EE559: Final Project

NBA Players Dataset - Regression & Classification

Shauryasikt Jena, jenas@usc.edu

May 9, 2023

1 Abstract

I worked on selective datasets from the NBA Players Datasets. I implemented Multinomial Ridge Regressor, RBF Networks, SVMs, KNN, Bayesian Regressor, and Neural Networks for the regression problem targeting the average points, rebounds, and assists of each player in a season. The metric used to evaluate these models was RMSE, and MAE and R2-score for each are also reported. For comparison between models, the R2-score is favored as it measures how well a model captures the variance of unseen test data.

Further, I implemented Multinomial Ridge Classifier, SVMs, KNN, Naive Bayes Classifier, and Neural Networks for the classification problem targeting the draft round and draft number of a player in the given season. I modeled the draft numbers into bins of tens for ease of classification, while minimizing the compromise of oversimplifying the outputs. The metric used to evaluate these models was Accuracy, and Macro-F1 scores and Confusion Matrices are also reported for each model. For comparison between models, the Macro-F1 score is favored as it does not favor classes with the majority of data samples but is a fair measure of correctness over all classes.

Upon empirical evidence, the use of k-fold cross-validation was discarded as the results did not show significant improvement. Most models are directly implemented from Scikit-learn with the code performing hyperparameter tuning and metric evaluation for model selection. The full project thoroughly covers almost all topics covered in class except for dimension-reduction techniques and parametric estimates.

Keywords: NBA Players Dataset, Regressor, Classifier, Multinomial Ridge, RBF, SVM, KNN, Bayesian, Naive Bayes, Neural Networks, RMSE, MAE, R2-score, Accuracy, Macro-F1 score, Confusion Matrix, Hyperparameters, Scikit-learn

Key: RBF - Radial Basis Function, SVM - Support Vector Machine, KNN - K-Nearest Neighbors, RMSE - Root Mean Squared Error, MAE - Mean Absolute Error

2 Introduction

The dataset(s) chosen for this project is the NBA Players Dataset, upon which two regression and one classification tasks were expected to be performed. The regression tasks include the prediction of players' average points, rebounds, and assists in a year based on - (1) Demographic Attributes, and (2) Statistical Figures from past seasons. The classification task is to predict the players' draft rounds and draft numbers in a given year.

There are five datasets in the collection - (1) Super-set Data from 1996-97 to 2021-22, (2) Data from 1996-97 to 2019-20 seasons, (3) Data from 2020-21 and 2021-22 seasons, (4) Roughly 80% samples the Super-Set, and (5) Roughly 20% samples from the Super-set. The last two are intended as a provided train-test split on the complete data.

2.1 Problem Assessment and Goals

For all problems, a player's name does not contribute any unique information that can not be covered by their demographic attributes (described in 2.1.1), so the name column was dropped for all training

and testing purposes.

Also, the draft year and given season convey how long a player has been playing. So the two columns were combined by taking the difference between the latter year of the given season and the draft year, into a new feature titled 'Experience' in years.

2.1.1 Regression based on Demographic Attributes

I considered all the features that do NOT count toward a player's career figures as their demographic attributes, namely - team, age, height, weight, college, country, draft round, and draft number. Out of these, only age, height, and weight are numerical data and the rest are categorical. Based only on these attributes, I had to predict a player's average points, rebounds, and assists for the given sample.

As it was possible some categorical information (such as one country) may be missed from either the train or the test set, and empirically observed from datasets (4) and (5), the super-set (1) was chosen to be preprocessed and split into train and test datasets for model evaluations.

2.1.2 Regression based on Statistical Figures

I considered all the features that count toward a player's career figures as their statistical figures, namely - games played, net rating, % offensive rebounds, % defensive rebounds, % team plays, % shooting efficiency, % assists, and Experience. All of these are numerical data. Based only on these figures, I had to predict a player's average points, rebounds, and assists for the given sample.

Here, it was perfectly acceptable to use the provided datasets (2) and (3) for separation into past and current seasons as no information loss is risked.

2.1.3 Classification

As the classification tasks of predicting the draft round and draft number of a player are dependent on other categorical information, I again chose to split the provided super-set (1) instead of using datasets (4) and (5) to avoid loss of information universally.

2.2 Literature Review

I have not explored any approaches to the NBA Players Datasets, however, I have relied on the inbuilt model classes from Scikit-learn to evaluate the performances of my applied Machine Learning techniques. Each method will be detailed in its corresponding subsection.

3 Approach and Implementation

As instructed, I have implemented one or more models from the following categories for the regression problems - (1) Non-probabilistic: Multinomial Ridge, RBF Network, (2) Support Vector Regressor, (3) Probabilistic: KNN, Regression with Gaussian priors, and (4) Neural Network. The loss metrics evaluated and reported are RMSE, MAE, and R2-Score and comparisons were primarily derived from RMSE and R2-Score.

I have implemented one or more models from the following categories for the classification problems - (1) Non-probabilistic: Multinomial Ridge, RBF Network, (2) Support Vector Classifier, (3) Probabilistic: KNN, Naive Bayes, Complement NB, and (4) Neural Network. The performance metrics evaluated and reported are Accuracy, Macro-F1 Score, and Confusion Matrix and comparisons were primarily derived from Accuracy and Macro-F1 (discussed later).

3.1 Dataset Usage

The full dataset, and the pre-seasons and recent seasons datasets have 21 attributes including the target variables. Following primary data cleaning involving feature engineering (3.2), I used the full dataset for a random 80-20 train-test split resulting in 9844 training samples and 2461 test samples

in regression on demographic attributes (3.1.1) and classification (3.1.3). The pre-seasons and recent seasons datasets were used respectively as trainset - 11160 samples, and testset - 1145 samples in regression on statistical figures (3.1.2), after data cleaning.

For validation splits in cross-validation, an 80-20 split was used to maintain train-test similarity during model performance evaluation. However, cross-validation provided empirically similar results to a simple validation split, and its use was discarded in the code for all models except the first to save computation costs and time for code reproducibility. Preprocessing (3.3) was performed after all data splits to prevent data snooping and cultivate the models to train on the ability to capture qualities of unseen data during validation and testing.

3.1.1 Regression - Demographic Attributes

The target variables are - (1) pts: average points scored, (2) rebs: average rebounds grabbed, and (3) ast: average assists distributed. After assigning these as target labels each for separate regression models, the attributes of players that contribute to season statistics were dropped from the training. The column with player names was also dropped as other given demographic attributes are sufficient to uniquely identify the players' information as vectors. The final training features are - team, age, height, weight, college, country, draft round, and draft number.

After the best trainable model for each technique with tuned hyperparameters was chosen based on the least RMSE, it was evaluated on the test set. Aside from the trivial system and two baseline systems (Linear Regression and 1-NN), the test set was used for the evaluation of 6 regression models.

3.1.2 Regression - Statistical Figures

For this task too, the target variables are - (1) pts: average points scored, (2) rebs: average rebounds grabbed, and (3) ast: average assists distributed. After assigning these as target labels each for separate regression models, only the attributes of players that do contribute to season statistics were included in the training. The column with player names was also dropped to encourage statistical similarity between similar feature vectors. The final training features are - games played, net rating, % offensive rebounds, % defensive rebounds, % team plays, % shooting efficiency, % assists, and Experience (in years).

After the best trainable model for each technique with tuned hyperparameters was chosen based on the least RMSE, it was evaluated on the test set. Aside from the trivial system and two baseline systems (Linear Regression and 1-NN), the test set was used for the evaluation of 6 regression models.

3.1.3 Classification

The target variables are - (1) draft round and (2) draft number. The column with player names was also dropped as other given demographic attributes are sufficient to uniquely identify the players' information as vectors.

After the best trainable model for each technique with tuned hyperparameters was chosen based on the best accuracy, it was evaluated on the test set. Aside from the trivial system and the baseline system (Nearest Means), the test set was used for the evaluation of 7 models.

3.2 Data Cleaning, Feature Engineering, and Dimensionality

Firstly, the datasets were checked for missing values and none were reported so this problem did not have to be handled.

The draft year and given season convey how long a player has been playing. So the two columns were combined by taking the difference between the latter year of the given season and the draft year, into a new feature titled 'Experience' in years. The draft year and season features were dropped.

As ML models traditionally work on numerical inputs, all categorical training features were one-hot encoded instead of label encoded, to prevent favorable rankings of different categorical classes.

3.2.1 Regression

The 'Undrafted' values in the draft round and draft number category were set to 0 so as not to initialize the effect of any weights for such players in training. Then, the draft numbers were binned in groups of 5 starting from 1 to categorize them into the stage of the draft. Then these two features were treated as categorical and also one-hot encoded for training. The number of one-hot features is 499.

3.2.2 Classification

The draft round and draft number are the targets for the classification problem. To maintain the integrity of this task, the draft numbers were binned in groups of 10 starting from 1-10 for classification, and each bin was assigned the value of its highest number divided by 10.

As undrafted would be desired lower in hierarchy than the last drafted person, draft round and draft number were set to the highest value of their respective data and put in the numerical class that would come after it. Mathematically expressed:

$$\begin{aligned} New\ Round &= Round, && Round \neq\ 'Undrafted' \\ &= 1 + max\{Rounds\}, && Round =\ 'Undrafted' \end{aligned} \tag{1}$$

$$\begin{aligned} New\ Number &= Number, && Number \neq\ 'Undrafted' \\ &= 10 + max\{Numbers\}, && Number =\ 'Undrafted' \end{aligned} \tag{2}$$

$$Draft\ Stage\ Bin = \lfloor New\ Number/10 \rfloor 10 \tag{3}$$

The draft stage bins were then label-encoded. The number of one-hot encoded training features is 465.

Dimensionality adjustment techniques were avoided as for demographic or all features, one-hot encoding generates a largely sparse portion of the dataset which yet contains instrumental information that cannot be risked being flattened. Further, for purely statistical data, there are very few dimensions from the start, and regularizations have been implemented ahead for appropriate feature importances in models.

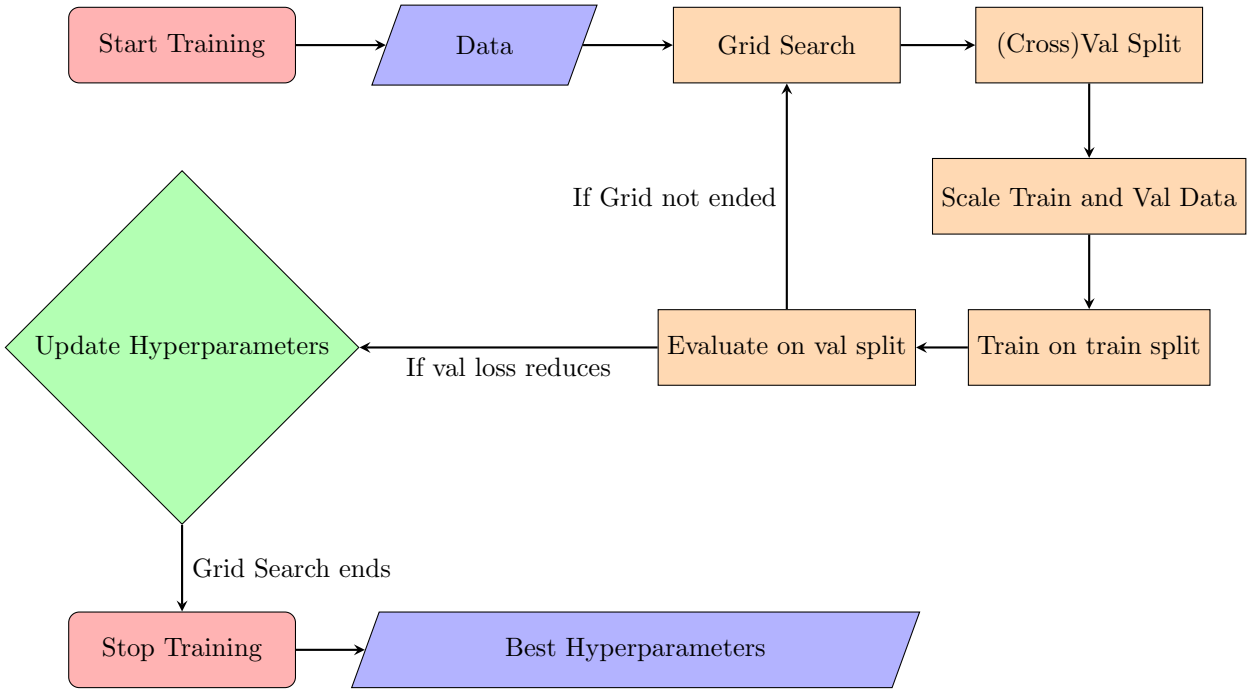
3.3 Preprocessing

All categorical information in training was one-hot encoded. For initial numerical data, the columns were normalized by making them zero-mean and scaling them on the respective column ranges. Standard Scaling was not used as it tends to fit the data to a standard normal curve even if it might not be a good fit. The scaling used by me retains the distribution pattern of the initial data. For continuous and numeric data:

$$Column = (Column\ values - Column\ Mean)/Column\ Range \tag{4}$$

3.4 Training and Model Selection

A general idea of the training process implemented by me for all algorithms:



3.5 Regression - Demographic Attributes

I modeled the predictions of average points, rebounds grabbed, and assists distributed per season as three separate regression problems for each of the following techniques. All the hyperparameters were tuned with Grid Search.

For all the problems,

Number of constraints = 80% of training samples due to train-val split = 7875

3.5.1 Trivial System

For all target labels, the mean target value across all training samples was used as the prediction value. To gauge if an ML model is actually learning something during training, its loss should be less than the trivial system. Training Results:

Target	RMSE	MAE	R2-score
Points	5.949	4.729	0
Rebounds	2.479	1.905	0
Assists	1.785	1.308	0

The RMSE values here reflect the standard deviations of each target.

3.5.2 Baseline System - 1NN

This is an implementation of K-Nearest Neighbors regression with K=1, i.e., each sample predicts the target based on the target value of its nearest neighbor by Euclidean distance in the feature vectors' dimension. Training results:

Target	RMSE	MAE	R2-score
Points	0	0	1
Rebounds	0	0	1
Assists	0	0	1

The training perfectly fits over the data over which it is trained.

- Degrees of Freedom = No. of features = 502
- Number of constraints/D.o.F $\geq$ (3-10) : Yes

3.5.3 Baseline System - Linear Regression

This is an implementation of Linear Regression with no regularization. The target is modeled as a linear function of all features. Training results:

Target	RMSE	MAE	R2-score
Points	4.818	3.719	0.343
Rebounds	1.868	1.402	0.432
Assists	1.317	0.945	0.455

- Degrees of Freedom = No. of features = 502
- Number of constraints/D.o.F $\geq$ (3-10) : Yes

3.5.4 Multinomial Ridge Regression

A polynomial kernel of suitable degree was applied to the numerical features and then the new featureset was trained by Ridge Regression with L2 regularization with a suitable λ, i.e., the target is modeled as a linear combination of all features and their polynomials. 5-fold cross-validation was used, but it gave comparable results to a simple train-val split. Training results:

Target	RMSE	MAE	R2-score
Points	4.639	3.576	0.391
Rebounds	1.838	1.381	0.450
Assists	1.280	0.913	0.485

Target	Degree	λ
Points	7	0.1
Rebounds	4	3
Assists	4	0.1

- Degrees of Freedom = No. of modified features $\approx$ 2000
- Number of constraints/D.o.F $\geq$ (3-10) : Yes
- Grid Search Degrees = [4, 5, 6, 7]
- Grid Search λs = [0.1, 0.3, 1, 3, 10]

3.5.5 RBF Network

An RBF kernel with a suitable parameter γ was applied to the featureset that took each sample as a basis center. Then the modified featureset was passed through Ridge Regression with a suitable λ for linear modeling of the features for estimating the target. 5-fold cross-validation was used, but it gave comparable results to a simple train-val split. Training results:

Target	RMSE	MAE	R2-score
Points	1.591	0.975	0.928
Rebounds	0.661	0.386	0.928
Assists	0.412	0.230	0.946

Target	γ	λ
Points	1	0.001
Rebounds	1	0.001
Assists	1	0.001

- Degrees of Freedom = No. of modified features $\approx$ inf
- Number of constraints/D.o.F $\geq$ (3-10) : Yes
- Grid Search γs = [0.01, 0.1, 1]
- Grid Search λs = [0.001, 0.01, 0.1]

Cross-validation was discarded from here on.

3.5.6 Support Vector Regressor

A Support Vector Machine acts as a regressor here with the tube method. The model was trained with suitable γ and C parameters to determine the tightness of fitting and the number of support vectors. A simple train-val split was used. Training results:

Target	RMSE	MAE	R2-score
Points	2.043	1.040	0.881
Rebounds	0.864	0.454	0.878
Assists	0.448	0.232	0.936

Target	Kernel	γ	C
Points	rbf	1	30
Rebounds	rbf	0.3	30
Assists	rbf	1	30

- Degrees of Freedom = VC dimesnion $\leq \infty$
- Number of constraints/D.o.F $\geq$ (3-10): No, but SVMs are extremely adept at sparse data handling
- Grid Search kernels = [rbf, sigmoid]
- Grid Search γs = [0.1, 0.3, 1]
- Grid Search Cs = [1, 10, 30]

3.5.7 K-Nearest Neighbors

This is an implementation of the K-Nearest Neighbors Regressor with a suitable value of K, i.e.. the target is modeled as the mean of targets of K nearest feature vectors in the Euclidean space. Training results:

Target	RMSE	MAE	R2-score
Points	0	0	1
Rebounds	0	0	1
Assists	0	0	1

Target	K
Points	7
Rebounds	8
Assists	9

- Degrees of Freedom = No. of features = 502
- Number of constraints/D.o.F $\geq$ (3-10) : Yes
- Grid Search Ks = [2, 3, 4, 5, 6, 7, 8, 9, 10]

3.5.8 Bayesian Regression

This is an implementation of Bayesian Regression that considers multivariate Gaussian priors over the featureset for given targets. Training results:

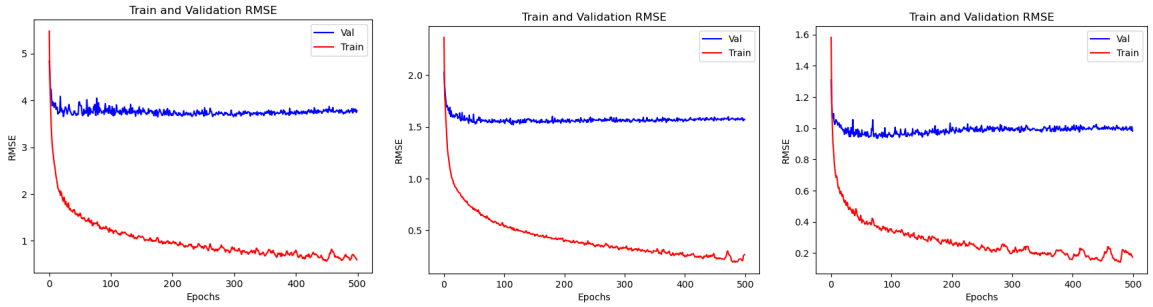
Target	RMSE	MAE	R2-score
Points	4.851	3.773	0.334
Rebounds	1.878	1.419	0.426
Assists	1.323	0.954	0.450

Target	α_1	α_2	λ_1	λ_2
Points	0.01	10^{-6}	10^{-6}	0.01
Rebounds	0.01	10^{-6}	10^{-6}	0.01
Assists	0.01	10^{-6}	10^{-6}	0.01

- Degrees of Freedom = No. of features = 502
- Number of constraints/D.o.F $\geq$ (3-10) : Yes
- Grid Search for all params: [10^{-6}, 10^{-4}, 10^{-2}]

3.5.9 Neural Network

This is an example of an MLP Regressor with a sequential architecture. The MLP approximates the curve in a piece-wise fashion through multiple linear models and non-linear activations. The architecture has two hidden layers with twice and the same times the nodes as input features respectively, as is the wont for a large number of features. All the activation layers are ReLU as this is a regression problem and SoftMax is not required. Batch size was chosen such that it gives the fastest convergence. Training results:



(a) Points (b) Rebounds (c) Assists

Figure 1: Training MLPs on Demographic Attributes

Target	RMSE	MAE	R2-score
Points	0.568	0.355	0.990
Rebounds	0.242	0.148	0.990
Assists	0.166	0.099	0.991

Target	Batch Size	η	λ
Points	128	0.01	0.001
Rebounds	128	0.001	0.001
Assists	128	0.01	0.0001

- Degrees of Freedom = $M_1 M_2 D \approx 1028096$
- Number of constraints/D.o.F $\geq$ (3-10): No, but it is fine as Neural Networks can adapt to sparse datasets
- Fastest Batch Sizes = [16, 32, 64, 128, 256]
- Grid Search ηs = [0.001, 0.01, 0.1]
- Grid Search λs = [0.0001, 0.001, 0.01, 0.1]

3.5.10 Analysis

Most models seem to be overfitting on the training data, so we need to evaluate the performance on the basis of test sets to choose the best model for this problem. Please note that I am not defining the models working in the next section as they are all the same models.

3.6 Regression - Statistical Figures

I modeled the predictions of average points, rebounds grabbed, and assists distributed per season as three separate regression problems for each of the following techniques. All the hyperparameters were tuned with Grid Search.

For all the problems,

Number of constraints = 80% of training samples due to train-val split = 8928

3.6.1 Trivial System

For all target labels, the mean target value across all training samples was used as the prediction value. To gauge if an ML model is actually learning something during training, its loss should be less than the trivial system. Training Results:

Target	RMSE	MAE	R2-score
Points	5.926	4.728	0
Rebounds	2.491	1.912	0
Assists	1.788	1.312	0

The RMSE values here reflect the standard deviations of each target.

3.6.2 Baseline System - 1NN

This is an implementation of K-Nearest Neighbors regression with K=1. Training results:

Target	RMSE	MAE	R2-score
Points	0	0	1
Rebounds	0	0	1
Assists	0	0	1

The training perfectly fits over the data over which it is trained.

- Degrees of Freedom = No. of features = 8
- Number of constraints/D.o.F $\geq$ (3-10) : Yes

3.6.3 Baseline System - Linear Regression

This is an implementation of Linear Regression with no regularization. Training results:

Target	RMSE	MAE	R2-score
Points	3.433	2.614	0.664
Rebounds	1.520	1.125	0.627
Assists	0.897	0.613	0.748

- Degrees of Freedom = No. of features = 8
- Number of constraints/D.o.F $\geq$ (3-10) : Yes

3.6.4 Multinomial Ridge Regression

A polynomial kernel of suitable degree was applied to the numerical features and then the new featureset was trained by Ridge Regression with L2 regularization with a suitable λ. 5-fold cross-validation was used, but it gave comparable results to a simple train-val split. Training results:

Target	RMSE	MAE	R2-score
Points	2.850	2.184	0.768
Rebounds	1.400	1.034	0.683
Assists	0.674	0.447	0.857

Target	Degree	λ
Points	6	3
Rebounds	6	10
Assists	5	1

- Degrees of Freedom = No. of modified features $\approx$ 4000
- Number of constraints/D.o.F $\geq$ (3-10) : Yes
- Grid Search Degrees = [3, 4, 5, 6, 7]
- Grid Search λs = [0.3, 1, 3, 10, 30]

3.6.5 RBF Network

An RBF kernel with a suitable parameter γ was applied to the featureset that took each sample as a basis center. Then the modified featureset was passed through Ridge Regression with a suitable λ. 5-fold cross-validation was used, but it gave comparable results to a simple train-val split. Training results:

Target	RMSE	MAE	R2-score
Points	2.683	2.054	0.794
Rebounds	1.251	0.917	0.747
Assists	0.681	0.451	0.854

Target	γ	λ
Points	0.223	0.01
Rebounds	0.223	0.01
Assists	0.223	0.01

- Degrees of Freedom = No. of modified features $\approx$ inf
- Number of constraints/D.o.F $\geq$ (3-10) : Yes
- Grid Search factor γs = [0.01, 0.1, 1]
- Grid Search λs = [0.001, 0.01, 0.1]

Cross-validation was discarded from here on.

3.6.6 Support Vector Regressor

A Support Vector Machine acts as a regressor here with the tube method. The model was trained with suitable γ and C parameters to determine the tightness of fitting and the number of support vectors. A simple train-val split was used. Training results:

Target	RMSE	MAE	R2-score
Points	3.100	2.234	0.727
Rebounds	1.287	0.900	0.733
Assists	0.684	0.433	0.853

Target	Kernel	γ	C
Points	rbf	3	0.1
Rebounds	rbf	3	0.1
Assists	rbf	3	0.1

- Degrees of Freedom = VC dimesnion $\leq \infty$
- Number of constraints/D.o.F $\geq$ (3-10): No, but SVMs are extremely adept handling sparse data
- Grid Search kernels = [rbf, sigmoid, polynomial]
- Grid Search γs = [0.3, 1, 3]
- Grid Search Cs = [0.01, 0.03, 0.1]

3.6.7 K-Nearest Neighbors

This is an implementation of the K-Nearest Neighbors Regressor with a suitable value of K. Training results:

Target	RMSE	MAE	R2-score
Points	0	0	1
Rebounds	0	0	1
Assists	0	0	1

Target	K
Points	15
Rebounds	15
Assists	15

- Degrees of Freedom = No. of features = 8
- Number of constraints/D.o.F $\geq$ (3-10) : Yes
- Grid Search Ks = [5, 10, 15, 20, 25, 30]

3.6.8 Bayesian Regression

This is an implementation of Bayesian Regression that considers multivariate Gaussian priors over the featureset and targets. Training results:

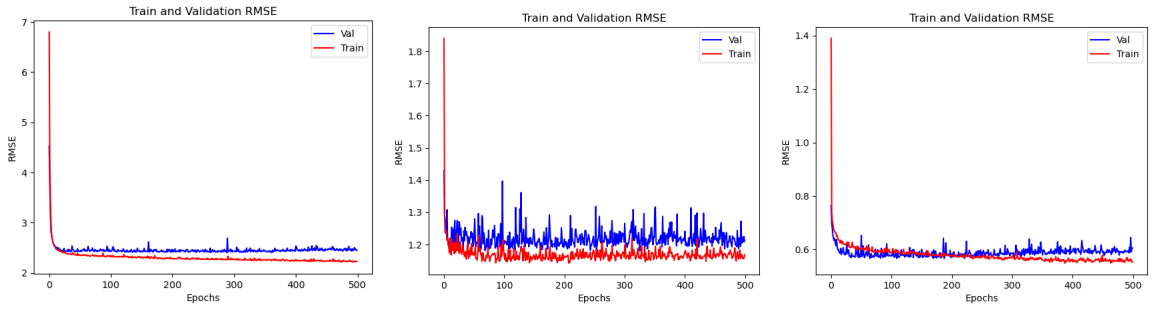
Target	RMSE	MAE	R2-score
Points	3.433	2.614	0.664
Rebounds	1.520	1.125	0.627
Assists	0.897	0.613	0.748

Target	α_1	α_2	λ_1	λ_2
Points	10^{-6}	1	1	1
Rebounds	10^{-6}	1	1	1
Assists	10^{-6}	1	1	10^{-6}

- Degrees of Freedom = No. of features = 8
- Number of constraints/D.o.F $\geq$ (3-10) : Yes
- Grid Search for all params: [10^{-6}, 10^{-4}, 10^{-2}, 1]

3.6.9 Neural Network

This is an example of an MLP regressor with a sequential architecture. The architecture has two hidden layers with 64 and 32 nodes respectively, as is the wont for small number of features. All the activation layers are ReLU as this is a regression problem and SoftMax is not required. Batch-size was chosen such that it gives fastest convergence. Training results:



(a) Points (b) Rebounds (c) Assists

Figure 2: Training MLPs on Statistical Figures

Target	RMSE	MAE	R2-score
Points	2.197	1.656	0.862
Rebounds	1.144	0.841	0.784
Assists	0.561	0.380	0.902

Target	Batch Size	η	λ
Points	256	0.01	0.01
Rebounds	256	0.1	0.01
Assists	128	0.01	0.01

- Degrees of Freedom = $M_1 M_2 D \approx 16384$
- Number of constraints/D.o.F $\geq$ (3-10): No, but it is fine as Neural Networks can adapt to sparse datasets
- Fastest Batch Sizes = [16, 32, 64, 128, 256]
- Grid Search ηs = [0.001, 0.01, 0.1]
- Grid Search λs = [0.0001, 0.001, 0.01, 0.1]

3.6.10 Analysis

The performance in these regression models is generally poorer than the ones trained on demographic attributes, implying a higher correlation between the targets (especially rebounds) and the demographic attributes than with statistical figures.

3.7 Classification

I modeled the predictions of draft rounds and draft numbers as two separate classification problems for each of the following techniques. All the hyperparameters were tuned with Grid Search.

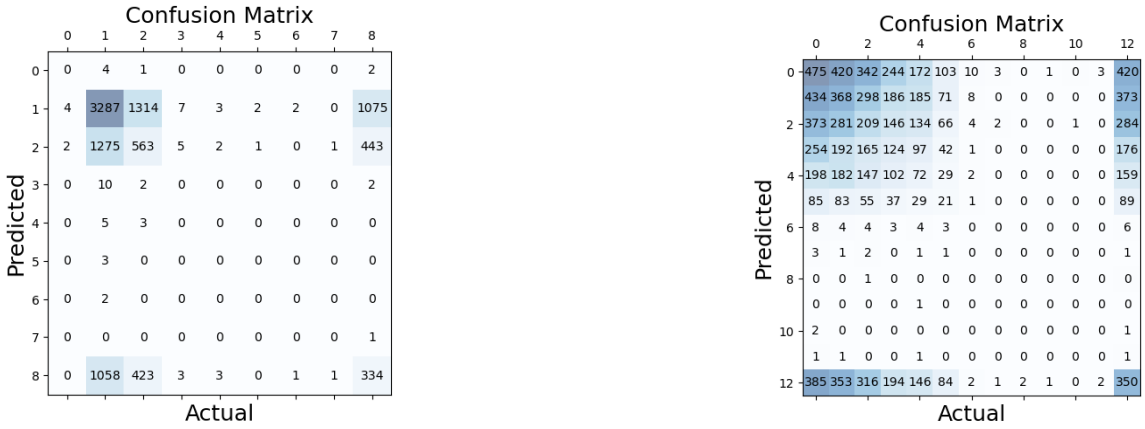
For all the problems,

Number of constraints = 80% of training samples due to train-val split = 7875

3.7.1 Trivial System

The predicted target label was randomly chosen with a chance equalling the prior of each label from the trainset. To gauge if an ML model is actually learning something during training, its accuracy should be greater than the trivial system. Training results:

Target	Accuracy	Macro-F1
Draft Round	0.425	0.111
Draft Stage	0.164	0.076



(a) Draft Round

(b) Draft Stage

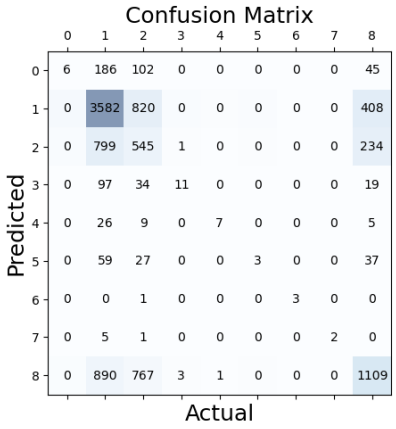
Figure 3: Training - Trivial

3.7.2 Baseline System - Nearest Means Classifier

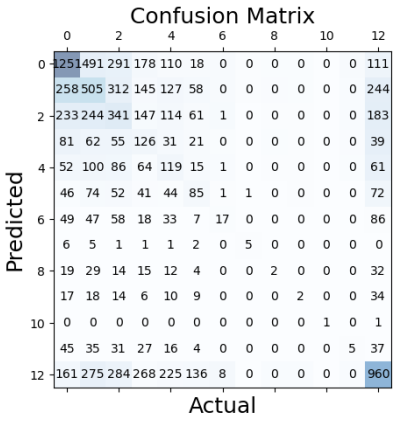
This is an implementation of multi-class nearest means classification, i.e., the feature vector is assigned to the class whose mean feature vector is closest to it in Euclidean space. Training results:

Target	Accuracy	Macro-F1
Draft Round	0.535	0.351
Draft Stage	0.347	0.255

- Degrees of Freedom = No. of features = 479
- Number of constraints/D.o.F $\geq$ (3-10) : Yes



(a) Draft Round



(b) Draft Stage

Figure 4: Training - NMC

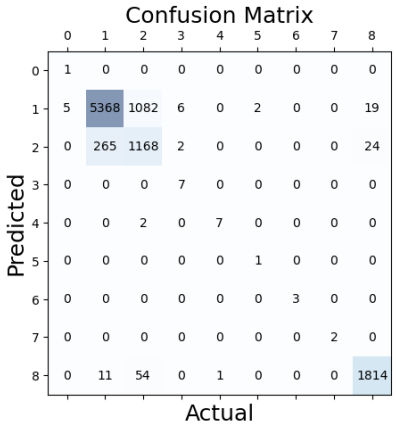
3.7.3 Ridge Classifier

A polynomial kernel of suitable degree was applied to the numerical features and then the new featureset was trained by Ridge Regression with L2 regularization with a suitable λ. A simple 80-20 train-val split was used. Then it was modeled as an OvR classifier for multiple classes. Training results:

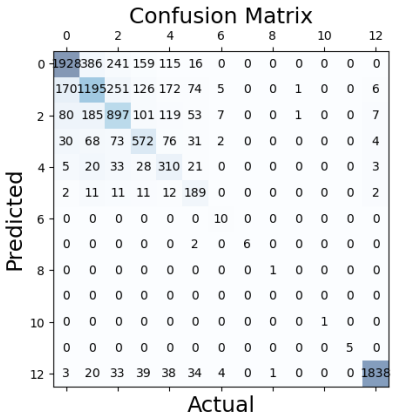
- Degrees of Freedom = No. of features $\approx$ 5000
- Number of constraints/D.o.F $\geq$ (3-10) : No, but regularization can handle the sparsity

Target	Accuracy	Macro-F1
Draft Round	0.850	0.746
Draft Stage	0.706	0.665

Target	Degree	λ
Draft Round	4	0.01
Draft Stage	4	0.001



(a) Draft Round



(b) Draft Stage

Figure 5: Training - Ridge

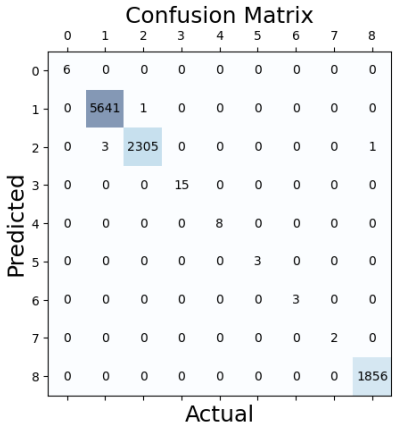
3.7.4 Support Vector Classifier

A Support Vector Machine acts as a classifier here, and fitted as an OvR classifier. The model was trained with suitable γ and C parameters to determine the tightness of fitting and the number of support vectors. A simple train-val split was used. Training results:

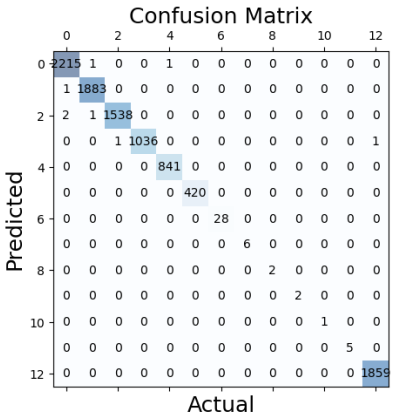
- Degrees of Freedom = VC dimesnion ≤ 5000
- Number of constraints/D.o.F $\geq$ (3-10) : No, but SVMs are extremely adept at handling sparse data
- Grid Search kernels = [rbf, sigmoid, polynomial]
- Grid Search degrees = [3, 4, 5]
- Grid Search γs = [0.3, 1, 3]
- Grid Search Cs = [0.01, 0.03, 0.1]

Target	Accuracy	Macro-F1
Draft Round	0.999	0.999
Draft Stage	0.999	0.999

Target	Kernel	Degree	γ	C
Draft Round	poly	4	3	0.01
Draft Stage	poly	4	3	0.01



(a) Draft Round



(b) Draft Stage

Figure 6: Training - SVC

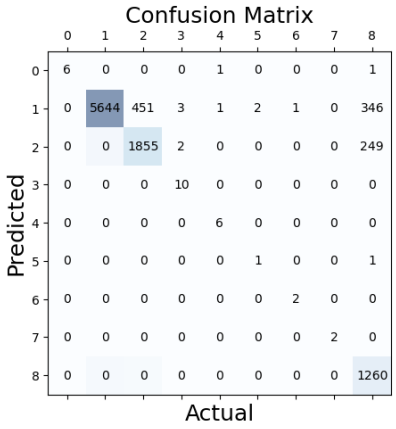
3.7.5 K-Nearest Neighbors

This is an implementation of the K-Nearest Neighbors Classifier with a suitable value of K. Training results:

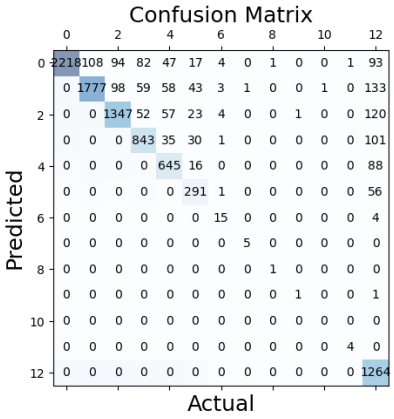
- Degrees of Freedom = No. of features = 479
- Number of constraints/D.o.F $\geq$ (3-10) : Yes
- Grid Search Ks = [2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20]

Target	Accuracy	Macro-F1
Draft Round	0.892	0.810
Draft Stage	0.854	0.726

Draft Round	K
Draft Stage	2
Rebounds	2



(a) Draft Round



(b) Draft Stage

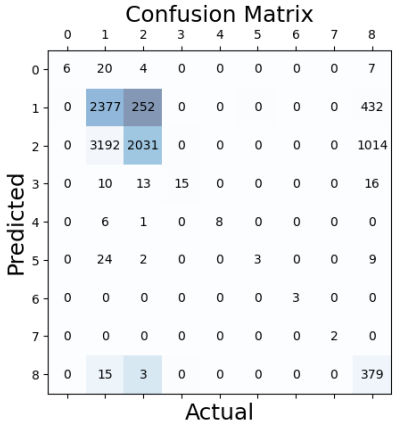
Figure 7: Training - KNN

3.7.6 Naive Bayes

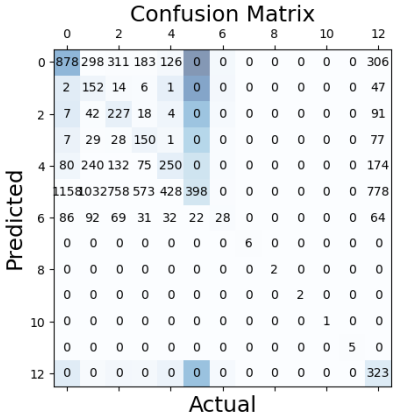
This is an implementation of Bayesian Classification that considers univariate Gaussian priors over each feature from the featureset and targets. Training results:

- Degrees of Freedom = No. of features = 479
- Number of constraints/D.o.F $\geq$ (3-10) : Yes

Target	Accuracy	Macro-F1
Draft Round	0.490	0.545
Draft Stage	0.246	0.525



(a) Draft Round



(b) Draft Stage

Figure 8: Training - Naive Bayes

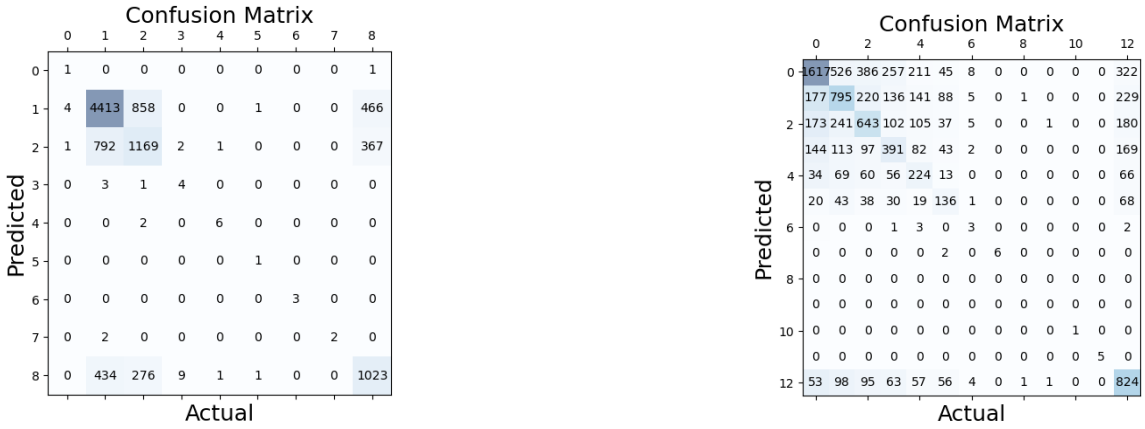
3.7.7 Complement Naive Bayes

This is an implementation of Bayesian Classification that considers multivariate Gaussian priors over all features from the complement featureset of the given feature and targets[1][2]. This is well-suited for imbalanced datasets like the one we have been given. Training results:

- Degrees of Freedom = No. of features = 479

- Number of constraints/D.o.F $\geq$ (3-10) : Yes

Target	Accuracy	Macro-F1
Draft Round	0.672	0.595
Draft Stage	0.471	0.464



(a) Draft Round

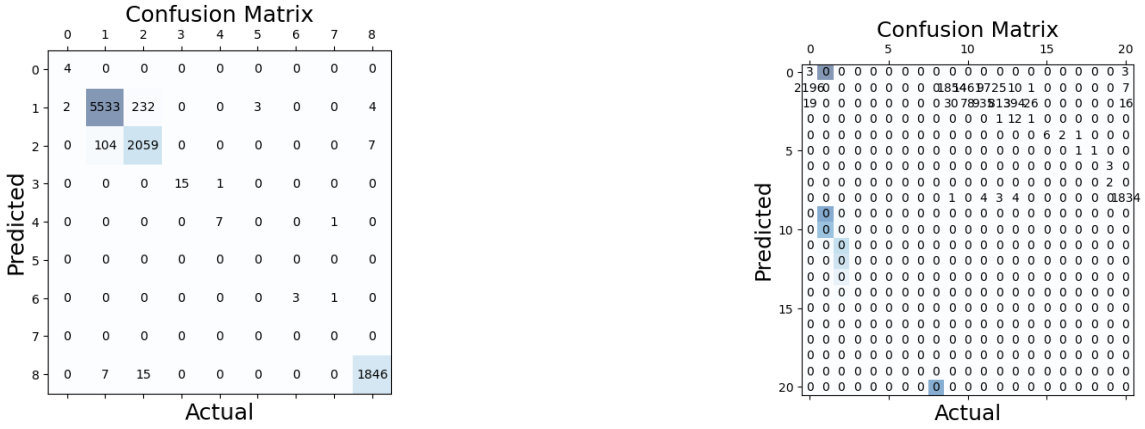
(b) Draft Stage

Figure 9: Training - Complement Naive Bayes

3.7.8 Neural Network

This is an example of an MLP classifier with a sequential architecture. The architecture has two hidden layers with (64, 32) nodes for the draft round and (128, 64) nides for the draft stage, as is the wont for small number of features. SoftMax actiivation is applied in the end for classification. Training results:

Target	Accuracy	Macro-F1
Draft Round	0.961	0.708
Draft Stage	0	0



(a) Draft Round

(b) Draft Stage

Figure 10: Training - MLP

3.7.9 Analysis

In training, most models seem to be on the verge of overfitting as the confusion matrices are almost diagonal for the best-performing classifiers.

4 Analysis: Comparison of Results, Interpretation

For each problem, I compared the performance metrics to discern the best model specific to that task and later discuss which technique performs the best across all tasks.

4.1 Regression - Demographic Attributes

Target	Trivial	1NN	Lin. Reg.	Ridge	RBF Net.	SVR	KNN	Bayes	MLP
Points	6.076	3.697	1.768e10	4.814	3.463	3.567	3.522	4.979	3.652
Rebounds	2.497	1.618	1.710e10	1.903	1.471	1.490	1.472	1.942	1.541
Assists	1.829	1.408	3.417e10	1.346	0.957	0.965	0.939	0.987	1.009

Table 1: Test RMSE

Target	Trivial	1NN	Lin. Reg.	Ridge	RBF Net.	SVR	KNN	Bayes	MLP
Points	4.827	2.718	0.103e10	3.727	2.597	2.684	2.686	3.905	2.800
Rebounds	1.890	1.126	0.099e10	1.416	1.055	1.083	1.080	1.457	1.142
Assists	1.339	0.682	0.199e10	0.955	0.655	0.674	0.668	0.987	0.708

Table 2: Test MAE

Target	Trivial	1NN	Lin. Reg.	Ridge	RBF Net.	SVR	KNN	Bayes	MLP
Points	0	0.629	-8.474e18	0.371	0.674	0.655	0.663	0.328	0.638
Rebounds	0	0.580	-4.690e19	0.419	0.652	0.643	0.652	0.395	0.619
Assists	0	0.671	-3.491e20	0.458	0.726	0.721	0.736	0.440	0.695

Table 3: Test R2-scores

The model performances on test sets are similar to the ones on validation sets, so they are not included in the analysis here. However, performances on training data are recorded in a previous section (3.5) for drawing comparisons.

As this is a regression task, all models that give a lower RMSE than the trivial system are actually learning information from the data, i.e., all models except simple linear regression without regularization. Also, as these are predictions about human data, R2-scores higher than 0.5 show capture of data variance by the models with high confidence[3].

Simple linear regression is not at all applicable to this data indicating non-linearity in the curve. Ridge Regression and Bayesian Regression are able to learn decently from the data but are not good enough for confident predictions, implying the curve is not close to a linear approximation and the features do not fit closely to Gaussian priors. RBF Networks perform the best here, closely followed by KNN, SVR (RBF Kernel), and MLP Regressors - all of which surpassed the baseline 1NN model. All these best-performing models have d.o.f >> no. of samples, and thus they train well on sparse data. I hypothesize that if the MLP was trained with more parameters, being a universal function approximator, it would eventually fit the closest to the curve and emerge as the best-performing model alongside SVR with properly tuned Support Vectors defining the curve of the target.

Best-performing regression models trained on demographic attributes, as per targets:

- Points: RBF Network - $[\gamma = 1, \lambda = 0.001]$
- Rebounds: RBF Network - $[\gamma = 1, \lambda = 0.001]$
- Assists: KNN - $[K = 9]$

4.2 Regression - Statistical Figures

Target	Trivial	1NN	Lin. Reg.	Ridge	RBF Net.	SVR	KNN	Bayes	MLP
Points	6.423	4.141	5.471	4.322	4.986	3.479	2.949	5.456	3.989
Rebounds	2.398	2.283	2.691	2.041	2.877	1.709	1.433	2.685	2.493
Assists	1.851	0.974	1.032	0.823	0.826	0.939	0.852	1.028	0.871

Table 4: Test RMSE

Target	Trivial	1NN	Lin. Reg.	Ridge	RBF Net.	SVR	KNN	Bayes	MLP
Points	4.916	3.032	3.696	3.349	3.866	2.618	2.117	3.687	2.857
Rebounds	1.821	1.545	1.887	1.525	2.210	1.383	1.038	1.883	1.634
Assists	1.333	0.655	0.682	0.578	0.567	0.576	0.548	0.679	0.541

Table 5: Test MAE

Target	Trivial	1NN	Lin. Reg.	Ridge	RBF Net.	SVR	KNN	Bayes	MLP
Points	-0.004	0.582	0.271	0.544	0.394	0.705	0.788	0.275	0.612
Rebounds	0	0.093	-0.258	0.275	-0.438	0.492	0.642	-0.252	-0.080
Assists	-0.004	0.721	0.687	0.801	0.799	0.741	0.786	0.690	0.777

Table 6: Test R2-scores

The model performances on test sets are similar to the ones on validation sets, so they are not included in the analysis here. However, performances on training data are recorded in a previous section (3.6) for drawing comparisons.

As this is a regression task, all models that give a lower RMSE than the trivial system are actually learning information from the data. Also, as these are predictions about human data, R2-scores higher than 0.5 show capture of data variance by the models with high confidence[3].

Simple linear regression and Bayesian Regression give passable performances for points and assists but not rebounds, implying the data is not close to a linear fit and that the features do not fit closely with Gaussian priors. Ridge Regression is able to learn decently from the data but is not good enough for confident predictions, except for predicting assists where it is the best-recorded model implying a polynomial curve. KNN performs the best here, followed by SVR (RBF Kernel), and MLP Regressors - all of which surpassed the baseline 1NN model. All these best-performing models have d.o.f >> no. of samples, and thus they train well on sparse data. I hypothesize that if the MLP was trained with more parameters, being a universal function approximator, it would eventually fit the closest to the curve and emerge as the best-performing model alongside SVR with properly tuned Support Vectors defining the curve of the target. Moreover, rebounds have a markedly poor correlation with the data for the least R2-scores across all models and assists can be approximated to a linear fit given its high R2-scores across all models, including the linear ones.

Best-performing regression models trained on statistical figures, as per targets:

- Points: KNN - $[K = 15]$
- Rebounds: KNN - $[K = 15]$
- Assists: Ridge - [degree= 5, $\lambda = 1$]

Observing both the regression tasks for the same target variables with different trainable features, I can make the counter-intuitive inference that the demographic attributes of players are more correlated to the players' season performance than their season stats. The model with the most consistently good performance is the KNN Regressor with one of the least computation times too.

4.3 Classification

Target	Trivial	NMC	Ridge	SVC	KNN	NB	Comp NB	MLP
Draft Round	0.419	0.548	0.626	0.825	0.780	0.466	0.658	0.672
Draft Stage	0.181	0.361	0.425	0.755	0.671	0.227	0.441	0

Table 7: Test Accuracy

Target	Trivial	NMC	Ridge	SVC	KNN	NB	Comp NB	MLP
Draft Round	0.124	0.313	0.507	0.573	0.430	0.329	0.465	0.305
Draft Stage	0.091	0.206	0.274	0.594	0.507	0.250	0.362	0

Table 8: Test Macro-F1 scores

The model performances on test sets are similar to the ones on validation sets, so they are not included in the analysis here. However, performances on training data are recorded in a previous section (3.7) for drawing comparisons.

As this is a classification task, all models that give a higher accuracy than the trivial system are actually learning information from the data - except for MLP on draft stages data.

From the confusion matrices observed during training (Figure 6), it is observable that the draft rounds are primarily distributed between rounds 1, 2, and undrafted. Consequently, so are draft numbers or draft stages are primarily distributed between 1-60 (bins 1-6) or undrafted. This heavy imbalance in data renders Macro-F1 scores as comparatively less indicative measures of correctness due to equal weightage on all classes. Hence, accuracy is the primary correctness metric for model judgment in this task.

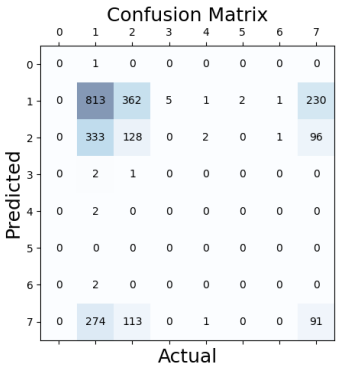
The Naive Bayes Classifier succeeds in outperforming the trivial system but not by a margin big enough to inject confidence into its predictions, and is consequently the poorest performing model - not even outperforming the baseline system. This implies the individual features do not fit closely to Gaussian priors. The Ridge Classifier performs decently implying that some decision boundaries might be closely fitting to polynomial curves, but is not the best model for classification. Even the MLP fails to learn anything from the draft stage information - indicative of wanting for better input data (different binning system) and/or better model architecture and hyperparameter tuning, although it captures some information from the most populated classes in draft rounds.

The best-performing model here is SVC (polynomial kernel), followed by KNN and Complement NB. This implies that there is useful information in assuming the complements of each feature in the featureset as fitting to multivariate Gaussian priors. SVC performing the best on a polynomial kernel substantiates the observation from the Ridge Classifier, leading to an inference that most decision boundaries can be closely fitted as polynomial functions of the features.

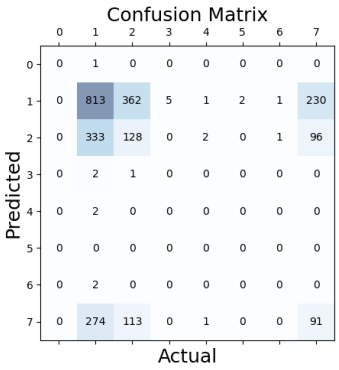
Best-performing classification models, as per targets:

- Draft Round: SVC - [kernel=polynomial, degree=4, $\gamma = 3$, $C = 0.01$]
- Draft Stage: SVC - [kernel=polynomial, degree=4, $\gamma = 3$, $C = 0.01$]

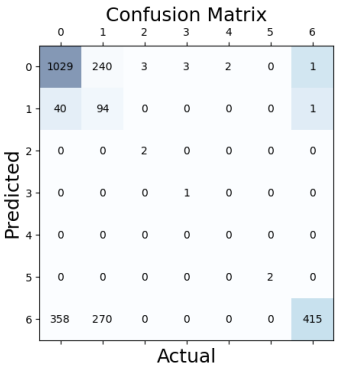
, essentially one same Support Vector Machine. The confusion matrices for test performances are on the following pages.



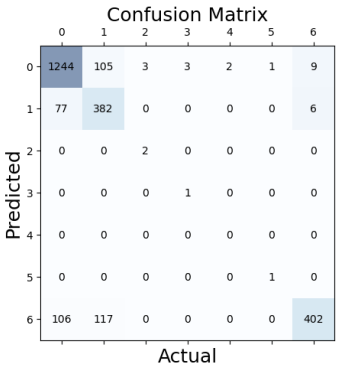
(a) Trivial



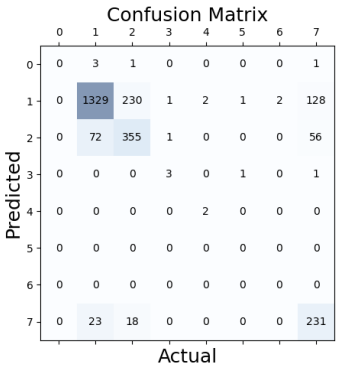
(b) NMC



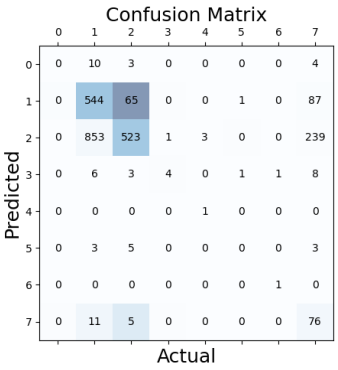
(c) Ridge



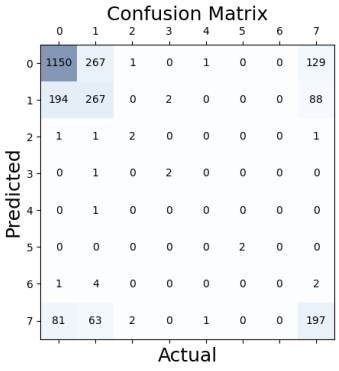
(d) SVC



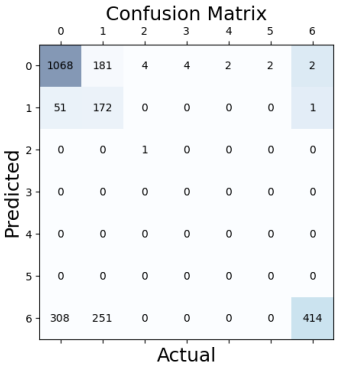
(e) KNN



(f) Naive Bayes

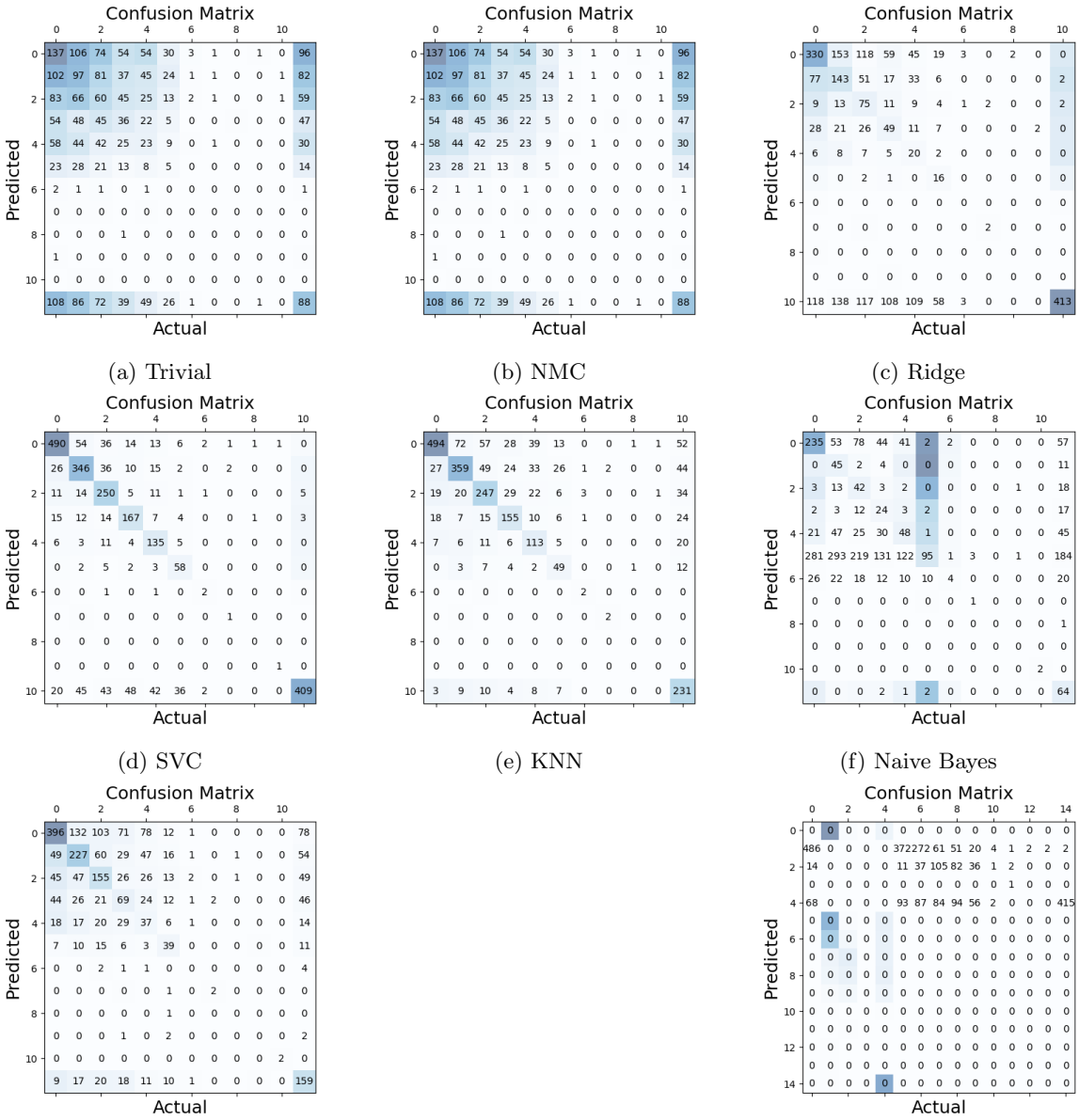


(g) Complement Naive Bayes



(h) MLP

Figure 11: Test Draft Rounds



(a) Trivial (b) NMC (c) Ridge

(d) SVC (e) KNN (f) Naive Bayes

(g) Complement Naive Bayes (h) MLP

Figure 12: Test Draft Stages

5 Libraries Used

The given list includes all automated processes that were coded:

- Numpy: For all linear algebra calculations
- Pandas: For dataset visualization in the notebook, at every stage of data cleaning and preprocessing
- Matplotlib: For plotting confusion matrices and train-val loss curves during MLP training
- Scikit-learn: For implementing most ML models and their evaluation by calling an object of these classes as models
  - Preprocessing: PolynomialFeatures - To apply polynomial kernel of the given degree to numerical feature columns

  - Model Selection: Train-test split, KFold - For creating folds for validation/cross-validation. Note that this only created folds, and validation/cross-validation is coded by myself in nested loops based on these folds.
  - Metrics: MSE, MAE, R2-score, Accuracy, Macro-F1 score, Confusion Matrix, RBF Kernel - To deliver evaluation metrics between predicted and target labels, and to apply RBF kernels
  - Linear Models: Ridge, ARD Regression, RidgeClassifier - Models for regression/classification (linear L2, Bayesian)
  - Neighbors: KNeighbors, Nearest Centroids - KNN Regressor/Classifier, and Nearest means classifier
  - SVM: SVR, SVC - Support Vector Regressor/Classifier
  - Naive Bayes: GaussianNB, ComplementNB - For Naive Bayes Classifiers
  - Neural Network: MLP Classifier
  - Multiclass: OvR Classifier - For multiclass classification by the One v. Rest method for C-classes
- Torch: For designing Sequential Neural Networks for Regression by defining Fully Connected Linear Layers and activation layers, along with many optimizer functions for Deep Learning

I coded all data cleaning and preprocessing functions by myself, Pandas was used primarily for loading and visualizing the datasets. The models I coded by myself using these libraries are - Multinomial Ridge, RBF Networks, and Neural Networks. I also coded all grid searches from scratch for hyperparameter tuning.

6 Summary and conclusions

In this project, I applied multiple ML regression and classification techniques for predicting the NBA players' average points, rebounds, and assists per game in a season given their demographic or statistical data and for predicting the draft round and draft stage of a player given all of their statistical and demographic information.

In preprocessing, I handled tasks such as data cleaning - dealing with missing and outlier values, feature engineering - obtaining useful numerical features from categorical information and applying kernels on them, and feature selection - handled during training where less important features are given much lower weights by means of regularization and weight decay.

Across all tasks, Support Vector Machines and K-Nearest Neighbors showed the most consistently good performance with sufficiently high confidence metrics. However, it has been discussed how Multi-Layer Perceptrons or Neural Networks may yet outperform these SVM and KNN models if given the computational resources to arrive at a nearly perfect set of hyperparameters given their ability as universal function approximators. In terms of speed, KNN outclassed almost all models while maintaining good correctness metrics.

For future work, I suggest that enough time is dedicated to Neural Network engineering for designing prospective models with the best correctness metrics. Given the data, several of the models listed in this report did provide good correctness metrics considering the fact that the input data is largely based on human behavior (statistical information) and includes some scientific data (absolute demographic information).

References

[1] *Complement Naive Bayes*, available at https://scikit-learn.org/stable/modules/naive_bayes.html

[2] Rennie, J. D., Shih, L., Teevan, J., & Karger, D. R. (2003). Tackling the Poor Assumptions of Naive Bayes Text Classifiers. in ICML (Vol. 3, pp. 616-623).

[3] Ozili, Peterson K (2023). The acceptable R-square in empirical modelling for social science research. in MPRA, paper no. 115769

A Appendix

- R2-score:

$$R^2 = 1 - \frac{\Sigma_i (y_i - \hat{y_i})^2}{\Sigma_i (y_i - \bar{y})^2}$$

- Macro-F1 score:

$$F_{1_c} = 2 \times \frac{precision_c \times recall_c}{precision_c + recall_c}$$

$$Macro - F_1 = \frac{\Sigma_{c=1}^{c=C} F_{1_c}}{C}$$