**WEATHER FORECASTING**

**SUBMITTED TO PROF TSENG-CHING SHEN**

**A finger pointing at a weather forecast

Description automatically generated**

**BY**

**PULAPALLI MOUNIKA**

**SONIKA**

**Objective:**

The main aim of the project is to forecast weather based on various weather parameters weather such as snowfall, fog, and cold. This project will design and implement a machine learning-driven weather forecasting system that harnesses big data analytics**.**

**Source of Data set:**

The dataset is downloaded from the Kaggle website. It's about 1.09 GB in size.

[**https://www.kaggle.com/datasets/sobhanmoosavi/us-weather-events/data**](https://www.kaggle.com/datasets/sobhanmoosavi/us-weather-events/data)

**A screenshot of a weather report

Description automatically generatedA screenshot of a computer

Description automatically generated**

**Functionalities:**

A weather forecasting project typically involves various functionalities aimed at predicting and presenting weather conditions. Here are some key functionalities often included in such projects:

* Gathering data from various sources like weather stations, satellites, radar, and sensors. This involves accessing real-time or historical weather data such as temperature, humidity, wind speed, precipitation, atmospheric pressure, etc.
* Processing the collected data using algorithms and models to derive meaningful insights. This could involve statistical analysis, machine learning techniques, or numerical weather prediction models to forecast future weather conditions.
* Short-term predictions (up to a week) typically involve forecasting temperature changes, precipitation, wind patterns, and more. Long-term forecasts may focus on broader climate trends.
* Presenting the weather information to users in an understandable and visually appealing manner. This includes maps, charts, graphs, and other visual aids to display current conditions and forecasts.
* Allowing users to access past weather data for analysis, comparison, or research purposes. This could involve providing access to archives of weather information.
* Constantly refining algorithms and models to enhance the accuracy of weather predictions. This might involve incorporating new data sources, improving algorithms, or refining machine learning models.
* Allowing users to set preferences such as units of measurement, favorite locations, specific weather parameters of interest, and personalization options.

**Architecture and Design:**

The project is built on top of AWS. The entire project runs on the Hadoop cluster built on top of AWS.

**A diagram of a company

Description automatically generated**

**The resources used from AWS are:**

1. AWS EMR
2. S3
3. EMR Studio
4. EMR Workspace
5. Jupyter Notebook

**AWS EMR:**

Apache™ Hadoop® is an open-source software project that can process large datasets efficiently. Instead of using one large computer to process and store the data, Hadoop allows the clustering of commodity hardware together to analyze massive data sets in parallel.

There are many applications and execution engines in the Hadoop ecosystem, providing a variety of tools to match the needs of your analytics workloads. [Amazon EMR](https://aws.amazon.com/emr/) makes it easy to create and manage fully configured, elastic clusters of Amazon EC2 instances running Hadoop and other applications in the Hadoop ecosystem.

**A diagram of an overview of an emr

Description automatically generated**

**AWS S3:**

Amazon Simple Storage Service (Amazon S3) is an object storage service offering industry-leading scalability, data availability, security, and performance. Customers of all sizes and industries can store and protect data for virtually any use case, such as data lakes, cloud-native applications, and mobile apps. With cost-effective storage classes and easy-to-use management features, you can optimize costs, organize data, and configure fine-tuned access controls to meet specific business, organizational, and compliance requirements.

**A screenshot of a logo

Description automatically generated**

**AWS EMR STUDIO:**

Amazon EMR Studio is a web-based integrated development environment (IDE) for fully managed Jupyter notebooks that run on Amazon EMR clusters. You can set up an EMR Studio for your team to develop, visualize, and debug applications written in R, Python, Scala, and PySpark.

**A diagram of a computer network

Description automatically generated**

**AWS EMR WORKSPACE :**

You can use Amazon EMR Notebooks with Amazon EMR clusters running [Apache Spark](http://aws.amazon.com/emr/features/spark/) to create and open [Jupyter](https://jupyter.org/" \t "_blank) Notebook and JupyterLab interfaces within the Amazon EMR console. An EMR notebook is a "serverless" notebook that you can use to run queries and code. Unlike a traditional notebook, the contents of an EMR notebook — the equations, queries, models, code, and narrative text within notebook cells — run in a client. The commands are executed using a kernel on the EMR cluster. Notebook contents are also saved to Amazon S3 separately from cluster data for durability and flexible reuse.

You can start a cluster, attach an EMR notebook for analysis, and then terminate the cluster. You can also close a notebook attached to one running cluster and switch to another. Multiple users can attach notebooks to the same cluster simultaneously and share notebook files in Amazon S3 with each other. These features let you run clusters on-demand to save cost and reduce the time spent re-configuring notebooks for different clusters and datasets.

**Deployment Instructions:**

Series of steps are involved in the development of this project.

1. Create an AWS account.
2. Log in to the account as an AWS IAM User
3. Create AWS S3 bucket for data storage
4. Set up the environment to create a Hadoop cluster on AWS EMR.
5. Create Aws EMR Studio to launch the workspace.
6. Launch the Jupyter Notebook and attach the Hadoop cluster and AWS S3 to proceed with data processing using Apache Pyspark Kernel.

**How to create S3 Bucket for data Storage:**

1. Sign in to the  [AWS Management Console](https://console.aws.amazon.com/).
2. Under **Storage & Content Delivery**, choose **S3** to open the Amazon S3 console.

If you are using the **Show All Services** view, your screen looks like this:

A screenshot of a computer

Description automatically generated

1. From the Amazon S3 console dashboard, choose **Create Bucket**.
2. In **Create a Bucket**, type a bucket name in **Bucket Name**.
3. In **Region**, choose **particular region**
4. Choose **Create**.

When Amazon S3 successfully creates your bucket, the console displays your empty bucket in the **Buckets** pane.

**How to Set up Hadoop Cluster:**

###### To launch a cluster with Spark installed with the new console

1. Sign in to the AWS Management Console, and open the Amazon EMR console at <https://console.aws.amazon.com/emr>.
2. Under **EMR on EC2** in the left navigation pane, choose **Clusters**, and then choose **Create cluster**.
3. On the **Create Cluster** page, note the default values for **Release**, **Instance type**, **Number of instances**, and **Permissions**. These fields automatically populate with values that work for general-purpose clusters.
4. In the **Cluster name** field, enter a unique cluster name to help you identify your cluster, such as *My first cluster*.
5. Under **Applications**, choose the **Spark** option to install Spark on your cluster.

###### Note

Choose the applications you want on your Amazon EMR cluster before you launch the cluster. You can't add or remove applications from a cluster after launch.

1. Under **Cluster logs**, select the Publish cluster-specific logs to Amazon S3 check box. Replace the **Amazon S3 location** value with the Amazon S3 bucket you created, followed by **/logs**. For example, **s3://DOC-EXAMPLE-BUCKET/logs**. Adding **/logs** creates a new folder called 'logs' in your bucket, where Amazon EMR can copy the log files of your cluster.
2. Under **Security configuration and permissions**, choose your **EC2 key pair**. In the same section, select the **Service role for Amazon EMR** dropdown menu and choose **EMR\_DefaultRole**. Then, select the **IAM role for instance profile** dropdown menu and choose **EMR\_EC2\_DefaultRole**.
3. Choose **Create cluster** to launch the cluster and open the cluster details page.
4. Find the cluster **Status** next to the cluster name. The status changes from **Starting** to **Running** to **Waiting** as Amazon EMR provisions the cluster. You may need to choose the refresh icon on the right or refresh your browser to see status updates.

Your cluster status changes to **Waiting** when the cluster is up, running, and ready to accept work.

A successful Amazon EMR cluster follows this process:

1. Amazon EMR first provisions EC2 instances in the cluster for each instance according to your specifications. For more information, see [Configure cluster hardware and networking](https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-plan-instances.html). For all instances, Amazon EMR uses the default AMI for Amazon EMR or a custom Amazon Linux AMI that you specify. For more information, see [Using a custom AMI](https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-custom-ami.html). During this phase, the cluster state is STARTING.
2. Amazon EMR runs bootstrap actions that you specify on each instance. You can use bootstrap actions to install custom applications and perform customizations that you require. For more information, see [Create bootstrap actions to install additional software](https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-plan-bootstrap.html). During this phase, the cluster state is BOOTSTRAPPING.
3. Amazon EMR installs the native applications that you specify when you create the cluster, such as Hive, Hadoop, Spark, and so on.
4. After bootstrap actions are successfully completed and native applications are installed, the cluster state is RUNNING. At this point, you can connect to cluster instances, and the cluster sequentially runs any steps that you specified when you created the cluster. You can submit additional steps, which run after any previous steps complete. For more information, see [Submit work to a cluster](https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-work-with-steps.html).
5. After steps run successfully, the cluster goes into a WAITING state. If a cluster is configured to auto-terminate after the last step is complete, it goes into a TERMINATING state and then into the TERMINATED state. If the cluster is configured to wait, you must manually shut it down when you no longer need it. After you manually shut down the cluster, it goes into the TERMINATING state and then into the TERMINATED state.

A failure during the cluster lifecycle causes Amazon EMR to terminate the cluster and all of its instances unless you enable termination protection. If a cluster terminates because of a failure, any data stored on the cluster is deleted, and the cluster state is set to TERMINATED\_WITH\_ERRORS. If you enabled termination protection, you can retrieve data from your cluster, and then remove termination protection and terminate the cluster.

**A diagram of a process

Description automatically generated**

###### 3. To set up an EMR Studio

1. [Choose an authentication mode for Amazon EMR Studio](https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-studio-authentication.html)
2. Create the following Studio resources.
   * [Create an EMR Studio service role](https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-studio-service-role.html)
   * [Configure EMR Studio user permissions for Amazon EC2 or Amazon EKS](https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-studio-user-permissions.html)
   * (Optional) [Define security groups to control EMR Studio network traffic](https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-studio-security-groups.html).
3. [Create an EMR Studio](https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-studio-create-studio.html)
4. [Assign a user or group to an EMR Studio](https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-studio-manage-users.html#emr-studio-assign-users-groups).

###### To create an EMR notebook

1. Open the Amazon EMR console at <https://console.aws.amazon.com/elasticmapreduce/>.
2. Choose Notebooks, Create notebook.
3. Enter a Notebook name and an optional Notebook description.
4. If you have an active cluster to which you want to attach the notebook, leave the default Choose an existing cluster selected, click Choose, select a cluster from the list, and then click Choose cluster. OR

Choose Create a Cluster, enter a Cluster name, and choose options according to the following guidelines. The cluster is created in the default VPC for the account using On-Demand instances.

###### To open the notebook editor for an EMR notebook

1. Select a notebook with a **Status** of **Ready** or **Pending** from the **Notebooks** list.
2. Choose **Open in JupyterLab** or **Open in Jupyter**.

A new browser tab opens to the JupyterLab or Jupyter Notebook editor.

1. From the **Kernel** menu, choose **Change kernel** and then select the kernel for your programming language.

You are now ready to write and run code from within the notebook editor.

**STEPS TO RUN THE APPLICATION:**

**The architecture to build this**

1. Login to the AWS account.
2. Start the cluster.
3. Launch the workspace from AWS EMR studio in order to start the jupyter notebook.
4. Attach the cluster to the workspace.
5. Start the pyspark kernel in the workspace.
6. Start coding and perform the analysis.

**Results:**

**S3 For Storage:**

**A screenshot of a computer

Description automatically generated**

**Hadoop Cluster**

**A screenshot of a computer

Description automatically generated**

**Cluster configuration :**

**A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated**

**EMR STUDIO:**

**A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated**

**Pyspark Jupyter Notebook:**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**Data Preprocessing**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated**

**Data Visualization:**

**A screenshot of a computer program

Description automatically generated**

**A diagram of a weather event distribution

Description automatically generated**

**A graph of different colored squares

Description automatically generated**

**A white background with red text

Description automatically generated**

**ML MODEL TRAINING -RANDOM FOREST CLASSIFIER**

**A screenshot of a computer

Description automatically generated**

**A screen shot of a computer program

Description automatically generated**