Improving Conservation of Animal Species: understanding the decision factors for migration

COMP 551: Applied Machine Learning

Arnaud Yoh Massenet-Oshima
arnaud.massenet@mail.mcgill.ca
260608613

Fan Ma
fan.ma@mail.mcgill.ca
260589203

Theophile Gervet
theophile.gervet@mail.mcgill.ca
260612917

I. Introduction and Related Works

To successfully conserve and manage migrating animal populations, it is essential to understand when animals move, where they move, and why they move. Indeed, migrating species are at higher risk of extinction : because they exploit multiple seasonal habitats, they are particularly susceptible to habitat loss and fragmentation [6]. With better understanding of a species behavior, we could better protect its seasonal habitats, and more accurately forecast population status and habitat needs in future conditions (e.g predict the impact of climate change) [2].

Environmental variables have a direct influence on the need to move : they govern the food availability as well as the general life conditions at the place of departure. They also directly affect the capacity to move : variation in wind speed and uplift can half or double the energetic cost for aerial migrants [3], [4].

Large-scale movements like migrations have always been some of the most challenging movements to study in their environmental context. This stems from the fact that environmental conditions cannot be measured locally as part of the study, they are needed over extensive areas.

With the rapid improvement of tracking technology, movement ecology is entering the big data area. We are obtaining more and more tracking data at previously unseen spatial and temporal resolution. At the same time, large remote sensing data sets recording environmental variables across time and space become widely available [2].

Combining these data sets should allow us to develop more accurate models of migrational behavior through better representation of the interaction between animal movement and the environment.

An interesting past approach has been to represent alternative hypotheses regarding the environmental drivers of animal movement as an array of competing agent-based models [5]. These models are then scored based on their ability to reproduce observed animal movement given known environmental conditions. These models performed fairly well and gave empirical evidence that environmental variables (vegetation indexes, precipitation measures) influence the migrating behavior of zebras [6] and elephants [7] in central Africa.

We would like to see if environmental variables influence the departure date of migrating barnacle geese in Northern Europe.

II. Problem Representation

A. Tracking data

The barnacle goose is a medium sized goose 55-70 cm long with a wingspan of 130-145 cm and a body mass of 1.21-2.23 kg. Barnacle geese breed mainly in the Arctic islands of the North Atlantic. There are three main populations, all represented on MoveBank (figure 1), with separate breeding and wintering ranges, from west to east :

- Breeding in eastern Greenland, wintering in Scotland and Ireland (7 individuals on Movebank).
- Breeding on Svalbard, wintering in England and Scotland (21 individuals on Movebank).
- Breeding on Novaya Zemlya, wintering in the Netherlands (12 individuals on Movebank).

Figure 1: Barnacle geese seasonal migrations between 2008 and 2011 in Northern Europe.

The three data sets are pretty sparse when it comes to the trajectory points, most of the points are clustered at one end or

the other. However compared to most other data sets available on Movebank, we have many examples of migrations. With one round-trip a year per bird for 40 birds over 3 years, we have about 120 individual round trip examples and 240 individual trip examples.

We had to come up with a question that could leverage data agglomerated at the end points of the trajectory and many individual examples of migrations. We decided to predict the departure date of a trip from time-series of environmental data.

Before we investigate in greater detail what environmental variables we decided to use, let us first motivate our approach.

B. Two Different Approaches

Our first most natural approach was to predict the departure date as a function of environmental variables over a given period of time at the place of departure. Inputs in our model would be time series of environmental variables over the place of departure, the output would be the date of departure. Remark that for this approach to work, it is essential that we have environmental data at the place of departure even after the bird has left.

It turns out that this kind of data is pretty painful to gather. We contacted Movebank for some advice about how to get such data and they pointed out it is much easier to get environmental data along the tracks of the animals than over a fixed gridded area. This leads to our second and final approach.

We want to predict if yes or no a bird will migrate at some given date, from time series of environmental variables along its tracks for the k previous days. We went from a regression task to a binary classification task.

The goal is to both understand what environment data influences the choice of the "departure date", and how well we can make predictions even with a small amount of data.

C. Environmental Data Annotation

We hypothesize geese move South when 1) it gets too cold and 2) food starts lacking. When the temperature drops below a certain threshold, lakes where they take refuge from predators freeze and vegetation dies. Since geese are primarily herbivores, a good indicator for food availability would be vegetation indexes, as measured by the Normalized Difference Vegetation Index (NDVI).

We will use the surface air temperature provided by the European Centre for Medium-Range Weather Forecasts (ECWMF).

The NDVI index is provided by MODIS (satellite imaging data from the NASA). It exploits a contrast in reflectance in the near-infrared (Rnir), and red portions of the electromagnetic spectrum (Rred) which indicates photo-synthetically active vegetation. Hence NDVI, defined as $(Rnir - Rred)/(Rnir + Rred)$ is high when there is a lot of photo-synthetically active vegetation, and low when there is not. It is available at high spatial and decent temporal resolution from remote sensing and it has been successfully used across multiple similar studies [3], [6], [7].

Both NDVI and temperature data values were annotated to the tracks using inverse weighted distance interpolation in space and time. This was done for us by the Env-DATA system provided by Movebank [2].

D. Data Preprocessing

The most challenging task of the preprocessing was to label the data points. Since we are doing a binary classification we had to separate the data points in two groups : "Departure point", labeled 1, and "Non-Departure point", labeled 0. We define "Departure point" as the last data point before a goose starts migrating.

We also had to differentiate between "Departure points" for north-bound migration from "Departure points" for south-bound migration. The reason behind this separation is that, even if both are "Departure points", they don't represent the same phenomenon and might not be influenced by environmental variables in the same way.

In order to perform this labeling task we tried separating the data set into three clusters : 1) flying, 2) resting in the north and 3) resting in the south, using k-means on longitude and latitude values. Due to the poor results, we opted to separate our points through a deterministic algorithm. For point n_i to be a departure point, it had to meet the following conditions.

- n_i and n_{i+1}, n_{i+1} and n_{i+2}, n_{i+2} and n_{i+3} all have to be more than 1 degree in latitude OR longitude farther.
- n_i and n_{i-1} have to be less than 1 degree in latitude AND longitude farther.

After this first step, we gathered data points into groups as follows :

- For each "Departure point" create a group
- Add data points corresponding to the 30 days preceding the "Departure Point" to this group

For every positive example ("Departure point") extracted from our data set, we randomly sampled a negative example. To do so we sampled "Non Departure points" for negative examples randomly in the range of "Departure points" and extracted data points for the previous 30 days as we did with positive examples. As a result our training data is balanced between positive and negative samples, and reflects the same date range for both classes.

We also modified the representation of time-stamps. The original data has time-stamps based on the ISO format (i.e. $dd/mm/yyyy$). This format does not capture the period of the year, and it distinguishes between different years. Our goal is to use data that can represent a "period" in any given year, we do not want to distinguish between different years. This lead us to use the Julian calendar format which represents dates in a very simple manner : January 1st is represented by 1 while December 31st is 365 (or 366 if we have a February 29th in our year).

E. Unsupervised K-means clustering of departure and non-departure points

Our script to label departure is very specific (different north to south, different from animals, different at different times).

Because of this, we attempted to generalize the task of labeling departure points by applying K-means clustering on the raw data. K-means cannot be used to answer our main question because we are using information after onset of the departure point to predict whether a point is a departure point. We can't use future data to make our predictions in a real life scenario. We tried to use longitude, latitude, heading and speed as features in our K-means. We labeled a subset of our data manually and compared it to the results in our K-means. We also compared it to one generated by our script. The results were not good enough. 60% of the data points manually labeled were in accordance to the result given by our K-means. We need this number to be much higher and decided to discontinue the usage of K-means clustering to label our data.

F. Final Set of Features

For each data point in each group we kept the following features :

longitude gradient (Naive Bayes only due to time constraint) Velocity, calculated by taking the instantaneous change of longitude.

latitude gradient (Naive Bayes only due to time constraint) Velocity, calculated by taking the instantaneous change of latitude.

Time-stamp We keep the time-stamp so that we can compare our predictions using only environmental variables to the obvious baseline : predicting if a bird will leave or not at a certain date from the date itself.

Temperature We hypothesize that birds will leave an area because of low temperatures during multiple consecutive days. The data we extracted confirms our observation since, birds seem to leave when temperatures are going just below 10 °C.

Vegetation indexes on Land Another critical environment element that should influence migration dates is food abundance. The diet of a goose is mostly vegetarian, it is made of grass, sedge, roots, moss and herbs [12]. Thus land vegetation indexes can well describe the abundance of food for the geese we are studying.

Vegetation indexes in Aquatic Environment Additionally the diet of a goose can also contain seaweeds and maritime plant shoots [12].

However, one considerable pre-processing technique was to replace longitude and latitude by the instantaneous change in longitude and latitude. Fig.2 shows a visualization of this feature transformation. The different positions of the geese are plotted as points. To that , we add the vector of the geese's velocity at that point represented by arrows. There are 2 useful pieces of information that can be surfaced with this pre-processing step. The difference between positive examples and negative examples reside in their homogeneity and the magnitude of vectors. For homogeneity, what we mean is that the positive examples have vectors in similar directions while the negative examples have them more spread out erratically. Keep in mind that these are all points preceding the departure point,

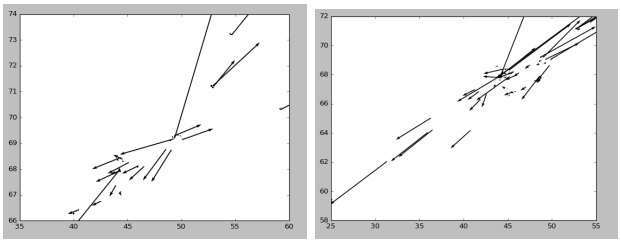

Figure 2: Map of instantaneous velocity gradient vectors for one example set. x-axis: longitude, y-axis: latitude. [Left] Negative example. [Right] Positive example. Arrows are not on the same scale as the axis (position).

which means there are no actual migration vectors (vectors during migration). This observation suggests that there may be some sort of pattern in the goose's displacement prior to migration. By looking at the graph, it seems that geese fly less erratically prior to migration. The second piece of information is magnitude. It goes without saying that the displacement of the last point (the departure point) is considerably larger than the displacement of a goose in its "everyday life". This piece of information is less obvious from Fig.2, but this is a useful feature that our pre-processing technique of calculating instantaneous velocity added to our model.

III. Algorithm Selection and Implementation

We selected the following algorithms for two different tasks. One group of algorithm was selected based on the easiness of the features' weights interpretation, while the other was selected based on prediction performance. The first group consists of Naive Bayes, Logistic Regression and Decision Tree. The second group comprises Gaussian SVM and RNN with LSTM. For both groups we will select the best hyper-parameters for prediction accuracy on a validation set. This is critical for the first group because if a learner has a good prediction accuracy, then it understands well how geese decide when to leave, thus the weights of the features are well picked. This also critical for the second group as we want to achieve the best prediction performance possible.

A. Baseline Linear Classification Algorithms

We chose three different classifiers : Naive Bayes, Support Vector Machines and Logistic Regression. The motivation behind these choices is that Logistic Regression is easy to interpret, thus it allow us to better learn about the important factors influencing the departure date of the geese. We added the Support Vector Machines and Naive Bayes. Naive Bayes is a simple and fast learner that can make us realize how hard the prediction task is, while SVM is a learner that usually outperforms the two other ones.

1) Naïve Bayes: We used Scikit-Learn's implementation of Naïve Bayes. We attempted to fit both Gaussian and Bernoulli distributions to our model. The Bernoulli distribution fits our use case best because of the binomial outcome, but

the Guassian distribution can capture better our "flattened timeseries".

2) Logistic Regression: We decided to consider the Scikit-Learn implementation of Logistic Regression with a the "liblinear" solver. This solver uses a coordinate descent. It will update x_n as follows:

$$x_{k+1,i} = argminf(x_{k+1,1}, ..., x_{k+1,i-1}, y, x_{k+1,i+1}, ..., x_{k,n})$$

We use this solver because it usually performs better than the other ones when used with small amount of data [17]. We chose to use the $L2$ regularization over the $L1$ as we want to keep non-zero values for the weights to better understand the influence of each feature on the output.

Another important hyper-parameter is C. C represents a trade-off between low training error and the simplicity of the resulting model. A high value for C will give low training error, while a low value will create a smoother decision boundary. As you can expect, a too high C value will eventually cause over-fitting while a too small C will give poor classification. This hyper-parameter was chosen based on validation error using a 5-fold cross validation (*see figure 2*). We can see that the sweet spot is located at $C = 10$.

Figure 3: Accuracy of the prediction of Logistic Regression on a leave-one-out cross-validation with values of C ranging from 10^{-4} to 10^4 using a logarithmic steps.

From *Figure 2*, we can see that the sweet spot is located at $C = 10$.

Lastly we had to choose the size of the look-back window k. We fixed $C = 10$ and after using leave-one-out cross validation, we found that $k = 7$ was an optimal choice.

3) Support Vector Machine: We decided to consider the Scikit-Learn implementation of the Gaussian-Kernel SVM. The argument supporting the choice of this kernel is the prediction performance of non-linear kernels. Indeed non-linear kernels usually perform better than linear-kernels, especially if our feature space is not-linearly separable [15].

We decided to use the C-SVM implementation of scikit-learn. Note that the hyper-parameter C in C-SVM has the same role as the C in our Logistic Regression.

Similarly to Logistic Regression, C is the main hyper-parameter needed to be set in the C-SVM implementation of scikit-learn. C represents the trade-off we are willing to make between training error and complexity if the decision boundary.

In order to determine the best value for C we performed multiple leave-one-out cross validation and then we examined the validation and the training error (*see Figure 3*).

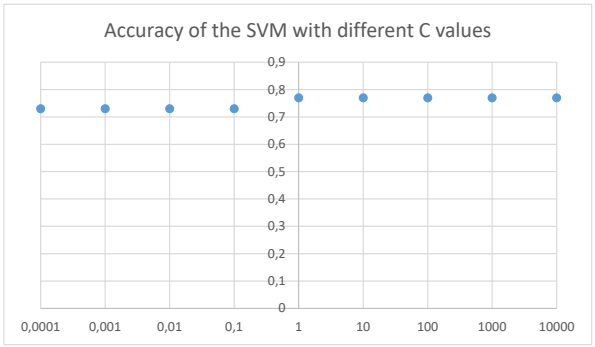

Figure 4: Accuracy of the prediction of the SVM on a leave-one-out cross-validation with values of C ranging from 10^{-4} to 10^4 using a logarithmic steps.

The last hyper-parameter we will discuss is γ. γ is a parameter of the Gaussian Kernel that takes care of the non-linear classification [18]. A larger γ gives smoother curves. Thus we will have a high bias and low variance with a high γ value, while a low γ value will give a low bias and a high variance.

We will determine the optimal parameters C and γ by using a $5x5$ exhaustive grid search with 5 values of C and 5 values for γ. The results can be seen in **Table I**.

According to the various results we computed by tweaking C and γ, it seems that our sweet spot is when we have $C = 1$ and $\gamma = 0.001$

Lastly we selected the size of the look-back window k. We fixed $C = 1$ and $\gamma = 0.001$ and after using leave-one-out cross validation, we found that $k = 6$ was an optimal choice.

B. Non-Linear Classification Algorithm: Decision Tree

Along with linear classifiers, we decided to include non-linear learners. This was motivated by the idea that non-linear

Validation Score % over C and γ						
	γ					
C		0.001	0.1	1	10	100
	0.1	0.81	0.65	0.58	0.58	0.5
	1.0	0.85	0.81	0.58	0.54	0.5
	10	0.81	0.77	0.58	0.54	0.5
	100	0.77	0.77	0.58	0.54	0.5
	1000	0.77	0.77	0.58	0.54	0.5

Table I: Validation Accuracy Score with different value for C and γ

classifiers could interpret the data differently and give weights in a different manner.

This is why we choose to use decision trees, which, instead of using regularization, use Information Gain when deciding what feature to use. Note that we decided to avoid using the random forest. Again, the reason we decided to avoid using Random Forest, is the scarcity of data we have. Random forest tend to perform poorly compared to decision trees when it comes to small data [13].

The other advantage of decision trees over random forest is the easiness of interpreting the model to understand which features have the biggest influence on the outcome [13].

We chose to use the scikit learn implementation of decision trees. For the choice of features, we decided to try both the "Entropy" and the "Gini" method. The "Entropy" method selects features with the best "Information Gain" while "Gini" selects features with the best "Gini Score". Both scores are in fact similar and use impurity based splitting criteria. They have different impurity functions which are particular values of a more generalized measure called *Tsallis' Entropy*:

$$H_\beta(E) = \frac{1}{\beta - 1}\left(1 - \sum_{j=1}^{c} p_j^\beta\right)$$

Gini's Score is when we set $\beta = 2$:

$$Gini(E) = 1 - \sum_{j=1}^{c} p_j^2$$

And the Entropy is defined when $\beta \to 1$:

$$Entropy(E) = -\sum_{j=1}^{c} p_j \log p_j$$

We then use leave-one-out cross validation to select our "lookback window" size k. This gave use $k = 10$ for both methods, giving us 88% accuracy for the "Gini" method and 85% for the "Entropy" method.

C. Recurrent Neural Network Approach

The task at hand can be qualified as supervised sequence labeling : given values for environmental variables over multiple time-stamps, we want to classify the sequence as positive (the goose migrates at the end of the sequence) or negative (the goose stays at the end of the sequence).

Recurrent neural networks (RNNs) are able to incorporate context information in a flexible way, and are robust to localized distortions in the input data, which makes them particularly well suited for this kind of task [8]. Long short-term memory networks (LSTMs) get rid of the vanishing/exploding gradient problem and allow us to classify more accurately longer sequences. Figure 2 illustrates the form the network takes : red vectors are inputs (environmental variables at subsequent timestamps), green vectors represent the recurrent hidden layer, and the blue vector is the binary output.

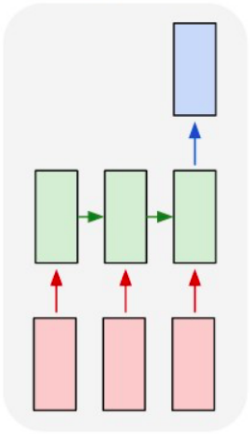

Figure 5: RNN model for sequence classification (from blog post by A. Karpathy)

We implemented our LSTM network using the Keras deep-learning library, built on top of Theano and Tensorflow. The network takes as input a sequence of time-stamps of environmental variables and outputs the probability that the bird migrates given this sequence.

We kept the architecture simple given our limited amount of data : one layer of h LSTM nodes where h is to be determined by cross-validation, followed by a sigmoid output layer that allows us to interpret the output as a probability. We used binary cross-entropy as the loss function and the recent Adam [9] algorithm based on the unpublished RMSprop algorithm for optimization (per parameter adaptive learning rate method). We will use a batch size of one so that the internal state of the LSTM units resets after each example sequence (Keras' implementation of LSTM resets internal state after each batch by default).

For regularization, we will use Srivastava et al.'s dropout technique [10] which consists in randomly dropping units from layers in the neural network, along with their connections. This technique is reported to prevent neurons from developing overly complex co-adaptations. In an LSTM network, dropout can be applied either in between layers, or more specifically to the input and recurrent connections of the memory units. We will cross-validate over these different schemes.

Speaking about cross-validation, we also need to select the best k (length of the sequences) and h (number of nodes in the LSTM layer), as well as the optimization hyper-parameters. The modest size of our data set allows us to perform an extensive grid search with 5 fold cross-validation.

IV. Results

Before we present any results using environmental variables to make predictions, let us fix a simple baseline. Because of the way we extracted our training data (balanced classes and same date ranges for both classes), the simplest classifier which uses only the date of the departure (or non departure) point achieves 50% accuracy. We would like to best this random predictor.

Note that the date would be a very strong indicator if we sampled negative examples from the whole year, but since we sampled negative examples from the date range of positive examples, it is equivalent to a random classifier.

A. Baseline Linear Classification Algorithms

1) Naïve Bayes: We obtained satisfactory results using Naïve Bayes without regularization. The results are in accordance with other linear classifiers. Naïve Bayes makes the strong assumption of independence of features. However, this is not true for all features. Features like temperature, aquatic vegetation and land vegetation have some elements of dependence. They are not strongly dependent of each other, which is why the Naïve Bayes model still performs admirably. We chose to fit a Bernoulli distribution because of our binomial outcome. The Bernoulli Naïve Bayes outperformed the Gaussian Naïve Bayes by 20% on the validation set and remained constant with it on the training set.

Precision	Accuracy	F1-score	Recall
0.70	0.80	0.74	0.71

Table II: Validation Score of Bernoulli Naïve Bayes

2) Logistic Regression: Using the hyper-parameters we found in the previous section $IV.A.2$) we can extract the weight of the feature we found. Recall that the optimal value we have are $C = 10$ and $k = 7$. Using these value we got the scores for the prediction on the validation set. The results can be seen in **Table III**.

Precision	Accuracy	F1-score	Recall
0.73	0.85	0.74	0.77

Table III: Validation Score of our Logistic Regression with $C = 10$ and $k = 7$

It seems that our results seems promising given the scarcity of the data we had to face.

After these results, we computed the weights that were assigned to our feature using the Logistic Regression classifier $coef\_$ attribute (*see Table IV*) .

Timestamp	Temperature	Vegetation on Land	Vegetation in Water
0.62	0.95	0.15	0.13

Table IV: Weights of the features computed for Logistic Regression

This results are a bit surprising as we learn that the vegetation indices seems less important than what we assumed at the beginning of the project. However it is clear that temperature plays a big role in the departure date decision. Note also that the value $k = 7$ we found also tells us that birds are likely to make the decision to leave based on the environment conditions of the past 7 days.

3) Support Vector Machines: Using the hyper-parameters we found in the previous section $IV.A.3$) we got the scores for the prediction on the validation set. Recall that the optimal value we have are $C = 1$ $\gamma = 0.001$ and $k = 6$.The results can be seen in **Table V**.

Precision	Accuracy	F1-score	Recall
0.80	0.85	0.81	0.78

Table V: Validation Score of our SVM with $C = 1$, $\gamma = 0.001$ and $k = 6$

Surprisingly the Gaussian-SVM classifier did not performed as well as expected since it was outperformed by the Logistic Regression.

B. Non-Linear Classification Algorithm: Decision Tree

Using the best value for the size of the "lookback window" $k = 10$ for both type of decision trees, we computed their prediction performance (*see Table VI and VII*

Precision	Accuracy	F1-score	Recall
0.81	0.85	0.80	0.92

Table VI: Validation Score of the "Gini" Decision Tree using $k = 10$

Precision	Accuracy	F1-score	Recall
0.77	0.85	0.85	0.92

Table VII: Validation Score of our SVM with $C = 1$, $\gamma = 0.001$ and $k = 6$

Interestingly, we can see that we have quite similar performance between the two methods. Especially we have the same score for Accuracy and the Recall score is very high in both cases.

Now we computed the $feature\_importance\_$ attribute of both classifiers.This attribute presents itself as an array with a value that can be mapped to each feature. This allows us to learn about the weight each feature had in the decision tree. The higher the value, the more important is a feature. We can the results in the Table VIII and the Table IX

Timestamp	Temperature	Vegetation on Land	Vegetation in Water
0.05	0.00	0.05	0.00

Table VIII: Weights of the features computed for the "Gini" Decision Tree

Timestamp	Temperature	Vegetation on Land	Vegetation in Water
0.05	0.00	0.05	0.00

Table IX: Weights of the features computed for the "Entropy" Decision Tree

These results show a flaws in our idea that decision trees will inform well about the weight of each features. The issue here is the way our data is represented. Recall that each input is a aggregation of data from different dates put one after another. This means that the learner will consider the temperature on two different dates as completely unrelated features. The issue now is that because of the scarcity of our data, the tree is more likely to select fewer features (it actually selected 2 features out of 40, considering that we used 10 days, 4 features per day).

C. Recurrent Neural Network Approach

We selected the best values for k (length of the sequence) and h (number of nodes in the LSTM layer) using an extensive grid search with 5-fold cross validation. The best k turned out to be 30, we couldn't try out higher values of k since we extracted only 30 days in our preprocessing step, and the best h was 4. Our final model used 0.2 dropout at the input and recurrent gates of the LSTM units, since this scheme performed better than the classic 0.5 dropout between layers.

With these settings, taking the mean over all cross-validation folds, we obtained almost 100% training accuracy and around 90% validation accuracy. A training run of our final model is illustrated below (*see Figure 6*). Both training and validation accuracy start at 50% since we constructed balanced training and validation sets. After about 60 epochs, we reach 90% validation accuracy before we start over-training and validation accuracy goes down again.

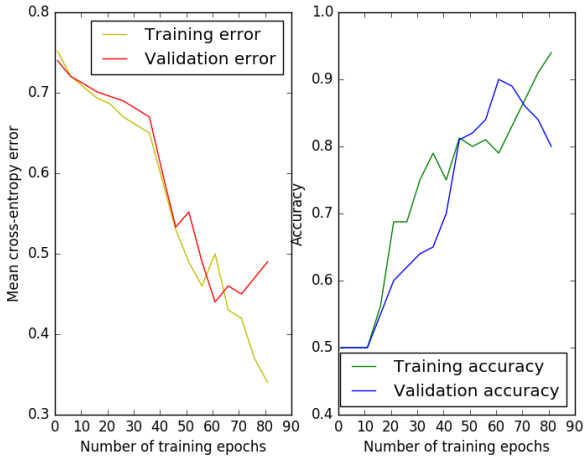

Figure 6: Accuracy and mean cross-entropy loss as a function of the number of training epochs.

V. Discussion

Being aware that the data provided by Movebank is scarce and the amount of actual usable data is scarcer, we are pleasantly surprised by the results we got through the various supervised machine learning methods we used. By far, the best method is the LSTM model which reaches 90% validation accuracy and 100% training accuracy. It should come as no surprise that the LSTM's ability to incorporate longer spans of sequences comes in handy in a task which makes high use of temporally dependent data. The second advantage of LSTMs is their ability to learn more complex decision boundaries. The fact that LSTMs do so much better than our other classifiers hint at the suspicion that to best predict when an animal migrates, it actually takes more than the simple intuition of migrating when the temperature reaches a certain point, when vegetation below a threshold or what is the pattern of movements of the animal at rest. And of course, probably the biggest player is the temporal aspect. Indeed, even if our linear classifiers incorporate more than just one data point in the past, it does so in a "flattened" way and cannot learn a decision boundary that is super linear.

Still for the LSTM, we selected many features, but that did not turn our to be our bottleneck. Indeed, as we were testing LSTMs, we reached our maximum window of 30 days. Unfortunately, the trend was still growing, which means that the LSTM might have performed even better if our preprocessing steps accommodated a window larger than 30 days.

We also had to face a failure, which is best represented by the decision tree algorithm. The way we constructed our data, for the learners other than the RNN, was not suitable for decision trees given the small amount of data we had.

However, the weight we gather from Logistic Regression seems to support the fact that temperature really plays a big role in the "departure date" decision making, which is validated by observations we can make about geese. That is not to say other features play less roles. Another thing to consider is interaction effects between features.

The goal of this project was to understand how geese decide when to start migrating by using Machine Learning. Obviously this project could be in fact applied to many more species that have migration habits, like birds, whales or dolphins. However, in order to have good results, it would be advisable to have much more data to avoid biased outcomes.

Future Work

We believe that our model and methodology can be applied to a wide variety of migratory species with strong environmental cues. We would also like to explore in detail the effects of context window size for LSTMs as we focused more on feature engineering in this project. The window size became a possible bottleneck, something that can only be confirmed if we ran more experiments.

VI. Statement of Contributions

Arnaud Yoh Massenet-Oshima

I implemented and evaluated the data preprocessing and the feature selection steps along with the Logistic Regression, the SVM and the Decision Tree. I was in charge of describing these works in the report and part of the discussion section.

Fan Ma

Fan implemented and wrote about K-means, gradient map features preprocessing, Naive Bayes and part of the discussion.

Theophile Gervet

I was primarily responsible for the implementation, evaluation and analysis of the RNN approach, as well as for the introduction and related works of the report.

We hereby state that all the work presented in this report is that of the authors.

REFERENCES

[1] Christopher D. Manning, Prabhakar Raghavan and Hinrich Schutze, *Introduction to Information Retrieval*, Text classification and Naive Bayes, Chapter 13, Cambridge University Press, 2008.

[2] Dodge S., Bohrer G., Weinzierl R., Davidson S.C., Kays R., Douglas D., Cruz S., Han J., Brandes D., and Wilkelski M., *The Environmental-Data Automated Track Annotation (Env-DATA) System: Linking animal tracks with environmental data*, Movement Ecology, v. 1:3, 2013.

[3] Somayeh Dodge, Gil Bohrer, Keith Bildstein, Sarah C. Davidson, Rolf Weinzierl, Marc J. Bechard, David Barber, Roland Kays, David Brandes, Jiawei Han, Martin Wikelski, *Environmental drivers of variability in the movement ecology of turkey vultures (Cathartes aura) in North and South America* The Royal Society, 2014.

[4] Greg W Mitchell, Bradley K Woodworth, Philip D Taylor and D Ryan Norris, *Automated telemetry reveals age specific differences in flight duration and speed are driven by wind conditions in a migratory songbird*, Movement ecology, 2015.

[5] Bunnefeld N, Boerger L, van Moorter B, Rolandsen CM, Dettki H, Solberg EJ, Ericsson G, *A model-driven approach to quantify migration patterns: individual, regional and yearly differences*, J Anim Ecol, 2001.

[6] Hattie L. A. Bartlam-Brooks, Pieter S. A. Beck, Gil Bohrer, Stephen Harris, *In search of greener pastures: Using satellite images to predict the effects of environmental change on zebra migration*, Journal of Geophysical Research: Biogeosciences, vol 118, 2013.

[7] Gil Bohrer, Pieter SA Beck, Shadrack M Ngene, Andrew K Skidmore, Ian Douglas-Hamilton, *Elephant movement closely tracks precipitation-driven vegetation dynamics in a Kenyan forest-savanna landscape*, Movement Ecology, 2014.

[8] Alex Graves, *Supervised Sequence Labelling with Recurrent Neural Networks*, Springer, 2012.

[9] Diederik P. Kingma, Jimmy Lei Ba, *Adam: A Method for Stochastic Optimization*, ICLR, 2015.

[10] N Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, *Dropout: A simple Way to Prevent Neural Networks from Overfitting*, The Journal of Machine Learning Research, 2014.

[11] A. Krizhevsky, I. Sutskever, G. E. Hinton, *ImageNet Classification with Deep Convolutional Neural Networks*, NIPS, 2012.

[12] Gill, F and D Donsker, *IOC World Bird List (v4.3)*, 2014

[13] Jehad Ali,Rehanullah Khan, Nasir Ahmad,Imran Maqsood *Random Forests and Decision Trees* , IJCSI International Journal of Computer Science Issue, 2012

[14] Nidhi . Ruparel, Nitin M. Shahane,Devyani P. Bhamare, *Learning from Small Data Set to Build Classification Model: a survey*, International Journal of Computer Application, 2013

[15] Keerthi, S. Sathiya, and Chih-Jen Lin, *Asymptotic behaviors of support vector machines with Gaussian kernel*, Neural computation 15.7 (2003): 1667-1689.

[16] Chih-Chung Chang,Chih-Jen Lin, *Training Nu-Support Vector Classifiers: Theory and Algorithms*,

[17] *sklearn linearmodel LogisticRegression*, http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[18] Pedregosa, Fabian, et al. "Scikit-learn: Machine learning in Python." Journal of Machine Learning Research 12.Oct (2011): 2825-2830.
Random Forests and Decision Trees