CLIMATE PREDICTION SYSTEM

SIVA KARTHIK GADE

VISHAL DESAI
PRASHANT KAGWAD

MADHUMITA DANGE

GAURAV DEY

PRASANJIT KHUNTIYA

KARAN LUNIYA

Understanding the Data Set and Pre-Processing:

The dataset is from NCDC is responsible for preserving, monitoring, assessing, and providing public access to the Nation's treasure of climate and historical weather data and information from 2007-2014. NCDC puts a priority on interpreting and applying scientific understanding to our extensive array of climate datasets. Some of the products and output derived from this effort include extreme event and dynamically generated climate information. Detailed descriptions of the information available are below. The climate of the United States varies by location and by time of year. Climate Normals, monthly climate reports, and drought information are a few of the many datasets and products found under our climate section. NCDC offers several types of climate information generated from examination of the data in the archives. These types of information include record temperatures, record precipitation and snowfall, climate extremes statistics, and other derived climate products. The data had a lot of discrepancy and therefore we need to provide a systematic approach to resolve the bugs and to gather the useful data. The data did not have the State location mapped to the dataset, instead the data was provided with a station Id. These weather stations have to be mapped with the states and then can be used for our further analysis. The weather data had few records which were not mapping with any state so we had to consider those data as different states. Then the data was loaded in chunks since the data was huge in numbers therefore we wrote a java code to process the data. The data was assumed to be in slots since we had continuous values for each variable therefore we found the maximum and minimum of each variable. Then we created slots of 10-20 for each variable and this data was fed into our algorithm to create structural algorithms. This algorithm is based on Chow Liu Learning which expects the data in a range values. Data pre-processing is an important step in the data mining process. The phrase "garbage in, garbage out" is particularly applicable to data mining and machine learning projects. Data-gathering methods are often loosely controlled, resulting in out-of-range values (e.g., Income: −100), impossible data combinations (e.g., Sex: Male, Pregnant: Yes), missing values, etc. Analyzing data that has not been carefully screened for such problems can produce misleading results. Thus, the representation and quality of data is first and foremost before running an analysis. If there is much irrelevant and redundant information present or noisy and unreliable data, then knowledge discovery during the training phase is more difficult. Data preparation and filtering steps can take considerable amount of processing time. Data pre-processing includes cleaning, normalization, transformation, feature extraction and selection, etc. The product of data pre-processing is the final training set.

Model generation using Structure Learning and Parameter Learning:

Chow Liu algorithm for structure learning:

Chow and Liu show how to select second-order terms for the product approximation so that, among all such second-order approximations (first-order dependency trees), the constructed approximation P' has the minimum Kullback–Leibler distance to the actual distribution P, and is thus the closest approximation in the classical information-theoreticsense. The Kullback–Leibler distance between a second-order product approximation and the actual distribution is shown to be

$$D(P \parallel P') = -\sum I(X_i; X_{j(i)}) + \sum H(X_i) - H(X_1, X_2, \ldots, X_n)$$

where $I(X_i; X_{j(i)})$ is the mutual information between variable X_i and its parent $X_{j(i)}$ and $H(X_1, X_2, \ldots, X_n)$ is the joint entropy of variable set $\{X_1, X_2, \ldots, X_n\}$. Since the terms $\sum H(X_i)$ and $H(X_1, X_2, \ldots, X_n)$ are independent of the dependency ordering in the tree, only the sum of the pairwise mutual informations, $\sum I(X_i; X_{j(i)})$, determines the quality of the approximation. Thus, if every branch (edge) on the tree is given a weight corresponding to the mutual information between the variables at its vertices, then the tree which provides the optimal second-order approximation to the target distribution is just the maximum-weight tree. The equation above also highlights the role of the dependencies in the approximation: When no dependencies exist, and the first term in the equation is absent, we have only an approximation based on first-order marginals, and the distance between the approximation and the true distribution is due to the redundancies that are not accounted for when the variables are treated as independent. As we specify second-order dependencies, we begin to capture some of that structure and reduce the distance between the two distributions.

Chow and Liu provide a simple algorithm for constructing the optimal tree; at each stage of the procedure the algorithm simply adds the maximum mutual information pair to the tree. See the original paper, Chow

& Liu (1968), for full details. A more efficient tree construction algorithm for the common case of sparse data was outlined in Meilă (1999).

Chow and Wagner proved in a later paper Chow & Wagner (1973) that the learning of the Chow–Liu tree is consistent given samples (or observations) drawn i.i.d. from a tree-structured distribution. In other words, the probability of learning an incorrect tree decays to zero as the number of samples tends to infinity. The main idea in the proof is the continuity of the mutual information in the pairwise marginal distribution. Recently, the exponential rate of convergence of the error probability was provided.

MLE for parameter learning:

In statistics, maximum-likelihood estimation (MLE) is a method of estimating the parameters of a statistical model. When applied to a data set and given a statistical model, maximum-likelihood estimation provides estimates for the model's parameters.

The method of maximum likelihood corresponds to many well-known estimation methods in statistics. For example, one may be interested in the heights of adult female penguins, but be unable to measure the height of every single penguin in a population due to cost or time constraints. Assuming that the heights are normally (Gaussian) distributed with some unknown mean and variance, the mean and variance can be estimated with MLE while only knowing the heights of some sample of the overall population. MLE would accomplish this by taking the mean and variance as parameters and finding particular parametric values that make the observed results the most probable (given the model).

In general, for a fixed set of data and underlying statistical model, the method of maximum likelihood selects the set of values of the model parameters that maximizes the likelihood function. Intuitively, this maximizes the "agreement" of the selected model with the observed data, and for discrete random variables it indeed maximizes the probability of the observed data under the resulting distribution. Maximum-likelihood estimation gives a unified approach to estimation, which is well-defined in the case of the normal distribution and many other problems. However, in some complicated problems, difficulties do occur: in such problems, maximum-likelihood estimators are unsuitable or do not exist.

To use the method of maximum likelihood, one first specifies the joint density function for all observations. For an independent and identically distributed sample, this joint density function is

$$f(x_1, x_2, \ldots, x_n \mid \theta) = f(x_1|\theta) \times f(x_2|\theta) \times \cdots \times f(x_n|\theta).$$

Now we look at this function from a different perspective by considering the observed values x_1, x_2, …, x_n to be fixed "parameters" of this function, whereas θ will be the function's variable and allowed to vary freely; this function will be called the likelihood:

$$\mathcal{L}(\theta\,;\,x_1, \ldots, x_n) = f(x_1, x_2, \ldots, x_n \mid \theta) = \prod_{i=1}^{n} f(x_i|\theta).$$

Note $;$ denotes a separation between the two input arguments: θ and the vector-valued input $x_1, \ldots, x_n$.

In practice it is often more convenient to work with the logarithm of the likelihood function, called the **log-likelihood**:

$$\ln \mathcal{L}(\theta\,;\,x_1, \ldots, x_n) = \sum_{i=1}^{n} \ln f(x_i|\theta),$$

or the average log-likelihood:

$$\hat{\ell} = \frac{1}{n} \ln \mathcal{L}.$$

The hat over ℓ indicates that it is akin to some estimator. Indeed, $\hat{\ell}$ estimates the expected log-likelihood of a single observation in the model.

The method of maximum likelihood estimates θ_0 by finding a value of θ that maximizes $\hat{\ell}(\theta;x)$. This method of estimation defines a **maximum-likelihood estimator** (**MLE**) of θ_0 …

$$\{\hat{\theta}_{\mathrm{mle}}\} \subseteq \{\underset{\theta\in\Theta}{\arg\max}\ \hat{\ell}(\theta\,;\,x_1, \ldots, x_n)\}.$$

… if any maximum exists. An MLE estimate is the same regardless of whether we maximize the likelihood or the log-likelihood function, since log is a strictly monotonically increasing function.

For many models, a maximum likelihood estimator can be found as an explicit function of the observed data x_1, …, x_n. For many other models, however, no closed-form solution to the maximization problem is known or available, and an MLE has to be found numerically using optimization methods. For some problems, there may be multiple estimates that maximize the likelihood. For other problems, no maximum likelihood estimate exists (meaning that the log-likelihood function increases without attaining the supremum value).

Bayesian network:

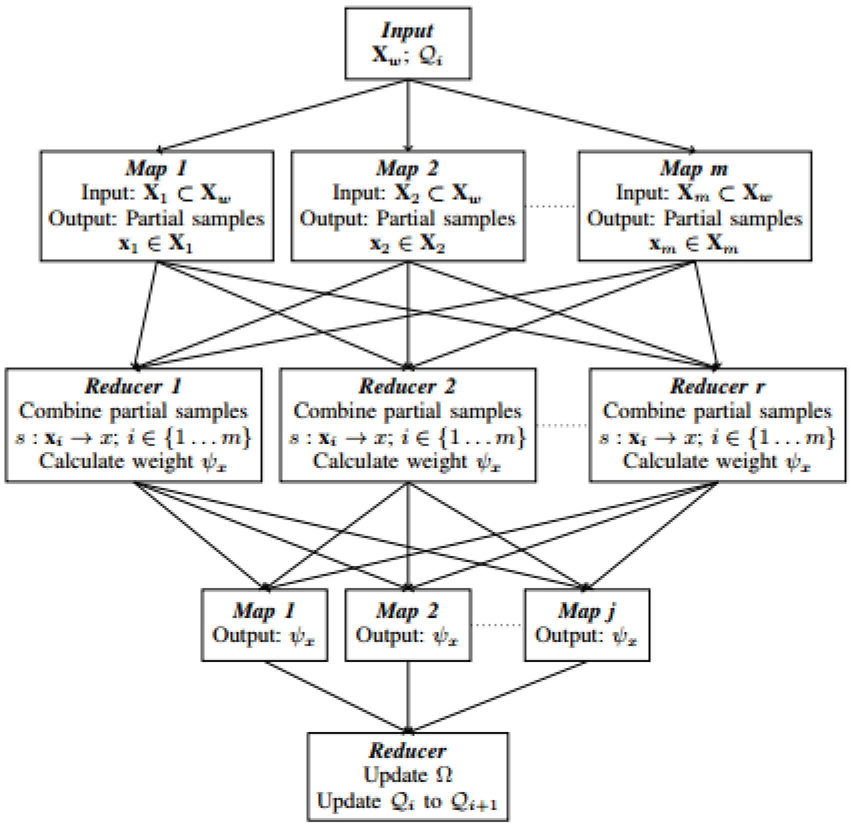
A Bayesian network, Bayes network, belief network, Bayes(ian) model or probabilistic directed acyclic graphical model is a probabilistic graphical model (a type of statistical model) that represents a set of random variables and their conditional dependencies via a directed acyclic graph (DAG). For example, a Bayesian network could represent the probabilistic relationships between diseases and symptoms. Given symptoms, the network can be used to compute the probabilities of the presence of various diseases.

Formally, Bayesian networks are DAGs whose nodes represent random variables in the Bayesian sense: they may be observable quantities, latent variables, unknown parameters or hypotheses. Edges represent conditional dependencies; nodes that are not connected represent variables that are conditionally independent of each other. Each node is associated with a probability function that takes, as input, a particular set of values for the node's parent variables, and gives (as output) the probability (or probability distribution, if applicable) of the variable represented by the node. For example, if m parent nodes represent m Boolean variables then the probability function could be represented by a table of 2^m entries, one entry for each of the 2^m possible combinations of its parents being true or false. Similar ideas may be applied to undirected, and possibly cyclic, graphs; such are called Markov networks.

Efficient algorithms exist that perform inference and learning in Bayesian networks. Bayesian networks that model sequences of variables (e.g. speech signals or protein sequences) are called dynamic Bayesian networks. Generalizations of Bayesian networks that can represent and solve decision problems under uncertainty are called influence diagrams.

Adaptive Importance Sampling Parallel Version Using Map Reduce:

Performing inference on a data model is exponential in tree width complexity for time and space. As model size increases, inference becomes intractable. So, have come up with an approximate inference technique which is adaptive and implemented in parallel way using map reduce. Have implement Adaptive Importance Sampling using Likelihood Weighting proposal distribution and sampling.



Input
$\mathbf{X}_w$; Q_i
Map 1
Input: $\mathbf{X}_1 \subset \mathbf{X}_w$
Output: Partial samples
$\mathbf{x}_1 \in \mathbf{X}_1$
Map 2
Input: $\mathbf{X}_2 \subset \mathbf{X}_w$
Output: Partial samples
$\mathbf{x}_2 \in \mathbf{X}_2$
Map m
Input: $\mathbf{X}_m \subset \mathbf{X}_w$
Output: Partial samples
$\mathbf{x}_m \in \mathbf{X}_m$
Reducer 1
Combine partial samples
$s : \mathbf{x}_i \rightarrow x; i \in \{1 \ldots m\}$
Calculate weight ψ_x
Reducer 2
Combine partial samples
$s : \mathbf{x}_i \rightarrow x; i \in \{1 \ldots m\}$
Calculate weight ψ_x
Reducer r
Combine partial samples
$s : \mathbf{x}_i \rightarrow x; i \in \{1 \ldots m\}$
Calculate weight ψ_x
Map 1
Output: ψ_x
Map 2
Output: ψ_x
Map j
Output: ψ_x
Reducer
Update Ω
Update Q_i to Q_{i+1}

Hadoop Implementation:

We designed several map reduce jobs using java to implement our algorithm and model generation.

This implementation was done using eclipse and vmware workstation.

Servlets were also implemented with hadoop map reduce logic.

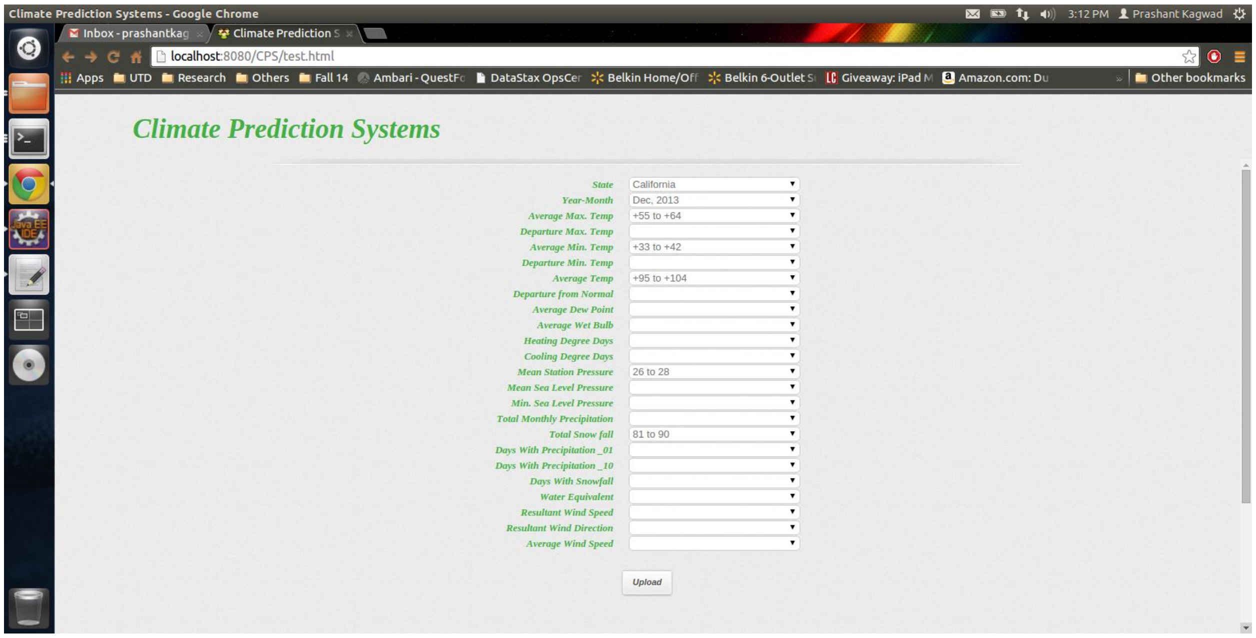
We also tried to implement our structural prediction algorithm using Apache Spark and found out various discrepancies with Java implementation of Spark.

GUI IMPLEMENTATION:

On the GUI, User is given a choice to set the ranges of the climate attributes. (Setting evidence)

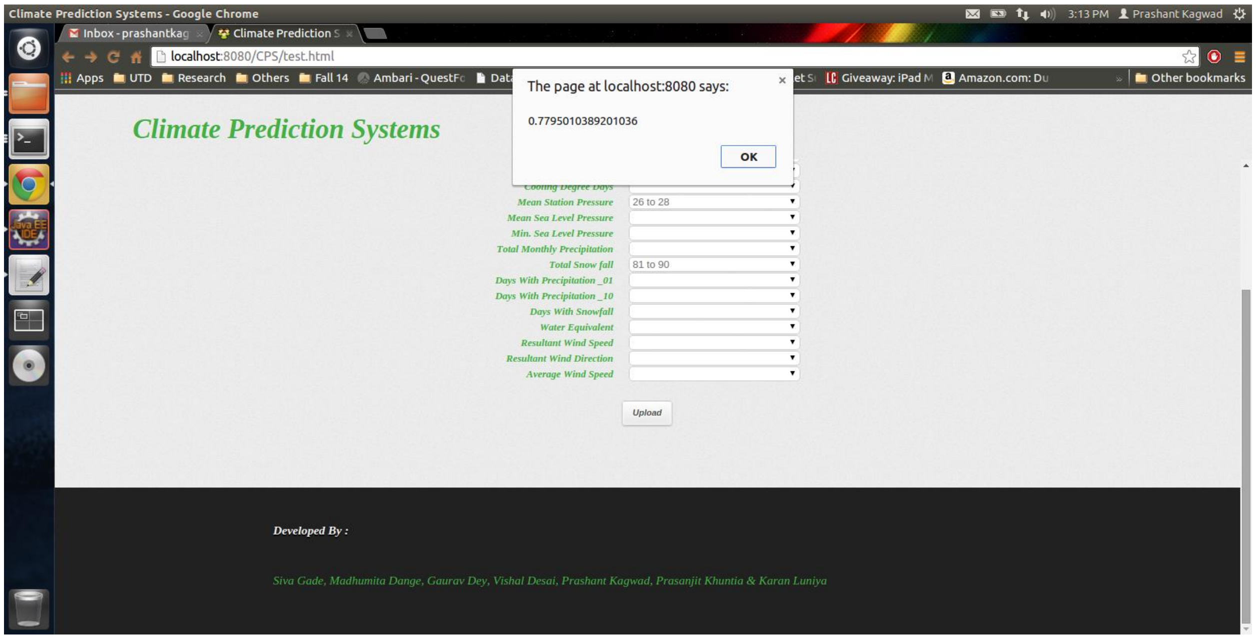
The evidence variables and values chosen is sent as input to the server code which invokes Hadoop map reduce jobs to perform inference on the model using the provided evidence.

Result (POE) is returned to server.


Climate Prediction Systems - Google Chrome
localhost:8080/CPS/test.html
Upload

Climate Prediction Systems

State	California
Year-Month	Dec, 2013
Average Max. Temp	+55 to +64
Departure Max. Temp	
Average Min. Temp	+33 to +42
Departure Min. Temp	
Average Temp	+95 to +104
Departure from Normal	
Average Dew Point	
Average Wet Bulb	
Heating Degree Days	
Cooling Degree Days	
Mean Station Pressure	26 to 28
Mean Sea Level Pressure	
Min. Sea Level Pressure	
Total Monthly Precipitation	
Total Snow fall	81 to 90
Days With Precipitation _01	
Days With Precipitation _10	
Days With Snowfall	
Water Equivalent	
Resultant Wind Speed	
Resultant Wind Direction	
Average Wind Speed	



Climate Prediction Systems - Google Chrome

localhost:8080/CPS/test.html

Climate Prediction Systems

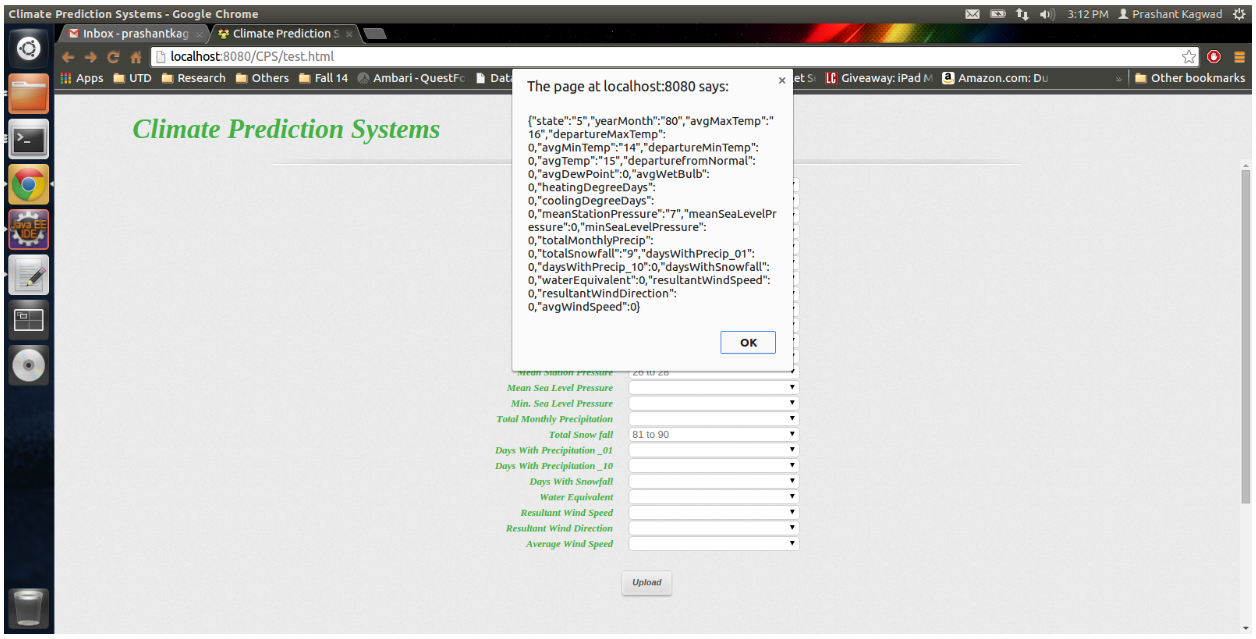
The page at localhost:8080 says:

```
0.7795010389201036
```

Mean Station Pressure	26 to 28
Mean Sea Level Pressure	
Min. Sea Level Pressure	
Total Monthly Precipitation	
Total Snow fall	81 to 90
Days With Precipitation _01	
Days With Precipitation _10	
Days With Snowfall	
Water Equivalent	
Resultant Wind Speed	
Resultant Wind Direction	
Average Wind Speed	

Developed By :

Siva Gade, Madhumita Dange, Gaurav Dey, Vishal Desai, Prashant Kagwad, Prasanjit Khuntia & Karan Luniya



Climate Prediction Systems - Google Chrome

localhost:8080/CPS/test.html

Climate Prediction Systems

The page at localhost:8080 says:

```
{"state":"5","yearMonth":"80","avgMaxTemp":"
16","departureMaxTemp":
0,"avgMinTemp":"14","departureMinTemp":
0,"avgTemp":"15","departurefromNormal":
0,"avgDewPoint":0,"avgWetBulb":
0,"heatingDegreeDays":
0,"coolingDegreeDays":
0,"meanStationPressure":"7","meanSeaLevelPr
essure":0,"minSeaLevelPressure":
0,"totalMonthlyPrecip":
0,"totalSnowfall":"9","daysWithPrecip_01":
0,"daysWithPrecip_10":0,"daysWithSnowfall":
0,"waterEquivalent":0,"resultantWindSpeed":
0,"resultantWindDirection":
0,"avgWindSpeed":0}
```

Mean Station Pressure	26 to 28
Mean Sea Level Pressure	
Min. Sea Level Pressure	
Total Monthly Precipitation	
Total Snow fall	81 to 90
Days With Precipitation _01	
Days With Precipitation _10	
Days With Snowfall	
Water Equivalent	
Resultant Wind Speed	
Resultant Wind Direction	
Average Wind Speed	

CHALLENGES FACED:

Spark implementation for Structural Learning Algorithm using Java.

Servlets implementation over Hadoop.

Efficiency of Chow-Liu Algorithm for 24 variables.

Unstructured data for the pre-processing.

References:

- Cutset Networks: A Simple, Tractable and Scalable Approach for Improving the Accuracy of Chow-Liu Trees - http://www.hlt.utdallas.edu/~vgogate/papers/ecml14-a.pdf
- NCDC – National Climatic Data Center - http://www.ncdc.noaa.gov/climate-information
- ieeebigdata2014.pdf (Distributed Adaptive Importance Sampling on Graphical Models Using MapReduce)
- ssci2014.pdf (MapReduce Guided Approximate Inference Over Graphical Models)