Draft version October 2, 2019
Typeset using LaTeX **twocolumn** style in AASTeX62

Automated MK Classification of Observed Stellar Spectra collected by the Sloan Digital Sky Survey using a Single Classifier*

Michael J. Brice[1] and Răzvan Andonie[1]

[1] *Computer Science Department*
Central Washington University
Ellensburg, WA 98926, USA

(Received June 22, 2019; Revised August 18, 2019; Accepted August 30, 2019)

Submitted to AJ

ABSTRACT

The classification of stellar spectra is a fundamental task in stellar astrophysics. Stellar spectra from the Sloan Digital Sky Survey (SDSS) are applied to standard classification methods K-Nearest Neighbors and Random Forest, to automatically classify the spectra. Stellar spectra are high dimensional data and the dimensionality is reduced using astronomical knowledge because classifiers work in low dimensional space. These methods are utilized to classify the stellar spectra into a complete MK classification (spectral and luminosity) using a single classifier. The motion of stars (radial velocity) causes machine learning complications through the feature matrix when classifying stellar spectra. Due to the nature of stellar classification and radial velocity, these complications cannot be corrected. However, classifiers utilizing a large set of observed stellar spectra that has had astronomical specific feature selection applied, performed computationally fast with extremely high accuracy.

Keywords: Stellar Classification, Morgan Keenan Classification, Radial Velocity, Sky Surveys, Computational Astronomy, Random Forest

1. INTRODUCTION

Stellar classification is a fundamental task in stellar astrophysics. Traditionally, stellar spectra are classified by determining the wavelengths of absorption lines using wavelet transformations, statistical analysis, and using references to the Morgan Keenan (MK) Classification scheme (Morgan et al. 1943) or they are classified by comparing the best fit of the spectra to that of templates using statistical tests (Duan et al. 2009). The traditional classification schemes require complex data transformations and analysis to identify the class of a star based on its spectrum.

The amount of astronomical data and dimensionality of said data is growing rapidly through more and more ambitious astronomical surveys. The Sloan Digital Sky Survey (SDSS) is an example of an ambitious astronomical survey with high quantity and dimensional data.

Presently, SDSS is creating the most detailed three-dimensional maps of the Universe ever made, with deep multi-color images of one-third of the sky, and spectra for more than three million astronomical objects[1] York et al. (2000). The SDSS provides stellar spectra with observed wavelengths. The following experiments will classify stars using SDSS data run 14 optical spectra dataset.

The SDSS and other large astronomical surveys create challenging problems for a thorough and speedy analysis. As such, automated classification methods are explored. However, some classification algorithms are limited to low dimensional data, making the use of feature selection and feature extraction essential.

Radial velocity (RV) creates complications for the automated classification of stellar spectra through the Feature Matrix. The automated process for identifying RV and stellar class that the SDSS uses is as follows (Bolton et al. (2012) and SkyServer: Redshifts, Classifications, and Velocity Dispersions[2]) :

1. Redshift and classification templates for galaxy, quasar, and CV star classes are constructed by performing a rest-frame principal-component analysis (PCA: Shlens (2014)) of training samples of known redshift.

* Released on TBD

[1] https://www.sdss.org/

[2] https://www.sdss.org/dr12/algorithms/redshifts/

2. The combination of redshift and template class that yields the overall best fit (in terms of lowest reduced chi-squared) is adopted as the pipeline measurement of the redshift and classification of the spectrum.

3. The most common warning flag is set to indicate that the change in reduced chi-squared between the best and next-best redshift/classification is either less than 0.01 in an absolute sense, or less than 1% of the best model reduced chi-squared, which indicates a poorly determined redshift.

This paper proposes a novel approach to stellar classification characterized by the following:

- Avoids complex transformation and statistical analysis of the spectra space by using machine learning.
- Use spectra without RV corrections.
- Uses astronomical knowledge to perform feature selection

Stellar spectra are classified into a complete MK Classification (spectral and luminosity) using a single classifier method. Astronomical knowledge is used to reduce the number of flux measurements. This results in key aspects of the spectra being preserved for classification which allows for a complete spectral and luminosity classification to be possible. However, the work conducted here deals with spectra with RV in the range of $\approx \pm$ 240 km/s.

The structure of this paper is as follows. Section 2 describes the approach to classification. Section 3 describes the Experimental setup and the results. Section 4 provides a discussion of the results. Finally Section 5 provides the conclusions.

2. APPROACH TO CLASSIFICATION

In this section, the data pre-processing, machine learning classifiers, and feature selection are described.

2.1. *Data Pre-Processing*

The only data pre-processing required for this approach is flux scaling using eq. (1). If a sample is known to have missing or corrupt flux measurements around the absorption lines used for Feature Selection, then an Imputer[3] method is required to fill in missing values. None of the samples in the dataset used in this analysis required an Imputer.

$$f_{i,scaled} = \frac{f_i - f_{min}}{f_{max} - f_{min}} \quad (1)$$

[3] An Imputer is used to fill in missing values in a Feature Matrix. Brice & Andonie (2019) utilize a moving average Imputer for missing values in stellar spectra.

where f_i is the ith flux measurement, f_{max} and f_{min} are the maximum and minimum flux measurements respectively, and $f_{i,scaled}$ is the resulting scaled flux.

2.2. *Machine Learning and Feature Selection*

The classifier methods of K-Nearest Neighbors (KNN) and Random Forest (RF) are used in this approach. KNN classifies using the K nearest known samples. More in depth explanations of KNN can be found in Marsland (2015); Ivezic et al. (2014); Goldberger et al. (2005). RF classifies using a forest of decision trees, where each tree votes on the classification. More in depth explanations of RF can be found in Marsland (2015); Ivezic et al. (2014); Breiman (2001). KNN and RF were chosen because they are widely used in astronomy (Ivezic et al. 2014; Bai et al. 2019; Yi & Pan 2010), and they work well in low dimensional spaces (Goldberger et al. 2005; Breiman 2001). RF was also chosen because it demonstrated good results in earlier work found in Brice & Andonie (2019).

The difference between the work presented here and the work of other authors is feature selection. Feature Selection is the act of taking a set of Attributes or Features and extracting or transforming the most relevant ones for classification and to reduce the number of dimensions for the input space for the classifier model (Bolón-Canedo et al. 2015). Authors Bazarghan & Gupta (2008), Almeida & Prieto (2013), and Yi & Pan (2010) do not use feature selection, rather they use the full range of wavelengths. This causes the input space for the classifier models to be very large in dimension, which makes the algorithms slow. Bazarghan & Gupta (2008) re-binned the SDSS spectra to have the same resolution as the Jacoby (Jacoby et al. 1984) spectra. One could argue that this is feature selection because they are converting and reducing the number of measurements, but a more accurate description would be that they are simply spectra fitting and not significantly reducing the number of dimensions. Almeida & Prieto (2013) use K Means clustering to classify SDSS stellar spectra and do not reduce the number of dimensions. Yi & Pan (2010) utilized Random Forest to classify stellar spectra. The authors also compared Random Forest to Neural Networks (Multi-layer Perceptron (MLP)).

Authors Xing & Guo (2004), Bazarghan (2008), and Bailer-Jones et al. (1998) use PCA to reduce the number of flux measurements, but they maintain the shape and structure of the overall spectrum. Xing & Guo (2004) also uses a wavelet transformation to reduce noisy flux measurements, but again they maintain the shape and structure of the spectrum. Others such as Zhang et al. (2008) normalized the continuum of the spectra. It is important to note that this does not reduce the dimensions of the spectra. Bai et al. (2019) uses a color space rather than spectra to classify stars. The authors use 9 color bands (i.e. g - r, r - i, etc.) as their features.

Elting et al. (2008) also uses photometric data instead of spectra to classify stars. Schierscher & Paunzen (2011) actually reduces the spectra using a similar approach to the work in this paper by using absorption lines, but they reduce the dimensions from 2,400 to 435, where this paper reduces down to 34, as explained in Section 3. However, Schierscher & Paunzen (2011) classifies effective temperature ranges and not direct spectral and luminosity classes.

From an astronomical point of view, spectra contain two features: Flux and Wavelength. From a machine learning point of view, spectra contain N features, where N is the number of Flux measurements. In earlier work conducted in Brice & Andonie (2019), standard machine learning feature selection methods are used which uses a statistical approach to rank correlation between Flux measurements and spectral classes. Then the K most correlated features (Flux measurements) are taken as the input space to the classifier model. This approach does not maintain the shape and structure of the spectra. The work presented in this paper does not use machine learning feature selection, rather astronomical knowledge of the spectra to reduce the number of dimensions in the input space.

The input space to classifier models is known as a feature matrix. Each column of the feature matrix is a unique feature/attribute/dimension of the object to be classified and the rows are the individual samples. For spectra, these features are the flux measurements, as seen in Table 1. It is important to notice that the wavelength values are used as the title of the unique features, not as the features themselves.

The stellar spectra found in the SDSS dataset contain on average 4,617 flux measurements or in another term the input space has 4,617 dimensions. As stated above, feature selection is used to reduce these dimensions. Standard machine learning feature selection methods used in Brice & Andonie (2019) did not work for these experiments because the spectral classes overshadow the luminosity classes. This means that statistical correlation between specific flux measurements and spectral class is stronger than the same flux measurement and luminosity class. This is apparent because the luminosity classes are based on the width of the absorption lines, which makes it difficult for individual flux measurements to be correlated to both luminosity and spectral classes. Therefore, flux measurements around an absorption line is used rather than the ones that are statistically correlated.

Since there is not one absorption line that all spectral classes share, two absorption line regions are used. The two absorption lines with rest vacuum wavelengths are H_δ (4102 $\mathring{A}$) and Ca I (4227 $\mathring{A}$). B and A stars have H_δ, K and M have Ca I, and F and G have both absorption lines. Spectral classes are separated because of the intensity of the flux in the regions and Luminosity classes are separated because of the widths

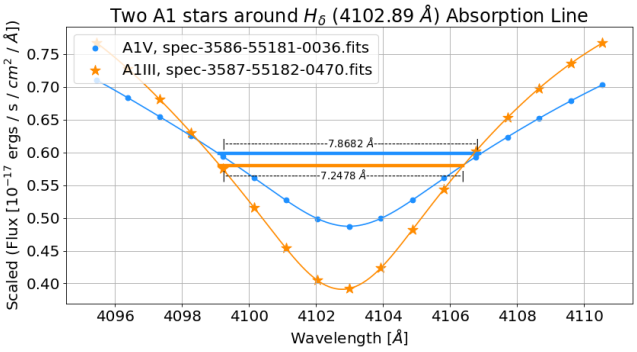

Figure 1. Example of the same spectral class with different wavelength width (Full Width Half Max) for the same absorption line for different MK classes.

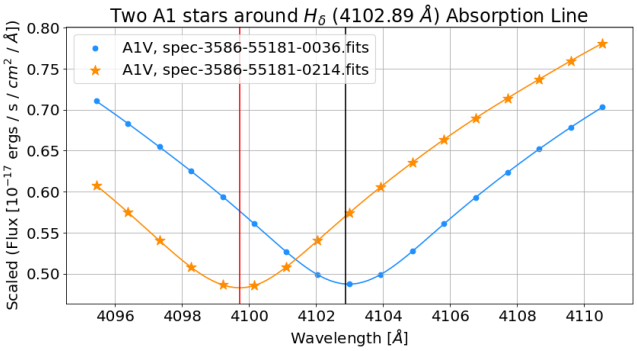

Figure 2. Example of how RV is accounted for in the feature selection window

.

of the absorption line. Figure 1 shows how the width of the absorption line changes the flux measurements for two A type stars. Using these two regions the feature matrix can be built. The flux measurements from the H_δ region and the flux measurements from the Ca I region are combined to create a single flux array per sample, as seen in Table 1. This feature selection must also be able to incorporate the shifting of the spectrum due to RV. Figure 2 shows that with a sufficiently sized region, RV can be incorporated.

The feature matrix can be represented as an N dimensional hyper-cube. Where each feature is a dimension in this hyper-cube. As mentioned before, the wavelengths are used as dimension labels, where the flux measurement at that wavelength is the magnitude of the vector in that dimension. This results in each reduced "spectrum" being represented as a N dimensional point. As the shape and intensity of the flux in these regions change due to spectral class, the positions of these reduced spectra change in this N dimensional hyper-cube. The same happens when the width of the absorption line changes with luminosity class.

A problem arises with the feature matrix because RV causes the flux measurements to be shifted in

Table 1. Example of the feature matrix using two sets of wavelengths around two absorption lines. The 34 features result in a 34 column feature matrix.

	Wavelength: 4,095.43 $\AA$	...	Wavelength: 4,110.55 $\AA$	Wavelength: 4,219.88 $\AA$	...	Wavelength: 4,235.45 $\AA$
Spectrum 1	Flux	...	Flux	Flux	...	Flux
Spectrum 2	Flux	...	Flux	Flux	...	Flux

Table 2. Example of the feature matrix with observed A0 spectra. Note for the A0 stars, RV causes the H_δ absorption line to be modeled with different wavelength features.

Star Class	...	Wavelength: 4,101.00 $\AA$	Wavelength: 4,101.94 $\AA$	Wavelength: 4,102.89 $\AA$	Wavelength: 4,103.83 $\AA$	Wavelength: 4,104.77 $\AA$	...
A0	...		H_δ				...
A0	...	H_δ					...
A0	...			H_δ			...
A0	...				H_δ		...
A0	...					H_δ	...

wavelength. This is illustrated in Table 2. As mentioned earlier, each feature is unique. The problem that RV causes is that it breaks this feature uniqueness. To overcome this problem, a sufficient number of samples of the exact same class with different RV that span the range of realistic RV are required. This gives the training dataset for the classifier model sufficient samples of spectral and luminosity classes at different RV. In terms of KNN, this allows for sufficient neighbors to be nearby when classifying. In terms of RF, this forces the splitting threshold of each decision tree's node to include spectra with RV.

3. EXPERIMENTS

In this section, the experimental setup is described and the results are presented.

3.1. *Experimental Setup*

The dataset used in these experiments comes from SDSS data run 14, which was collected using the BOSS spectrograph[4] (Smee et al. 2013). Data run 14 contains a total of 335,844 spectra. Some of the data was rejected because it was not a spectral class of O, B, A, F, G, K, and M with subclass of 0 - 9 combined with luminosity class of I, II, III, IV, V, VII or that the data was missing a large portion of its spectrum, similarly to the work done in Brice & Andonie (2019). The data was also pre-processed by SDSS scientists through the methods presented by Dawson et al. (2013) and Stoughton et al. (2002).

The usable dataset contains 168,982 stellar spectra and 46 of the 420 class combinations. It is important to note that this is real collected data and not simulated data. The spectra are first pre-processed by scaling the flux to ensure that similar classes have similar flux measurements using eq. (1).

Then feature selection is performed using Algorithm 1, where bounds is the number of flux measurements before and after the absorption line. Bounds is set to 8 to cover the RV range of -552 km/s to 552 km/s and ensure that sufficient flux measurements are recorded. This results in a region of 17 flux measurements around the H_δ (4102 $\AA$) and a region of 17 flux measurements around the Ca I (4227 $\AA$) absorption line, which is combined to form a 34 dimension feature matrix.

After the feature selection phase, the dataset is split into 2 subsets for RV. One for RV less than 200 km/s and one for RV greater than 200 km/s. Each subset is again divided into 10 subsets for 10 fold cross validation (Kohavi 1995). One subset from each RV set is taken as testing sets respectively. The remaining subsets are combined into one training set. Due to the dataset being imbalanced (Figure 3), the training set is balanced using a *Undersampling* method (Japkowicz 2000) (randomly removing samples), an *Oversampling* method (randomly duplicating samples using SMOTE (Chawla et al. 2002)), and a *Hybrid* method (Undersampling + Oversampling). Then, KNN and RF classifiers are applied to the training set and tested using the RV less than 200 km/s test set and RV greater than 200 km/s test set as well as a combined test set. These steps are repeated 10 times with different subsets used for testing for 10 fold cross validation. The accuracy, precision, recall, and F1 score (defined below) are averaged over each cross validation. Misclassification costs for incorrectly classifying a spectrum is not explored.

[4] Simplified Boss Spectrograph explanation: https://www.sdss.org/instruments/boss_spectrograph/

Algorithm 1 Flux Feature Selection

```
function FEATURE_SELECTION(flux_Arr, wavelength_Arr, absorption_Line)
    bounds = 8 // range of flux measurements before and after the absorption line
    for 0 < i < flux_Arr_Length do
        index = Find_Nearest(wavelength_Arr[i], absorption_Line)
        for index − bounds < j < index + bounds do
            new_flux_Arr[i][j] = flux_Arr[i][j]
            new_wavelength_Arr[i][j] = wavelength_Arr[i][j]
        end for
    end for
    return new_flux_Arr, new_wavelength_Arr
end function
function FIND_NEAREST(array, value)
    min = ∞
    index = −1
    for 0 < i < array_Length do
        if array[i] − value < min then
            index = i
        end if
    end for
    return index
end function
procedure MAIN
    flux_Arr_1, wavelength_Arr_1 = Feature_Selection(flux_Arr,
                                                     wavelength_Arr, 4102.89)
    flux_Arr_2, wavelength_Arr_2 = Feature_Selection(flux_Arr,
                                                     wavelength_Arr, 4227.79)
    flux_Arr = Append(flux_Arr_1, flux_Arr_2)
    wavelength_Arr = Append(wavelength_Arr_1, wavelength_Arr_2)
end procedure
```

The experiments are implemented in Python, using scikit-learn (Pedregosa et al. 2011). Due to the size of the dataset (16.7 GB, which is larger than the RAM used in these experiments), the Python NumPy memmap[5] (Oliphant 2015) module was used to read very large arrays from storage rather than RAM. The experiments are performed on a personal computer with the following relevant specifications: AMD Ryzen 7 1800x 16 logical core CPU, 16 GB RAM, and 1 TB Samsung 860 EVO Solid State Drive.

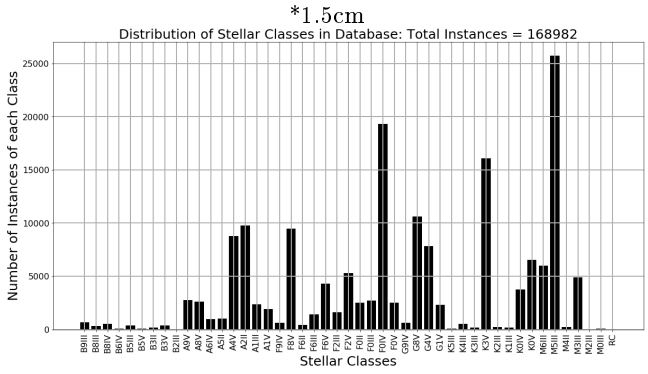

Figure 3. Distribution of classes in the dataset. RC = Remaining Classes of 0 instances

[5] Memmap: https://docs.scipy.org/doc/numpy-1.14.0/reference/generated/numpy.memmap.html

KNN and RF use the scikit-learn default parameters (Pedregosa et al. 2011). Precision[6], recall[7], and F1 score[8] are computed using functions implemented by the scikit-learn sklearn.metrics package (Pedregosa et al. 2011). Feature selection is implemented using Algorithm 1 in Python and uses the Python multiprocessing package[9] for parallelization.

Precision is the measure of how well a predicted class compares to actual classes. (Marsland 2015). For example, in the binary class case, precision is the number of samples correctly predicted as class 1 divided by the total number of samples predicted as class 1 (eq. 2). Recall is the measure of how well an actual class compares to predicted classes. (Marsland 2015). For example, in the binary class case, recall is the number of samples correctly predicted as class 1 divided by the total number of samples that are actually class 1 (eq. 3). F1 score is a type of harmonic mean that combines precision and recall into a single metric (eq. 4) (Marsland 2015). The closer precision, recall, and F1 score is to 1, the more accurate the model. In terms of this work, precision, recall, and F1 score are reported as an average of all the individual precision, recall, and F1 score measurements associated with each of the 46 classes respectively. Precision, Recall, and F1 score are defined in the binary class case as follows:

$$Precision = \frac{Total\_TP}{Total\_TP + Total\_FP} \tag{2}$$

$$Recall = \frac{Total\_TP}{Total\_TP + Total\_FN} \tag{3}$$

$$F_1 = 2\frac{Precision * Recall}{Precision + Recall} \tag{4}$$

where TP (True Positive) is defined as a prediction of class 1 being correctly classified as class 1, FP (False Positive) is defined as a prediction of class 1 being incorrectly classified as class 2, and FN (False Negative) is defined as a sample of class 1 being incorrectly predicted as class 2. In the multi-class problem, False Positive is defined as a prediction of class i being incorrectly classified as class $\neq$ i and False Negative is defined as a sample of class i being incorrectly predicted as class $\neq$ i. More information regarding precision, recall, and F1 score can be found here: Powers & Ailab (2011).

[6] Precision SciKit-Learn: https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html

[7] Recall SciKit-Learn: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html#sklearn.metrics.recall_score

[8] F1 Score SciKit-Learn: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html#sklearn.metrics.f1_score

[9] Multiprocessing: https://docs.python.org/3.7/library/multiprocessing.html#module-multiprocessing

4. DISCUSSION

Tables 3 and 6 show that classification using KNN has essentially the same accuracy as RF when using hybrid and oversampling balancing. These tables demonstrate that using KNN and RF along side Algorithm 1 for feature selection are viable options for the automated classification of stellar spectra because of the high accuracy achieved. Table 3 demonstrate that using three neighbors for KNN classification performs the best for KNN. Table 6 shows that changing the number of trees used in RF does not significantly change the classification accuracy. Tables 3 and 6 shows that Oversampling balancing outperforms Hybrid balancing.

Tables 4 and 7 show that the precision and recall is less than accuracy (Tables 3 and 6). The precision implies on average that the RF model's prediction of class i is 84.91% (Table 7) actually class i. Where recall implies on average, the RF model predicts 86.63% (Table 7) of the samples that is class i as class i. It is important to note that the test sets for all RV, RV $<$ 200 km/s, and RV $\geq$ 200 km/s metrics are tested using the same trained model. As seen in these tables, this approach accurately classifies both RV $<$ 200 km/s and RV $\geq$ 200 km/s. However, RV $\geq$ 200 km/s has a higher accuracy, precision, and recall (Table 7) than RV $<$ 200 km/s. This could be the result of an imbalance between low RV and high RV samples and overfitting the model in the low RV range. Future work will attempt to increase precision and recall for low RV, which includes balancing the RV range.

Tables 5 and 8 show the execution times for each experiment. These tables show that KNN performs much faster than RF. KNN has a faster train time than RF, but RF has a faster test time than KNN. For both KNN and RF, feature selection takes approximately the same amount of time, which is expected since they both use the same feature selection.

This approach takes considerably fewer steps than the one in Bolton et al. (2012) and produces excellent results. The execution times and the obtained accuracy demonstrate that, for a real application of this work, the automated classification of observed stellar spectra into a complete MK classification using a single classifier not only achieves a high accuracy but is also fast.

These experiments yielded an accuracy of 97.16% for RF (Table 6) using Oversampling Balancing and 150 decision trees (Table 6), which is significantly better than Xing & Guo (2004), which reports 81.66% accuracy for just Support Vector Machine (SVM) with no data reduction, 93.26% for wavelet + SVM and 81.30% for PCA + SVM. Schierscher & Paunzen (2011) are able to achieve an 85% match in comparison with SEGUE Stellar Parameter Pipeline (SSPP) using an Artificial Neural Network (ANN), but this is not comparable since they did not classify into direct spectral and

Table 3. 10-Fold cross validation Accuracy for KNN.

Balance Method	RV	Accuracy (%) for K Neighbors					
		3.0	5.0	7.0	10.0	15.0	20.0
Undersampled	All	52.66	52.31	51.47	50.73	49.63	48.49
	< 200 km/s	53.32	52.99	52.17	51.45	50.35	49.23
	≥ 200 km/s	23.97	22.94	21.18	19.38	17.98	16.67
Hybrid	All	93.02	92.63	92.31	91.85	91.26	90.73
	< 200 km/s	92.92	92.56	92.26	91.82	91.28	90.77
	≥ 200 km/s	97.15	95.62	94.44	92.94	90.54	88.96
Oversampled	All	95.48	95.00	94.54	93.98	93.24	92.58
	< 200 km/s	95.46	94.99	94.56	94.02	93.32	92.69
	≥ 200 km/s	96.41	95.28	93.91	92.18	89.54	87.78

Table 4. 10-Fold cross validation Precision, Recall, and F1 Score for KNN using Oversampling.

	RV	K Neighbors					
		3.0	5.0	7.0	10.0	15.0	20.0
Precision	All	0.805493	0.795833	0.789050	0.781626	0.767124	0.756264
	< 200 km/s	0.803331	0.793968	0.787575	0.780606	0.766780	0.756458
	≥ 200 km/s	0.926071	0.897225	0.872481	0.838610	0.793230	0.756975
Recall	All	0.835391	0.834499	0.834981	0.835475	0.833052	0.829150
	< 200 km/s	0.832799	0.831953	0.832601	0.833312	0.831288	0.829150
	≥ 200 km/s	0.941410	0.921932	0.905449	0.879239	0.853662	0.959524
F1 Score	All	0.811617	0.805477	0.801665	0.796227	0.786297	0.776893
	< 200 km/s	0.809250	0.803309	0.799771	0.794651	0.785290	0.776297
	≥ 200 km/s	0.989692	0.985351	0.981322	0.975739	0.967290	0.827752

Table 5. 10-Fold cross validation Execution Times for KNN using Oversampling for all RV.

	Time in seconds for K Neighbors					
	3.0	5.0	7.0	10.0	15.0	20.0
Feature Selection	90.89	90.89	90.89	90.89	90.89	90.89
Train	6.91	6.91	6.95	6.94	6.88	6.82
Test	3.03	3.22	3.41	3.62	3.92	4.19

luminosity classes. Schierscher & Paunzen (2011) also never addresses the fact that their data is imbalanced.

Other authors such as Bazarghan & Gupta (2008); Yi & Pan (2010); Zhang et al. (2008) do not report metrics that are comparable to the work conducted in this paper. Bazarghan & Gupta (2008) used a Probabilistic Neural Network implemented in MATLAB and they used a χ^2 value to determine classification accuracy. They make the assumption that a χ^2 value of 0.002 or lower is considered classified correctly, then they achieved a success rate of about 88% in only a few seconds. Yi & Pan (2010) find that RF performed better than the MLP with Root Mean Square Error (RMSE) of 1.04 and 1.36 respectively. Zhang et al. (2008) separated the classification into two classifiers. For the spectral classes, the authors used a non-parameter regression method. For the luminosity classes, the authors removed or normalized the continuum of the spectra and used a partial least-squared regression method. Zhang et al. (2008) used three spectra data sources, Silva & Cornell (1992), Pickles (1998), and Jacoby et al. (1984). They achieved a standard deviation for the spectral classes of $\sigma = 0.7994$ and for the luminosity classes of $\sigma = 0.58159$.

As described above in Section 3, these experiments deal with data collected by a real astronomical survey. As such, when an astronomical survey points their telescopes into the sky, they get the samples (classes) that they get. The experiments presented here deal

Table 6. 10-Fold cross validation Accuracy for RF.

Balance Method	RV	Accuracy (%) for N Trees					
		10.0	50.0	100.0	150.0	200.0	250.0
Undersampled	All	52.45	58.13	58.75	59.01	59.09	59.21
	< 200 km/s	53.09	58.78	59.39	59.65	59.72	59.83
	≥ 200 km/s	24.62	29.79	30.85	31.48	31.77	31.91
Hybrid	All	93.83	94.44	94.48	94.48	94.50	94.49
	< 200 km/s	93.73	94.35	94.38	94.39	94.41	94.41
	≥ 200 km/s	98.39	98.36	98.36	98.44	98.61	98.41
Oversampled	All	96.39	97.05	97.12	97.16	97.15	97.16
	< 200 km/s	96.34	97.01	97.09	97.13	97.12	97.12
	≥ 200 km/s	98.49	98.65	98.62	98.62	98.65	98.65

Table 7. 10-Fold cross validation Precision, Recall, and F1 Score for for RF using Oversampling.

	RV	N Trees					
		10.0	50.0	100.0	150.0	200.0	250.0
Precision	All	0.827606	0.845460	0.849466	0.849149	0.850188	0.850051
	< 200 km/s	0.825895	0.844077	0.848296	0.847941	0.848908	0.848765
	≥ 200 km/s	0.968922	0.971134	0.967166	0.968655	0.971852	0.971852
Recall	All	0.849646	0.862420	0.866250	0.866379	0.868213	0.868580
	< 200 km/s	0.847837	0.861001	0.865014	0.865182	0.866917	0.867305
	≥ 200 km/s	0.973081	0.973082	0.970220	0.972075	0.972433	0.972433
F1 Score	All	0.832101	0.847679	0.851134	0.851346	0.852966	0.853006
	< 200 km/s	0.830256	0.846204	0.849857	0.850088	0.851616	0.851663
	≥ 200 km/s	0.969936	0.970727	0.967720	0.969363	0.970591	0.970591

Table 8. 10-Fold cross validation Execution Times for RF using Oversampling for all RV.

	Time in seconds for N Trees					
	10.0	50.0	100.0	150.0	200.0	250.0
Feature Selection	90.89	90.89	90.89	90.89	90.89	90.89
Train	129.23	605.32	1226.75	1857.21	2448.32	2960.94
Test	0.65	2.91	6.03	9.16	11.99	14.46

with a subset of all possible class combinations. It is important to note that not all possible class combinations (O, B, A, F, G, K, and M with sub - classes of 0 - 9 combined with I, II, III, IV, V, VII) are common or even found in nature. Therefore, even though this approach yielded great results, there cannot be a claim that this approach will guarantee work for all stellar classes. There is, however, some theoretical validity to this approach.

Referencing Figure 3, O type stars are the only spectral class not found in the dataset. Figure 4 shows that O type stars also contain the H_δ absorption line. Therefore, the missing spectral and luminosity classes are compatible with this approach because every spectral major class has at least one absorption line in the feature selection regions, which allows for the variation in width for luminosity classes.

5. CONCLUSION

The results shown in this paper support that accurate automatic stellar classification can be obtained using astronomical specific feature selection. Compared to previous work of other authors, there are four interesting conclusions:

1. A high level of accuracy can be obtained by considering only flux measurements at wavelengths near the H_δ and Ca I absorption lines.

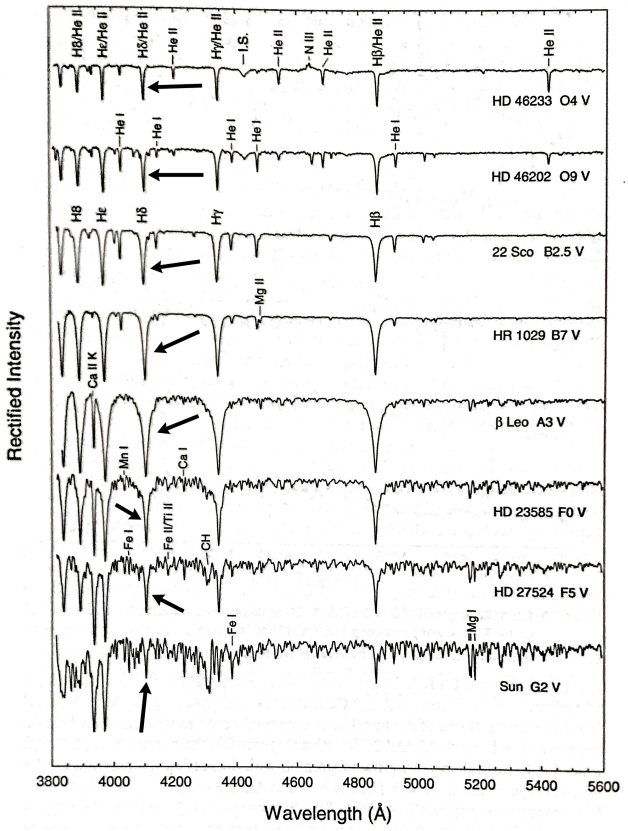

Figure 4. Sample of continuum normalized spectra from O - G type stars, Gray & Corbally (2009). The arrows point to the H_δ absorption line

2. A complete MK classification can be identified using a single classifier with a high level of accuracy for stars with small RV (the range of $\approx \pm$ 240 km/s). However, for stars with large RV (outside the range of $\approx \pm$ 240 km/s), increasing the value of bounds in Algorithm 1 should compensate for large RV, but at the cost of an increase to the computational time because of the increase in dimensions.

3. Correcting for RV is not necessary because of a sufficient distribution of samples with different RV.

4. Aside from flux scaling, any additional spectrum pre-processing after recombining and re-binning the spectra as presented by Dawson et al. (2013) and Stoughton et al. (2002) is unnecessary for SDSS stellar spectra.

Therefore, this new approach for the automatic classification of stellar spectra is feasible, useful, and accurate. Future work will be conducted with the concepts of this approach to automatically classify stars and large redshift objects such as galaxies and quasars. Future work will also include building the training set with an equal distribution of RV and experiment using SMOTE or similar algorithms to build a training set using a small number of real samples to simulate how this approach could work for a new spectroscopic survey.

REFERENCES

Almeida, J. S., & Prieto, C. A. 2013, ApJ, 763, 50. http://stacks.iop.org/0004-637X/763/i=1/a=50

Bai, Y., Liu, J., Wang, S., & Yang, F. 2019, AJ, 157, 9, doi: 10.3847/1538-3881/aaf009

Bailer-Jones, C. A. L., Irwin, M., & von Hippel, T. 1998, Monthly Notices of the Royal Astronomical Society, 298, 361, doi: 10.1046/j.1365-8711.1998.01596.x

Bazarghan, M. 2008, arXiv e-prints, arXiv:0804.2742. https://arxiv.org/abs/0804.2742

Bazarghan, M., & Gupta, R. 2008, Astrophysics and Space Science, 315, 201, doi: 10.1007/s10509-008-9816-5

Bolón-Canedo, V., Sánchez-Maroño, N., & Alonso-Betanzos, A. 2015, Feature Selection for High-Dimensional Data (Springer International Publishing), doi: 10.1007/978-3-319-21858-8

Bolton, A. S., Schlegel, D. J., Aubourg, É., et al. 2012, AJ, 144, 144, doi: 10.1088/0004-6256/144/5/144

Breiman, L. 2001, Machine Learning, 45, 5, doi: 10.1023/A:1010933404324

Brice, M., & Andonie, R. 2019, in 2019 International Joint Conference on Neural Networks (IJCNN) (Budapest, Hungary: IEEE), 1–8, doi: 10.1109/IJCNN.2019.8852407

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. 2002, Journal of Artificial Intelligence Research, 16, 321–357, doi: 10.1613/jair.953

Dawson, K. S., Schlegel, D. J., Ahn, C. P., et al. 2013, AJ, 145, 10, doi: 10.1088/0004-6256/145/1/10

Duan, F.-Q., Liu, R., Guo, P., Zhou, M.-Q., & Wu, F.-C. 2009, Research in Astronomy and Astrophysics, 9, 341, doi: 10.1088/1674-4527/9/3/009

Elting, C., Bailer-Jones, C. A. L., & Smith, K. W. 2008, AIP Conference Proceedings, 1082, 9, doi: 10.1063/1.3059095

Goldberger, J., Hinton, G. E., Roweis, S. T., & Salakhutdinov, R. R. 2005, in Advances in Neural Information Processing Systems 17, ed. L. K. Saul, Y. Weiss, & L. Bottou (MIT Press), 513–520. http://papers.nips.cc/paper/2566-neighbourhood-components-analysis.pdf

Gray, R. O., & Corbally, J., C. 2009, Stellar Spectral Classification (Princeton University Press)

Ivezic, Z., Connolly, A., VanderPlas, J., & Gray, A. 2014, Statistics, Data Mining, and Machine Learning in Astronomy: A Practice Python Guide for the Analysis of Survey Data (Princeton University Press), doi: 10.23943/princeton/9780691151687.001.0001

Jacoby, G. H., Hunter, D. A., & Christian, C. A. 1984, apjs, 56, 257, doi: 10.1086/190983

Japkowicz, N. 2000, in Papers from the AAAI Workshop Technical Report WS-00-05 (AAAI Press), 10–15

Kohavi, R. 1995, in Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2, IJCAI'95 (San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.), 1137–1143. http://dl.acm.org/citation.cfm?id=1643031.1643047

Marsland, S. 2015, Machine Learning: An Algorithmic Perspective, 2nd edn. (CRC Press)

Morgan, W. W., Keenan, P. C., & Kellman, E. 1943, An atlas of stellar spectra, with an outline of spectral classification (Chicago, III., The University of Chicago Press)

Oliphant, T. E. 2015, A guide to NumPy (Published by Continuum Press, a division of Continuum Analytics, Inc.)

Pedregosa, F., Varoquaux, G., Gramfort, A., et al. 2011, J. Mach. Learn. Res., 12, 2825. http://dl.acm.org/citation.cfm?id=1953048.2078195

Pickles, A. J. 1998, pasp, 110, 863, doi: 10.1086/316197

Powers, D., & Ailab. 2011, J. Mach. Learn. Technol, 2, 2229, doi: 10.9735/2229-3981

Schierscher, F., & Paunzen, E. 2011, Astronomische Nachrichten, 332, 597, doi: 10.1002/asna.201011556

Shlens, J. 2014, arXiv e-prints, arXiv:1404.1100. https://arxiv.org/abs/1404.1100

Silva, D. R., & Cornell, M. E. 1992, Astrophysical Journal Supplement Series, 81, 865, doi: 10.1086/191706

Smee, S. A., et al. 2013, AJ, 146, 1, doi: 10.1088/0004-6256/146/2/32

Stoughton, C., Lupton, R. H., Bernardi, M., et al. 2002, AJ, 123, 485, doi: 10.1086/324741

Xing, F., & Guo, P. 2004, in Advances in Neural Networks – ISNN 2004 (Berlin, Heidelberg: Springer Berlin Heidelberg), 616–621, doi: 10.1007/978-3-540-28647-9_101

Yi, Z., & Pan, J. 2010, 978-1-4244-6516-3/10/26.00 2010 IEEE, 3129

York, D. G., et al. 2000, AJ, 120, 1579, doi: 10.1086/301513

Zhang, J. N., Luo, A. L., & Tu, L. P. 2008, in 2008 International Congress on Image and Signal Processing (CISP 2008), 249–252