

**WEATHER PREDICTION**

Team name: Black Sharks

Team members:

1. Sudheer Kumar Reddy Reddymalla  
[sredd1@unh.newhaven.edu](mailto:sredd1@unh.newhaven.edu)  
00719922
2. Durga Guna Sekhar Reddy Savanam  
[dsava2@unh.newhaven.edu](mailto:dsava2@unh.newhaven.edu)  
00755099
3. Vamsi Krishna Regalla  
[vregal@unh.newhaven.edu](mailto:vregal@unh.newhaven.edu)  
00730353

**Research Question:**

To predict the weather based on the patterns previously collected using data mining techniques. Here, in this case we use Seattle's weather data.

**Data Set:**

The data set we taken is from Kaggle website. Here is the link for our data set: <https://www.kaggle.com/datasets/ananthr1/weather-prediction> It has 6 column attributes.

**Data Mining Techniques:**

- Naïve Bayes Classification
- K-Star algorithm
- SMO model
- Decision Tree and
- Random forest

**Parameters and hyperparameters:**

Precipitation: All forms in which water falls on the land surface and open water bodies as rain, sleet, snow, hail, or drizzle.

temp\_max : shows the results which has maximum temperature it may range from -1.6 to 35.6 .

temp\_min : shows the results which has minimum temperature it may range from -7 to 18.3 .

wind: shows the wind speed on those specific dates it may range from 0.4 to 9.5.

other parameters such as drizzle, rain, sun, snow, fog.

## Hardware:

Processor: M1 chip

Ram: 8GB

## Outcomes and Visualization Techniques:

**Naïve Bayes Classification:** A probabilistic classifier is the Naive Bayes algorithm for classification. It is based on probability models that make substantial assumptions about independence. The independence presumptions frequently do not affect reality. They are therefore viewed as being naive.

By using Naive bayes model, correctly classified instances are 1269 which is 86.85%.

The screenshot displays the Weka Explorer interface with the NaiveBayes classifier selected. The 'Classify' tab is active, and the 'Test options' section shows 'Use training set' selected. The 'Classifier output' section provides a summary of the model's performance on the training set.

**Classifier output**

Time taken to build model: 0.01 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.05 seconds

=== Summary ===

Correctly Classified Instances	1269	86.8583 %
Incorrectly Classified Instances	192	13.1417 %
Kappa statistic	0.7739	
Mean absolute error	0.0784	
Root mean squared error	0.1912	
Relative absolute error	32.1333 %	
Root relative squared error	54.7807 %	
Total Number of Instances	1461	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.264	0.005	0.667	0.264	0.378	0.407	0.934	0.453	drizzle
	0.914	0.007	0.990	0.914	0.951	0.917	0.985	0.987	rain
	0.984	0.205	0.789	0.984	0.876	0.777	0.969	0.955	sun
	0.846	0.008	0.667	0.846	0.746	0.746	0.991	0.882	snow
	0.168	0.000	1.000	0.168	0.288	0.398	0.968	0.842	fog
Weighted Avg.	0.869	0.093	0.885	0.869	0.848	0.798	0.975	0.942	

=== Confusion Matrix ===

	a	b	c	d	e	<-- classified as
14	0	39	0	0	0	a = drizzle
0	586	47	8	0	0	b = rain
6	2	630	2	0	0	c = sun
0	4	0	22	0	0	d = snow
1	0	82	1	17	0	e = fog

Status: OK

Log x 0

**K-Star algorithm:** K-star is an instance-based classifier, meaning that it bases the classification of a test instance on the classification of training instances that are like it as specified by some similarity function. Use of a modelling and simulation distance function sets it apart from other instance-based learners. By using K-star algorithm, correctly classified instances are 100%.

**Weka Explorer**

Preprocess **Classify** Cluster Associate Select attributes Visualize

Classifier: Choose **KStar -B 20 -M a**

Test options:  
☒ Use training set  
☐ Supplied test set (Set...)  
☐ Cross-validation (Folds: 10)  
☐ Percentage split (%: 66)  
 More options...

(Nom) weather  
 Start Stop

Result list (right-click for options):  
 15:13:33 - trees.J48  
 14:57:14 - misc.InputMappedClassifier  
 14:57:35 - misc.InputMappedClassifier  
 15:00:18 - misc.InputMappedClassifier  
 15:01:52 - bayes.NaiveBayes  
**15:45:48 - lazy.KStar**

**Classifier output**  
 KStar options: -B 20 -M a  
 Time taken to build model: 0 seconds  
 === Evaluation on training set ===  
 Time taken to test model on training data: 3.57 seconds  
 === Summary ===  
 Correctly Classified Instances 1461 100 %  
 Incorrectly Classified Instances 0 0 %  
 Kappa statistic 1  
 Mean absolute error 0.0041  
 Root mean squared error 0.0105  
 Relative absolute error 1.6945 %  
 Root relative squared error 3.0191 %  
 Total Number of Instances 1461  
 === Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	drizzle
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	rain
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	sun
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	snow
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	fog
Weighted Avg.	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	

=== Confusion Matrix ===  
 a b c d e <-- classified as  
 53 0 0 0 0 | a = drizzle  
 0 641 0 0 0 | b = rain  
 0 0 640 0 0 | c = sun  
 0 0 0 26 0 | d = snow  
 0 0 0 0 101 | e = fog

**SMO model:** is a method for resolving the quadratic programming (QP) problem that appears during training support vector machines (SVM). Support vector machines are frequently trained using SMO, which is implemented by the well-liked LIBSVM tool. As a result, that earlier SVM training techniques were far more complicated and needed pricey third-party QP solvers. In this model, correctly classified instances are 99%.

The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. The classifier chosen is 'SMO -C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V -1 -W 1 -K "weka.classifiers.functions.supportVector.PolyKernel -E 1.0 -C 250007" -calibrator "weka.classifiers.functions.SMO"'. The test options are set to 'Use training set' with 'Folds' set to 10 and 'Percentage split' set to 66. The result list on the left shows several classifiers, with '17:08:47 - functions.SMO' selected. The classifier output on the right displays the following information:

Time taken to build model: 4.25 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.18 seconds

=== Summary ===

Correctly Classified Instances	1454	99.5209 %
Incorrectly Classified Instances	7	0.4791 %
Kappa statistic	0.9921	
Mean absolute error	0.2402	
Root mean squared error	0.3165	
Relative absolute error	98.463 %	
Root relative squared error	90.6705 %	
Total Number of Instances	1461	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.981	0.000	1.000	0.981	0.990	0.990	1.000	1.000	drizzle
	0.995	0.000	1.000	0.995	0.998	0.996	0.998	0.998	rain
	1.000	0.009	0.989	1.000	0.995	0.990	0.996	0.989	sun
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	snow
	0.970	0.000	1.000	0.970	0.985	0.984	1.000	0.999	fog
Weighted Avg.	0.995	0.004	0.995	0.995	0.995	0.992	0.997	0.994	

=== Confusion Matrix ===

	a	b	c	d	e	← classified as
52	0	1	0	0	0	a = drizzle
0	638	3	0	0	0	b = rain
0	0	640	0	0	0	c = sun
0	0	0	26	0	0	d = snow
0	0	3	0	98	0	e = fog

**Random Forest:** Supervised machine learning algorithms like random forest are frequently employed in classification and regression issues. It creates decision trees from several samples, using the majority vote for classification and the average in the case of regression.

In this random forest model, correctly classified instances are 100%.

The screenshot shows the WEKA software interface with the 'Classify' tab selected. The 'Classifier' dropdown is set to 'RandomForest'. The 'Test options' section shows 'Use training set' selected. The 'Result list' on the left shows the 'trees.RandomForest' model selected. The 'Classifier output' pane displays the following results:

weka.classifiers.trees.RandomForest -K 0 -M 1.0 -V 0.001 -S 1 -no-check-capabilities

Time taken to build model: 2.08 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.05 seconds

=== Summary ===

Correctly Classified Instances	1461	100	%
Incorrectly Classified Instances	0	0	%
Kappa statistic	1		
Mean absolute error	0.0729		
Root mean squared error	0.1077		
Relative absolute error	29.8909 %		
Root relative squared error	30.8456 %		
Total Number of Instances	1461		

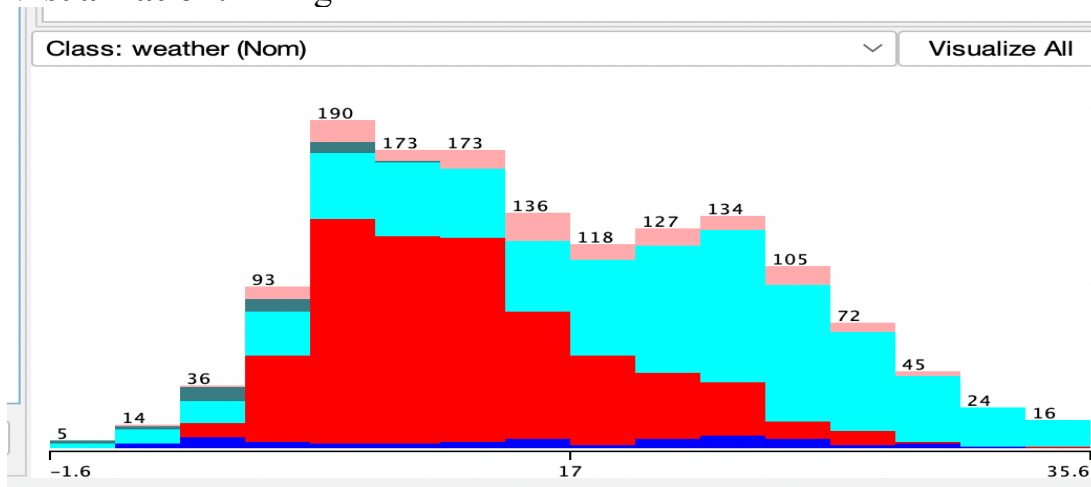
=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	drizzle
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	rain
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	sun
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	snow
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	fog
Weighted Avg.	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	

=== Confusion Matrix ===

a	b	c	d	e	← classified as
53	0	0	0	0	a = drizzle
0	641	0	0	0	b = rain
0	0	640	0	0	c = sun
0	0	0	26	0	d = snow
0	0	0	0	101	e = fog

**Visualization:** This gives overall information about the weather.



### Conclusion:

By comparing all these modelling techniques, we can say that Random Forest and k-star have most instances with 100%. Whereas, Naive Bayes have 86% and SMO having 99 % correctly classified instances.

By using these modelling techniques in Weka tool, we can predict weather.

GitHub:

<https://github.com/sudheerredde/BLACKSHARKS.git>