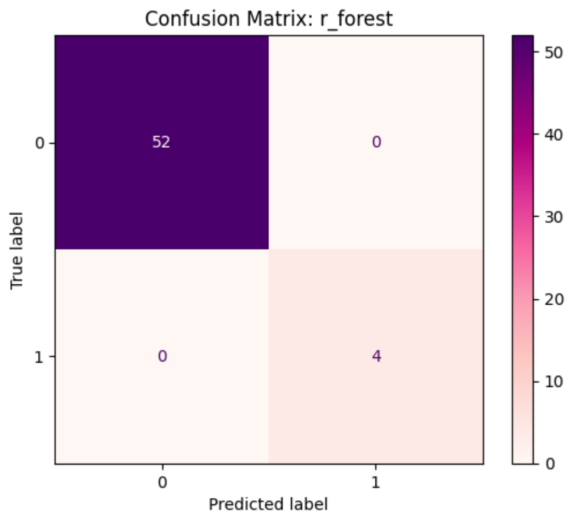
Experiment 1: Confusion Matrices by Model





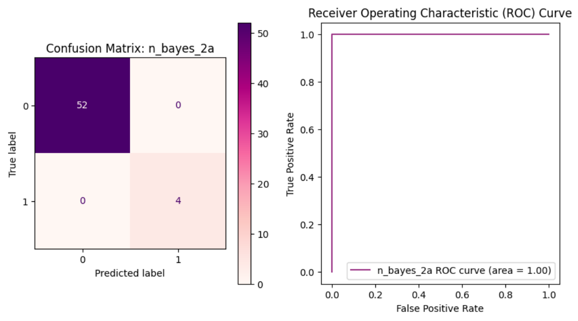
[3] https://github.com/CS-Edwards/ml_final





In part two of the second experiment, each classifier was tuned separately using GridSearchCV, and Leave One Out Cross Validation. Again, the F1 score was used as the scoring metric in GridSearchCV. And the test data was applied to the classifiers best performing hyper parameters.

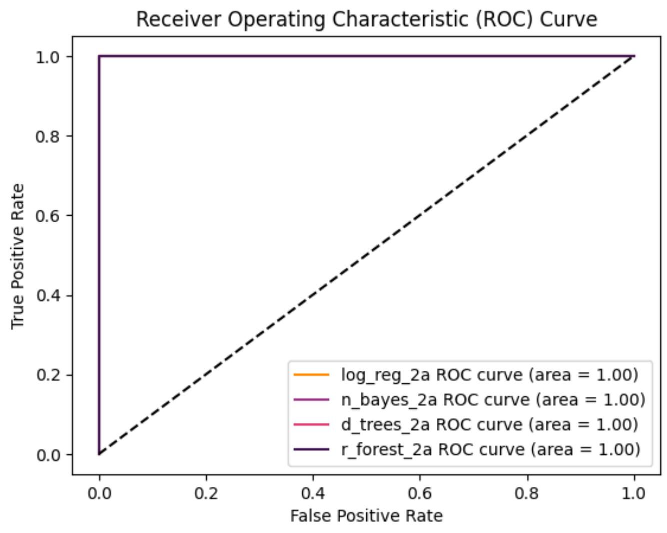
Experiment 2a : Cross Validation and Optimization - Naive Bayes



In the third and final experiment, we reset random state variable, and test size of the training and testing data. Using the most performative tuning in each classifier from experiment two, we re-trained and re-tested and recorded the performance results.

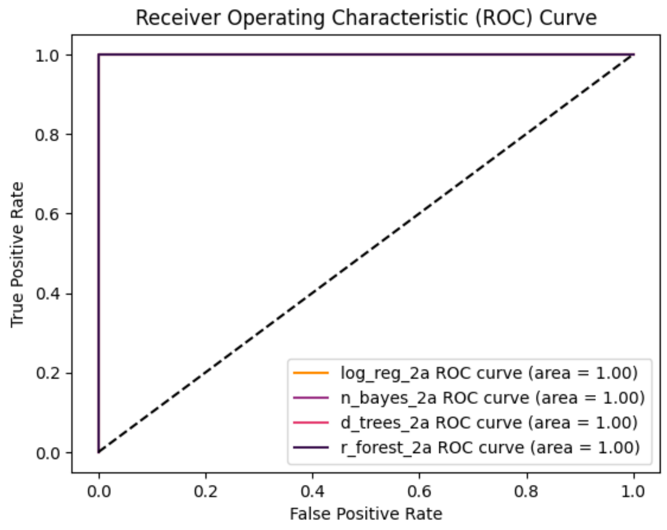
Results:
Summary results for each experiment can be found in Appendix B. The following figure shows the resulting ROC curve for all the classifiers tuned using stratified k-fold cross validation.

Experiment 2: K-Fold ROC - All Models



The subsequent figure shows the resulting curve for all the classifiers tuned using stratified LOOCV cross validation.

Experiment 2: LOOCV ROC - All Models



Note, in each ROC Curve there are four curves however they are overlapping.

Using log loss as the determinant, the result of Experiment 3 revealed that the most performant model overall for the dataset was the Decision Tree Classifier (see Appendix B). The Decision Tree classifier scored a 1.0 in accuracy, precision, recall and f1, and produced the lowest log loss of the classifiers. According to the Decision Tree, the most important features of the dataset were ‘OUT’, a feature which indicated if a contestant had left the competition, and ‘avg_rank’, the average rank of the performer for their season.

Discussion & Conclusion:

This project applied four classical machine learning algorithms to the source dataset: Logistic Regression, Naive Bayes, Decision Trees and Random Forest. The first experiment utilized the default hyperparameter settings for each model in the Sci-kit Learn library. The second experiment performed model specific hyperparameter tuning for each classifier, using first Stratified K-Fold Cross Validation, then Leave One Out Cross Validation – holding all else equal. The third experiment provided a model comparison to identify the best performing model on the dataset.

The greatest challenge of this dataset was the small size. We can see in each ROC curve , all models achieved an area =1, indicating a perfect classification; however, overfitting cannot be ruled out.

A larger dataset would help improve understanding of the impact of hyperparameter tuning of each classifier. Future work on this dataset can include increasing the number of observations by including data from dozen-plus international Drag Race spinoffs.

Additionally, analysis of the contest judge critiques using NLP to create a sentiment analysis metric, could also have an impact on the model. However, at this time this is limited transcription data available for series episodes.

[INTENTIONALLY BLANK]

Project Code:
Full project coding notebook: Link.

APPENDIX A:
Coding notebook for data from Bal et al. [7]. Link.

APPENDIX B: Results:
Summary results by experiment.

Experiment 1: Vanilla Classifiers
```
           log_reg        d_tree       n_bayes  r_forest
Accuracy   0.946429  1.000000e+00  1.000000e+00  1.000000
Recall     0.250000  1.000000e+00  1.000000e+00  1.000000
Precision  1.000000  1.000000e+00  1.000000e+00  1.000000
F1         0.400000  1.000000e+00  1.000000e+00  1.000000
Log-loss   0.070506  2.220446e-16  7.649225e-08  0.039307
```

Experiment 2a: Stratified K-Fold Cross Validation
```
           log_reg_2a    n_bayes_2a     d_tree_2a  r_forest_2a
Accuracy     0.910714  1.000000e+00  1.000000e+00     1.000000
Recall       1.000000  1.000000e+00  1.000000e+00     1.000000
Precision    0.444444  1.000000e+00  1.000000e+00     1.000000
F1           0.615385  1.000000e+00  1.000000e+00     1.000000
Log-loss     0.407539  7.649225e-08  2.220446e-16     0.037536
```

Experiment 2b: Leave One Out Cross Validation
```
           log_reg_2b    n_bayes_2b     d_tree_2b  r_forest_2b
Accuracy     0.875000  1.000000e+00  1.000000e+00     0.982143
Recall       1.000000  1.000000e+00  1.000000e+00     0.750000
Precision    0.363636  1.000000e+00  1.000000e+00     1.000000
F1           0.533333  1.000000e+00  1.000000e+00     0.857143
Log-loss     0.397526  7.649225e-08  2.220446e-16     0.040056
```

Experiment 3: Best Overall Classifier
```
           log_reg        d_tree       n_bayes  r_forest
Accuracy   0.975904  1.000000e+00  1.000000e+00  0.987952
Recall     0.714286  1.000000e+00  1.000000e+00  0.857143
Precision  1.000000  1.000000e+00  1.000000e+00  1.000000
F1         0.833333  1.000000e+00  1.000000e+00  0.923077
Log-loss   0.060990  2.220446e-16  5.603559e-08  0.044188
```

Results by Model: Logistic Regression
```
           log_reg_1  log_reg_2a  log_reg_2b
Accuracy    0.946429    0.910714    0.875000
Recall      0.250000    1.000000    1.000000
Precision   1.000000    0.444444    0.363636
F1          0.400000    0.615385    0.533333
Log-loss    0.070506    0.407539    0.397526
```

```
Results by Model:  Naive Bayes
           n_bayes_1    n_bayes_2a    n_bayes_2b
Accuracy   1.000000e+00  1.000000e+00  1.000000e+00
Recall     1.000000e+00  1.000000e+00  1.000000e+00
Precision  1.000000e+00  1.000000e+00  1.000000e+00
F1         1.000000e+00  1.000000e+00  1.000000e+00
Log-loss   7.649225e-08  7.649225e-08  7.649225e-08
```

```
Results by Model: Decision Trees
            d_tree_1     d_tree_2a     d_tree_2b
Accuracy   1.000000e+00  1.000000e+00  1.000000e+00
Recall     1.000000e+00  1.000000e+00  1.000000e+00
Precision  1.000000e+00  1.000000e+00  1.000000e+00
F1         1.000000e+00  1.000000e+00  1.000000e+00
Log-loss   2.220446e-16  2.220446e-16  2.220446e-16
```

```
Results by Model: Random Forest
          r_forest_1  r_forest_2a  r_forest_2b
Accuracy     1.000000     1.000000     0.982143
Recall       1.000000     1.000000     0.750000
Precision    1.000000     1.000000     1.000000
F1           1.000000     1.000000     0.857143
Log-loss     0.039307     0.037536     0.040056
```

References:

[1] Harris, J.K., "Primer on binary logistic regression," Fam Med Community Health, vol. 9, no. Suppl 1, pp. e001290, Dec. 2021. doi: 10.1136/fmch-2021-001290. PMID: 34952854; PMCID: PMC8710907. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8710907/

[2] Huang Y, Pepe MS, Feng Z. LOGISTIC REGRESSION ANALYSIS WITH STANDARDIZED MARKERS. Ann Appl Stat. 2013 Sep 1;7(3):10.1214/13-AOAS634SUPP. doi: 10.1214/13-AOAS634SUPP. PMID: 24204441; PMCID: PMC3817965. [Online]. Available:https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3817965/

[3] Costa E Silva E, Lopes IC, Correia A, Faria S. "A logistic regression model for consumer default risk." J Appl Stat. 2020 May 5;47(13-15):2879-2894. doi: 10.1080/02664763.2020.1759030. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9041570/

[4] Ho, Yaoshiang & Wookey, Samuel. (2019). The Real-World-Weight Cross-Entropy Loss Function: Modeling the Costs of Mislabeling. IEEE Access. PP. 1-1. 10.1109/ACCESS.2019.2962617. [Online]. Available: https://www.researchgate.net/publication/338205326_The_Real-World-Weight_Cross-Entropy_Loss_Function_Modeling_the_Costs_of_Mislabeling

[5] Kim, T., & Lee, J.-S. (2023). Maximizing AUC to learn weighted naive Bayes for imbalanced data classification. Expert Systems with Applications, 217, 119564. https://doi.org/10.1016/j.eswa.2023.119564

[6] I. Chaabane, R. Guermazi and M. Hammami, "Adapted pruning scheme for the framework of imbalanced data-sets," Procedia Computer Science, vol. 112, pp. 1542-1553, 2017, doi: 10.1016/j.procs.2017.08.060.

[7] M. Bal, M.F. Amasyali, H. Sever, G. Kose and A. Demirhan, "Performance evaluation of the machine learning algorithms used in inference mechanism of a medical decision support system," ScientificWorldJournal, vol. 2014, p. 137896, 2014. doi: 10.1155/2014/137896. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4177776/

[8] Ala'raj, M., Majdalawieh, M. & Abbod, M.F. Improving binary classification using filtering based on k-NN proximity graphs. J Big Data 7, 15 (2020). https://doi.org/10.1186/s40537-020-00297-7 [Online]. Available: https://journalofbigdata.springeropen.com/articles/10.1186/s40537-020-00297-7

[9] S. Arlot and A. Celisse, "A survey of cross-validation procedures for model selection," 2010, pp. 40-79.[Online]. Available: https://projecteuclid.org/journals/statistics-surveys/volume-4/issue-none/A-survey-of-cross-validation-procedures-for-model-selection/10.1214/09-SS054.full

[10] M. Stone, "Cross-Validatory Choice and Assessment of Statistical Predictions," Journal of the Royal Statistical Society. Series B (Methodological), vol. 36, no. 2, pp. 111-147, 1974. [Online]. Available: http://www.jstor.org/stable/2984809.

[11] H. Cheng, D. J. Garrick, and R. L. Fernando, "Efficient strategies for leave-one-out cross validation for genomic best linear unbiased prediction," J. Anim. Sci. Biotechnol., vol. 8, p. 38, May 2017. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5414316/.