**ABSTRACT**

India Meteorological Department has implemented state level medium range rainfall forecast system applying multi model ensemble technique, making use of model outputs of state-of-the-art global models from the five leading global NWP centers. The pre-assigned grid point weights on the basis of anomaly correlation coefficients (CC) between the observed values and forecast values are determined for each constituent model utilizing two season datasets and the multi model ensemble forecasts are generated at the same resolution on a real-time basis. The ensemble forecast fields are then used to prepare forecasts for each state, taking the average value of all grid points falling in a particular district. In this paper, we describe the development strategy of the technique and performance skill of the system during 15 years of rain fall at different states in india. The study demonstrates the potential of the system for predicting future rainfall forecasts for upcoming years and scale over Indian region. District wise performance of the ensemble rainfall forecast reveals that the technique, in general, is capable of providing reasonably good forecast skill over most states of the country, particularly over the states where the monsoon systems are more dominant.

Rain fall forecasting is one of the most important analysis and prediction method which is required for predicting upcoming rainfall in coming years based on previous dataset. In this project past 10 years dataset for different states and districts are taken as input and weather forecast is predicted. In present studies rainfall prediction was based on NWP numerical weather prediction methods are used which has given better results and played important role. Even though this methods are used in present systems there are many areas accuracy can be increased like dealing with indian monsoon. This is because large variation of data at different times and limitation of NWP models.

**1. INTRODUCTION**

There has been long demand from the user community for district level quantitative weather forecasts in short to medium range time scale. The quantitative rainfall forecast for smaller spatial distribution such as district level over highly complex inhomogeneous region like India is a very challenging task. For the generation of district level quantitative rainfall forecasts, one has to depend on the forecasts from dynamical Numerical Weather Prediction (NWP) models. During the last two decades, NWP methods have acquired greater skills and are playing an increasingly important role in the operational weather forecasting. But rainfall prediction skill of NWP models is still not adequate to satisfactorily address detailed aspects of Indian summer monsoon. This is because of large spatial and temporal variability of rainfall and some inherent limitations of NWP models. There are various factors like topography, prevailing synoptic situation and its interaction with mesoscale systems, lack of observations, etc., are some of the key factors which pose difficulties for numerical weather prediction of any region, and so Indian region is not an exception. Considering the need of farming sector, India Meteorological Department (IMD) has upgraded the Agro-Meteorological Advisory Service from agro climate zone to district level.

**1.1 Background**

As a major step, IMD started issuing district level weather forecasts from 1 June 2008 for meteorological parameters such as rainfall, maximum and minimum temperature, relative humidity, surface wind and cloud octa up to 5 days in quantitative terms (Roy Bhowmik et al 2009). These forecasts are generated through multi-model ensemble (MME) system making use of model outputs of state-ofthe-art three global models from the leading global NWP centres. These forecasts are made available on the national website of IMD (www.imd.gov.in). During summer monsoon 2009, the number of ensemble members is increased from three to five. In the present study, we describe the development strategy of the MME technique, used for high resolution rainfall forecasts over Indian region and demonstrate the prediction skill of the technique during summer monsoon 2009. In our previous study (Roy Bhowmik and Durai 2008, 2010), performance skill of MME at 50 km horizontal resolution for district level short range rainfall forecasts during summer monsoon 2007 was demonstrated from the use of four coarser grid models namely (i) IMD limited area model at 75 km horizontal resolution, (ii) IMD MM5 at 45 km horizontal resolution, (iii) National Centre for Medium Range Weather Forecasting (NCMRWF) MM5 at 30 km resolution, and (iv) NCMRWF T-80 (grid space ∼156 over the tropics). At 50 km resolution, MME could cover only 250 districts. The encouraging results of the study motivated authors for further research to increase the forecast period and model resolution using improved rainfall outputs of state-of-the-art high resolution global models from leading NWP centres to meet the operational requirement of farming community.

The strategy for multimodel super ensemble (Krishnamurti et al 1999, 2000), involves two phases. In the first phase, known as training period, utilizes the direct model outputs and the corresponding observed fields to derive the statistics. The weight for each model is generated from the least square minimization of the difference between the analysis (observed field) and model output utilizing a training period of 120 days. In the work of Krishnamurti et al (1999, 2000), daily rainfall analysis generated on the basis of Tropical Rainfall Measurement Mission (TRMM) and Special Sensor Microwave Imager (SSMI) dataset are used as observed fields. These data are also used by them for physical initialization of Florida State University (FSU) Global model. The second phase, called the forecast phase, utilizes the multimodel forecasts and aforementioned weights to obtain the final ensemble forecast. In the present study, weight for each constituent model at each grid point is derived on the basis of anomaly correlation coefficients (CC) between the observed values (analysis fields) and forecast values based on the training period datasets. The gridded data with the use of high dense land rain gauge observations has the capability to capture large scale as well as mesoscale features of monsoon precipitation systems, and is superior to other rainfall products like, Climate Prediction Centre (CPC), for computation of anomaly CC, observed climatology (on the basis of gridded observed rainfall fields of 15 years) is used. Five NWP models considered for this development work are: (i) National Centre for Medium Range Weather Forecasting (NCMRWF T-254), (ii) European Centre For Medium Range Weather Forecasting (ECMWF T-799), (iii) Japan Meteorological Agency (JMA T-959), (iv) United Kingdom Meteorological Office (UKMO), and (v) National Centre for Environmental Prediction Global Forecast\System (NCEP GFS T-382). Under a joint collaborative research project, IMD has been receiving global model outputs (in the GRIB format) of ECMWF and JMA. The outputs (GRIB) of NCEP GFS are available freely from the Internet. The model outputs of these models are post-processed using GRIB decoder. IMD receives NCMRWF T-254 and UKMO model outputs online from NCMRWF, Noida. As the model outputs available are at different resolutions, in the first step, model outputs of the constituent models are interpolated at the uniform grid resolution of 0.25◦ ×0.25◦ lat./long. for the domain covering from lat. 0◦–40◦N and long. 60◦–100◦E. In the second step, the weight for each model at each grid is determined objectively by computing the anomaly correlation coefficient between the predicted rainfall and observed rainfall. High resolution (lat./long.) gridded data (Rajeevan et al 2006; Rajeevan and Bhate 2009) produced operationally at National Climate Data Centre of IMD Pune are used for the development and validation of forecasts. The gridded data with the use of high dense land rain gauge observations has the capability to capture large scale as well as mesoscale features of monsoon precipitation systems, and is superior to other rainfall products like, Climate Prediction Centre (CPC), Merged Analysis of Precipitation (CMAP), Global Precipitation Climate Product (GPCP) data to bring out characteristic features of Indian monsoon (Roy Bhowmik and Das 2007).

**1.2 Purpose**

Indian meteorological department provides forecasting data required for project. In this project we are planning to work on long term predictions of rainfall.The main motive of the project is to predict the amount of rainfall in a particular division or state well in advance. We predict the amount of rainfall using past data.There has been long demand from the user community for state level quantitative weather forecasts in short to medium range time scale. The quantitative rainfall forecast for smaller spatial distribution such as state level over highly complex inhomogeneous region like India is a very challenging task.

**1.3 Existing system**

There has been long demand from the user community for state level quantitative weather forecasts in short to medium range time scale. The quantitative rainfall forecast for smaller spatial distribution such as state level over highly complex inhomogeneous region like India is a very challenging task. For the generation of district level quantitative rainfall forecasts, one has to depend on the forecasts from dynamical Numerical Weather Prediction (NWP) models. During the last two decades, NWP methods have acquired greater skills and are playing an increasingly important role in the operational weather forecasting. But rainfall prediction skill of NWP models is still not adequate to satisfactorily address detailed aspects of Indian summer monsoon.

In our previous study (Roy Bhowmik and Durai 2008, 2010), performance skill of MME at 50 km horizontal resolution for district level short range rainfall forecasts during summer monsoon 2007 was demonstrated from the use of four coarser grid models namely (i) IMD limited area model at 75 km horizontal resolution, (ii) IMD MM5 at 45 km horizontal resolution, (iii) National Centre for Medium Range Weather Forecasting (NCMRWF) MM5 at 30 km resolution, and (iv) NCMRWF T-80 (grid space ∼156 over the tropics). At 50 km resolution, MME could cover only 250 districts

**Disadvantages :**

1. Existing methods are covering for only few districts which are not using machine learning techniques and there was no prediction for rainfall .
2. In preset system data set was limited for few districts which are restricted for data analysis.
3. The primary problem with Numerical Weather Prediction (NWP) models is,it takes long time to produce its results.
4. Existing methods are covering for only few districts, which in turn are restricting data analysis and data prediction process.
5. There are various factors like topography, its interaction with mesoscale systems, lack of observations, etc., are some of the key factors which pose difficulties.

**1.4 Proposed system**

Proposed system is designed for long term prediction by applying machine learning models like :

1. **Linear regression** - In statistics, linear regression is a linear approach to modeling the relationship between a scalar response and one or more explanatory variables.
2. **Artificial Neural Network** - ANNs are considered nonlinear statistical data modeling tools where the complex relationships between inputs and outputs are modeled or patterns are found.
3. **Support Vector Machine** - Support Vector Machines are a type of supervised machine learning algorithm that provides analysis of data for classification and regression analysis.

In the present study, we describe the development strategy of the MME technique, used for high resolution rainfall forecasts over Indian region and demonstrate the prediction skill.

**Advantages :**

1)Ability to work with incomplete knowledge

2)Stability :- A small change to the data does not greatly affect the hyperplane and hence the SVM.

1. ANN takes data samples rather than entire data sets to arrive at solutions, which saves both time and money.
2. ANN's are considered fairly simple mathematical models to enhance existing data analysis technologies.

**2. Literature survey**

**Improved weather and seasonal climate forecasts from multimodel super ensemble**

India Meteorological Department has implemented district level medium range rainfall forecast system applying multimodel ensemble technique, making use of model outputs of state-of-the-art global models from the five leading global NWP centres. The pre-assigned grid point weights on the basis of anomaly correlation coefficients (CC) between the observed values and forecast values are determined for each constituent model at the resolution of 0.25° ×0.25° utilizing two season datasets (1 June–30 September, 2007 and 2008) and the multimodel ensemble forecasts (day-1 to day-5 forecasts) are generated at the same resolution on a real-time basis. The ensemble forecast fields are then used to prepare forecasts for each district, taking the average value of all grid points falling in a particular district. In this paper, we describe the development strategy of the technique and performance skill of the system during summer monsoon 2009. The study demonstrates the potential of the system for improving rainfall forecasts at five days time scale over Indian region. Districtwise performance of the ensemble rainfall forecast reveals that the technique, in general, is capable of providing reasonably good forecast skill over most states of the country, particularly over the states where the monsoon systems are more dominant.

**Improving tropical precipitation forecasts from a multi analysis super ensemble**

This paper utilizes forecasts from a multianalysis system to construct a superensemble of precipitation forecasts. This method partitions the computations into two time lines. The first of those is a control (or a training) period and the second is a forecast period. The multianalysis is derived from a physical initialization–based data assimilation of “observed rainfall rates.” The different members of the reanalysis are produced by using different rain-rate algorithms for physical initialization. The basic rain-rate datasets are derived from satellites’ microwave radiometers, including those from the Tropical Rainfall Measuring Mission (TRMM) satellites and the Special Sensor Microwave Imager (SSM/I) data from three current U.S. Air Force Defense Meteorological Satellite Program (DMSP) satellites. During the training period, 155 experiments were conducted to find the relationship between forecasts from the multianalysis dataset and the best “observed” estimates of daily rainfall totals. This relationship is based on multiple regression and defined by statistical weights (which vary in space.) The forecast phase utilizes the multianalysis forecasts and the statistics from the training period to produce superensemble forecasts of daily rainfall totals. The results for day 1, day 2, and day 3 forecasts are compared to various conventional forecasts with a global model. The superensemble day 3 forecasts of precipitation clearly have the highest skill in such comparisons.

**Experimental realtime multi-model ensemble (MME) prediction of rainfall during monsoon 2008: Large scale medium range aspects**

Realistic simulation/prediction of the Asian summer monsoon rainfall on various space–time scales is a challenging scientific task. Compared to mid-latitudes, a proportional skill improvement in the prediction of monsoon rainfall in the medium range has not happened in recent years. Global models and data assimilation techniques are being improved for monsoon/tropics. However, multi-model ensemble (MME) forecasting is gaining popularity, as it has the potential to provide more information for practical forecasting in terms of making a consensus forecast and handling model uncertainties. As major centers are exchanging model output in near real-time, MME is a viable inexpensive way of enhancing the forecasting skill and information content. During monsoon 2008, on an experimental basis, an MME forecasting of large-scale monsoon precipitation in the medium range was carried out in real-time at National Centre for Medium Range Weather Forecasting (NCMRWF), India. Simple ensemble mean (EMN) giving equal weight to member models, bias-corrected ensemble mean (BCEMn) and MME forecast, where different weights are given to member models, are the products of the algorithm tested here. In general, the aforementioned products from the multi-model ensemble forecast system have a higher skill than individual model forecasts. The skill score for the Indian domain and other sub-regions indicates that the BCEMn produces the best result, compared to EMN and MME. Giving weights to different models to obtain an MME product helps to improve individual member models only marginally. It is noted that for higher rainfall values, the skill of the global model rainfall forecast decreases rapidly beyond day-3, and hence for day-4 and day-5, the MME products could not bring much improvement over member models. However, up to day-3, the MME products were always better than individual member models.

**High resolution daily gridded rainfall data for Indian Region: Analysis of break and active monsoon spells**

In this communication, we discuss the development of a very high resolution (0.5° x 0.5°) daily rainfall data-set for mesoscale meteorological studies over the Indian region. The dataset was developed using quality-controlled rainfall data from more than 3000 rain gauge stations over India. The analysis consists of daily rainfall data for all the seasons for the period 1971-2005. A well-tested interpolation method (Shepard's method) was used to interpolate the station data into regular grids of 0.5° x 0.5° lat. x long. After proper validation, it has been found that the present dataset is better compared to other available datasets. A few case studies have been shown to demonstrate the utility of the dataset for different mesoscale meteorological analyses. However, since the data density is not kept uniform, there is a possibility of temporal inhomogeneity and therefore, the present dataset cannot be used for trend analysis. The dataset is freely available from the India Meteorological Department, Pune.

**Rainfall analysis for Indian monsoon region using the merged rain gauge observations and satellite estimates**

Objective analysis of daily rainfall at the resolution of 1° grid for the Indian monsoon region has been carried out merging dense land rainfall observations and INSAT derived precipitation estimates. This daily analysis, being based on high dense rain gauge observations was found to be very realistic and able to reproduce detailed features of Indian summer monsoon. The inter-comparison with the observations suggests that the new analysis could distinctly capture characteristic features of the summer monsoon such as north-south oriented belt of heavy rainfall along the Western Ghats with sharp gradient of rainfall between the west coast heavy rain region and the rain shadow region to the east, pockets of heavy rainfall along the location of monsoon trough/low, over the east central parts of the country, over north-east India, along the foothills of Himalayas and over the north Bay of Bengal. When this product was used to assess the quality of other available standard climate products (CMAP and ECMWF reanalysis) at the gird resolution of 2.5°, it was found that the orographic heavy rainfall along Western Ghats of India was poorly identified by them. However, the GPCC analysis (gauge only) at the resolution of 1° grid closely discerns the new analysis. This suggests that there is a need for a higher resolution analysis with adequate rain gauge observations to retain important aspects of the summer monsoon over India. The case studies illustrated show that the daily analysis is able to capture large-scale as well as mesoscale features of monsoon precipitation systems. This study with data of two seasons (2001 and 2003) has shown sufficiently promising results for operational application, particularly for the validation of NWP models.

1. **SYSTEM STUDY**

**3.1 FEASIABILITY STUDY**

Preliminary investigation examines project feasibility; the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All systems are feasible if they are given unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

* Technical Feasibility
* Operation Feasibility
* Economical Feasibility

**3.1.1 Technical Feasibility**

The technical issue usually raised during the feasibility stage of the investigation includes the following:

* Does the necessary technology exist to do what is suggested?
* Do the proposed equipment have the technical capacity to hold the data required to use the new system?
* Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
* Can the system be upgraded if developed?

**3.1.2 Operation Feasibility**

**User Friendly**

Customer will use the forms for their various transactions i.e. for adding new routes, viewing the routes details. Also the Customer wants the reports to view the various transactions based on the constraints. These forms and reports are generated as user-friendly to the Client.

**Reliability**

The package wills pick-up current transactions on line. Regarding the old transactions, User will enter them in to the system.

**Security**

The web server and database server should be protected from hacking, virus etc

**Portability**

The application will be developed using standard open source software (Except Oracle) like Java, tomcat web server, Internet Explorer Browser etc these software will work both on Windows and Linux o/s. Hence portability problems will not arise.

**Availability**

This software will be available always.

**Maintainability**

The system uses the 2-tier architecture. The 1st tier is the GUI, which is said to be front-end and the 2nd tier is the database, which uses My-Sql, which is the back-end. The front-end can be run on different systems (clients). The database will be running at the server. Users access these forms by using the user-ids and the passwords.

**3.1.3 Economic feasibility**

The computerized system takes care of the present existing system’s data flow and procedures completely and should generate all the reports of the manual system besides a host of other management reports. It should be built as a web based application with separate web server and database server. This is required as the activities are spread throughout the organization customer wants a centralized database. Further some of the linked transactions take place in different locations. Open source software like TOMCAT, JAVA, Mysql and Linux is used to minimize the cost for the Customer. All systems are feasible if they are given unlimited resources and infinite time. The interpretation of the results from the automated accessibility testing tools requires experience in accessibility techniques with an understanding of technical and usability issues.

**3.2 Functional Requirements**

The functional requirement refers to the system needs in an exceedingly computer code engineering method. The key goal of determinant “functional requirements” in an exceedingly product style and implementation is to capture the desired behavior of a software package in terms of practicality and also the technology implementation of the business processes.

**3.3 Non-Functional Requirements**

All the other requirements which do not form a part of the above specification are categorized as Non-Functional needs. A system perhaps needed to gift the user with a show of the quantity of records during info. If the quantity must be updated in real time, the system architects should make sure that the system is capable of change the displayed record count at intervals associate tolerably short interval of the quantity of records dynamic. Comfortable network information measure may additionally be a non-functional requirement of a system.

The following are the features:

* Accessibility
* Availability
* Maintenance
* Computer Performance
* Portability
* Reliability
* Response Time
* Robustness
* Scalability
* Security

**Accessibility**

Accessibility testing is a subset of usability testing where in the users under consideration are people with all abilities and disabilities. The significance of this testing is to verify both usability and accessibility. Accessibility testing is a subset of usability testing where in the users under consideration are people with all abilities and disabilities. The significance of this testing is to verify both usability and accessibility. The above said automated accessibility testing tools are very good at identifying pages and lines of code that need to be manually checked for accessibility. The interpretation of the results from the automated accessibility testing tools requires experience in accessibility techniques with an understanding of technical and usability issues.

**Availability**

Availability indicates when a system is operational as well as how reliable it is during operational period, What are the hours that a given system will be available? What days will the system be operational? Not all systems operate on a 24/7 basis. Some internal facing systems may only be needed when there are people in place to operate them. During a system's hours of operation, what reliability (excluding planned outages) is needed? Reliability is usually measured as a percentage. The higher the number, the greater the cost. Many people ask for 99% reliability but do you actually need it? [37signals](http://37signals.com/svn/) has a [great post](http://37signals.com/svn/archives2/dont_scale_99999_uptime_is_for_walmart.php) explaining this concept. The question to ask your clients is, "What level of reliability is justifiable from a business perspective?" Will the world end if your web site goes down unexpectedly for an hour? The answer will be different for different systems and companies.

**Maintenance**

The technical meaning of maintenance involves functional checks, servicing, repairing or replacing of necessary devices, equipment, machinery, building infrastructure, and supporting utilities in industrial, business, governmental, and residential installations. Over time, this has come to include multiple wordings that describe various cost-effective practices to keep equipment operational; these activities take place either before or [after a failure](https://en.wikipedia.org/wiki/Maintenance_(technical)" \l "breakdown_maintenance).

The marine and air transportation, offshore structures, industrial plant and [facility management](https://en.wikipedia.org/wiki/Facility_management" \o "Facility management) industries depend on maintenance, repair and overhaul (MRO) including scheduled or preventive [paint](https://en.wikipedia.org/wiki/Paint" \o "Paint) maintenance programs to maintain and restore [coatings](https://en.wikipedia.org/wiki/Coatings" \o "Coatings) applied to steel in environments subject to attack from erosion, corrosion and environmental pollution. [Architectural conservation](https://en.wikipedia.org/wiki/Architectural_conservation" \o "Architectural conservation) employs MRO to preserve, rehabilitate, restore, or reconstruct historical structures with stone, brick, glass, metal, and wood which match the original constituent materials where possible, or with suitable polymer technologies when not.

**Computer Performance**

In [computing](https://en.wikipedia.org/wiki/Computing" \o "Computing), computer performance is the amount of useful work accomplished by a computer system. Outside of specific contexts, computer performance is estimated in terms of accuracy, efficiency and speed of executing more of the following factors might be involved:

• Short [response time](https://en.wikipedia.org/wiki/Response_time_(technology)" \o "Response time (technology)) for a given piece of work.

• High [throughput](https://en.wikipedia.org/wiki/Throughput" \o "Throughput) (rate of processing work).

• Low utilization of [computing resource](https://en.wikipedia.org/wiki/Computing_resource" \o "Computing resource)(s).

• [High availability](https://en.wikipedia.org/wiki/High_availability" \o "High availability) of the computing system or application.

• Fast (or highly compact) [data compression](https://en.wikipedia.org/wiki/Data_compression" \o "Data compression) and decompression.

• High [bandwidth](https://en.wikipedia.org/wiki/Bandwidth_(computing)" \o "Bandwidth (computing)).

• Short [data transmission](https://en.wikipedia.org/wiki/Data_transmission" \o "Data transmission) time.

**Portability**

Portability in [high-level computer programming](https://en.wikipedia.org/wiki/High-level_programming_language" \o "High-level programming language) is the usability of the same [software](https://en.wikipedia.org/wiki/Software" \o "Software) in different environments. The requirement for portability is the generalized [abstraction](https://en.wikipedia.org/wiki/Abstraction_(computer_science)" \o "Abstraction (computer science)) between the application logic and [system interfaces](https://en.wikipedia.org/wiki/Interface_(computer_science)" \o "Interface (computer science)). When software with the same functionality is produced for several [computing platforms](https://en.wikipedia.org/wiki/Computing_platform" \o "Computing platform), portability is the key issue for development cost reduction.When [operating systems](https://en.wikipedia.org/wiki/Operating_system" \o "Operating system) of the same family are installed on two computers with [processors](https://en.wikipedia.org/wiki/CPU" \o "CPU) with similar [instruction sets](https://en.wikipedia.org/wiki/Instruction_set" \o "Instruction set) it is often possible to transfer the files implementing program files between them.

Porting is no more than transferring specified directories and their contents. Software installed on portable [mass storage](https://en.wikipedia.org/wiki/Mass_storage" \o "Mass storage) devices such as [USB sticks](https://en.wikipedia.org/wiki/USB_stick" \o "USB stick) can be used on any compatible computer on simply plugging the storage device in, and stores all configuration information on the removable device. Hardware- and software-specific information is often stored in [configuration files](https://en.wikipedia.org/wiki/Configuration_file" \o "Configuration file) in specified locations (e.g. the [registry](https://en.wikipedia.org/wiki/Windows_registry" \o "Windows registry) on machines running [Microsoft Windows](https://en.wikipedia.org/wiki/Microsoft_Windows" \o "Microsoft Windows)). Software which is not portable in this sense will have to be transferred with modifications to support the environment on the destination machine.

**Reliability**

Reliability in [statistics](https://en.wikipedia.org/wiki/Statistics" \o "Statistics) and psychometrics is the overall consistency of a measure. A measure is said to have a high reliability if it produces similar results under consistent conditions. "It is the characteristic of a set of test scores that relates to the amount of random error from the measurement process that might be embedded in the scores. Scores that are highly reliable are accurate, reproducible, and consistent from one testing occasion to another. That is, if the testing process were repeated with a group of test takers, essentially the same results would be obtained. Various kinds of reliability coefficients, with values ranging between 0.00 (much error) and 1.00 (no error), are usually used to indicate the amount of error in the scores." For example, measurements of people's height and weight are often extremely reliable.

**Response Time**

Response time is the total amount of time it takes to respond to a request for service. That service can be anything from a memory fetch, to a disk IO, to a complex database query, or loading a full web page. Ignoring transmission time for a moment, the response time is the sum of the service time and wait time. The service time is the time it takes to do the work you requested. For a given request the service time varies little as the workload increases – to do X amount of work it always takes X amount of time. The wait time is how long the request had to wait in a [queue](https://en.wikipedia.org/wiki/Queue_(abstract_data_type)" \o "Queue (abstract data type)) before being serviced and it varies from zero, when no waiting is required, to a large multiple of the service time, as many requests are already in the queue and have to be serviced first.

With basic queueing theory math you can calculate how the average wait time increases as the device providing the service goes from 0-100% busy. As the device becomes busier, the average wait time increases in a [non-linear](https://en.wikipedia.org/wiki/Nonlinear_system" \o "Nonlinear system) fashion. The busier the device is, the more dramatic the response time increases will seem as you approach 100% busy; all of that increase is caused by increases in wait time, which is the result of all the requests waiting in queue that have to run first.

**Robustness**

Robustness of a [biological system](https://en.wikipedia.org/wiki/Biological_system" \o "Biological system) (also called biological or genetic robustness) is the persistence of a certain characteristic or trait in a system under perturbations or conditions of uncertainty. Robustness in development is known as [canalization](https://en.wikipedia.org/wiki/Canalisation_(genetics)" \o "Canalisation (genetics)). According to the kind of perturbation involved, robustness can be classified as [mutational](https://en.wikipedia.org/wiki/Mutation" \o "Mutation), [environmental](https://en.wikipedia.org/wiki/Environment_(biophysical)" \o "Environment (biophysical)), [recombinational](https://en.wikipedia.org/wiki/Genetic_recombination" \o "Genetic recombination), or [behavioral](https://en.wikipedia.org/wiki/Behavior" \o "Behavior) robustness etc. Robustness is achieved through the combination of many [genetic](https://en.wikipedia.org/wiki/Genetics" \o "Genetics) and [molecular mechanisms](https://en.wikipedia.org/wiki/Molecular_biology" \o "Molecular biology) and can [evolve](https://en.wikipedia.org/wiki/Evolution" \o "Evolution) by either direct or indirect [selection](https://en.wikipedia.org/wiki/Natural_selection" \o "Natural selection). Several [model systems](https://en.wikipedia.org/wiki/Model_organism" \o "Model organism) have been developed to experimentally study robustness and its evolutionary consequences.

**Scalability**

Scalability is the property of a system to handle a growing amount of work by adding resources to the system. In an [economic](https://en.wikipedia.org/wiki/Economics" \o "Economics) context, a scalable business model implies that a company can increase sales given increased resources. For example, a package delivery system is scalable because more packages can be delivered by adding more delivery vehicles. However, if all packages had to first pass through a single warehouse for sorting, the system would not be scalable, because one warehouse can handle only a limited number of packages.

In computing, scalability is a characteristic of computers, networks, [algorithms](https://en.wikipedia.org/wiki/Algorithm" \o "Algorithm), [networking protocols](https://en.wikipedia.org/wiki/Protocol_(computing)" \o "Protocol (computing)), [programs](https://en.wikipedia.org/wiki/Computer_program" \o "Computer program) and applications. An example is a search engine, which must support increasing numbers of users, and the number of topics it indexes. In mathematics, scalability mostly refers to closure under [scalar multiplication](https://en.wikipedia.org/wiki/Scalar_multiplication" \o "Scalar multiplication).

**Security**

Security is freedom from, or resilience against, [potential](https://en.wikipedia.org/wiki/Potential" \o "Potential) harm (or other unwanted [coercive](https://en.wikipedia.org/wiki/Coercion" \o "Coercion) change) caused by others. Beneficiaries of security may be of persons and social groups, objects and institutions, ecosystems or any other entity or phenomenon vulnerable to unwanted change. Security mostly refers to protection from hostile forces, but it has a wide range of other senses: for example, as the absence of harm (e.g. [freedom from want](https://en.wikipedia.org/wiki/Freedom_from_want" \o "Freedom from want)); as the presence of an essential good (e.g. [food security](https://en.wikipedia.org/wiki/Food_security" \o "Food security)); as [resilience](https://en.wikipedia.org/wiki/Resilience_(organizational)" \o "Resilience (organizational)) against potential damage or harm (e.g. secure foundations); as secrecy (e.g. a [secure telephone line](https://en.wikipedia.org/wiki/Telephone_tapping" \o "Telephone tapping)); as containment (e.g. a secure room or [cell](https://en.wikipedia.org/wiki/Prison_cell" \o "Prison cell)); and as a state of mind (e.g. [emotional security](https://en.wikipedia.org/wiki/Emotional_security" \o "Emotional security)). The term is also used to refer to acts and systems whose purpose may be to provide security (e.g. [security forces](https://en.wikipedia.org/wiki/Security_forces" \o "Security forces); [security guard](https://en.wikipedia.org/wiki/Security_guard" \o "Security guard); [cyber security systems](https://en.wikipedia.org/wiki/Computer_security" \o "Computer security); [security cameras;](https://en.wikipedia.org/wiki/Closed-circuit_television" \o "Closed-circuit television) [remote guarding](https://en.wikipedia.org/wiki/Remote_guarding" \o "Remote guarding)).

**3.4 Python**

**3.4.1 Script**

Up to this point, I have concentrated on the interactive programming capability of Python.  This is a very useful capability that allows you to type in a program and to have it executed immediately in an interactive mode

**Scripts are reusable**

Basically, a script is a text file containing the statements that comprise a Python program.  Once you have created the script, you can execute it over and over without having to retype it each time.

**Scripts are editable**

Perhaps, more importantly, you can make different versions of the script by modifying the statements from one file to the next using a text editor.  Then you can execute each of the individual versions. In this way, it is easy to create different programs with a minimum amount of typing. Security is freedom from, or resilience against, [potential](https://en.wikipedia.org/wiki/Potential" \o "Potential) harm (or other unwanted [coercive](https://en.wikipedia.org/wiki/Coercion" \o "Coercion) change) caused by others.

**You will need a text editor**

Just about any text editor will suffice for creating Python script files.You can use Microsoft Notepad, Microsoft WordPad, Microsoft Word, or just about any word processor if you want to.

**Difference between a script and  a program**

**Script**

Scripts are distinct from the core code of the application, which is usually written in a different language, and are often created or at least modified by the end-user. Scripts are often interpreted from source code or bytecode, where as the applications they control are traditionally compiled to native machine code.

**Program**

The program has an executable form that the computer can use directly to execute the instructions. The same program in its human-readable source code form, from which executable programs are derived (e.g., compiled)

**Python**

What is Python? Chances you are asking yourself this. You may have found this book because you want to learn to program but don’t know anything about programming languages. Or you may have heard of programming languages like C, C++, C#, or Java and want to know what Python is and how it compares to “big name” languages. Hopefully I can explain it for you.

**Python concepts**

If your not interested in the hows and whys of Python, feel free to skip to the next chapter. In this chapter I will try to explain to the reader why I think Python is one of the best languages available and why it’s a great one to start programming with.

• Open source general-purpose language.

 • Object Oriented, Procedural, Functional

• Easy to interface with C/ObjC/Java/Fortran

 • Easy-ish to interface with C++ (via SWIG)

• Great interactive environment

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English key words frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

• **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

• **Python is Object - Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

• **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

**History of Python**

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages. Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL). Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

**Python Features**

Python's features include −

• **Easy-to-learn** − Python has few keywords, simple structure, and a clearly   defined syntax. This allows the student to pick up the language quickly.

• **Easy-to-read** − Python code is more clearly defined and visible to the eyes.

• **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.

• **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

• **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

• **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.

• **Databases** − Python provides interfaces to all major commercial databases.

• **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and window systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

• **Scalable** − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

•  It supports functional and structured programming methods as well as OOP.

•  It can be used as a scripting language or can be compiled to byte-code for building large applications.

• It provides very high-level dynamic data types and supports dynamic type checking.

**Dynamic vs Static**

Types Python is a dynamic-typed language. Many other languages are static typed, such as C/C++ and Java. A static typed language requires the programmer to explicitly tell the computer what type of “thing” each data value is. For example, in C if you had a variable that was to contain the price of something, you would have to declare the variable as a “float” type. This tells the compiler that the only data that can be used for that variable must be a floating point number, i.e. a number with a decimal point. If any other data value was assigned to that variable, the compiler would give an error when trying to compile the program.

Python, however, doesn’t require this. You simply give your variables names and assign values to them. The interpreter takes care of keeping track of what kinds of objects your program is using. This also means that you can change the size of the values as you develop the program. Say you have another decimal number (a.k.a. a floating point number) you need in your program. With a static typed language, you have to decide the memory size the variable can take when you first initialize that variable. A double is a floating point value that can handle a much larger number than a normal float (the actual memory sizes depend on the operating environment). If you declare a variable to be a float but later on assign a value that is too big to it, your program will fail; you will have to go back and change that variable to be a double. With Python, it doesn’t matter. You simply give it whatever number you want and Python will take care of manipulating it as needed. It even works for derived values. For example, say you are dividing two numbers. One is a floating point number and one is an integer. Python realizes that it’s more accurate to keep track of decimals so it automatically calculates the result as a floating point number

**Variables**

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory. Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals or characters in these variables.

**Standard Data Types**

The data stored in memory can be of many types. For example, a person's age is stored as a numeric value and his or her address is stored as alphanumeric characters. Python has various standard data types that are used to define the operations possible on them and the storage method for each of them.

**Python has five standard data types −**

• Numbers

• String

• List

• Tuple

• Dictionary

**Python Numbers**

Number data types store numeric values. Number objects are created when you assign a value to them

**Python Strings**

Strings in Python are identified as a contiguous set of characters represented in the quotation marks. Python allows for either pairs of single or double quotes. Subsets of strings can be taken using the slice operator ([ ] and [:] ) with indexes starting at 0 in the beginning of the string and working their way from -1 at the end.

**Python Lists**

Lists are the most versatile of Python's compound data types. A list contains items separated by commas and enclosed within square brackets ([]). To some extent, lists are similar to arrays in C. One difference between them is that all the items belonging to a list can be of different datatype. The values stored in a list can be accessed using the slice operator ([ ] and [:]) with indexes starting at 0 in the beginning of the list and working their way to end -1. The plus (+) sign is the list concatenation operator, and the asterisk (\*) is the repetition operator.

**Python Tuples**

A tuple is another sequence data type that is similar to the list. A tuple consists of a number of values separated by commas. Unlike lists, however, tuples are enclosed within parentheses. The main differences between lists and tuples are: Lists are enclosed in brackets ( [ ] ) and their elements and size can be changed, while

**Python Dictionary**

Python's dictionaries are kind of hash table type. They work like associative arrays or hashes found in Perl and consist of key-value pairs. A dictionary key can be almost any Python type, but are usually numbers or strings. Values, on the other hand, can be any arbitrary Python object. Dictionaries are enclosed by curly braces ({ }) and values can be assigned and accessed using square braces ([]).

**Different modes in python**

Python has two basic modes: normal and interactive. The normal mode is the mode where the scripted and finished .py files are run in the Python interpreter. Interactive mode is a command line shell which gives immediate feedback for each statement, while running previously fed statements in active memory. As new lines are fed into the interpreter, the fed program is evaluated both in part and in whole

**Python libraries**

1. Requests. The most famous http library written by kenneth reitz. It’s a must have for every python developer.

2. Scrapy. If you are involved in web scraping then this is a must have library for you. After using this library you won’t use any other.

3. wxPython. A gui toolkit for python. I have primarily used it in place of tkinter. You will really love it.

4. Pillow. A friendly fork of PIL (Python Imaging Library). It is more user friendly than PIL and is a must have for anyone who works with images.

5. SQLAlchemy. A database library. Many love it and many hate it. The choice is yours.

6. BeautifulSoup. I know it’s slow but this xml and html parsing library is very useful for beginners.

7. Twisted. The most important tool for any network application developer. It has a very beautiful api and is used by a lot of famous python developers.

8. NumPy. How can we leave this very important library ? It provides some advance math functionalities to python.

9. SciPy. When we talk about NumPy then we have to talk about scipy. It is a library of algorithms and mathematical tools for python and has caused many scientists to switch from ruby to python.

10. matplotlib. A numerical plotting library. It is very useful for any data scientist or any data analyzer.

11. Pygame. Which developer does not like to play games and develop them ? This library will help you achieve your goal of 2d game development.

20. IPython. I just can’t stress enough how useful this tool is. It is a python prompt on steroids. It has completion, history, shell capabilities, and a lot more. Make sure that you take a look at it.

**Here’s a brief list of Python OOP ideas:**

• The class statement creates a class object and gives it a name. This creates a new namespace.

• Assignments within the class create class attributes. These attributes are accessed by qualifying the name using dot syntax: ClassName.Attribute.

• Class attributes export the state of an object and its associated behavior. These attributes are shared by all instances of a class.

• Calling a class (just like a function) creates a new instance of the class. This is where the multiple copies part comes in.

• Each instance gets ("inherits") the default class attributes and gets its own namespace. This prevents instance objects from overlapping and confusing the program.

• Using the term self identifies a particular instance, allowing for per-instance attributes. This allows items such as variables to be associated with a particular instance.

**Exceptions**

I’ve talked about exceptions before but now I will talk about them in depth. Essentially, exceptions are events that modify program’s flow, either intentionally or due to errors. They are special events that can occur due to an error, e.g. trying to open a file that doesn’t exist, or when the program reaches a marker, such as the completion of a loop. Exceptions, by definition, don’t occur very often; hence, they are the "exception to the rule" and a special class has been created for them. Exceptions are everywhere in Python. Virtually every module in the standard Python library uses them, and Python itself will raise them in a lot of different circumstances.

**Here are just a few examples:**

• Accessing a non−existent dictionary key will raise a KeyError exception.

• Searching a list for a non−existent value will raise a ValueError exception

• Calling a non−existent method will raise an AttributeError exception.

• Referencing a non−existent variable will raise a NameError exception.

• Mixing datatypes without coercion will raise a TypeError exception.

One use of exceptions is to catch a fault and allow the program to continue working; we have seen this before when we talked about files. This is the most common way to use exceptions. When programming with the Python command line interpreter, you don’t need to worry about catching exceptions. Your program is usually short enough to not be hurt too much if an exception occurs. Plus, having the exception occur at the command line is a quick and easy way to tell if your code logic has a problem.

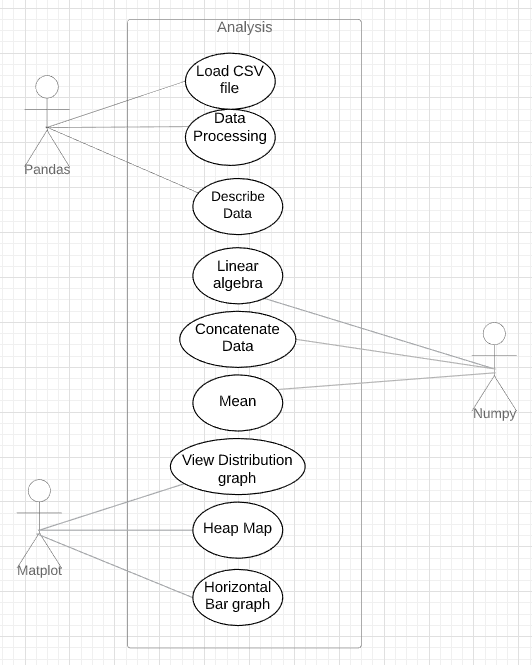
However, if the same error occurred in your real program, it will fail and stop working. Exceptions can be created manually in the code by raising an exception. It operates exactly as a system-caused exceptions, except that the programmer is doing it on purpose. This can be for a number of reasons. One of the benefits of using exceptions is that, by their nature, they don’t put any overhead on the code processing.

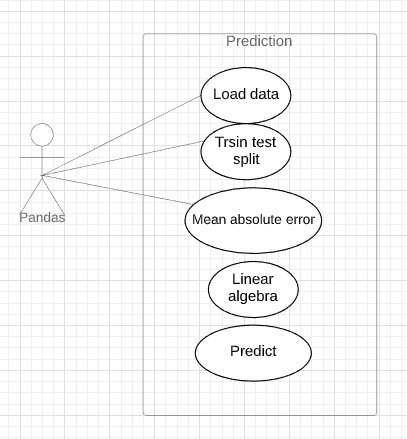
Exceptions can be thought of as a special form of the if/elif statements. You can realistically do the same thing with if blocks as you can with exceptions. However, as already mentioned, exceptions aren’t processed until they occur; if blocks are processed all the time. Proper use of exceptions can help the performance of your program. The more infrequent the error might occur, the better off you are to use exceptions; using if blocks requires Python to always test extra conditions before continuing. Exceptions also make code management easier: if your programming logic is mixed in with error-handling if statements, it can be difficult to read, modify, and debug your program.

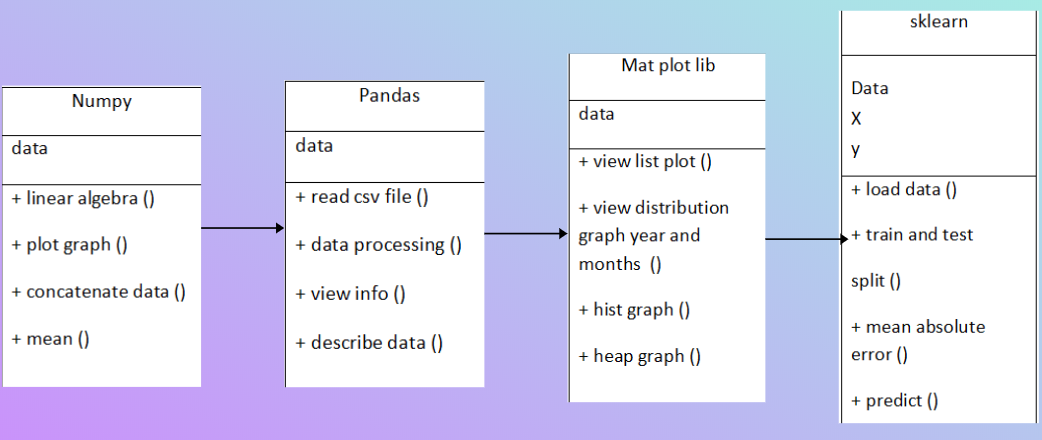
**User-Defined Exceptions**

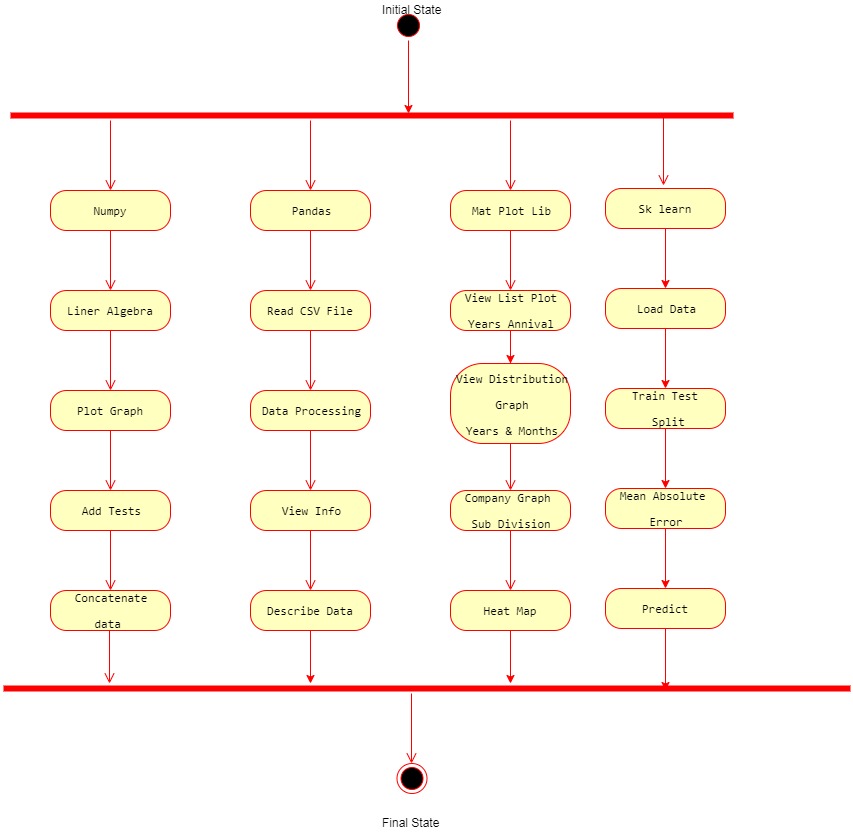
I won’t spend too much time talking about this, but Python does allow for a programmer to create his own exceptions. You probably won’t have to do this very often but it’s nice to have the option when necessary.

However, before making your own exceptions, make sure there isn’t one of the built-in exceptions that will work for you. They have been "tested by fire" over the years and not only work effectively, they have been optimized for performance and are bug-free. Making your own exceptions involves object-oriented programming, which will be covered in the next chapter. To make a custom exception, the programmer determines which base exception to use as the class to inherit from, e.g. making an exception for negative numbers or one for imaginary numbers would probably fall under the Arithmetic Error exception class. To make a custom exception, simply inherit the base exception and define what it will do.









**SYSTEM TESTING**

**Introduction to Testing**

Testing is a process, which reveals errors in the program. Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. The increasing visibility of software as a system element and attendant costs associated with a software failure are motivating factors for we planned, through testing. Testing is the process of executing a program with the intent of finding an error. The design of tests for software and other engineered products can be as challenging as the initial design of the product itself It is the major quality measure employed during software development. During testing, the program is executed -with a set of test cases and the output of the program for the test cases is evaluated to determine if the program is performing as it is expected to perform.A test case contains all the information necessary to verify some particular functionality of the software.

**Purpose -** Describe the features of the software to be tested, and the particular behavior being verified by this test.

**Requirement Traceability -** A cross reference to the numbers of the requirements (in the system specification) which are being verified in this test.

**Setup -** Describe all the steps necessary to setup the software environment necessary to carry out the test.

**Test data -** Write the actual input data to be provided and the expected output for your actual working product. You must provide the actual input data values, not just a description.

**Test Plan -** A test plan outlines the strategy that will be used to test an application, the resources that will be used, the test environment in which testing will be performed, and the limitations of the testing and the schedule of testing activities.

**.2 Testing Strategy**

The development of software systems involves of a series of production activities where opportunities for injection of human falibilities are enormous. Errors may begin to occur at the very inception of the process where the objectives may be erroneously or imperfectly specified, as well as in later design and development stages. Test techniques include the process of executing a program or application with the intent of finding software bugs (errors or other defects).In order to make sure that the system does not have errors, the different levels of testing strategies that are applied at differing phases of software development are:

**Unit Testing**

Unit Testing is done on individual modules as they are completed and become executable. It is confined only to the designer's requirements.

**Functional Testing**

This is a type of black-box testing that is based on the specifications of the software that is to be tested. The application is tested by providing input and then the results are examined that need to conform to the functionality it was intended for. Functional testing of a software is conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements.

**Black Box Testing**

In this strategy some test cases are generated as input conditions that fully execute all functional requirements for the program. This testing has been uses to find errors in the following categories Incorrect or missing functions

• Interface errors

• Errors in data structure or external database access

• Performance errors

**White Box Testing**

In this the test cases are generated on the logic of each module by drawing flow graphs of that module and logical decisions are tested on all the cases. It has been uses

to generate the test cases in the following cases:

*  Guarantee that all independent paths have been Executed.
*  Execute all logical decisions on their true and false Sides.

**Regression Testing**

Whenever a change in a software application is made, it is quite possible that other areas within the application have been affected by this change. Regression testing is performed to verify that a fixed bug hasn't resulted in another functionality or business rule violation.

**Acceptance Testing**

This is arguably the most important type of testing, as it is conducted by the Quality Assurance Team who will gauge whether the application meets the intended specifications and satisfies the client’s requirement.

**Performance Testing**

It is mostly used to identify any bottlenecks or performance issues rather than finding bugs in a software. There are different causes that contribute in lowering the performance of a software:

• Network delay

 • Client-side processing

• Database transaction processing

• Load balancing between servers

• Data rendering

• Load Testing

It is a process of testing the behaviour of a software by applying maximum load in terms of software accessing and manipulating large input data.

**Usability Testing**

Usability testing is a black-box technique and is used to identify any error(s) and improvements in the software. Usability can be defined in terms of five factors, i.e. efficiency of use, learn-ability, memory-ability, errors/safety, and satisfaction.

**Security Testing**

Security testing involves testing a software in order to identify any flaws and gaps from security and vulnerability point of view. Listed below are the main aspects that security testing should ensure:

• Confidentiality

• Integrity

• Authentication

• Availability

• Integrating Testing

Integration testing ensures that software and subsystems work together a whole. It tests the interface of all the modules to make sure that the modules behave properly when integrated together.

**System Testing**

System testing involves in-house testing of the entire system before delivery to the user. Its aim is to satisfy the user the system meets all requirements of the client's specifications.

**Acceptance Testing**

It is a pre-delivery testing in which entire system is tested at client's site on real world data to find errors.

**.3 Test Approach**

Testing can be done in two ways :

**.3.1 Bottom up Approach**

Testing can be performed starting from smallest and lowest level modules and proceeding one at a time. When bottom level modules are tested attention turns to those on the next level that use the lower level ones they are tested individually and then linked with the previously examined lower level modules.

**.3.2 Top down Approach**

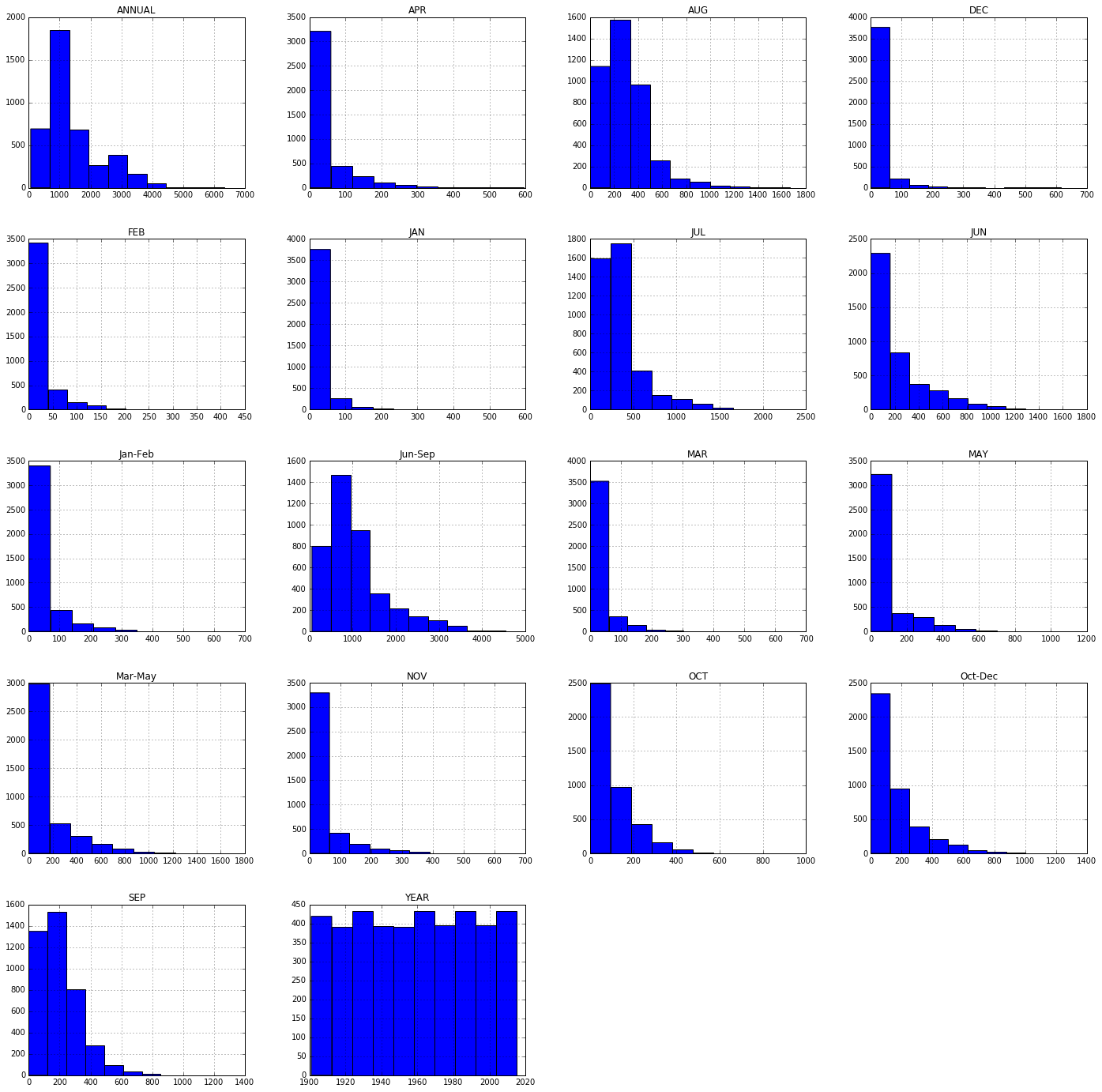
This type of testing starts from upper level modules. Since the detailed activities usually performed in the lower level routines are not provided stubs are written. A stub is a module shell called by upper level module and that when reached properly will return a message to the calling module indicating that proper interaction occurred.

**.4 Validation**

The system has been tested and implemented successfully and thus ensured that all the requirements as listed in the software requirements specification are completely fulfilled.

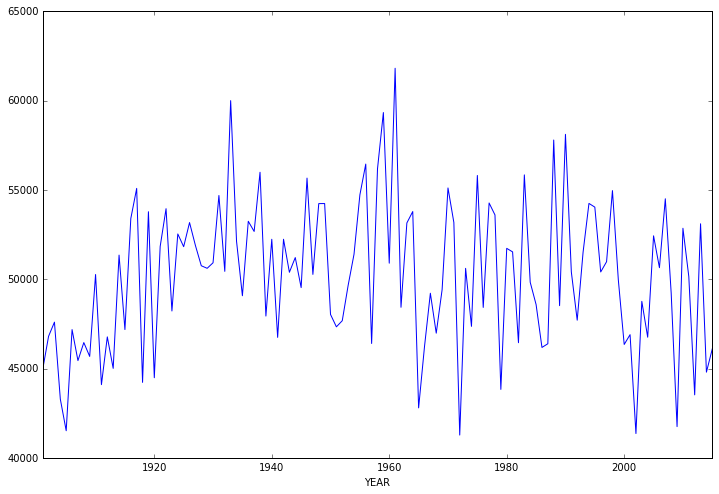
1. **SCREENSHOTS**

These are the following results which we get during the execution of the process which are mentioned in level by level detail.

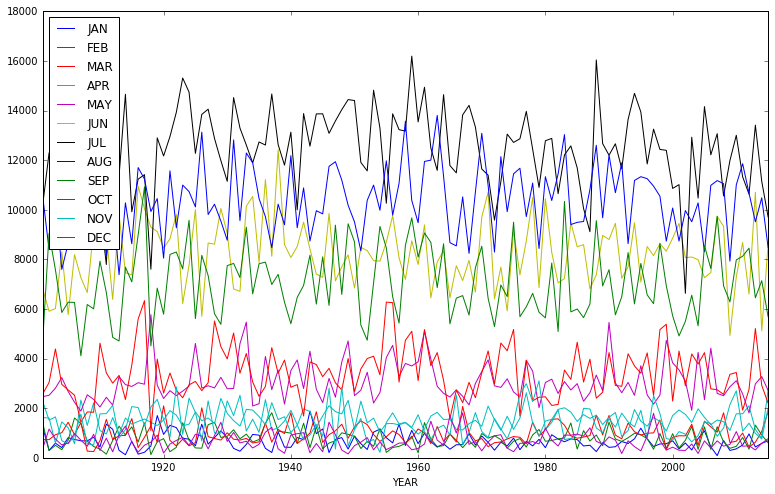


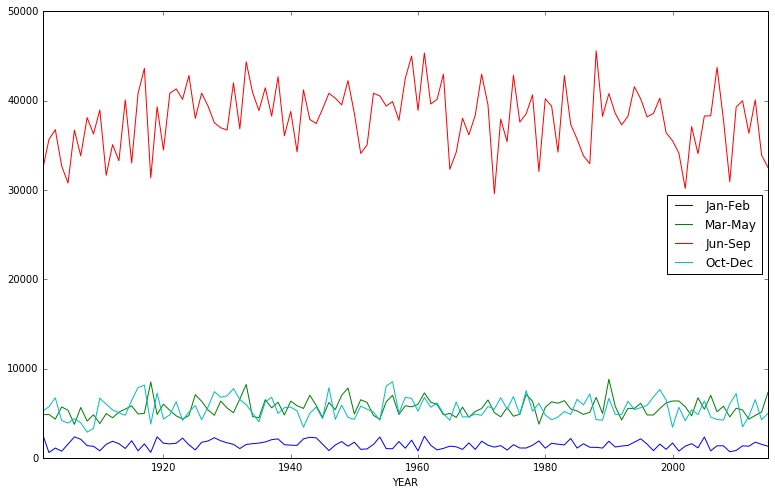
Histogram representation of rainfall

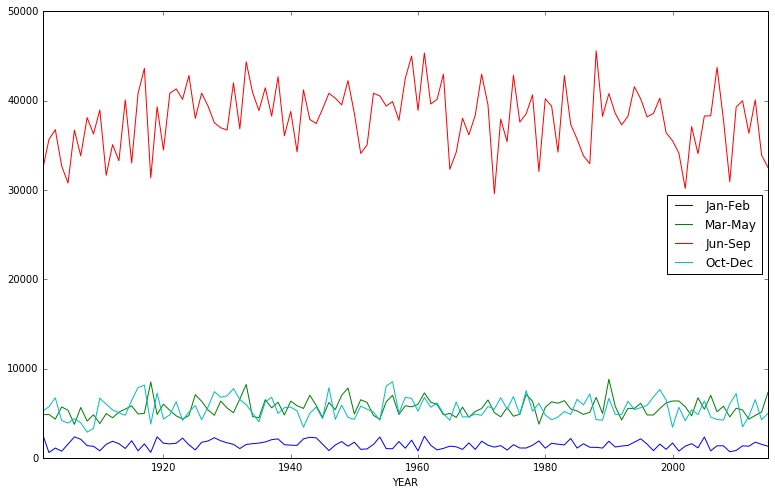
Above histograms show the distribution of rainfall over months.Observed increase in amount of rainfall over months July, August, September.

 Plot graph

Shows distribution of rainfall over years. Observed high amount of rainfall in 1950s.

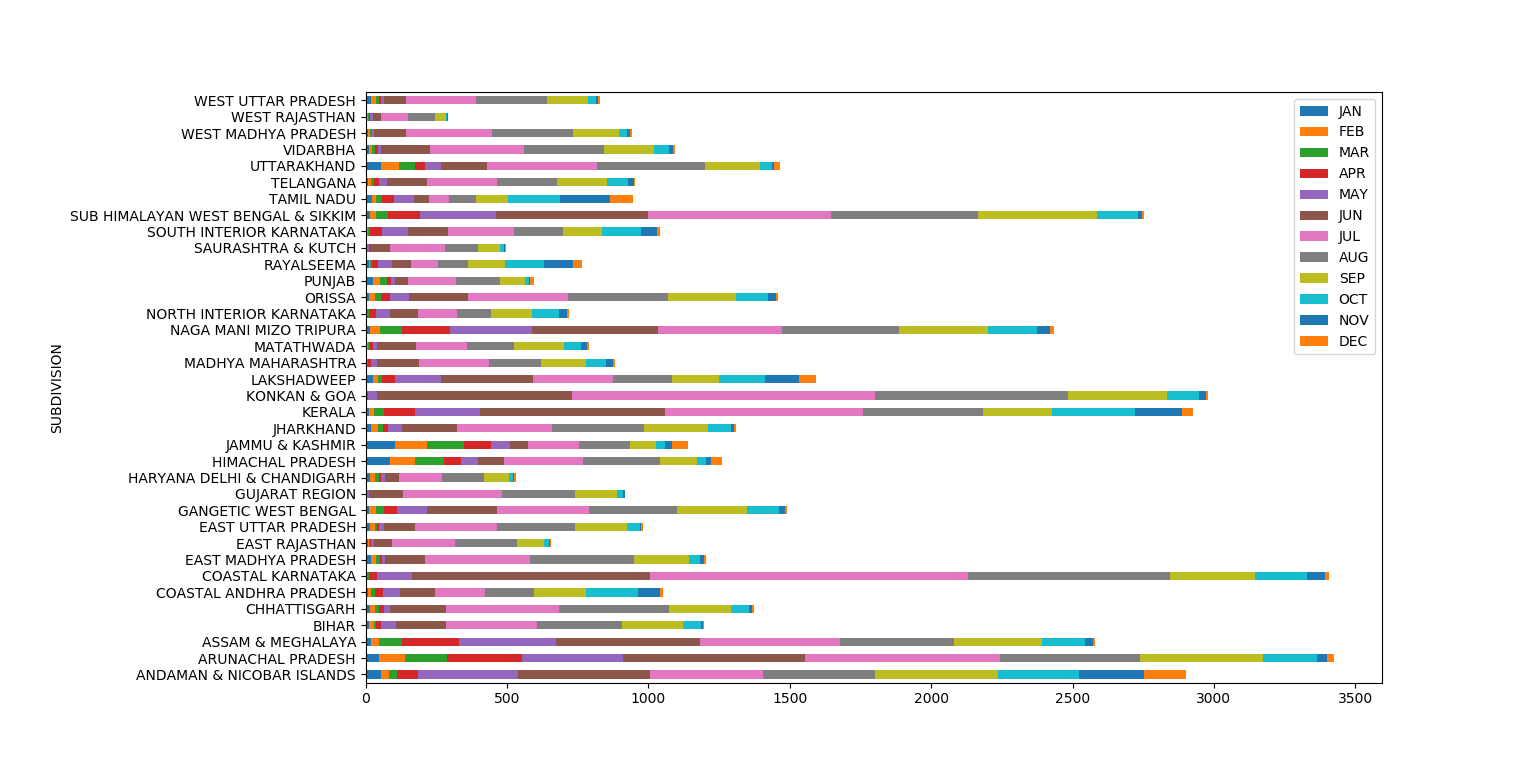
****

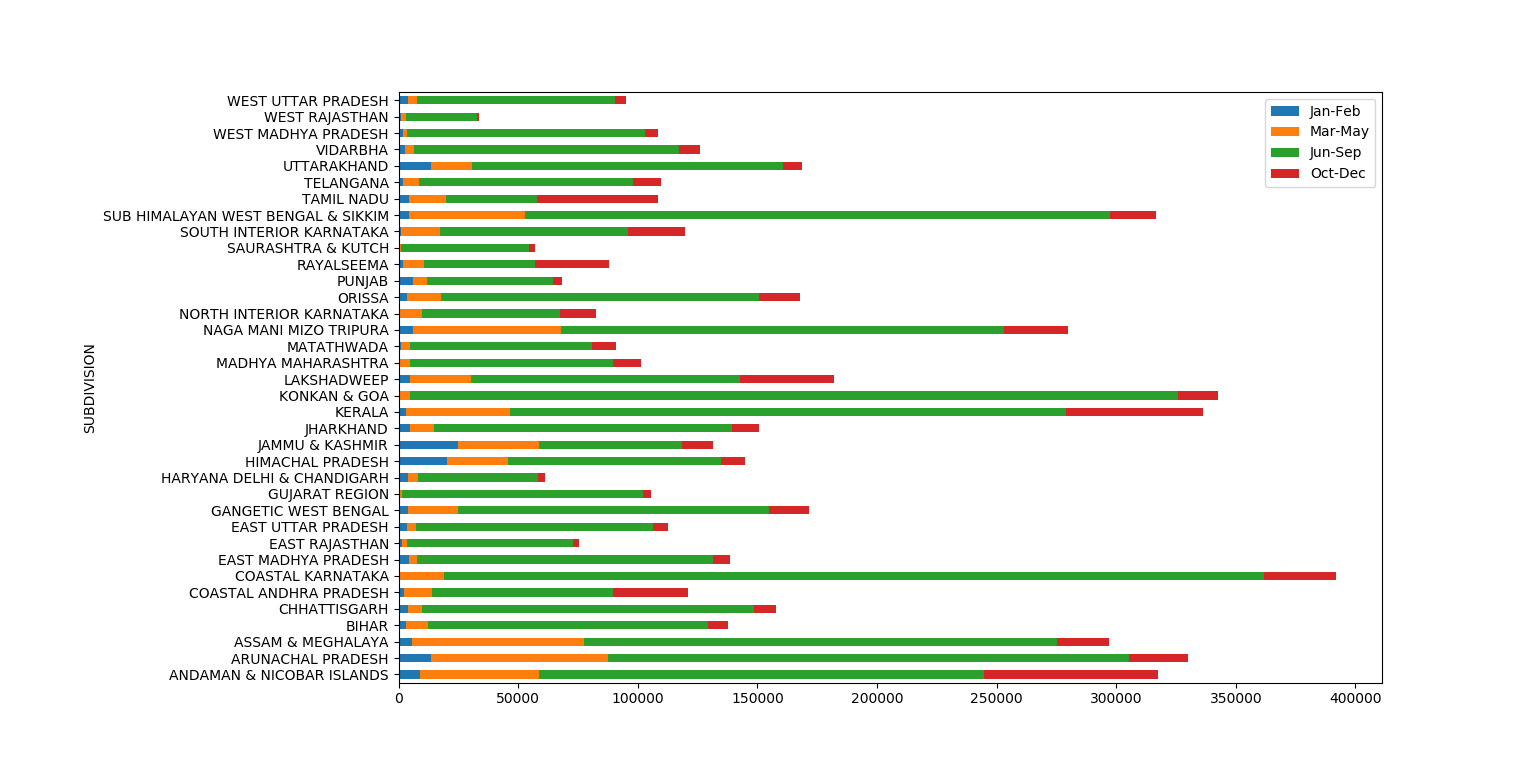




Distribution of rainfall over months

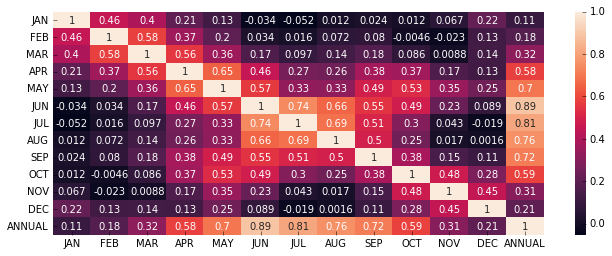
The graphs clearly shows that amount of rainfall in high in the months july, aug, sep which is monsoon season in India.

****

****

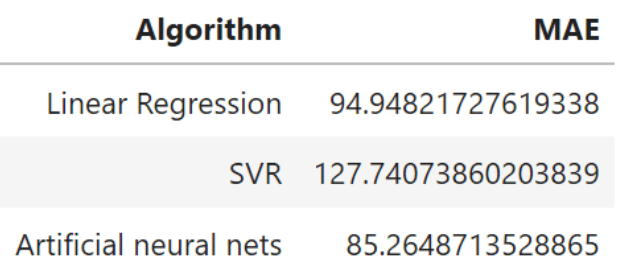
Horizontal bar graph

Above two graphs shows that the amount of rainfall is reasonably good in the months of march, april, may in eastern India.

 ****

Heap graph

Heat Map shows the co-relation (dependency) betwenn the amounts of rainfall over months. From above it is clear that if amount of rainfall is high in the months of july, august, september then the amount of rainfall will be high annually. It is also obwserved that if amount of rainfall in good in the months of october, november, december then the rainfall is going to b good in the overall year.



Mean Absolute Error

In statistics, mean absolute error is a measure of difference between two continuous variables.The mean absolute error is given by:

